MT390 (DIP): Tutorial 2

Chapter 3: Intensity Transformations and Spatial Filtering

Introduction:

Image Enhancement?

☞ **Enhancement تحسين الصورة**: is to process an image so that the result is more suitable than the original image for a *specific* application.

Enhancement techniques fall into 2 types:

- Spatial domain: direct manipulation of pixels in the image plane
- Frequency domain: modifying Fourier transform of the image.

☞ *In this chapter, we are going to discuss spatial domain techniques*

Preview

Spatial Domain:

- Refers to Image Plane
- Direct Manipulation of Pixels

2 Categories of Spatial Domain Processing:
- Intensity (gray-level) Transformations
- Spatial Filtering (Neighborhood Processing or Spatial Convolution)

3.1 Background

3.1.1 The Basics of Intensity Transformations and Spatial Filtering

Spatial domain: aggregate of pixels composing an image

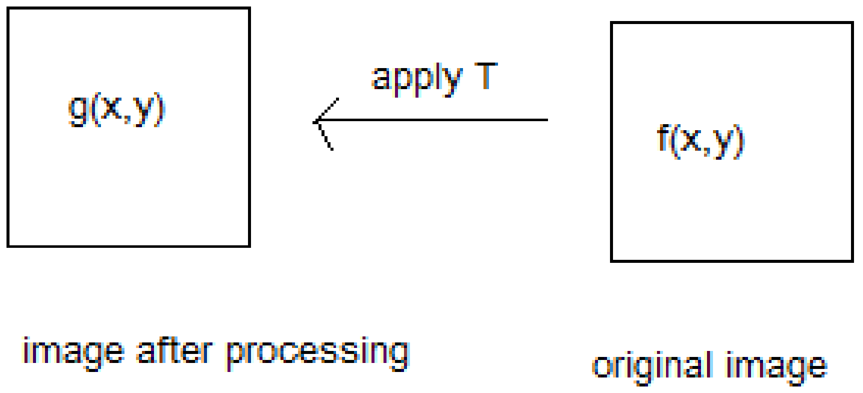
Spatial domain processes:

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

Processed (output) image

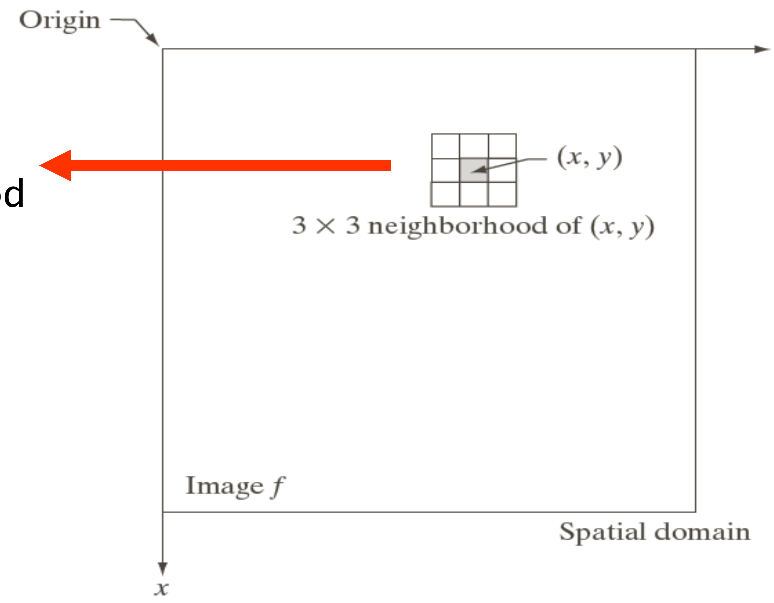
Operator T defined on some neighborhood of f(x,y)

Input image



Defining a neighborhood (T) :

This rectangle is called neighborhood or mask



Operator T is applied at each location (x,y) to produce the output g at that location.

types of neighborhood:

1. **intensity transformation:** neighborhood of size 1x1
2. **spatial filter** (or mask ,kernel, template or window): neighborhood of larger size , like in the above example.

Spatial Filtering:

- It is the **procedure** of moving the location of neighbourhood and performing a predefined operation.
- The neighbourhood, along with the predefined operation is called a **spatial filter**.
- **Other names** for spatial filter are spatial mask, kernel, template or window.

Intensity transformation function:

- Let the neighbourhood be of size 1x1 (that is, a single pixel).
- In this case, g depends only on the value of f at (x, y), and **T becomes a intensity** (also called a gray-level or mapping) transformation function of the form:

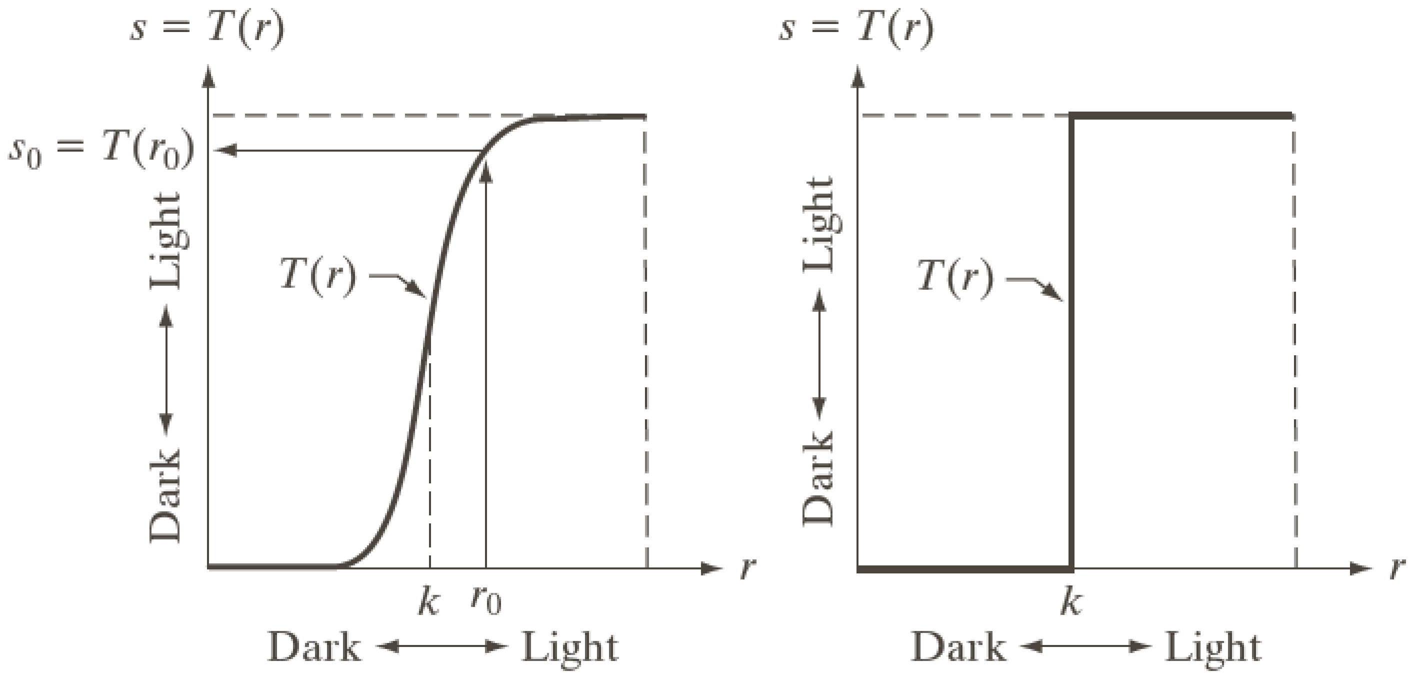
$$\underbrace{g(x,y)}_{s} = T[\underbrace{f(x,y)}_{r}]$$

s=T(r)

- where, for simplicity in notation, r and s are variables denoting, respectively, the gray level of f(x, y) and g(x, y) at any point (x, y).

a b

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



Example of Fig 3.2:

- **Contrast stretching 3.2(a):** An image of higher contrast than the original is produced by darkening the levels **below k** and brightening the levels **above k** in the original image.
- Values of r below k are compressed by the transformation function into a narrow **range** of s, toward black.
- The opposite effect takes place for values of r above k.

- **Thresholding 3.2(b):** In the limiting case, T(r) produces a **two-level** (binary) image.

About Examples in this Chapter

- **Image Enhancement**.
- It is the process of **manipulating** an image so that the result is **more suitable** than original.
- There is **no “general theory**” of Image Enhancement.

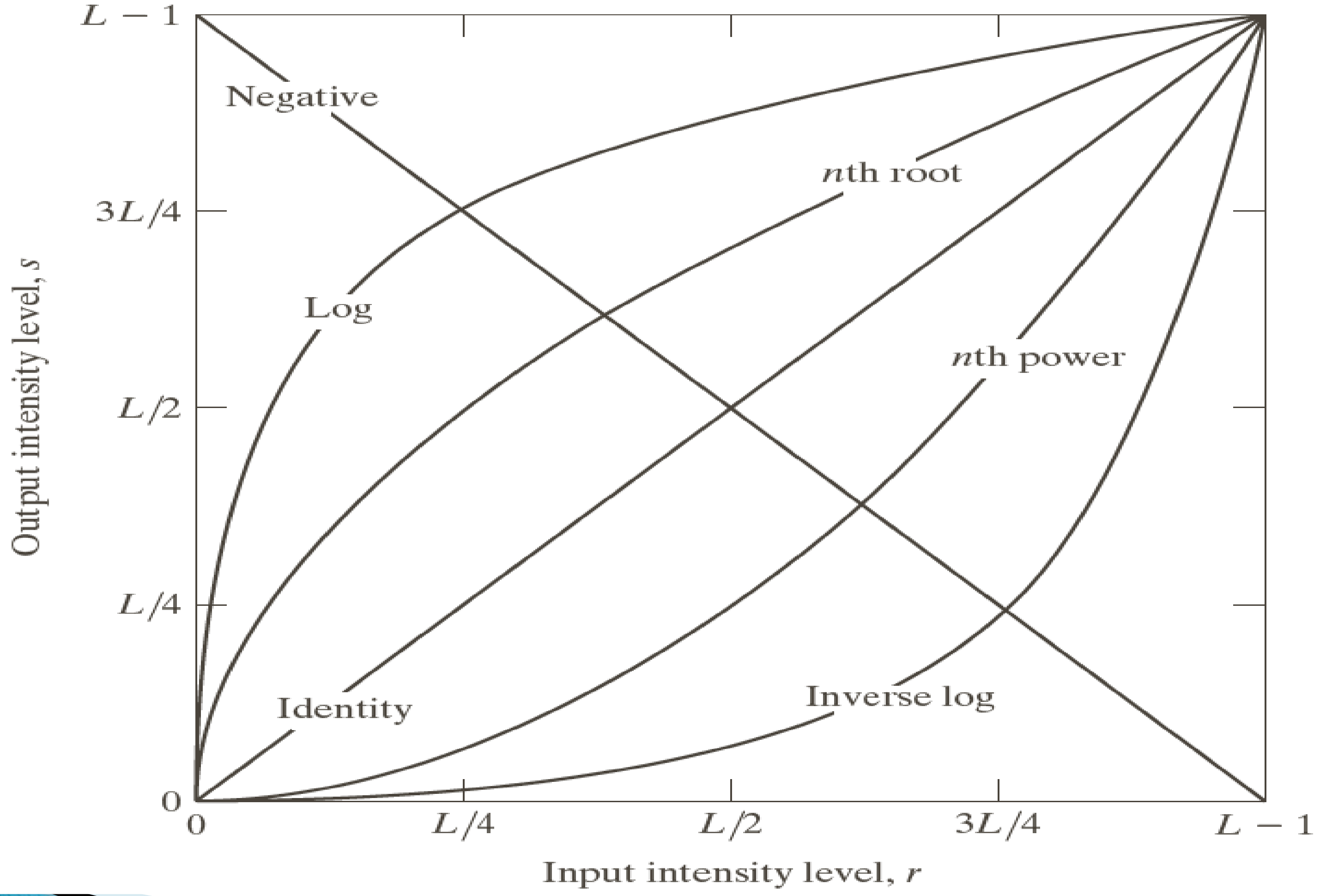
3.2 Some Basic Intensity Transformation Functions

Intensity transformation functions fall into 2 approaches:

1) **Basic intensity transformations**
   - a) Linear (negative and identity).
   - b) logarithmic (Log and Inverse Log) .
   - c) Power(nth power and nth root).

2) piecewise Linear transformation functions.
   - a) Contrast stretching, thresholding
   - b) intensity-level slicing
   - c) Bit-plane slicing

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



3.2.1 Linear (Image Negatives)

- The negative of an image with gray levels in the range [0,L-1]is obtained by using the negative transformation shown in Fig. 3.3, which is given by the **expression**: $s = L - 1 - r.$
- **Reversing** the intensity levels of an image in this manner produces the equivalent of a photographic negative.

Image (r)

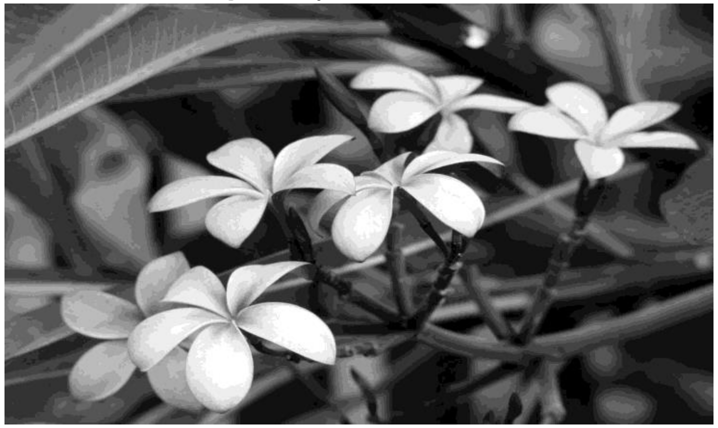
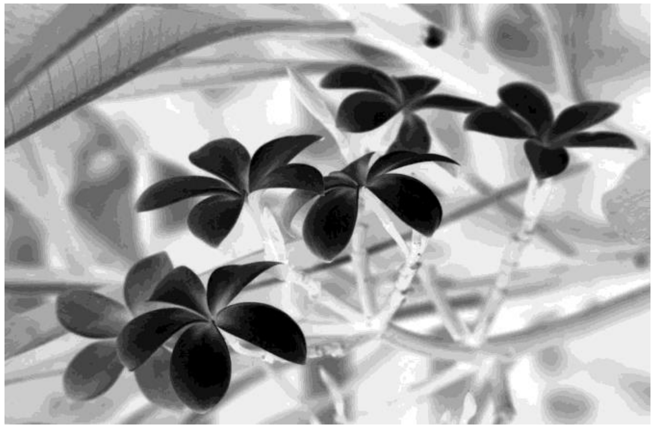


Image (s) after applying T (negative)



<u>Advantages of negative</u> :
✓ Produces an equivalent of a photographic negative.
✓ Enhances white or gray detail embedded in dark regions.

Illustrations on linear (negative images):

The negative of an image with intensity levels in the range [0,L-1] is obtained by using the negative transformation :

$$s = L-1-r$$

Example
the following matrix represents the pixels values of an 8-bit image (r) , apply negative transform and find the resulting image pixel values.

Image (r)

100	110	90	95
98	140	145	135
89	90	88	85
102	105	99	115

solution:
$L = 2^8 = 256$

$s=L-1-r$
$s = 255-r$

Apply this transform to each pixel to find the negative

Image (s)

155	145	165	160
157	115	110	120
166	165	167	170
153	150	156	140

Activity on obtaining negative image:

Exercise:

the following matrix represents the pixels values of a 5-bit image (r) , apply negative transform and find the resulting image pixel values.

Image (r)

21	26	29	30
19	21	20	30
16	16	26	31
19	18	27	23

solution:

Image (s)

Example 2

- Apply Negative IT to the 5-bit image of size 3x3 given below:

28	5	4
16	8	26
11	15	7

Ans.:

S=L-1-r

S1=32-1-28=31-28=3

S2=32-1-5=31-5=26

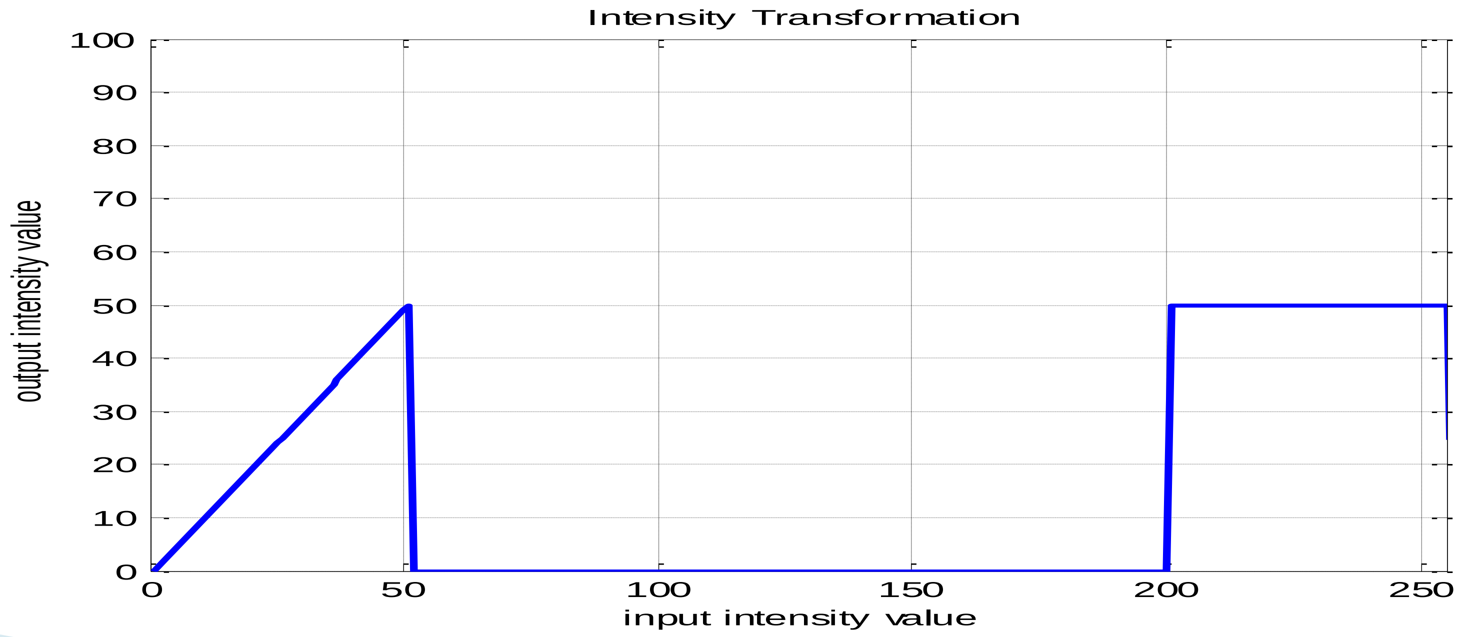
3	26	

Example 3

Consider the image given above

45	210	110
33	78	220
116	150	13

Apply the following intensity transformation to the image given above



Ans.:

You should apply the IT and find the result.

45	50	

Example 4: Given the following image:

45	210	110
33	78	220
116	150	13

The given Intensity Transformation is:

$$s = T(r) = \begin{cases} 10 & 0 \leq r \leq 50 \\ 70 & 51 \leq r \leq 170 \\ 20 & 171 \leq r \leq 255 \end{cases}$$

Ans.: **You should apply the IT and find the result.**

10	20	

Log Transformations

- The general form is:

$$s = c\log(1 + r)$$

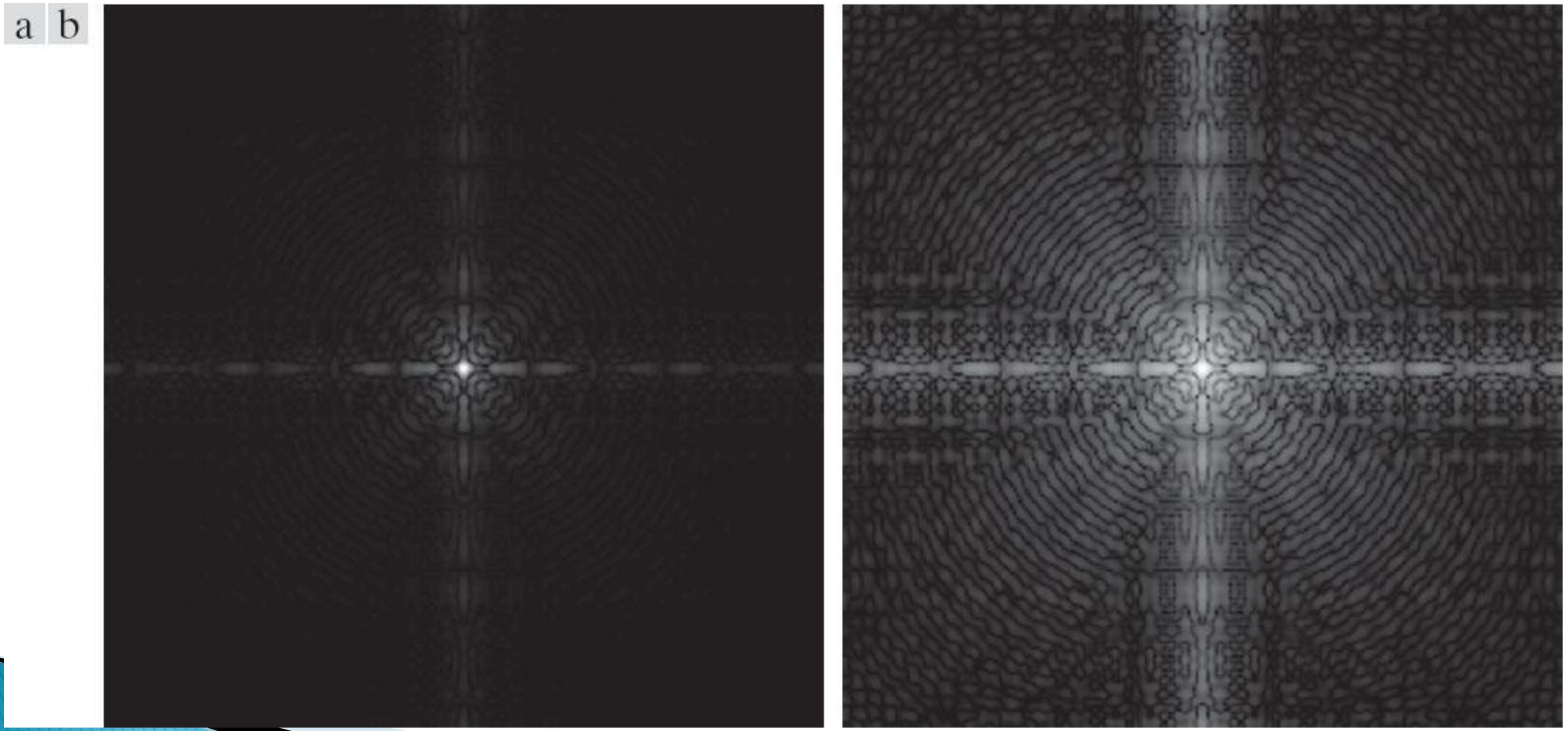
- where *c is a constant, and it is assumed that* $r \geq 0$.
- *It* maps a narrow range of low gray-level values in the input image into a wider range of output levels.
- The opposite is true of higher values of input levels.
- Used to expand the values of dark pixels in an image while compressing the higher-level values.
- The opposite is true of the inverse log.
- The power-law transformations are much more versatile (useful) for this purpose than the log transformation

<u>Logarithmic Transformations Usage:</u>

- To compress dynamic range
- For example Fourier spectrum can have values in range [0 10^6]
- By using the log, the dynamic range is reduced to about 14.

Figure 3.5

(a) Fourier spectrum displayed as a grayscale image.
(b) Result of applying the log transformation in Eq. (3-4) with **c** = 1. Both images are scaled to the range [0, 255].



Example:

Assuming log base10 (widely available on calculators) and c=1, apply the Logarithmic IT Transformation to the following 3x3 image and find the output.

5	1	7
3	2	7
5	4	0

Answer:

S1=1. log10(1+5)=.7781

S2=1. log10(1+1)=.3010

.

.

.7781	.3010	

Power-Law Transformations

- Power-law transformations have the basic **form**

$$s = cr^{\gamma}$$

- Where c and gamma are positive constants.
- Plots of *s versus r for various* values of gamma are shown in **Fig. 3.6**.
- As in the case of the log transformation, power-law curves with fractional values of gamma map a **narrow range** of dark input values into a **wider range** of output values, with the opposite being true for higher values of input levels.

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.

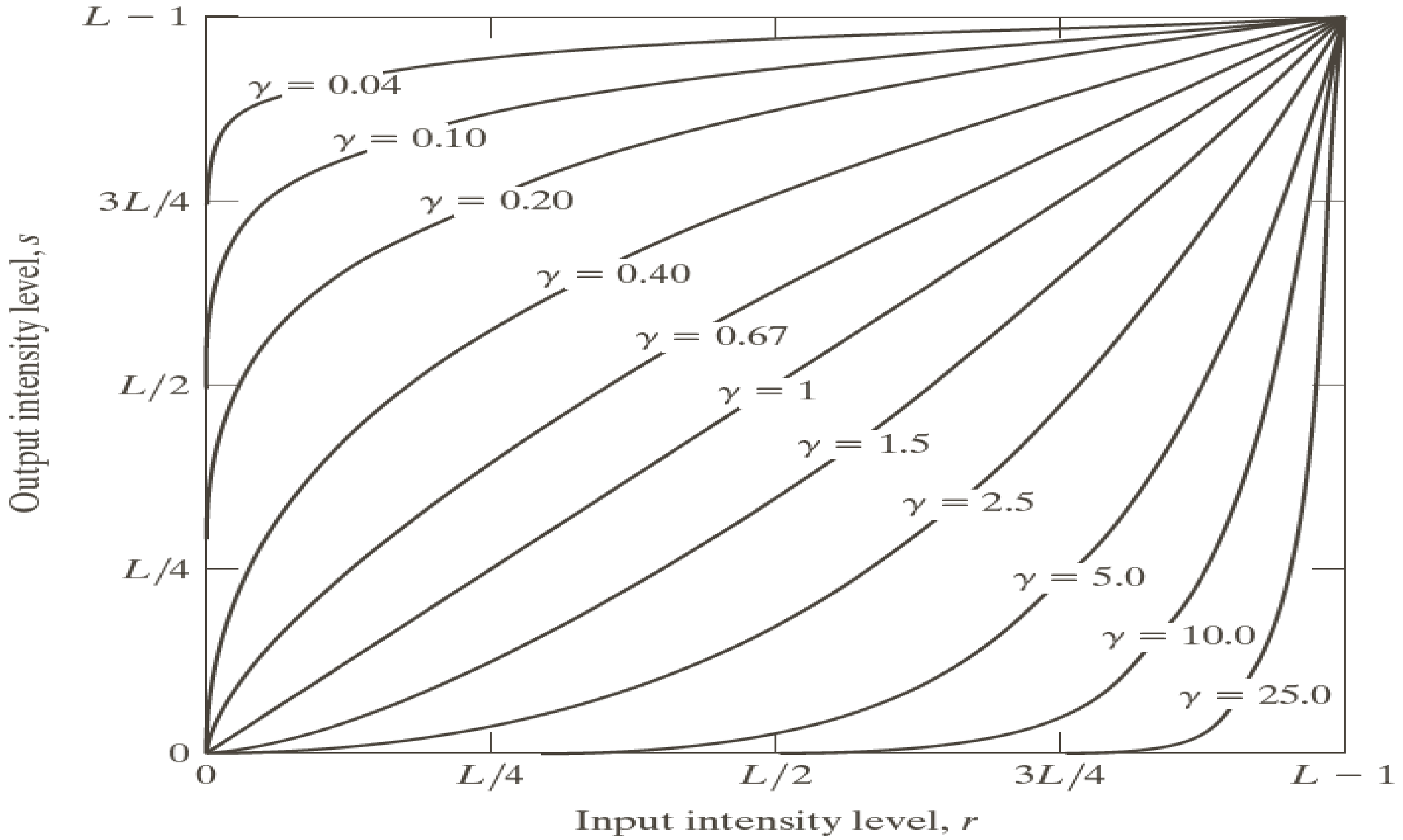


Figure 3.7

(a) Image of a human retina. (b) Image as it appears on a monitor with a gamma setting of 2.5 (note the darkness). (c) Gamma-corrected image. (d) Corrected image, as it appears on the same monitor (compare with the original image). (Image (a) courtesy of the National Eye Institute, NIH)

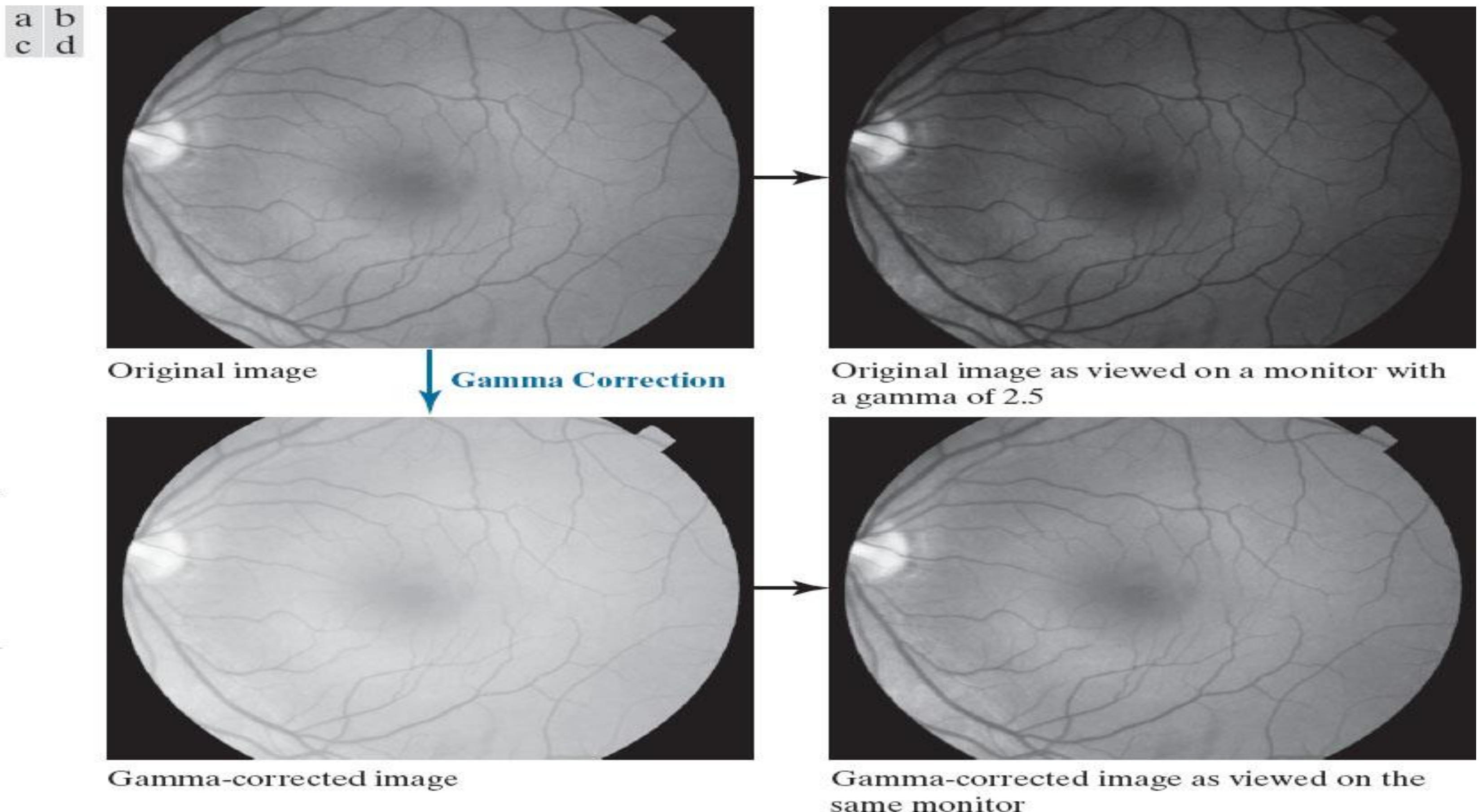

a b
c d
Original image
Gamma Correction
Original image as viewed on a monitor with a gamma of 2.5
Gamma-corrected image
Gamma-corrected image as viewed on the same monitor

Figure 3.8

(a) Magnetic resonance image (MRI) of a fractured human spine (the region of the fracture is enclosed by the circle). (b)–(d) Results of applying the transformation in Eq. (3-5) with c = 1and $\gamma = 0.6, 0.4,$

and 0.3, respectively. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

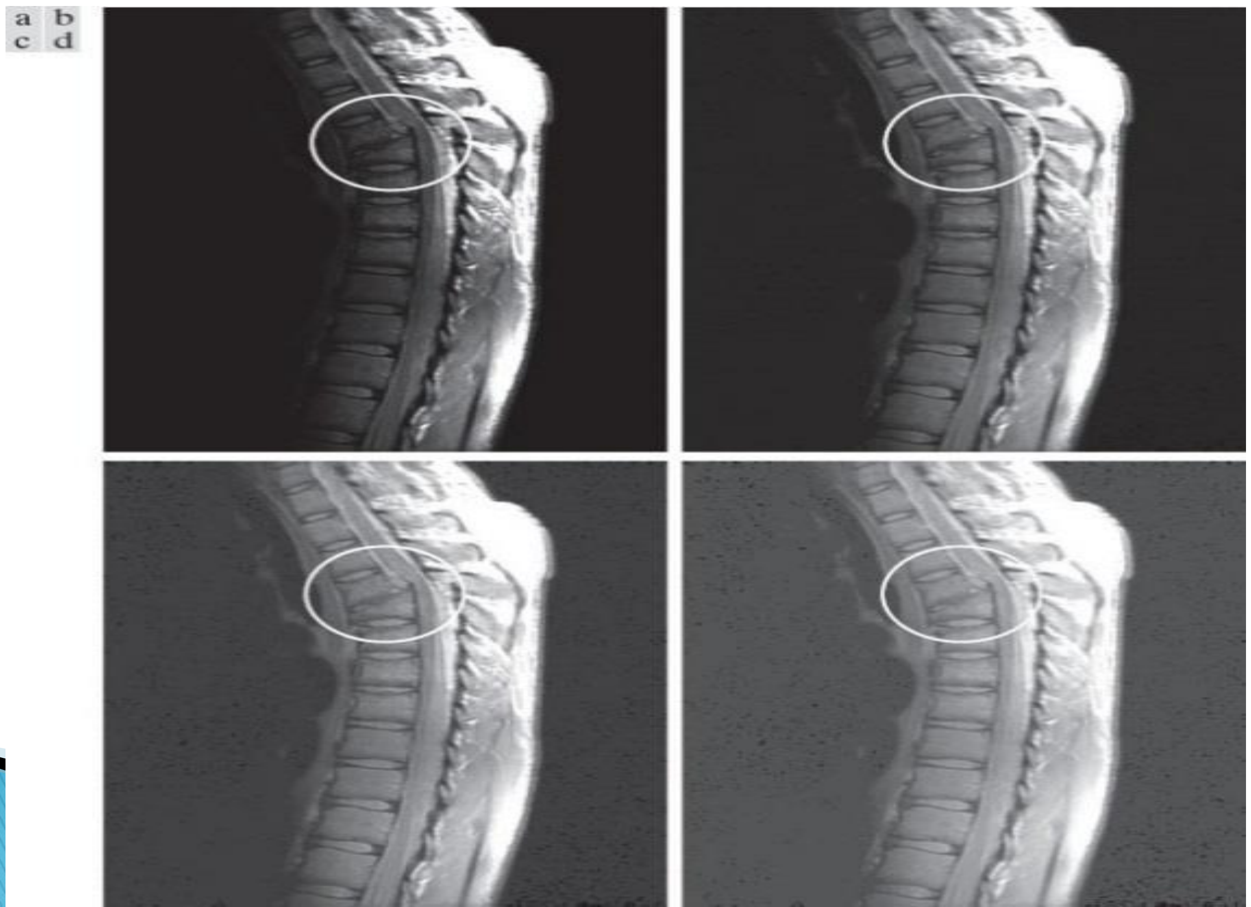
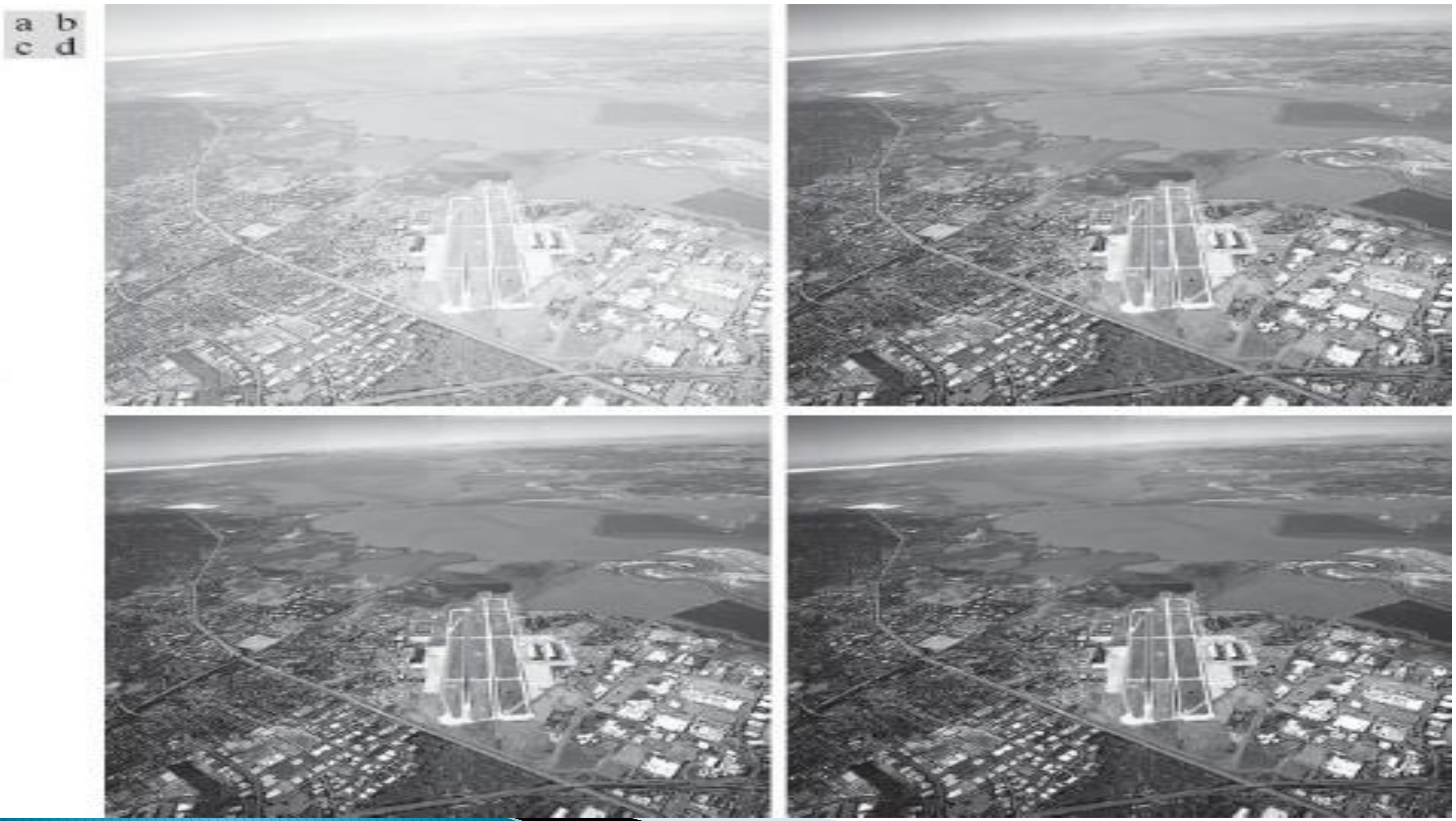


Figure 3.9

(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3–5) with $\gamma = 3.0, 4.0, \text{and } 5.0,$ respectively. (c = 1 in all cases.)

(Original image courtesy of NASA.)



Piecewise-Linear Transformation Functions

- It is **Piecewise-Linear** (Fig 3.10a).
- 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 (CS):

- One of the simplest **piecewise linear** functions is a contrast-stretching transformation.
- **Low-contrast** images can result from poor illumination, lack of dynamic range in the imaging sensor, or even wrong setting of a lens aperture during image acquisition.
- The idea behind contrast stretching is to **increase** the **dynamic range** of the gray levels in the image being processed.
- **Fig 3.10.**

CS:

a	b
c	d

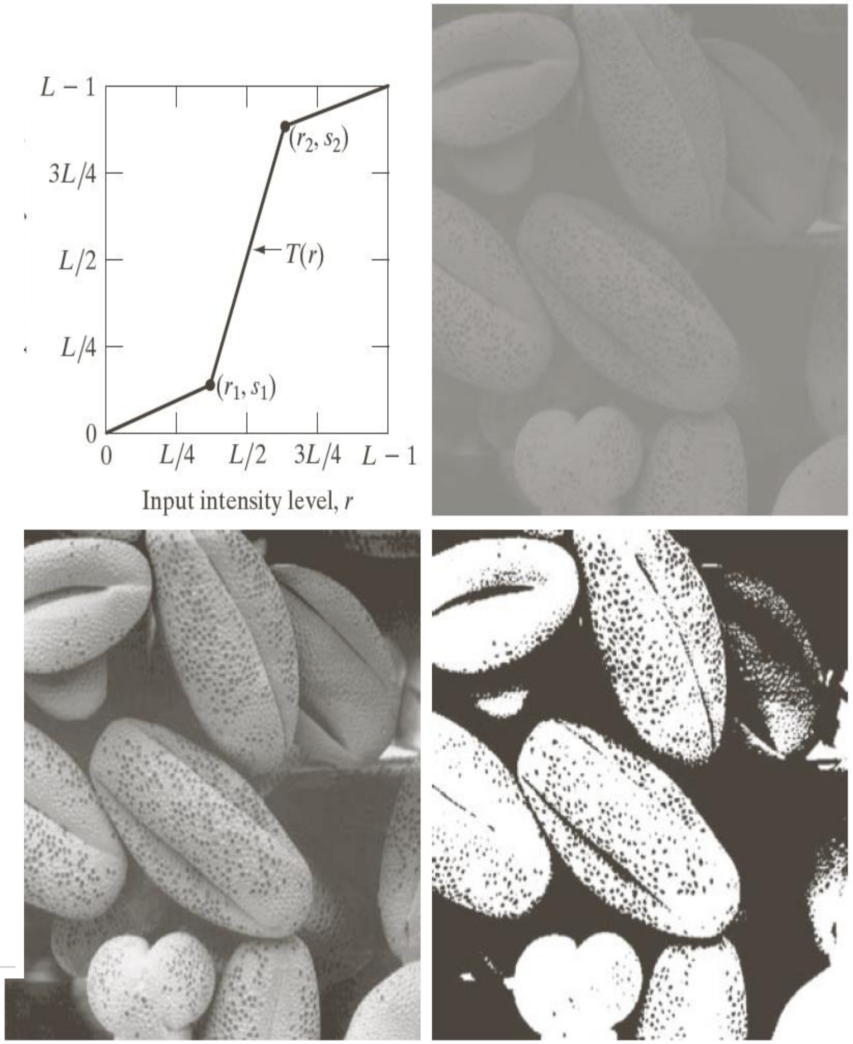


FIGURE 3.10 Contrast stretching. (a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

Contrast-Stretching Transformations:

- It expands a narrow range of input values into a wide (stretched) range of output.
- Result is an image of Higher Contrast.
- In the limiting case it results in **Binary** Image.
- This Limiting function is called **Thresholding** Function. See Figure below:

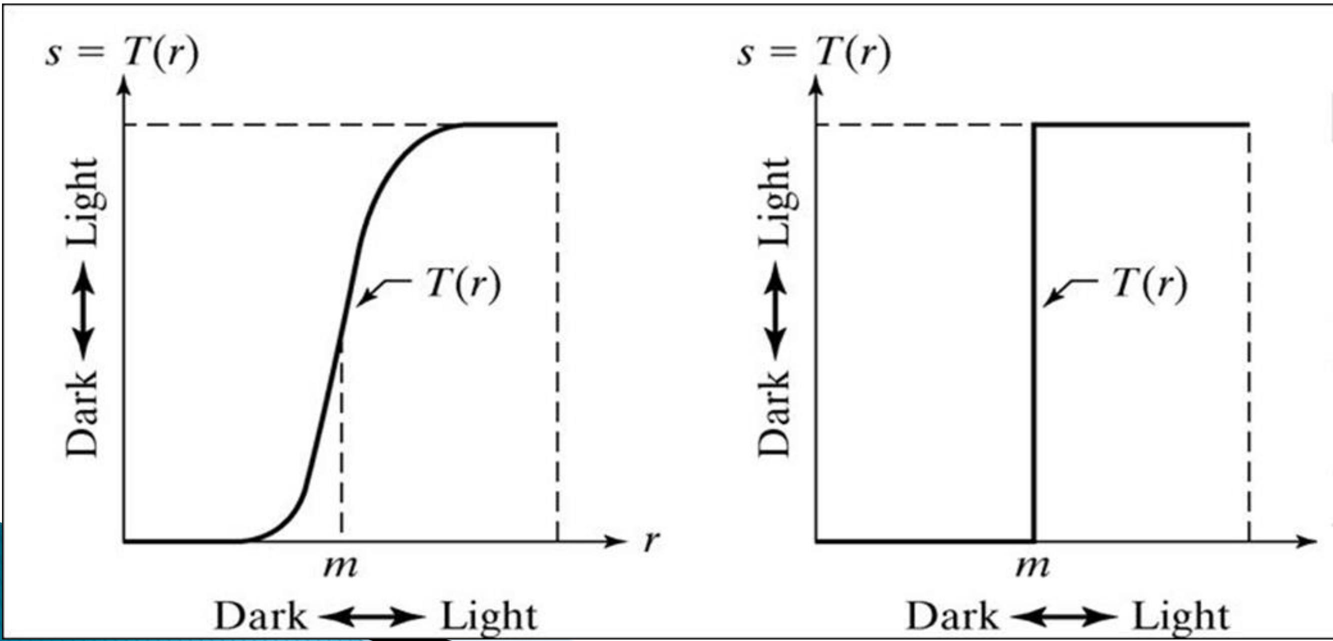


FIGURE 3.4 (a) Contrast-stretching transformation. (b) Thresholding transformation.

Example on CS:

Remember that:

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

Or

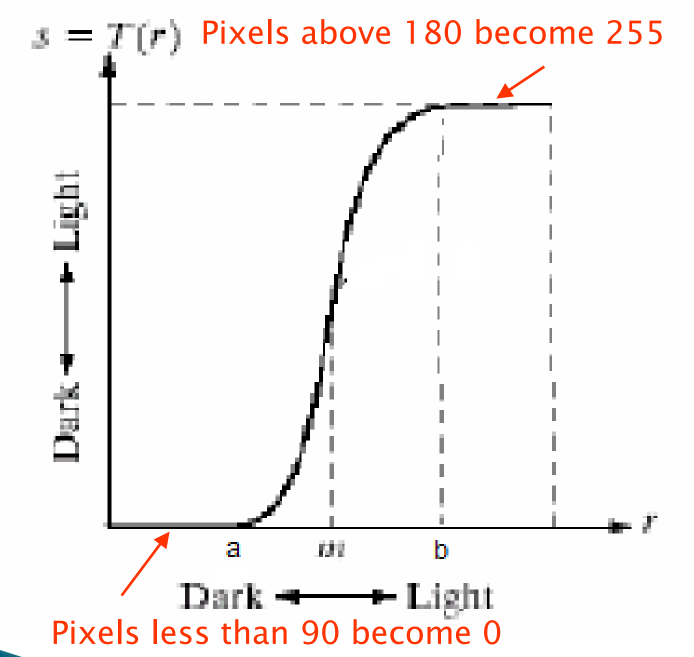
$s = T(r)$

Example: in the graph, suppose we have the following intensities : a=90, b=180, m=100

- ✓ if r is above 180 ,it becomes 255 in s.
- ✓ If r is below 90 , it becomes 0,
- ✓ If r is between 90, 180 , T applies as follows:

when r < 100 , s closes to zero (darker)

when r>100 , s closes to 255 (brighter)



$$T = \begin{cases} \text{If } r > 180;\ s = 255 \\ \text{If } r < 180 \text{ and } r > 90;\ s = T(r) \\ \text{If } r < 90;\ s = 0 \end{cases}$$

This is called contrast stretching, which means that the bright pixels in the image will become brighter and the dark pixels will become darker, this means : higher contrast image.

Image (r)

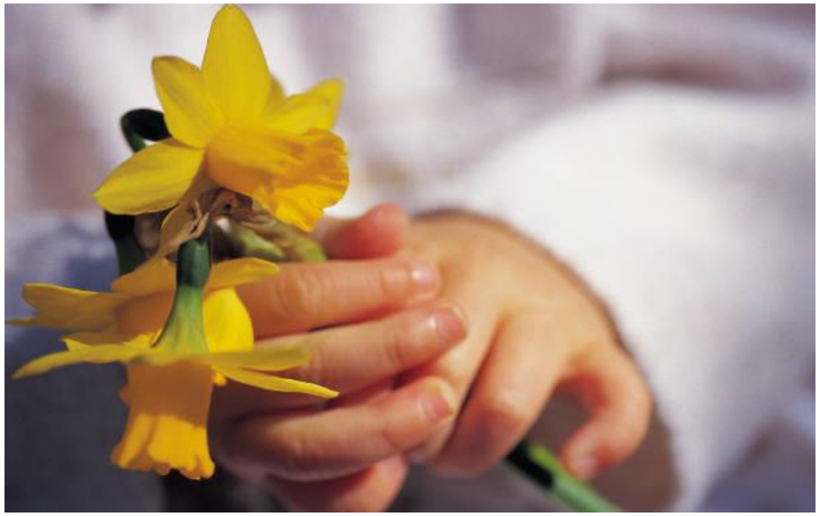
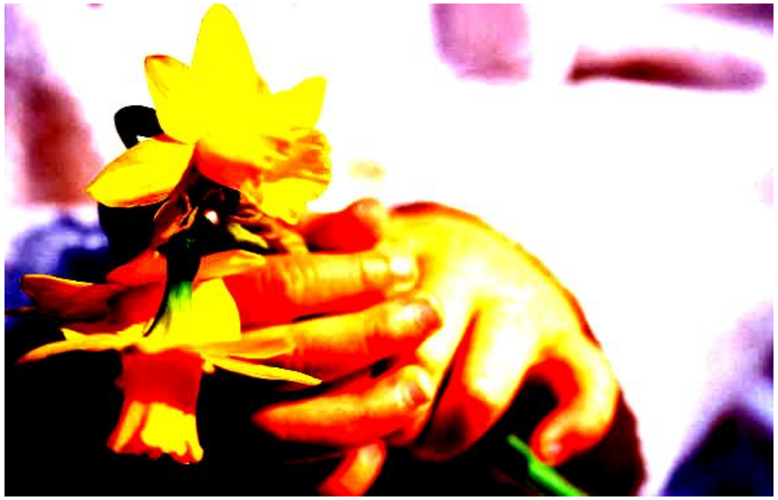


Image (s) after applying T (contrast stretching)



Notice that the intensity transformation function T, made the pixels with dark intensities darker and the bright ones even more brighter, this is called contrast stretching.

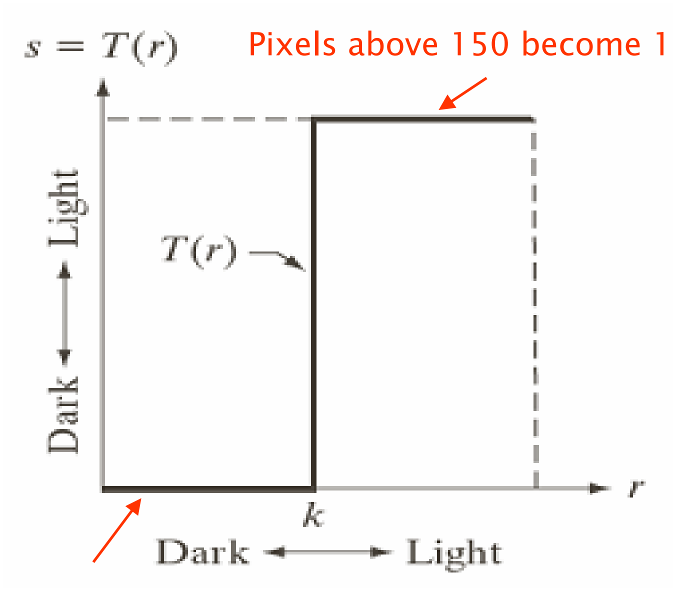
Example on thresholding:

☝ Remember that:

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

✍ **Example**: suppose m= 150 (called threshold),

- if r (or pixel intensity in image f) is above this threshold it becomes 1 in s (or pixel intensity in image g), otherwise it becomes zero.



Pixels less than 150 become 0

$$T = \begin{cases} \text{If } f(x,y) > 150;\ g(x,y) = 1 \\ \text{If } f(x,y) < 150;\ g(x,y) = 0 \end{cases}$$

Or simply…

$$T = \begin{cases} \text{If } r > 150;\ s = 1 \\ \text{If } r < 150;\ s = 0 \end{cases}$$

This is called thresholding, and it produces a binary image!

Image (r)

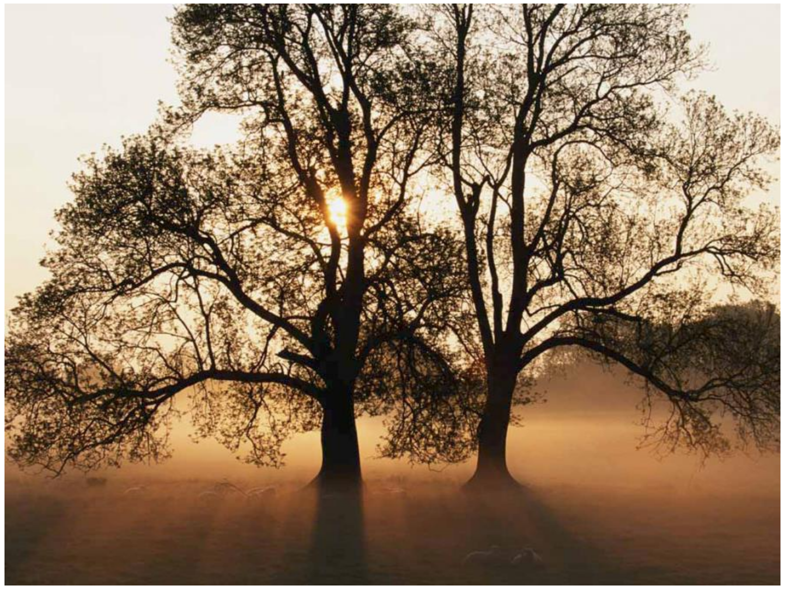
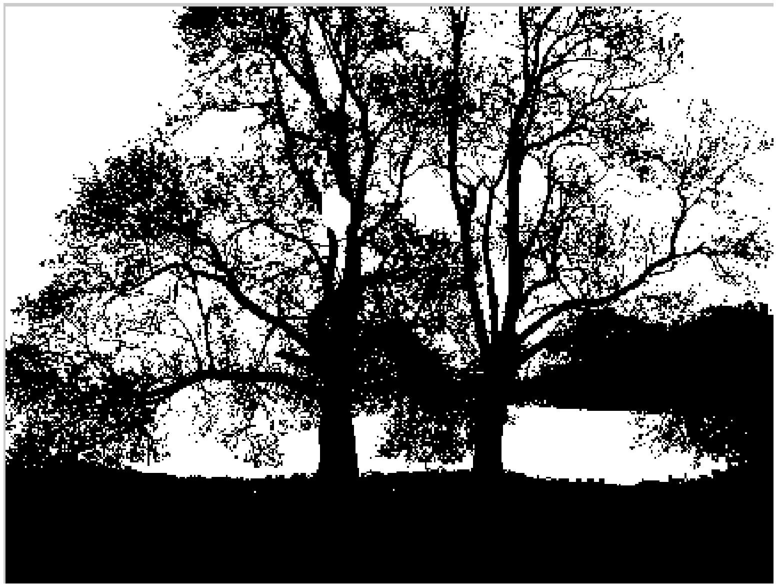
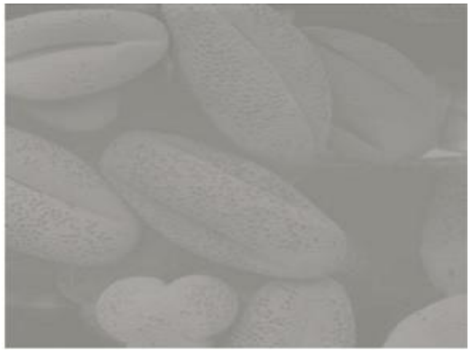


Image (s) after applying T (Thresholding)

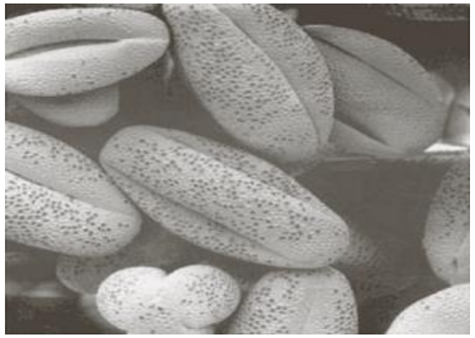


Notice that the intensity transformation function T, convert the pixels with dark intensities into black and the bright pixels into white. Pixels above threshold is considered bright and below it is considered dark, and this process is called thresholding.

Application on Contrast stretching and thresholding:

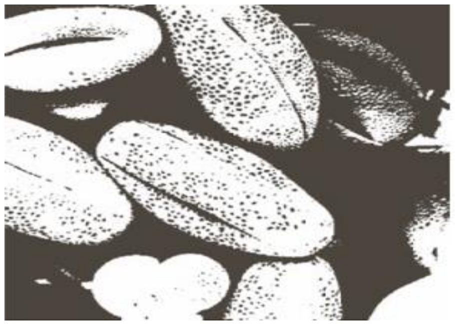


8-bit image with low contrast



After contrast stretching

$(r1,s1)=(r_{min},0)$, $(r2,s2)=(r_{max},L-1)$



Thresholding function

$(r1,s1)=(m,0)$, $(r2,s2)=(m,L-1)$

m : mean intensity level in the image

Exercise on Contrast stretching and thresholding:

Exercise:

the following matrix represents the pixels values of a 8-bit image (r) , apply thresholding transform assuming that the threshold m=95, find the resulting image pixel values.

Image (r)

110	120	90	130
91	94	98	200
90	91	99	100
82	96	85	90

solution:

Image (s)

Example:

Apply the following IT to the 3x3 image to produce the Binary image using threshold equal to 55:

65	78	43
21	99	12
200	31	77

Answer: *Resultant Binary Image is:*

1	1	0

Intensity-level slicing: (gray-level slicing)

- Highlighting a **specific range** of gray levels in an image.
- **Applications** include enhancing features such as masses of water in satellite imagery and enhancing flaws in X-ray images.
- There are several ways of doing level slicing, but most of them are variations of **two basic themes**.
- **One** approach shown in (**Fig. 3.11(a))** is to display a high value for all gray levels in the range of interest and a low value for all other gray levels producing a **binary** image.
- The **second approach shown in (Fig. 3.11(b))** brightens the desired range of gray levels but **preserves** the background and gray-level tonalities in the image.

Fig 3.11:

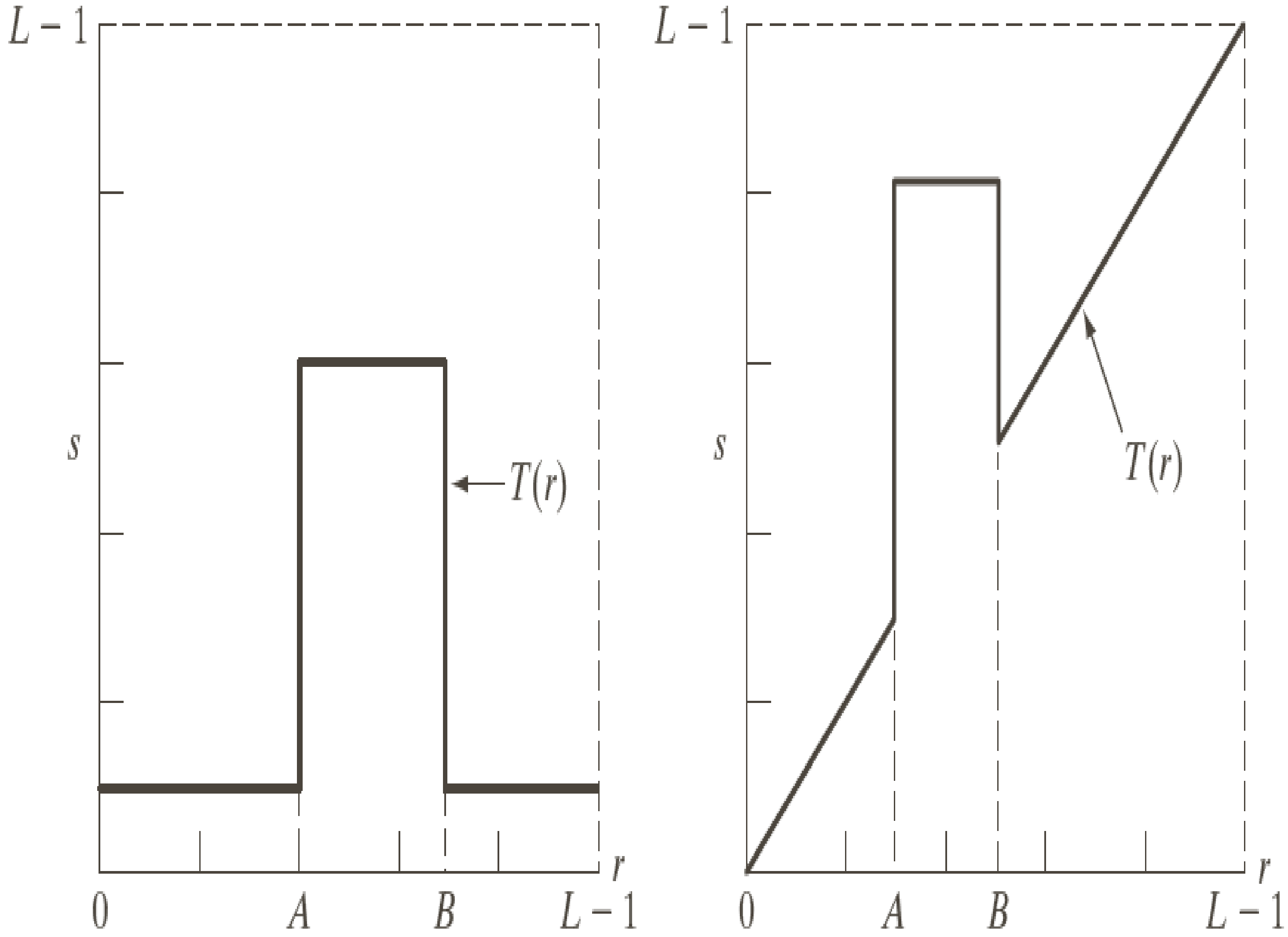
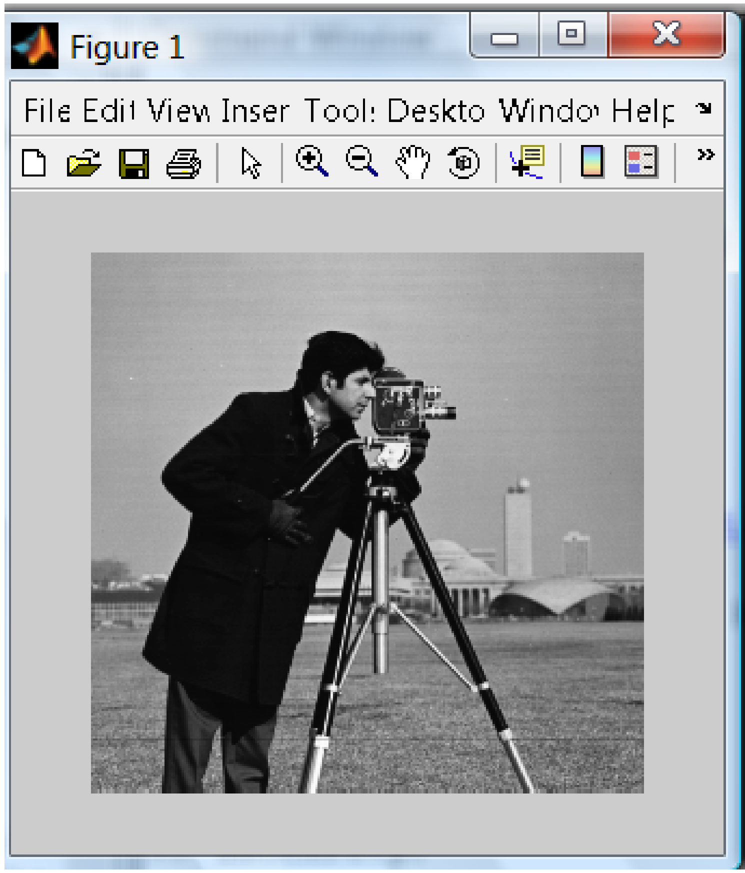
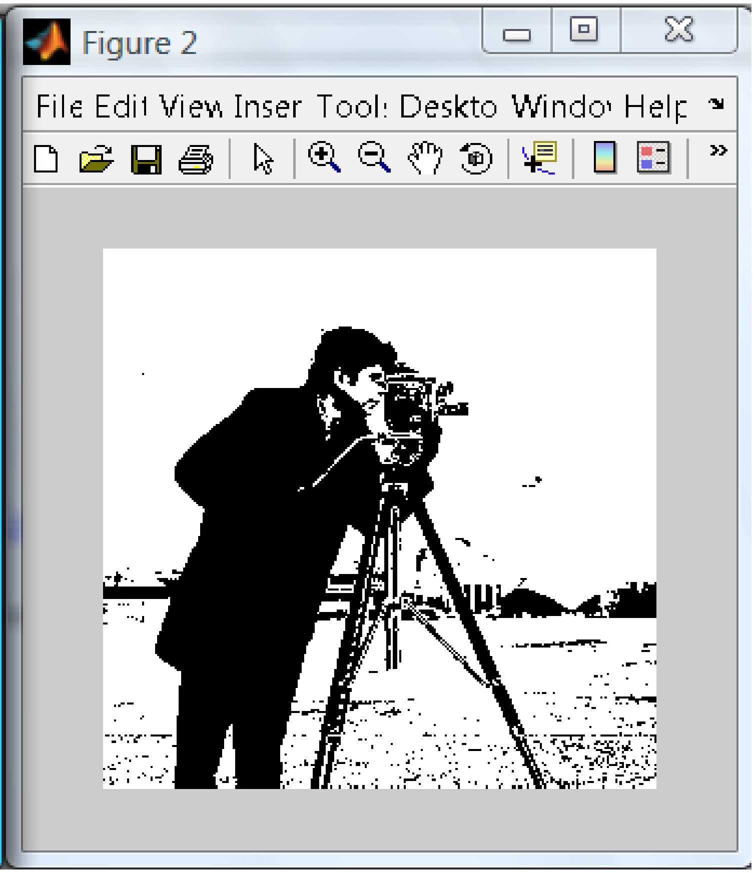


FIGURE 3.11 (a) This transformation highlights intensity range $[A, B]$ and reduces all other intensities to a lower level. (b) This transformation highlights range $[A, B]$ and preserves all other intensity levels.

Application of intensity level slicing:

Approach 1:





Application of intensity level slicing:

Approach 2:

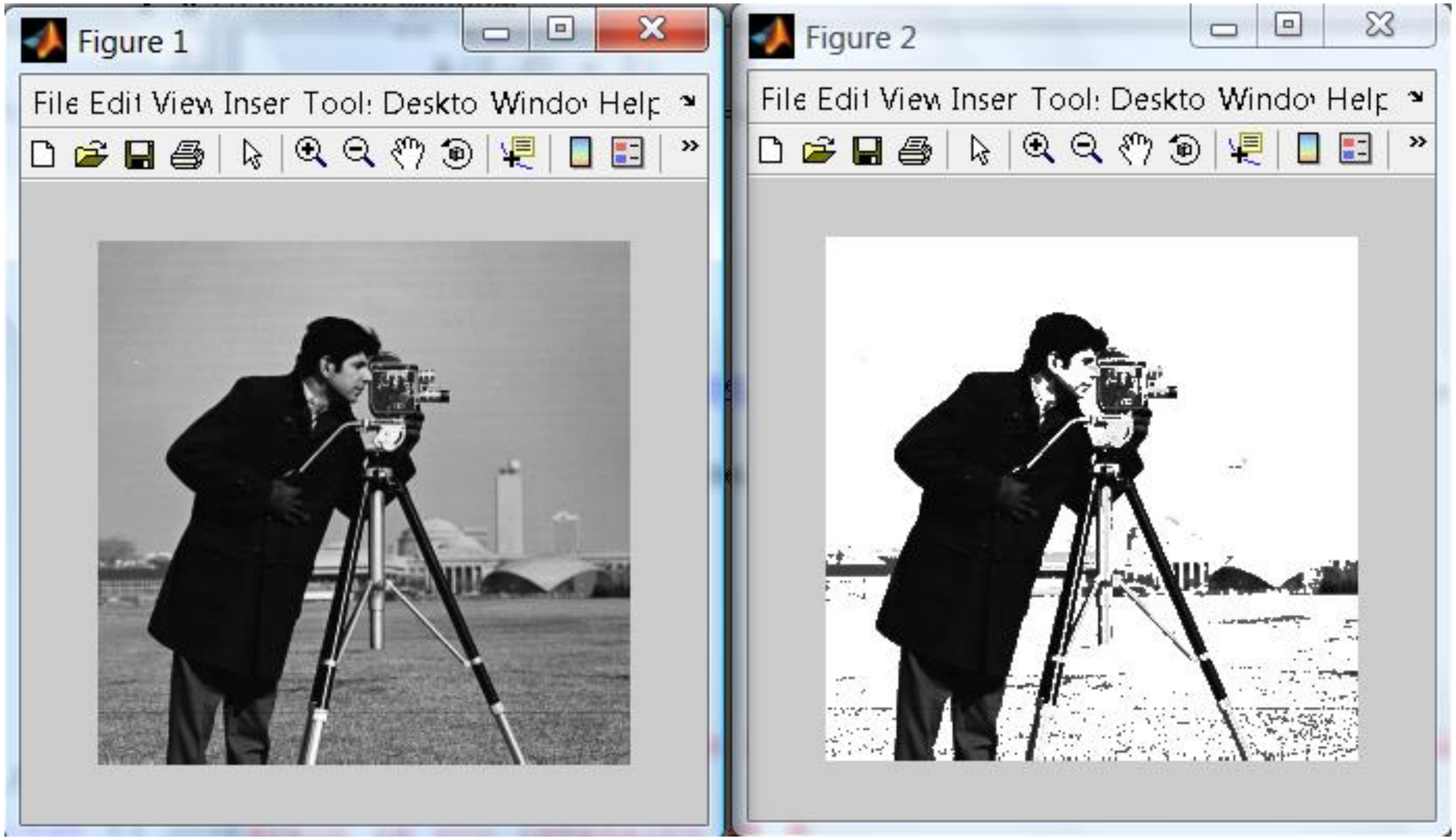
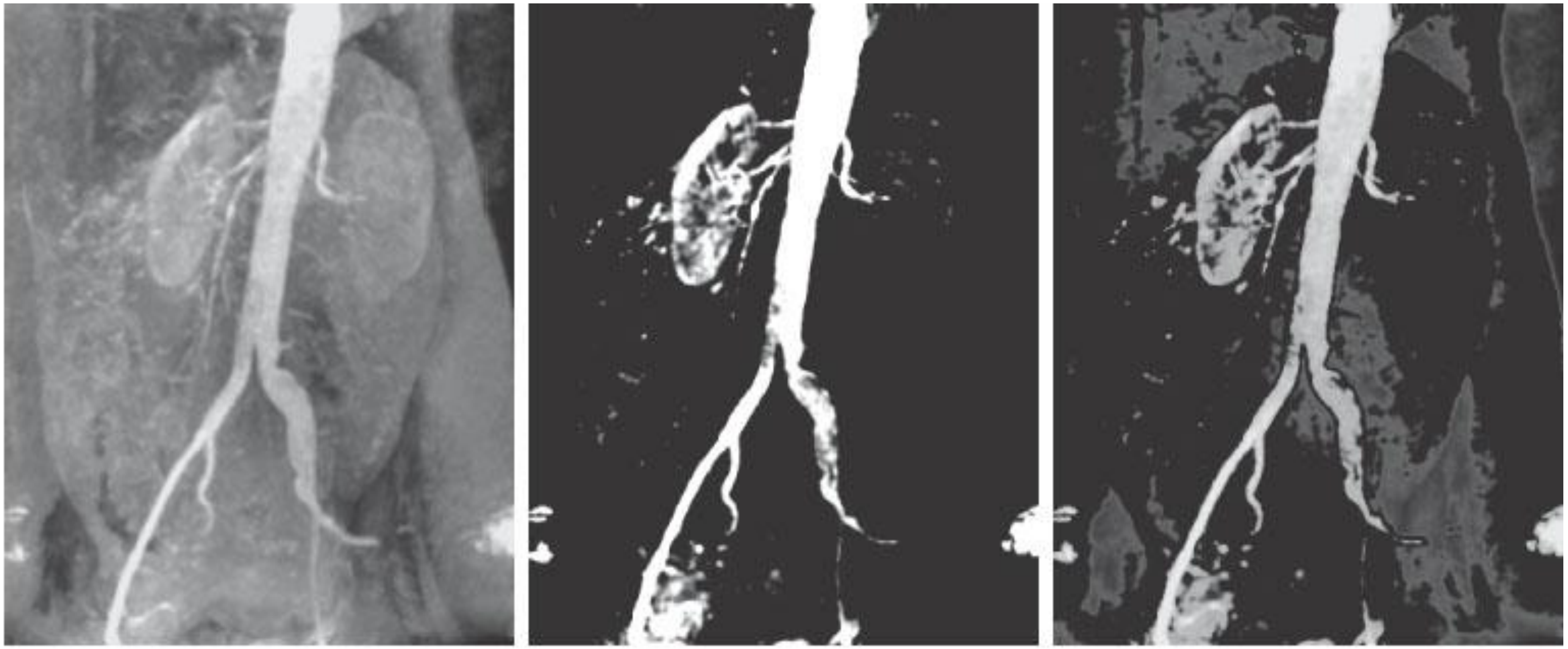


Figure 1
File Edit View Inser Tool: Deskto Windo Help
Figure 2
File Edit View Inser Tool: Deskto Windo Help

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 range set near black, so that the 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.)



a b c

Bit-plane slicing (Figure 3.13) :

- Suppose that each pixel in an image is represented by **8 bits**.
- Imagine that the image is composed of **eight 1-bit planes,** ranging from bit-plane 0 to bit-plane 7
- **plane 0** contains all the lowest order bits and **plane 7** contains all the high-order bits.
- The **higher-order** bits (especially the top four) contain the majority of the **visually significant data**.
- The other bit planes contribute to more **subtle details** in the image.
- **Separating** a digital image into its bit planes is useful for analyzing the relative importance played by each bit of the image, a process that aids in determining the adequacy of the number of bits used to quantize each pixel. Also, this type of decomposition is useful for image **compression**.

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

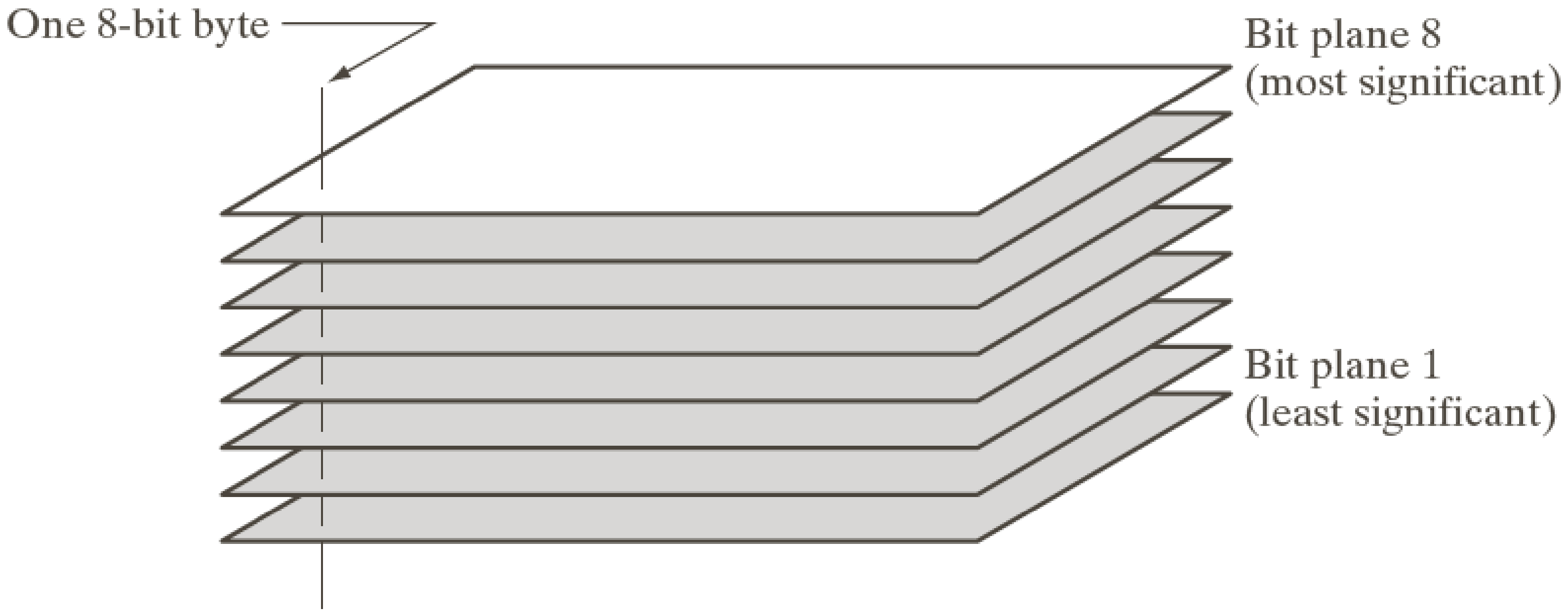
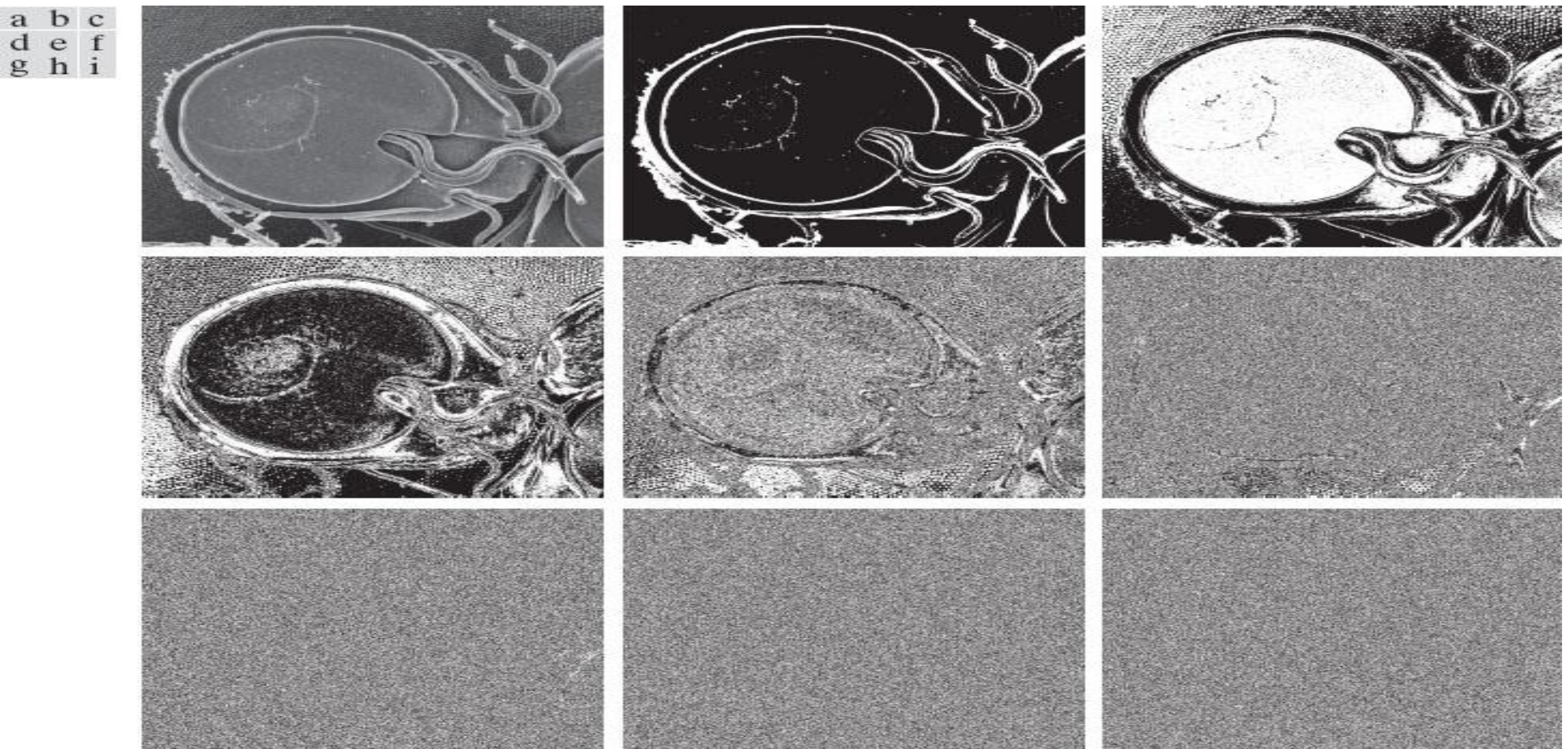
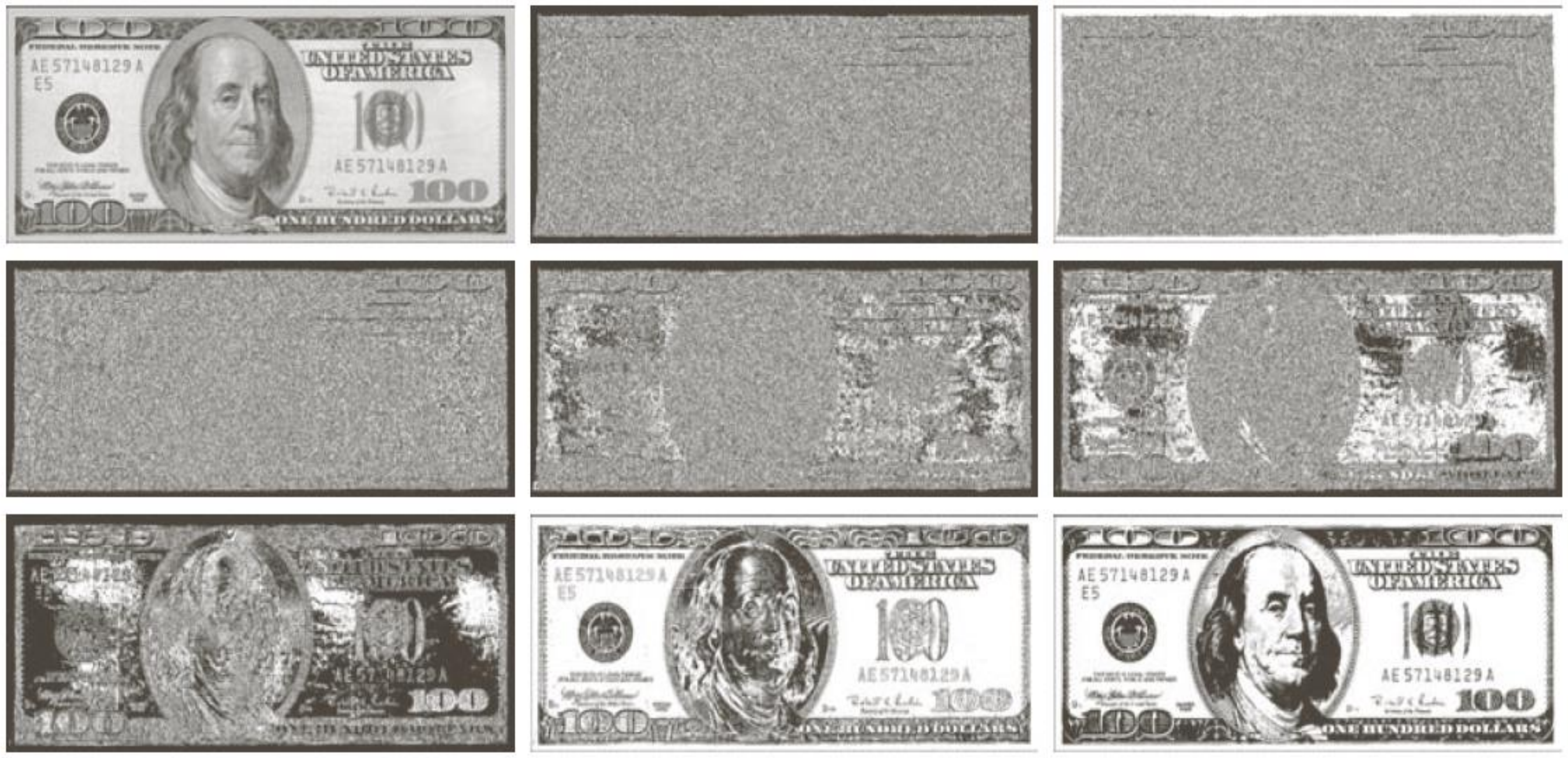


Figure 3.14

(a) An 8-bit grayscale image of size 837×988 pixels. (b) through (i) Bit planes 8 through 1, respectively, where plane 1 contains the least significant bit. Each bit plane is a binary image. Figure (a) is an SEM image of a trophozoite that causes a disease called giardiasis. (Courtesy of Dr. Stan Erlandsen, U.S. Center for Disease Control and Prevention.)



Application of 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.

Bit-Plane Slicing (example)

100			

0 1 1 0 0 1 0 0

Image of bit1:
00000000

0			

Image of bit2:
00000000

0			

Image of bit3:
00000100

4			

Image of bit4:
00000000

0			

Image of bit5:
00000000

0			

Image of bit6:
00100000

32			

Image of bit7:
01000000

64			

Image of bit8:
00000000

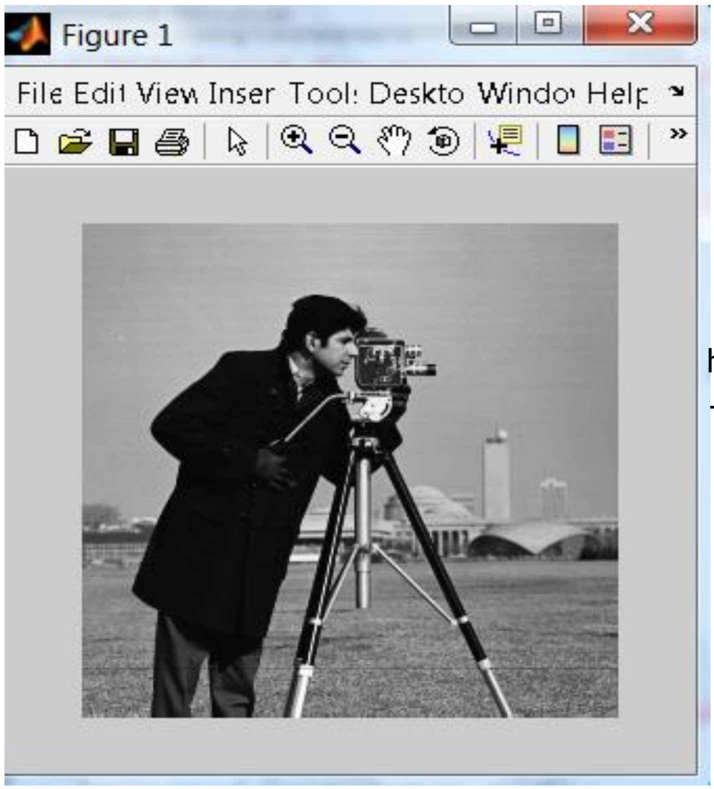
0			

3.3 Histogram Processing:

The histogram of a digital image with gray levels in the range $[0, L-1]$ is a discrete function $h(r_k) = n_k$, where r_k is the kth gray level and n_k is the number of pixels in the image having gray level r_k. It is common practice to normalize a histogram by dividing each of its values by the total number of pixels in the image, denoted by n. Thus, a normalized histogram is given by $p(r_k) = n_k/n$, for $k = 0, 1, \dots, L-1$. Loosely speaking, $p(r_k)$ gives an estimate of the probability of occurrence of gray level r_k. Note that the sum of all components of a normalized histogram is equal to 1.

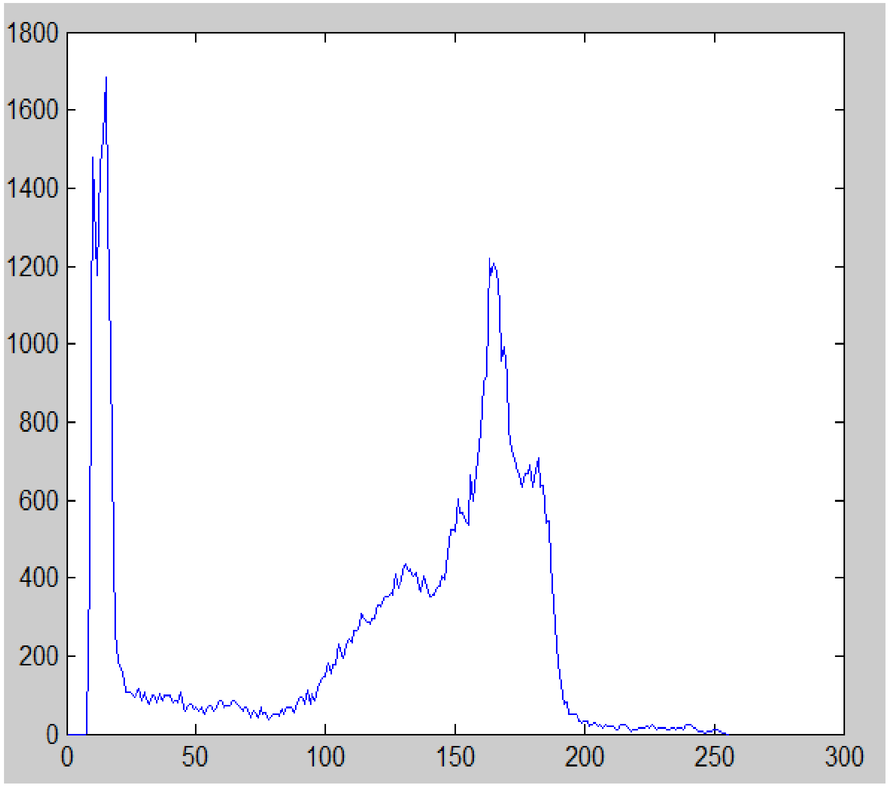
$n=MN$.

Histogram of the Image:



histogram

$$\xrightarrow{\hspace{2cm}}$$



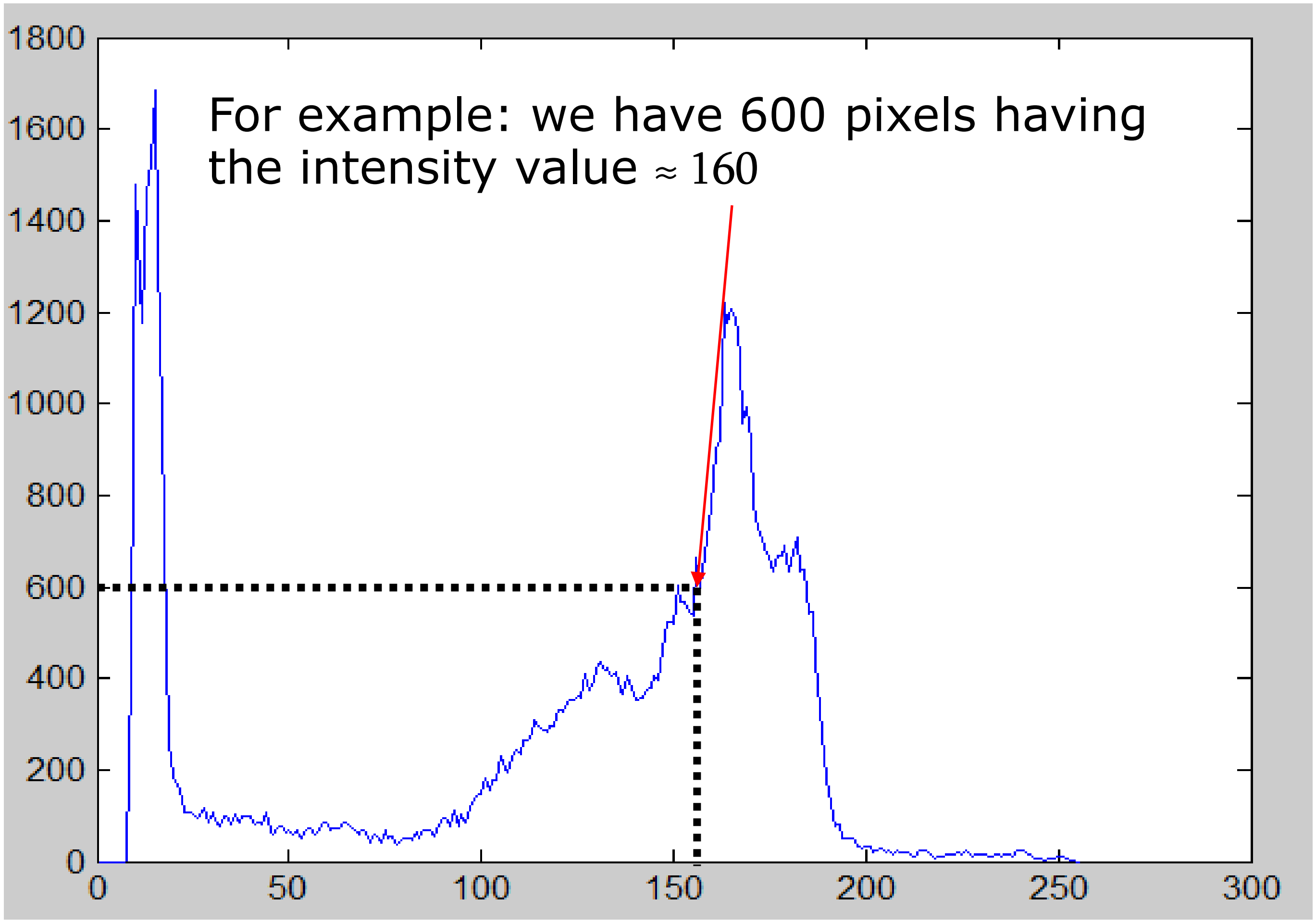
$h(r_k) = n_k$ number of pixels for each value of gray levels

Where:

r_k : kth gray level

n_k : # of pixels with having gray level rk

Histogram of the image:



Example:

Consider the following intensity distribution for a 2-bit image of size 32x32 pixels and find its Normalized Histogram:

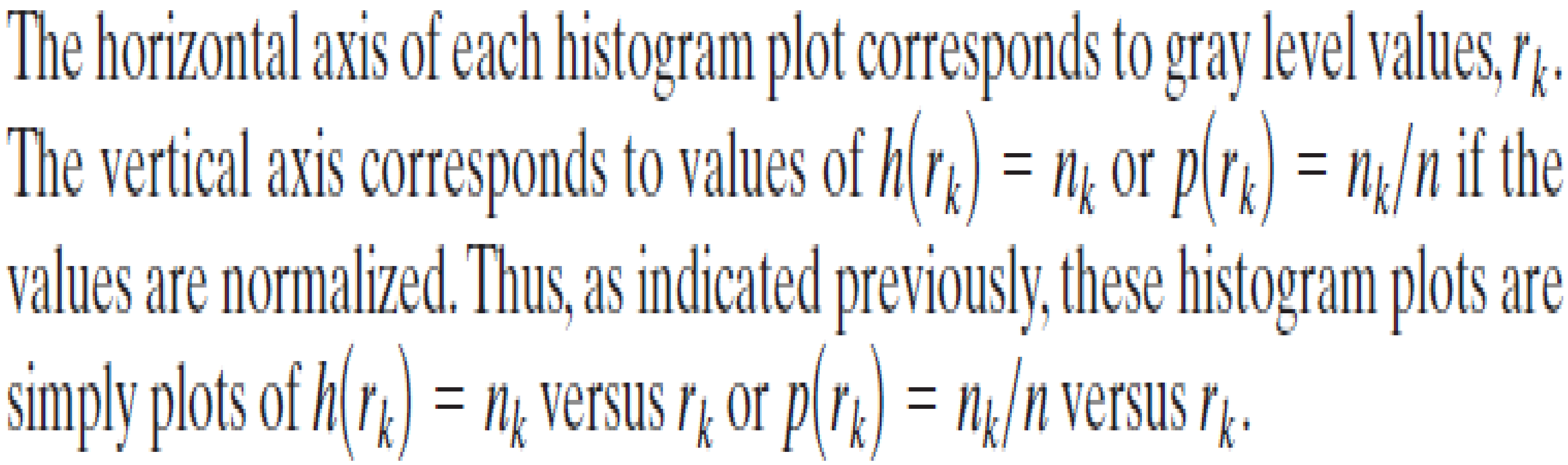
r_k	n_k
0	500
1	500
2	22
3	2

Ans. *The Normalized Histogram is:*

r_k	N_k	N_k/MN
0	500	500/1024=.4883
1	500	500/1024=.4883
2	22	22/1024=.0215
3	2	2/1024=.0020

Role of histogram processing in image enhancement:

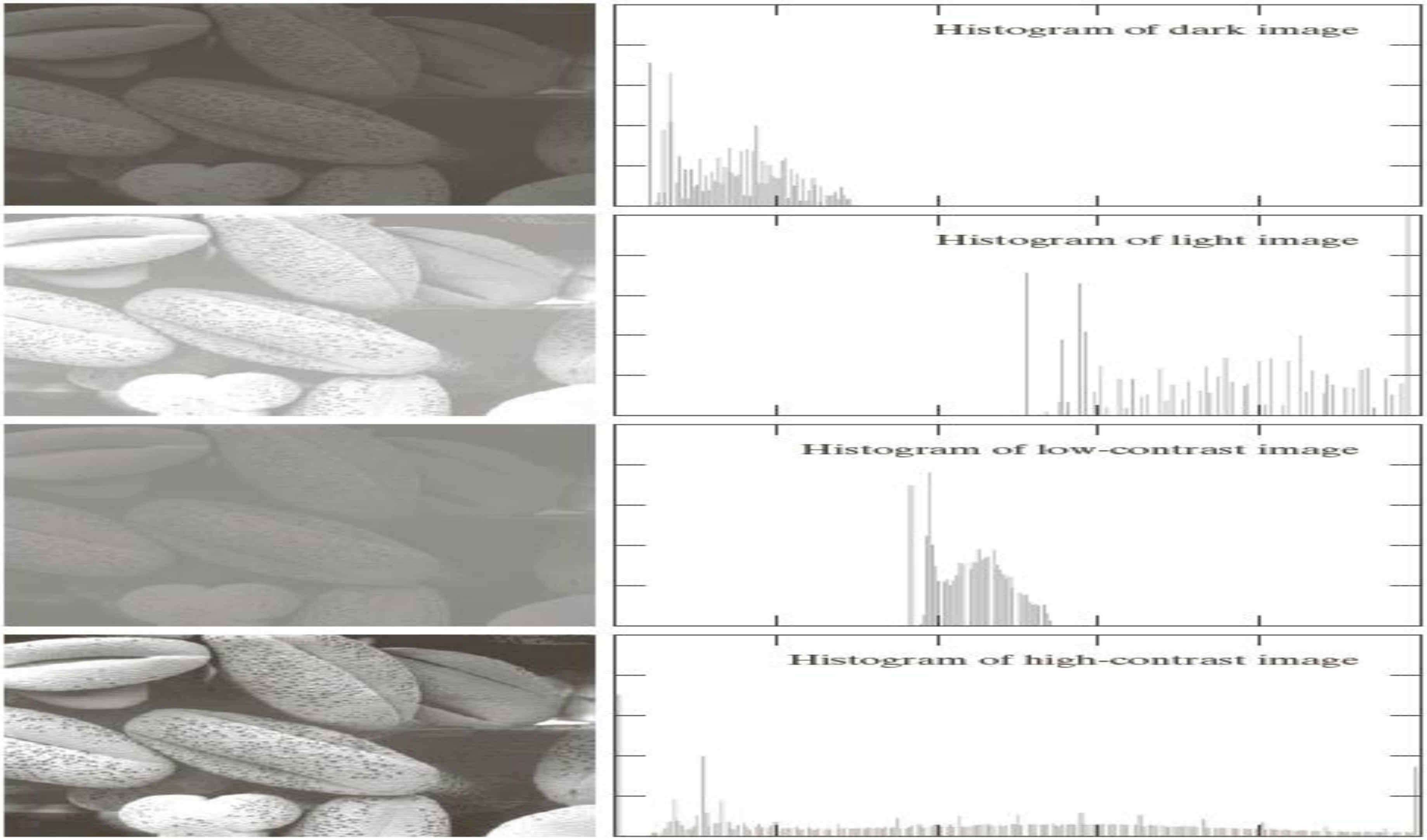
- Consider **Fig. 3.16.**
- It is shown in **four** basic gray-level characteristics: dark, light, low contrast, and high contrast.



The horizontal axis of each histogram plot corresponds to gray level values, r_k. The vertical axis corresponds to values of $h(r_k) = n_k$ or $p(r_k) = n_k/n$ if the values are normalized. Thus, as indicated previously, these histogram plots are simply plots of $h(r_k) = n_k$ versus r_k or $p(r_k) = n_k/n$ versus r_k.

Four Basic Image Types:

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



Histogram Equalization

- For discrete values, a transformation function of particular importance in image processing has the form **(equation 3.15):**

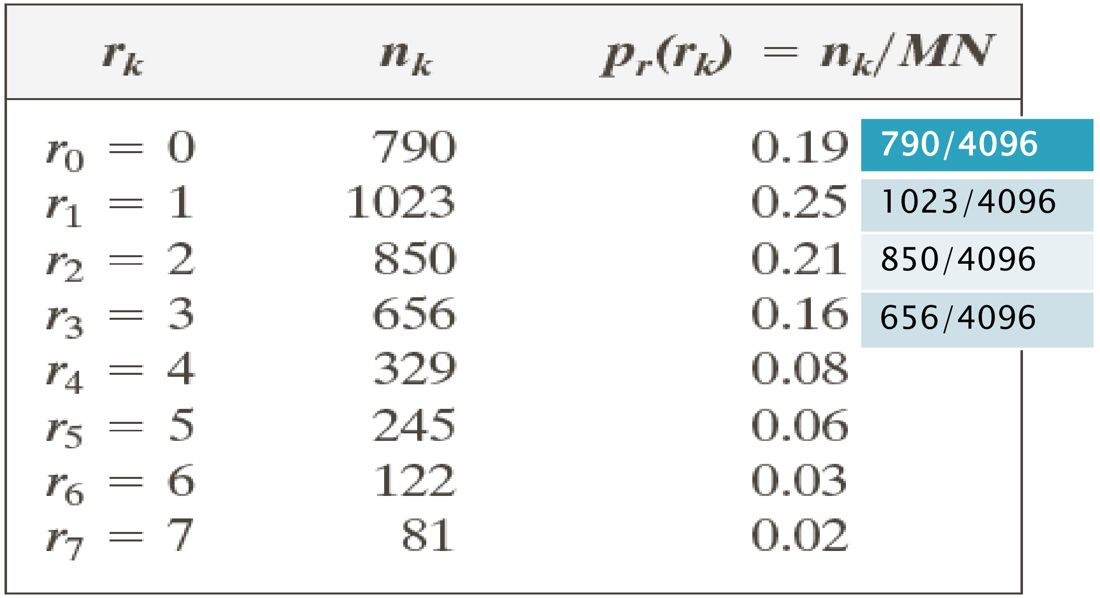
$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) = \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j$$

- Where k=0,1,2,....,L-1

The above results in Histogram Equalization.

Example 3.5:

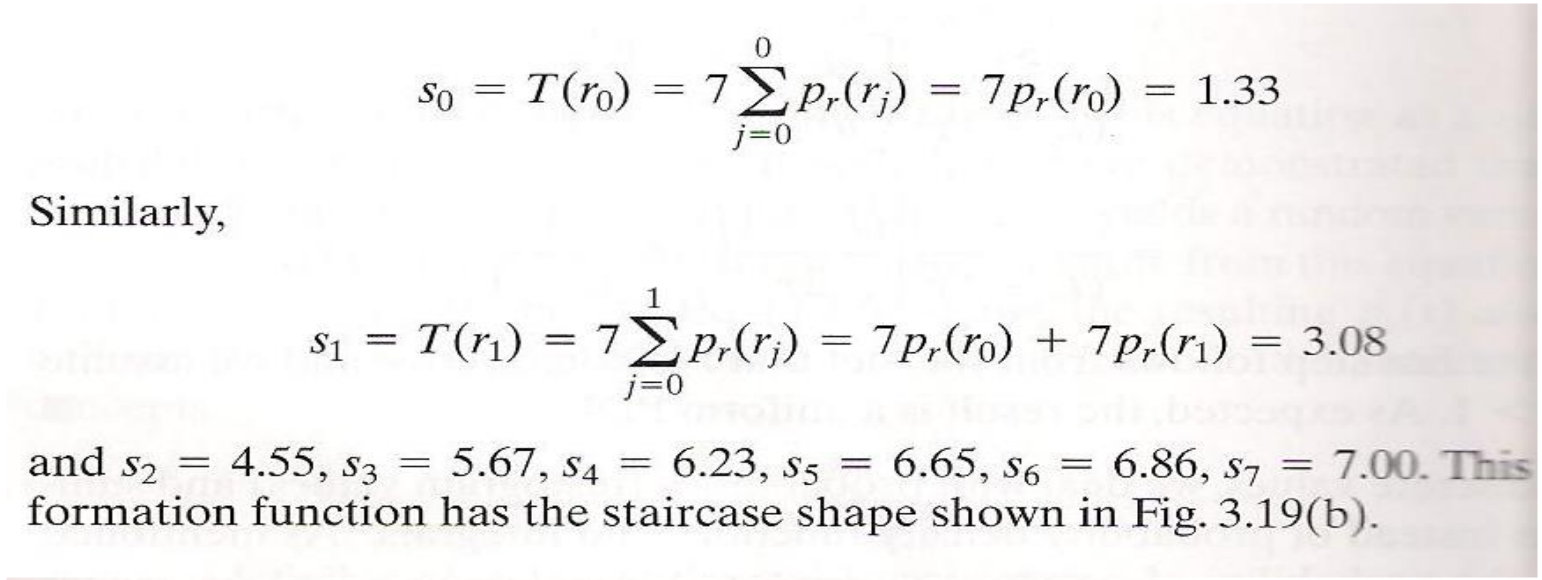
Suppose a 3-bit image of size 64x64 pixels (MN=4096) has the intensity distribution shown in Table 3.1, where the intensity levels are integers in the range [0, L-1]= [0,7]. The histogram of the hypothetical image is sketched in Fig.3.19 (a). Values of the histogram equalization transformation function are obtained using EQ. (3.15)



r_k	n_k	$p_r(r_k) = n_k/MN$	
$r_0 = 0$	790	0.19	790/4096
$r_1 = 1$	1023	0.25	1023/4096
$r_2 = 2$	850	0.21	850/4096
$r_3 = 3$	656	0.16	656/4096
$r_4 = 4$	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.

For example:

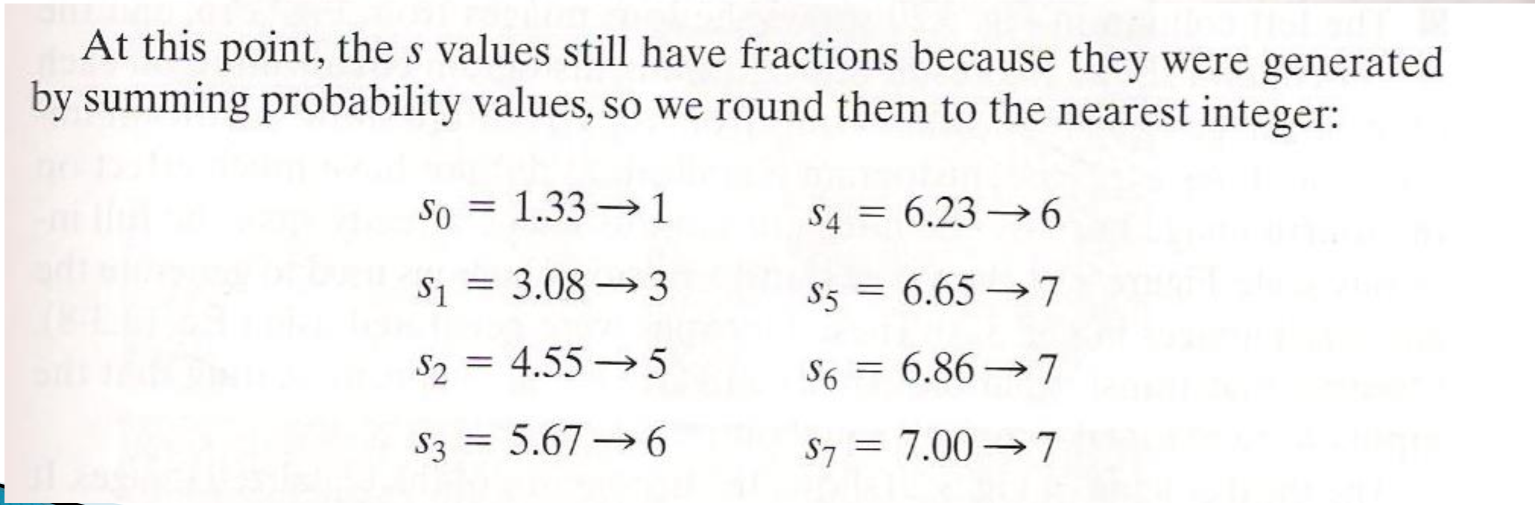


$$s_0 = T(r_0) = 7\sum_{j=0}^{0} p_r(r_j) = 7p_r(r_0) = 1.33$$

Similarly,

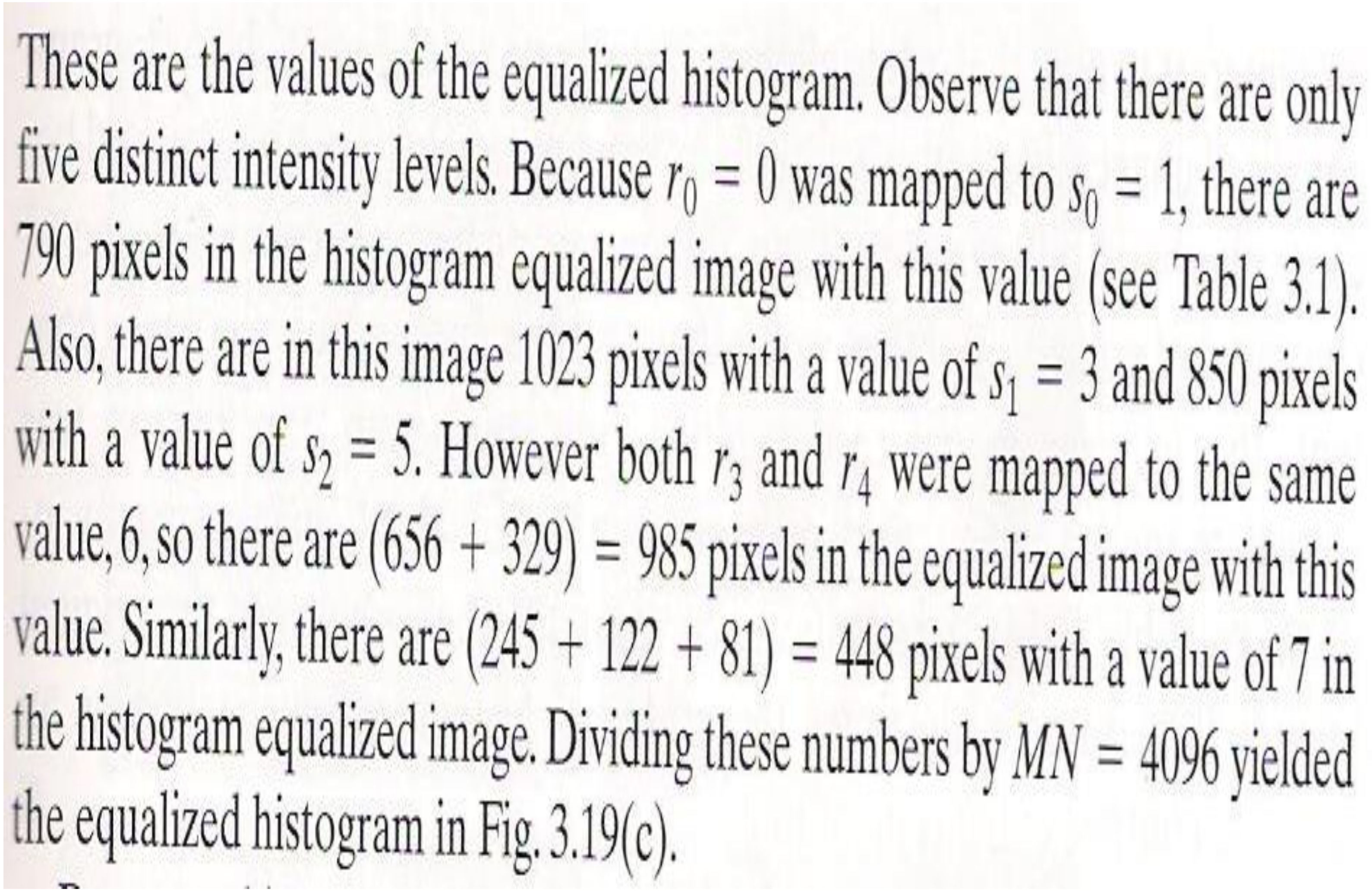
$$s_1 = T(r_1) = 7\sum_{j=0}^{1} p_r(r_j) = 7p_r(r_0) + 7p_r(r_1) = 3.08$$

and $s_2 = 4.55$, $s_3 = 5.67$, $s_4 = 6.23$, $s_5 = 6.65$, $s_6 = 6.86$, $s_7 = 7.00$. This formation function has the staircase shape shown in Fig. 3.19(b).



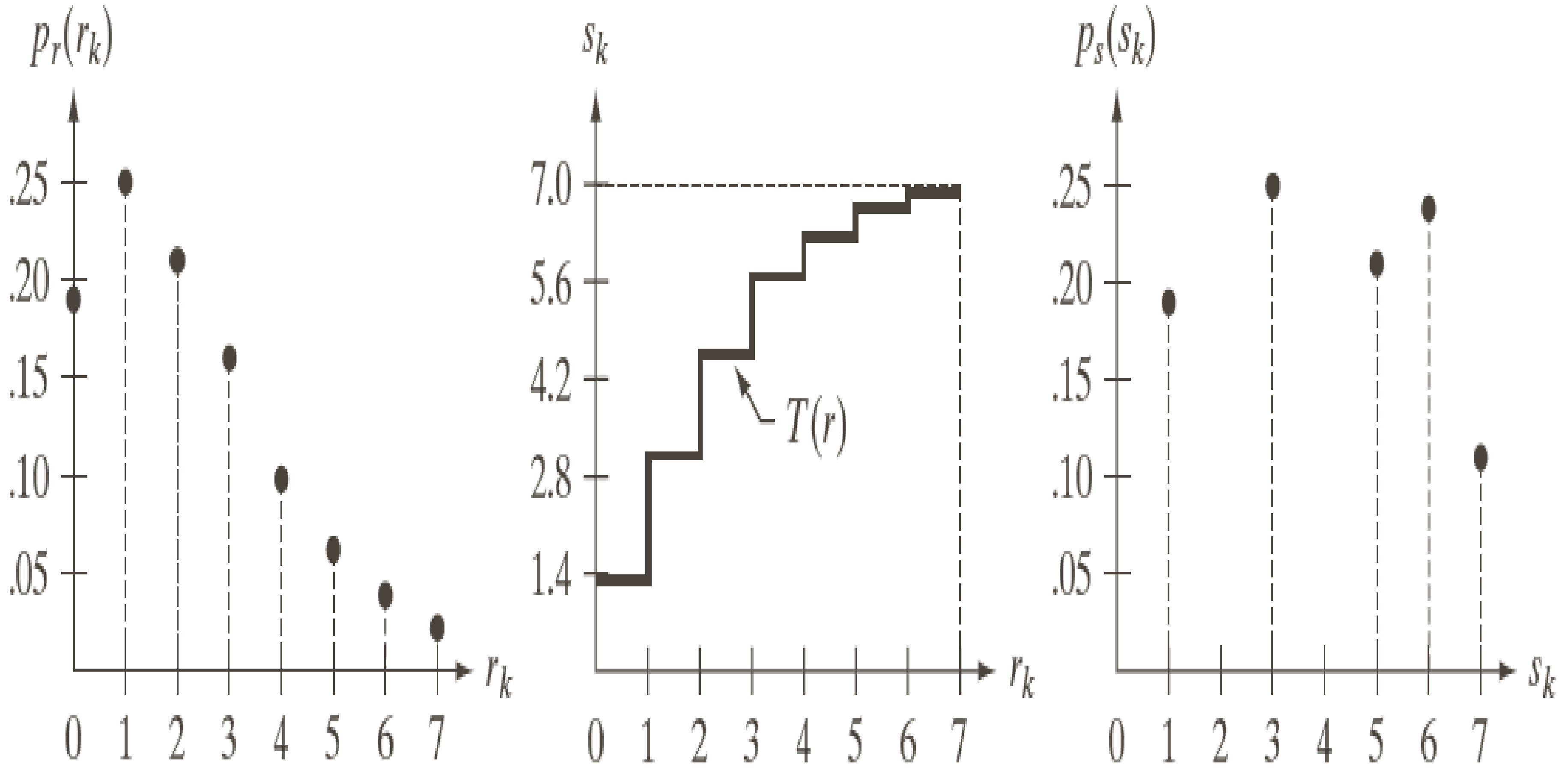
At this point, the s values still have fractions because they were generated by summing probability values, so we round them to the nearest integer:

$$\begin{array}{ll} s_0 = 1.33 \rightarrow 1 & s_4 = 6.23 \rightarrow 6 \\ s_1 = 3.08 \rightarrow 3 & s_5 = 6.65 \rightarrow 7 \\ s_2 = 4.55 \rightarrow 5 & s_6 = 6.86 \rightarrow 7 \\ s_3 = 5.67 \rightarrow 6 & s_7 = 7.00 \rightarrow 7 \end{array}$$



These are the values of the equalized histogram. Observe that there are only five distinct intensity levels. Because $r_0 = 0$ was mapped to $s_0 = 1$, there are 790 pixels in the histogram equalized image with this value (see Table 3.1). Also, there are in this image 1023 pixels with a value of $s_1 = 3$ and 850 pixels with a value of $s_2 = 5$. However both r_3 and r_4 were mapped to the same value, 6, so there are $(656 + 329) = 985$ pixels in the equalized image with this value. Similarly, there are $(245 + 122 + 81) = 448$ pixels with a value of 7 in the histogram equalized image. Dividing these numbers by $MN = 4096$ yielded the equalized histogram in Fig. 3.19(c).

Solution:



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.

Example 3.6:

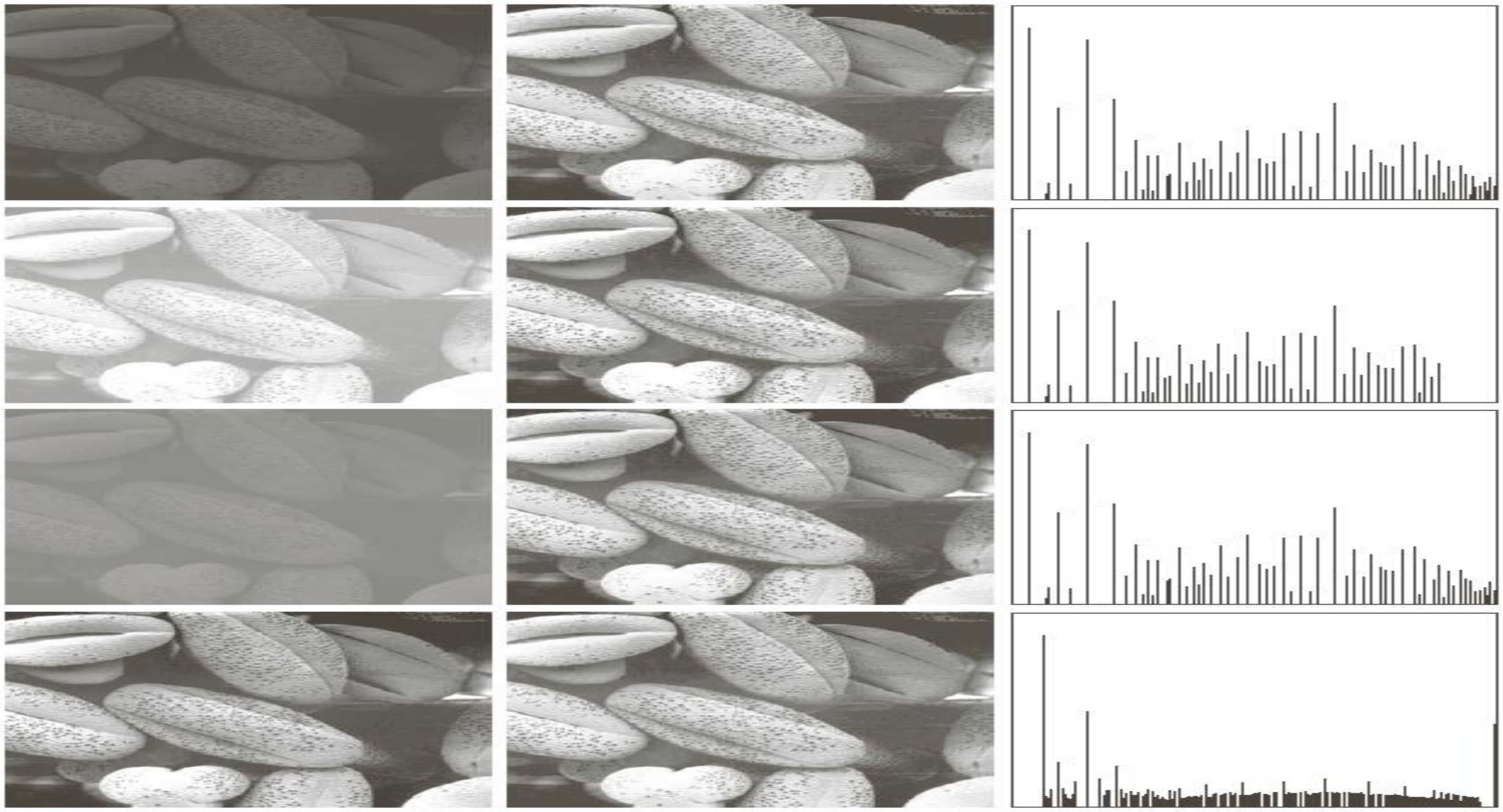
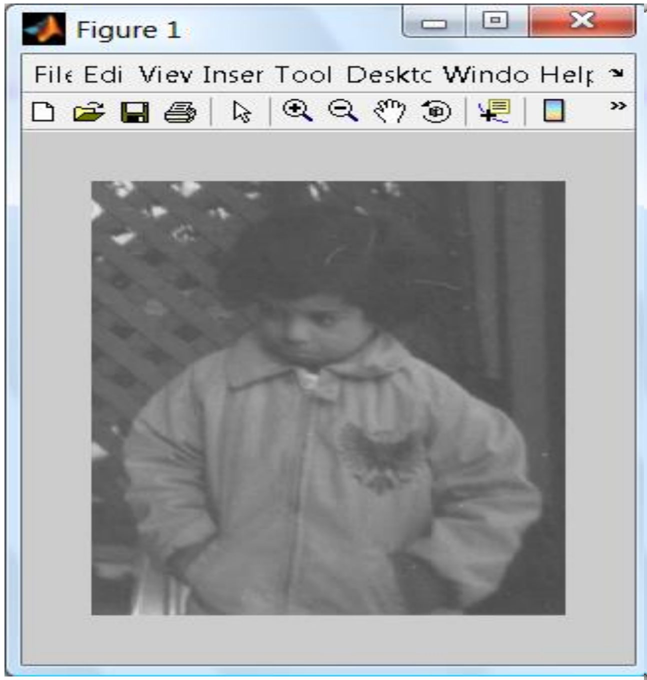
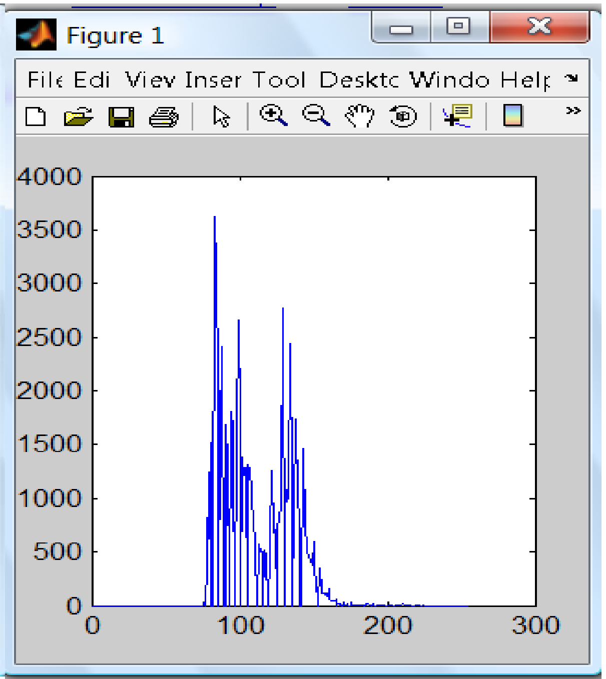


FIGURE 3.20 Left column: images from Fig. 3.16. Center column: corresponding histogram-equalized images. Right column: histograms of the images in the center column.

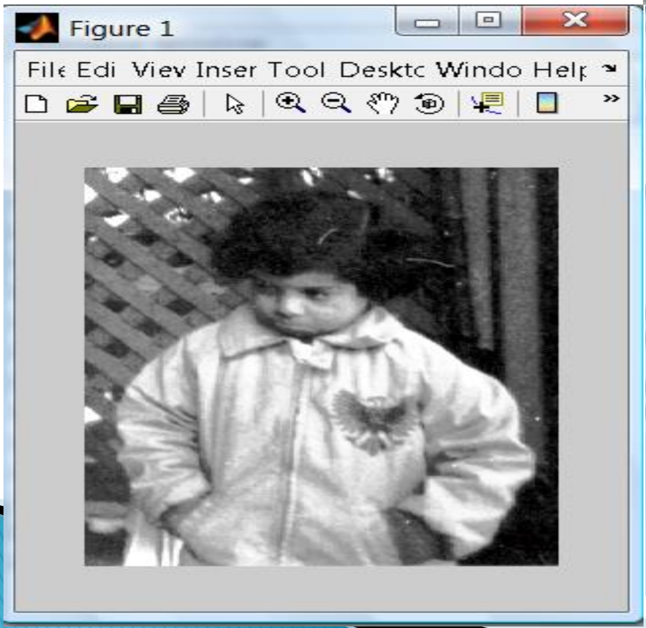
Another example of equalized histogram

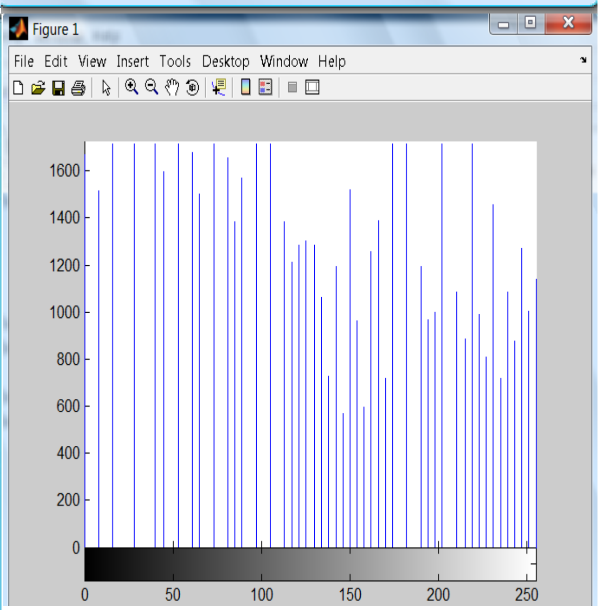




Notice that the pixels intensity values are concentrated on the middle

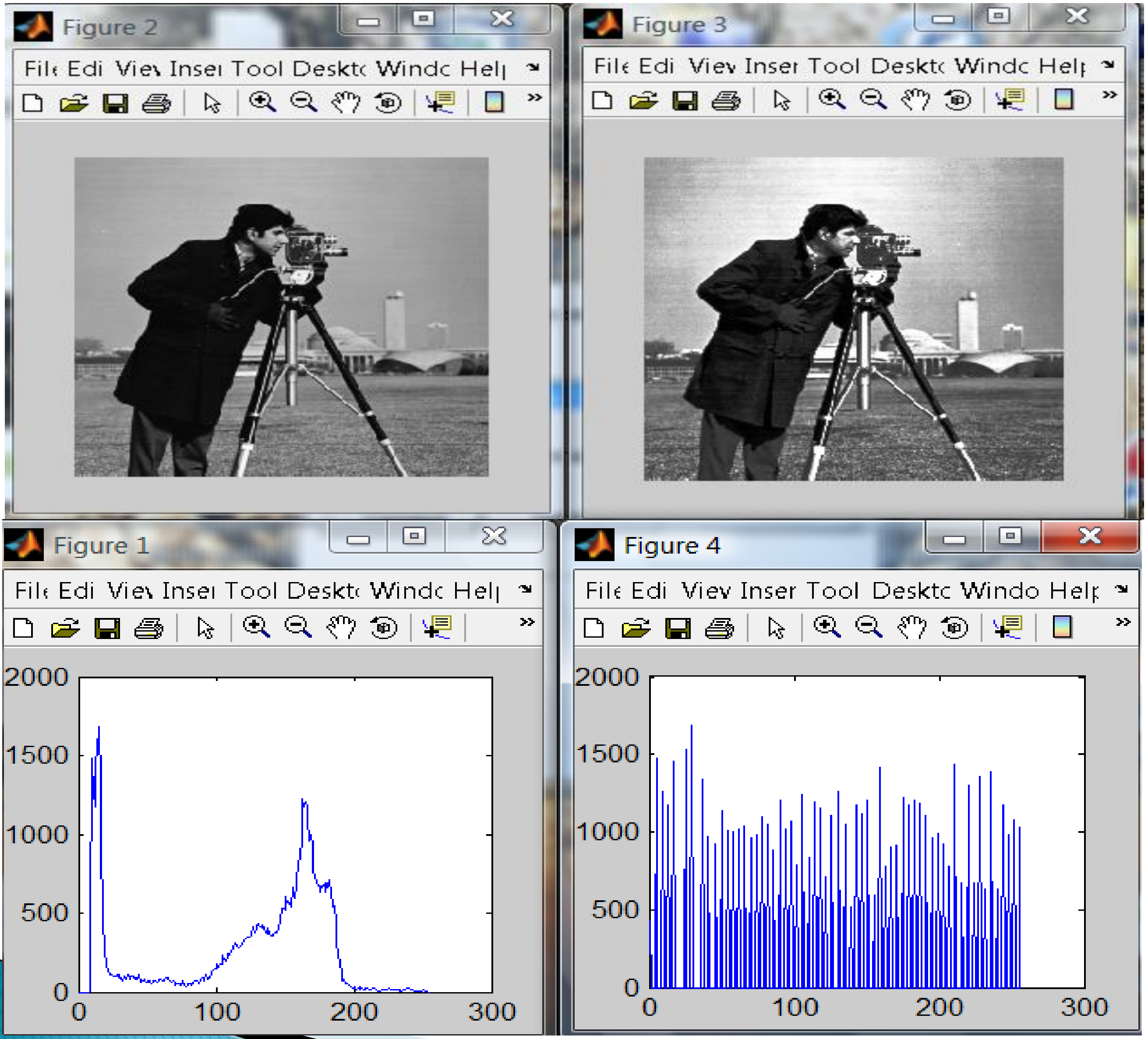
(low contrast)





Histogram produces pixels having values that are distributed throughout the range

Another example of equalized histogram



Notice that histogram equalization does not always produce a good result

Histogram Matching (Specification)

Histogram **equalization** seeks to produce a **uniform** histogram.

Sometimes, attempting to base enhancement on a uniform histogram is **not** the best approach.

It is useful sometimes to be able to specify the **shape** of the histogram that we wish the processed image to have.

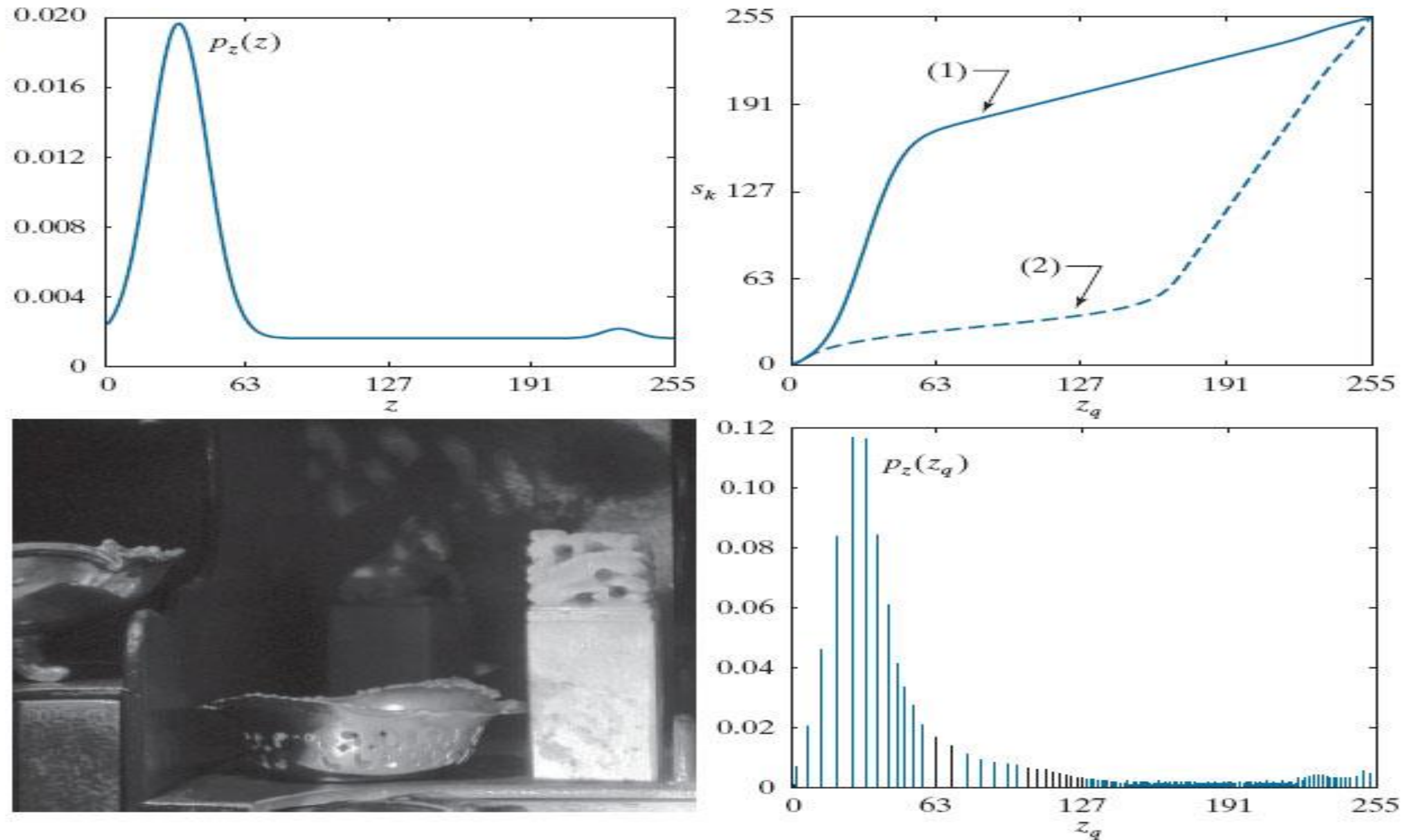
The method used to **generate a processed image** that has a specified histogram is called *histogram matching or histogram specification.*

It is usually a ***trail-and-error*** *process.*

Figure 3.25

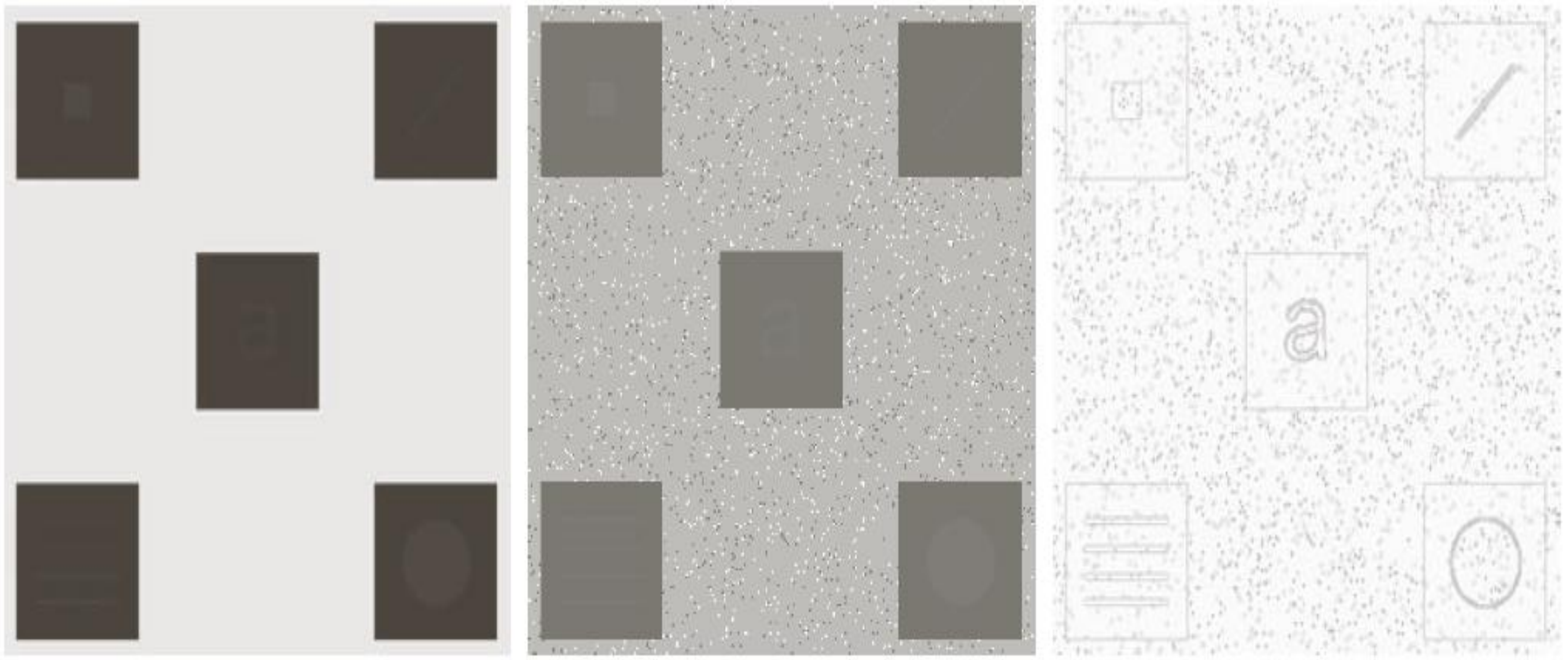
Histogram specification. (a) Specified histogram. (b) Transformation labeled (1), and $G^{-1}(s_k)$, labeled (2). (c) Result of histogram specification. (d) $G(z_q)$, Histogram of image (c).

a b
c d



Histogram Processing

- **Global:** pixels are modified based on entire image
- **Local:** neighbourhood of a pixel is used



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.

Use of Histogram Statistics for Image Enhancement

- **From Histogram:** *The nth moment of a random variable r about its mean is defined as:*

$$\mu_n(r) = \sum_{i=0}^{L-1} (r_i - m)^n p(r_i)$$

where m is the mean (measure of average intensity) value of r (i.e., the average intensity of the pixels in the image):

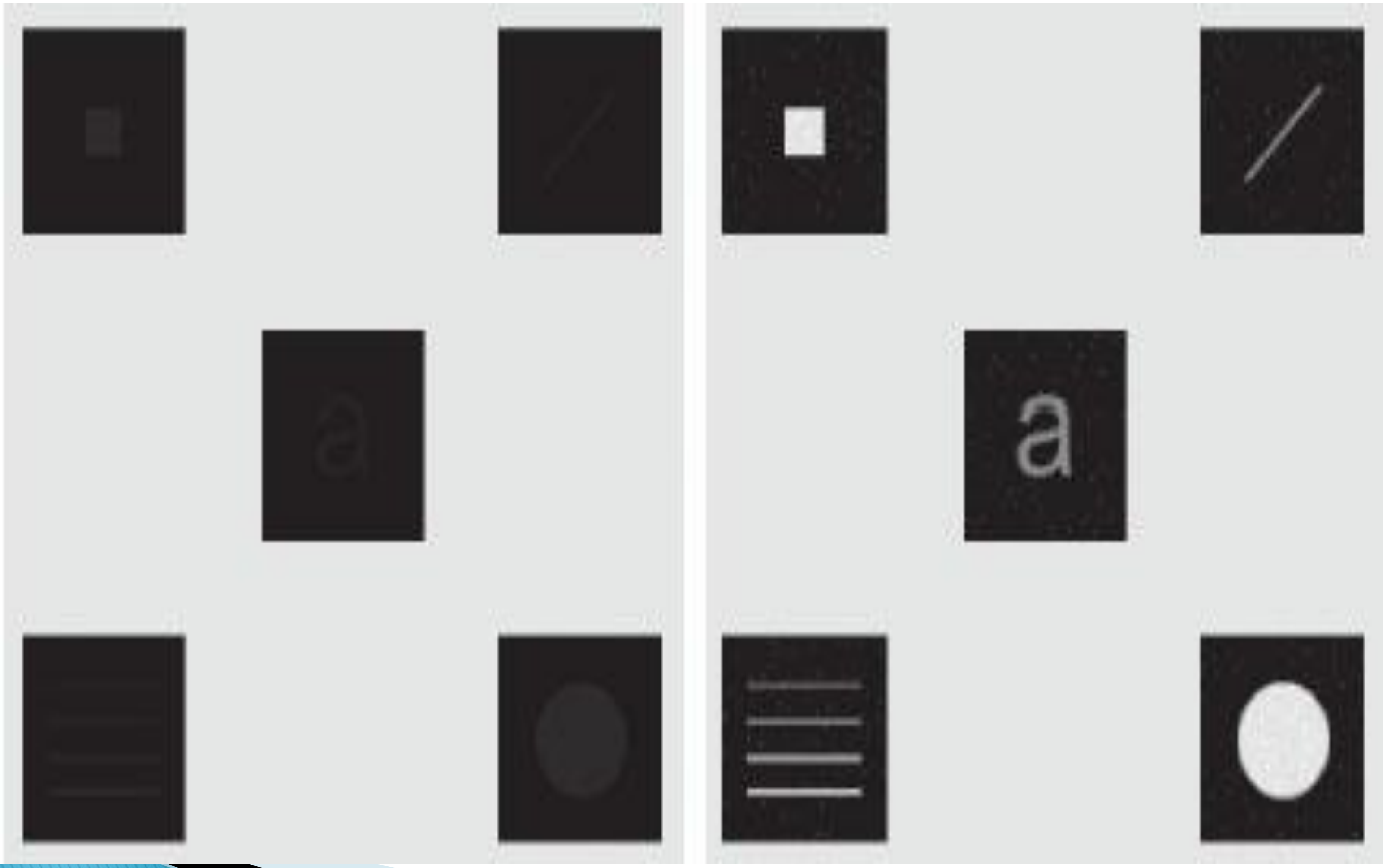
$$m = \sum_{i=0}^{L-1} r_i p(r_i)$$

- The standard deviation is a measure of image contrast.
- See Example next for use in enhancement.

Figure 3.27

(a) Original image. (b) Result of local enhancement based on local histogram statistics. Compare (b) with Fig. 3.26(c).

a b



Sample Mean and Variance:

$$m = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)$$

$$\sigma^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[f(x,y) - m\right]^2$$

Matlab:

adapthisteq, brighten, imadjust, imhist, histeq,

conv, convn, filter2, xcorr2, std2, mean2, subplot, mesh, imshow, imview, subimage, image, imagesc.