EE 604, Digital Image Processing

Chapter 3: Intensity Transformations and Spatial Filtering

Dr. W. David Pan
Dept. of ECE
UAH

Preview

- **Spatial domain** refers to the image plane itself, and image processing methods in this category are based on direct manipulation of pixels in an image.
- Image processing in a **transform domain** involves first transforming an image into the transform domain, doing the processing there, and obtaining the inverse transform to bring the results back into the spatial domain.
- Two principal categories of spatial processing are intensity transformations and spatial filtering.
  - Intensity transformations operate on **single pixels** of an image, principally for the purpose of contrast manipulation and image thresholding.
  - Spatial filtering deals with performing operations, such as image sharpening, by working in a **neighborhood** of every pixel in an image

Topics

- Background
- Basic Intensity Transformation Functions
- Histogram Processing
- Spatial Filtering
- Smoothing Spatial Filters
- Sharpening Spatial Filters
- Combining Spatial Enhancement Methods

Intensity Transform

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

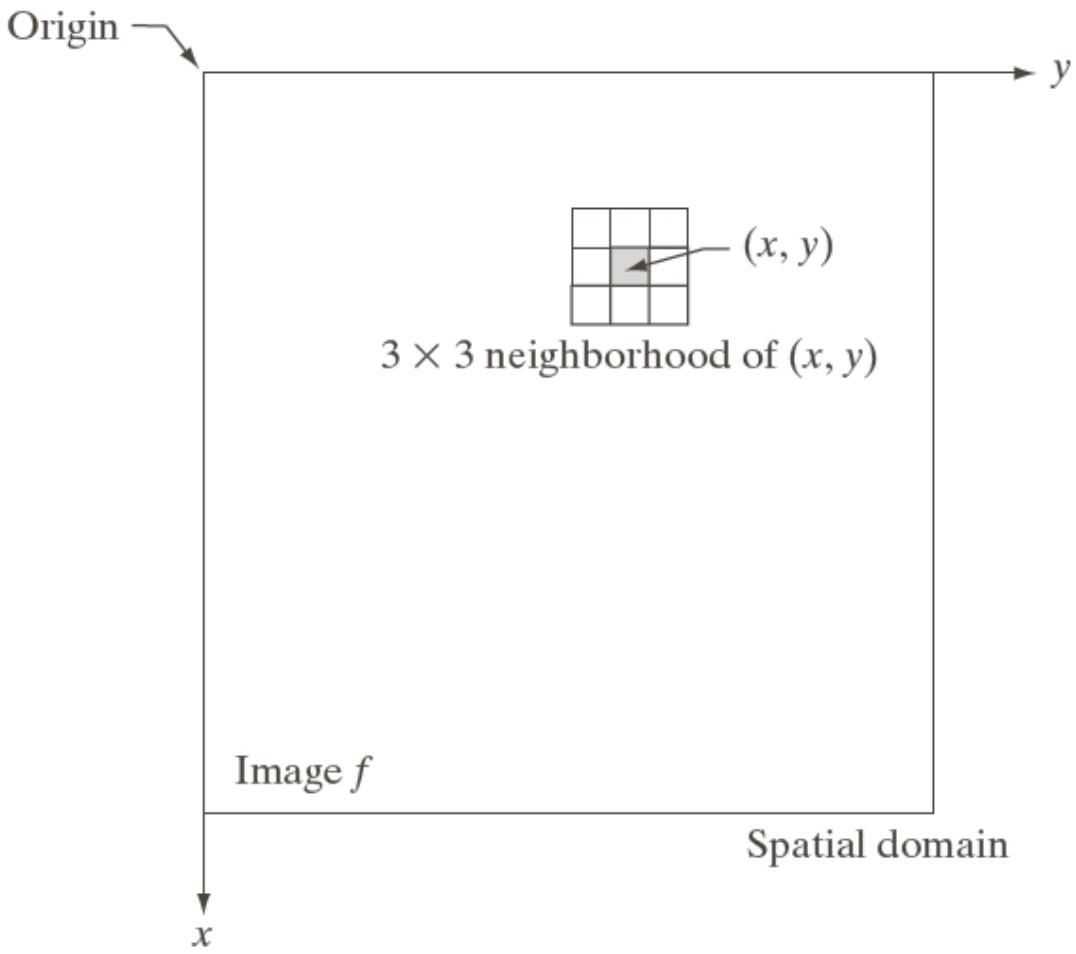
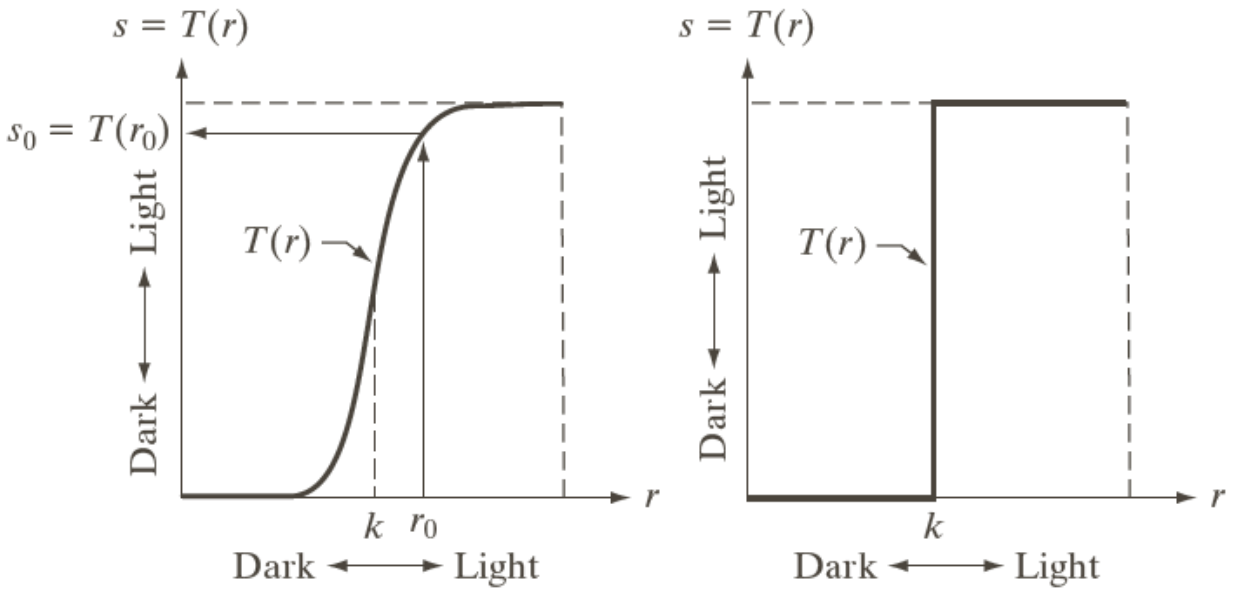


FIGURE 3.1 A 3×3 neighborhood about a point (x, y) in an image in the spatial domain. The neighborhood is moved from pixel to pixel in the image to generate an output image.

Contrast Stretching and Thresholding

$$s = T(r)$$



a b

FIGURE 3.2 Intensity transformation functions. (a) Contrast-stretching function. (b) Thresholding function.

Basic Transformation Functions

- Image Negatives: $s = L - 1 - r$
- Log Transformations: $s = c\ log(1 + r)$
- Power-Law (Gamma) Transformations

$$s = c\ r^{\gamma}$$

- Piecewise-Linear Transformation Functions
  - Contrast Stretching
  - Intensity-Level Slicing
  - Bit-plane Slicing

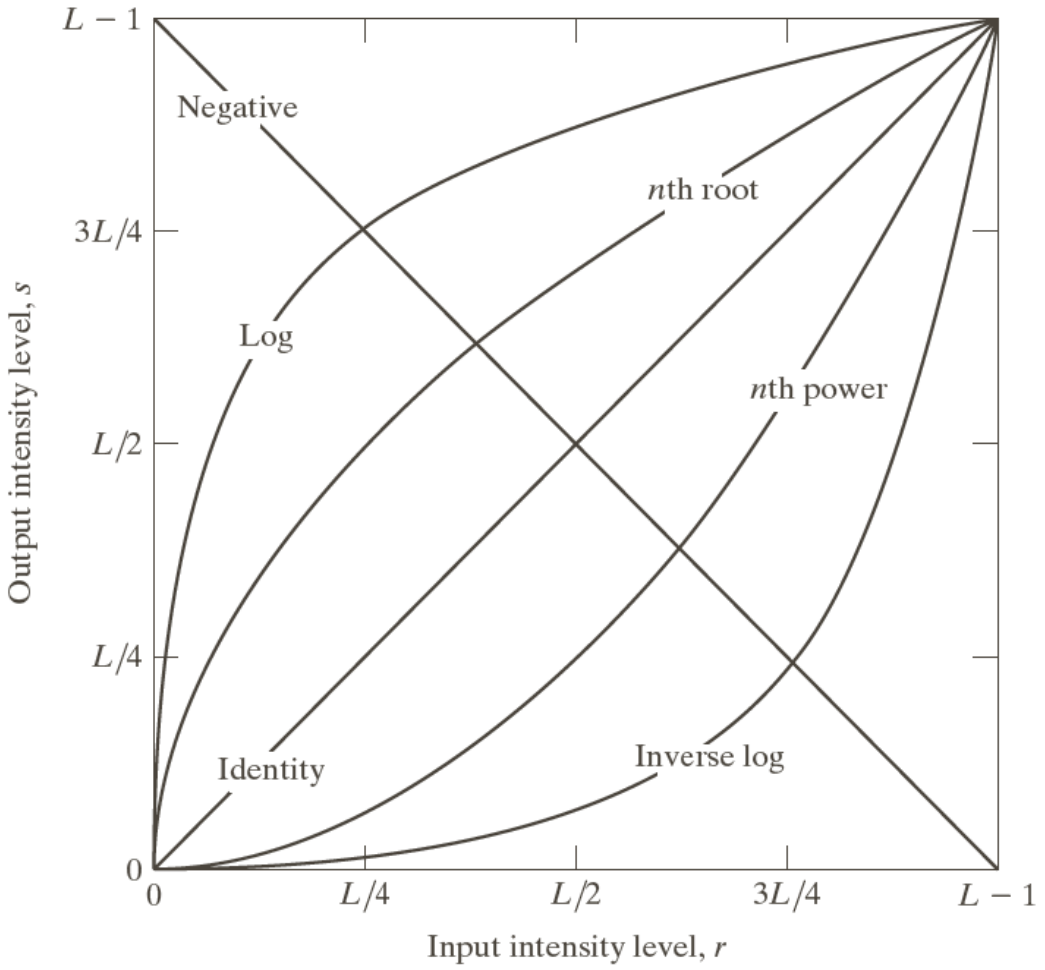


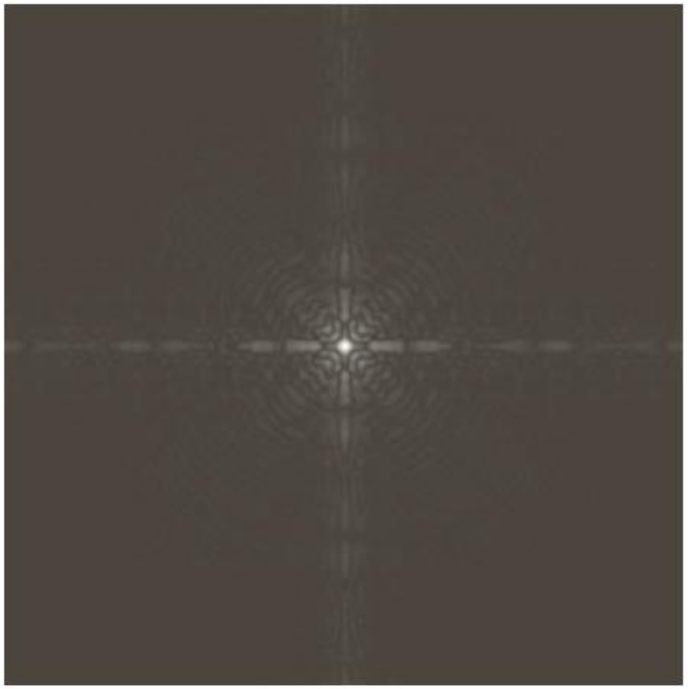
FIGURE 3.3 Some basic intensity transformation functions. All curves were scaled to fit in the range shown.

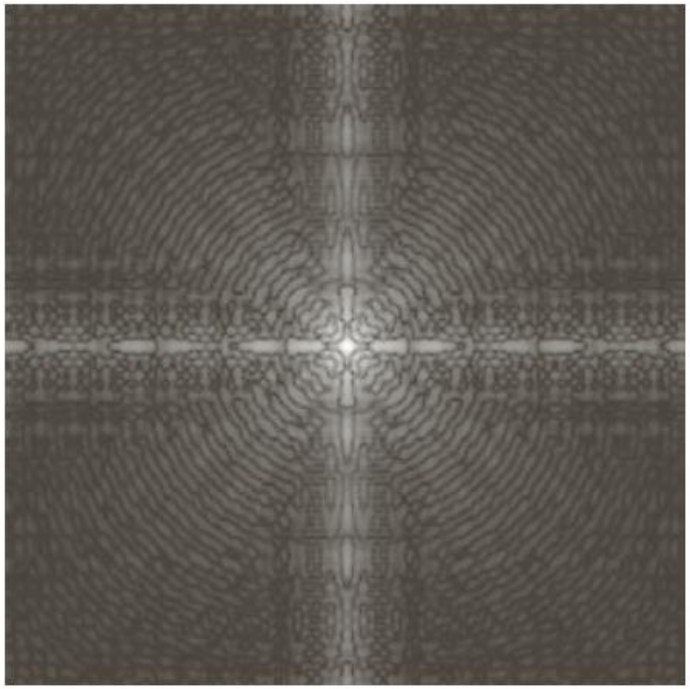
Negatives

a b

FIGURE 3.4
(a) Original digital mammogram. (b) Negative image obtained using the negative transformation in Eq. (3.2-1). (Courtesy of G.E. Medical Systems.)

Log Transformation





a b

FIGURE 3.5
(a) Fourier spectrum.
(b) Result of applying the log transformation in Eq. (3.2-2) with $c = 1$.

Used to expand the values of dark pixels while compressing the higher-level values

Gamma Transformations

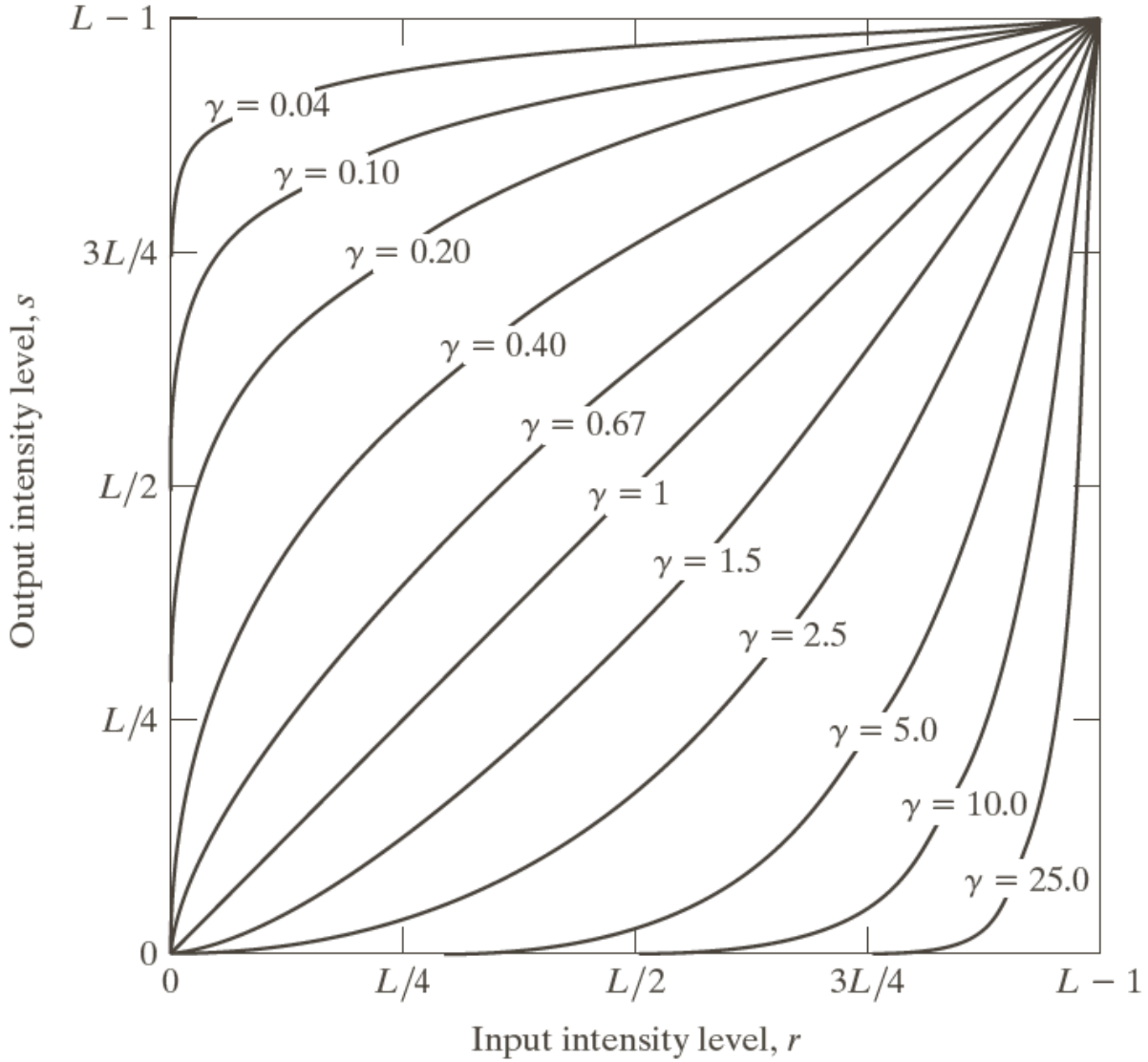
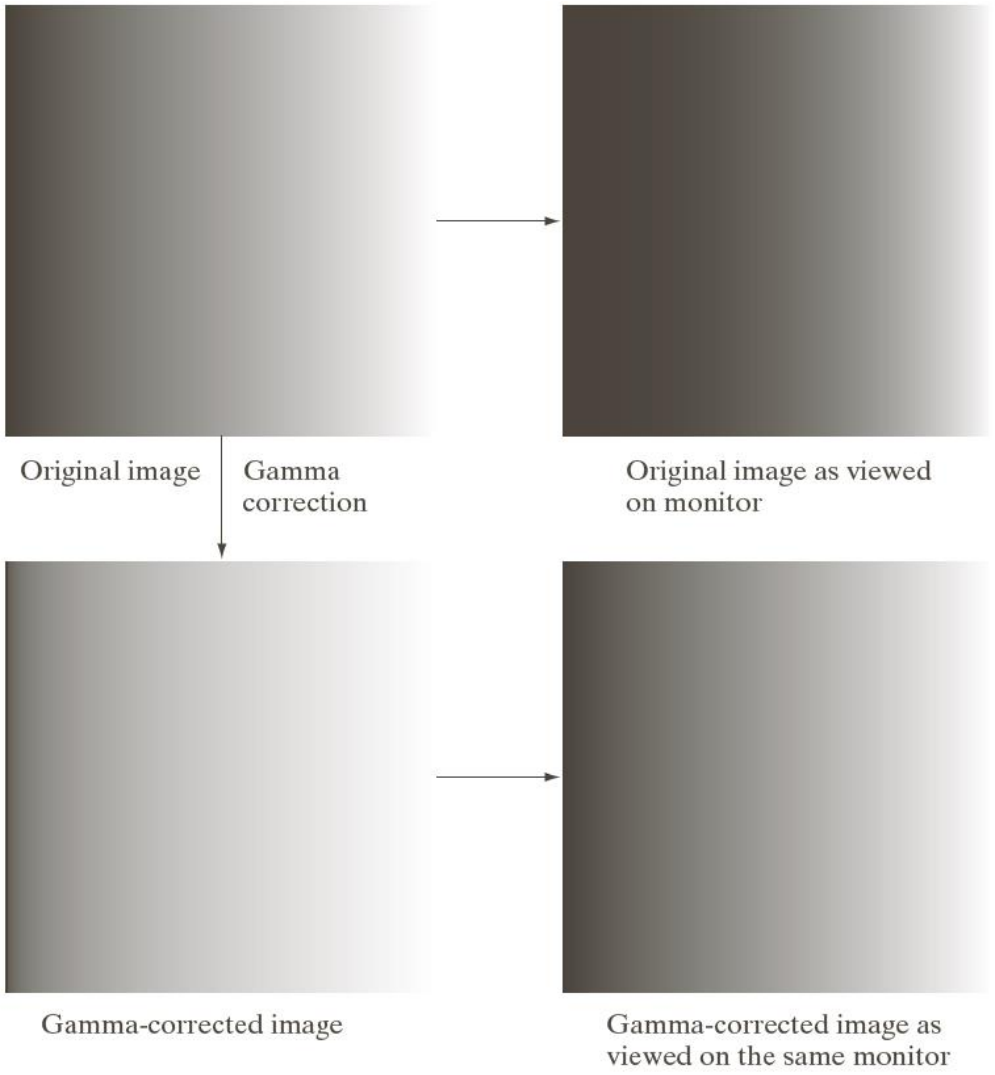


FIGURE 3.6 Plots of the equation $s = cr^{\gamma}$ for various values of γ ($c = 1$ in all cases). All curves were scaled to fit in the range shown.

Gamma Correction

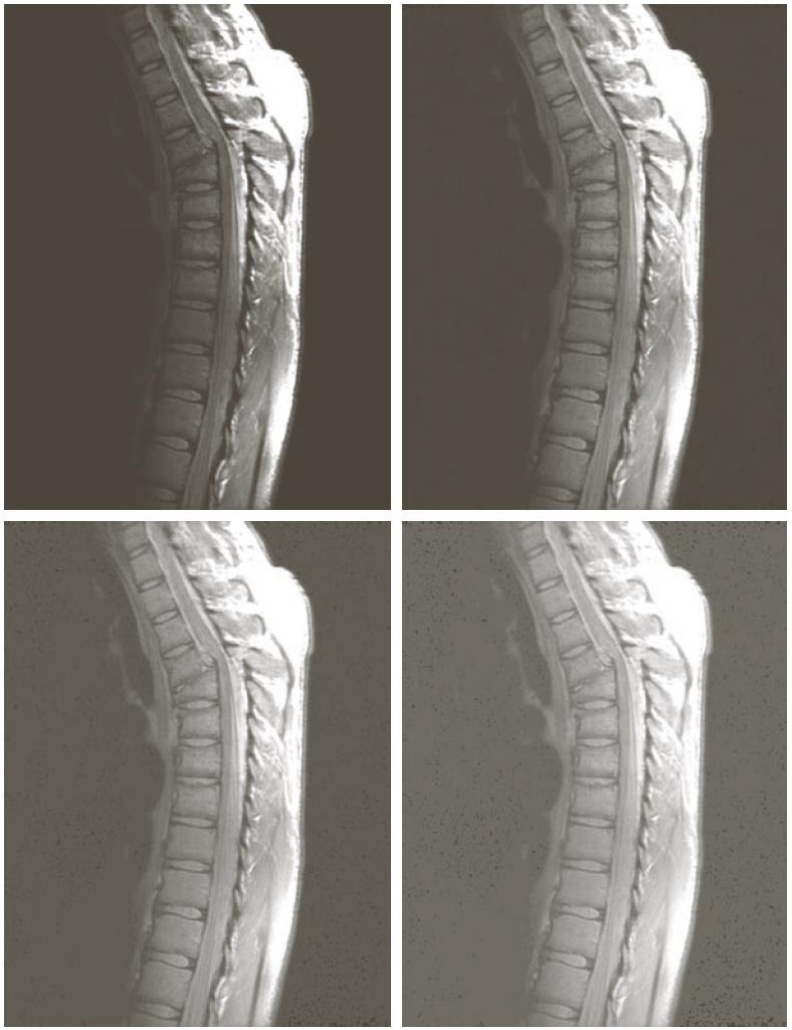


a b
c d

FIGURE 3.7
(a) Intensity ramp image. (b) Image as viewed on a simulated monitor with a gamma of 2.5. (c) Gamma-corrected image. (d) Corrected image as viewed on the same monitor. Compare (d) and (a).

$$s = r^{1/2.5} = r^{0.4}$$

Contrast Manipulation



a b
c d

FIGURE 3.8
(a) Magnetic resonance image (MRI) of a fractured human spine.
(b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 0.6$, 0.4, and 0.3, respectively.
(Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

See Matlab code

Compression of Intensity Levels

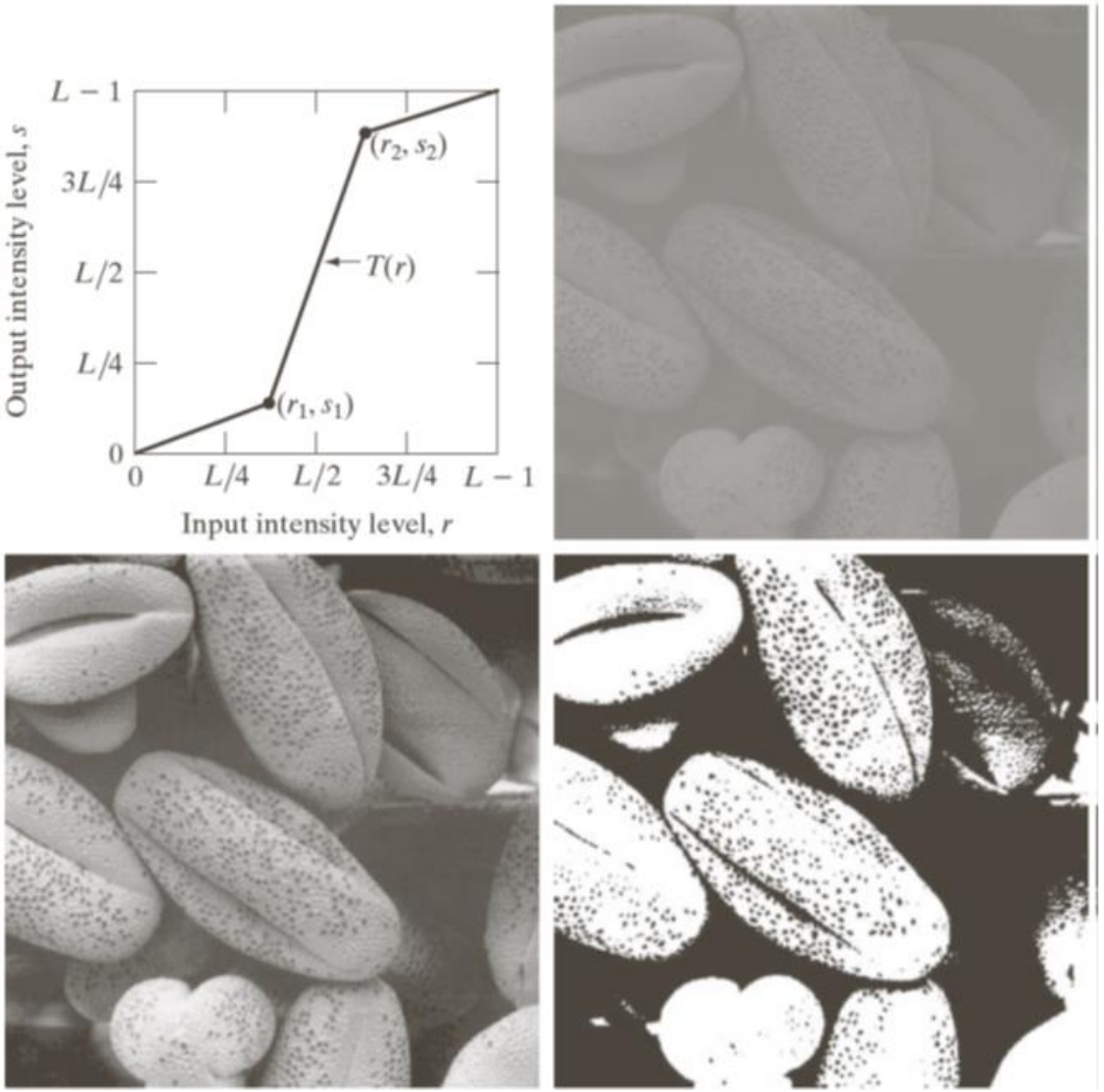
a b
c d

FIGURE 3.9
(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 3.0$, 4.0, and 5.0, respectively. (Original image for this example courtesy of NASA.)

Piecewise-Linear Transformation Functions

- A complementary approach to the methods discussed previously is to use piecewise linear functions.
- The principal advantage of piecewise linear functions is that the form of piecewise functions can be arbitrarily complex.
- In fact, a practical implementation of some important transformations can be formulated only as piecewise functions.
- The principal disadvantage of piecewise functions is that their specification requires considerably more user input.

Contrast Stretching



a b
c d

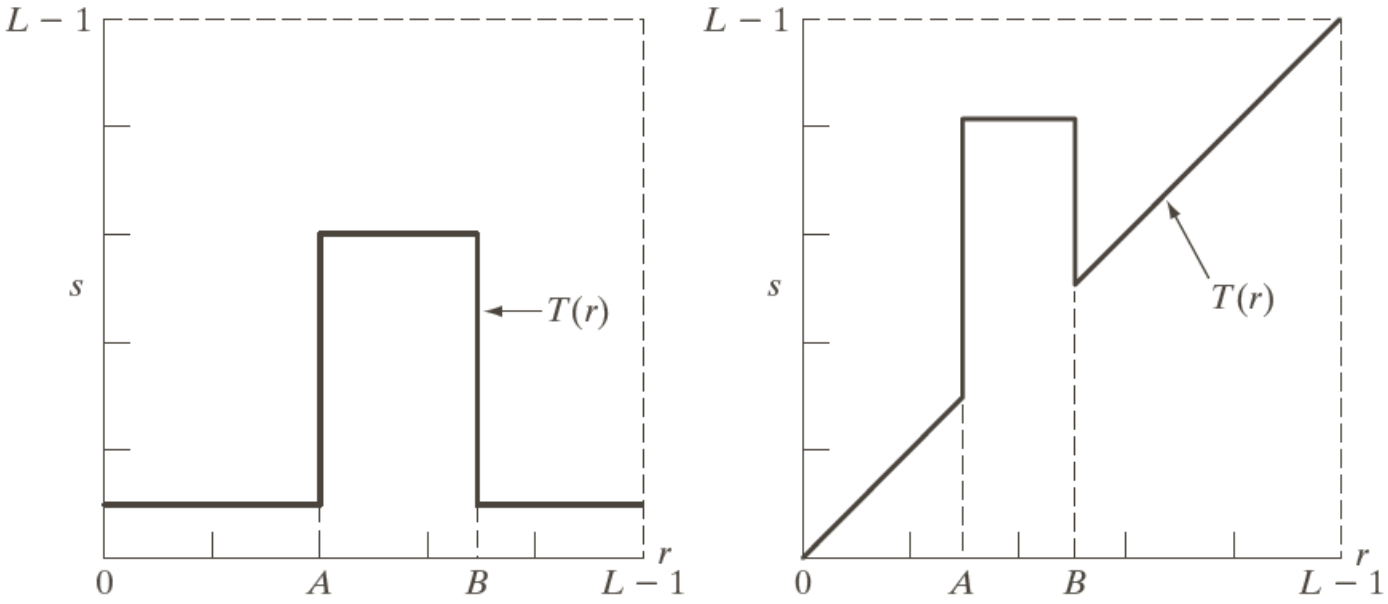
FIGURE 3.10 Contrast stretching. (a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

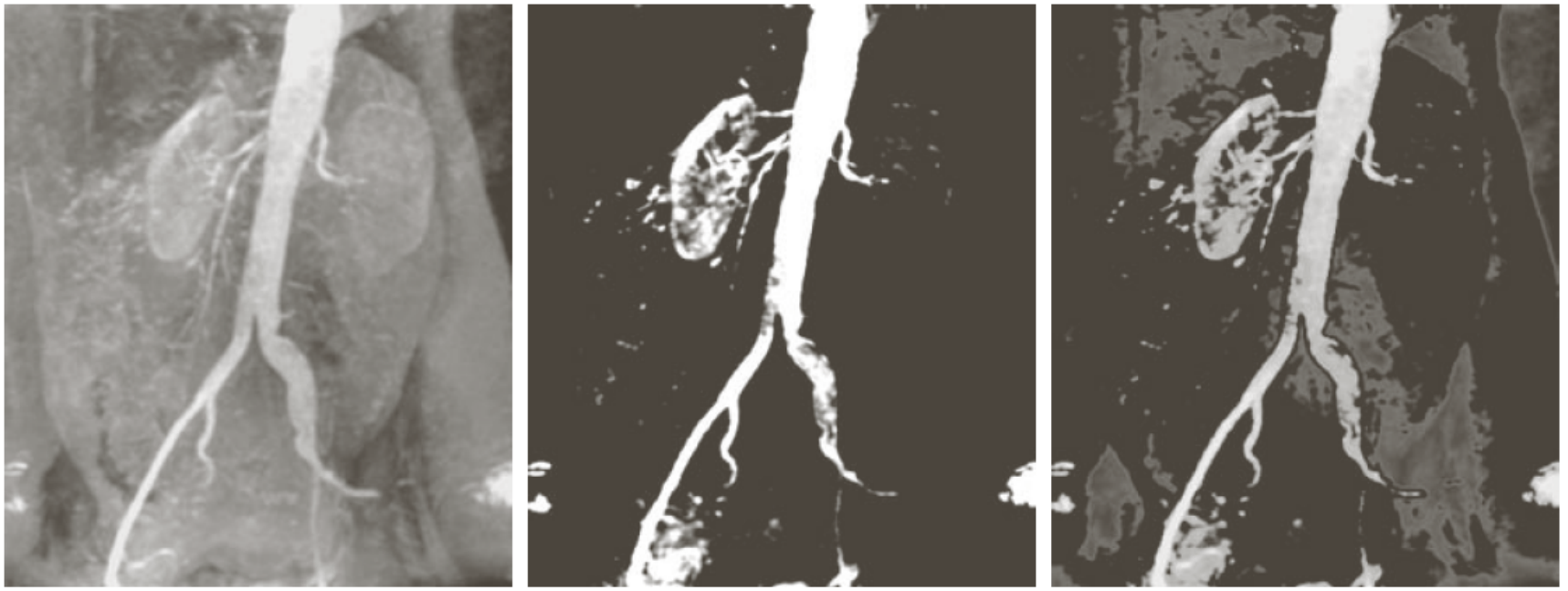
See Matlab code

Intensity-Level Slicing

a b

FIGURE 3.11 (a) This transformation highlights intensity range $[A, B]$ and reduces all other intensities to a lower level. (b) This transformation highlights range $[A, B]$ and preserves all other intensity levels.





a b c

FIGURE 3.12 (a) Aortic angiogram. (b) Result of using a slicing transformation of the type illustrated in Fig. 3.11(a), with the range of intensities of interest selected in the upper end of the gray scale. (c) Result of using the transformation in Fig. 3.11(b), with the selected area set to black, so that grays in the area of the blood vessels and kidneys were preserved. (Original image courtesy of Dr. Thomas R. Gest, University of Michigan Medical School.)

Bit-Plane Slicing

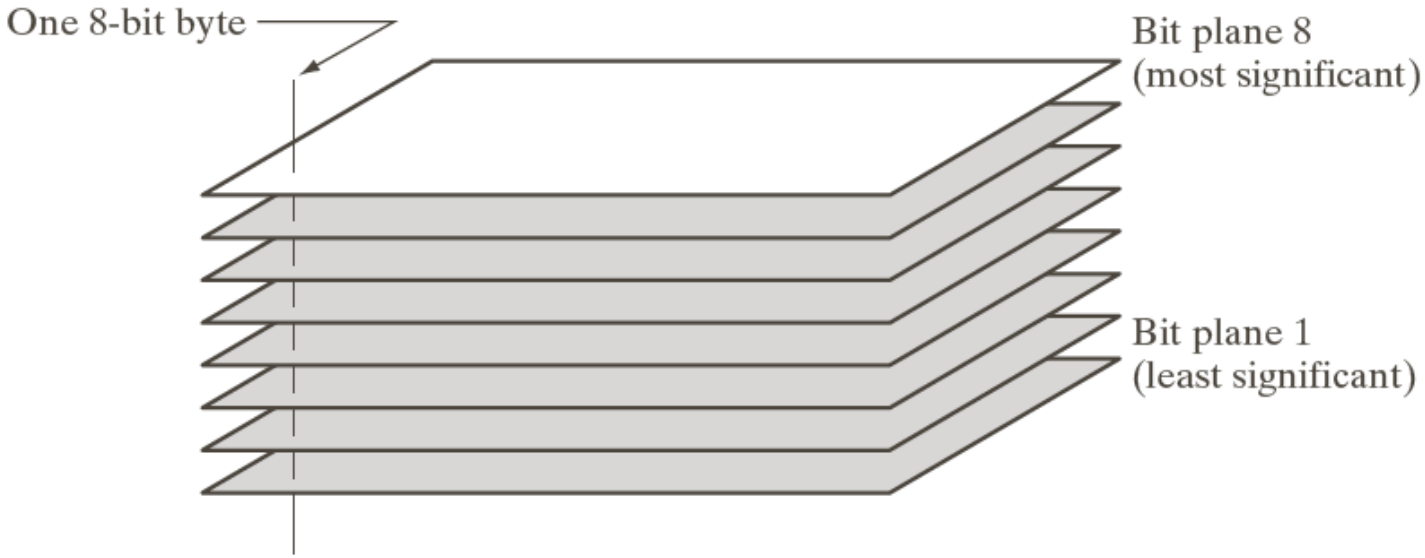


FIGURE 3.13
Bit-plane representation of an 8-bit image.

a b c
d e f
g h i

See Matlab code

FIGURE 3.14 (a) An 8-bit gray-scale image of size 500 × 1192 pixels. (b) through (i) Bit planes 1 through 8, with bit plane 1 corresponding to the least significant bit. Each bit plane is a binary image.

Reconstruction

a b c

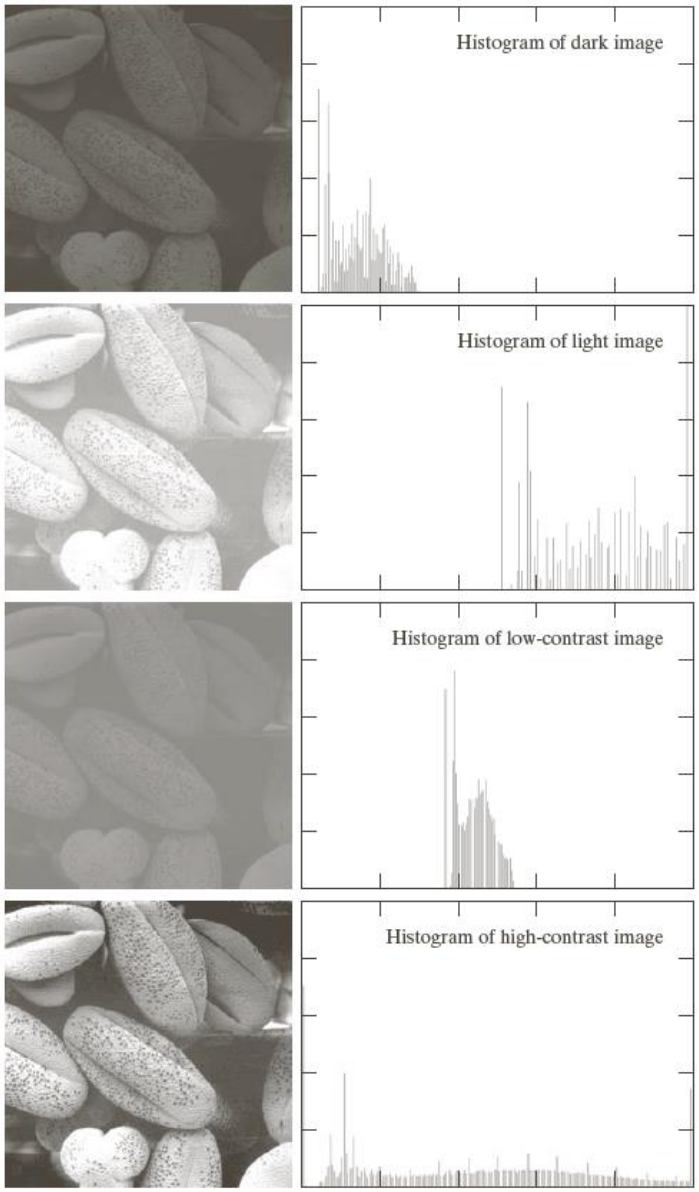
FIGURE 3.15 Images reconstructed using (a) bit planes 8 and 7; (b) bit planes 8, 7, and 6; and (c) bit planes 8, 7, 6, and 5. Compare (c) with Fig. 3.14(a).

Topics

- Background
- Basic Intensity Transformation Functions
- **Histogram Processing**
- Spatial Filtering
- Smoothing Spatial Filters
- Sharpening Spatial Filters
- Combining Spatial Enhancement Methods

Histograms

- Histogram gives an estimate of the probability of occurrence of intensity.
- Histograms are the basis for numerous spatial domain processing techniques
  - Histogram manipulation can be used for image enhancement
  - In addition to providing useful image statistics, the information inherent in histograms also is quite useful in other image processing applications, such as image compression and segmentation.
- Histograms are simple to calculate in software and also lend themselves to economic hardware implementations, thus making them a popular tool for real-time image processing.



Note: imhist ( ) might not show you the whole picture:

The maximum counts displayed is calculated below. This prevents a few large counts from drowning out the rest of the histogram, at the expense of not being able to see the whole plot.

In "imhist.m":

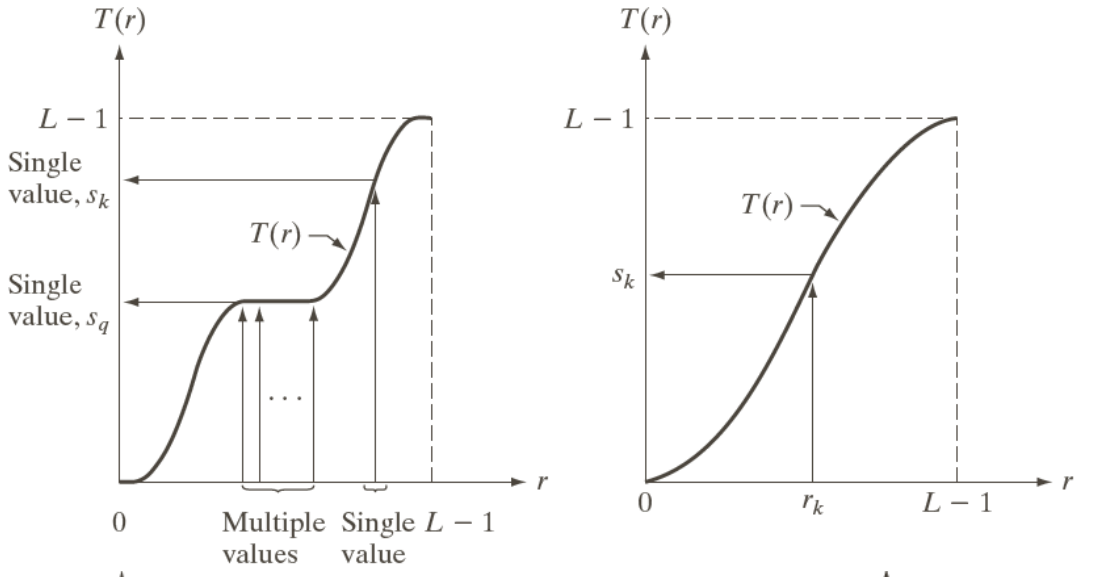
*var = sqrt(y'*y/length(y));*
*limits(4) = 2.5*var;*
axis(hist_axes,limits);

To see the whole picture, use:

```
>> [counts, bins] = imhist(I);
>> stem (bins, counts);
```

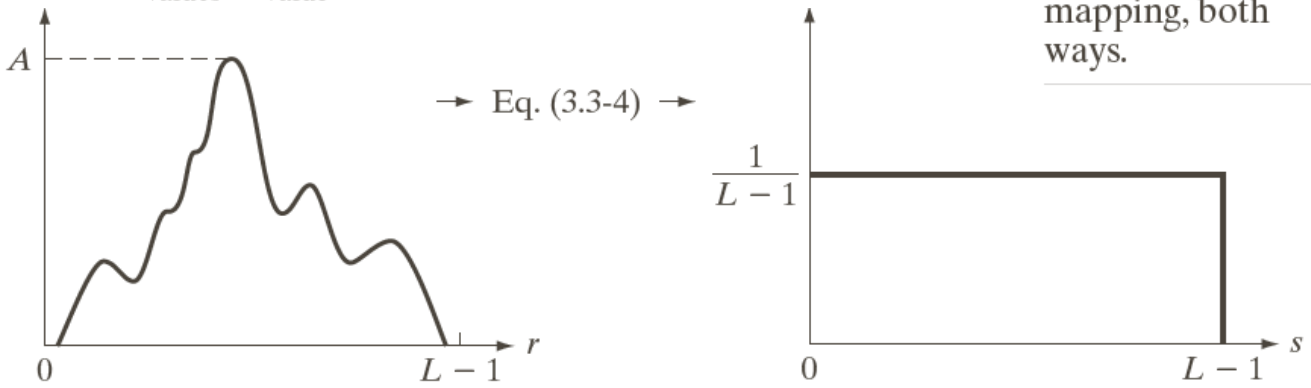
FIGURE 3.16 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms.

Histogram Equalization



a b

FIGURE 3.17 (a) Monotonically increasing function, showing how multiple values can map to a single value. (b) Strictly monotonically increasing function. This is a one-to-one mapping, both ways.



a b

FIGURE 3.18 (a) An arbitrary PDF. (b) Result of applying the transformation in Eq. (3.3-4) to all intensity levels, r. The resulting intensities, s, have a uniform PDF, independently of the form of the PDF of the r's.

Review of Probability Theory

- Random Variables
- Distribution Function
- CDF
- PDF
- Functions of random variables

Continuous vs. Discrete RV

- In probability and statistics, a random variable is a variable whose value is subject to variations due to chance.
- A random variable can take on a set of possible different values, each with an associated probability.
- If a random variable can assume any value within a specified range (possibly infinite), then it will be designated as a continuous random variable.
- Discrete random variables are those assuming one of a countable set of values.

Distribution Functions

- Let X be a random variable and x be any allowed value of this random variable.
- The probability distribution function (also known as Cumulative Distribution Function, or **CDF**) is defined as the probability of the event that the observed random variable X is less than or equal to the allowed value x.

$$F_X(x) = \Pr(X \leq x)$$

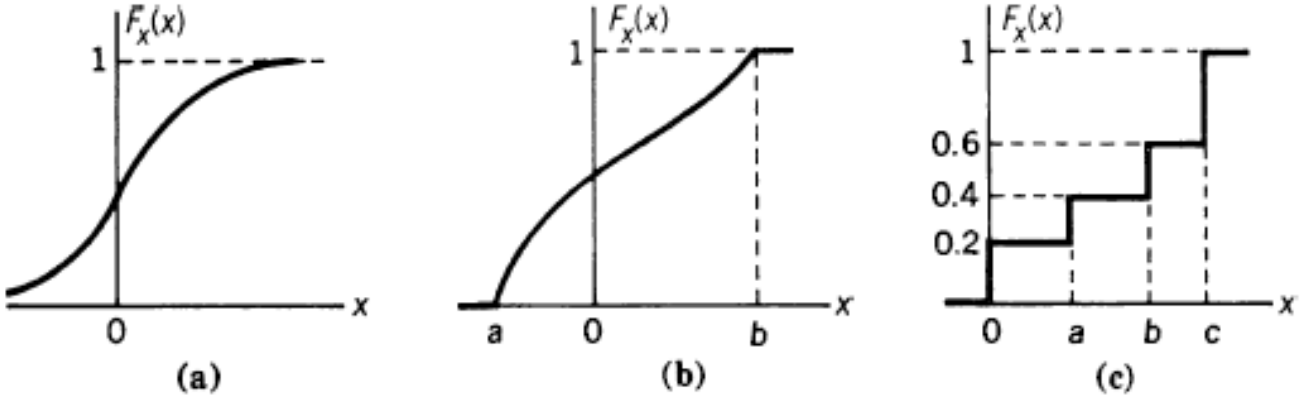
- The subscript X denotes the random variable while the argument x could be any other symbol.
- Sometimes it is convenient to suppress the subscript X when no confusion will result. Thus $F_X(x)$ will often be written as $F(x)$.
- Since the probability distribution function is a probability, it must satisfy the basic axioms.

Properties of $F_X(x)$

$$F_X(x) = \Pr(X \leq x)$$

- $0 \leq F_X(x) \leq 1, \quad -\infty < x < \infty$
- $F_X(-\infty) = 0, \quad F_X(\infty) = 1$
- $F_X(x)$ is non-decreasing as x increases.
- $\Pr(x_1 < X \leq x_2) = F_X(x_2) - F_X(x_1)$.
- $\Pr(X > x) = 1 - F_X(x)$.

- Examples:



Example Distribution

- A particular random variable has a probability distribution function given by

$$F_X(x) = \begin{cases} 0 & -\infty < x \leq 0 \\ 1 - e^{-2x} & 0 \leq x < \infty \end{cases}$$

- What is the probability that $X > 0.5$?

$$\Pr(X > 0.5) = 1 - F_X(0.5) = 1 - (1 - e^{-2*0.5}) = 0.3679$$

- What is the probability that $0.5 < X \leq 0.51$?

$$\Pr(0.5 < X \leq 0.51) = F_X(0.51) - F_X(0.5) = ?$$

Density Functions

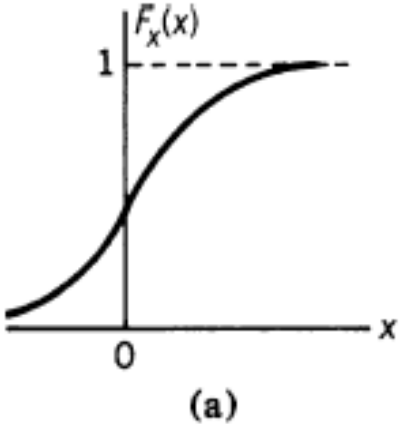
- Although the distribution function is a complete description of the probability model for a single random variable, it is not convenient for many calculations of interest.
- It may be preferable to use the derivative of $F_X(x)$ rather than $F_X(x)$ itself. This derivative is called the probability density function (**PDF**). When it exists, it is defined by

$$f_X(x) = \lim_{e \to 0} \frac{F_X(x+e) - F_X(x)}{e} = \frac{dF_X(x)}{dx}$$

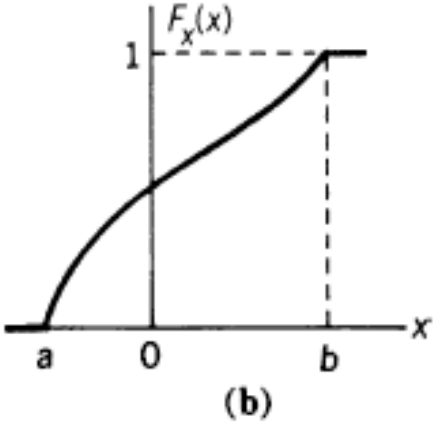
Properties of PDF

- The physical significance of the PDF is best described in terms of the probability element, $f_X(x)dx = \Pr(x < X \leq x + dx)$.
- Properties
  1. $f_X(x) \geq 0,\ -\infty < x < \infty$
  2. $\int_{-\infty}^{\infty} f_X(x)dx = 1$
  3. $F_X(x) = \int_{-\infty}^{x} f_X(u)du$
  4. $\int_{x_1}^{x_2} f_X(x)dx = \Pr(x_1 < X \leq x_2)$

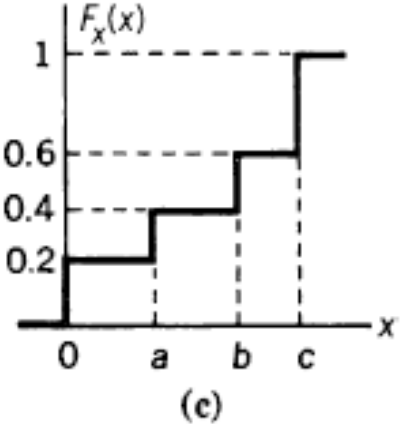
Examples of CDF's and PDF's



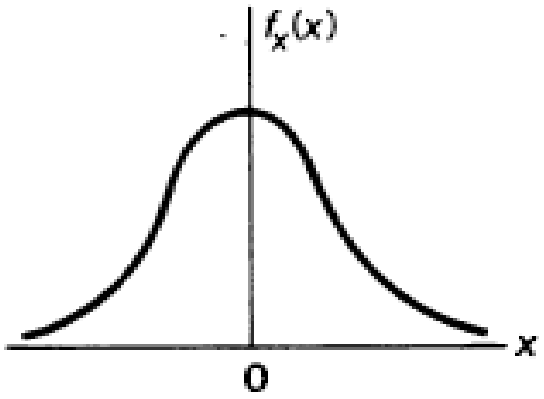
(a)



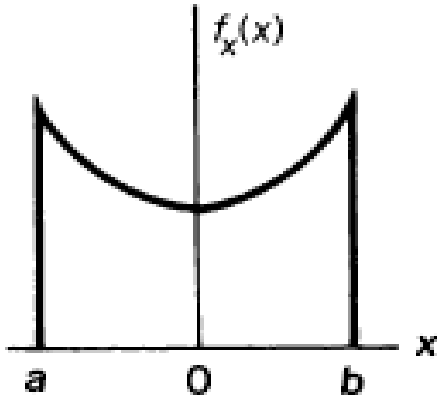
(b)



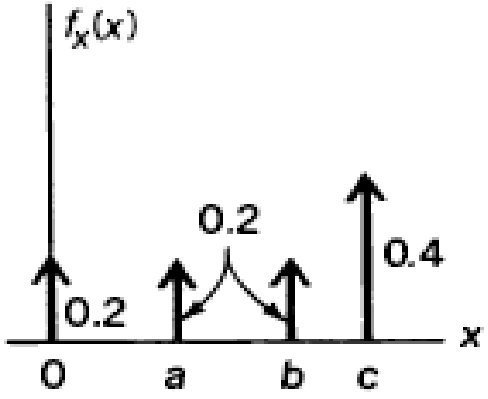
(c)



(a)



(b)



(c)

Another Example

- $Fx(X) = \begin{cases} 0 & -\infty < x \leq 0 \\ 1 - e^{-2x} & 0 \leq x < \infty \end{cases}$
- PDF:

$$f_X(x) = \frac{dFx(X)}{dx} = \begin{cases} 0 & -\infty < x \leq 0 \\ 2e^{-2x} & 0 \leq x < \infty \end{cases}$$

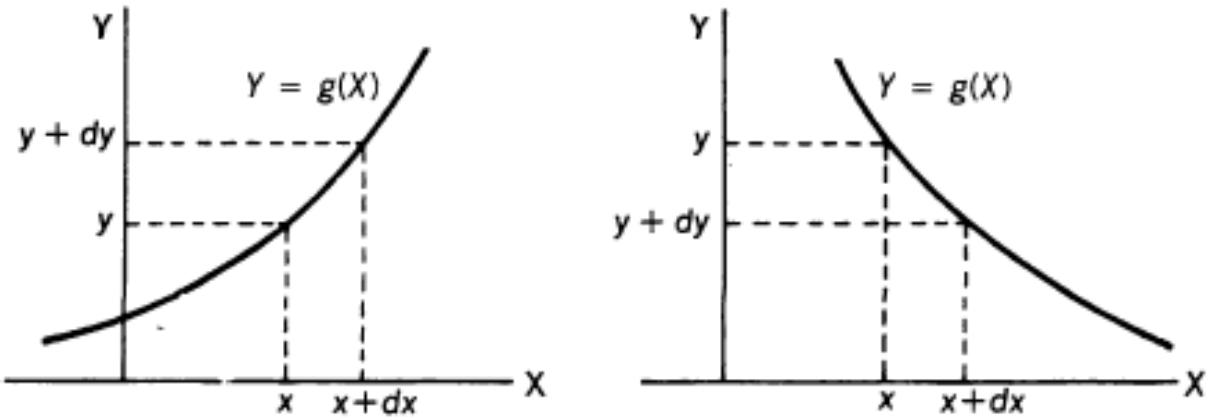
How about the following properties?

1. $f_X(x) \geq 0, \ -\infty < x < \infty$
2. $\int_{-\infty}^{\infty} f_X(x)dx = 1$

$$\int_0^{\infty} 2e^{-2x}dx = -e^{-2x}\Big|_0^{\infty} = ?$$

$Y = g(X)$

- Given the PDF of X is known as $f_X(x)$, find the PDF of Y, which is denoted by $f_Y(y)$.
- It is clear that whenever the random variable X lies between x and $x + dx$, the random variable Y will lie between y and $y + dy$.
- Since the probabilities of these events are $f_X(x)dx$ and $f_Y(y)dy$, $f_X(x)dx = f_Y(y)dy$.
- Therefore, $f_Y(y) = f_X(x)\frac{dx}{dy}$.
- In general, , $f_Y(y) = f_X(x)\left|\frac{dx}{dy}\right|$



Example

$$f_X(x) = e^{-x}u(x)$$

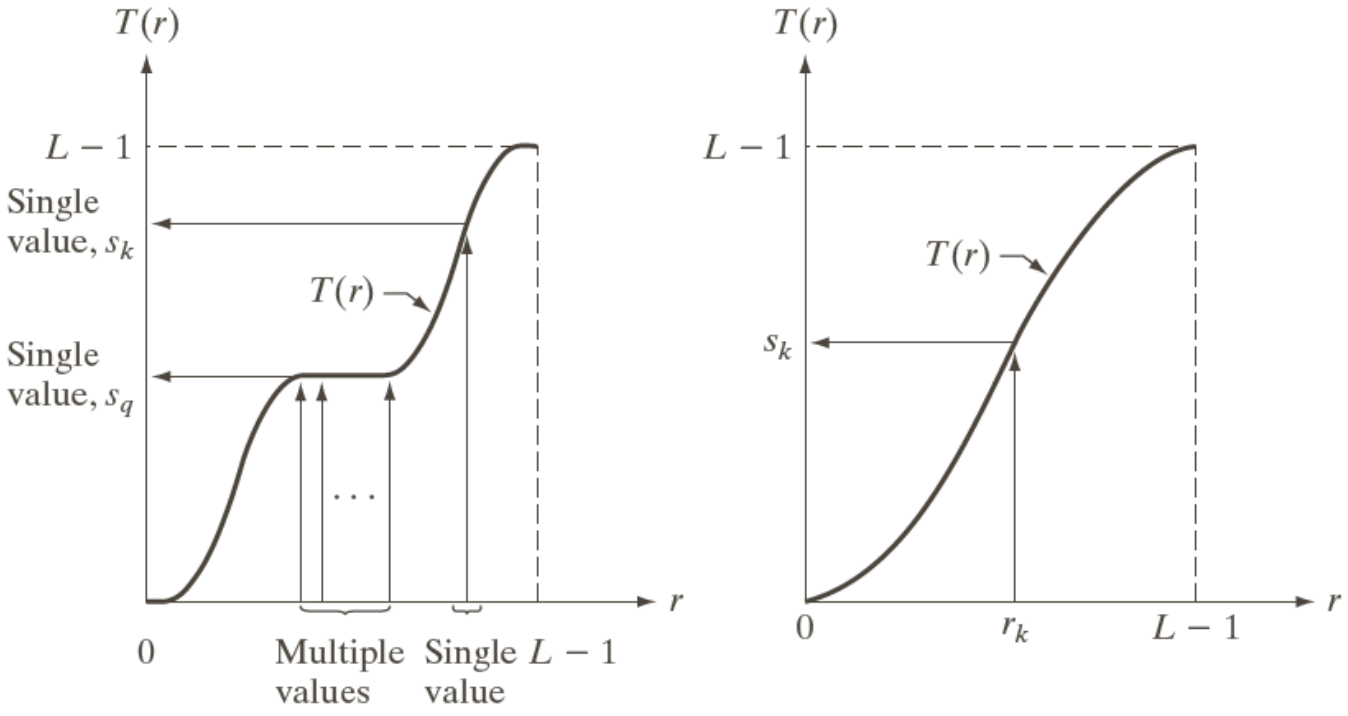
$$Y = X^3$$

$$\frac{dy}{dx} = 3x^2$$

$$\frac{dx}{dy} = \frac{1}{3x^2} = \frac{1}{3y^{2/3}}$$

$$f_Y(y) = \frac{e^{-y^{1/3}}}{3} y^{-2/3} u(y)$$

Returning to Intensity Transform



a b

FIGURE 3.17 (a) Monotonically increasing function, showing how multiple values can map to a single value. (b) Strictly monotonically increasing function. This is a one-to-one mapping, both ways.

$$s = T(r), \qquad 0 \le r \le L - 1$$

$$p_s(s) = p_r(r) \left| \frac{dr}{ds} \right|$$

Continuous Intensity Values

$$s = T(r) = (L-1)\int_0^r p_r(\omega)d\omega$$

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$

$$\frac{ds}{dr} = \frac{dT(r)}{dr}$$

$$= (L-1)\frac{d}{dr}\left[\int_0^r p_r(w)\,dw\right]$$

$$= (L-1)p_r(r)$$

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$

$$= p_r(r)\left|\frac{1}{(L-1)p_r(r)}\right|$$

$$= \frac{1}{L-1} \qquad 0 \le s \le L-1$$

Histogram Equalization

Example

$$p_r(r) = \begin{cases} \dfrac{2r}{(L-1)^2} & \text{for } 0 \le r \le L-1 \\ 0 & \text{otherwise} \end{cases}$$

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw = \frac{2}{L-1}\int_0^r w\,dw = \frac{r^2}{L-1}$$

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right| = \frac{2r}{(L-1)^2}\left|\left[\frac{ds}{dr}\right]^{-1}\right|$$

$$= \frac{2r}{(L-1)^2}\left|\left[\frac{d}{dr}\frac{r^2}{L-1}\right]^{-1}\right|$$

$$= \frac{2r}{(L-1)^2}\left|\frac{(L-1)}{2r}\right| = \frac{1}{L-1}$$

Discrete Intensity Values

$$p_r(r_k) = \frac{n_k}{MN} \qquad k = 0, 1, 2, \ldots, L - 1$$

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

$$= \frac{(L - 1)}{MN}\sum_{j=0}^{k} n_j \qquad k = 0, 1, 2, \ldots, L - 1$$

Illustration

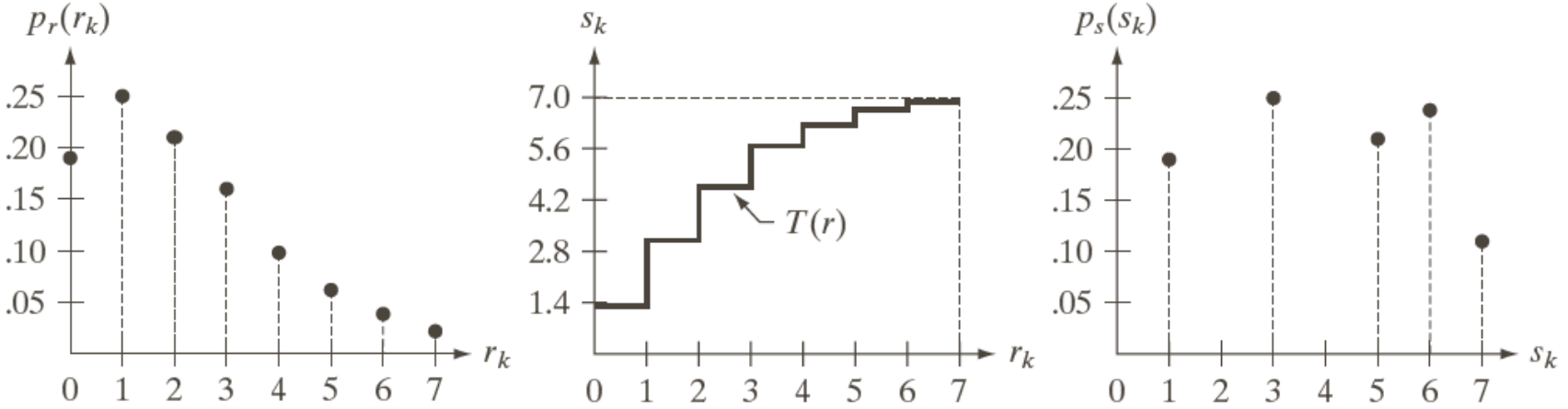
r_k	n_k	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02

TABLE 3.1
Intensity distribution and histogram values for a 3-bit, 64×64 digital image.

$$s_0 = T(r_0) = 7\sum_{j=0}^{0} p_r(r_j) = 7p_r(r_0) = 1.33$$

$$s_1 = T(r_1) = 7\sum_{j=0}^{1} p_r(r_j) = 7p_r(r_0) + 7p_r(r_1) = 3.08$$

$$s_2 = 4.55,\ s_3 = 5.67,\ s_4 = 6.23,\ s_5 = 6.65,\ s_6 = 6.86,\ s_7 = 7.00.$$



a b c

See Matlab code

FIGURE 3.19 Illustration of histogram equalization of a 3-bit (8 intensity levels) image. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

$$s_0 = 1.33 \rightarrow 1 \qquad s_4 = 6.23 \rightarrow 6$$

$$s_1 = 3.08 \rightarrow 3 \qquad s_5 = 6.65 \rightarrow 7$$

$$s_2 = 4.55 \rightarrow 5 \qquad s_6 = 6.86 \rightarrow 7$$

$$s_3 = 5.67 \rightarrow 6 \qquad s_7 = 7.00 \rightarrow 7$$

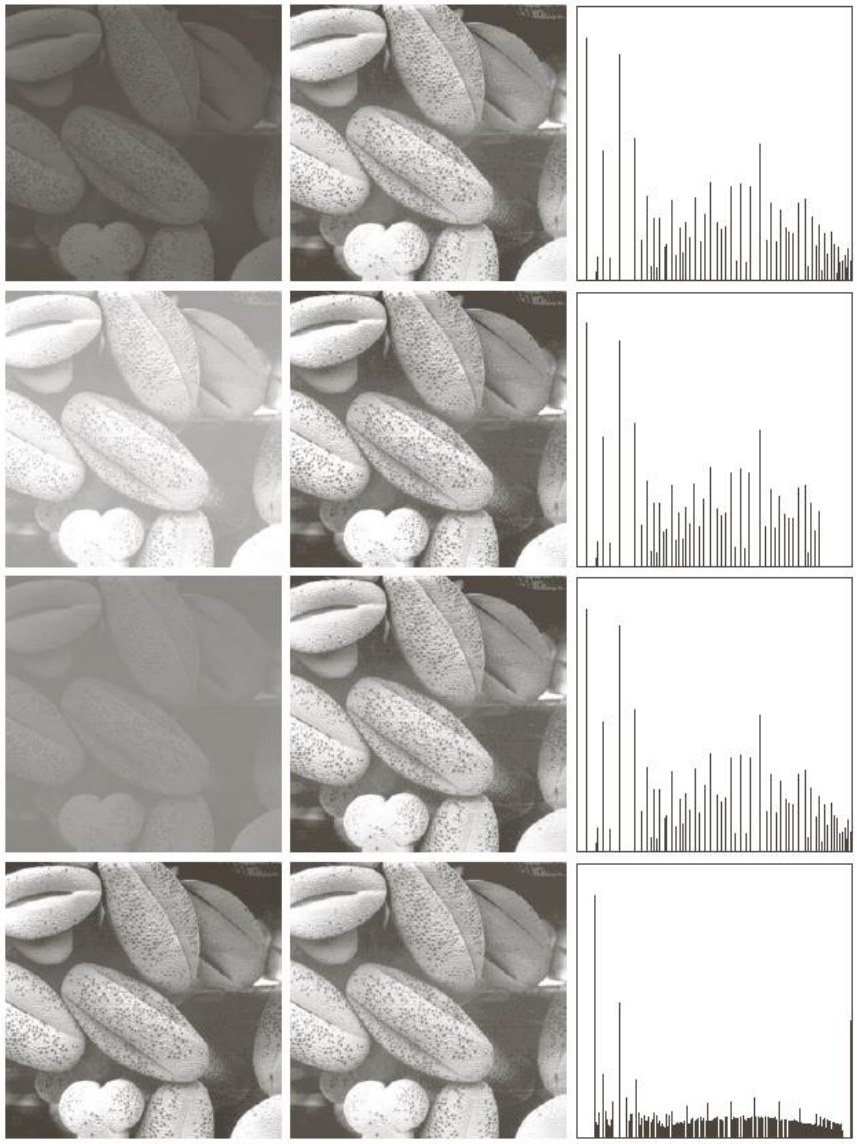


FIGURE 3.20 Left column: images from Fig. 3.16. Center column: corresponding histogram-equalized images. Right column: histograms of the images in the center column.

histeq

```
>> I = imread('Fig0320(2)(2nd_from_top).tif');
>> imshow(I)
>> figure; imhist(I)
>> J = histeq(I);
>> figure;
>> imshow(J)
>> figure; imhist(J)
```

Topics

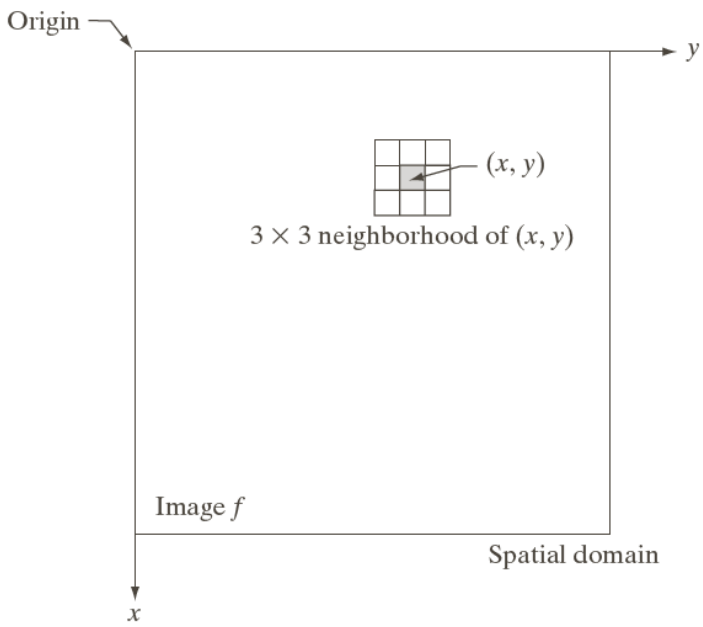
- Background
- Basic Intensity Transformation Functions
- Histogram Processing
- **Spatial Filtering**
- Smoothing Spatial Filters
- Sharpening Spatial Filters
- Combining Spatial Enhancement Methods

Fundamentals of Spatial Filtering

- The name “filter” is borrowed from frequency domain processing, which is the topic of the next chapter, where "filtering" refers to accepting (passing) or rejecting certain frequency components.
- For example, a filter that passes low frequencies is called a lowpass filter. The net effect produced by a lowpass filter is to blur (smooth) an image.
- We can accomplish a similar smoothing directly on the image itself by using spatial filters (also called spatial masks, kernels, templates, and windows).
- Spatial filters offer considerably more versatility because they can be used also for nonlinear filtering, which we cannot do in the frequency domain.

Mechanics of Spatial Filtering

- a spatial filter consists of
  - A neighborhood, (typically a small rectangle), and
  - a predefined operation that is performed on the image pixels encompassed by the neighborhood.
- Filtering creates a new pixel with coordinates equal to the coordinates of the center of the neighborhood, and whose value is the result of the filtering operation.
- A processed (filtered) image is generated as the center of the filter visits each pixel in the input image.
- If the operation performed on the image pixels is linear, then the filter is called a linear spatial filter. Otherwise, the filter is nonlinear.



$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t)$$

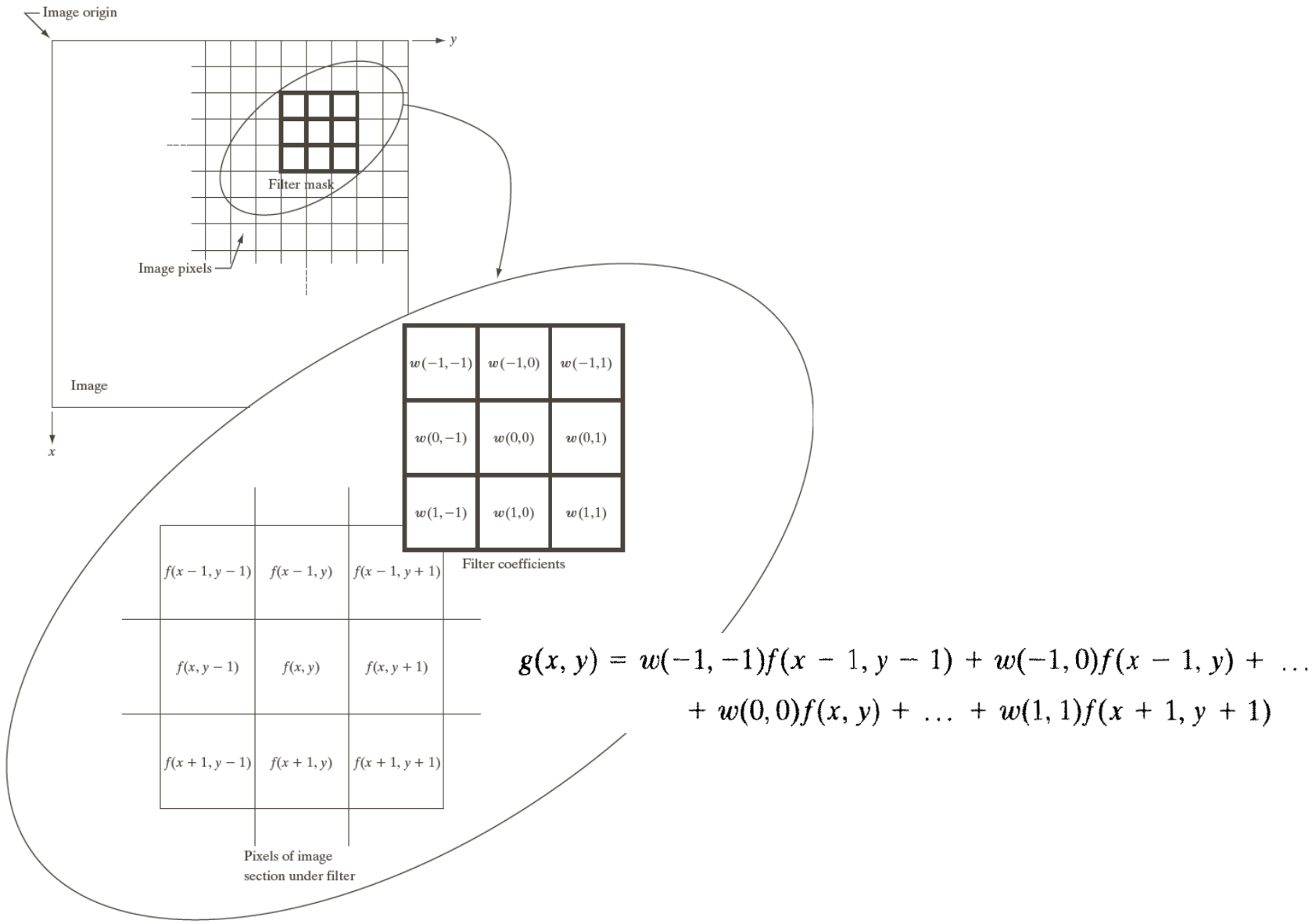


FIGURE 3.28 The mechanics of linear spatial filtering using a 3×3 filter mask. The form chosen to denote the coordinates of the filter mask coefficients simplifies writing expressions for linear filtering.

Spatial Correlation and Convolution

- There are two closely related concepts that must be understood when performing linear spatial filtering, One is *correlation* and the other is *convolution.*
- Correlation is the process of moving a filter mask over the image and computing the sum of products at each location.

$$w(x, y) \star f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t)$$

- The mechanics of convolution are the same, except that the filter is first rotated by 180°.

$$w(x, y) \star f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x - s, y - t)$$

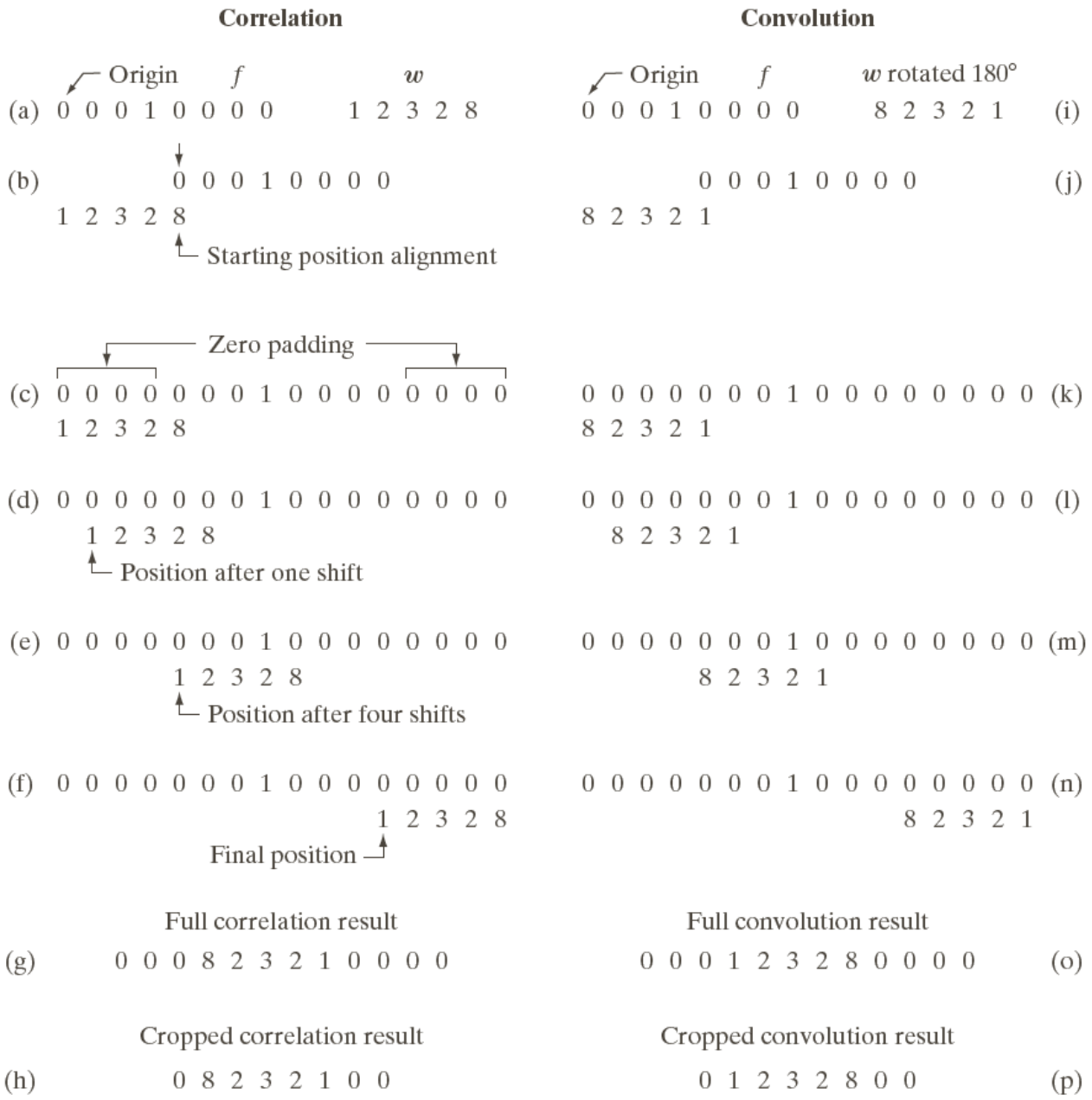


FIGURE 3.29 Illustration of 1-D correlation and convolution of a filter with a discrete unit impulse. Note that correlation and convolution are functions of *displacement*.

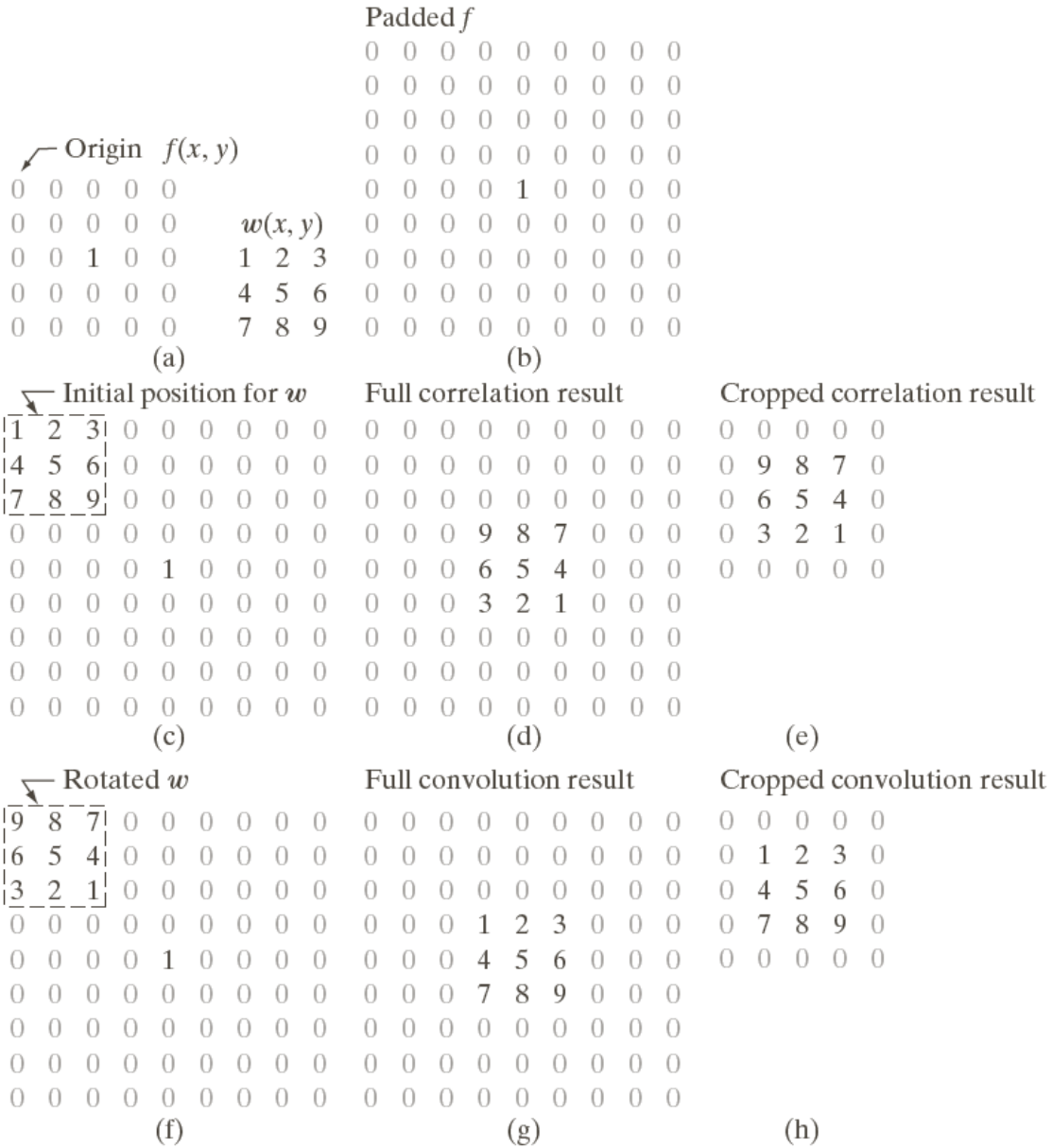


FIGURE 3.30 Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.

Vector-Representation of Linear Filtering

When interest lies in the characteristic response, R, of a mask either for correlation or convolution, it is convenient sometimes to write the sum of products as

$$R = w_1 z_1 + w_2 z_2 + \ldots + w_{mn} z_{mn}$$

$$= \sum_{k=1}^{mn} w_k z_k$$

$$= \mathbf{w}^T \mathbf{z}$$

$$R = w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9$$

$$= \sum_{k=1}^{9} w_k z_k$$

$$= \mathbf{w}^T \mathbf{z}$$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

FIGURE 3.31 Another representation of a general 3×3 filter mask.

Generating Spatial Filter Masks

- Generating an $m \times n$ linear spatial filter requires that we specify mn mask coefficients.
- In turn, these coefficients are selected based on what the filter is supposed to do – all we can do with linear filtering is to implement a sum of products.
- Generating a nonlinear filter requires that we specify the size of a neighborhood and the operation(s) to be performed on the image pixels contained in the neighborhood.
  - For example, a 5×5 max filter

Topics

- Background
- Basic Intensity Transformation Functions
- Histogram Processing
- Spatial Filtering
- **Smoothing Spatial Filters**
- Sharpening Spatial Filters
- Combining Spatial Enhancement Methods

Smoothing Spatial Filters

- Smoothing filters are used for blurring and for noise reduction.
  - Blurring is used in preprocessing tasks, such as removal of small details from an image prior to (large) object extraction, and bridging of small gaps in lines or curves.
  - Noise reduction can be accomplished by blurring with a linear filter and also by nonlinear filtering.
- The output (response) of a smoothing, linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask. These filters sometimes are called *averaging filters.*
- Order-statistic filters arc nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result. The best-known filter in this category is the median filler.

Linear Filters

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

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 3.32 Two 3×3 smoothing (averaging) filter masks. The constant multiplier in front of each mask is equal to 1 divided by the sum of the values of its coefficients, as is required to compute an average.

Matlab Filtering Functions

- MATLAB has several two-dimensional and multidimensional filtering functions.
- The function **filter2** performs two-dimensional correlation.
- **conv2** performs twodimensional convolution, and convn performs multidimensional convolution.
- Each of these filtering functions always converts the input to double, and the output is always double.
- The above filtering functions always assume the input is zero padded, and they do not support other padding options.
- In contrast, the **imfilter** function does not convert input images to double.
- The **imfilter** function also offers a flexible set of boundary padding options.

Imfilter (I, w, corr options, boundary options, size_options)

Boundary Options

Option	Description
Boundary Options	
X	Input array values outside the bounds of the array are implicitly assumed to have the value X. When no boundary option is specified, the default is 0.
`'symmetric'`	Input array values outside the bounds of the array are computed by mirror-reflecting the array across the array border.
`'replicate'`	Input array values outside the bounds of the array are assumed to equal the nearest array border value.
`'circular'`	Input array values outside the bounds of the array are computed by implicitly assuming the input array is periodic.
Output Size	
`'same'`	The output array is the same size as the input array. This is the default behavior when no output size options are specified.
`'full'`	The output array is the full filtered result, and so is larger than the input array.
Correlation and Convolution Options	
`'corr'`	`imfilter` performs multidimensional filtering using correlation, which is the same way that `filter2` performs filtering. When no correlation or convolution option is specified, `imfilter` uses correlation.
`'conv'`	`imfilter` performs multidimensional filtering using convolution.

padarray

```
>> A = [1 2; 3 4]

>> padarray(A, [2 2])
```

0	0	0	0	0	0
0	0	0	0	0	0
0	0	**1**	**2**	0	0
0	0	**3**	**4**	0	0
0	0	0	0	0	0
0	0	0	0	0	0

```
>> padarray(A, [2 2],'replicate')
```

1	1	1	2	2	2
1	1	1	2	2	2
1	1	**1**	**2**	2	2
3	3	**3**	**4**	4	4
3	3	3	4	4	4
3	3	3	4	4	4

```
>> A = [1 2; 3 4]
```

```
>> padarray(A, [2 2], 'symmetric')
```

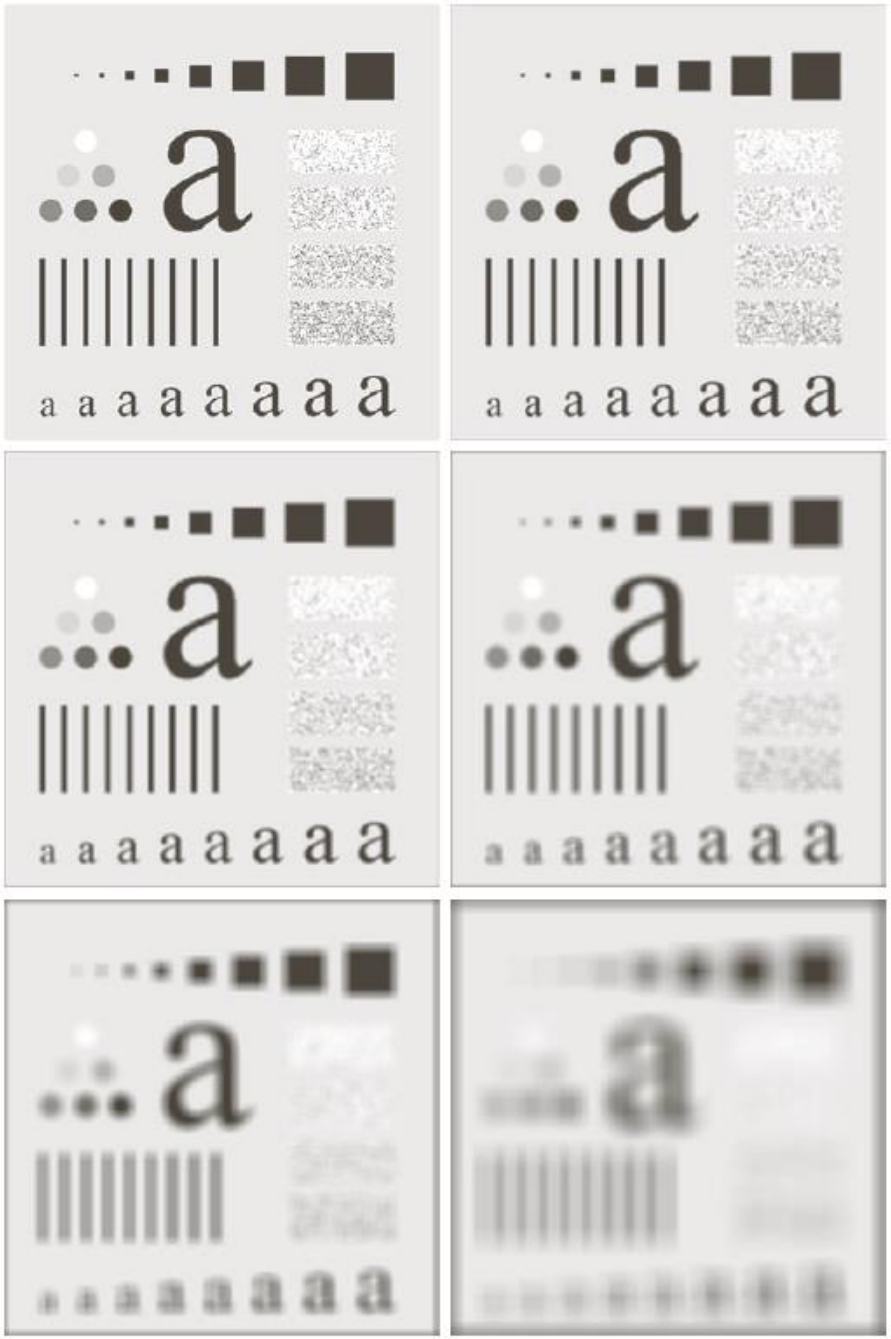
4	3	3	4	4	3
2	1	1	2	2	1
2	1	**1**	**2**	2	1
4	3	**3**	**4**	4	3
4	3	3	4	4	3
2	1	1	2	2	1

```
>> padarray(A, [2 2], 'circular')
```

1	2	1	2	1	2
3	4	3	4	3	4
1	2	**1**	**2**	1	2
3	4	**3**	**4**	3	4
1	2	1	2	1	2
3	4	3	4	3	4

3×3 Averaging Filter

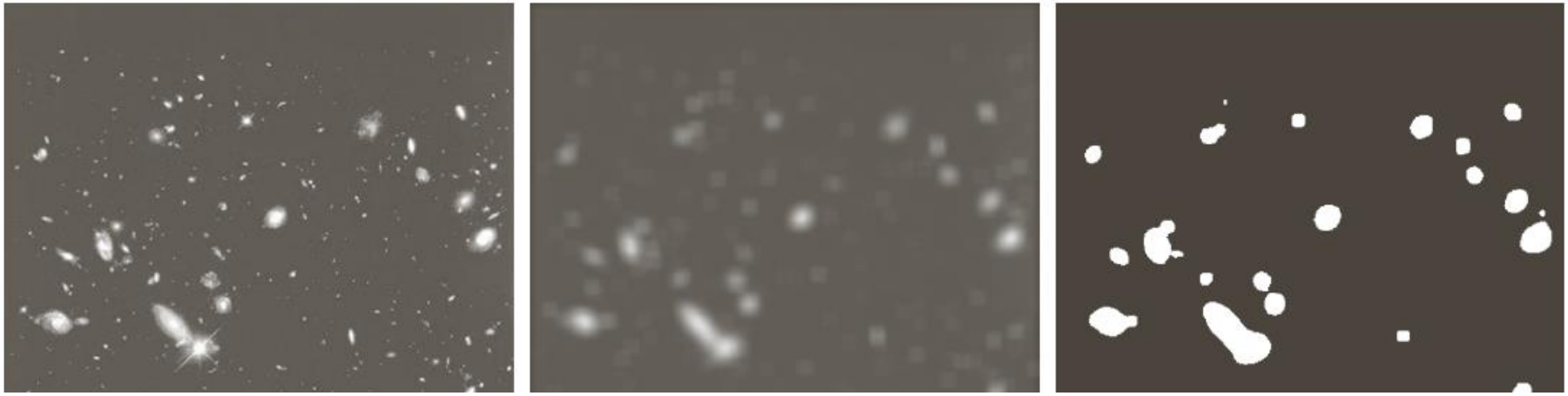
```
>> I = imread ('Fig0335(a)(ckt_board_saltpep_prob_pt05).tif');
>> h = ones(3,3) / 9;
>> J = imfilter (I, h, 'symmetric');
>> imshowpair (I,J,'montage')
```



a b
c d
e f

FIGURE 3.33 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $m = 3, 5, 9, 15$, 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 intensity 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.

Smoothing Followed by Thresholding



a b c

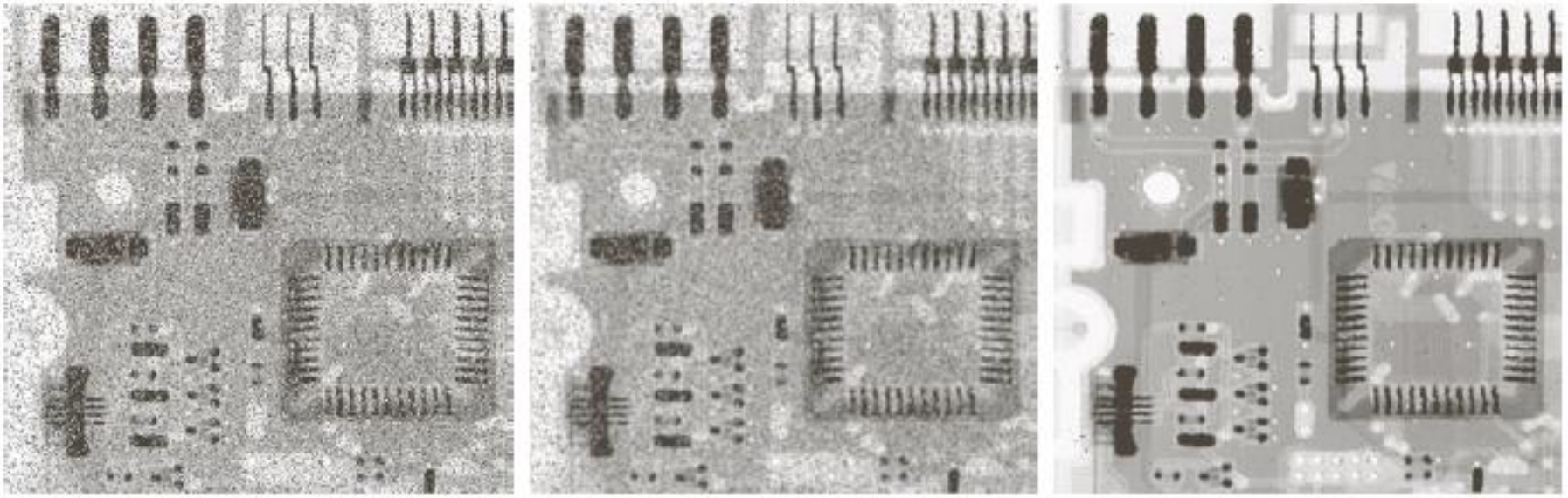
FIGURE 3.34 (a) Image of size 528 × 485 pixels from the Hubble Space Telescope. (b) Image filtered with a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

Median Filters

- The best-known order-statistic (non-linear) filter is the median filler, which replaces the value of a pixel by the median of the intensity values in the neighborhood of that pixel (the original value of the pixel is included in the computation of the median).
- Median filters are quite popular because for certain types of random noise, they provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters of similar size.
- Median filters are particularly effective in the presence of impulse noise, also called **salt-and-pepper** noise because of its appearance as white and black dots superimposed on an image.

- The median ζ of a set of values is such that half the values in the set are less than or equal to ζ and half are greater than or equal to ζ
- In order to perform median filtering at a point in an image, we first sort the values of the pixel in the neighborhood, determine the median, and assign the value to the corresponding pixel in the filtered image.
- For example, in a 3×3 neighborhood, the median is the 5^{th} largest value.
- (10, 15, 20, 20, 20, 20, 20, 25, 100) results in a median of 20.
- The median filters force points with distinct intensity levels to be more like their neighbors.

Denoising



a b c

FIGURE 3.35 (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.)

medfilt2

```
>> I = imread ('coins.png');
>> J = imnoise (I, 'salt & pepper', 0.2);
>> K = medfilt2 (J);
>> imshowpair (J, K,'montage')


>> I = imread ('Fig0335(a)(ckt_board_saltpep_prob_pt05).tif');
>> K = medfilt2(I);
>> imshowpair(I,K,'montage')

>> KS = medfilt2(I,'symmetric');        % symmetric extension
>> imshowpair(K,KS,'montage')
```

Topics

- Background
- Basic Intensity Transformation Functions
- Histogram Processing
- Spatial Filtering
- Smoothing Spatial Filters
- **Sharpening Spatial Filters**
- Combining Spatial Enhancement Methods

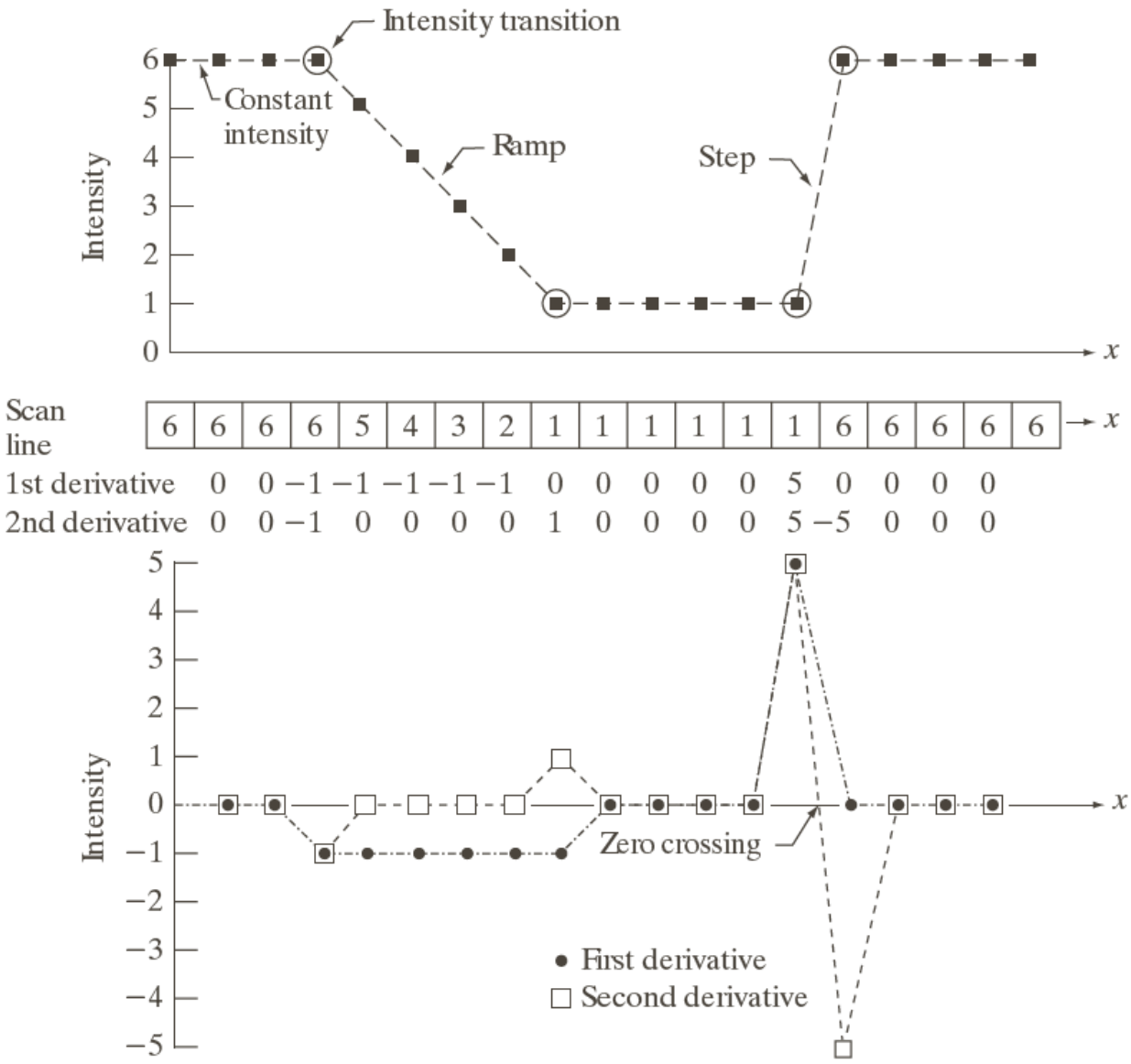
Sharping Spatial Filters

- The principal objective of sharpening is to highlight transitions in intensity.
- Uses of image sharpening vary and include applications from electronic printing and medical imaging to industrial inspection and autonomous guidance in military systems.
- While image blurring can be accomplished by pixel averaging, sharpening can be achieved by spatial differentiation.

Foundation

- First-order derivative of a one-dimensional function is: $\frac{\partial f}{\partial x} = f(x+1) - f(x)$
- Second-order derivative of the function is:

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



a
b
c

FIGURE 3.36 Illustration of the first and second derivatives of a 1-D digital function representing a section of a horizontal intensity profile from an image. In (a) and (c) data points are joined by dashed lines as a visualization aid.

Edges In Digital Images

- Edges in digital images often are ramp-like transitions in intensity, where the first derivative of the image would result in thick edges because the derivative is nonzero along a ramp.
- On the other hand, the second derivative would produce a double edge one pixel thick, separated by zeros.
- Therefore, the second derivative enhances fine detail much better than the first derivative, a property that is ideally suited for sharpening images.

The Laplacian

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

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

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

$$\nabla^2 f(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

0	1	0
1	−4	1
0	1	0

Extension of Laplacian Filter Mask

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 3.37
(a) Filter mask used to implement Eq. (3.6-6).
(b) Mask used to implement an extension of this equation that includes the diagonal terms.
(c) and (d) Two other implementations of the Laplacian found frequently in practice.

Image Sharpening Using Laplacian

- Because the Laplacian is a derivative operator, its use highlights intensity discontinuities in an image and deemphasizes regions with slowly varying intensity levels.
- This will tend to produce images that have grayish edge lines and other discontinuities, all superimposed on a dark, featureless background.
- Background features can be "recovered" while still preserving the sharpening effect of the Laplacian simply by adding the Laplacian image to the original.
- $g(x,y) = f(x,y) - [\nabla^2 f(x,y)]$, if the center coefficient is negative.
- $g(x,y) = f(x,y) + [\nabla^2 f(x,y)]$, if the center coefficient is positive.

h = fspecial (type, parameter)

`h = fspecial(type)` creates a two-dimensional filter `h` of the specified `type`. `fspecial` returns `h` as a correlation kernel, which is the appropriate form to use with `imfilter`. *`type`* is a string having one of these values.

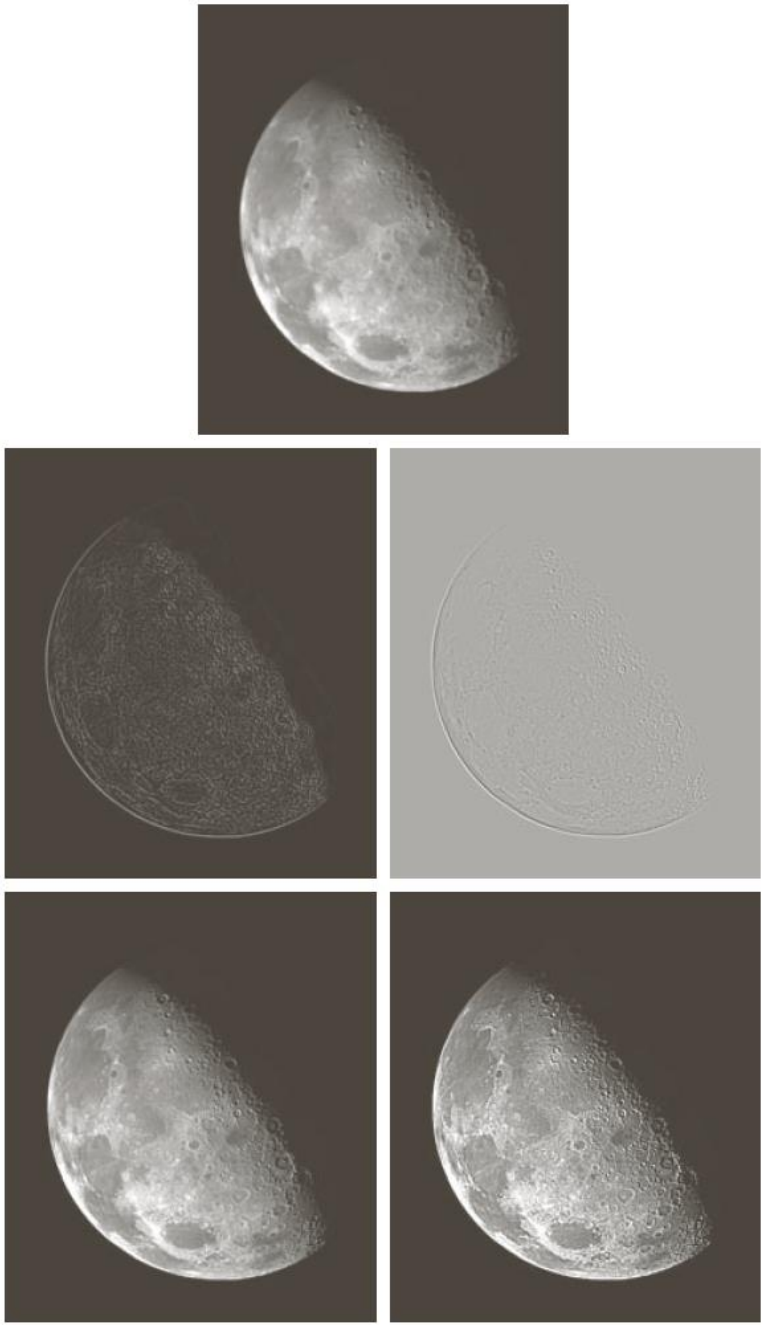
Value	Description
`average`	Averaging filter
`disk`	Circular averaging filter (pillbox)
`gaussian`	Gaussian lowpass filter
`laplacian`	Approximates the two-dimensional Laplacian operator
`log`	Laplacian of Gaussian filter
`motion`	Approximates the linear motion of a camera
`prewitt`	Prewitt horizontal edge-emphasizing filter
`sobel`	Sobel horizontal edge-emphasizing filter

`h = fspecial('laplacian', alpha)` returns a 3-by-3 filter approximating the shape of the two-dimensional Laplacian operator. The parameter `alpha` controls the shape of the Laplacian and must be in the range 0.0 to 1.0. The default value for `alpha` is 0.2.

```
>> I = imread('Fig0338(a)(blurry_moon).tif');
>> h = fspecial('laplacian', 0)
h =
    0     1     0
    1    -4     1
    0     1     0
>> J1 = imfilter(I, h, 'replicate');
>> imshow(J1, [])                % Truncation problem

>> I2 = im2double(I);
>> J2 = imfilter(I2, h, 'replicate');
>> imshow(J2, [])

>> G = I2 - J2;
>> imshow(G)
```



$$f_m = f - \min(f)$$

$$f_s = K\left[\frac{f_m}{\max(f_m)}\right], \text{ where } K = 255$$

a
b c
d e

FIGURE 3.38
(a) Blurred image of the North Pole of the moon. (b) Laplacian without scaling. (c) Laplacian with scaling (d) Image sharpened using the mask in Fig. 3.37(a). (e) Result of using the mask in Fig. 3.37(b). (Original image courtesy of NASA.)

Unsharp Masking

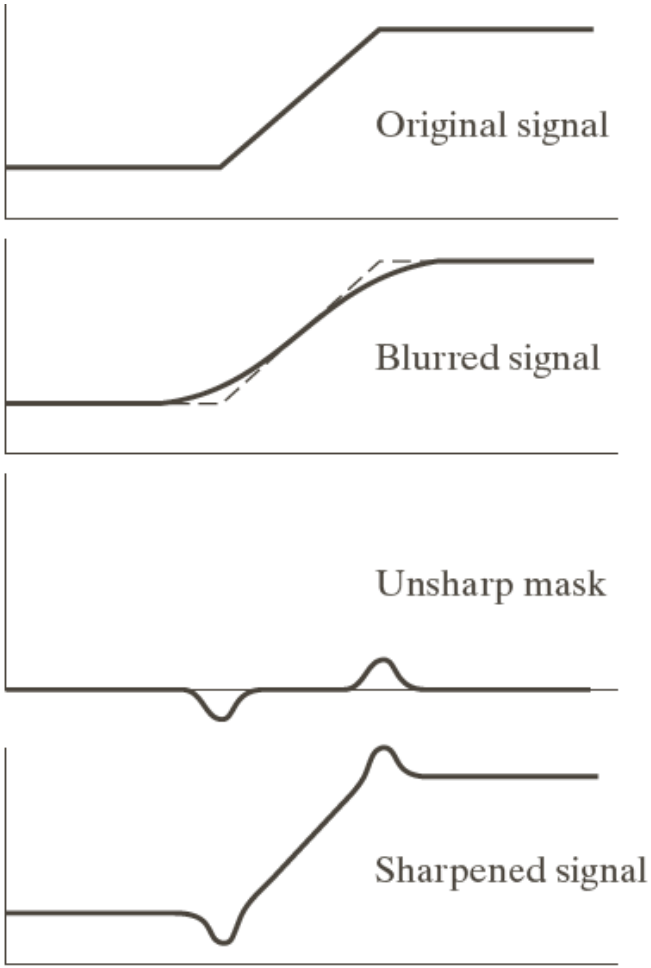
- A process that has been used in the printing and publishing industry to sharpen images consists of subtracting an unsharp (smoothed) version of an image from the original image. This process, called *unsharp masking,* consists of the following steps:
  - Blur the original image
  - Subtract the blurred image from the original (the resulting difference is called the masks)

$$g_{mask}(x, y) = f(x, y) - \bar{f}(x, y)$$

  - Add the mask to the original:

$$g(x, y) = f(x, y) + k \times g_{mask}(x, y)$$

  - When $k = 1$, we have **unsharp masking**; when $k > 1$, the process is called **highboost filtering**.



a
b
c
d

FIGURE 3.39 1-D illustration of the mechanics of unsharp masking. (a) Original signal. (b) Blurred signal with original shown dashed for reference. (c) Unsharp mask. (d) Sharpened signal, obtained by adding (c) to (a).

a
b
c
d
e

FIGURE 3.40
(a) Original image.
(b) Result of blurring with a Gaussian filter.
(c) Unsharp mask. (d) Result of using unsharp masking.
(e) Result of using highboost filtering.

The Gradient for Image Sharpening

- First derivatives in image processing are implemented using the magnitude of the gradient.
- For a function $f(x, y)$, the gradient of f at coordinates (x, y) is defined as the two-dimensional column vector

$$\nabla f \equiv \text{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- This vector has the important geometrical property that it points in the direction of the greatest rate of change of f at location (x, y).
- The magnitude of the vector is $M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$ called the *gradient image*, or simply as *gradient*.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0
0	1

0	−1
1	0

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

a
b c
d e

FIGURE 3.41
A 3 × 3 region of an image (the zs are intensity values).
(b)–(c) Roberts cross gradient operators.
(d)–(e) Sobel operators. All the mask coefficients sum to zero, as expected of a derivative operator.

```
>> hy = -fspecial('sobel')
>> hx = hy'
```

Sobel Operators

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

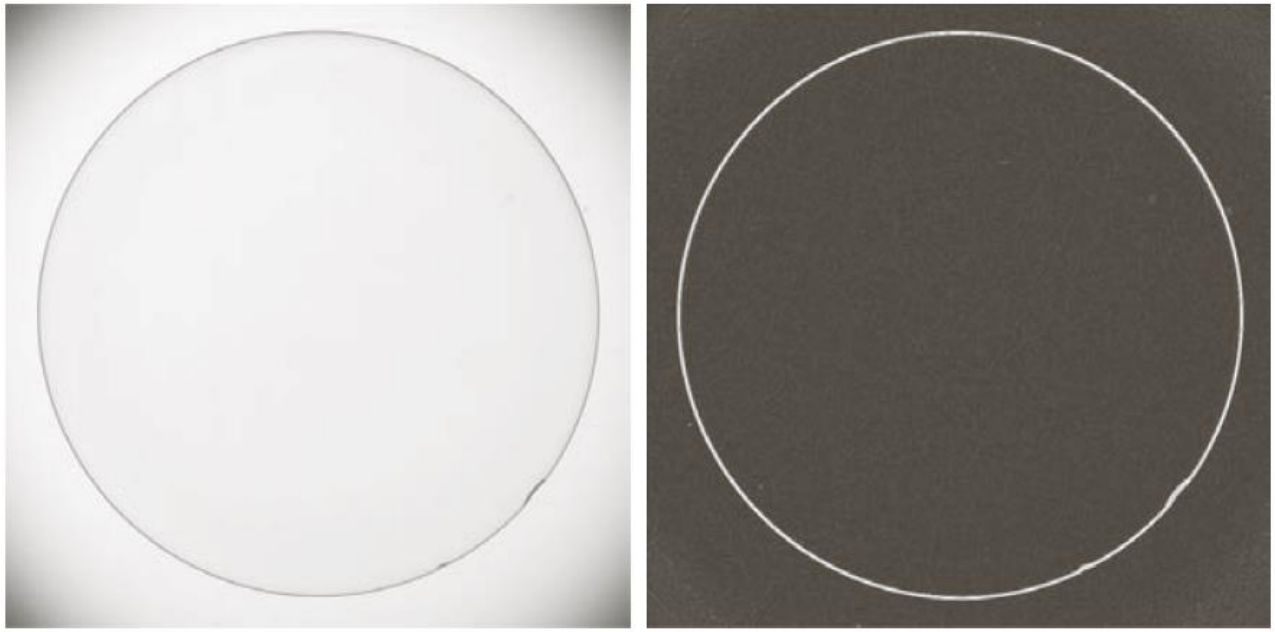
$$M(x, y) \approx |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|$$

imgradient & imgradientxy

```
[Gx,Gy] = imgradientxy(I)
[Gmag,Gdir] = imgradient(I)
```

```
>> [Gx,Gy]= imgradientxy (I,'sobel');
>> imshowpair (Gx,Gy,'montage')
```

```
>> I = imread('Fig0342(a)(contact_lens_original).tif');
>> sobelGradient = imgradient(I);
>> imshow (sobelGradient,[ ])
```

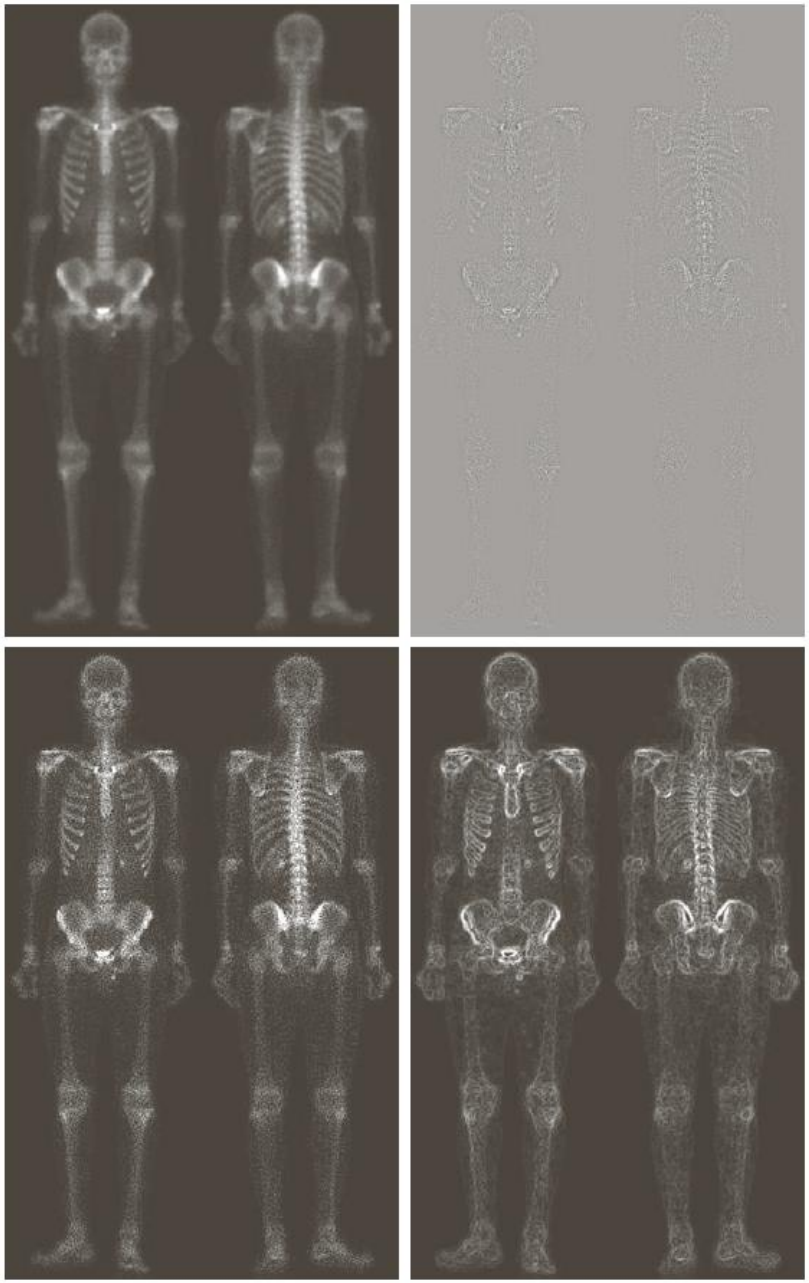


a b

FIGURE 3.42
(a) Optical image of contact lens (note defects on the boundary at 4 and 5 o'clock). (b) Sobel gradient. (Original image courtesy of Pete Sites, Perceptics Corporation.)

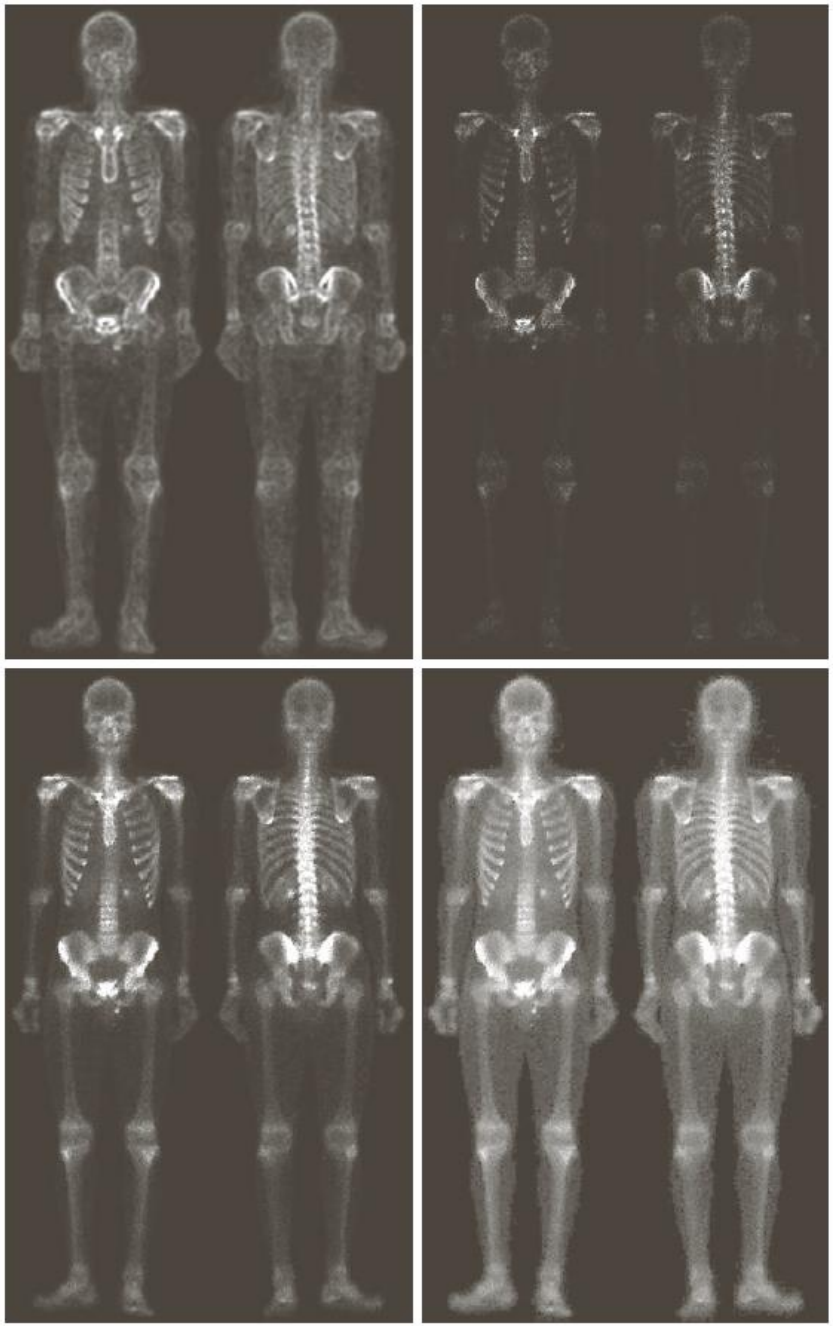
Topics

- Background
- Basic Intensity Transformation Functions
- Histogram Processing
- Spatial Filtering
- Smoothing Spatial Filters
- Sharpening Spatial Filters
- **Combining Spatial Enhancement Methods**



a b
c d

FIGURE 3.43
(a) Image of whole body bone scan.
(b) Laplacian of (a). (c) Sharpened image obtained by adding (a) and (b).
(d) Sobel gradient of (a).



e f
g h

FIGURE 3.43
(Continued)
(e) Sobel image smoothed with a 5×5 averaging filter. (f) Mask image formed by the product of (c) and (e). (g) Sharpened image obtained by the sum of (a) and (f). (h) Final result obtained by applying a power-law transformation to (g). Compare (g) and (h) with (a). (Original image courtesy of G.E. Medical Systems.)