Spatial Domain Filtering

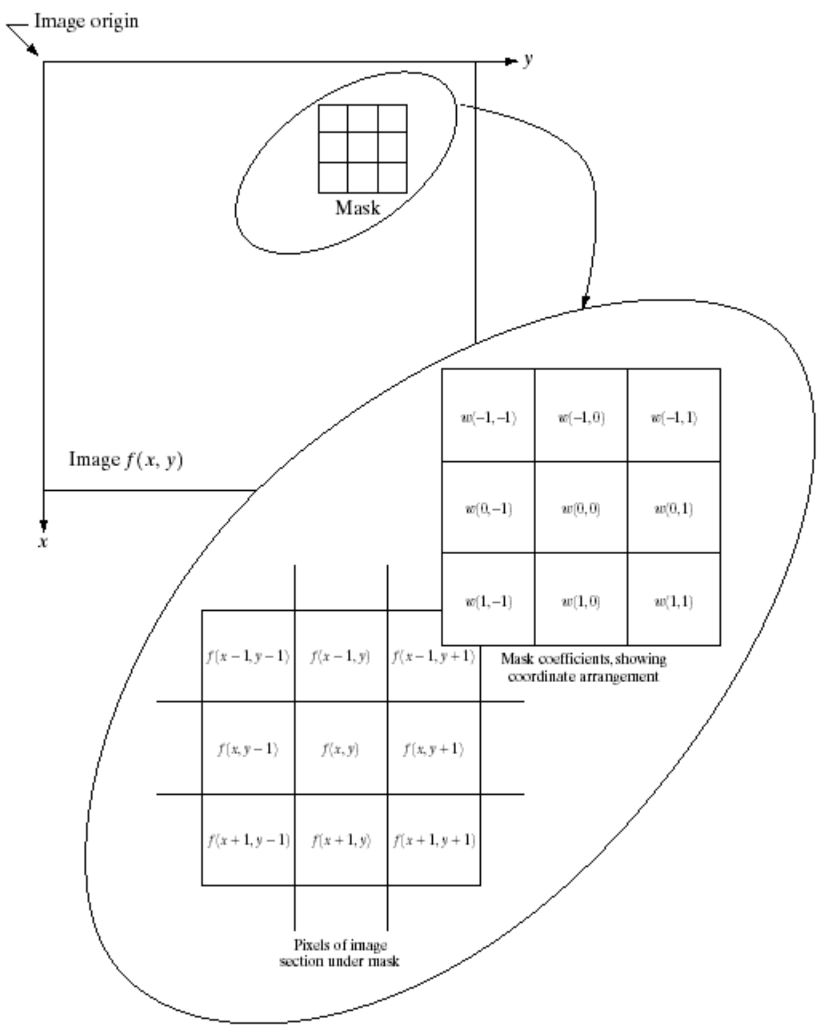


FIGURE 3.32 The mechanics of spatial filtering. The magnified drawing shows a 3×3 mask and the image section directly under it; the image section is shown displaced out from under the mask for ease of readability.

Spatial Domain Filtering

$\frac{1}{9} \times$

1	1	1
1	1	1
1	1	1

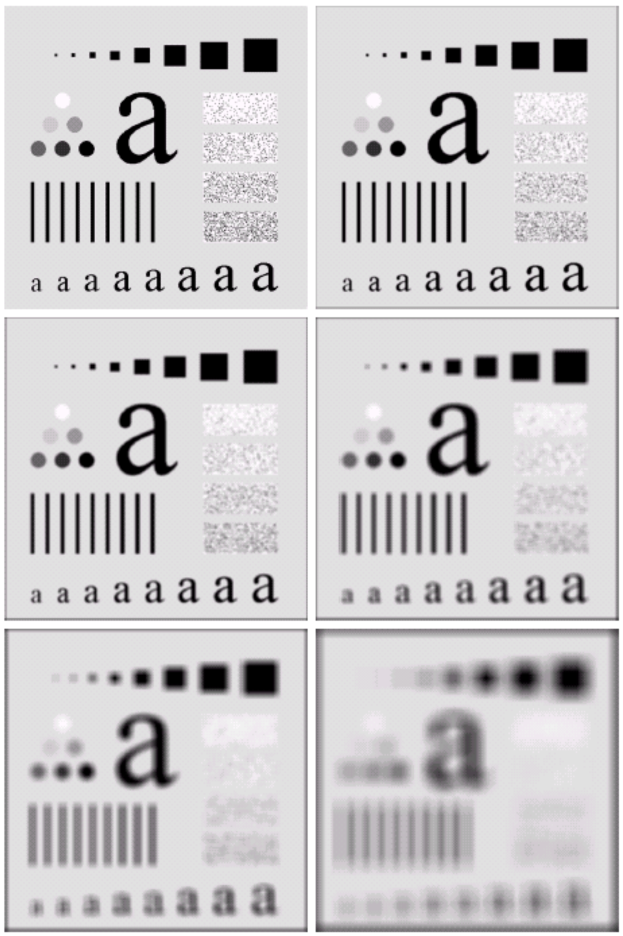
$\frac{1}{16} \times$

1	2	1
2	4	2
1	2	1

a b

FIGURE 3.34 Two 3×3 smoothing (averaging) filter masks. The constant multipli er in front of each mask is equal to the sum of the values of its coefficients, as is required to compute an average.

Spatial Domain Filtering



a b
c d
e f

FIGURE 3.35 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $n = 3, 5, 9, 15$, and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their gray levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

Fourier Transform

3.1 INTRODUCTION TO THE FOURIER TRANSFORM

Let $f(x)$ be a continuous function of a real variable x. The *Fourier transform* of $f(x)$, denoted by $\mathfrak{F}\{f(x)\}$, is defined by the equation

$$\mathfrak{F}\{f(x)\} = F(u) = \int_{-\infty}^{\infty} f(x) \exp[-j2\pi ux]\, dx \tag{3.1-1}$$

Example: Consider the simple function shown in Fig. 3.1(a). Its Fo transform is obtained from Eq. (3.1-1) as follows:

$$F(u) = \int_{-\infty}^{\infty} f(x) \exp[-j2\pi ux]\, dx$$

$$= \int_{0}^{X} A \exp[-j2\pi ux]\, dx$$

$$= \frac{-A}{j2\pi u}\left[e^{-j2\pi ux}\right]_0^X = \frac{-A}{j2\pi u}\left[e^{-j2\pi uX} - 1\right]$$

$$= \frac{A}{j2\pi u}\left[e^{j\pi uX} - e^{-j\pi uX}\right]e^{-j\pi uX}$$

$$= \frac{A}{\pi u} \sin(\pi uX)\, e^{-j\pi uX}$$

which is a complex function. The Fourier spectrum is given by

$$|F(u)| = \frac{A}{\pi u} |\sin(\pi uX)|\, |e^{-j\pi uX}|$$

$$= AX \left| \frac{\sin(\pi uX)}{(\pi uX)} \right|$$

A plot of $|F(u)|$ is shown in Fig. 3.1(b).

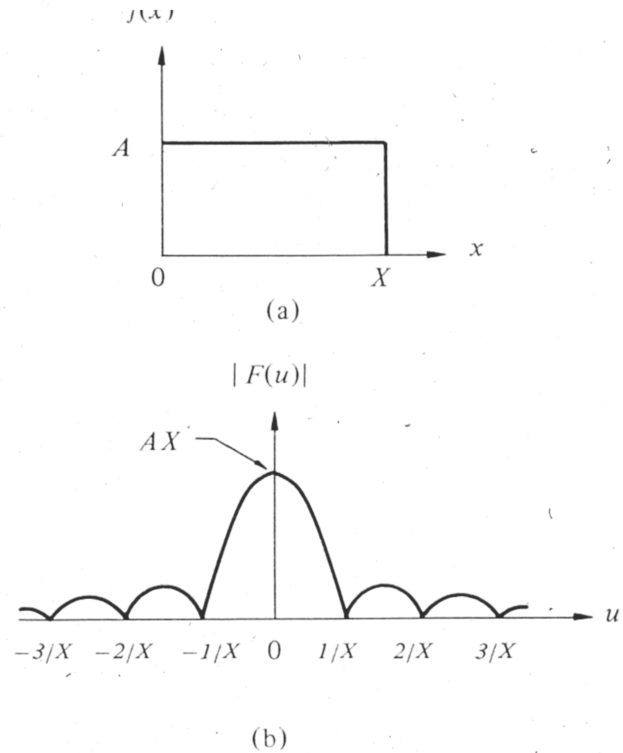


Figure 3.1. A simple function and its Fourier spectrum.

2-D Fourier Transform

Example: The Fourier transform of the function shown in Fig. 3.2(
given by

$$F(u, v) = \int\int_{-\infty}^{\infty} f(x, y) \exp[-j2\pi(ux + vy)]\, dx\, dy$$

$$= A \int_0^X \exp[-j2\pi ux]\, dx \int_0^Y \exp[-j2\pi vy]\, dy$$

$$= A\left[\frac{e^{-j2\pi ux}}{-j2\pi u}\right]_0^X \left[\frac{e^{-j2\pi vy}}{-j2\pi v}\right]_0^Y$$

$$= \frac{A}{-j2\pi u}\left[e^{-j2\pi uX} - 1\right] \frac{1}{-j2\pi v}\left[e^{-j2\pi vY} - 1\right]$$

$$= AXY\left[\frac{\sin(\pi uX)\, e^{-j\pi uX}}{(\pi uX)}\right]\left[\frac{\sin(\pi vY)\, e^{-j\pi vY}}{(\pi vY)}\right]$$

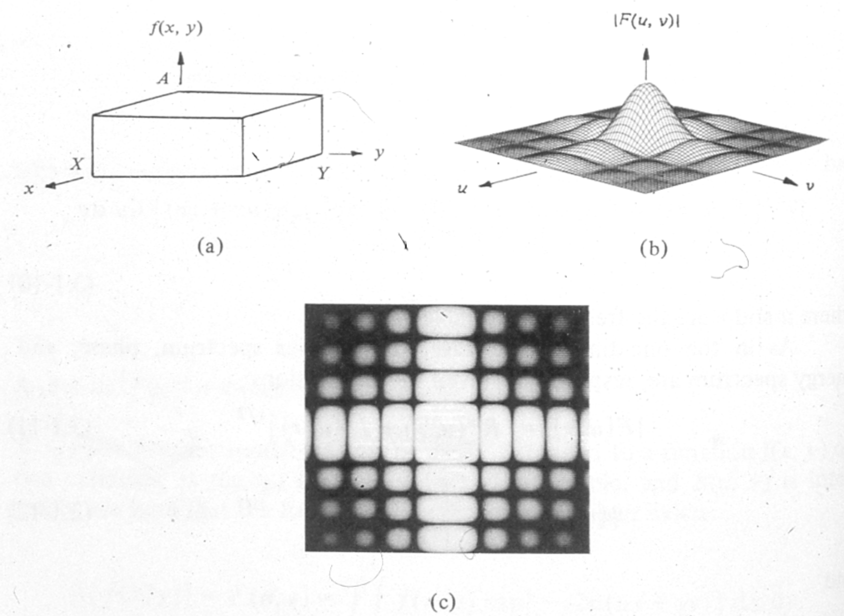
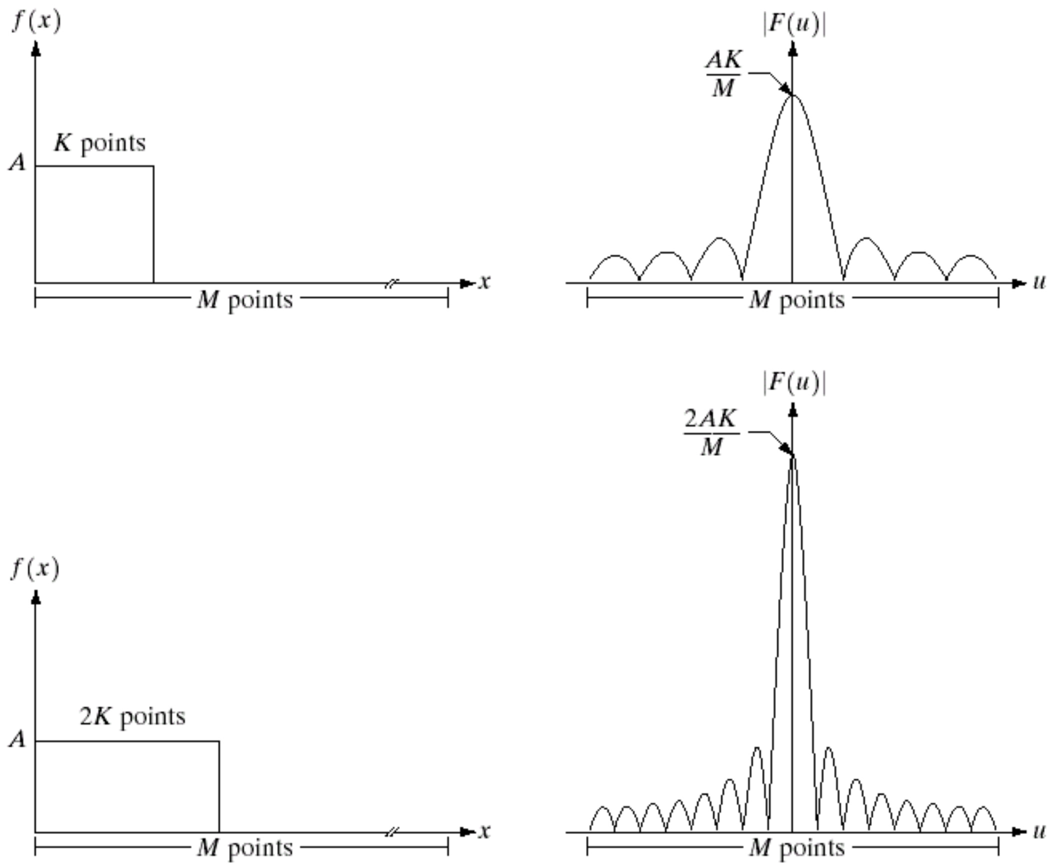


Figure 3.2. (a) A two-dimensional function, (b) its Fourier spectrum, and (c) the spectrum displayed as an intensity function.

Spatial Frequency



a	b
c	d

FIGURE 4.2 (a) A discrete function of *M* points, and (b) its Fourier spectrum. (c) A discrete function with twice the number of nonzero points, and (d) its Fourier spectrum.

$$\Delta u = \frac{1}{M\Delta x}$$

Fourier Transform Shift

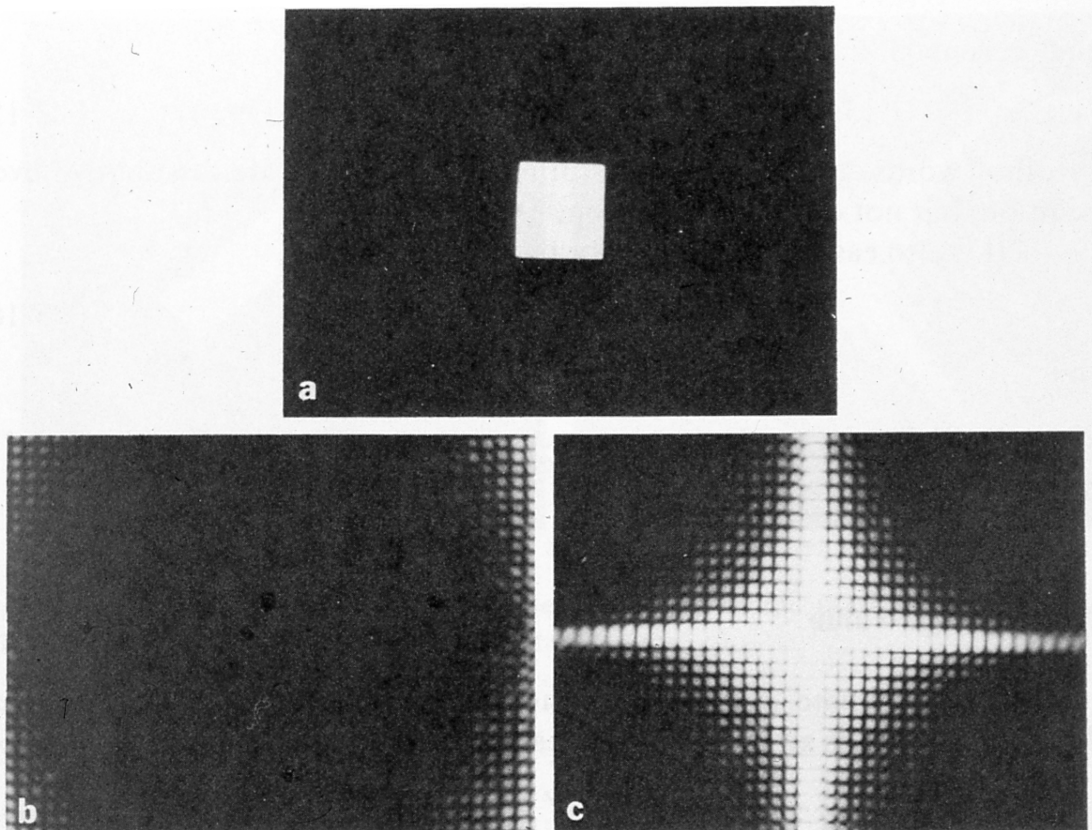
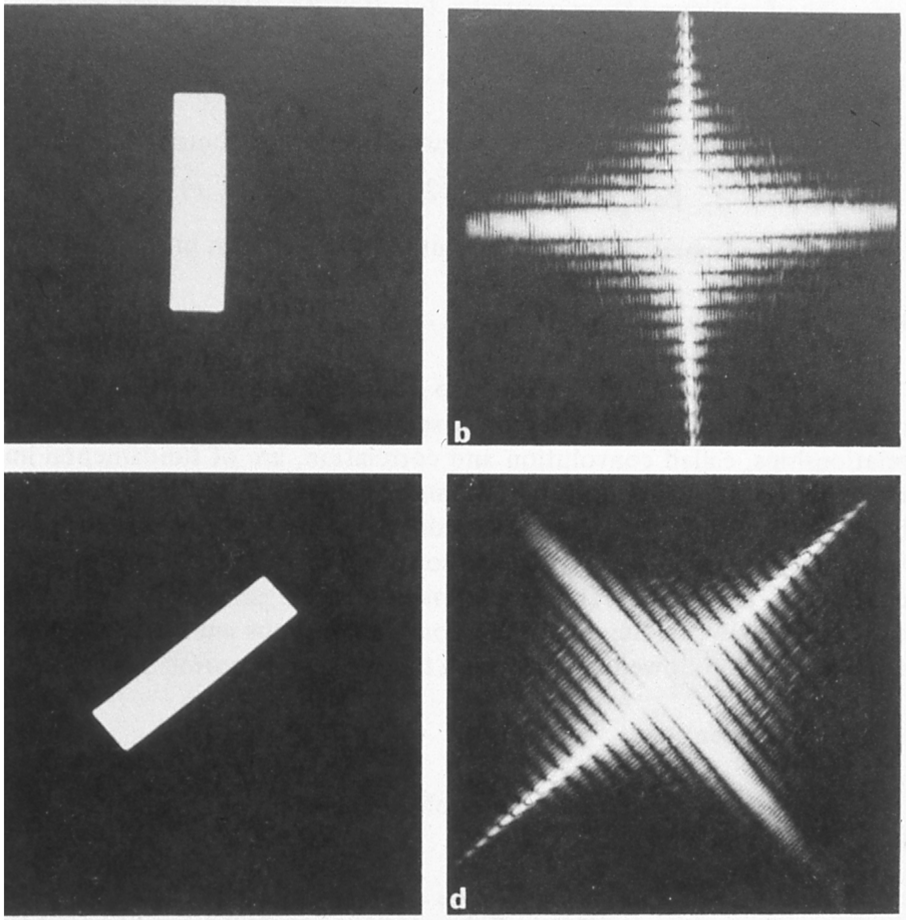


Figure 3.11. (a) A simple image. (b) Fourier spectrum without shifting. (c) Fourier spectrum shifted to the center of the frequency square.

Fourier Transform Rotation



re 3.12. Rotational properties of the Fourier transform. (a) A simple image. (b) :trum. (c) Rotated image. (d) Resulting spectrum.

Sample 2-D Fourier Transforms

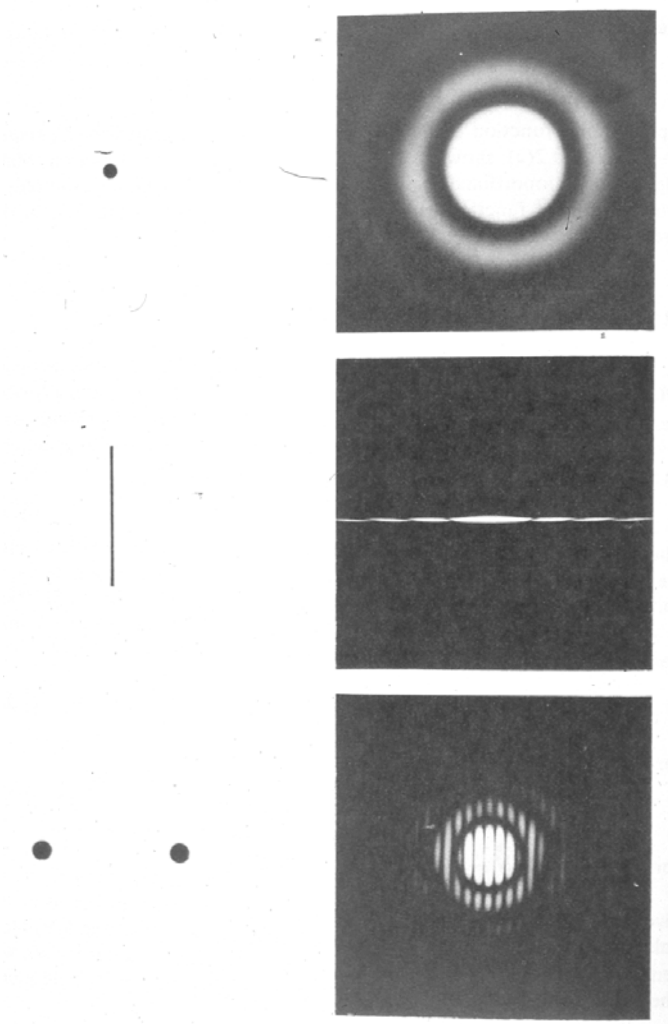


Figure 3.3. Some two-dimensional functions and their Fourier spectra.

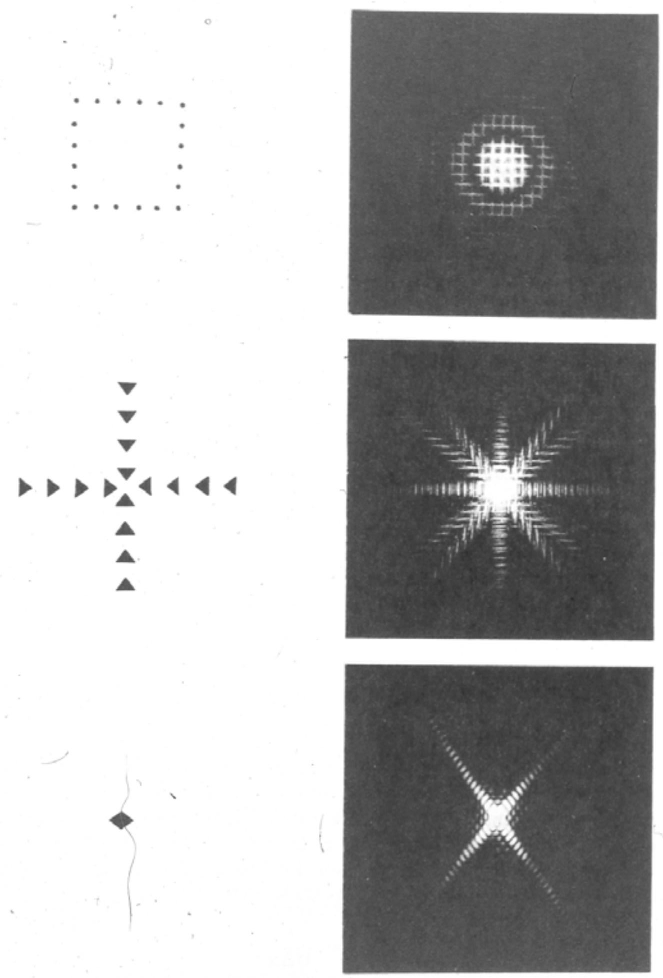
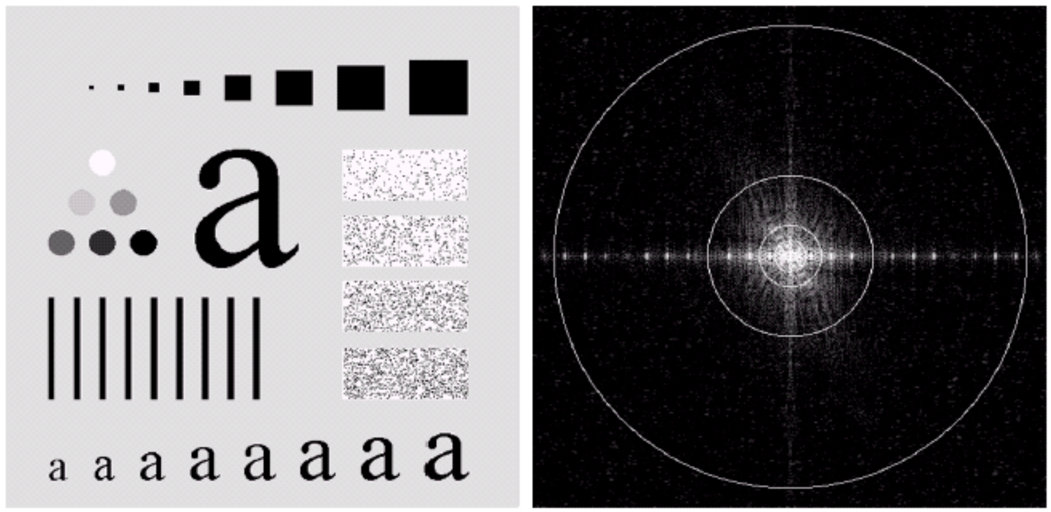


Figure 3.3. (Continued.)

Power Spectra



a b

FIGURE 4.11 (a) An image of size 500 × 500 pixels and (b) its Fourier spectrum. The superimposed circles have radii values of 5, 15, 30, 80, and 230, which enclose 92.0, 94.6, 96.4, 98.0, and 99.5% of the image power, respectively.

Image Enhancement in the Frequency Domain

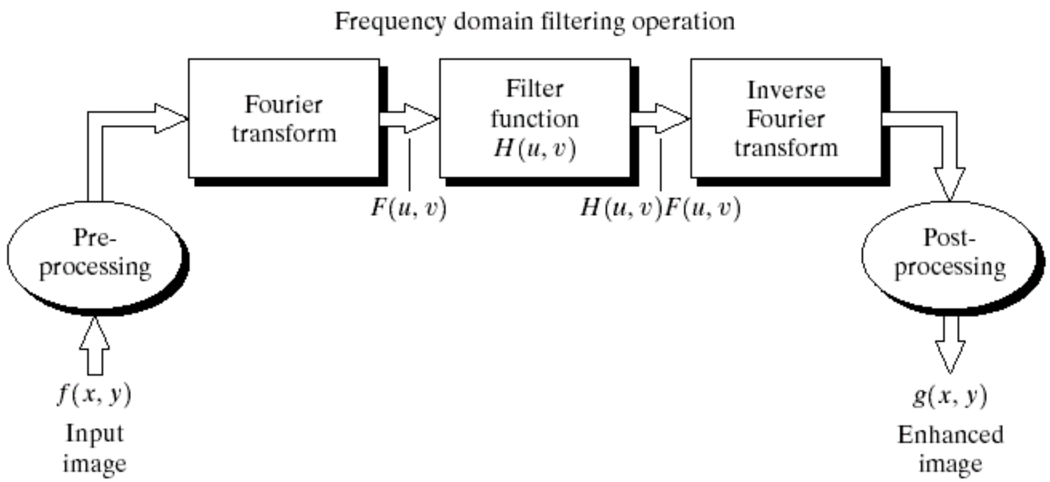
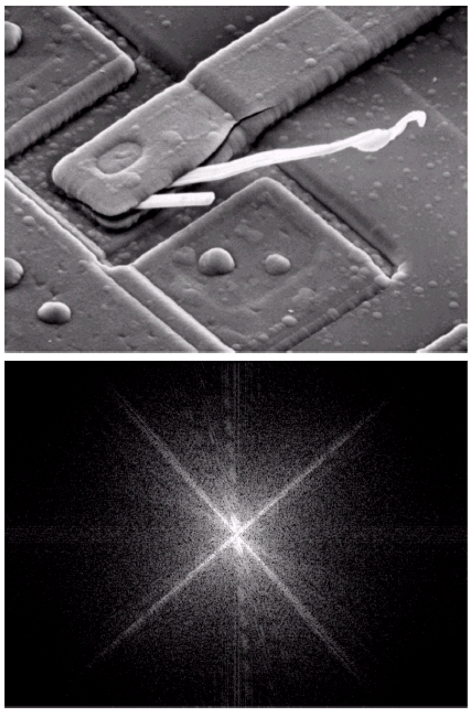


FIGURE 4.5 Basic steps for filtering in the frequency domain.

2-D Fourier Transform



a
b

FIGURE 4.4
(a) SEM image of a damaged integrated circuit. (b) Fourier spectrum of (a). (Original image courtesy of Dr. J. M. Hudak, Brockhouse Institute for Materials Research, McMaster University, Hamilton, Ontario, Canada.)

2-D High- & Low-Pass Filters

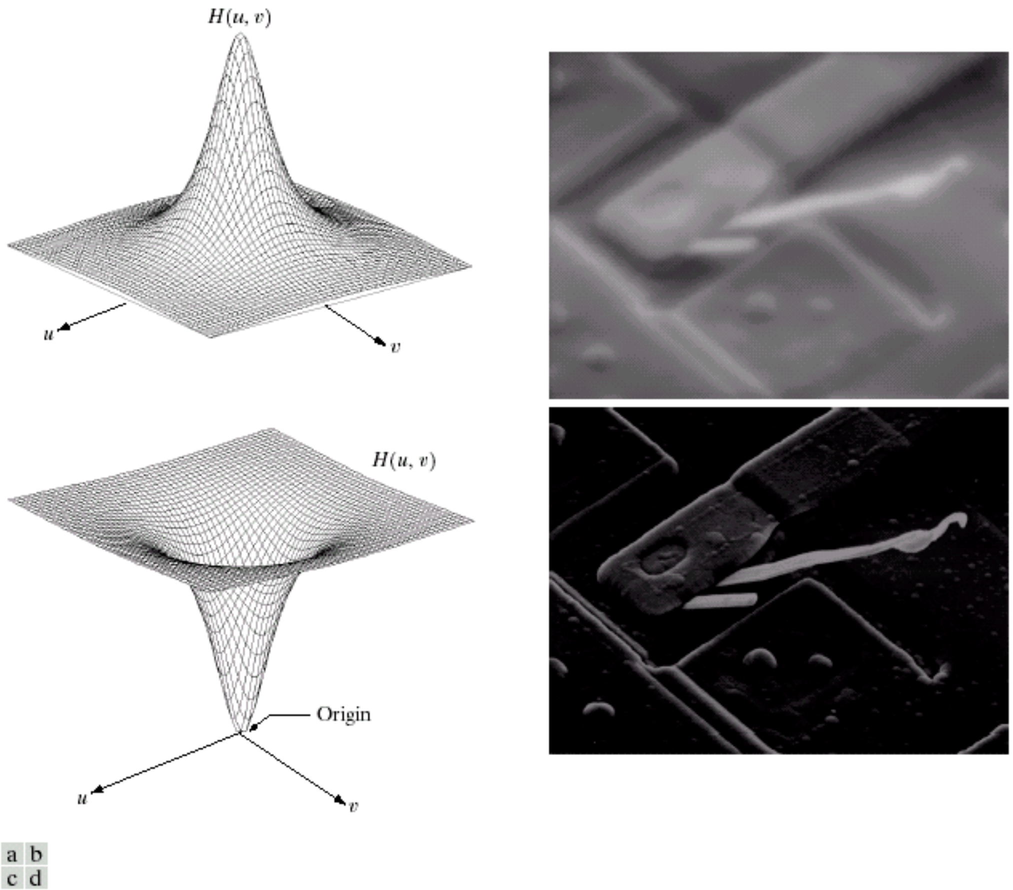
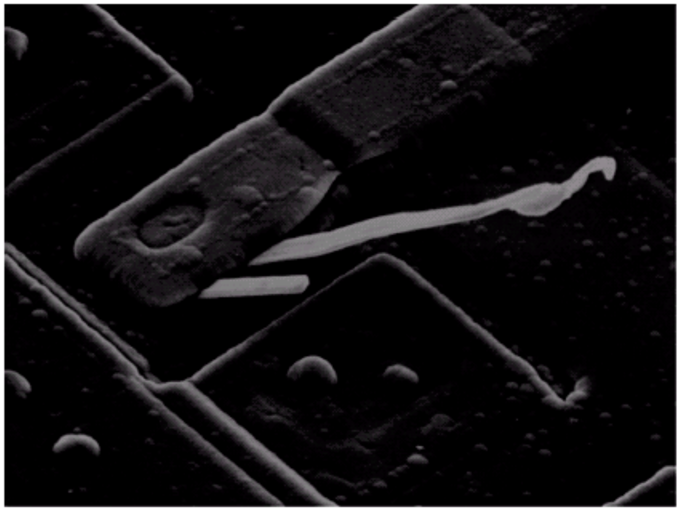


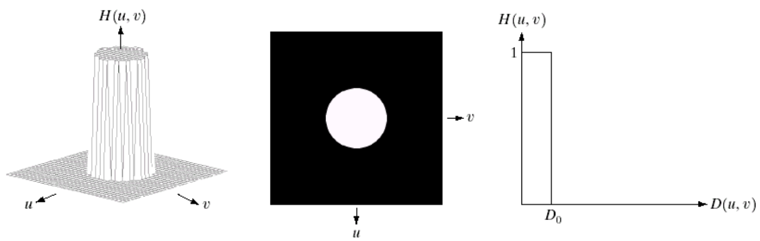
FIGURE 4.7 (a) A two-dimensional lowpass filter function. (b) Result of lowpass filtering the image in Fig. 4.4(a). (c) A two-dimensional highpass filter function. (d) Result of highpass filtering the image in Fig. 4.4(a).

2-D Notch Filter

FIGURE 4.6 Result of filtering the image in Fig. 4.4(a) with a notch filter that set to 0 the $F(0, 0)$ term in the Fourier transform.

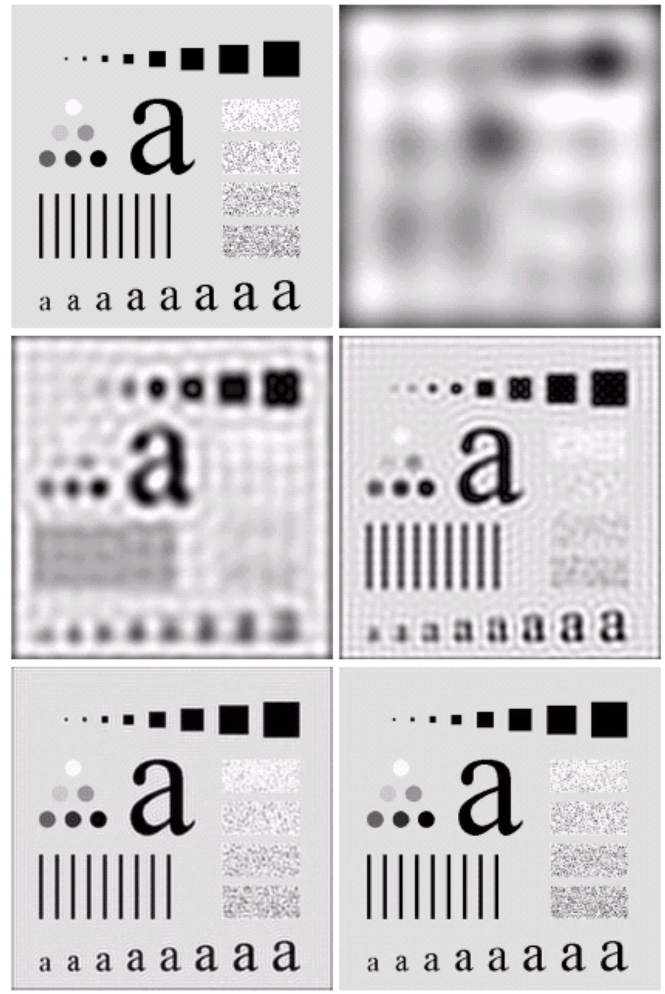


Low-Pass Filtering in Frequency Domain



a b c

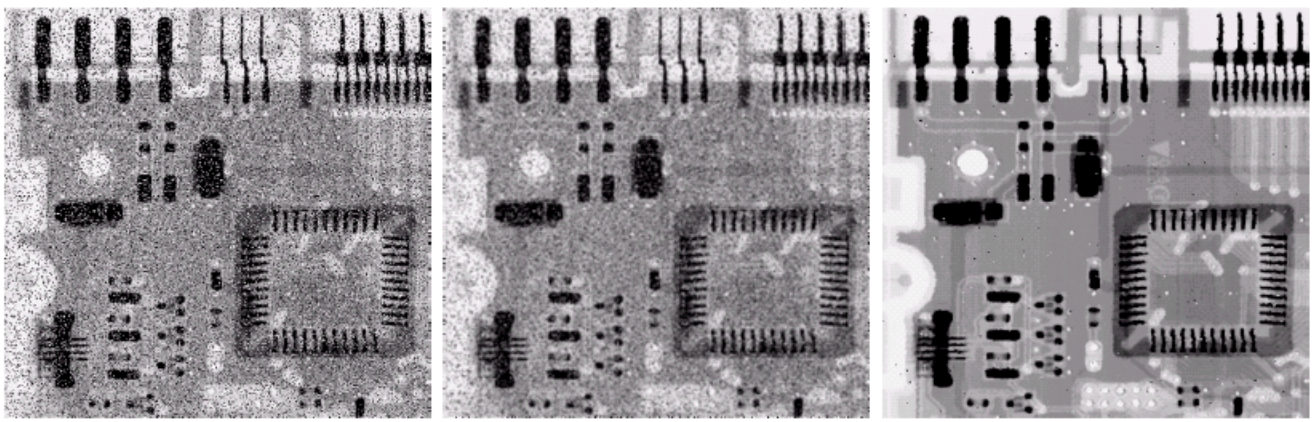
FIGURE 4.10 (a) Perspective plot of an ideal lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.



a b
c d
e f

FIGURE 4.12 (a) Original image. (b)–(f) Results of ideal lowpass filtering with cutoff frequencies set at radii values of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). The power removed by these filters was 8, 5.4, 3.6, 2, and 0.5% of the total, respectively.

Non-linear Filtering



a b c

FIGURE 3.37 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3 × 3 averaging mask. (c) Noise reduction with a 3 × 3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

1D Convolution

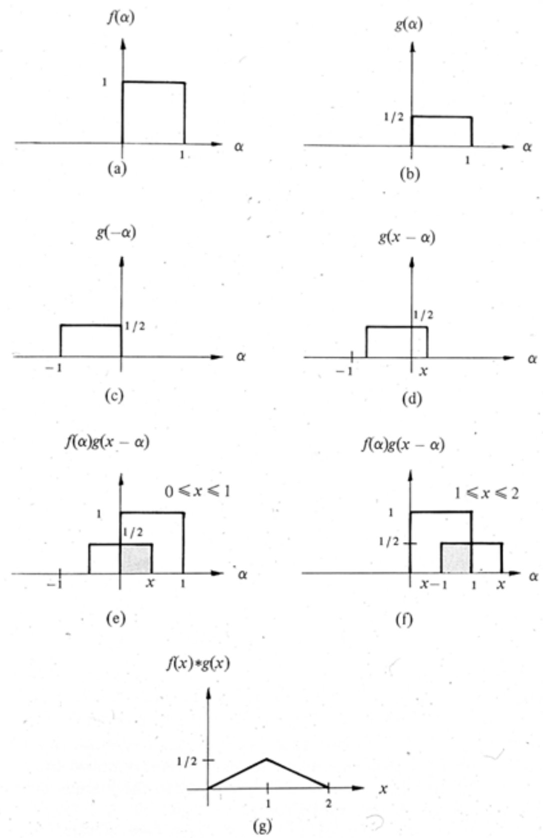
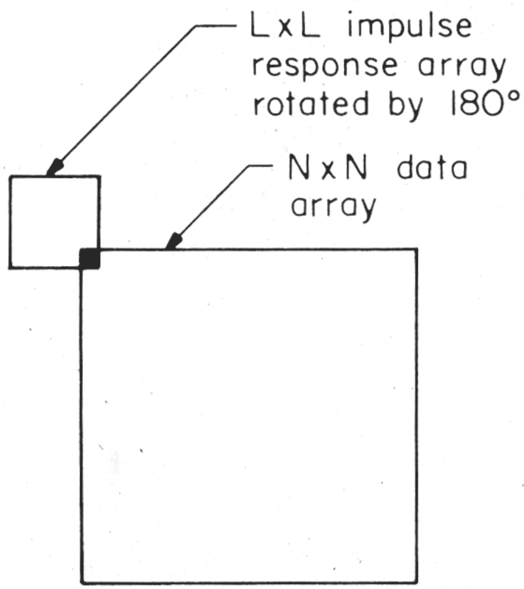


Figure 3.13. Graphical illustration of convolution. The shaded areas indicate regions where the product is not zero.

2D Convolution

Finite Area Superposition Operator



$$\begin{aligned} p(1,1) = {} & p(0,0)*k(0,0) + p(1,0)*k(1,0) \\ & + p(2,0)*k(2,0) + p(0,1)*k(0,1) \\ & + p(1,1)*k(1,1) + p(2,1)*k(2,1) \\ & + p(0,2)*k(0,2) + p(1,2)*k(1,2) \\ & + p(2,2)*k(2,2) \end{aligned}$$

or

$$p(1,1) = \sum_{m,n=0}^{2} k(m,n)*p(m,n)$$

FIGURE 9.1-1. Relationships between input data array and impulse response array for finite area superposition.

Computation Requirements

$$p(x,y) = \sum_{m,n=0}^{2} k(m,n)*p(x+m,y+n)$$

Convolving an area of size X by Y with a kernel of size n by m requires $X*Y*n*m$ multiplies and adds. Thus, a 256 by 256 image with a 3 by 3 kernel requires 589,824 multiply/add operations; this can take a long time on a computer without fast multiplication hardware.

2D Transforms as 1D Computations

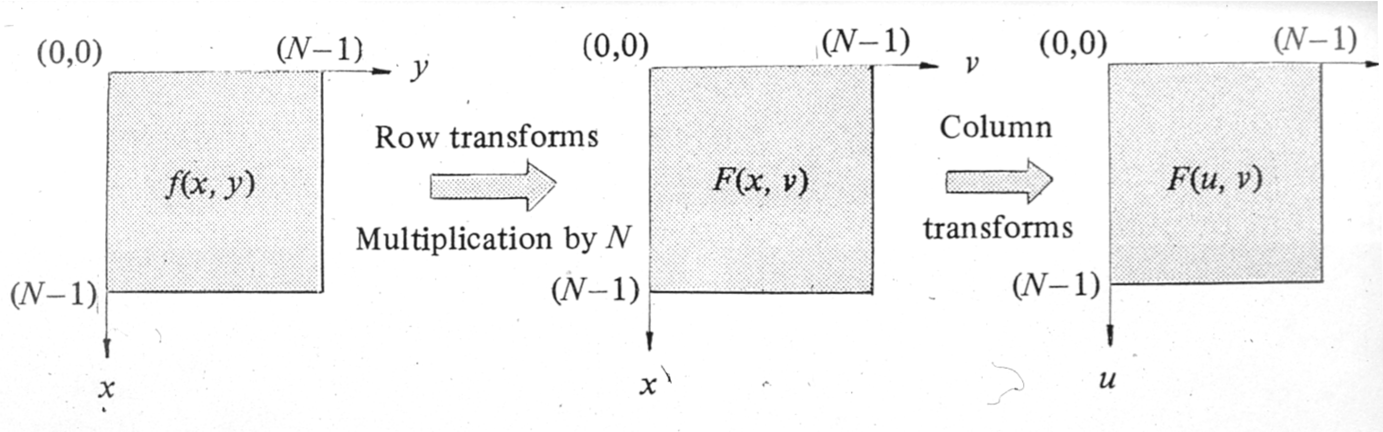


Figure 3.9. Computation of the two-dimensional Fourier transform as a series of o
dimensional transforms.

Sample C Code for Spatial Processing

Listing 5: *A C code fragment for a 3 by 3 convolution algorithm that uses separate source and destination memories to avoid overlapping the output convolution values with the inputs to the convolution.*

```
/* Set up kernel for "sharpening" (high-frequency boosting)
   the image */
  static int kernel[9] = {-1,-1,-1,
                          -1, 9,-1,
                          -1,-1,-1,};

/* Increment starting position and decrement image size
   to accommodate the convolution edge effects */
  x++; y++; dx--; dy--;
/* Set up address offsets for the output */
  xx = 0; yy = 0;
/* Scan through source image, output to destination */
  for (i = y ; i < y+dy ; i++) {
     xx = 0;                    /* Reset x output index */
     for (j = x ; j < x+dx ; j++) {
      sum = 0;                  /* Zero convolution sum */
      k_pointer = kernel;       /* Pointer to kernel values */
/* Inner loop to do convolution (correlation!) */
      for ( n = -1 ; n <= 1 ; n++) {
        for (m = -1 ; m <= 1 ; m++)
           sum = sum + read_pixel(i+m,j+n)*(*k_pointer++);
```

Spatial Derivatives

a
b c
d e

FIGURE 3.44
A 3 × 3 region of an image (the z's are gray-level values) and masks used to compute the gradient at point labeled z_5. All masks coefficients sum to zero, as expected of a derivative operator.

(a)

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

(b)

−1	0
0	1

(c)

0	−1
1	0

(d)

−1	−2	−1
0	0	0
1	2	1

(e)

−1	0	1
−2	0	2
−1	0	1

Edge Processing

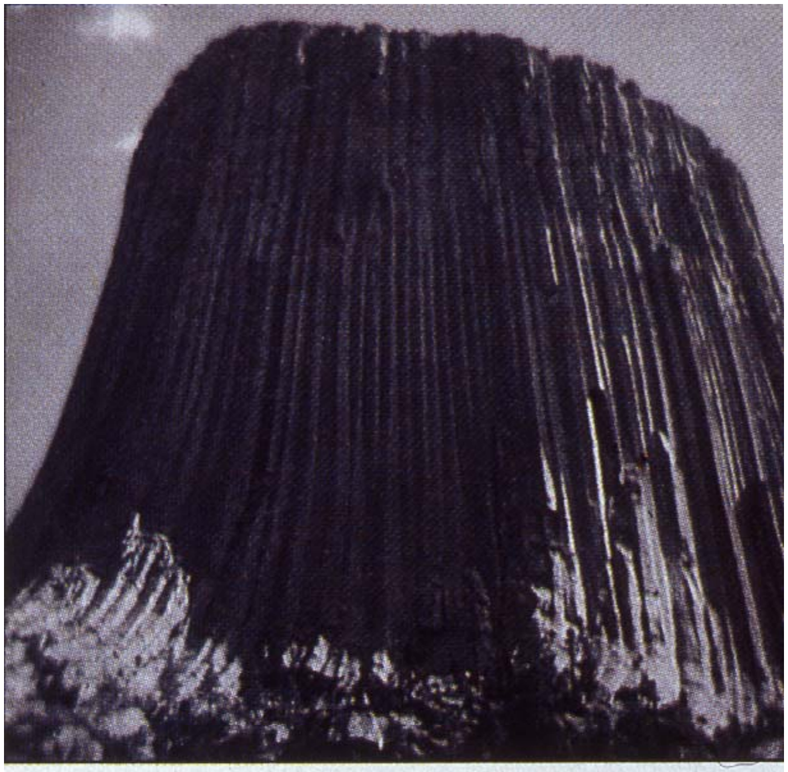


Photo 7: *An image of Devil's Tower National Monument in Wyoming, before image processing.*

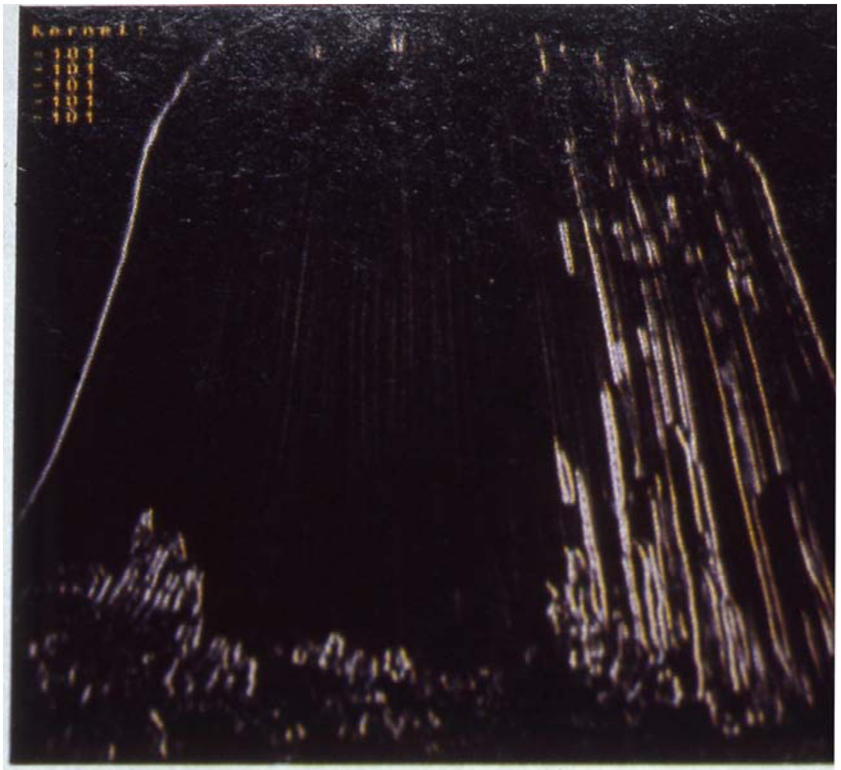
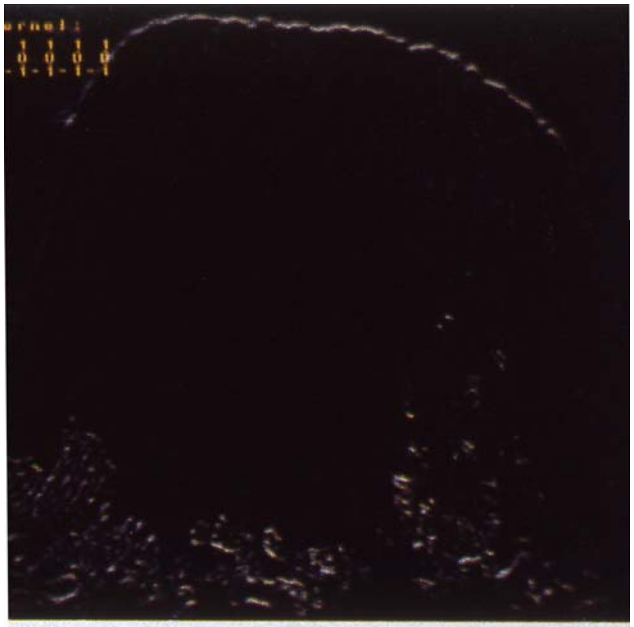


Photo 8: *Convolution of photo 7 with a kernel (shown in the upper left corner) that amplifies vertical edges.*

More Edge Processing



oto 9: *Convolution of photo 7 with kernel (shown in the upper left rner) that amplifies horizontal edges you can see, this image doesn't ve many horizontal edges.*

Second Order Derivatives

0	1	0
1	−4	1
0	1	0

Figure 7.37 Mask used to compute the Laplacian.

2D Edge Finding

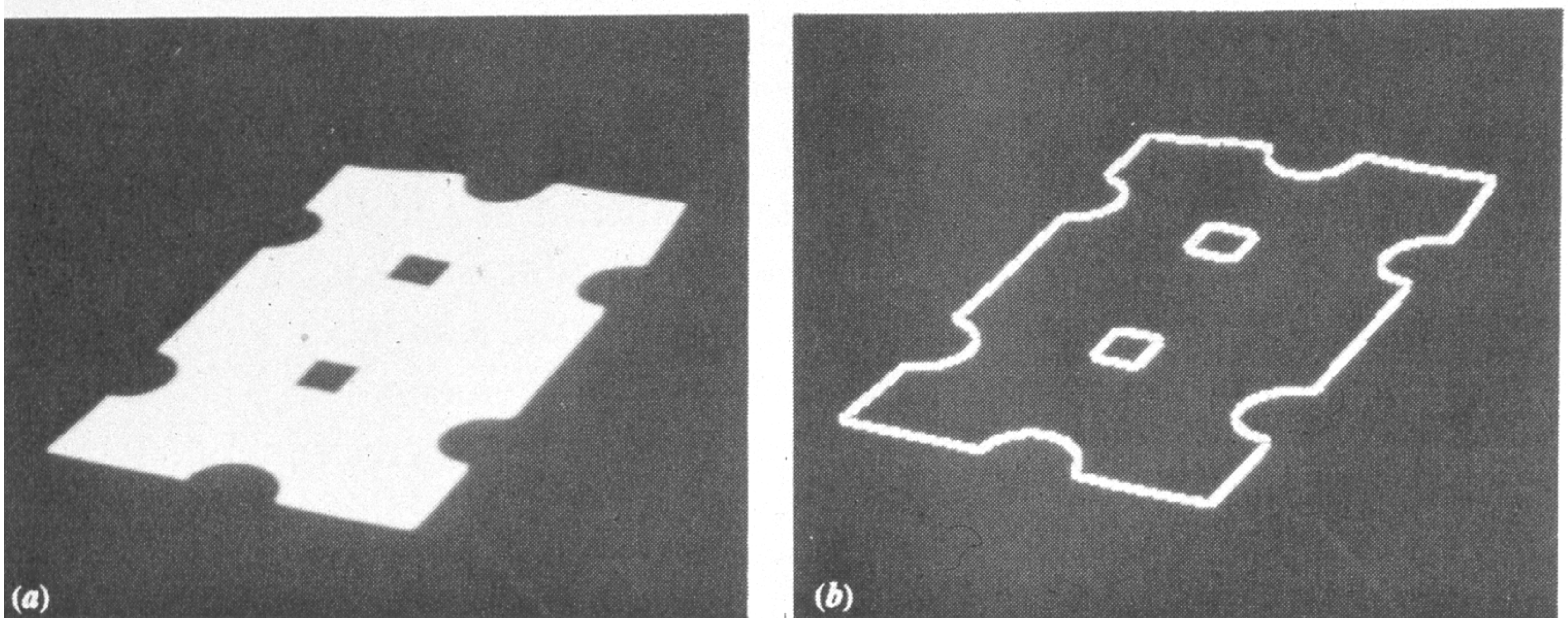


Figure 7.36 (*a*) Input image. (*b*) Result of using Eq. (7.6-44).

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

a	b
c	d

FIGURE 3.39 (a) Filter mask used to implement the digital Laplacian, as defined in Eq. (3.7-4). (b) Mask used to implement an extension of this equation that includes the diagonal neighbors. (c) and (d) Two other implementations of the Laplacian.

Edge Location

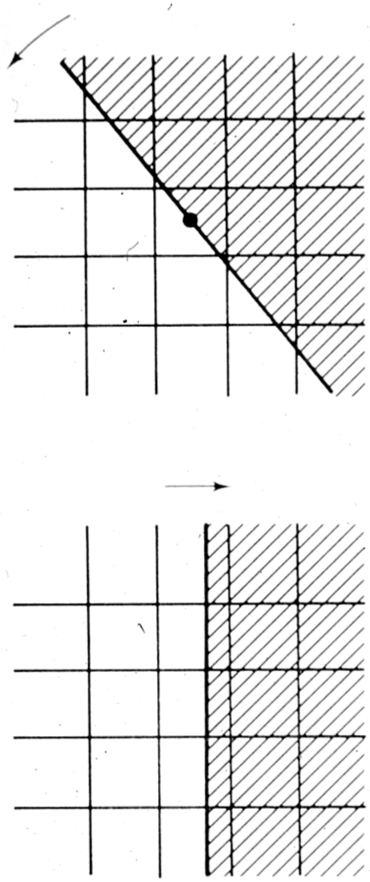


Fig. 3.11 Edge models for orientation and displacement sensitivity analyses.

Sharpening

a b
c d

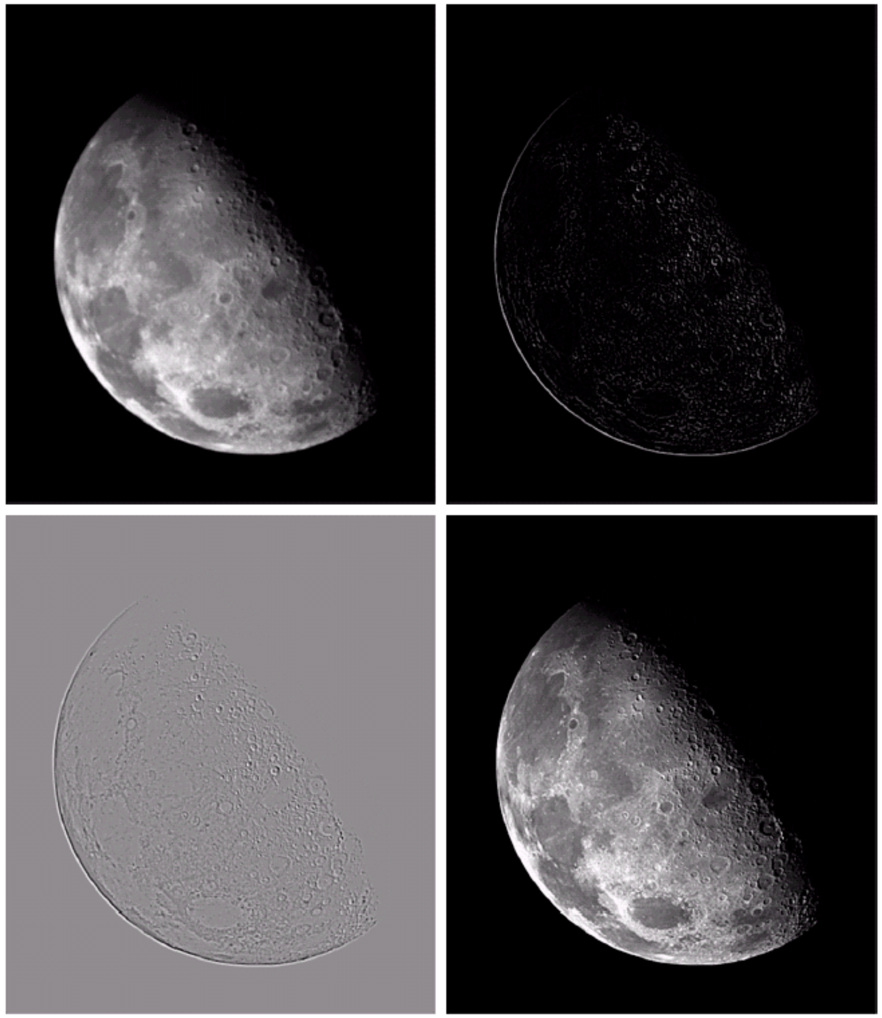


FIGURE 3.40
(a) Image of the North Pole of the moon.
(b) Laplacian-filtered image.
(c) Laplacian image scaled for display purposes.
(d) Image enhanced by using Eq. (3.7-5).
(Original image courtesy of NASA.)

Chapter 2: Digital Image Fundamentals

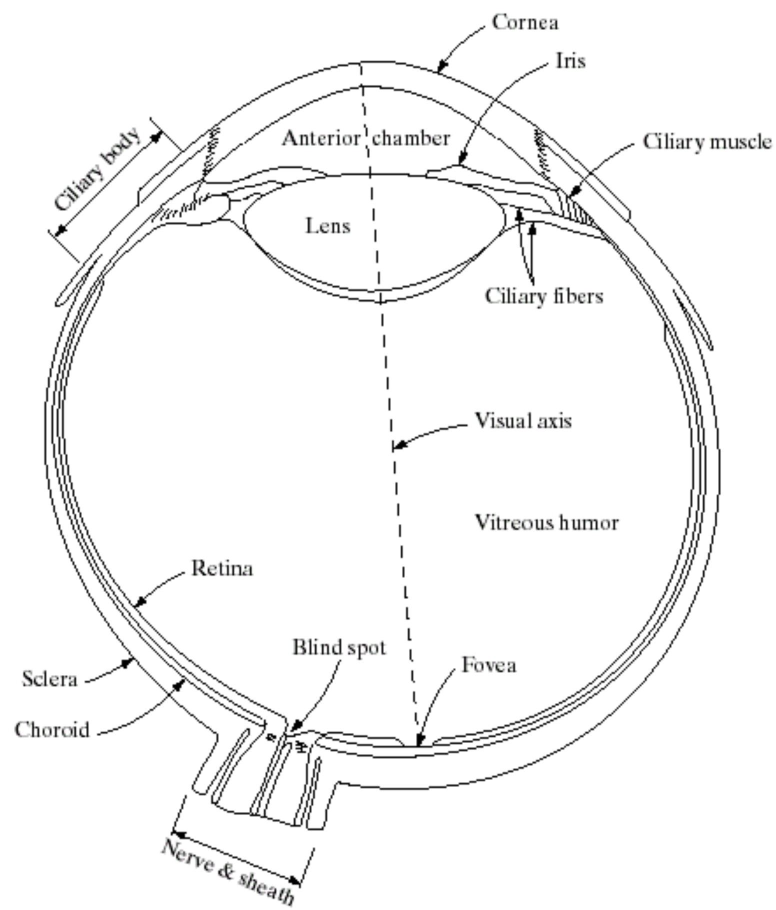


FIGURE 2.1 Simplified diagram of a cross section of the human eye.

Chapter 2: Digital Image Fundamentals

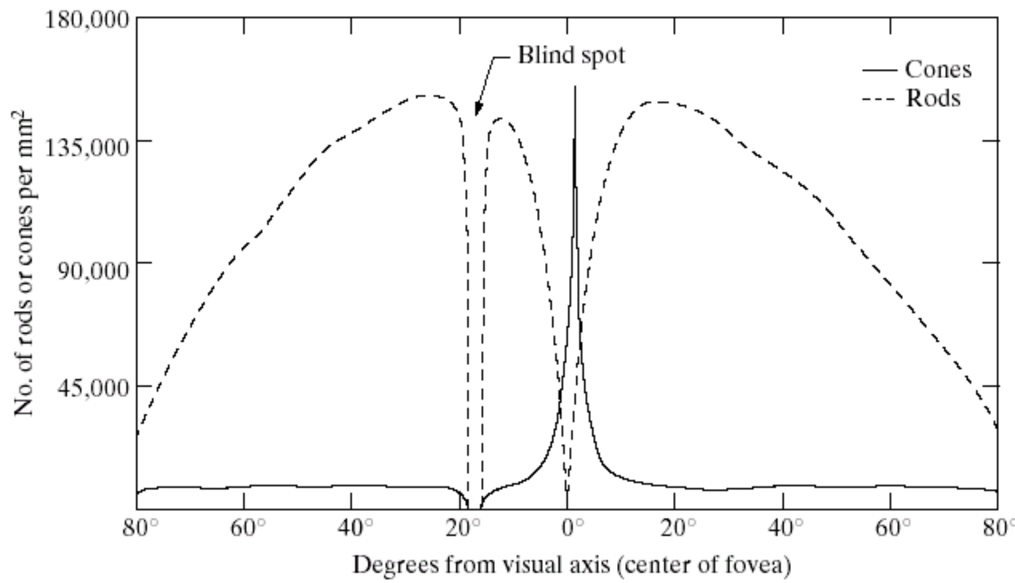
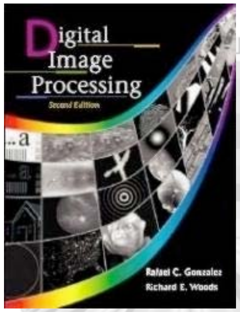
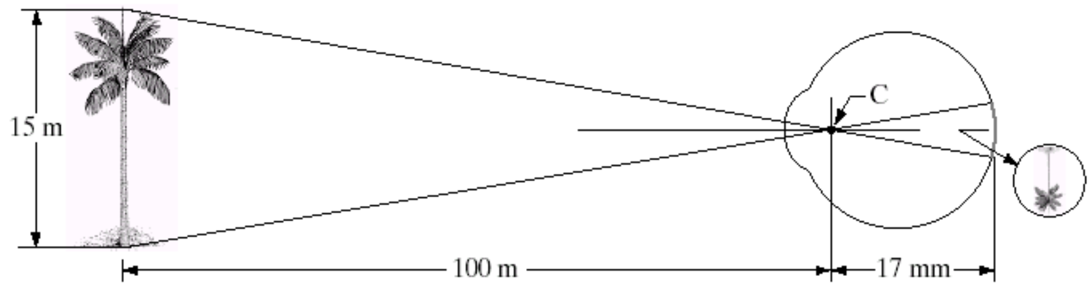


FIGURE 2.2 Distribution of rods and cones in the retina.



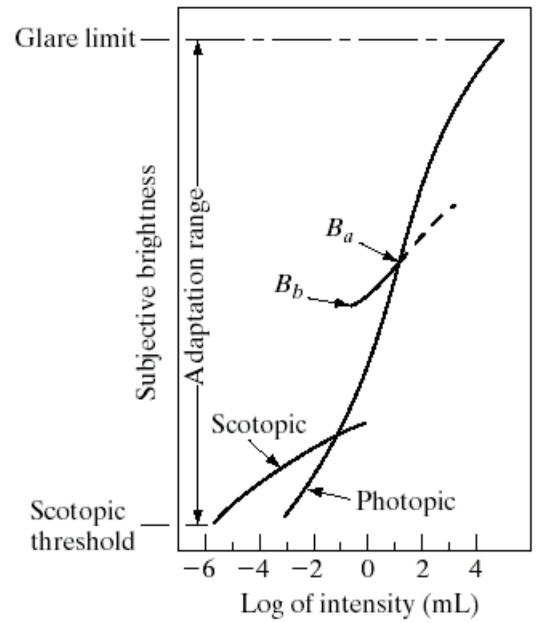
Chapter 2: Digital Image Fundamentals

FIGURE 2.3 Graphical representation of the eye looking at a palm tree. Point *C* is the optical center of the lens.



Chapter 2: Digital Image Fundamentals

FIGURE 2.4 Range of subjective brightness sensations showing a particular adaptation level.



Chapter 2: Digital Image Fundamentals

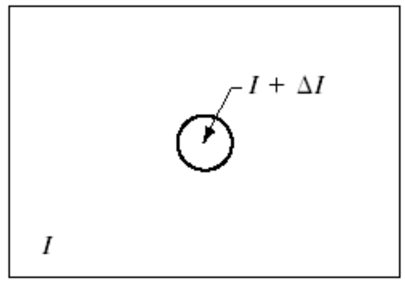


FIGURE 2.5 Basic experimental setup used to characterize brightness discrimination.

Chapter 2: Digital Image Fundamentals

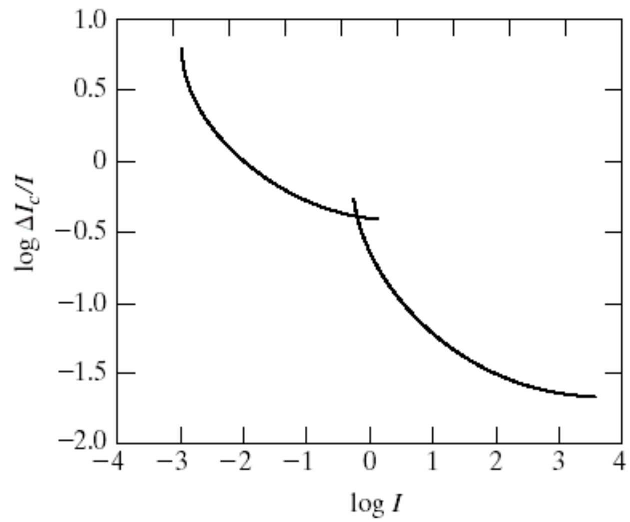
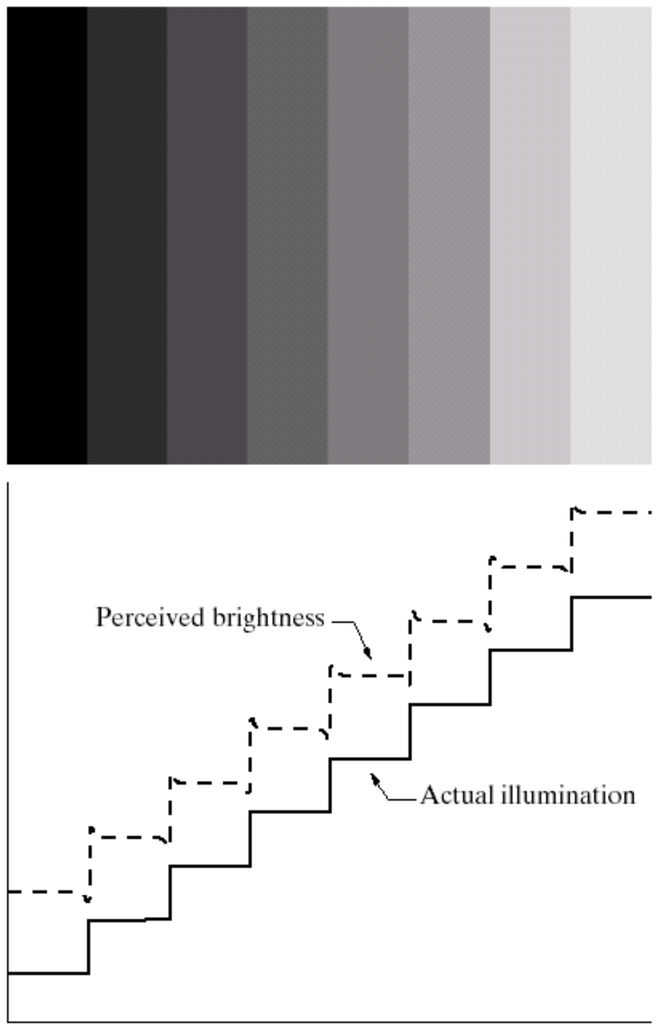


FIGURE 2.6 Typical Weber ratio as a function of intensity.

Chapter 2: Digital Image Fundamentals



a
b

FIGURE 2.7
(a) An example showing that perceived brightness is not a simple function of intensity. The relative vertical positions between the two profiles in (b) have no special significance; they were chosen for clarity.

Chapter 2: Digital Image Fundamentals

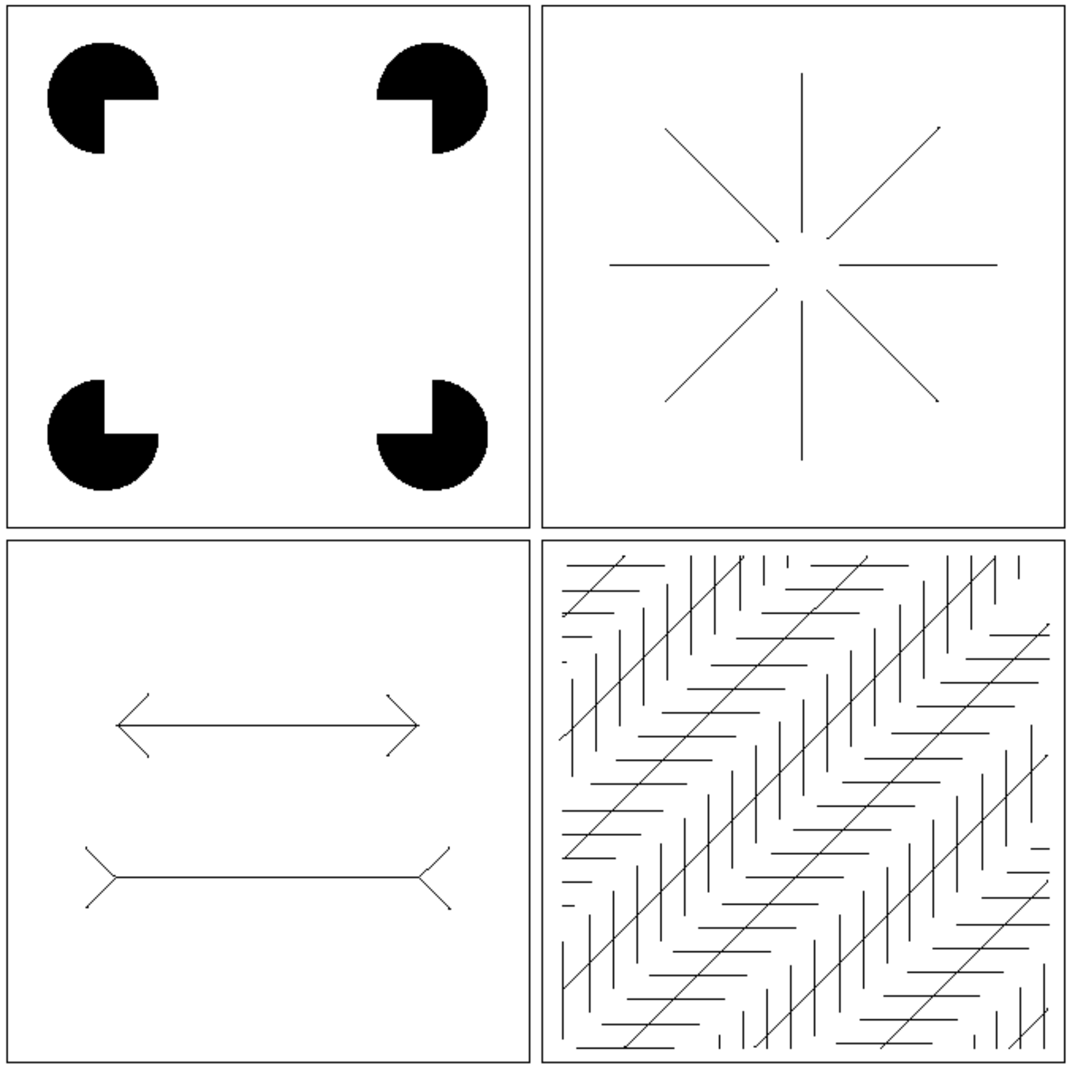
a b c

FIGURE 2.8 Examples of simultaneous contrast. All the inner squares have the same intensity, but they appear progressively darker as the background becomes lighter.

Chapter 2: Digital Image Fundamentals

a b
c d

FIGURE 2.9 Some well-known optical illusions.



Chapter 2: Digital Image Fundamentals

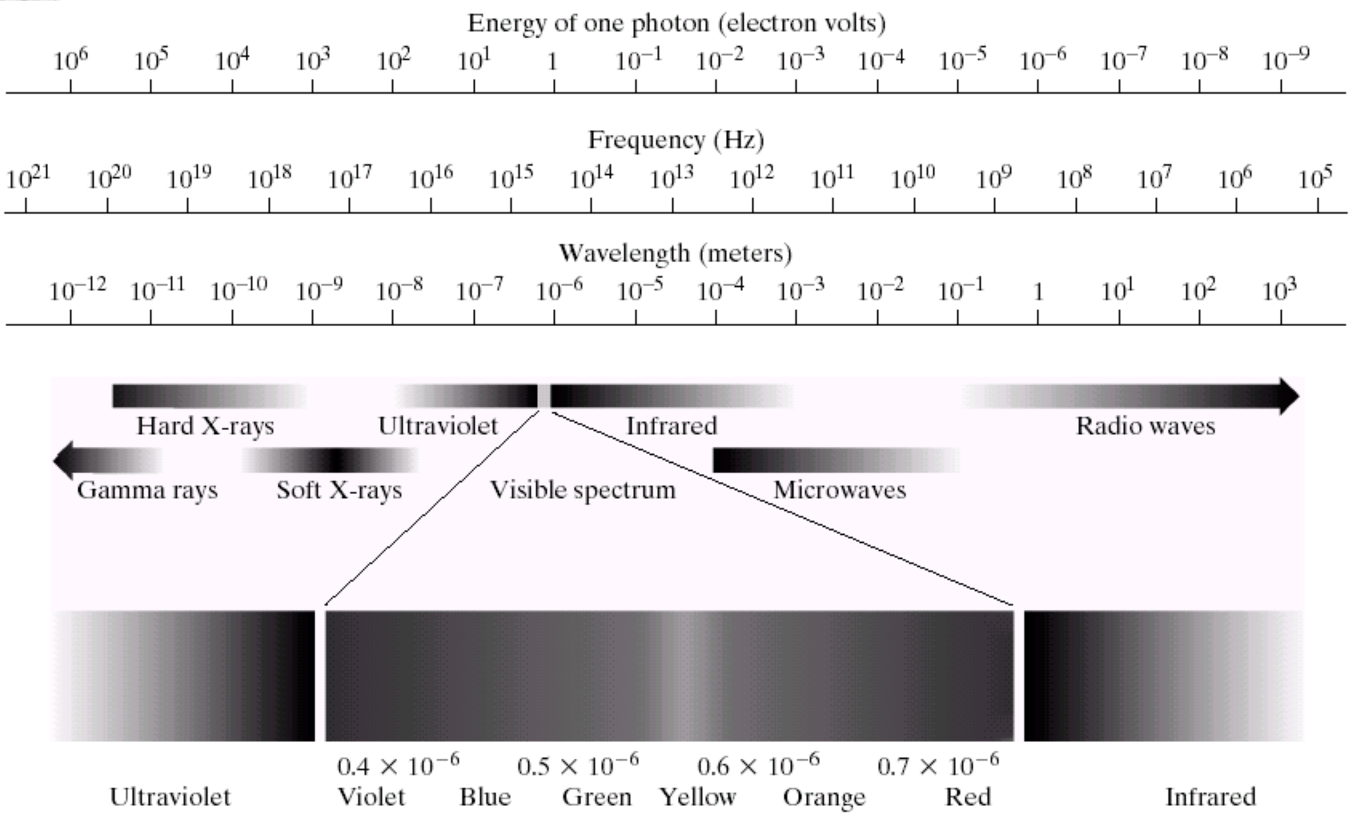


FIGURE 2.10 The electromagnetic spectrum. The visible spectrum is shown zoomed to facilitate explanation, but note that the visible spectrum is a rather narrow portion of the EM spectrum.

Chapter 2: Digital Image Fundamentals

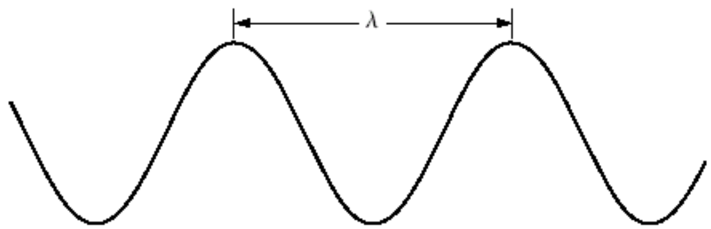
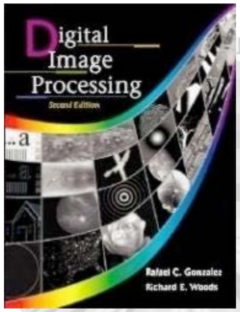


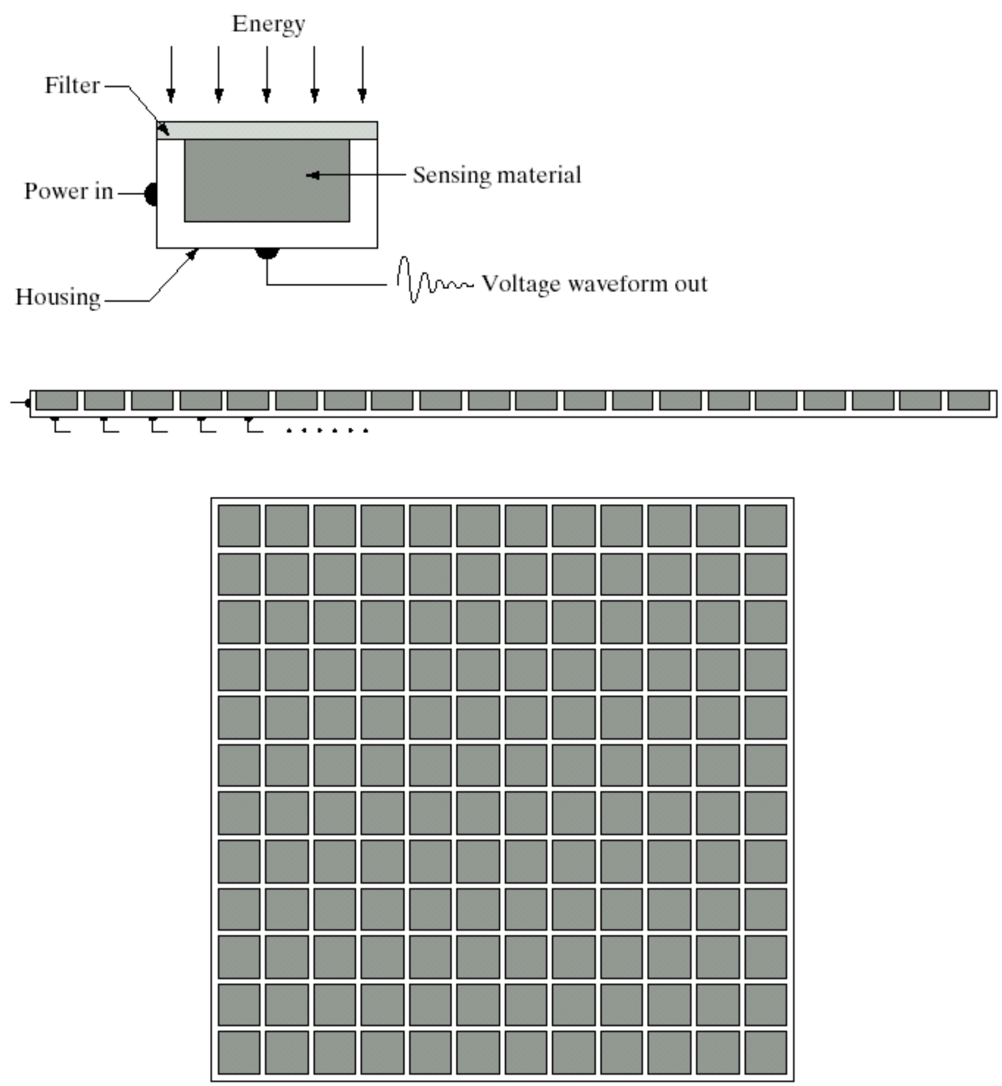
FIGURE 2.11 Graphical representation of one wavelength.



Chapter 2: Digital Image Fundamentals

a
b
c

FIGURE 2.12
(a) Single imaging sensor.
(b) Line sensor.
(c) Array sensor.



Chapter 2: Digital Image Fundamentals

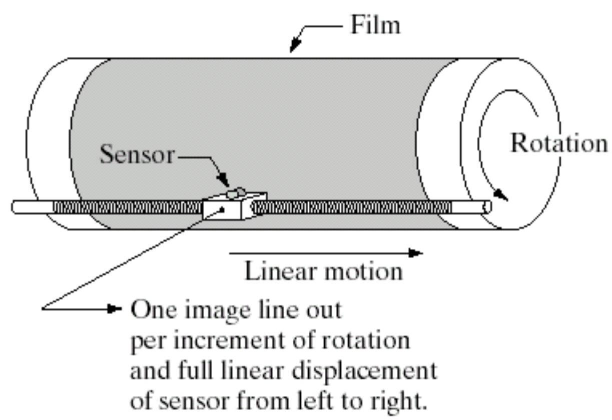
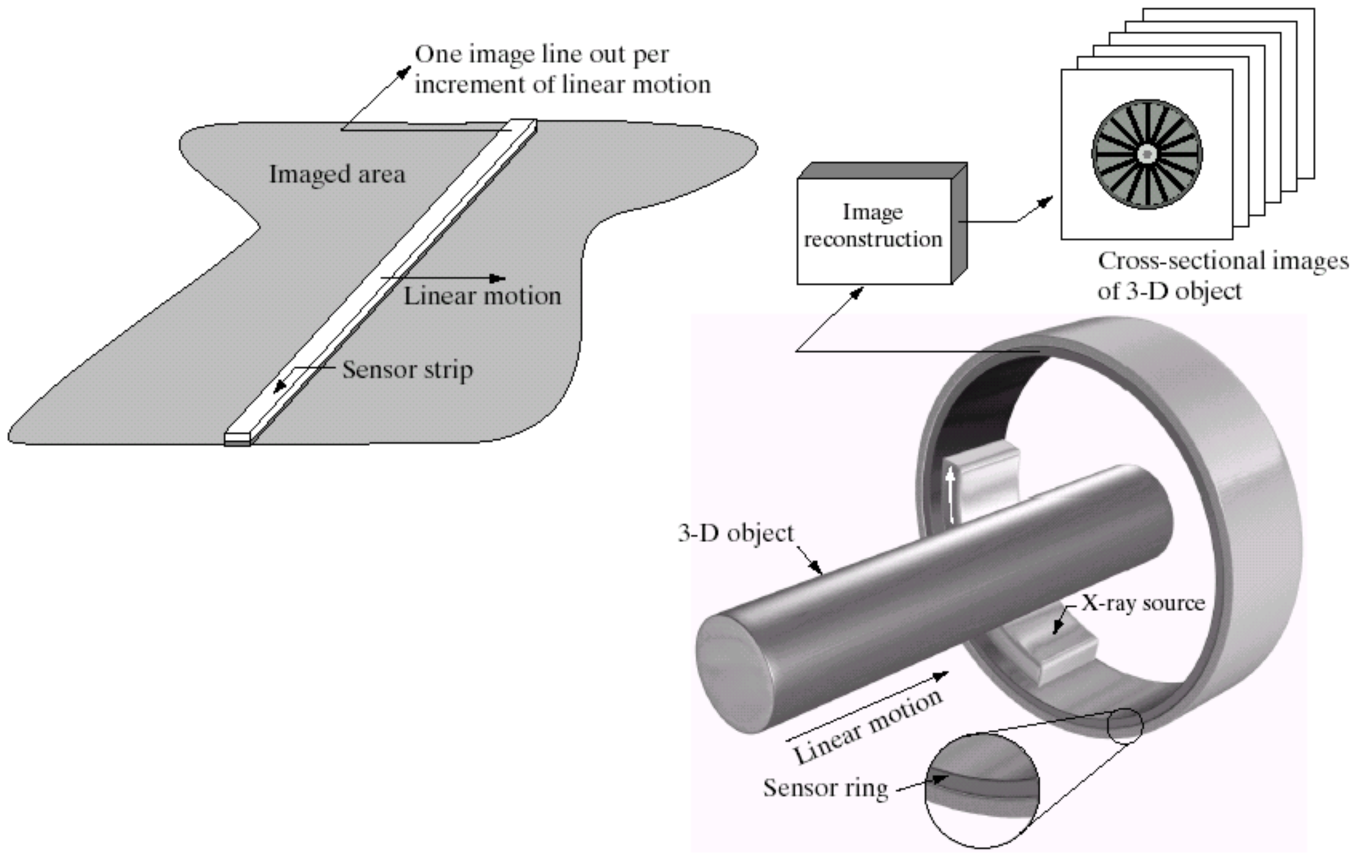


FIGURE 2.13 Combining a single sensor with motion to generate a 2-D image.

Chapter 2: Digital Image Fundamentals



a b

FIGURE 2.14 (a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

Chapter 2: Digital Image Fundamentals

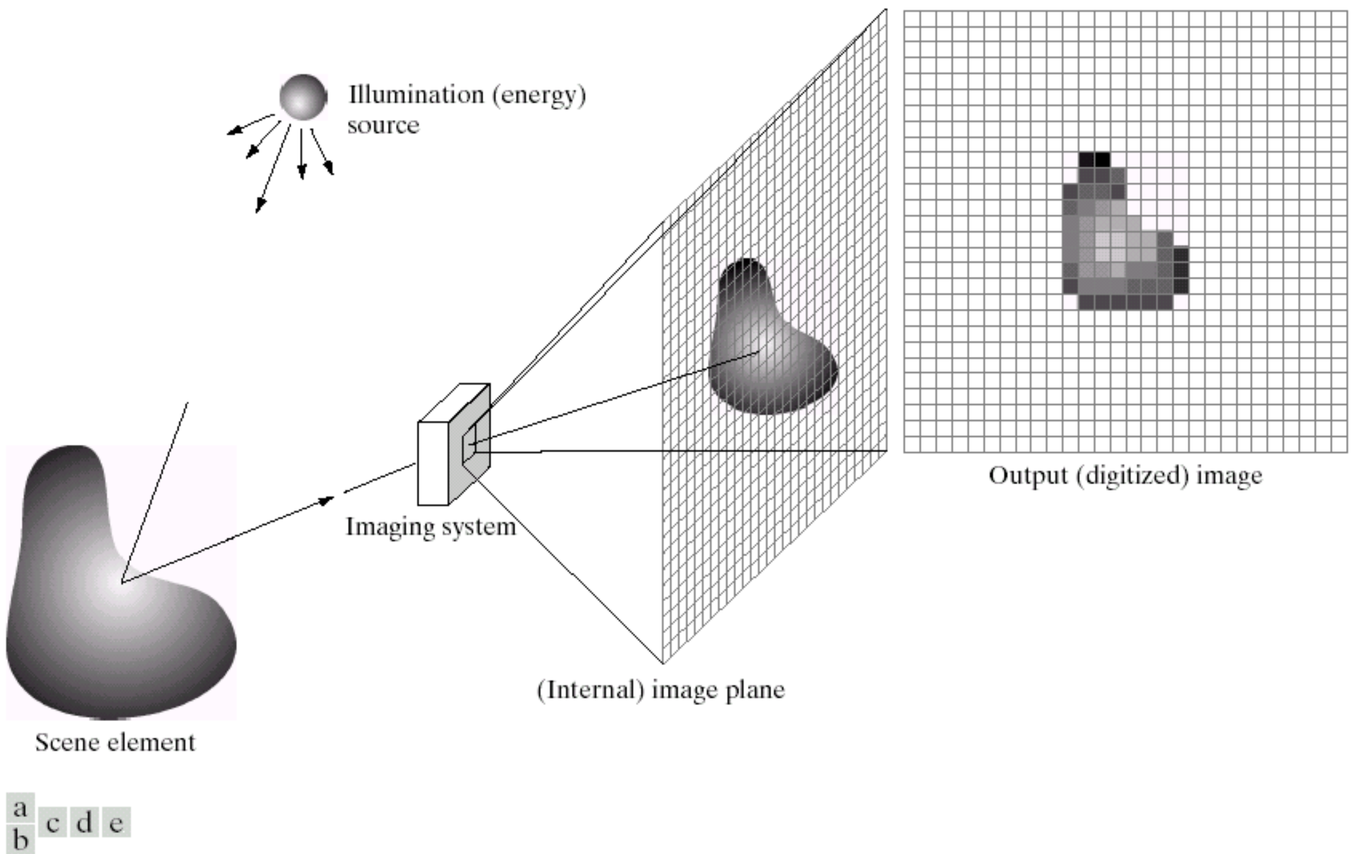


FIGURE 2.15 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

Chapter 2: Digital Image Fundamentals

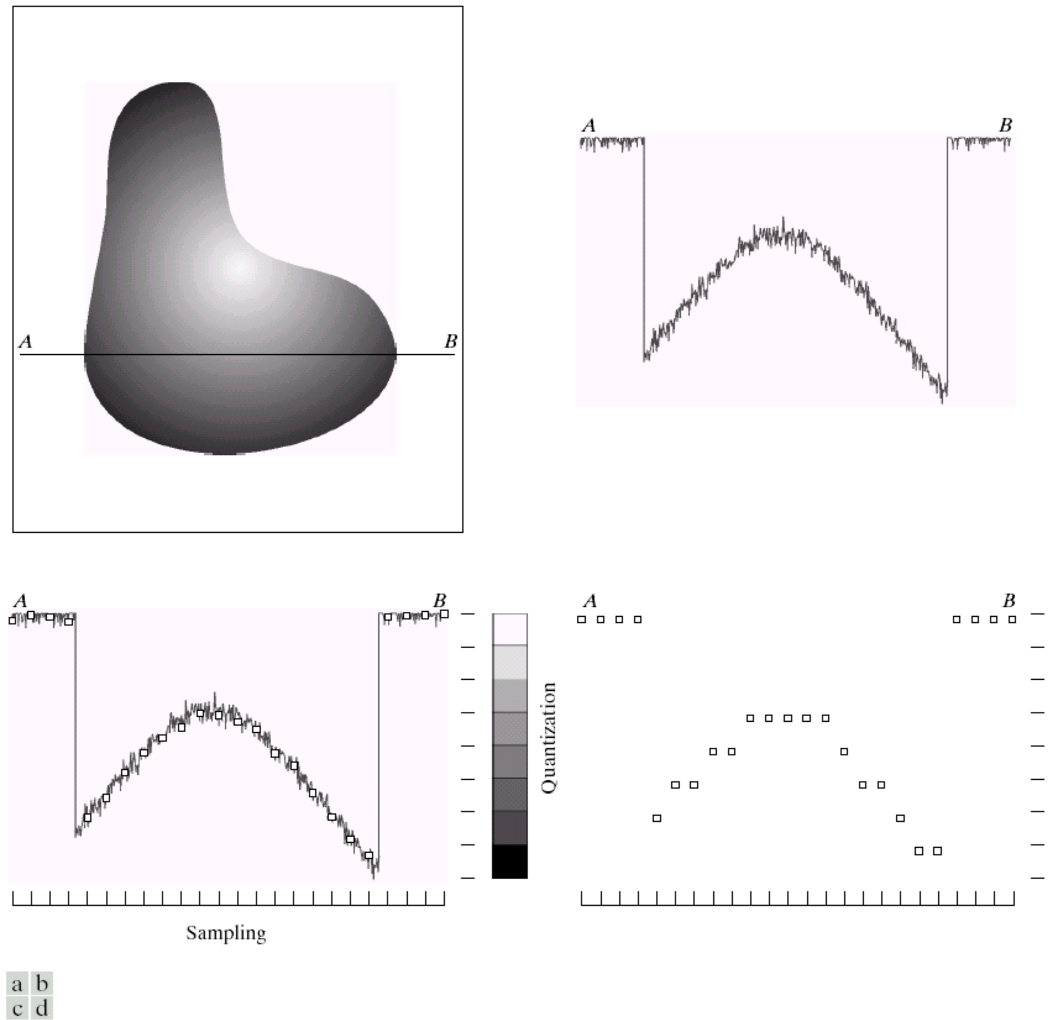
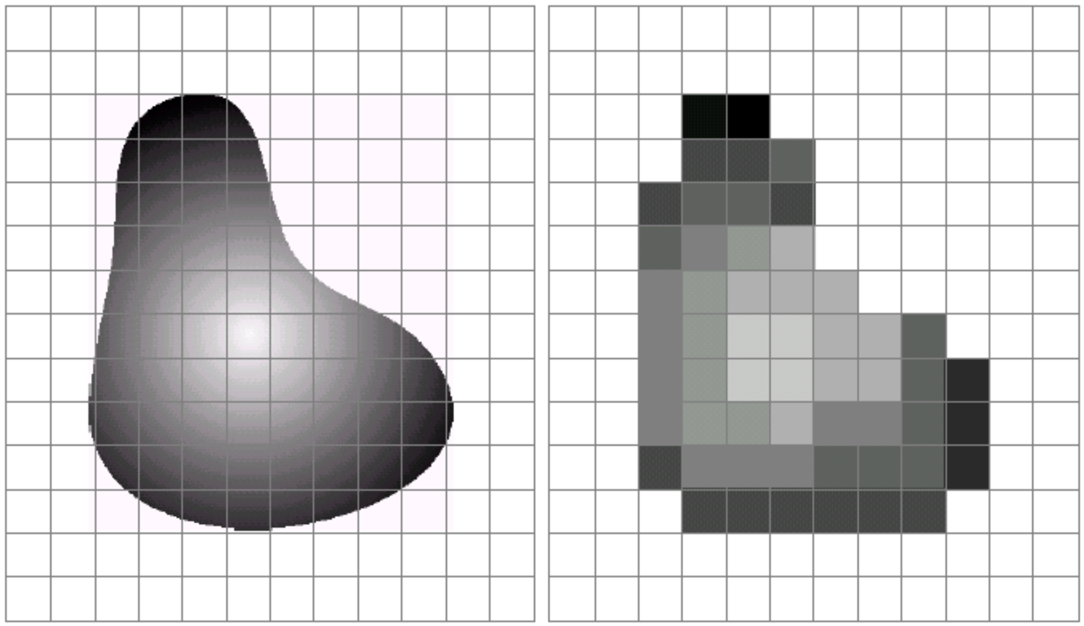


FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from *A* to *B* in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

Chapter 2: Digital Image Fundamentals



a b

FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

Chapter 2: Digital Image Fundamentals

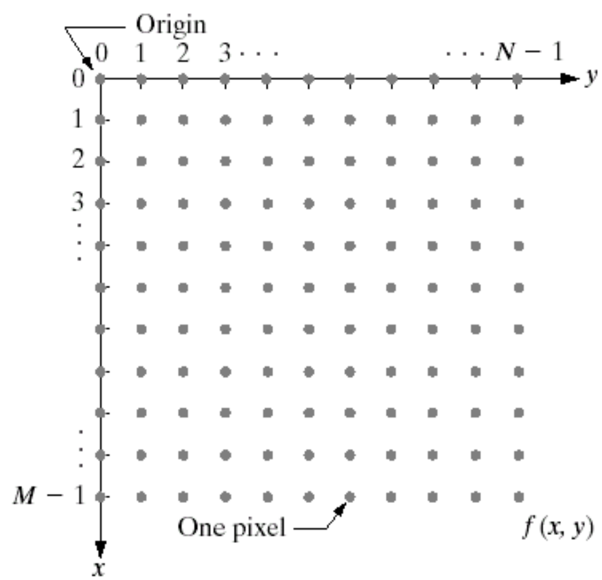


FIGURE 2.18 Coordinate convention used in this book to represent digital images.

Chapter 2: Digital Image Fundamentals

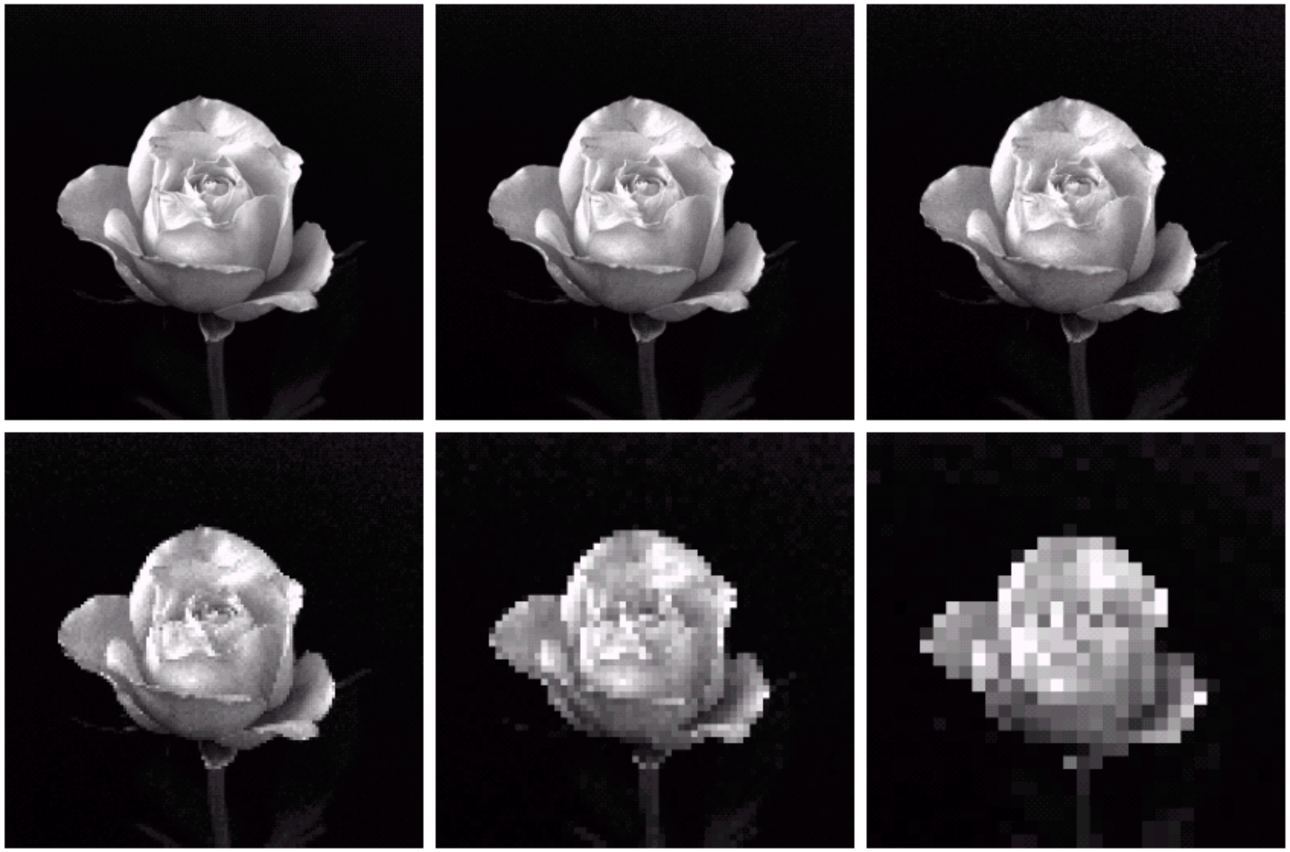
TABLE 2.1
Number of storage bits for various values of N and k.

N/k	1 (L = 2)	2 (L = 4)	3 (L = 8)	4 (L = 16)	5 (L = 32)	6 (L = 64)	7 (L = 128)	8 (L = 256)
32	1,024	2,048	3,072	4,096	5,120	6,144	7,168	8,192
64	4,096	8,192	12,288	16,384	20,480	24,576	28,672	32,768
128	16,384	32,768	49,152	65,536	81,920	98,304	114,688	131,072
256	65,536	131,072	196,608	262,144	327,680	393,216	458,752	524,288
512	262,144	524,288	786,432	1,048,576	1,310,720	1,572,864	1,835,008	2,097,152
1024	1,048,576	2,097,152	3,145,728	4,194,304	5,242,880	6,291,456	7,340,032	8,388,608
2048	4,194,304	8,388,608	12,582,912	16,777,216	20,971,520	25,165,824	29,369,128	33,554,432
4096	16,777,216	33,554,432	50,331,648	67,108,864	83,886,080	100,663,296	117,440,512	134,217,728
8192	67,108,864	134,217,728	201,326,592	268,435,456	335,544,320	402,653,184	469,762,048	536,870,912

Chapter 2: Digital Image Fundamentals

FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

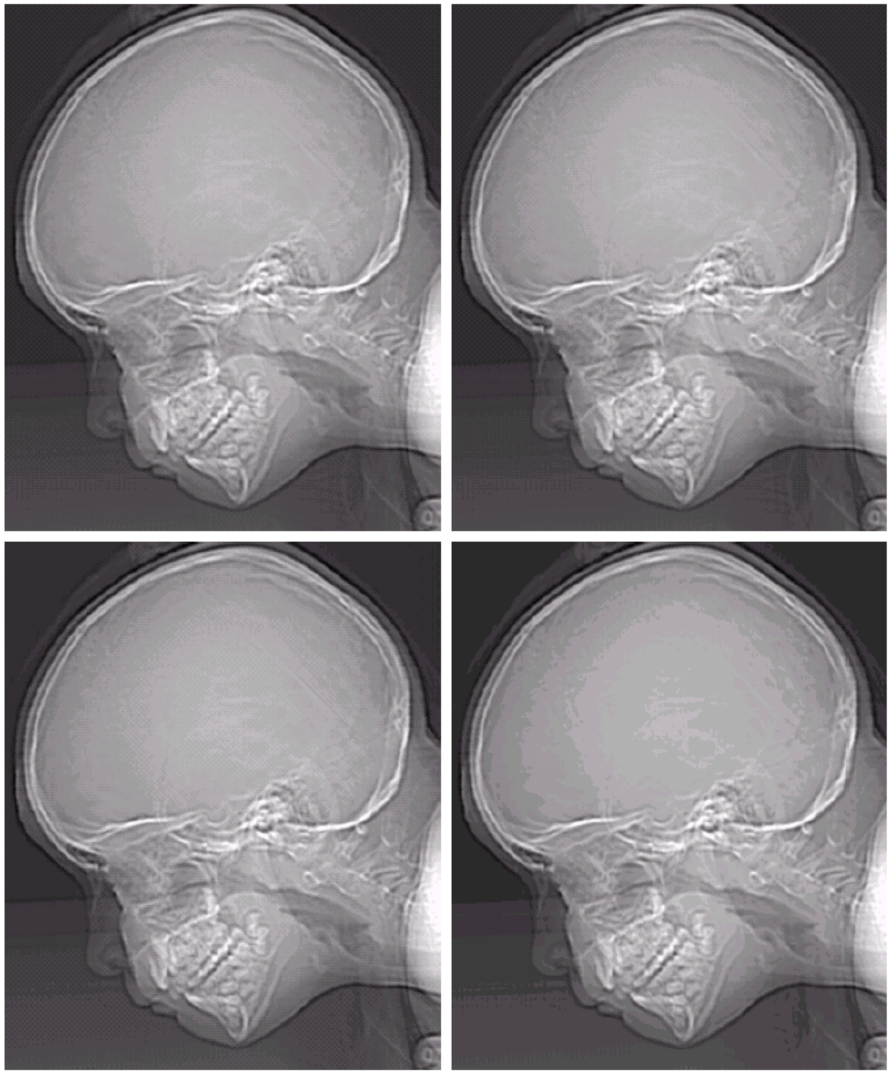
Chapter 2: Digital Image Fundamentals



a b c
d e f

FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

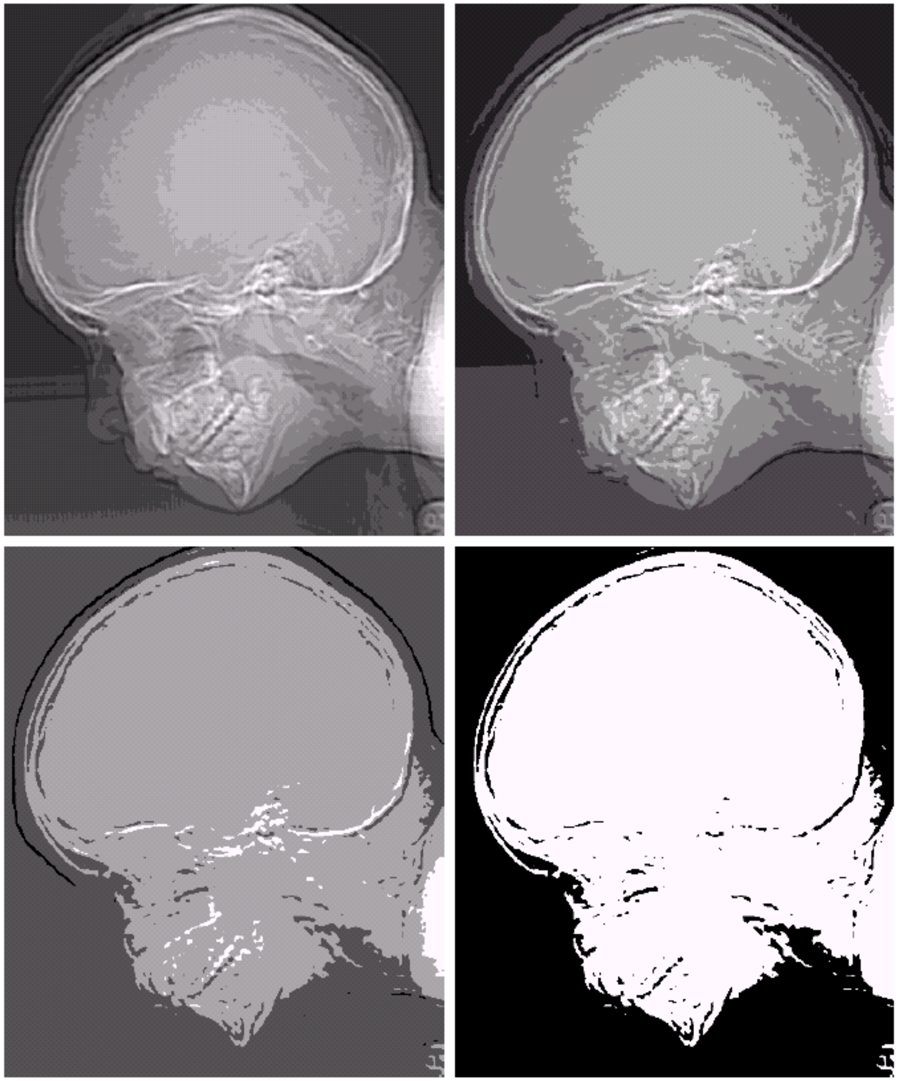
Chapter 2: Digital Image Fundamentals



a b
c d

FIGURE 2.21 (a) 452 × 374, 256-level image. (b)–(d) Image displayed in 128, 64, and 32 gray levels, while keeping the spatial resolution constant.

Chapter 2: Digital Image Fundamentals



e f
g h

FIGURE 2.21 *(Continued)* (e)–(h) Image displayed in 16, 8, 4, and 2 gray levels. (Original courtesy of Dr. David R. Pickens, Department of Radiology & Radiological Sciences, Vanderbilt University Medical Center.)

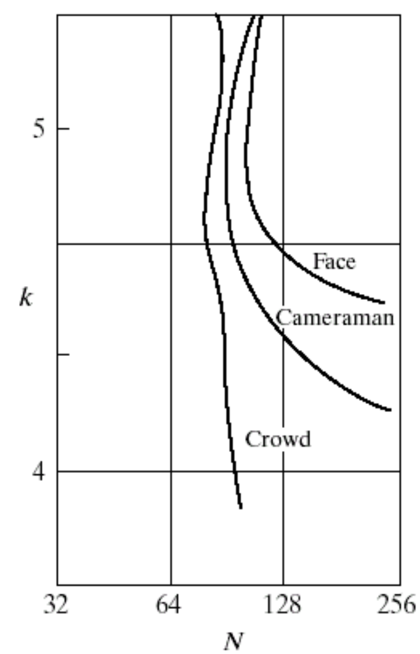
Chapter 2: Digital Image Fundamentals

a b c

FIGURE 2.22 (a) Image with a low level of detail. (b) Image with a medium level of detail. (c) Image with a relatively large amount of detail. (Image (b) courtesy of the Massachusetts Institute of Technology.)

Chapter 2: Digital Image Fundamentals

FIGURE 2.23 Representative isopreference curves for the three types of images in Fig. 2.22.



Chapter 2: Digital Image Fundamentals

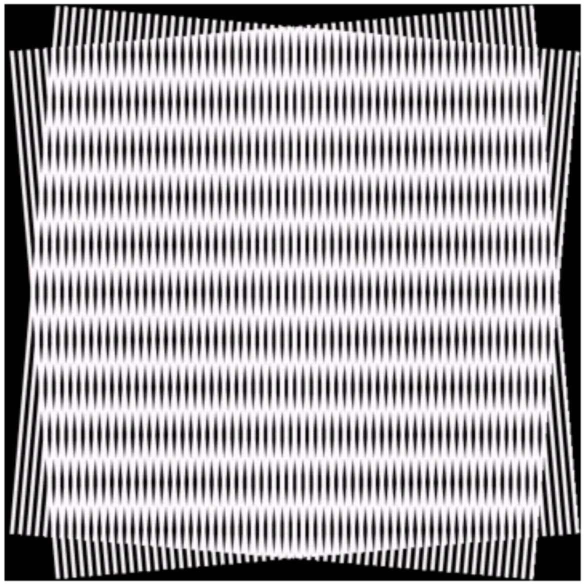
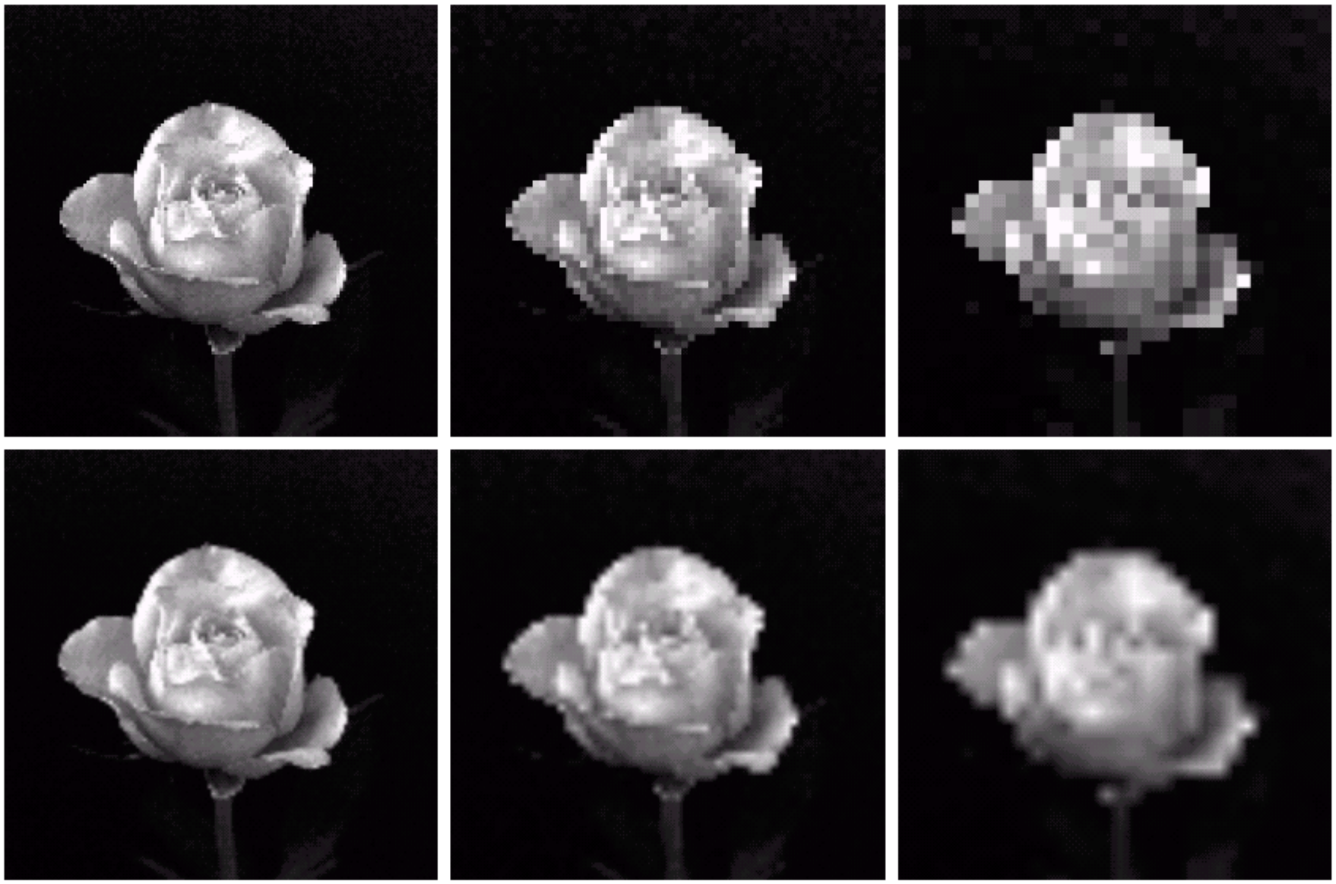


FIGURE 2.24 Illustration of the Moiré pattern effect.

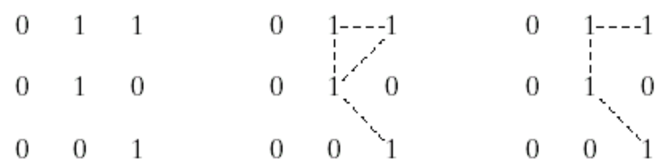
Chapter 2: Digital Image Fundamentals



a b c
d e f

FIGURE 2.25 Top row: images zoomed from 128 × 128, 64 × 64, and 32 × 32 pixels to 1024 × 1024 pixels, using nearest neighbor gray-level interpolation. Bottom row: same sequence, but using bilinear interpolation.

Chapter 2: Digital Image Fundamentals



a b c

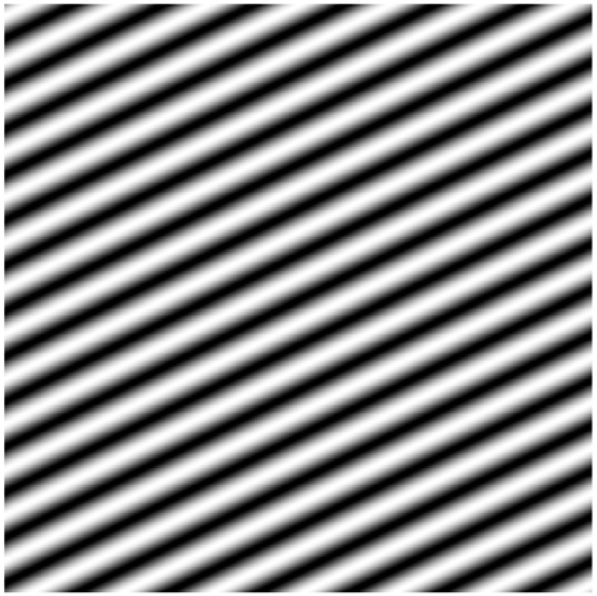
FIGURE 2.26 (a) Arrangement of pixels; (b) pixels that are 8-adjacent (shown dashed) to the center pixel; (c) *m*-adjacency.

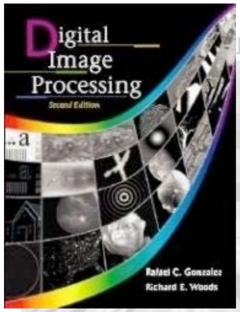
MATLAB/Image Processing Toolbox

```
function [rt, f, g] = twodsin(A,u0, v0, M, N)
% TWODSIN Compares for loops vs. vectorization.
% The comparison is based on implementing the function
% f(x,y)=Asin(u0x+v0y) for x=0,1,2,...,M-1 and
% y=0,1,2,...,N-1.  The inputs to the function are
% M and N and the constants in the function.
% GWE, Example 2.13, p.57

% First implement using for loops
tic %start timing
for r=1:M
   u0x=u0*(r-1);
   for c=1:N
     v0y=v0*(c-1);
     f(r,c)=A*sin(u0x+v0y);
   end
end
t1=toc; % End timing

% Now implement using vectorization
tic %start timing
r=0:M-1;
c=0:N-1;
[C,R]=meshgrid(c,r);
%special MATLAB function for fast 2D function evaluations
% creates all the (x,y) pairs for function evaluation
g=A*sin(u0*R+v0*C);

t2=toc; %End timing

%compute the ratio of the two times
rt=t1/(t2+eps); %use eps in case t2 is close to zero.
```

MATLAB/Image Processing Toolbox

```
>> [rt,f,g]=twodsin(1, 1/(2*pi), 1/(4*pi), 512, 512);
>> rt
rt =
    34.2520   % I only got ~19 on my old machine.
>>g=mat2gray(g);
>> imshow(g)  %show in separate window.
```

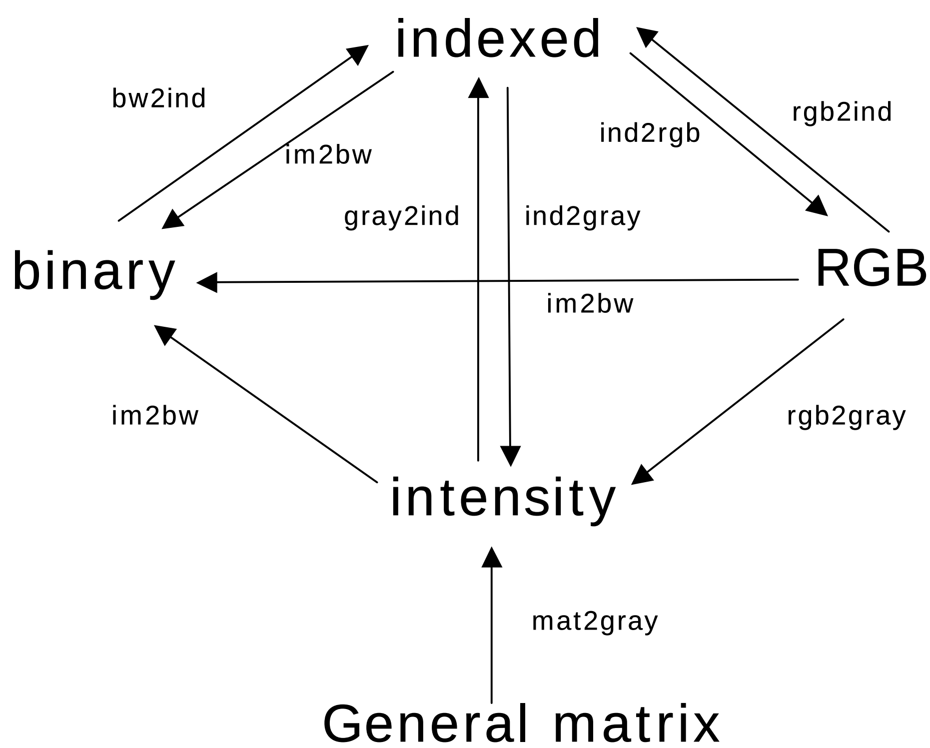




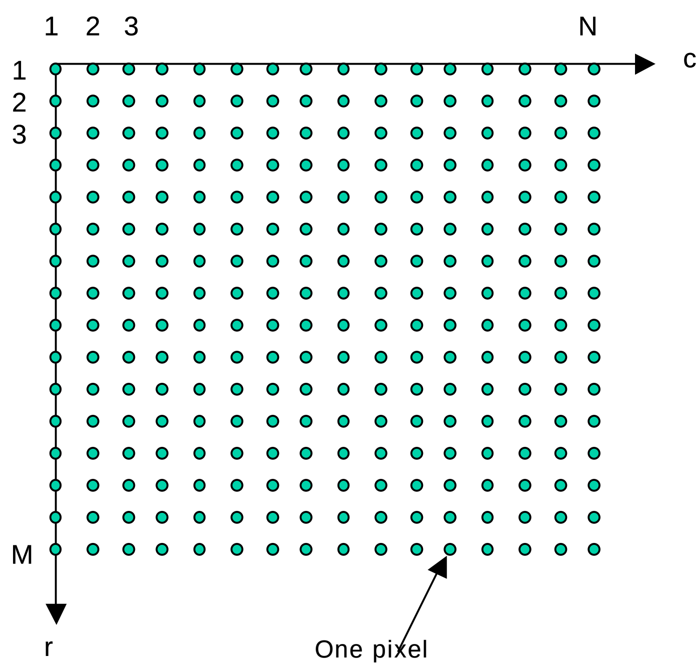
MATLAB/Image Processing Toolbox

```
imshow (f)
%f is an image array

imwrite(f, 'filename')
% filename MUST contain a recognized file format extension
% .tif or .tiff identify TIFF
% .jpg identifies JPEG
% additional parameters for tiff and jpeg identify compression, etc.

imfinfo filename
% returns all kind of useful file information such as size

g=imread('filename')
% filename MUST contain an appropriate extension
```

MATLAB/Image Processing Toolbox



MATLAB/Image Processing Toolbox



MATLAB/Image Processing Toolbox

```
>> h=imhist(f)                        %any previously loaded image
                                       % must be gray scale image
>> h1=h(1:10:256)                     %create bins for horiz axis
>> horz=1:10:256;                     %
>> bar(horz, h1)                      %
>> axis([0 255 0 15000])               %expand lower range of y-axis
>> set(gca, 'xtick', 0:50:255)        %gca means 'get current axis'
>> set(gca, 'ytick', 0:2000:15000) %lab h & v ticks
```

See GWE, p.77-78