5

Discrete Transforms

5.1 Introduction and Overview

A transform is simply another term for a mathematical mapping process. Most of the transforms discussed in this chapter are used in image analysis and processing to provide information regarding the rate at which the gray levels change within an image—the spatial frequency or sequency. However, the principal components transform (PCT) is included and its primary purpose is to decorrelate the data between image bands. Additionally, the wavelet and the Haar transforms are different in that they retain both spatial and frequency information.

In general, a transform maps image data into a different mathematical space via a transformation equation. In Chapter 2, we discussed transforming image data into alternate color spaces. However, those color transforms mapped data from one color space to another color space with a one-to-one correspondence between a pixel in the input and the output. Basically, most of these transforms map the image data from the spatial domain to the frequency domain (also called the spectral domain), where *all* the pixels in the input (spatial domain) contribute to *each* value in the output (frequency domain). This is illustrated in Figure 5.1-1.

These transforms are used as tools in many areas of engineering and science, including digital image processing. Originally defined in their continuous forms, they are commonly used today in their discrete (sampled) forms. The large number of arithmetic operations required for the discrete transforms, combined with the massive amounts of data in an image, require a great deal of computer power. The ever-increasing computer power, memory capacity, and disk storage available today make the use of these transforms much more feasible than in the past.

The discrete form of these transforms is created by sampling the continuous form of the functions on which these transforms are based; that is, the *basis functions*. The functions used for these transforms are typically sinusoidal or rectangular, and the sampling process, for the one-dimensional (1-D) case, provides us with *basis vectors*. When we extend these into two-dimensions, as we do for images, they are *basis matrices* or *basis images* (see Figure 5.1-2). The process of transforming the image data into another domain, or mathematical space, amounts to projecting the image onto the basis images. The mathematical term for this projection process is an *inner product*, and is identical to what was done with Frei–Chen edge and line masks in Chapter 4. The Frei–Chen projections are performed to uncover edge and line information in the image, and use 3×3 image blocks. The frequency transforms considered here use the entire image, or blocks that are typically at least 8×8, and are used to discover spatial frequency information.

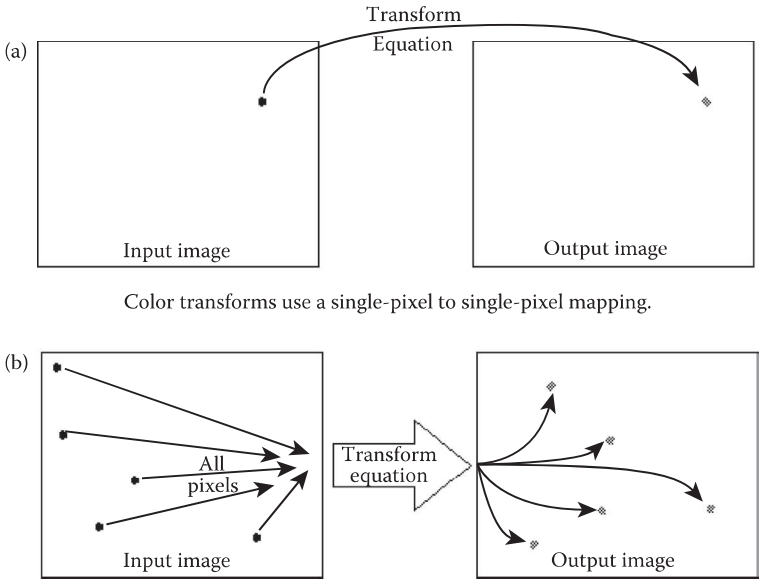


FIGURE 5.1-1
Discrete transforms.

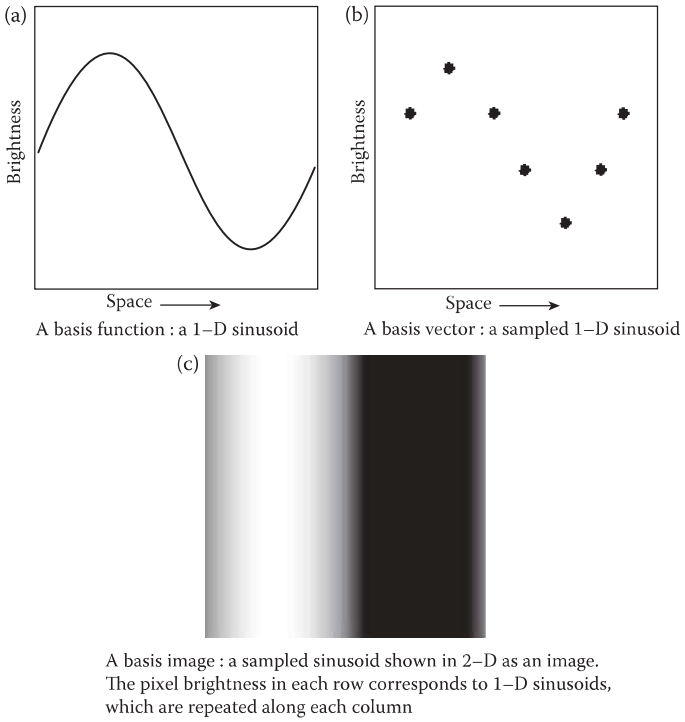


FIGURE 5.1-2
Basis vectors and images.

The ways in which the image brightness levels change in space define the spatial frequency. For example, rapidly changing brightness corresponds to high spatial frequency, whereas slowly changing brightness levels relate to low frequency information. The lowest spatial frequency, called the zero frequency term, corresponds to an image with a constant value. These concepts are illustrated in Figure 5.1-3, using square waves and sinusoids as basis vectors.

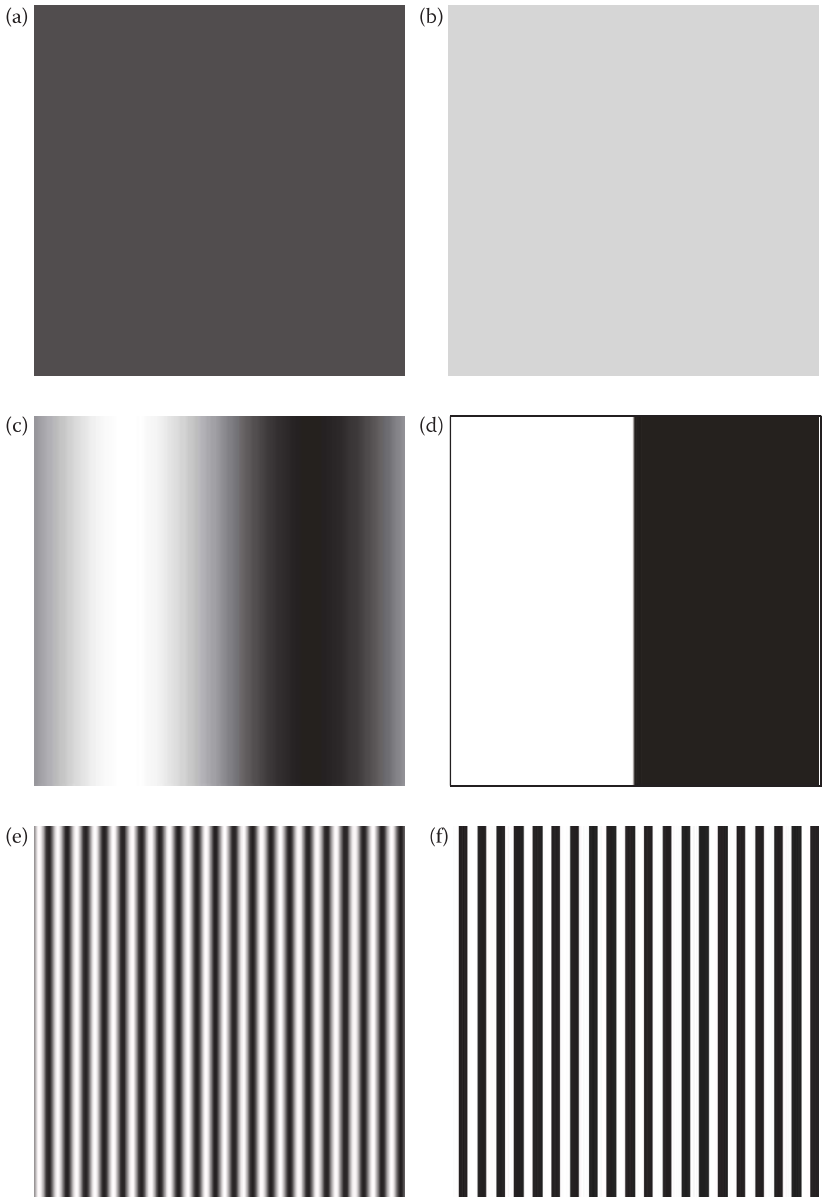


FIGURE 5.1-3
Spatial frequency. (a) frequency = 0, gray level = 51, (b) frequency = 0, gray level = 204, (c) frequency = 1, horizontal sine wave, (d) frequency = 1, horizontal square wave, (e) frequency = 20, horizontal sine wave, (f) frequency = 20, horizontal square wave.

The general form of the transformation equation, assuming an $M \times N$ image, is given by

$$T(u,v) = k \sum_{r=0}^{M-1} \sum_{c=0}^{N-1} I(r,c) B(r,c;u,v)$$

Here, u and v are the frequency domain variables, k is a constant that is transform dependent, $T(u,v)$ are the transform coefficients, and $B(r,c;u,v)$ correspond to the basis images. The notation $B(r,c;u,v)$ defines a set of basis images, corresponding to each different value for u and v, and the size of each is r by c (Figure 5.1-4). The transform coefficients, $T(u,v)$, are the projections of $I(r,c)$ onto each $B(u,v)$. This is illustrated in Figure 5.1-5.

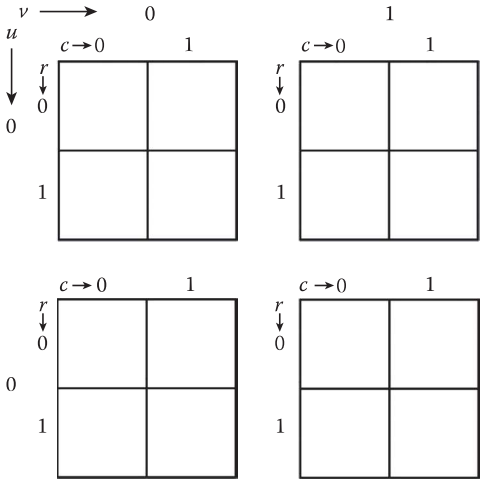


FIGURE 5.1-4
A set of basis images B(r,c;u,v). Size of generic basis images for a 2 × 2 transform.

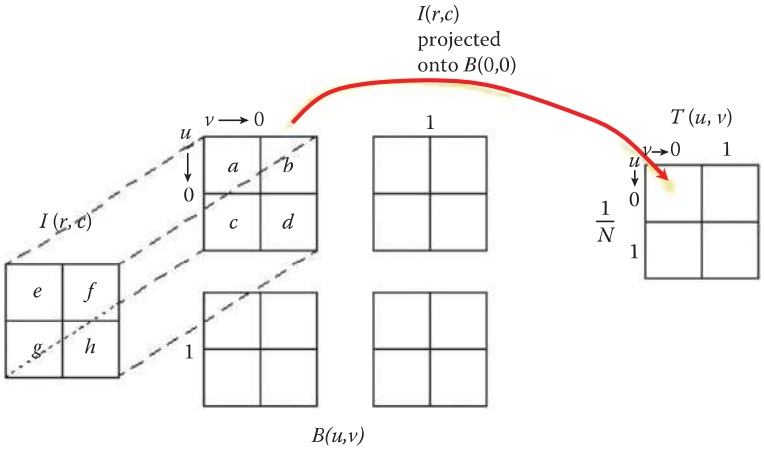


FIGURE 5.1-5
Transform coefficients. To find the transform coefficients, $T(u,v)$, we project the image, $I(r,c)$, onto the basis images, $B(u,v)$. For example, $T(0,0)$ is the projection of $I(r,c)$ onto $B(0,0)$, which equals $(ea + fb + gc + hd)$.

These coefficients tell us how similar the image is to the basis image; the more alike they are, the bigger the coefficient. This transformation process amounts to decomposing the image into a weighted sum of the basis images, where the coefficients $T(u,v)$ are the weights.

Example 5.1.1

Let $I(r,c) = \begin{bmatrix} 5 & 3 \\ 1 & 2 \end{bmatrix}$

And let $B(u,v,r,c) = \left\{ \begin{bmatrix} +1 & +1 \\ +1 & +1 \end{bmatrix} \begin{bmatrix} +1 & -1 \\ +1 & -1 \end{bmatrix} \right.$
$\left. \begin{bmatrix} +1 & +1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} +1 & -1 \\ -1 & +1 \end{bmatrix} \right\}$

Then $T(u,v) = \left\{ \begin{array}{l} T(0,0) = 5(1) + 3(1) + 1(1) + 2(1) = 11 \\ T(0,1) = 5(1) + 3(-1) + 1(1) + 2(-1) = 1 \\ T(1,0) = 5(1) + 3(1) + 1(-1) + 2(-1) = 5 \\ T(1,1) = 5(1) + 3(-1) + 1(-1) + 2(1) = 3 \end{array} \right\} = \begin{bmatrix} 11 & 1 \\ 5 & 3 \end{bmatrix}$

To obtain the image from the transform coefficients we apply the inverse transform equation:

$$I(r,c) = T^{-1}[T(u,v)] = k' \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} T(u,v)\, B^{-1}(r,c;u,v)$$

Here the $T^{-1}[T(u, v)]$ represents the inverse transform, and the $B^{-1}[(r, c, u, v)]$ represents the inverse basis images, and the k' is a constant that is transform dependent. In many cases, the inverse basis images are the same as the forward ones, and in cases where they are not, they are very similar.

Example 5.1.2

From the previous example, we have $T(u,v) = \begin{bmatrix} 11 & 1 \\ 5 & 3 \end{bmatrix}$

And let $B^{-1}(u,v,r,c) = \left\{ \begin{bmatrix} +1 & +1 \\ +1 & +1 \end{bmatrix} \begin{bmatrix} +1 & -1 \\ +1 & -1 \end{bmatrix} \right.$
$\left. \begin{bmatrix} +1 & +1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} +1 & -1 \\ -1 & +1 \end{bmatrix} \right\}$

$$\text{Then } I(r,c) = \begin{Bmatrix} I(0,0) = 11(1) + 1(1) + 5(1) + 3(1) = 20 \\ I(0,1) = 11(1) + 1(-1) + 5(1) + 3(-1) = 12 \\ I(1,0) = 11(1) + 1(1) + 5(-1) + 3(-1) = 4 \\ I(1,1) = 11(1) + 1(-1) + 5(-1) + 3(1) = 8 \end{Bmatrix} = \begin{bmatrix} 20 & 12 \\ 4 & 8 \end{bmatrix}?$$

$$\text{Is this correct? No, since } I(r,c) = \begin{bmatrix} 5 & 3 \\ 1 & 2 \end{bmatrix}$$

Comparing our results we see that we must multiply our answer by 1/4. What does this tell us?—It tells us that the transform pair, $B(u,v;r,c)$ and $B^{-1}(u,v;r,c)$ are not properly defined, we need to be able to recover our original image to have a proper transform pair. We can solve this by letting $k' = 1/4$, or by letting $k = k' = 1/2$. Note that 1/2 will normalize the magnitude of the basis images to 1. Remember that the magnitude of a vector can be found by the square root of the sum of the squares of the vector components; in this case:

Magnitude of the basis images = $\sqrt{1^2 + (\pm 1)^2 + (\pm 1)^2 + (\pm 1)^2} = \sqrt{4} = 2$. Therefore, to normalize the magnitude to 1, we need to divide by 2, or multiply by 1/2.

Two important attributes for basis images are that they be orthogonal and orthonormal. If basis images are *orthogonal*, it means the vector inner product of each one with every other one is equal to zero. Basis images that are *orthonormal* are orthogonal and have magnitudes equal to one. In the above example we saw why we want the basis images to have a magnitude of one, but what does orthogonality really mean and why is it important for basis images? Orthogonality means that the projection of one basis image onto another has a zero result—the two have nothing in common, they are uncorrelated. In Figure 5.1-6 we see an illustration of the vector inner product in a two-dimensional (2-D) mathematical (x,y) space. Given two vectors, $f_1(x_1,y_1)$ and $f_2(x_2,y_2)$, we can find the vector inner product by the following equation:

$$\text{inner product or projection} = |f_1||f_2|\cos\theta = x_1x_2 + y_1y_2$$

In the figure we see that the projection consists of what is common between the two vectors, and that if they are perpendicular, then the inner product is zero and they have nothing in common. This is important for basis images because we are decomposing a

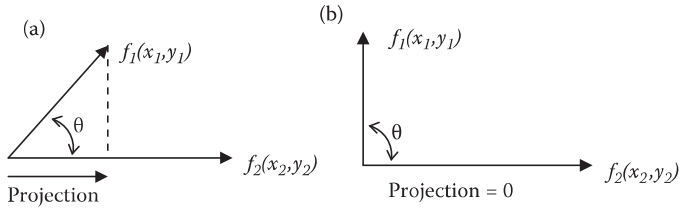


FIGURE 5.1-6
Vector inner product/projection. (a) Given two vectors, $f_1(x_1,y_1)$ and $f_2(x_2,y_2)$, we can find the vector inner product by the following equation: $|f_1||f_2|\cos\theta = x_1x_2 + y_1y_2$, here we see the projection of f_1 onto f_2, with θ less than 90°, (b) If the two vectors are perpendicular, then the inner product is zero because the cos(90°) = 0, and the two vectors have nothing in common.

complex function into a weighted sum of these basis images, and if the basis images are not orthogonal then these weights, $T(u,v)$, will contain redundant information. This will become clearer as we look at the specific transforms.

5.2 Fourier Transform

The Fourier transform is the best known, and the most widely used, of the transforms considered here. It was developed by Jean Baptiste Joseph Fourier (1768–1830) to explain the distribution of temperature and heat conduction. Since that time the Fourier transform has found numerous uses, including vibration analysis in mechanical engineering, circuit analysis in electrical engineering, and here in digital image processing. The Fourier transform decomposes a complex signal into a weighted sum of a zero frequency term (the DC term that is related to the average value), and sinusoidal terms, the basis functions, where each sinusoid is a harmonic of the fundamental. The *fundamental* is the basic or lowest frequency, and the *harmonics* are frequency multiples of the fundamental (the fundamental is also called the first harmonic). We can recreate the original signal by adding the fundamental and all the harmonics, with each term weighted by its corresponding transform coefficient. This is shown in Figure 5.2-1.

Fourier transform theory begins with the 1-D continuous transform, defined as follows:

$$F(v) = \int_{-\infty}^{\infty} I(c)e^{-j2\pi vc}dc$$

The basis functions, $e^{-j2\pi vc}$, are complex exponentials and will be defined in the next section, but for now suffice it to say that they are sinusoidal in nature. Also note that continuous Fourier transform theory assumes that the functions start at $-\infty$ and go to $+\infty$, so they are continuous and everywhere. This aspect of the underlying theory is important for the periodic property of the Fourier transform discussed later.

Example 5.2.1

Given the simple rectangle function shown in Figure 5.2-2a, we can find the Fourier transform by applying the equation defined above:

$$\begin{aligned} F(v) &= \int_{-\infty}^{\infty} I(c)e^{-j2\pi vc}dc \\ &= \int_{0}^{C} Ae^{-j2\pi vc}dc \\ &= -\frac{-A}{j2\pi v}\left[e^{-j2\pi vc}\right]_0^C = \frac{-A}{j2\pi v}\left[e^{-j2\pi vC} - 1\right] \\ &= \frac{A}{j2\pi v}\left[e^{j\pi vC} - e^{-j\pi vC}\right]e^{-j\pi vC} \end{aligned}$$

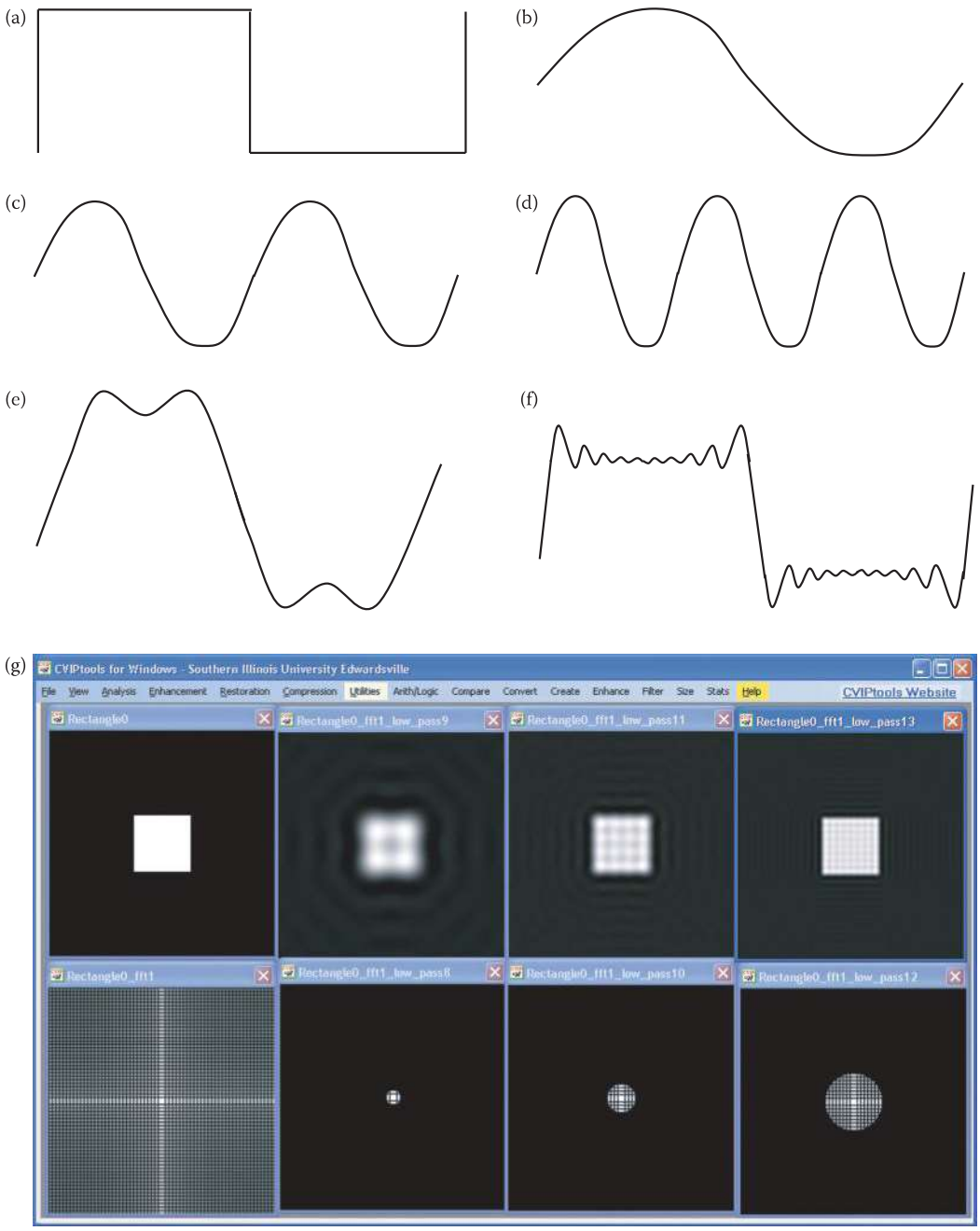


FIGURE 5.2-1
Decomposing a square wave with a Fourier transform. (a) The square wave, (b) the fundamental, or first harmonic, (c) the second harmonic, (d) the third harmonic, (e) approximation to the sum of the fundamental and the first three harmonic harmonics, (f) approximation to sum of the first 20 harmonics, (g) CVIPtools screen capture of a square and successively adding more harmonics. Across the top are the reconstructed squares with approximately 8, 16, and then 32 harmonics. Across the bottom are the corresponding Fourier transform magnitude images.

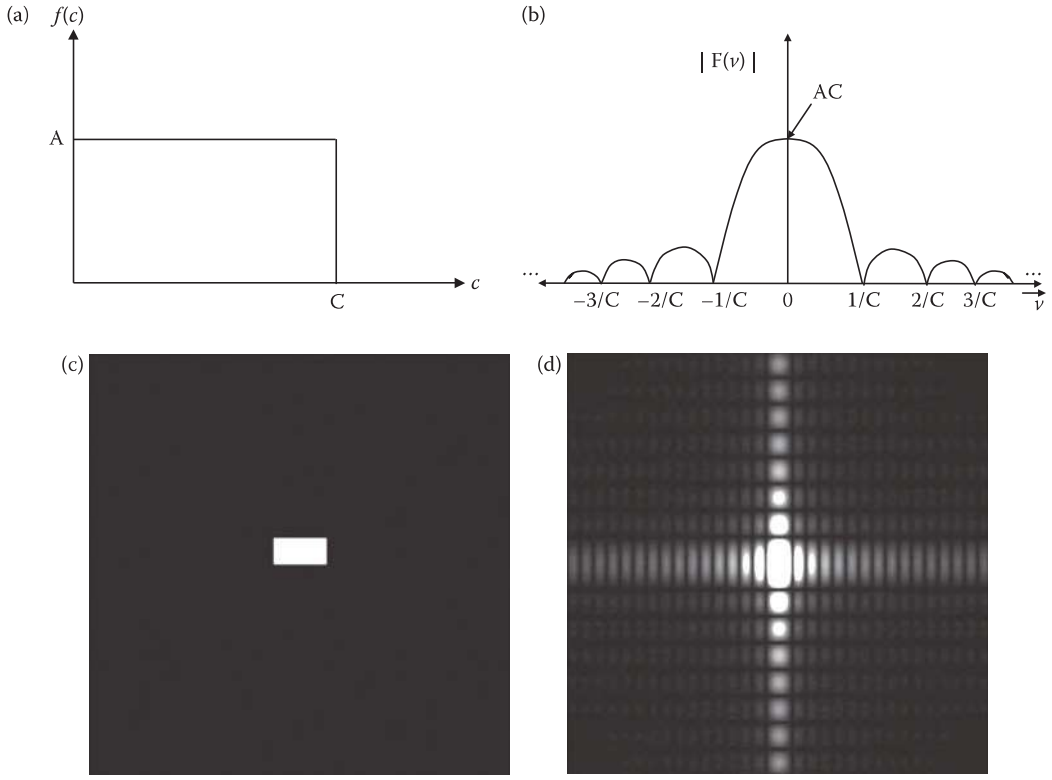


FIGURE 5.2-2
Fourier transform example. (a) The one-dimensional rectangle function, (b) the magnitude of Fourier transform of the 1-D rectangle function: $|F(v)| = AC|\sin(\pi vC/\pi vC)| = AC|\text{sinc}(vC)|$ (c) Two-dimensional rectangle function as an image, (d) the magnitude of the Fourier transform, called the Fourier spectrum, of the 2-D rectangle.

then we use the trigonometric identity, $\sin\theta = (e^{j\theta} - e^{-j\theta})/2j$

$$= \frac{A}{\pi v}\sin(\pi vC)e^{-j\pi vC}$$

This result is complex function, and here we are interested in the magnitude (defined in the next section), which is

$$|F(v)| = \left|\frac{A}{\pi v}\right|\left|\sin(\pi vC)\right|\left|e^{-j\pi vC}\right|$$

Now we multiply through by C/C, and the magnitude of $e^{-j\pi vC} = 1$, we can get it in the form of a sinc function:

$$= AC\left|\frac{\sin(\pi vC)}{(\pi vC)}\right| = AC|\text{sinc}(\text{v}C)|$$

Figure 5.2-2b shows this result.

Figure 5.2-2c shows the 2-D rectangle function, with the brightness of the image representing the magnitude of the function. In Figure 5.2-2d we see the magnitude of the Fourier spectrum in image form. It is customary to display the magnitude only of a Fourier spectrum, as the Fourier transform contains complex terms, which have real and imaginary

parts. The magnitude is however a real quantity and is also referred to as the Fourier spectrum or frequency spectrum.

The reasons for introducing this example here are as follows: (1) to illustrate the continuous and infinite nature of the basis functions in the underlying theory, and (2) to illustrate that when we have a function that ends abruptly in one domain, such as the function $F(c)$, it leads to a continuous series of decaying ripples in the other domain as shown in Figure 5.2-2b and 5.2-2d, and (3) to show that the width of the rectangle in one domain is inversely proportional to the spacing of the ripples in the other domain. As you will see, this will be useful in understanding the nature of phenomena that occurs in images at object boundaries, especially when we apply filters; but first we will explore the details of the discrete Fourier transform (DFT).

5.2.1 One-Dimensional Discrete Fourier Transform

The equation for the 1-D DFT is

$$F(v) = \frac{1}{N}\sum_{c=0}^{N-1} I(c)e^{-j2\pi vc/N}$$

The inverse DFT is given by

$$F^{-1}[F(v)] = I(c) = \sum_{v=0}^{N-1} F(v)e^{j2\pi vc/N}$$

where the $F^{-1}[\ ]$ notation represents the inverse transform. These equations correspond to one row of an image; note that as we move across a row, the column coordinate is the one that changes. The base of the natural logarithmic function, e, is about 2.71828; j, the imaginary coordinate for a complex number, equals $\sqrt{-1}$. The basis functions are sinusoidal in nature, as can be seen by Euler's identity:

$$e^{j\theta} = \cos(\theta) + j\sin(\theta)$$

Putting this equation into the DFT equation by substituting $\theta = -2\pi vc/N$, and remembering that $\cos(\theta) = \cos(-\theta)$ and $\sin(-\theta) = -\sin(\theta)$, the 1-D DFT equation can be written as

$$F(v) = \frac{1}{N}\sum_{c=0}^{N-1} I(c)[\cos(2\pi vc/N) - j\sin(2\pi vc/N)] = \mathrm{Re}(v) + j\mathrm{Im}(v)$$

In this case, $F(v)$ is also complex, with the real part corresponding to the cosine terms, and the imaginary part corresponding to the sine terms. If we represent a complex spectral component by $F(v) = \mathrm{Re}(v) + j\mathrm{Im}(v)$, where $\mathrm{Re}(v)$ is the real part and $\mathrm{Im}(v)$ is the imaginary part, then we can define the magnitude and phase of a complex spectral component as

$$MAGNITUDE = |F(v)| = \sqrt{[\mathrm{Re}(v)]^2 + [\mathrm{Im}(v)]^2}$$

and

$$PHASE = \phi(v) = \mathrm{Tan}^{-1}\left[\frac{\mathrm{Im}(v)}{\mathrm{Re}(v)}\right]$$

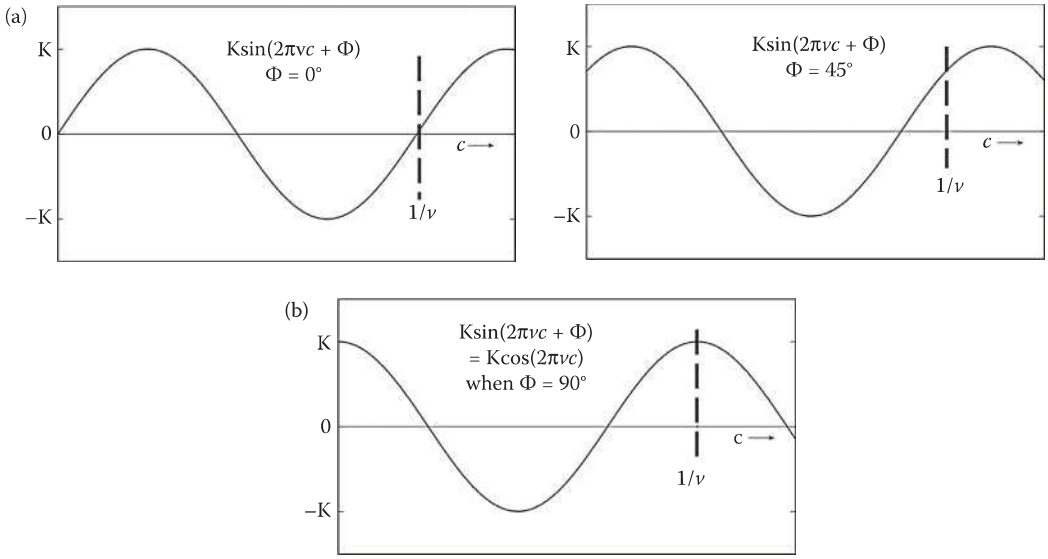


FIGURE 5.2-3
Magnitude and phase of sinusoidal waves.

The magnitude of a sinusoid is simply its peak value, and the phase determines where the origin is, or where the sinusoid starts (see Figure 5.2-3). Keep in mind that the basis functions are simply sinusoids at varying frequencies, the complex exponential notation, e^{jx}, is simply a mathematical notational tool to make it easier to write and manipulate the equations. In Figure 5.2-4 we see that a complex number can be expressed in rectangular form, described by the real and imaginary part; or in exponential form, by the magnitude and phase. A memory aid for evaluating $e^{j\theta}$ is given in Figure 5.2-5.

Example 5.2.2

Given $I(c) = [3,2,2,1]$, corresponding to the brightness values of one row of a digital image. Find $F(v)$ in both rectangular form, and in exponential form.

$$F(v) = \frac{1}{N}\sum_{c=0}^{N-1} I(c)e^{-j2\pi vc/N}$$

$$F(0) = \frac{1}{4}\sum_{c=0}^{3} I(c)e^{-j2\pi vc/4} = \frac{1}{4}\sum_{c=0}^{3} I(c)e^{0} = \frac{1}{4}[I(0)+I(1)+I(2)+I(3)]1 = \frac{1}{4}[3+2+2+1] = 2$$

$$F(1) = \frac{1}{4}\sum_{c=0}^{3} I(c)e^{-j2\pi(1)c/4} = \frac{1}{4}\left[3e^{0}+2e^{-j\pi/2}+2e^{-j\pi}+1e^{-j\pi 3/2}\right] = \frac{1}{4}[3+2(-j)+2(-1)+1(j)]$$

$$= \frac{1}{4}[1-j]$$

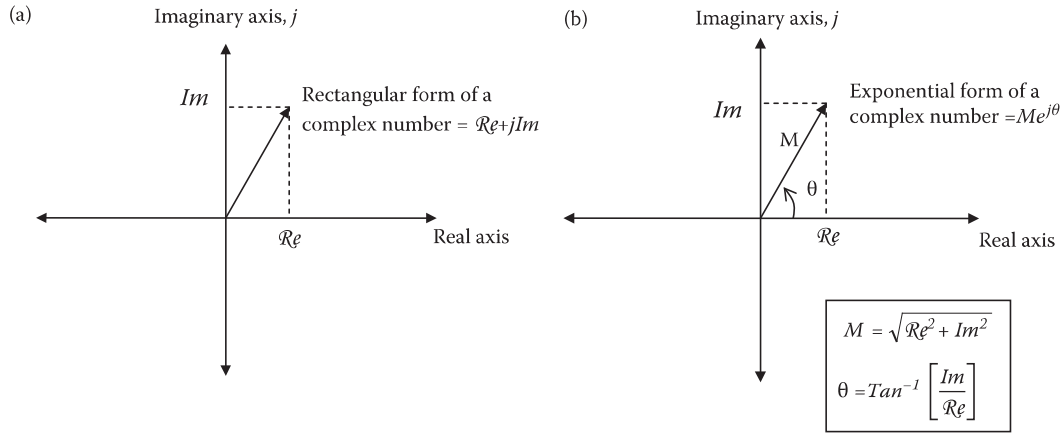


FIGURE 5.2-4
Complex numbers. (a) A complex number shown as a vector and expressed in rectangular form, in terms of the real, Re, and imaginary components, *Im*, (b) a complex number expressed in exponential form in terms of magnitude, *M*, and angle, θ. Note that θ is measured from the real axis counterclockwise.

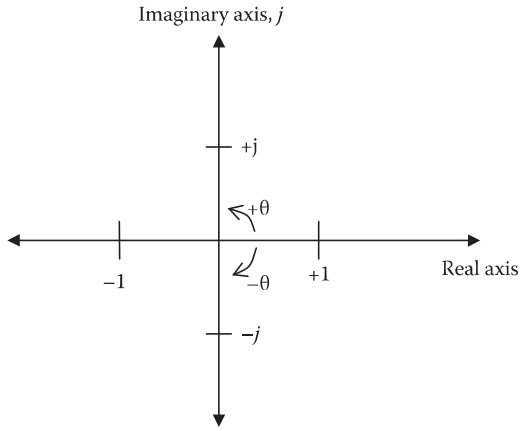


FIGURE 5.2-5
A memory aid for evaluating $e^{j\theta}$ The angle is measured from the real axis counterclockwise, so $\theta = 0 \Rightarrow e^{j\theta} = +1$, $\theta = \pi/2 \Rightarrow e^{j\theta} = +j$, $\theta = \pi \Rightarrow e^{j\theta} = -1$, $\theta = 3\pi/2 \Rightarrow e^{j\theta} = -j$, $\theta = 2\pi/2 \Rightarrow e^{j\theta} = +1$, $\theta = 5\pi/2 \Rightarrow e^{j\theta} = +j$, and so on…

$$F(2) = \frac{1}{4}\sum_{c=0}^{3} I(c)e^{-j2\pi(2)c/4} = \frac{1}{4}\left[3e^{0} + 2e^{-j\pi} + 2e^{-j2\pi} + 1e^{-j3\pi}\right] = \frac{1}{4}[3 + (-2) + 2 + (-1)] = \frac{1}{2}$$

$$F(3) = \frac{1}{4}\sum_{c=0}^{3} I(c)e^{-j2\pi(3)c/4} = \frac{1}{4}\left[3e^{0} + 2e^{-j\pi 3/2} + 2e^{-j3\pi} + 1e^{-j\pi 9/2}\right] = \frac{1}{4}[3 + 2j + 2(-1) + 1(-j)] = \frac{1}{4}[1 + j]$$

Therefore we have

$$F(v) = \left[2, \frac{1}{4}[1-j], \frac{1}{2}, \frac{1}{4}[1+j]\right]$$

Next, put these into exponential form:

$$F(0) = 2 = 2 + 0j \Rightarrow M = \sqrt{2^2 + 0^2} = 2;\ \theta = \tan^{-1}\left[\frac{0}{2}\right] = 0$$

$$F(1) = \frac{1}{4}[1 - j] = \frac{1}{4} - \frac{1}{4}j \Rightarrow M = \sqrt{\left(\frac{1}{4}\right)^2 + \left(-\frac{1}{4}\right)^2} \cong 0.35;\theta = \tan^{-1}\left[\frac{-\frac{1}{4}}{\frac{1}{4}}\right] = -\pi/4$$

$$F(2) = \frac{1}{2} = \frac{1}{2} + 0j \Rightarrow M = \sqrt{\left(\frac{1}{2}\right)^2 + 0^2} = 0.5;\ \theta = \tan^{-1}\left[\frac{0}{\frac{1}{2}}\right] = 0$$

$$F(3) = \frac{1}{4}[1 + j] = \frac{1}{4} + \frac{1}{4}j \Rightarrow M = \sqrt{\left(\frac{1}{4}\right)^2 + \left(\frac{1}{4}\right)^2} \cong 0.35;\ \theta = \tan^{-1}\left[\frac{\frac{1}{4}}{\frac{1}{4}}\right] = \pi/4$$

Therefore, we have

$$F(v) = \left[2,\ 0.35e^{-j\pi/4},\ 0.5,\ 0.35e^{j\pi/4}\right]$$

5.2.2 Two-Dimensional Discrete Fourier Transform

Extending the DFT to the 2-D case for images, we can decompose an image into a weighted sum of 2-D sinusoidal terms. The physical interpretation of a 2-D sinusoid is shown in Figure 5.2-6. Here we see that a sinusoid that is not directly on the u or the v axis can be broken down into separate frequency terms by finding the period along each axis. Assuming a square $N \times N$ image, the equation for the 2-D DFT is

$$F(u,v) = \frac{1}{N}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)e^{-j2\pi(ur+vc)/N}$$

As before, we can also write the Fourier transform equation as

$$F(u,v) = \frac{1}{N}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)\left[\cos\left(\frac{2\pi}{N}(ur+vc)\right) - j\sin\left(\frac{2\pi}{N}(ur+vc)\right)\right]$$

Now, $F(u,v)$ is also complex, with the real part corresponding to the cosine terms, and the imaginary part corresponding to the sine terms. If we represent a complex spectral component by $F(u,v) = \mathrm{Re}(u,v) + j\mathrm{Im}(u,v)$, where $\mathrm{Re}(u,v)$ is the real part and $\mathrm{Im}(u,v)$ is the imaginary part, then we can define the magnitude and phase of a complex spectral component as

$$MAGNITUDE = |F(u,v)| = \sqrt{[\mathrm{Re}(u,v)]^2 + [\mathrm{Im}(u,v)]^2}$$

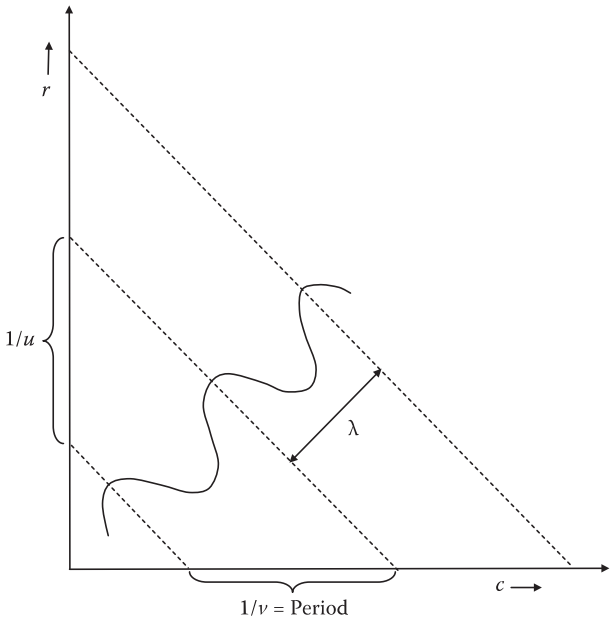


FIGURE 5.2-6
Physical interpretation of a two-dimensional sinusoid. The wavelength of the sinusoid is $\lambda = 1/\sqrt{u^2 + v^2}$ where (u,v) are the frequencies along (r,c), the periods are $1/u$ and $1/v$.

and

$$PHASE = \phi(u,v) = \text{Tan}^{-1}\left[\frac{\text{Im}(u,v)}{\text{Re}(u,v)}\right]$$

Figure 5.2-7 shows images recovered with the phase or magnitude only. With phase only we lose the relative magnitudes, which results in a loss of contrast (see Figure 5.2-7b), but we retain the relative placement of objects; in other words, the phase data contain information about *where objects are* in an image. With the magnitude only image we retain the contrast, but lose all the important detail that is essential to image understanding.

Once we perform the transform, if we want to get our original image back, we need to apply the *inverse transform*. The inverse 2-D DFT is given by

$$F^{-1}[F(u,v)] = I(r,c) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ur+vc)/N}$$

The $F^{-1}[\ ]$ notation represents the inverse transform. This equation illustrates that the function, $I(r,c)$, is represented by a weighted sum of the basis functions, and that the transform coefficients, $F(u,v)$, are the weights. With the inverse Fourier transform, the sign on the basis functions' exponent is changed from –1 to +1. However, this only corresponds to the phase and not the frequency and magnitude of the basis functions (see Figure 5.2-7 and the magnitude and phase equations above).

One important property of the Fourier transform is called *separability*; which means that the 2-D basis image can be decomposed into two product terms where each term depends only on

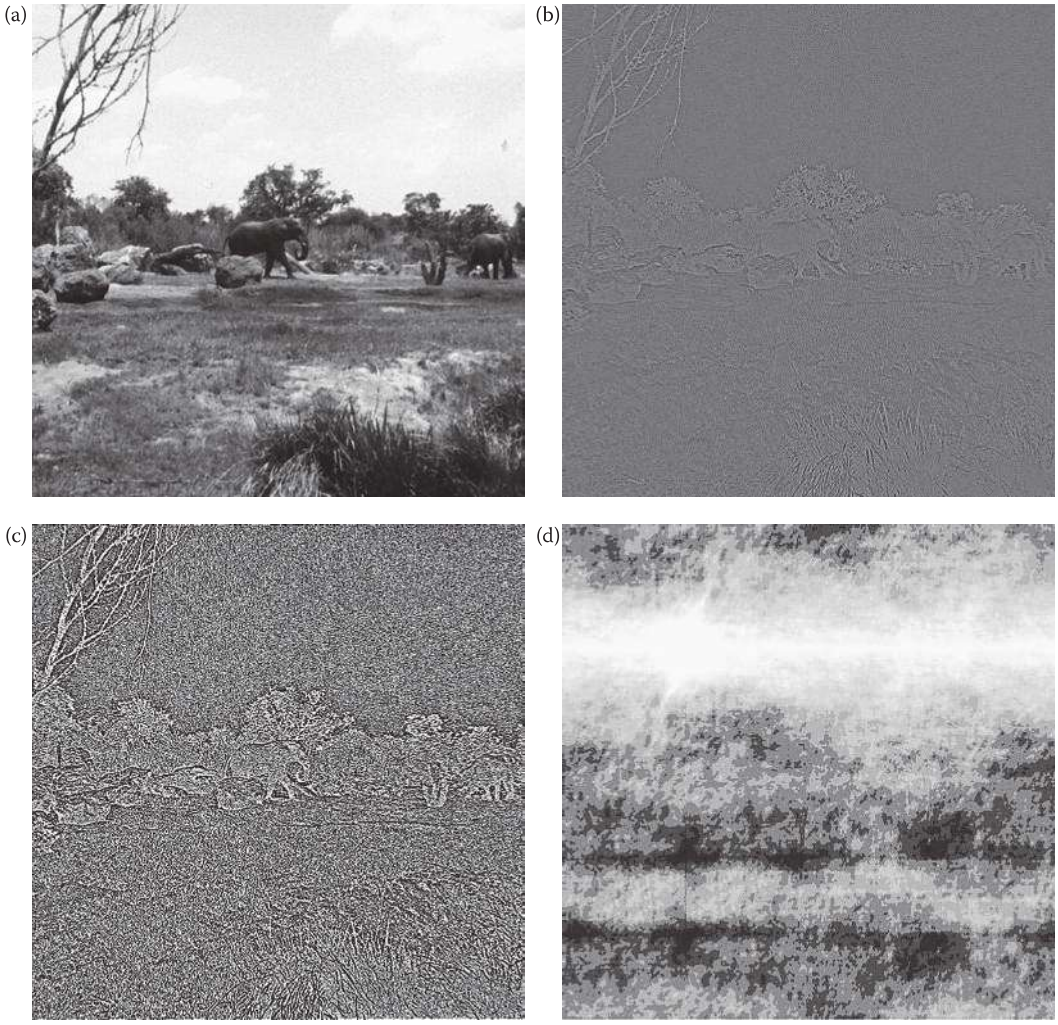


FIGURE 5.2-7
Fourier transform phase and magnitude image information. (a) Original image, (b) Phase only image, (c) Contrast enhanced version of image (b) to show detail, (d) magnitude only image after histogram equalization. The phase only image is created by taking a Fourier transform, setting all the magnitudes equal to 1, and performing an inverse Fourier transform. The magnitude only image is created by taking a Fourier transform, setting the phase to a fixed value, such as 0, then performing an inverse Fourier transform.

the rows or columns. Also, if the basis images are *separable*, then the result can be found by successive applications of two, one-dimensional transforms. This is illustrated by first separating the basis image term (also called the transform kernel) into a product, as follows:

$$e^{-j2\pi(ur+vc)/N} = e^{-j2\pi ur/N}\, e^{-j2\pi vc/N}$$

Next, we write the Fourier transform equation in the following form:

$$F(u,v) = \frac{1}{N}\sum_{r=0}^{N-1}\left(e^{-j2\pi ur/N}\right)\sum_{c=0}^{N-1} I(r,c)e^{-j2\pi vc/N}$$

The advantage of the separability property is that $F(u,v)$ or $I(r,c)$ can be obtained in two steps by successive applications of the 1-D Fourier transform or its inverse. Expressing the equation as

$$F(u,v) = \frac{1}{N}\sum_{r=0}^{N-1} F(r,v)e^{-j2\pi ur/N}$$

where

$$F(r,v) = (N)\left(\frac{1}{N}\right)\sum_{c=0}^{N-1} I(r,c)e^{-j2\pi vc/N}$$

For each value of r, the expression inside the brackets is a 1-D transform with frequency values $v = 0,1,2,3, \ldots N-1$. Hence the 2-D function $F(r,v)$ is obtained by taking a transform along each row of $I(r,c)$ and multiplying the result by N. The desired result, $F(u,v)$ is obtained by taking a transform along each column of $F(r,v)$.

Often, the DFT is implemented as a Fast Fourier Transform (FFT). There are fast algorithms for most of the transforms described here, and many are based on the input data having a number of elements that are a power of 2, which is common for images. In general, these algorithms take advantage of the many redundant calculations involved and operate to eliminate this redundancy. The transforms in Computer Vision and Image Processing tools (CVIPtools) are implemented with fast algorithms based on powers of 2, which means that any image that is not a power of 2 will be zero-padded. Details of these algorithms can be found in the references.

5.2.3 Fourier Transform Properties

A Fourier transform pair refers to an equation in one domain, either spatial or spectral, and its corresponding equation in the other domain. This implies that if we know what is done in one domain, we know what will occur in the other domain.

5.2.3.1 Linearity

The Fourier transform is a linear operator and is shown by the following equations:

$$F\left[aI_1(r,c) + bI_2(r,c)\right] = aF_1(u,v) + bF_2(u,v)$$
$$aI_1(r,c) + bI_2(r,c) = F^{-1}\left[aF_1(u,v) + bF_2(u,v)\right]$$

where a and b are constants.

5.2.3.2 Convolution

Convolution in one domain is the equivalent of multiplication in the other domain, this is what allows us to perform filtering in the spatial domain with convolution masks (see Section 5.7). Using * to denote the convolution operation, and $F[\ ]$ for the forward

Fourier transform and $F^{-1}[\ ]$ for the inverse Fourier transform, these equations define this property:

$$F[I_1(r,c) * I_2(r,c)] = F_1(u,v)F_2(u,v)$$

$$I_1(r,c) * I_2(r,c) = F^{-1}[F_1(u,v)F_2(u,v)]$$

$$F[I_1(r,c)I_2(r,c)] = F_1(u,v) * F_2(u,v)$$

$$I_1(r,c)I_2(r,c) = F^{-1}[F_1(u,v) * F_2(u,v)]$$

Note that it may be computationally less intensive to apply filters in the spatial domain of the image rather than the frequency domain of the image, especially if parallel hardware is available.

5.2.3.3 Translation

The translation property of the Fourier transform is given by the following equations:

$$F[I(r - r_0, c - c_0)] = F(u,v)e^{-j2\pi(ur_0+vc_0)/N}$$

$$I(r - r_0, c - c_0) = F^{-1}\left[F(u,v)e^{-j2\pi(ur_0+vc_0)/N}\right]$$

These equations tell us that if the image is moved, the resulting Fourier spectrum undergoes a phase shift, but the magnitude of the spectrum remains the same. This is shown in Figure 5.2-8.

5.2.3.4 Modulation

The modulation property, also called the frequency translation property, is given by

$$F\left[I(r,c)e^{j2\pi(u_0r+v_0c)/N}\right] = F(u - u_0, v - v_0)$$

$$I(r,c)e^{j2\pi(u_0r+v_0c)/N} = F^{-1}\left[F(u - u_0, v - v_0)\right]$$

These equations tell us that if the image is multiplied by a complex exponential (remember this is really a form of a sinusoid), its corresponding spectrum is shifted. This property is illustrated in Figure 5.2-9.

5.2.3.5 Rotation

The rotation property can be easily illustrated by using polar coordinates:

$$r = x\cos(\theta),\ c = x\sin(\theta)$$

$$u = w\cos(\phi),\ v = w\sin(\phi)$$

The Fourier transform pair $I(r,c)$ and $F(u\ v)$ become $I(x,\theta)$ and $F(w,\phi)$, respectively, and we can write a Fourier transform pair to illustrate the rotation property as follows:

$$I(x, \theta + \theta_0) = \mathrm{F}^{-1}\left[F(w,\ \phi + \theta_0)\right]$$

$$F[I(x, \theta + \theta_0)\ ] = F(w,\ \phi + \theta_0)$$

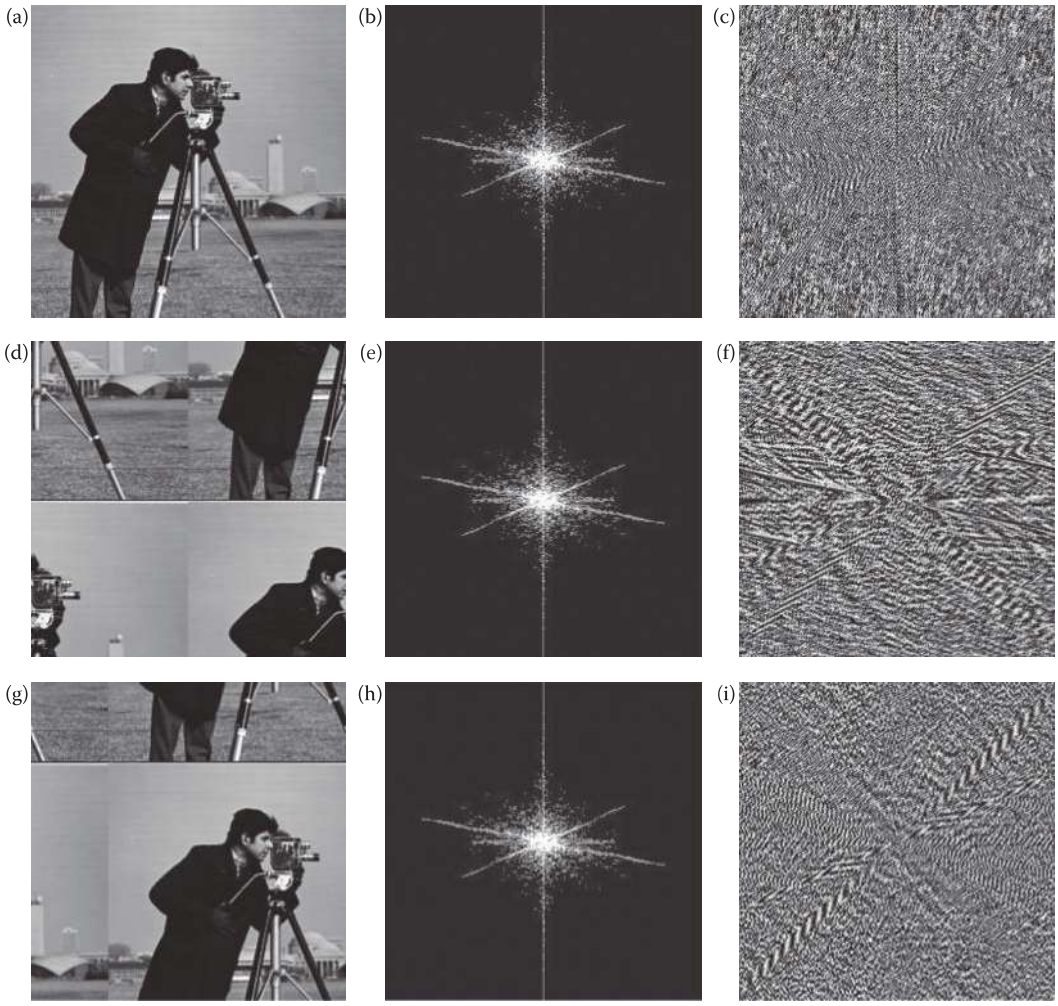


FIGURE 5.2-8
Translation property results in a phase shift of the spectrum. (a) Original image, (b) the magnitude of the Fourier spectrum from (a) represented as an image, (c) the phase of the Fourier spectrum from (a) represented by an image, (d) original image shifted by 128 rows and 128 columns, (e) the magnitude of the Fourier spectrum from (d) represented as an image, (f) the phase of the Fourier spectrum from (d) represented by an image, (g) the original image shifted by 64 columns and 64 rows, (h) the magnitude of the Fourier spectrum from (g) represented as an image, (i) the phase of the Fourier spectrum from (g) represented by an image. These images illustrate that when an image is translated, the phase changes, even though magnitude remains the same.

This property tells us that if an image is rotated by an angle θ_0, then $F(u,v)$ is rotated by the same angle, and vice verse. This is shown in Figure 5.2-10.

5.2.3.6 Periodicity

The DFT is periodic with period N, for an $N \times N$ image. This means,

$$F(u,v) = F(u+N,v) = F(u,v+N) = F(u+N,v+N)\ldots$$

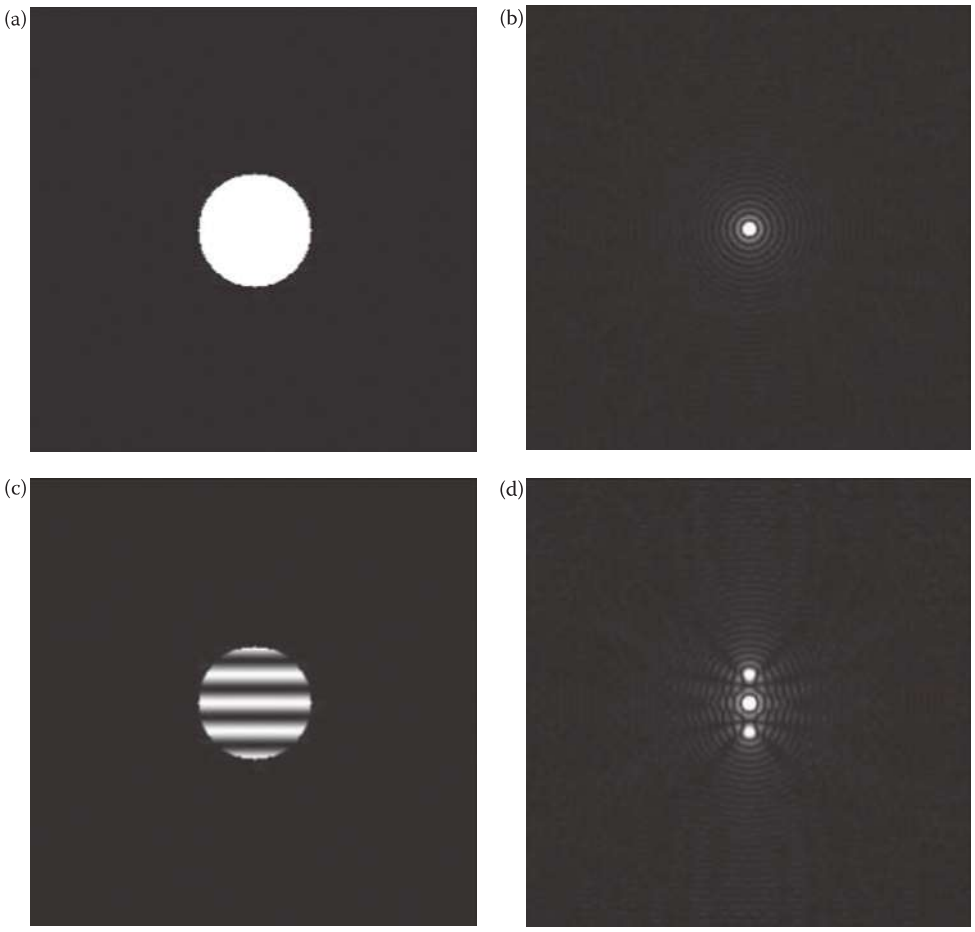


FIGURE 5.2-9
Modulation property results in frequency shift. (a) Original image, (b) magnitude of Fourier spectrum of (a) represented as an image, (c) original image multiplied by a vertical cosine wave at a relative frequency of 16 (16 cycles per image), (d) magnitude of Fourier spectrum of (c) represented as an image. Note that the spectrum has been shifted by 16 above and below the origin (in these spectral images the origin is in the center of the image).

This is shown in Figure 5.2-11a. This figure shows nine periods, but the theoretical implication is that it continues in all directions to infinity. This property defines the implied symmetry in the Fourier spectrum that results from certain theoretical considerations, which have not been rigorously developed here. We will, however, examine the practical implications of these theoretical aspects.

5.2.3.7 Sampling and Aliasing

We can see in Figure 5.2-11d that the range of frequencies in the DFT for a square image is from $-N/2$ to $N/2$. This follows from sampling theory in digital signal processing that states that we must sample a continuous signal with a sampling rate that is at least twice the highest frequency contained in the signal. This sampling rate is called the *Nyquist rate*.

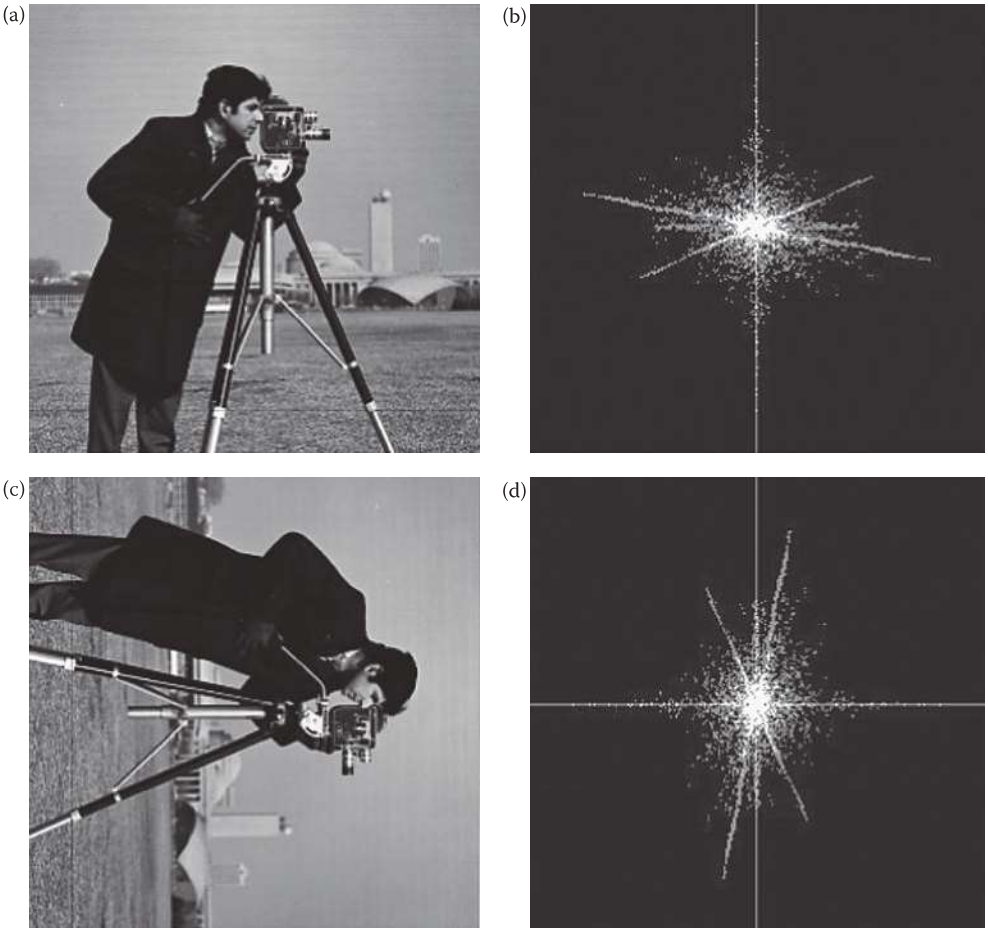


FIGURE 5.2-10
Rotation property results in corresponding rotations with image and spectrum. (a) Original image, (b) Fourier spectrum image of original image, (c) original image rotated by 90°, (d) Fourier spectrum image of rotated image.

If the Nyquist rate is violated, aliasing occurs. Aliasing occurs when the continuous signal is sampled at a rate less than twice the highest frequency in the signal and appears as false information in the high frequencies.

Figure 5.2-12 illustrates the concept of aliasing. For simplicity we will look at two periods and only consider the frequencies in the horizontal direction. By applying the periodicity of the Fourier transform, which says it is periodic with period of $N \times N$ for an $N \times N$ image, if there is information in the signal at frequencies greater than $N/2$, overlap will occur. This overlap causes the aliasing, which manifests itself as false information in the high frequencies. In Figure 5.2-12a we show the spectrum of two periods where the Nyquist criteria has been met. Figure 5.2-12b shows what happens if the original signal actually contains frequencies greater than N/2 in the horizontal direction—adjacent periods overlap

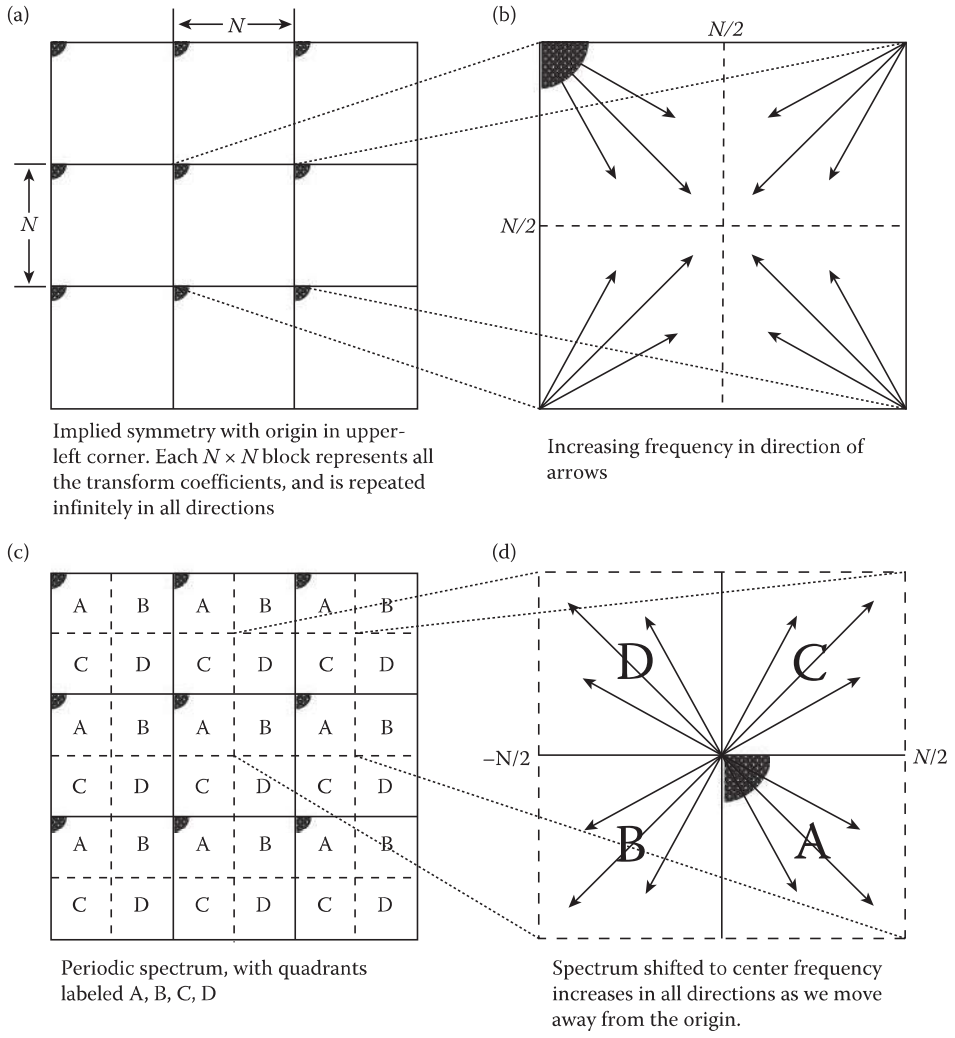


FIGURE 5.2-11
Periodicity and implied symmetry for the Fourier transform.

and aliasing occurs. Note that one method to avoid aliasing is to bandlimit the original signal with a lowpass filter so that it does not contain any frequencies above the Nyquist frequency.

5.2.4 Displaying the Discrete Fourier Spectrum

The Fourier spectrum consists of complex floating point numbers, which are stored in CVIPtools as a single band image with a *data format* of *complex*. What we usually see in a spectral image is actually the magnitude data that has been remapped in a way that makes visualization easier. For displaying the magnitude of the Fourier spectrum, we usually shift

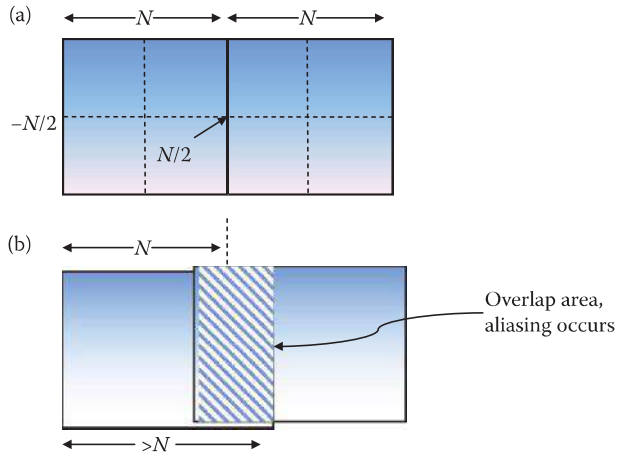


FIGURE 5.2-12
Spectral aliasing. (a) Two periods of the Fourier spectrum of an NxN image,sampled by the Nyquist rate, no frequencies in the signal are greater than N/2, (b) Two periods of the Fourier spectrum of an NxN image, sampled by a rate less than the Nyquist rate. Here the period implied is still NxN, from –N/2 to N/2, but there are actual frequencies in the original image greater than N/2. In this case the periods overlap causing false frequency information in the high frequencies. Since one period overlaps the next period we get contributions from both which creates the false information.

the origin to the center. Applying the periodicity property and the modulation property with $u_0 = v_0 = N/2$, we obtain,

$$I(r,c)e^{j2\pi(Nr/2+Nc/2)/N}$$
$$= I(r,c)e^{j\pi(r+c)}$$
$$= I(r,c)(-1)^{(r+c)}$$

In other words, we can shift the spectrum of $N/2$ by multiplying the original image by $(-1)^{(r+c)}$, which will shift the origin to the center of the image (shown in Figure 5.2-11). This is how it is done in CVIPtools for various reasons: (1) it is easier to understand the spectral information with the origin in the center and frequency increasing from the center out toward the edges, (2) it makes it easier to visualize the filters (Section 5.7), and (3) it looks better.

The actual dynamic range of the Fourier spectrum is much greater than the 256 gray levels (8-bits) available with most image display devices. Thus, when we remap it to 256 levels, we can only see the largest values, which are typically the low frequency terms around the origin and/or terms along the u and v axis. Figure 5.2-13a shows a Fourier magnitude image that has been directly remapped to 0–255 where all we see is the zero frequency term. We can apply contrast enhancement techniques to show more information, as in Figure 5.2-13b through d, but we are still missing much of the visual information due to the limited dynamic range and the human visual system's response.

To take advantage of the human visual system's response to brightness we can greatly enhance the visual information available by displaying the following log transform of the spectrum:

$$\text{Log}(u,v) = k \log[1 + |F(u,v)|]$$

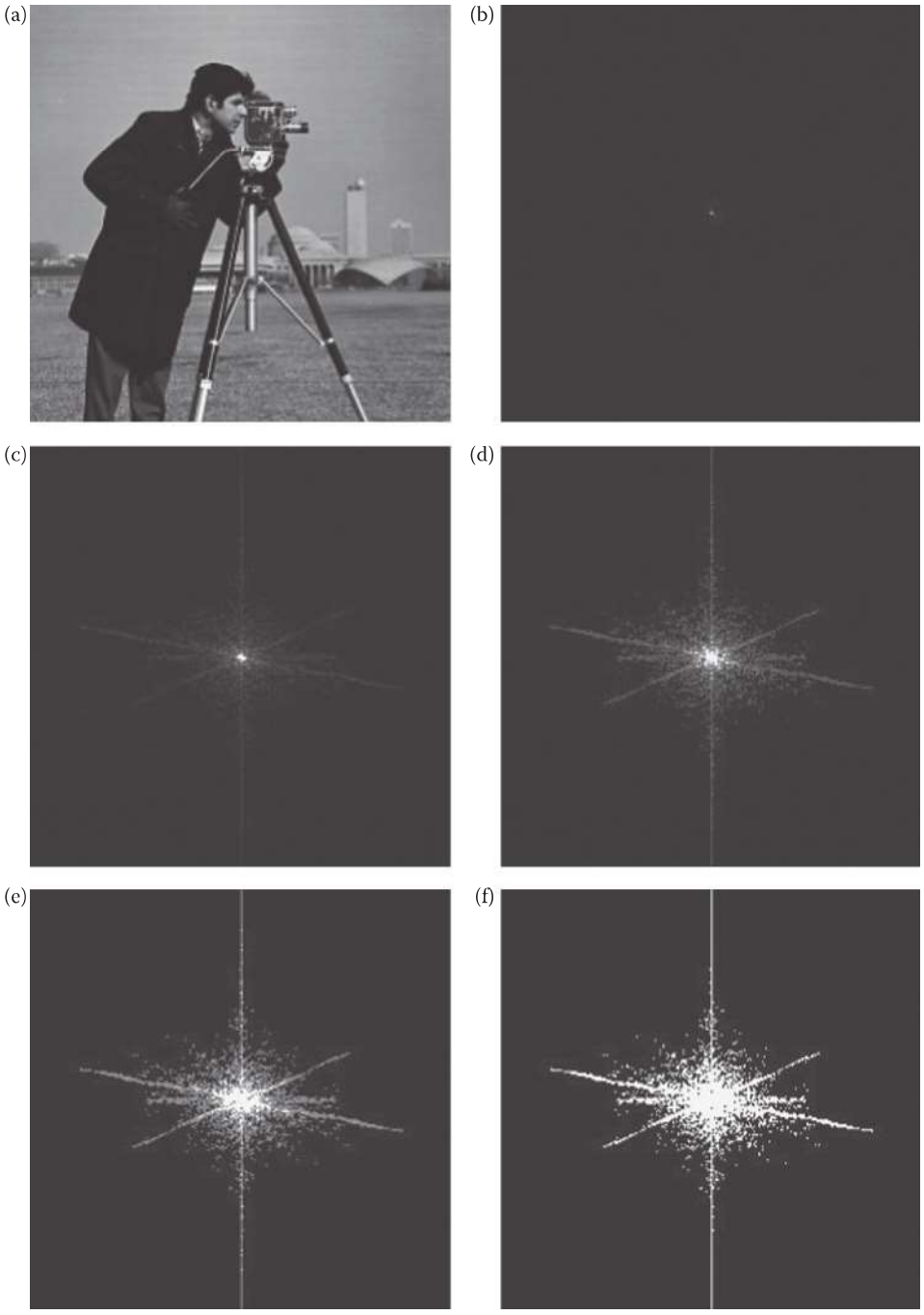


FIGURE 5.2-13
Direct mapping of Fourier magnitude data. (a) Original image, (b) the Fourier magnitude directly remapped to 0.255 without any enhancement, (c–f) contrast enhanced versions of (b). Note that in (f), where we can see the most, the image is visually reduced to being either black or white, most of the dynamic range is lost.

The log function compresses the data, and the scaling factor k remaps the data to the 0–255 range. In Figure 5.2-14 we show the comparison of displaying the magnitude of the spectrum by direct remapping and contrast enhancement versus the log remap method. Here we see that the log remap method shows much more information visually. This effect is most prominent with the spectra from natural images (corresponding to a–c and g–i), as compared with artificial images (corresponding to d–f and j–l). Can you guess the shapes of the artificial images that created the spectra in j–l and d–f? Remember the first example we saw with the continuous Fourier transform, where we learned that a function that ends abruptly in one domain results in rippling in the other domain that corresponds to the edges but the spacing is inversely proportional? In Figure 5.2-15 are images of simple geometric shapes and their corresponding spectral images. Examine them carefully and apply what you have learned thus far to understand them.

In addition to the magnitude information, the phase information is available in the Fourier spectrum. Typically, this information is not displayed as an image, but we have found it useful to illustrate phase changes, as was shown in Figure 5.2-8. The phase information has a range of 0–360°, or 0–2π radians. It is floating point data, so it has a larger dynamic range than the 256 levels typically available for display.

5.3 Discrete Cosine Transform

The cosine transform, like the Fourier transform, uses sinusoidal basis functions. The difference is that the cosine transform basis functions are not complex; they use only cosine functions, and not sine functions. The 2-D discrete cosine transform (DCT) equation for an $N \times N$ image is given by

$$C(u,v) = \alpha(u)\alpha(v)\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)\cos\left[\frac{(2r+1)u\pi}{2N}\right]\cos\left[\frac{(2c+1)v\pi}{2N}\right]$$

where

$$\alpha(u),\alpha(v) = \begin{cases} \sqrt{\frac{1}{N}} \text{ for } u,v = 0 \\ \sqrt{\frac{2}{N}} \text{ for } u,v = 1,2,\ldots,N-1 \end{cases}$$

Since this transform uses only the cosine function it can be calculated using only real arithmetic, instead of complex arithmetic as the DFT requires. The cosine transform can be derived from the Fourier transform by assuming that the function (the image) is mirrored about the origin, thus making it an even function, which means it is symmetric about the origin. This has the effect of canceling the odd terms, which correspond to the sine terms (imaginary terms) in the Fourier transform. This also affects the implied symmetry of the transform, where we now have a function that is implied to be $2N \times 2N$. In Figure 5.3-1 we see the meaning of mirroring, or folding, a function about the origin creating a $2N \times 2N$ function from an original $N \times N$ one.

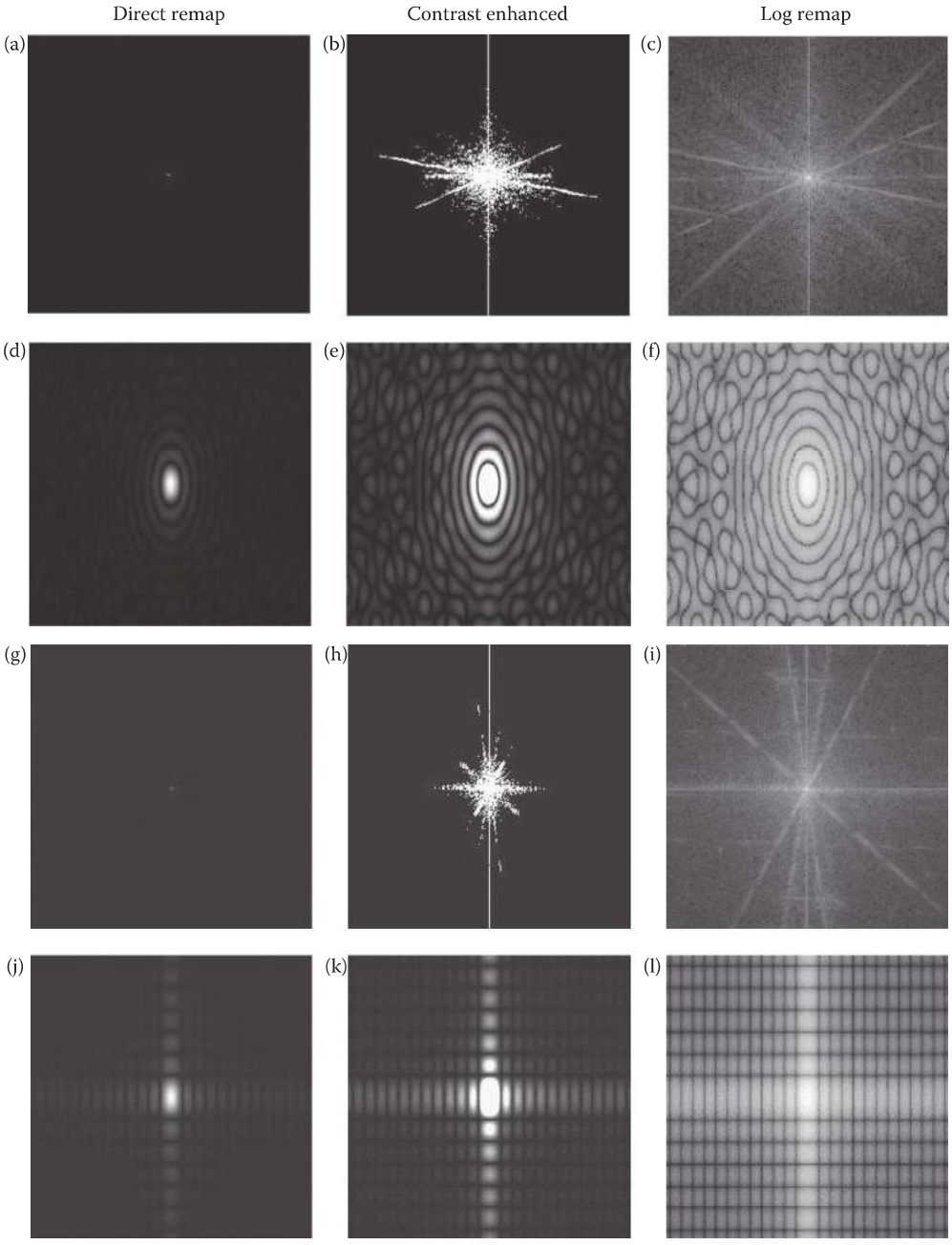


FIGURE 5.2-14
Displaying DFT spectrum with various remap methods. (a) Fourier magnitude spectrum of cam.pgm, direct remap to byte, (b) contrast enhanced version of (a), (c) log remapped version of cam.pgm DFT spectrum, (d) Fourier magnitude spectrum of an ellipse, direct remap to byte, (e) contrast enhanced version of (d), (f) log remapped version of an ellipse DFT spectrum, (g) Fourier magnitude spectrum of house.pgm, direct remap to byte, (h) contrast enhanced version of (g), (i) log remapped version of house.pgm DFT spectrum, (j) Fourier magnitude spectrum of a rectangle, direct remap to byte, (k) contrast enhanced version of (j), (l) log remapped version of a rectangle DFT spectrum.

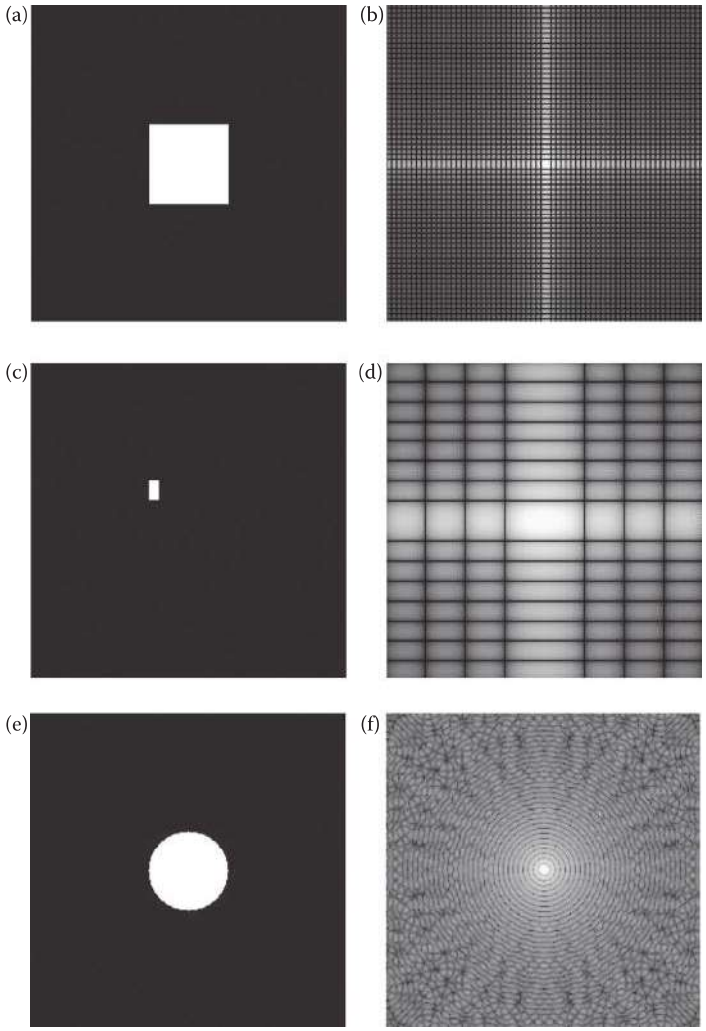


FIGURE 5.2-15
Images of simple geometric shapes and their Fourier spectral images. (a) An image of square, (b) the log remapped spectrum of the square, (c) a small rectangle, (d) the log remapped spectrum of the small rectangle, (e) an image of a circle, (f) the log remapped spectrum of the circle image, (g) a small circle, (h) the log remapped spectrum of the small circle, (i) a small ellipse, (j) the log remapped spectrum of the small ellipse, (k) an image of a vertical sine wave, (l) the magnitude of the spectrum of the sine wave.

Now, we are only interested in an $N \times N$ portion of this spectrum, which corresponds to our image, since the other quadrants are redundant. Understand that we do not want to shift the origin to the center for the cosine transform, or we lose information (see Figure 5.3-2).

The cosine transform is often used in image compression, in particular in the first version of the Joint Photographers Expert Group (JPEG) image compression method, which has been established as an international standard (the newer JPEG2000 method uses the wavelet transform). In digital image processing we often represent the basis matrices as images, called basis images, where we use various gray values to represent the different values in the basis matrix. The 2-D basis images for the cosine transform are shown in

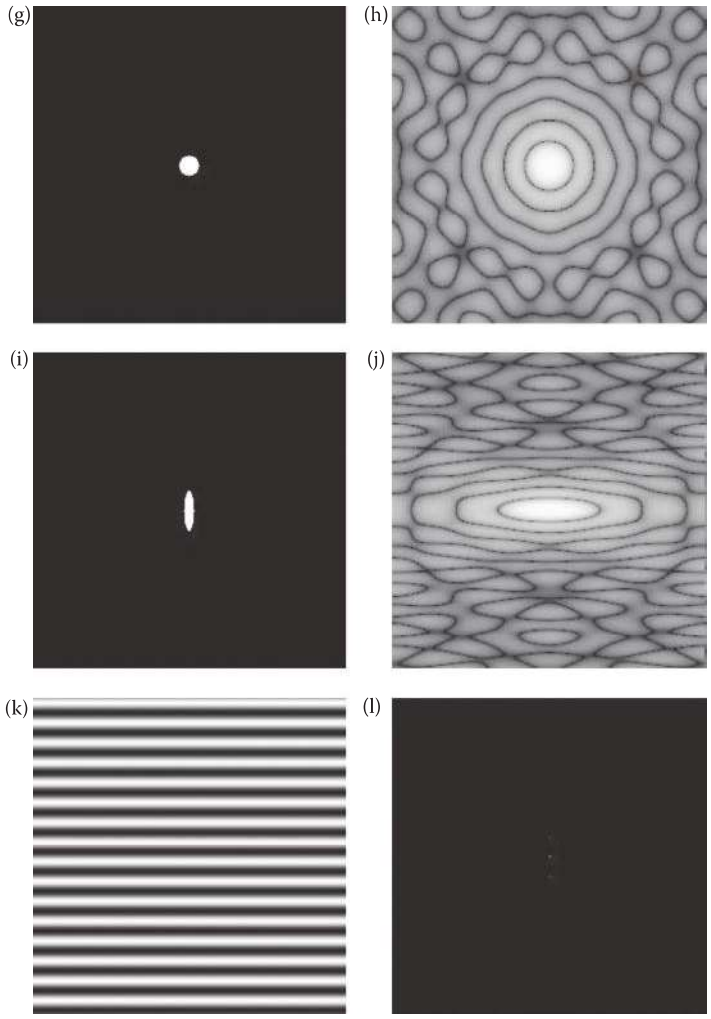


FIGURE 5.2-15 (CONTINUED)
Images of simple geometric shapes and their Fourier spectral images. (a) An image of square, (b) the log remapped spectrum of the square, (c) a small rectangle, (d) the log remapped spectrum of the small rectangle, (e) an image of a circle, (f) the log remapped spectrum of the circle image, (g) a small circle, (h) the log remapped spectrum of the small circle, (i) a small ellipse, (j) the log remapped spectrum of the small ellipse, (k) an image of a vertical sine wave, (l) the magnitude of the spectrum of the sine wave.

Figure 5.3-3 for a 4×4 image, where the actual values have been remapped for illustration purposes by the legend at the bottom of the figure. Remember that the transform actually projects the image onto each of these basis images (see Figure 5.1-5), so the transform coefficients, $C(u,v)$, tell us the amount of that particular basis image that the original image, $I(r,c)$, contains.

The inverse cosine transform is given by

$$C^{-1}[C(u,v)] = I(r,c) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos\left[\frac{(2r+1)u\pi}{2N}\right]\cos\left[\frac{(2c+1)v\pi}{2N}\right]$$

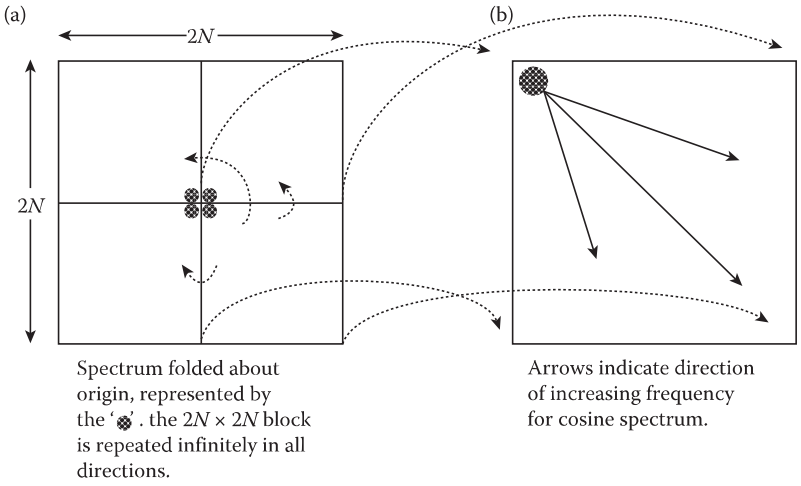


FIGURE 5.3-1
Cosine symmetry.

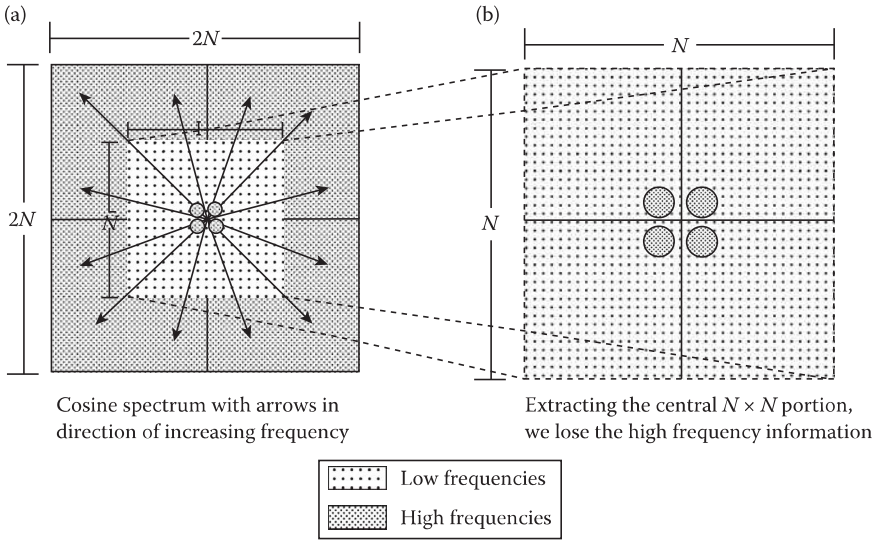
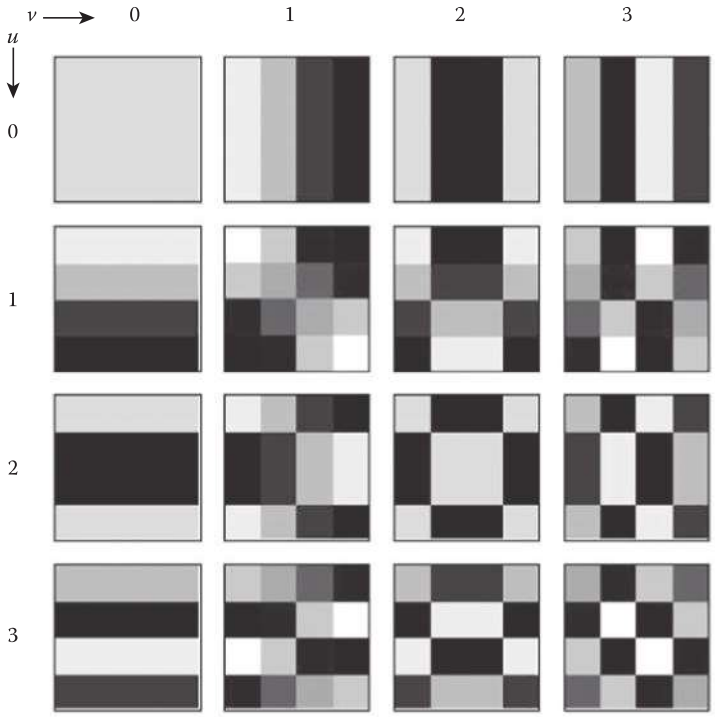


FIGURE 5.3-2
Cosine spectrum should not be shifted to center.



B(r,c)	−0.43	−0.33	−0.25	−0.18	−0.14	−0.07	0.07	0.14	0.18	0.25	0.33	0.43
Gray level	0	30	54	75	86	106	149	170	181	202	225	255

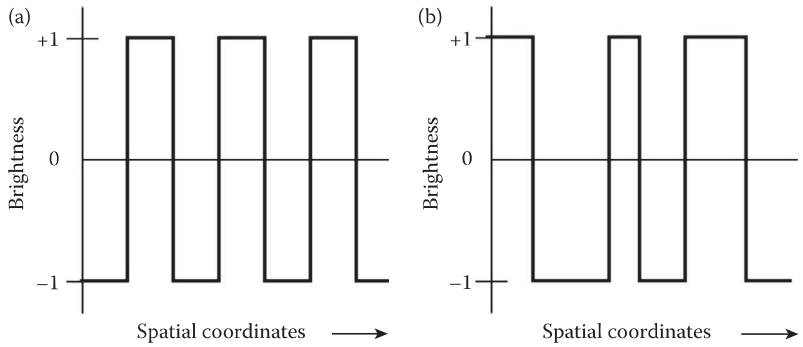
FIGURE 5.3-3
Discrete cosine transform basis images.

5.4 Discrete Walsh–Hadamard Transform

The Walsh–Hadamard transform (WHT) differs from the Fourier and cosine transforms in that the basis functions are not sinusoids. The basis functions are based on square or rectangular waves with peaks of ±1 (see Figure 5.4-1). Here the term rectangular wave refers to any function of this form, where the width of the pulse may vary. One primary advantage of a transform with these types of basis functions is that the computations are very simple. When we project the image onto the basis functions, all we need to do is to multiply each pixel by ±1, as in seen in he WHT equation:

$$WH(u,v) = \frac{1}{N}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)\,(-1)^{\sum_{i=0}^{n-1}[b_i(r)p_i(u)+b_i(c)p_i(v)]}$$

where $N = 2^n$, the exponent on the (–1) is performed in modulo 2 arithmetic, and $b_i(r)$ is found by considering r as a binary number, and finding the ith bit.



A square wave

Representation of a rectangular wave. The width of each pulse may vary.

FIGURE 5.4-1
Form of the Walsh–Hadamard basis functions.

Example 5.4.1

$n = 3$ (3 bits, so $N = 8$), and $r = 4$
r in binary is 100, so $b_2(r) = 1$, $b_1(r) = 0$, and $b_0(r) = 0$

Example 5.4.2

$n = 4$, (4 bits, so $N = 16$), and $r = 2$
r in binary is 0010, so $b_3(r) = 0$, $b_2(r) = 0$, $b_1(r) = 1$, and $b_0(r) = 0$

$p_i(u)$ is found as follows:

$$p_0(u) = b_{n-1}(u)$$

$$p_1(u) = b_{n-1}(u) + b_{n-2}(u)$$

$$p_2(u) = b_{n-2}(u) + b_{n-3}(u)$$

$$\vdots$$

$$p_{n-1}(u) = b_1(u) + b_0(u)$$

The sums are performed in modulo 2 arithmetic, and the values for $b_i(c)$ and $p_i(v)$ are found in a similar manner. Strictly speaking we cannot call the WHT a frequency transform, as the basis functions do not exhibit the frequency concept in the manner of sinusoidal functions. However, we define an analogous term for use with these types of functions. If we consider the number of zero crossings (or sign changes) we have a measure that is comparable to frequency, and we call this *sequency*. In Figure 5.4-2 we see the 1-D Walsh–Hadamard basis functions for $N = 4$, and the corresponding sequency. We can see that the basis functions are in the order of increasing sequency, much like the sinusoidal functions are in order of increasing frequency. In Figure 5.4-3, we have the basis images for the WHT for a 4×4 image; we use white for the + 1 and black for the –1.

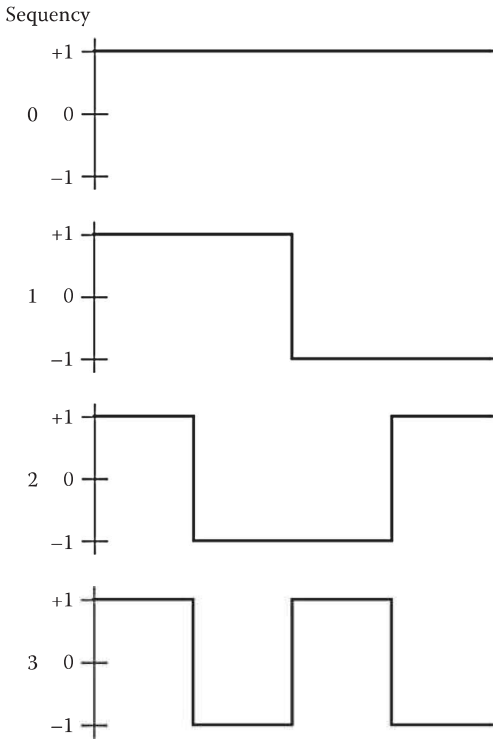


FIGURE 5.4-2
1-D Walsh–Hadamard basis functions.

It may be difficult to see how the 2-D basis images are generated from the 1-D basis vectors. For the terms that are along the u or v axis, we simply repeat the 1-D function along all the rows or columns. For the basis images that are not along the u or v axis we perform a *vector outer product* on the corresponding 1-D vectors. We have seen that a *vector inner product* is what we call a projection, and is performed by overlaying, multiplying coincident terms, and summing the results—this gives us a scalar, or a single number, for a result. The vector outer product gives us a matrix, which is obtained as follows:

Example 5.4.3

For $(u,v) = (3,2)$, see Figure 5.4-4. If we look along one row of the $v = 2$ ($u = 0$) basis image in Figure 5.4-3 we find the following numbers: + 1 –1 –1 + 1. Then if we look along one column in the u direction for $u = 3$ ($v = 0$), we see + 1 –1 + 1 –1. These are the corresponding 1-D basis vectors. We then put the row vector across the top and the column vector down the left side and fill in the matrix by multiplying the column by the corresponding row element, as in Figure 5.4-4. The resulting matrix is the vector outer product. Compare this to the corresponding basis image in Figure 5.4-3.

This process can be used to generate the 2-D basis images for any function that has a separable basis. Remember that *separable* means that the basis function can be expressed as a product of terms that depend only on one of the variable pairs, r,u or c,v, and that this separability also allows us to perform the 2-D transformation by two 1-D transforms. This is accomplished by first doing a 1-D transform on the rows, and then performing the 1-D transform on the resulting columns as was shown in the DFT section.

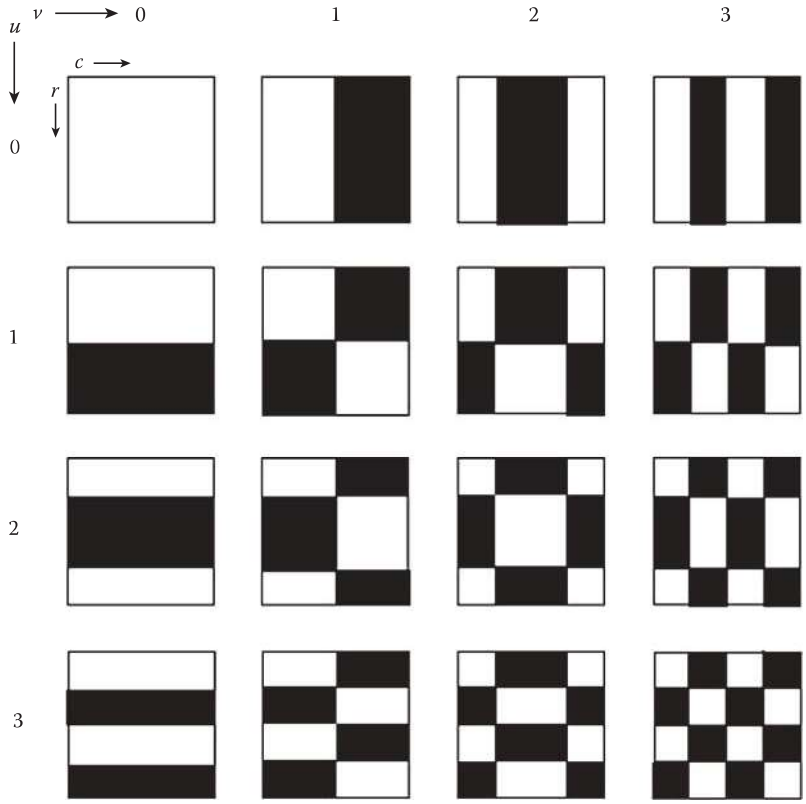


FIGURE 5.4-3
Walsh–Hadamard basis images.

It is interesting to note that with the WHT there is another visual method to find the off axis basis images, by assuming that the black in Figure 5.4-3 corresponds to 0 and the white corresponds to 1. The basis images not along the u or v axis can be obtained by taking the corresponding basis images on these axes, overlaying them, and performing an XOR followed by a NOT. For example, to find the Walsh–Hadamard basis image corresponding to $(u,v) = (3,2)$, we take the basis image along the u axis for $u = 3$, and the basis image along the v axis for $v = 2$, overlay them, XOR the images, and then perform a NOT. This is illustrated in Figure 5.4-5.

The inverse WHT equation is

$$WH^{-1}[WH(u,v)] = I(r,c) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} WH(u,v)(-1)^{\sum_{i=0}^{n-1}[b_i(r)p_i(u)+b_i(c)p_i(v)]}$$

In CVIPtools there is a separate Walsh transform and a separate Hadamard transform. Even though (if N is power of 2) they both have the same basis functions, as initially defined, the basis functions were in different orders. The Hadamard ordering was not sequency based, so this ordering is not really that useful for image processing. It was originally defined for the ease of generating the basis vectors. The standard here is to use the sequency ordered basis functions and call it the WHT. In the CVIPtools the transform called the Walsh is sequency ordered, and the one called the Hadamard is in standard "Hadamard ordering"—not sequency based.

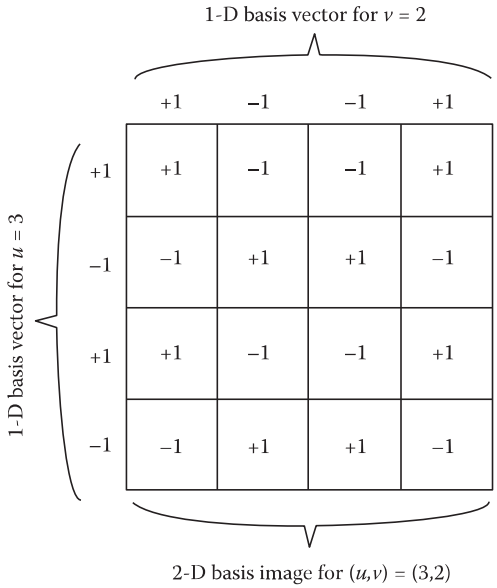


FIGURE 5.4-4
Vector outer product.

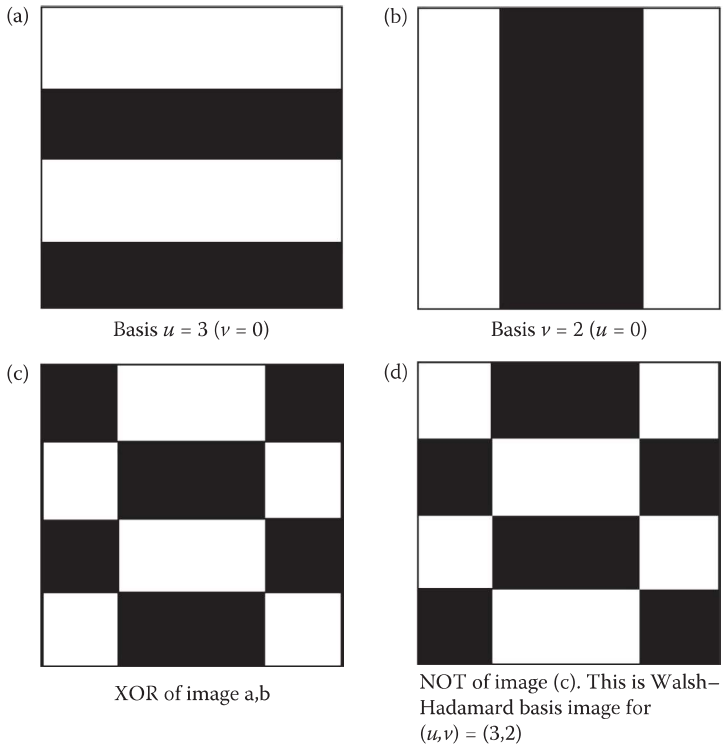


FIGURE 5.4-5
Finding an off-axis Walsh–Hadamard basis image.

5.5 Discrete Haar Transform

The Haar transform has rectangular waves as basis functions, similar to the WHT. The primary differences are that the basis vectors contain not just + 1 and –1, but also contain zeros. The Haar transform is derived from the Haar matrices; in these matrices each row represents a 1-D basis vector. The following shows the basis vectors for a Haar transform of 2 basis vectors ($N = 2$), 4 basis vectors ($N = 4$), and 8 basis vectors ($N = 8$).

$$Haar2 \Rightarrow \frac{1}{\sqrt{2}} \begin{Bmatrix} +1 & +1 \\ +1 & -1 \end{Bmatrix}$$

$$Haar4 \Rightarrow \frac{1}{\sqrt{4}} \begin{Bmatrix} +1 & +1 & +1 & +1 \\ +1 & +1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{Bmatrix}$$

$$Haar8 \Rightarrow \frac{1}{\sqrt{8}} \begin{Bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ +2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & +2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & +2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & +2 & -2 \end{Bmatrix}$$

The Haar basis vectors can be extended to higher orders by following the same patterns shown in the above. Note that as the order increases the number of zeros in the basis vectors increase. This has the unique effect of allowing a multiresolution decomposition of an image (explored more in Section 5.8), and is best illustrated by example. In Figure 5.5-1 we see the log remapped Haar spectrum. Here we see that the Haar provides edge information at increasing levels of resolution.

5.6 Principal Components Transform

The principal components transform (PCT) is also referred to as the Hotelling, Karhunen-Loeve, or eigenvector transform. This is because it was first derived by Karhunen and Loeve for continuous signals, and later developed for discrete signals by Hotelling. Mathematically it involves finding eigenvectors of covariance matrices, hence eigenvector transform, and it results in decomposing the image into its principal components, hence PCT. It differs from the transforms that we have considered thus far, as it is not related to extracting frequency or sequency information from images, but is a mathematical

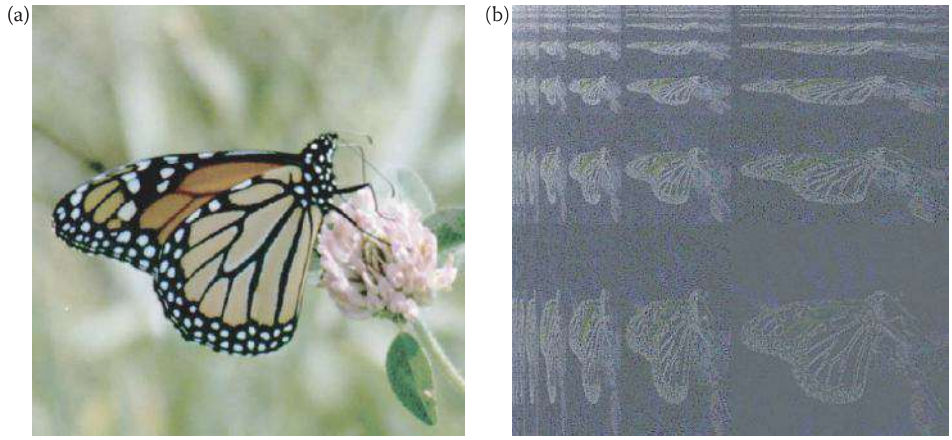


FIGURE 5.5-1
Haar transform. (a) Original image, (b) Haar transform image.

transform that decorrelates multiband image data. It can, however, be used to find optimal basis images for a specific image, but this use of it is not very practical due to the extensive processing required.

Applying the PCT to multiband images, color images or multispectral images, provides a linear transform matrix that will decorrelate the input data. In most color images there is a high level of correlation between the red, green, and blue bands. This can be seen in Figure 5.6-1 where we show the brightness values in the red, green, and blue bands of a color image with each band presented as a monochrome image, and the three bands after the PCT. Here we see that the red, green, and blue bands are highly correlated—they look similar; whereas with the PCT bands most of the visual information is in band 1, some information is in band 2, and practically no visual information in band 3. This is what it means when we say that the PCT decorrelates the data and puts most of the information into the principal component band.

The three step procedure for finding the PCT for a color, RGB, image is as follows:

1. Find the covariance matrix in RGB space, given by

$$[COV]_{RGB} = \begin{bmatrix} C_{RR} & C_{GR} & C_{BR} \\ C_{RG} & C_{GG} & C_{BG} \\ C_{RB} & C_{GB} & C_{BB} \end{bmatrix}$$

where

$$C_{RR} = \frac{1}{P}\sum_{i=1}^{P}\left(R_i - m_R\right)^2$$

P = the number of pixels in the image

R_i = the red value for the ith pixel

$$m_R = \text{Red mean (average)} = \frac{1}{P}\sum_{i=1}^{P} R_i$$

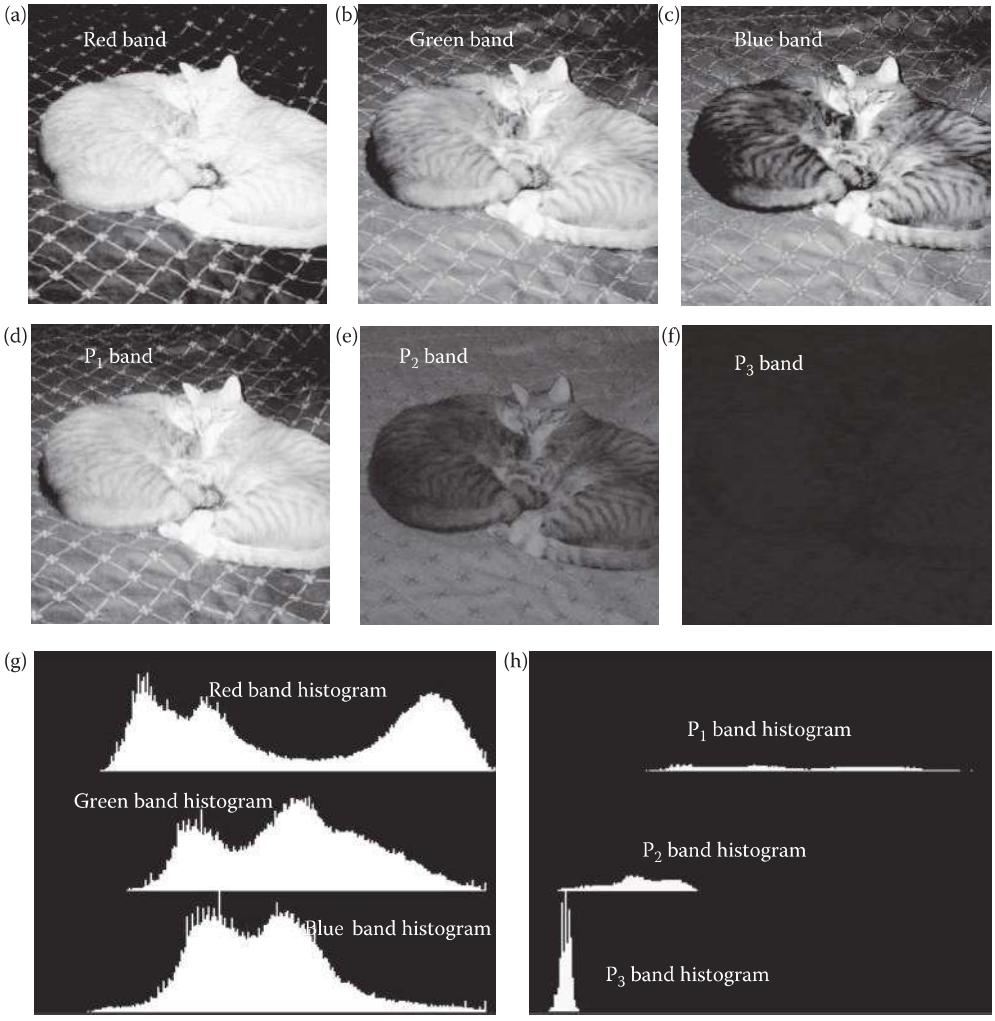


FIGURE 5.6-1
Principal components transform – PCT. (a) Red band of a color image, (b) green band, (c) blue band, (d) principal component band 1, (e) principal component band 2, (f) principal component band 3. Note that the red, green, and blue bands are highly correlated—they look similar; whereas with the PCT bands most of the visual information is in band 1, some in band 2 and none in band 3. This is what it means when we say that the PCT decorrelates the data and puts most of the information into the principal component band. (g) Histogram of the original RGB image, (h) histogram of the PCT image. Note that in the RGB image histogram the data are fairly equally spread out, each band has a similar variance. With the PCT histogram, we can see the decreasing variance as shown by the decreasing spread in the data.

Similar equations are used for C_{GG} and C_{BB} (the autocovariance terms). The elements of the covariance matrix that involve more than one of the RGB variables, C_{GR}, C_{BR}, C_{RG}, C_{BG}, C_{RB}, and C_{GB}, are called cross-covariance terms and are found as follows:

$$C_{XY} = \frac{1}{P}\left[\sum_{i=1}^{P} X_i Y_i\right] - m_x m_y$$

with the means defined as above.

2. Find the eigenvalues of the covariance matrix, e_1, e_2, and e_3, and their corresponding eigenvectors:

$$e_1 \Rightarrow [E_{11}, E_{12}, E_{13}]$$
$$e_2 \Rightarrow [E_{21}, E_{22}, E_{23}]$$
$$e_3 \Rightarrow [E_{31}, E_{32}, E_{33}]$$

   Order them such that e_1 is the largest eigenvalue, and e_3 is the smallest.

3. Perform the linear transform on the RGB data by using the eigenvectors as follows:

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} = \begin{bmatrix} E_{11} & E_{12} & E_{13} \\ E_{21} & E_{22} & E_{23} \\ E_{31} & E_{32} & E_{33} \end{bmatrix} \begin{bmatrix} R_i \\ G_i \\ B_i \end{bmatrix} = \begin{bmatrix} E_{11}R_i + E_{12}G_i + E_{13}B_i \\ E_{21}R_i + E_{22}G_i + E_{23}B_i \\ E_{31}R_i + E_{32}G_i + E_{33}B_i \end{bmatrix}$$

Now the PCT data are P_1, P_2, and P_3 where the P_1 data are the principal component and contains the most variance (as illustrated in Figure 4.3.9). In pattern recognition theory the measure of variance is considered to be a measure of information, so we can say that the principal component data contains the most information, as shown in Figure 5.6-1. The PCT is easily extended to data of any dimensionality (the above example is three-dimensional, RGB), so it can be applied to multispectral images in a similar manner. For a multispectral image this means we can use the PCT to reduce the dimensionality of the image and still retain maximal information.

This allows the PCT to be used in image compression (see Chapter 10), since this transform is optimal in the least-square-error sense. For image compression we simply retain the components with the most information, and discard those with the least information that correspond to the smaller eigenvalues. The mean square error in the reconstructed image can be found as the sum of the eigenvalues associated with the eigenvectors that are discarded. For example, in one application involving a database of medical images, it was experimentally determined that the dimension with the largest variance after the PCT was performed contained approximately 91% of the variance. This would allow at least a 3:1 compression and still retain 91% of the information.

5.7 Filtering

After the image has been transformed into the frequency or sequency domain, we may want to modify the resulting spectrum. Filtering modifies the frequency or sequency spectrum by selectively retaining, removing or scaling the various components of the spectrum. High frequency information can be removed with a lowpass filter, which will have the effect of blurring an image, or low frequency information can be removed with a highpass filter, which will tend to sharpen the image. We may want to extract the frequency information in specific parts of the spectrum by bandpass filtering. Alternately, band-reject filtering can be employed to eliminate specific parts of the

spectrum, for example, to remove unwanted noise. All of these types of filters will be explored here.

Before we explore the filters we need to be aware of the implied symmetry for each of the transforms of interest. We will assume that the Fourier spectrum has been shifted to the center and exhibits the symmetry shown in Figure 5.2-11. Both the cosine and the Walsh–Hadamard are assumed to have the symmetry shown in Figure 5.3-1. The Haar transform is unique, but also has the origin in the upper left corner (see Figure 5.5-1) as the cosine and Walsh–Hadamard. The PCT as defined does not lend itself to the type of filtering under discussion here.

5.7.1 Lowpass Filters

Lowpass filters tend to blur images. They pass low frequencies, and attenuate or eliminate the high frequency information. They are used for image compression, or for mitigating noise effects. Visually they blur the image, although this blur is sometimes considered an enhancement as it imparts a softer effect to the image (see Figure 5.7-1). Lowpass filtering is performed by multiplying the spectrum by a filter, and then applying the inverse transform to obtain the filtered image. The ideal filter function is shown in Figure 5.7-2; note the two types of symmetry in the filter to match the type of symmetry in the spectrum. The frequency at which we start to eliminate information is called the *cutoff frequency*, f_0. The frequencies in the spectrum that are not filtered out are in the *passband*, while the spectral components that do get filtered out are in the *stopband*. We can represent the filtering process by the following equation:

$$I_{fil}(r,c) = T^{-1}[T(u,v)H(u,v)]$$

where $I_{fil}(r,c)$ is our filtered image, $H(u,v)$ is the filter function, $T(u,v)$ is the transform, and $T^{-1}[\,]$ represents the inverse transform. The multiplication, $T(u,v)H(u,v)$, is performed with a point-by-point method. That is, $T(0,0)$ is multiplied by $H(0,0)$, $T(0,1)$ is multiplied by $H(0,1)$, and so on. The resulting products are placed into an array at the same (r,c) location.

Example 5.7.1

Let $H(u,v)$ and $T(u,v)$ be the following 2×2 images.

$$H(u,v) = \begin{bmatrix} 2 & -3 \\ 4 & 1 \end{bmatrix} \quad T(u,v) = \begin{bmatrix} 4 & 6 \\ -5 & 8 \end{bmatrix}$$

Then $T(u,v)H(u,v)$ is equal to: $\begin{bmatrix} 8 & -18 \\ -20 & 8 \end{bmatrix}$

Note that for ideal filters in Figure 5.7-2 the $H(u,v)$ matrix will contain only 1s and 0s, but, as in the above example, the matrix can contain any numbers.

The ideal filter is called ideal because the transition from the passband to the stopband in the filter is perfect, it goes from 0 to 1 instantly. Although this type of filter is not realizable in physical systems, such as with electronic filters, it is a reality for digital image processing applications, where we need only multiply numbers in software. However, the ideal filter leaves undesirable artifacts in images. This artifact appears in the lowpass filtered

(a)

Original image

(b)

Filtered image, using a non-ideal lowpass filter. Note the blurring that softens the image.

(c)

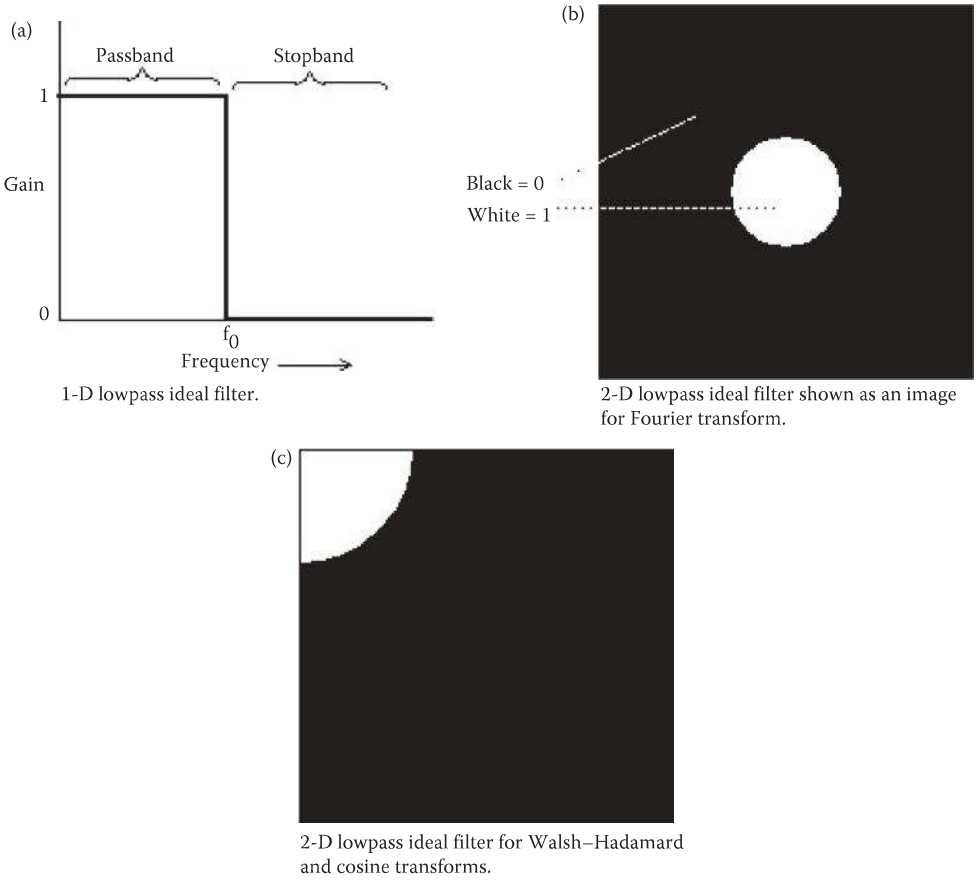
Ideal lowpass-filtered image shows the ripple artifacts at boundaries. Frequency cutoff = 32.

FIGURE 5.7-1
Lowpass filtering.

image in Figure 5.7-1c as ripples, or waves, wherever there is a boundary in the image. This problem can be avoided by using a "nonideal" filter that does not have perfect transition, as is shown in Figure 5.7-3. The image created in Figure 5.7-1b was generated using a non-ideal filter of a type called a Butterworth filter.

With the Butterworth filter we can specify the *order* of the filter, which determines how steep the slope is in the transition of the filter function. A higher order to the filter creates a steeper slope, and the closer we get to an ideal filter. The filter function of a Butterworth lowpass filter of order n is given by the following equation:

$$H(u,v) = \frac{1}{1 + \left[\dfrac{\sqrt{u^2 + v^2}}{f_0}\right]^{2n}}$$



1-D lowpass ideal filter.

2-D lowpass ideal filter shown as an image for Fourier transform.

2-D lowpass ideal filter for Walsh–Hadamard and cosine transforms.

FIGURE 5.7-2
Ideal lowpass filters.

Note that $\sqrt{u^2 + v^2}$ is the distance from the origin, so the gain falls off as we get farther away from the zero frequency term, which is what we expect for a lowpass filter, to cut the high frequencies. Also note that for the Fourier spectrum shifted to the center, but still indexing our matrix with (0,0) in the upper left corner (as we may do in a computer program), we need to replace u with $(u - N/2)$ and v with $(v - N/2)$ in the above equation.

In Figure 5.7-4 we compare the results of different orders of Butterworth filters. We see that as we get closer to an ideal filter, the blurring effect becomes more prominent due to the elimination of even partial high frequency information. Another effect that is most noticeable in the 8th-order filter, is the appearance of waves, or ripples, wherever boundaries occur in the image. This artifact is called ringing and increases as the Butterworth filter's order increases.

In Figure 5.7-5 we see the result of using a third order Butterworth filter, but decreasing the cutoff frequency. As the cutoff frequency is lowered the image becomes more and more blurry because we are keeping less and less of the high frequency information.

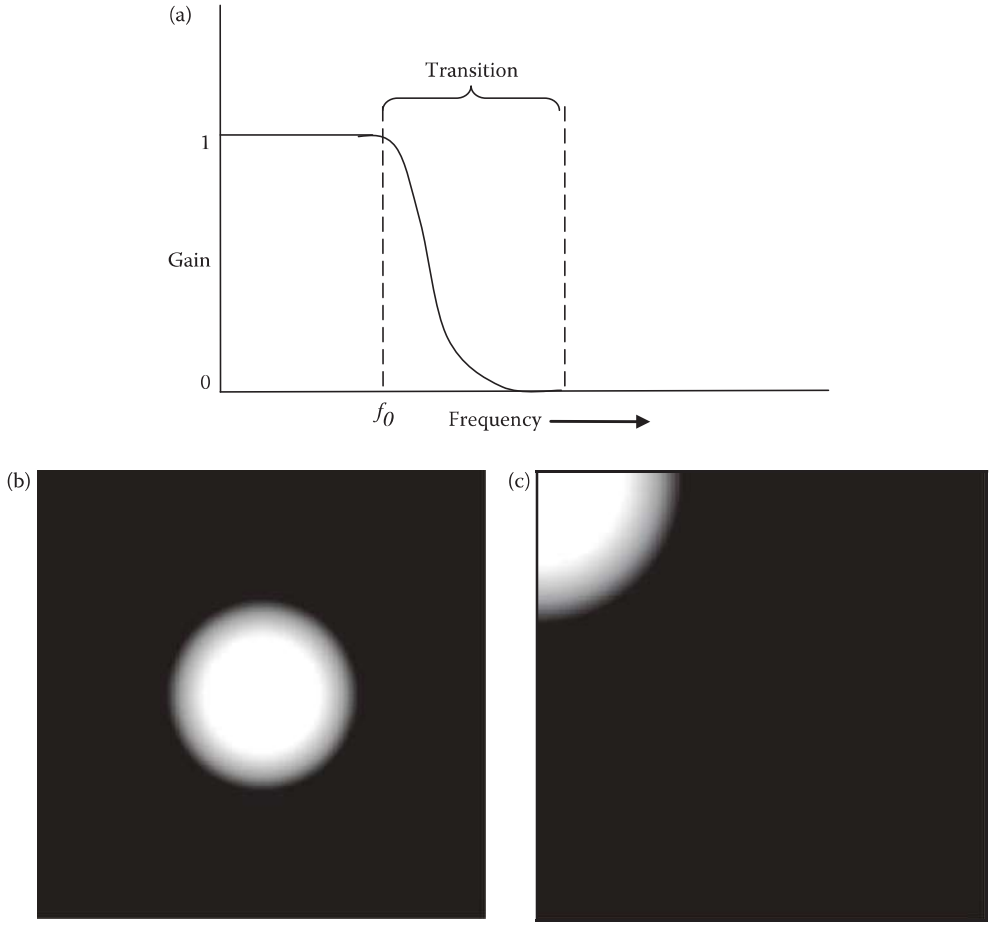


FIGURE 5.7-3
Nonideal lowpass filters. (a) 1-D nonideal filter, (b) 2-D lowpass nonideal filter shown as an image for Fourier symmetry, in the image shown black = 0, white = 1, and the gray values in between represent the transition band, (c) 2-D lowpass nonideal filter shown as an image for cosine and Walsh–Hadamard symmetry, in the image shown black = 0, white = 1, and the gray values in between represent the transition band.

5.7.2 Highpass Filters

Highpass filters will keep high frequency information, which corresponds to areas of rapid change in brightness, such as edges or fine textures. The highpass filter functions are shown in Figure 5.7-6, where we see both ideal and Butterworth filter functions. A highpass filter can be used for edge enhancement, since it passes only high frequency information, corresponding to places where gray levels are changing rapidly (edges in images are characterized by rapidly changing gray levels). The Butterworth filter of order n for the highpass filter is

$$H(u,v) = \frac{1}{1 + \left[\dfrac{f_0}{\sqrt{u^2 + v^2}}\right]^{2n}}$$

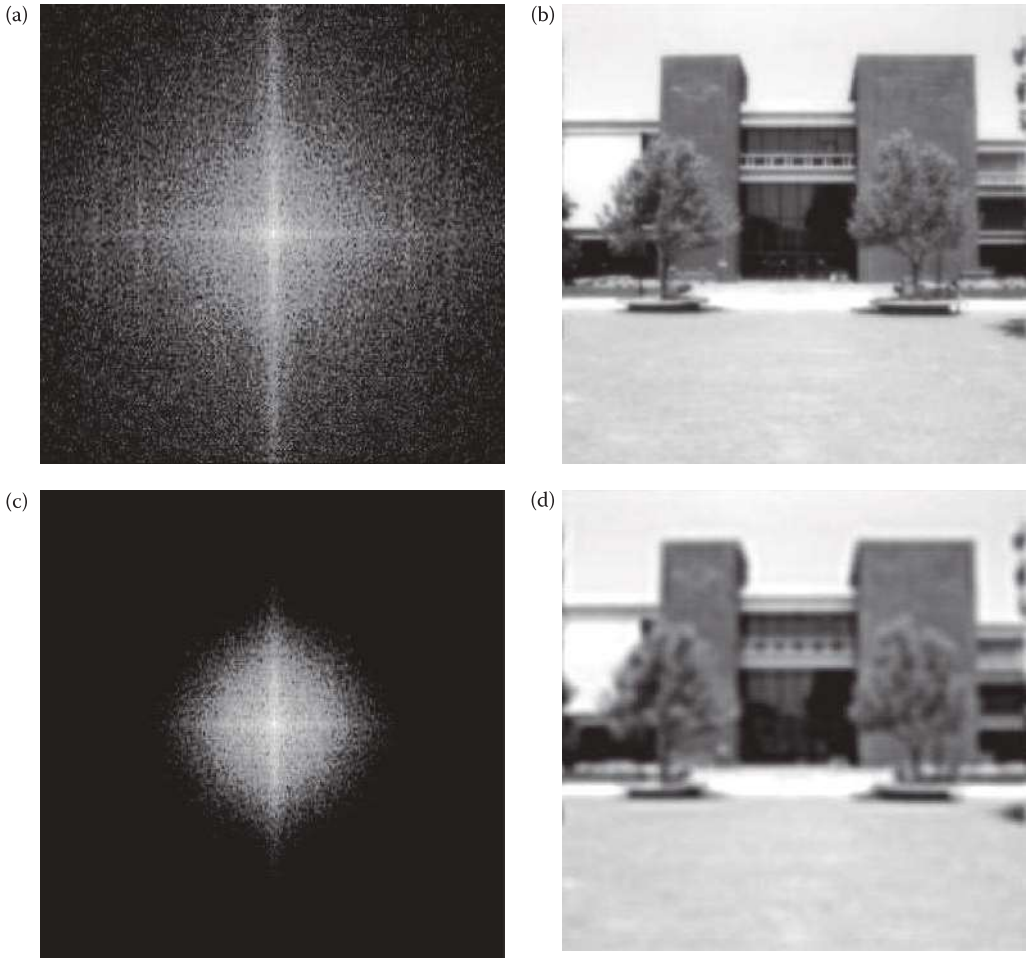


FIGURE 5.7-4
Lowpass butterworth filters. (a) Fourier spectrum, filter order = 1, (b) resultant image with order = 1, (c) Fourier spectrum, filter order = 3, (d) resultant image with order = 3, (e) Fourier spectrum, filter order = 5, (f) resultant image with order= 5, (g) Fourier spectrum, filter order = 8, (h) resultant image with order = 8.

Note that this filter gain is very small for frequencies much smaller than f_0, and approaches a gain of one as the frequencies get much larger than f_0. Also note that for the Fourier spectrum shifted to the center, but still indexing our matrix with (0,0) in the upper left corner (as we may do in a computer program), we need to replace u with $(u - N/2)$ and v with $(v - N/2)$ in the above equation.

The function for a special type of highpass filter, called a high frequency emphasis filter, is shown in Figure 5.7-7. This filter function boosts the high frequencies and retains some of the low frequency information and by adding an offset value to the function, so we do not lose the overall image information. The results from applying these types of filters are shown in Figure 5.7-8. The original is shown in Figure 5.7-8a. Figure 5.7-8b and c shows the results from a Butterworth and an ideal filter function. Here we can see the edges enhanced, and the ripples that occur from using an ideal filter (Figure 5.7-8c), but note a loss in the overall contrast of the image. In Figure 5.7-8d and e, we see the contrast

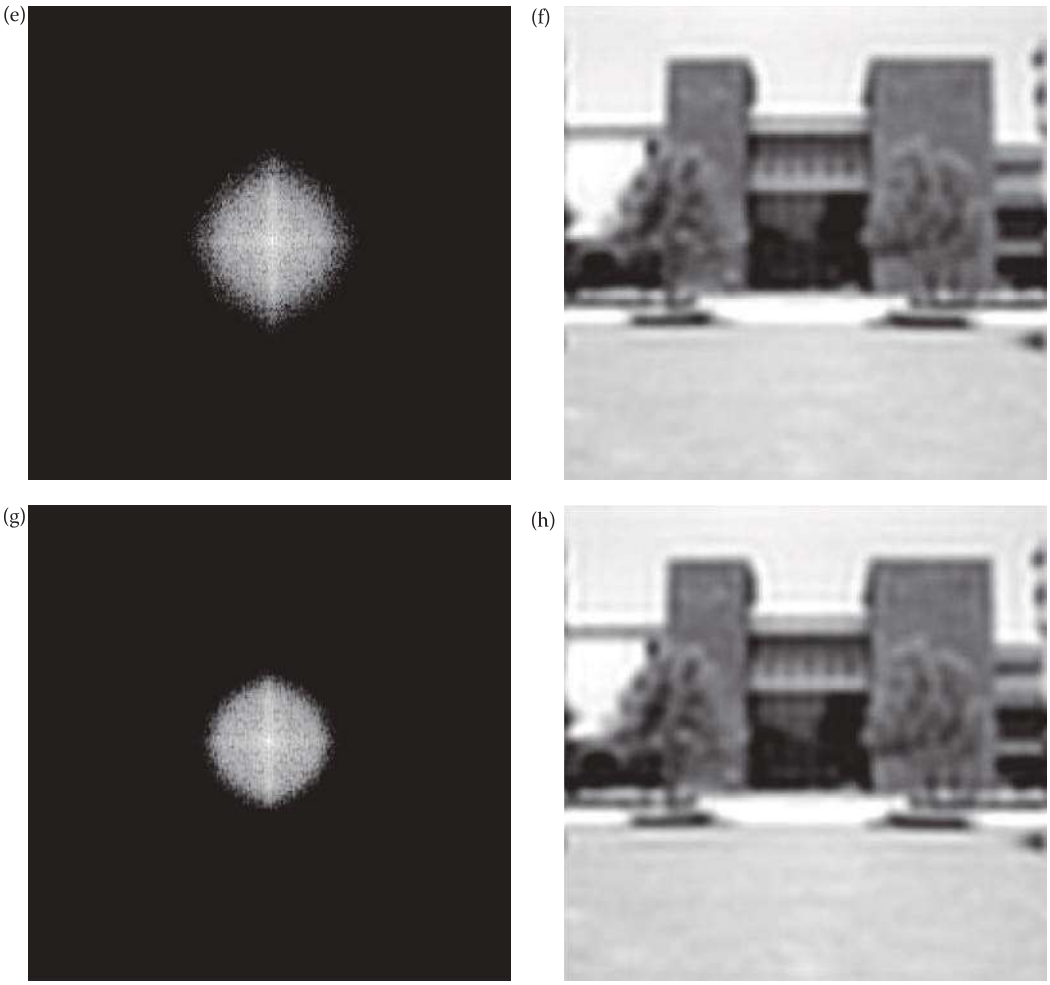


FIGURE 5.7-4 (CONTINUED)
Lowpass butterworth filters. (a) Fourier spectrum, filter order = 1, (b) resultant image with order = 1, (c) Fourier spectrum, filter order = 3, (d) resultant image with order = 3, (e) Fourier spectrum, filter order = 5, (f) resultant image with order= 5, (g) Fourier spectrum, filter order = 8, (h) resultant image with order = 8.

added back to the image by using the high frequency emphasis filter function. This is because we kept more of the low frequency information from the original image.

5.7.3 Bandpass and Bandreject Filters

The bandpass and bandreject filters are specified by two cutoff frequencies, a low cutoff and a high cutoff, shown in Figure 5.7-9. These filters can be modified into nonideal filters by making the transitions gradual at the cutoff frequencies, as was shown for the lowpass filter in Figure 5.7-3 and the highpass in Figure 5.7-6. A special form of these filters is called a notch filter, because it only notches out, or passes, specific frequencies (see Figure 5.7-9g and h). These filters are useful for retaining bandpass, or removing bandreject, specific frequencies of interest that are typically application dependent—one common application

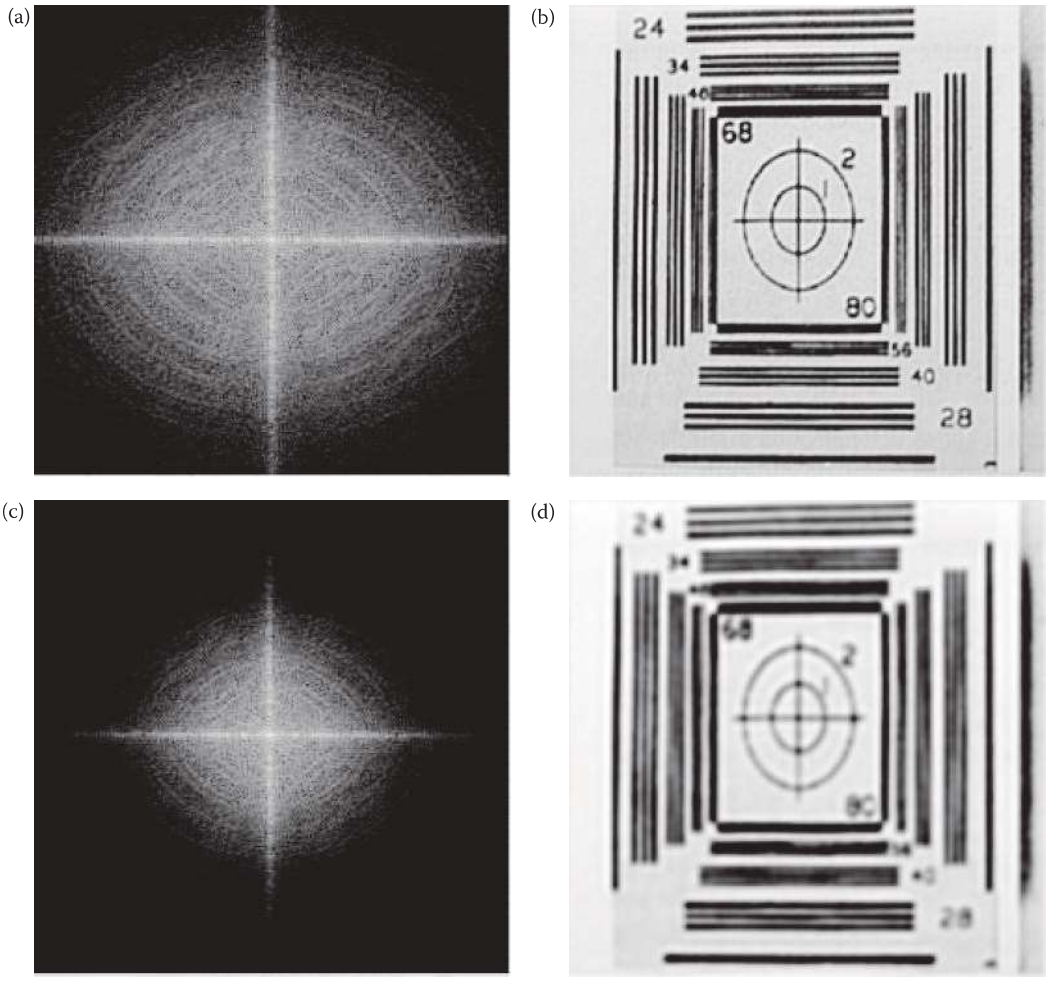


FIGURE 5.7-5
Butterworth lowpass filtering, filter order = 3, various cutoff frequencies. (a) Fourier spectrum, cutoff frequency = 64, (b) resultant image with cutoff frequency = 64, (c) Fourier spectrum, cutoff frequency = 32, (d) resultant image with cutoff frequency = 32, (e) Fourier spectrum, cutoff frequency = 16, (f) resultant image with cutoff frequency = 16 (g) Fourier spectrum, cutoff frequency = 8, (h) resultant image with cutoff frequency = 8.

is for noise removal. These three types of filters are typically used in image restoration, enhancement, and compression, and examples can be seen in Chapters 8, 9, and 10.

5.8 Discrete Wavelet Transform

The wavelet transform is really a family of transforms that satisfy specific conditions. From our perspective we can describe the *wavelet transform* as a transform that has basis functions that are shifted and expanded versions of themselves. Because of this, the

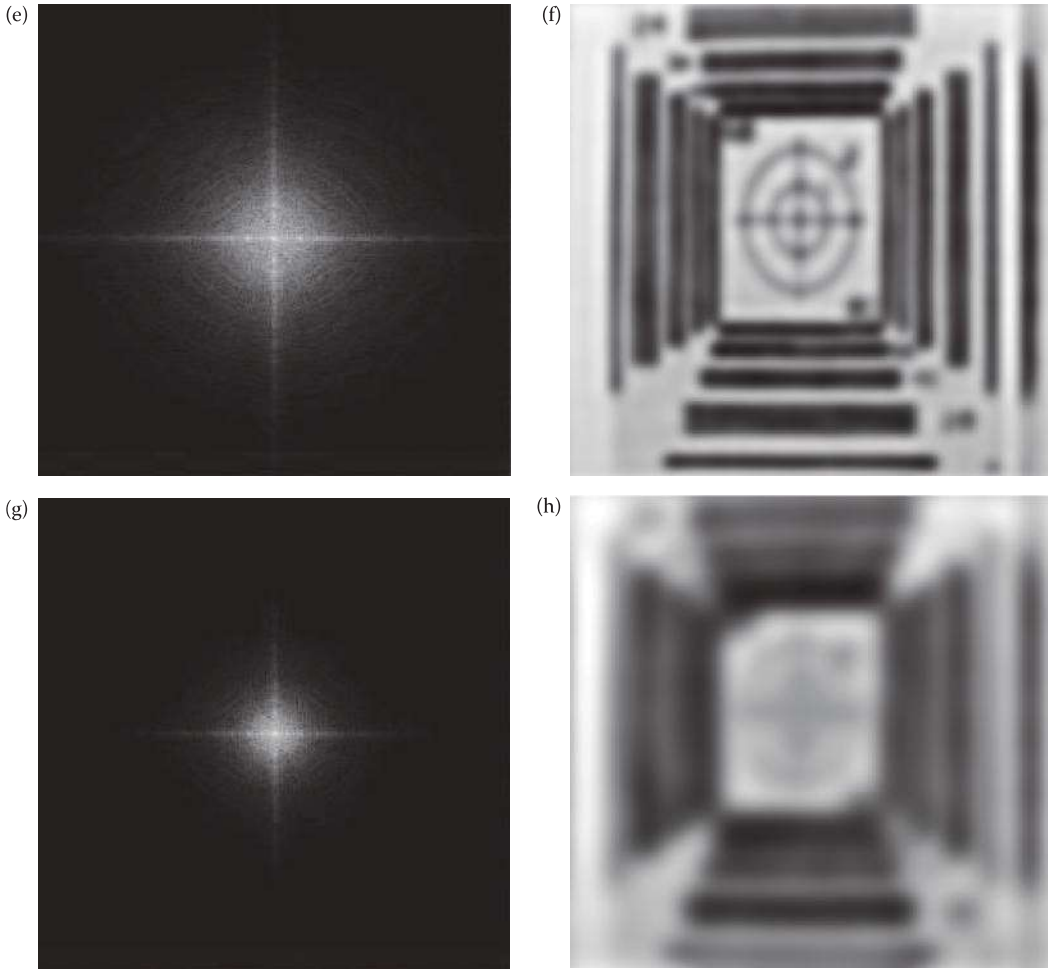


FIGURE 5.7-5 (CONTINUED)
Butterworth lowpass filtering, filter order = 3, various cutoff frequencies. (a) Fourier spectrum, cutoff frequency = 64, (b) resultant image with cutoff frequency = 64, (c) Fourier spectrum, cutoff frequency = 32, (d) resultant image with cutoff frequency = 32, (e) Fourier spectrum, cutoff frequency = 16, (f) resultant image with cutoff frequency = 16 (g) Fourier spectrum, cutoff frequency = 8, (h) resultant image with cutoff frequency = 8.

wavelet transform contains not just frequency information, but spatial information as well. Additionally, for application to digital images, we will need to use the *discrete* wavelet transform.

One of the most common models for a wavelet transform uses the Fourier transform and highpass and lowpass filters. To satisfy the conditions for a wavelet transform, the filters must be *perfect reconstruction filters*, which means that any distortion introduced by the forward transform will be canceled in the inverse transform (an example of these types of filters are *quadrature mirror filters*). For the specific examples explored here we will perform the filtering in the spatial domain, with convolution filters.

The discrete wavelet transform breaks an image down into four subsampled, or decimated, images. They are subsampled by keeping every other pixel. The results consist of

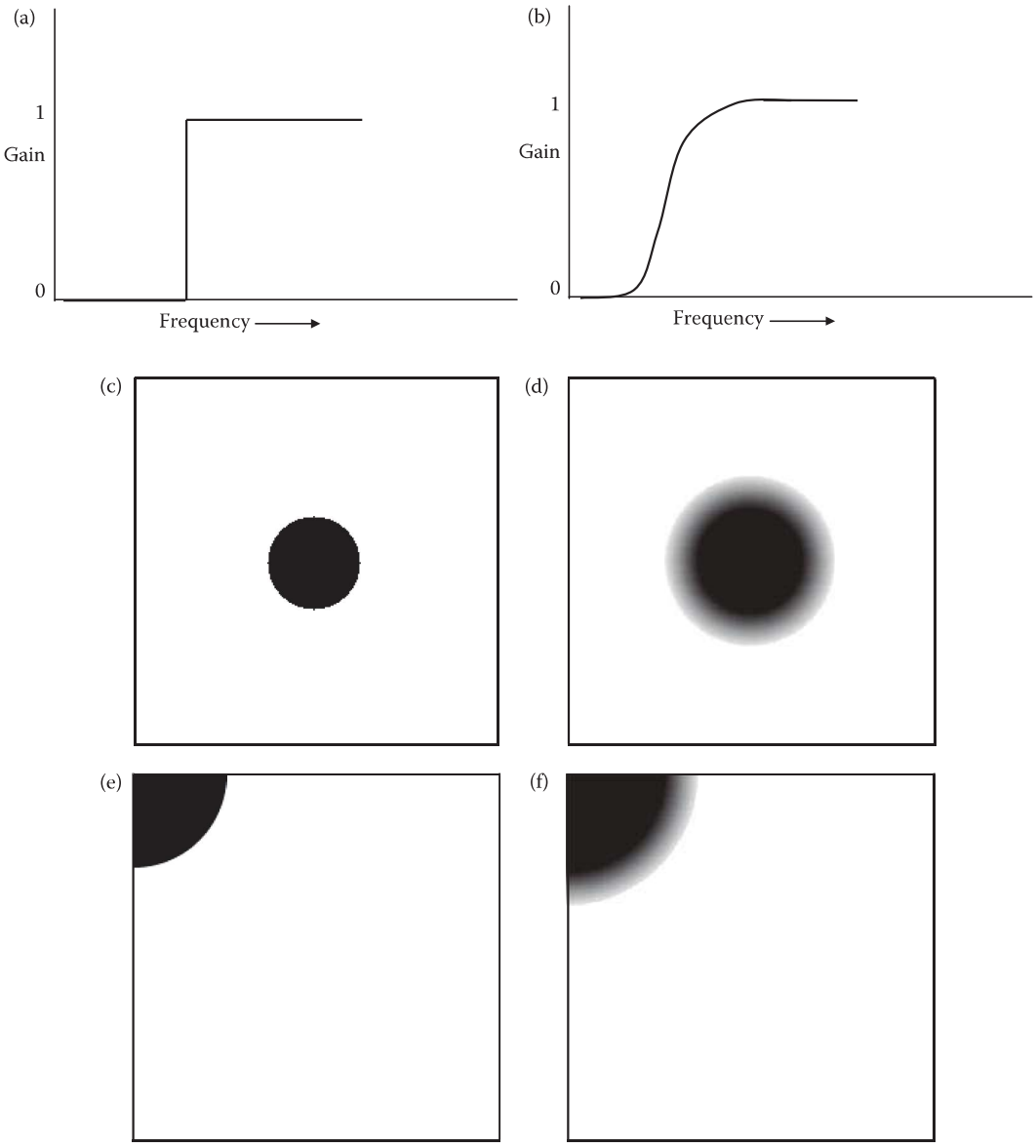


FIGURE 5.7-6
Highpass filter functions. (a) 1-D ideal highpass filter, (b) 1-D non-ideal highpassfilter, (c) 2-D ideal highpass filter for Fourier symmetry, shownas an image, (d) 2-D nonideal highpass filter for Fourier symmetry, shown as an image, (e) 2-D ideal highpass filter for cosine and Walsh–Hadamard symmetry, (f) 2-D nonideal highpass filter for cosine and Walsh–Hadamard symmetry. Note: for the filters shown as images, white =1, black = 0, and gray values in between represent values between 0 and 1, corresponding to the transition band in nonideal filters.

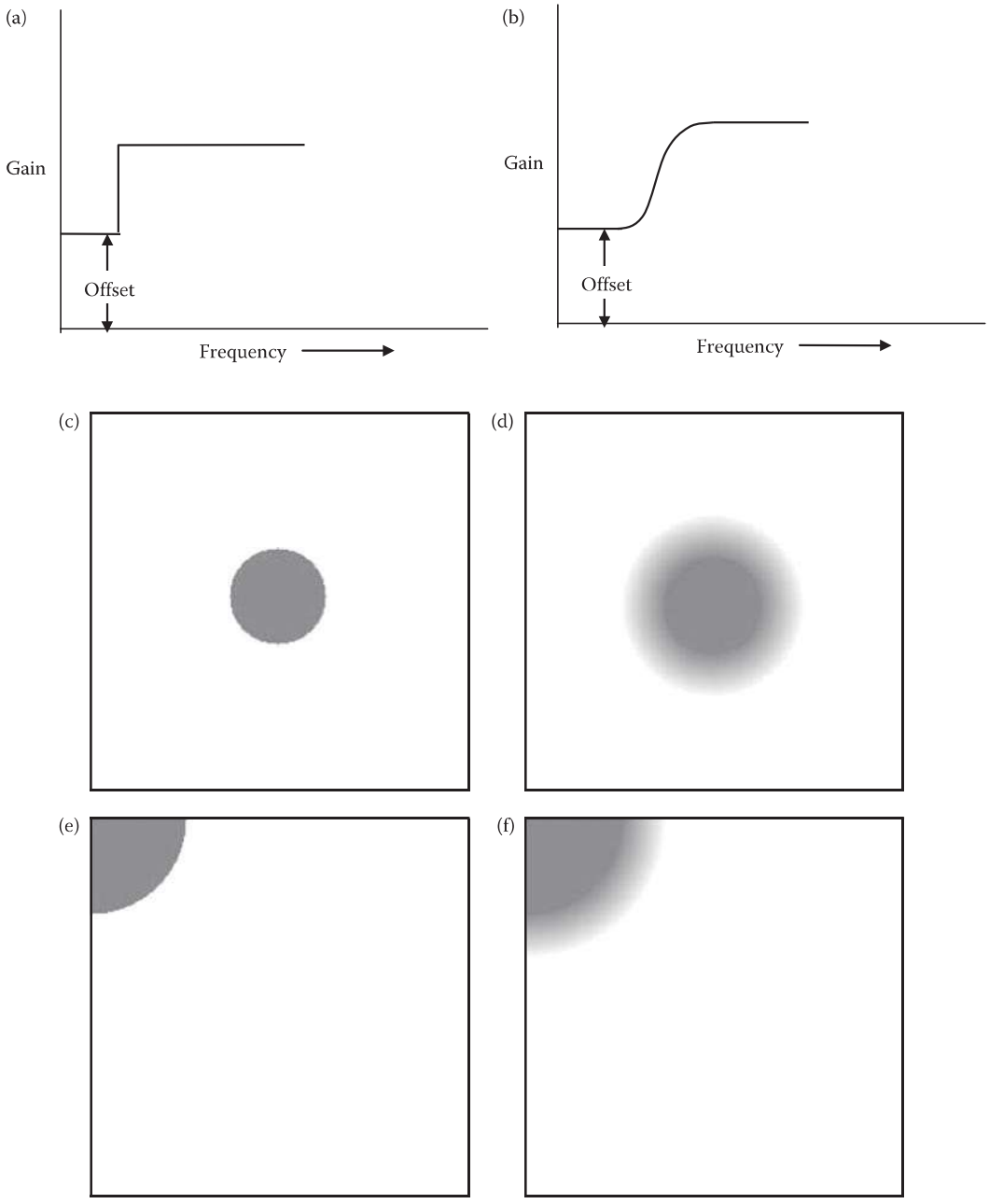


FIGURE 5.7-7
High frequency emphasis filter functions. (a) 1-D ideal high frequency emphasis filter, (b) 1-D nonideal high-frequency emphasis filter, (c) 2-D ideal high frequency emphasis filter for Fourier symmetry, shown as an image, (d) 2-D nonideal high frequency emphasis filter for Fourier symmetry, (e) 2-D ideal high frequency emphasis filter for cosine and Walsh–Hadamard symmetry, (f) 2-D nonideal high frequency emphasis filter for cosine and Walsh–Hadamard symmetry.

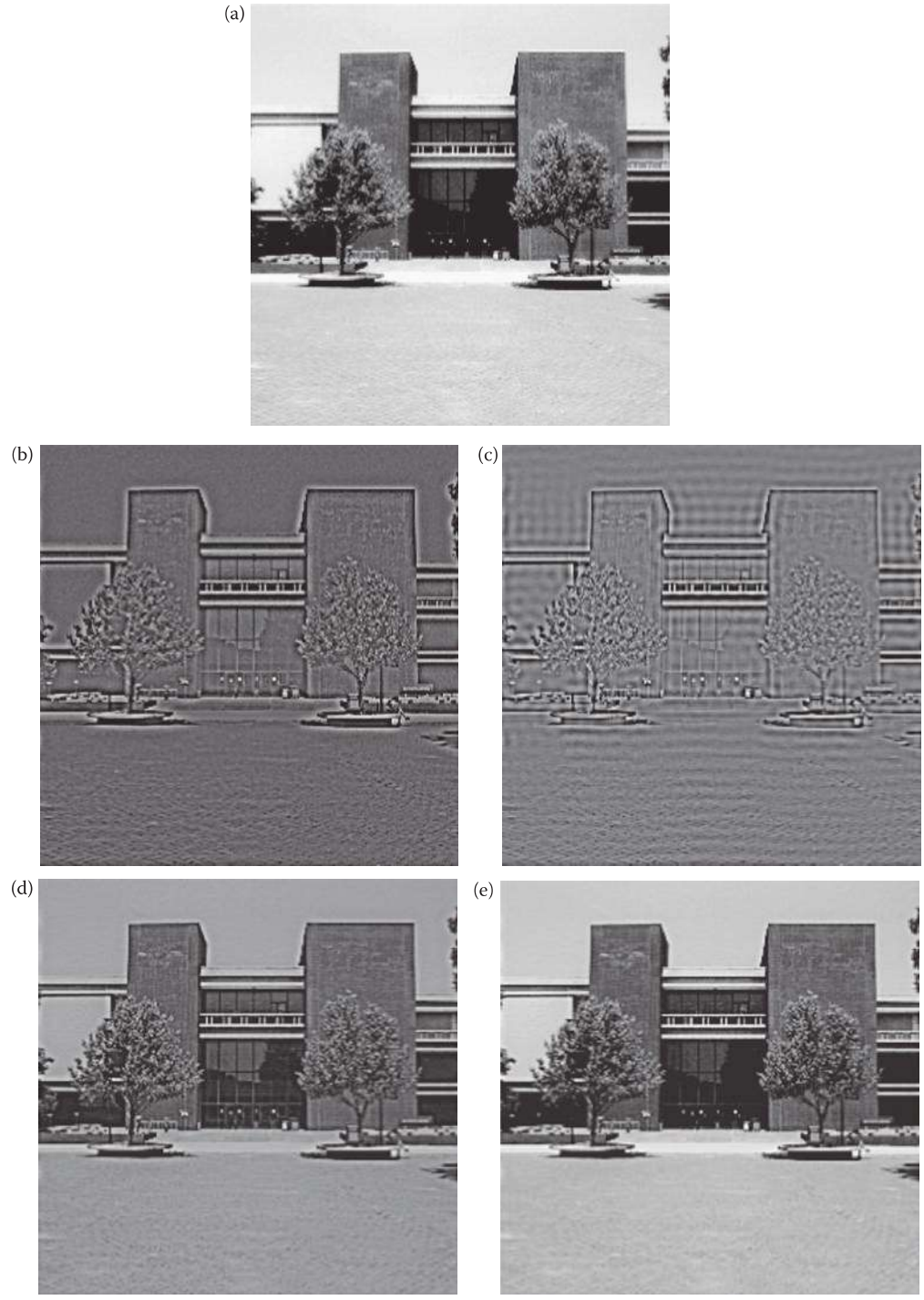


FIGURE 5.7-8
Highpass filtering. (a) Original image, (b) Butterworth filter; order = 2; cutoff = 32, (c) ideal filter; cutoff = 32, (d) high frequency emphasis filter; offset = 0.5, order = 2, cutoff = 32, (e) high frequency emphasis filter; offset = 1.5, order = 2, cutoff = 32. Note that the high frequency emphasis filter retains more of the original image.

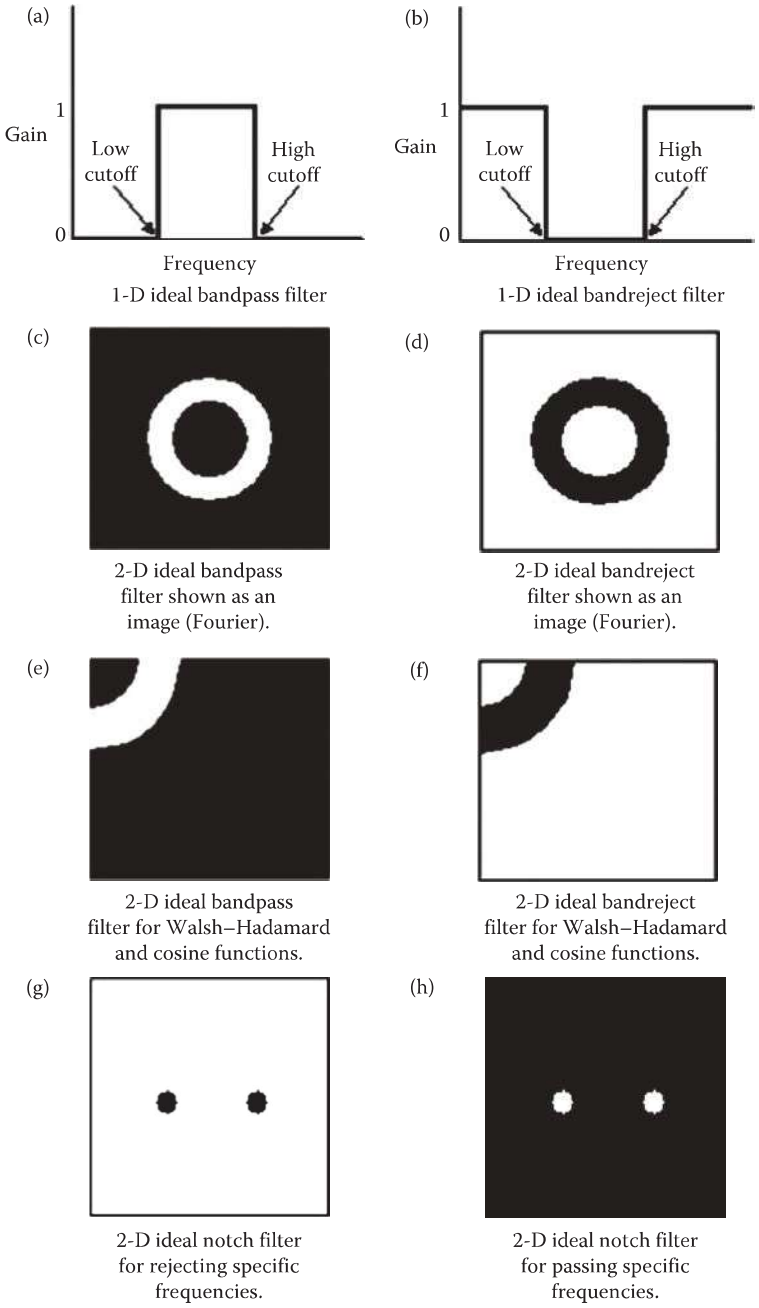


FIGURE 5.7-9
Bandpass, bandreject, and notch filters.

one image that has been highpass filtered in both the horizontal and vertical directions, one that has been highpass filtered in the vertical and lowpassed in the horizontal, one that has been highpassed in the horizontal and lowpassed in the vertical, and one that has been lowpass filtered in both directions.

This transform is typically implemented in the spatial domain by using 1-D convolution filters. In the section on edge detection we looked at 2-D convolution masks that mark places in the image where the gray levels are changing rapidly. These rapid changes correspond to high frequency information, so edge detectors are basically highpass filters. To do this we apply the convolution theorem that is an important Fourier transform property. As we have seen, the *convolution theorem* states that convolution in the spatial domain is the equivalent of multiplication in the frequency domain. We have seen that multiplication in the frequency domain is used to perform filtering; the convolution theorem tells us that we can also perform filtering in the spatial domain via convolution, such as we have already seen with spatial convolution masks. Therefore, if we can define convolution masks that satisfy the wavelet transform conditions, we can implement the wavelet transform in the spatial domain. We have also seen that if the transform basis functions are separable, we can perform the 2-D transform by using two 1-D transforms. An additional benefit of convolution versus frequency domain filtering is that, if the convolution mask is short, it is much faster.

In order to perform the wavelet transform with convolution filters, a special type of convolution called circular convolution must be used. *Circular convolution* is performed by taking the underlying image array and extending it in a periodic manner to match the symmetry implied by the DFT (see Figure 5.8-1a and 5.8-1b). The convolution process

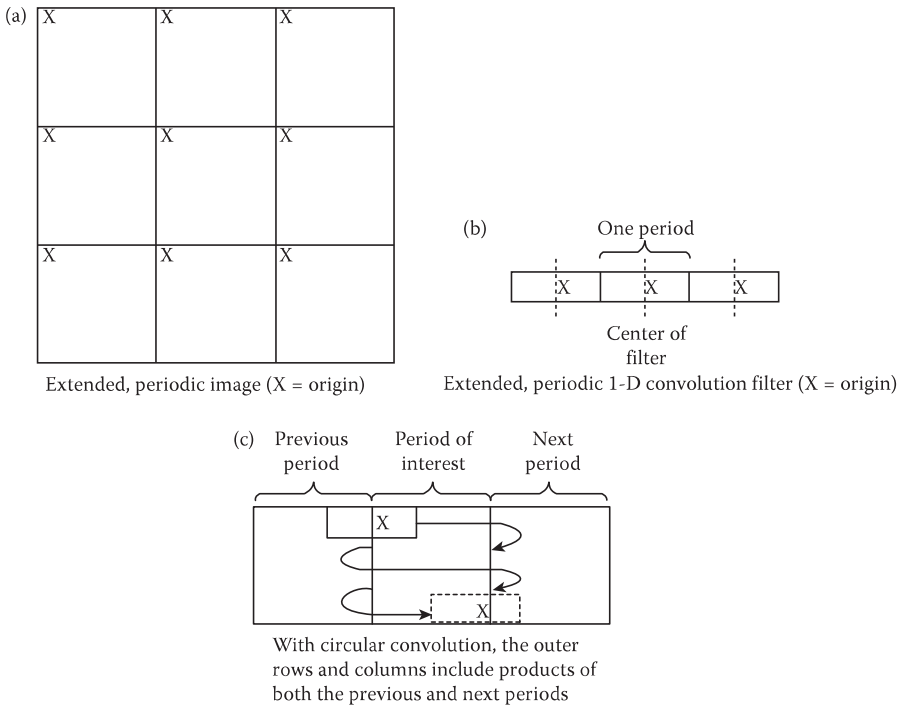


FIGURE 5.8-1
Circular convolution.

starts with the origin of the image and the convolution mask aligned, so that the first value contains contributions from the "previous" copy of the periodic image (see Figure 5.8-1c). In Figure 5.8-1c we see that the last value(s) contain contributions from the "next" copy of the extended, periodic image. Performing circular convolution allows us to retain the outer rows and columns, unlike the previously used method where the outer rows and columns where ignored. This is important since we may want to perform the wavelet transform on small blocks, and eliminating the outer row(s) and column(s) is not practical.

Many different convolution filters are available for use with the wavelet transform. Here we will consider two examples based on the Daubechies and the Haar functions. These are separable, so they can be used to implement a wavelet transform by first convolving them with the rows and then the columns. The Haar basis vectors are simple:

$$LOWPASS: \quad \frac{1}{\sqrt{2}}[1 \quad 1]$$

$$HIGHPASS: \quad \frac{1}{\sqrt{2}}[1 \quad -1]$$

An example of Daubechies basis vectors are

$$LOWPASS: \quad \frac{1}{4\sqrt{2}}[1+\sqrt{3}, \quad 3+\sqrt{3}, \; 3-\sqrt{3}, \; 1-\sqrt{3}]$$

$$HIGHPASS: \quad \frac{1}{4\sqrt{2}}[1-\sqrt{3}, \; \sqrt{3}-3, \; 3+\sqrt{3}, \; -1-\sqrt{3}]$$

To use the basis vectors to implement the wavelet transform they must be zero-padded to be the same size as the image (or subimage). Also note that the origin of the basis vectors is in the center, corresponding to the value to the right of the middle of the vector.

Example 5.8.1

We want to use the Haar basis vectors to perform a wavelet transform on an image by dividing it into 4×4 blocks. The basis vectors need to be zero-padded so that they have a length of 4, as follows:

$$LOWPASS: \quad \frac{1}{\sqrt{2}}[1 \quad 1 \quad 0 \quad 0]$$

$$HIGHPASS: \quad \frac{1}{\sqrt{2}}[1 \quad -1 \quad 0 \quad 0]$$

$$\uparrow$$

origin

These are aligned with the image so that the origins coincide, and the result from the first vector inner product is placed into the location corresponding to the origin. Note that when the vector is zero-padded on the right, the origin is no longer to the right of the center of the resulting vector. The origin is determined by selecting the coefficient corresponding to the right of center *before* zero-padding.

Example 5.8.2

To use the Daubechies basis vectors to do a wavelet transform on an image by dividing it into 8×8 blocks, we need to zero-pad them to a length of 8, as follows:

$$\textit{LOWPASS}: \quad \frac{1}{4\sqrt{2}}\,[1+\sqrt{3},\ \ 3+\sqrt{3},\ 3-\sqrt{3},\ 1-\sqrt{3},\ \ 0,\ \ 0,\ \ 0,\ \ 0]$$

$$\textit{HIGHPASS}: \quad \frac{1}{4\sqrt{2}}\,[1-\sqrt{3},\ \sqrt{3}-3,\ 3+\sqrt{3},\ -1-\sqrt{3},\ \ 0,\ \ 0,\ \ 0,\ \ 0]$$

$$\uparrow$$

origin

Note that the origin is the value to the right of the center of the original vector before zero-padding. Since these are assumed periodic for circular convolution, we could zero-pad equally on both ends, then the origin *is* to the right of the center of the zero-padded vector, as follows:

$$\textit{LOWPASS}: \quad \frac{1}{4\sqrt{2}}\,[0,\ \ 0,\ \ 1+\sqrt{3},\ \ 3+\sqrt{3},\ 3-\sqrt{3},\ 1-\sqrt{3},\ \ 0,\ \ 0]$$

$$\textit{HIGHPASS}: \quad \frac{1}{4\sqrt{2}}\,[0,\ \ 0,\ \ 1-\sqrt{3},\ \sqrt{3}-3,\ 3+\sqrt{3},\ -1-\sqrt{3},\ \ 0,\ \ 0]$$

$$\uparrow$$

origin

After the basis vectors have been zero-padded (if necessary), the wavelet transform is performed by doing the following:

1. Convolve the lowpass filter with the rows (remember that this is done by sliding, multiplying coincident terms, and summing the results) and save the results. (Note: For the basis vectors as given, they do *not* need to be reversed for convolution.)
2. Convolve the lowpass filter with the columns (of the results from Step 1), and subsample this result by taking every other value; this gives us the lowpass–lowpass version of the image.
3. Convolve the result from Step 1, the lowpass filtered rows, with the highpass filter on the columns. Subsample by taking every other value to produce the lowpass–highpass image.
4. Convolve the original image with the highpass filter on the rows, and save the result.
5. Convolve the result from Step 4 with the lowpass filter on the columns; subsample to yield the highpass–lowpass version of the image.
6. To obtain the highpass–highpass version, convolve the columns of the result from Step 4 with the highpass filter.

In practice the convolution sum of every other pixel is not performed, since the resulting values are not used. This is typically done by shifting the basis vector by 2, instead of by 1 at each convolution step. Note that with circular convolution the basis vector will overlap the extended periodic copies of the image when both the first and last convolution sums are calculated.

The convention for displaying the wavelet transform results, as an image, is shown in Figure 5.8-2. In Figure 5.8-3, we see the results of applying the wavelet transform to an image. In Figure 5.8-3b we can see the lowpass–lowpass image in the upper left corner, the lowpass–highpass images on the diagonals, and the highpass–highpass in the lower right corner. We can continue to run the same wavelet transform on the lowpass–lowpass version of the image to get seven subimages, as in Figure 5.8-3c, or perform it another time to get ten subimages, as in Figure 5.8-3d. This process is called *multiresolution decomposition*, and can continue to achieve 13, 16, or as many subimages as are practical. The *decomposition level* refers to how many times we have performed the wavelet transform, where each successive decomposition level means that the wavelet is performed on the lowpass-lowpass version of the image as shown in Figure 5.8-3.

We can see in the resulting images that the transform contains spatial information, as the image itself is still visible in the transform domain. This is similar to what we saw with the Haar transform (Figure 5.5-1), but the Fourier, Walsh–Hadamard, and cosine spectrum does not necessarily have any visible correlation to the image itself when performed on the entire image (Figure 5.8-4). However, if we perform these transforms using small blocks, the resulting spectrum will resemble the image primarily due to the zero frequency term's magnitude (Figure 5.8-5).

The inverse wavelet transform is performed by enlarging the wavelet transform data to its original size. Insert zeros between each value, convolve the corresponding (lowpass and highpass) inverse filters to each of the four subimages, and sum the results to obtain the original image. For the Haar filter, the inverse wavelet filters are identical to the forward filters; for the Daubechies example given, the inverse wavelet filters are

$$LOWPASS_{inv}: \frac{1}{4\sqrt{2}}[3-\sqrt{3},\quad 3+\sqrt{3},\ 1+\sqrt{3},\ 1-\sqrt{3}]$$

$$HIGHPASS_{inv}: \frac{1}{4\sqrt{2}}[1-\sqrt{3},\quad -1-\sqrt{3},\ 3+\sqrt{3},\quad -3+\sqrt{3}]$$

Low/ Low	High/ Low
Low/ High	High/ High

FIGURE 5.8-2
Wavelet transform display. Location of frequency bands in a four-band wavelet transformed image. Designation is row/column.

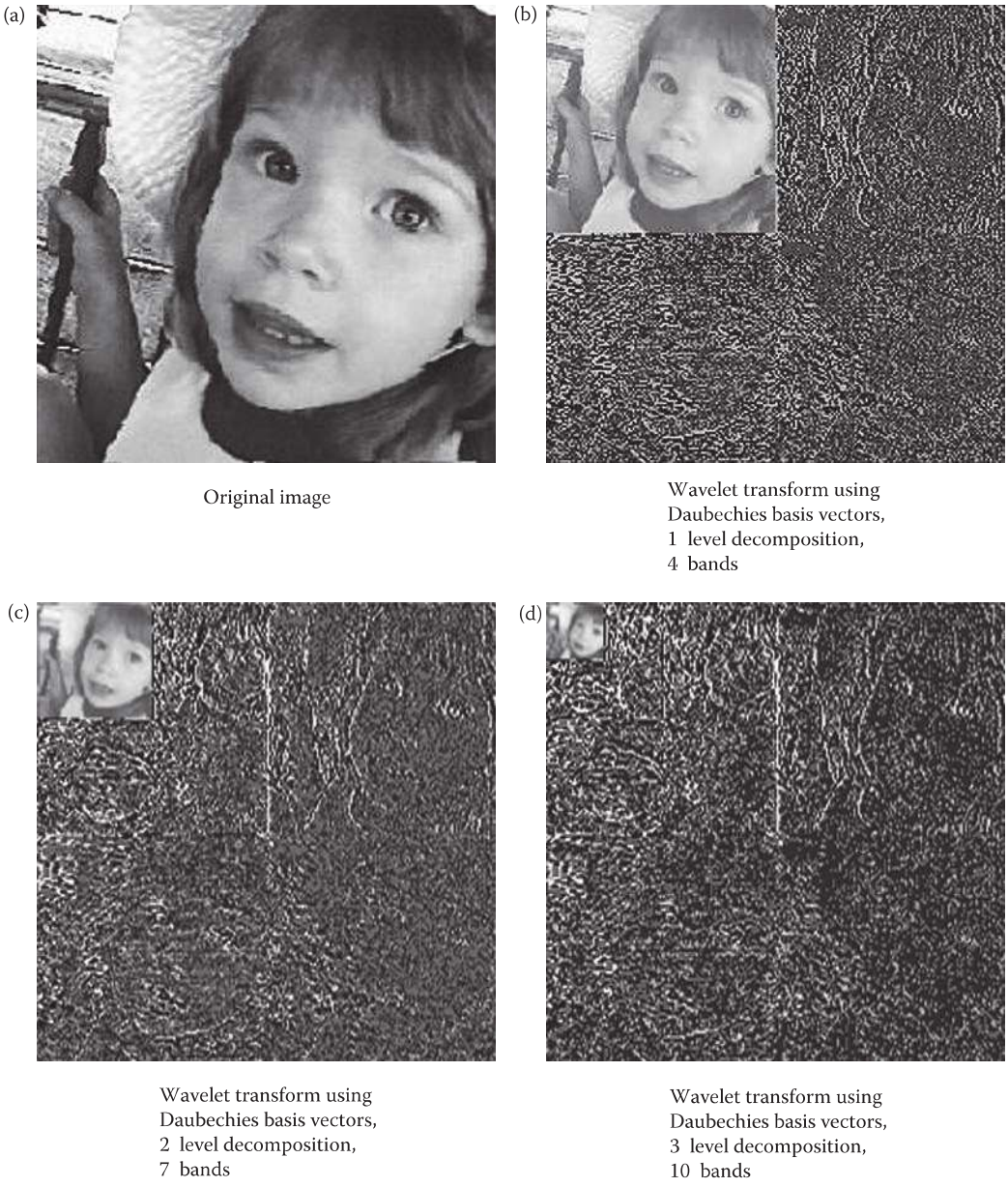


FIGURE 5.8-3
Wavelet transform.

The use of the wavelet transform is increasingly popular for image compression, a very active research area today. The computer revolution, along with the increasing ubiquity of the Internet, multimedia applications, and high definition television, all contribute to the high level of interest in image compression. The multiresolution decomposition property of the wavelet transform, which separates low-resolution information from more detailed

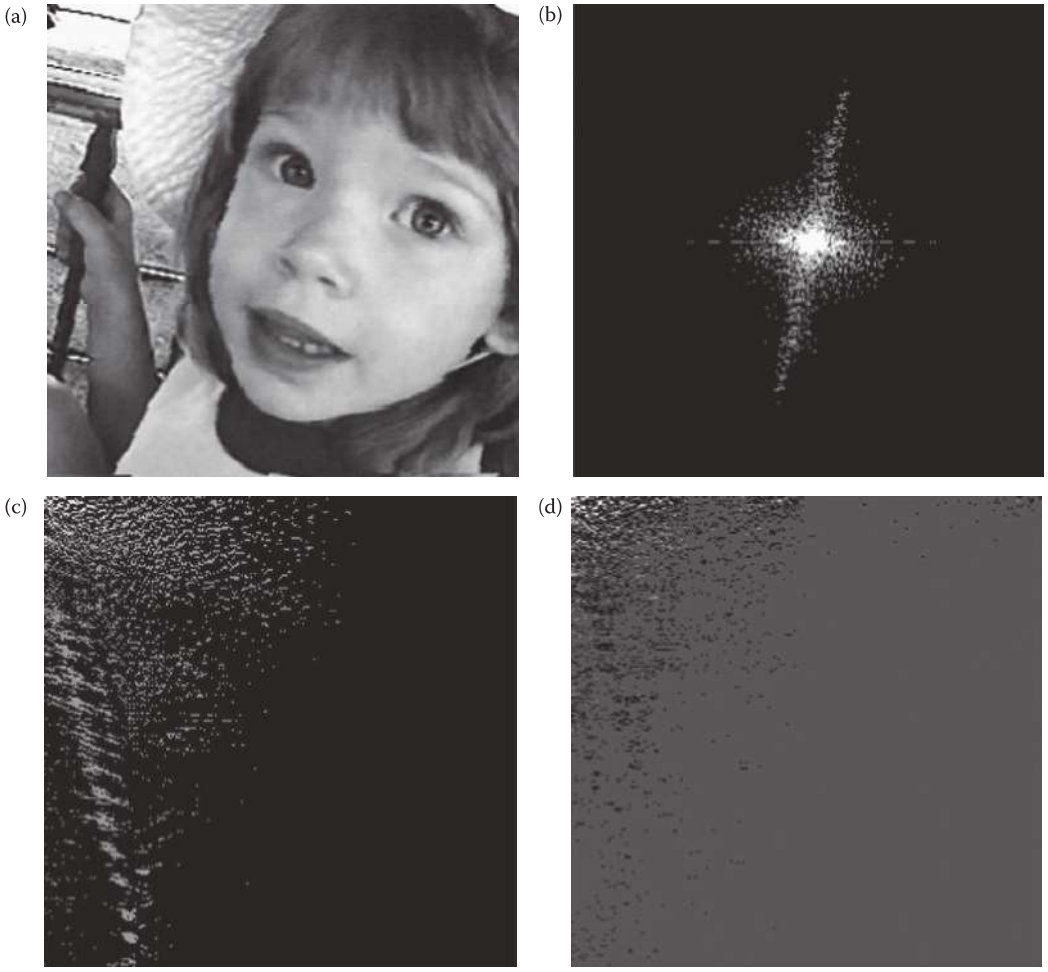


FIGURE 5.8-4
Fourier, Walsh–Hadamard, cosine spectra contain no obvious spatial information, (a) Original image, (b) Fourier spectrum, (c) Walsh–Hadamard spectrum, (d) cosine spectrum.

information, makes it useful in applications where it is desirable to have coarse information available fast such as perusing an image database or progressively transmitting images on the Internet. The wavelet transform is one of the relatively new transforms being explored for image compression applications; as mentioned before it is used in the JPEG compression standard JPEG2000.

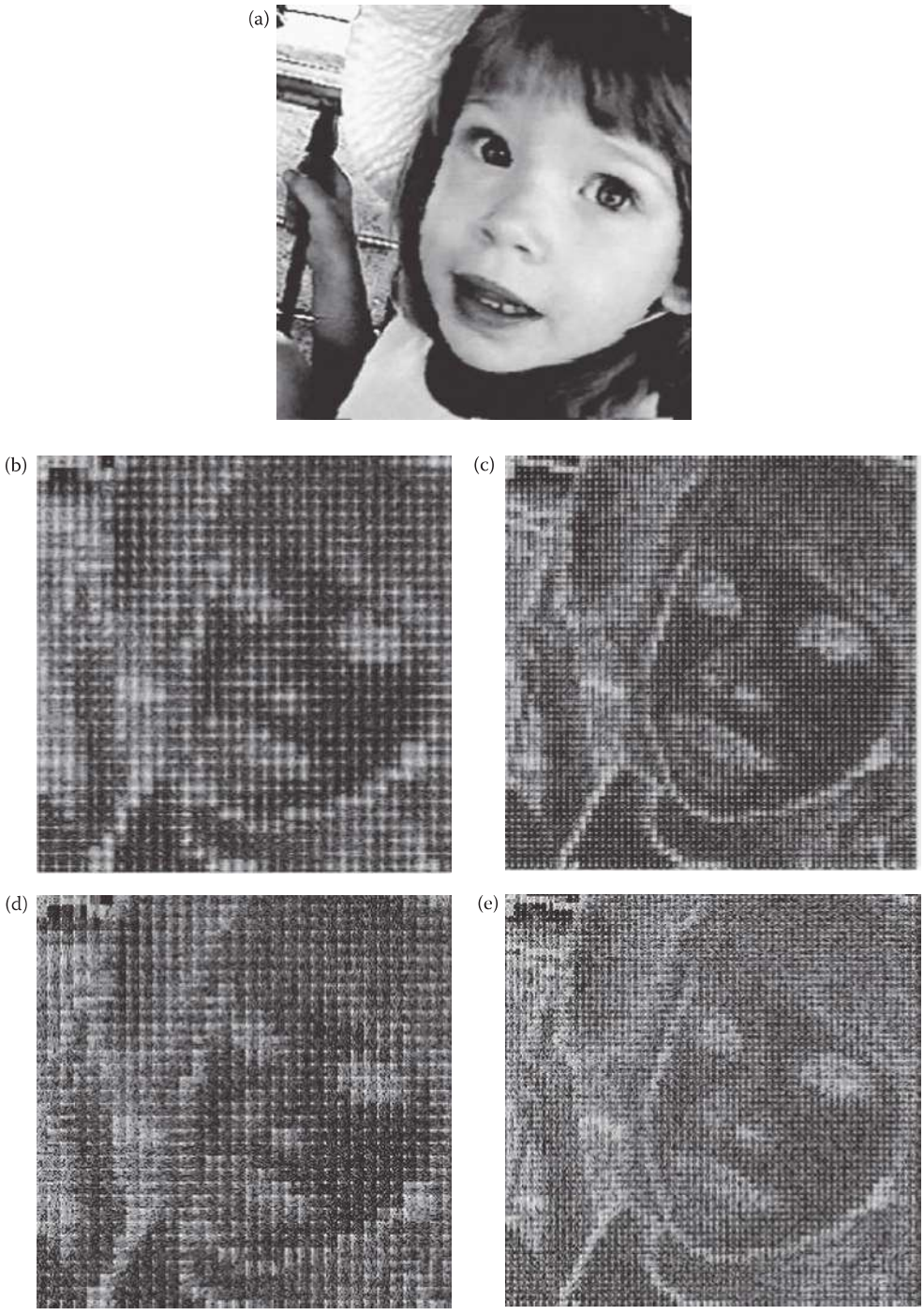


FIGURE 5.8-5
Fourier, Walsh–Hadamard, cosine spectra performed on small blocks resembles image. (a) Original image, (b) Fourier spectrum 8×8 blocks, (c) Fourier spectrum 4×4 blocks, (d) Walsh–Hadamard spectrum 8×8 blocks, (e) Walsh–Hadamard spectrum 4×4 blocks, (f) cosine spectrum, 8×8 blocks, (g) cosine spectrum 4×4 blocks. *Note: all spectra are log remapped.*

(f)  (g) 

FIGURE 5.8-5 (CONTINUED)
Fourier, Walsh–Hadamard, cosine spectra performed on small blocks resembles image. (a) Original image, (b) Fourier spectrum 8 × 8 blocks, (c) Fourier spectrum 4 × 4 blocks, (d) Walsh–Hadamard spectrum 8 × 8 blocks, (e) Walsh–Hadamard spectrum 4 × 4 blocks, (f) cosine spectrum, 8 × 8 blocks, (g) cosine spectrum 4 × 4 blocks. *Note: all spectra are log remapped.*

5.9 Key Points

OVERVIEW: DISCRETE TRANSFORMS

- Most of the discrete transforms provide information regarding spatial frequency content of an image.
- The principal component transform decorrelates multiband image data.
- The wavelet and the Haar transforms retain both spatial and frequency information.
- Most of the discrete transforms map image data into a frequency or sequency mathematical space where all the pixels contribute to each value in the transform domain.
- Spatial frequency and sequency relate to how brightness levels change relative to spatial coordinates.
- Frequency is the term for sinusoidal transforms, sequency for rectangular wave transforms.
- Rapidly changing brightness values correspond to high frequency (or sequency) terms, slowly changing brightness values correspond to low frequency (or sequency) terms.
- A constant brightness value is called the zero frequency (sequency) term, or the DC term.
- Most of the discrete transforms decompose an image into a weighted sum of basis images.
- Basis images are two-dimensional (2-D) versions of basis vectors.
- Basis vectors are sampled versions of basis functions.

- The weights for the basis images are found by projecting the basis image onto the image being transformed.
- Mathematically, the projection process is performed by calculating the vector inner product of the basis image and the image being transformed.
- Basis images should be orthogonal and orthonormal.
- Orthogonal basis images have vector inner products equal to zero—they have nothing in common, they are uncorrelated.
- Orthonormal basis images are orthogonal and have magnitudes of one.

FOURIER TRANSFORM

- The Fourier transform decomposes an image into complex sinusoidal terms.
- These terms include a zero frequency term, also called the DC term, related to the average value.
- The higher order terms include the fundamental or lowest frequency term, and harmonics that are multiples of the fundamental.

One-Dimensional DFT

- The one-dimensional (1-D) DFT corresponds to one row (or column) of an image.
- Basis vectors are complex sinusoids, defined by Euler's Identity:

$$e^{j\theta} = \cos(\theta) + j\sin(\theta)$$

- Forward:

$$F(v) = 1/N \sum_{c=0}^{N-1} I(c) e^{-j2\pi vc/N}$$

$$= \frac{1}{N}\sum_{c=0}^{N-1} I(c)[\cos(2\pi vc/N) - j\sin(2\pi vc/N)] = \mathrm{Re}(v) + j\,\mathrm{Im}(v)$$

- Inverse: $F^{-1}[F(v)] = I(c) = \sum_{c=0}^{N-1} F(v) e^{j2\pi vc/N}$
- The $F(v)$ terms can be broken down into a magnitude and phase component:
  - $MAGNITUDE = |F(v)| = \sqrt{[\mathrm{Re}(v)]^2 + [\mathrm{Im}(v)]^2}$, also called the *Fourier spectrum* or *frequency spectrum.*
  - $PHASE = \phi(v) = \mathrm{Tan}^{-1}\left[\frac{\mathrm{Im}(v)}{\mathrm{Re}(v)}\right]$

Two-Dimensional DFT

- Basis images are complex sinusoids:

$$e^{-j2\pi(ur+vc)/N} = \cos\left(\frac{2\pi}{N}(ur+vc)\right) - j\sin\left(\frac{2\pi}{N}(ur+vc)\right)$$

- Forward:

$$F(u,v) = \frac{1}{N}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)e^{-j2\pi(ur+vc)/N}$$

$$= \frac{1}{N}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)\left[\cos\left(\frac{2\pi}{N}(ur+vc)\right) - j\sin\left(\frac{2\pi}{N}(ur+vc)\right)\right]$$

- Inverse: $F^{-1}[F(u,v)] = I(r,c) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ur+vc)/N}$
- $F(u,v) = \mathrm{Re}(u,v) + jIm(u,v)$, $\mathrm{Re}(u,v)$ is the real part and $\mathrm{Im}(u,v)$ is the imaginary part, then we can define the magnitude and phase of a complex spectral component as:
  - $MAGNITUDE = |F(u,v)| = \sqrt{[\mathrm{Re}(u,v)]^2 + [\mathrm{Im}(u,v)]^2}$, also called the *Fourier spectrum* or *frequency spectrum.*
  - $PHASE = \phi(u,v) = \mathrm{Tan}^{-1}\left[\mathrm{Im}(u,v)/\mathrm{Re}(u,v)\right]$
- The 2-D DFT is separable, which means the basis image can be broken down into product terms where each term depends only on the rows or columns:

$$e^{-j2\pi(ur+vc)/N} = e^{-j2\pi ur/N}e^{-j2\pi vc/N}$$

- Separability also implies that the 2-D DFT can be found by successive application of two 1-D DFTs.

Fourier Transform Properties

- Linearity: $F\left[aI_1(r,c) + bI_2(r,c)\right] = aF_1(u,v) + bF_2(u,v)$ where a and b are constants.

  $aI_1(r,c) + bI_2(r,c) = F^{-1}\left[aF_1(u,v) + bF_2(u,v)\right]$

- Convolution: $F[I_1(r,c) * I_2(r,c)] = F_1(u,v)F_2(u,v)$

  $I_1(r,c) * I_2(r,c) = F^{-1}[F_1(u,v)F_2(u,v)]$

  $F[I_1(r,c)I_2(r,c)] = F_1(u,v) * F_2(u,v)$

  $I_1(r,c)I_2(r,c) = F^{-1}[F_1(u,v) * F_2(u,v)]$

- Translation: $F[I(r-r_0, c-c_0)] = F(u,v)e^{-j2\pi(ur_0+vc_0)/N}$

  $I(r-r_0, c-c_0) = F^{-1}\left[F(u,v)e^{-j2\pi(ur_0+vc_0)/N}\right]$

- Modulation: $F\left[I(r,c)e^{j2\pi(u_0r+v_0c)/N}\right] = F(u-u_o, v-v_0)$

  $I(r,c)e^{j2\pi(u_0r+v_0c)/N} = F^{-1}\left[F(u-u_o, v-v_0)\right]$

- Rotation: Let $r = x\cos(\theta), c = x\sin(\theta)$

  $u = w\cos(\phi), v = w\sin(\phi)$

  - $I(x, \theta + \theta_0) = F^{-1}\left[F(w, \phi + \theta_0)\right]$

    $F[I(x, \theta + \theta_0)] = F(w, \phi + \theta_0)$

- Periodicity: $F(u,v) = F(u+N,v) = F(u,v+N) = F(u+N,v+N)\ldots$
- Sampling and Aliasing: To avoid aliasing false frequencies, we must sample a continuous signal with a sampling rate that is at least twice the highest frequency contained in the signal, called the *Nyquist rate* (see Figure 5.2-12).

Displaying the Fourier Spectrum

- The Fourier spectrum consists of complex floating point numbers, stored in CVIPtools as a two band image—one for the real part and one for the imaginary part.
- In CVIPtools we shift the origin to the center of the image by applying the properties of periodicity and modulation with $u_0 = v_0 = N/2$:

$$I(r,c)e^{j2\pi(Nr/2+Nc/2)/N} = I(r,c)e^{j\pi(r+c)} = I(r,c)(-1)^{(r+c)}$$

- To take advantage of the human visual system's response to brightness we can greatly enhance the visual information available by performing a log remap by displaying the following log transform of the spectrum:

$$\mathrm{Log}(u,v) = k\log[1+|F(u,v)|]$$

- The phase can be displayed primarily to illustrate phase changes.

COSINE TRANSFORM

- The cosine transform uses cosine functions as basis functions and can be derived by using a Fourier transform and extending the original $N \times N$ image to an image that is $2N \times 2N$ by folding it about the origin.
- Extending the image to $2N \times 2N$ has the effect of creating an even function, one that is symmetric about the origin, so the imaginary terms in the DFT cancel out.
- The cosine transform requires only real arithmetic.
- The basis images are separable.
- The DCT has been used historically in image compression, such as JPEG.
- Forward: $C(u,v) = \alpha(u)\alpha(v)\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)\cos\left[\frac{(2r+1)u\pi}{2N}\right]\cos\left[\frac{(2c+1)v\pi}{2N}\right]$

where $\alpha(u), \alpha(v) = \begin{cases} \sqrt{\frac{1}{N}} & for\ u,v = 0 \\ \sqrt{2/N} & for\ u,v = 1,2,\ldots,N-1 \end{cases}$

- Inverse: $C^{-1}[C(u,v)] = I(r,c) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos\left[\frac{(2r+1)u\pi}{2N}\right]\cos\left[\frac{(2c+1)v\pi}{2N}\right]$

WALSH–HADAMARD TRANSFORM

- The Walsh–Hadamard transform (WHT) uses rectangular functions for basis functions.
- Instead of frequency terms, we have sequency terms.
- Sequency is the number of zero crossings.
- A 2-D basis image is found from two 1-D basis vectors by performing a vector outer product of the two.
- The WHT is separable.
- Forward: $WH(u,v) = \frac{1}{N}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)(-1)^{\sum_{i=0}^{n-1}[b_i(r)p_i(u)+b_i(c)p_i(v)]}$
- Inverse: $WH^{-1}[WH(u,v)] = I(r,c) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} WH(u,v)(-1)^{\sum_{i=0}^{n-1}[b_i(r)p_i(u)+b_i(c)p_i(v)]}$

HAAR TRANSFORM

- The Haar transform has rectangular waves as basis functions.
- The Haar transform is derived from the Haar matrices.
- The Haar transform retains both spatial and sequency information.
- In the Haar matrices each row is a 1-D basis vector, for example:
- $Haar8 \Rightarrow \frac{1}{\sqrt{8}}\begin{Bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ +2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & +2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & +2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & +2 & -2 \end{Bmatrix}$
- The Haar transform allows for multiresolution decomposition of an input image (see Figure 5.5-1).

PRINCIPAL COMPONENTS TRANSFORM

- The principal components transform (PCT) differs from the previous transforms, as it is not related to extracting frequency or sequency information from images, but is a mathematical transform that decorrelates multiband image data.
- The PCT provides a linear transform matrix that will decorrelate the bands in the input image.
- The linear transform matrix is found by a three step procedure; for example in a three band color, RGB, image: (1) Find the covariance matrix in RGB space, (2) Find the eigenvalues of the covariance matrix, and their corresponding eigenvectors, and (3) use the eigenvectors as a linear transform on the RGB data.
- The PCT data will have the most variance in the principal band.
- In pattern recognition theory variance is a measure of information, in this sense most information is in the principal band.
- Another use of the PCT is to find optimal basis images for a specific image, but this use is not very practical due to the extensive processing required.

FILTERING

- Filtering modifies the frequency or sequency spectrum by selectively retaining, removing, or scaling the various components of the spectrum.
- The shape of the filter depends on the implied symmetry in the transform used.
- Ideal filters have abrupt transitions in the filter function.
- Ideal filters cause artifacts that appear as ripples or waves at edges in the image.
- Nonideal filters have gradual changes in the filter function.
- A commonly used nonideal filter is called a Butterworth filter.
- For a Butterworth filter the order determines the slope of the transition, a higher order is a steeper slope.

Lowpass Filters

- A lowpass filter keeps low frequencies and attenuates high frequencies.
- Lowpass filters will blur the image by removing fast brightness changes that correspond to image detail.
- Butterworth lowpass filter function of order n, and cutoff frequency f_0:

$$H(u,v) = \frac{1}{1 + \left[\dfrac{\sqrt{u^2 + v^2}}{f_0}\right]^{2n}}$$

Highpass Filters

- A highpass filter keeps the high frequencies and attenuates low frequencies.
- Highpass filters will tend to sharpen the image by retaining areas of rapid change in brightness that correspond to edges.
- Butterworth highpass filter function of order n, and cutoff frequency f_0:

- $H(u,v) = \dfrac{1}{1 + \left[\dfrac{f_0}{\sqrt{u^2 + v^2}}\right]^{2n}}$
- A high frequency emphasis filter is a highpass filter that retains some of the low frequency information and boosts the gain of the high frequencies by including an offset value in the filter function (Figure 5.7-7).

Bandpass and Bandreject Filters

- Bandpass filtering will retain specific parts of the spectrum.
- Bandreject filters will remove specific parts of the spectrum.
- Bandpass and bandreject filters require high and low frequency cutoff values.
- Bandreject filters are often used for noise removal.
- A special type of bandreject filter is a notch filter that only removes specific frequencies.

WAVELET TRANSFORM

- The discrete wavelet transform (DWT) is a family of transforms that satisfy specific conditions.
- The *wavelet transform* has basis functions that are shifted and expanded versions of themselves.
- The wavelet transform contains not just frequency information, but also spatial information.
- The wavelet transform breaks an image down into four subsampled, or decimated, images by keeping every other pixel.
- The wavelet results consist of one subsampled image that has been highpass filtered in both the horizontal and vertical directions, one that has been highpass filtered in the vertical and lowpassed in the horizontal, one that has been highpassed in the horizontal and lowpassed in the vertical, and one that has been lowpass filtered in both directions.
- One of the most common models for a wavelet transform uses the Fourier transform and highpass and lowpass filters.
- This model uses the convolution property of the Fourier transform to perform the wavelet transform in the spatial domain.
- This model uses the separable property to perform a 2-D wavelet with two 1-D filters.
- Circular convolution must be used that requires zero-padding.
- The filters discussed include the Haar and Daubechies:

  Haar:

$$\begin{aligned} &\textit{LOWPASS}: && \frac{1}{\sqrt{2}}[1 \quad 1] \\ &\textit{HIGHPASS}: && \frac{1}{\sqrt{2}}[1 \quad -1] \end{aligned} \qquad \text{(inverse same as forward)}$$

Daubechies:

$$LOWPASS: \quad \frac{1}{4\sqrt{2}}[1+\sqrt{3},\ \ 3+\sqrt{3},\ 3-\sqrt{3},\ 1-\sqrt{3}]$$

$$HIGHPASS: \quad \frac{1}{4\sqrt{2}}[1-\sqrt{3},\ \sqrt{3}-3,\ 3+\sqrt{3},\ -1-\sqrt{3}]$$

$$LOWPASS_{inv}: \quad \frac{1}{4\sqrt{2}}[3-\sqrt{3},\ \ 3+\sqrt{3},\ 1+\sqrt{3},\ 1-\sqrt{3}]$$

$$HIGHPASS_{inv}: \quad \frac{1}{4\sqrt{2}}[1-\sqrt{3},\ -1-\sqrt{3},\ 3+\sqrt{3},\ -3+\sqrt{3}]$$

- The algorithm described for the wavelet transform can be performed in six steps:
  1. Convolve the lowpass filter with the rows (remember that this is done by sliding, multiplying coincident terms, and summing the results) and save the results. (Note: For the basis vectors as given, they do *not* need to be reversed for convolution.)
  2. Convolve the lowpass filter with the columns (of the results from Step 1), and subsample this result by taking every other value; this gives us the lowpass–lowpass version of the image.
  3. Convolve the result from Step 1, the lowpass filtered rows, with the highpass filter on the columns. Subsample by taking every other value to produce the lowpass-highpass image.
  4. Convolve the original image with the highpass filter on the rows, and save the result.
  5. Convolve the result from Step 4 with the lowpass filter on the columns; subsample to yield the highpass–lowpass version of the image.
  6. To obtain the highpass–highpass version, convolve the columns of the result from step 4 with the highpass filter.
- The wavelet transform is used in image compression, for example in JPEG2000.

Exercises

Problems

1. When transforming an image into the frequency domain, how does this differ from a color transform?
2. Define basis function, basis vector, basis image, and vector inner product. Explain how these relate to discrete image transforms.
3. (a) What is spatial frequency? (b) What frequency is an area of constant brightness in an image? (c) Are edges in an image primarily high or low frequency?
4. (a) Find the projection, $T(u,v)$, of the image, $I(r,c)$, onto the basis images:

$$\text{Let } I(r,c) = \begin{bmatrix} 7 & 3 \\ 2 & 8 \end{bmatrix}$$

$$\text{And let } B(u,v,r,c) = \begin{cases} \begin{bmatrix} +1 & +1 \\ +1 & +1 \end{bmatrix} \begin{bmatrix} +1 & -1 \\ +1 & -1 \end{bmatrix} \\ \begin{bmatrix} +1 & +1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} +1 & -1 \\ -1 & +1 \end{bmatrix} \end{cases}$$

(b) Using the following inverse basis images, project $T(u,v)$ from above onto them to recover the image.

$$B^{-1}(u,v,r,c) = \begin{cases} \begin{bmatrix} +1 & +1 \\ +1 & +1 \end{bmatrix} \begin{bmatrix} +1 & -1 \\ +1 & -1 \end{bmatrix} \\ \begin{bmatrix} +1 & +1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} +1 & -1 \\ -1 & +1 \end{bmatrix} \end{cases}$$

Did you get the original image back? Why or why not? Are these basis images orthogonal? Are they orthonormal?

5. (a) Find the DFT of the following row of an image: [2 2 2 2]. (b) Do the inverse DFT on the result. Did you get your image row back? Why or why not? (c) Find the DFT of the following row of an image: [2 4 4 2]. (d) Do the inverse DFT on the result. Did you get your image row back? Why or why not?
6. For the 4×4 image shown below, do the following:

$$\begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 4 & 4 & 2 \\ 2 & 4 & 4 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix}$$

(a) Perform the 2-D DFT. Show the results after transforming each row, and after each column. Leave answers in $R + jI$ form.

(b) Perform the inverse DFT on result of (a). Did you get the same data back? Why or why not?

(c) Multiply each element in the original image by $(-1)^{(r+c)}$ and repeat (a). Calculate $F(u,v)$. Is it shifted to the center? Where is the "center" on a 4×4 grid?

7. Use CVIPtools to explore the Fourier spectra of simple objects.

(a) Create an image of a vertical line with *Utilities→ Create*. Perform the FFT on this image (use *Analysis→ Transforms*). In which direction do you see the frequency components?

(b) Create an image of a horizontal line, and perform the FFT. Now what direction do you see the frequency components?

(c) Create a 256×256 image of a rectangle, using the default values. Perform an FFT on this image. In which direction are the frequency components? Use *File→ Show spectrum* to look at just the magnitude image. Next, select the magnitude image with the mouse, and press "e" on the keyboard (this performs a contrast enhancement). Does this help to see the primary frequency components?

(d) Create an image of a vertical cosine wave of frequency 64, perform the FFT. Now do the same with a horizontal cosine. Do the resulting spectra look correct? Explain.

(e) Create circles, ellipses, and checkerboard images. Perform the FFT and view the spectra and phase images. Are the results what you expected? Explain.

8. Use CVIPtools to illustrate the linearity property of the Fourier transform.

   (a) Create a horizontal line and a rectangle image using *Utilities→ Create.* Add these two images together with *Utilities→ Arith/Logic*. Perform the FFT on the resultant image.

   (b) Perform the FFT on original line and original rectangle image. Add the two resulting spectra. Does the result look like the spectrum from (a)? Why or why not? (Hint: log remap.)

   (c) Perform the inverse FFT on the spectra from (a) and (b). Did you get the image back?

9. Use CVIPtools to compare filters in the frequency and spatial domain, which illustrates the convolution property. Use a square image whose size is a power of 2; for example 256×256. Note that an image can be resized with *Utilities→ Size*. Use the default blocksize, which is equal to the image size.

   (a) Apply a lowpass spatial filter mask using the mean filter under *Utilities→ Filters*, to an image. Apply lowpass filtering in the frequency domain with *Analysis→ Transforms*. This is done by first performing the Fourier transform (FFT), followed by the filter on the output Fourier spectrum. Note that the filter automatically performs the inverse transform. Compare the resultant images. Experiment with the mask size of the spatial filter, and the cutoff frequency and type, of the frequency domain lowpass filter. Adjust these parameters until the resultant images look similar. Perform a Fourier transform on the resultant images and compare the spectra.

   (b) Apply a highpass spatial filter mask, using *Utilities→ Filters→ Specify a filter*, to an image. This is done by holding the mouse button on the drop-down arrow to select a filter type, then select the mask size, followed by a click on the *OK* button. Apply highpass filtering in the frequency domain. Follow a process similar to what was done in (a) above, and then compare the resultant images and spectra.

10. Use CVIPtools to create frequency domain filters.

    (a) Open an image of your choice and resize it to 256×256 with *Utilities→ Resize*. Perform the FFT on the 256×256 image.

    (b) Use *Utilities→ Create* to create a circle. Use the default parameter values to create circles—if they have been modified, click the RESET button on the Utilities window, this will reset all the parameters to the default values—if in doubt, killing the window with a click on the **X** in the upper right corner will do a hard reset

on the window. Create a circle in the center of the image with a radius of 32. Next, check the *Blur radius* checkbox, set blur radius value to 64, click *Apply.*

(c) Use *Utilities→ Arith/Logic* to multiply the FFT spectrum with the circle images. Select the FFT spectrum as the *current image* (click on the image itself or the name in the image queue), and select the circle as the *second image.* Perform the multiplication with the spectrum and both circles. Note that these multiplied images are both filtered spectra—use the "e" option on the keyboard to enhance these images.

(d) Perform the inverse FFT transform on the two multiplied images. Look at the output images and compare. How do they differ? What type of filters are these? Why do the two output images differ?

(e) Repeat steps (c) and (d) but first perform a logical NOT on the two circle images, using *Utilities→ Arith/Logic*.

11. Use CVIPtools to illustrate the translation property of the Fourier transform.

(a) Translate an image in the spatial domain, using *Analysis→ Geometry→ Translate an image* with the default Wrap-around option. Now perform an FFT on the original image and the translated image. Compare the spectra of these images. Are they the same? In addition to the log remapped magnitude (the default spectral display), use the *File→ Show spectrum* in the main window to compare the phase images. Are the phase images the same?

(b) Change the translation values, the amount you move the image right and down, and repeat part (a).

12. Use CVIPtools to illustrate the modulation property of the Fourier transform.

(a) Use *Utilities→ Create* to create an image of a circle and an image of a horizontal cosine wave of frequency 32. Multiply these two images together.

(b) Find the FFT spectra of the circle and of the two images multiplied together. Examine the modulation property of the Fourier transform. Do these images look correct? Why or why not?

(c) Use *File→ Show spectrum* to look at the magnitude images of the spectra. Do these images look correct? Why or why not?

13. Use CVIPtools to illustrate the rotation property of the Fourier transform.

(a) Create a 256×256 image of a rectangle, using the default values. Perform an FFT on this image.

(b) Rotate the rectangle image, using *Analysis→ Geometry*, by 45°. Next, crop a 256×256 image from the center of this rotated image with *Utilities→ Size* [e.g., use (*r*,*c*) = (64,64) for the upper left corner and a size of 256×256], and perform the FFT.

(c) Compare the resulting spectra. Did the rotation cause any artifacts that may be affecting the spectrum? Look at the phase image by using *File→ Show Spectrum* for both the original and rotated rectangle spectra, what do you see? Does this seem reasonable?

14. (a) Is the cosine an even or an odd function? (b) Do you think that the Fourier or the cosine transform is faster to compute? Explain.

15. For an $N \times N$ image, we assume a Fourier symmetry that repeats the $N \times N$ pattern. For an $N \times N$ image, what size is the pattern that repeats for the cosine transform?

16. (a) What is the general form of the Walsh–Hadamard basis functions? (b) Do you think that the Walsh–Hadamard or the cosine transform is faster to compute? Explain.
17. Let the rows of the following matrix be basis vectors:

$$\begin{Bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & -1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \end{Bmatrix}$$

    (a) What is the sequency of each row? Going from top to bottom are they in sequency order?
    (b) Are they orthogonal? Why or why not?
    (c) Are the orthonormal? Why or why not? If not, how can we make them orthonormal?
    (d) To which transform do these basis vectors belong?
    (e) Find the vector outer product of the first and last row.
18. Of the transforms described in this chapter, which ones are separable? Explain what this means.
19. Name the two transforms discussed that provide a multiresolution decomposition of the input image. Explain what this means.
20. Sketch the (a) Fourier, (b) cosine, and (c) Walsh–Hadamard transform spectral image of the following image:

$$\begin{bmatrix} 255 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 \end{bmatrix}$$

21. (a) Sketch the Fourier spectrum for a horizontal sine wave of frequency 32 on a 256×256 image, (b) the spectrum for a vertical sine wave of frequency 32 on a 256×256 image, and (c) apply the convolution theorem to find the image that results from the convolution of the horizontal and vertical sine wave.
22. (a) Where does the PCT get its name? Explain. (b) What are the three steps to performing the PCT?
23. Use CVIPtools to explore the PCT transform with color images.
    (a) Open a color image of your choice. Use *Utilities→ Convert→ Color space* to apply the PCT to the image. Next, do the inverse PCT on the output image, by unchecking the *Forward* checkbox. Is the PCT an invertible transform? In other words, did you get your image back OK?
    (b) Use *Utilities→ Create→ Extract band* to get all three bands from the original image as well as the PCT image. Place the RGB bands above and the three PCT bands below and compare. Are the RGB bands correlated; that is, do they look

similar? Are the PCT bands correlated; that is, do they look similar? Examine the data range on each image, which band has the largest data range? (Note: the data range is seen at the bottom of the main window next to the data type.) Why do you think this band has the largest data range?

24. (a) What is the difference between an ideal filter and a nonideal filter? (b) Are there any advantages to one over the other? What are they?
25. (a) What is the *order* of a Butterworth filter? (b) How does increasing the order of the filter affect the image?
26. Name an application of a bandreject, or a notch, filter. Explain.
27. (a) What is the difference between a highpass filter and a high frequency emphasis filter? (b) How does this difference affect the resultant image?
28. Use CVIPtools to explore a bandreject filter.
    (a) Create a 256×256 image of a horizontal sinusoidal wave using *Utilities→ Create* in CVIPtools. Select *Analysis→ Transforms* and perform the FFT on the image with a blocksize of 256 (the entire image). View the magnitude of the FFT output using *File→ Show spectrum→ Magnitude* on the main window. Notice the frequency component corresponding to the horizontal frequency of the sinusoid. What is the approximate location of this frequency component? Notice also the location of the DC component (center). As is the case in most images, the DC component is the largest component in the transformed image.
    (b) Now filter the transformed image with a bandreject filter. Check the *Keep DC* checkbox, so the average value is retained. Choose an ideal bandreject filter. Choose cutoff frequencies to remove the horizontal sinusoidal frequency. Perform the inverse FFT on the image. Compare the resulting image with the original image. Did you succeed in removing most of the sinusoid? (Perfect results are not attainable for this experiment, because of quantization noise.)
29. Use CVIPtools to see how the DC terms affects filtering. Use the *Analysis→ Transforms* window.
    (a) Load a 256×256 image of your choice and perform an FFT using a block size of 64. Filter the spectrum with an ideal lowpass filter with a cutoff of 16, and keep the DC by checking the *Keep DC* checkbox. Next, use the same filter, but without keeping the DC term. Compare the data range of the two resulting images. How does the DC term affect the data range? Compare how the images look—do they look different? Why or why not?
    (b) Perform a DCT using a block size of 64. Filter the spectrum with an ideal lowpass filter with a cutoff of 32, and keep the DC by checking the *Keep DC* checkbox. Next, use the same filter, but without keeping the DC term. Compare the data range of the two resulting images. How does the DC term affect the data range? Compare how the images look—do they look different? Why or why not? Compare the DCT and FFT results, what do you see? Explain the results.
    (c) Perform a WHT using a block size of 64. Filter the spectrum with an ideal lowpass filter with a cutoff of 32, and keep the DC by checking the *Keep DC* checkbox. Next, use the same filter, but without keeping the DC term. Compare the data range of the two resulting images. How does the DC term affect the data range? Compare how the images look—do they look different? Why or why not? Compare these results with the DCT and FFT results, what do you see? Explain the results.

30. Use CVIPtools to illustrate the effects of low-pass filtering on a real image and compare the effects of ideal filters to those of Butterworth filters.
    (a) Load a 256×256 complex grayscale image and perform the FFT with the *Analysis→ Transforms* window, using a blocksize of 16. Filter the transformed image using an ideal lowpass filter with a cutoff of 4 and keep the DC value. Notice the absence of highfrequency information and the ringing effect, which appears as waves emanating from edges in the image, caused by the sharp frequency cutoff.
    (b) Now apply a Butterworth lowpass filter of order 1 to the Fourier-transformed image. Use the same cutoff as in part 2(a). Keep the DC during filtering. Compare the result with that of part (a). Is the ringing effect as noticeable now? Why/why not?
    (c) Repeat part (b) using a 6th order Butterworth filter instead of a 1st order filter. Compare the result to the result from parts (a) and (b). Because the frequency response of a 6th-order Butterworth filter is close to that of an ideal filter, the image should be similar to that of part (a). (d) Repeat (a), (b), and (c) using a blocksize of 256×256 and cutoff of 64.
31. Use CVIPtools to see the relationship between transform-domain filtering in the DCT, FFT, and WHT domain. Use the *Analysis→ Transforms* window. Note that the origin for the FFT, corresponding to the zero-frequency term (DC component), is shifted to the center, while for all other transforms the origin is in the upper left corner. Since all the transforms implemented are fast transforms based on powers of 2, the blocksize must be a power of 2. If the selected blocksize will not evenly cover the image, then the image is zero-padded as required. For DC components located in the upper-left-hand corner, CVIPtools lets you specify cutoff frequencies ranging from 1 to *blocksize*. For the FFT, with the DC term in the center, the range is from 1 to *blocksize/2*.
    (a) Load any image of your choice. If the size is not 256×256, use *Utilities→ Size* to resize it to 256×256.
    (b) Choose the ideal lowpass filter type, any blocksize, and a cutoff frequency (CF) divisible by 2. Apply this filter to the image in the Walsh domain. Repeat this procedure using the DCT and the FFT, but use a cutoff frequency equal to CF/2 for the FFT.
    (c) Compare the images resulting from filtering with different transforms.
    (d) Which transform resulted in the best quality of the filtered image? Which transform resulted in the poorest quality filtered image? Compare your answers with what you know about the properties of the DCT, FFT, and Walsh Transform. Do your answers agree with what you would expect?
32. Use CVIPtools to compare highpass and high frequency emphasis filters.
    (a) Perform a DCT on a 256×256 image using 256 as the block size.
    (b) Apply a Butterworth highpass filter with a cutoff of 64.
    (c) Apply a high frequency emphasis filter with a cutoff of 64, use the same order used in (b), and compare results to (b). What do you see? Is this what you expected? Explain.
    (d) Add the original image to the result from (b), and compare the added image to the one from (c). Are they similar? Why or why not?

33. Use CVIPtools to see how the DC term affects highpass filters, review data remapping. Perform a DCT on a 256×256 image, and apply a highpass filter with a cutoff of 64. Do it with and without keeping the DC term. Do the two filtered images look similar? Explain. Compare the data range on the two images by clicking on each image (the image information appears at the bottom of the main window).
34. a) Does the wavelet transform have a unique set of basis functions? (b) Name two functions commonly used for wavelet filters. (c) Of these two, with which one is the wavelet transform faster to calculate? (d) Does the wavelet transform provide spatial or frequency information?
35. (a) Describe the process to implement a wavelet transform using 1-D filters. (b) What is the criterion that the filters must meet for a wavelet transform, and why? (c) How is circular convolution performed? (d) What does the term *decomposition level* mean as related to the wavelet transform?
36. Use CVIPtools to explore the wavelet transform.
    (a) Open an image of your choice and resize it to 256×256. Perform the wavelet transform using *Analysis→ Transforms*. Use decomposition levels of 1, 2, 3, and 4. Compare the wavelet images. What exactly is the *decomposition level*?
    (b) Perform the wavelet transform twice with a decomposition level of 9 on your 256×256 image, once with the Haar basis and once with the Daubechies. Note that the Haar uses filters with two coefficients, and the Daubechies uses four. Perform an ideal lowpass filter with a cutoff of 32 on the wavelet images and compare the filtered images. Are they different? How? Why?
    (c) Repeat (b), but use a first order Butterworth filter.

Programming Exercises

Filtering

1. Write a function to create frequency domain ideal filter masks that will work with the Fourier transform in CVIPtools. Let the user specify the size, the filter type: highpass, lowpass, or bandreject, and the cutoff frequencies. Be sure the output images are floating point.
2. Test these filter mask images using CVIPtools. Use the FFT, and multiply to test the filters.
3. Repeat (1) and (2) for DCT symmetry, and test with the DCT.
4. Repeat 1–3, but create Butterworth filter masks. Let the user specify the filter order.

Fourier Transform

1. Use the CVIPtools libraries functions to put the *fft_transform* and *ifft_transform* functions into your CVIPlab program.
2. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

Discrete Cosine Transform

1. Use the CVIPtools libraries functions to put the *dct_transform* and *idct_transform* functions into your CVIPlab program.
2. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

Walsh–Hadamard Transform

1. Use the CVIPtools libraries functions to put the *walhad_transform* function into your CVIPlab program.
2. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

Haar Transform

1. Use the CVIPtools libraries functions to put the *haar_transform* function into your CVIPlab program.
2. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

Wavelet Transform

1. Use the CVIPtools libraries functions to put the *wavhaar_transform* and *wavdaub4_transform* functions into your CVIPlab program.
2. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

CVIPtools Library Filter Functions

1. Use the CVIPtools functions in the *TransformFilter* library to perform filtering in your CVIPlab program.
2. Compare your results to those obtained with CVIPtools, and with the filter functions you wrote.

Supplementary Exercises

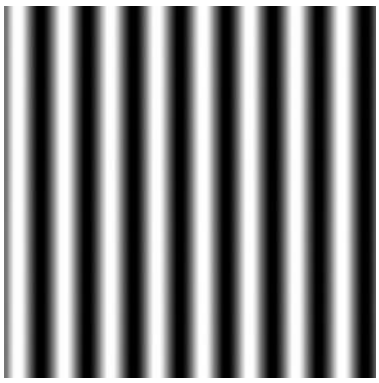
Supplementary Problems

1. The Fourier transform of the "triangle" function below is a squared sinc function. Show that we can verify this by using the Fourier transform of a rectangle function.

"Triangle" 

2. Describe or sketch the image that results from the convolution of the following two images:

   Image #1: a horizontal sine wave of frequency 8; Image #2: a vertical sine wave of frequency 16.



3. The following two images resulted from using the same transform-domain filter; but one used the Fourier and one used the Walsh transform. (a) What image used what transform? (b) Was the filter a highpass or lowpass? (c) Was it an ideal or nonideal filter?

Image #1

Image #2

4. Derive the 1-D discrete cosine transform using a 1-D discrete Fourier transform of length 2*N*.
5. The eigenvalues of a multispectral six-band image are [3.2, 0.976, 122, 0.45, 1.15, 110]. What will the RMS error be if we apply a PCT for a 3:1 data compression?
6. Use CVIPtools *Utilities→ Create→ Assemble Bands* and the images in Figure 5.6-1 to create the original RGB image. (a) Next use *Utilities→ Stats→ Image Statistics* to find the statistics of the original image for each band. (b) Perform the PCT with *Utilities→ Convert→ Color Space* with a PCT transform, and find the image statistics. (c) Compare the results from (a) and (b) and discuss the results. Do they make sense? Why or why not? (Note: Did you find the image statistics on the original data or on data remapped to byte?)
7. You are given a TV camera that is suspected of having an interlace problem where one of the fields is slightly brighter than the other. Assume you have a frame grabber to digitize single frames that does not introduce any noise or other problems into the resulting digital image. Devise a method using frequency or sequency

transforms to determine: (a) If this is actually the problem. (b) If so, how much brighter is the one field compared to the other?

8. A bicycle manufacturer requires precision parts for its gear assembly that it receives from an outside vendor. Occasionally, in transit the parts get bent or damaged, or manufacturing defects occur. Before the gear assembly is put together they want to be certain the parts are without defects—some defects may not be easily visible and they have an automated production line. Devise a computer imaging algorithm using the Fourier transform to test these parts.

Supplementary Programming Exercises

Fourier Transform

1. Write a function to implement the Fourier transform and its inverse. Allow the user to specify the block size.
2. Use the references to research the fast Fourier transform (FFT). Modify the function to implement an FFT.
3. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

Cosine Transform

1. Write a function to implement the cosine transform and its inverse. Allow the user to specify the block size.
2. Use the references to research the fast discrete cosine transform (FDCT). Modify the function to implement an FDCT.
3. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

Walsh–Hadamard Transform

1. Write a function to implement the WHT and its inverse. Allow the user to specify the block size.
2. Use the references to research the fast discrete Walsh–Hadamard transform (FDWHT). Modify the function to implement a FDWHT.
3. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

Haar Transform

1. Use the references to research the Haar transform and its inverse. Write a function to implement the Haar transform.
2. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

Principal Components Transform

1. Write a function to implement a PCT, and its inverse, on a multiband image. Note that the CVIPtools library, *libmatrix*, can be used. These functions are of particular interest: *covariance_Matrix, eigenSystem_Matrix*

2. Compare your results to those obtained with CVIPtools (see *Utilities→ Convert→ Color Space*). Are they the same? Why or why not? If they are different, what can be done to make them the same?

Wavelet Transform

1. Using the two filter types discussed, write a function to implement a wavelet transform and its inverse. Let the user specify the number of decompositions.
2. Compare your results to those obtained with CVIPtools. Are they the same? Why or why not? If they are different, what can be done to make them the same?

References

Banks, S., *Signal Processing, Image Processing and Pattern Recognition,* Upper Saddle River, NJ: Prentice Hall, 1990.

Bracewell, R. N., *Two-Dimensional Imaging*, Upper Saddle River, NJ: Prentice Hall, 1995.

Castleman, K. R., *Digital Image Processing*, Upper Saddle River, NJ: Prentice Hall, 1996.

Gonzalez, R. C., and Woods, R. E., *Digital Image Processing*, Upper Saddle River, NJ: Prentice Hall, 2008.

Gonzalez, R. C., and Woods, R. E., *Digital Image Processing*, Reading, MA: Addison Wesley, 1992.

Jain, A. K., *Fundamentals of Digital Image Processing*, Upper Saddle River, NJ: Prentice Hall, 1989.

Kjoelen, A., *Wavelet Based Compression of Skin Tumor Images*, Master's Thesis in Electrical Engineering, Southern Illinois University at Edwardsville, 1995.

Lim, J. S., *Two-Dimensional Signal and Image Processing*, Upper Saddle River, NJ: PTR Prentice Hall, 1990.

Masters, T., *Signal and Image Processing with Neural Networks*, New York, NY: Wiley, 1994.

Oppenheim, A. V, and Schafer, R. W., *Discrete Time Signal Processing,* 3rd ed., Upper Saddle River, NJ: Prentice Hall, 2009.

Petrou, M., and Bosdogianni, P., *Image Processing: The Fundamentals*, New York, NY: Wiley, 1999.

Pratt, W. K., *Digital Image Processing*, New York, NY: Wiley, 1991.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Numerical Recipes in C*, New York, NY: Cambridge University Press, 1992.

Rosenfeld, A., and Kak, A. C., *Digital Picture Processing*, San Diego, CA: Academic Press, 1982.

Sonka, M., Hlavac, V., and Boyle, R., *Image Processing, Analysis and Machine Vision*, Toronto, Canada: Thomson, 2008.

Taubman, D. S., and Marcellin, M. W., *JPEG2000: Image Compression Fundamentals, Standards and Practice*, Norwell, MA: Kluwer Academic Publishers, 2002.

Further Reading

For discrete transforms, many excellent texts are available, including Gonzalez and Woods (2008), Sonka, Hlavac, and Boyle (2008), Castleman (1996), Bracewell (1995), Pratt (1991), Jain (1989), and Rosenfeld and Kak (1982). For details regarding the fast implementation of the transforms see Gonzalez and Woods (2008), Petrou and Bosdogianni (1999), and Press et al. (1992). See Gonzalez and Woods (1992) and Pratt (1991) for details on the separate Walsh

and Hadamard transforms. More details on the Haar transform can be found in Gonzalez and Woods (2008), Castleman (1996), Pratt (1991), and Jain (1989). Additional detail on the PCT can be found in Gonzalez and Woods (2008), and (applied to feature analysis) Sonka, Hlavac, and Boyle (2008). For a more in depth treatment of sampling and aliasing see Oppenheim and Schafer (2009).

For more information on filters see Gonzalez and Woods (2008), Pratt (1991), Lim (1990), and Banks (1990). For an excellent and detailed discussion on the need for zero-padding with convolution filters see Gonzalez and Woods (2008). For mathematical details on implementing filters in the spatial domain from the frequency domain specifications see Petrou and Bosdogianni (1999) and Gonzalez and Woods (1992). The wavelet transform as implemented in CVIPtools is described in Kjoelen (1995). More information on wavelet transforms is found in Gonzalez and Woods (2008), Masters (1994), and Castleman (1996). Implementation details for the wavelet as applied in JPEG2000 are found in Taubman and Marcellin (2002).

6

Feature Analysis and Pattern Classification

6.1 Introduction and Overview

Feature analysis and pattern classification are often the final steps in the image analysis process. *Feature analysis* involves examining the features extracted from the images and determining if and how they can be used to solve the imaging problem under consideration. In some cases the extracted features may not solve the problem and the information gained by analyzing the features can be used to determine further analysis methods that may prove helpful, including additional features that may be needed. *Pattern classification*, often called pattern recognition, involves the classification of objects into categories. For many imaging applications this classification needs to be done automatically, via computer. The patterns to be classified consist of the extracted feature information, which are associated with image objects and the classes or categories will be application dependent.

As discussed in Chapter 3, the goal in image analysis is to extract information useful for solving application-based problems. This is done by intelligently reducing the amount of image data with the tools we have explored, including image segmentation (Chapter 4) and transforms (Chapter 5). Once we have performed these operations, we have modified the image from the lowest level of pixel data into higher-level representations. Now, we can consider extraction of features that can be useful for solving computer imaging problems. Image segmentation allows us to look at object features, and the image transforms provide us with features based on spatial frequency information—spectral features. The object features of interest include the geometric properties of binary objects, histogram features, spectral features, texture features, and color features. Once we have extracted the features of interest, we can analyze the image.

Exactly what we do with the features will be application-dependent. If we are working on a computer vision problem, the end goal may be the generation of a classification rule in order to identify objects. If we are working to develop a new image compression algorithm, we may want to determine what image data is important; the insignificant information can be compressed or eliminated completely. For image restoration we may want to determine the type of noise that exists in the image, or how the image has been degraded. Image analysis may help us to solve an image enhancement problem by allowing us to determine exactly what it is that makes images visually pleasing.

As was shown in Figure 3.1-3, feature extraction is part of the data reduction process and is followed by feature analysis. One of the important aspects of feature analysis is to determine exactly what features are important, so the analysis is not complete until we incorporate application-specific feedback into the system (see Figure 6.1-1). In this chapter we will discuss feature extraction and analysis, as well as provide an introduction to pattern