Course 3:

Spatial Processing

Outline

- Intensity transformations
  - Point processing, mainly operate on single pixels, for contrast manipulation, image thresholding and so on
- Spatial filtering
  - Neighborhood processing, operate on neighborhood of every pixel, for image enhancement and many other target-oriented applications

Basic formulation

- Spatial processing:

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

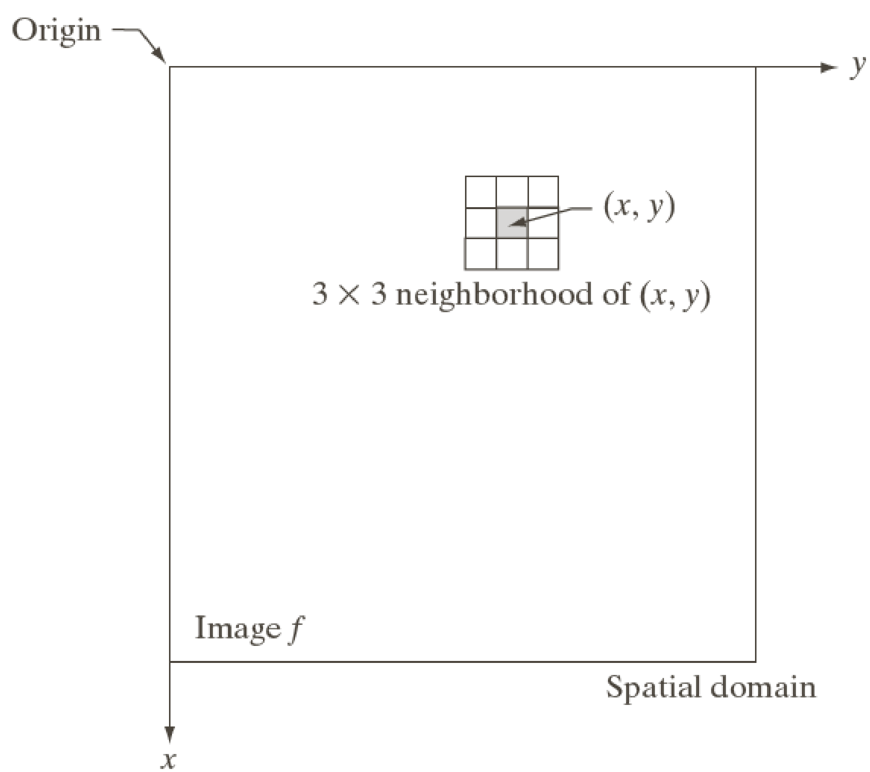
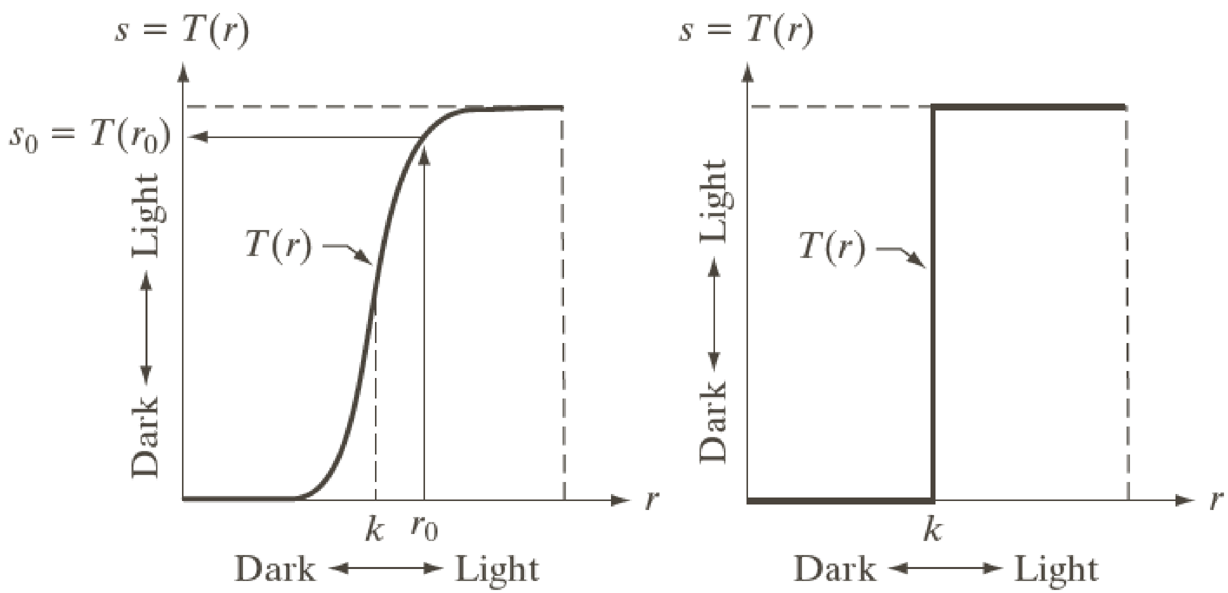


FIGURE 3.1 A 3 × 3 neighborhood about a point (x, y) in an image in the spatial domain. The neighborhood is moved from pixel to pixel in the image to generate an output image.

Basic formulation

- Intensity transformation:

$$s = T(r)$$



a b

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

Basic intensity transformations

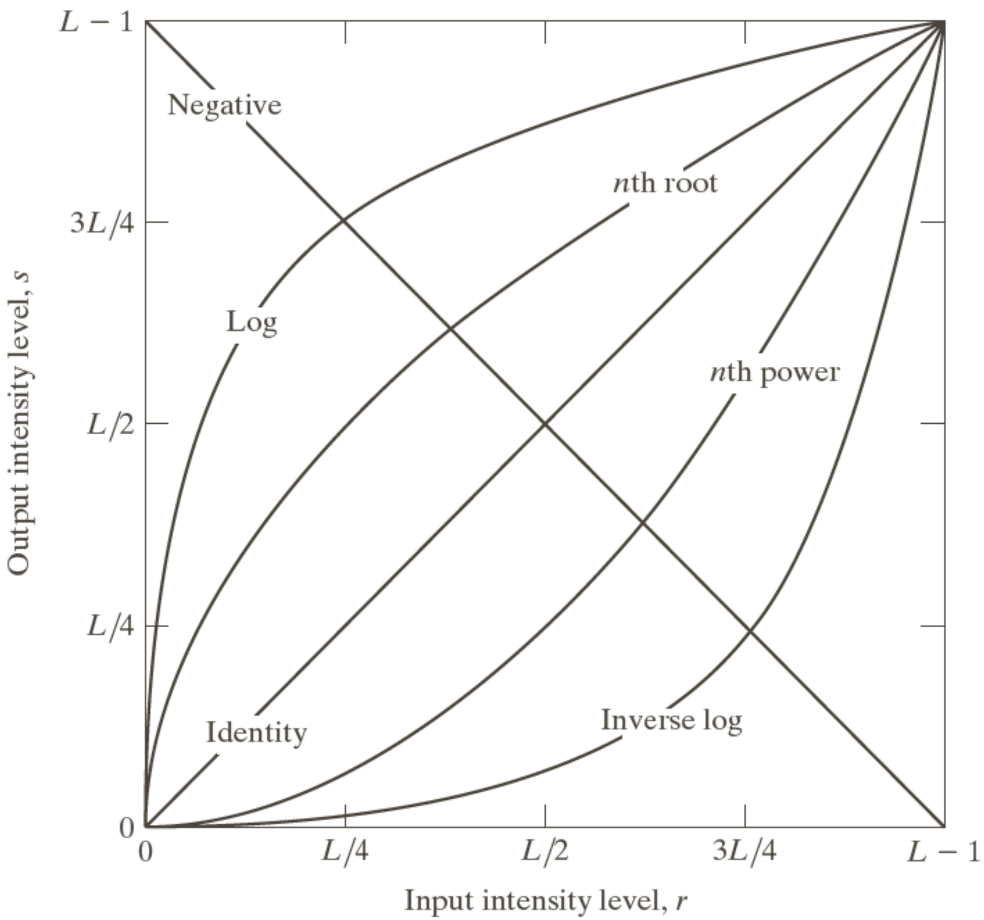
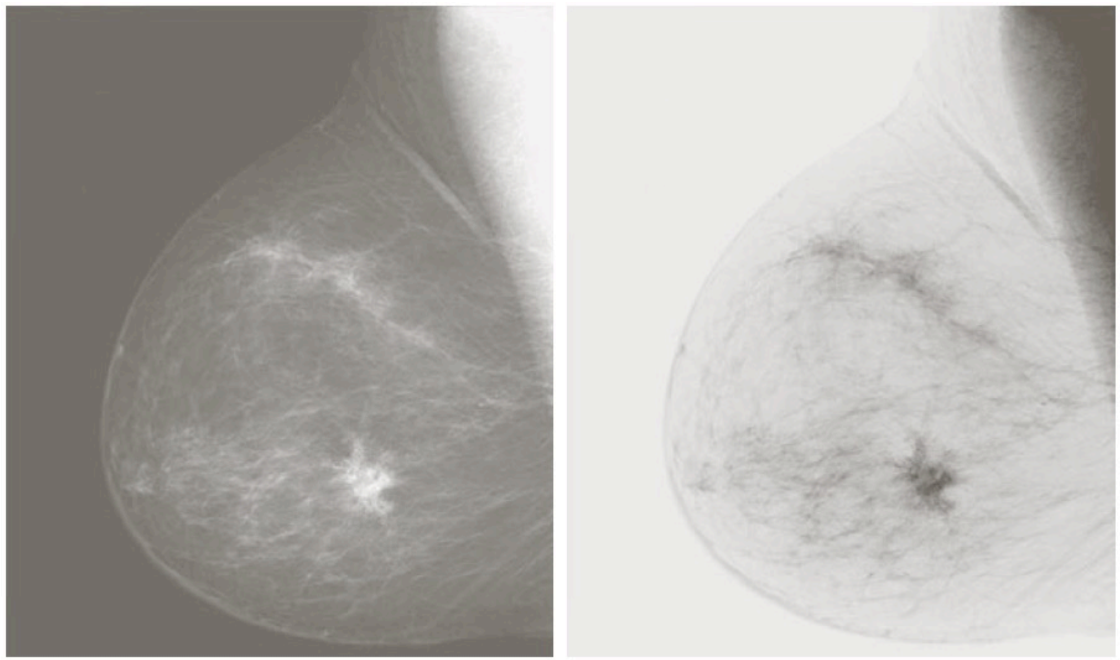


FIGURE 3.3 Some basic intensity transformation functions. All curves were scaled to fit in the range shown.

Image Negatives : s=L-1-r

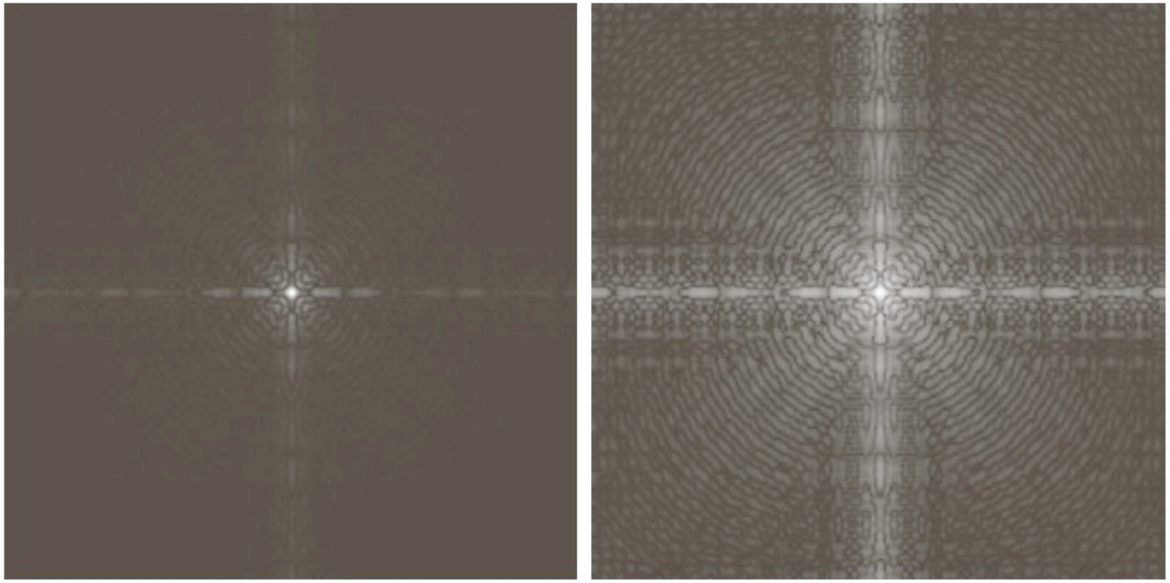


a b

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.)

For enhancing white or gray detail embedded in dark regions of an image, especially when the black areas are dominant in size.

Log Transformations: $s = c * \log(1 + r)$



a b

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

To expand the values of dark pixels in an image while compressing the higher-level values. The opposite is true of the inverse log transformation.

Power-Law (Gamma) Transformations : s = c r^{gamma}

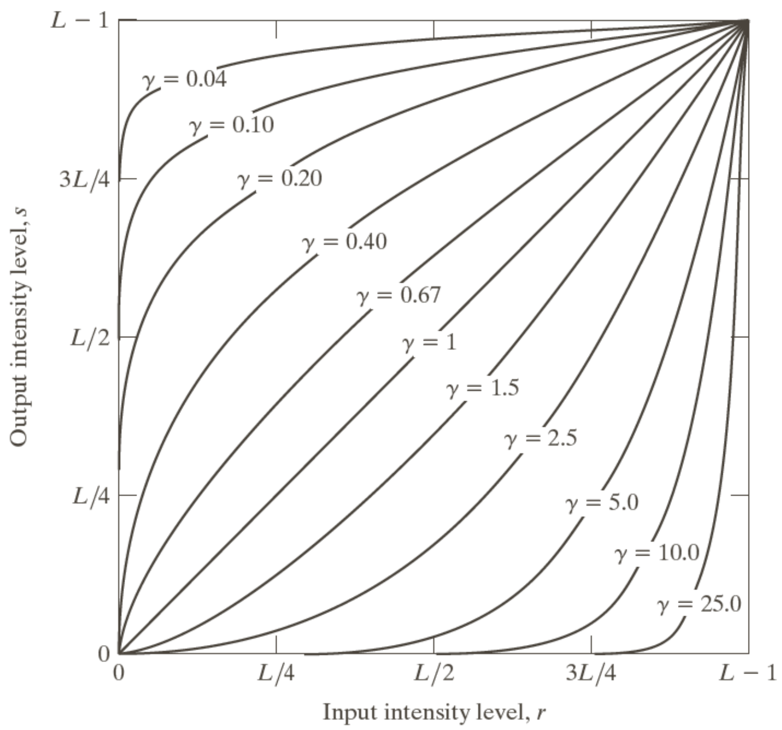
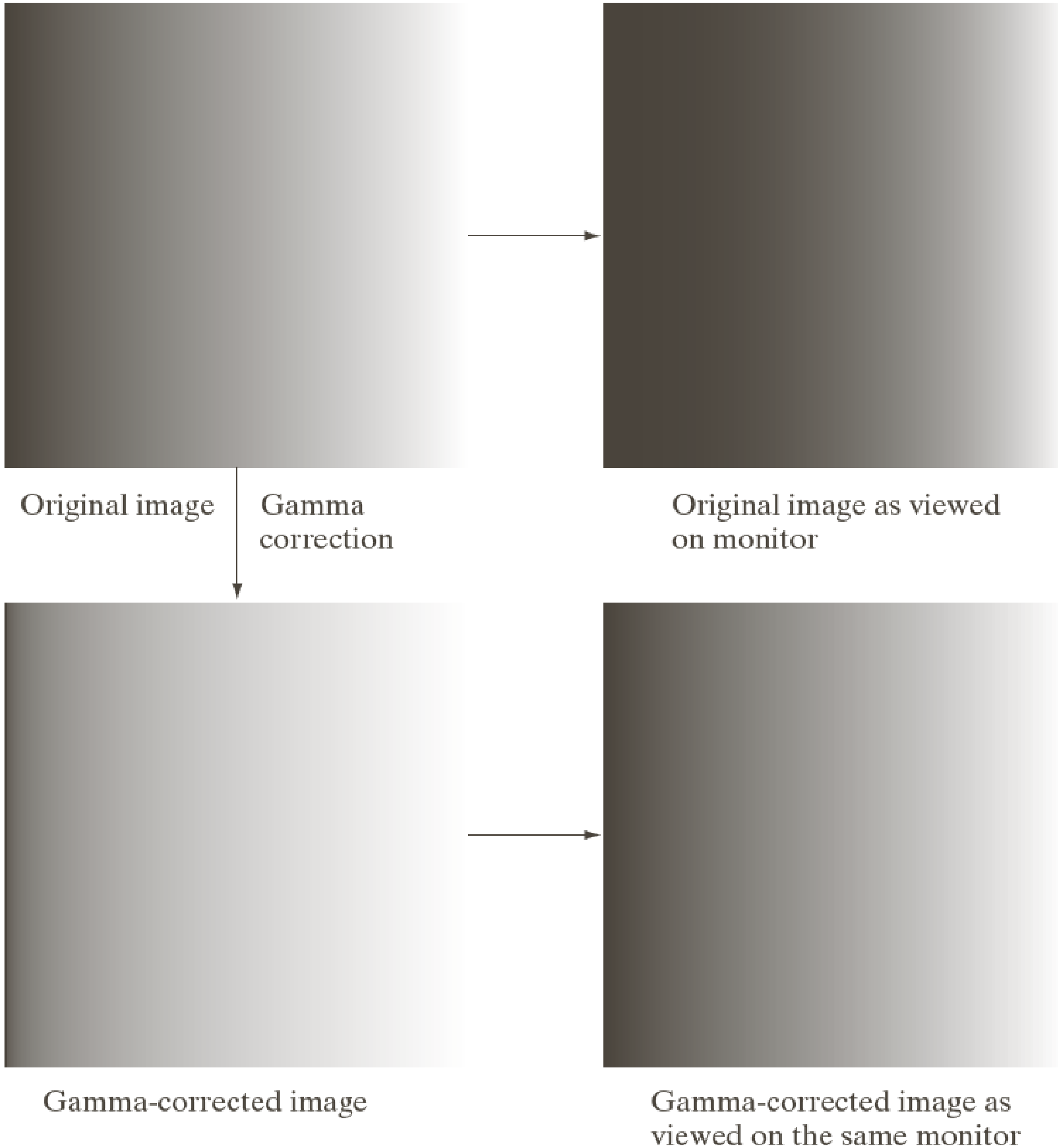


FIGURE 3.6 Plots of the equation $s = cr^{\gamma}$ for various values of γ ($c = 1$ in all cases). All curves were scaled to fit in the range shown.



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).

gamma correction

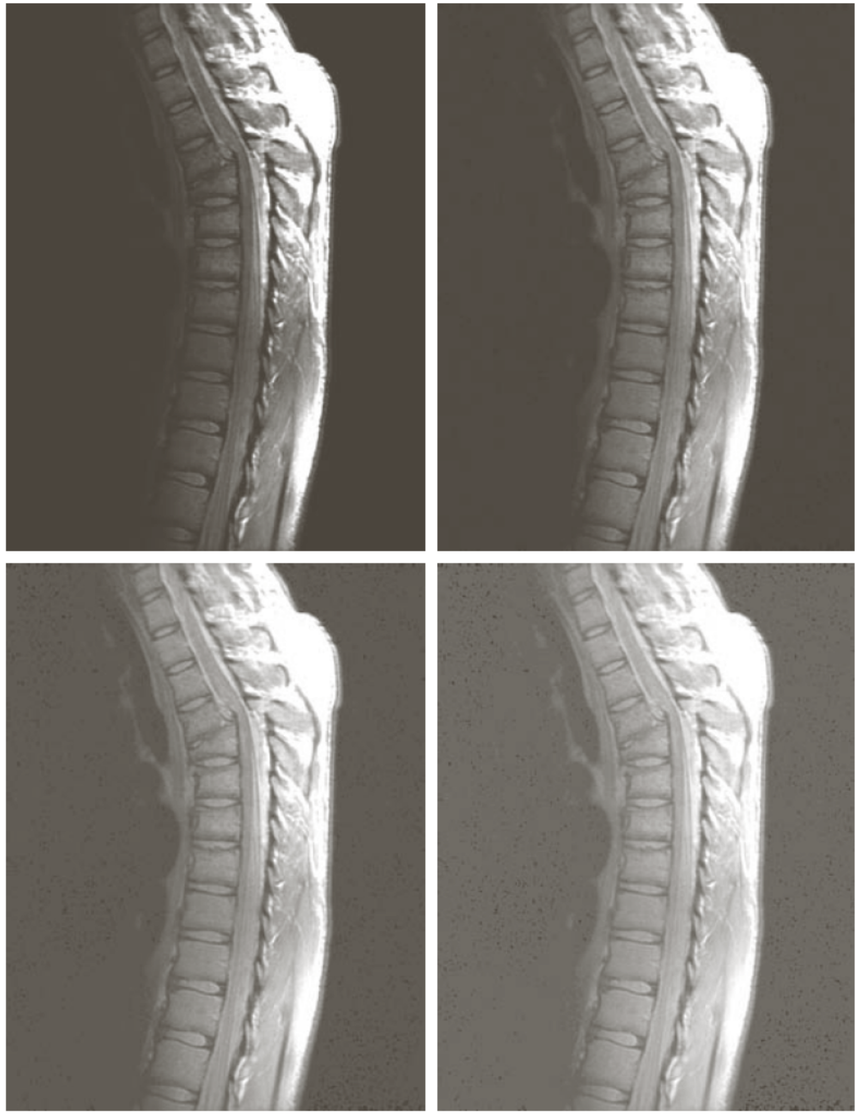


FIGURE 3.8 (a) Magnetic resonance image (MRI) of a fractured human spine. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 0.6, 0.4,$ and 0.3, respectively. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt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

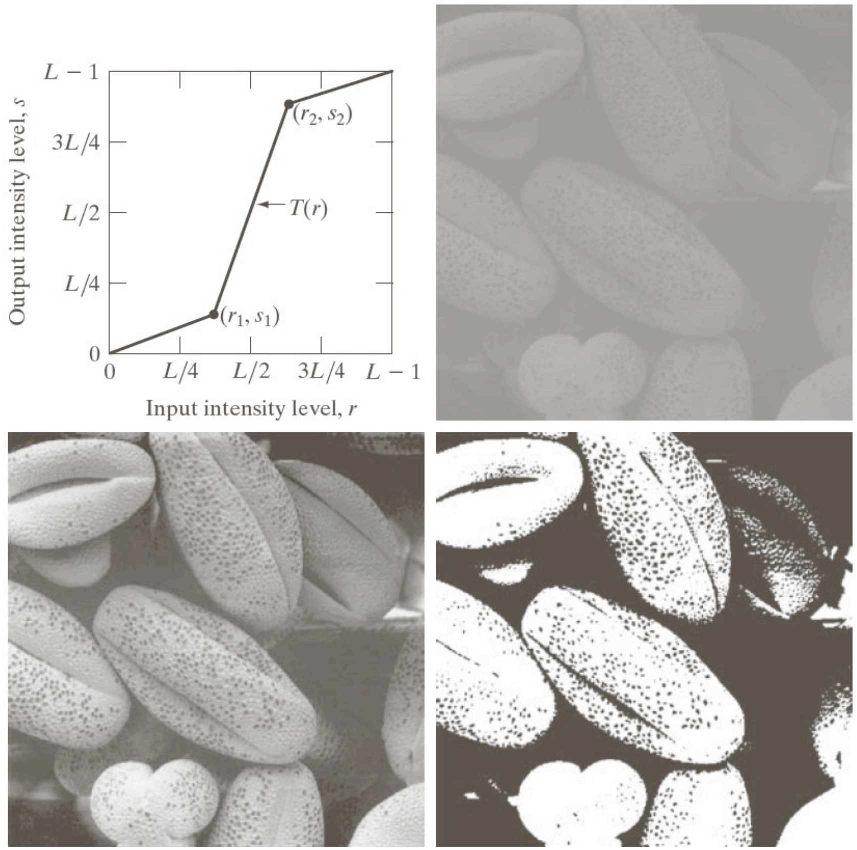
contrast manipulation

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.)

contrast manipulation

Piecewise-Linear Transformation Functions



a b
c d

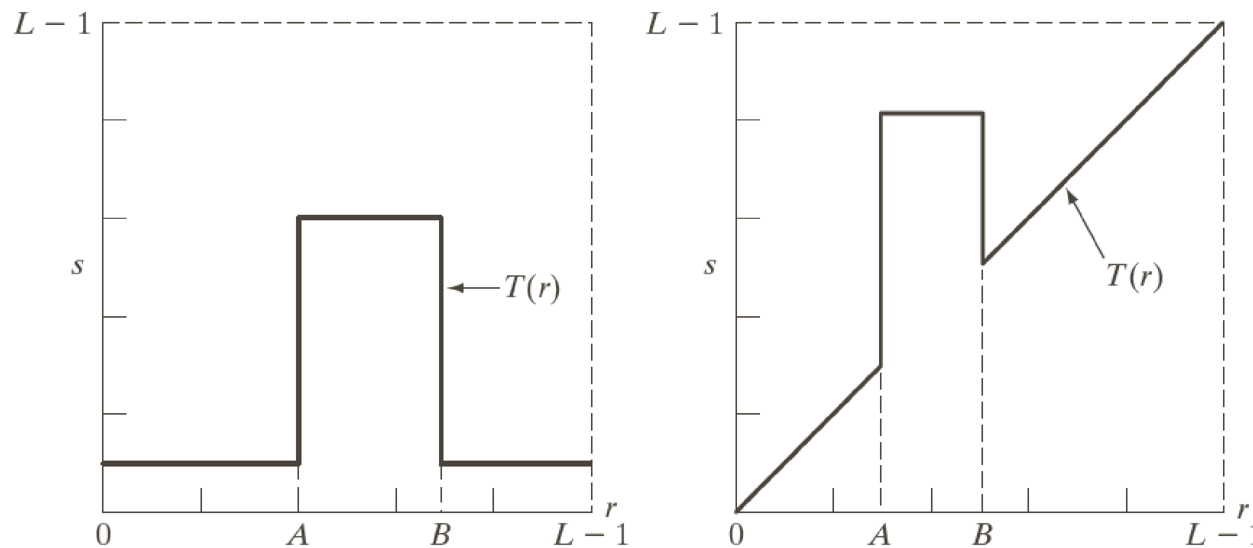
FIGURE 3.10 Contrast stretching. (a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

Contrast stretching is a process that expands the range of intensity levels in an image so that it spans the full intensity range of the recording medium or display device.

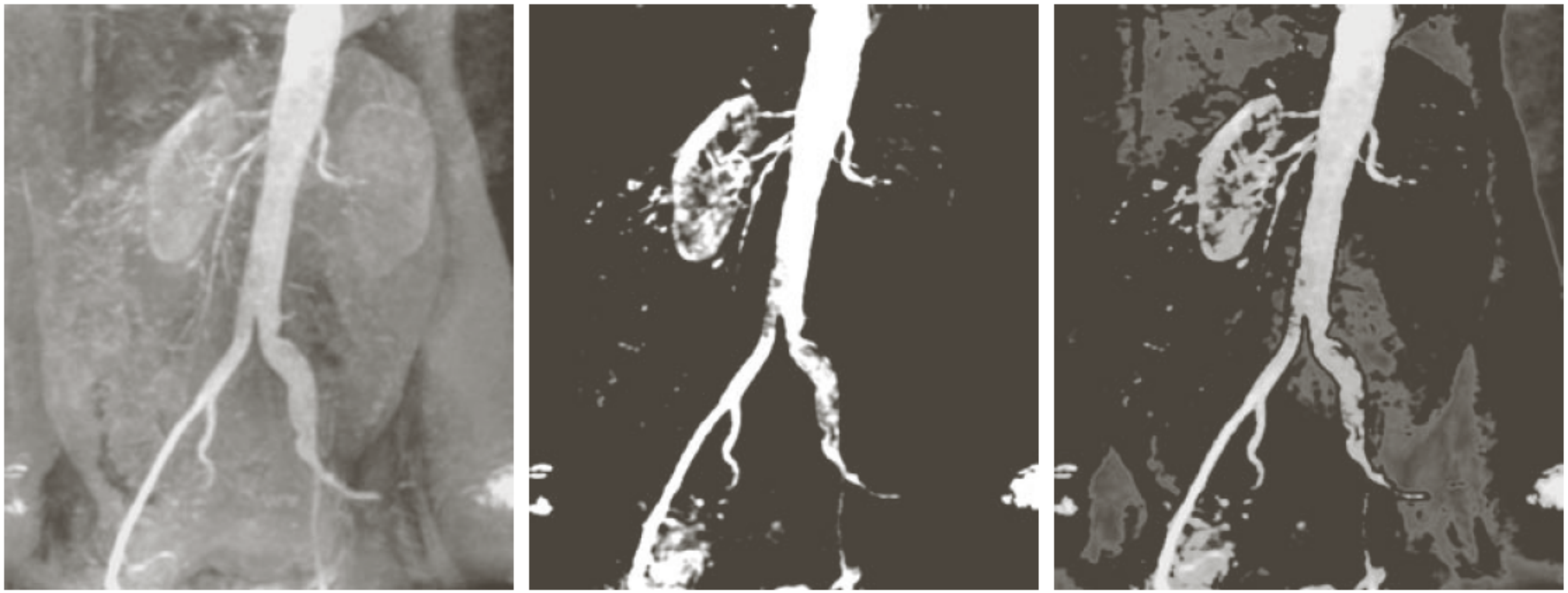
Piecewise-Linear Transformation Functions

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.



Intensity-level slicing



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.)

Piecewise-Linear Transformation Functions

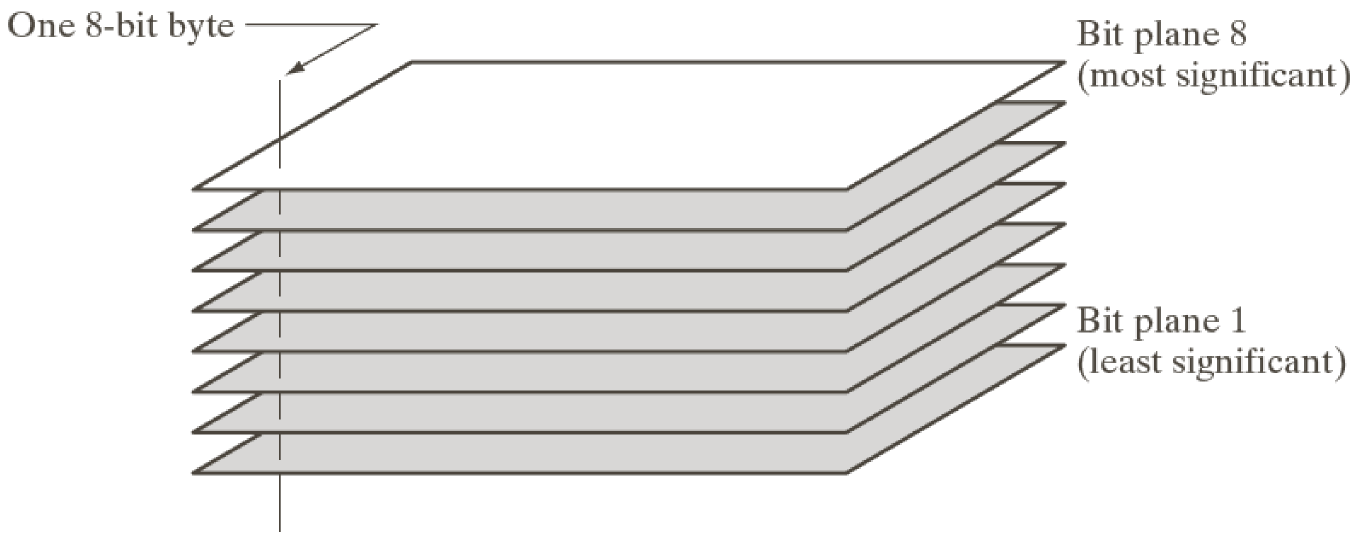


FIGURE 3.13
Bit-plane representation of an 8-bit image.

Bit-plane slicing

a b c
d e f
g h i

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.

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).

Reconstruction

Histogram processing

- Histogram: statistical description about the occurrences of intensity value of image pixels

$$h(r_k) = n_k$$

$$p(r_k) = n_k/M*N$$

- Histogram can be used for image enhancement, image compression, segmentation and representation.

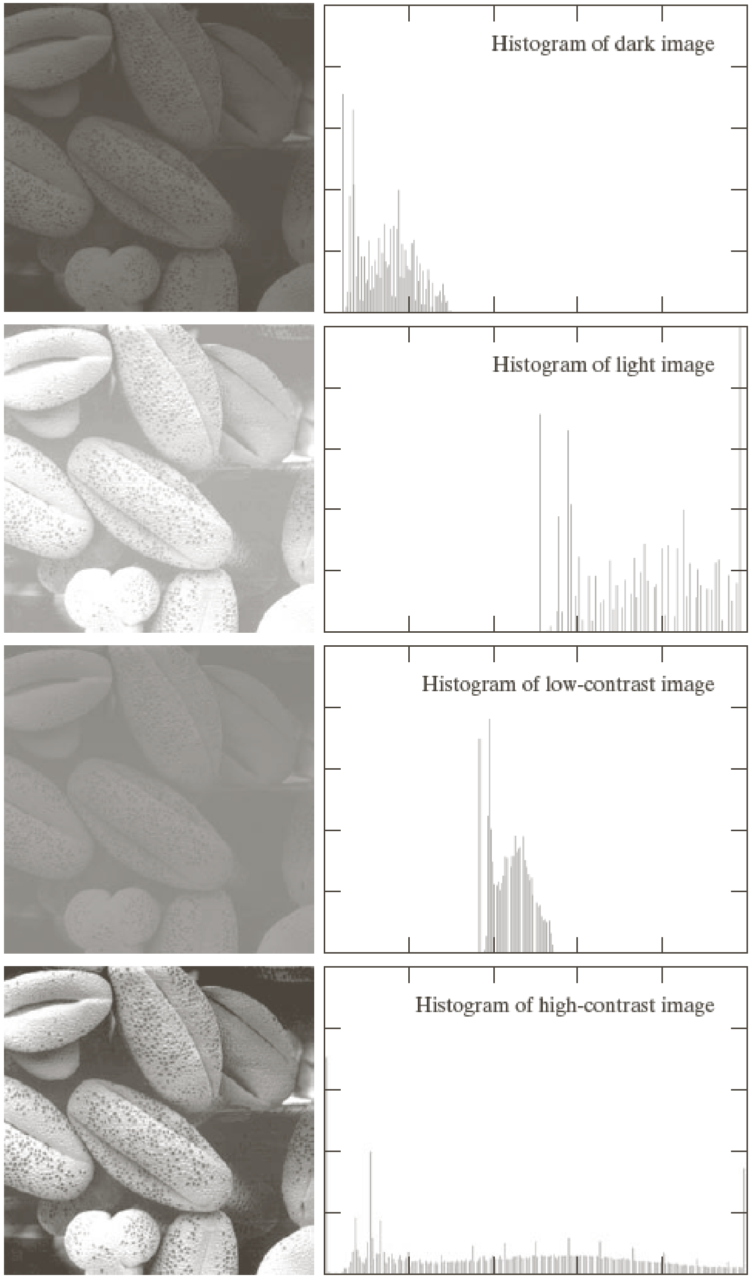


FIGURE 3.16 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms.

"Intuitively, it is reasonable to conclude that an image whose pixels tend to occupy the entire range of possible intensity levels and, in addition, tend to be distributed uniformly, will have an appearance of high contrast and will exhibit a large variety of gray tones. "

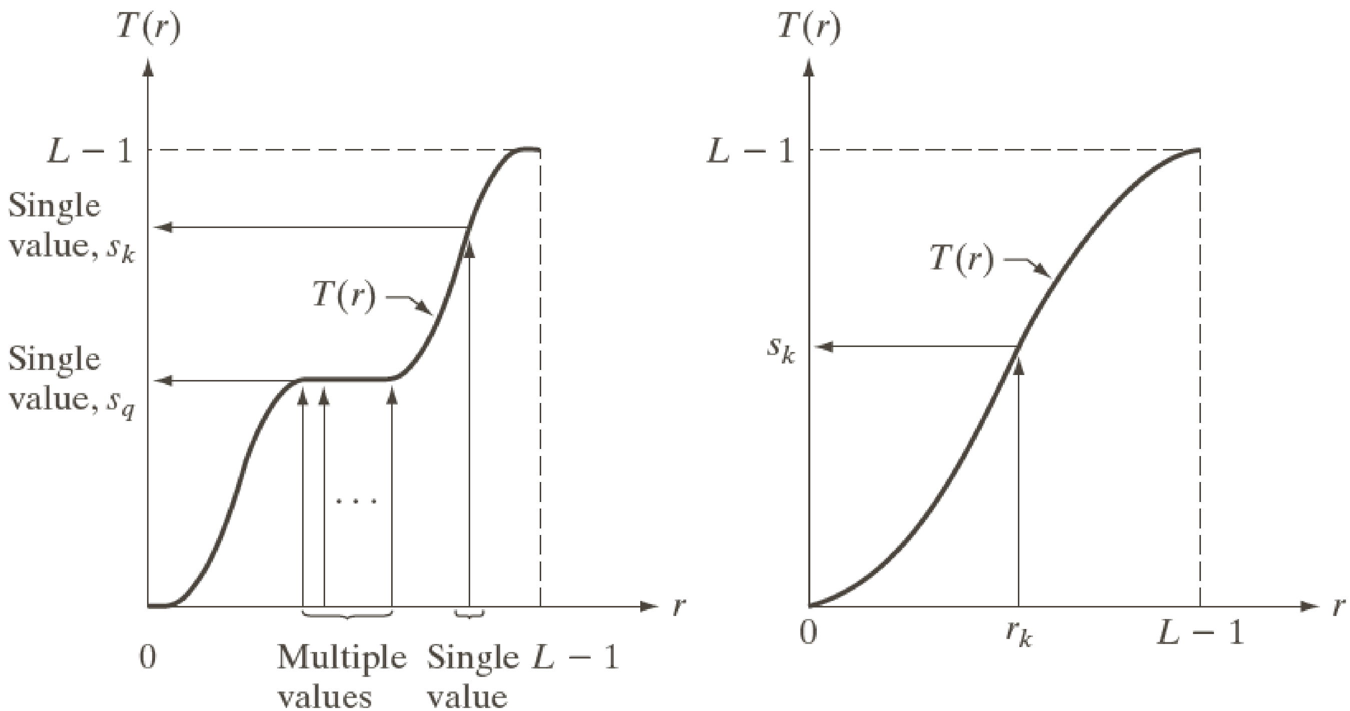
Histogram equalization

- Intensity transformation function

$$s = T(r) \quad 0 \leq r \leq L - 1$$

should satisfy

a) T(r) is a monotonically increasing function in the interval 0 <= r<= L - 1; and
b) 0 <=T(r)<= L – 1, for 0<= r <=L - 1.



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.

Probability description

- Intensity levels, r and s, can be viewed as random variables in the interval [0, L-1]
- $P_s(s)$ and $P_r(r)$ denote the PDFs of r and s, so we have

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

- An important transformation function

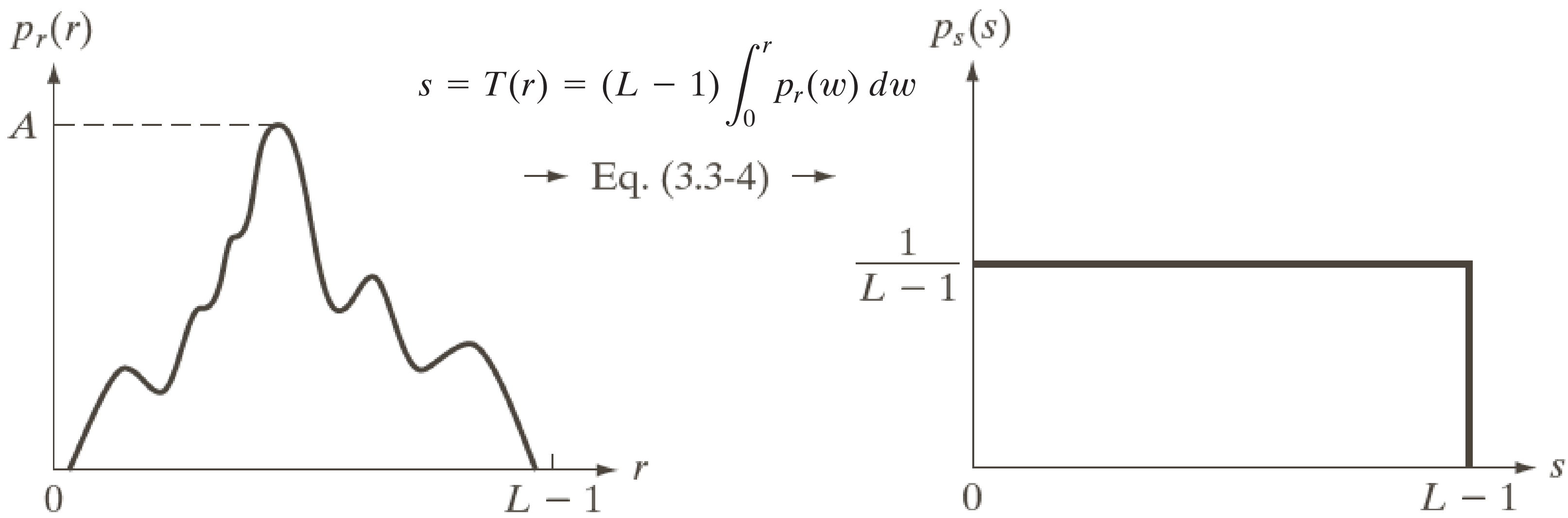
$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw$$

Probability description

- The derivation of $p_s(s)$

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

$$\begin{aligned}p_s(s) &= p_r(r)\left|\frac{dr}{ds}\right| \\ &= p_r(r)\left|\frac{1}{(L-1)p_r(r)}\right| \\ &= \frac{1}{L-1} \qquad 0 \le s \le L-1\end{aligned}$$



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.

Discrete form

- The discrete form of the transformation

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

is

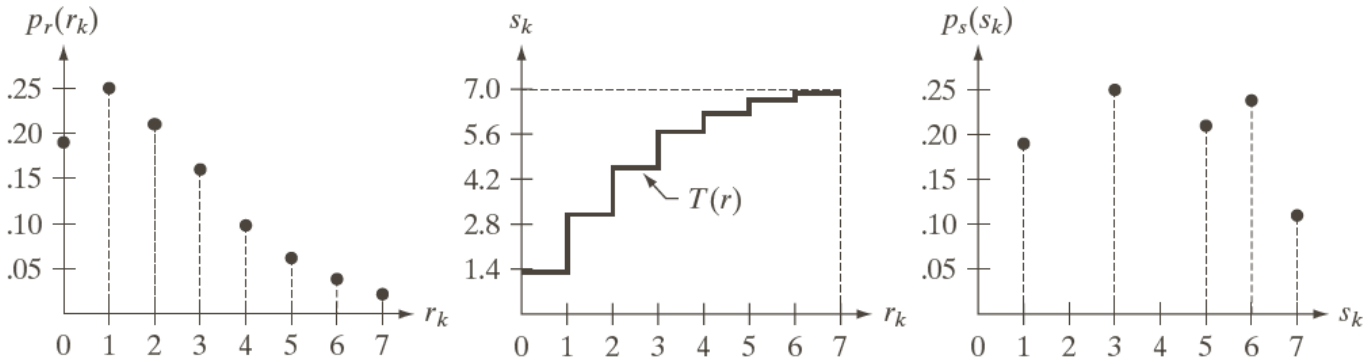
$$s_k = T(r_k) = (L - 1)\sum_{j=0}^{k} p_r(r_j)$$

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

as

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

A simple example



a b c

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

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

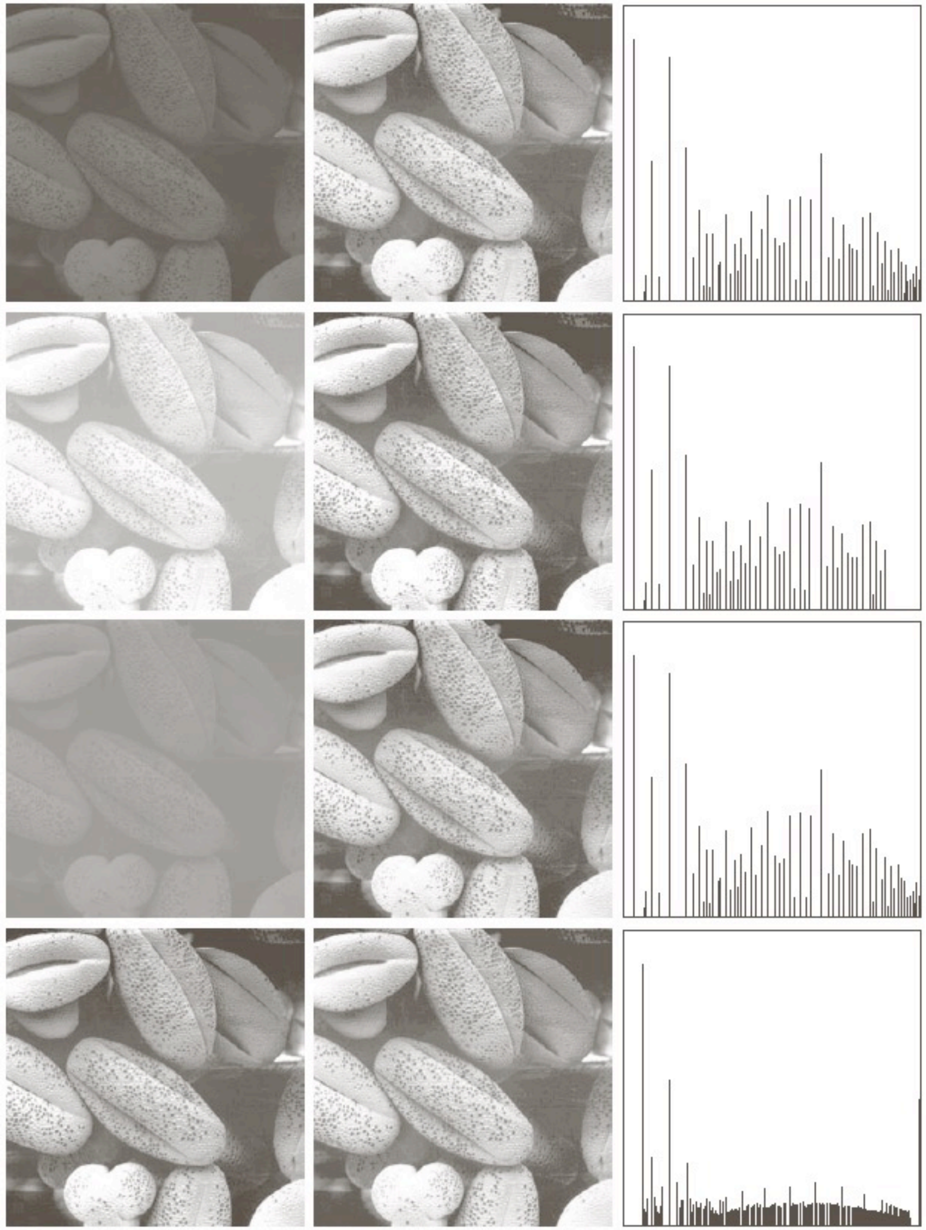


FIGURE 3.20 Left column: images from Fig. 3.16. Center column: corresponding histogram-equalized images. Right column: histograms of the images in the center column.

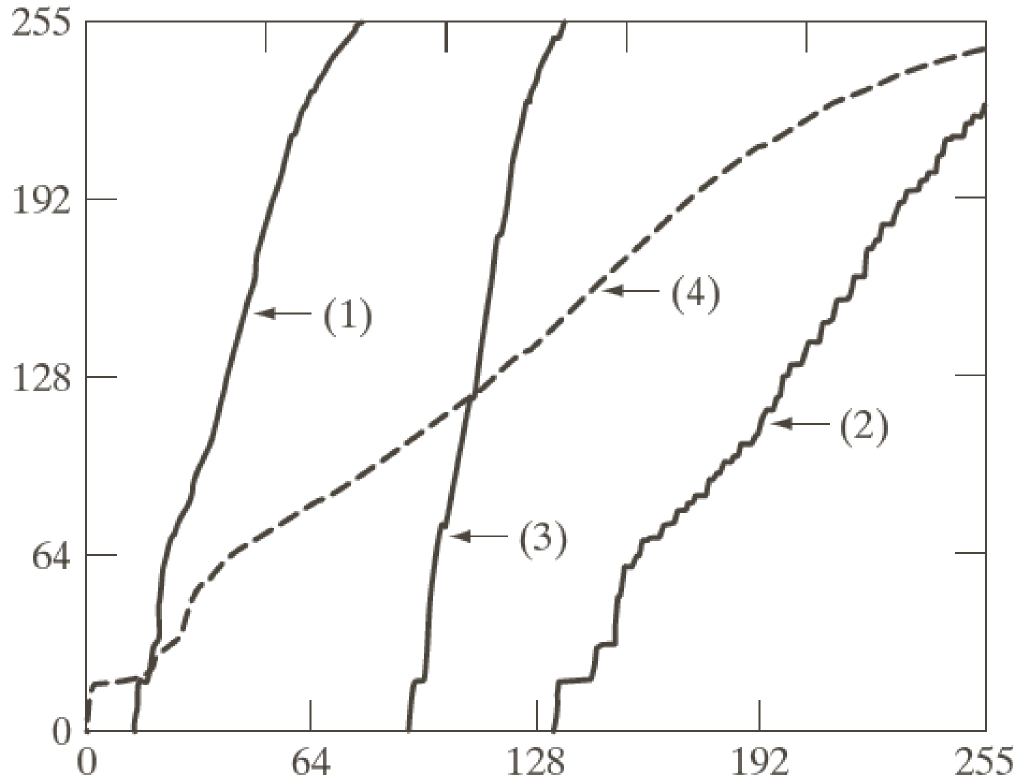


FIGURE 3.21 Transformation functions for histogram equalization. Transformations (1) through (4) were obtained from the histograms of the images (from top to bottom) in the left column of Fig. 3.20 using Eq. (3.3-8).

Histogram matching (specification)

- The method used to generate a processed image that has a specified histogram is called *histogram matching* or *histogram specification*.

$$s = T(r) = (L - 1)\int_0^r p_r(w)\, dw \qquad (3.3\text{-}10)$$

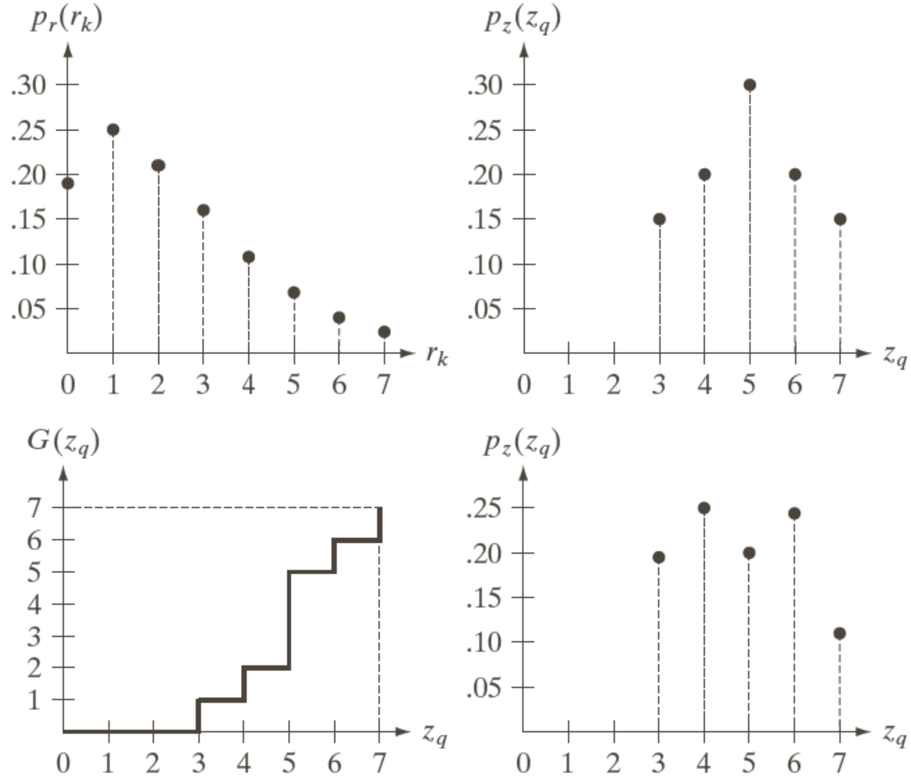
$$G(z) = (L - 1)\int_0^z p_z(t)\, dt = s \qquad (3.3\text{-}11)$$

$$z = G^{-1}[T(r)] = G^{-1}(s) \qquad (3.3\text{-}12)$$

Histogram matching (specification)

- Procedure
  ① Obtain $p_r(r)$ from the input image and use Eq. (3.3-10) to obtain the values of *s*.
  ② Use the specified PDF in Eq.(3.3 11) to obtain the transformation function *G*(*z*).
  ③ Obtain the inverse transformation $z = G^{-1}(s)$; because z is obtained from s, this process is a mapping from s to z, the latter being the desired values.
  ④ Obtain the output image by first equalizing the input image using Eq. (3.3-10); the pixel values in this image are the s values. For each pixel with value s in the equalized image, perform the inverse mapping z = G-1(s) to obtain the corresponding pixel in the output image.

An example



a b
c d

FIGURE 3.22 (a) Histogram of a 3-bit image. (b) Specified histogram. (c) Transformation function obtained from the specified histogram. (d) Result of performing histogram specification. Compare (b) and (d).

z_q	Specified $p_z(z_q)$	Actual $p_z(z_k)$
$z_0 = 0$	0.00	0.00
$z_1 = 1$	0.00	0.00
$z_2 = 2$	0.00	0.00
$z_3 = 3$	0.15	0.19
$z_4 = 4$	0.20	0.25
$z_5 = 5$	0.30	0.21
$z_6 = 6$	0.20	0.24
$z_7 = 7$	0.15	0.11

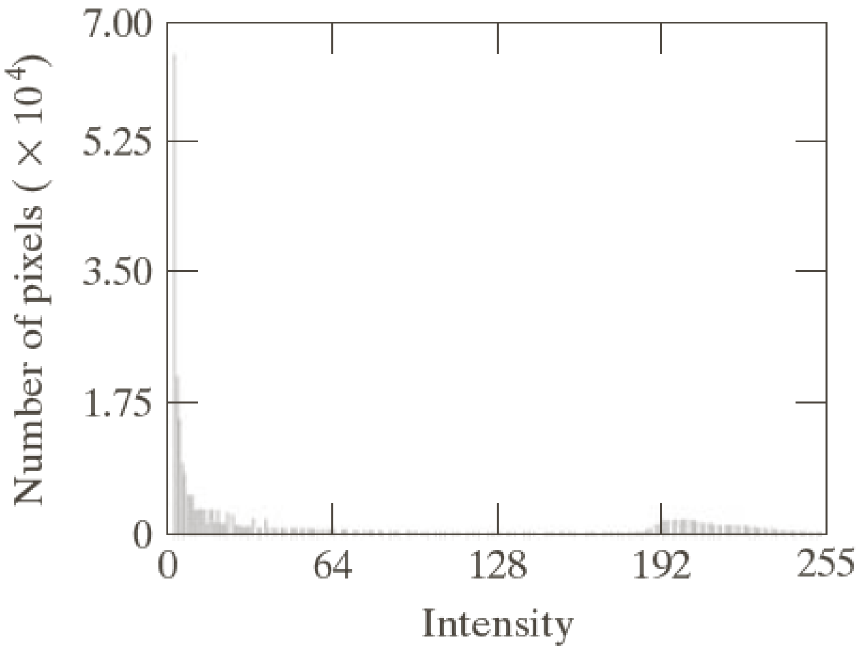
TABLE 3.2 Specified and actual histograms (the values in the third column are from the computations performed in the body of Example 3.8).

z_q	$G(z_q)$
$z_0 = 0$	0
$z_1 = 1$	0
$z_2 = 2$	0
$z_3 = 3$	1
$z_4 = 4$	2
$z_5 = 5$	5
$z_6 = 6$	6
$z_7 = 7$	7

TABLE 3.3 All possible values of the transformation function G scaled, rounded, and ordered with respect to z.

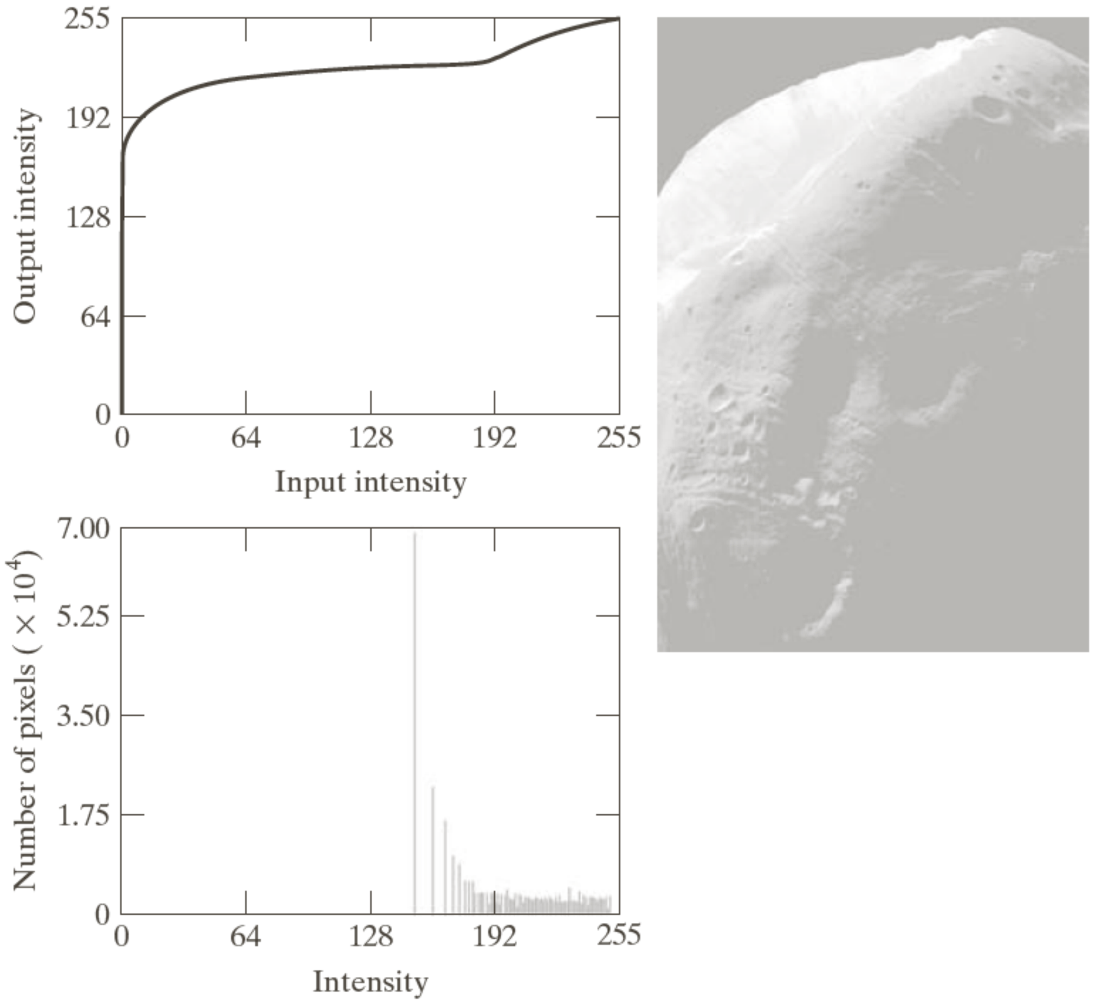
s_k	$\rightarrow$	z_q
1	$\rightarrow$	3
3	$\rightarrow$	4
5	$\rightarrow$	5
6	$\rightarrow$	6
7	$\rightarrow$	7

TABLE 3.4 Mappings of all the values of s_k into corresponding values of z_q.



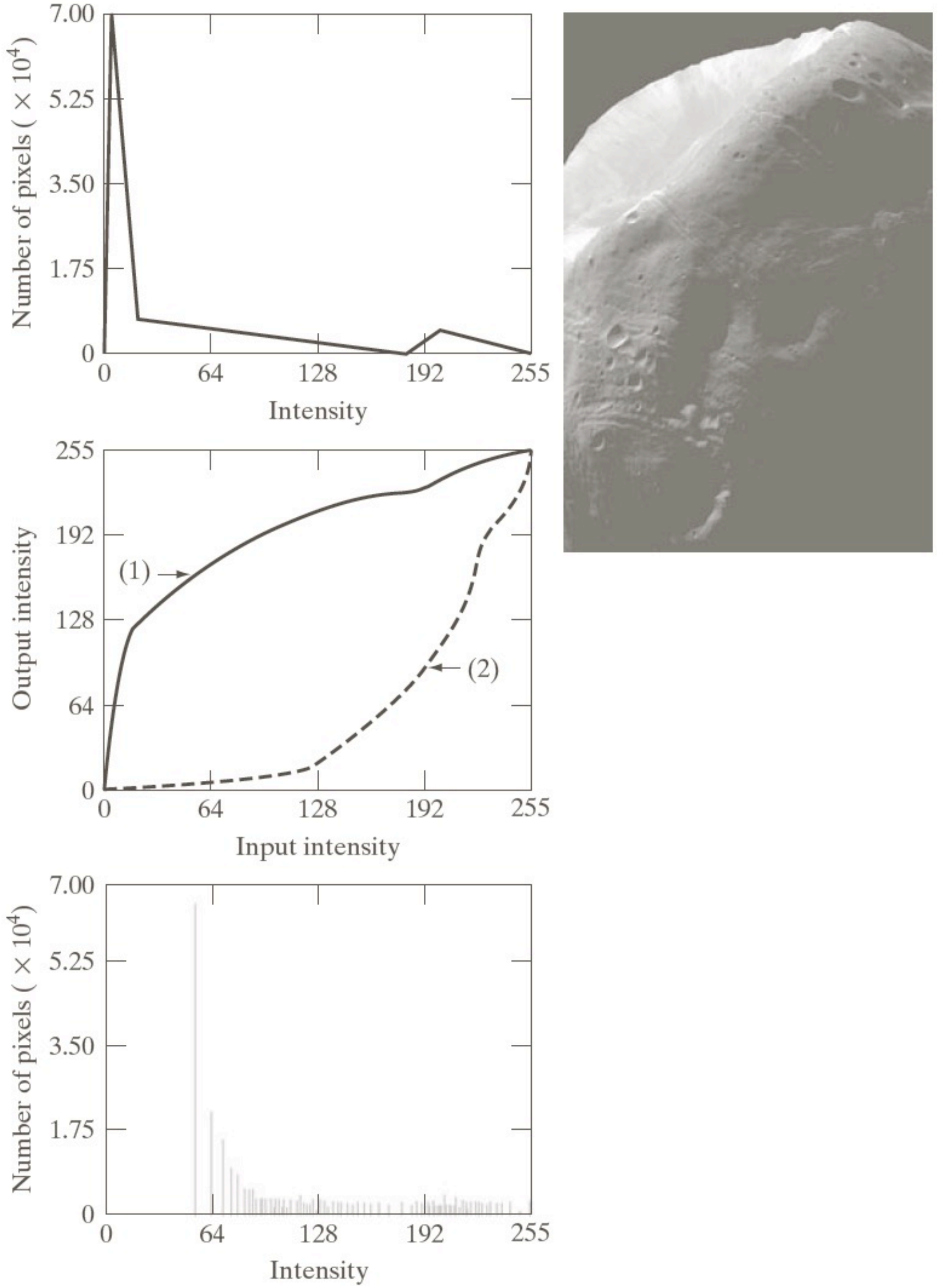
a b

FIGURE 3.23 (a) Image of the Mars moon Phobos taken by NASA's *Mars Global Surveyor.* (b) Histogram. (Original image courtesy of NASA.)



a b
c

FIGURE 3.24
(a) Transformation function for histogram equalization.
(b) Histogram-equalized image (note the washed-out appearance).
(c) Histogram of (b).

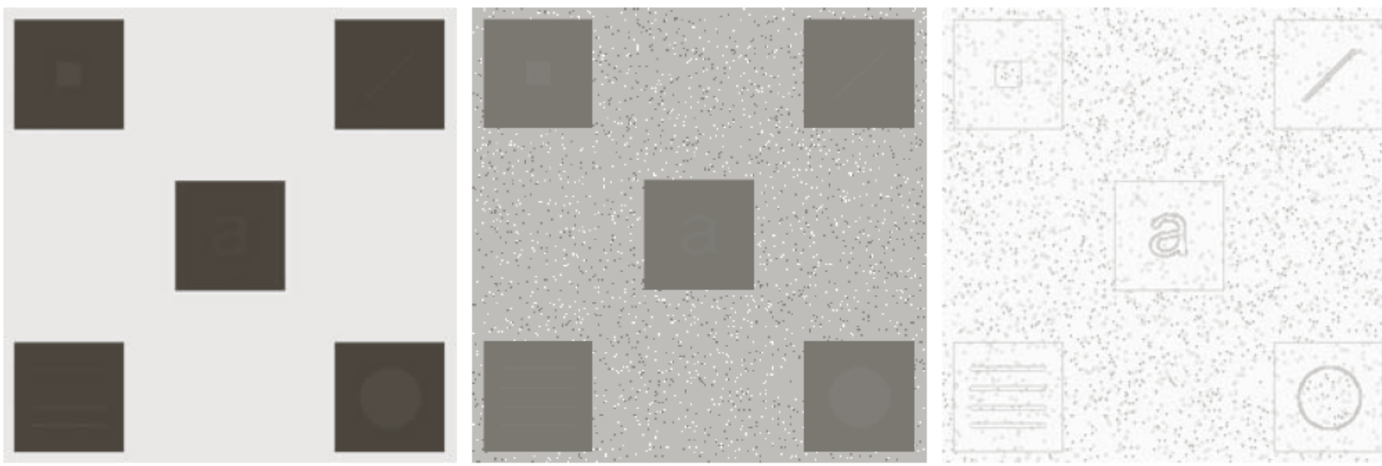


a c
b
d

FIGURE 3.25
(a) Specified histogram.
(b) Transformations.
(c) Enhanced image using mappings from curve (2).
(d) Histogram of (c).

Local histogram processing

- Compute the histogram within a local neighborhood



a b c

FIGURE 3.26 (a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization applied to (a), using a neighborhood of size 3 × 3.

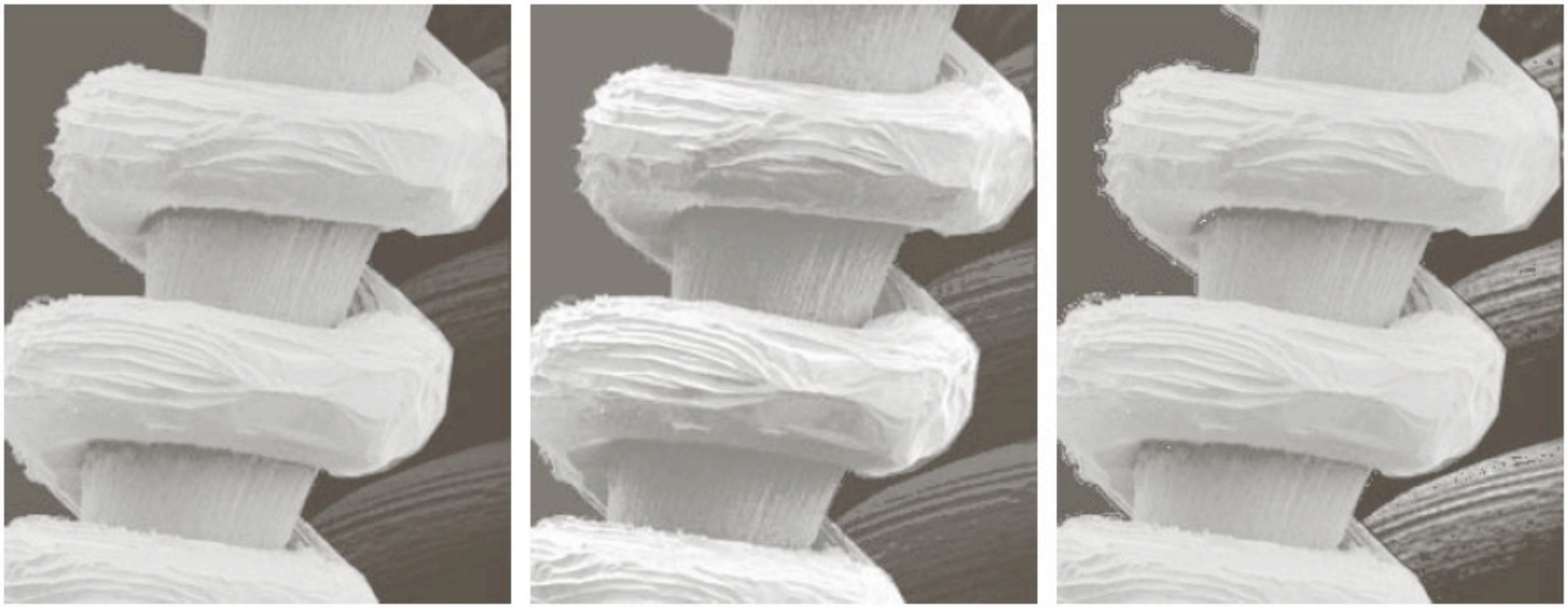
Using histogram statistics for image enhancement

- Mean and variance of intensity values are important measures for evaluating the brightness and contrast of an image respectively

$$m_{S_{xy}} = \sum_{i=0}^{L-1} r_i \, p_{S_{xy}}(r_i)$$

$$\sigma^2_{S_{xy}} = \sum_{i=0}^{L-1} (r_i - m_{S_{xy}})^2 \, p_{S_{xy}}(r_i)$$

$$g(x, y) = \begin{cases} E \cdot f(x, y) & \text{if } m_{S_{xy}} \le k_0 m_G \text{ AND } k_1 \sigma_G \le \sigma_{S_{xy}} \le k_2 \sigma_G \\ f(x, y) & \text{otherwise} \end{cases}$$



a b c

FIGURE 3.27 (a) SEM image of a tungsten filament magnified approximately 130×. (b) Result of global histogram equalization. (c) Image enhanced using local histogram statistics. (Original image courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)

Spatial filtering

- Spatial filter: the name is come from frequency domain processing. Also named as mask, template, kernel and window.
- The linear filtering operation can be expressed as

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

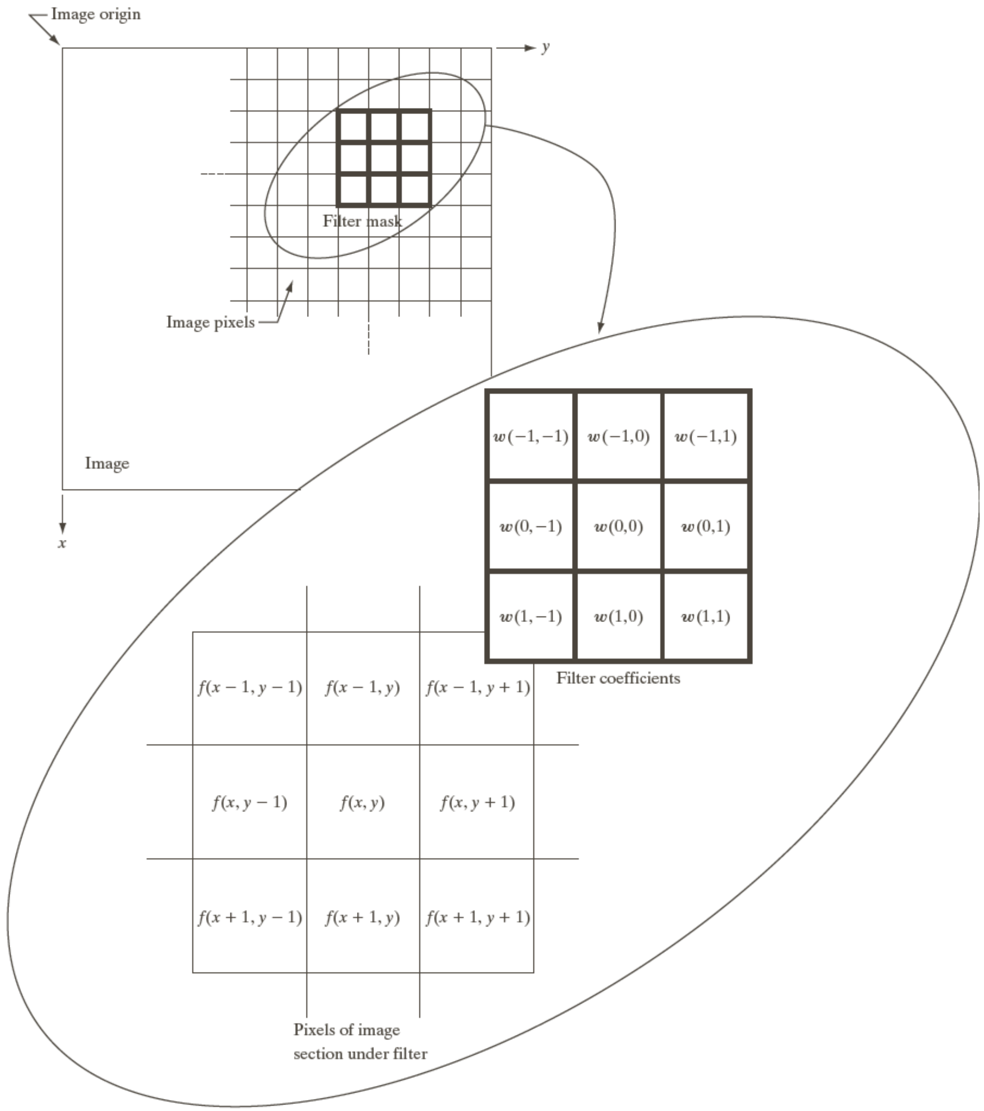


FIGURE 3.28 The mechanics of linear spatial filtering using a 3×3 filter mask. The form chosen to denote the coordinates of the filter mask coefficients simplifies writing expressions for linear filtering.

Correlation and convolution

- Correlation is the process of moving a filter mask over the image and computing the sum of products at each location. The mechanics of convolution are the same, except that the filter is first rotated by 180° .

Correlation:

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

Convolution:

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

	Correlation	Convolution	
	Origin $\quad f \quad\quad w$	Origin $\quad f \quad\quad w$ rotated 180°	
(a)	0 0 0 1 0 0 0 0 $\quad$ 1 2 3 2 8	0 0 0 1 0 0 0 0 $\quad$ 8 2 3 2 1	(i)
(b)	0 0 0 1 0 0 0 0 1 2 3 2 8 Starting position alignment	0 0 0 1 0 0 0 0 8 2 3 2 1	(j)
(c)	Zero padding 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 2 3 2 8	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 8 2 3 2 1	(k)
(d)	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 2 3 2 8 Position after one shift	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 8 2 3 2 1	(l)
(e)	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 2 3 2 8 Position after four shifts	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 8 2 3 2 1	(m)
(f)	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 2 3 2 8 Final position	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 8 2 3 2 1	(n)
	Full correlation result	Full convolution result	
(g)	0 0 0 8 2 3 2 1 0 0 0 0	0 0 0 1 2 3 2 8 0 0 0 0	(o)
	Cropped correlation result	Cropped convolution result	
(h)	0 8 2 3 2 1 0 0	0 1 2 3 2 8 0 0	(p)

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*.

Notes

- First, correlation is a function of *displacement* of the filter. In other words, the first value of correlation corresponds to zero displacement of the filter, the second corresponds to one unit displacement, and so on.
- The second thing to notice is that correlating a filter w with a function that contains all 0s and a single 1 yields a result that is a *copy* of w, but *rotated* by 180°.

Origin $f(x, y)$

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

$w(x, y)$

1	2	3
4	5	6
7	8	9

(a)

Padded f

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(b)

Initial position for w

1	2	3	0	0	0	0	0	0
4	5	6	0	0	0	0	0	0
7	8	9	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(c)

Full correlation result

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	9	8	7	0	0	0
0	0	0	6	5	4	0	0	0
0	0	0	3	2	1	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(d)

Cropped correlation result

0	0	0	0	0
0	9	8	7	0
0	6	5	4	0
0	3	2	1	0
0	0	0	0	0

(e)

Rotated w

9	8	7	0	0	0	0	0	0
6	5	4	0	0	0	0	0	0
3	2	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(f)

Full convolution result

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	2	3	0	0	0
0	0	0	4	5	6	0	0	0
0	0	0	7	8	9	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(g)

Cropped convolution result

0	0	0	0	0
0	1	2	3	0
0	4	5	6	0
0	7	8	9	0
0	0	0	0	0

(h)

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.

Smoothing filters

- For noise reduction and image blurring
  - **Salt and pepper noise**: random occurrences of black and white pixels
  - **Impulse noise**: random occurrences of white pixels
  - **Gaussian noise**: variations in intensity drawn from a Gaussian normal distribution

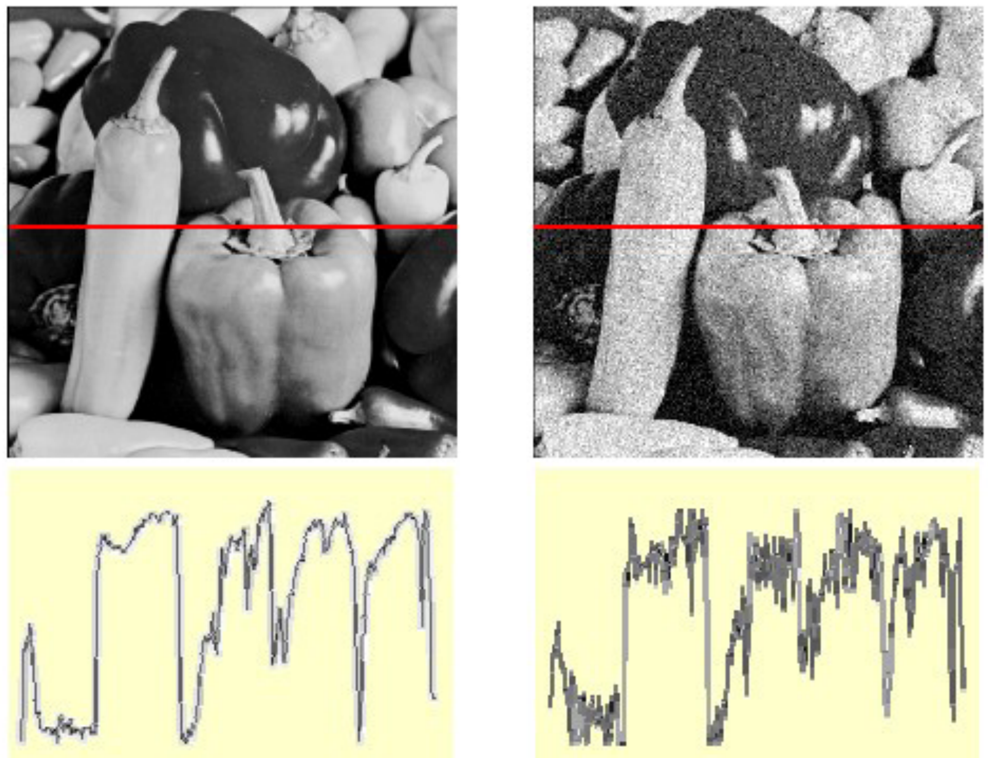

Original


Salt and pepper noise


Impulse noise


Gaussian noise

Gaussian noise

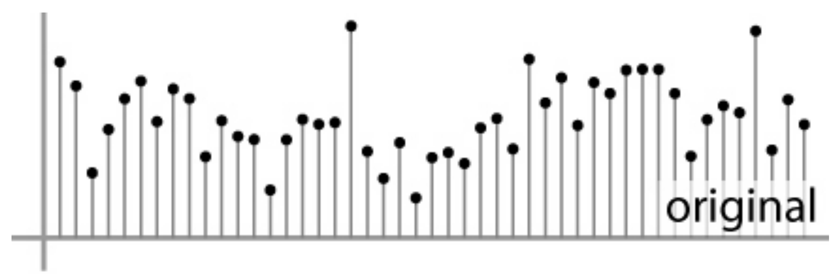


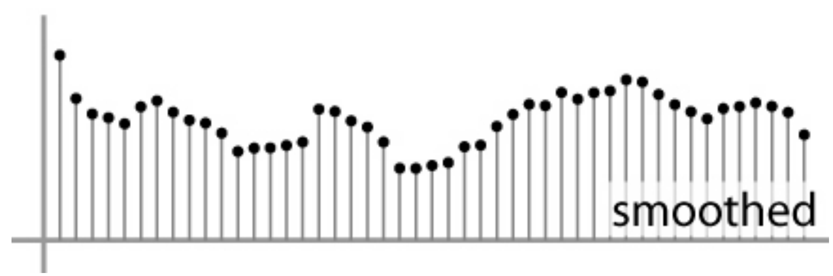
$$f(x,y) = \overbrace{\hat{f}(x,y)}^{\text{Ideal Image}} + \overbrace{\eta(x,y)}^{\text{Noise process}}$$

Gaussian i.i.d. ("white") noise:
$\eta(x,y) \sim \mathcal{N}(\mu, \sigma)$

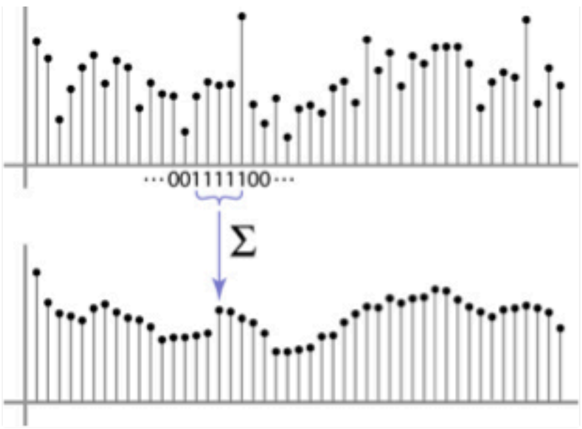
How could we reduce the noise?

- Let's replace each pixel with an average of all the values in its neighborhood
- Moving average in 1D

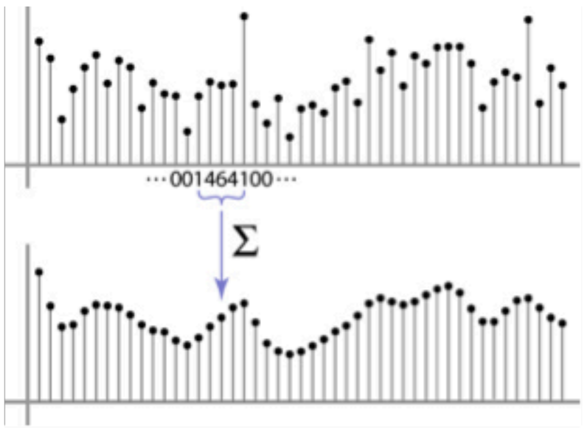




Weighted moving average



Weights [1, 1, 1, 1, 1] / 5



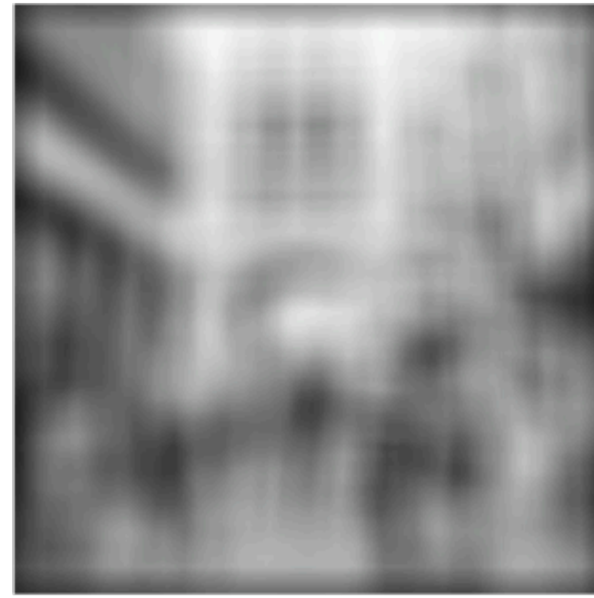
Non-uniform weights [1, 4, 6, 4, 1] / 16

Box filter

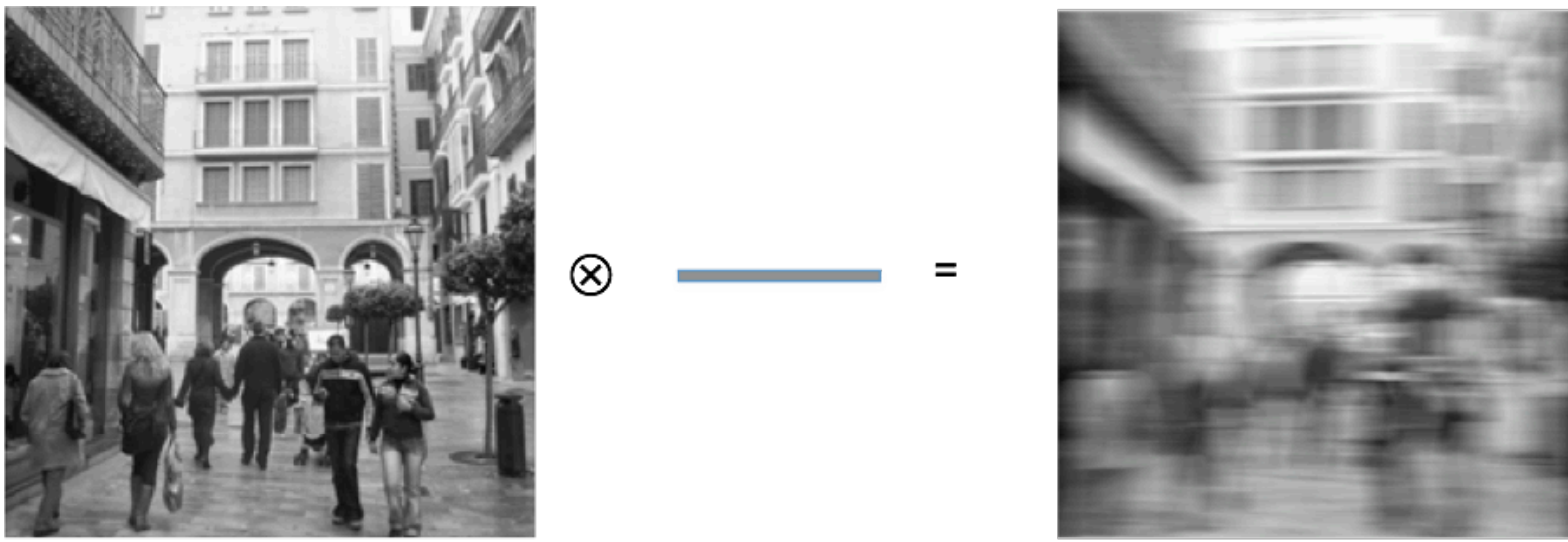
$$\frac{1}{K^2}\begin{array}{|c|c|c|c|}\hline 1 & 1 & \cdots & 1 \\ \hline 1 & 1 & \cdots & 1 \\ \hline \vdots & \vdots & 1 & \vdots \\ \hline 1 & 1 & \cdots & 1 \\ \hline \end{array}$$

$\otimes$

$=$



Antonio Torralba



Antonio Torralba

Antonio Torralba

Integral image

- Summed area table
- Benefits:
  - **fast**: for repeatedly convolved with different box filters

3	2	7	2	3
1	5	1	3	4
5	1	**3**	5	1
4	3	2	1	6
2	4	1	4	8

(a) S = 24

3	5	12	14	17
4	*11*	**19**	24	31
9	**17**	28	38	46
13	24	37	48	62
15	30	44	59	81

(b) s = 28

3	5	12	*14*	17
4	11	19	24	31
9	17	28	38	46
13	24	37	**48**	62
15	30	44	59	81

(c) S = 24

$$s(i,j) = \sum_{k=0}^{i} \sum_{l=0}^{j} f(k,l)$$

$$s(i,j) = s(i-1,j) + s(i,j-1) - s(i-1,j-1) + f(i,j)$$

$$S(i_0 \ldots i_1, j_0 \ldots j_1) = \sum_{i=i_0}^{i_1} \sum_{j=j_0}^{j_1} s(i_1,j_1) - s(i_1, j_0 - 1) - s(i_0 - 1, j_1) + s(i_0 - 1, j_0 - 1)$$

Bilinear filter (tent function)

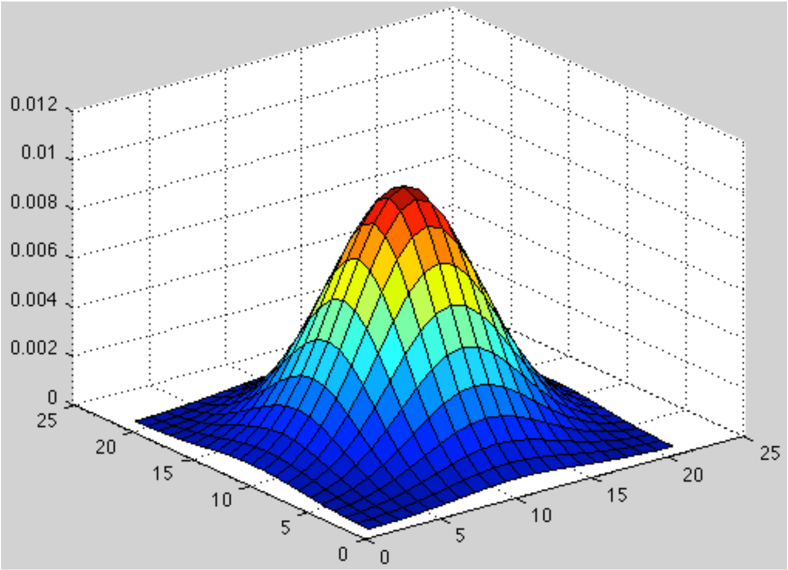
- Outer product of two linear splines

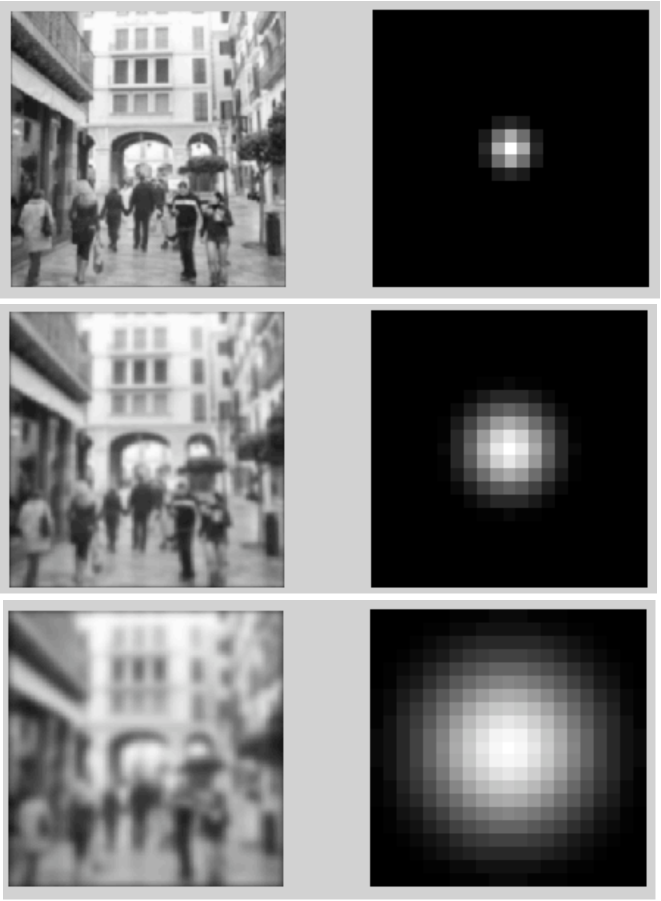
$$\frac{1}{16}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

$$\frac{1}{4}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline \end{array}$$

Gaussian filter

- Rotational symmetric

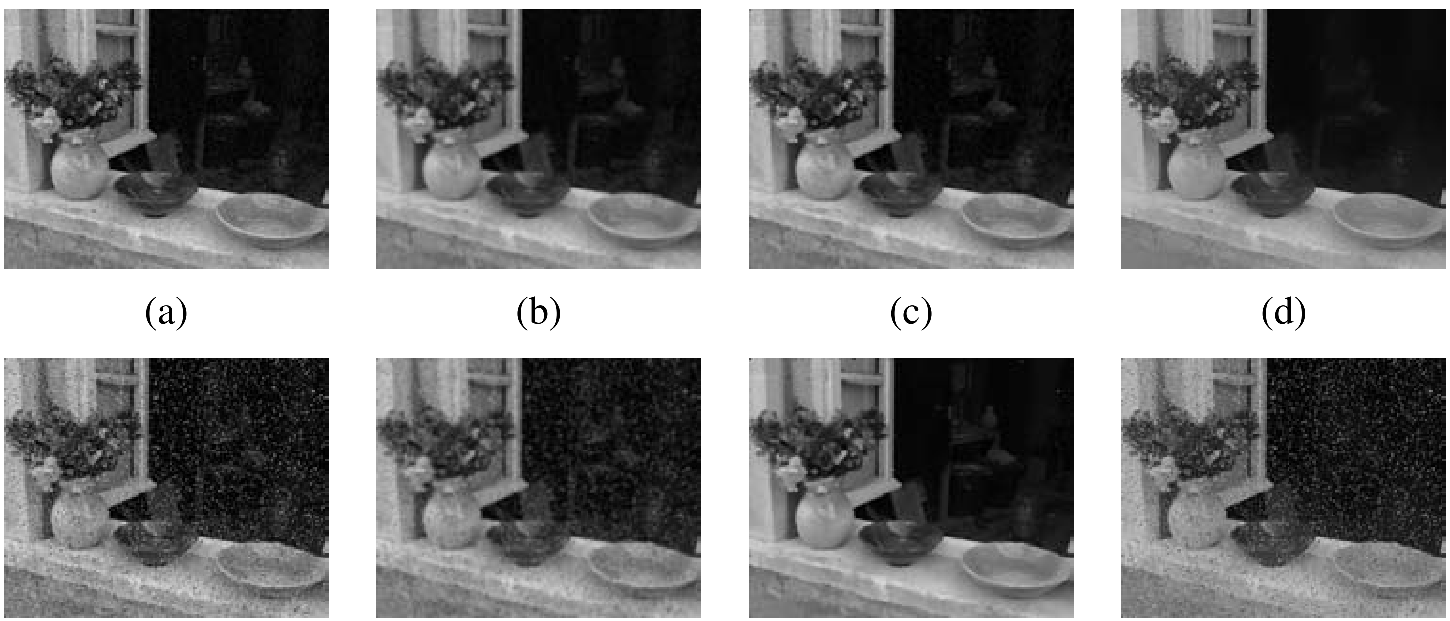




$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Non-linear filters

- Sometimes non-linear filtering can achieve better performance than linear.



(a) (b) (c) (d)

(e) (f) (g) (h)

Median filter

- Select the median value from each pixel's neighborhood
- Very efficient to remove salt-and–pepper noise
- Variants: α-trimmed mean, weighted median

1	2	1	2	4
2	1	3	5	8
1	3	7	6	9
3	4	8	6	7
4	5	7	8	9

(a) median = 4

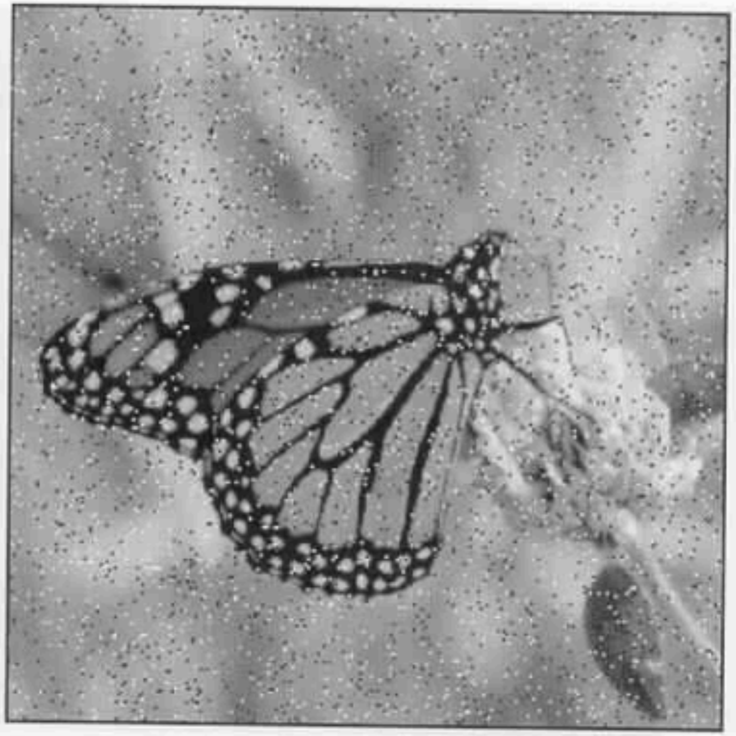
1	2	1	2	4
2	1	3	5	8
1	3	7	6	9
3	4	8	6	7
4	5	7	8	9

(b) α-mean= 4.6

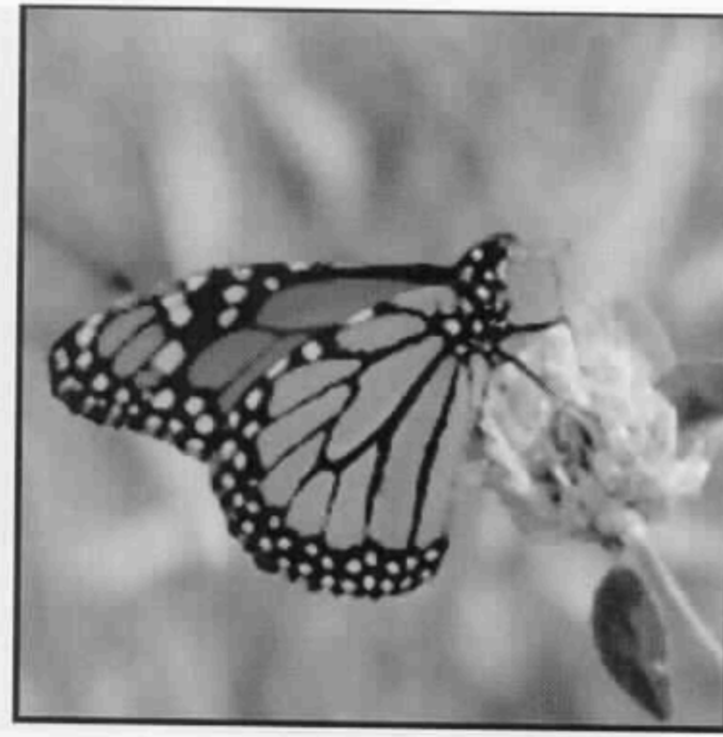
Similar non-linear filters

- Minimum filter (remove salt noise)
- Maximum filter (remove pepper noise)
- Midpoint filter (remove Gaussian and uniform noise)

$$\hat{f}(x, y) = \frac{1}{2}\left[\max_{(s,t)\in S_{xy}} \{g(s, t)\} + \min_{(s,t)\in S_{xy}} \{g(s, t)\}\right]$$



a. Original image with added salt-and-pepper noise.



b. Median-filtered image using a 3 × 3 kernel.

Median filter on salt-and-pepper noise corrupted image

a. Image with salt noise; probability of salt = .04.

b. Result of minimum filtering image (a); mask size = 3 × 3.

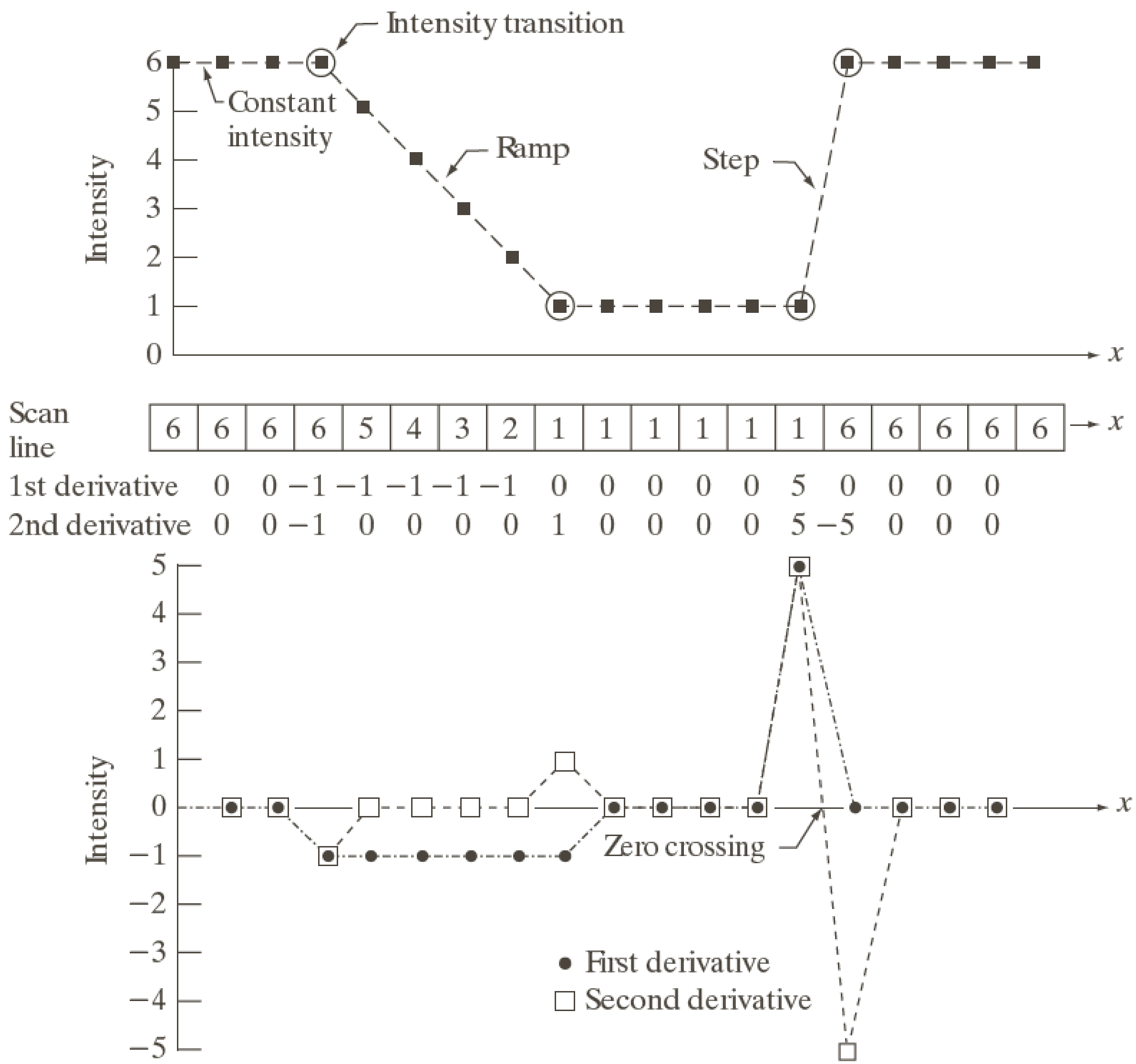
Minimum filter on salt noise corrupted image

Image sharpening

- In contrast to the integration operation for image smoothing, image sharpening works on differentiation. For example, the 1-dim situation:

First order: $$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

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



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.

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.

$$\nabla^2 f(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y) \tag{3.6-6}$$

Sharpening using Laplacian

- Adding the Laplacian image to the original image, we can get the sharpened image

$$g(x, y) = f(x, y) + c\left[\nabla^2 f(x, y)\right]$$

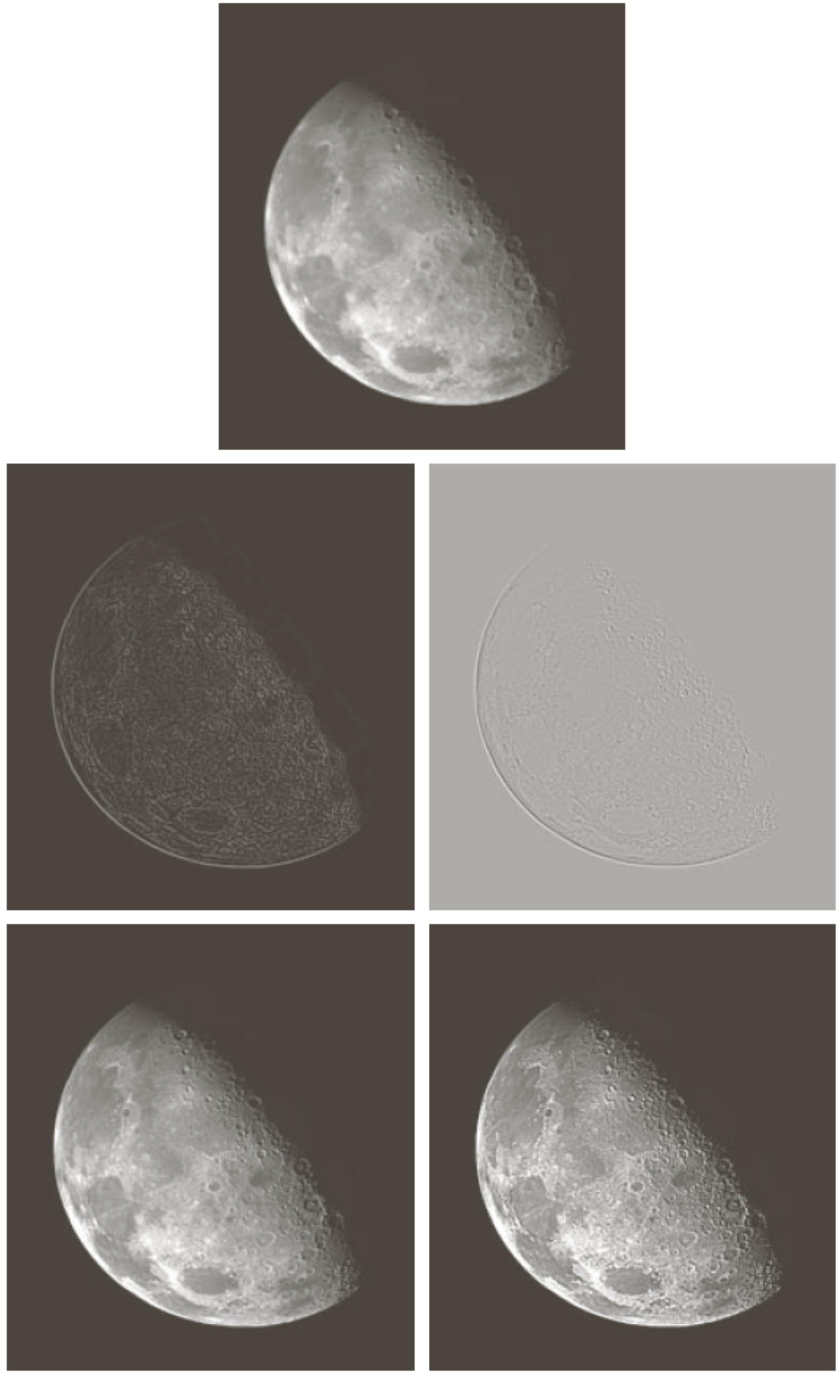


FIGURE 3.38 (a) Blurred image of the North Pole of the moon. (b) Laplacian without scaling. (c) Laplacian with scaling. (d) Image sharpened using the mask in Fig. 3.37(a). (e) Result of using the mask in Fig. 3.37(b). (Original image courtesy of NASA.)

Sharpening using highboost filtering

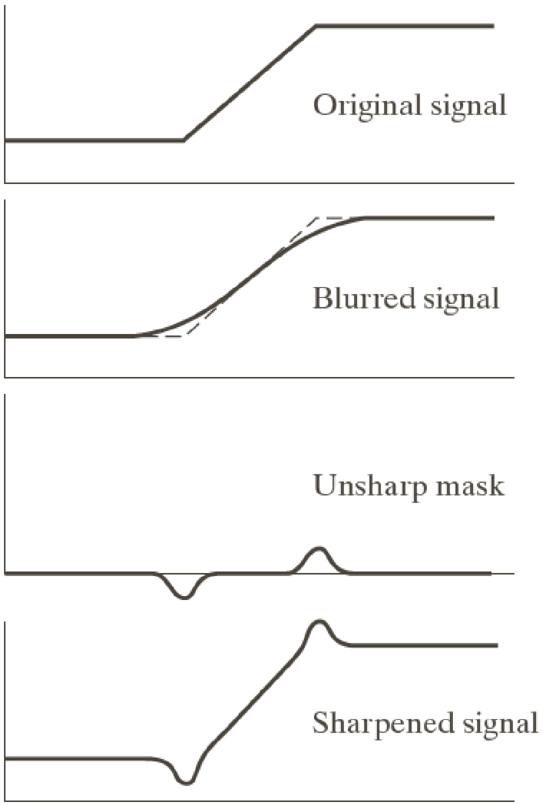
① Blur the original image.

② Subtract the blurred image from the original (the resulting difference is called the *mask*.)

③ Add the mask to the original.

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

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



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.