ECEN 642, Fall 2019

Texas A&M University

Electrical and Computer Engineering Department

Dr. Raffaella Righetti

Due: 10/25/2019 (before class)

Assignment #4

Gonzalez and Woods (4th edition), projects: 4.1, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8

Gonzalez and Woods (4th edition), problems: 4.4, 4.6, 4.9, 4.24, 4.25, 4.51

Please clearly indicate your processing parameters (if there is any) in your solutions

For the assignments, you will need to use MATLAB. You can access it on campus through the open access lab (OAL) or remotely through the virtual open access lab (VOAL). The link below will guide you into configuring the Horizon client for VOAL remote connection step by step on your PC or mac:

https://tamu.service-now.com/tamu-selfservice/knowledge_detail.do?sysparm_document_key=kb_knowledge,6f7e0c6adbce5f84778ff5961d96199f#

After you connect to the server, you will see the MATLAB icon. If you don't, you can connect to your VOAL desktop (VOAL icon) and start MATLAB from there.

For projects from the 4th edition, photo copies of the project statements are provided. For projects from the 3rd edition, you can access them via the link:
http://www.imageprocessingplace.com/DIP-3E/dip3e_student_projects.htm#02-04

Projects

MATLAB solutions to the projects marked with an asterisk (*) are in the DIP4E Student Support Package (consult the book website: www.ImageProcessingPlace.com).

4.1 Write a function **g = minusOne4e(f)** that multiplies **f** by $(-1)^{x+y}$ to produce **g**. Array **f** can be 1-D (row or column) or 2-D. The input image must be floating point, so your function should perform a validation check for this.

4.2 Implementation and testing of the 2-D FFT and its inverse using a 1-D FFT algorithm.

(a)* Obtain a routine that computes the 1-D FFT in the language you are using for projects. For example, excellent FFT implementations in C are available from www.fftw.org. If you are working in MATLAB, use function **fft**. Use the 1-D FFT routine to implement a function **F = dft2D4e(f)** that computes the 2-D forward FFT of image **f**, as explained in Section 4.11.

(b) Write a function **f = idft2D4e(F)** that computes the inverse FFT of an input transform **F**. (*Hint:* Work with the conjugate of **F** so that you can use the forward FFT function from (a) to compute the inverse, as explained in Section 4.11.)

(c) Read the image **rose512.tif** and scale it to the range [0,1] using the default settings of function **intScaling4e**. Denote the result by **f**. Test your functions by (1) computing **F**, the forward FFT of **f**, and (2) obtaining **g**, the real part of the inverse FFT of **F**. Display **f**, **g**, and the difference, **d = f - g**, of the two. Display the maximum and minimum values of **d**. The displays of **f** and **g** should look identical, and **d** should appear as a black image.

(d)* Compute the centered transform and display the spectrum of **F** as **S = log(1 + abs(F))**. Scale **S** using the **'full'** option in function **intScaling4e** before displaying it.

4.3 Lowpass filter transfer functions.

(a)* Write a function **H = lpFilterTF4e(type,P,Q,param)** to generate a **P** × **Q** lowpass filter transfer function, **H**, with the following properties. If **type** = **'ideal'**, **param** should be a scalar equal to the cut-off frequency D_0 in Eq. (4-111). If **type** = **'gaussian'**, **param** should be a scalar equal to the standard deviation D_0 in Eq. (4-116). If **type** = **'butterworth'**, **param** should be a 1×2 array (vector) containing the cutoff frequency and filter order, $[D_0, n]$, in Eq. (4-117).

(b)* Generate a lowpass ideal filter transfer function of size 512×512 with $D_0 = 96$. Display your result as an image.

(c) Generate a lowpass Gaussian filter transfer function of size 512×512 with $D_0 = 96$. Display your result as an image.

(d) Generate a lowpass Butterworth filter transfer function of size 512×512. Choose $D_0 = 96$ and $n = 2$. Display your result as an image.

4.4 Highpass filter transfer functions.

(a)* Write a function **H = hpFilterTF4e(type,P,Q,param)** to generate a **P** × **Q** highpass filter transfer function, **H**, with the following properties. If **type** = **'ideal'**, **param** should be a scalar equal to the cut-off frequency D_0 in Eq. (4-119). If **type** = **'gaussian'**, **param** should be a scalar equal to the standard deviation D_0 in Eq. (4-120). If **type** = **'butterworth'**, **param** should be a 1×2 array (vector) the cutoff frequency and filter order, $[D_0, n]$, in Eq. (4-121).

(b)* Generate an ideal highpass filter transfer function of size 512×512 with $D_0 = 96$. Display your result as an image.

(c) Generate a highpass Gaussian filter transfer function of size 512×512 with $D_0 = 96$. Display your result as an image.

(d) Generate a highpass Butterworth filter transfer function of size 512×512. Choose $D_0 = 96$ and $n = 2$.

4.5 Frequency domain filtering package.

(a)* Write a function, **g = dftFiltering4e(f,H,padmode scaling)** to filter image **f** with a given filter transfer function **H**. Your function should implement the seven steps in the filtering algorithm discussed in Section 4.7. **If padmode = 'replicate'** or is not included in the argument, then replicate padding should be used. If **padmode = 'zeros'**, zero padding should be used.

The padding used in either case should be of the **'post'** type discussed in project function **imPad4e** from Chapter 3. No padding should be used if **padmode = 'none'**.

(b)* To test your function, read the image **testpattern512.tif**, filter it with a Butterworth lowpass filter with cutoff a frequency of 32 and order 2. Display the result.

4.6 Lowpass filtering in the frequency domain. Show the image resulting after each of the following:

(a) Read the image **testpattern1024.tif** and lowpass filter it using a Gaussian filter so that the large letter "a" is barely readable, and the other letters are not.

(b)* Read the image **testpattern1024.tif**. Lowpass filter it using a Butterworth filter of your specification so that, when thresholded, the filtered image contains only part of the large square on the top, right. (*Hint:* It is more intuitive to work with the negative of the original image.)

(c) Read the image **checkerboard1024-shaded.tif** and reproduce the results in Example 3.18 using frequency-domain filtering. (*Hint:* To obtain images like the ones in the example, scale the shading pattern and the processed image to the full [0,1] intensity range—you can use project function **intScaling4e** for this.)

4.7 Unsharp masking and high-boost filtering in the frequency domain.

(a)* Read the image **blurry-moon.tif** and sharpen it using unsharp masking. Use a Gaussian lowpass filter of your choice for the blurring step. Display your final result.

(b) Improve the sharpness of your result using high-boost filtering. Display the final result. [You should be able to achieve a result close to Fig. 4.56(c).]

4.8 Highpass filtering in the frequency domain.

(a) Read the image **thumbprint.tif** and duplicate the results Example 4.20. (*Hint:* Use the **'no'** scaling option in function **dftFiltering4e** so that negative values are preserved.)

(b)* Write a function **H = laplacianTF4e(P,Q)** that computes the Laplacian filter transfer function, **H**, in the frequency domain. **P** and **Q** are positive integers.

(c)* Read the image **blurry-moon.tif** and sharpen it in the frequency domain using the Laplacian transfer function from (b) (see Example 4.21). Display your result. [*Hint:* Use the **'no'** scaling option in function **dftFiltering4e** so that negative values are preserved for proper scaling, as discussed in connection with Eq. (4-126).]

(d) Read the image **chestXray.tif** and duplicate the high-frequency-emphasis results given in Example 4.22.

4.9 Bandreject and bandpass filtering.

(a)* With reference to Table 4.7, write a function **H = brFilterTF4e(type,M,N,C0,W,n)** for performing bandreject filtering, where **type** refers to one of the three categories in the table, **M** and **N** are the usual image dimensions, and the other parameters are as explained in Table 4.7.

(b) Use the results from (a) to write a bandpass function, **H = bpFilterTF4e(type,M,N,C0,W,n)**, where the parameters are as explained in (a).

(c)* Generate a 512×512 ideal bandreject filter transfer function of your choice and display it as an image.

(d) Generate a 512×512 Gaussian bandreject filter transfer function using the same parameters as in (c) and show your result as an image.

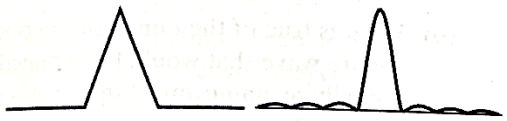
(e) Generate a 512×512 Butterworth bandreject filter transfer function using the same parameters as in (c) and $n = 1$. Show your result as an image.

(f) **through (h)** Repeat (c)-(e) but for bandpass filter transfer functions instead.

4.10 The objective of this project is to duplicate the results in Fig. 3.62 (Example 3.23) using frequency domain filtering. The image used is **zoneplate.tif**. For consistency scale the intensity range of this image to [0,1] using MATLAB function **im2double** or project function **intScaling4e** with the default setting. To simplify experimenting, do the filtering without padding. (*Hint:* To separate frequency bands requires filters with sharp cutoffs, so use Butterworth filter transfer functions with $n = 3$ throughout.)

4.3 Repeat Example 4.1, but using the function $f(t) = A$ for $0 \le t < T$ and $f(t) = 0$ for all other values of t. Explain the reason for any differences between your results and the results in the example.

4.4 As the figure below shows, the Fourier transform of a "tent" function (on the left) is a squared sinc function (on the right). Advance an argument that shows that the Fourier transform of a tent function can be obtained from the Fourier transform of a box function. (*Hint:* The tent itself can be generated by convolving two equal boxes.)



4.5 What is the convolution of two, 1-D impulses:

(a)* $\delta(t)$ and $\delta(t - t_0)$?

(b) $\delta(t - t_0)$ and $\delta(t + t_0)$?

4.6* Use the sifting property of the impulse to show that convolving a 1-D continuous function, $f(t)$, with an impulse located at t_0 shifts the function so that its origin is moved to the location of the impulse (if the impulse is at the origin, the function is not shifted).

4.7* With reference to Fig. 4.9, give a graphical illustration of an aliased pair of functions that are not periodic.

4.8 Do two or more functions have to be band-limited for them to possibly be aliased? Explain.

4.9 With reference to Fig. 4.11:

(a)* Redraw the figure, showing what the dots would look like for a sampling rate that exceeds the Nyquist rate slightly.

(b) What is the *approximate* sampling rate represented by the large dots in Fig. 4.11?

(c) *Approximately*, what would be the lowest sampling rate that you would use so that (1) the Nyquist rate is satisfied, and (2) the samples look like a sine wave?

4.10 A function, $f(t)$, is formed by the sum of three functions, $f_1(t) = A\sin(\pi t)$, $f_2(t) = B\sin(4\pi t)$, and $f_3(t) = C\cos(8\pi t)$.

(a) Assuming that the functions extend to infinity in both directions, what is the highest frequency of $f(t)$? (*Hint:* Start by finding the period of the sum of the three functions.)

(b)* What is the Nyquist rate corresponding to your result in (a)? (Give a numerical answer.)

(c) At what rate would you sample $f(t)$ so that perfect recovery of the function from its samples is possible?

4.11* Show that $\Im\{e^{j2\pi t_0 t}\} = \delta(\mu - t_0)$, where t_0 is a constant. (*Hint:* Study Example 4.2.)

4.12 Show that the following expressions are true. (*Hint:* Make use of the solution to Problem 4.11):

(a)* $\Im\{\cos(2\pi\mu_0 t)\} = \frac{1}{2}[\delta(\mu - \mu_0) + \delta(\mu + \mu_0)]$

(b) $\Im\{\sin(2\pi\mu_0 t)\} = \frac{1}{2j}[\delta(\mu - \mu_0) - \delta(\mu + \mu_0)]$

4.13 Consider the function $f(t) = \sin(2\pi n t)$, where n is an integer. Its Fourier transform, $F(\mu)$, is purely imaginary (see Problem 4.12). Because the transform, $\tilde{F}(\mu)$, of sampled data consists of periodic copies of $F(\mu)$, it follows that $\tilde{F}(\mu)$ will also be purely imaginary. Draw a diagram similar to Fig. 4.6, and answer the following questions based on your diagram (assume that sampling starts at $t = 0$).

(a)* What is the period of $f(t)$?

(b)* What is the frequency of $f(t)$?

(c)* What would the sampled function and its Fourier transform look like in general if $f(t)$ is sampled at a rate higher than the Nyquist rate?

(d) What would the sampled function look like in general if $f(t)$ is sampled at a rate lower than the Nyquist rate?

(e) What would the sampled function look like if $f(t)$ is sampled at the Nyquist rate, with samples taken at $t = 0, \pm\Delta T, \pm 2\Delta T, \ldots$?

4.14* Prove the validity of the convolution theorem of one continuous variable, as given in Eqs. (4-25) and (4-26).

4.15 We explained in the paragraph after Eq. (4-36) that arbitrarily limiting the duration of a band-limited function by multiplying it by a box function would cause the function to cease being band limited. Show graphically why this is so by limit-

ing the duration of the function $f(t) = \cos(2\pi\mu_0 t)$ [the Fourier transform of this function is given in Problem 4.12(a)]. (*Hint:* The transform of a box function is given in Example 4.1. Use that result in your solution, and also the fact that convolution of a function with an impulse shifts the function to the location of the impulse, in the sense discussed in the solution of Problem 4.6.)

4.16* Complete the steps that led from Eq. (4-37) to Eq. (4-38).

4.17 Show that $\tilde{F}(\mu)$ in Eq. (4-40) is infinitely periodic in both directions, with period $1/\Delta T$.

4.18 Do the following:

(a) Show that Eqs. (4-42) and (4-43) are a Fourier transform pair: $f_n \Leftrightarrow F_m$.

(b)* Show that Eqs. (4-44) and (4-45) also are a Fourier transform pair: $f(x) \Leftrightarrow F(u)$.

You will need the following orthogonality property in both parts of this problem:

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

4.19 Show that both $F(u)$ and $f(x)$ in Eqs. (4-44) and (4-45) are infinitely periodic with period M; that is, $F(u) = F(u + kM)$ and $f(x) = f(x + M)$, where k is an integer. [See Eqs. (4-46) and (4-47).]

4.20 Demonstrate the validity of the translation (shift) properties of the following 1-D, discrete Fourier transform pairs. (*Hint:* It is easier in part (b) to work with the IDFT.)

(a)* $f(x)e^{j2\pi u_0 x/M} \Leftrightarrow F(u - u_0)$

(b) $f(x - x_0) \Leftrightarrow F(u)e^{-j2\pi u x_0/M}$

4.21 Show that the 1-D convolution theorem given in Eqs. (4-25) and (4-26) also holds for discrete variables, but with the right side of Eq. (4-26) multiplied by $1/M$. That is, show that

(a)* $(f \star h)(x) \Leftrightarrow (F \cdot H)(u)$, and

(b) $(f \cdot h)(x) \Leftrightarrow \frac{1}{M}(F \star H)(u)$

4.22* Extend the expression for 1-D convolution [see Eq. (4-24)] to two continuous variables. Use t and z for the variables on the left side of the expression and α and β for the variables in the 2-D integral.

4.23 Use the sifting property of the 2-D impulse to show that convolution of a 2-D continuous function, $f(t,z)$, with an impulse shifts the function so that its origin is located at the location of the impulse. (If the impulse is at the origin, the function is copied exactly as it was.) (*Hint:* Study the solution to Problem 4.6).

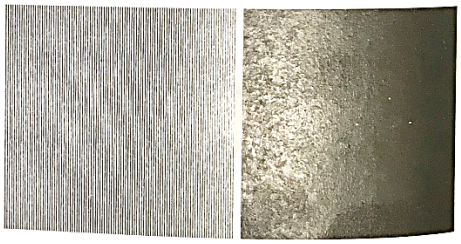
4.24 Answer the following:

(a)* Let K denote the sides of the squares in the 96×96-pixel image in Example 4.6. What are all the valid values of K that will yield images free of aliasing?

(b) What is true of the number of periods of the square wave that would be obtained by horizontally scanning any of the images resulting from (a)?

(c) Consider a checkerboard image in which each square is of size 1×1 mm. Assuming that the image extends infinitely in both coordinate directions, what is the minimum sampling rate (in samples/mm) required to avoid aliasing?

4.25 The image on the left in the figure below consists of alternating stripes of black/white, each stripe

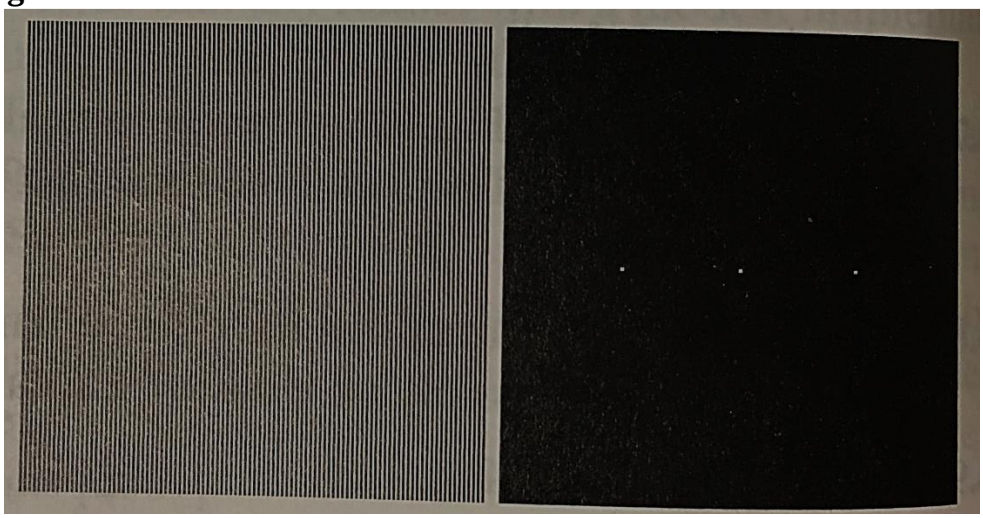


being two pixels wide. The image on the right is the Fourier spectrum of the image on the left, showing the dc term and the frequency terms corresponding to the stripes. (Remember, the spectrum is symmetric so all components, other than the dc term, appear in two symmetric locations.)

(a)* Suppose that the stripes of an image of the same size are four pixels wide. Sketch what the spectrum of the image would look like, including only the dc term and the two highest-value frequency terms, which correspond to the two spikes in the spectrum above.

(b) Why are the components of the spectrum limited to the horizontal axis?

4.25 figure



4.51* Consider a 3×3 spatial kernel that averages the four closest neighbors of a point (x, y), but excludes the point itself from the average.

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

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

Figures & formulas

IDEAL LOWPASS FILTERS

A 2-D lowpass filter that passes without attenuation all frequencies within a circle of radius from the origin, and "cuts off" all frequencies outside this, circle is called an *ideal lowpass filter* (ILPF); it is specified by the transfer function

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases} \tag{4-111}$$

where D_0 is a positive constant, and $D(u,v)$ is the distance between a point (u,v) in the frequency domain and the center of the $P \times Q$ frequency rectangle; that is,

$$D(u,v) = \left[(u - P/2)^2 + (v - Q/2)^2\right]^{1/2} \tag{4-112}$$

GAUSSIAN LOWPASS FILTERS

Gaussian lowpass filter (GLPF) transfer functions have the form

$$H(u,v) = e^{-D^2(u,v)/2\sigma^2} \tag{4-115}$$

where, as in Eq. (4-112), $D(u,v)$ is the distance from the center of the $P \times Q$ frequency rectangle to any point, (u,v), contained by the rectangle. Unlike our earlier expressions for Gaussian functions, we do not use a multiplying constant here in order to be consistent with the filters discussed in this and later sections, whose highest value is 1. As before, σ is a measure of spread about the center. By letting $\sigma = D_0$, we can express the Gaussian transfer function in the same notation as other functions in this section:

$$H(u,v) = e^{-D^2(u,v)/2D_0^2} \tag{4-116}$$

where D_0 is the cutoff frequency. When $D(u,v) = D_0$, the GLPF transfer function is down to 0.607 of its maximum value of 1.0.

From Table 4.4, we know that the inverse Fourier transform of a frequency-domain Gaussian function is Gaussian also. This means that a spatial Gaussian filter kernel, obtained by computing the IDFT of Eq. (4-115) or (4-116), will have no ringing. As property 13 of Table 4.4 shows, the same inverse relationship explained earlier for ILPFs is true also of GLPFs. Narrow Gaussian transfer functions in the frequency domain imply broader kernel functions in the spatial domain, and vice

BUTTERWORTH LOWPASS FILTERS

The transfer function of a Butterworth lowpass filter (BLPF) of order n, with cutoff frequency at a distance D_0 from the center of the frequency rectangle, is defined as

$$H(u,v) = \frac{1}{1 + \left[D(u,v)/D_0\right]^{2n}} \tag{4-117}$$

where $D(u,v)$ is given by Eq. (4-112). Figure 4.45 shows a perspective plot, image display, and radial cross sections of the BLPF function. Comparing the cross section plots in Figs. 4.39, 4.43, and 4.45, we see that the BLPF function can be controlled to approach the characteristics of the ILPF using higher values of n, and the GLPF for lower values of n, while providing a smooth transition in from low to high frequencies. Thus, we can use a BLPF to approach the sharpness of an ILPF function with considerably less ringing.

IDEAL, GAUSSIAN, AND BUTTERWORTH HIGHPASS FILTERS FROM LOWPASS FILTERS

As was the case with kernels in the spatial domain (see Section 3.7), subtracting a lowpass filter transfer function from 1 yields the corresponding highpass filter transfer function in the frequency domain:

$$H_{HP}(u,v) = 1 - H_{LP}(u,v) \tag{4-118}$$

where $H_{LP}(u,v)$ is the transfer function of a lowpass filter. Thus, it follows from Eq. (4-111) that an ideal highpass filter (IHPF) transfer function is given by

$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \le D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases} \tag{4-119}$$

where, as before, $D(u,v)$ is the distance from the center of the $P \times Q$ frequency rectangle, as given in Eq. (4-112). Similarly, it follows from Eq. (4-116) that the transfer function of a Gaussian highpass filter (GHPF) transfer function is given by

$$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2} \tag{4-120}$$

and, from Eq. (4-117), that the transfer function of a Butterworth highpass filter (BHPF) is

$$H(u,v) = \frac{1}{1 + \left[D_0/D(u,v)\right]^{2n}} \tag{4-121}$$

Figure 4.51 shows 3-D plots, image representations, and radial cross sections for the preceding transfer functions. As before, we see that the BHPF transfer function in the third row of the figure represents a transition between the sharpness of the IHPF and the broad smoothness of the GHPF transfer function.

It follows from Eq. (4-118) that the spatial kernel corresponding to a highpass filter transfer function in the frequency domain is given by

SUMMARY OF STEPS FOR FILTERING IN THE FREQUENCY DOMAIN

The process of filtering in the frequency domain can be summarized as follows:

1. Given an input image $f(x,y)$ of size $M \times N$, obtain the padding sizes P and Q using Eqs. (4-102) and (4-103); that is, $P = 2M$ and $Q = 2N$.
2. Form a padded† image $f_p(x,y)$ of size $P \times Q$ using zero-, mirror-, or replicate padding (see Fig. 3.45 for a comparison of padding methods).
3. Multiply $f_p(x,y)$ by $(-1)^{x+y}$ to center the Fourier transform on the $P \times Q$ frequency rectangle.
4. Compute the DFT, $F(u,v)$, of the image from Step 3.
5. Construct a real, symmetric filter transfer function, $H(u,v)$, of size $P \times Q$ with center at $(P/2, Q/2)$.
6. Form the product $G(u,v) = H(u,v)F(u,v)$ using elementwise multiplication; that is, $G(i,k) = H(i,k)F(i,k)$ for $i = 0, 1, 2, \ldots, M-1$ and $k = 0, 1, 2, \ldots, N-1$.
7. Obtain the filtered image (of size $P \times Q$) by computing the IDFT of $G(u,v)$:

$$g_p(x,y) = \left(\text{real}\left[\Im^{-1}\{G(u,v)\}\right]\right)(-1)^{x+y}$$

8. Obtain the final filtered result, $g(x,y)$, of the same size as the input image, by extracting the $M \times N$ region from the top, left quadrant of $g_p(x,y)$.

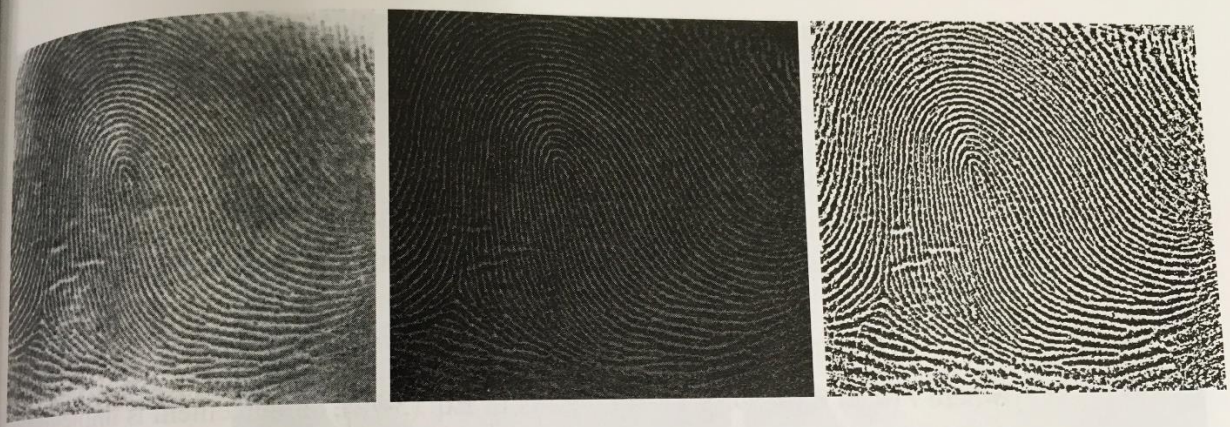
a b

FIGURE 4.56 (a) Original, blurry image. (b) Image enhanced using the Laplacian in the frequency domain. Compare with Fig. 3.52(d). (Original image courtesy of NASA.)

encompasses a very small neighborhood, while the formulation in Eqs. (4-125) and (4-126) encompasses the entire image.

EXAMPLE 4.20: Using highpass filtering and thresholding for image enhancement.

Figure 4.55(a) is a 962×1026 image of a thumbprint in which smudges (a typical problem) are evident. A key step in automated fingerprint recognition is enhancement of print ridges and the reduction of smudges. In this example, we use highpass filtering to enhance the ridges and reduce the effects of



a b c

FIGURE 4.55 (a) Smudged thumbprint. (b) Result of highpass filtering (a). (c) Result of thresholding (b). (Original image courtesy of the U.S. National Institute of Standards and Technology.)

smudging. Enhancement of the ridges is accomplished by the fact that their boundaries are characterized by high frequencies, which are unchanged by a highpass filter. On the other hand, the filter reduces low frequency components, which correspond to slowly varying intensities in the image, such as the background and smudges. Thus, enhancement is achieved by reducing the effect of all features except those with high frequencies, which are the features of interest in this case.

Figure 4.55(b) is the result of using a Butterworth highpass filter of order 4 with a cutoff frequency of 50. A fourth-order filter provides a sharp (but smooth) transition from low to high frequencies, with filtering characteristics between an ideal and a Gaussian filter. The cutoff frequency chosen is about 5% of the long dimension of the image. The idea is for D_0 to be close to the origin so that low frequencies are attenuated but not completely eliminated, except for the DC term which is set to 0, so that tonality differences between the ridges and background are not lost completely. Choosing a value for D_0 between 5% and 10% of the long dimension of the image is a good starting point. Choosing a large value of D_0 would highlight fine detail to such an extent that the definition of the ridges would be affected. As expected, the highpass filtered image has negative values, which are shown as black by the display.

A simple approach for highlighting sharp features in a highpass-filtered image is to threshold it by setting to black (0) all negative values and to white (1) the remaining values. Figure 4.55(c) shows the result of this operation. Note how the ridges are clear, and how the effect of the smudges has been reduced considerably. In fact, ridges that are barely visible in the top, right section of the image in Fig. 4.55(a) are nicely enhanced in Fig. 4.55(c). An automated algorithm would find it much easier to follow the ridges on this image than it would on the original.

THE LAPLACIAN IN THE FREQUENCY DOMAIN

In Section 3.6, we used the Laplacian for image sharpening in the spatial domain. In this section, we revisit the Laplacian and show that it yields equivalent results using frequency domain techniques. It can be shown (see Problem 4.56) that the Laplacian can be implemented in the frequency domain using the filter transfer function

$$H(u,v) = -4\pi^2(u^2 + v^2) \tag{4-123}$$

or, with respect to the center of the frequency rectangle, using the transfer function

$$\begin{aligned} H(u,v) &= -4\pi^2\left[(u-P/2)^2+(v-Q/2)^2\right] \\ &= -4\pi^2 D^2(u,v) \end{aligned} \tag{4-124}$$

where $D(u,v)$ is the distance function defined in Eq. (4-112). Using this transfer function, the Laplacian of an image, $f(x,y)$, is obtained in the familiar manner:

$$\nabla^2 f(x,y) = \Im^{-1}\left[H(u,v)F(u,v)\right] \tag{4-125}$$

where $F(u,v)$ is the DFT of $f(x,y)$. As in Eq. (3-77), enhancement is implemented using the equation

$$g(x,y) = f(x,y) + c\nabla^2 f(x,y) \tag{4-126}$$

Here, $c = -1$ because $H(u,v)$ is negative. In Chapter 3, $f(x,y)$ and $\nabla^2 f(x,y)$ had comparable values. However, computing $\nabla^2 f(x,y)$ with Eq. (4-125) introduces DFT scaling factors that can be several orders of magnitude larger than the maximum value of f. Thus, the differences between f and its Laplacian must be brought into comparable ranges. The easiest way to handle this problem is to normalize the values of $f(x,y)$ to the range $[0, 1]$ (before computing its DFT) and divide $\nabla^2 f(x,y)$ by its maximum value, which will bring it to the approximate range $[-1, 1]$. (Remember, the Laplacian has negative values.) Equation (4-126) can then be used.

We can write Eq. (4-126) directly in the frequency domain as

$$\begin{aligned} g(x,y) &= \Im^{-1}\left\{F(u,v) - H(u,v)F(u,v)\right\} \\ &= \Im^{-1}\left\{\left[1-H(u,v)\right]F(u,v)\right\} \\ &= \Im^{-1}\left\{\left[1+4\pi^2 D^2(u,v)\right]F(u,v)\right\} \end{aligned} \tag{4-127}$$

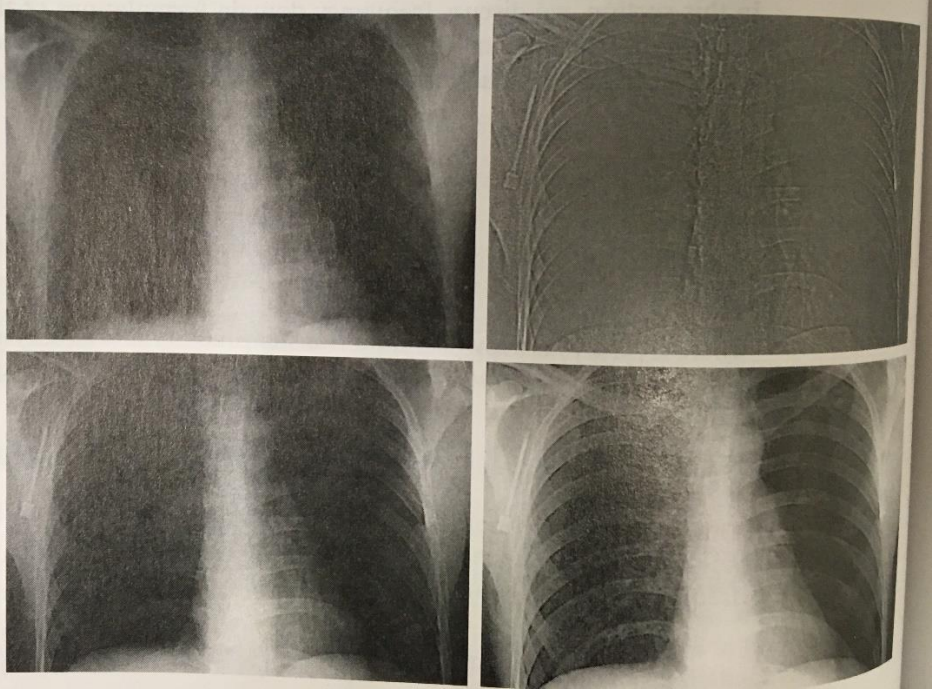
Although this result is elegant, it has the same scaling issues just mentioned, compounded by the fact that the normalizing factor is not as easily computed. For this reason, Eq. (4-126) is the preferred implementation in the frequency domain, with $\nabla^2 f(x,y)$ computed using Eq. (4-125) and scaled using the approach mentioned in the previous paragraph.

EXAMPLE 4.21: Image sharpening in the frequency domain using the Laplacian.

Figure 4.56(a) is the same as Fig. 3.54(a), and Fig. 4.56(b) shows the result of using Eq. (4-126), in which the Laplacian was computed in the frequency domain using Eq. (4-125). Scaling was done as described in connection with Eq. (4-126). We see by comparing Figs. 4.56(b) and 3.54(d) that the frequency-domain result is superior. The image in Fig. 4.56(b) is much sharper, and shows details that are barely visible in 3.54(d), which was obtained using the Laplacian kernel in Fig. 3.53(b), with a -8 in the center. The significant improvement achieved in the frequency domain is not unexpected. The spatial Laplacian kernel

EXAMPLE 4.22: Image enhancement using high-frequency-emphasis filtering.

Figure 4.57(a) shows a 503×720-pixel chest X-ray image with a narrow range of intensity levels. The objective of this example is to enhance the image using high-frequency-emphasis filtering. X-rays cannot be focused in the same manner that optical lenses can, and the resulting images generally tend to be slightly blurred. Because the intensities in this particular image are biased toward the dark end of the



a b
c d

FIGURE 4.57
(a) A chest X-ray. (b) Result of filtering with a GHPF function. (c) Result of high-frequency-emphasis filtering using the same GHPF. (d) Result of performing histogram equalization on (c). (Original image courtesy of Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School.)

gray scale, we also take this opportunity to give an example of how spatial domain processing can be used to complement frequency-domain filtering.

Image artifacts, such as ringing, are unacceptable in medical image processing, so we use a Gaussian highpass filter transfer function. Because the spatial representation of a GHPF function is Gaussian also, we know that ringing will not be an issue. The value chosen for D_0 should provide enough filtering to sharpen boundaries while at the same time not over-sharpening minute details (such as noise). We used $D_0 = 70$, approximately 10% of the long image dimension, but other similar values would work also. Figure 4.57(b) is the result of highpass filtering the original image (scaled as the images in Fig. 4.54). As expected, the image is rather featureless, but the important boundaries (e.g., the edges of the ribs) are clearly delineated. Figure 4.57(c) shows the advantage of high-frequency-emphasis filtering, where we used Eq. (4-133) with $k_1 = 0.5$ and $k_2 = 0.75$. Although the image is still dark, the gray-level tonality has been restored, with the added advantage of sharper features.

As we discussed in Section 3.3, an image characterized by intensity levels in a narrow range of the gray scale is an ideal candidate for histogram equalization. As Fig. 4.57(d) shows, this was indeed an appropriate method to further enhance the image. Note the clarity of the bone structure and other details that simply are not visible in any of the other three images. The final enhanced image is a little noisy, but this is typical of X-ray images when their gray scale is expanded. The result obtained using a combination of high-frequency-emphasis and histogram equalization is superior to the result that would be obtained by using either method alone.

EXAMPLE 4.6: Aliasing in images.

Consider an imaging system that is perfect, in the sense that it is noiseless and produces an exact digital image of what it sees, but the number of samples it can take is fixed at 96×96 pixels. For simplicity, assume that pixels are little squares of unit width and length. We want to use this system to digitize checkerboard images of alternating black and white squares. Checkerboard images can be interpreted as periodic, extending infinitely in both dimensions, where one period is equal to adjacent black/white pairs. If we specify "valid" digitized images as being those extracted from an infinite sequence in such a way that the image contains an integer multiple of periods, then, based on our earlier comments, we know that properly sampled periodic images will be free of aliasing. In the present example, this means that the sizes of the squares must be such that dividing 96 by the size yields an even number. This will give an integer number of periods (pairs of black/white squares). The smallest size of squares under the stated conditions is 1 pixel.

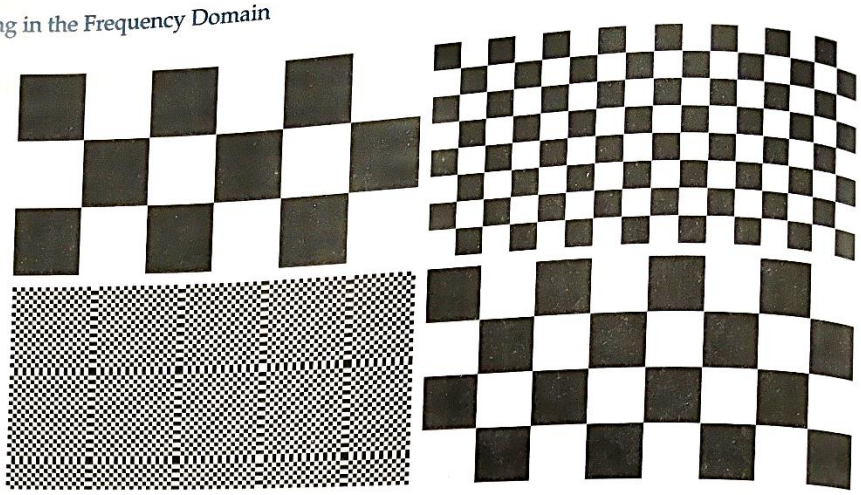
The principal objective of this example is to examine what happens when checkerboard images with squares of sizes less than 1 pixel on the side are presented to the system. This will correspond to the undersampled case discussed earlier, which will result in aliasing. A horizontal or vertical scan line of the checkerboard images results in a 1-D square wave, so we can focus the analysis on 1-D signals.

To understand the capabilities of our imaging system in terms of sampling, recall from the discussion of the 1-D sampling theorem that, given the sampling rate, the maximum frequency allowed before aliasing occurs in the sampled signal has to be less than one-half the sampling rate. Our sampling rate is fixed, at one sample per unit of the independent variable (the units are pixels). Therefore, the maximum frequency our signal can have in order to avoid aliasing is 1/2 cycle/pixel.

We can arrive at the same conclusion by noting that the most demanding image our system can handle is when the squares are 1 unit (pixel) wide, in which case the period (cycle) is two pixels. The frequency is the reciprocal of the period, or 1/2 cycle/pixel, as in the previous paragraph.

Figures 4.18(a) and (b) show the result of sampling checkerboard images whose squares are of sizes 16×16 and 6×6 pixels, respectively. The frequencies of scan lines in either direction of these two images are 1/32 and 1/6 cycles/pixel. These are well below the 1/2 cycles/pixel allowed for our system. Because, as mentioned earlier, the images are perfectly registered in the field of view of the system, the results are free of aliasing, as expected.

When the size of the squares is reduced to slightly less than one pixel, a severely aliased image results, as Fig. 4.18(c) shows (the squares used were approximately of size 0.95×0.95 pixels). Finally, reducing



a b
c d

FIGURE 4.18
Aliasing. In (a) and (b) the squares are of sizes 16 and 6 pixels on the side. In (c) and (d) the squares are of sizes 0.95 and 0.48 pixels, respectively. Each small square in (c) is one pixel. Both (c) and (d) are aliased. Note how (d) masquerades as a "normal" image.

the size of the squares to slightly less than 0.5 pixels on the side yielded the image in Fig. 4.18(d). In this case, the aliased result looks like a normal checkerboard pattern. In fact, this image would result from sampling a checkerboard image whose squares are 12 pixels on the side. This last image is a good reminder that aliasing can create results that may be visually quite misleading.

The effects of aliasing can be reduced by slightly defocusing the image to be digitized so that high frequencies are attenuated. As explained in Section 4.3, anti-aliasing filtering has to be done at the "front-end," *before* the image is sampled. There are no such things as after-the-fact software anti-aliasing filters that can be used to reduce the effects of aliasing caused by violations of the sampling theorem. Most commercial digital image manipulation packages do have a feature called "anti-aliasing." However, as illustrated in Example 4.8 below, this term is related to blurring a digital image to reduce additional aliasing artifacts caused by resampling. The term does not apply to reducing aliasing in the original sampled image. A significant number of commercial digital cameras have true anti-aliasing filtering built in, either in the lens or on the surface of the sensor itself. Even nature uses this approach to reduce the effects of aliasing in the human eye, as the following example shows.

EXAMPLE 4.7: Nature obeys the limits of the sampling theorem.

When discussing Figs. 2.1 and 2.2, we mentioned that cones are the sensors responsible for sharp vision. Cones are concentrated in the fovea, in line with the visual axis of the lens, and their concentration is measured in degrees off that axis. A standard test of visual acuity (the ability to resolve fine detail) in humans is to place a pattern of alternating black and white stripes in one degree of the visual field. If the total number of stripes exceeds 120 (i.e., a frequency of 60 cycles/degree), experimental evidence shows that the observer will perceive the image as a single gray mass. That is, the lens in the eye automatically lowpass filters spatial frequencies higher than 60 cycles/degree. Sampling in the eye is done by the cones, so, based on the sampling theorem, we would expect the eye to have on the order of 120 cones/degree in order to avoid the effects of aliasing. As it turns out, that is exactly what we have!