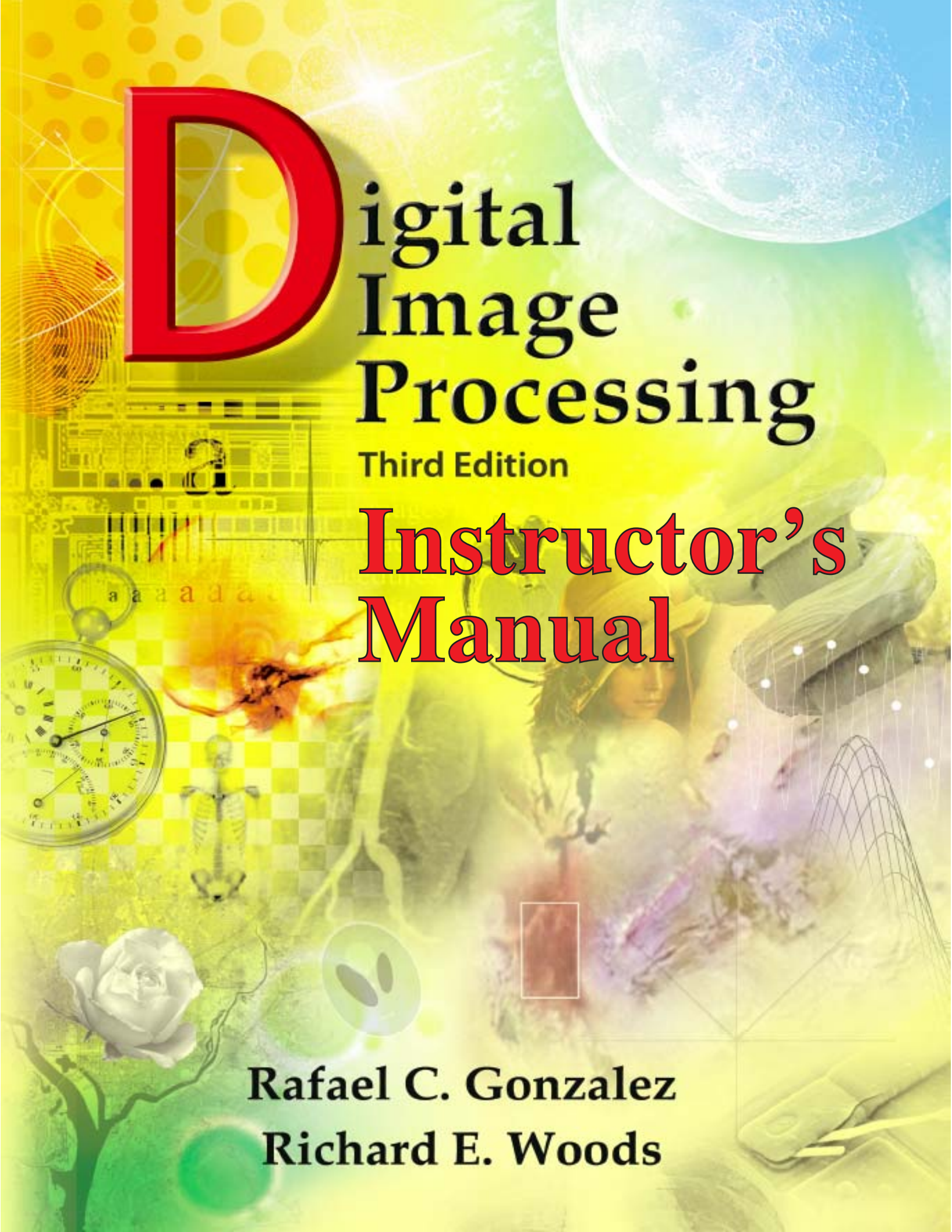

Digital Image Processing
Third Edition
Instructor's Manual
Rafael C. Gonzalez
Richard E. Woods

Digital Image Processing

Third Edition

Instructor's Manual

Version 3.0

Rafael C. Gonzalez
Richard E. Woods

Prentice Hall
Upper Saddle River, NJ 07458

www.imageprocessingplace.com

Chapter 1

Introduction

The purpose of this chapter is to present suggested guidelines for teaching material from *Digital Image Processing* at the senior and first-year graduate levels. We also discuss use of the book web site. Although the book is totally self-contained, the web site offers, among other things, complementary review material and computer projects that can be assigned in conjunction with classroom work. Detailed solutions to all problems in the book also are included in the remaining chapters of this manual.

1.1 Teaching Features of the Book

Undergraduate programs that offer digital image processing typically limit coverage to one semester. Graduate programs vary, and can include one or two semesters of the material. In the following discussion we give general guidelines for a one-semester senior course, a one-semester graduate course, and a full-year course of study covering two semesters. We assume a 15-week program per semester with three lectures per week. In order to provide flexibility for exams and review sessions, the guidelines discussed in the following sections are based on forty, 50-minute lectures per semester. The background assumed on the part of the student is senior-level preparation in mathematical analysis, matrix theory, probability, and computer programming. The Tutorials section in the book web site contains review materials on matrix theory and probability, and has a brief introduction to linear systems. PowerPoint classroom presentation material on the review topics is available in the Faculty section of the web site.

The suggested teaching guidelines are presented in terms of general objectives, and not as time schedules. There is so much variety in the way image processing material is taught that it makes little sense to attempt a breakdown of the material by class period. In particular, the organization of the present edition of

the book is such that it makes it much easier than before to adopt significantly different teaching strategies, depending on course objectives and student background. For example, it is possible with the new organization to offer a course that emphasizes spatial techniques and covers little or no transform material. This is not something we recommend, but it is an option that often is attractive in programs that place little emphasis on the signal processing aspects of the field and prefer to focus more on the implementation of spatial techniques.

1.2 One Semester Senior Course

A basic strategy in teaching a senior course is to focus on aspects of image processing in which both the inputs and outputs of those processes are images. In the scope of a senior course, this usually means the material contained in Chapters 1 through 6. Depending on instructor preferences, wavelets (Chapter 7) usually are beyond the scope of coverage in a typical senior curriculum. However, we recommend covering at least some material on image compression (Chapter 8) as outlined below.

We have found in more than three decades of teaching this material to seniors in electrical engineering, computer science, and other technical disciplines, that one of the keys to success is to spend at least one lecture on motivation and the equivalent of one lecture on review of background material, as the need arises. The motivational material is provided in the numerous application areas dis1.2 One Semester Senior Coursecussed in Chapter 1. This chapter was prepared with this objective in mind. Some of this material can be covered in class in the first period and the rest assigned as independent reading. Background review should cover probability theory (of one random variable) before histogram processing (Section 3.3). A brief review of vectors and matrices may be required later, depending on the material covered. The review material in the book web site was designed for just this purpose.

Chapter 2 should be covered in its entirety. Some of the material (Sections 2.1 through 2.3.3) can be assigned as independent reading, but more detailed explanation (combined with some additional independent reading) of Sections 2.3.4 and 2.4 through 2.6 is time well spent. The material in Section 2.6 covers concepts that are used throughout the book and provides a number of image processing applications that are useful as motivational background for the rest of the book

Chapter 3 covers spatial intensity transformations and spatial correlation and convolution as the foundation of spatial filtering. The chapter also covers a number of different uses of spatial transformations and spatial filtering for image enhancement. These techniques are illustrated in the context enhancement

(as motivational aids), but it is pointed out several times in the chapter that the methods developed have a much broader range of application. For a senior course, we recommend covering Sections 3.1 through 3.3.1, and Sections 3.4 through 3.6. Section 3.7 can be assigned as independent reading, depending on time.

The key objectives of Chapter 4 are (1) to start from basic principles of signal sampling and from these derive the discrete Fourier transform; and (2) to illustrate the use of filtering in the frequency domain. As in Chapter 3, we use mostly examples from image enhancement, but make it clear that the Fourier transform has a much broader scope of application. The early part of the chapter through Section 4.2.2 can be assigned as independent reading. We recommend careful coverage of Sections 4.2.3 through 4.3.4. Section 4.3.5 can be assigned as independent reading. Section 4.4 should be covered in detail. The early part of Section 4.5 deals with extending to 2-D the material derived in the earlier sections of this chapter. Thus, Sections 4.5.1 through 4.5.3 can be assigned as independent reading and then devote part of the period following the assignment to summarizing that material. We recommend class coverage of the rest of the section. In Section 4.6, we recommend that Sections 4.6.1-4.6.6 be covered in class. Section 4.6.7 can be assigned as independent reading. Sections 4.7.1-4.7.3 should be covered and Section 4.7.4 can be assigned as independent reading. In Sections 4.8 through 4.9 we recommend covering one filter (like the ideal lowpass and highpass filters) and assigning the rest of those two sections as independent reading. In a senior course, we recommend covering Section 4.9 through Section 4.9.3 only. In Section 4.10, we also recommend covering one filter and assigning the rest as independent reading. In Section 4.11, we recommend covering Sections 4.11.1 and 4.11.2 and mentioning the existence of FFT algorithms. The $\log_2$ computational advantage of the FFT discussed in the early part of Section 4.11.3 should be mentioned, but in a senior course there typically is no time to cover development of the FFT in detail.

Chapter 5 can be covered as a continuation of Chapter 4. Section 5.1 makes this an easy approach. Then, it is possible to give the student a "flavor" of what restoration is (and still keep the discussion brief) by covering only Gaussian and impulse noise in Section 5.2.1, and two of the spatial filters in Section 5.3. This latter section is a frequent source of confusion to the student who, based on discussions earlier in the chapter, is expecting to see a more objective approach. It is worthwhile to emphasize at this point that spatial enhancement and restoration are the same thing when it comes to noise reduction by spatial filtering. A good way to keep it brief and conclude coverage of restoration is to jump at this point to inverse filtering (which follows directly from the model in Section 5.1) and show the problems with this approach. Then, with a brief explanation

regarding the fact that much of restoration centers around the instabilities inherent in inverse filtering, it is possible to introduce the "interactive" form of the Wiener filter in Eq. (5.8-3) and discuss Examples 5.12 and 5.13. At a minimum, we recommend a brief discussion on image reconstruction by covering Sections 5.11.1-5.11-2 and mentioning that the rest of Section 5.11 deals with ways to generated projections in which blur is minimized.

Coverage of Chapter 6 also can be brief at the senior level by focusing on enough material to give the student a foundation on the physics of color (Section 6.1), two basic color models (RGB and CMY/CMYK), and then concluding with a brief coverage of pseudocolor processing (Section 6.3). We typically conclude a senior course by covering some of the basic aspects of image compression (Chapter 8). Interest in this topic has increased significantly as a result of the heavy use of images and graphics over the Internet, and students usually are easily motivated by the topic. The amount of material covered depends on the time left in the semester.

1.3 One Semester Graduate Course (No Background in DIP)

The main difference between a senior and a first-year graduate course in which neither group has formal background in image processing is mostly in the scope of the material covered, in the sense that we simply go faster in a graduate course and feel much freer in assigning independent reading. In a graduate course we add the following material to the material suggested in the previous section.

Sections 3.3.2-3.3.4 are added as is Section 3.3.8 on fuzzy image processing. We cover Chapter 4 in its entirety (with appropriate sections assigned as independent readying, depending on the level of the class). To Chapter 5 we add Sections 5.6-5.8 and cover Section 5.11 in detail. In Chapter 6 we add the HSI model (Section 6.3.2) , Section 6.4, and Section 6.6. A nice introduction to wavelets (Chapter 7) can be achieved by a combination of classroom discussions and independent reading. The minimum number of sections in that chapter are 7.1, 7.2, 7.3, and 7.5, with appropriate (but brief) mention of the existence of fast wavelet transforms. Sections 8.1 and 8.2 through Section 8.2.8 provide a nice introduction to image compression.

If additional time is available, a natural topic to cover next is morphological image processing (Chapter 9). The material in this chapter begins a transition from methods whose inputs and outputs are images to methods in which the inputs are images, but the outputs are attributes about those images, in the sense defined in Section 1.1. We recommend coverage of Sections 9.1 through 9.4, and

some of the algorithms in Section 9.5.

1.4 One Semester Graduate Course (with Student Having Background in DIP)

Some programs have an undergraduate course in image processing as a prerequisite to a graduate course on the subject, in which case the course can be biased toward the latter chapters. In this case, a good deal of Chapters 2 and 3 is review, with the exception of Section 3.8, which deals with fuzzy image processing. Depending on what is covered in the undergraduate course, many of the sections in Chapter 4 will be review as well. For Chapter 5 we recommend the same level of coverage as outlined in the previous section.

In Chapter 6 we add full-color image processing (Sections 6.4 through 6.7). Chapters 7 and 8 are covered as outlined in the previous section. As noted in the previous section, Chapter 9 begins a transition from methods whose inputs and outputs are images to methods in which the inputs are images, but the outputs are attributes about those images. As a minimum, we recommend coverage of binary morphology: Sections 9.1 through 9.4, and some of the algorithms in Section 9.5. Mention should be made about possible extensions to gray-scale images, but coverage of this material may not be possible, depending on the schedule. In Chapter 10, we recommend Sections 10.1 through 10.4. In Chapter 11 we typically cover Sections 11.1 through 11.4.

1.5 Two Semester Graduate Course (No Background in DIP)

In a two-semester course it is possible to cover material in all twelve chapters of the book. The key in organizing the syllabus is the background the students bring to the class. For example, in an electrical and computer engineering curriculum graduate students have strong background in frequency domain processing, so Chapter 4 can be covered much quicker than would be the case in which the students are from, say, a computer science program. The important aspect of a full year course is exposure to the material in all chapters, even when some topics in each chapter are not covered.

1.6 Projects

One of the most interesting aspects of a course in digital image processing is the pictorial nature of the subject. It has been our experience that students truly enjoy and benefit from judicious use of computer projects to complement the

material covered in class. Because computer projects are in addition to course work and homework assignments, we try to keep the formal project reporting as brief as possible. In order to facilitate grading, we try to achieve uniformity in the way project reports are prepared. A useful report format is as follows:

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.1.2 One Semester Senior Course

Project resources available in the book web site include a sample project, a list of suggested projects from which the instructor can select, book and other images, and MATLAB functions. Instructors who do not wish to use MATLAB will find additional software suggestions in the Support/Software section of the web site.

1.7 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.

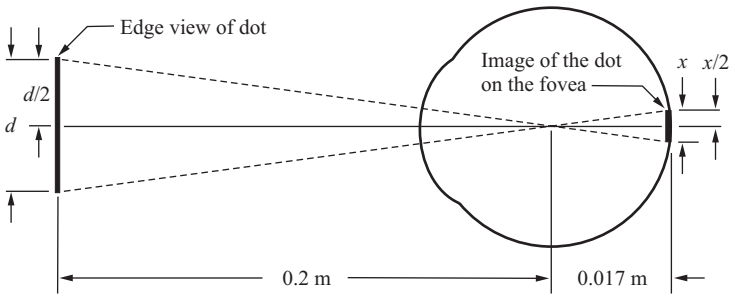


Figure P2.1

Problem 2.2

Brightness adaptation.

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 2.4

(a) From the discussion on the electromagnetic spectrum in Section 2.2, the source of the illumination required to see an object must have wavelength the same size or smaller than the object. Because interest lies only on the boundary shape and not on other spectral characteristics of the specimens, a single illumination source in the far ultraviolet (wavelength of .001 microns or less) will be able to detect all objects. A far-ultraviolet camera sensor would be needed to image the specimens.

(b) No answer is required because the answer to (a) is affirmative.

Problem 2.5

From the geometry of Fig. 2.3, $(7\,\text{mm})/(35\,\text{mm}) = (z)/(500\,\text{mm})$, or $z = 100\,\text{mm}$. So the target size is 100 mm on the side. We have a total of 1024 elements per line, so the resolution of 1 line is $1024/100 = 10$ elements/mm. For line pairs we divide by 2, giving an answer of 5 lp/mm.

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 com-

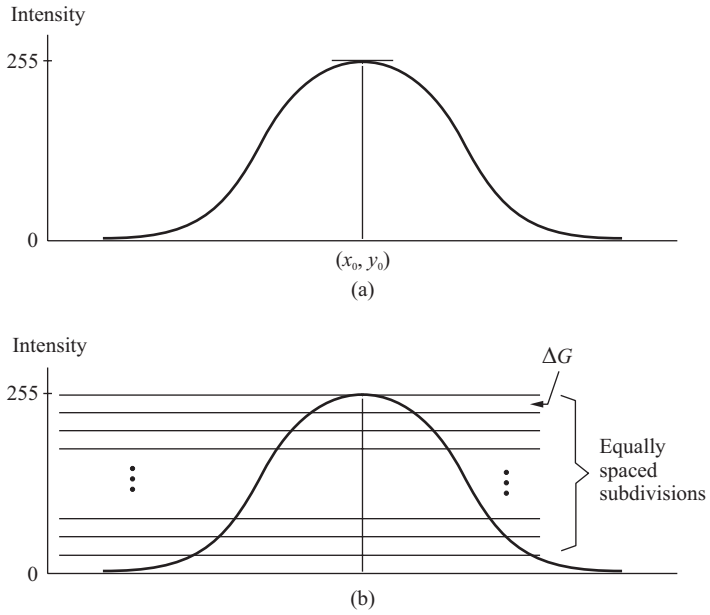


Figure P2.7

pletely 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 2.7

The image in question is given by

$$\begin{aligned} f(x,y) &= i(x,y)r(x,y) \\ &= 255\, e^{-[(x-x_0)^2+(y-y_0)^2]} \times 1.0 \\ &= 255\, e^{-[(x-x_0)^2+(y-y_0)^2]} \end{aligned}$$

A cross section of the image is shown in Fig. P2.7(a). If the intensity is quantized using m bits, then we have the situation shown in Fig. P2.7(b), where $\triangle G = (255+1)/2^m$. Since an abrupt change of 8 intensity levels is assumed to be detectable by the eye, it follows that $\triangle G = 8 = 256/2m$, or $m = 5$. In other words, 32, or fewer, intensity levels will produce visible false contouring.

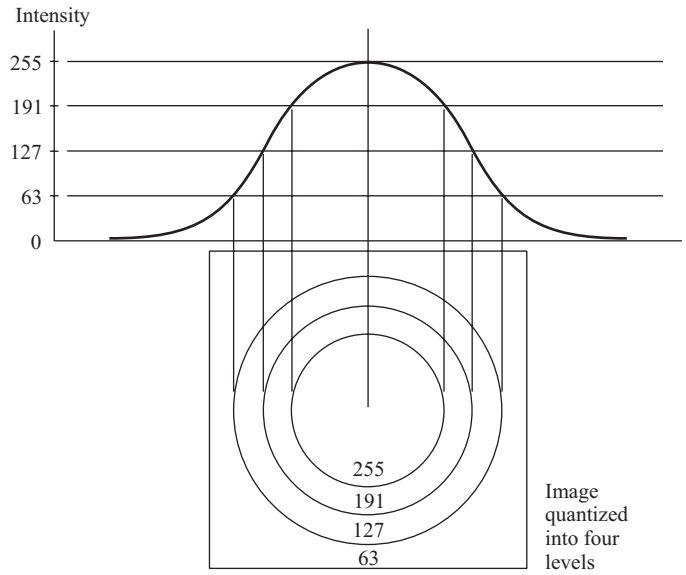


Figure P2.8

Problem 2.8

The use of two bits ($m = 2$) of intensity resolution produces four intensity levels in the range 0 to 255. One way to subdivide this range is to let all levels between 0 and 63 be coded as 63, all levels between 64 and 127 be coded as 127, and so on. The image resulting from this type of subdivision is shown in Fig. P2.8. Of course, there are other ways to subdivide the range $[0, 255]$ into four bands.

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 2.10

The width-to-height ratio is 16/9 and the resolution in the vertical direction is 1125 lines (or, what is the same thing, 1125 pixels in the vertical direction). It is given that the resolution in the horizontal direction is in the 16/9 proportion, so the resolution in the horizontal direction is $(1125) \times (16/9) = 2000$ pixels per line. The system "paints" a full 1125×2000, 8-bit image every 1/30 sec for each of the

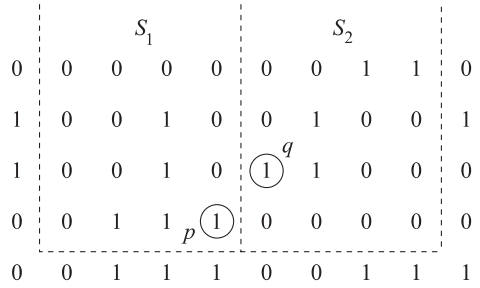


Figure P2.11

red, green, and blue component images. There are 7200 sec in two hours, so the total digital data generated in this time interval is $(1125)(2000)(8)(30)(3)(7200) = 1.166 \times 10^{13}$ bits, or 1.458×10^{12} bytes (i.e., about 1.5 terabytes). These figures show why image data compression (Chapter 8) is so important.

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.

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.

Problem 2.13

The solution to this problem is the same as for Problem 2.12 because converting from an m-connected path to a 4-connected path simply involves detecting diagonal segments and converting them to the appropriate 4-connected segment.

Problem 2.14

The difference between the pixels in the background that are holes and pixels that are not holes is than no paths exist between hole pixels and the boundary of the image. So, the definition could be restated as follows: The subset of pixels

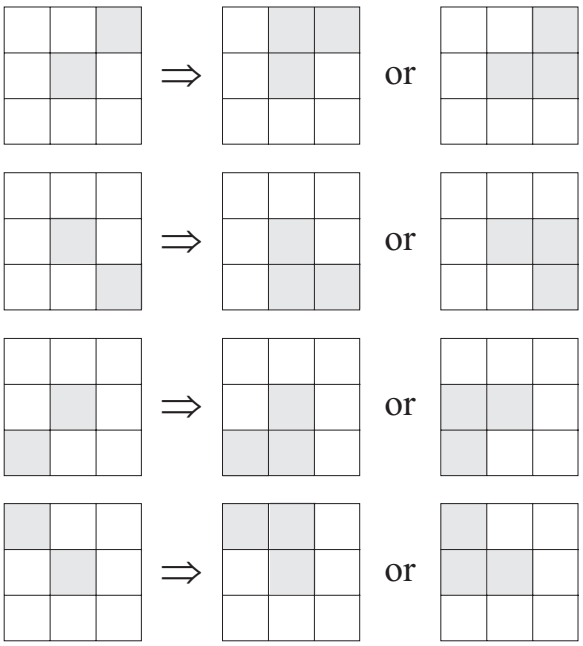


Figure P2.12

of $(R_U)^c$ that are connected to the border of the image is called the *background*. All other pixels of $(R_U)^c$ are called *hole* pixels.

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.

(b) One possibility for the shortest 4-path when $V = \{1, 2\}$ is shown in Fig. P2.15(c); its length is 6. It is easily verified that another 4-path of the same length exists between p and q. One possibility for the shortest 8-path (it is not unique) is shown in Fig. P2.15(d); its length is 4. The length of a shortest m-path (shown dashed) is 6. This path is not unique.

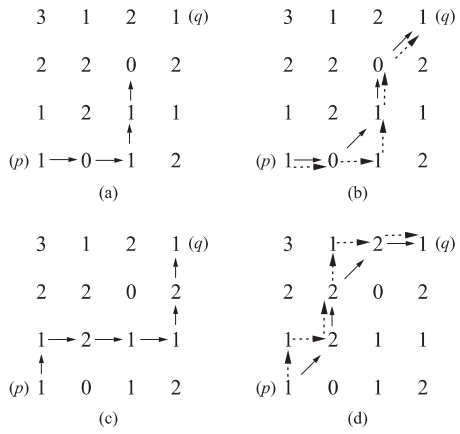


Figure P.2.15

Problem 2.16

(a) A shortest 4-path between a point p with coordinates (x,y) and a point q with coordinates (s,t) is shown in Fig. P2.16, where the assumption is that all points along the path are from V. The length of the segments of the path are $|x-s|$ and $\left|y-t\right|$, respectively. The total path length is $|x-s|+\left|y-t\right|$, 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 $|x-s|+\left|y-t\right|$. This occurs whenever we can get from p to q by following a path whose elements (1) are from V, and (2) are arranged in such a way that we can traverse the path from p to q by making turns in at most two directions (e.g., right and up).

(b) The path may or may not be unique, depending on V and the values of the points along the way.

Problem 2.17

(a) The D_8 distance between p and q [see Eq. (2.5-3) and Fig. P2.16] is

$$D_8(p,q) = \max\left(|x-s|,\left|y-t\right|\right).$$

Recall that the D_8 distance (unlike the Euclidean distance) counts diagonal segments the same as horizontal and vertical segments, and, as in the case of the D_4 distance, is independent of whether or not a path exists between p and q. As in the previous problem, the shortest 8-path is equal to the D_8 distance when the path length is $\max\left(|x-s|,\left|y-t\right|\right)$. This occurs when we can get from p to q

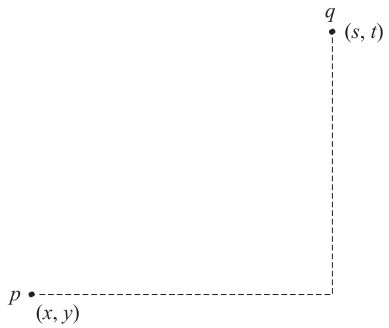


Figure P2.16

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 traveling diagonally in only one direction and, whenever diagonal travel is not possible, by making turns in the horizontal or vertical (but not both) direction.

(b) The path may or may not be unique, depending on V and the values of the points along the way.

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 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 2.19

The median, ζ, of a set of numbers is such that half the values in the set are below ζ and the other half are above it. A simple example will suffice to show that Eq. (2.6-1) is violated by the median operator. Let $S_1 = \{1,-2,3\}$, $S_2 = \{4,5,6\}$, and $a = b = 1$. In this case H is the median operator. We then have $H(S_1 + S_2)$ =median$\{5,3,9\} = 5$, where it is understood that $S_1 + S_2$ is the array sum of S_1 and S_2. Next, we compute $H(S_1) =$ median$\{1,-2,3\} = 1$ and $H(S_2) =$ median$\{4,5,6\} = 5$. Then, because $H(aS_1 + bS_2) \neq aH(S_1) + bH(S_2)$, it follows that Eq. (2.6-1) is violated and the median is a nonlinear 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

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

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

Problem 2.21

(a) Pixels are integer values, and 8 bits allow representation of 256 contiguous integer values. In our work, the range of intensity values for 8-bit images is $[0, 255]$. The subtraction of values in this range cover the range $[-255, 255]$. This range of values cannot be covered by 8 bits, but it is given in the problem statement that the result of subtraction has to be represented in 8 bits also, and, consistent with the range of values used for 8-bit images throughout the book, we assume that values of the 8-bit difference images are in the range $[0, 255]$. What this means is that any subtraction of 2 pixels that yields a negative quantity will be clipped at 0.

The process of repeated subtractions of an image $b(x, y)$ from an image $a(x, y)$ can be expressed as

$$\begin{aligned} d_K(x, y) &= a(x, y) - \sum_{k=1}^{K} b(x, y) \\ &= a(x, y) - K \times b(x, y) \end{aligned}$$

where $d_K(x, y)$ is the difference image resulting after K subtractions. Because image subtraction is an array operation (see Section 2.6.1), we can focus attention on the subtraction of any corresponding pair of pixels in the images. We have already stated that negative results are clipped at 0. Once a 0 result is obtained, it will remain so because subtraction of any nonnegative value from 0 is a negative quantity which, again, is clipped at 0. Similarly, any location (x_0, y_0) for which $b(x_0, y_0) = 0$, will produce the result $d_K(x_0, y_0) = a(x_0, y_0)$. That is, repeatedly subtracting 0 from any value results in that value. The locations in $b(x, y)$ that are not 0 will eventually decrease the corresponding values in $d_K(x, y)$ until they are 0. The maximum number of subtractions in which this takes place in the context of the present problem is 255, which corresponds to the condition at a location in which $a(x, y)$ is 255 and $b(x, y)$ is 1. Thus, we conclude from the preceding discussion that repeatedly subtracting an image from another will result in a difference image whose components are 0 in the locations in $b(x, y)$ that are not zero and equal to the original values of $a(x, y)$ at the locations in $b(x, y)$ that are 0. This result will be achieved in, at most, 255 subtractions.

(b) The order does matter. For example, suppose that at a pair of arbitrary coordinates, (x_0, y_0), $a(x_0, y_0) = 128$ and $b(x_0, y_0) = 0$. Subtracting $b(x_0, y_0)$ from $a(x_0, y_0)$ will result in $d_K(x_0, y_0) = 128$ in the limit. Reversing the operation will result in a value of 0 in that same location.

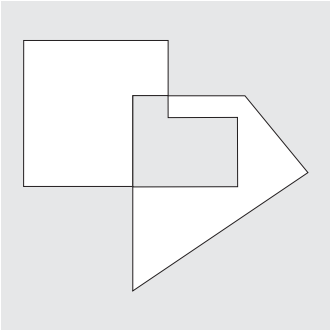
Problem 2.22

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, 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.



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

Figure P2.23

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.

(b) With reference to the sets in the problem statement, the answers are, from left to right,

$$(A \cap B \cap C) - (B \cap C);$$

$$(A \cap B \cap C) \cup (A \cap C) \cup (A \cap B);$$

$$\{B \cap (A \cup C)^c\} \cup \{(A \cap C) - [(A \cap C) \cap (B \cap C)]\}.$$

Problem 2.24

Using triangular regions means three tiepoints, so we can solve the following set of linear equations for six coefficients:

$$\begin{aligned} x' &= c_1 x + c_2 y + c_3 \\ y' &= c_4 x + c_5 y + c_6 \end{aligned}$$

to implement spatial transformations. Intensity interpolation is implemented using any of the methods in Section 2.4.4.

Problem 2.25

The Fourier transformation kernel is separable because

$$\begin{aligned} r(x,y,u,v) &= e^{-j2\pi(ux/M+vy/N)} \\ &= e^{-j2\pi(ux/M)}e^{-j2\pi(vy/N)} \\ &= r_1(x,u)r_2(y,v). \end{aligned}$$

It is symmetric because

$$\begin{aligned} e^{-j2\pi(ux/M+vy/N)} &= e^{-j2\pi(ux/M)}e^{-j2\pi(vy/N)} \\ &= r_1(x,u)r_1(y,v). \end{aligned}$$

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

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

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

Problem 2.27

The geometry of the chips is shown in Fig. P2.27(a). From Fig. P2.27(b) and the geometry in Fig. 2.3, we know that

$$\Delta x=\frac{\lambda\times 80}{\lambda-z}$$

where Δx is the side dimension of the image (assumed square because the viewing screen is square) impinging on the image plane, and the 80 mm refers to the size of the viewing screen, as described in the problem statement. The most inexpensive solution will result from using a camera of resolution 512×512. Based on the information in Fig. P2.27(a), a CCD chip with this resolution will be of size $(16\mu)\times(512)=8$ mm on each side. Substituting $\Delta x=8$ mm in the above equation gives $z=9\lambda$ as the relationship between the distance z and the focal length of the lens, where a minus sign was ignored because it is just a coordinate inversion. If a 25 mm lens is used, the front of the lens will have to be located at approximately 225 mm from the viewing screen so that the size of the

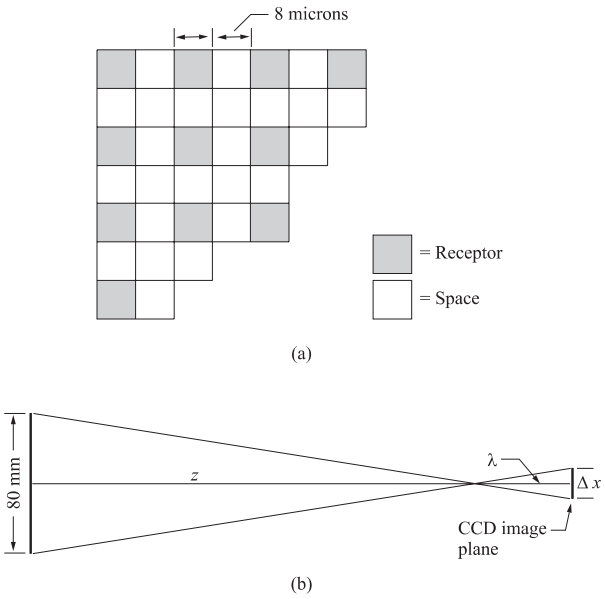


Figure P2.27

image of the screen projected onto the CCD image plane does not exceed the 8 mm size of the CCD chip for the 512×512 camera. This value of z is reasonable, but any other given lens sizes would be also; the camera would just have to be positioned further away.

Assuming a 25 mm lens, the next issue is to determine if the smallest defect will be imaged on, at least, a 2×2 pixel area, as required by the specification. It is given that the defects are circular, with the smallest defect having a diameter of 0.8 mm. So, all that needs to be done is to determine if the image of a circle of diameter 0.8 mm or greater will, at least, be of size 2×2 pixels on the CCD imaging plane. This can be determined by using the same model as in Fig. P2.27(b) with the 80 mm replaced by 0.8 mm. Using $\lambda = 25$ mm and $z = 225$ mm in the above equation yields $\Delta x = 100\ \mu$. In other words, a circular defect of diameter 0.8 mm will be imaged as a circle with a diameter of 100 μ on the CCD chip of a 512×512 camera equipped with a 25 mm lens and which views the defect at a distance of 225 mm.

If, in order for a CCD receptor to be activated, its area has to be excited in its entirety, then, it can be seen from Fig. P2.27(a) that to guarantee that a 2×2 array of such receptors will be activated, a circular area of diameter no less than $(6)(8) = 48\ \mu$ has to be imaged onto the CCD chip. The smallest defect is imaged as a circle with diameter of 100 μ, which is well above the 48 μ minimum requirement.

Therefore, we conclude that a CCD camera of resolution 512×512 pixels, using a 25 mm lens and imaging the viewing screen at a distance of 225 mm, is sufficient to solve the problem posed by the plant manager.

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:

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

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 $[0, L-1]$

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

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

Problem 3.2

(a) General form: $s = T(r) = Ae^{-Kr^2}$. For the condition shown in the problem figure, $Ae^{-KL_0^2} = A/2$. Solving for K yields

$$\begin{aligned} -KL_0^2 &= \ln(0.5) \\ K &= 0.693/L_0^2. \end{aligned}$$

Then,

$$s = T(r) = Ae^{-\frac{0.693}{L_0^2} r^2}.$$

(b) General form: $s = T(r) = B(1 - e^{-Kr^2})$. For the condition shown in the prob-

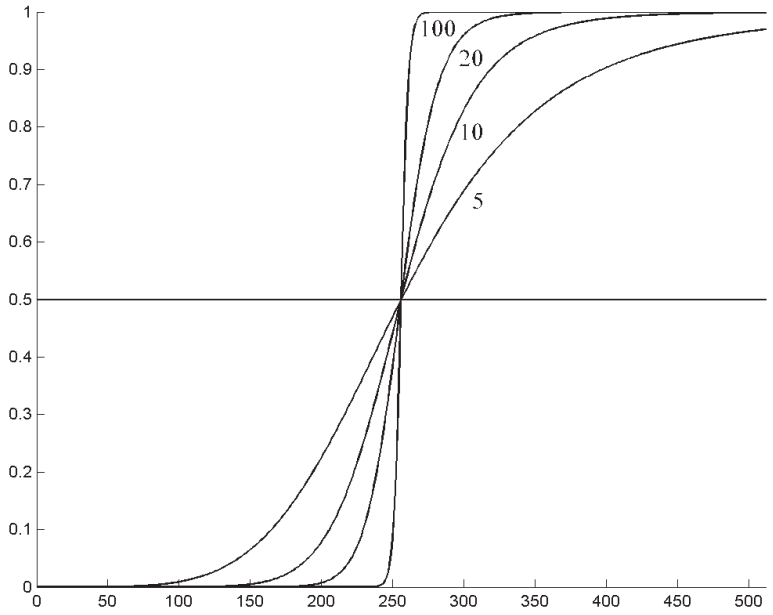


Figure P3.3

lem figure, $B(1-e^{-KL_0^2}) = B/2$. The solution for K is the same as in (a), so

$$s = T(r) = B(1 - e^{-\frac{0.693}{L_0^2} r^2})$$

(c) General form: $s = T(r) = (D-C)(1-e^{-Kr^2}) + C$.

Problem 3.3

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

(b) See Fig. P3.3.

(c) We want s to be 0 for $r < m$, and s to be 1 for values of $r > m$. When $r = m$, $s = 1/2$. But, because the values of r are integers, the behavior we want is

$$s = T(r) = \begin{cases} 0.0 & \text{when } r \leq m-1 \\ 0.5 & \text{when } r = m \\ 1.0 & \text{when } r \geq m+1. \end{cases}$$

The question in the problem statement is to find the smallest value of E that will make the threshold behave as in the equation above. When $r = m$, we see from (a) that $s = 0.5$, regardless of the value of E. If C is the smallest positive

number representable in the computer, and keeping in mind that s is positive, then any value of s less than $C/2$ will be called 0 by the computer. To find the smallest value of E for which this happens, simply solve the following equation for E, using the given value $m = 128$:

$$\frac{1}{1+[m/(m-1)]^E} < C/2.$$

Because the function is symmetric about m, the resulting value of E will yield $s = 1$ for $r \geq m+1$.

Problem 3.4

The transformations required to produce the individual bit planes are nothing more than mappings of the truth table for eight binary variables. In this truth table, the values of the 8th bit are 0 for byte values 0 to 127, and 1 for byte values 128 to 255, thus giving the transformation mentioned in the problem statement. Note that the given transformed values of either 0 or 255 simply indicate a binary image for the 8th bit plane. Any other two values would have been equally valid, though less conventional.

Continuing with the truth table concept, the transformation required to produce an image of the 7th bit plane outputs a 0 for byte values in the range [0, 63], a 1 for byte values in the range [64, 127], a 0 for byte values in the range [128, 191], and a 1 for byte values in the range [192, 255]. Similarly, the transformation for the 6th bit plane alternates between eight ranges of byte values, the transformation for the 5th bit plane alternates between 16 ranges, and so on. Finally, the output of the transformation for the lowest-order bit plane alternates between 0 and 255 depending on whether the byte values are even or odd. Thus, this transformation alternates between 128 byte value ranges, which explains why an image of that bit plane is usually the "busiest" looking of all the bit plane images.

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.

(b) The most visible effect would be significant darkening of the image. For example, dropping the highest bit would limit the brightest level in an 8-bit im-

age to be 127. Because the number of pixels would remain constant, the height of some of the histogram peaks would increase. The general shape of the histogram would now be taller and narrower, with no histogram components being located past 127.

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 3.7

Let $n = MN$ be the total number of pixels and let n_{r_j} be the number of pixels in the input image with intensity value r_j. Then, the histogram equalization transformation is

$$s_k = T(r_k) = \sum_{j=0}^{k} n_{r_j}/n = \frac{1}{n}\sum_{j=0}^{k} n_{r_j}.$$

Because every pixel (and no others) with value r_k is mapped to value s_k, it follows that $n_{s_k} = n_{r_k}$. A second pass of histogram equalization would produce values v_k according to the transformation

$$v_k = T(s_k) = \frac{1}{n}\sum_{j=0}^{k} n_{s_j}.$$

But, $n_{s_j} = n_{r_j}$, so

$$v_k = T(s_k) = \frac{1}{n}\sum_{j=0}^{k} n_{r_j} = s_k$$

which shows that a second pass of histogram equalization would yield the same result as the first pass. We have assumed negligible round-off errors.

Problem 3.8

The general histogram equalization transformation function is

$$s = T(r) = \int_0^r p_r(w)\,dw.$$

There are two important points about which the student must show awareness in answering this problem. First, this equation assumes only positive values for r. However, the Gaussian density extends in general from $-\infty$ to ∞. Recognition of this fact is important. Once recognized, the student can approach this difficulty in several ways. One good answer is to make some assumption, such as the standard deviation being small enough so that the area of the curve under $p_r(r)$ for negative values of r is negligible. Another is to scale up the values until the area under the negative part of the curve is negligible. The second major point is to recognize is that the transformation function itself,

$$s = T(r) = \frac{1}{\sqrt{2\pi}\sigma}\int_0^r e^{-\frac{(w-m)^2}{2\sigma^2}}\,dw$$

has no closed-form solution. This is the cumulative distribution function of the Gaussian density, which is either integrated numerically, or its values are looked up in a table. A third, less important point, that the student should address is the high-end values of r. Again, the Gaussian PDF extends to $+\infty$. One possibility here is to make the same assumption as above regarding the standard deviation. Another is to divide by a large enough value so that the area under the positive part of the PDF past that point is negligible (this scaling reduces the standard deviation).

Another approach the student can take is to work with histograms, in which case the transformation function would be in the form of a summation. The issue of negative and high positive values must still be addressed, and the possible answers suggested above regarding these issues still apply. The student needs to indicate that the histogram is obtained by sampling the continuous function, so some mention should be made regarding the number of samples (bits) used. The most likely answer is 8 bits, in which case the student needs to address the scaling of the function so that the range is [0, 255].

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

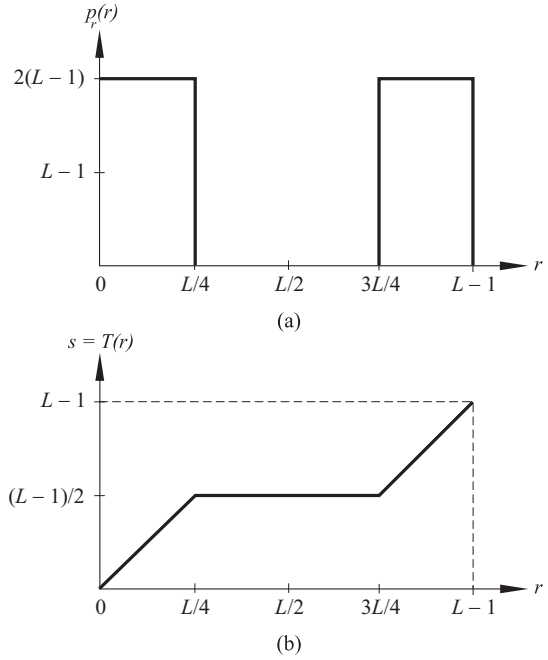


Figure P3.9.

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 conditions (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

(a) We need to show that the transformation function in Eq. (3.3-8) is monotonic in the range $[0, L-1]$, and that its values are in the range $[0, L-1]$. From Eq. (3.3-8)

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j)$$

Because all the $p_r(r_j)$ are positive, it follows that $T(r_k)$ is monotonic in the range $k \in [0, L-1]$. Because the sum of all the values of $p_r(r_j)$ is 1, it follows that $0 \leq s_k \leq L-1$.

(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 3.11

First, we obtain the histogram equalization transformation:

$$s = T(r) = \int_0^r p_r(w)\,dw = \int_0^r (-2w+2)\,dw = -r^2 + 2r.$$

Next we find

$$v = G(z) = \int_0^z p_z(w)\,dw = \int_0^z 2w\,dw = z^2.$$

Finally,

$$z = G^{-1}(v) = \pm\sqrt{v}.$$

But only positive intensity levels are allowed, so $z = \sqrt{v}$. Then, we replace v with s, which in turn is $-r^2 + 2r$, and we have

$$z = \sqrt{-r^2 + 2r}.$$

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.

(b) Similarly, the histogram $h_{\text{diff}}(u_k)$ of the difference has the same components as h_f but their locations are moved left by an amount c as a result of the subtraction operation.

(c) Following the same reasoning, the values (heights) of the components of histogram $h_{\text{prod}}(u_k)$ of the product are the same as h_f, but their locations are at $u_k = c \times r_k$. Note that while the spacing between components of the resulting histograms in (a) and (b) was not affected, the spacing between components of $h_{\text{prod}}(u_k)$ will be spread out by an amount c.

(d) Finally, assuming that $c \neq 0$, the components of $h_{\text{div}}(u_k)$ are the same as those of h_f, but their locations will be at $u_k = r_k/c$. Thus, the spacing between components of $h_{\text{div}}(u_k)$ will be compressed by an amount equal to $1/c$. The preceding solutions are applicable if image $f(x, y)$ is constant also. In this case

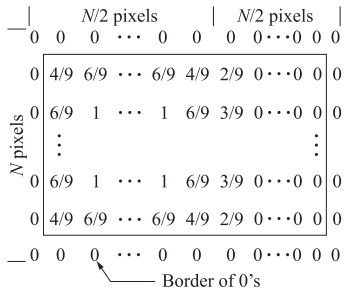


Figure P3.14

the four histograms just discussed would each have only one component. Their location would be affected as described (a) through (d).

Problem 3.14

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

(b) To handle the border effects, we surround the image with a border of 0s. We assume that image is of size $N \times N$ (the fact that the image is square is evident from the right image in the problem statement). Blurring is implemented by a 3×3 mask whose coefficients are 1/9. Figure P3.14 shows the different types of values that the blurred left image will have (see image in the problem statement). The values are summarized in Table P3.14-1. It is easily verified that the sum of the numbers on the left column of the table is N^2. A histogram is easily constructed from the entries in this table. A similar (tedious) procedure yields the results in Table P3.14-2.

Table P3.14-1

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

Table P3.14-2

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

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.

(b) The computational advantage is

$$A = \frac{n^2 - 1}{n+1} = \frac{(n+1)(n-1)}{(n+1)} = n - 1.$$

The plot of A as a function of n is a simple linear function starting at $A = 1$ for $n = 2$.

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.

(b) The only difference between convolution and correlation is that the mask is rotated by 180°. This does not affect the conclusions reached in (a), so correlating an image with a mask whose coefficients sum to zero will produce a correlation image whose elements also sum to zero.

Problem 3.17

One of the easiest ways to look at repeated applications of a spatial filter is to use superposition. Let $f(x, y)$ and $h(x, y)$ denote the image and the filter function, respectively. Assuming square images of size $N \times N$ for convenience, we can express $f(x, y)$ as the sum of at most N^2 images, each of which has only one nonzero pixel (initially, we assume that N can be infinite). Then, the process of running $h(x, y)$ over $f(x, y)$ can be expressed as the following convolution:

$$h(x,y) \star f(x,y) = h(x,y) \star \left[f_1(x,y) + f_2(x,y) + \cdots + f_{N^2}(x,y) \right].$$

Suppose for illustrative purposes that $f_i(x,y)$ has value 1 at its center, while the other pixels are valued 0, as discussed above (see Fig. P3.17a). If $h(x,y)$ is a 3×3 mask of 1/9's (Fig. P3.17b), then convolving $h(x,y)$ with $f_i(x,y)$ will produce an image with a 3×3 array of 1/9's at its center and 0s elsewhere, as Fig. P3.17(c) shows. If $h(x,y)$ is now applied to this image, the resulting image will be as shown in Fig. P3.17(d). Note that the sum of the nonzero pixels in both Figs. P3.17(c) and (d) is the same, and equal to the value of the original pixel. Thus, it is intuitively evident that successive applications of $h(x,y)$ will "diffuse" the nonzero value of $f_i(x,y)$ (not an unexpected result, because $h(x,y)$ is a blurring filter). Since the sum remains constant, the values of the nonzero elements will become smaller and smaller, as the number of applications of the filter increases. The overall result is given by adding all the convolved $f_k(x,y)$, for $k = 1, 2, ..., N^2$.

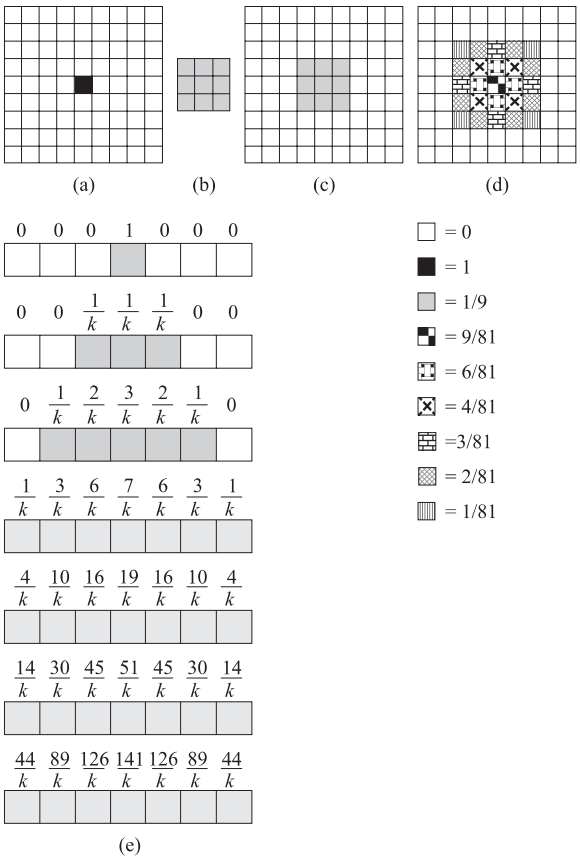


Figure P3.17

It is noted that every iteration of blurring further diffuses the values outwardly from the starting point. In the limit, the values would get infinitely small, but, because the average value remains constant, this would require an image of infinite spatial proportions. It is at this junction that border conditions become important. Although it is not required in the problem statement, it is instructive to discuss in class the effect of successive applications of $h(x,y)$ to an image of finite proportions. The net effect is that, because the values cannot diffuse outward past the boundary of the image, the denominator in the successive applications of averaging eventually overpowers the pixel values, driving the image to zero in the limit. A simple example of this is given in Fig. P3.17(e), which shows an array of size 1×7 that is blurred by successive applications of the 1×3 mask $h(y) = \frac{1}{3}[1,1,1]$. We see that, as long as the values of the blurred 1 can diffuse out, the sum, S, of the resulting pixels is 1. However, when the boundary is met,

an assumption must be made regarding how mask operations on the border are treated. Here, we used the commonly made assumption that pixel value immediately past the boundary are 0. The mask operation does not go beyond the boundary, however. In this example, we see that the sum of the pixel values begins to decrease with successive applications of the mask. In the limit, the term $1/(3)^n$ would overpower the sum of the pixel values, yielding an array of 0s.

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.

(b) For the conclusion reached in (a) to hold, the number of points that we consider cluster (object) points can never exceed $(n^2-1)/2$. Thus, two or more different clusters cannot be in close enough proximity for the filter mask to encompass points from more than one cluster at any mask position. It then follows that no two points from different clusters can be closer than the diagonal dimension of the mask minus one cell (which can be occupied by a point from one of the clusters). Assuming a grid spacing of 1 unit, the minimum distance between any two points of different clusters then must greater than $\sqrt{2}(n-1)$. In other words, these points must be separated by at least the distance spanned by $n-1$ cells along the mask diagonal.

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 in the appropriate locations in the sorted array.

Problem 3.20

(a) The most extreme case is when the mask is positioned on the center pixel of a 3-pixel gap, along a thin segment, in which case a 3×3 mask would encompass a completely blank field. Since this is known to be the largest gap, the next (odd) mask size up is guaranteed to encompass some of the pixels in the segment. Thus, the smallest mask that will do the job is a 5×5 averaging mask.

(b) The smallest average value produced by the mask is when it encompasses only two pixels of the segment. This average value is a gray-scale value, not binary, like the rest of the segment pixels. Denote the smallest average value by A_{min}, and the binary values of pixels in the thin segment by B. Clearly, A_{min} is less than B. Then, setting the binarizing threshold slightly smaller than A_{min} will create one binary pixel of value B in the center of the mask.

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.

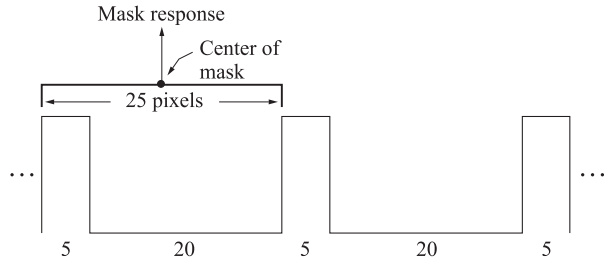


Figure P3.21

Problem 3.22

There are at most q^2 points in the area in which we want to reduce the intensity level of each pixel to one-tenth its original value. Consider an averaging mask of size $n \times n$ encompassing the $q \times q$ neighborhood. The averaging mask has n^2 points of which we are assuming that q^2 points are from the object and the rest from the background. Note that this assumption implies separation between objects that, at a minimum, is equal to the area of the mask all around each object. The problem becomes intractable unless this assumption is made. This condition was not given in the problem statement on purpose in order to force the student to arrive at that conclusion. If the instructor wishes to simplify the problem, this should then be mentioned when the problem is assigned. A further simplification is to tell the students that the intensity level of the background is 0.

Let B represent the intensity level of background pixels, let a_i denote the intensity levels of points inside the mask and o_i the levels of the objects. In addition, let S_a denote the set of points in the averaging mask, S_o the set of points in the object, and S_b the set of points in the mask that are not object points. Then, the response of the averaging mask at any point on the image can be written as

$$
\begin{aligned}
R &= \frac{1}{n^2}\sum_{a_i \in S_a} a_i \\
&= \frac{1}{n^2}\left[\sum_{o_j \in S_o} o_j + \sum_{a_k \in S_b} a_k\right] \\
&= \frac{1}{n^2}\left[\frac{q^2}{q^2}\sum_{o_j \in S_o} o_j\right] + \frac{1}{n^2}\left[\sum_{a_k \in S_b} a_k\right] \\
&= \frac{q^2}{n^2}\overline{Q} + \frac{1}{n^2}\left[(n^2 - q^2)B\right]
\end{aligned}
$$

where $\overline{Q}$ denotes the average value of object points. Let the maximum expected average value of object points be denoted by $\overline{Q}_{\max}$. Then we want the response of the mask at any point on the object under this maximum condition to be less than one-tenth $\overline{Q}_{\max}$, or

$$\frac{q^2}{n^2}\overline{Q}_{\max} + \frac{1}{n^2}\left[(n^2 - q^2)B\right] < \frac{1}{10}\overline{Q}_{\max}$$

from which we get the requirement

$$n > q\left[\frac{10(\overline{Q}_{\max} - B)}{(\overline{Q}_{\max} - 10B)}\right]^{1/2}$$

for the minimum size of the averaging mask. Note that if the background intensity is 0, then the minimum mask size is $n < \sqrt{10}q$. If this was a fact specified by the instructor, or the student made this assumption from the beginning, then this answer follows almost by inspection.

Problem 3.23

The student should realize that both the Laplacian and the averaging process are linear operations, so it makes no difference which one is applied first.

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.

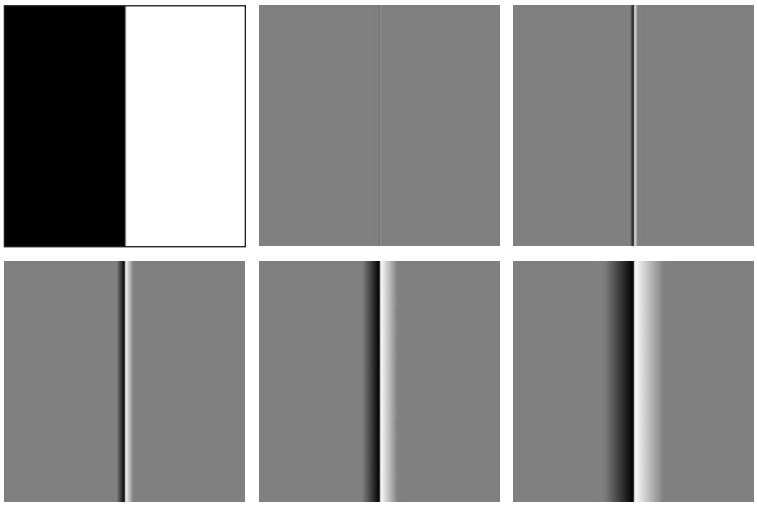


Figure P3.26

Problem 3.26

(a) The size and coefficients of the 3×3 Laplacian mask come directly from the *definition* of the Laplacian given in Eq. (3.6-6). In other words, the number of coefficients (and thus size of the mask) is a direct result of the definition of the second derivative. A larger "Laplacian-like" mask would no longer be implementing a second derivative, so we cannot expect that in general it will give sharper results. In fact, as explained in part (b), just the opposite occurs.

(b) In general, increasing the size of the "Laplacian-like" mask produces blurring. To see why this is so, consider an image consisting of two vertical bands, a black band on the left and a white band on the right, with the transition between the bands occurring through the center of the image. That is, the image has a sharp vertical edge through its center. From Fig. 3.36 we know that a second derivative should produce a double edge in the region of the vertical edge when a 3×3 Laplacian mask is centered on the edge. As the center of the mask moves more than two pixels on either side of the edge the entire mask will encompass a constant area and its response would be zero, as it should. However, suppose that the mask is much larger. As its center moves through, say, the black (0) area, one half of the mask will be totally contained in that area. However, depending on its size, part of the mask will be contained in the white (255) area. The sum of products will therefore be different from 0. This means that there will be a response in an area where the response should have been 0 because the mask is centered on a constant area. Figure P3.26 shows these effects. The

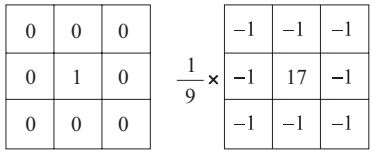


Figure P3.27

figure on the top left is the band just mentioned and the figure next to it is the result of using a 3×3 Laplacian mask with an 8 in the center. The other figures shows the results of using "Laplacian-like" masks of sizes 15×15, 35×35, 75×75, and 125×125, respectively. The progressively increasing blurring as a result of mask size is evident in these results.

Problem 3.27

With reference to Eqs. (3.6-8) and (3.6-9), and using $k = 1$ we can write the following equation for unsharp masking:

$$\begin{aligned} g(x,y) &= f(x,y) + f(x,y) - \bar{f}(x,y) \\ &= 2f(x,y) - \bar{f}(x,y) \end{aligned}$$

Convolving $f(x,y)$ with the mask in Fig. P3.27(a) produces $f(x,y)$. Convolving $f(x,y)$ with the mask in Fig. 3.32(a) produces $\bar{f}(x,y)$. Then, because these operations are linear, we can use superposition, and we see from the preceding equation that using two masks of the form in Fig. P3.27(a) and the mask in Fig. 3.32(a) produces the composite mask in Fig. P3.27(b). Convolving this mask with $f(x,y)$ produces $g(x,y)$, the unsharp result.

Problem 3.28

Consider the following equation:

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

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

where $\overline{f}(x,y)$ denotes the average of $f(x,y)$ in a predefined neighborhood 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 3.29

(a) From Problem 3.24, $f(x,y)$

$$\frac{\partial f}{\partial x'} = \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta$$

and

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

from which it follows that

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

or

$$\left[\left(\frac{\partial f}{\partial x'}\right)^2 + \left(\frac{\partial f}{\partial y'}\right)^2\right]^{1/2} = \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2}.$$

Thus, we see that the magnitude of the gradient is an isotropic operator.

(b) From Eq. (3.6-10), (3.6-12), and the preceding results,

$$|G_x| = \left|\frac{\partial f}{\partial x}\right| \qquad |G_y| = \left|\frac{\partial f}{\partial y}\right|,$$

$$|G_{x'}| = \left|\frac{\partial f}{\partial x'}\right| = \left|\frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta\right|,$$

and

$$\left|G_{y'}\right| = \left|\frac{\partial f}{\partial y'}\right| = \left|-\frac{\partial f}{\partial x}\sin\theta + \frac{\partial f}{\partial y}\cos\theta\right|.$$

Clearly, $|G_{x'}| + \left|G_{y'}\right| \neq |G_x| + \left|G_y\right|$.

Problem 3.30

It is given that the range of illumination stays in the linear portion of the camera response range, but no values for the range are given. The fact that images stay in the linear range implies that images will not be saturated at the high end or be driven in the low end to such an extent that the camera will not be able to respond, thus losing image information irretrievably. The only way to establish a benchmark value for illumination is when the variable (daylight) illumination is not present. Let $f_0(x,y)$ denote an image taken under artificial illumination only, with no moving objects (e.g., people or vehicles) in the scene. This becomes the standard by which all other images will be normalized. There are numerous ways to solve this problem, but the student must show awareness that areas in the image likely to change due to moving objects should be excluded from the illumination-correction approach.

One way is to select various representative subareas of $f_0(x,y)$ not likely to be obscured by moving objects and compute their average intensities. We then select the minimum and maximum of all the individual average values, denoted by, $\overline{f}_{\min}$ and $\overline{f}_{\max}$. The objective then is to process any input image, $f(x,y)$, so that its minimum and maximum will be equal to $\overline{f}_{\min}$ and $\overline{f}_{\max}$, respectively. The easiest way to do this is with a linear transformation function of the form

$$f_{\text{out}}(x,y) = af(x,y) + b$$

where f_{out} is the scaled output image. It is easily verified that the output image will have the required minimum and maximum values if we choose

$$a = \frac{\overline{f}_{\max} - \overline{f}_{\min}}{f_{\max} - f_{\min}}$$

and

$$b = \frac{\overline{f}_{\min} f_{\max} - \overline{f}_{\max} f_{\min}}{f_{\max} - f_{\min}}$$

where $f_{\max}$ and $f_{\min}$ are the maximum and minimum values of the input image.

Note that the key assumption behind this method is that all images stay within the linear operating range of the camera, thus saturation and other nonlinearities are not an issue. Another implicit assumption is that moving objects comprise a relatively small area in the field of view of the camera, otherwise these objects would overpower the scene and the values obtained from $f_0(x,y)$ would not make sense. If the student selects another automated approach (e.g., histogram equalization), he/she must discuss the same or similar types of assumptions.

Problem 3.31

Substituting directly into Eq. (3.8-9)

$$\begin{aligned} 2\left[\frac{b-a}{c-a}\right]^2 &= \frac{1}{2} \\ \left[\frac{b-a}{c-a}\right] &= \frac{1}{2} \\ b &= \frac{a+c}{2}. \end{aligned}$$

Problem 3.32

All figures can be constructed using various membership functions from Fig. 3.46 and the definition of fuzzy union (the pointwise maxima between the curves), as follows.

(a) The figure in part (a) of the problem statement can be constructed using two trapezoidal functions [Figs. 3.46(b)] and one triangular function [Fig. 3.46(a)], as Fig. P3.32(a) shows.

(b) The figure in part (b) of the problem statement can be constructed using two trapezoidal functions in which the one on the right terminates vertically, as Fig. P3.32(b) shows.

(c) The figure in part (c) of the problem statement can be constructed using two triangular functions, as Fig. P3.32(c) shows.

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

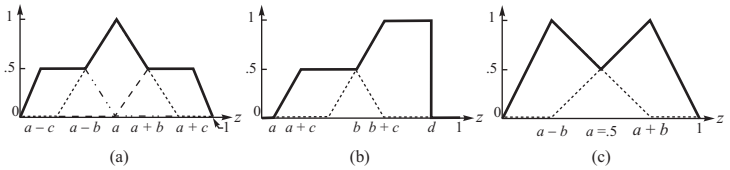


Figure P3.32

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

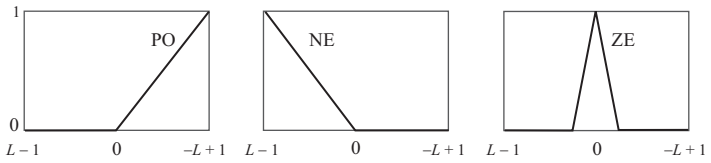
IF d_2 is PO AND d_4 is PO AND d_6 is PO AND d_8 is PO THEN v is PO
IF d_2 is NE AND d_4 is NE AND d_6 is NE AND d_8 is NE THEN v is NE
ELSE v is ZR.

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).

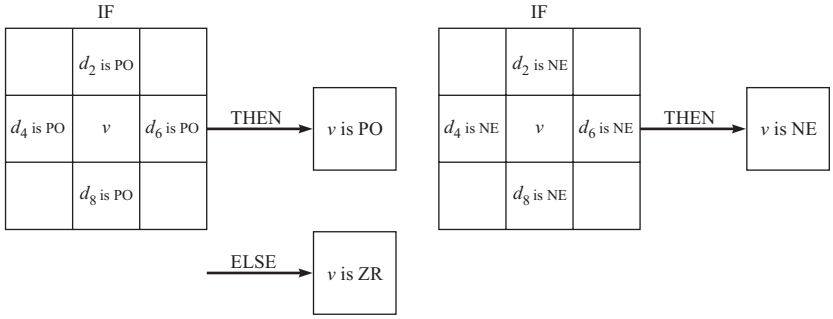
(c) The membership functions PO and NE are triangles from -1 to 0 and 0 to 1, respectively. Membership function ZR is also a triangle. It is centered on 0 and overlaps the other two slightly. Figure P3.34(a) shows these three membership functions.

(d) Figure P3.34(b) shows in graphic form the three rules from part (b). The off-center elements in the 3×3 neighborhoods are differences $d_i = z_i - z_5$, for $i = 2, 4, 6, 8$. The value v in the center is the correction that will be added to z_5 to obtain the new intensity, z_5' , in that location.

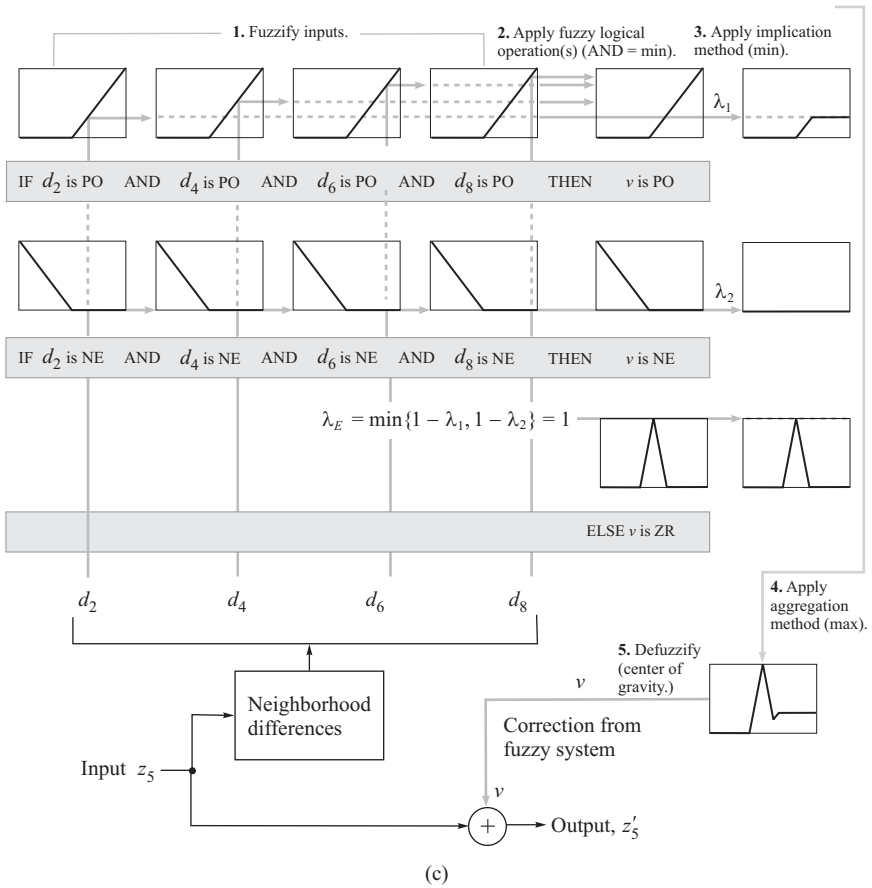
(e) Figure P3.34(c) describes the fuzzy system responsible for generating the correction factor, v. This diagram is similar to Fig. 3.52, with two main differences: (1) the rules are composed of ANDs, meaning that we have to use the min operation when applying the logical operations (step 2 in Fig. 3.52); and (2) we show how to implement an ELSE rule. This rule is nothing more that computing 1 minus the minimum value of the outputs of step 2, and using the result to clip the ZR membership function. It is important to understand that the output of the fuzzy system is the center of gravity of the result of aggregation (step 4 in Fig. 3.52). Thus, for example, if both the first and second rules were to produce a minimum of 0, the "strength" of the ELSE rule would be 1. This would produce the complete ZR membership function in the implication step (step 3 in Fig. 3.52). The other two results would be zero, so the result of aggregation would be the ZR function. The center of gravity of this function is 0, so the output of the system would be $v = 0$, indicating no correction, as it should be. For the particular values of the $d_i's$ shown in Fig. P3.34(c), the aggregated function is biased to the right, so the value of the correction (center of gravity) will be positive. This is as it should be because the differences are all positive, indicating that the value of z_5 is less than the value of its 4-neighbors.



(a)



(b)



(c)

Figure 3.34

Chapter 4

Problem Solutions

Problem 4.1

$$
\begin{aligned}
F(\mu) &= \int_{-\infty}^{\infty} f(t)e^{-j2\pi\mu t}\,dt \\
&= \int_{0}^{T} Ae^{-j2\pi\mu t}\,dt \\
&= \frac{-A}{j2\pi\mu}\left[e^{-j2\pi\mu t}\right]_0^T = \frac{-A}{j2\pi\mu}\left[e^{-j2\pi\mu t}-1\right] \\
&= \frac{A}{j2\pi\mu}\left[e^{j\pi\mu T}-e^{-j\pi\mu T}\right]e^{-j\pi\mu T} \\
&= \frac{A}{\pi\mu}\sin\left(\pi\mu T\right)e^{-j\pi\mu T} \\
&= AT\left[\frac{\sin\left(\pi\mu T\right)}{\left(\pi\mu T\right)}\right]e^{-j\pi\mu T}.
\end{aligned}
$$

If we let $T = W$, the only difference between this result and the result in Example 4.1 is the exponential term. It is a phase term that accounts for the shift in the function. The magnitude of the Fourier transform is the same in both cases, as expected.

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),

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

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 we have 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),

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

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),

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

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 $(1)\, e^{j2\pi n t}$, which is $\delta(\mu - n)$, and similarly, the second integral is the transform of $(1)\, e^{-j2\pi n t}$, or $\delta(\mu + n)$. Combining all results yields

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

as desired.

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 $\Delta T =$

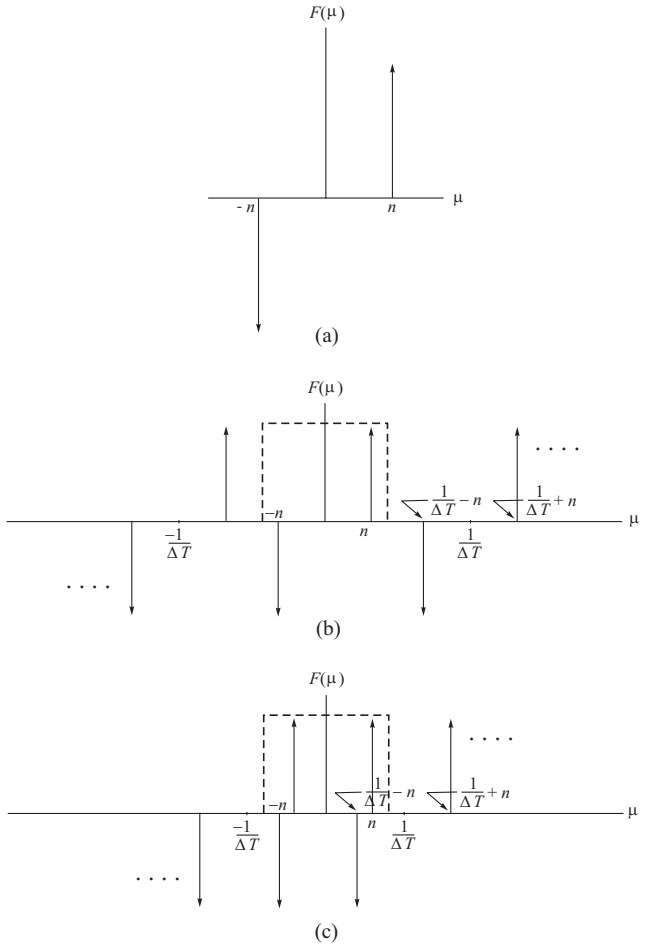


Figure P4.4

$1/2n$ 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.

(d) When the sampling rate is less than the Nyquist rate, we can have a situation such as the one illustrated in Fig. P4.4(c), which is the sum of two sine waves in this case. For some values of sampling, the sum of the two sines combine to form a single sine wave and a plot of the samples would appear as in Fig. 4.8 of the book. Other values would result in functions whose samples can describe any shape obtainable by sampling the sum of two sines.

Problem 4.5

Starting from Eq. (4.2-20),

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

The Fourier transform of this expression is

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

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

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

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 4.6

$$\begin{aligned}
f(t) &= h(t)\bigstar\tilde{f}(t) \\
&= \int_{-\infty}^{\infty} h(z)\tilde{f}(t-z)dz \\
&= \int_{-\infty}^{\infty} \frac{\sin(\pi z/\Delta T)}{(\pi z/\Delta T)} \sum_{n=-\infty}^{\infty} f(t-z)\delta(t-n\Delta T-z)dz \\
&= \sum_{n=-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{\sin(\pi z/\Delta T)}{(\pi z/\Delta T)} f(t-z)\delta(t-n\Delta T-z)dz \\
&= \sum_{n=-\infty}^{\infty} f(n\Delta T)\frac{\sin[\pi(t-n\Delta T)/\Delta T]}{[\pi(t-n\Delta T)/\Delta T]} \\
&= \sum_{n=-\infty}^{\infty} f(n\Delta T)\mathrm{sinc}[(t-n\Delta T)/\Delta T].
\end{aligned}$$

Problem 4.7

The tent function is obtained by convolving two boxes, and recall that the transform of a box is a sinc function. Because, by the convolution theorem, the Fourier transform of the spatial convolution of two functions is the product their transforms, it follows that the Fourier transform of a tent function is a sinc function squared.

Problem 4.8

(a) We solve this problem by direct substitution using orthogonality. Substituting Eq. (4.4-5) into (4.4-4) yields

$$\begin{aligned}
F_m &= \sum_{n=0}^{M-1}\left[\frac{1}{M}\sum_{r=0}^{M-1} F_r e^{-j2\pi rn/M}\right] e^{-j2\pi mn/M} \\
&= \frac{1}{M}\sum_{r=0}^{M-1} F_r\left[\sum_{n=0}^{M-1} e^{-j2\pi rn/M}e^{-j2\pi mn/M}\right] \\
&= F_m
\end{aligned}$$

where the last step follows from the orthogonality condition given in the problem statement. Substituting Eq. (4.4-4) into (4.6-5) and using the same basic procedure yields a similar identity for f_n.

(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 4.9

To prove infinite periodicity we have to show that $F(u+kM) = F(u)$ for $k = 0, \pm 1, \pm 2, \ldots$. We do this by direct substitution into Eq. (4.4-6):

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

where the last step follows from the fact that $e^{-j2\pi kx} = 1$ because k and x are integers. Note that this holds for positive *and* negative values of k. We prove the validity of Eq. (4.4-9) in a similar way.

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 4.12

The limit of the imaging system is 1 pixel per square, so each row and column in the resulting image would be of the form . . . 101010101 . . . . The period of this waveform is $P = 2$ mm, so the max frequency is $\mu = 1/P = 0.5$ cycles/mm. To avoid aliasing we have to sample at a rate that exceeds twice this frequency or $2(0.5) = 1$ sample/mm. That is, the sampling rate would have to exceed 1 sample/mm. So, each square has to correspond to slightly more than one pixel in the imaging system.

Problem 4.13

Shrinking can cause aliasing because the effective sampling rate is reduced. This is not the case in zooming, which introduces additional samples. Although no new detail in introduced by zooming, it certainly does not reduce the sampling rate, so zooming cannot result in aliasing.

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

$$\Im\left[a_1f_1(t,z)+a_2f_2(t,z)\right]=a_1\Im\left[f_1(t,z)\right]+a_2\Im\left[f_2(t,z)\right].$$

Substituting into the definition of the Fourier transform yields

$$\begin{aligned}\Im\left[a_1f_1(t,z)+a_2f_2(t,z)\right] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[a_1f_1(t,z)+a_2f_2(t,z)\right]\\ &\qquad\times e^{-j2\pi(\mu t+vz)}dt\,dz\\ &= a_1\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(t,z)e^{-j2\pi(\mu t+vz)}dt\,dz\\ &\qquad+a_2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f_2(t,z)e^{-j2\pi(\mu t+vz)}dt\,dz\\ &= a_1\Im\left[f_1(t,z)\right]+a_2\Im\left[f_2(t,z)\right].\end{aligned}$$

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

$$\begin{aligned}\Im\left[a_1f_1(x,y)+a_2f_2(x,y)\right] &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[a_1f_1(x,y)+a_2f_2(x,y)\right]e^{-j2\pi(ux/M+vy/N)}\\ &= a_1\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f_1(x,y)e^{-j2\pi(ux/M+vy/N)}\\ &\qquad+a_2\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f_2(x,y)e^{-j2\pi(ux/M+vy/N)}\\ &= a_1\Im\left[f_1(x,y)\right]+a_2\Im\left[f_2(x,y)\right].\end{aligned}$$

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

Problem 4.15

Select an image of your choice and compute its average value:

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

Compute the DFT of $f(x,y)$ and obtain $F(0,0)$. If $F(0,0) = MN\bar{f}(x,y)$, then $1/MN$ was included in front of the IDFT [see Eqs. (4.5-15), (4.5-16) and (4.6-21)]. Similarly, if $F(0,0) = \bar{f}(x,y)$ the 1/MN term was included in front of the DFT. Finally, if $F(0,0)=\sqrt{MN}f(x,y)$, the term $1/\sqrt{MN}$ was included in the formulation of both the DFT and IDFT.

Problem 4.16

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

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

(b) From Eq. (4.5-16),

$$\begin{aligned}\Im^{-1}\left[F(u,v)e^{-j2\pi(ux_0+uy_0)}\right] &= \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\left[F(u,v)e^{-j2\pi(ux_0+uy_0)}\right] \\ &\qquad \times e^{j2\pi(ux/M+vy/N)} \\ &= \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v) \\ &\qquad \times e^{j2\pi[(u(x-x_0)/M+v(y-y_0)/N)]} \\ &= f(x-x_0,y-y_0).\end{aligned}$$

Problem 4.17

From Eq. (4.5-7),

$$\begin{aligned}F(\mu,v) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(t,z)\,e^{-j2\pi(\mu t+vz)}dt\,dz \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sin\left(2\pi\mu_0 t+2\pi v_0 z\right)e^{-j2\pi(\mu t+vz)}dt\,dz.\end{aligned}$$

Expressing the sine function in terms of exponentials yields

$$\begin{aligned} F(\mu,\nu) &= \frac{-j}{2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[e^{j2\pi(\mu_0 t+\nu_0 z)}-e^{-j2\pi(\mu_0 t+\nu_0 z)}\right]e^{-j2\pi(\mu t+\nu z)}dt\,dz \\ &= \frac{-j}{2}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{j2\pi(\mu_0 t+\nu_0 z)}e^{-j2\pi(\mu t+\nu z)}dt\,dz\right] \\ &\quad -\frac{-j}{2}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-j2\pi(\mu_0 t+\nu_0 z)}e^{-j2\pi(\mu t+\nu z)}dt\,dz\right]. \end{aligned}$$

We recognize the terms inside the brackets as the Fourier transforms of

$$(1)e^{j2\pi(\mu_0 t+\nu_0 z)}$$

and

$$(1)e^{-j2\pi(\mu_0 t+\nu_0 z)}.$$

From the problem statement we know that the $\Im[1]=\delta(\mu,\nu)$, and from Table 4.3 we know that the exponential just introduces a shift in the origin of the transform. Therefore, $\Im\left[(1)e^{j2\pi(\mu_0 t+\nu_0 z)}\right]=\delta(\mu-\mu_0,\nu-\nu_0)$, and similarly for the other transform. Plugging these results in the preceding equation yields,

$$\Im\left[\sin\left(2\pi\mu_0 t+2\pi\nu_0 z\right)\right]=\frac{j}{2}\left[\delta(\mu+\mu_0)-\delta(\nu-\nu_0)\right].$$

Problem 4.18

We consider the 1-D case first. From Eq. (4.4-6),

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

When $f(x)=1$ and $u=0$, $F(u)=1$. When $f(x)=1$ and $u\neq 0$,

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

This expression is 0 for any integer value of u in the range $[0,N-1]$. There are various ways of proving this. One of the most intuitive is based on using Euler's formula to express the exponential term as

$$e^{-j2\pi ux/M}=\cos(2\pi ux/M)-j\sin(2\pi ux/M).$$

This expression describes a unit vector in the complex plane. The vector is centered at the origin and its direction depends on the value of the argument. For any integer value of u in the range $[0, N-1]$ the argument ranges over integer values of x in the same range. This means that the vector makes an integer number of revolutions about the origin in equal increments. Thus, for any positive value of $\cos(2\pi ux/M)$, there will be a corresponding negative value of this term. This produces a zero sum for the real part of the exponent. Similar comments apply the imaginary part. Therefore, when $f(x) = 0$ and $u \neq 0$, it follows that $F(u) = 0$. Thus, we have shown that for discrete quantities,

$$\Im[1] = \delta(u) = \begin{cases} 1 & \text{if } u = 0 \\ 0 & \text{if } u \neq 0. \end{cases}$$

A similar procedure applies in the case of two variables, and we have that

$$\Im[1] = \delta(u,v) = \begin{cases} 1 & \text{if } u = 0 \text{ and } v = 0 \\ 0 & \text{otherwise.} \end{cases}$$

Problem 4.19

From Eq. (4.5-15), and using the exponential representation of the sine function, we have

$$\begin{aligned}
F(u,v) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} \sin\left(2\pi u_0 x + 2\pi v_0 y\right) e^{-j2\pi(ux/M+vy/N)} \\
&= \frac{-j}{2}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[e^{j2\pi(u_0x+v_0y)} - e^{-j2\pi(u_0x+v_0y)}\right] e^{-j2\pi(ux/M+vy/N)} \\
&= \frac{-j}{2}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} e^{j2\pi(Mu_0x/M+Nv_0y/N)} e^{-j2\pi(ux/M+vy/N)} \\
&\quad -\frac{-j}{2}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} e^{-j2\pi(Mu_0x/M+Nv_0y/N)} e^{-j2\pi(ux/M+vy/N)} \\
&= \frac{-j}{2}\Im\left[(1)e^{j2\pi(Mu_0x/M+Nv_0y/N)}\right] + \frac{j}{2}\Im\left[(1)e^{-j2\pi(u_0x+v_0y)}\right] \\
&= \frac{j}{2}\left[\delta(u+Mu_0, v+Nv_0) - \delta(u-Mu_0, v-Nv_0)\right]
\end{aligned}$$

where the fourth step follows from the discussion in Problem 4.17, and the last line follows from Table 4.3.

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).

(c) *Property 5:* $f(-x,-y) \Leftrightarrow F^*(u,v)$. That is, if $f(x,y)$ is real and its transform is $F(u,v)$, then the transform of $f(-x,-y)$ is $F^*(u,v)$. And conversely. *Proof:* From Example 4.12,

$$\Im\left[f(-x,-y)\right] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)e^{j2\pi(um/M+vn/N)}.$$

To see what happens when $f(x,y)$ is real, we write the right side of this equation as

$$\begin{aligned}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)e^{j2\pi(um/M+vn/N)} &= \left[\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f^*(m,n)e^{-j2\pi(um/M+vn/N)}\right]^* \\ &= \left[\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)e^{-j2\pi(um/M+vn/N)}\right]^* \\ &= F^*(u,v)\end{aligned}$$

where the second step follows from the fact that $f(x,y)$ is real. Thus, we have shown that

$$\Im\left[f(-x,-y)\right] = F^*(u,v).$$

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

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

(e) *Property 9:* If $f(x,y)$ is real and odd, then $F(u,v)$ is imaginary and odd, and conversely. *Proof:* Because $f(x,y)$ is real, we know that the real part of $F(u,v)$ is even and its imaginary part is odd. If we can show that F is purely imaginary, then we will have completed the proof.

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

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

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

(f) *Property 10:* If $f(x,y)$ is imaginary and even, then $F(u,v)$ is imaginary and even, and conversely. *Proof:* We know that when $f(x,y)$ is imaginary, then the real part of $F(u,v)$ is odd and its imaginary part is even. If we can show that the real part is 0, then we will have proved this property. Because $f(x,y)$ is imaginary, we can express it as $j\,g(x,y)$, where g is a real function. Then,

$$
\begin{aligned}
F(u,v) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)} \\
&= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[j\,g(x,y)\right]\left[e^{-j2\pi(ux/M)}\right]\left[e^{-j2\pi(vy/N)}\right] \\
&= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[j\,\text{even}][\text{even}-j\,\text{odd}][\text{even}-j\,\text{odd}] \\
&= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[j\,\text{even}]\left[(\text{even})(\text{even})-2j(\text{even})(\text{odd})-(\text{odd})(\text{odd})\right] \\
&= j\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[(\text{even})(\text{even})]+2\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[(\text{even})(\text{odd})] \\
&\qquad -j\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[(\text{even})(\text{even})] \\
&= \text{imaginary}
\end{aligned}
$$

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

(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).

(h) *Property 12:* If $f(x,y)$ is complex and even, then $F(u,v)$ is complex and even, and conversely. *Proof:* Here, we have to prove that both the real and imaginary parts of $F(u,v)$ are even. Recall that if $f(x,y)$ is an even function, both its real and imaginary parts are even. Thus, we can write this function as $f(x,y)=f_{re}(x,y)+jf_{ie}(x,y)$. Then,

$$\begin{aligned}
F(u,v) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[f_{re}(x,y)+jf_{ie}(x,y)]e^{-j2\pi(ux/M+vy/N)} \\
&= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f_{re}(x,y)e^{-j2\pi(ux/M+vy/N)} \\
&\qquad +\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}jf_{ie}(x,y)e^{-j2\pi(ux/M+vy/N)}.
\end{aligned}$$

The first term of this result is recognized as the DFT of a real, even function, which we know is a real, even function. The second term is the DFT of a purely imaginary even function, which we know is imaginary and even. Thus, we see that the the transform of a complex, even function, has an even real part and an even imaginary part, and is thus a complex even function. This concludes the proof.

(i) *Property 13:* If $f(x,y)$ is complex and odd, then $F(u,v)$ is complex and odd,

and conversely. *Proof*: The proof parallels the proof in (h).

$$\begin{aligned} F(u,v) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[f_{ro}(x,y)+jf_{io}(x,y)]e^{-j2\pi(ux/M+vy/N)} \\ &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f_{ro}(x,y)e^{-j2\pi(ux/M+vy/N)} \\ &\quad + \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}jf_{io}(x,y)e^{-j2\pi(ux/M+vy/N)}. \end{aligned}$$

The first term is the DFT of an odd, real function, which know is imaginary and odd. The second term is the DFT of purely imaginary odd function, which we know is real and odd. Thus, the sum of the two is a complex, odd function, as we wanted to prove.

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.

(b) Yes, they are equal. We know that $F(0,0) = MN\bar{f}(x,y)$ and $F_p(0,0) = PQ\bar{f}_p(x,y)$. And, from part (a), $\bar{f}_p(x,y) = MN\bar{f}(x,y)/PQ$. Then,

$$\begin{aligned}\frac{F_p(0,0)}{PQ} &= \frac{MN}{PQ}\frac{F(0,0)}{MN} \\ F_p(0,0) &= F(0,0).\end{aligned}$$

Problem 4.24

We solve the problem by direct substitution into the definition of the 2-D DFT:

$$\begin{aligned}F(u+k_1M, v+k_2N) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi([u+k_1M]x/M+[v+k_2N]y/N)} \\ &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}e^{-j2\pi k_1x}e^{-j2\pi k_2y} \\ &= F(u,v)\end{aligned}$$

with k_1 and k_2 having values $0, \pm 1, \pm 2, \ldots$. The last step follows from the fact that k_1x and k_2y are integers, which makes the two rightmost exponentials equal to 1.

Problem 4.25

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

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

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

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

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

(b) As in (a),

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

(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,

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

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

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

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

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

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

(a) From Chapter 3, the Laplacian of a function $f(t,z)$ of two continuous variables is defined as

$$\nabla^2 f(t,z) = \frac{\partial^2 f(t,z)}{\partial t^2} + \frac{\partial^2 f(t,z)}{\partial z^2}.$$

We obtain the Fourier transform of the Laplacian using the result from Problem 4.25 (entry 12 in Table 4.3):

$$\begin{aligned}\Im\left[\nabla^2 f(t,z)\right] &= \Im\left[\frac{\partial^2 f(t,z)}{\partial t^2}\right]+\Im\left[\frac{\partial^2 f(t,z)}{\partial z^2}\right]\\ &= (j2\pi\mu)^2 F(\mu,\nu)+(j2\pi\nu)^2 F(\mu,\nu)\\ &= -4\pi^2(\mu^2+\nu^2)F(\mu,\nu).\end{aligned}$$

We recognize this as the familiar filtering expression $G(\mu,\nu)=H(\mu,\nu)F(\mu,\nu)$, in which $H(\mu,\nu)=-4\pi^2(\mu^2+\nu^2)$.

(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 4.28

(a) As in Problem 4.27, the filtered image is given by:

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

From property 3 in Table 4.3,

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

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

$$\begin{aligned} H(u,v) &= \left[(e^{j2\pi u/M} - 1) + (e^{j2\pi v/N} - 1)\right] \\ &= 2j\left[\sin(\pi u/M)e^{j\pi u/M} + \sin(\pi v/N)e^{j\pi v/N}\right]. \end{aligned}$$

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

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

The function is 0 at the center of the filter $u = M/2$). As u and v increase, the value of the filter decreases, reaching its limiting value of close to $-4j$ when $u = M-1$ and $v = M-1$. The negative limiting value is due to the order in which the derivatives are taken. If, instead we had taken differences of the form $f(x,y) - f(x+1,y)$ and $f(x,y) - f(x,y+1)$, the filter would have tended toward a positive limiting value. The important point here is that the dc term is eliminated and higher frequencies are passed, which is the characteristic of a highpass filter.

Problem 4.29

The filtered function is given by

$$g(x,y) = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y).$$

As in Problem 4.28,

$$G(u,v) = H(u,v)F(u,v)$$

where

$$\begin{aligned} H(u,v) &= \left[e^{j2\pi u/M} + e^{-j2\pi u/M} + e^{j2\pi v/N} + e^{-j2\pi v/N} - 4\right] \\ &= 2\left[\cos(2\pi u/M) + \cos(2\pi v/N) - 2\right]. \end{aligned}$$

Shifting the filter to the center of the frequency rectangle gives

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

When $(u,v) = (M/2, N/2)$ (the center of the shifted filter), $H(u,v) = 0$. For values away from the center, $H(u,v)$ decreases (as in Problem 4.28) because of the order in which derivatives are taken. The important point is the the dc term is eliminated and the higher frequencies are passed, which is the characteristic of a highpass filter.

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

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

in the preceding integral yields

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

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

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

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

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

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

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

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

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

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

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

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 4.32

The spatial filter is obtained by taking the inverse Fourier transform of the frequency-domain filter:

$$\begin{aligned} h_{\mathrm{HP}}(t,z) &= \Im^{-1}\left[1 - H_{\mathrm{LP}}(\mu,\nu)\right] \\ &= \Im^{-1}[1] - \Im^{-1}\left[H_{\mathrm{LP}}(\mu,\nu)\right] \\ &= \delta(0) - A2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)}. \end{aligned}$$

This result is for continuous functions. To use them with discrete variables we simply sample the function into its desired dimensions.

Problem 4.33

The complex conjugate simply changes j to $-j$ in the inverse transform, so the image on the right is given by

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

which simply mirrors $f(x,y)$ about the origin, thus producing the image on the right.

Problem 4.34

The equally-spaced, vertical bars on the left, lower third of the image.

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 4.36

(a) The ring in fact has a dark center area as a result of the highpass operation only (the following image shows the result of highpass filtering only). However, the dark center area is averaged out by the lowpass filter. The reason the final result looks so bright is that the discontinuity (edge) on boundaries of the ring are much higher than anywhere else in the image, thus dominating the display of the result.

(b) Filtering with the Fourier transform is a linear process. The order does not matter.

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

Figure P4.36.

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.

(b) To guarantee the result in (a), K has to be chosen large enough so that the filter becomes a notch pass filter (at the origin) for all values of $D(u,v)$. Keeping in mind that increments of frequencies are in unit values, this means

$$H_K(u,v) = e^{-KD^2(u,v)/2D_0^2} = \begin{cases} 1 & \text{if } (u,v) = (0,0) \\ 0 & \text{Otherwise.} \end{cases}$$

Because u and v are integers, the conditions on the second line in this equation are satisfied for all $u > 1$ and/or $v > 1$. When $u = v = 0$, $D(u,v) = 0$, and $H_K(u,v) = 1$, as desired. We want all values of the filter to be zero for all values of the distance from the origin that are greater than 0 (i.e., for values of u and/or v greater than 0). However, the filter is a Gaussian function, so its value is always greater than 0 for all finite values of $D(u,v)$. But, we are dealing with digital numbers, which will be designated as zero whenever the value of the filter is less than one-half the smallest positive number representable in the computer being used. As given in the problem statement, the value of this number is $c_{\min}$. So, values of K for which for which the filter function is greater than $0.5 \times c_{\min}$will

suffice. That is, we want the minimum value of K for which

$$e^{-KD^2(u,v)/2D_0^2} < 0.5c_{\min}$$

or

$$\begin{aligned} K &> -\frac{\ln(0.5c_{\min})}{D^2(u,v)/2D_0^2} \\ &> -\frac{2D_0^2\ln(0.5c_{\min})}{D^2(u,v)}. \end{aligned}$$

As noted above, we want this equation for hold for all values of $D^2(u,v) > 0$. Because the exponential decreases as a function of increasing distance from the origin, we choose the smallest possible value of $D^2(u,v)$, which is 1. This gives the result

$$K > -2D_0^2\ln(0.5c_{\min})$$

which yields a positive number because $c_{\min} << 1$. This result guarantees that the lowpass filter will act as a notch pass filter, leaving only the value of the transform at the origin. The image will not change past this value of K.

Problem 4.38

(a) The key for the student to be able to solve this problem is to treat the number of applications (denoted by K) of the highpass filter as 1 minus K applications of the corresponding lowpass filter, so that

$$\begin{aligned} H_K(u,v) &= H_K(u,v)F(u,v) \\ &= \left[1 - e^{-KD^2(u,v)/2D_0^2}\right] H(u,v) \end{aligned}$$

where the Gaussian lowpass filter is from Problem 4.37. Students who start directly with the expression of the Gaussian highpass filter $\left[1 - e^{-KD^2(u,v)/2D_0^2}\right]$ and attempt to raise it to the Kth power will run into a dead end.

The solution to the problem parallels the solution of Problem 4.37. Here, however, the filter will approach a notch filter that will take out $F(0,0)$ and thus will produce an image with zero average value (this implies negative pixels). So, there is a value of K after which the result of repeated highpass filtering will simply produce a constant image.

(b) The problem here is to determine the value of K for which

$$H_K(u,v) = 1 - e^{-KD^2(u,v)/2D_0^2} = \begin{cases} 0 & \text{if } (u,v) = (0,0) \\ 1 & \text{otherwise.} \end{cases}$$

Because u and v are integers, the conditions on the second line in this equation have to be satisfied for all $u \geq 1$ and/or $v \geq 1$. When $u = v = 0$, $D(u,v) = 0$, and $H_K(u,v) = 0$, as desired. We want all values of the filter to be 1 for all values of the distance from the origin that are greater than 0 (i.e., for values of u and/or v greater than 0). For $H_K(u,v)$ to become 1, the exponential term has to become 0 for values of u and/or v greater than 0. This is the same requirement as in Problem 4.37, so the solution of that problem applies here as well.

Problem 4.39

(a) Express filtering as convolution to reduce all processes to the spatial domain. Then, the filtered image is given by

$$g(x,y) = h(x,y) \star f(x,y)$$

where h is the spatial filter (inverse Fourier transform of the frequency-domain filter) and f is the input image. Histogram processing this result yields

$$\begin{aligned} g'(x,y) &= T\left[g(x,y)\right] \\ &= T\left[h(x,y) \star f(x,y)\right], \end{aligned}$$

where T denotes the histogram equalization transformation. If we histogram-equalize first, then

$$g(x,y) = T\left[f(x,y)\right]$$

and

$$g'(x,y) = h(x,y) \star T\left[f(x,y)\right].$$

In general, T is a nonlinear function determined by the nature of the pixels in the image from which it is computed. Thus, in general, $T\left[h(x,y) \star f(x,y)\right] \neq h(x,y) \star T\left[f(x,y)\right]$ and the order does matter.

(b) As indicated in Section 4.9, highpass filtering severely diminishes the contrast of an image. Although high-frequency emphasis helps some, the improvement is usually not dramatic (see Fig. 4.59). Thus, if an image is histogram equalized first, the gain in contrast improvement will essentially be lost in the filtering process. Therefore, the procedure in general is to filter first and histogram-equalize the image after that.

Problem 4.40

From Eq. (4.9-3), the transfer function of a Butterworth highpass filter is

$$H(u,v) = \frac{1}{1 + \left[\frac{D_0}{D(u,v)}\right]^{2n}}.$$

We want the filter to have a value of γ_L when $D(u,v)=0$, and approach γ_H for high values of $D(u,v)$. The preceding equation is easily modified to accomplish this:

$$H(u,v)=\gamma_L+\frac{(\gamma_H-\gamma_L)}{1+\left[\frac{D_0}{D(u,v)}\right]^{2n}}.$$

The value of n controls the sharpness of the transition between γ_L and γ_H.

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.

Problem 4.42

Consider a single star, modeled as an impulse $\delta(x-x_0, y-y_0)$. Then,

$$f(x,y) = K\delta(x-x_0, y-y_0)$$

from which

$$\begin{aligned} z(x,y) &= \ln f(x,y) = \ln K + \ln \delta(x-x_0, y-y_0) \\ &= K' + \delta'(x-x_0, y-y_0). \end{aligned}$$

Taking the Fourier transform of both sides yields

$$\begin{aligned} \Im\left[z(x,y)\right] &= \Im\left[K'\right] + \Im\left[\delta'(x-x_0, y-y_0)\right] \\ &= \delta(0,0) + e^{-2\pi(ux_0+vy_0)}. \end{aligned}$$

From this result, it is evident that the contribution of illumination is an impulse at the origin of the frequency plane. A notch filter that attenuates only this component will take care of the problem. Extension of this development to multiple impulses (stars) is implemented by considering one star at a time. The form of the filter will be the same. At the end of the procedure, all individual images are combined by addition, followed by intensity scaling so that the relative brightness between the stars is preserved.

Problem 4.43

The problem can be solved by carrying out the following steps:

1. Perform a median filtering operation.
2. Follow (1) by high-frequency emphasis.
3. Histogram-equalize this result.
4. Compute the average gray level, K_0. Add the quantity $(K-K_0)$ to all pixels.

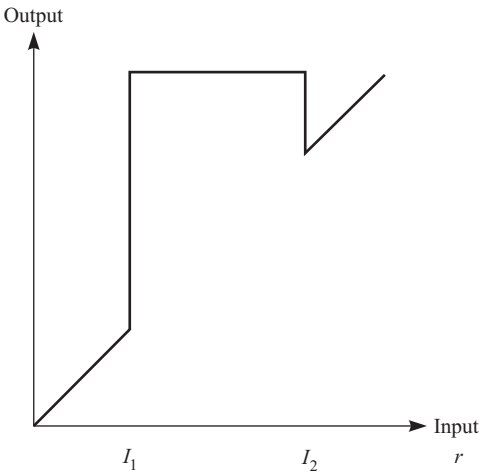


Figure P4.43

5. Perform the transformations shown in Fig. P4.43, where r is the input gray level, and R, G, and B are fed into an RGB color monitor.

Chapter 5

Problem Solutions

Problem 5.1

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

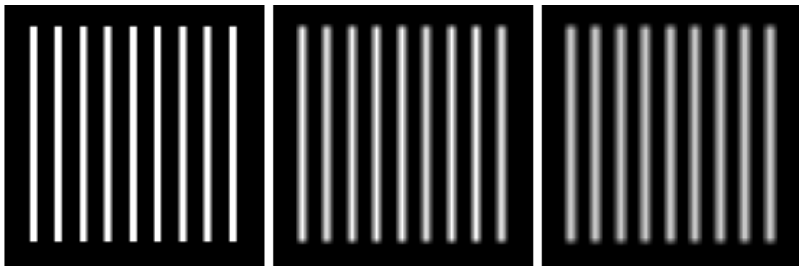


Figure P5.1

Problem 5.2

The solutions are shown in the following figure, from left to right.

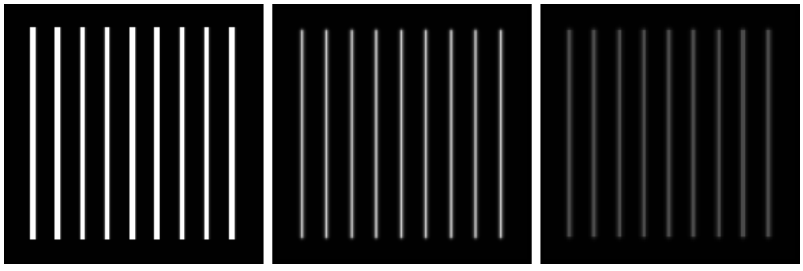


Figure P5.2

Problem 5.3

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

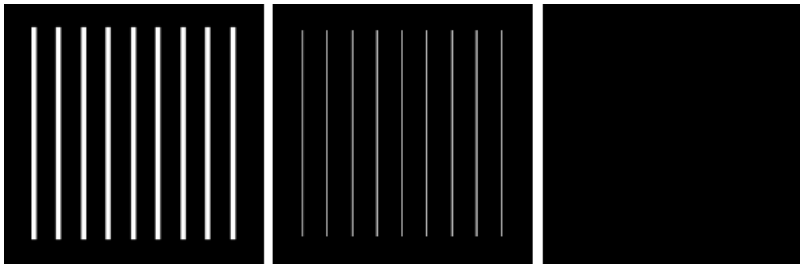


Figure P5.3

Problem 5.4

The solutions are shown in Fig. P5.4, from left to right.

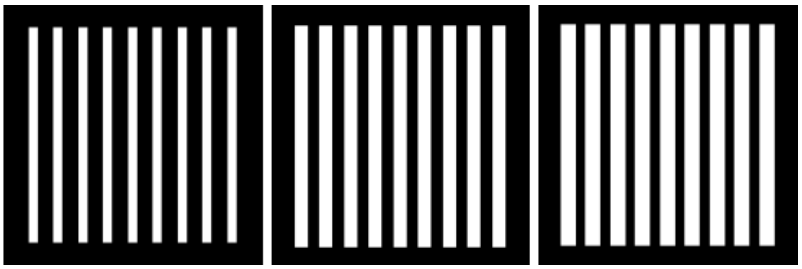


Figure P5.4

Problem 5.5

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

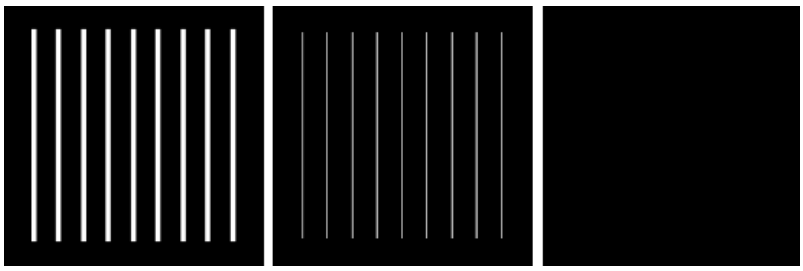


Figure P5.5

Problem 5.6

The solutions are shown in Fig. P5.6, from left to right.

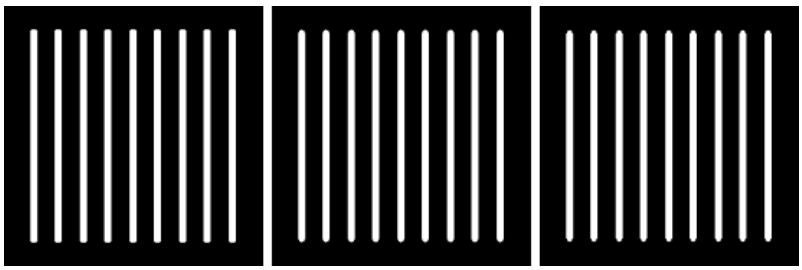


Figure P5.6

Problem 5.7

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

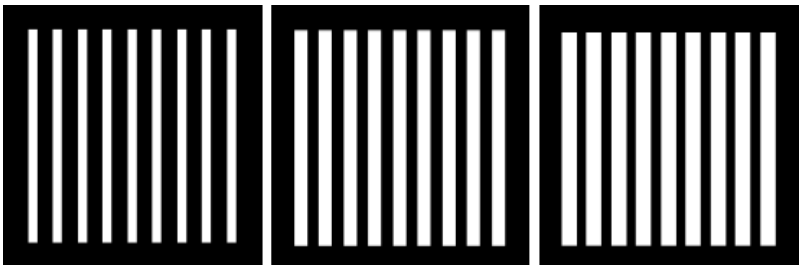


Figure P5.7

Problem 5.8

The solutions are shown in Fig. P5.8, from left to right.

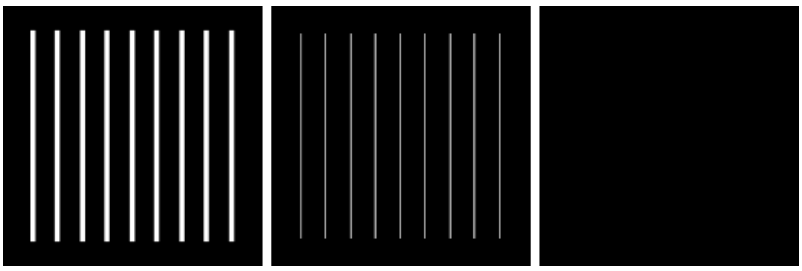


Figure P5.8

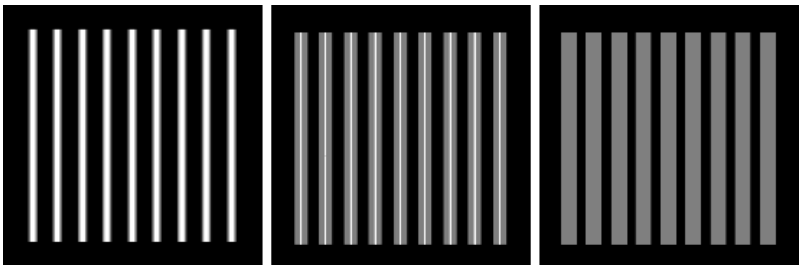


Figure P5.9

Problem 5.9

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

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).

(b) Black is 0, so the geometric mean will return values of 0 as long as at least one pixel in the window is black. Because the center of the mask can be outside the original black area when this happens, the figure will be thickened.

Problem 5.11

The key to understanding the behavior of the contra-harmonic filter is to think of the pixels in the neighborhood surrounding a noise impulse as being constant, with the impulse noise point being in the center of the neighborhood. For the noise spike to be visible, its value must be considerably larger than the value of its neighbors. Also keep in mind that the power in the numerator is 1 plus the power in the denominator.

(a) By definition, pepper noise is a low value (really 0). It is most visible when surrounded by light values. The center pixel (the pepper noise), will have little influence in the sums. If the area spanned by the filter is approximately constant, the ratio will approach the value of the pixels in the neighborhood—thus reducing the effect of the low-value pixel. For example, here are some values of the filter for a dark point of value 1 in a 3×3 region with pixels of value 100: For $Q = 0.5$, filter $= 98.78$; for $Q = 1$, filter $= 99.88$, for $Q = 2$, filter $= 99.99$; and for $Q = 5$, filter $= 100.00$.

(b) The reverse happens when the center point is large and its neighbors are small. The center pixel will now be the largest. However, the exponent is now negative, so the small numbers will dominate the result. The numerator can then be thought of a constant raised to the power $Q+1$ and the denominator as a the same constant raised to the power Q. That constant is the value of the pixels in the neighborhood. So the ratio is just that value.

(c) When the wrong polarity is used, the large numbers in the case of the salt noise will be raised to a positive power, thus the noise will overpower the result. For salt noise the image will become very light. The opposite is true for pepper noise—the image will become dark.

(d) When $Q=-1$, the value of the numerator at any location is equal to the number of pixels in the neighborhood (mn). The terms of the sum in the denominator are 1 divided by individual pixel values in the neighborhood. For example, for a 3×3 enighborhoood, the response of the filter when $Q=-1$ is: $9/[1/p1+1/p2+\cdots 1/p9]$ where the p's are the pixel values in the neighborhood. Thus, low pixel values will tend to produce low filter responses, and vice versa. If, for example, the filter is centered on a large spike surrounded by zeros, the response will be a low output, thus reducing the effect of the spike.

(e) In a constant area, the filter returns the value of the pixels in the area, independently of the value of Q.

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 5.13

We use highpass filters to construct notch filters, as indicated in Eq. (4.10.2). The ideal notch reject fitter is given by

$$H_{\mathrm{INR}}(u,v)=\prod_{k=1}^{3}H_k(u,v)H_{-k}(u,v)$$

where

$$H_k(u,v)=\begin{cases} 0 & \text{if } D_k(u,v)\le D_0 \\ 1 & \text{if } D_k(u,v)>D_0 \end{cases}$$

with

$$D_k(u,v)=\left[(u-M/2-u_k)^2+(v-N/2-v_k)^2\right]$$

in which (u_k,v_k) are the centers of the notches. For the Gaussian filter,

$$H_k(u,v)=1-e^{-D_k^2(u,v)/2D_0^2}$$

and the notch reject filter is given by

$$H_{\mathrm{GNR}}(u,v)=\prod_{k=1}^{3}\left[1-e^{-D_k^2(u,v)/2D_0^2}\right]\left[1-e^{-D_{-k}^2(u,v)/2D_0^2}\right].$$

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 5.15

From Eq. (5.4-19)

$$\sigma^2=\frac{1}{(2a+1)(2b+1)}\sum\sum\left\{\left[g(\gamma)-w\eta(\gamma)\right]-\left[\overline{g}-w\overline{\eta}\right]\right\}^2$$

where "γ" indicates terms affected by the summations. Letting $K = 1/(2a+1)(2b+1)$, taking the partial derivative of σ^2 with respect to w and setting the result equal to zero gives

$$\begin{aligned}
\frac{\partial \sigma^2}{\partial w} &= K\sum\sum 2\left[g(\gamma) - w\eta(\gamma) - \overline{g} + w\overline{\eta}\right]\left[-\eta(\gamma) + \overline{\eta}\right] = 0 \\
&= K\sum\sum -g(\gamma)\eta(\gamma) + g(\gamma)\overline{\eta} + w\eta^2(\gamma) - w\eta(\gamma)\overline{\eta} + \\
&\qquad \overline{g}\eta(\gamma) - \overline{g}\,\overline{\eta} - w\overline{\eta}\eta(\gamma) + w\overline{\eta}^2 \\
&= -\overline{g\eta} + \overline{g}\,\overline{\eta} + w\overline{\eta^2} - w\overline{\eta}^2 + \overline{g}\,\overline{\eta} - \overline{g}\,\overline{\eta} - w\overline{\eta}^2 + w\overline{\eta}^2 \\
&= -\overline{g\eta} + \overline{g}\,\overline{\eta} + w\left(\overline{\eta^2} - \overline{\eta}^2\right) \\
&= 0
\end{aligned}$$

where, for example, we used the fact that

$$\frac{1}{(2a+1)(2b+1)}\sum\sum g(\gamma)\eta(\gamma) = \overline{g\eta}.$$

Solving for w gives us

$$w = \frac{\overline{g\eta} - \overline{g}\,\overline{\eta}}{\overline{\eta^2} - \overline{\eta}^2}.$$

Finally, inserting the variables x and y,

$$w(x,y) = \frac{\overline{g(x,y)\eta(x,y)} - \overline{g}(x,y)\overline{\eta}(x,y)}{\overline{\eta^2}(x,y) - \overline{\eta}^2(x,y)}$$

which agrees with Eq. (5.4-21).

Problem 5.16

From Eq. (5.5-13),

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

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,

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

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

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

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

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

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

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

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

which is a blurred version of the original image.

Problem 5.17

Following the image coordinate convention in the book, vertical motion is in the x-direction and horizontal motion is in the y-direction. Then, the components of motion are as follows:

$$
x_0(t) = \begin{cases} \frac{at}{T_1} & 0 \le t \le T_1 \\ a & T_1 < t \le T_1 + T_2 \end{cases}
$$

and

$$
y_0(t) = \begin{cases} 0 & 0 \le t \le T_1 \\ \frac{b(t-T_1)}{T_1} & T_1 < t \le T_1 + T_2. \end{cases}
$$

Then, substituting these components of motion into Eq. (5.6-8) yields

$$\begin{aligned} H(u,v) &= \int_0^{T_1} e^{-j2\pi[uat/T_1]}dt + \int_{T_1}^{T_1+T_2} e^{-j2\pi[ua+vb(t-T_1)/T_2)}dt \\ &= \frac{T_1}{\pi ua}\sin(\pi ua)e^{-j\pi ua} + e^{-j2\pi ua}\int_{T_1}^{T_1+T_2} e^{-j2\pi vb(t-T_1)/T_2}dt \\ &= \frac{T_1}{\pi ua}\sin(\pi ua)e^{-j\pi ua} + e^{-j2\pi ua}\int_0^{T_2} e^{-j2\pi vb\tau/T_2}d\tau \\ &= \frac{T_1}{\pi ua}\sin(\pi ua)e^{-j\pi ua} + e^{-j2\pi ua}\frac{T_2}{\pi vb}\sin(\pi vb)e^{-j\pi vb} \end{aligned}$$

where in the third line we made the change of variables $\tau = t - T_1$. The blurred image is then

$$g(x,y) = \Im^{-1}[H(u,v)F(u,v)]$$

where $F(u,v)$ is the Fourier transform of the input image.

Problem 5.18

Following the procedure in Section 5.6.3,

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

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 5.19

A basic approach for restoring a rotationally blurred image is to convert the image from rectangular to polar coordinates. The blur will then appear as one-dimensional uniform motion blur along the θ-axis. Any of the techniques discussed in this chapter for handling uniform blur along one dimension can then be applied to the problem. The image is then converted back to rectangular coordinates after restoration. The mathematical solution is simple. For any pixel with rectangular coordinates (x, y) we generate a corresponding pixel with polar coordinates (r, θ), where

$$r = \sqrt{x^2 + y^2}$$

and

$$\theta = \tan^{-1}\left(\frac{y}{x}\right).$$

A display of the resulting image would show an image that is blurred along the θ-axis and would, in addition, appear distorted due to the coordinate conversion. Because the extent of the rotational blur is known (it is given as $\pi/8$ radians), we can use the same solution we used for uniform linear motion (Section 5.6.3), with $x = \theta$ and $y = r$ to obtain the transfer function. Any of the methods in Sections 5.7 through 5.9 then become applicable.

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 5.21

The key to solving this problem is to recognize that the given function,

$$h(x,y) = \frac{x^2+y^2-2\sigma^2}{\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

is the the second derivative (Laplacian) of the function (see Section 3.6.2 regarding the Laplacian)

$$s(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}.$$

That is,

$$\begin{aligned} \nabla^2[s(x,y)] &= \left[\frac{\partial^2 s(x,y)}{\partial x^2} + \frac{\partial^2 s(x,y)}{\partial y^2}\right] \\ &= \frac{x^2+y^2-2\sigma^2}{\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}. \end{aligned}$$

(This result is derived in Section 10.2.6). So, it follows that

$$\begin{aligned} H(u,v) &= \Im\left[h(x,y)\right] \\ &= \Im\left[\nabla^2 s(x,y)\right]. \end{aligned}$$

But, we know from the statement of Problem 4.26(a) that

$$\Im\left[\nabla^2 s(x,y)\right] = -4\pi^2(u^2+v^2)F(u,v)$$

where

$$\begin{aligned} F(u,v) &= \Im\left[s(x,y)\right] \\ &= \Im\left[e^{-\frac{x^2+y^2}{2\sigma^2}}\right]. \end{aligned}$$

Therefore, we have reduced the problem to computing the Fourier transform of a Gaussian function. From the basic form of the Gaussian Fourier transform pair given in entry 13 of Table 4.3 (note that (x,y) and (u,v) in the present problem are the reverse of the entry in the table), we have

$$\Im\left[e^{-\frac{x^2+y^2}{2\sigma^2}}\right] = 2\pi\sigma^2 e^{-2\pi^2\sigma^2(u^2+v^2)}$$

so we have the final result

$$\begin{aligned} H(u,v) &= -4\pi^2(u^2+v^2)F(u,v) \\ &= \left[-4\pi^2(u^2+v^2)\right]\left[2\pi\sigma^2 e^{-2\pi^2\sigma^2(u^2+v^2)}\right] \\ &= -8\pi^3\sigma^2(u^2+v^2)e^{-2\pi^2\sigma^2(u^2+v^2)} \end{aligned}$$

as desired. Keep in mind that the preceding derivations are based on assuming continuous variables. A discrete filter is obtained by sampling the continuous function.

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 5.23

This also is a simple plug in problem, whose purpose is the same as the previous problem. From Eq. (5.9-4)

$$\begin{aligned} H_C(u,v) &= \frac{H^*(u,v)}{|H(u,v)|^2+\gamma|P(u,v)|^2} \\ &= -\frac{8\pi^2\sigma^2(u^2+v^2)e^{-2\pi\sigma^2(u^2+v^2)}}{64\pi^4\sigma^4(u^2+v^2)^2 e^{-4\pi^2\sigma^2(u^2+v^2)}+\gamma|P(u,v)|^2} \end{aligned}$$

where $P(u,v)$ is the Fourier transform of the Laplacian operator [Eq. (5.9-5)]. This is as far as we can reasonably carry this problem. It is worthwhile pointing out to students that a filter in the frequency domain for the Laplacian operator is discussed in Section 4.9.4 [see Eq. (4.9-5)]. However, substituting that solution for $P(u,v)$ here would only increase the number of terms in the filter and would not help in simplifying the expression.

Problem 5.24

Because the system is assumed linear and position invariant, it follows that Eq. (5.5-17) holds. Furthermore, we can use superposition and obtain the response of the system first to $F(u,v)$ and then to $N(u,v)$ because we know that the image and noise are uncorrelated. The sum of the two individual responses then gives the complete response. First, using only $F(u,v)$,

$$G_1(u,v) = H(u,v)F(u,v)$$

and

$$|G_1(u,v)|^2 = |H(u,v)|^2 |F(u,v)|^2 .$$

Then, using only $N(u,v)$,

$$G_2(u,v) = N(u,v)$$

and,

$$|G_2(u,v)|^2 = |N(u,v)|^2$$

so that,

$$\begin{aligned} |G(u,v)|^2 &= |G_1(u,v)|^2 + |G_2(u,v)|^2 \\ &= |H(u,v)|^2 |F(u,v)|^2 + |N(u,v)|^2 . \end{aligned}$$

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

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

(b)

$$\begin{aligned} \hat{F}(u,v) &= R(u,v)G(u,v) \\ &= \left[\frac{|F(u,v)|^2}{|H(u,v)|^2 |F(u,v)|^2 + |N(u,v)|^2}\right]^{1/2} G(u,v) \\ &= \left[\frac{1}{|H(u,v)|^2 + \frac{|N(u,v)|^2}{|F(u,v)|^2}}\right]^{1/2} G(u,v) \end{aligned}$$

and, because $|F(u,v)|^2 = S_f(u,v)$ and $|N(u,v)|^2 = S_\eta(u,v)$,

$$\hat{F}(u,v) = \left[\frac{1}{|H(u,v)|^2 + \frac{S_\eta(u,v)}{S_f(u,v)}} \right]^{1/2} G(u,v).$$

Problem 5.26

One possible solution: (1) Perform image averaging to reduce noise. (2) Obtain a blurred image of a bright, single star to simulate an impulse (the star should be as small as possible in the field of view of the telescope to simulate an impulse as closely as possible. (3) The Fourier transform of this image will give $H(u,v)$. (4) Use a Wiener filter and vary K until the sharpest image possible is obtained.

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

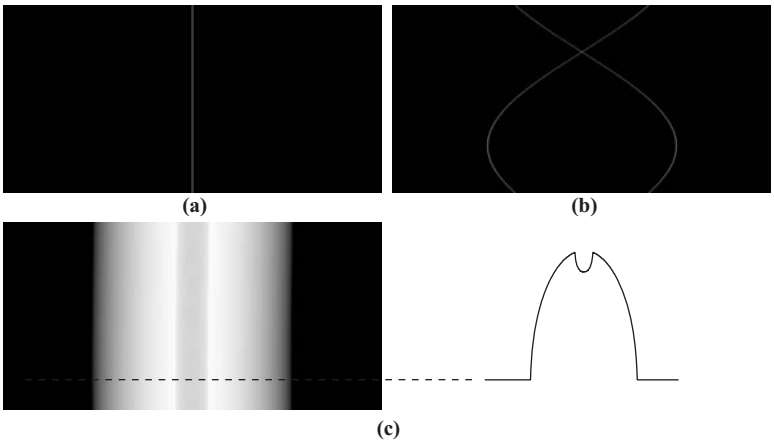


Figure P5.28

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

The solutions are shown in the following figures. 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°. In (b) 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.

Problem 5.29

Because $f(x,y)$ is rotationally symmetric, its projections are the same for all angles, so all we have to do is obtain the projection for $\theta = 0°$. Equation (5.11-3)

then becomes

$$\begin{aligned}\Re\{f(x,y)\} = g(\rho,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x-\rho)dxdy \\ &= \int_{-\infty}^{\infty} f(\rho,y)dy \\ &= A\int_{-\infty}^{\infty} e^{(-\rho^2-y^2)}dy \\ &= Ae^{-\rho^2}\int_{-\infty}^{\infty} e^{-y^2}dy.\end{aligned}$$

Because

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

it follows by letting $\sigma^2 = 1/2$ in this equation that,

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

Then,

$$\int_{-\infty}^{\infty} e^{-y^2}dy = \sqrt{\pi}$$

and

$$g(\rho,\theta) = A\sqrt{\pi}e^{-\rho^2}.$$

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.

(b) Similarly, substituting into Eq. (5.11-3),

$$\begin{aligned}\Re\{f(x,y)\} = g(\rho,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos\theta + y\sin\theta - \rho)dxdy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(x-x_0,y-y_0)\delta(x\cos\theta + y\sin\theta - \rho)dxdy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} 1\times\delta(x_0\cos\theta + y_0\sin\theta - \rho)dxdy.\end{aligned}$$

From the definition of the impulse, this result is 0 unless

$$\rho = x_0\cos\theta + y_0\sin\theta$$

which is the equation of a sinusoidal curve in the $\rho\theta$-plane.

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.

(b) Let $p = x - x_0$ and $q = y - y_0$. Then $dp = dx$ and $dq = dy$. From Eq. (5.11-3), the Radon transform of $f(x-x_0, y-y_0)$ is

$$\begin{aligned}g(\rho,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x-x_0,y-y_0)\delta(x\cos\theta + y\sin\theta - \rho)dxdy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(p,q)\delta\left[(p+x_0)\cos\theta + (q+y_0)\sin\theta - \rho)\right]dpdq \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(p,q)\delta\left[p\cos\theta + q\sin\theta - (\rho - x_0\cos\theta - y_0\sin\theta)\right]dpdq \\ &= g(\rho - x_0\cos\theta - y_0\sin\theta, \theta).\end{aligned}$$

(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,

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

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

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 5.32

The solution is as follows:

$$\begin{aligned}
f(x,y) &= \int_0^{2\pi}\int_0^\infty G(\omega,\theta)e^{j2\pi\omega(x\cos\theta+y\sin\theta)}\omega d\omega d\theta \\
&= \int_0^{\pi}\int_0^\infty G(\omega,\theta)e^{j2\pi\omega(x\cos\theta+y\sin\theta)}\omega d\omega d\theta \\
&\qquad + \int_0^{\pi}\int_0^\infty G(\omega,\theta)e^{j2\pi\omega(x\cos[\theta+180^\circ]+y\sin[\theta+180^\circ)}\omega d\omega d\theta.
\end{aligned}$$

But $G(\theta+180^\circ,\theta) = G(-\omega,\theta)$, so the preceding equation can be expressed as

$$f(x,y) = \int_0^{\pi}\int_{-\infty}^\infty |\omega|\,G(\omega,\theta)e^{j2\pi\omega(x\cos\theta+y\sin\theta)}d\omega d\theta$$

which agrees with Eq. (5.11-15).

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).

Problem 5.34

From the explanation of Eq. (5.11-18),

$$s(\rho)=\int_{-\infty}^{\infty}|\omega|\,e^{j2\pi\omega\rho}\,d\omega.$$

Let $\rho = R\sin\alpha$, and keep in mind that $\alpha/R\sin\alpha$ is always positive. Then,

$$s(R\sin\alpha)=\int_{-\infty}^{\infty}|\omega|\,e^{j2\pi\omega R\sin\alpha}\,d\omega.$$

Next, define the transformation

$$\omega'=\frac{\omega R\sin\alpha}{\alpha}.$$

Then,

$$d\omega=\frac{\alpha}{R\sin\alpha}d\omega'$$

and we can write

$$\begin{aligned} s(R\sin\alpha) &= \left[\frac{\alpha}{R\sin\alpha}\right]^2\int_{-\infty}^{\infty}\left|\omega'\right|e^{j2\pi\omega'\alpha}\,d\omega' \\ &= \left[\frac{\alpha}{R\sin\alpha}\right]^2 s(\alpha) \end{aligned}$$

as desired.

Chapter 6

Problem Solutions

Problem 6.1

From Fig. 6.5 in the book, $x = 0.43$ and $y = 0.4$. Since $x + y + z = 1$, it follows that $z = 0.17$. These are the trichromatic coefficients. We are interested in tristimulus values X, Y, and Z, which are related to the trichromatic coefficients by Eqs. (6.1-1) through (6.1-3). Note however, that all the tristimulus coefficients are divided by the same constant, so their percentages relative to the trichromatic coefficients are the same as those of the coefficients. Therefore, the answer is $X = 0.43$, $Y = 0.40$, and $Z = 0.17$.

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.

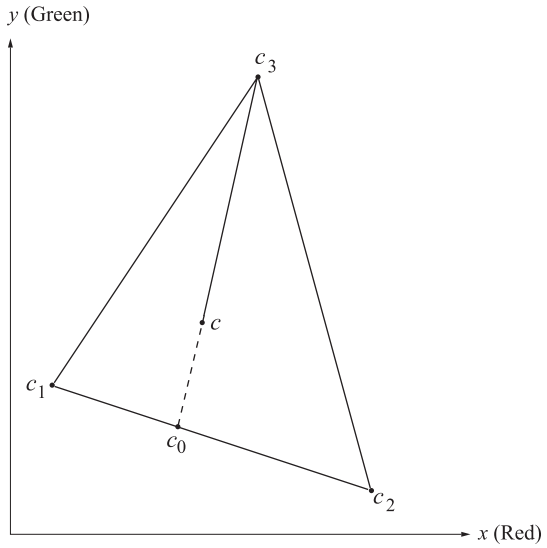


Figure P6.3

Problem 6.3

Consider Fig. P6.3, in which c_1, c_2, and c_3 are the given vertices of the color triangle and c is an arbitrary color point contained within the triangle or on its boundary. The key to solving this problem is to realize that any color on the border of the triangle is made up of proportions from the two vertices defining the line segment that contains the point. The contribution to a point on the line by the color vertex opposite this line is 0% .

The line segment connecting points c_3 and c is shown extended (dashed segment) until it intersects the line segment connecting c_1 and c_2. The point of intersection is denoted c_0. Because we have the values of c_1 and c_2, if we knew c_0, we could compute the percentages of c_1 and c_2 contained in c_0 by using the method described in Problem 6.2. Let the ratio of the content of c_1 and c_2 in c_0 be denoted by R_{12}. If we now add color c_3 to c_0, we know from Problem 6.2 that the point will start to move toward c_3 along the line shown. For any position of a point along this line we could determine the percentage of c_3 and c_0, again, by using the method described in Problem 6.2. What is important to keep in mind that the ratio R_{12} will remain the same for any point along the segment connecting c_3 and c_0. The color of the points along this line is different for each position, but the *ratio* of c_1 to c_2 will remain constant.

So, if we can obtain c_0, we can then determine the ratio R_{12}, and the percentage of c_3, in color c. The point c_0 is not difficult to obtain. Let $y = a_{12}x + b_{12}$ be

the straight line containing points c_1 and c_2, and $y = a_{3c}x + b_{3c}$ the line containing c_3 and c. The intersection of these two lines gives the coordinates of c_0. The lines can be determined uniquely because we know the coordinates of the two point pairs needed to determine the line coefficients. Solving for the intersection in terms of these coordinates is straightforward, but tedious. Our interest here is in the fundamental method, not the mechanics of manipulating simple equations so we do not give the details.

At this juncture we have the percentage of c_3 and the ratio between c_1 and c_2. Let the percentages of these three colors composing c be denoted by p_1, p_2, and p_3 respectively. We know that $p_1 + p_2 = 100 - p_3$, and that $p_1/p_2 = R_{12}$, so we can solve for p_1 and p_2. Finally, note that this problem could have been solved the same way by intersecting one of the other two sides of the triangle. Going to another side would be necessary, for example, if the line we used in the preceding discussion had an infinite slope. A simple test to determine if the color of c is equal to any of the vertices should be the first step in the procedure; in this case no additional calculations would be required.

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.

Problem 6.5

At the center point we have

$$\frac{1}{2}R + \frac{1}{2}B + G = \frac{1}{2}(R + G + B) + \frac{1}{2}G = \text{midgray} + \frac{1}{2}G$$

which looks to a viewer like pure green with a boost in intensity due to the additive gray component.

Table P6.6

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

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 6.7

There are $2^8 = 256$ possible values in each 8-bit image. For a color to be gray, all RGB components have to be equal, so there are 256 shades of gray.

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

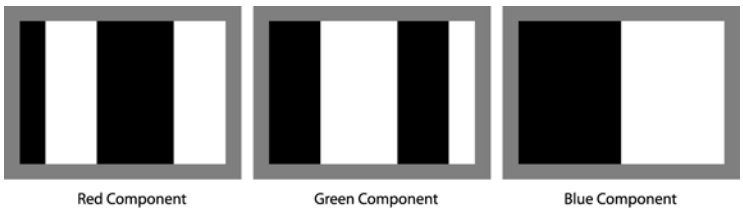


Figure P6.6

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.

(b) Let the axis numbering be the same as in Fig. 6.7 in the book. Then: $(0,0,0) =$ white, $(1,1,1,) =$ black, $(1,0,0) =$ cyan, $(1,1,0) =$ blue, $(1,0,1) =$ green, $(0,1,1) =$ red, $(0,0,1) =$ yellow, $(0,1,0) =$ magenta.

(c) The ones that do not contain the black or white point are fully saturated. The others decrease in saturation from the corners toward the black or white point.

Table P6.9

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

Problem 6.9

(a) For the image given, the maximum intensity and saturation requirement means that the RGB component values are 0 or 1. We can create Table P6.9 using Eq. (6.2-1). Thus, we get the monochrome displays shown in Fig. P6.9(a).

(b) The resulting display is the complement of the starting RGB image. From left to right, the color bars are (in accordance with Fig. 6.32) white, cyan, blue, magenta, red, yellow, green, and black. The middle gray background is unchanged.

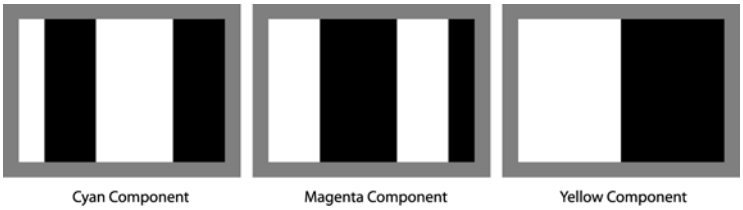


Figure P6.9

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 6.11

(a) The purest green is 00FF00, which corresponds to cell (7, 18).

(b) The purest blue is 0000FF, which corresponds to cell (12, 13).

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

Problem 6.13

With reference to the HSI color circle in Fig. 6.14(b), deep purple is found at approximately 270°. To generate a color rectangle with the properties required in the problem statement, we choose a fixed intensity I, and maximum saturation (these are spectrum colors, which are supposed to be fully saturated), S. The first column in the rectangle uses these two values and a hue of 270°. The next column (and all subsequent columns) would use the same values of I and S, but the hue would be decreased to 269°, and so on all the way down to a hue of 0°, which corresponds to red. If the image is limited to 8 bits, then we can only have 256 variations in hue in the range from 270° down to 0°, which will require a different uniform spacing than one degree increments or, alternatively, starting at a 255° and proceed in increments of 1, but this would leave out most of the purple. If we have more than eight bits, then the increments can be smaller. Longer strips also can be made by duplicating column values.

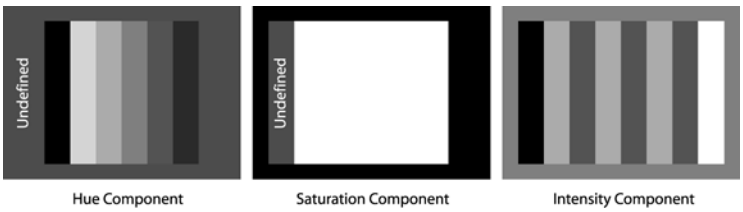


Figure P6.12

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 6.15

The hue, saturation, and intensity images are shown in Fig. P6.15, from left to right.

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

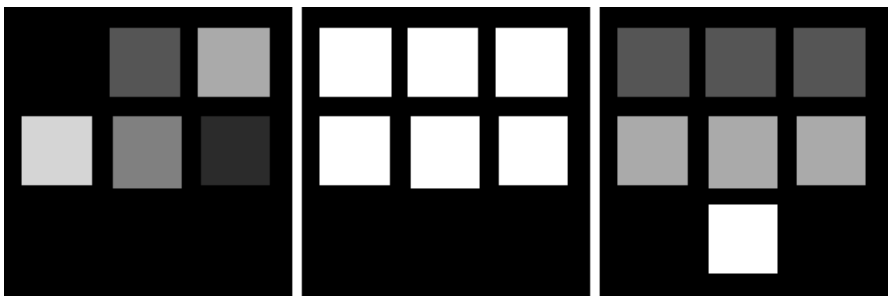


Figure P6.15

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.

(b) The colors are spectrum colors, so they are fully saturated. Therefore, the values 255 shown apply‘ to all circle regions. The region in the center of the color image is white, so its saturation is 0.

(c) The key to getting the values in this figure is to realize that the center portion of the color image is white, which means equal intensities of fully saturated red, green, and blue. Therefore, the value of both darker gray regions in the intensity image have value 85 (i.e., the same value as the other corresponding region). Similarly, equal proportions of the secondaries yellow, cyan, and magenta produce white, so the two lighter gray regions have the same value (170) as the region shown in the figure. The center of the image is white, so its value is 255.

Problem 6.17

(a) Because the infrared image which was used in place of the red component image has very high gray-level values.

(b) The water appears as solid black (0) in the near infrared image [Fig. 6.27(d)]. Threshold the image with a threshold value slightly larger than 0. The result is shown in Fig. P6.17. It is clear that coloring all the black points in the desired shade of blue presents no difficulties.

(c) Note that the predominant color of natural terrain is in various shades of red. We already know how to take out the water from (b). Therefore, a method that actually removes the “background” of red and black would leave predominantly the other man-made structures, which appear mostly in a bluish light color. Removal of the red [and the black if you do not want to use the method as in (b)] can be done by using the technique discussed in Section 6.7.2.

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 begin-

Figure P6.17

ning 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 6.19

The complement of a color is the color opposite it on the color circle of Fig. 6.32. The hue component is the angle from red in a counterclockwise direction normalized by 360 degrees. For a color on the top half of the circle (i.e., $0 \leq H \leq 0.5$), the hue of the complementary color is $H + 0.5$. For a color on the bottom half of the circle (i.e., for $0.5 \leq H \leq 1$), the hue of the complement is $H - 0.5$.

Problem 6.20

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

$$s_i = 1 - r_i$$

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

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

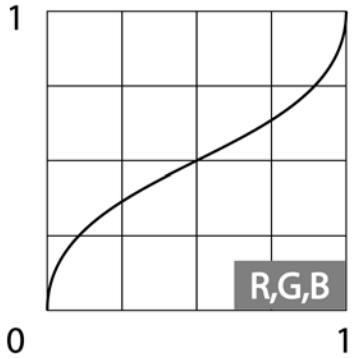


Figure P6.21

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 6.21

The RGB transformation should darken the highlights and lighten the shadow areas, effectively compressing all values toward the midtones. The red, green, and blue components should be transformed with the same mapping function so that the colors do not change. The general shape of the curve would be as shown in Fig. P6.21.

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 6.23

The $L^*a^*b^*$ components are computed using Eqs. (6.5-9) through (6.5-12). Reference white is $R = G = B = 1$. The computations are best done in a spreadsheet, as shown in Table P6.23.

Table P6.23

Color	R	G	B	X	Y	Z	$\frac{X}{X_w}$	$\frac{Y}{Y_w}$	$\frac{Z}{Z_w}$	$h\left(\frac{X}{X_w}\right)$	$h\left(\frac{Y}{Y_w}\right)$	$h\left(\frac{Z}{Z_w}\right)$	L^*	a^*	b^*
Ref.	1	1	1	0.95	1.00	1.10	1	1	1	1	1	1	100	0	0
Black	0	0	0	0	0	0	0	0	0	0.14	0.14	0.14	0	0	0
Red	1	0	0	0.59	0.29	0	0.62	0.29	0	0.85	0.66	0.14	83	95	105
Yellow	1	1	0	0.77	0.90	0.07	0.81	0.90	0.06	0.93	0.96	0.40	92	-16	113
Green	0	1	0	0.18	0.61	0.07	0.19	0.61	0.06	0.57	0.85	0.40	51	-136	90
Cyan	0	1	1	0.36	0.71	1.09	0.38	0.71	1	0.73	0.89	1	68	-84	-22
Blue	0	0	1	0.18	0.11	1.02	0.19	0.11	0.94	0.58	0.47	0.98	51	53	-101
Magenta	1	0	1	0.77	0.40	1.02	0.81	0.40	0.94	0.93	0.73	0.98	92	100	-49
White	1	1	1	0.95	1.00	1.10	1	1	1	1	1	1	100	0	0
Gray	0.5	0.5	0.5	0.48	0.50	0.55	0.5	0.5	0.5	0.79	0.79	0.79	76	0	0

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 6.25

The given color image is shown in Fig. P6.25(a). Assume that the component image values of the HSI image are in the range $[0, 1]$. Call the component images H (hue), S (saturation), and I (intensity).

(a) It is given that the image is fully saturated, so image H will be constant with value 1. Similarly, all the squares are at their maximum value so, from Eq. (6.2-4), the intensity image also will be constant, with value 1/3 [the maximum value of any (normalized) pixel in the RGB image is 1, and it is given that none of the squares overlap]. The hue component image, H, is shown in Fig. P6.25(b). Recall from Fig. 6.14 that the value of hue is an angle. Because the range of values of H is normalized to [0, 1], we see from that figure, for example, that as we go around the circle in the counterclockwise direction a hue value of 0 corresponds to red, a value of 1/3 to green, and a value of 2/3 to blue. Thus, the important point to be made in Fig. P6.25(b) is that the gray value in the image corresponding to the red square should be black, the value corresponding to the green green square should be lower-mid gray, and the value of the the square corresponding to blue should be a lighter shade of gray than the square corresponding to green. As you can see, this indeed is the case for the squares in Fig. P6.25(b). For the shades of red, green, and blue in Fig. P6.25(a), the exact values are $H = 0$, $H = 0.33$, and $H = 0.67$, respectively.

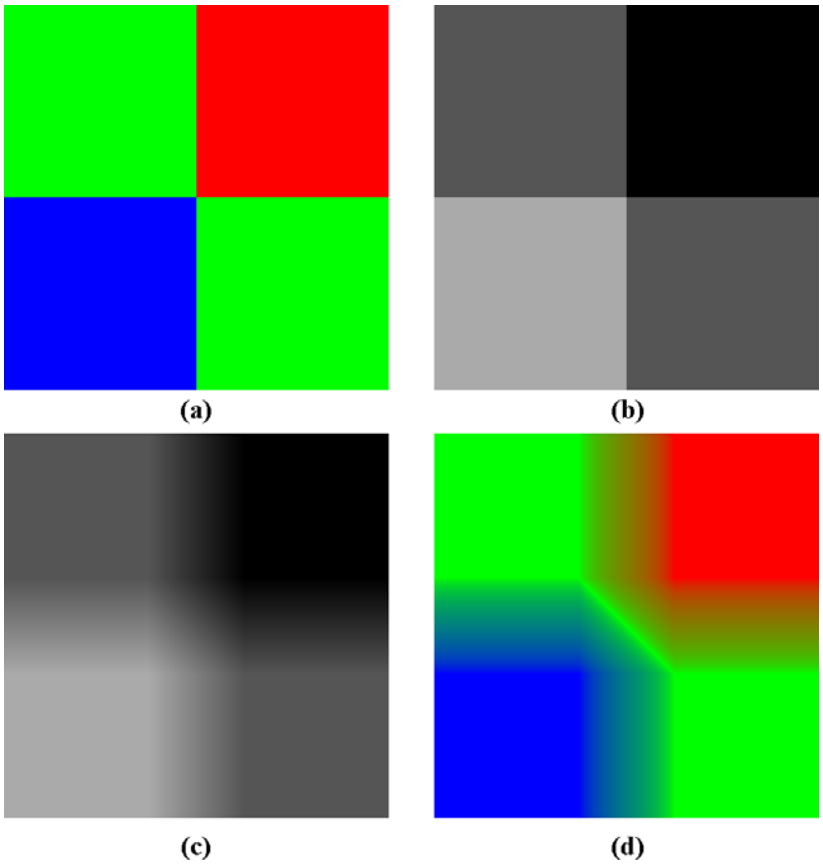


Figure P6.25

(b) The saturation image is constant, so smoothing it will produce the same constant value .

(c) Figure P6.25(c) shows the result of blurring the hue image. When the averaging mask is fully contained in a square, there is no blurring because the value of each square is constant. When the mask contains portions of two or more squares the value produced at the center of the mask will be between the values of the two squares, and will depend the relative proportions of the squares occupied by the mask. To see exactly what the values are, consider a point in the center of red mask in Fig. P6.25(c) and a point in the center of the green mask on the top left. We know from (a) above that the value of the red point is 0 and the value of the green point is 0.33. Thus, the values in the blurred band between red and green vary from 0 to 0.33 because averaging is a linear operation. Figure P6.25(d) shows the result of generating an RGB image with the blurred

hue component images and the original saturation and intensity images. The values along the line just discussed are transitions from green to red. From Fig. 6.14 we see that those transitions encompass the spectrum from green to red that includes colors such as yellow [all those colors are present in Fig. P6.25(d), although they are somewhat difficult to see]. The reason for the diagonal green line in this figure is that the average values along that region are nearly midway between red and blue, which we know from from Fig. 6.14 is green.

Problem 6.26

This is a simple problem to encourage the student to think about the meaning of the elements in Eq. (6.7-2). When $\mathbf{C} = \mathbf{I}$, it follows that $\mathbf{C}^{-1} = \mathbf{I}$ and Eq. (6.7-2) becomes

$$D(\mathbf{z}, \mathbf{a}) = \left[(\mathbf{z} - \mathbf{a})^T (\mathbf{z} - \mathbf{a})\right]^{1/2}.$$

But the term inside the brackets is recognized as the inner product of the vector $(\mathbf{z} - \mathbf{a})$ with itself, which, by definition, is equal to the right side of Eq. (6.7-1).

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) $\mathbf{z}$ in RGB space lying on the plane. If an RGB point $\mathbf{z}$ does not lie on the plane, and its coordinates are substituted in the preceding equation, the equation will give either a positive or a negative value; it will not yield zero. We say that $\mathbf{z}$ lies on the positive or negative side of the plane, depending on whether the result is positive or negative. We can change the positive side of a plane by multiplying its coefficients (except d) by -1. Suppose that we test the point $\mathbf{a}$ given in the problem statement to see whether it is on the positive or negative side each of the six planes composing the box, and change the coefficients of any plane for which the result is negative. Then, $\mathbf{a}$ will lie on the positive side of all planes composing the bounding box. In fact all points inside the bounding box will yield positive values when their coordinates are substituted in the equations of the planes. Points outside the box will give at least one negative (or zero 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

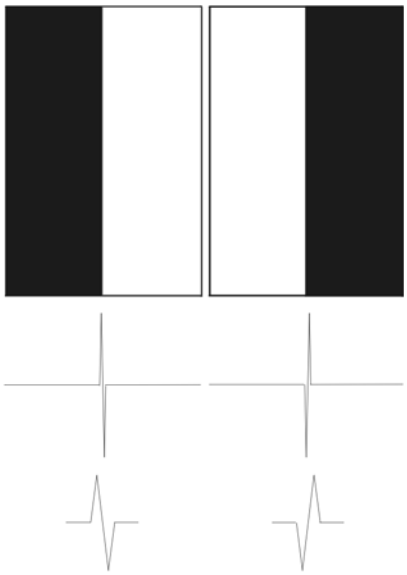


Figure P6.29

for in the problem statement to make it simpler to evaluate the student's line of reasoning.

(b) If the box is lined up with the RGB coordinate axes, then the planes intersect the RGB coordinate planes perpendicularly. The intersections of pairs of parallel planes establish a range of values along each of the RGB axis that must be checked to see if the if an unknown point lies inside the box or not. This can be done on an image per image basis (i.e., the three component images of an RGB image), designating by 1 a coordinate that is within its corresponding range and 0 otherwise. These will produce three binary images which, when ANDed, will give all the points inside the box.

Problem 6.28

The sketch is an elongated ellipsoidal figure in which the length lined up with the Red (R) axis is 8 times longer that the other two dimensions. In other words, the figure looks like a blimp aligned with the R-axis.

Problem 6.29

Set one of the three primary images to a constant value (say, 0), then consider the two images shown in Fig. P6.29. If we formed an RGB composite image by

letting the image on the left be the red component and the image on the right the green component, then the result would be an image with a green region on the left separated by a vertical edge from a red region on the right. To compute the gradient of each component image we take second-order partial derivatives. In this case, only the component of the derivative in the horizontal direction is nonzero. If we model the edge as a ramp edge then a profile of the derivative image would appear as shown in Fig. P6.29 (see Section 10.2.4 for more detail on the profile of edges). The magnified view shows clearly that the derivatives of the two images are mirrors of each other. Thus, if we computed the gradient vector of each image and added the results as suggested in the problem statement, the components of the gradient would cancel out, giving a zero gradient for a color image that has a clearly defined edge between two different color regions. This simple example illustrates that the gradient vector of a color image is not equivalent to the result of forming a color gradient vector from the sum of the gradient vectors of the individual component images.

Chapter 7

Problem Solutions

Problem 7.1

Following the explanation in Example 7.1, the decoder is as shown in Fig. P7.1.

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)

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

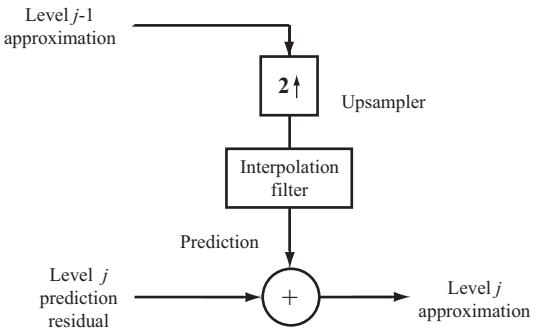


Figure P7.1

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 approximation (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.

Problem 7.4

First check for orthonormality. If the filters are orthonormal, they are biorthogonal by definition. Because $K_{even} = 2$, $n = 0, 1$, and $g_0(n) = \left\{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right\}$ for $n = 0, 1$, Eq. (7.1-14) yields:

$$(-1)^n g_0(1-n) = (-1)^n g_0(1-n) = \{g_0(0), -g_0(1)\} = \left\{\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right\} = g_1(n)$$
$$g_0(1-n) = \left\{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right\} = h_0(n)$$
$$g_1(1-n) = \left\{\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right\} = h_1(n).$$

Thus, the filters are orthonormal and will also satisfy Eq. (7.1-13). In addition, they will satisfy the biorthogonality conditions stated in Eqs. (7.1-12) and (7.1-11), but not (7.1-10). The filters are both orthonormal and biorthogonal.

Problem 7.5

(a) $f_a(n) = -f(n) = \{0, -0.5, -0.25, -1\}$.

(b) $f_b(n) = f(-n) = \{1, 0.25, 0.5, 0\}$.

(c) $f_c(n) = (-1)^n f(n) = \{0, -0.5, 0.25, -1\}$.

(d) $f_d(n) = f_c(-n) = \{-1, 0.25, -0.5, 0\}$.

(e) $f_e(n) = (-1)^n f_b(n) = \{1, -0.25, 0.5, 0\}$.

(f) Filter $f_e(n)$ in (e) corresponds to Eq. (7.1-9).

Problem 7.6

Table 7.1 in the book defines $g_0(n)$ for $n = 0, 1, 2, ..., 7$ to be about {0.23, 0.72, 0.63, -0.03, -0.19, 0.03, 0.03, -0.01}. Using Eq, (7.1-14) with $K_{even} = 8$, we can

write

$$g_1(n) = (-1)^n g_0(7-n).$$

Thus $g_1(n)$ is an order-reversed and modulated copy of $g_0(n)$—that is, {–0.01, -0.03, 0.03, 0.19, -0.03, -0.63, 0.72, -0.23}.

To numerically prove the orthonormality of the filters, let $m = 0$ in Eq. (7.1-13):

$$\left\langle g_i(n)\, g_j(n) \right\rangle = \delta\left(i-j\right) \quad \text{for } i, j = \{0, 1\}.$$

Iterating over i and j, we get

$$\begin{aligned} \sum_n g_0^2(n) &= \sum_n g_1^2(n) = 1 \\ \sum_n g_0(n)\, g_1(n) &= 0. \end{aligned}$$

Substitution of the filter coefficient values into these equations yields:

$$\begin{aligned} \sum_n g_0(n)\, g_1(n) &= (0.23)(-0.01) + (0.72)(-0.03) + (0.63)(0.03) \\ &\quad + (-0.03)(0.19) + (-0.19)(-0.03) + (0.03)(-0.63) \\ &\quad + (0.03)(0.72) + (-0.01)(-0.23) \\ &= 0 \end{aligned}$$

$$\begin{aligned} \sum_n g_0^2(n) &= \sum_n g_1^2(n) \\ &= (\pm 0.23)^2 + (0.72)^2 + (\pm 0.63)^2 + (-0.03)^2 \\ &\quad + (\pm 0.19)^2 + (0.03)^2 + (\pm 0.03)^2 + (-0.01)^2 \\ &= 1. \end{aligned}$$

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.

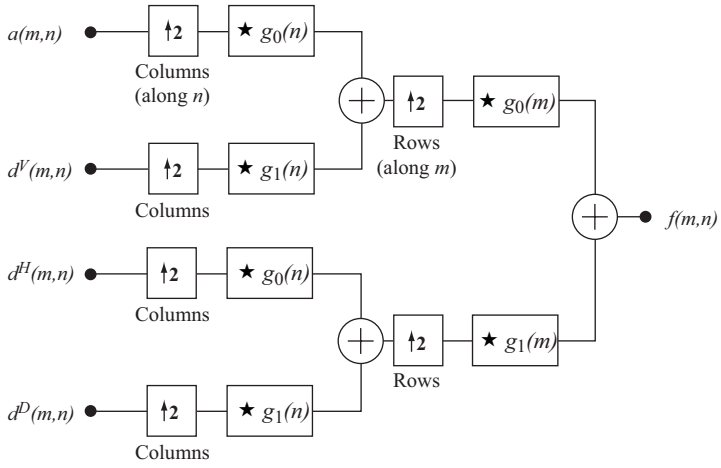


Figure P7.7

Problem 7.8

The Haar transform matrix for $N = 8$ is

$$\mathbf{H}_8 = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix}.$$

Problem 7.9

(a) Equation (7.1-18) defines the 2×2 Haar transformation matrix as

$$\mathbf{H}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Thus, using Eq. (7.1-17), we get

$$\mathbf{T} = \mathbf{HFH}^T = \left(\frac{1}{\sqrt{2}}\right)^2 \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 3 & -1 \\ 6 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

$$= \begin{bmatrix} 5 & 4 \\ -3 & 0 \end{bmatrix}$$

(b) First, compute

$$\mathbf{H}_2^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

such that

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Solving this matrix equation yields

$$\mathbf{H}_2^{-1} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \mathbf{H}_2 = \mathbf{H}_2^T.$$

Thus,

$$\begin{aligned} \mathbf{F} &= \mathbf{H}^T \mathbf{T} \mathbf{H} \\ &= \left(\frac{1}{\sqrt{2}}\right)^2 \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 5 & 4 \\ -3 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ &= \begin{bmatrix} 3 & -1 \\ 6 & 2 \end{bmatrix}. \end{aligned}$$

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,

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

(b) The basis is biorthonormal and the coefficients are computed by the vector equivalent of Eq. (7.2-3):

$$\begin{aligned}\alpha_0 &= \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \\ &= 1 \\ \alpha_1 &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \\ &= 2\end{aligned}$$

so,

$$\begin{aligned}\varphi_0 + 2\varphi_1 &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 2 \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 3 \\ 2 \end{bmatrix}.\end{aligned}$$

(c) The basis is overcomplete and the coefficients are computed by the vector equivalent of Eq. (7.2-3):

$$\begin{aligned}\alpha_0 &= \begin{bmatrix} \frac{2}{3} & 0 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \\ &= 2 \\ \alpha_1 &= \begin{bmatrix} -\frac{1}{3} & \frac{\sqrt{3}}{3} \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \\ &= -1 + \frac{2\sqrt{3}}{3} \\ \alpha_2 &= \begin{bmatrix} -\frac{1}{3} & -\frac{\sqrt{3}}{3} \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \\ &= -1 - \frac{2\sqrt{3}}{3}\end{aligned}$$

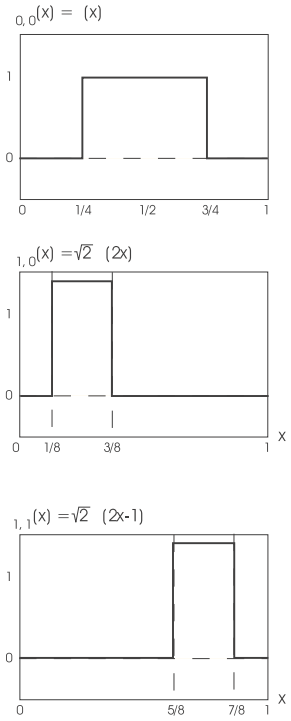


Figure P7.11

so,

$$2\varphi_0 + \left[-1+\frac{2\sqrt{3}}{3}\right]\varphi_1 + \left[-1-\frac{2\sqrt{3}}{3}\right]\varphi_2 = 2\begin{bmatrix}1\\0\end{bmatrix} + \left[-1+\frac{2\sqrt{3}}{3}\right]\begin{bmatrix}-\frac{1}{2}\\ \frac{\sqrt{3}}{2}\end{bmatrix} + \left[-1-\frac{2\sqrt{3}}{3}\right]\begin{bmatrix}-\frac{1}{2}\\ -\frac{\sqrt{3}}{2}\end{bmatrix} = \begin{bmatrix}3\\2\end{bmatrix}.$$

Problem 7.11

As can be seen in Fig. P7.11, scaling function $\varphi_{0,0}(x)$ cannot be written as a sum of double resolution copies of itself. Note the gap between $\varphi_{1,0}(x)$ and $\varphi_{1,1}(x)$.

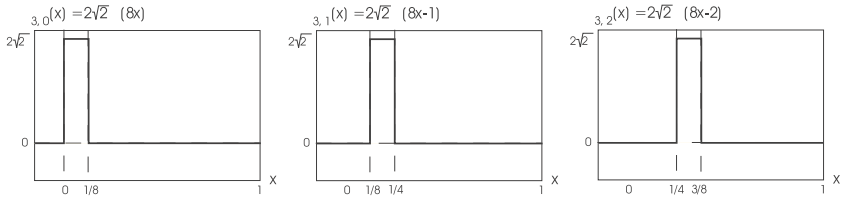


Figure P7.12

Problem 7.12

Substituting $j=3$ into Eq. (7.2-13), we get

$$V_3 = \overline{\underset{k}{Span}\{\varphi_{3,k}(x)\}}$$
$$V_3 = \overline{\underset{k}{Span}\left\{2^{\frac{3}{2}}\varphi\left(2^3x-k\right)\right\}}$$
$$V_3 = \overline{\underset{k}{Span}\left\{2\sqrt{2}\varphi(8x-k)\right\}}.$$

Using the Haar scaling function [Eq. (7.2-14)], we then get the result shown in Fig. P7.12.

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

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

Problem 7.14

Using Eq. (7.2-22),

$$\begin{aligned}V_3 &= V_2 \oplus W_2\\ &= V_1 \oplus W_1 \oplus W_2\\ &= V_0 \oplus W_0 \oplus W_1 \oplus W_2\end{aligned}$$

The scaling and wavelet functions are plotted in Fig. P7.14.

Problem 7.15

With $j_0 = 1$ the approximation coefficients are $c_1(0)$ and $c_1(1)$:

$$c_1(0) = \int_0^{1/2} x^2\sqrt{2}\,dx = \frac{\sqrt{2}}{24}$$

$$c_1(1) = \int_{1/2}^{1} x^2\sqrt{2}\,dx = \frac{7\sqrt{2}}{24}.$$

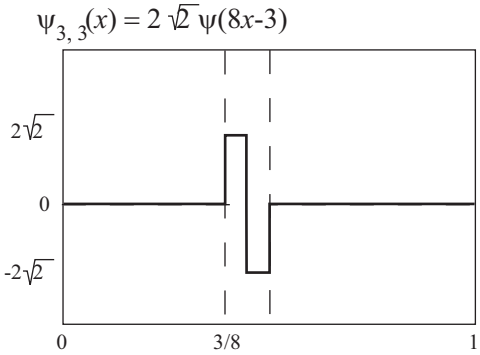


Figure P7.13

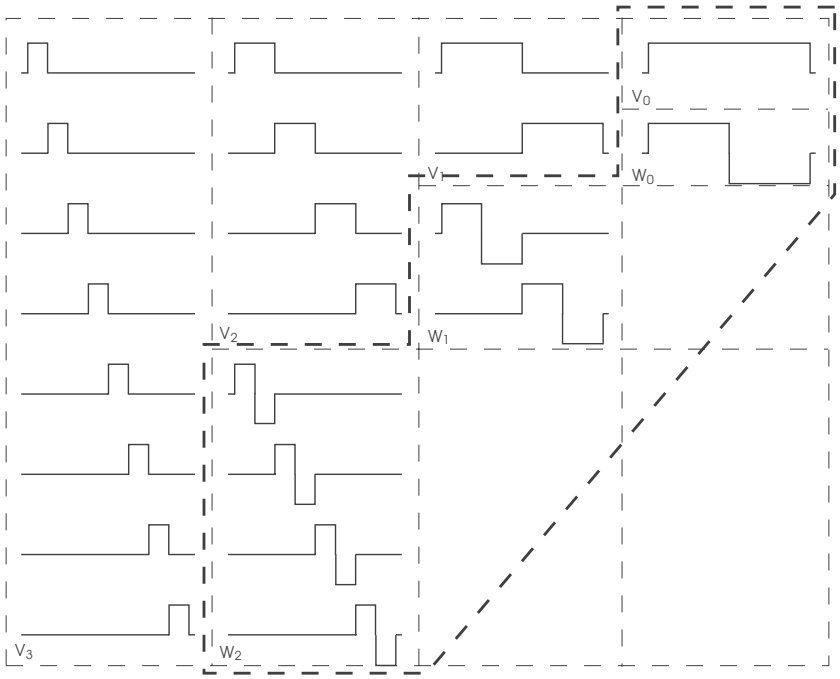


Figure P7.14

Therefore, the V_1 approximation is

$$\frac{\sqrt{2}}{24}\varphi_{1,0}(x)+\frac{7\sqrt{2}}{24}\varphi_{1,1}(x)$$

which, when plotted, is identical to the V_1 approximation in Fig. 7.15(d). The last two coefficients are $d_1(0)$ and $d_1(1)$, which are computed as in the example. Thus, the expansion is

$$y=\frac{\sqrt{2}}{24}\varphi_{1,0}(x)+\frac{7\sqrt{2}}{24}\varphi_{1,1}(x)+\left[\frac{-\sqrt{2}}{32}\psi_{1,0}(x)-\frac{3\sqrt{2}}{32}\psi_{1,1}(x)\right]+\cdots$$

Problem 7.16

(a) Because $M=4$, $J=2$, and $j_0=1$, the summations in Eqs. (7.3-5) through (7.3-7) are performed over $n=0,1,2,3$, $j=1$, and $k=0,1$. Using Haar functions and assuming that they are distributed over the range of the input sequence, we

get

$$\begin{aligned}
W_\varphi(1,0) &= \frac{1}{2}\left[f(0)\varphi_{1,0}(0)+f(1)\varphi_{1,0}(1)+f(2)\varphi_{1,0}(2)+f(3)\varphi_{1,0}(3)\right] \\
&= \frac{1}{2}\left[(1)(\sqrt{2})+(4)(\sqrt{2})+(-3)(0)+(0)(0)\right] = \frac{5\sqrt{2}}{2} \\
W_\varphi(1,1) &= \frac{1}{2}\left[f(0)\varphi_{1,1}(0)+f(1)\varphi_{1,1}(1)+f(2)\varphi_{1,1}(2)+f(3)\varphi_{1,1}(3)\right] \\
&= \frac{1}{2}\left[(1)(0)+(4)(0)+(-3)(\sqrt{2})+(0)(\sqrt{2})\right] = \frac{-3\sqrt{2}}{2} \\
W_\psi(1,0) &= \frac{1}{2}\left[f(0)\psi_{1,0}(0)+f(1)\psi_{1,0}(1)+f(2)\psi_{1,0}(2)+f(3)\psi_{1,0}(3)\right] \\
&= \frac{1}{2}\left[(1)(\sqrt{2})+(4)(-\sqrt{2})+(-3)(0)+(0)(0)\right] = \frac{-3\sqrt{2}}{2} \\
W_\psi(1,1) &= \frac{1}{2}\left[f(0)\psi_{1,1}(0)+f(1)\psi_{1,1}(1)+f(2)\psi_{1,1}(2)+f(3)\psi_{1,1}(3)\right] \\
&= \frac{1}{2}\left[(1)(0)+(4)(0)+(-3)(\sqrt{2})+(0)(-\sqrt{2})\right] = \frac{-3\sqrt{2}}{2}
\end{aligned}$$

so that the DWT is $\{5\sqrt{2}/2, -3\sqrt{2}/2, -3\sqrt{2}/2, -3\sqrt{2}/2\}$.

(b) Using Eq. (7.3-7),

$$\begin{aligned}
f(n) = \frac{1}{2}[&W_\varphi(1,0)\varphi_{1,0}(n)+W_\varphi(1,1)\varphi_{1,1}(n)+ \\
&W_\psi(1,0)\psi_{1,0}(n)+W_\psi(1,1)\psi_{1,1}(n)]
\end{aligned}$$

which, with $n=1$, becomes

$$\begin{aligned}
f(1) &= \frac{\sqrt{2}}{4}\left[(5)(\sqrt{2})+(-3)(0)+(-3)(\sqrt{2})+(-3)(0)\right] \\
&= \frac{2(\sqrt{2})^2}{4} = 1.
\end{aligned}$$

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.

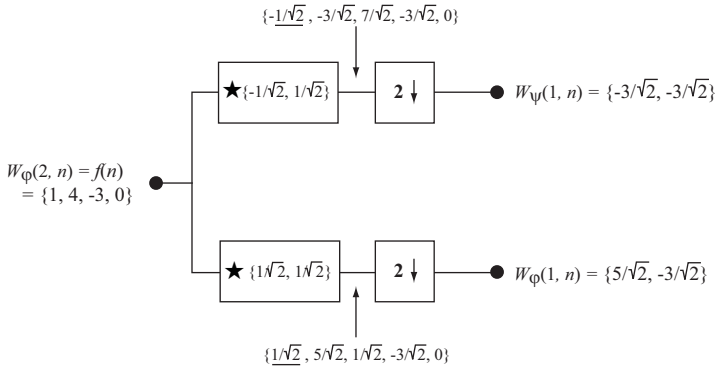


Figure P7.19

Problem 7.18

(a) The scale and translation parameters are continuous, which leads to the overcompleteness of the transform.

(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 7.20

The complexity is determined by the number of coefficients in the scaling and wavelet vectors—that is, by n in Eqs. (7.2-18) and (7.2-28). This defines the number of taps in filters $h_\psi(-n)$, $h_\varphi(-n)$, $h_\psi(n)$, and $h_\varphi(n)$.

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 $j_0 = k = 0$.

(b) Using the same reasoning as in part (a), the transform is $\{0, 2\sqrt{2}, 0, 0, 0, 0, 0, 0\}$.

(c) For the given transform, $W_\psi(2,2) = B$ and all other transform coefficients are 0. Thus, the input must be proportional to $\psi_{2,2}(x)$. The input sequence must be of the form $\{0,0,0,0,C,-C,0,0\}$ for some C. To determine C, use Eq. (7.3-6) to write

$$\begin{aligned} W_\psi(2,2) &= \frac{1}{\sqrt{8}}\{f(0)\psi_{2,2}(0) + f(1)\psi_{2,2}(1) + f(2)\psi_{2,2}(2) + f(3)\psi_{2,2}(3) + \\ &\quad f(4)\psi_{2,2}(4) + f(5)\psi_{2,2}(5) + f(6)\psi_{2,2}(6) + f(7)\psi_{2,2}(7)\} \\ &= \frac{1}{\sqrt{8}}\{(0)(0) + (0)(0) + (0)(0) + (0)(0) + (C)(2) + (-C)(-2) + \\ &\quad (0)(0) + (0)(0)\} \\ &= \frac{1}{\sqrt{8}}\{2C + 2C\} = \frac{4C}{\sqrt{8}} = \sqrt{2}C. \end{aligned}$$

Because this coefficient is known to have the value B, we have that $\sqrt{2}C = B$ or

$$C = \frac{\sqrt{2}}{2}B.$$

Thus, the input sequence is $\{0,0,0,0,\sqrt{2}B/2,-\sqrt{2}B/2,0,0\}$. To check the result substitute these values into Eq. (7.3-6):

$$\begin{aligned} W_\psi(2,2) &= \frac{1}{\sqrt{8}}\{(0)(0) + (0)(0) + (0)(0) + (0)(0) + (\frac{\sqrt{2}}{2}B)(2) + \\ &\quad (-\frac{\sqrt{2}}{2}B)(-2) + (0)(0) + (0)(0)\} \\ &= \frac{1}{\sqrt{8}}\{\sqrt{2}B + \sqrt{2}B\} \\ &= B. \end{aligned}$$

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

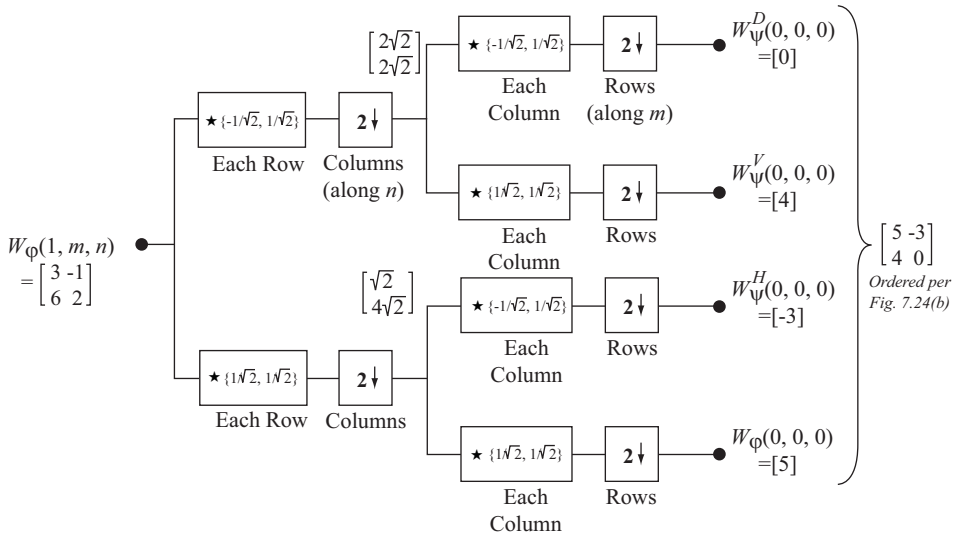


Figure P7.23

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 7.23

One pass through the FWT 2-d filter bank of Fig. 7.24(a) is all that is required (see Fig. P7.23).

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.

Problem 7.25

The table is completed as shown in Fig. P7.25. The functions are determined using Eqs. (7.2-18) and (7.2-28) with the Haar scaling and wavelet vectors from Examples 7.5 and 7.6:

$$\varphi(x) = \varphi(2x) + \varphi(2x-1)$$
$$\psi(x) = \varphi(2x) - \varphi(2x-1).$$

To order the wavelet functions in frequency, count the number of transitions that are made by each function. For example, V_0 has the fewest transitions (only 2) and lowest frequency content, while $W_{2,AA}$ has the most (9) transitions and correspondingly highest frequency content. From top to bottom in the figure, there are 2, 3, 5, 4, 9, 8, 6, and 7 transitions, respectively. Therefore, the frequency ordered subspaces are (from low to high frequency) V_0, W_0, $W_{1,D}$, $W_{1,A}$, $W_{2,DA}$, $W_{2,DD}$, $W_{2,AD}$, and $W_{2,AA}$.

Problem 7.26

(a) The analysis tree is shown in Fig. P7.26(a).

(b) The corresponding frequency spectrum is shown in Fig. P7.26(b).

Problem 7.27

First use the entropy measure to find the starting value for the input sequence, which is

$$E\{f(n)\} = \sum_{n=0}^{7} f^2(n) \ln\left[f^2(n)\right] = 2.7726.$$

Then perform an iteration of the FWT and compute the entropy of the generated approximation and detail coefficients. They are 2.0794 and 0, respectively. Since

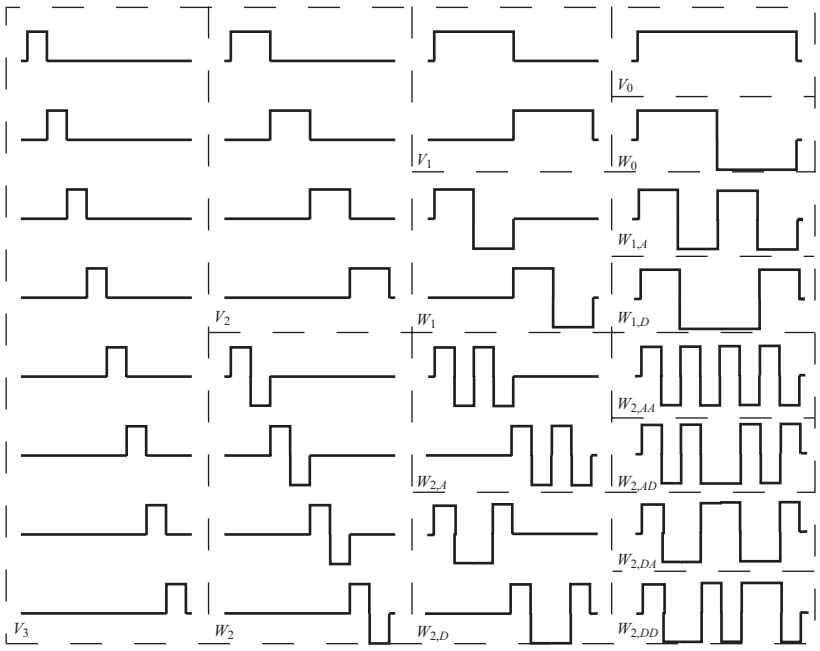


Figure P7.25

their sum is less than the starting entropy of 2.7726, we will use the decomposition.

Because the detail entropy is 0, no further decomposition of the detail is warranted. Thus, we perform another FWT iteration on the approximation to see if it should be decomposed again. This process is then repeated until no further decompositions are called for. The resulting optimal tree is shown in Fig. P7.27.

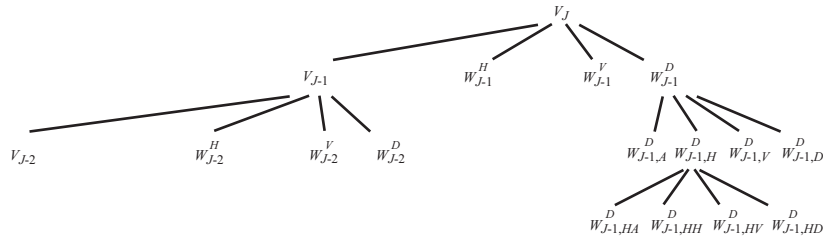


Figure P7.26(a)

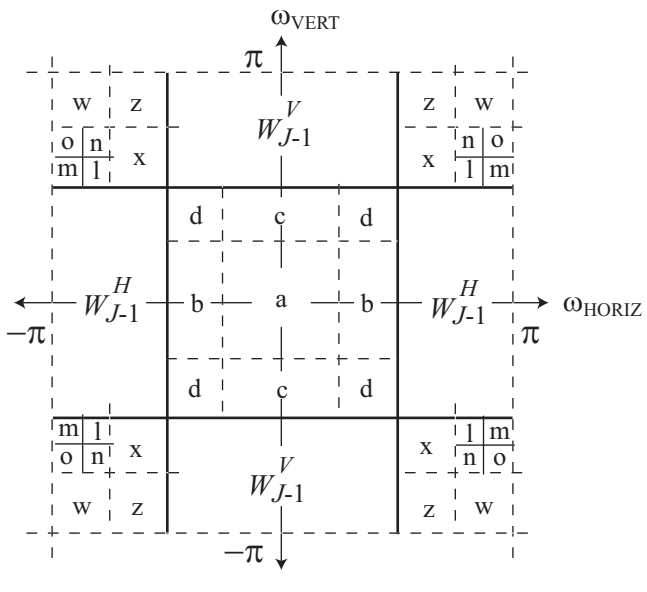


Figure P7.26(b)

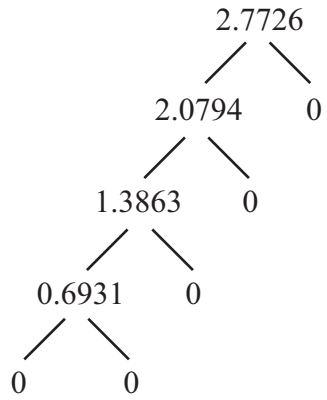


Figure P7.27

Chapter 8

Problem Solutions

Problem 8.1

(a) A histogram equalized image (in theory) has an intensity distribution which is uniform. That is, all intensities are equally probable. Eq. (8.1-4) thus becomes

$$L_{avg} = \frac{1}{2^n} \sum_{k=0}^{2^n-1} l\left(r_k\right)$$

where $1/2^n$ is the probability of occurrence of any intensity. Since all intensities are equally probable, there is no advantage to assigning any particular intensity fewer bits than any other. Thus, we assign each the fewest possible bits required to cover the 2^n levels. This, of course is n bits and L_{avg} becomes n bits also:

$$\begin{aligned} L_{avg} &= \frac{1}{2^n} \sum_{k=0}^{2^n-1} (n) \\ &= \frac{1}{2^n} \left(2^n\right) n \\ &= n. \end{aligned}$$

(b) Since spatial redundancy is associated with the geometric arrangement of the intensitess in the image, it is possible for a histogram equalized image to contain a high level of spatial redundancy - or none at all.

Problem 8.2

(a) A single line of raw data contains $n_1 = 2^n$ bits. The maximum run length would be 2^n and thus require n bits for representation. The starting coordinate

of each run also requires n bits since it may be arbitrarily located within the 2^n pixel line. Since a run length of 0 can not occur and the run-length pair $(0,0)$ is used to signal the start of each new line - an additional $2n$ bits are required per line. Thus, the total number of bits required to code any scan line is

$$\begin{aligned} n_2 &= 2n + N_{avg}(n+n) \\ &= 2n\left(1 + N_{avg}\right) \end{aligned}$$

where N_{avg} is the average number of run-length pairs on a line. To achieve some level of compression, C must be greater than 1. So,

$$\begin{aligned} C &= \frac{n_1}{n_2} \\ &= \frac{2^n}{2n\left(1+N_{avg}\right)} > 1 \end{aligned}$$

and

$$N_{avg} < \frac{2^{n-1}}{n} - 1.$$

(b) For $n = 10$, N_{avg} must be less than 50.2 run-length pairs per line.

Problem 8.3

The original pixel intensities, their 4-bit quantized counterparts, and the differences between them are shown in Table P8.3. Note that the quantized intensities must be multiplied by 16 to decode or decompress them for the rms error and signal-to-noise calculations.

Table P8.3

$f(x,y)$		$\hat{f}(x,y)$		$16\hat{f}(x,y) - f(x,y)$
Base 10	Base 2	Base 2	Base 10	Base 10
108	01101100	0110	6	-12
139	10001011	1000	8	-11
135	10000111	1000	8	-7
244	11110100	1111	15	-4
172	10101100	1010	10	-12
173	10101101	1010	10	-13
56	00111000	0011	3	-8
99	01100011	0110	6	-3

Using Eq. (8.1-10), the rms error is

$$\begin{aligned} e_{rms} &= \sqrt{\frac{1}{8}\sum_{x=0}^{0}\sum_{y=0}^{7}\left[16\hat{f}(x,y)-f(x,y)\right]^2} \\ &= \sqrt{\frac{1}{8}\left[(-12)^2+(-11)^2+(-7)^2+(-4)^2+(-12)^2+(-13)^2+(-8)^2+(-3)^2\right]} \\ &= \sqrt{\frac{1}{8}(716)} \\ &= 9.46 \end{aligned}$$

or about 9.5 intensity levels. From Eq. (8.1-11), the signal-to-noise ratio is

$$\begin{aligned} SNR_{ms} &= \frac{\sum_{x=0}^{0}\sum_{y=0}^{7}\left[16\hat{f}(x,y)\right]^2}{\sum_{x=0}^{0}\sum_{y=0}^{7}\left[16\hat{f}(x,y)-f(x,y)\right]^2} \\ &= \frac{96^2+128^2+128^2+240^2+160^2+160^2+48^2+96^2}{716} \\ &= \frac{162304}{716} \\ &\simeq 227. \end{aligned}$$

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 8.5

(a) The maximum compression with Huffman coding is $C = \frac{8}{5.3} = 1.509$.

(b) No, but Huffman coding does provide the smallest possible number of bits per intensity value subject to the constraint that the intensities are coded one at a time.

(c) One possibility is to eliminate spatial redundancies in the image before Huffman encoding. For instance, compute the differences between adjacent pixels and Huffman code them.

Problem 8.6

The conversion factors are computed using the logarithmic relationship

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

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

Problem 8.7

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

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

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 8.8

(a) There are two unique codes.

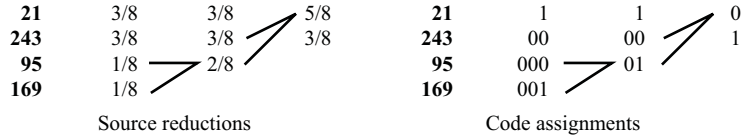


Figure P8.9

(b) The codes are: (1) 0, 11, 10 and (2) 1, 00, 01. The codes are complements of one another. They are constructed by following the Huffman procedure for three symbols of arbitrary probability.

Problem 8.9

(a) The entropy of the image is estimated using Eq. (8.1-7) to be

$$\begin{aligned}
\tilde{H} &= -\sum_{k=0}^{255} p_r(r_k)\log_2 p_r(r_k) \\
&= -\left[\frac{12}{32}\log_2\frac{12}{32}+\frac{4}{32}\log_2\frac{4}{32}+\frac{4}{32}\log_2\frac{4}{32}+\frac{12}{32}\log_2\frac{12}{32}\right] \\
&= -[-0.5306-0.375-0.375-0.5306] \\
&= 1.811 \text{ bits/pixel.}
\end{aligned}$$

The probabilities used in the computation are given in Table P8.9-1.

Table P8.9-1

Intensity	Count	Probability
21	12	3/8
95	4	1/8
169	4	1/8
243	12	3/8

(b) Figure P8.9 shows one possible Huffman source reduction and code assignment. Use the procedures described in Section 8.2.1. The intensities are first arranged in order of probability from the top to the bottom (at the left of the source reduction diagram). The least probable symbols are them combined to create a reduced source and the process is repeated from left to right in the diagram. Code words are then assigned to the reduced source symbols from right to left. The codes assigned to each intensity value are read from the left side of the code assignment diagram.

(c) Using Eq. (8.1-4), the average number of bits required to represent each pixel in the Huffman coded image (ignoring the storage of the code itself) is

$$L_{avg} = 1\left(\frac{3}{8}\right) + 2\left(\frac{3}{8}\right) + 3\left(\frac{1}{8}\right) + 3\left(\frac{1}{8}\right) = \frac{15}{8} = 1.875 \text{ bits/pixel.}$$

Thus, the compression achieved is

$$C = \frac{8}{1.875} = 4.27.$$

Because the theoretical compression resulting from the elimination of all coding redundancy is $\frac{8}{1.811} = 4.417$, the Huffman coded image achieves $\frac{4.27}{4.417} \times 100$ or 96.67% of the maximum compression possible through the removal of coding redundancy alone.

(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.

(e) Construct a difference image by replicating the first column of the original image and using the arithmetic difference between adjacent columns for the remaining elements. The difference image is

$$\begin{array}{cccccccc} 21 & 0 & 0 & 74 & 74 & 74 & 0 & 0 \\ 21 & 0 & 0 & 74 & 74 & 74 & 0 & 0 \\ 21 & 0 & 0 & 74 & 74 & 74 & 0 & 0 \\ 21 & 0 & 0 & 74 & 74 & 74 & 0 & 0 \end{array}$$

The probabilities of its various elements are given in Table 8.9-3.

Table P8.9-3

Intensity difference	Count	Probability
21	4	1/8
0	16	1/2
74	12	3/8

The entropy of the difference image is estimated using Eq. (8.1-7) to be

$$\tilde{H} = -\left[\frac{1}{8}\log_2\frac{1}{8} + \frac{1}{2}\log_2\frac{1}{2} + \frac{3}{8}\log_2\frac{3}{8}\right] = 1.41 \text{ bits/pixel.}$$

(f) The entropy calculated in (a) is based on the assumption of statistically independent pixels. The entropy (of the pixel pairs) computed in (d), which is smaller that the value found in (a), reveals that the pixels are not statistically independent. There is at least $(1.811 - 1.25) = 0.56$ bits/pixel of spatial redundancy in the image. The difference image mapping used in (e) removes most of that spatial redundancy, leaving only $(1.41 - 1.25) = 0.16$ bits/pixel.

Problem 8.10

The decoded message is $a_3a_6a_6a_2a_5a_2a_2a_2a_4$.

Problem 8.11

The Golomb code is shown in Table P8.11. It is computed by the procedure outlined in Section 8.2.2 in conjunction with Eq. (8.2-1).

Table P8.11

Integer n	Unary code of $\lfloor \frac{n}{m} \rfloor$	Truncated $n \bmod m$	Golomb code $G_3(n)$
0	0	0	00
1	0	10	010
2	0	11	011
3	10	0	100
4	10	10	1010
5	10	11	1011
6	110	0	1100
7	110	10	11010
8	110	101	11011
9	1110	0	11100
10	1110	10	111010
11	1110	101	111011
12	11110	0	111100
13	11110	10	1111010
14	11110	11	1111011
15	111110	0	1111100

Problem 8.12

To decode $G_m(n)$, let $k = \lceil \log_2 m \rceil$ and $c = 2^k - m$. Then:

1. Count the number of 1s in a left-to-right scan of a concatenated $G_m(n)$ bit sequence before reaching the first 0, and multiply the number of 1s by m.

2. If the decimal equivalent of the next $k-1$ bits is less than c, add it to result from step 1; else add the decimal equivalent of the next k bits and subtract c.

For example, to decode the first $G_3(n)$ code in the Golomb coded bit sequence 1111010011..., let $k = \lceil \log_2 3 \rceil = 2$, $c = 2^2 - 3 = 1$, and note that there are 4 1s in a left-to-right scan of the bit stream before reaching the first 0. Multiplying the number of 1s by 3 yields 12 (the result of step 1). The bit following the 0 identified in step 1 is a 1, whose decimal equivalent is not less than c, i.e., $1 \not< 1$. So add the decimal equivalent of the 2 bits following the 0 identified in step 1 and subtract 1. Thus, the first integer is $12 + 2 - 1 = 13$. Repeat the process for the next code word, which begins with bits 011...

Problem 8.13

The probability mass function defined by Eq. (8.2-2) is for an infinite set of integers. It is not possible to list an infinite set of source symbols and perform the source reductions required by Huffman's approach.

Problem 8.14

The exponential Golomb code is shown in Table P8.14. It is computed by the procedure outlined in Section 8.2.2 in conjunction with Eqs. (8.2-5) and (8.2-6).

Table P8.14

Integer n	Parameter i	Golomb code $G^2_{exp}(n)$
0	0	000
1	0	001
2	0	010
3	0	011
4	1	10000
5	1	10001
6	1	10010
7	1	10011
8	1	10100
9	1	10101
10	1	10110
11	1	10111
12	2	1100000
13	2	1100001
14	2	1100010
15	2	1100011

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.

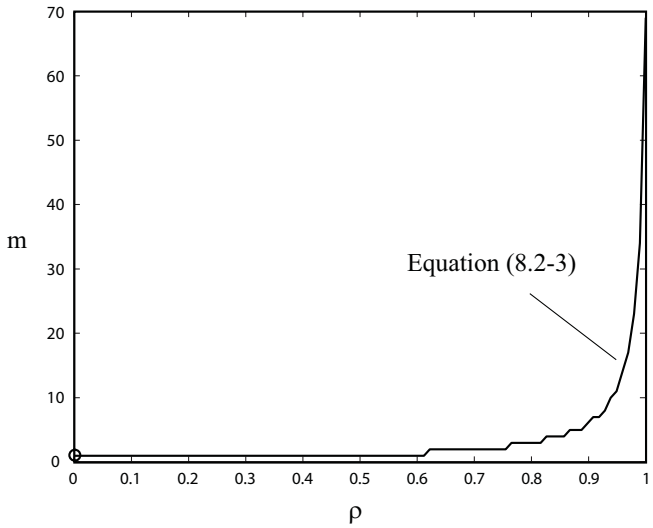


Figure P8.16

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 8.16

The graph shown in Figure P8.16 was obtained using MATLAB. Note that the function is not defined for $\rho = \{0, 1\}$.

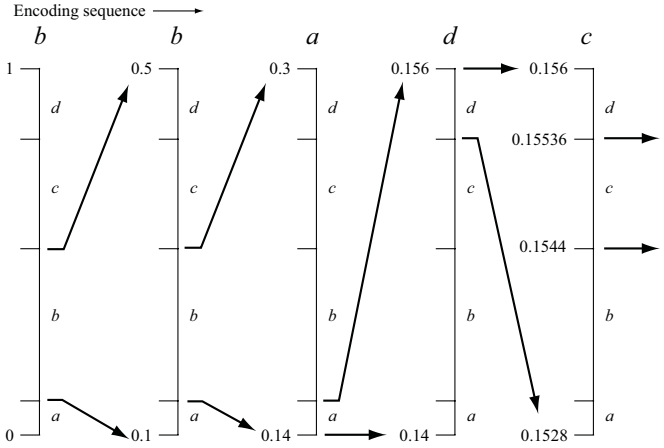


Figure P8.17

Problem 8.17

Figure P8.17 shows the arithmetic coding process as described in Section 8.2.3. Any value in the interval [0.1544, 0.15536) at the right side of the figure can be used to code the sequence. For example, the value 0.155.

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 8.19

Assume that the first 256 codes in the starting dictionary are the ASCII codes. If you assume 7-bit ASCII, the first 128 locations are all that are needed. In either case, the ASCII "a" corresponds to location 97. The coding proceeds as shown in Table P8.19. The encoded output is

97 256 257 258 97.

Table P8.19

Recognized	Character	Output	Dict. Address	Dict. Entry
	a			
a	a	97	256	aa
a	a			
aa	a	256	257	aaa
a	a			
aa	a			
aaa	a	257	258	aaaa
a	a			
aa	a			
aaa	a			
aaaa	a	258	259	aaaaa
a		97		

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 8.21

Using the BMP specification given in Example 8.8 of Section 8.2.5, the first two bytes indicate that the uncompressed data begins with a run of 4s with length 3. In a similar manner, the second two bytes call for a run of 6s with length 5. The first four bytes of the BMP encoded sequence are encoded mode. Because the 5th byte is 0 and the 6th byte is 3, absolute mode is entered and the next three values are taken as uncompressed data. Because the total number of bytes in absolute mode must be aligned on a 16-bit word boundary, the 0 in the 10th byte of the encoded sequence is padding and should be ignored. The final two bytes specify an encoded mode run of 47s with length 2. Thus, the complete uncompressed sequence is {4, 4, 4, 6, 6, 6, 6, 6, 103, 125, 67, 47, 47}.

Problem 8.22

(a) Using Eq. (8.2-9), form Table P8.22.

Table P8.22

Binary	Gray Code		Binary	Gray Code
0000	0000		1000	1100
0001	0001		1001	1101
0010	0011		1010	1111
0011	0010		1011	1110
0100	0110		1100	1010
0101	0111		1101	1011
0110	0101		1110	1001
0111	0100		1111	1000

(b) The procedure is to work from the most significant bit to the least significant bit using the equations:

$$a_{m-1} = g_{m-1}$$
$$a_i = g_i \oplus a_{i+1} \quad 0 \le i \le m-2.$$

The decoded binary value is thus 0101100111010.

Problem 8.23

Following the procedure in the flow chart of Fig. 8.14 in the book, the proper code is

0001 010 1 0011000011 0001

where the spaces have been inserted for readability alone. The coding mode sequence is pass, vertical (1 left), vertical (directly below), horizontal (distances 3 and 4), and pass.

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 8.25

In general, the number of MAD computations for single-pixel precision and displacements $\pm dx$ and $\pm dy$ is $(2dx+1)(2dy+1)$. With both dx and dy equal to ± 8, the number of MAD computations for this problem is 17^2 or 289 per macroblock. With 8×8 macroblocks, each MAD computation involves $8^2(3)=192$ operations (64 subtractions, absolute values, and additions). So, the total number of math operations for 8×8 macroblocks and single pixel displacements $dx=dy=8$ is $289\times 192=55,488$.

For $\frac{1}{4}$ pixel precision, the number of MAD computations is multiplied by 16, yielding $16\times 289=4,624$ MAD computations per macroblock. The number of math operations for each MAD computation is increased from 192 to $192+8^2(4)=448$, where the additional operations are for bilinear interpolation. So, the total number of math operations for 8×8 macroblocks and $\frac{1}{4}$ pixel displacements $dx=dy=8$ (using bilinear interpolation) is $4624\times 448=2,071,552$.

Problem 8.26

There are several ways in which backward motion-compensated prediction can help:

1. It provides a means of predicting uncovered background when an object is moving.
2. It provides a means of predicting pixels on the edges of frames when the camera is panning.
3. Backward and forward prediction can average the noise in two reference frames to yield a more accurate prediction.

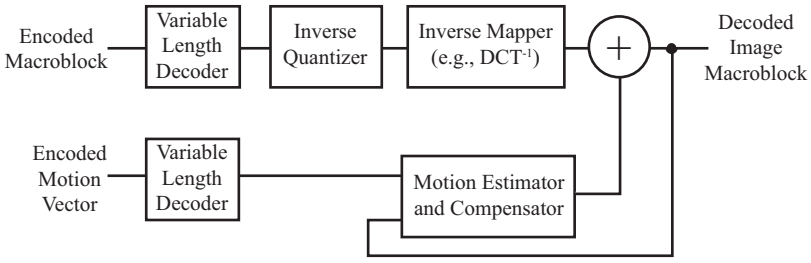


Figure P8.27

Problem 8.27

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

Problem 8.28

(a) Substituting $\rho_h = 0$ into Eq. (8.2-49) and evaluating it to form the elements of **R** and **r**, we get

$$\mathbf{R} = \sigma^2 \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$$

$$\mathbf{r} = \sigma^2 \begin{bmatrix} \rho \\ \rho^2 \end{bmatrix}.$$

(b) First form the inverse of **R**,

$$\mathbf{R}^{-1} = \frac{1}{\sigma^2\left(1-\rho^2\right)} \begin{bmatrix} 1 & -\rho \\ -\rho & 1 \end{bmatrix}.$$

Then, perform the matrix multiplication of Eq. (8.2-45):

$$\alpha = \mathbf{R}^{-1}\mathbf{r} = \frac{\sigma^2}{\sigma^2\left(1-\rho^2\right)} \begin{bmatrix} \rho\left(1-\rho^2\right) \\ 0 \end{bmatrix} = \begin{bmatrix} \rho \\ 0 \end{bmatrix}.$$

Thus, $\alpha_1 = \rho$ and $\alpha_2 = 0$.

(c) The variance is computed using Eq. (8.2-48):

$$\sigma_e^2 = \sigma^2 - \alpha^T\mathbf{r} = \begin{bmatrix} \rho & 0 \end{bmatrix} \begin{bmatrix} \rho \\ \rho^2 \end{bmatrix} = \sigma^2\left(1-\rho^2\right).$$

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 $s_1 \le A$)

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

$$\frac{1}{2A}\int_0^{\frac{1}{2}(t_1+t_2)} (s-t_1)\,ds = \frac{s^2}{2} - t_1 s \Big|_0^{\frac{1}{2}(t_1+t_2)} = 0$$

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

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:

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

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,

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

Problem 8.30

There are many ways to approach this problem. For example, JPEG-2000 compression could be used. Here's another alternative. Because the T1 transfer rate is 1.544 Mbit/sec, a 6 second transfer will provide

$$(1.544 \times 10^6)(6\,\text{sec}) = 9.264 \times 10^6\,\text{bits}$$

of data. The initial approximation of the X-ray must contain no more than this number of bits. The required compression ratio is thus

$$C = \frac{4096 \times 4096 \times 12}{9.264 \times 10^6} = 21.73$$

The JPEG transform coding approach of Section 8.2.8 can achieve this level of compression and provide reasonably good reconstructions. At the X-ray encoder, the X-ray can be JPEG compressed using a normalization array that yields about a 25:1 compression. While it is being transmitted over the T1 line to the remote viewing station, the encoder can decode the compressed JPEG data and identify the "differences" between the resulting X-ray approximation and the

original X-ray image. Since we wish to transmit these "differences" over a span of 1 minute with refinements every 5 - 6 seconds, there can be no more than

$$\begin{array}{ccc} \frac{60}{6} & \text{to} & \frac{60}{5} \\ \text{or } 10 & \text{to} & 12 \end{array}$$

refinements. If we assume that 12 refinements are made and that each refinement corresponds to the "differences" between one of the 12 bits in the original X-ray and the JPEG reconstructed approximation, then the compression that must be obtained per bit (to allow a 6 second average transfer time for each bit) is

$$C = \frac{4096 \times 4096 \times 1}{9.264 \times 10^6} = 1.81$$

where, as before, the bottom of the fraction is the number of bits that can be transmitted over a T1 line in 6 seconds. Thus, the "difference" data for each bit must be compressed by a factor just less than 2. One simple way to generate the "difference information" is to XOR the actual X-ray with the reconstructed JPEG approximation. The resulting binary image will contain a 1 in every bit position at which the approximation differs from the original. If the XOR result is transmitted one bit at a time beginning with the MSB and ending with the LSB, and each bit is compressed by an average factor of 1.81:1, we will achieve the performance that is required in the problem statement. To achieve an average error-free bit-plane compression of 1.81:1 (see Section 8.2.7), the XOR data can be Gray coded, run-length coded, and finally variable-length coded. A conceptual block diagram for both the encoder and decoder are given in Fig. P8.30. Note that the decoder computes the bit refinements by XORing the decoded XOR data with the reconstructed JPEG approximation.

Problem 8.31

To demonstrate the equivalence of the lifting based approach and the traditional FWT filter bank method, we simply derive general expressions for one of the odd and even outputs of the lifting algorithm of Eq. (8.2-62). For example, the $Y(0)$ output of step 4 of the algorithm can be written as

$$\begin{aligned} Y_4(0) &= Y_2(0) + \delta\left[Y_3(-1) + Y_3(1)\right] \\ &= X(0) + \beta\left[Y_1(-1) + Y_1(1)\right] + \delta\left[Y_3(-1) + Y_3(1)\right] \end{aligned}$$

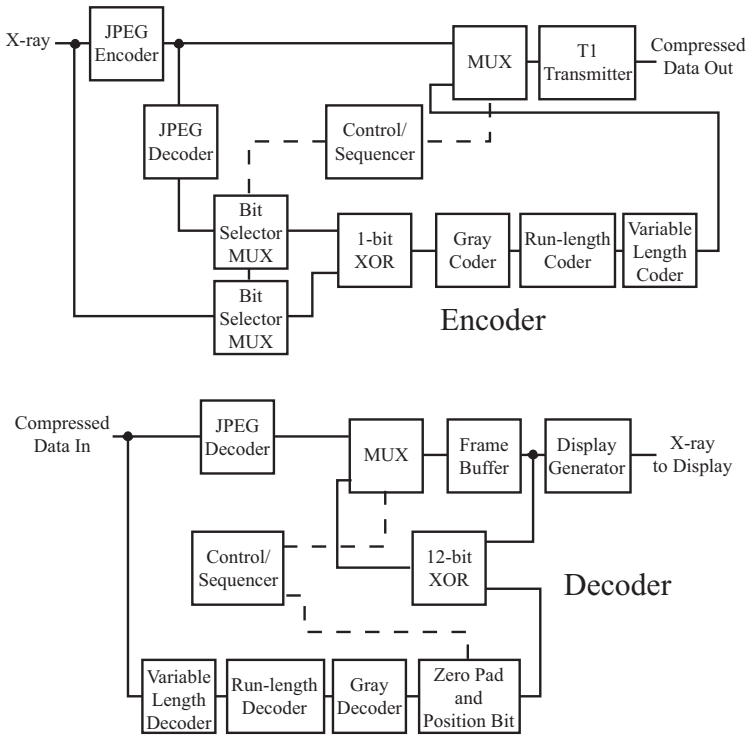


Figure P8.30

where the subscripts on the Y's have been added to identify the step of the lifting algorithm from which the value is generated. Continuing this substitution pattern from earlier steps of the algorithm until $Y_4(0)$ is a function of X's only, we get

$$\begin{aligned} Y(0) &= \left[1+2\alpha\beta+2\alpha\delta+6\alpha\beta\gamma\delta+2\gamma\delta\right]X(0) \\ &\quad +\left[\beta+3\beta\gamma\delta+\delta\right]X(1) \\ &\quad +\left[\alpha\beta+4\alpha\beta\gamma\delta+\alpha\delta+\gamma\delta\right]X(2) \\ &\quad +\left[\beta\gamma\delta\right]X(3) \\ &\quad +\left[\alpha\beta\gamma\delta\right]X(4) \\ &\quad +\left[\beta+3\beta\gamma\delta+\delta\right]X(-1) \\ &\quad +\left[\alpha\beta+4\alpha\beta\gamma\delta+\alpha\delta+\gamma\delta\right]X(-2) \\ &\quad +\left[\beta\gamma\delta\right]X(-3) \\ &\quad +\left[\alpha\beta\gamma\delta\right]X(-4). \end{aligned}$$

Thus, we can form the lowpass analysis filter coefficients shown in Table P8.31-1.

Table P8.31-1

Coefficient Index	Expression	Value
±4	$\alpha\beta\gamma\delta/K$	0.026748757
±3	$\beta\gamma\delta/K$	-0.016864118
±2	$(\alpha\beta+4\alpha\beta\gamma\delta+\alpha\delta+\gamma\delta)/K$	-0.07822326
±1	$(\beta+3\beta\gamma\delta+\delta)/K$	0.26686411
0	$(1+2\alpha\beta+2\alpha\delta+6\alpha\beta\gamma\delta+2\gamma\delta)/K$	0.60294901

Here, the coefficient expressions are taken directly from our expansion of $Y(0)$ and the division by K is in accordance with step 6 of Eq. (8.2-62). The coefficient values in column 3 are determined by substituting the values of α, β, γ, δ, and K from the text into the expressions of column 2. A similar derivation beginning with

$$Y_3(1) = Y_1(1) + \gamma\,[Y_2(0) + Y_2(2)]$$

yields

$$\begin{aligned} Y(1) \;=\; & [\alpha+3\alpha\beta\gamma+\delta]\,X(0) \\ & +[1+2\beta\gamma]\,X(1) \\ & +[\alpha+3\alpha\beta\gamma+\delta]\,X(2) \\ & +[\beta\gamma]\,X(3) \\ & +[\alpha\beta\gamma]\,X(4) \\ & +[\beta\gamma]\,X(-1) \\ & +[\alpha\beta\gamma]\,X(-2) \end{aligned}$$

from which we can obtain the highpass analysis filter coefficients shown in Table P8.31-2

Table P8.31-2

Coefficient Index	Expression	Value
-2	$-K(\alpha\beta\gamma)$	-0.091271762
-1	$-K(\beta\gamma)$	0.057543525
0	$-K(\alpha+3\alpha\beta\gamma+\delta)$	0.591271766
1	$-K(1+2\beta\gamma)$	-1.115087053
2	$-K(\alpha+3\alpha\beta\gamma+\delta)$	0.591271766
3	$-K(\beta\gamma)$	0.057543525
4	$-K(\alpha\beta\gamma)$	-0.091271762

Problem 8.32

From Eq. (8.6-5) and the problem statement, we get that

$$\mu_{2LL} = \mu_0 = 8$$
$$\epsilon_{2LL} = \epsilon_0 + 2 - 2 = \epsilon_0 = 8.$$

Substituting these values into Eq. (8.2-64), we find that for the $2LL$ subband

$$\Delta_{2LL} = 2^{(8+0)-8}\left[1 + \frac{8}{2^{11}}\right] = 1.00390625.$$

Here, we have assumed an 8-bit image so that $R_b = 8$. Likewise, using Eqs. (8.2-65), (8.2-64), and Fig. 8.48 (to find the analysis gain bits for each subband), we get

$$\Delta_{2HH} = 2^{(8+2)-8}\left[1 + \tfrac{8}{2^{11}}\right] = 4.015625$$
$$\Delta_{2HL} = \Delta_{2LH} = 2^{(8+1)-8}\left[1 + \tfrac{8}{2^{11}}\right] = 2.0078125$$
$$\Delta_{1HH} = 2^{(8+2)-8}\left[1 + \tfrac{8}{2^{11}}\right] = 4.015625$$
$$\Delta_{1HL} = \Delta_{1LH} = 2^{(8+1)-8}\left[1 + \tfrac{8}{2^{11}}\right] = 2.0078125.$$

Problem 8.33

One approach is to implement Eq. (8.3-1) using Fourier transforms. Using the properties of the Fourier transform:

$$\begin{aligned} f_w &= (1-\alpha)f + \alpha w \\ \Im[f_w] &= \Im[(1-\alpha)f + \alpha w] \\ F_w &= \Im[(1-\alpha)f] + \Im[\alpha w] \\ &= (1-\alpha)\Im[f] + \alpha\Im[w] \end{aligned}$$

So, visible watermarking in the transform domain can be accomplished by adding a scaled (by α) version of a watermark's Fourier transform to a scaled version (by $1-\alpha$) of an image's Fourier transform and taking the inverse transform of the sum. This is obviously more computationally demanding than the equivalent spatial domain approach [Eq. (8.3-1)].

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:

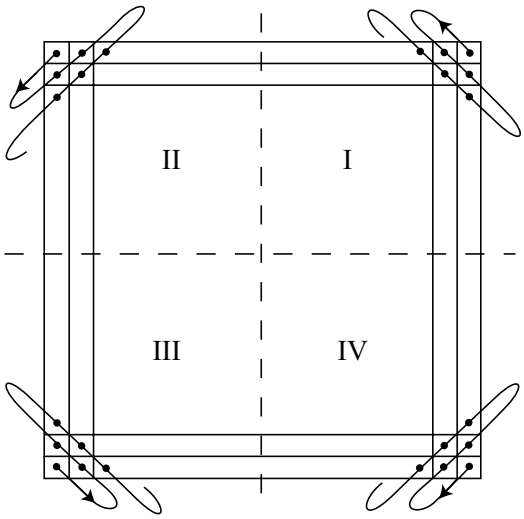


Figure P8.34

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.

Problem 8.35

There are many ways to watermark an image using DWTs. A simple example that has been reported in the literature uses the following watermark insertion technique:

1. Create a watermark by generating a 1000 element pseudo-random sequence of numbers, $\omega_1, \omega_2, ..., \omega_{1000}$, taken from a Gaussian distribution with zero mean and unit variance.

2. Compute the $L+1$-level DWT of the image to be watermarked. Choose L so that the number of approximation coefficients at level $J-L$ is about 1000. Recall from chapter 7 that these coefficients were denoted $W_\varphi(J-L, m, n)$ where $J = \log_2 N$ and the image is of size $N \times N$. For the purposes of this algorithm, we will call the selected coefficients $c_i(i)$ for $1 \leq i \leq 1000$.

3. Compute the average of the selected approximation coefficients $\overline{c}$.

4. Embed watermark ω into the selected approximation coefficients using

$$c_i' = \overline{c} + [c_i - \overline{c}](1 + \alpha\omega_i)$$

   where α determines the "intensity" of the watermark.

5. Compute the watermarked image by taking the inverse DWT with the marked approximation coefficients replacing the original unmarked coefficients.

Watermark detection is accomplished as follows:

1. Compute the $L+1$-level DWT of the image in question.

2. Compute the average $\bar{\hat{c}}$ and variance $\sigma(\hat{c})$ of the level $J-L$ approximation coefficients, $\hat{c}_1, \hat{c}_2, ..., \hat{c}_{1000}$.

3. Extract watermark $\hat{\omega}$ using

$$\hat{\omega}_i = \frac{\left(\hat{c}_i - \bar{\hat{c}}\right)\frac{\sigma(c)}{\sigma(\hat{c})} - (c_i - \bar{c})}{c_i - \bar{c}}$$

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

Chapter 9

Problem Solutions

Problem 9.1

(a) Converting a rectangular to a hexagonal grid basically requires that even and odd lines be displaced horizontally with respect to each other by one-half the horizontal distance between adjacent pixels (see the figure in the problem statement). Because in a rectangular grid there are no pixel values defined at the new locations, a rule must be specified for their creation. A simple approach is to double the image resolution in both dimensions by interpolation (see Section 2.4.4). Then, the appropriate 6-connected points are picked out of the expanded array. The resolution of the new image will be the same as the original (but the former will be slightly blurred due to interpolation). Figure P9.1(a) illustrates this approach. The black points are the original pixels and the white points are the new points created by interpolation. The squares are the image points picked for the hexagonal grid arrangement.

(b) Rotations in a 6-neighbor arrangement are invariant in 60° increments.

(c) Yes. Ambiguities arise when there is more than one path that can be followed from one 6-connected pixel to another. Figure P9.1(b) shows an example, in which the 6-connected points of interest are in black.

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

Figure P9.1

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.

(b) Only one pass is required. Application of the hit-or-miss transform using a given B^i finds all instances of occurrence of the pattern described by that structuring element.

(c) The order does matter. For example, consider the sequence of points in Fig. P9.2(b), and assume that we are traveling from left to right. If B^1 is applied first, point a will be deleted and point b will remain after application of all other structuring elements. If, on the other hand, B^3 is applied first, point b will be deleted and point a will remain. Thus, we would end up with different (but acceptable) m-paths.

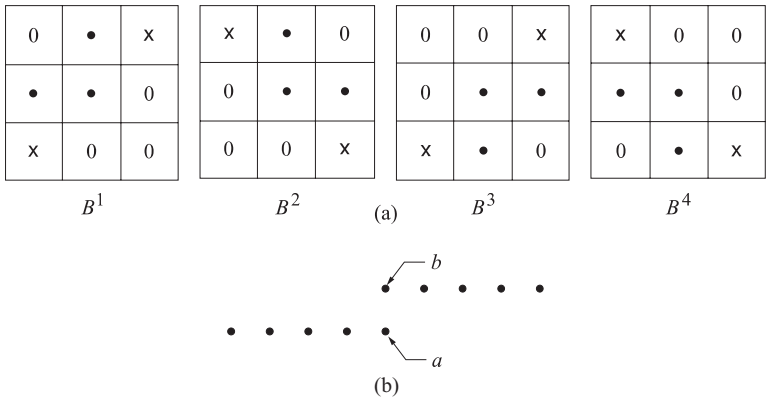


Figure P9.2

Problem 9.3

See Fig. P9.3. Keep in mind that erosion is the set formed from the locations of the *origin* of the structuring element, such that the structuring element is contained within the set being eroded.

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 continuous 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.

(c) See Fig. P9.4(b). Here, we see that the lower right point is not connected to the others.

(d) See Fig. 9.4(c). The two inner points are 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.

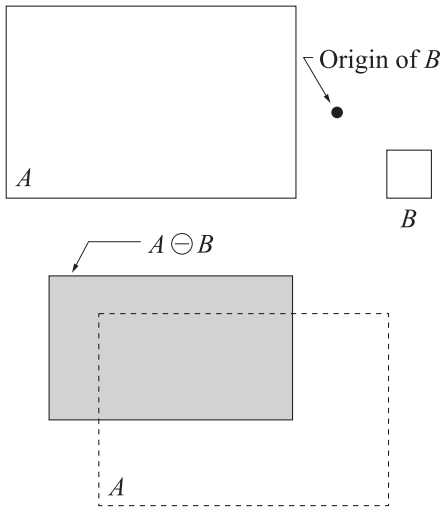


Figure P9.3

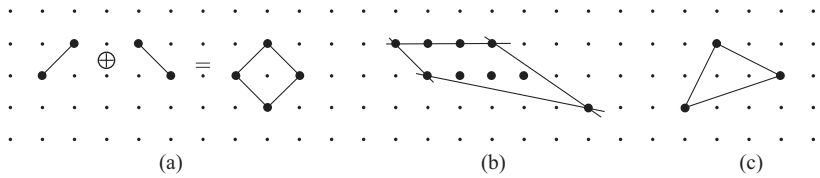


Figure P9.4

(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 9.6

The solutions to (a) through (d) are shown from top to bottom in Fig. P9.6.

Problem 9.7

(a) The dilated image will grow without bound.

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

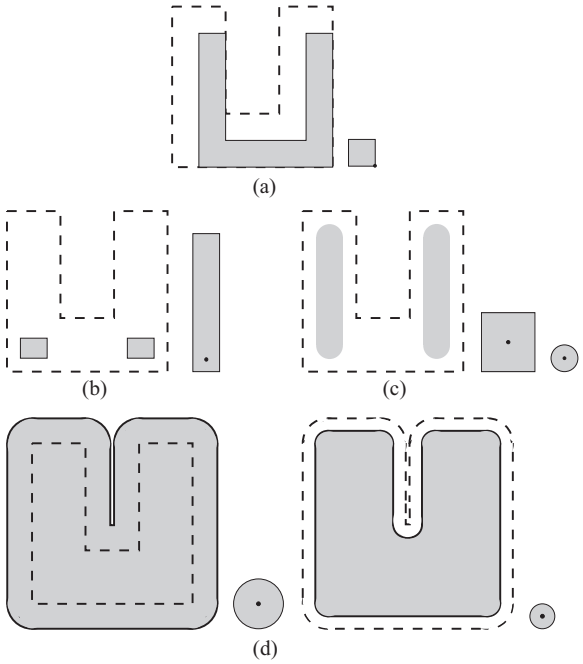


Figure P9.5

Problem 9.8

(a) The image will erode to one element.

(b) The smallest set that contains the structuring element.

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 9.10

(a) Let $x \in A \ominus B$. Then, from the definition of erosion given in the problem statement, for every $b \in B$, $x+b \in A$. But, $x+b \in A$ implies that $x \in (A)_{-b}$. Thus,

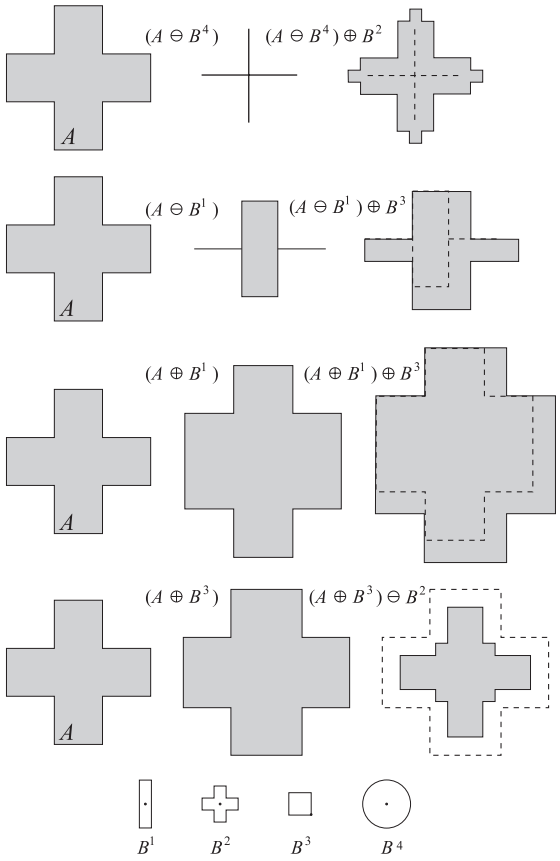


Figure P9.6

for *every* $b \in B$, $x \in (A)_{-b}$, which implies that $x \in \bigcap_{b \in B} (A)_{-b}$. Suppose now that $x \in \bigcap_{b \in B} (A)_{-b}$. Then, for every $b \in B$, $x \in (A)_{-b}$. Thus, for every $b \in B$, $x + b \in A$ which, from the definition of erosion, means that $x \in A \ominus B$.

(b) Suppose that $x \in A \ominus B = \bigcap_{b \in B} (A)_{-b}$. Then, for every $b \in B$, $x \in (A)_{-b}$, or $x + b \in A$. But, as shown in Problem 9.9, $x + b \in A$ for every $b \in B$ implies that $(B)_x \subseteq A$, so that $x \in A \ominus B = \{x \in Z^2 \,|\, (B)_x \subseteq A\}$. Similarly, $(B)_x \subseteq A$ implies that all elements of $(B)_x$ are contained in A, or $x + b \in A$ for every $b \in B$ or, as in (a), $x + b \in A$ implies that $x \in (A)_{-b}$. Thus, if for every $b \in B$, $x \in (A)_{-b}$, then $x \in \bigcap_{b \in B} (A)_{-b}$.

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 9.12

(a) Suppose that $x \in A \oplus B$. Then, for some $a \in A$ and $b \in B$, $x = a + b$. Thus, $x \in (A)_b$ and, therefore, $x \in \bigcup_{b \in B} (A)_b$. On the other hand, suppose that $x \in \bigcup_{b \in B} (A)_b$. Then, for some $b \in B$, $x \in (A)_b$. However, $x \in (A)_b$ implies that there exists an $a \in A$ such that $x = a + b$. But, from the definition of dilation given in the problem statement, $a \in A$, $b \in B$, and $x = a + b$ imply that $x \in A \oplus B$.

(b) Suppose that $x \in \bigcup_{b \in B} (A)_b$. Then, for some $b \in B$, $x \in (A)_b$. However, $x \in (A)_b$ implies that there exists an $a \in A$ such that $x = a + b$. But, 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 $x \in \left[(\hat{B})_x \cap A \neq \emptyset\right]$. Now, suppose that $x \in \left[(\hat{B})_x \cap A \neq \emptyset\right]$. The condition $(\hat{B})_x \cap A \neq \emptyset$ implies that for some $b \in B$, $x - b \in A$ or $x - b = a$ (i.e., $x = a + b$) for some $a \in A$. But, if $x = a + b$ for some $a \in A$ and $b \in B$, then $x \in (A)_b$ and, therefore, $x \in \bigcup_{b \in B} (A)_b$.

Problem 9.13

From the definition of dilation, Eq. (9.2-3),

$$(A \oplus B)^c = \left\{z \,\middle|\, (\hat{B})_z \cap A \neq \emptyset\right\}^c.$$

The *complement* of the set of z's that satisfy $(\hat{B})_z \cap A \neq \emptyset$ is the set of z's such that $(\hat{B})_z \cap A = \emptyset$. In turn, $(\hat{B})_z \cap A = \emptyset$ implies that that $(\hat{B})_z$ is contained in A^c. That is, $(\hat{B})_z \subseteq A^c$, and we can write

$$\begin{aligned}(A \oplus B)^c &= \left\{z \,\middle|\, (\hat{B})_z \subseteq A^c\right\} \\ &= A^c \ominus \hat{B}\end{aligned}$$

where the second step follows from the definition of erosion, Eq. (9.2-1). This completes the proof.

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.

(b) From Eq. (9.3-3),

$$C \circ B = \bigcup \{(B)_z \,|\, (B)_z \subseteq C\}$$

and

$$D \circ B = \bigcup \{(B)_z \,|\, (B)_z \subseteq D\}.$$

Therefore, if $C \subseteq D$, it follows that $C \circ B \subseteq D \circ B$.
(c) From (a), $(A \circ B) \circ B \subseteq (A \circ B)$. From the definition of opening,

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

But, the only way that $(A \circ B) \circ B \subseteq (A \circ B)$ and $(A \circ B) \circ B \supseteq (A \circ B)$ can hold is if $(A \circ B) \circ B = (A \circ B)$. The next to last step in the preceding sequence follows from the fact that the closing of a set by another contains the original set [this is from Problem 9.16(a)].

Problem 9.16

(a) From Problem 9.14, $(A \bullet B)^c = A^c \circ \hat{B}$, and, from Problem 9.15(a), it follows that

$$(A \bullet B)^c = A^c \circ \hat{B} \subseteq A^c.$$

Taking the complement of both sides of this equation reverses the inclusion sign and we have that $A \subseteq (A \bullet B)$, as desired.

(b) From Problem 9.15(b), if $D^c \subseteq C^c$, then $D^c \circ \hat{B} \subseteq C^c \circ \hat{B}$ where we used D^c, C^c, and $\hat{B}$ instead of C, D, and B. From Problem 9.15, $(C \bullet B)^c = C^c \circ \hat{B}$ and $(D \bullet B)^c = D^c \circ \hat{B}$. Therefore, if $D^c \subseteq C^c$ then $(D \bullet B)^c \subseteq (C \bullet B)^c$. Taking complements reverses the inclusion, so we have that if $C \subseteq D$, then $(C \bullet B) \subseteq (D \bullet B)$, as desired.

(c) Using results from Problems 9.14 and 9.15,

$$\begin{aligned}
(A \bullet B) \bullet B &= \left\{(A \bullet B)^c \circ \hat{B}\right\}^c \\
&= \left\{(A^c \circ \hat{B}) \circ \hat{B}\right\}^c \\
&= \left\{(A^c \circ \hat{B})\right\}^c \\
&= \left\{(A \bullet B)^c\right\}^c \\
&= (A \bullet B).
\end{aligned}$$

where the third step follows from Problem 9.15(c) and the fourth step follows from Problem 9.14.

Problem 9.17

Figure P9.17 shows the solution. Although the images shown could be sketched by hand, they were done in MATLAB for clarity of presentation. The MATLAB code follows:

```
>> f = imread('FigProb0917.tif');
>> se = strel('dis', 11, 0); % Structuring element.
>> fa = imerode(f, se);
>> fb = imdilate(fa, se);
>> fc = imdilate(fb, se);
>> fd = imerode(fc, se);
```

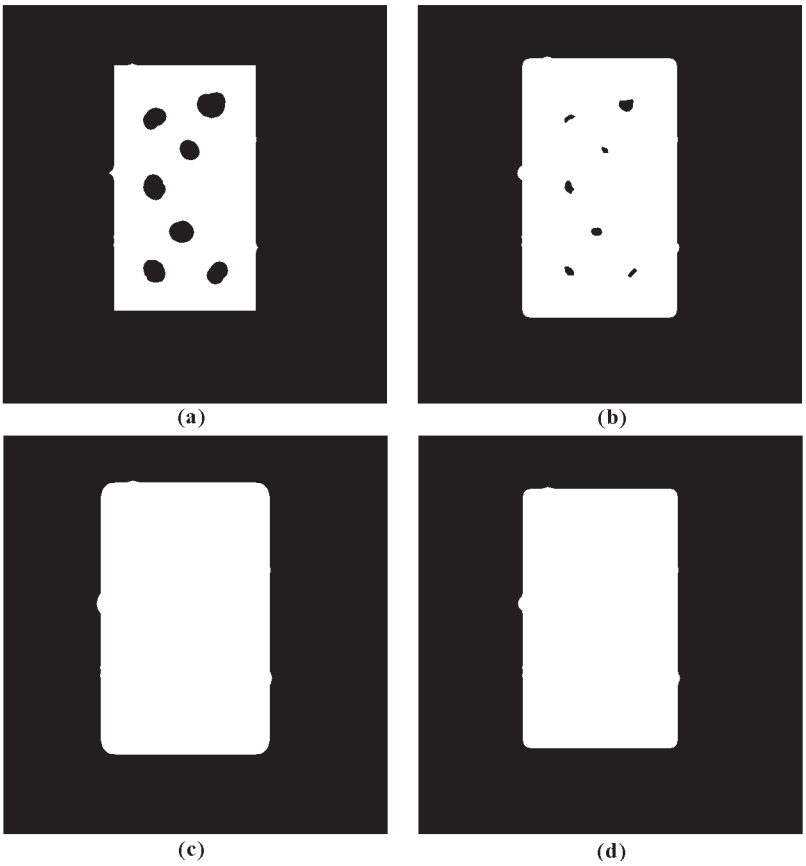


Figure P9.17

The size of the original image is 648×624 pixels. A disk structuring element of radius 11 was used. This structuring element was just large enough to encompass each noise element, as given in the problem statement. The images shown in Fig. P9.17 are: (a) erosion of the original, (b) dilation of the result, (c) another dilation, and finally (d) an erosion. The main points we are looking for from the student's answer are: The first erosion should take out all noise elements that do not touch the rectangle, should increase the size of the noise elements completely contained within the rectangle, and should decrease the size of the rectangle. If worked by hand, the student may or may not realize that some "imperfections" are left along the boundary of the object. We do not consider this an important issue because it is scale-dependent, and nothing is said in the problem statement about this. The first dilation should shrink the noise components that were increased in erosion, should increase the size of the rectangle,

and should round the corners. The next dilation should eliminate the internal noise components completely and further increase the size of the rectangle. The final erosion (bottom right) should then decrease the size of the rectangle. The rounded corners in the final answer are an important point that should be recognized by the student.

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 9.19

Select a one-pixel border around the structuring element (subimage of the T), and let the origin be located at the horizontal/vertical midpoint of this subimage. The result of applying the hit-or-miss transform would be a single point where the two T's were in perfect registration. The location of the point would be the same as the origin of the structuring element.

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:

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.

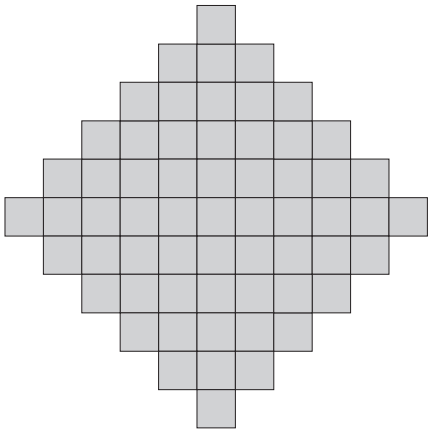


Figure P9.21

Problem 9.21

(a) The entire image would be filled with 1's.

(b) The background would be filled with 1's.

(c) See Fig. P9.21.

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).

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

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 connected components are labels with a value different from 1 or 0. The remaining

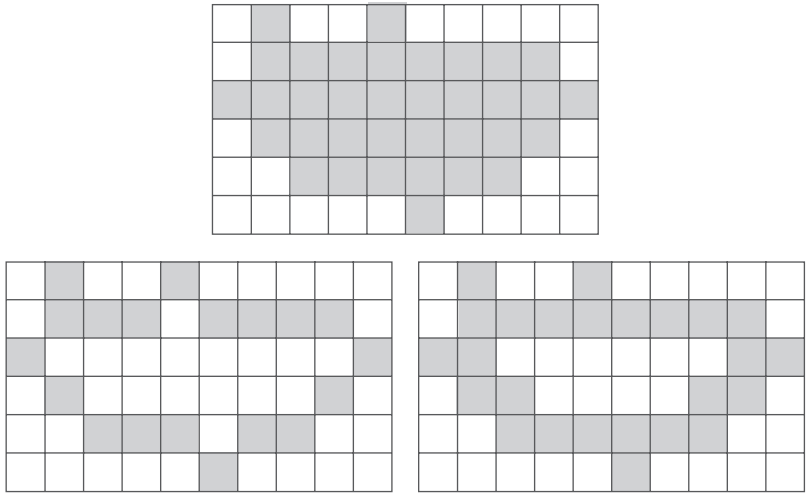


Figure P9.22

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.

(b) If the spheres are allowed to touch in arbitrary ways, a way must be found to separate them because they could create "pockets" of black points surrounded by all white or surrounded by all white and part of the boundary of the image. The simplest approach is to separate the spheres by preprocessing. One way to do this is to erode the white components of the image by one pass of a 3×3 structuring element, effectively creating a black border around the spheres, thus "separating" them. This approach works in this case because the objects are spherical, which implies that they have small areas of contact. To handle the case of spheres touching the border of the image, we simply set all border point to black. We then proceed to find all background points To do this, we pick a point on the boundary of the image (which we know is black due to preprocessing) and find all black points connected to it using a connected component algorithm (Section 9.5.3). These connected components are labels with a value different from 1 or 0. The remaining black points are interior to the spheres. We can fill all spheres with white by applying the hole filling algorithm in Section 9.5.2 until all such 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. Note that the erosion of white areas makes the black areas interior to the spheres grow, so the possibility exists that such an area near the border of a sphere could grow into the background. This issue introduces further complications that the student may not have the tools to solve yet. We recommend making the assumption that the interior black areas are small and near the center. Recognition of the potential problem by the student should be sufficient in the context of this problem.

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 9.25

From Section 9.5.9, the following expression is an image H in which all the holes in image f have been filled:

$$H = \left[R^D_{f^c}(F)\right]^c.$$

The only difference between this expression and the original image f is that the holes have been filled. Therefore, the intersection of f^c and H is an image containing only the (filled) holes. The complement of that result is an image containing only the holes.

$$f_{\text{holes}} = \left[f^c \cap \left[R^D_{f^c}(F)\right]^c\right]^c.$$

Figure P9.25 shows the holes extracted from Fig 9.31(a). Keep in mind that the holes in the original image are black. That's the reason for the complement in the preceding equation.

Figure P9.25

Problem 9.26

(a) If all the border points are 1, then F in Eq. (9.5-28) will be all 0s. The dilation of F by B will also be all 0s, and the intersection of this result with f^c will be all 0s also. Then H, which is the complement, will be all 1s.

(b) If all the points in the border of f are 1, that means that the interior of the entire image is a hole, so H in (a) will be the correct result. That is, the entire image should be filled with 1s. This algorithm is not capable of detecting holes within holes, so the result is as expected.

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 9.28

The dark image structures that are completely filled by morphological closing remain closed after the reconstruction.

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 9.30

$$\begin{aligned} R_G^D(F) &= D_G^{(k)}(F) \\ &= \left[E_{G^c}^{(1)}\left(E_{G^c}^{(k-1)}\left(F^c\right)\right)\right]^c \\ &= \left[E_{G^c}^{(k)}\left(F^c\right)\right]^c \\ &= \left[R_{G^c}^E\left(F^c\right)\right]^c \end{aligned}$$

where we used the result from Problem 9.29. The other duality property is proved in a similar manner.

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 9.32

$$\begin{aligned} O_R^{(n)}(F) &= R_F^D(F \ominus nB) \\ &= \left[R_{F^c}^E\left([F \ominus nB]^c\right)\right]^c \\ &= \left[R_{F^c}^E\left(F^c \oplus n\hat{B}\right)\right]^c \\ &= \left[R_{F^c}^E\left(F^c \oplus nB\right)\right]^c \\ &= \left[C_R^{(n)}\left(F^c\right)\right]^c \end{aligned}$$

where the second step follows from the duality of reconstruction by dilation (Problem 9.30), the third step follows from the result in Problem 9.31, the fourth step follows from the symmetry of B, and the last step follows from the definition of closing by reconstruction. The other duality property can be 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.

(b) We prove the second duality property:

$$\begin{aligned}(f \circ b)^c &= [(f \ominus b) \oplus b]^c \\ &= (f \ominus b)^c \ominus \hat{b} \\ &= \left(f^c \oplus \hat{b}\right) \ominus \hat{b} \\ &= f^c \bullet \hat{b}.\end{aligned}$$

The second and third steps follow from the duality property of dilation and erosion, and the last step follows from the definition of closing in Eq. (9.6-8). The other property in the problem statement is proved in a similar manner.

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

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

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

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

The other property is proved in a similar way.

(d) We prove the first property:

$$\begin{aligned}
R_g^D(f) &= D_g^{(k)}(f) \\
&= \left[E_{g^c}^{(1)}\left(E_{g^c}^{(k-1)}(f^c)\right)\right]^c \\
&= \left[E_{g^c}^{(k)}(f^c)\right]^c \\
&= \left[R_{g^c}^E(f^c)\right]^c.
\end{aligned}$$

The other property is proved in a similar manner.

(e) We prove the first property. Consider the case when $n = 2$

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

where the second and third lines follow from the duality property in Eq. (9.6-5). 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 $[f \ominus nb]^c = f^c \oplus n\hat{b}$. The other property is proved in a similar manner.

(f) We prove the first property:

$$\begin{aligned}O_R^{(n)}(f) &= R_f^D(f \ominus nb) \\ &= \left[R_{f^c}^E\left([f \ominus nb]^c\right)\right]^c \\ &= \left[R_{f^c}^E\left(f^c \oplus n\hat{b}\right)\right]^c \\ &= \left[R_{f^c}^E\left(f^c \oplus nb\right)\right]^c \\ &= \left[C_R^{(n)}\left(f^c n\right)\right]^c\end{aligned}$$

where the second step follows from the duality of reconstruction by dilation given in (d) , the third step follows from the result in (e), the fourth step follows from the symmetry of b, and the last step follows from the definition of closing by reconstruction. The other duality property can be proved in a similar manner.

Problem 9.34

The method is predicated on image openings and closings, having nothing to do with spatial arrangement. Because the blobs do not touch, there is no difference between the fundamental arrangement in Fig. 9.43 and the figure in this problem. The steps to the solution will be the same. The one thing to watch is that the SE used to remove the small blobs do not remove or severely attenuate large blobs and thus open a potential path to the boundary of the image larger than the diameter of the large blobs. In this case, disks of radius 30 and 60 (the same those used in Fig. 9.43) do the job properly. The solution images are shown in Fig. P9.34. The explanation is the same as Fig. 9.43. The MATLAB code used to generated the solution follows.

```
>> f = imread('FigP0934(blobs_in_circular_arrangement).tif');
>> figure, imshow(f)
>> % Remove small blobs.
>> fsm = imclose(f, strel('disk',30));
>> figure, imshow(fsm)
```

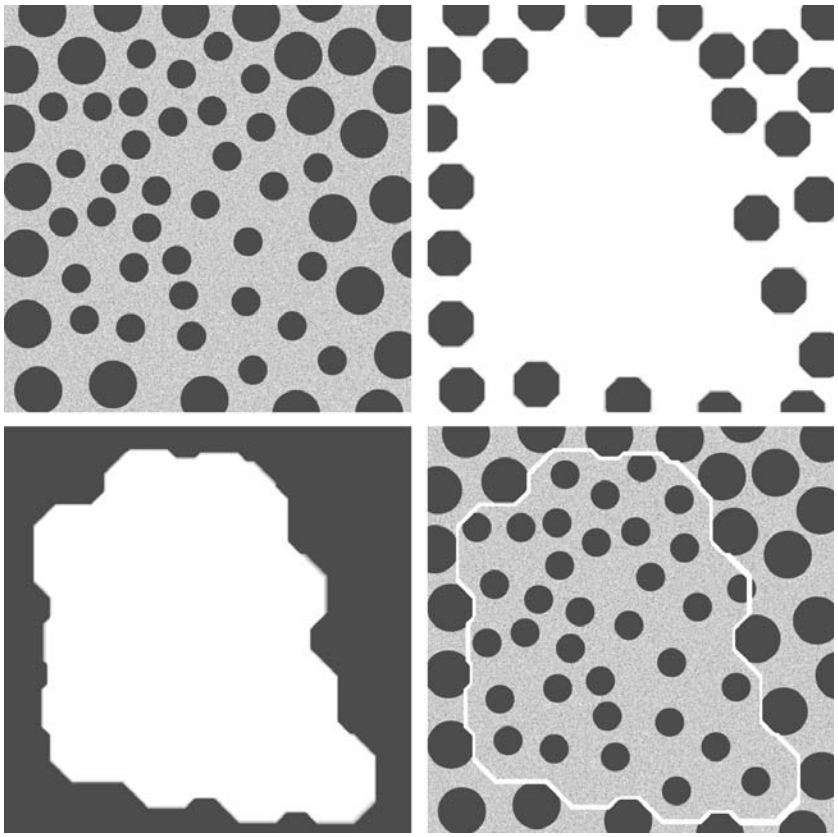


Figure P9.34

```
>> % Remove large blobs.
>> flrg = imopen(fsm, strel('disk',60));
>> figure, imshow(flrg)
>> % Use a morphological gradient to obtain the boundary
>> % between the regions.
>> se = ones(3);
>> grad = imsubtract(imdilate(flrg, se), imerode(flrg, se));
>> figure, imshow(grad)
>> % Superimpose the boundary on the original image.
>> idx = find(grad > 0);
>> final = f;
>> final(idx) = 255;
>> figure, imshow(final)
```

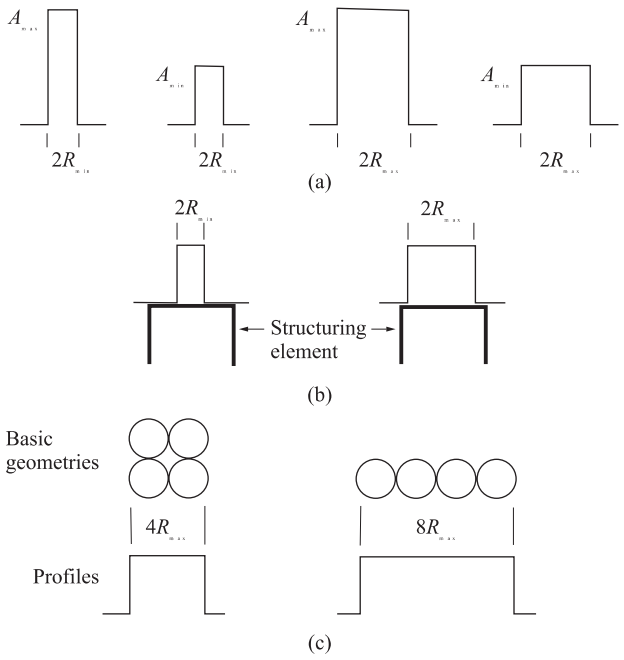


Figure P9.35

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.

(b) The basic solution is the same as in (a), but now we have to take into account the various possible overlapping geometries shown in Fig. P9.35(c). A structuring element like the one used in (a) but with radius slightly larger than $4R_{\max}$ will do the job. Note in (a) and (b) that other parts of the image would be affected by this approach. The bigger $R_{\max}$, the bigger the structuring element that would be needed and, consequently, the greater the effect on the image as a whole.

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.

(b) It is given that all particles are of the same size (this is done to simplify the problem; more general analysis requires tools from Chapter 11). Determine the area (number of pixels) of a single particle; denote the area by A. Eliminate from the image the particles that were merged with the border of the image. Apply the connected component algorithm. Count the number of pixels in each component. A component is then designated as a single particle if the number of pixels is less than or equal to $A+\varepsilon$, where ε is a small quantity added to account for variations in size due to noise.

(c) Subtract from the image single particles and the particles that have merged with the border, and the remaining particles are overlapping particles.

Problem 9.37

As given in the problem statement, interest lies in deviations from the round in the inner and outer boundaries of the washers. It is stated also that we can ignore errors due to digitizing and positioning. This means that the imaging system has enough resolution so that objectionable artifacts will not be introduced as a result of digitization. The mechanical accuracy similarly tells us that no appreciable errors will be introduced as a result of positioning. This is important if we want to do matching without having to register the images.

The first step in the solution is the specification of an illumination approach. Because we are interested in boundary defects, the method of choice is a backlighting system that will produce a binary image. We are assured from the problem statement that the illumination system has enough resolution so that we can ignore defects due to digitizing.

The next step is to specify a comparison scheme. The simplest way to match binary images is to AND one image with the complement of the other. Here, we match the input binary image with the complement of the golden image (this is more efficient than computing the complement of each input image and comparing it to the golden image). If the images are identical (and perfectly registered) the result of the AND operation will be all 0s. Otherwise, there will be 1s in the areas where the two images do not match. Note that this requires that the images be of the same size and be registered, thus the assumption of the mechanical accuracy given in the problem statement.

As noted, differences in the images will appear as regions of 1s in the AND image. These we group into regions (connected components) by using the algorithm given in Section 9.5.3. Once all connected components have been ex-

tracted, we can compare them against specified criteria for acceptance or rejection of a given washer. The simplest criterion is to set a limit on the number and size (number of pixels) of connected components. The most stringent criterion is 0 connected components. This means a perfect match. The next level for "relaxing" acceptance is one connected component of size 1, and so on. More sophisticated criteria might involve measures like the shape of connected components and the relative locations with respect to each other. These types of descriptors are studied in Chapter 11.

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 10.3

The key to solving this problem is to find all end points (see Section 9.5.8 for a definition of end point) of line segments in the image. Once all end points have been found, the D_8 distance between all pairs of such end points gives the lengths of the various gaps. We choose the smallest distance between end points of every pair of segments and any such distance less than or equal to

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

K satisfies the statement of the problem. This is a rudimentary solution, and numerous embellishments can be added to build intelligence into the process. For example, it is possible for end points of different, but closely adjacent, lines to be less than K pixels apart, and heuristic tests that attempt to sort out things like this are quite useful. Although the problem statement does not call for any such tests, they normally are needed in practice and it is worthwhile to bring this up in class if this problem is assigned as homework.

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.

(b) Via connectivity analysis.

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.

(b) The third row in Fig. P10.5 shows the angle images and their profiles.

All images were scaled for display, so only the shapes are of interest. In particular, the profile of the angle image of the roof edge has zero, negative, and positive values. The gray in the scaled angle image denotes zero, and the the light and dark areas correspond to positive and equally negative values, respectively. The black in all other images denotes 0 and the pure white is the maximum value (255 because these are 8-bit images). Grays are values in between.

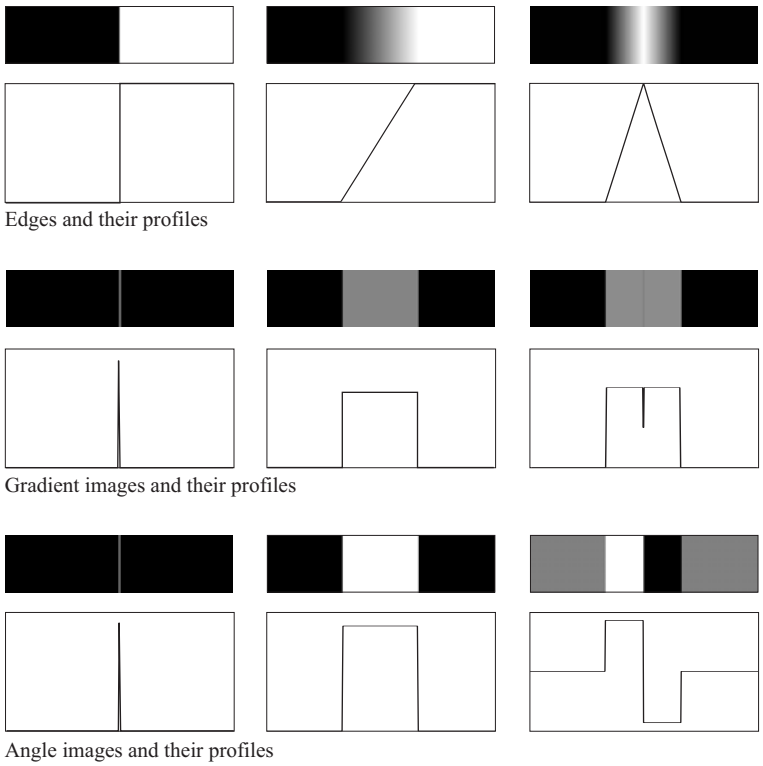


Figure P10.5

Problem 10.6

Figure P10.6 shows the solution.

Problem 10.7

Figure P10.7 shows the solution.

Problem 10.8

(a) Inspection of the Sobel masks shows that $g_x = 0$ for edges oriented vertically and $g_y = 0$ for edges oriented horizontally. Therefore, it follows in this case that, for vertical edges, $\nabla f = \sqrt{g_y^2} = |g_y|$, and similarly for horizontal edges.

(b) The same argument applies to the Prewitt masks.

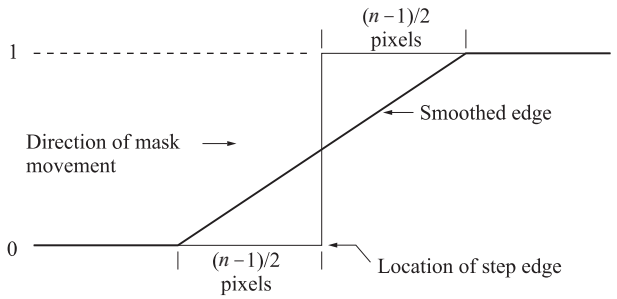


Figure P10.6

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.

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 10.10

Consider first the 3×3 smoothing mask mentioned in the problem statement, and a general 3×3 subimage area with intensities a through i, whose center point has value e, as Fig. P10.10 shows. Recall that value e is replaced by the response of the 3×3 mask when its center is at that location. Ignoring the 1/9

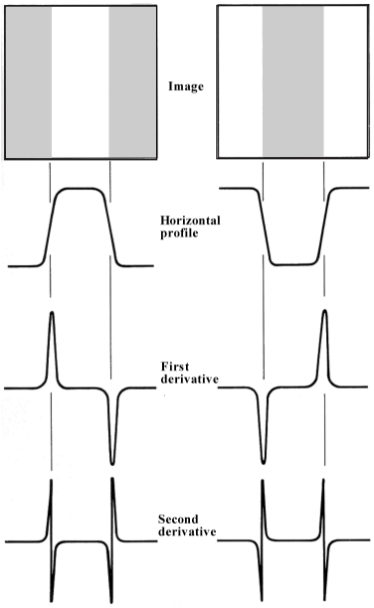


Figure P10.7

scale factor, the response of the mask when centered at that location is $(a+b+c+d+e+f+g+h+i)$.

The idea with the one-dimensional mask is the same: We replace the value of a pixel by the response of the mask when it is centered on that pixel. With this in mind, the mask [1 1 1] would yield the following responses when centered at the pixels with values *b*, *e*, and *h*, respectively: $(a+b+c)$, $(d+e+f)$, and $(g+h+i)$. Next, we pass the mask

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

through these results. When this mask is centered at the pixel with value e, its response will be $[(a+b+c)+(d+e+f)+(g+h+i)]$, which is the same as the result produced by the 3×3 smoothing mask.

Returning now to problem at hand, when the g_x Sobel mask is centered at the pixel with value e, its response is $g_x = (g+2h+i)-(a+2b+c)$. If we pass the one-dimensional differencing mask

$$\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

through the image, its response when its center is at the pixels with values d, e, and f, respectively, would be:$(g-a)$, $(h-b)$, and $(i-c)$. Next we apply the smoothing mask [1 2 1] to these results. When the mask is centered at the pixel with value e, its response would be $[(g-a)+2(h-b)+(i-c)]$ which is $[(g+2h+i)-(a+2b+c)]$. This is the same as the response of the 3×3 Sobel mask for g_x. The process to show equivalence for g_y is basically the same. Note, however, that the directions of the one-dimensional masks would be reversed in the sense that the differencing mask would be a column mask and the smoothing mask would be a row mask.

Problem 10.11

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

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

(b) The solution is as follows:

Compass gradient operators

E	NE	N	NW	W	SW	S	SE

Gradient direction

N	NW	W	SW	S	SE	E	NE

Problem 10.12

(a) The solution is shown in Fig. P10.12(a). The numbers in brackets are values of $[g_x, g_y]$.

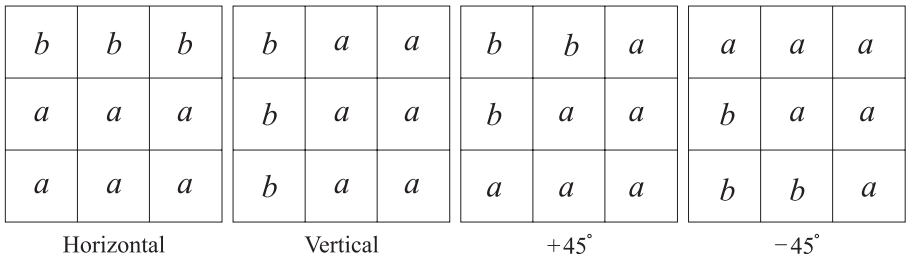


Figure P10.9

1	1	1
1	1	1
1	1	1

Smoothing mask (scaled by 1/9).

a	b	c
d	e	f
g	h	i

Subimage area under the mask at any one time.

Figure P10.10

(b) The solution is shown in Fig. P10.12(b). The angle was not computed for the trivial cases in which $g_x = g_y = 0$. The histogram follows directly from this table.

(c) The solution is shown in Fig. P10.12(c).

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

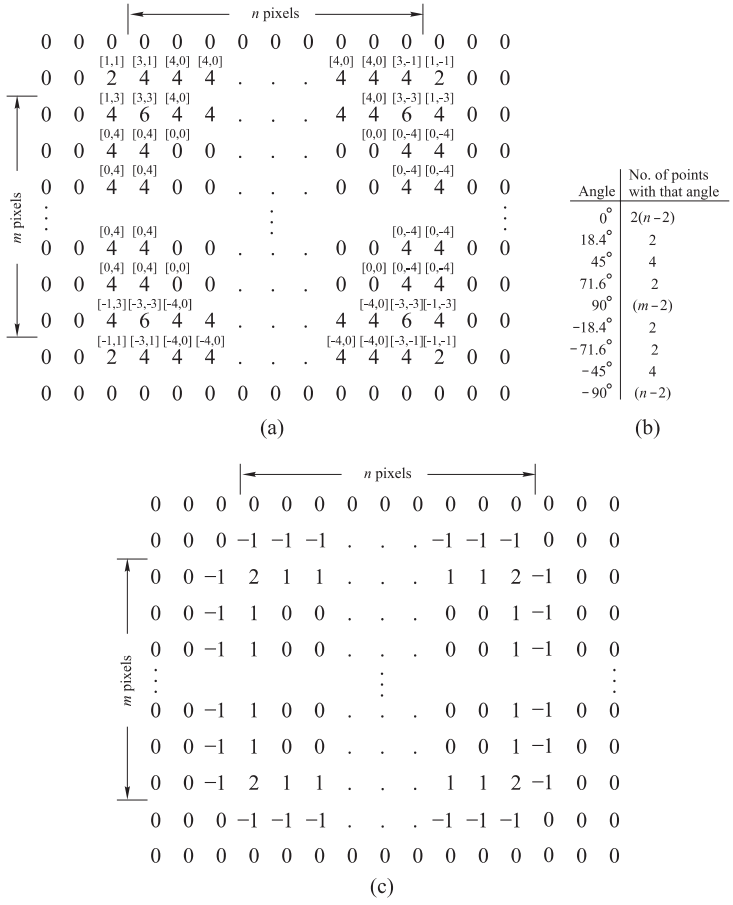
$$\partial \bar{f} / \partial x = \bar{f}(x+1, y) - \bar{f}(x, y)$$

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

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



Angle	No. of points with that angle
0°	$2(n-2)$
18.4°	2
45°	4
71.6°	2
90°	$(m-2)$
-18.4°	2
-71.6°	2
-45°	4
-90°	$(n-2)$

Figure P10.12

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

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

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

(b) From the preceding equation for $\partial \bar{f}/\partial x$, the maximum value this term can have is when each difference in the summation is maximum which, for an m-bit image, is $L = 2^m - 1$ (e.g., 255 for an 8-bit image). Thus,

$$\left[\partial \bar{f}/\partial x\right]_{\max} = \frac{1}{n^2}(nL) = \frac{L}{n}$$

and similarly for other derivative term. Therefore,

$$\bar{M}_{\max}(x,y) = \sqrt{\frac{L^2}{n^2} + \frac{L^2}{n^2}} = \frac{L}{n}\sqrt{2}.$$

For the original image, the maximum value that $\partial f/\partial x$ can have is L, and similarly for $\partial f/\partial y$. Therefore,

$$M_{\max}(x,y) = L\sqrt{2}$$

and the ratio is:

$$\frac{\bar{M}_{\max}(x,y)}{M_{\max}(x,y)} = \frac{1}{n}$$

which shows that the edge strength of a smoothed image compared to the original image is inversely proportional to the size of the smoothing mask.

Problem 10.14

(a) We proceeed as follows

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

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

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

(b) Convolving an image $f(x,y)$ with $\nabla^2 G(x,y)$ in the spatial domain is the same as multiplying the Fourier transforms of these two functions in the frequency domain. From Chapter 4, we know that the average value of a function in the frequency domain is proportional to the value of its transform at the origin of the frequency domain. We showed in (a) that the average value of $\nabla^2 G(x,y)$ is zero, so its Fourier transform at the origin must be zero. Therefore, the product of the Fourier transforms of $f(x,y)$ and $\nabla^2 G(x,y)$ evaluated at the origin is zero, from

which it follows that the convolution of these two functions has a zero average value.

(c) The answer is yes. From Problem 3.16 we know that convolving a mask (like the Laplacian masks in the problem statement) whose coefficients sum to zero with any image gives a result whose average value is zero. The result is digital, so if its average is zero, this means that all the elements sum to zero. Thus, convolving this result with any image also gives a result whose average value is zero.

Problem 10.15

(a) Let $g_L(x,y)$ denote the LoG image, as in Fig. 10.22(b). This image has both positive and negative values, and we know that the zero crossings are such that at least a pair of opposing locations in a 3×3 neighborhood differ in sign. Consider a *binary* image $g_P(x,y)$ formed by letting $g_P(x,y)=1$ if $g_L(x,y)>0$ and $g_P(x,y)=0$ otherwise. Figure P10.15(a) shows $g_P(x,y)$ for the Laplacian image in Fig. 10.22(b). The important thing about this image is that it consists of connected components. Futhermore (for the 3×3 mask used to detect zero crossings and a threshold of 0) each point on the boundary of these connected components is either a points of transition between positive and negative (i.e., it is the center point of the mask) or it is adjacent to the point of transition. In other words, all zero-crossing points are adjacent to the boundaris of the connected components just described. But, boundaries of connected components form a closed path (a path exists between any two points of a connected component, and the points forming the boundary of a connected components are part of the connected component, so the boundary of a connected a component is a closed path). Figure P10.15(b) shows the closed paths for the problem in question. Compare these closed paths and the binary regions in Fig. P10.15(a).

(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 function 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.

Figure P10.15

Problem 10.16

$$\begin{aligned} G'(r) &= \frac{-r}{\sigma^2} e^{-\frac{r^2}{2\sigma^2}} \\ \nabla^2 G(x,y) &= G''(r) \\ &= \left[\frac{r^2}{\sigma^4} - \frac{1}{\sigma^2}\right] e^{-\frac{r^2}{2\sigma^2}} \\ &= \left[\frac{r^2-\sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}} \end{aligned}$$

Letting $r^2 = x^2 + y^2$ we obtain

$$\nabla^2 G(x,y) = \frac{x^2+y^2-\sigma^2}{\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Comparing this derivation with the derivation of Eq. (10.2-15) we see that the reason for the difference is in the derivatives of r as opposed to the derivatives with respect to x and y. Another way of explaining the difference is that working with r is basically working with a radial slice of a 2-D function, which is the same as working with a 1-D Gaussian. The 2-D nature of the problem is thus lost. As it turns out, the difference of only 2 in the numerator multiplying sigma makes a significant difference in the filtered result.

Problem 10.17

(a) From Eq. (10.2-26), the DoG function is zero when

$$\frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} = \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}.$$

Taking the natural log of both sides yields,

$$\ln\left[\frac{1}{2\pi\sigma_1^2}\right]-\frac{x^2+y^2}{2\sigma_1^2}=\ln\left[\frac{1}{2\pi\sigma_2^2}\right]-\frac{x^2+y^2}{2\sigma_2^2}.$$

Combining terms,

$$\begin{aligned}\left(x^2+y^2\right)\left[\frac{1}{2\sigma_1^2}-\frac{1}{2\sigma_2^2}\right]&=\ln\left[\frac{1}{2\pi\sigma_1^2}\right]-\ln\left[\frac{1}{2\pi\sigma_2^2}\right]\\&=\ln\left[\frac{\sigma_1^2}{\sigma_2^2}\right].\end{aligned}$$

The LoG function [Eq. (10.2-23)] is zero when $x^2+y^2=2\sigma^2$. Then, from the preceding equation,

$$\sigma^2\left[\frac{1}{\sigma_2^2}-\frac{1}{\sigma_1^2}\right]=\ln\left[\frac{\sigma_1^2}{\sigma_2^2}\right].$$

Finally, solving for σ^2,

$$\sigma^2=\frac{\sigma_1^2\sigma_2^2}{\sigma_1^2-\sigma_2^2}\ln\left[\frac{\sigma_1^2}{\sigma_2^2}\right]$$

which agrees with Eq. (10.2-27).

(b) To obtain an expression in terms of k, we let $\sigma_1=k\sigma_2$ in the preceding equation:

$$\begin{aligned}\sigma^2&=\frac{k^2\sigma_2^4}{k^2\sigma_2^2-\sigma_2^2}\ln\left[\frac{k^2\sigma_2^2}{\sigma_2^2}\right]\\&=\frac{k^2}{k^2-1}\sigma_2^2\ln(k^2)\end{aligned}$$

with $k>1$.

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

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

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,

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

(b) Direct implementation of 2-D convolution requires n^2 multiplications at each location of $f(x,y)$, so the total number of multiplications is $n^2\times M\times N$. 1-D convolution requires n multiplications at each location of every row in the image, for a total of $n\times M\times N$ for the pass along the rows. Then, $n\times M\times N$ multiplications are required for the pass along the columns, for a total of $2nMN$ multiplications. The computational advantage, A, is then

$$A=\frac{n^2MN}{2nMN}=\frac{n}{2}$$

which is independent of image size. For example, if $n=25$, $A=12.5$, so it takes 12.5 more multiplications to implement 2-D convolution directly than it does to implement the procedure just outlined that uses 1-D convolutions.

Problem 10.19

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

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

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:

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

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 10.20

(a) The solution is basically the same as in Problem 10.19, but using the 1-D masks in Fig. 10.13 to implement Eqs. (10.2-12) and (10.2-13).

(b) Smoothing with a Gaussian filter (step 1 of the algorithm) requires convolving an $M \times N$ image with an $n \times n$ filter mask will require $n^2 \times M \times N$ multiplications as explained in the solution of Problem 10.19. Then two convolutions

with a 3×3 horizontal and vertical edge detector requires $2(9\times M\times N)=18MN$ multiplications, for a total of $(n^2+18)\times M\times N$ multiplications. The magnitude image requires two multiplications, one to compute $\partial f/\partial x$ times itself and $\partial f/\partial y$ times itself, for a total of $2MN$ multiplications. The total for the 2-D approach is then $(n^2+20)MN$ multiplications. The 1-D convolution for step 1 requires $2nMN$ multiplications (see the solution to Problem, 10.19). The y component of the gradient can be implemented using the difference $\partial f/\partial y = f(x,y+1)-f(x,y)$, which, as stated in Section 10.2.5, can be implemented by convolving the smoothed image with a 1-D mask with coefficients $[-1\ 1]$. This will require $2MN$ multiplications. Similarly, the other partial derivative can be implemented with a vertical mask with the same coefficients, for a total of $4NM$ multiplications. The squares are the same as for the 2-D implementation: $2MN$ multiplications. The total for the 1-D implementation is then $(2n+6)MN$. The ratio of 2-D to 1-D multiplications is

$$A=\frac{(n^2+20)MN}{(2n+6)MN}=\frac{n^2+20}{2n+6}$$

which is independent of image size. When, for example, $n=25$, then $A=11.52$, so it would take approximately 11 times more multiplications to implement the first part of the Canny algorithm with 2-D convolutions.

Problem 10.21

Parts (a) through (e) are shown in rows 2 through 6 of Fig. P10.21.

Problem 10.22

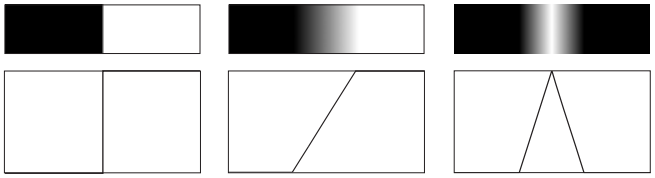
(a) Express $x\cos\theta+y\sin\theta=\rho$ in the form $y=-(\cot\theta)x+\rho/\sin\theta$. Equating terms with the slope-intercept form, $y=ax+b$, gives $a=-(\cot\theta)$ and $b=\rho/\sin\theta$. This gives $\theta=\cot^{-1}(a)$ and $\rho=b\sin\theta$. Once obtained from a and b of a given line, the parameters θ and ρ completely specify the normal representation of that line.

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

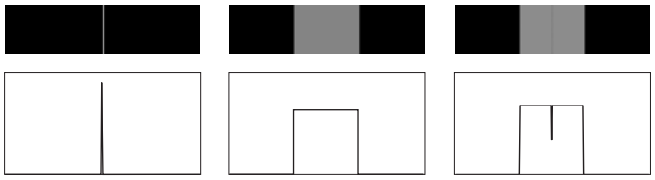
Problem 10.23

(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.

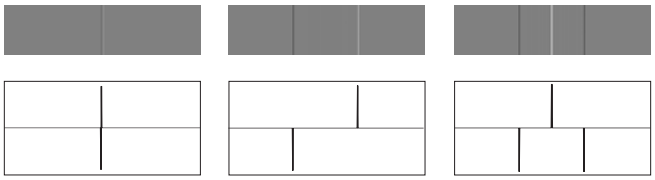
(b) Only the origin $(0,0)$ would yield this result.



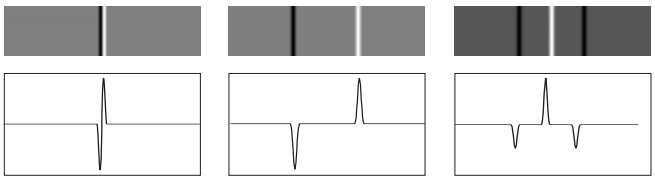
Edges and their profiles



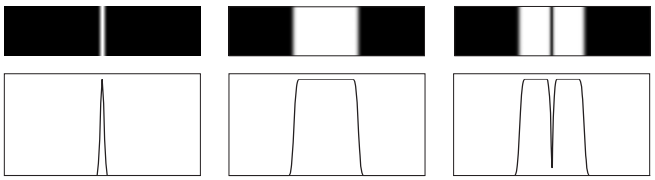
Gradient images and their profiles



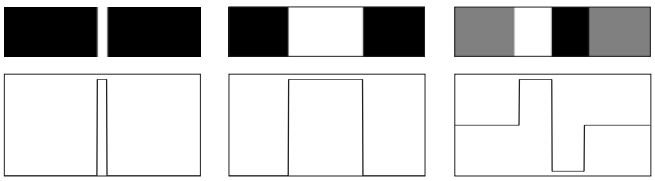
Laplacian images and their profiles



Images from Steps 1 and 2 of the Marr-Hildreth algorithm and their profiles



Images from Steps 1 and 2 of the Canny algorithm and their profiles



Canny angle images and their profiles

Figure P10.21

(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 10.24

Subdividing the θ-axis into K increments gives, for every point (x_k, y_k), K values of ρ corresponding to the K possible values of θ. With n image points, this method involves nK computations. Thus the procedure is *linear* in n.

Problem 10.25

This problem is a natural for the Hough transform, which is set up as follows: The θ axis is divided into six subdivisions, corresponding to the six specified directions and their error bands. For example (because the angle directions specified in the problem statement are with respect to the horizontal) the first band for angle θ extends from $-30°$ to $-20°$, corresponding to the $-25°$ direction and its $\pm 5°$ band. The ρ axis extends from $\rho = -\sqrt{D}$ to $\rho = +\sqrt{D}$, where D is the largest distance between opposite corners of the image, properly calibrated to fit the particular imaging set up used. The subdivisions in the ρ axis are chosen finely enough to resolve the minimum expected distance between tracks that may be parallel, but have different origins, thus satisfying the last condition of the problem statement.

Set up in this way, the Hough transform can be used as a "filter" to categorize all points in a given image into groups of points in the six specified directions. Each group is then processed further to determine if its points satisfy the criteria for a valid track: (1) each group must have at least 100 points; and (2) it cannot have more than three gaps, each of which cannot be more than 10 pixels long (see Problem 10.3 regarding the estimation of gaps of a given length).

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 $0 \leq i \leq L-1$, 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

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

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.

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.

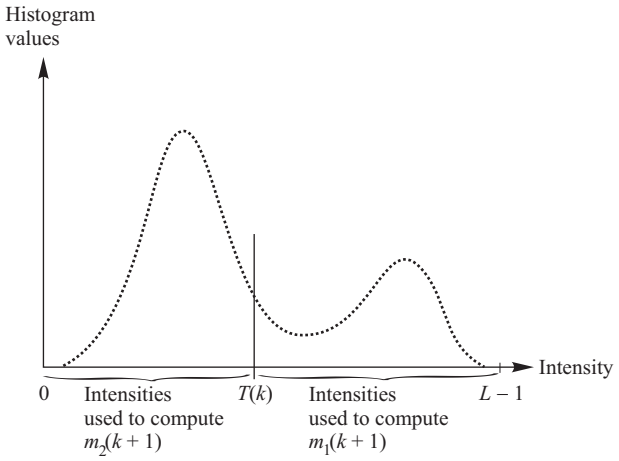


Figure P10.27

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.

Problem 10.28

Consider an image whose intensities (of both background and objects) are greater than $L/2$ and less than $L-1$, where L is the number of intensity levels and $L-1$ is the maximum intensity level (recall that we work with the intensity scale in the range $[0, L-1]$. Suppose that we choose the initial threshold as $T = 0$. Then

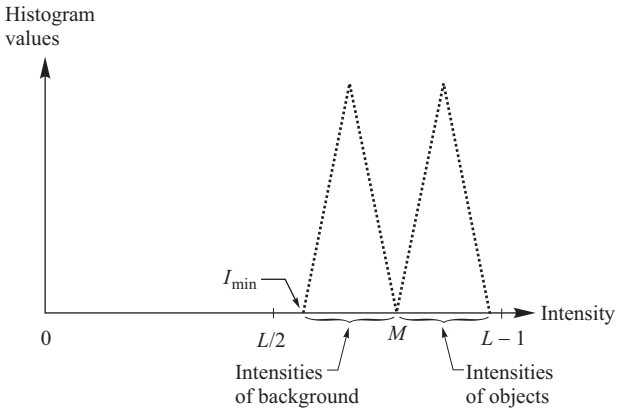


Figure P10.29

all pixels will be assigned to class 1 and none to class 2. So m_1 will be the mean value, M, of the image and m_2 will be zero because no pixels will be assigned to class 2. T will be the sum of the two means divided by 2, or $M/2$. However, the image mean, M, is between $L/2$ and $L-1$ because there are no pixels with values below $L/2$. So, the new threshold, $T = M/2$ will be below $L/2$. Because there are no pixels with values below $L/2$, m_2 will be zero again, and m_1 will be the mean value of the intensities and T will again be equal to $M/2$. So, the algorithm will terminate with the wrong value of threshold. The same concept is restated more formally and used as a counter-example in the solution of Problem 10.29.

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.

(b) The prove is similar to (a) because the modes are identical. When the algorithm arrives at the point between the two means, further motion of the threshold toward one or the other mean will cause one group to pick up and the other to lose the same "mass". Thus, their sum will be the same and the algorithm will stop.

Problem 10.31

(a) $A_1 = A_2$ and $\sigma_1 = \sigma_2 = \sigma$, which makes the two modes identical [this is the same as Problem 10.30(b)].

(b) You know from Problem 10.30 that if the modes were symmetric and identical, the algorithm would converge to the point midway bewteen the means. In the present problem, the modes are symmtric but they may not be identical. Then, all we can say is that the algorithm will converge to a point somewhere between the means (from Section 10.3.2 we know that the algorithm must start at some point between the minimum and maximum image intensities). So, if both A_1 and A_2 are greater than 0, we are assured that the algorithm will converge to a point somewhere between m_1 and m_2 .

(c) $\sigma_1 >> \sigma_2$. This will ”pull” the threshold toward m_1 during iteration.

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 10.33

From, Eq. (10.3-15),

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

As stated in the book, the measure σ_B^2 (or η) takes a minimum value of 0 when $m_1 = m_2$ which implies that there is only one class of pixels (the image is constant), in which case $P_1(k) = 1$ or $P_1(k) = 0$ for some value of k. In any other case, $P_1(k)[1-P_1(k)] > 0$ or, equivalently, for $0 < P_1(k) < 1$, so the measure takes on positive and bounded values, and a finite maximum is guaranteed to exist for some value of k if the image is not constant.

Problem 10.34

From the definition in Eq. (10.3-12),

$$\eta = \frac{\sigma_B^2}{\sigma_G^2}$$

where

$$\begin{aligned}\sigma_B^2 &= P_1(m_1-m_G)^2 + P_2(m_2-m_G)^2 \\ &= P_1P_2(m_1-m_2)^2\end{aligned}$$

and

$$\sigma_G^2 = \sum_{i=0}^{L-1}(i-m_G)^2 p_i.$$

As in the text, we have omitted k in σ_B^2 for the sake of notational clarity, but the assumption is that $0 \le k \le L-1$. As explained in the book, the minimum value of η is zero, and it occurs when the image is constant. It remains to be shown that the maximum value is 1, and that it occurs for two-valued images with values 0 and $L-1$.

From the second line of the expression for σ_B^2 we see that the maximum occurs when the quantity $(m_1-m_2)^2$ is maximum because P_1 and P_2 are positive. The intensity scale extends from 0 to $L-1$, so the maximum difference between means occurs when $m_1 = 0$ and $m_2 = L-1$. But the only way this can happen is if the variance of the two classes of pixels is zero, which implies that the image only has these two values, thus proving the assertion that the maximum occurs only when the image is two-valued with intensity values 0 and $L-1$. It remains to be shown that the maximum possible value of η is 1.

When $m_1 = 0$ and $m_2 = L-1$,

$$\begin{aligned}\sigma_B^2 &= P_1P_2(m_1-m_2)^2 \\ &= P_1P_2(L-1)^2.\end{aligned}$$

For an image with values 0 and $L-1$,

$$\begin{aligned}\sigma_G^2 &= \sum_{i=0}^{L-1}(i-m_G)^2p_i \\ &= \sum_{i=0}^{k}(i-m_G)^2P_1 + \sum_{i=k+1}^{L-1}(i-m_G)^2P_2 \\ &= (0-m_G)^2P_1 + (L-1-m_G)^2P_2\end{aligned}$$

From Eq. (10.3-10)

$$\begin{aligned}m_G &= P_1m_1 + P_2m_2 \\ &= P_2(L-1).\end{aligned}$$

So,

$$\begin{aligned}\sigma_G^2 &= (0-m_G)^2P_1 + (L-1-m_G)^2P_2 \\ &= P_2^2(L-1)^2P_1 + (L-1)^2(1-P_2)^2P_2 \\ &= (L-1)^2(P_2^2P_1 + P_1^2P_2) \\ &= (L-1)^2P_2P_1(P_2+P_1) \\ &= P_2P_1(L-1)^2\end{aligned}$$

where we used the fact that $P_1 + P_2 = 1$. We see from the preceding results for σ_B^2 and σ_G^2 that $\sigma_B^2/\sigma_G^2 = 1$ when the image is two-valued with values 0 and $L-1$. This completes the prrof.

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

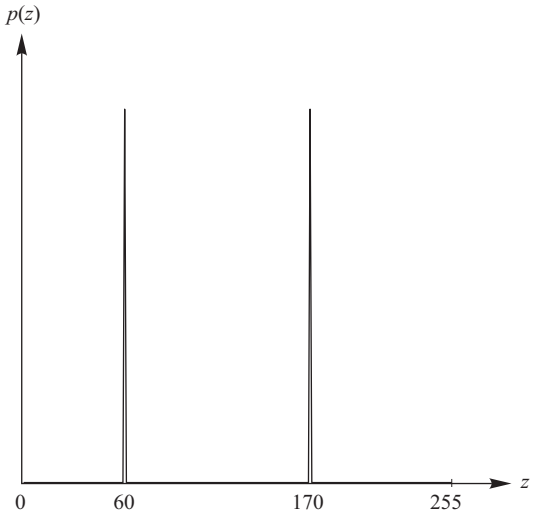


Figure P1036

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 10.36

With reference to Fig. P10.36, the key is to note that the means are more than 10 standard deviations apart, so the valley between the two modes of the histogram of the image is wide and deep. The simple global thresholding algorithm of Section 10.3.2 or Otsu's algorithm will give a performance that easily exceeds the 90% specification. In fact, the solution will give close to 100% accuracy.

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.

(b) The rightmost column would be 0 and all others would be 1, for the reason stated in (a).

(c) As in (a), all the pixels in the first column of the segmented image are labeled 0. Consider all other pixels in the first row of the segmented image and keep in mind that $n = 2$ and $b = 1$. Because the ramp in the original image increases in intensity to the right, the value of a pixel at location k in the image is always greater than the (moving) average of that pixel and its predecessor. Thus, all points along the first row, with the exception of the first are labeled 1. When the scan reaches the end of the first row and reverses direction as it starts the second row, the reverse condition will exist and all pixels along the second row of the segmented image will be labeled 0. The conditions in the third row are the same as in the first row, so the net effect is that the segmented image will consist of all 0s in the first column, and all 0s in alternating rows, starting with the second. All other pixels will be 1.

(d) The first pixel in the first row of the segmented pixel will be 0 and the rest will be 1s for the reason stated in (a). However, in subsequent rows the conditions change. Consider the change in direction between the end of the first row and the reverse scan of the second row. Because n is now much greater than 2, it will take “a while" before the elements in that row of the original image will become smaller than then running average, which reached is highest value at the end of the first row. The result will be K white pixels before the values in the original image drop below the running average and the pixels in the segmented image become black. This condition is reversed when the turn is made from the second to the third row. K black pixels will now be present in the first row before the values in the original image increase past the running average. Once they do, the corresponding locations in the segmented region will be labeled white through the end of that row. The segmented image will thus look as follows: The first pixel in the first row will be black and all others will be white. The next row will have a K black pixels on the left and K white pixels on the right, with all other pixels being black. The left and right of the next row will be the same, but the pixels in between will be white. These two alternating row patterns are applicable to the rest of the rows in the segmented image.

Problem 10.38

The means are at 60 and 170, and the standard deviation of the noise is 10 intensity levels. A range of $\pm 3\sigma$ about 60 gives the range [30, 90] and a similar range about 170 gives [140 200] so significant separation exists between the two intensity populations. So choosing 170 as the seed value from which to grow the objects is quite adequate. One approach is to grow regions by appending to a seed any pixel that is 8-connected to any pixel previously appended to that

seed, and whose intensity is $170 \pm 3\sigma$.

Problem 10.39

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

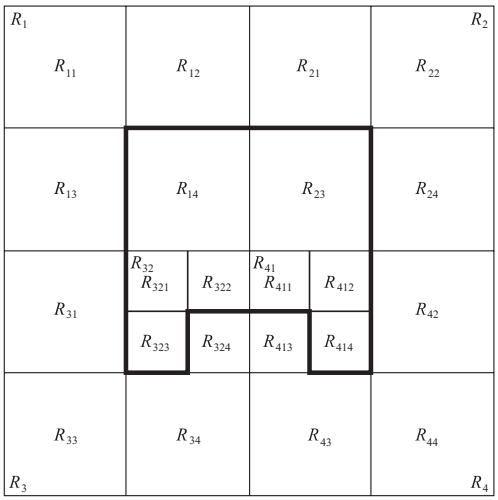
Problem 10.40

To obtain the sparse outer region we simply form the AND of the mask with the outer region. Because the mask completely separates the inner region from the background, its complement will produce two disjoint regions. We find these two regions by simply by using a connected-components algorithm (see Section 9.5.3). We then identify the region containing the background by determining which of the two regions contains at least one point on the boundary of the image (say, any one of the four corner points). We then AND an image containing only this region with the original to obtain the background. ANDing the other region with the image isolates only the inner region.

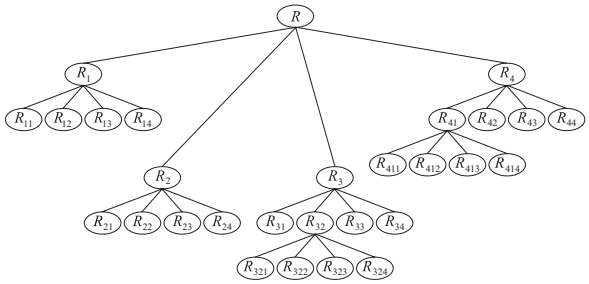
Problem 10.41

(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 never replaced, it follows that the elements in $C_n(M_i)$ are never replaced either. In addition, we see that $C_{n-1}(M_i) \subseteq C_n(M_i)$.

(b) This part of the problem is answered by the same argument as in (a). Because (1) n always increases; (2) the elements of neither $C_n(M_i)$ nor $T[n]$ are ever replaced; and (3) $T[n-1] \subseteq T[n]$ and $C_{n-1}(M_i) \subseteq C_n(M_i)$, it follows that the number of elements of both $C_n(M_i)$ and $T[n]$ either increases or remains the same.



(a)



(b)

Figure P10.39

Problem 10.42

Using the terminology of the watershed algorithm, a break in a boundary between two catchment basins would cause water between the two basins to merge. However, the heart of the algorithm is to build a dam higher than the highest intensity level in the image any time a break in such boundaries occurs. Because the entire topography is enclosed by such a dam, dams are built any time there is a break that causes water to merge between two regions, and segmentation boundaries are precisely the tops of the dams, so it follows that the watershed algorithm always produces closed boundaries.

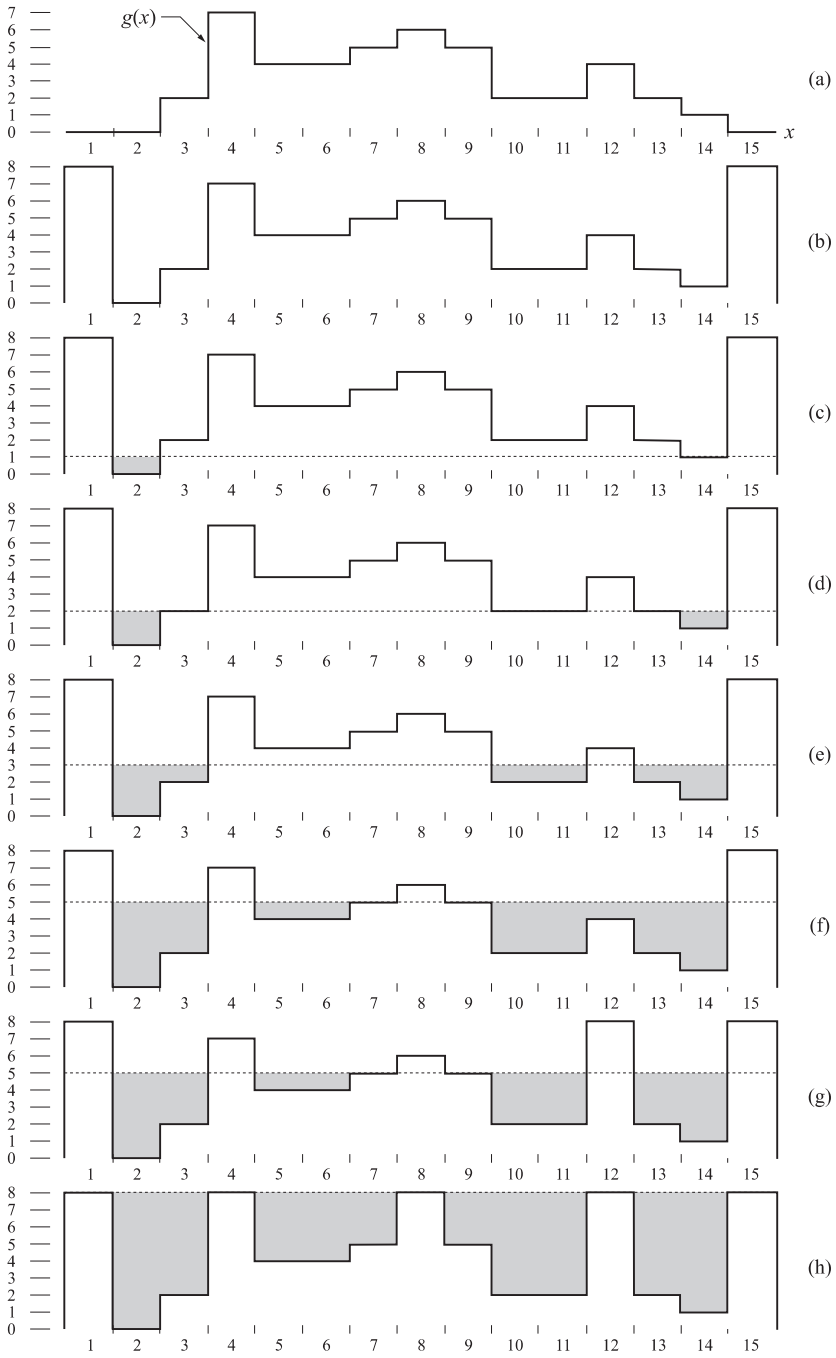


Figure P10.43

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

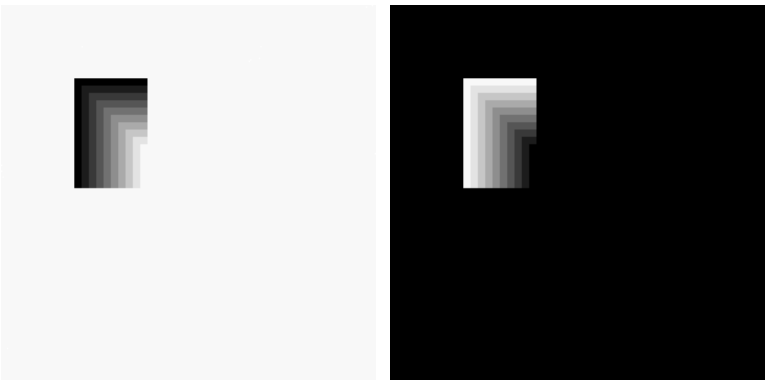


Figure P10.44

the water line. A final post-processing step would remove the outer dam walls to yield the inner edges of interest.

Problem 10.44

With reference to Eq. (10.6-4) , we see that comparing the negative ADI against a positive, rather than a negative, threshold would yield the image negative of the positive ADI [see Eq. (10.6-3)]. The result is shown in the left of Fig. P10.44. The image on the right is the positive ADI from Fig. 10.59(b). We have included it here for convenience in making the comparison.

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 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.

(b) True. The positive ADI is stationary and equal to the dimensions of the moving object because the differences between the reference and the moving object never exceed the threshold in areas that are background in the reference image [assuming, as in Eq. (10.6-3), that the background has lower intensity values than the object].

(c) True. From Eq. (10.6-4), we see that differences between the background and the object always will be negative [assuming, as in Eq. (10.6-4), that the intensity levels in the object exceed the value of the background]. Assuming also that the differences are more negative than the threshold, we see for the same reason as in (a) that all new background areas occupied by the moving object will have nonzero counts, thus increasing the dimension of the nonzero entries in the negative ADI (keep in mind that the values in this image are counts).

Problem 10.46

Consider first the fact that motion in the x-direction is zero. When all components of an image are stationary, $g_x(t, a_1)$ is a constant, and its Fourier transform yields an impulse at the origin. Therefore, Fig. 10.63 would now consists of a single impulse at the origin. The other two peaks shown in the figure would no longer be present. To handle the motion in the positive y-direction and its change in the opposite direction, recall that the Fourier transform is a linear process, so we can use superposition to obtain a solution. The first part of motion is in the positive y-direction at 1 pixel/frame. This is the same as in Example 10.28, so the peaks corresponding to this part of the motion are the same as the ones shown in Fig. 10.64. The reversal of motion is instantaneous, so the 33rd frame would show the object traveling in exactly the opposite direction. To handle this, we simply change a_2 to $-a_2$ in Eq. (10.6-7). Based on the discussion in connection with Eq. (10.6-5), all this change would do is produce peaks at frequencies $u = -a_2 V_2$ and $K + a_2 V_2$. From Example 10.28 we know that the value of a_2 is 4. From the problem statement, we know that $V_2 = 1$ and $K = 32$. Thus, we have two new peaks added to Fig. 10.64: one at $u = -4$ and the other at $u = 36$. As noted above, the original peaks correspond to the motion in the positive y-direction given in the problem statement, which is the same as in Example 10.28. Note that the frame count was restarted from 0 to 31 with the change in direction.

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 deriva-

tive 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 10.48

Set up a small target of known, uniform reflectivity, R, near a corner of the viewing area, at coordinates (x_0, y_0). The small target will become part of the images captured by the system and can thus be monitored by the software. With reference to the image model discussed in Section 2.3.4, you can determine $A(0)$ by imaging the target when the bulb is new ($t = 0$) because then $f(x0, y0) = iR = A(0)R$ and f and R will be known in the region of the imaged target. The exponential term is known, so can compute it once and store it. The image at any time t will be given by

$$f(x,y) = i(x,y)r(x,y) = \left(A(t) - t^2 e^{-\left[(x-M/2)^2+(y-N/2)^2\right]}\right) r(x,y).$$

It is given that Otsu's algorithm works well when $f = A(0)r(x,y)$ (i.e., when the lamp is new (and $t = 0$) so, if we can force the time and spatially varying component of the previous equation to have the value $A(0)$, then segmentation of the corrected image will be as desired. Consider the portion of the image at containing the known reflectivity target at any time t_n

$$f_n(x_0,y_0) = \left(A(t_n) - t_n^2 e^{-\left[(x_0-M/2)^2+(y_0-N/2)^2\right]}\right) R.$$

(Note: in practice we would use the average value of f around (x_0, y_0) instead of $f(x_0, y_0)$ to reduce the effects of noise.) The exponential term is known, and so are R, $f_n(x_0, y_0)$, and t_n. So we can solve for $A(t_n)$. Then we can correct the input image as follows:

$$\begin{aligned} g(x,y) &= \frac{A(0)f(x,y)}{A(t_n) - t_n^2 e^{-\left[(x_0-M/2)^2+(y_0-N/2)^2\right]}} \\ &= A(0)\left[\frac{A(t) - t^2 e^{-\left[(x-M/2)^2+(y-N/2)^2\right]}}{A(t_n) - t_n^2 e^{-\left[(x_0-M/2)^2+(y_0-N/2)^2\right]}}\right] r(x,y). \end{aligned}$$

At $t = t_n$, $g(x,y) = A(0)r(x,y)$, and we know that Otsu's algorithm will perform satisfactorily. Because we don't know how $A(t)$ behaves as a function of time and

the lamps are experimental, the safest course of action is to perform the preceding correction as many times as possible during the time the lamps are operating. This requires frequent computation of $A(t_n)$ using information obtained from the reflective target, as we did above. If continuous correction presents a computational burden that is too great, or too expensive to implement, then performing experiments with various lamps can help in determining a longer acceptable period between corrections. It is likely that longer periods between corrections will be acceptable because the change in physical devices generally is much slower than the speed of imaging devices.

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, $(1.02 \times 10^6$ pixels/sec) $\times$ K sec ≤ 1 pixel. So, $K \leq 9.8 \times 10^{-7}$ sec.

(b) The frame rate must be fast enough to capture at least two images of the bullet in successive frames so that the speed can be computed. If the frame rate is set so that the bullet cannot travel a distance longer (between successive frames) than one half the width of the image, then we have the cases shown in Fig. P10.49. In cases A and E we get two shots of the entire bullet in frames t_2 and t_3 and t_1 and t_2, respectively. In the other cases we get partial bullets. Although these cases could be handled with some processing (e.g., by determining size, leading and trailing edges, and so forth) it is possible to guarantee that at least two complete shots of every bullet will be available by setting the frame rate so that a bullet cannot travel more than one half the width of the frame, minus the length of the bullet. The length of the bullet in pixels is (2.5 cm)/(0.098 cm/pixel) $\approx$ 26 pixels. One half of the image frame is 128 pixels, so the maximum travel distance allowed is 102 pixels. Because the bullet travels at a maximum speed of 1.02×10^6 pixels/sec, the minimum frame rate is $1.02 \times 10^6/102 = 10^4$ frames/sec.

(c) In a flashing situation with a reflective object, the images will tend to be dark, with the object shining brightly. The techniques discussed in Section 10.6.1 would then be quite adequate.

(d) First we have to determine if a partial or whole image of the bullet has been obtained. After the pixels corresponding to the object have been identified us-

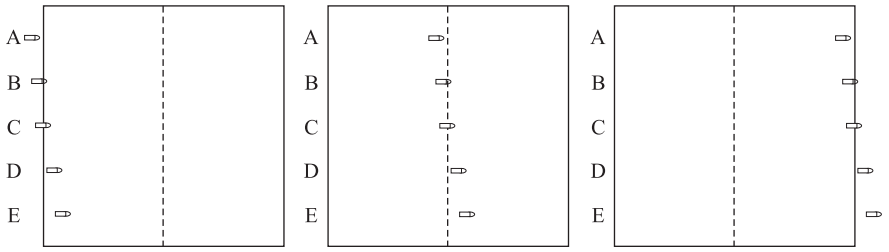


Figure P10.49

ing motion segmentation, we determine if the object runs into the left boundary (see the solution to Problem 9.36 regarding a method for determining if a binary object runs into the boundary of an image). If it does, we look at the next two frames, with the assurance that a complete image of the bullet has been obtained in each because of the frame rate in (b). If the object does not run into the left boundary, we are similarly assured of two full shots in two of the three frames. We then compute the centroid of the object in each image and count the number of pixels between the centroids. Because the distance between pixels and the time between frames are known, computation of the speed is a trivial problem. The principal uncertainty in this approach is how well the object is segmented. However, because the images are of the same object in basically the same geometry, consistency of segmentation between frames can be expected.

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.

(b) Code: 11076765543322. The starting point is 0, yielding the sequence

07676554332211.

Problem 11.2

(a) The first difference only counts the number of directions that separate adjacent elements of the code. Because the counting process is independent of direction, the first difference is independent of boundary rotation. (It is worthwhile to point out to students that the assumption here is that rotation does not change the code itself).

(b) Code: 0101030303323232212111. Difference: 3131331313031313031300. The code was treated as a circular sequence, so the first element of the difference is the transition between the last and first element of the code, as explained in the text.

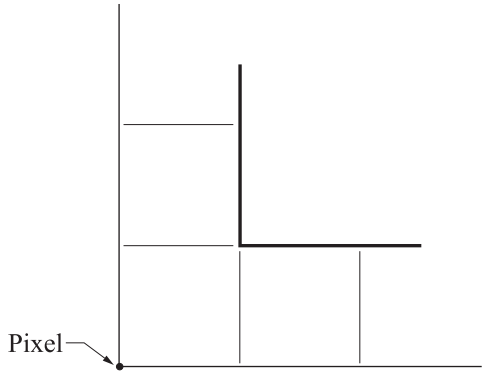


Figure P11.3

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.

(b) If a corner of a cell is centered at a pixel on the boundary, and the cell is such that the rubber band is tightened on the opposite corner, we would have the situation in Fig. P11.3. Assuming that the cell is of size $d \times d$, the maximum difference between the pixel and the boundary in that cell is $\sqrt{2}d$. If cells are centered on pixels, the maximum difference is $(\sqrt{2}d)/2$.

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. 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 ver-

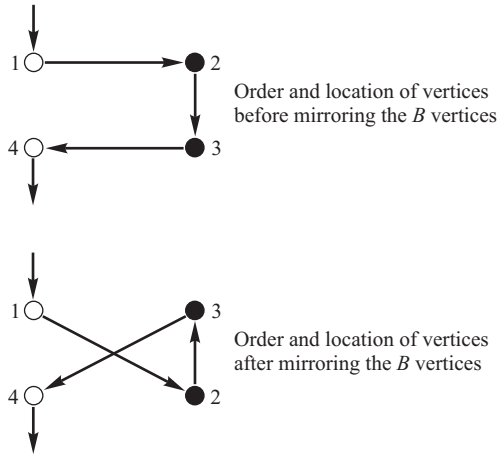


Figure P11.4

tex 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.

(c) When the B vertices of a 1-pixel deep protrusion are mirrored, they become aligned with the two convex vertices of the protrusion, which are known to be vertices of the MPP (otherwise they would be the corners of the protrusion). A line passing from the first convex vertex to the previous MPP vertex would show that the corresponding mirrored B vertex would be outside the MPP, so it could not be a vertex of the MPP. A similar conclusion is reached by passing a line from the second corner to the next vertex of the MPP, so the second mirrored vertex could not be part of the MPP. The conclusion is that the small protrusion would be missed by the algorithm.

(d) A drawing similar to the one used in (b) would show that both vertices of the protrusion would be detected normally for any protrusion extending for more than one pixel.

Problem 11.5

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

(b) Actually, in both cases the resulting polygon would contain all the boundary pixels.

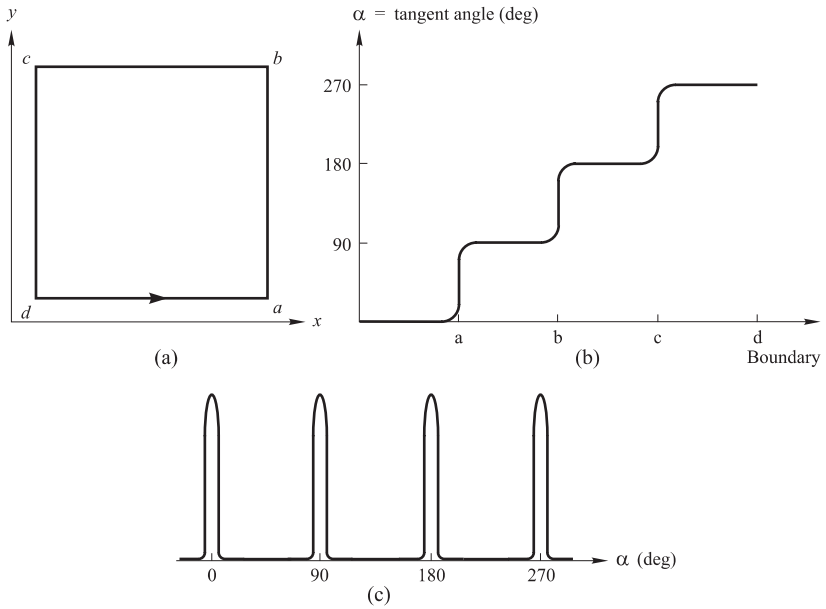


Figure P11.6

Problem 11.6

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

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

Problem 11.7

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

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

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.

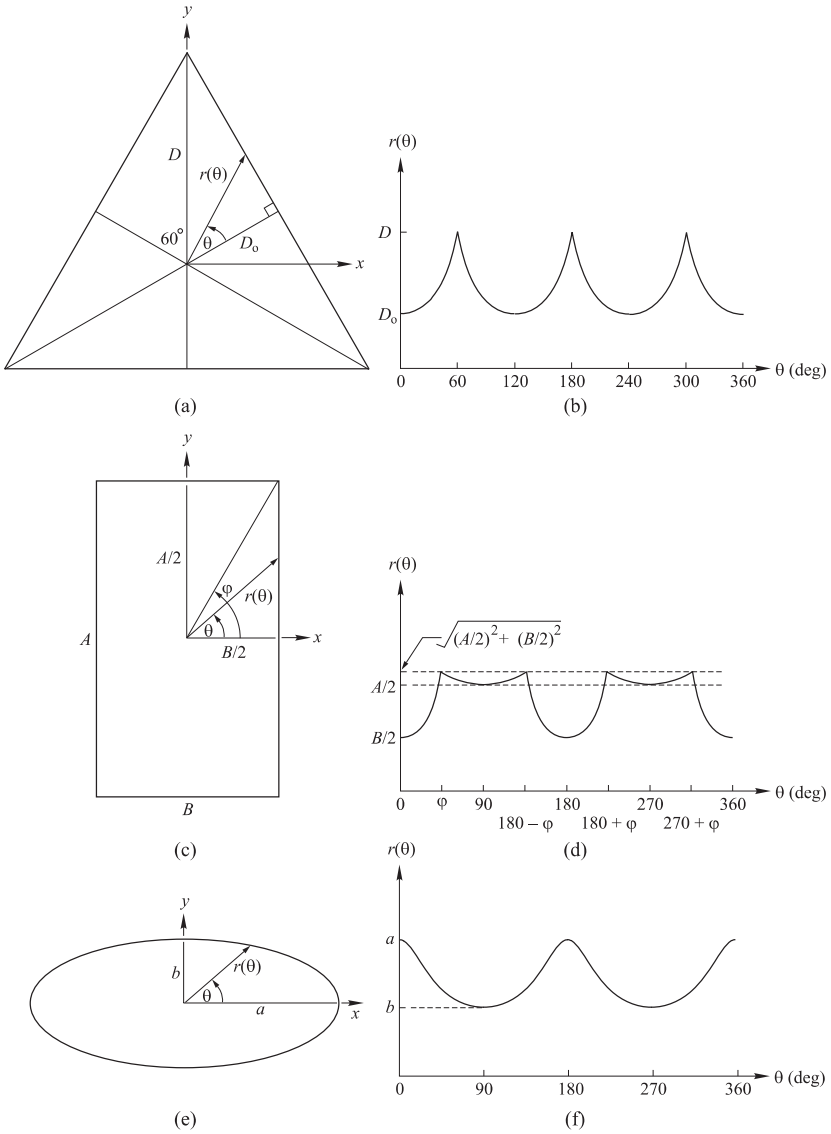


Figure P11.7

(b) From Fig. P11.7(c),

$$\begin{aligned}
r(\theta) &= \frac{B}{2\cos\theta} && 0^\circ \le \theta < \varphi \\
&= \frac{A}{2\cos(90^\circ-\theta)} && \varphi \le \theta < 90^\circ \\
&= \frac{A}{2\cos(\theta-90^\circ)} && 90^\circ \le \theta < (180^\circ-\varphi) \\
&= \frac{B}{2\cos(180^\circ-\theta)} && (180^\circ-\varphi) \le \theta < 180^\circ \\
&= \frac{B}{2\cos(\theta-180^\circ)} && 180^\circ \le \theta < 180^\circ+\varphi \\
&= \frac{A}{2\cos(270^\circ-\theta)} && 180^\circ+\varphi \le \theta < 270^\circ \\
&= \frac{A}{2\cos(\theta-270)} && 270^\circ \le \theta < 270^\circ+\varphi \\
&= \frac{B}{2\cos(360^\circ-\theta)} && 270^\circ+\varphi \le \theta < 360^\circ.
\end{aligned}$$

where $\varphi = \tan^{-1}(A/B)$.

(c) The equation of the ellipse in Fig. P11.7(e) is

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}=1.$$

We are interested in the distance from the origin to an arbitrary point (x,y) on the ellipse. In polar coordinates,

$$x = r\cos\theta$$

and

$$y = r\sin\theta$$

where r is the distance from the origin to (x,y):

$$r=\sqrt{x^2+y^2}.$$

Substituting into the equation of the ellipse we obtain

$$\frac{r^2\cos^2\theta}{a^2}+\frac{r^2\sin^2\theta}{b^2}=1$$

from which we obtain the desired result:

$$r(\theta)=\frac{1}{\left[\left(\dfrac{\cos\theta}{a}\right)^2+\left(\dfrac{\sin\theta}{b}\right)^2\right]^{1/2}}.$$

When $b=a$, we have the familiar equation of a circle, $r(\theta)=a$, or $x^2+y^2=a^2$.

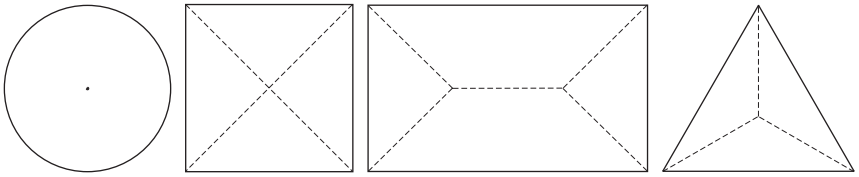


Figure P11.8

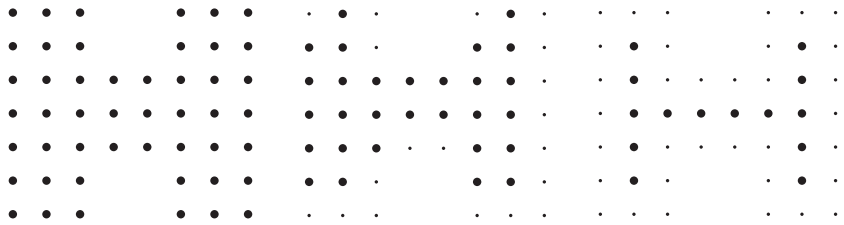


Figure P11.10

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.

(b) In the first case $p2 \cdot p6 \cdot p8 = 1$ so condition (d′) in Eq. (11.1-6) is violated and p is left unchanged. In the second case $N(p)=1$ so p is left unchanged. In the third case (c′) and (d′) are violated and p is left unchanged. In the fourth case $S(p)=2$ and p is left unchanged.

Problem 11.10

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

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

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.

(b) Starting at the top left corner,

Chain code:	000332123211
Difference:	300303311330
Shape number:	003033113303

Problem 11.12

With reference to Chapter 4, the DFT can be real only if the data sequence is conjugate symmetric. Only contours that are symmetric with respect to the origin have this property. The axis system of Fig. 11.19 would have to be set up so that this condition is satisfied for symmetric figures. This can be accomplished by placing the origin at the center of gravity of the contour.

Problem 11.13

Suppose that we have a particle moving in the xy-plane. At each time t the particle will be at some point whose coordinates can be written as $[x(y), y(t)]$. The situation with which we are dealing is in the complex plane. Recalling that a point in the complex plane is written at $c = x + jy$, we can write the location of the moving particle in the complex plane as $z(t) = x(t) + jy(t)$. As t varies, $z(t)$ describes a path in the complex plane. This is called the *parametric representation* of a curve because it depends on parameter t.

The "standard" equation of a circle with center at (b, c) and radius r is given by $(x-b)^2 + (y-c)^2 = r^2$. Using polar coordinates, we can write

$$\begin{aligned} x(\theta) &= b + r\cos\theta \\ y(\theta) &= c + r\sin\theta. \end{aligned}$$

Then, the circle can be represented in parametric form as the pair $\big(x(\theta), y(\theta)\big)$, with θ being the parameter. The circle can thus be represented in the complex plane, as the curve

$$\begin{aligned} z(\theta) &= x(\theta) + jy(\theta) \\ &= (b+c) + r(\cos\theta + j\sin\theta). \end{aligned}$$

Using only two descriptors in Eq. (11.2-5) gives

$$\begin{aligned}\hat{s}(k) &= \frac{1}{P}\left[a(0)+a(1)\left(\cos\frac{2\pi k}{K}+j\sin\frac{2\pi k}{K}\right)\right] \\ &= \frac{a(0)}{P}+\frac{a(1)}{P}\left(\cos\frac{2\pi k}{K}+j\sin\frac{2\pi k}{K}\right)\end{aligned}$$

which we see is in the form of a circle by letting $(b+c)=a(0)/P$, $r=a(1)/P$, and $\theta=2\pi k/K$.

Problem 11.14

The mean is sufficient.

Problem 11.15

Two ellipses with different, say, major axes, have signatures with the same mean and third statistical moment descriptors (both due to symmetry) but different second moment (due to spread).

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

(a) Co-occurrence matrix:

$$\begin{matrix} 19600 & 200 \\ 0 & 20000 \end{matrix}$$

(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 $p_{22} = 0.5025$.

(c) Descriptors:
(1) *Maximum probability:* 0.05025.
(2) *Correlation:* The means are

$$\begin{aligned} m_r &= \sum_{i=1}^{2} i \sum_{j=1}^{2} p_{ij} \\ &= 1(p_{11}+p_{12})+2(p_{21}+p_{22}) \\ &= 1(.4925+0.005)+2(0+0.5025) \\ &= 1.5025 \end{aligned}$$

and

$$\begin{aligned} m_c &= \sum_{i=1}^{2} j \sum_{i=1}^{2} p_{ij} \\ &= 1(p_{11}+p_{21})+2(p_{12}+p_{22}) \\ &= 1.5075. \end{aligned}$$

Similarly, the standard deviations are:

$$\begin{aligned} \sigma_r^2 &= \sum_{i=1}^{2} (i-m_r)^2 \sum_{j=1}^{2} p_{ij} \\ &= (1-m_r)(p_{11}+p_{12})+(2-m_r)(p_{21}+p_{22}) \\ &= 0.5000 \end{aligned}$$

and

$$\begin{aligned} \sigma_c^2 &= \sum_{j=1}^{2} (j-m_c)^2 \sum_{i=1}^{2} p_{ij} \\ &= (1-mc)(p11+p21)+(2-mc)(p12+p22) \\ &= 0.4999. \end{aligned}$$

Then, the correlation measure is computed as

$$\begin{aligned}\sum_{i=1}^{2}\sum_{j=1}^{2}\frac{(i-m_r)(j-m_c)\,p_{ij}}{\sigma_r\sigma_c} &= \frac{1}{\sigma_r\sigma_c}\sum_{i=1}^{2}\sum_{j=1}^{2}(i-m_r)(j-m_c)\,p_{ij}\\ &= 0.9900.\end{aligned}$$

(3) *Contrast:*

$$\begin{aligned}\sum_{i=1}^{2}\sum_{j=1}^{2}(i-j)^2\,p_{ij} &= (1-1)^2p_{11}+(1-2)^2p_{12}+(2-1)^2p_{21}+(2-2)^2p_{22}\\ &= 0.005.\end{aligned}$$

(4) *Uniformity:*

$$\begin{aligned}\sum_{i=1}^{2}\sum_{j=1}^{2}p_{ij}^2 &= (p_{11})^2+(p_{12})^2+(p_{21})^2+(p_{22})^2\\ &= 0.4951.\end{aligned}$$

(5) *Homogeneity:*

$$\begin{aligned}\sum_{i=1}^{2}\sum_{i=1}^{2}\frac{p_{ij}}{1+|i-j|} &= \frac{p11}{1}+\frac{p12}{2}+\frac{p21}{2}+\frac{p22}{1}\\ &= 0.9975.\end{aligned}$$

(6) *Entropy:*

$$\begin{aligned}-\sum_{i=1}^{2}\sum_{i=1}^{2}p_{ij}\log_2 p_{ij} &= -[p_{11}\log_2 p_{11}+p_{12}\log_2 p_{12}+p_{21}\log_2 p_{21}\\ &\qquad +p_{22}\log_2 p_{22}]\\ &= 1.0405\end{aligned}$$

where we used $0\log_2 0\equiv 0$.

Problem 11.18

We can use the position operator P: "$2m$ pixels to the right and $2m$ pixels below."

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 $\mathbf{G}$ 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

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

(b) In this case, g_{11} is the number of 0's two pixels to the right of a pixel valued 0. By inspection, $g_{11} = 8$. Similarly, $g_{12} = 0$, $g_{21} = 0$, and $g_{22} = 7$. The number of pixels satisfying P is 15, so the normalized co-occurrence matrix is

$$\mathbf{G} = \begin{bmatrix} 8/15 & 0 \\ 0 & 7/15 \end{bmatrix}.$$

Problem 11.20

When assigning this problem, the Instructor may wish to point the student to the review of matrices and vectors in the book web site.

From Eq. (11.4-6),

$$\mathbf{y} = \mathbf{A}(\mathbf{x} - \mathbf{m}_\mathbf{x}).$$

Then,

$$\begin{aligned} \mathbf{m}_\mathbf{y} &= E\{\mathbf{y}\} = E\{\mathbf{A}(\mathbf{x} - \mathbf{m}_\mathbf{x})\} \\ &= \mathbf{A}[E\{\mathbf{x}\} - E\{\mathbf{m}_\mathbf{x}\}] \\ &= \mathbf{A}(\mathbf{m}_\mathbf{x} - \mathbf{m}_\mathbf{x}) \\ &= \mathbf{0}. \end{aligned}$$

This establishes the validity of Eq. (11.4-7).

To prove the validity of Eq. (11.4-8), we start with the definition of the covariance matrix given in Eq. (11.4-3):

$$\mathbf{C}_\mathbf{y} = E\{(\mathbf{y} - \mathbf{m}_\mathbf{y})(\mathbf{y} - \mathbf{m}_\mathbf{y})^T\}.$$

Because $\mathbf{m_y} = \mathbf{0}$, it follows that

$$\begin{aligned}\mathbf{C_y} &= E\{\mathbf{y}\mathbf{y}^T\} \\ &= E\{[\mathbf{A}(\mathbf{x}-\mathbf{m_x})][\mathbf{A}(\mathbf{x}-\mathbf{m_x})]^T\} \\ &= \mathbf{A}\,E\{(\mathbf{x}-\mathbf{m_x})(\mathbf{x}-\mathbf{m_x})^T\}\mathbf{A}^T \\ &= \mathbf{A}\mathbf{C_x}\mathbf{A}^T.\end{aligned}$$

Showing the validity of Eq. (11.4-9) is a little more complicated. We start by noting that covariance matrices are real and symmetric. From basic matrix algebra, it is known that a real symmetric matrix of order n has n linearly independent eigenvectors (which are easily orthonormalized by, say, the Gram-Schmidt procedure). The rows of matrix $\mathbf{A}$ are the orthonormal eigenvectors of $\mathbf{C_x}$. Then,

$$\begin{aligned}\mathbf{C_x}\mathbf{A}^T &= \mathbf{C_x}[\mathbf{e}_1, \mathbf{e}_2, \ldots \mathbf{e}_n] \\ &= [\mathbf{C_x}\mathbf{e}_1, \mathbf{C_x}\mathbf{e}_2, \ldots \mathbf{C_x}\mathbf{e}_n] \\ &= [\lambda_1\mathbf{e}_1, \lambda_2\mathbf{e}_2, \ldots, \lambda_n\mathbf{e}_n] \\ &= \mathbf{A}^T\mathbf{D}\end{aligned}$$

where we used the definition of an eigenvector (i.e., $\mathbf{C_x}\mathbf{e}_i = \lambda_i\mathbf{e}_i$) and $\mathbf{D}$ is a diagonal matrix composed of the eigenvalues of $\mathbf{C_x}$:

$$\mathbf{D} = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}.$$

Premultiplying both sides of the preceding equation by matrix $\mathbf{A}$ gives

$$\begin{aligned}\mathbf{A}\mathbf{C_x}\mathbf{A}^T &= \mathbf{A}\mathbf{A}^T\mathbf{D} \\ &= \mathbf{D}\end{aligned}$$

where we used the fact that $\mathbf{A}^T\mathbf{A} = \mathbf{A}\mathbf{A}^T = \mathbf{I}$ because the rows of $\mathbf{A}$ are orthonormal vectors. Therefor, because $\mathbf{C_y} = \mathbf{A}\mathbf{C_x}\mathbf{A}^T$, we have shown that $\mathbf{C_y}$ is a diagonal matrix that is produced by diagonalizing matrix $\mathbf{C_x}$ using a transformation matrix composed of its eigenvectors. The eigenvalues of $\mathbf{C_y}$ are seen to be the same as the eigenvalues of $\mathbf{C_x}$. (Recall that the eigenvalues of a diagonal matrix are its diagonal terms). The fact that $\mathbf{C_y}\mathbf{e}_i = \mathbf{D}\mathbf{e}_i = \lambda_i\mathbf{e}_i$ shows that the eigenvectors of $\mathbf{C_y}$ are equal to the eigenvectors of $\mathbf{C_x}$.

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 11.22

This problem is similar to the previous one. The covariance matrix is of order 4096×4096 because the images are of size 64×64. It is given that the covariance matrix is the identity matrix, so all its 4096 eigenvalues are equal to 1. From Eq. (11.4-12), the mean square error is

$$\begin{aligned} e_{ms} &= \sum_{j=1}^{4096} \lambda_j - \sum_{i=1}^{2048} \lambda_i \\ &= 2048. \end{aligned}$$

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 11.24

A solution using the relationship "connected to," is shown in Fig. P11.24.

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

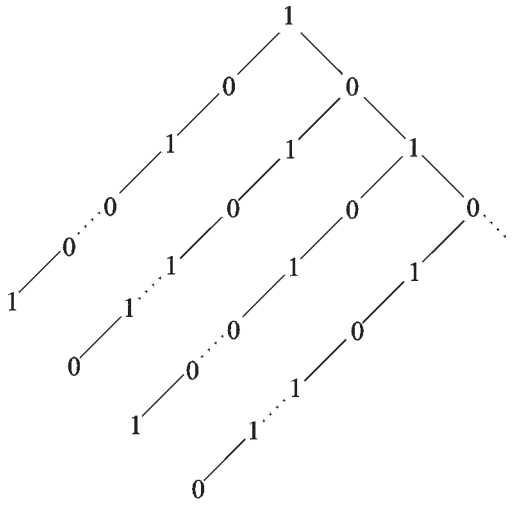


Figure P11.24

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.

Problem 11.26

This problem has four major parts. (1) Detecting individual bottles in an image; (2) finding the top each bottle; (3) finding the neck and shoulder of each bottle; and (4) determining the level of the liquid in the region between the neck and the shoulder.

(1) *Finding individual bottles.* Note that the background in the sample image is much darker than the bottles. We assume that this is true in all images. Then, a simple way to find individual bottles is to find vertical black stripes in the image having a width determined by the average separation between bottles, a number that is easily computable from images representative of the actual setup during

operation. We can find these stripes in various ways. One way is to smooth the image to reduce the effects of noise (we assume that, say, a 3×3 or 5×5 averaging mask is sufficient). Then, we run a horizontal scan line through the middle of the image. The low values in the scan line will correspond to the black or nearly black background. Each bottle will produce a significant rise and fall of intensity level in the scan line for the width of the bottle. Bottles that are fully in the field of view of the camera will have a predetermined average width. Bottles that are only partially in the field of view will have narrower profiles, and can be eliminated from further analysis (but we need to make sure that the trailing incomplete bottles are analyzed in the next image; presumably, the leading partial bottle was already processed.).

(2) *Finding the top of each bottle.* Once the location of each (complete or nearly complete) bottle is determined, we again can use the contrast between the bottles and the background to find the top of the bottle. One possible approach is to compute a gradient image (sensitive only to horizontal edges) and look for a horizontal line near the top of the gradient image. An easier method is to run a vertical scan line through the center of the locations found in the previous step. The first major transition in gray level (from the top of the image) in the scan line will give a good indication of the location of the top of a bottle.

(3) *Finding the neck and shoulder of a bottle.* In the absence of other information, we assume that all bottles are of the same size, as shown in the sample image. Then, once we now where the top of a bottle is, the location of the neck and shoulder are known to be at a fixed distance from the bottle top.

(4) *Determining the level of the liquid.* The area defined by the bottom of the neck and the top of the shoulder is the only area that needs to be examined to determine acceptable vs. unacceptable fill level in a given bottle. In fact, As shown in the sample image, an area of a bottle that is void of liquid appears quite bright in an image, so we have various options. We could run a single vertical scan line again, but note that the bottles have areas of reflection that could confuse this approach. This computation is at the core of what this system is designed to do, so a more reliable method should be used. One approach is to threshold the area spanning a rectangle defined by the bottom of the neck, the shoulder, and sides of the bottle. Then, we count the number of white pixels above the midpoint of this rectangle. If this number is greater than a pre-established value, we know that enough liquid is missing and declare the bottle improperly filled. A slightly more sophisticated technique would be to actually find the level of the liquid. This would consist of looking for a horizontal edge in the region within the bottle defined by the sides of the bottle, the bottom of the neck, and a line passing midway between the shoulder and the bottom of the neck. A gradient/edge-linking approach, as described in Chapter 10, would be suitable. Note however,

that if no edge is found, the region is either filled (dark values in the region) or completely void of liquid (white, or near white values in the region). A computation to resolve these two possible conditions has to follow if the system fails to find an edge.

Problem 11.27

The key specification of the desired system is that it be able to detect individual bubbles. No specific sizes are given. We assume that bubbles are nearly round, as shown in the test image. One solution consists of (1) segmenting the image; (2) post-processing the result; (3) finding the bubbles and bubble clusters, and determining bubbles that merged with the boundary of the image; (4) detecting groups of touching bubbles; (5) counting individual bubbles; and (6) determining the ratio of the area occupied by all bubbles to the total image area.

(1) *Segmenting the image*. We assume that the sample image is truly representative of the class of images that the system will encounter. The image shown in the problem statement is typical of images that can be segmented by a global threshold. As shown by the histogram in Fig. P11.27, the intensity levels of the objects of interest are high on the gray scale. A simple adaptive threshold method for data that is that high on the scale is to choose a threshold equal to the mean plus a multiple of the standard deviation. We chose a threshold equal to $m+2\sigma$, which, for the image in the problem statement, was 195. The segmented result is shown on the right of Fig. P11.27. Obviously this is not the only approach we could take, but this is a simple method that adapts to overall changes in intensity.

(2) *Post-processing*. As shown in the segmented image of Fig. P11.27, many of the bubbles appear as broken disks, or disks with interior black components. These are mostly due either to reflection or actual voids within a bubble. We could attempt to build a morphological procedure to repair and/or fill the bubbles. However, this can turn into a computationally expensive process that is not warranted unless stringent measurement standards are required, a fact not mentioned in the problem statement. An alternative is to calculate, on average (as determined from a set of sample images), the percentage of bubble areas that are filled with black or have black "bays" which makes their black areas merge with the background. Then, once the dimensions of each bubble (or bubble cluster) have been established, a correction factor based on area would be applied.

(3) *Finding the bubbles*. Refer to the solution to Problem 9.36. The solution is based on connected components, which also yields all bubbles and bubble clusters.

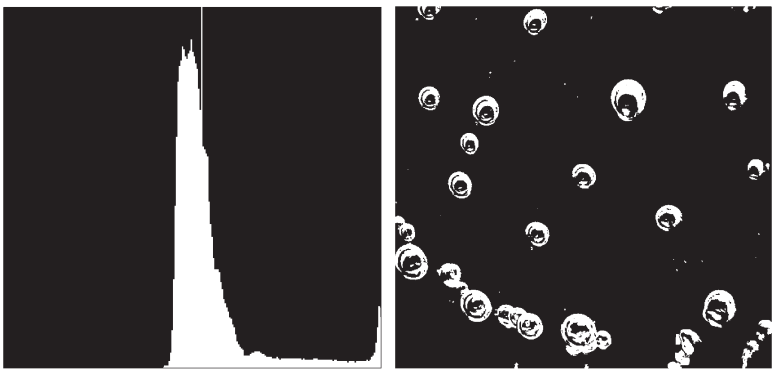


Figure P11.27

(4) In order to detect bubble clusters we make use of shape analysis. For each connected component, we find the eigen axes (see Section 11.4) and the standard deviation of the data along these axes (square root of the eigenvalues of the covariance matrix). One simple solution is to compute the ratio of the large to the small variance of each connected component along the eigen axes. A single, uniformly-filled, perfectly round bubble will have a ratio of 1. Deviations from 1 indicate elongations about one of the axes. We look for elliptical shapes as being formed by clusters of bubbles. A threshold to classify bubbles as single vs. clusters has to be determined experimentally. Note that single pixels or pixel streaks one pixel wide have a standard deviation of zero, so they must be processed separately. We have the option of considering connected components that consist of only one pixel to be either noise, or the smallest detectable bubble. No information is given in the problem statement about this. In theory, it is possible for a cluster to be formed such that its shape would be symmetrical about both axes, in which case the system would classify the cluster as a single bubble. Resolution of conflicts such as this would require additional processing. However, there is no evidence in the sample image to suggest that this in fact is a problem. Bubble clusters tend to appear as elliptical shapes. In cases where the ratio of the standard deviations is close to the threshold value, we could add additional processing to reduce the chances of making a mistake.

(5) *Counting individual bubbles.* A bubble that does not merge with the border of the image or is not a cluster, is by definition a single bubble. Thus, counting these bubbles is simply counting the connected components that have not been tagged as clusters or merged with the boundary of the image.

(6) *Ratio of the areas.* This ratio is simply the number of pixels in all the connected components plus the correction factors mentioned in (2), divided by the total number of pixels in the image.

The problem also asks for the size of the smallest bubble the system can detect. If, as mentioned in (4), we elect to call a one-pixel connected component a bubble, then the smallest bubble dimension detectable is the physical size of one pixel. From the problem statement, 700 pixels cover 7 cm, so the dimension of one pixel is 10 mm.

Chapter 12

Problem Solutions

Problem 12.1

(a) By inspection, the mean vectors of the three classes are, approximately, $\mathbf{m}_1 = (1.5, 0.3)^T$, $\mathbf{m}_2 = (4.3, 1.3)^T$, and $\mathbf{m}_3 = (5.5, 2.1)^T$ for the classes Iris setosa, versicolor, and virginica, respectively. The decision functions are of the form given in Eq. (12.2-5). Substituting the preceding values of mean vectors gives:

$$\begin{aligned} d_1(\mathbf{x}) &= \mathbf{x}^T\mathbf{m}_1 - \frac{1}{2}\mathbf{m}_1^T\mathbf{m}_1 = 1.5x_1 + 0.3x_2 - 1.2 \\ d_2(\mathbf{x}) &= \mathbf{x}^T\mathbf{m}_2 - \frac{1}{2}\mathbf{m}_2^T\mathbf{m}_2 = 4.3x_1 + 1.3x_2 - 10.1 \\ d_3(\mathbf{x}) &= \mathbf{x}^T\mathbf{m}_3 - \frac{1}{2}\mathbf{m}_3^T\mathbf{m}_3 = 5.5x_1 + 2.1x_2 - 17.3. \end{aligned}$$

(b) The decision boundaries are given by the equations

$$\begin{aligned} d_{12}(\mathbf{x}) &= d_1(\mathbf{x}) - d_2(\mathbf{x}) = -2.8x_1 - 1.0x_2 + 8.9 = 0 \\ d_{13}(\mathbf{x}) &= d_1(\mathbf{x}) - d_3(\mathbf{x}) = -4.0x_1 - 1.8x_2 + 16.1 = 0 \\ d_{23}(\mathbf{x}) &= d_2(\mathbf{x}) - d_3(\mathbf{x}) = -1.2x_1 - 0.8x_2 + 7.2 = 0. \end{aligned}$$

Figure P12.1 shows a plot of these boundaries.

Problem 12.2

From the definition of the Euclidean distance,

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

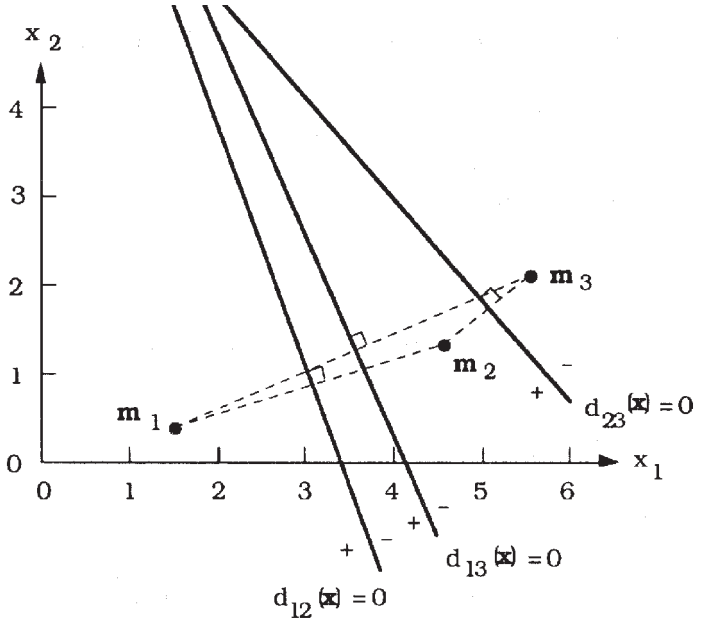


Figure P12.1

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 12.3

The equation of the decision boundary between a pair of mean vectors is

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

The midpoint between $\mathbf{m}_i$ and $\mathbf{m}_j$ is $(\mathbf{m}_i + \mathbf{m}_j)/2$ (see Fig. P12.3) . First, we show that this point is on the boundary by substituting it for $\mathbf{x}$ in the above equation

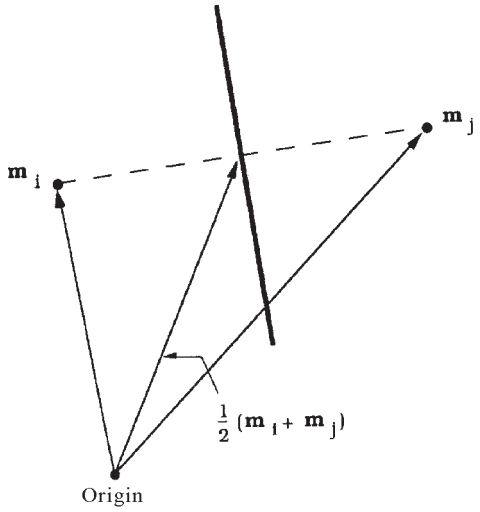


Figure P12.3

and showing that the result is equal to 0:

$$\begin{aligned}\frac{1}{2}(\mathbf{m}_i^T\mathbf{m}_i-\mathbf{m}_j^T\mathbf{m}_j)-\frac{1}{2}(\mathbf{m}_i^T\mathbf{m}_i-\mathbf{m}_j^T\mathbf{m}_j) &= \frac{1}{2}(\mathbf{m}_i^T\mathbf{m}_i-\mathbf{m}_j^T\mathbf{m}_j)\\ &\quad -\frac{1}{2}(\mathbf{m}_i^T\mathbf{m}_i-\mathbf{m}_j^T\mathbf{m}_j)\\ &= 0.\end{aligned}$$

Next, we show that the vector $(\mathbf{m}_i-\mathbf{m}_j)$ is perpendicular to the hyperplane boundary. There are several ways to do this. Perhaps the easiest is to show that $(\mathbf{m}_i-\mathbf{m}_j)$ is in the same direction as the unit normal to the hyperplane. For a hyperplane with equation $w_1x_1+w_2x_2+...w_nx_n+w_{n+1}=0$, the unit normal is

$$\mathbf{u}=\frac{\mathbf{w}_o}{\|\mathbf{w}_o\|}$$

where $\mathbf{w}_o=(w_1,w_2,...,w_n)^T$. Comparing the above equation for $d_{ij}(\mathbf{x})$ with the general equation of a hyperplane just given, we see that $\mathbf{w}_o=(\mathbf{m}_i-\mathbf{m}_j)$ and $w_{n+1}=-(\mathbf{m}_i^T\mathbf{m}_i-\mathbf{m}_j^T\mathbf{m}_j)/2$. Thus, the unit normal of our decision boundary is

$$\mathbf{u}=\frac{(\mathbf{m}_i-\mathbf{m}_j)}{\left\|\mathbf{m}_i-\mathbf{m}_j\right\|}$$

which is in the same direction as the vector $(\mathbf{m}_i-\mathbf{m}_j)$. This concludes the proof.

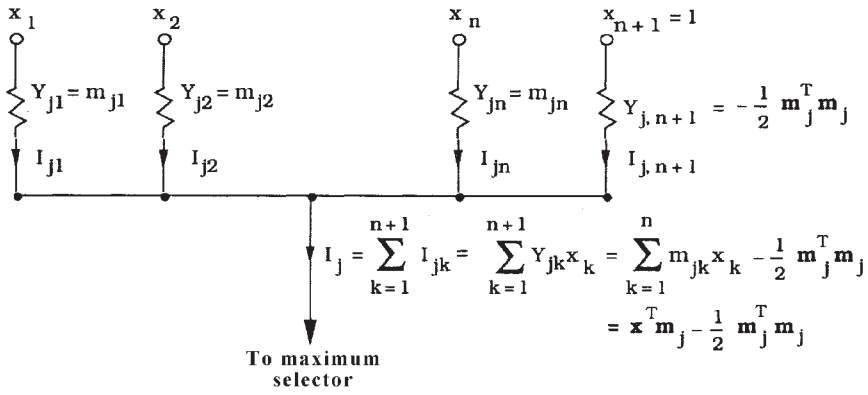


Figure P12.4

Problem 12.4

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

Problem 12.5

Assume that the mask is of size $J \times K$. For any value of displacement (s,t), we can express the area of the image under the mask, as well as the mask $w(x,y)$, in vector form by letting the first row of the subimage under the mask represent the first K elements of a column vector $\mathbf{a}$, the elements of next row the next K elements of $\mathbf{a}$, and so on. At the end of the procedure we subtract the average value of the intensity levels in the subimage from every element of $\mathbf{a}$. The vector $\mathbf{a}$ is of size $(J \times K) \times 1$. A similar approach yields a vector, $\mathbf{b}$, of the same size, for the mask $w(x,y)$ minus its average. This vector does not change as (s,t) varies because the coefficients of the mask are fixed. With this construction in mind, we see that the numerator of Eq. (12.2-8) is simply the vector inner-product $\mathbf{a}^T\mathbf{b}$. Similarly, the first term in the denominator is the norm squared of $\mathbf{a}$, denoted $\mathbf{a}^T\mathbf{a} = \|\mathbf{a}\|^2$, while the second term has a similar interpretation for $\mathbf{b}$. The correlation coefficient then becomes

$$\gamma(s,t) = \frac{\mathbf{a}^T\mathbf{b}}{\left[(\mathbf{a}^T\mathbf{a})(\mathbf{b}^T\mathbf{b})\right]^{1/2}}.$$

When $\mathbf{a} = \mathbf{b}$ (a perfect match), $\gamma(s,t) = \|\mathbf{a}\|^2 / \|\mathbf{a}\| \|\mathbf{a}\| = 1$, which is the maximum value obtainable by the above expression. Similarly, the minimum value occurs

when $a = -b$, in which case $\gamma(s,t) = -1$. Thus, although the vector $\mathbf{a}$ varies in general for every value of (s,t), the values of $\gamma(s,t)$ are all in the range $[-1,1]$.

Problem 12.6

The solution to the first part of this problem is based on being able to extract connected components (see Chapters 2 and 11) and then determining whether a connected component is convex or not (see Chapter 11). Once all connected components have been extracted we perform a convexity check on each and reject the ones that are not convex. All that is left after this is to determine if the remaining blobs are complete or incomplete. To do this, the region consisting of the extreme rows and columns of the image is declared a region of 1's. Then if the pixel-by-pixel AND of this region with a particular blob yields at least one result that is a 1, it follows that the actual boundary touches that blob, and the blob is called incomplete. When only a single pixel in a blob yields an AND of 1 we have a marginal result in which only one pixel in a blob touches the boundary. We can arbitrarily declare the blob incomplete or not. From the point of view of implementation, it is much simpler to have a procedure that calls a blob incomplete whenever the AND operation yields one or more results valued 1. After the blobs have been screened using the method just discussed, they need to be classified into one of the three classes given in the problem statement. We perform the classification problem based on vectors of the form $\mathbf{x} = (x_1, x_2)^T$, where x_1 and x_2 are, respectively, the lengths of the major and minor axis of an elliptical blob, the only type left after screening. Alternatively, we could use the eigen axes for the same purpose. (See Section 11.2.1 on obtaining the major axes or the end of Section 11.4 regarding the eigen axes.) The mean vector of each class needed to implement a minimum distance classifier is 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 12.7

(a) Because it is given that the pattern classes are governed by Gaussian densities, only knowledge of the mean vector and covariance matrix of each class are

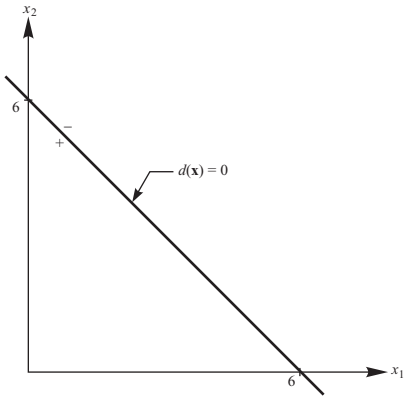


Figure P12.7

required to specify the Bayes classifier. Substituting the given patterns into Eqs. (12.2-22) and (12.2-23) yields

$$\mathbf{m}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\begin{aligned} \mathbf{m}_2 &= \begin{bmatrix} 5 \\ 5 \end{bmatrix} \\ \mathbf{C}_1 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \mathbf{C}_1^{-1} \end{aligned}$$

and

$$\mathbf{C}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \mathbf{C}_2^{-1}.$$

Because $\mathbf{C}_1 = \mathbf{C}_2 = \mathbf{I}$, the decision functions are the same as for a minimum distance classifier:

$$d_1(\mathbf{x}) = \mathbf{x}^T\mathbf{m}_1 - \frac{1}{2}\mathbf{m}_1^T\mathbf{m}_1 = 1.0x_1 + 1.0x_2 - 1.0$$

and

$$d_2(\mathbf{x}) = \mathbf{x}^T\mathbf{m}_2 - \frac{1}{2}\mathbf{m}_2^T\mathbf{m}_2 = 5.0x_1 + 5.0x_2 - 25.0.$$

The Bayes decision boundary is given by the equation $d(\mathbf{x}) = d_1(\mathbf{x}) - d_2(\mathbf{x}) = 0$, or

$$d(\mathbf{x}) = -4.0x_1 - 4.0x_2 + 24.0 = 0.$$

(b) Figure P12.7 shows a plot of the boundary.

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 12.9

The basic mechanics are the same as in Problem 12.6, but we have the additional requirement of computing covariance matrices from the training patterns of each class.

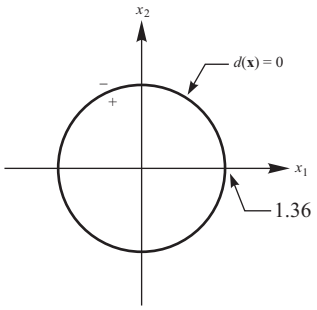


Figure P12.8

Problem 12.10

From basic probability theory,

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

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

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

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

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

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 $j = 1, 2, ..., W$, 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 12.11

(a) For class ω_1 we let $\mathbf{y}(1) = (0,0,0,1)^T$, $\mathbf{y}(2) = (1,0,0,1)^T$, $\mathbf{y}(3) = (1,0,1,1)T$, $\mathbf{y}(4) = (1,1,0,1)T$. Similarly, for class ω_2, $\mathbf{y}(5) = (0,0,1,1)^T$, $\mathbf{y}(6) = (0,1,1,1)^T$, $\mathbf{y}(7) = (0,1,0,1)^T$, $\mathbf{y}(8) = (1,1,1,1)^T$. Then, using $c = 1$ and

$$\mathbf{w}(1) = (-1,-2,-2,0)^T$$

it follows from Eqs. (12.2-34) through (12.2-36) that:

$$\begin{aligned}
&\mathbf{w}(1)^T\mathbf{y}(1)=0, && \mathbf{w}(2)=\mathbf{w}(1)+\mathbf{y}(1)=(-1,-2,-2,1)^T;\\
&\mathbf{w}(2)^T\mathbf{y}(2)=0, && \mathbf{w}(3)=\mathbf{w}(2)+\mathbf{y}(2)=(0,-2,-2,2)^T;\\
&\mathbf{w}(3)^T\mathbf{y}(3)=0, && \mathbf{w}(4)=\mathbf{w}(3)+\mathbf{y}(3)=(1,-2,-1,3)^T;\\
&\mathbf{w}(4)^T\mathbf{y}(4)=2, && \mathbf{w}(5)=\mathbf{w}(4)=(1,-2,-1,3)^T;\\
&\mathbf{w}(5)^T\mathbf{y}(5)=2, && \mathbf{w}(6)=\mathbf{w}(5)-\mathbf{y}(5)=(-1,-2,-2,2)^T;\\
&\mathbf{w}(6)^T\mathbf{y}(6)=-2, && \mathbf{w}(7)=\mathbf{w}(6)=(-1,-2,-2,2)^T;\\
&\mathbf{w}(7)^T\mathbf{y}(7)=0, && \mathbf{w}(8)=\mathbf{w}(7)-\mathbf{y}(7)=(1,-3,-2,1)^T;\\
&\mathbf{w}(8)^T\mathbf{y}(8)=-3, && \mathbf{w}(9)=\mathbf{w}(8)=(1,-3,-2,1)^T.
\end{aligned}$$

A complete iteration through all patterns with no errors was not achieved. Therefore, the patterns are recycled by letting $\mathbf{y}(9)=\mathbf{y}(1)$, $\mathbf{y}(10)=\mathbf{y}(2)$, and so on, which gives

$$\begin{aligned}
&\mathbf{w}(9)^T\mathbf{y}(9)=1, && \mathbf{w}(10)=\mathbf{w}(9)=(1,-3,-2,1)^T;\\
&\mathbf{w}(10)^T\mathbf{y}(10)=2, && \mathbf{w}(11)=\mathbf{w}(10)=(1,-3,-2,1)^T;\\
&\mathbf{w}(11)^T\mathbf{y}(11)=0, && \mathbf{w}(12)=\mathbf{w}(11)+\mathbf{y}(11)=(2,-3,-1,2)^T;\\
&\mathbf{w}(12)^T\mathbf{y}(12)=1, && \mathbf{w}(13)=\mathbf{w}(12)=(2,-3,-1,2)^T;\\
&\mathbf{w}(13)^T\mathbf{y}(13)=1, && \mathbf{w}(14)=\mathbf{w}(13)-\mathbf{y}(13)=(2,-3,-2,1)^T;\\
&\mathbf{w}(14)^T\mathbf{y}(14)=-4, && \mathbf{w}(15)=\mathbf{w}(14)=(2,-3,-2,1)^T;\\
&\mathbf{w}(15)^T\mathbf{y}(15)=-2, && \mathbf{w}(16)=\mathbf{w}(15)=(2,-3,-2,1)^T;\\
&\mathbf{w}(16)^T\mathbf{y}(16)=-2, && \mathbf{w}(17)=\mathbf{w}(16)=(2,-3,-2,1)^T.
\end{aligned}$$

Again, a complete iteration over all patterns without an error was not achieved, so the patterns are recycled by letting $\mathbf{y}(17)=\mathbf{y}(1)$, $\mathbf{y}(18)=\mathbf{y}(2)$, and so on, which gives:

$$\begin{aligned}
&\mathbf{w}(17)^T\mathbf{y}(17)=1, && \mathbf{w}(18)=\mathbf{w}(17)=(2,-3,-2,1)^T;\\
&\mathbf{w}(18)^T\mathbf{y}(18)=3, && \mathbf{w}(19)=\mathbf{w}(18)=(2,-3,-2,1)^T;\\
&\mathbf{w}(19)^T\mathbf{y}(19)=1, && \mathbf{w}(20)=\mathbf{w}(19)=(2,-3,-2,1)^T;\\
&\mathbf{w}(20)^T\mathbf{y}(20)=0, && \mathbf{w}(21)=\mathbf{w}(20)+\mathbf{y}(20)=(3,-2,-2,2)^T;\\
&\mathbf{w}(21)^T\mathbf{y}(21)=0, && \mathbf{w}(22)=\mathbf{w}(21)-\mathbf{y}(21)=(3,-2,-3,1)^T.
\end{aligned}$$

It is easily verified that no more corrections take place after this step, so $\mathbf{w}(22)=(3,-2,-3,1)^T$ is a solution weight vector.

(b) The decision surface is given by the equation

$$\mathbf{w}^T\mathbf{y}=3y_1-2y_2-3y_3+1=0.$$

A section of this surface is shown schematically in Fig. P12.11. The positive side of the surface faces the origin.

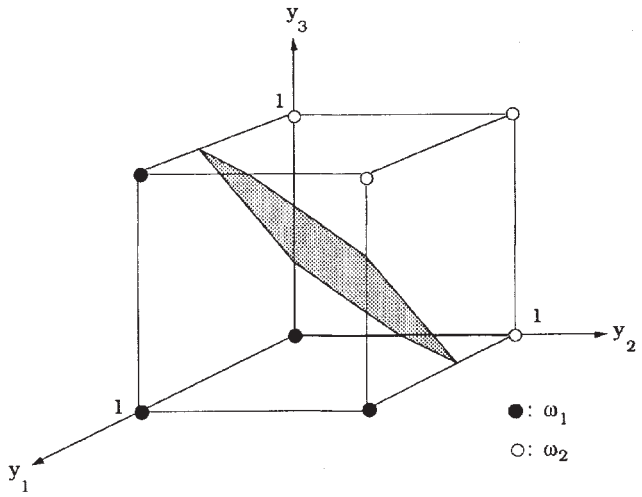


Figure P12.11

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 12.13

Let the training set of patterns be denoted by $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N$. It is assumed that the training patterns of class ω_2 have been multiplied by -1. If the classes are linearly separable, we want to prove that the perceptron training algorithm yields

a solution weight vector, $\mathbf{w}^*$, with the property

$$\mathbf{w}^{*T}\mathbf{y}_i \geq T_0$$

where T_0 is a nonnegative threshold. With this notation, the Perceptron algorithm (with $c = 1$) is expressed as $\mathbf{w}(k+1) = \mathbf{w}(k)$ if $\mathbf{w}^T(k)\mathbf{y}_i(k) \geq T_0$ or $\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{y}_i(k)$ otherwise.

Suppose that we retain only the values of k for which a correction takes place (these are the only indices of interest). Then, re-adapting the index notation, we may write

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{y}_i(k)$$

and

$$\mathbf{w}^T(k)\mathbf{y}i(k) \leq T_0.$$

With these simplifications in mind, the proof of convergence is as follows: From the above equation,

$$\mathbf{w}(k+1) = \mathbf{w}(1) + \mathbf{y}_i(1) + \mathbf{y}_i(2) + \cdots + \mathbf{y}_i(k).$$

Taking the inner product of the solution weight vector with both sides of this equation gives

$$\mathbf{w}^T(k+1)\mathbf{w}^* = \mathbf{w}^T(1)\mathbf{w}^* + \mathbf{y}_i^T(1)\mathbf{w}^* + \mathbf{y}_i^T(2)\mathbf{w}^* + \cdots + \mathbf{y}_i^T(k)\mathbf{w}^*.$$

Each term $\mathbf{y}_i^T(j)\mathbf{w}^*$, $j = 1, 2, ..., k$, is less than T_0, so

$$\mathbf{w}^T(k+1)\mathbf{w}^* \geq \mathbf{w}^T(1)\mathbf{w}^* + k\,T_0.$$

Using the Cauchy-Schwartz inequality, $\|\mathbf{a}\|^2 \|\mathbf{b}\|^2 \geq (\mathbf{a}^T\mathbf{b})^2$, results in

$$\left[\mathbf{w}^T(k+1)\mathbf{w}^*\right]^2 \leq \left\|\mathbf{w}^T(k+1)\right\|^2 \left\|\mathbf{w}^*\right\|^2$$

or

$$\left\|\mathbf{w}^T(k+1)\right\|^2 \geq \frac{\left[\mathbf{w}^T(k+1)\mathbf{w}^*\right]^2}{\|\mathbf{w}^*\|^2}.$$

Another line of reasoning leads to a contradiction regarding $\left\|\mathbf{w}^T(k+1)\right\|^2$. From above,

$$\left\|\mathbf{w}(j+1)\right\|^2 = \left\|\mathbf{w}(j)\right\|^2 + 2\mathbf{w}^T(j)\mathbf{y}_i(j) + \left\|\mathbf{y}_i(j)\right\|^2$$

or

$$\left\|\mathbf{w}(j+1)\right\|^2 - \left\|\mathbf{w}(j)\right\|^2 = 2\mathbf{w}^T(j)\mathbf{y}_i(j) + \left\|\mathbf{y}_i(j)\right\|^2.$$

Let $Q = \max_i \|\mathbf{y}_i(j)\|^2$. Then, because $\mathbf{w}^T(j)\mathbf{y}_i(j) \leq T_0$,

$$\left\|\mathbf{w}(j+1)\right\|^2 - \left\|\mathbf{w}(j)\right\|^2 \leq 2T_0 + Q.$$

Adding these inequalities for $j = 1, 2, \ldots, k$ yields

$$\left\|\mathbf{w}(j+1)\right\|^2 \leq \|\mathbf{w}(1)\|^2 + [2T_0 + Q]\, k.$$

This inequality establishes a bound on $\left\|\mathbf{w}(j+1)\right\|^2$ that conflicts for sufficiently large k with the bound established by our earlier inequality. In fact, k can be no larger than k_m, which is a solution to the equation

$$\frac{\left[\mathbf{w}^T(k+1)\mathbf{w}^* + k_m T_0\right]^2}{\|\mathbf{w}^*\|^2} = \|\mathbf{w}(1)\|^2 + [2T_0 + Q]\, k_m.$$

This equation says that k_m is finite, thus proving that the perceptron training algorithm converges in a finite number of steps to a solution weight vector $\mathbf{w}^*$ if the patterns of the training set are linearly separable.

Note: The special case with $T_0 = 0$ is proved in a slightly different manner. Under this condition we have

$$\mathbf{w}^T(k+1)\mathbf{w}^* \geq \mathbf{w}^T(1)\mathbf{w}^* + ka$$

where

$$a = \min_i \left[\mathbf{y}_i^T(j)\mathbf{w}^*\right].$$

Because, by hypothesis, $\mathbf{w}^*$ is a solution weight vector, we know that $\left[\mathbf{y}_i^T(j)\mathbf{w}^*\right] \geq 0$. Also, because $\mathbf{w}^T(j)\mathbf{y}_i(j) \leq (T = 0)$,

$$\begin{aligned} \left\|\mathbf{w}(j+1)\right\|^2 - \left\|\mathbf{w}(j)\right\|^2 &\leq \left\|\mathbf{y}_i(j)\right\|^2 \\ &\leq Q. \end{aligned}$$

The rest of the proof remains the same. The bound on the number of steps is the value of k_m that satisfies the following equation:

$$\frac{\left[\mathbf{w}^T(1)\mathbf{w}^* + k_m a\right]^2}{\|\mathbf{w}^*\|^2} = \|\mathbf{w}(1)\|^2 + Qk_m.$$

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 12.15

The approach to solving this problem is basically the same as in Problem 12.14. The idea is to combine the decision functions in the form of a hyperplane and then equate coefficients. For equal covariance matrices, the decision function for two pattern classes is obtained Eq. (12.2-27):

$$\begin{aligned} d_{ij}(\mathbf{x}) &= d_i(\mathbf{x}) - d_j(\mathbf{x}) = \ln P(\omega_i) - \ln P(\omega_j) + \mathbf{x}^T\mathbf{C}^{-1}(\mathbf{m}_i - \mathbf{m}_j) \\ &\quad -\frac{1}{2}(\mathbf{m}_i - \mathbf{m}_j)^T\mathbf{C}^{-1}(\mathbf{m}_i - \mathbf{m}_j). \end{aligned}$$

As in Problem 12.14, this is recognized as a linear decision function of the form

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

which is implemented by a single layer perceptron with coefficients

$$w_k = v_k \qquad k = 1, 2, \ldots, n$$

and

$$\theta = w_{n+1} = \ln P(\omega_i) - \ln P(\omega_j) + \mathbf{x}^T\mathbf{C}^{-1}(\mathbf{m}_i - \mathbf{m}_j)$$

where the v_k are elements of the vector

$$\mathbf{v} = \mathbf{C}^{-1}(\mathbf{m}_i - \mathbf{m}_j).$$

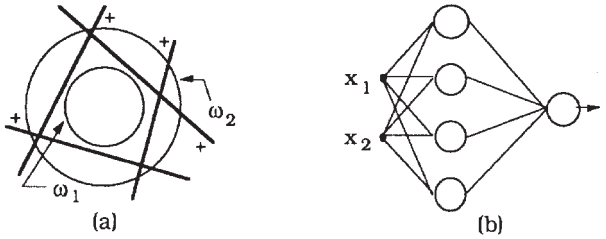


Figure P12.17

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 12.17

The classes and boundary needed to separate them are shown in Fig. P12.17(a). The boundary of minimum complexity in this case is a triangle, but it would be so tight in this arrangement that even small perturbations in the position of the patterns could result in classification errors. Thus, we use a network with the capability to implement 4 surfaces (lines) in 2D. The network, shown in Fig. P12.17(b), is an extension of the concepts discussed in the text in connection with Fig. 12.22. In this case, the output node acts like an AND gate with 4 inputs. The output node outputs a 1 (high) when the outputs of the preceding 4 nodes are all high simultaneously. This corresponds to a pattern being on the + side of all 4 lines and, therefore, belonging to class ω_1. Any other combination yields a 0 (low) output, indicating class ω_{21}.

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 neu-

ral 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 12.19

This problem, although it is a simple exercise in differentiation, is intended to help the student fix in mind the notation used in the derivation of the generalized delta rule. From Eq. (12.2-50), with $\theta_0 = 1$,

$$h_j(I_j) = \frac{1}{1 + e^{-\left[\sum_{k=1}^{N_K} w_{jk} O_k + \theta_j\right]}}.$$

Because, from Eq. (12.2-48),

$$I_j = \sum_{k=1}^{N_K} w_{jk} O_k$$

it follows that

$$h_j(I_j) = \frac{1}{1 + e^{-[I_j + \theta_j]}}.$$

Taking the partial derivative of this expression with respect to I_j gives

$$h'_j(I_j) = \frac{\partial h_j(I_j)}{\partial I_j} = \frac{e^{-[I_j + \theta_j]}}{\left[1 + e^{-[I_j + \theta_j]}\right]^2}.$$

From Eq. (12.2-49)

$$O_j = h_j(I_j) = \frac{1}{1 + e^{-[I_j + \theta_j]}}.$$

It is easily shown that

$$O_j(1 - O_j) = \frac{e^{-[I_j + \theta_j]}}{\left[1 + e^{-[I_j + \theta_j]}\right]^2}$$

so

$$h'_j(I_j) = O_j(1 - O_j).$$

This completes the proof.

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

Problem 12.21

$Q = 0$ implies that $\max(|A|, |B|) = M$. Suppose that $|A| > |B|$. Then, it must follow that $|A| = M$ and, therefore, that $M > |B|$. But M is obtained by matching A and B, so it must be bounded by $M \leq \min(|A|, |B|)$. Because we have stipulated that $|A| > |B|$, the condition $M \leq \min(|A|, |B|)$ implies $M \leq |B|$. But this contradicts the above result, so the only way for $\max(|A|, |B|) = M$ to hold is if $|A| = |B|$. This, in turn, implies that A and B must be identical strings ($A \equiv B$) because $|A| = |B| = M$ means that all symbols of A and B match. The converse result that if $A \equiv B$ then $Q = 0$ follows directly from the definition of Q.

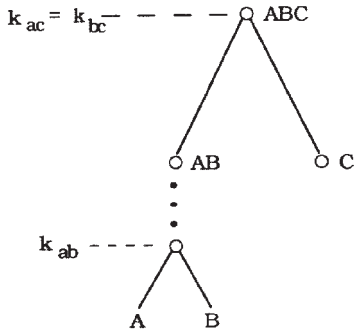


Figure P12.20

Problem 12.22

There are various possible approaches to this problem, and our students have shown over the years a tendency to surprise us with new and novel approaches to problems of this type. We give here a set of guidelines that should be satisfied by most practical solutions, and also offer suggestions for specific solutions to various parts of the problem. Depending on the level of maturity of the class, some of these may be offered as "hints" when the problem is assigned.

Because speed and cost are essential system specifications, we conceptualize a binary approach in which image acquisition, preprocessing, and segmentation are combined into one basic operation. This approach leads us to global thresholding as the method of choice. In this particular case this is possible because we can solve the inspection problem by concentrating on the white parts of the flag (stars and white stripes). As discussed in Chapter 10, uniform illumination is essential, especially when global thresholding is used for segmentation. The student should mention something about uniform illumination, or compensation for nonuniform illumination. A discussion by the student of color filtering to improve contrast between white and (red/blue/background) parts of an image is a plus in the design.

The first step is to specify the size of the viewing area, and the resolution required to detect the smallest components of interest, in this case the stars. Because the images are moving and the exact location of each flag is not known, it is necessary to specify a field of view that will guarantee that every image will contain at least one complete flag. In addition, the frame rate must be fast enough so that no flags are missed. The first part of the problem is easy to solve. The field of view has to be wide enough to encompass an area slightly greater across than two flags plus the maximum separation between them. Thus, the width, W, of the viewing area must be at least $W = 2(5) + 2.05 = 12.1$in. If we use

a standard CCD camera of resolution 640×480 elements and view an area 12.8 in. wide, this will give us a sampling rate of approximately 50 pixels/inch, or 250 pixels across a single flag. Visual inspection of a typical flag will show that the blue portion of a flag occupies about 0.4 times the length of the flag, which in this case gives us about 100 pixels per line in the blue area. There is a maximum of six stars per line, and the blue space between them is approximately 1.5 times the width of a star, so the number of pixels across a star is $100/([1+1.5]\times 6) \simeq 6$ pixels/star.

The next two problems are to determine the shutter speed and the frame rate. Because the number of pixels across each object of interest is only 6, we fix the blur at less than one pixel. Following the approach used in the solution of Problem 10.49, we first determine the distance between pixels as $(12.8\,\text{in})/640\,\text{pixels} = 0.02\,\text{in/pixel}$. The maximum speed of the flags is 21 in/sec. At this speed, the flags travel $21/0.02 = 1,050$ pixels/sec. We are requiring that a flag not travel more than one pixel during exposure; that is $(1,050\,\text{pixels/sec}) \times T\,\text{sec} \leq 1\,\text{pixel}$. So, $T \leq 9.52 \times 10^{-4}$ sec is the shutter speed needed.

The frame rate must be fast enough to capture an image of every flag that passes the inspection point. It takes a flag $(21\,\text{in/sec})/(12.8\,\text{in}) \simeq 0.6$ sec to cross the entire field of view, so we have to capture a frame every 0.3 sec in order to guarantee that every image will contain a whole flag, and that no flag will be missed. We assume that the camera is computer controlled to fire from a clock signal. We also make the standard assumption that it takes $1/30\,\text{sec} \simeq 330 \times 10^{-4}$ sec to read a captured image into a frame buffer. Therefore, the total time needed to acquire an image is $(330+9.5)\times 10^{-4} \simeq 340 \times 10^{-4}$ sec. Subtracting this quantity from the 0.3 sec frame rate leaves us with about 0.27 sec to do all the processing required for inspection, and to output an appropriate signal to some other part of the manufacturing process.

Because a global thresholding function can be incorporated in most digitizers as part of the data acquisition process, no additional time is needed to generate a binary image. That is, we assume that the digitizer outputs the image in binary form. The next step is to isolate the data corresponding to a complete flag. Given the imaging geometry and frame rate discussed above, four basic binary image configurations are expected: (1) part of a flag on the left of the image, followed by a whole flag, followed by another partial flag; (2) one entire flag touching the left border, followed by a second entire flag, and then a gap before the right border; (3) the opposite of (2); and (4) two entire flags, with neither flag touching the boundary of the image. Cases (2), (3), and (4) are not likely to occur with any significant frequency, but we will check for each of these conditions. As will be seen below, Cases (2) and (3) can be handled the same as Case (1), but, given the tight bounds on processing time, the output each time Case (4) occurs

will be to reject both flags.

To handle Case (1) we have to identify a whole flag lying between two partial flags. One of the quickest ways to do this is to run a window as long as the image vertically, but narrow in the horizontal direction, say, corresponding to 0.35 in. (based on the window size 1/2 of $[12.8 - 12.1]$), which is approximately $(0.35)(640)/12.8 \simeq 17$ pixels wide. This window is used look for a significant gap between a high count of 1's, and it is narrow enough to detect Case (4). For Case (1), this approach will produce high counts starting on the left of the image, then drop to very few counts (corresponding to the background) for about two inches, pick up again as the center (whole flag) is encountered, go like this for about five inches, drop again for about two inches as the next gap is encountered, then pick up again until the right border is encountered. The 1's between the two inner gaps correspond to a complete flag and are processed further by the methods discussed below; the other 1's are ignored. (A more elegant and potentially more rugged way is to determine all connected components first, and then look for vertical gaps, but time and cost are fundamental here). Cases (2) and (3) are handled in a similar manner with slightly different logic, being careful to isolate the data corresponding to an entire flag (i.e., the flag with a gap on each side). Case (4) corresponds to a gap-data-gap-data-gap sequence, but, as mentioned above, it is likely that time and cost constraints would dictate rejecting both flags as a more economical approach than increasing the complexity of the system to handle this special case. Note that this approach to extracting 1's is based on the assumption that the background is not excessively noisy. In other words, the imaging system must be such that the background is reliably segmented as black, with acceptable noise.

With reference to Fig. 1.23, the preceding discussion has carried us through the segmentation stage. The approach followed here for description, recognition, and the use of knowledge, is twofold. For the stars we use connected component analysis. For the stripes we use signature analysis. The system knows the coordinates of two vertical lines which contain the whole flag between them. First, we do a connected components analysis on the left half of the region (to save time) and filter out all components smaller and larger than the expected size of stars, say (to give some flexibility), all components less than 9 (3×3) pixels and larger than 64 (8×8) pixels. The simplest test at this point is to count the number of remaining connected components (which we assume to be stars). If the number is 50 we continue with the next test on the stripes. Otherwise, we reject the flag. Of course, the logic can be made much more complicated than this. For instance, it could include a regularity analysis in which the relative locations of the components are analyzed. There are likely to be as many answers here as there are students in the class, but the key objective should be to base

the analysis on a rugged method such as connected component analysis.

To analyze the stripes, we assume that the flags are printed on white stock material. Thus, "dropping a stripe" means creating a white stripe twice as wide as normal. This is a simple defect detectable by running a vertical scan line in an area guaranteed to contain stripes, and then looking at the intensity signature for the number of pulses of the right height and duration. The fact that the data is binary helps in this regard, but the scan line should be preprocessed to bridge small gaps due to noise before it is analyzed. In spite of the $\pm 15°$ variation in direction, a region, say, 1 in. to the right of the blue region is independent enough of the rotational variation in terms of showing only stripes along a scan line run vertically in that region.

It is important that any answer to this problem show awareness of the limits in available computation time. Because no mention is made in the problem statement about available processors, it is not possible to establish with absolute certainty if a solution will meet the requirements or not. However, the student should be expected to address this issue. The guidelines given in the preceding solution are among the fastest ways to solve the problem. A solution along these lines, and a mention that multiple systems may be required if a single system cannot meet the specifications, is an acceptable solution to the problem.