Point Processing

S.K. VIPPARTHI

Point Processing

- Spatial Operation
- Gray-Level Mapping
- Non-linear Gray-Level Mapping
  - Gamma Mapping
  - Logarithmic Mapping
  - Exponential Mapping
- The Image Histogram
- Histogram Stretching
- Histogram Equalization
- Thresholding
- Color Thresholding

Spatial Operation

- Single Pixel Operation
- Neighborhood Operation
- Geometric Spatial Transformation
  - A Spatial Transform of Coordinates
  - Intensity Interpolation
- Geometric Spatial Transformation
  - Affine Transform
    - Forward Mapping
    - Inverse Mapping

Gray-Level Mapping

$$g(x, y) = f(x, y) + b \qquad g(x, y) = a \cdot f(x, y) + b$$

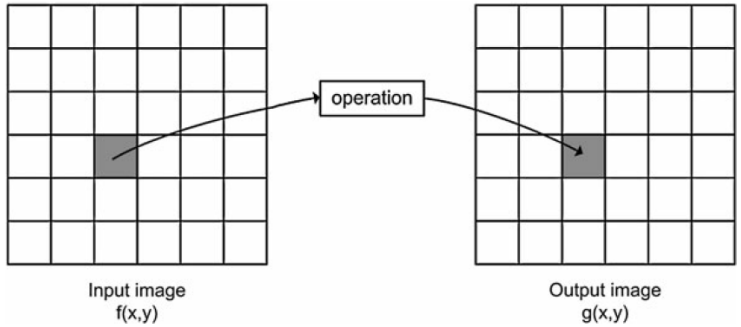
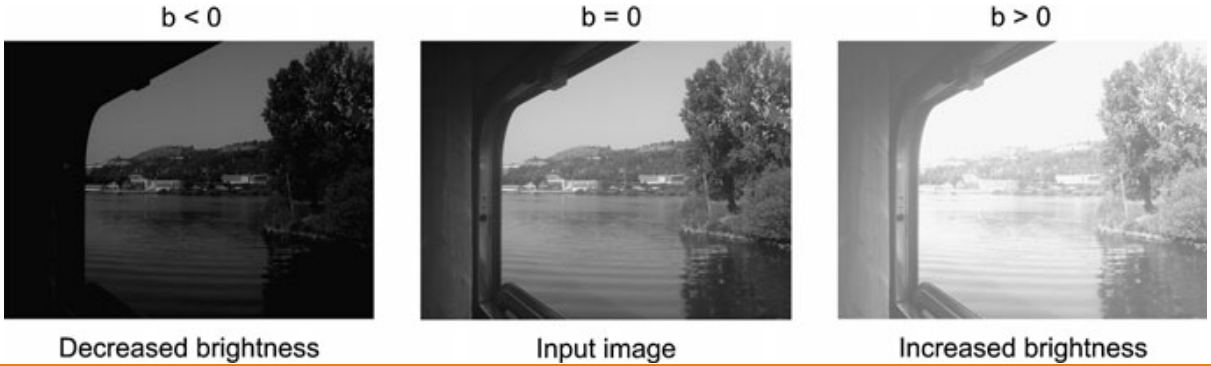


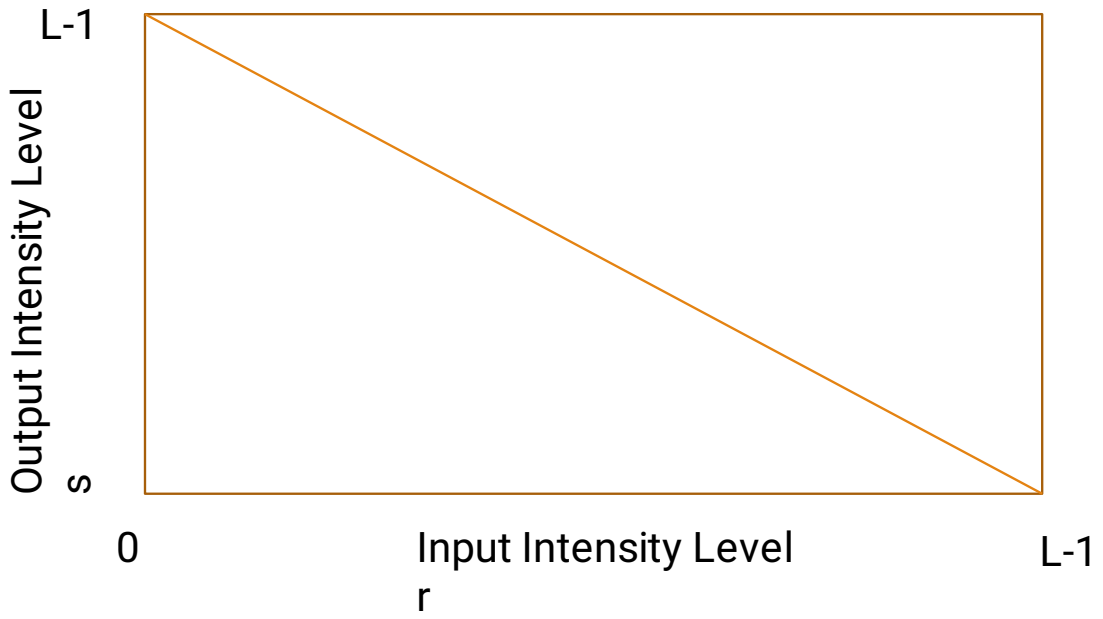
Fig. The principle of point processing.

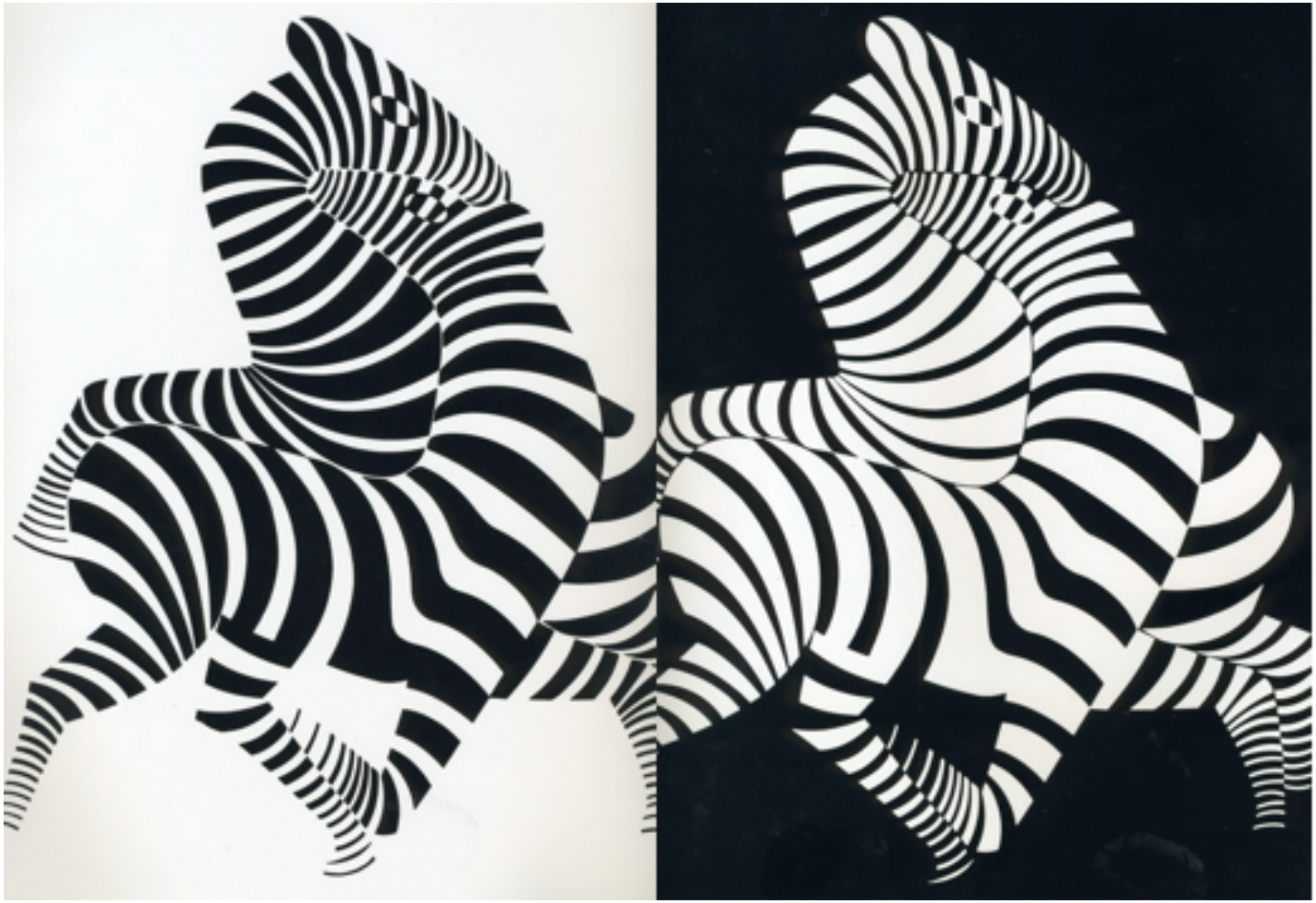


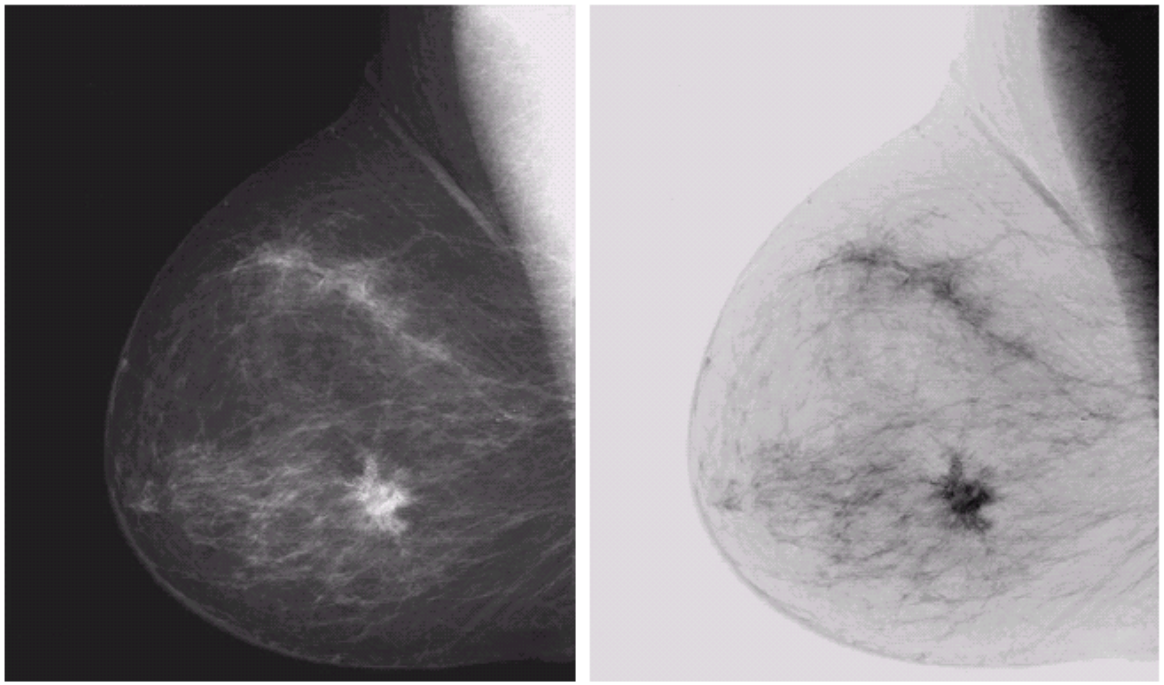
Negative Transformation

- Image negative is obtained by the given expression:

$s = L - 1 - r$, where intensity level range is [0,L-1]







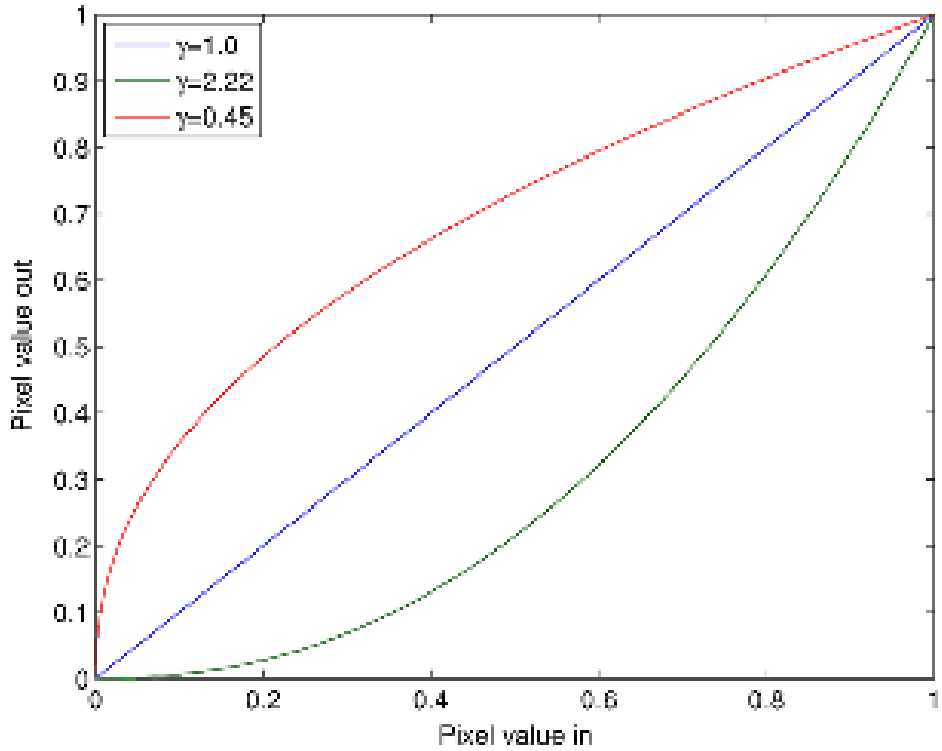
a b

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.)

Non-linear Gray-Level Mapping

Power Law (Gamma Mapping) Transformation

$$g(x, y) = f(x, y)^{\gamma}$$



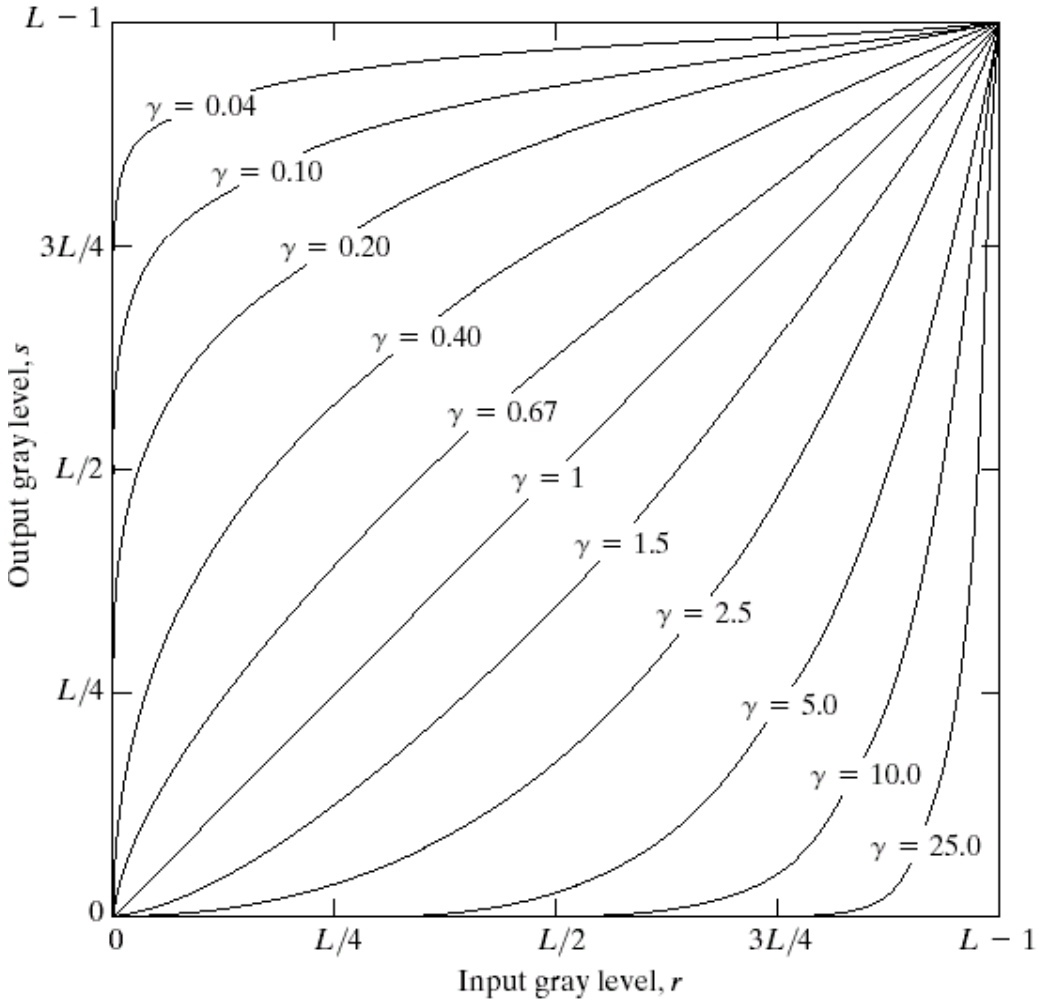


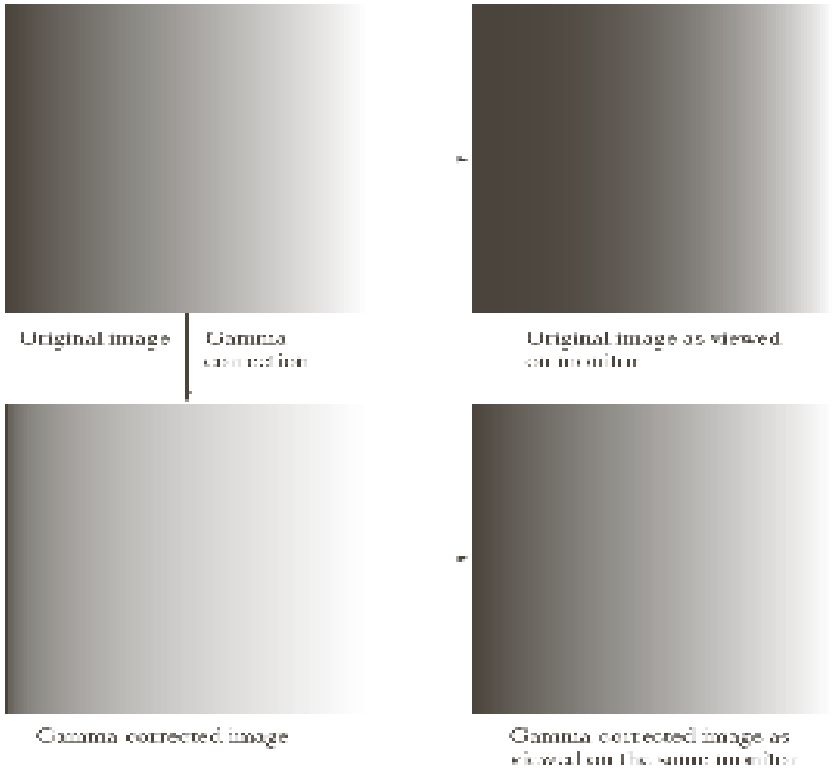
FIGURE 3.6 Plots of the equation $s = cr^{\gamma}$ for various values of γ ($c = 1$ in all cases).

Gamma value: 0.45

No gamma correction

Gamma value: 2.22

Gamma(Power-Law) Transformation



a b
c d

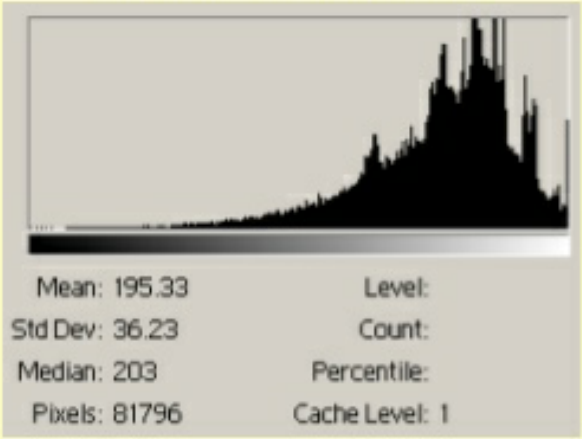
FIGURE 3.7
(a) Intensity ramp image. (b) Image as viewed on a simulated monitor with a gamma of 2.5. (c) Gamma-corrected image. (d) Corrected image as viewed on the same monitor. Compare (d) and (a).

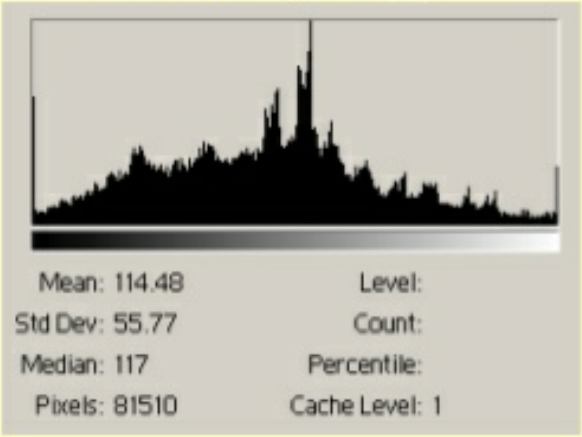
Gamma Transform - Example

Gamma Transform, c =1, γ =4.0

Original Aerial image

Transformed image





Logarithmic Mapping

$$g(x, y) = c \cdot \log\big(1 + f(x, y)\big)$$

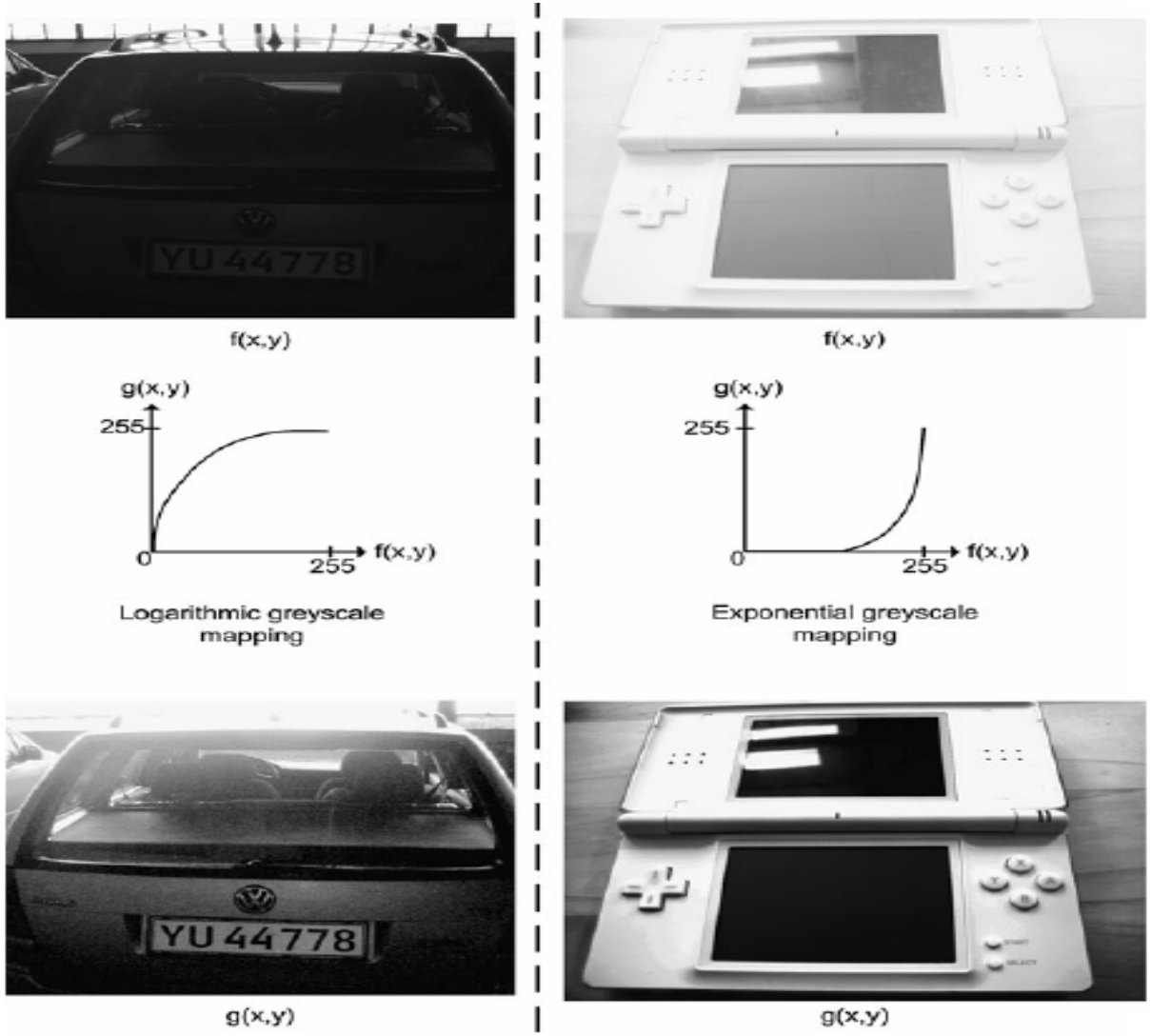
$$c = \frac{255}{\log(1 + v_{\max})}$$

Exponential Mapping

$$g(x, y) = c \cdot \left(k^{f(x,y)} - 1\right)$$

$$c = \frac{255}{k^{v_{\max}} - 1}$$

Fig.: Examples of logarithmic and exponential gray-level mappings. Logarithmic mapping is useful for bringing out details in dark images and exponential mapping is useful for bringing out details in bright images.



Piecewise-Linear Transformation Functions

- Intensity Level Slicing
- Contrast Stretching / Histogram stretching
- Bit-plane slicing

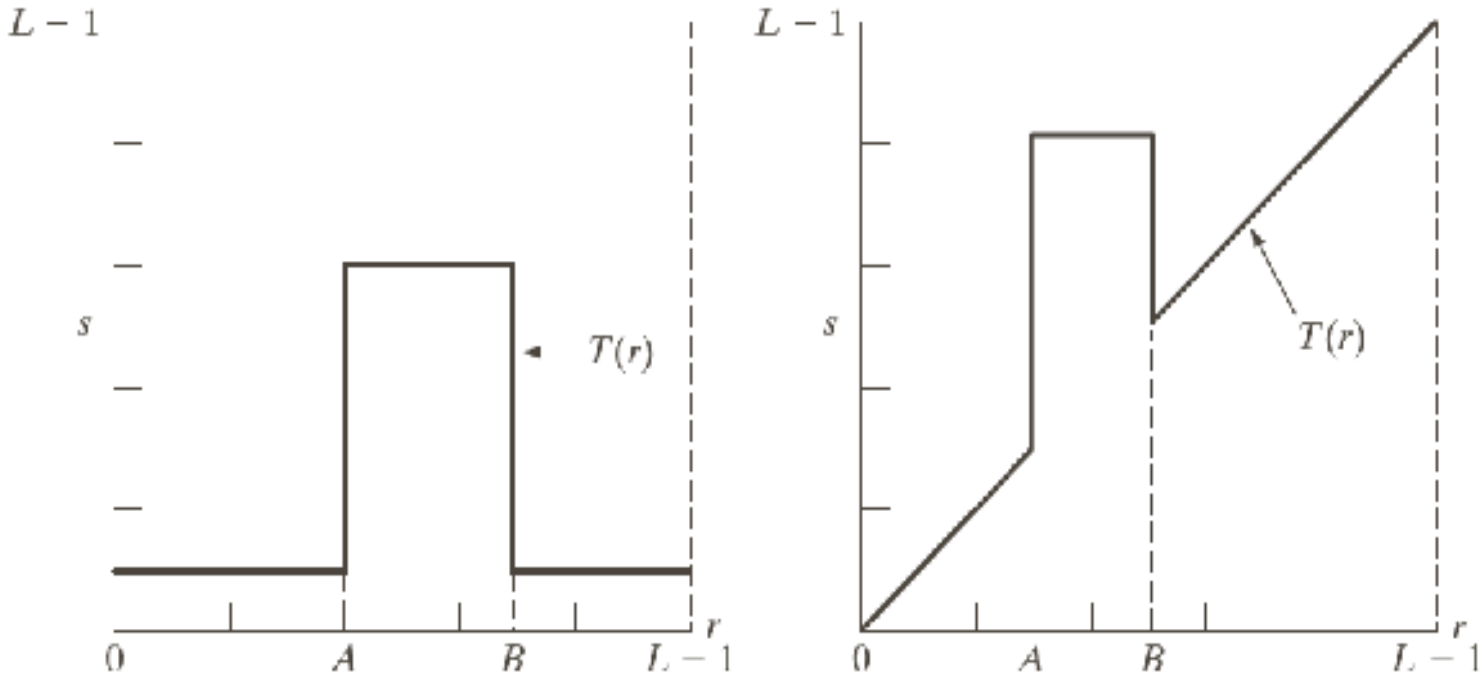
Intensity-level Slicing

- Highlighting a specific range of intensities of an image.
- There are two main different approaches:
  - Highlight a **range of intensities** while **diminishing all others to a constant low level**.
  - Highlight a range of intensities but **preserve all others**.

Intensity-level Slicing

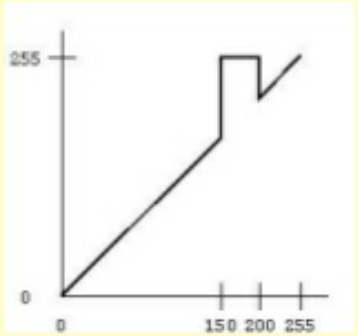
a b

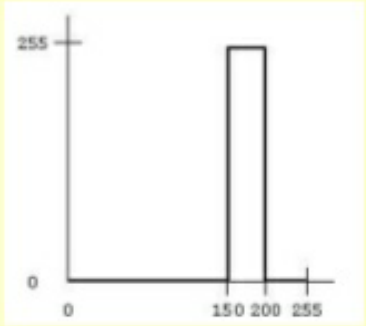
FIGURE 3.11 (a) This transformation highlights intensity range $[A, B]$ and reduces all other intensities to a lower level. (b) This transformation highlights range $[A, B]$ and preserves all other intensity levels.



Gray Level Slicing - Examples

Original image





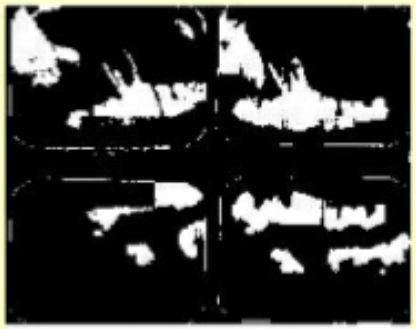
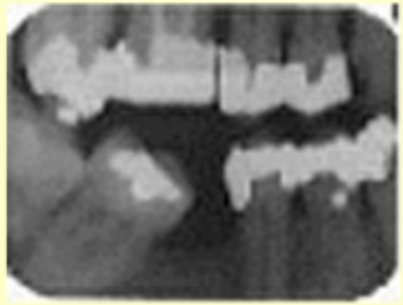


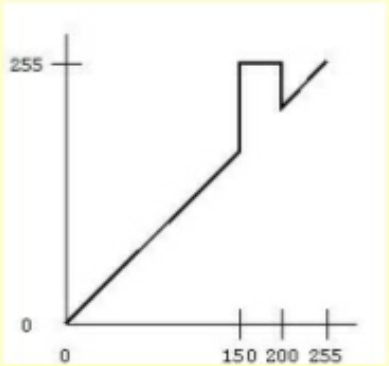
Image sliced to emphasize gray values from 150 to 200; background changed to black.

Background changed to black

Gray Level Slicing Effects - Examples



Original image



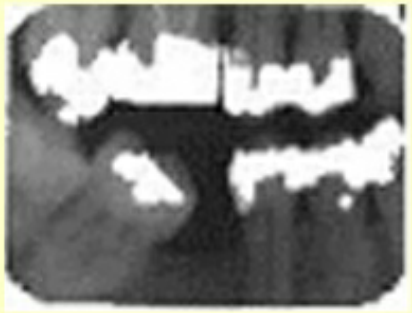
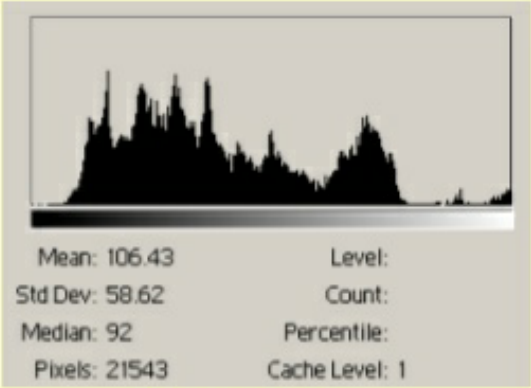
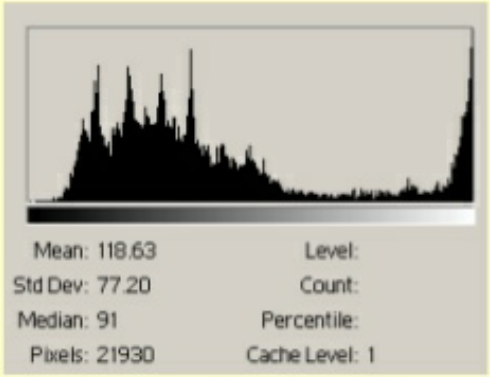


Image after slicing

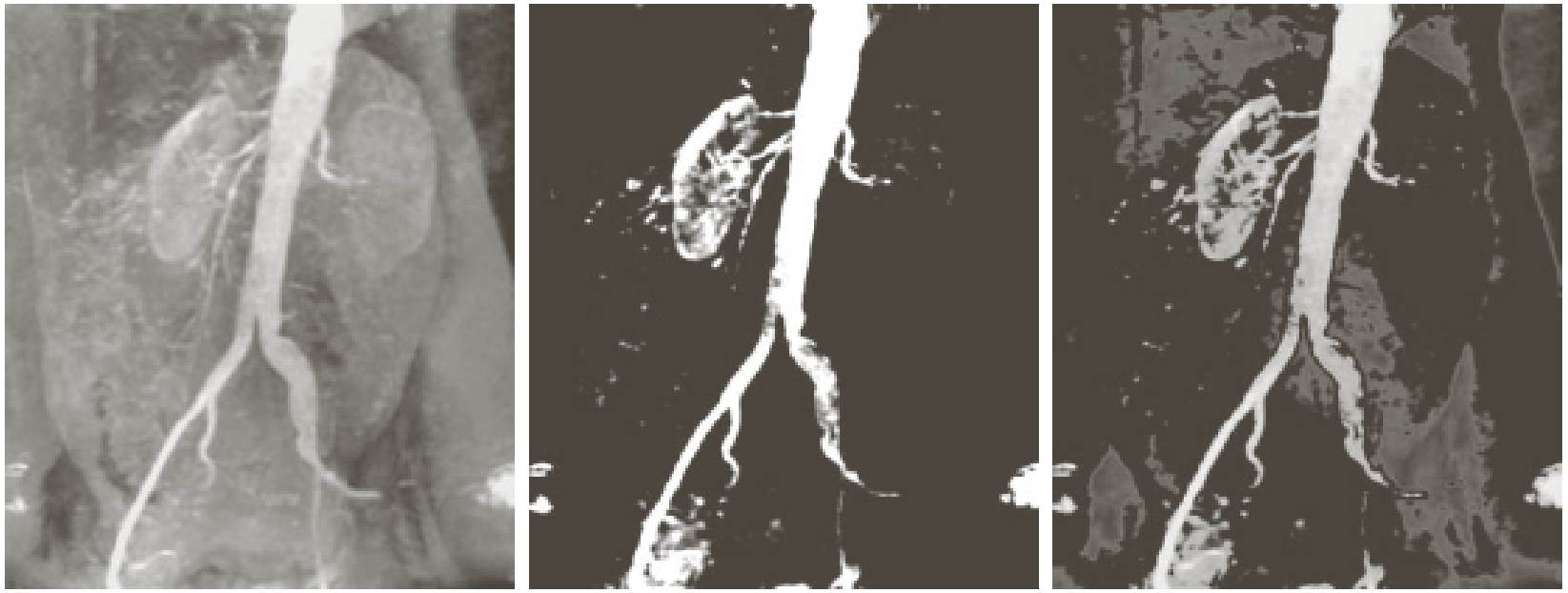


Histogram Before



Histogram after slicing

Background kept as it is



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

- Highlighting the contribution made to total image appearance by **specific bits**.
- Suppose that each pixel in an image is represented by 8 bits. Imagine the image is composed of 8, 1-bit planes ranging from bit plane 0 (LSB) to bit plane 7 (MSB).

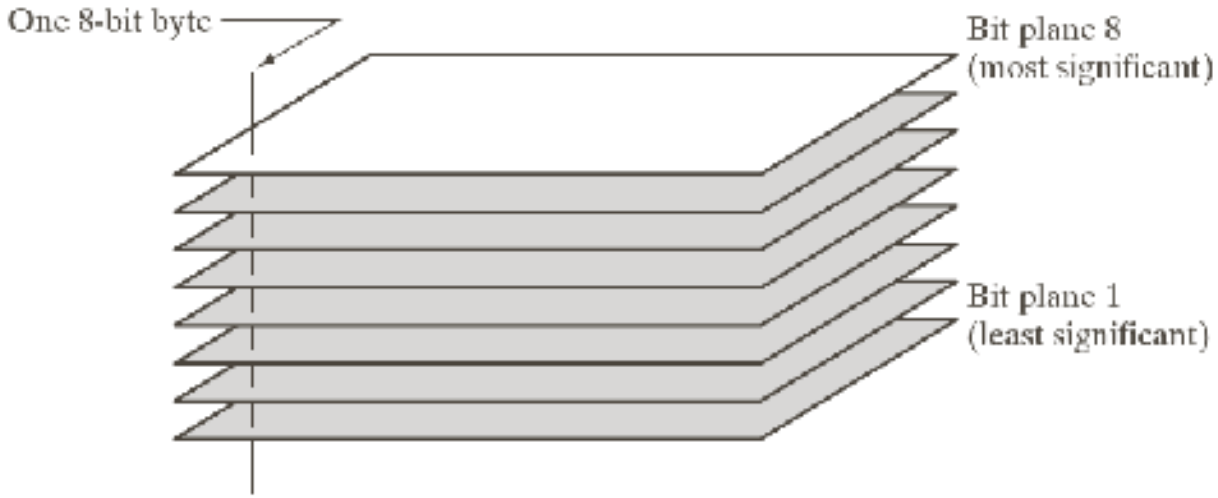
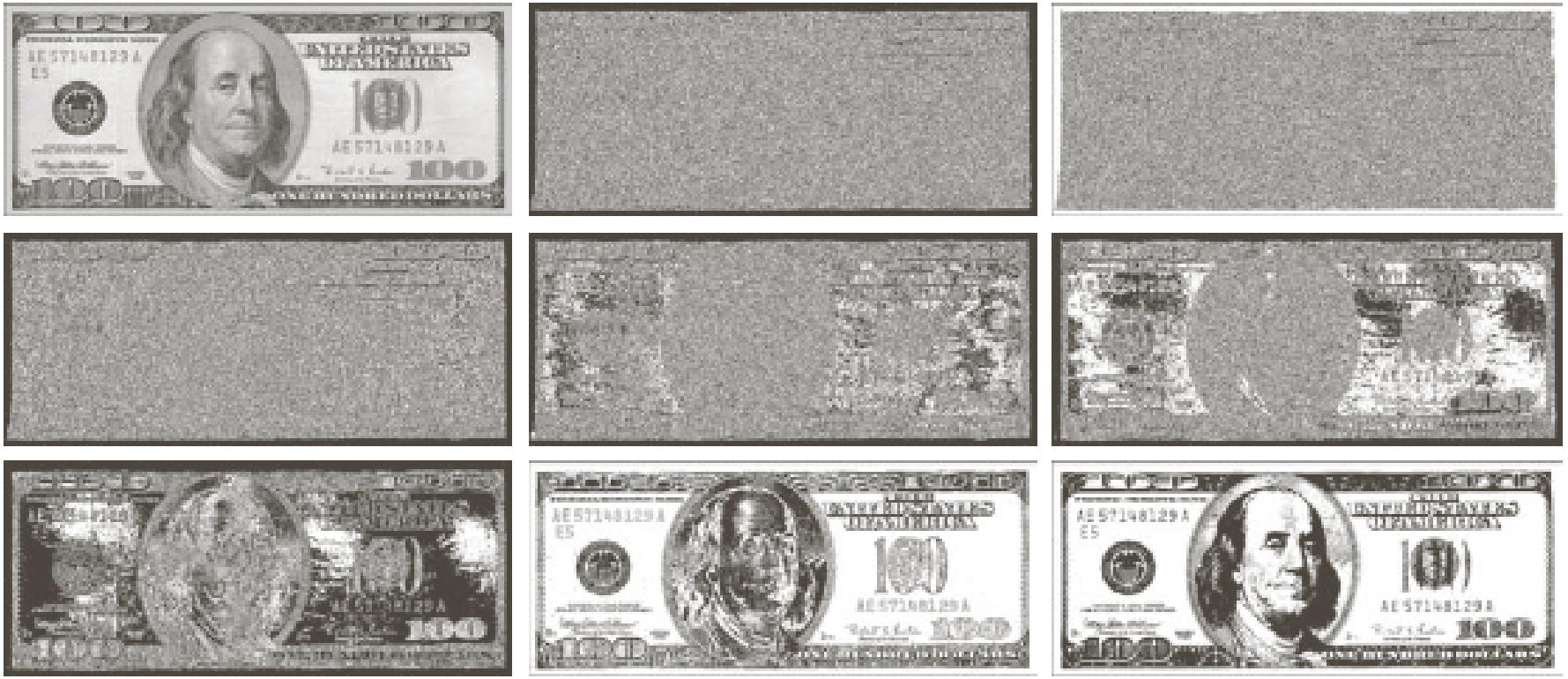


FIGURE 3.13
Bit-plane representation of an 8-bit image.

- What is the **importance** of it?
  - Separating a digital image into its bit planes is useful for analysing the relative importance played by each bit of the image.
  - It determines the adequacy of numbers of bits used to quantize each pixel , **useful for image compression**.

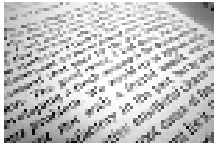
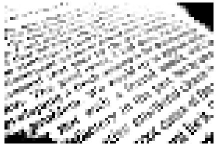
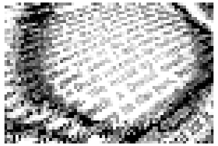
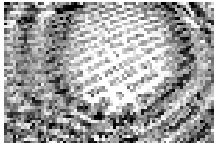
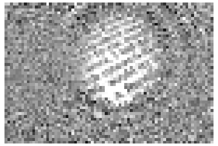
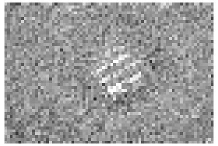
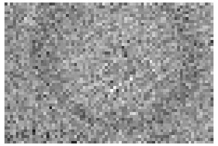
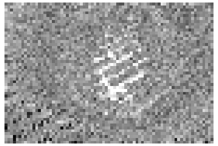
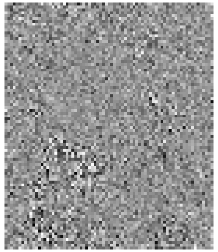
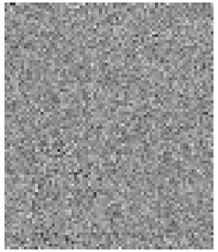
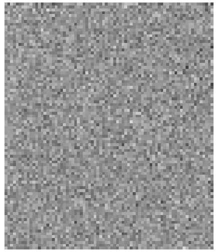
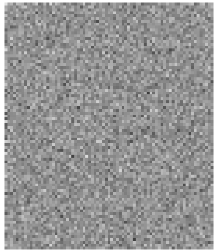
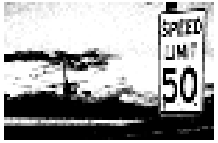
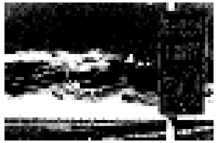
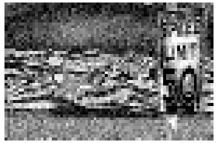
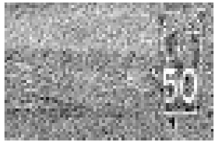
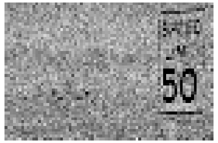
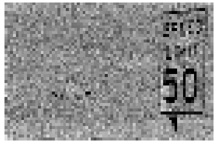


a b c
d e f
g h i

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).

Original Image	Bit Plane 8	Bit Plane 7	Bit Plane 6	Bit Plane 5	Bit Plane 4	Bit Plane 3	Bit Plane 2	Bit Plane 1
								
								
								

Histogram Processing

- The Histogram of a digital image with intensity level in the range [0,L-1] is a discrete function

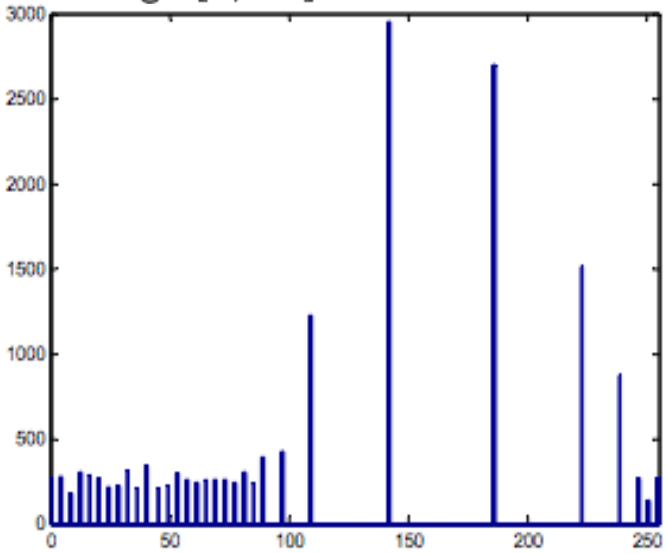
$$h(r_k) = n_k$$

where r_k is the k[th] intensity value and n_k is the number of pixels in the range [0,L-1] with the intensity r_k.

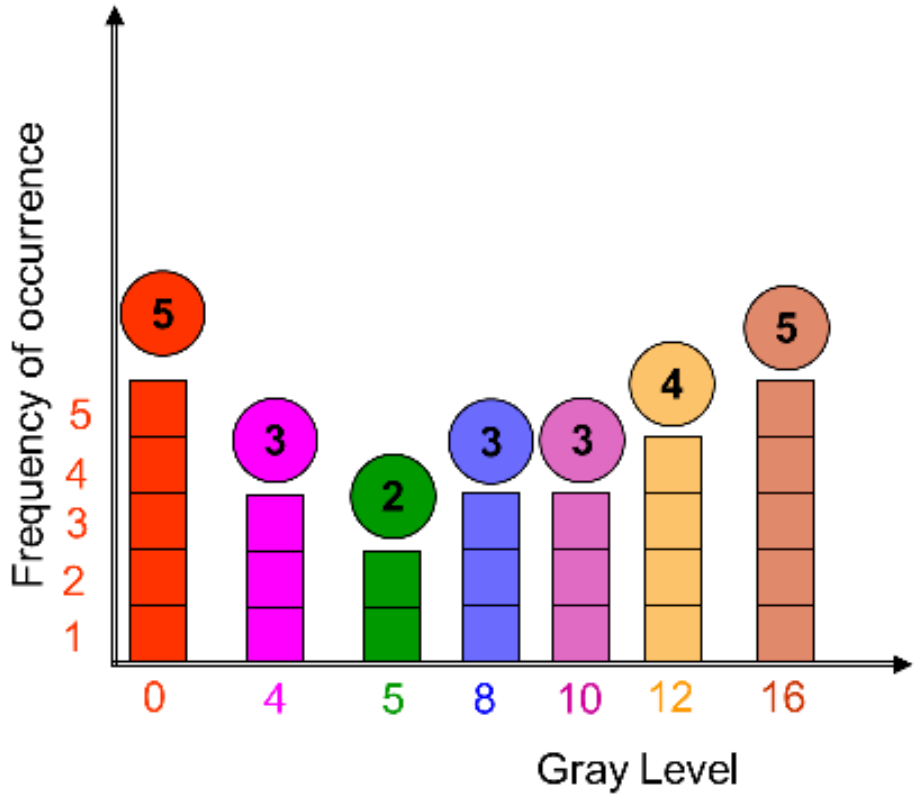
- A normalized histogram is given by:

$$p(r_k) = \frac{n_k}{MN},$$

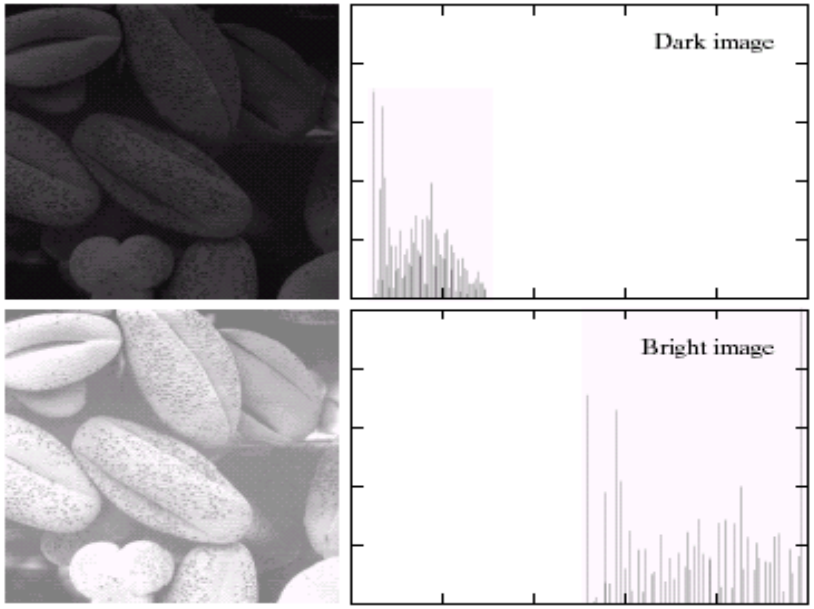
where *MN* is total number of pixels in the Image.

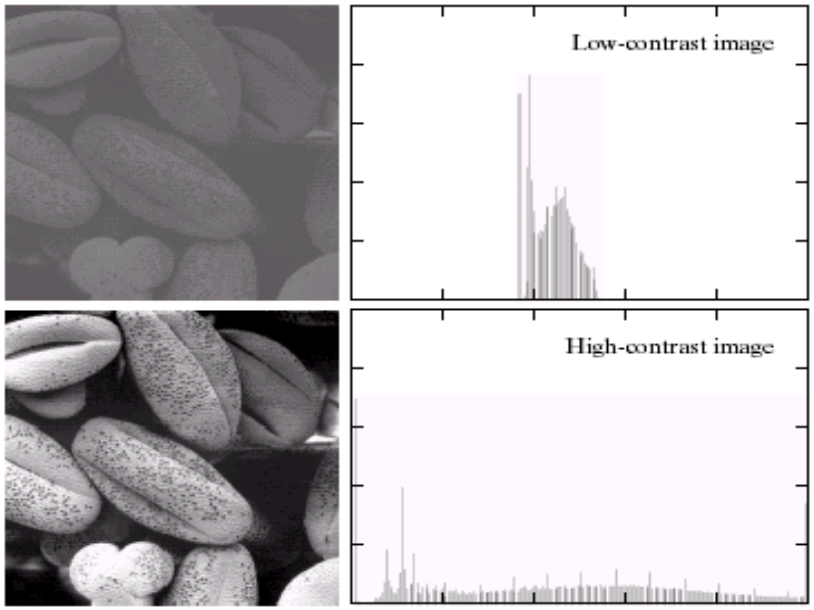


0	4	8	10	12
12	16	5	0	16
4	16	8	5	10
10	0	4	12	16
12	0	16	0	8

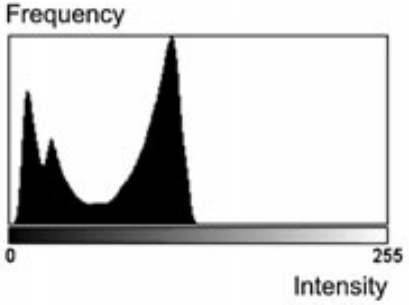


Histogram Processing

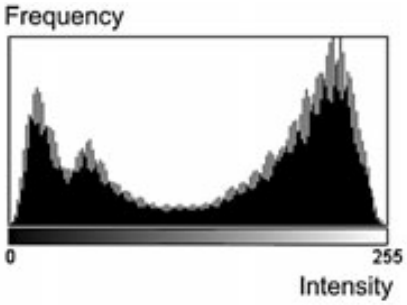




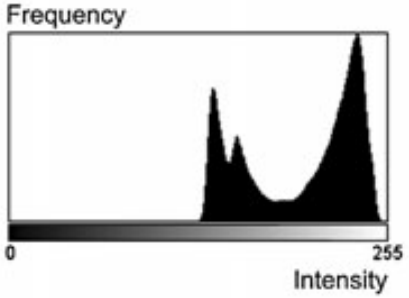
Dark image



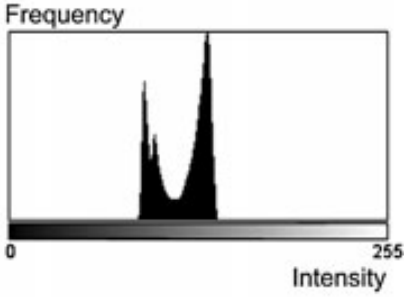
High contrast image



Bright image



Low contrast image



Contrast Stretching

- Contrast/Histogram stretching is a process that expands the range of intensity levels in an image so that it spans the full intensity range of the display device.

  r_{min} *is the minimum intensity level and* r_{max} *is the maximum intensity level of the image.*

- The contrast stretching is obtained by setting:

$$(r_1, s_1) = (r_{min}, 0) \text{ and } (r_2, s_2) = (r_{max}, L-1)$$

- The transformation function stretches the levels linearly

From their original range to the full range [0,L-1].

$$s = \frac{255}{r_{max} - r_{min}} (r - r_{min})$$

Histogram Stretching

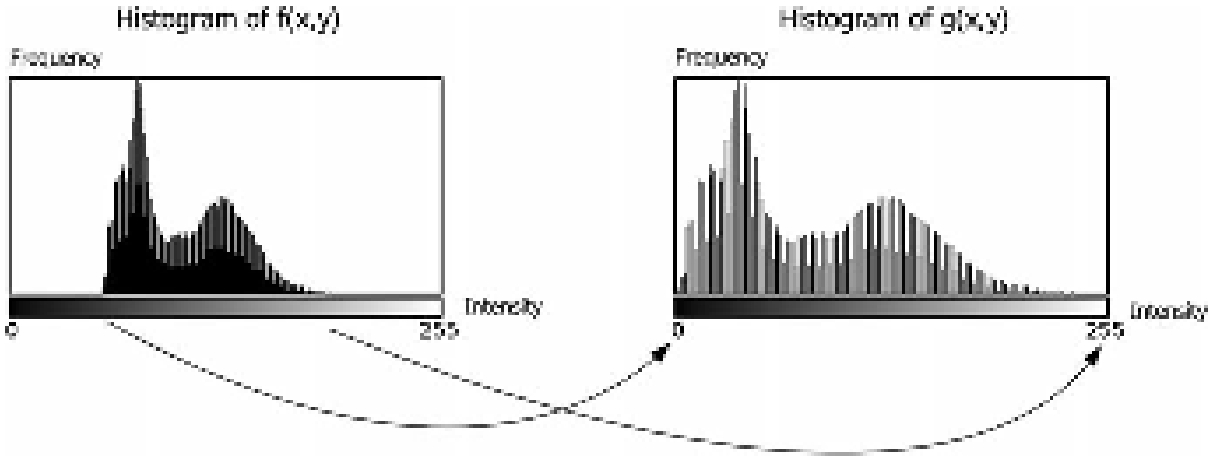
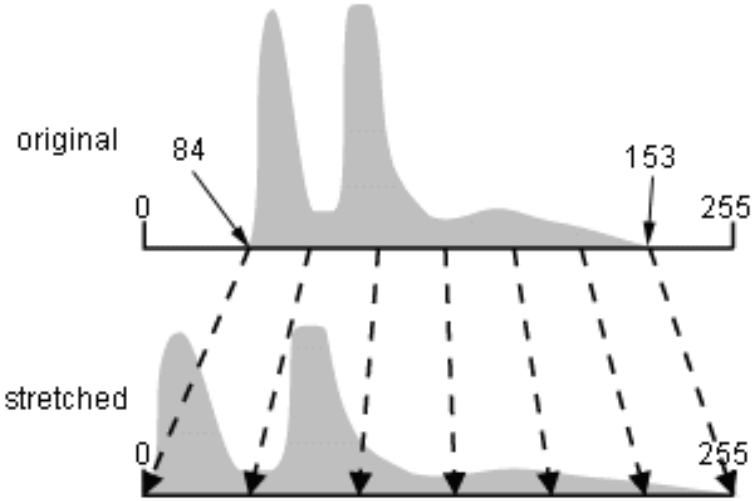


Fig. The concept of histogram stretching

$$g(x,y) = \frac{255}{f_2 - f_1} \cdot (f(x,y) - f_1)$$



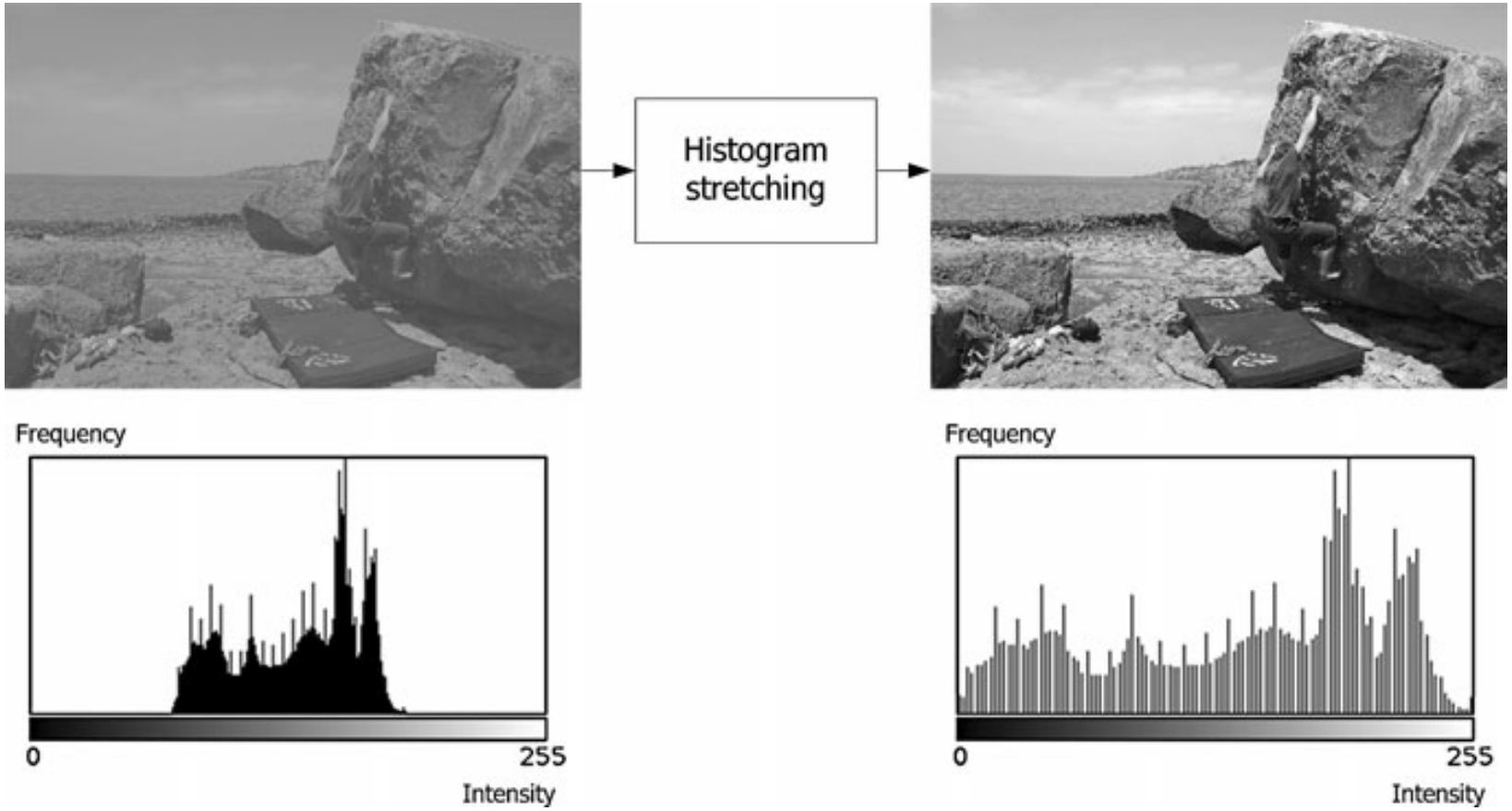


Fig. An example of histogram stretching

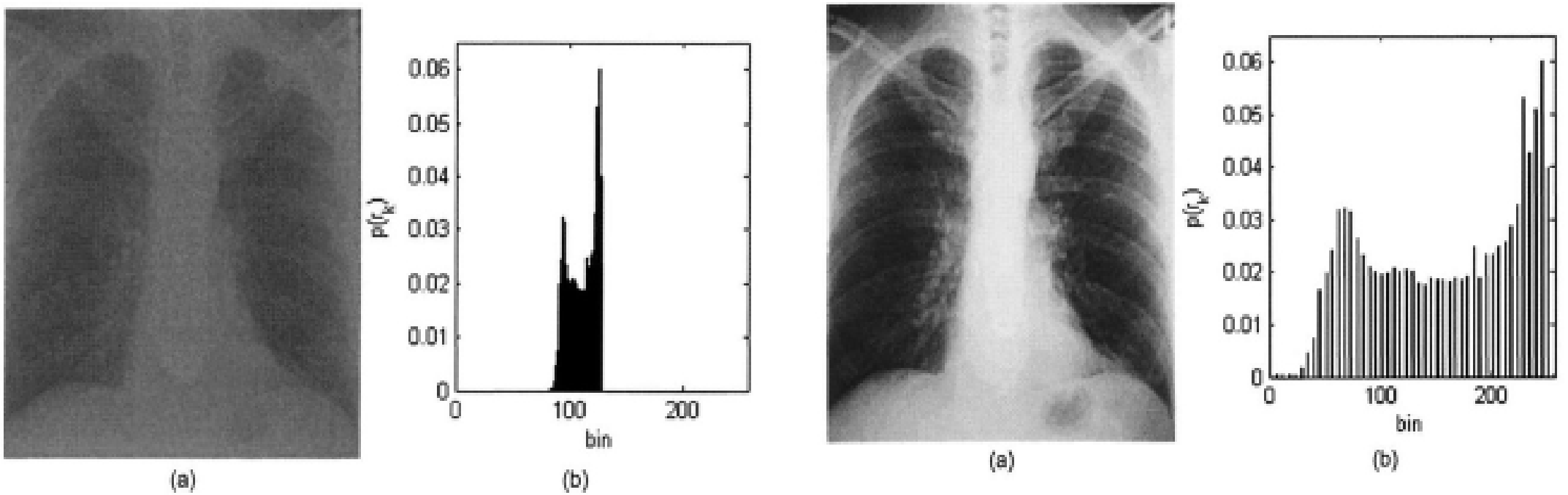
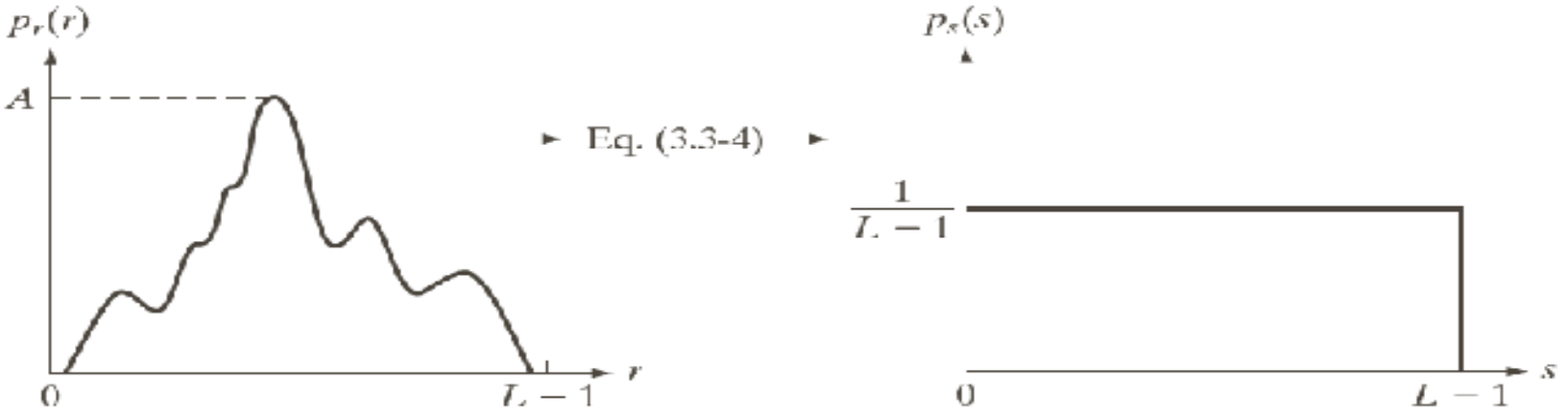


Fig. An example of histogram stretching

- If just one pixel has the value 0 and another 255, histogram stretching will not work, since *f2 −f1 = 255.*
- *A solution is modified histogram stretching where small* bins in the histogram are removed by changing their values to those of larger bins.
- But if a significant number of pixels with very small and very high values exit, we still have *f2−f1 = 255, and hence the histogram (and image) remains the same*.
- A more robust method to improve the histogram (and image) is therefore to apply *histogram equalization.*

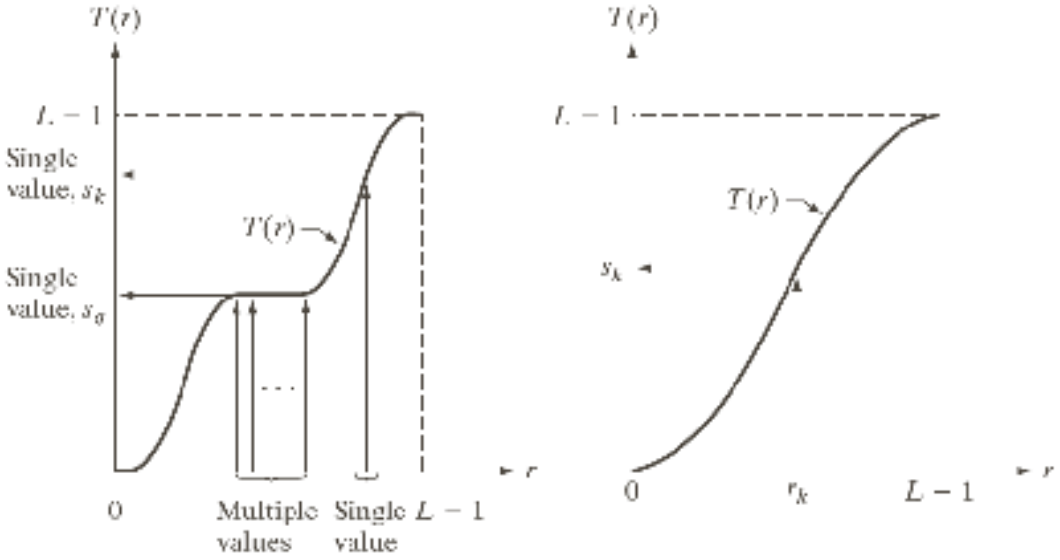
Histogram Equalization

- The histogram equalization is an approach to enhance a given image.
- The approach is to design a transformation *T(.)* such that the gray values in the output is **uniformly distributed** in [0, 1].
- This usually results in an enhanced image, with an **increase in the dynamic range of pixel values**.



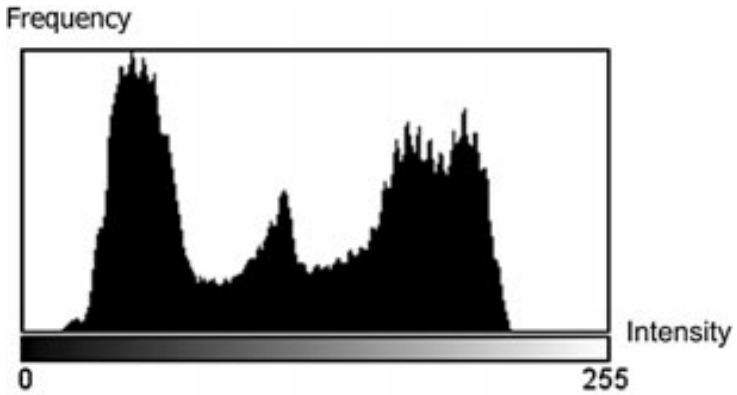
a b

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.



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.



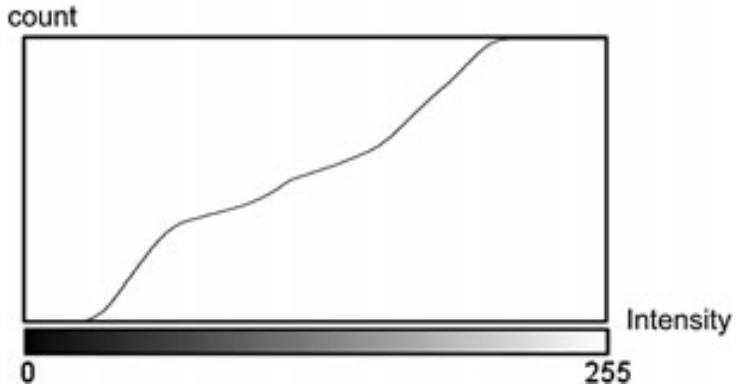


Image size = MxN

Histogram = $H(k), k = 0,1,2 \dots 255$

Normalized Histogram = $p(r_k) = \frac{n_k}{MN}$

Histogram Equalization:

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

$$s_k = \frac{L-1}{MN} \sum_{j=0}^{k} n_j$$

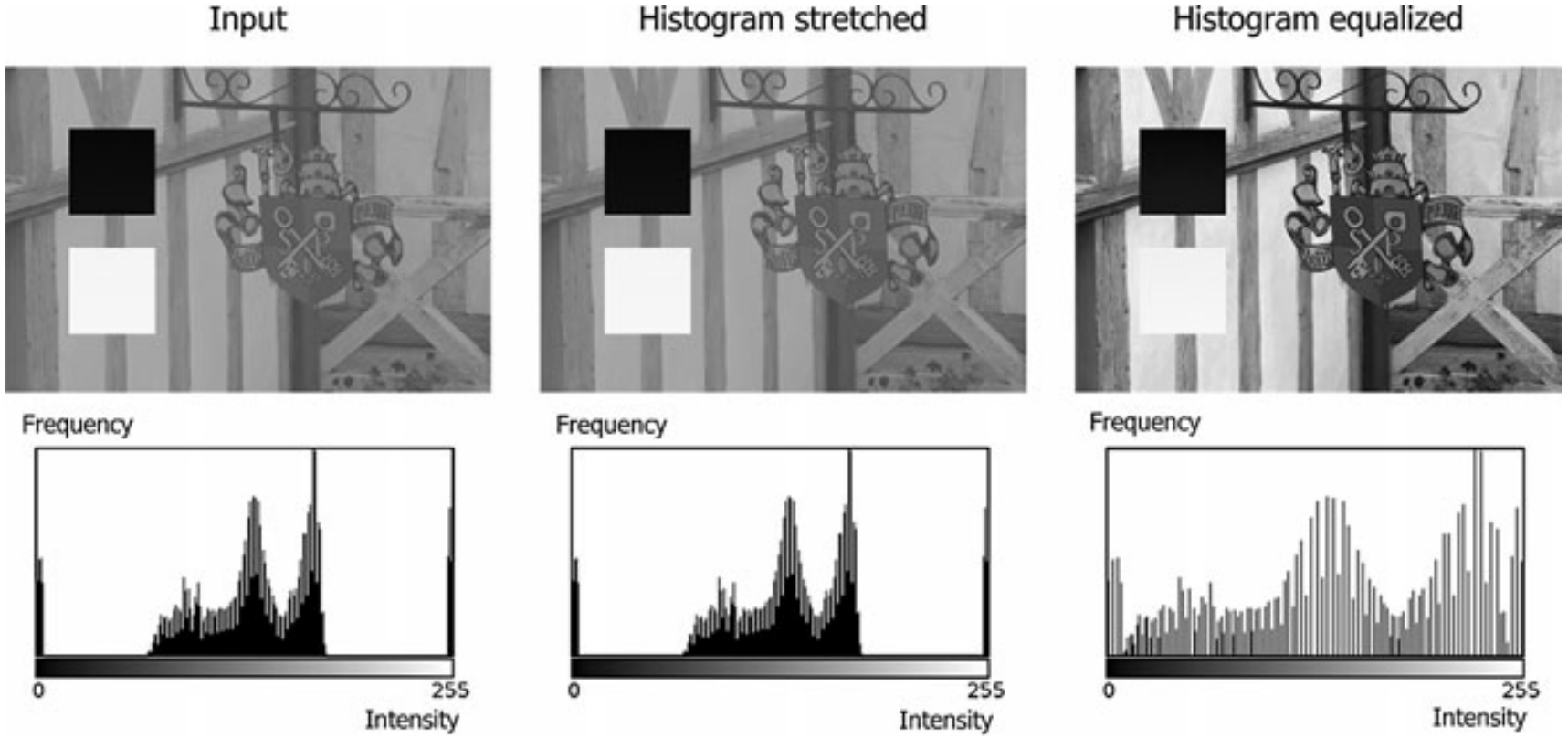


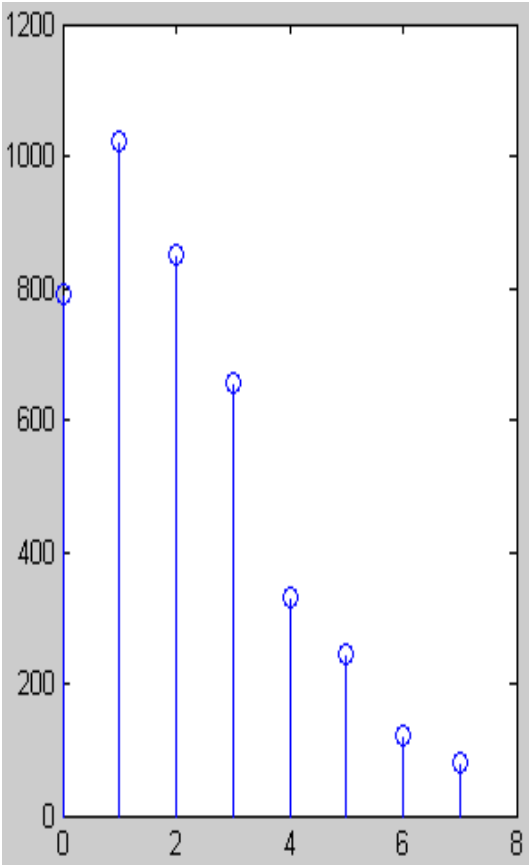
Fig. The effect of histogram stretching and histogram equalization on an input image with both very high and very low pixel values

Example:

- Consider an 8-level 64 x 64 image with gray values (0, 1, …,7). The normalized gray values are (0, 1/7, 2/7, …, 1). The normalized histogram is given below:

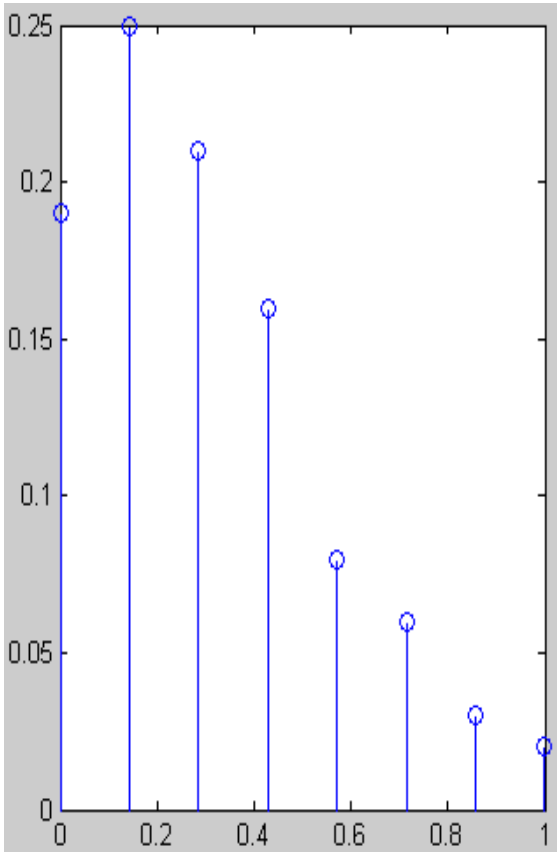
k	r_k	n_k	$p(r_k) = n_k/n$
0	0	790	0.19
1	1/7	1023	0.25
2	2/7	850	0.21
3	3/7	656	0.16
4	4/7	329	0.08
5	5/7	245	0.06
6	6/7	122	0.03
7	1	81	0.02

pixels



Gray value

Fraction of # pixels



Normalized gray value

Example: Histogram Equalization

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

$$s_0 = T(r_0) = 7\sum_{j=0}^{0} p_r(r_j) = 7 \times 0.19 = 1.33 \rightarrow 1$$

$$s_1 = T(r_1) = 7\sum_{j=0}^{1} p_r(r_j) = 7 \times (0.19 + 0.25) = 3.08 \rightarrow 3$$

$$s_2 = 4.55 \rightarrow 5 \qquad s_3 = 5.67 \rightarrow 6$$

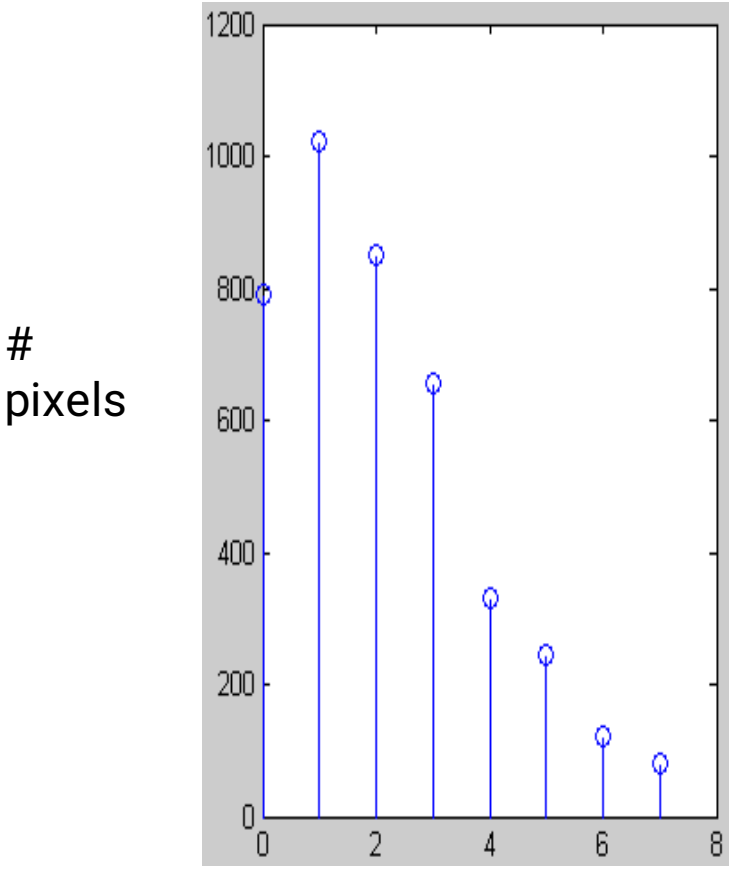
$$s_4 = 6.23 \rightarrow 6 \qquad s_5 = 6.65 \rightarrow 7$$

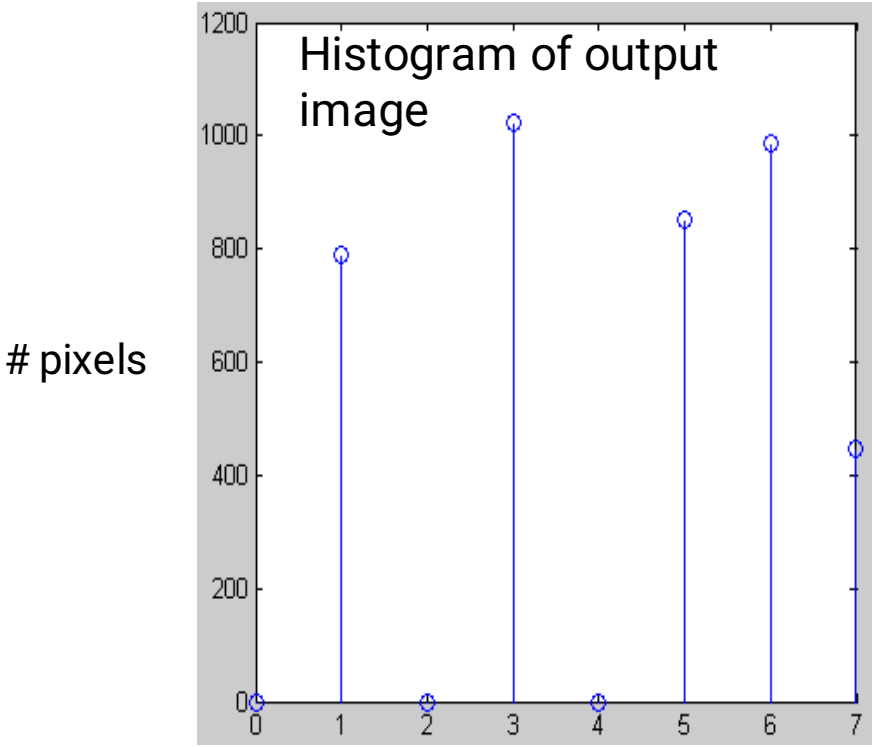
$$s_6 = 6.86 \rightarrow 7 \qquad s_7 = 7.00 \rightarrow 7$$

- Notice that there are only five distinct gray levels -- (1, 3,5,6,7) in the output image. We will relabel them as $(s_0, s_1, \ldots, s_4)$.

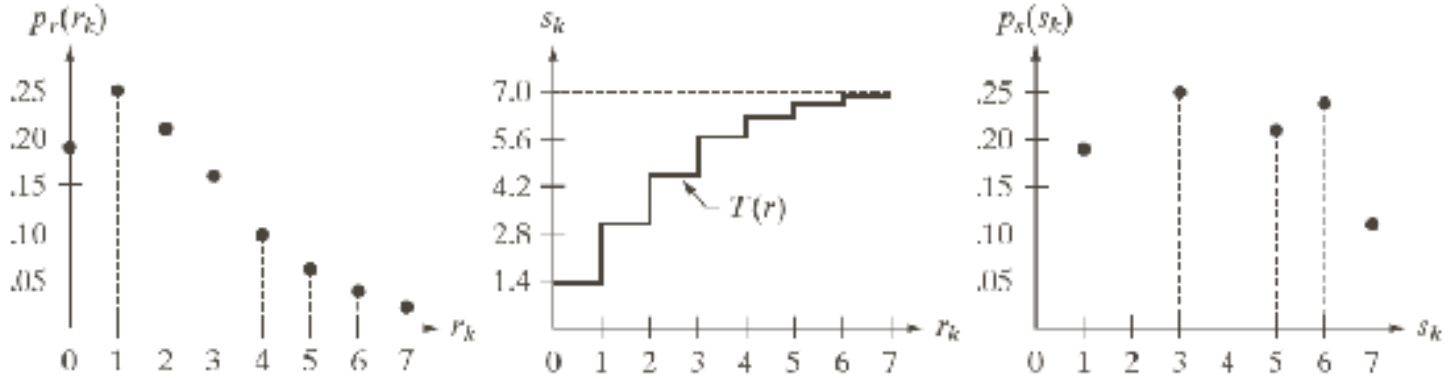
- With this transformation, the output image will have histogram

k	s_k	n_k	$p(s_k) = n_k/n$
0	1/7	790	0.19
1	3/7	1023	0.25
2	5/7	850	0.21
3	6/7	985	0.24
4	1	448	0.11





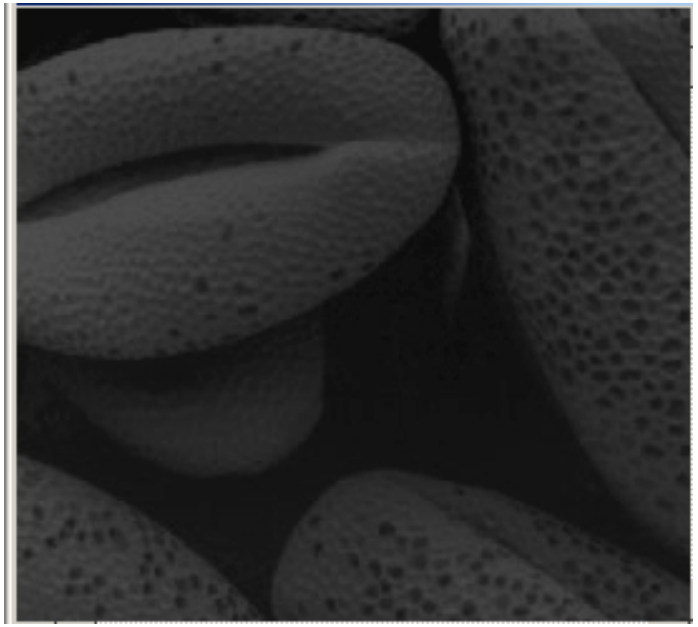
- Note that the histogram of output image is only approximate, and not exactly, uniform.

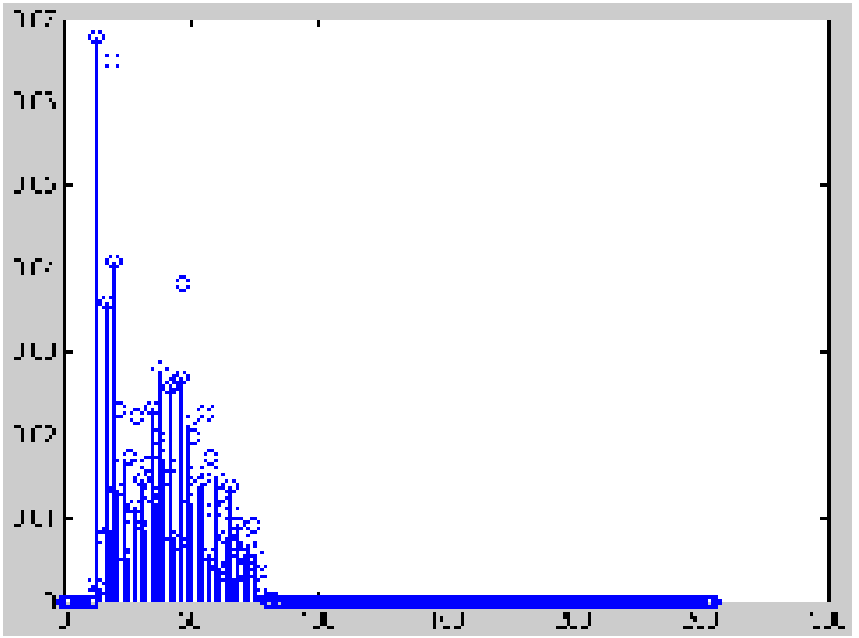


a b c

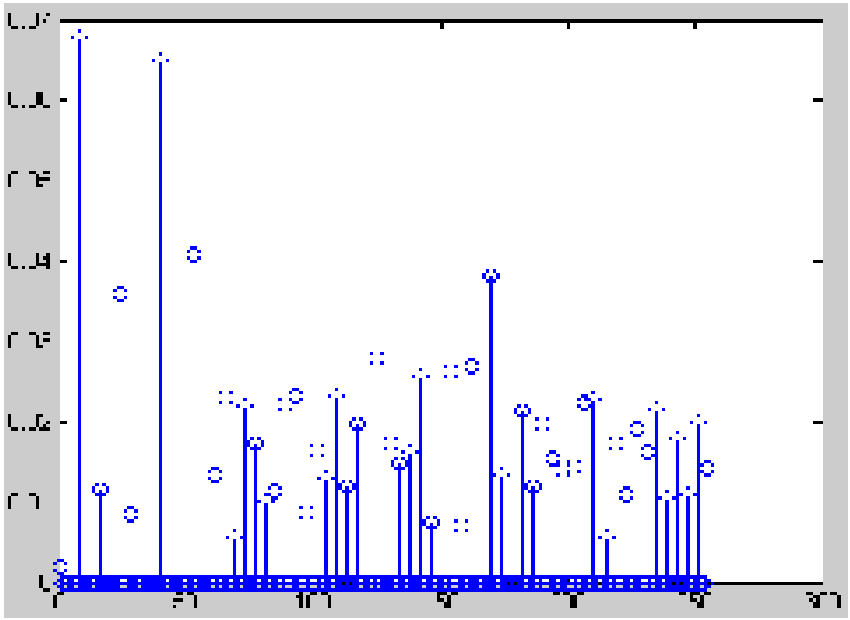
FIGURE 3.19 Illustration of histogram equalization of a 3-bit (8 intensity levels) image. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

Original image and its Histogram



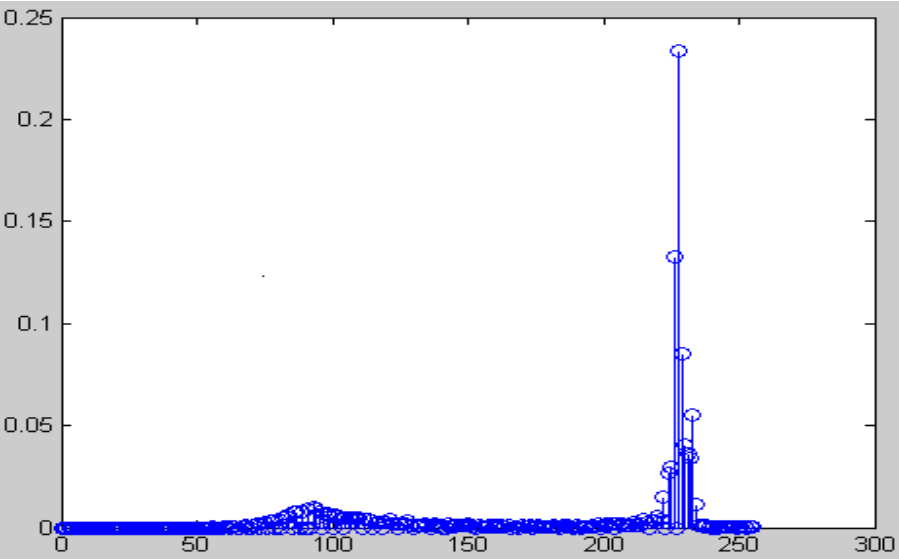


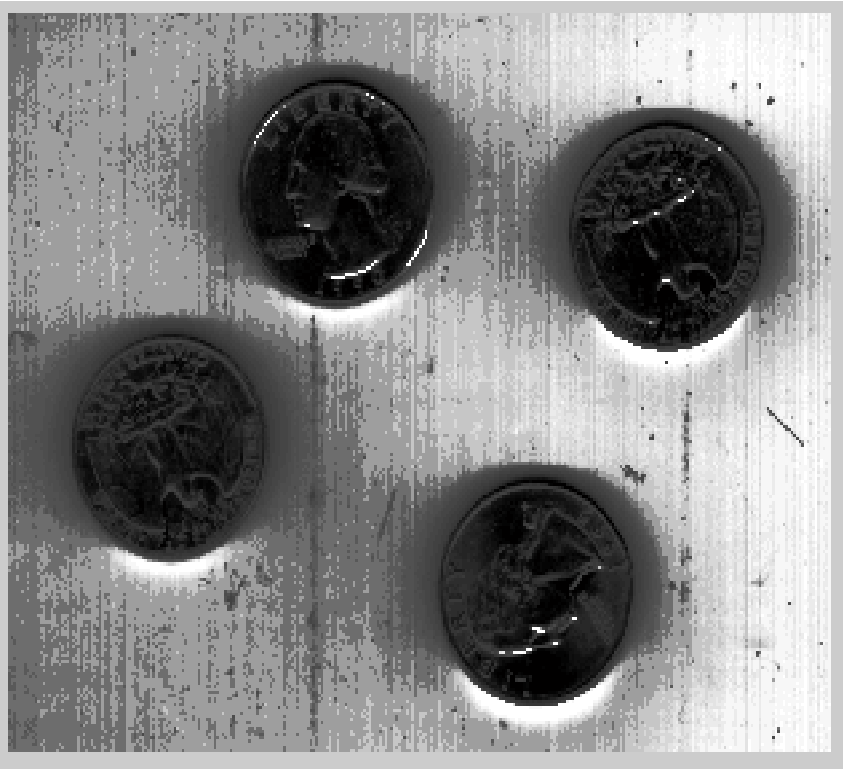
Histogram equalized image and its Histogram

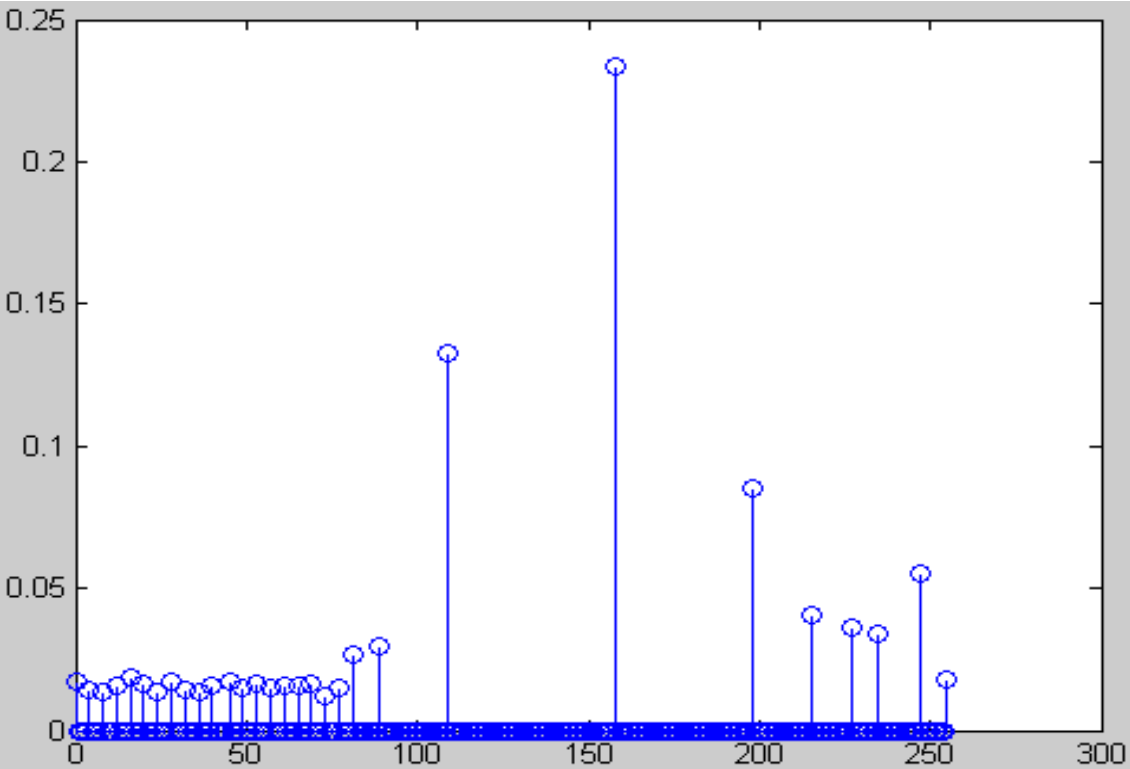


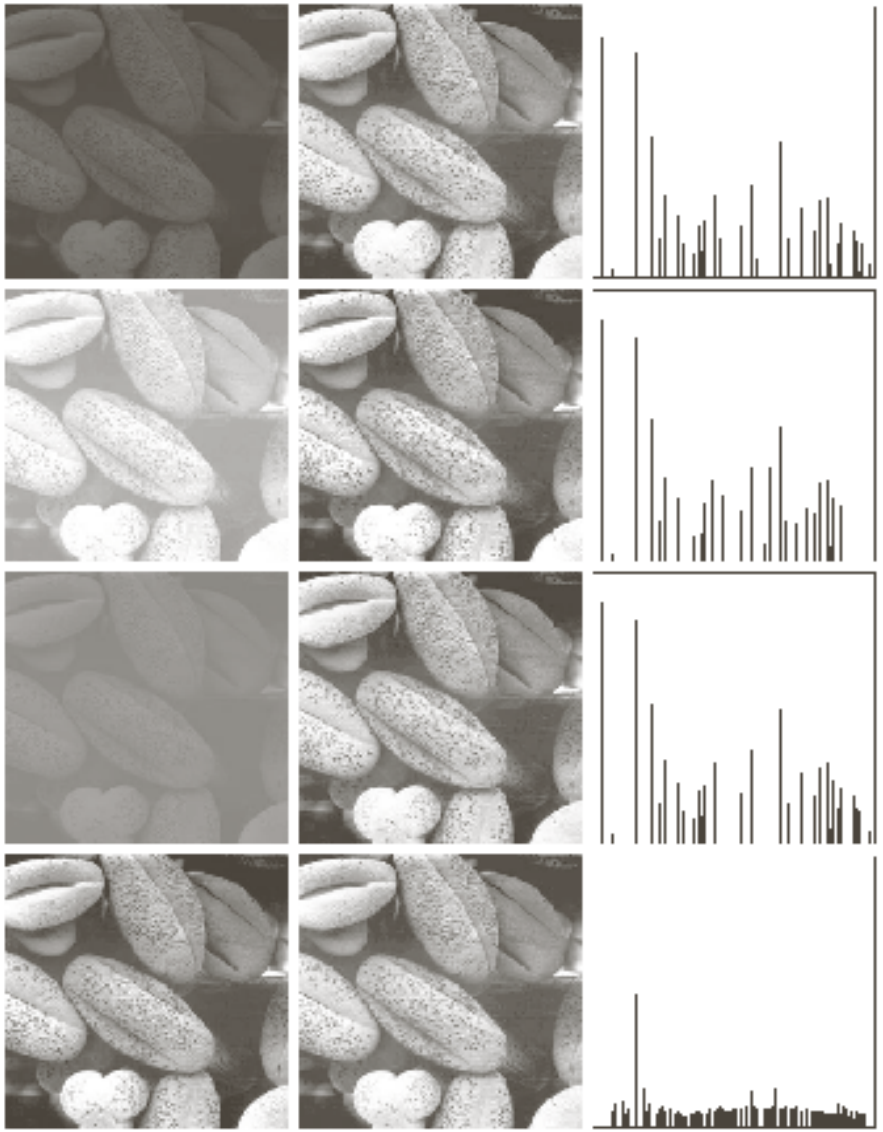
Comments:

- Histogram equalization may not always produce desirable results, particularly if the given histogram is very narrow.
- It can produce false edges and regions. It can also increase image “graininess” and “patchiness.”






0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300



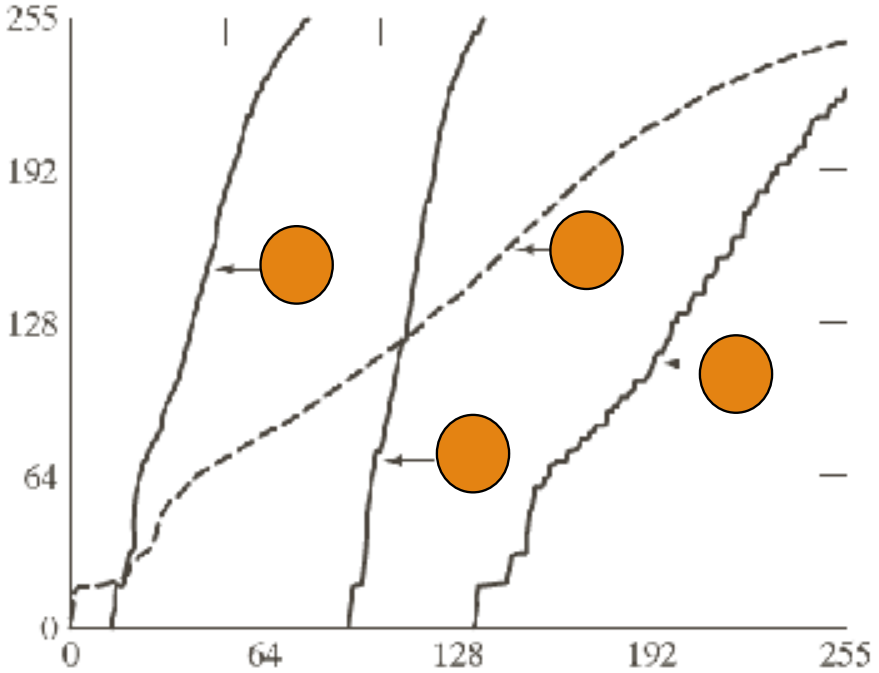


FIGURE 3.21 Transformation functions for histogram equalization. Transformations (1) through (4) were obtained from the histograms of the images (from top to bottom) in the left column of Fig. 3.20 using Eq. (3.3-8).

Histogram Specification

- Histogram equalization yields an image whose pixels are (in theory) uniformly distributed among all gray levels.

- Sometimes, this may not be desirable. Instead, we may want a transformation that yields an **output image** with a **pre-specified histogram**. This technique is called **histogram specificatio**n.

- In other words Transforming the intensity values so that the histogram of the output image aapproximately matches a specified histogram.

Given Information:

(1) Input image from which we can compute its histogram .

(2) Desired histogram.

Goal:

Derive a point operation, $H(r)$, that maps the input image into an output image that has the user-specified histogram.

Histogram Specification: Procedure

- Equalize the levels of the original image.
- Specify the desired density function (histogram) and obtain the transformation function G(z).
- Apply the inverse transformation function z = G(s) to the levels obtained in first step.

Histogram Specification: Procedure

- Obtain $p_r(r_j)$ from the input image and then obtain the values of s_k, round the value to the integer range [0, L-1].

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

- Use the specified PDF and obtain the transformation function $G(z_q)$, round the value to the integer range [0, L-1].

$$G(z_q) = (L-1)\sum_{i=0}^{q} p_z(z_i) = s_k$$

Where $p_z(z)$ is the specified PDF.

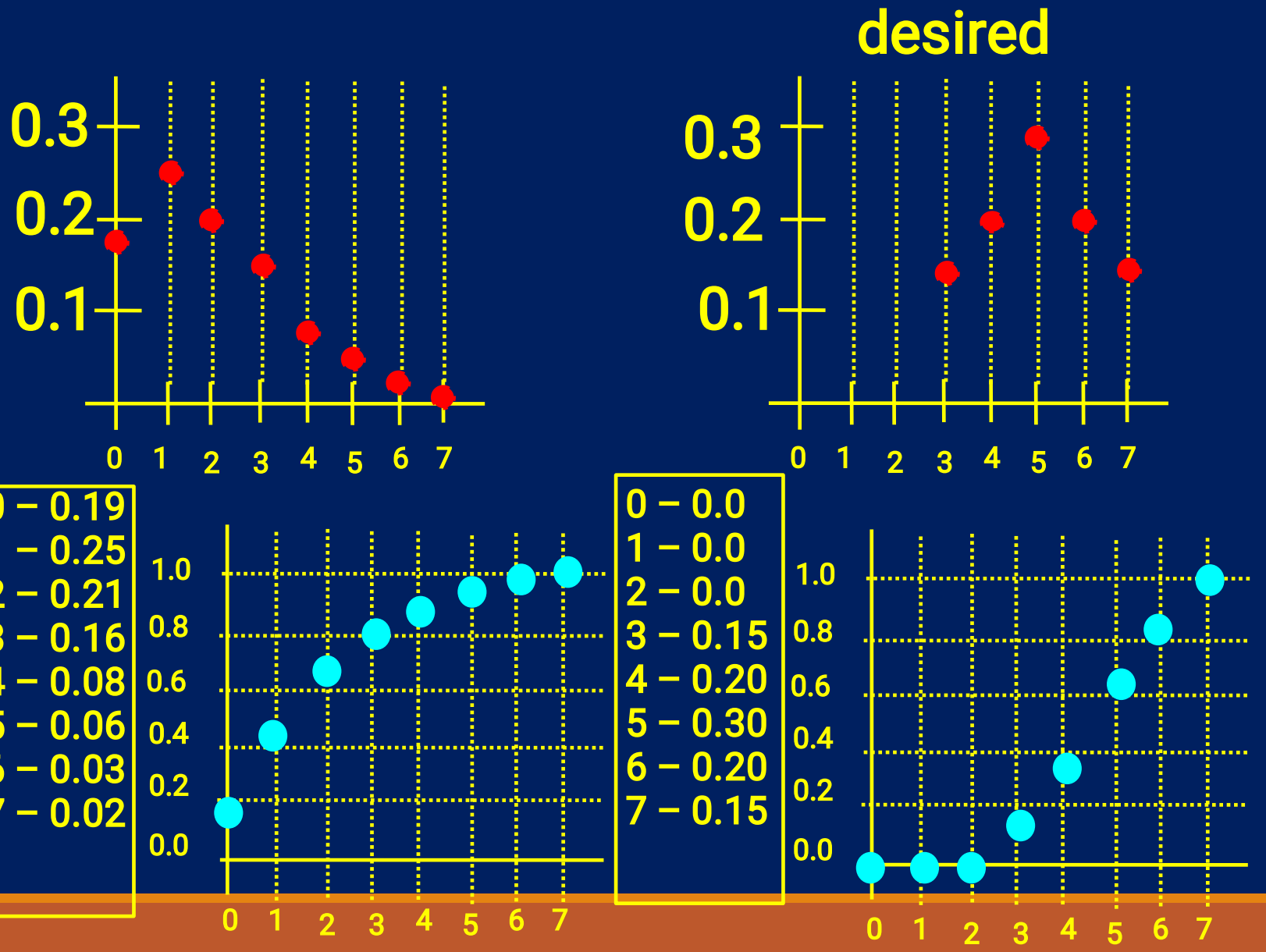
Histogram Specification: Procedure

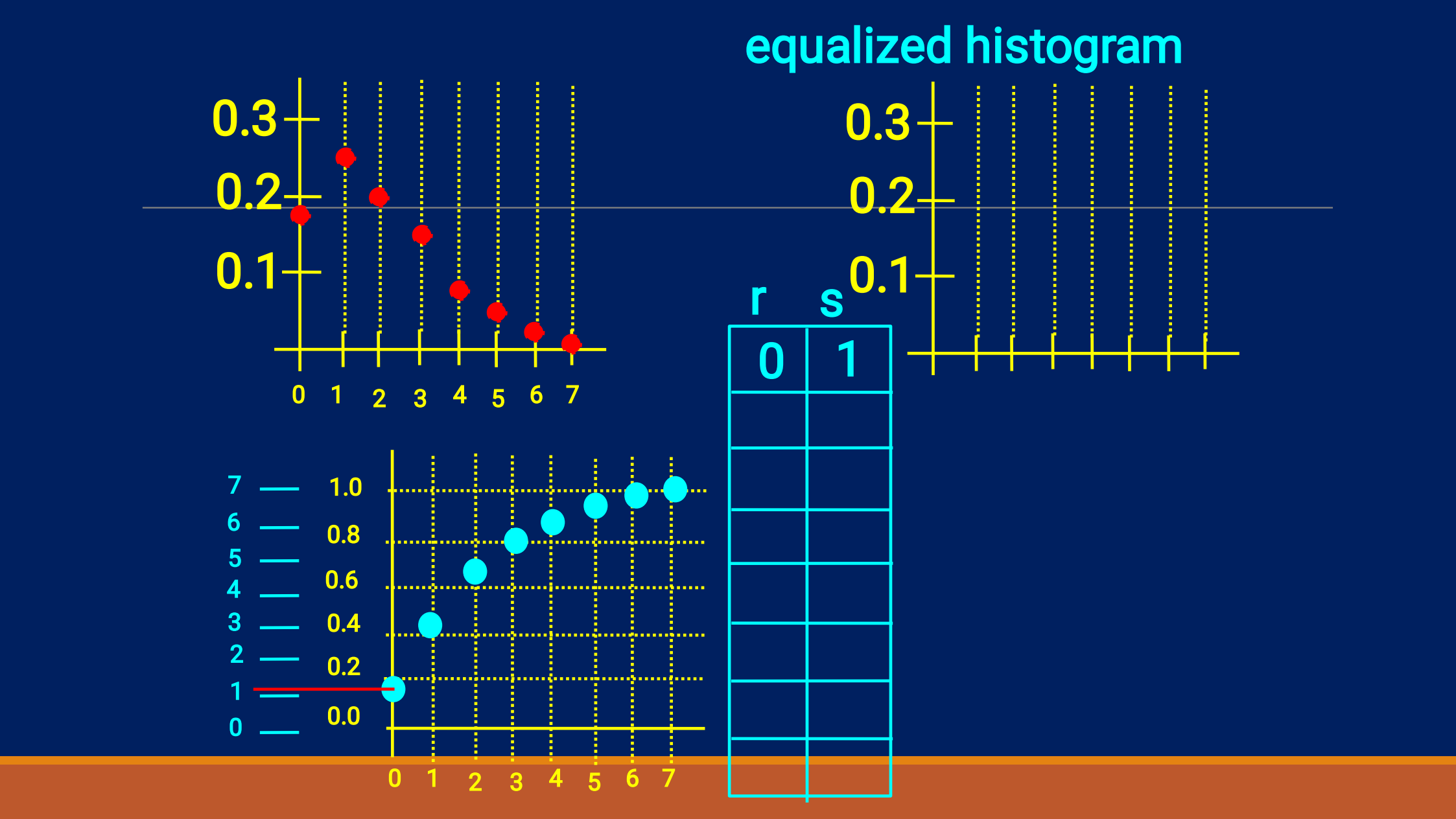
- Mapping from s_k to z_q

$$z_q = G^{-1}(s_k)$$

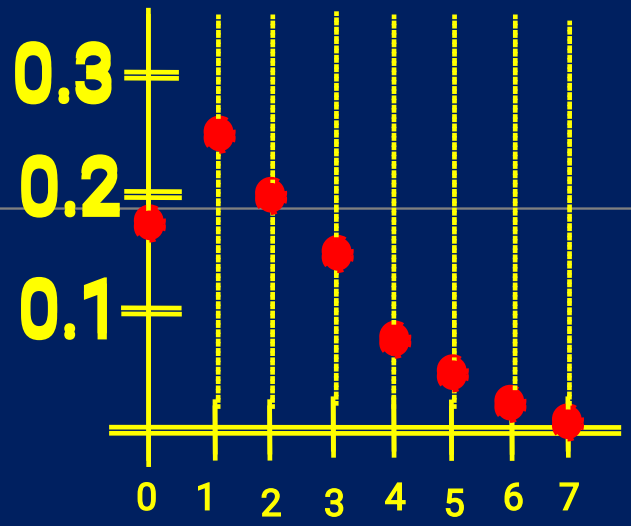
• If the transformation $z_k \rightarrow G(z_k)$ is one-to-one, the inverse transformation $s_k \rightarrow G^{-1}(s_k)$, can be easily determined, since we are dealing with a small set of discrete gray values.

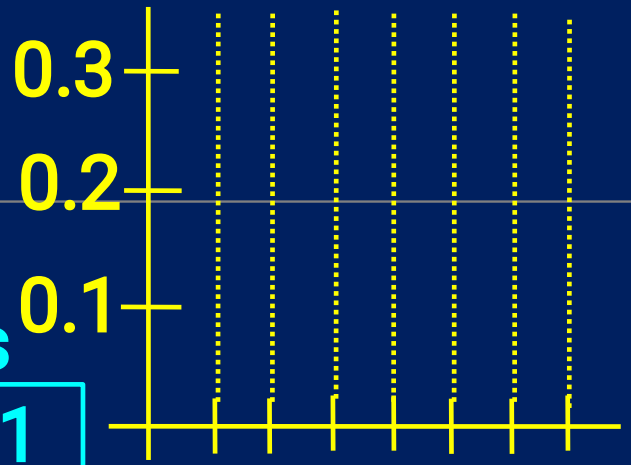
• In practice, this is not usually the case (i.e. $z_k \rightarrow G(z_k)$ is not one-to-one) and we assign gray values to match the given histogram, as closely as possible.

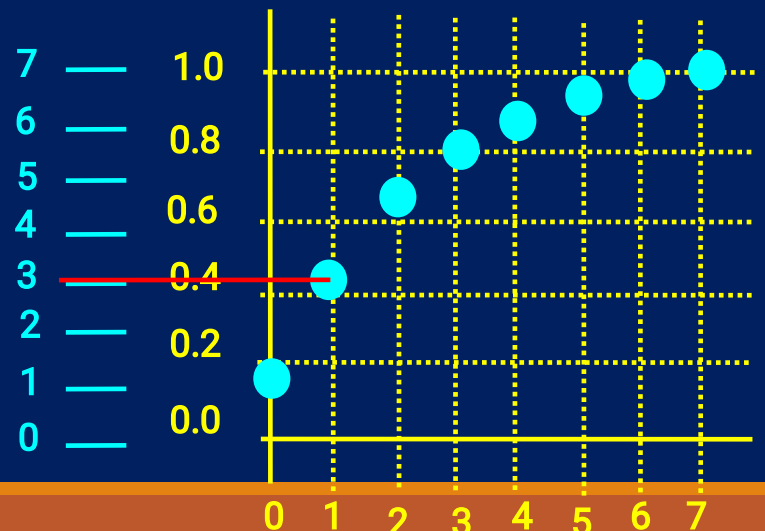

desired
0.3
0.2
0.1
0 1 2 3 4 5 6 7
0.3
0.2
0.1
0 1 2 3 4 5 6 7
0 – 0.19
1 – 0.25
2 – 0.21
3 – 0.16
4 – 0.08
5 – 0.06
6 – 0.03
7 – 0.02
1.0
0.8
0.6
0.4
0.2
0.0
0 1 2 3 4 5 6 7
0 – 0.0
1 – 0.0
2 – 0.0
3 – 0.15
4 – 0.20
5 – 0.30
6 – 0.20
7 – 0.15
1.0
0.8
0.6
0.4
0.2
0.0
0 1 2 3 4 5 6 7


equalized histogram
0.3
0.2
0.1
0 1 2 3 4 5 6 7
0.3
0.2
0.1
r s
0 1
7 6 5 4 3 2 1 0
1.0 0.8 0.6 0.4 0.2 0.0
0 1 2 3 4 5 6 7

equalized histogram

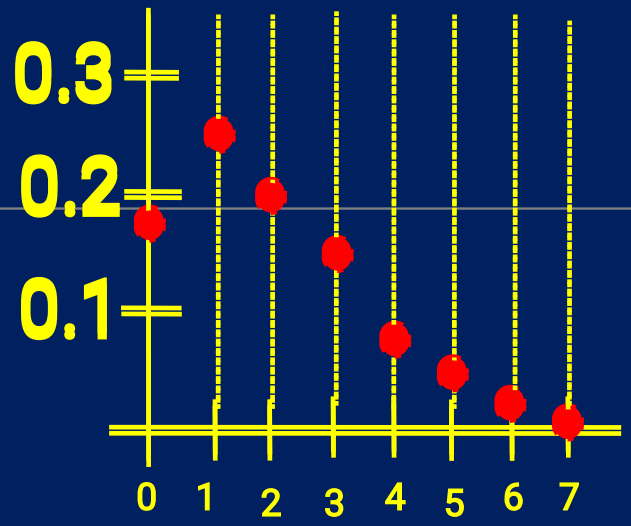


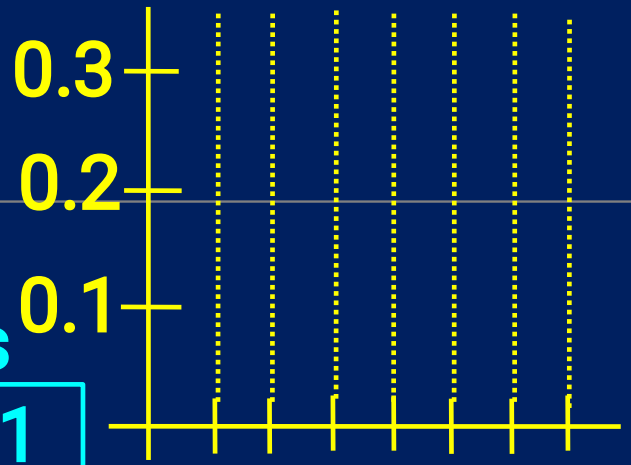


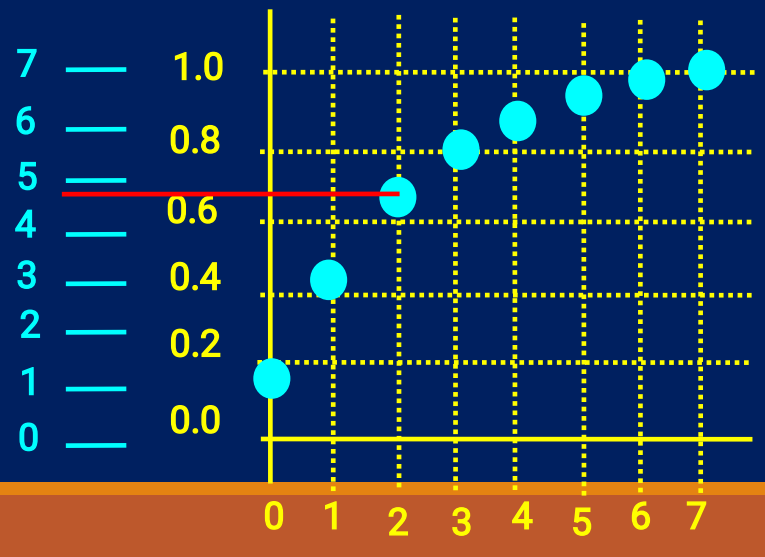


r	s
0	1
1	3

equalized histogram

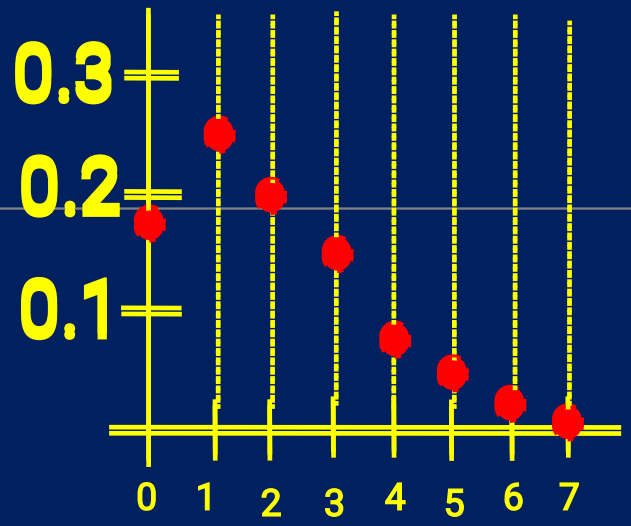


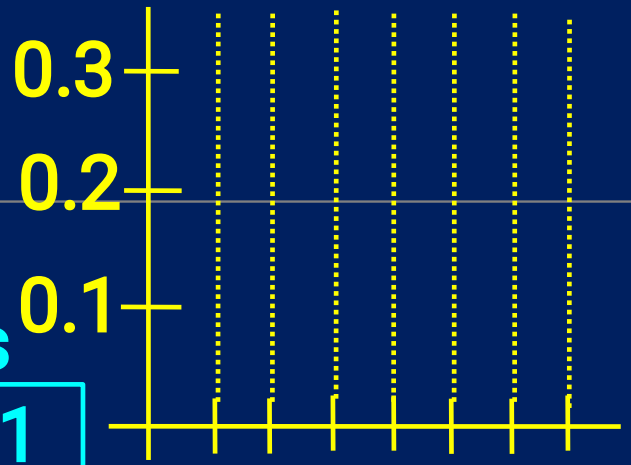


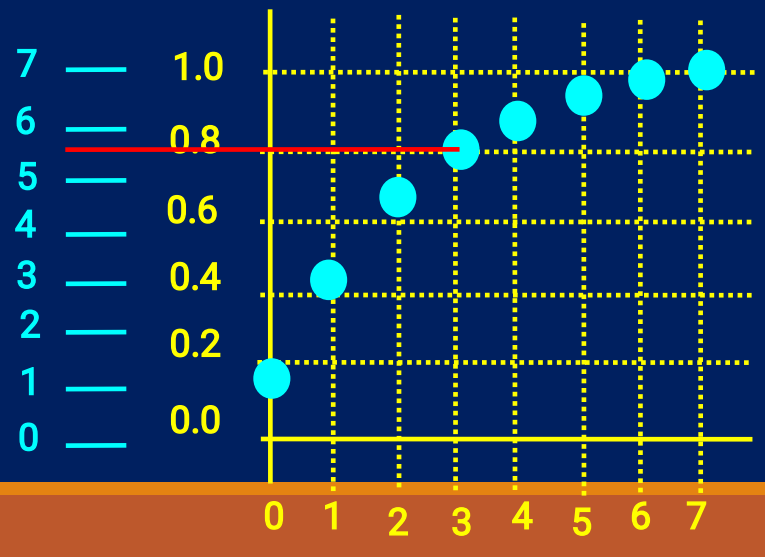


r	s
0	1
1	3
2	5

equalized histogram

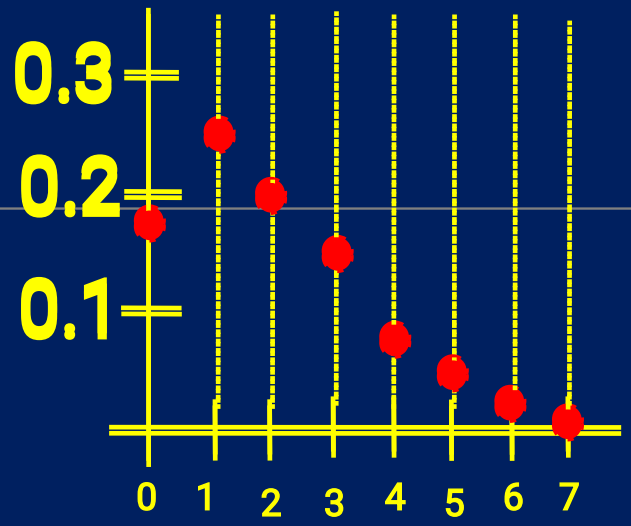


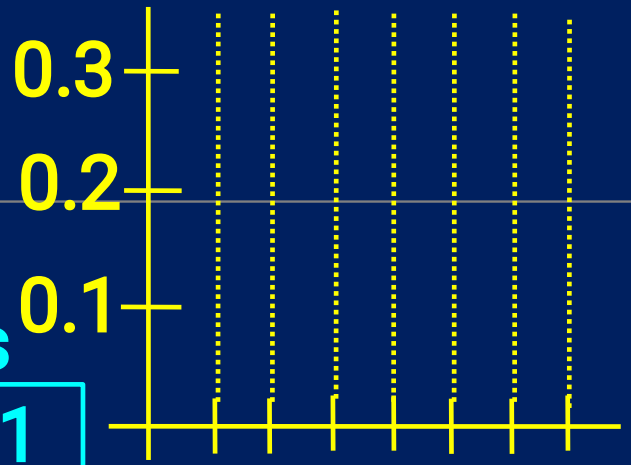


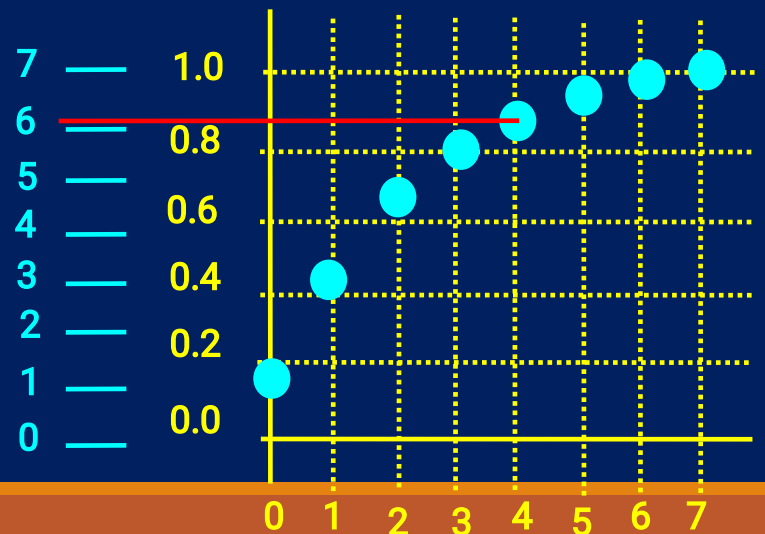


r	s
0	1
1	3
2	5
3	6

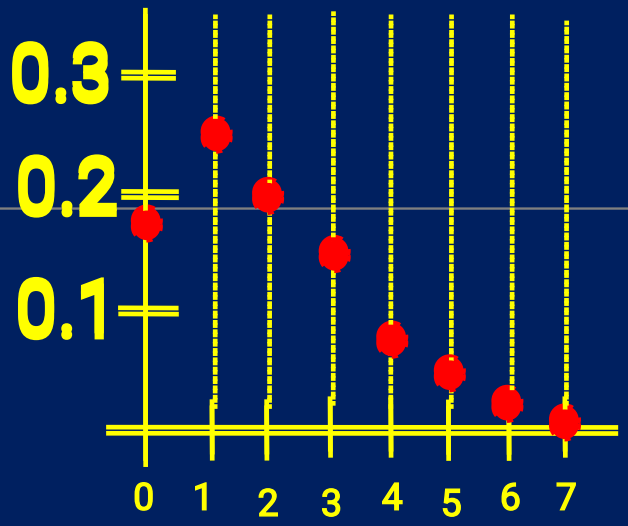
equalized histogram

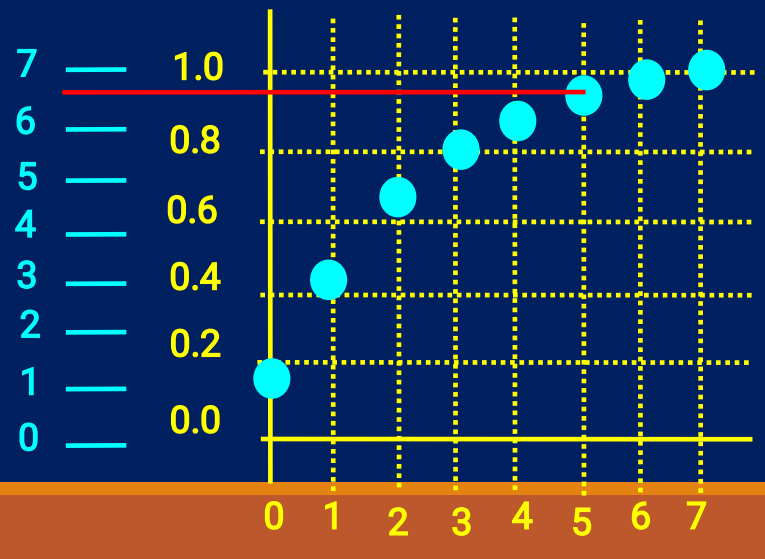






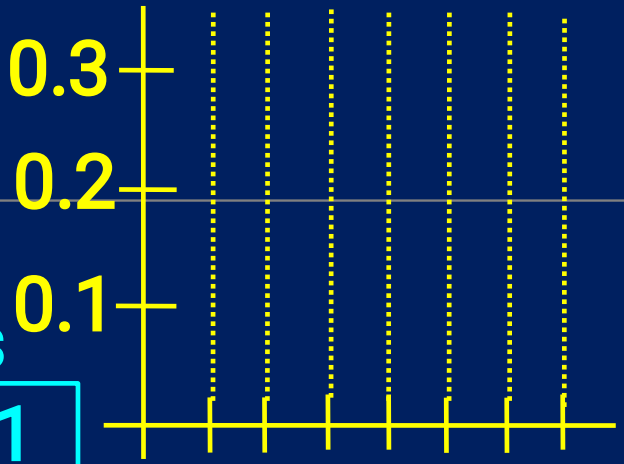
r	s
0	1
1	3
2	5
3	6
4	6

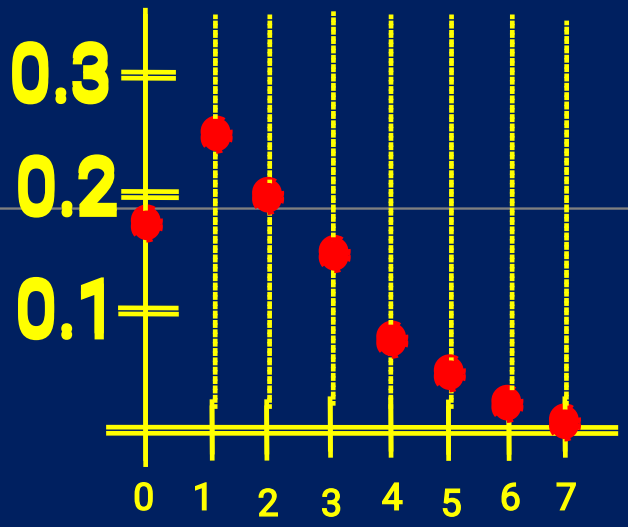


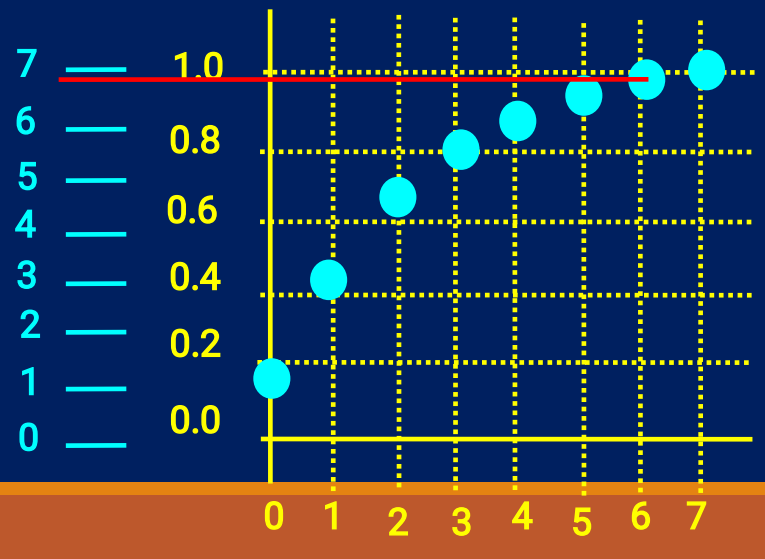


r	s
0	1
1	3
2	5
3	6
4	6
5	7

equalized histogram

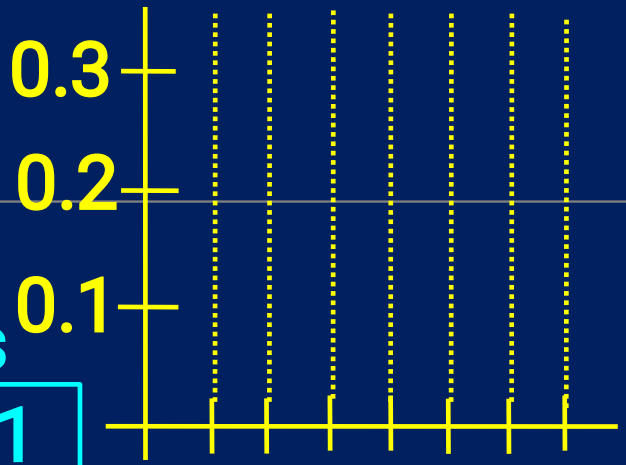


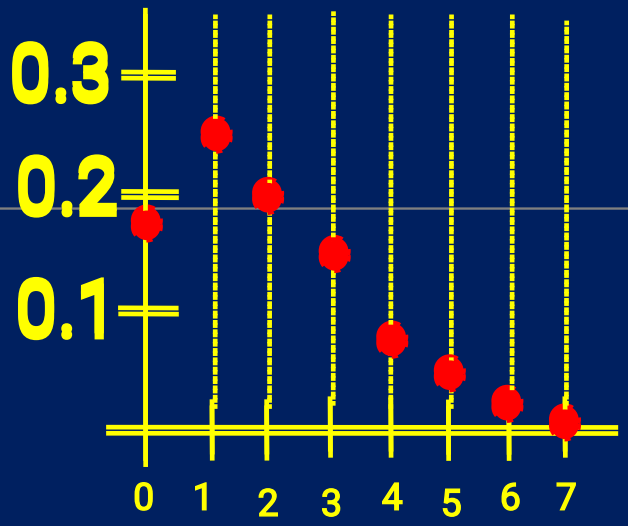


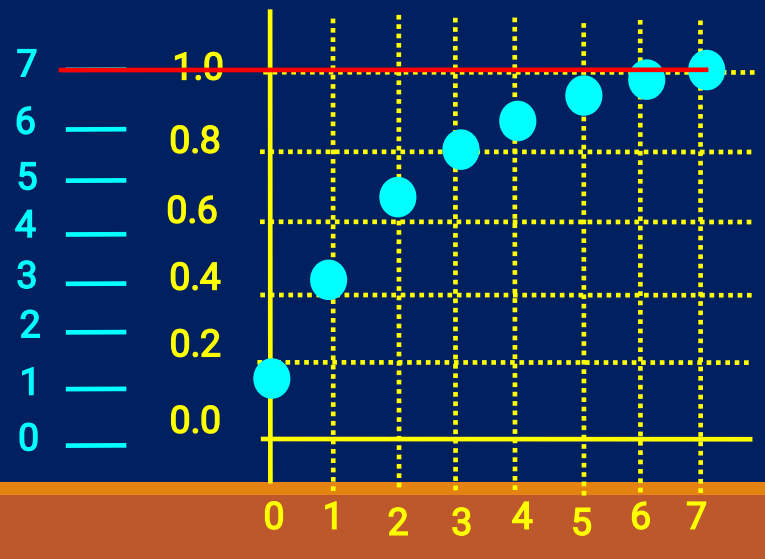


r	s
0	1
1	3
2	5
3	6
4	6
5	7
6	7

equalized histogram

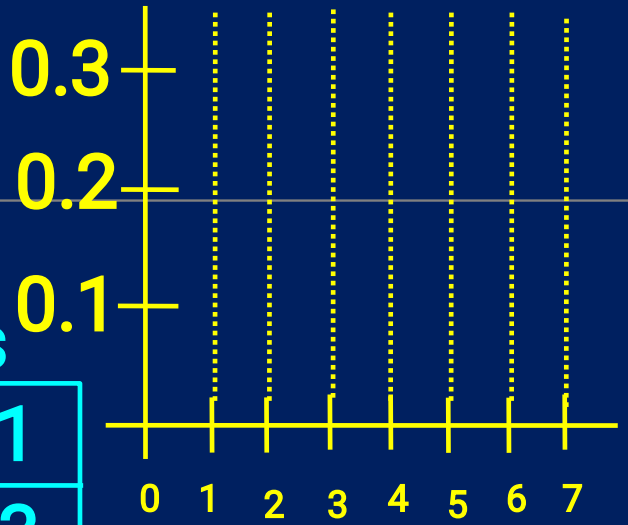




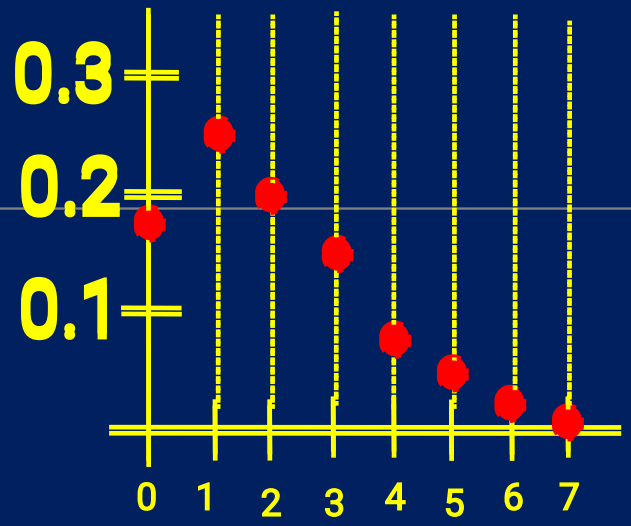


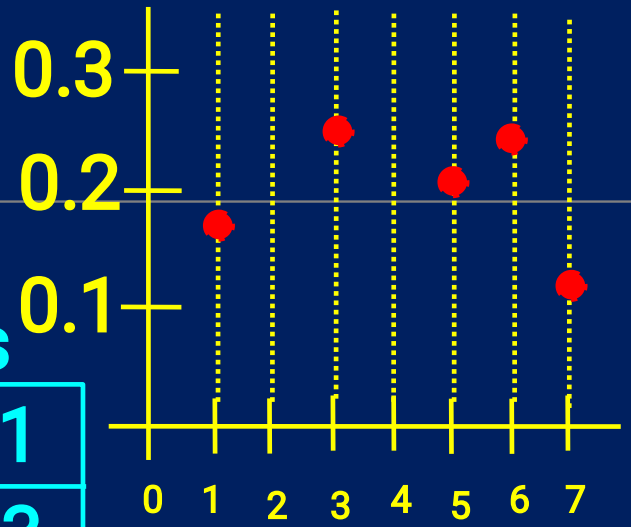
r	s
0	1
1	3
2	5
3	6
4	6
5	7
6	7
7	7

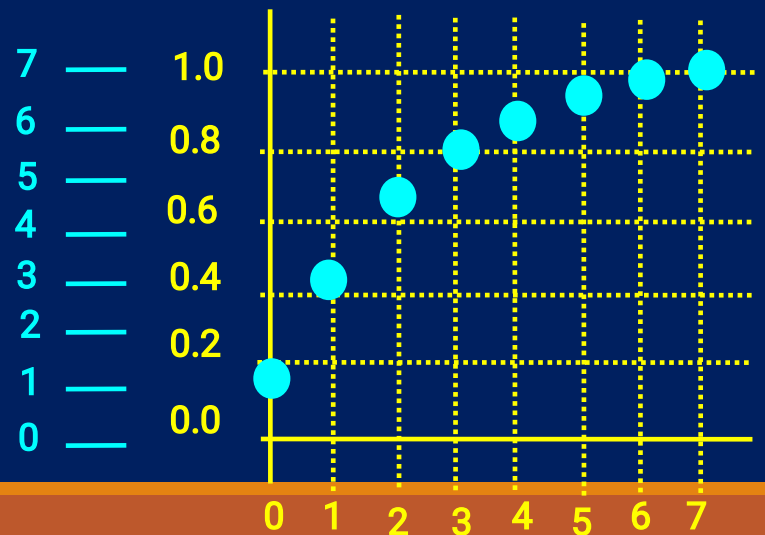
equalized histogram



equalized histogram

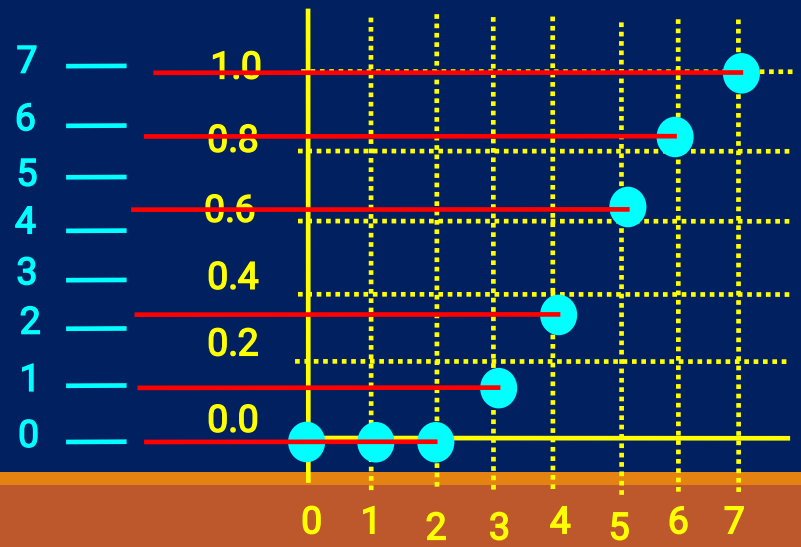




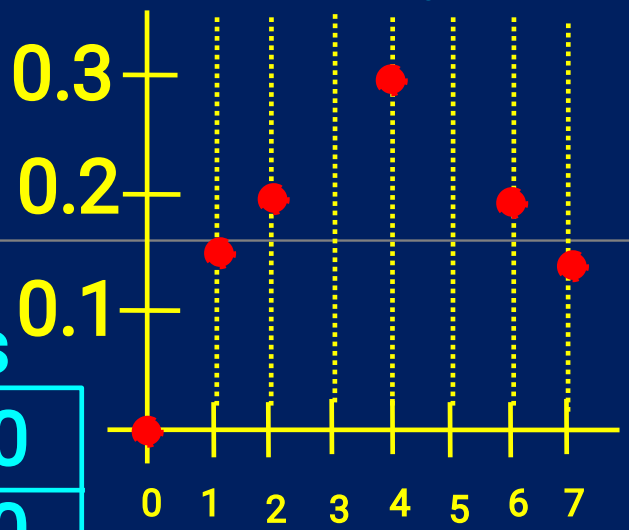


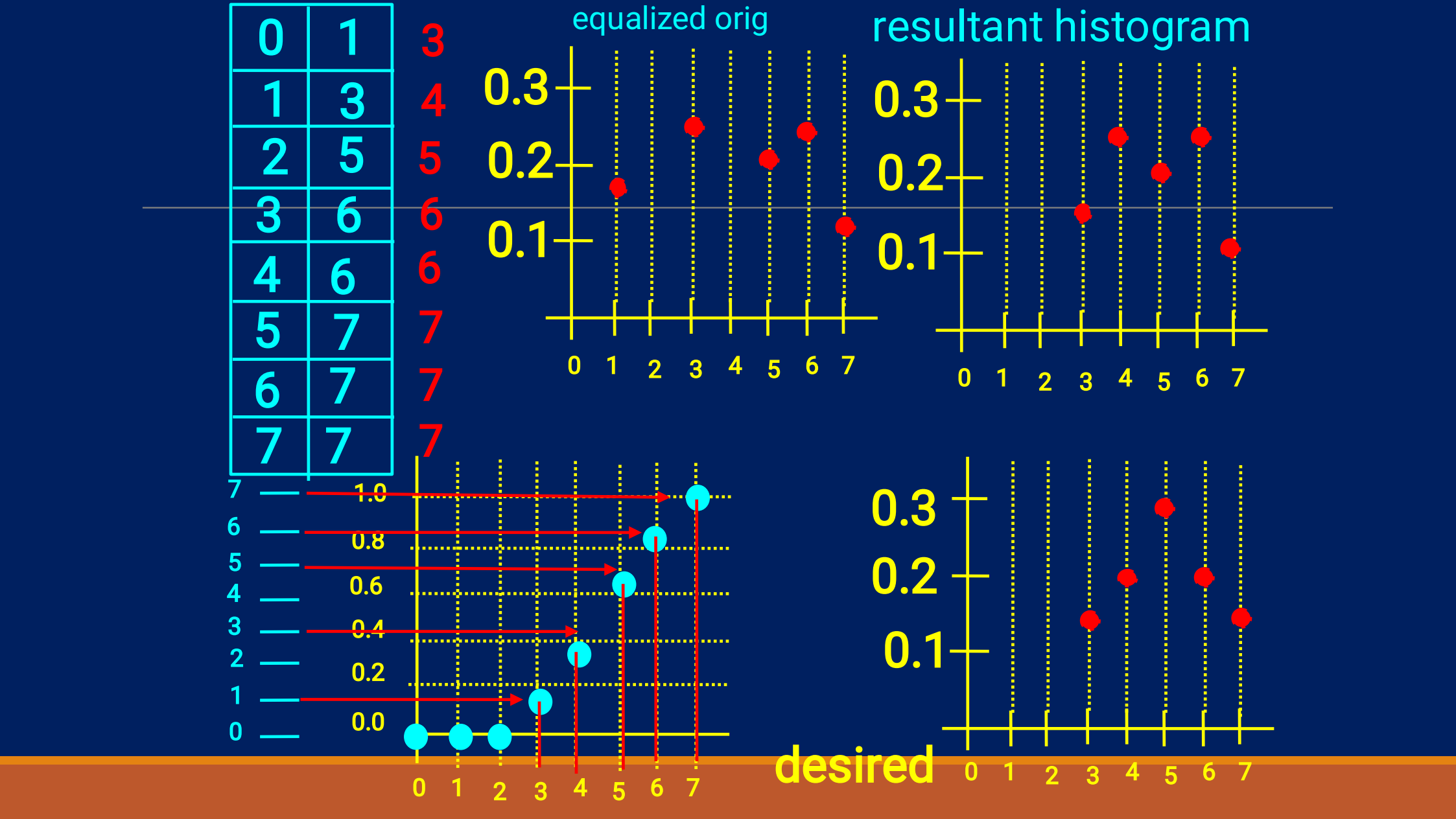
r	s
0	1
1	3
2	5
3	6
4	6
5	7
6	7
7	7

1 = 0.19
3 = 0.25
5 = 0.21
6 = .16+.08=.24
7 = .06+.03+.02
= 0.11



r	s
0	0
1	0
2	0
3	1
4	2
5	4
6	6
7	7




0 1
1 3
2 5
3 6
4 6
5 7
6 7
7 7
3
4
5
6
6
7
7
7
equalized orig
0.3
0.2
0.1
0 1 2 3 4 5 6 7
resultant histogram
0.3
0.2
0.1
0 1 2 3 4 5 6 7
7
6
5
4
3
2
1
0
1.0
0.8
0.6
0.4
0.2
0.0
0 1 2 3 4 5 6 7
0.3
0.2
0.1
desired
0 1 2 3 4 5 6 7

Example: Histogram Matching

Suppose that a 3-bit image (L=8) of size 64 × 64 pixels (MN = 4096) has the intensity distribution shown in the following table (on the left). Get the histogram transformation function and make the output image with the specified histogram, listed in the table on the right.

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

Specified	
z_q	$p_z(z_q)$
$z_0 = 0$	0.00
$z_1 = 1$	0.00
$z_2 = 2$	0.00
$z_3 = 3$	0.15
$z_4 = 4$	0.20
$z_5 = 5$	0.30
$z_6 = 6$	0.20
$z_7 = 7$	0.15

Example: Histogram Matching

Obtain the scaled histogram-equalized values,

$$s_0 = 1,\ s_1 = 3,\ s_2 = 5,\ s_3 = 6,\ s_4 = 7,$$

$$s_5 = 7,\ s_6 = 7,\ s_7 = 7.$$

Compute all the values of the transformation function G,

$$G(z_0) = 7\sum_{j=0}^{0} p_z(z_j) = 0.00 \rightarrow 0$$

$$G(z_1) = 0.00 \rightarrow 0 \qquad G(z_2) = 0.00 \rightarrow 0$$

$$G(z_3) = 1.05 \rightarrow 1 \quad s_0 \qquad G(z_4) = 2.45 \rightarrow 2 \quad s_1$$

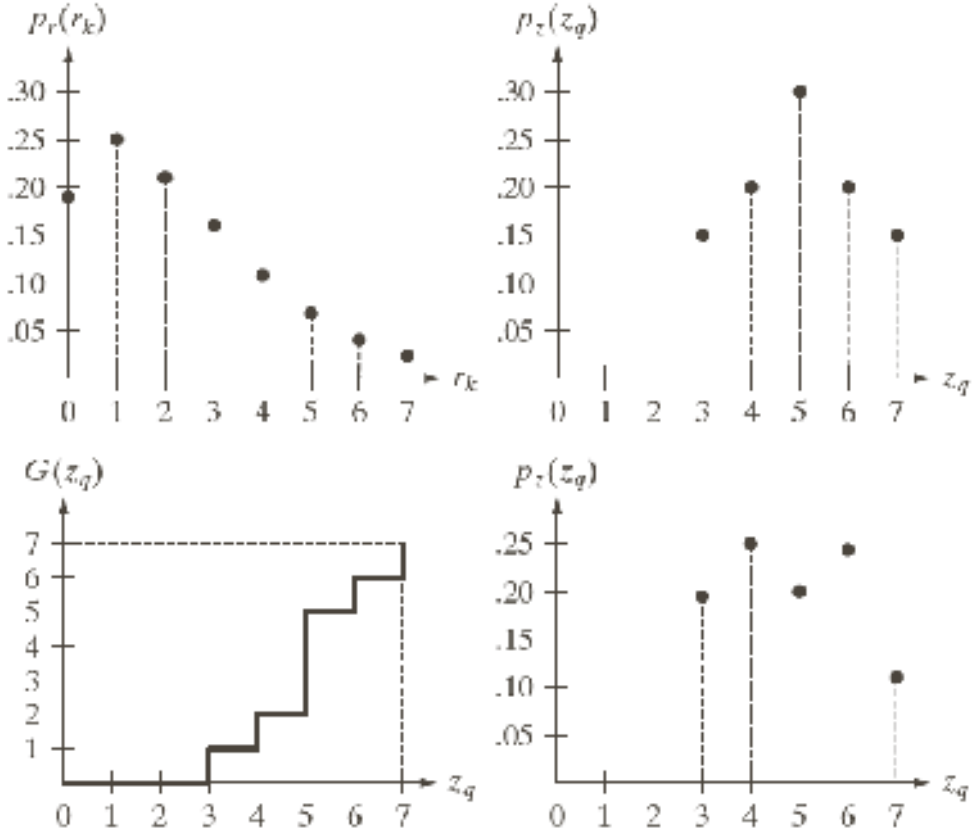
$$G(z_5) = 4.55 \rightarrow 5 \quad s_2 \qquad G(z_6) = 5.95 \rightarrow 6 \quad s_3$$

$$G(z_7) = 7.00 \rightarrow 7 \quad s_4 \quad s_5 \quad s_6 \quad s_7$$

	Specified
z_q	$p_z(z_q)$
$z_0 = 0$	0.00
$z_1 = 1$	0.00
$z_2 = 2$	0.00
$z_3 = 3$	0.15
$z_4 = 4$	0.20
$z_5 = 5$	0.30
$z_6 = 6$	0.20
$z_7 = 7$	0.15

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

Example: Histogram Matching



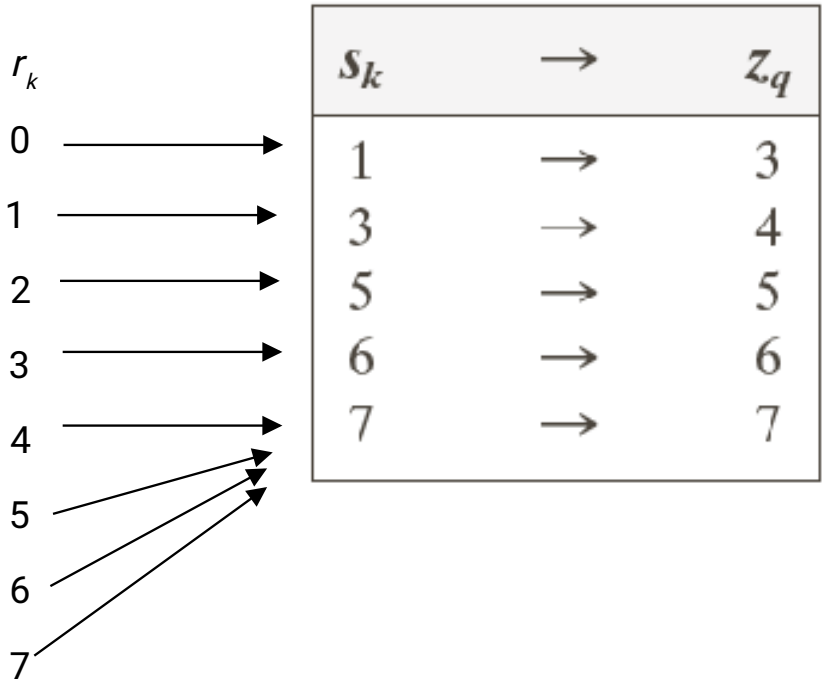
a b
c d

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).

Example: Histogram Matching

$$s_0 = 1,\ s_1 = 3,\ s_2 = 5,\ s_3 = 6,\ s_4 = 7,$$

$$s_5 = 7,\ s_6 = 7,\ s_7 = 7.$$



s_k	→	z_q
1	→	3
3	→	4
5	→	5
6	→	6
7	→	7

Example: Histogram Matching

$r_k \rightarrow z_q$

$0 \rightarrow 3$

$1 \rightarrow 4$

$2 \rightarrow 5$

$3 \rightarrow 6$

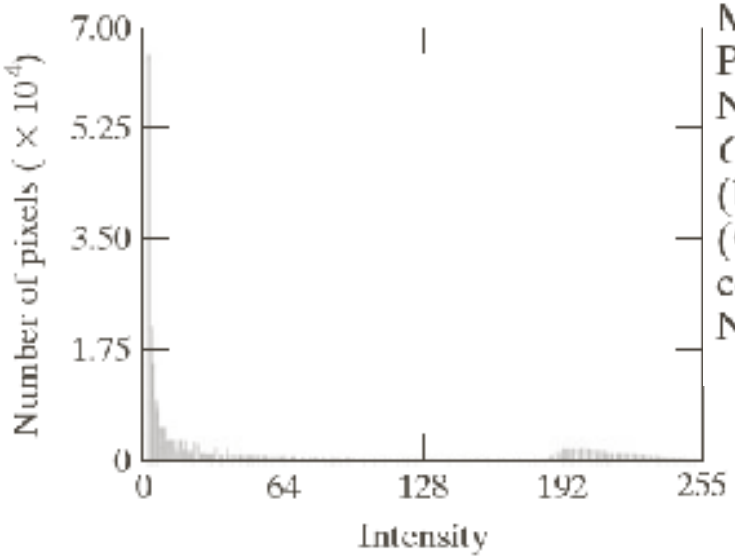
$4 \rightarrow 7$

$5 \rightarrow 7$

$6 \rightarrow 7$

$7 \rightarrow 7$

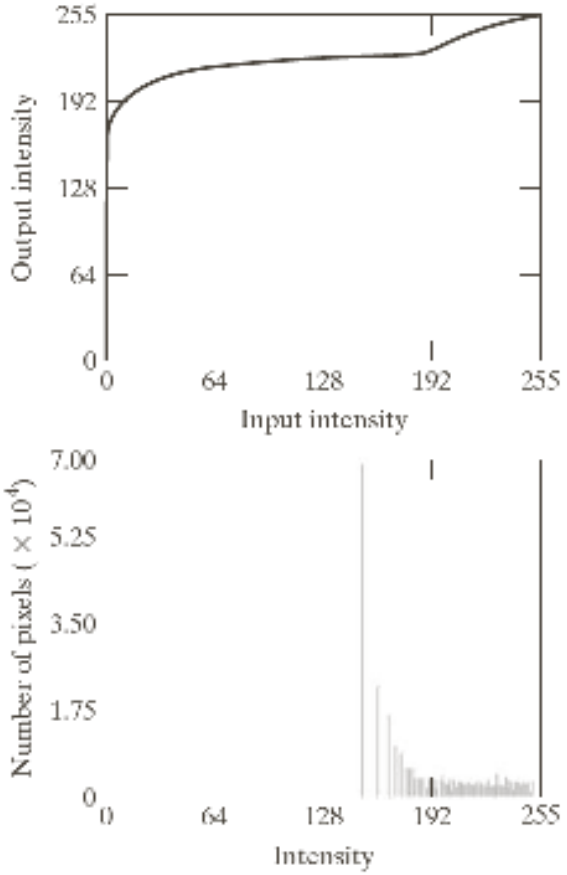
Example: Histogram Matching



a b

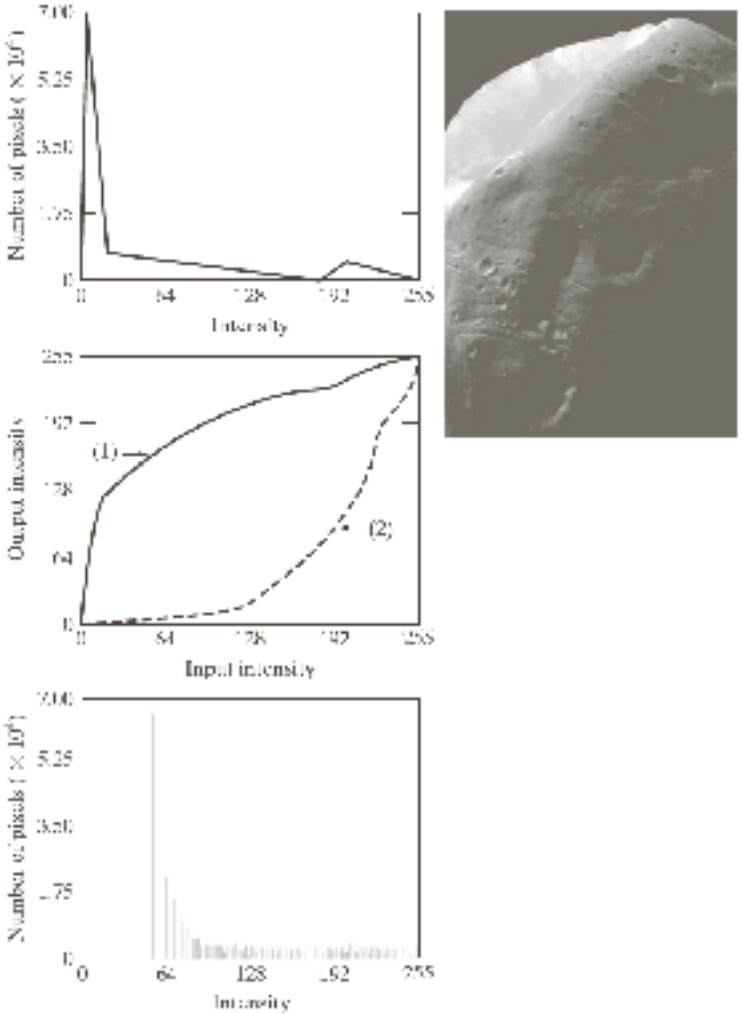
FIGURE 3.23
(a) Image of the Mars moon Phobos taken by NASA's *Mars Global Surveyor*. (b) Histogram. (Original image courtesy of NASA.)

Example: Histogram Matching



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).

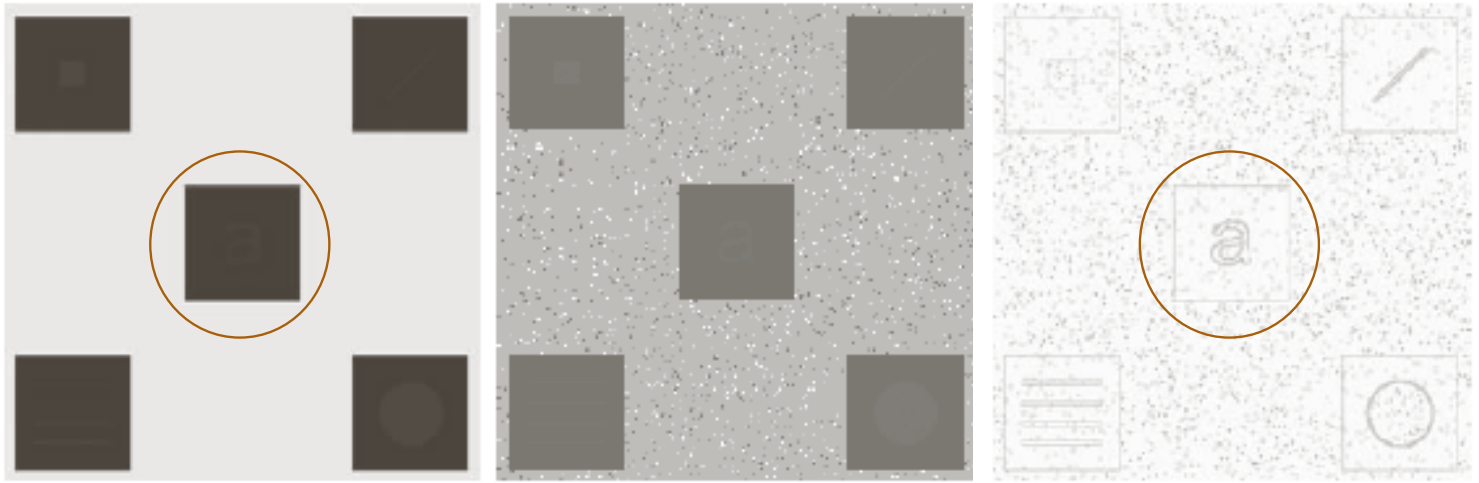
Local Histogram Processing

- Sometimes only a part of an image may require enhancement. In that case we can perform local histogram processing.

- Define a neighborhood and move its center from pixel to pixel.

- At each location, the histogram of the points in the neighborhood is computed. Either histogram equalization or histogram specification transformation function is obtained.

- Map the intensity of the pixel centered in the neighborhood. Move to the next location and repeat the procedure.

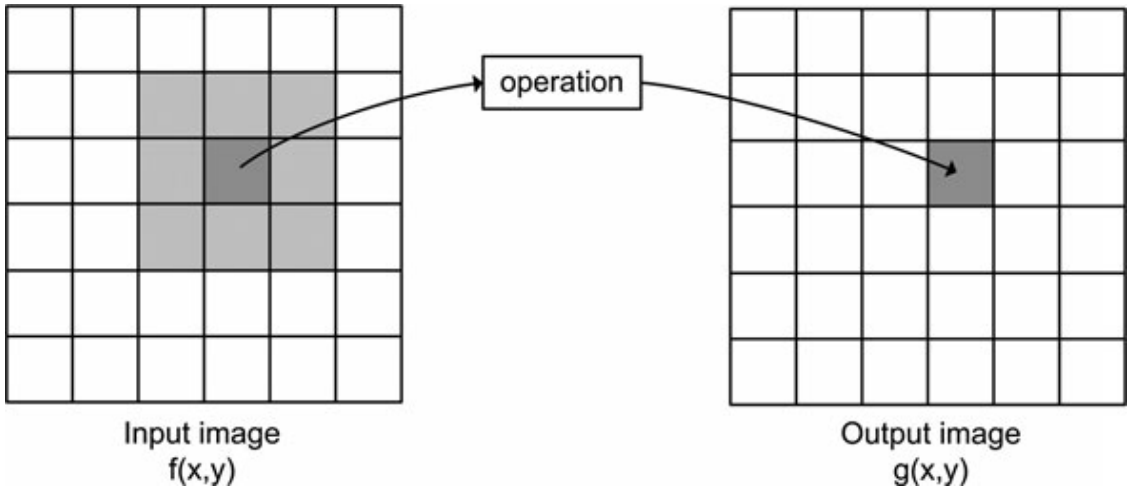


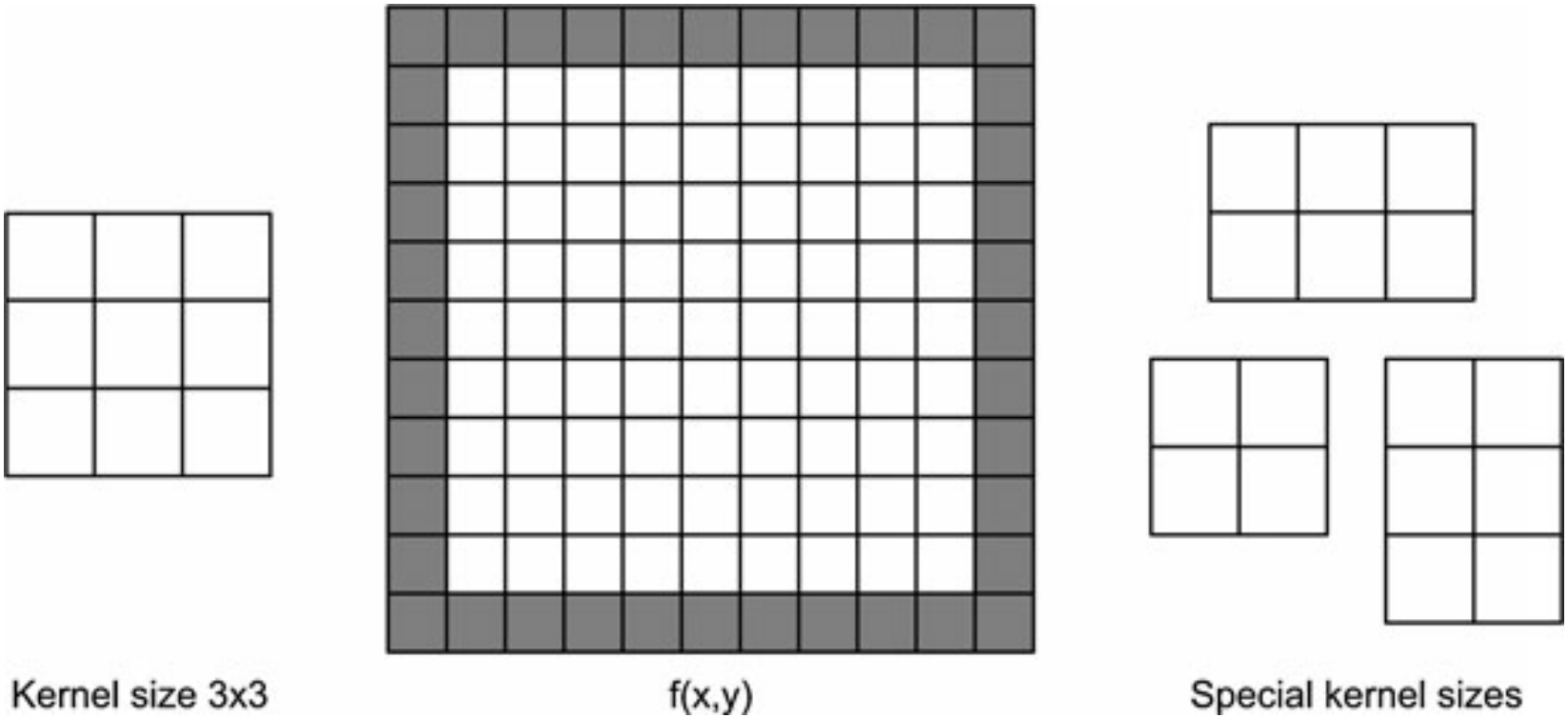
a b c

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.

Spatial Filtering

- A spatial filter consists of (a) **a neighborhood**, and (b) **a predefined operation**.
- Median Filter
- Mean Filter
- Correlation/Convolution
  - ✓ Template Matching
  - ✓ Edge Detection
    - Gradients
    - Image Edges
- Image Sharpening




Kernel size 3x3
f(x,y)
Special kernel sizes

Correlation

Difference between Correlation and Convolution

$$g(x, y) = \sum_{j=-R}^{R} \sum_{i=-R}^{R} h(i, j) \cdot f(x + i, y + j)$$

$$g'(x, y) = \frac{g(x, y)}{\sum_{k} \sum_{j} f(x + j, y + k)}$$

Convolution

Difference between Correlation and Convolution

$$g(x,y) = \sum_{j=-R}^{R} \sum_{j=-R}^{R} h(i,j).f(x-i,y-j)$$

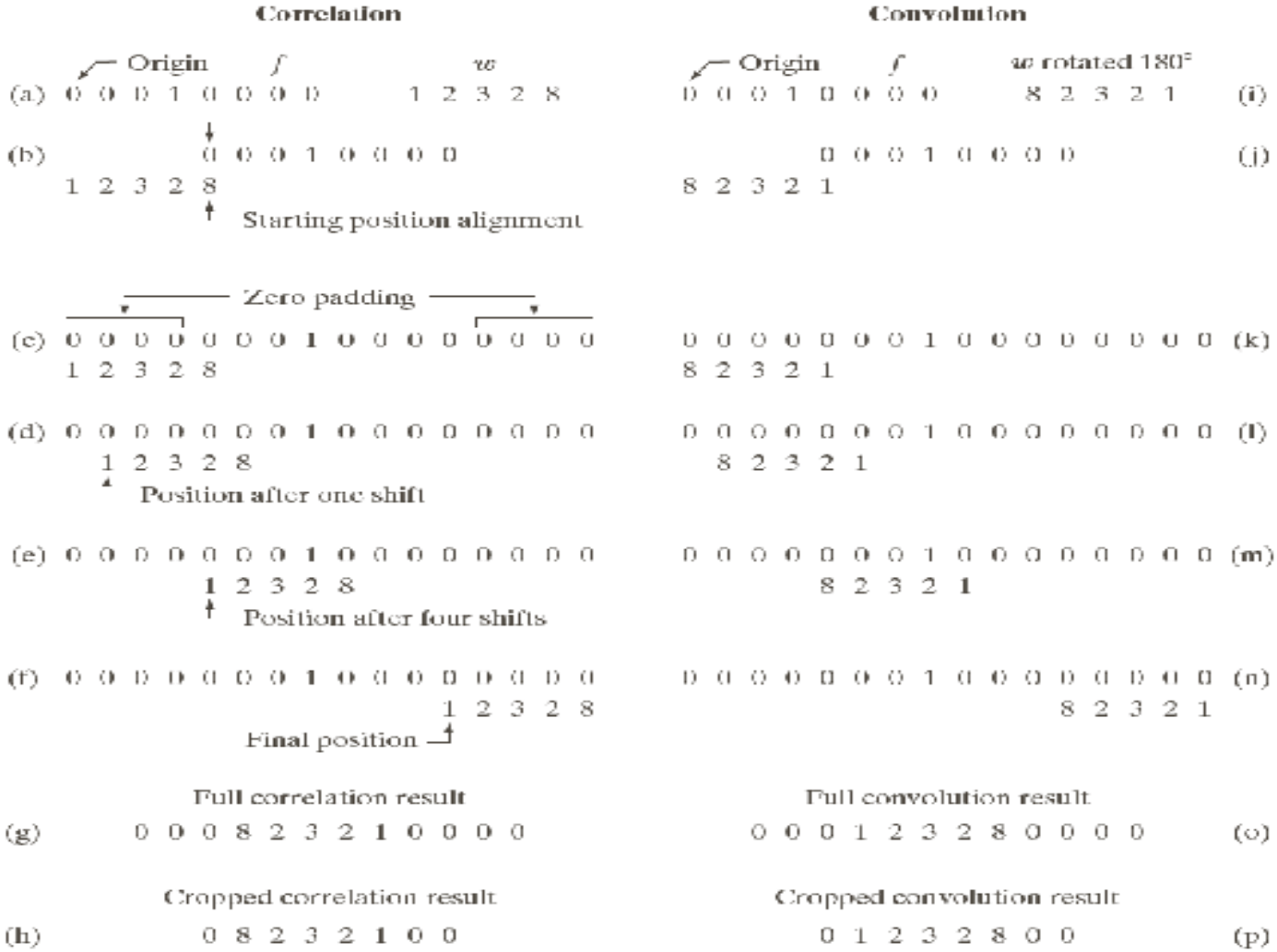
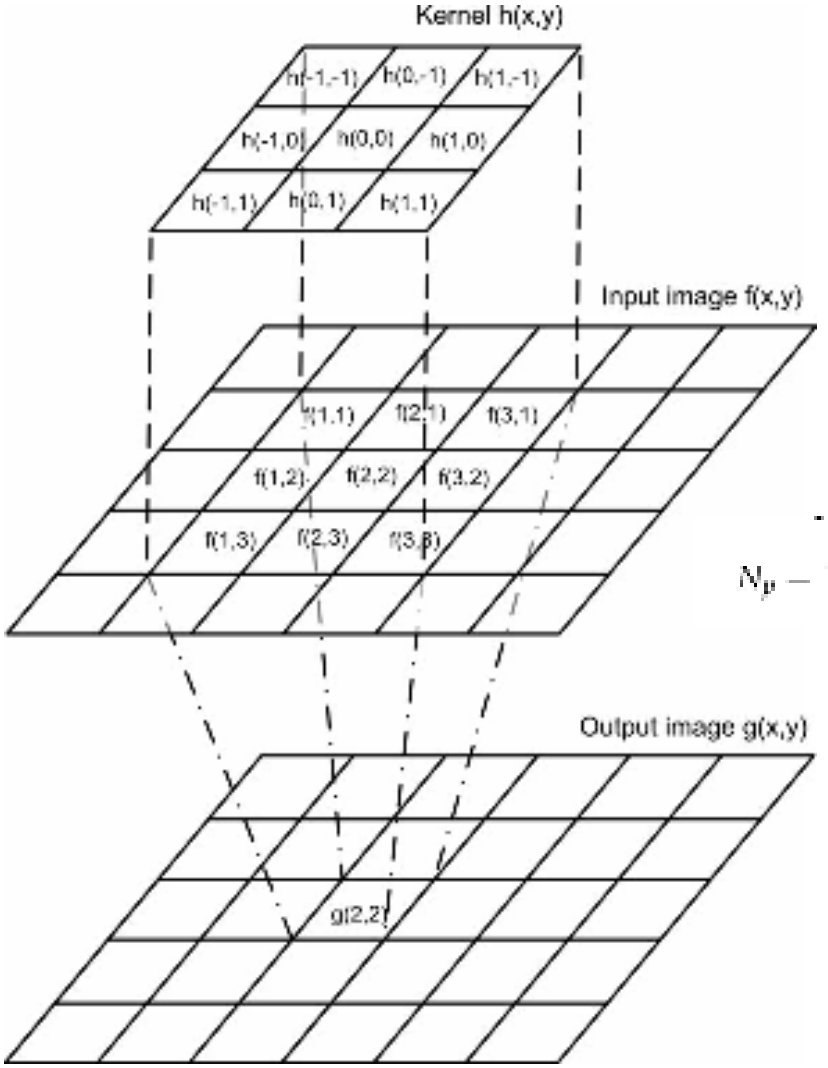
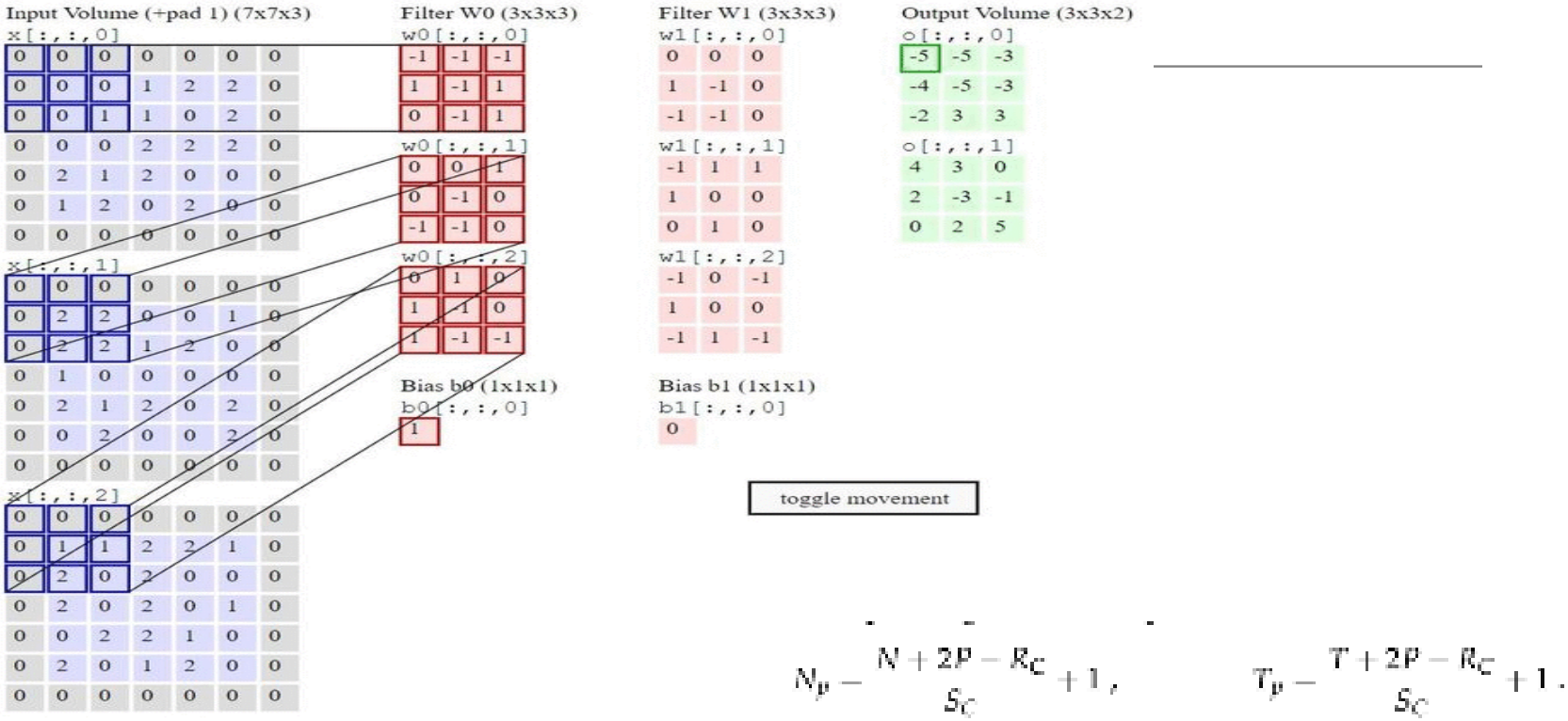


FIGURE 3.29 Illustration of 1-D correlation and convolution of a filter with a discrete unit impulse. Note that correlation and convolution are functions of *displacement*.



$$N_p = \frac{N + 2P - R_C}{S_C} + 1, \qquad T_p = \frac{T + 2P - R_C}{S_C} + 1.$$

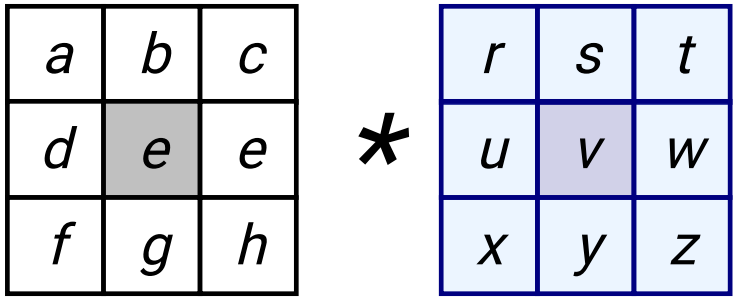


$$N_p = \frac{N + 2P - R_C}{S_C} + 1, \qquad T_p = \frac{T + 2P - R_C}{S_C} + 1.$$

Fig. Convolution Layer

Difference between Convolution and Correlation

Convolution is a similar operation, with just one subtle difference:



Original Image Pixels | **Filter**

Correlation

$$e_{processed} = v*e + z*h + y*g + x*f + w*e + u*d + t*c + s*b + r*a$$

Convolution

$$e_{processed} = v*e + z*a + y*b + x*c + w*d + u*e + t*f + s*g + r*h$$

For symmetric filters it makes no difference

Convolution and Correlation

- filtering and edge detection
- the output gray-level at a pixel is a weighted sum of the gray-levels of pixels in the neighborhood
- kernel elements are taken left-to-right, top-to-bottom
- image elements (in the neighborhood) are taken right-to-left, bottom-to-top

- feature recognition
- the kernel is now called a template
- algorithm is the same as convolution except for the pairing of weights with pixels:
- kernel elements are taken left-to-right, top-to-bottom
- image elements (in the neighborhood) are taken left-to-right, top-to-bottom
- you can get correlation using the convolution algorithm by rotating the kernel 180°
- the disadvantage of being sensitive to changes in the amplitude
- need to normalize the output

Applications

h_h{1,1}=[-1,1,0];

h_h{1,2}=[0,0,0;0,1,0;-1,0,0];

h_h{1,3}=[0;1;-1];

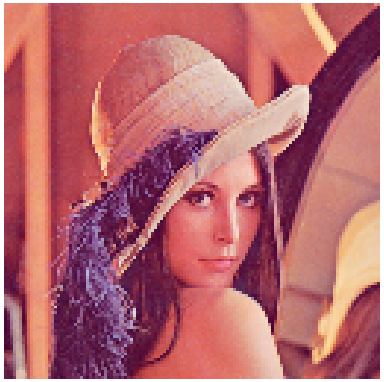
h_h{1,4}=[0,0,0;0,1,0;0,0,-1];

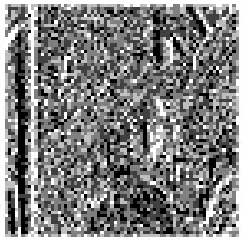
h_h{1,5}=[0,1,-1];

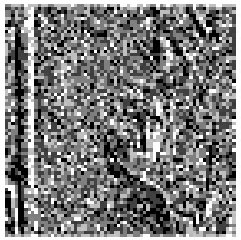
h_h{1,6}=[0,0,-1,;0,1,0;0,0,0];

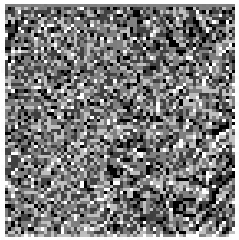
h_h{1,7}=[-1;1;0];

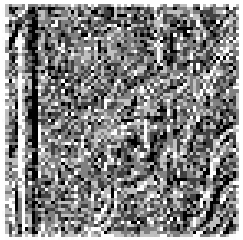
h_h{1,8}=[-1,0,0;0,1,0;0,0,0];

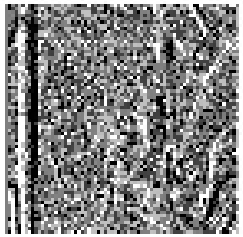


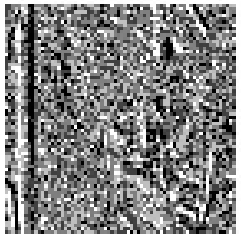


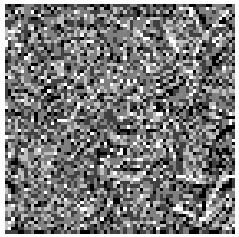


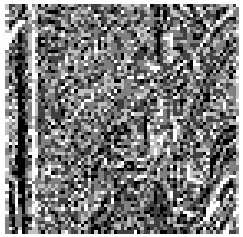












Mean Filter

- Also called Smoothing spatial filters and used for blurring and for noise reduction.
- Blurring is used in removal of small details and bridging of small gaps in lines or curves

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

1	1	1
1	1	1
1	1	1

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

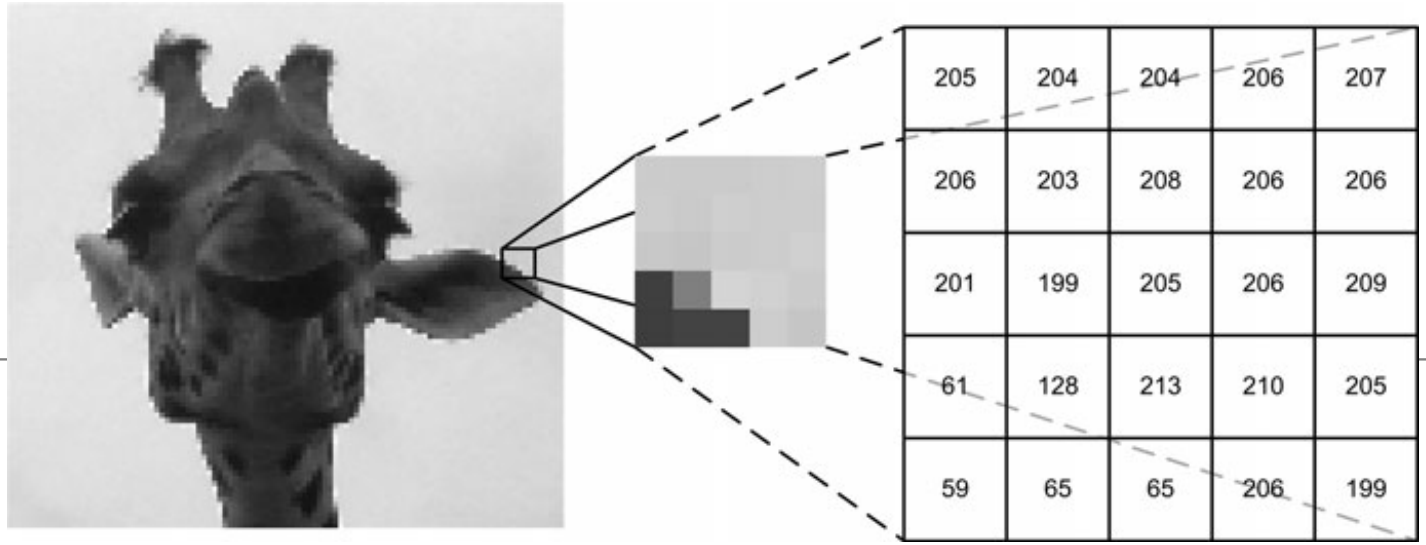
1	2	1
2	4	2
1	2	1

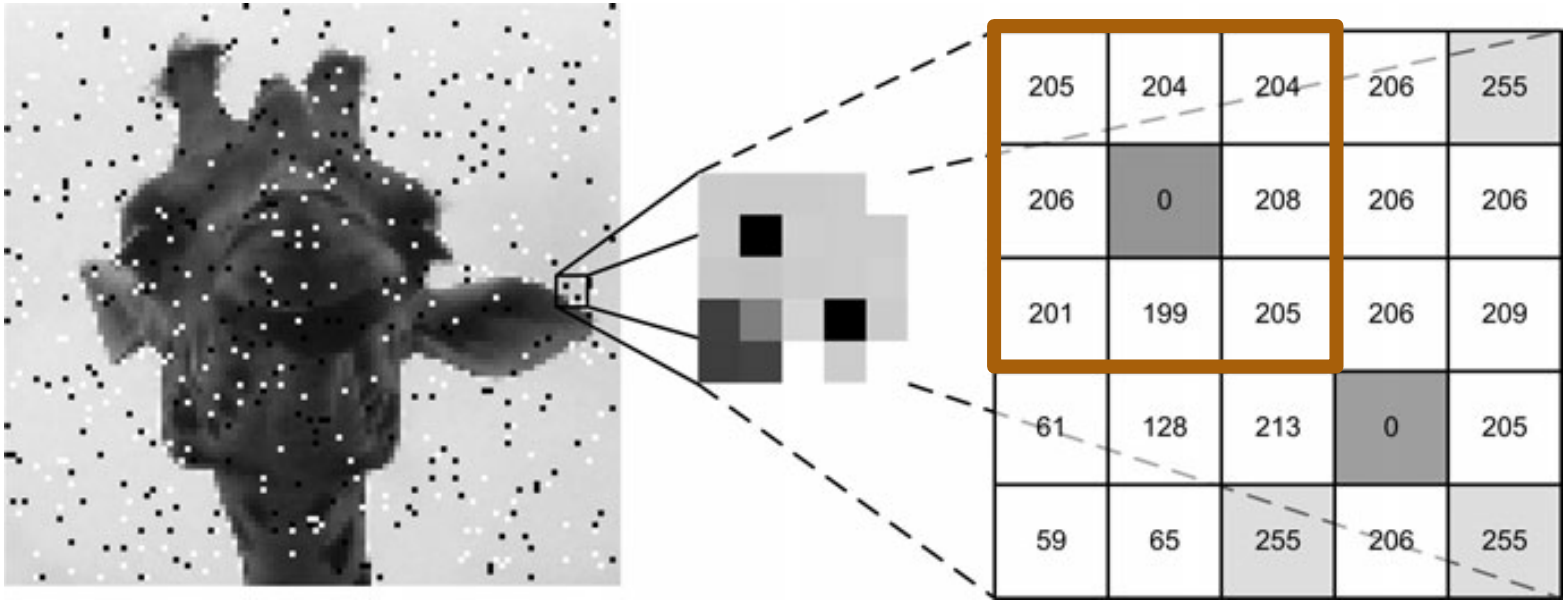
a b

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.

The Median Filter

Application: Image Denoising





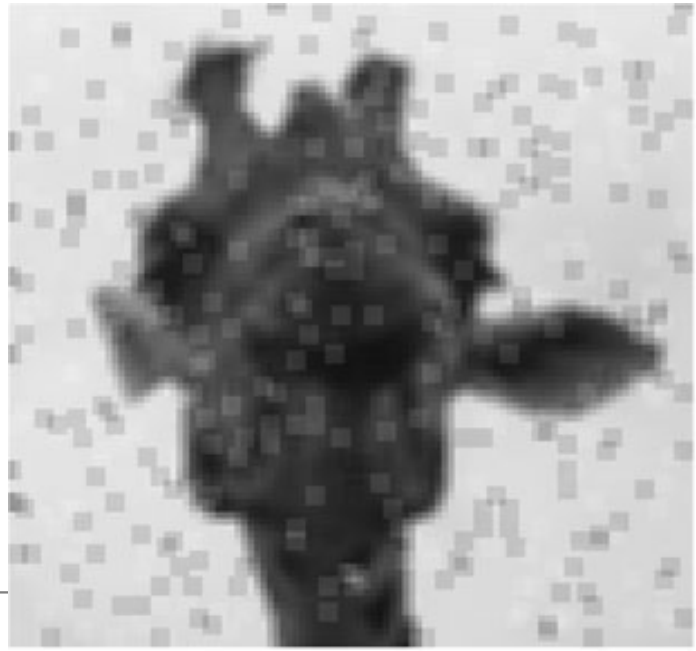
Ordering : [0, 199, 201, 204, *204*, 205, 205, 206, 208]

Median = 204

$$\text{Mean value} = \frac{205 + 204 + 204 + 206 + 0 + 208 + 201 + 199 + 205}{9}$$

$$= 181.3 \simeq 181$$

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



Mean filtered

Median filtered

Mean Filter

Original Image

3 x 3
Smoothing
filter

Smoothened Image

Mean Filter

Original Image

5 x 5
Smoothing
filter

Smoothened Image

Average Vs Weighted Average

Original Image

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Fig. An example of how a mean filter can be used to hide the identity of a person.

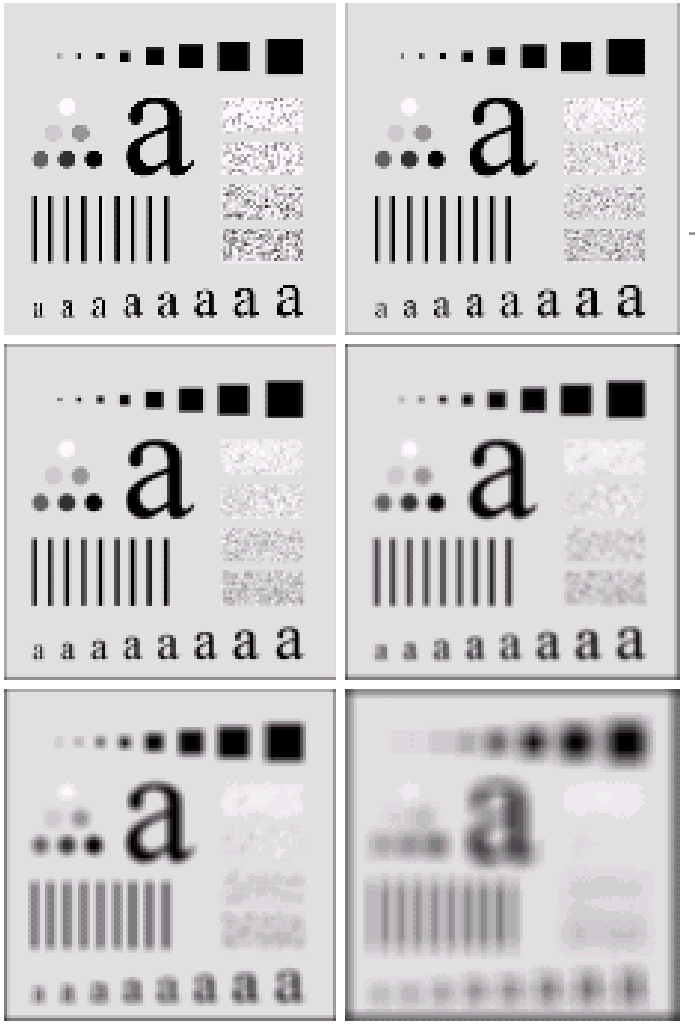
Image Smoothing Exam

The image at the top left
is an original image of
size 500*500 pixels

The subsequent images show the image after filtering with an averaging filter of increasing sizes

- 3, 5, 9, 15 and 35

Notice how detail begins to disappear



Edge Detection

Image Edges: Gradients

$$\text{Magnitude} = \sqrt{g_x^2 + g_y^2}$$

$$\text{Approximated magnitude} = |g_x| + |g_y|$$

$$g_x(x, y) \approx f(x+1, y) - f(x-1, y)$$

$$g_y(x, y) \approx f(x, y+1) - f(x, y-1)$$

Prewitt

Vertical

-1	0	1
-1	0	1
-1	0	1

Horizontal

-1	-1	-1
0	0	0
1	1	1

Sobel

Vertical

-1	0	1
-2	0	2
-1	0	1

Horizontal

-1	-2	-1
0	0	0
1	2	1

Fig. Prewitt and Sobel kernels

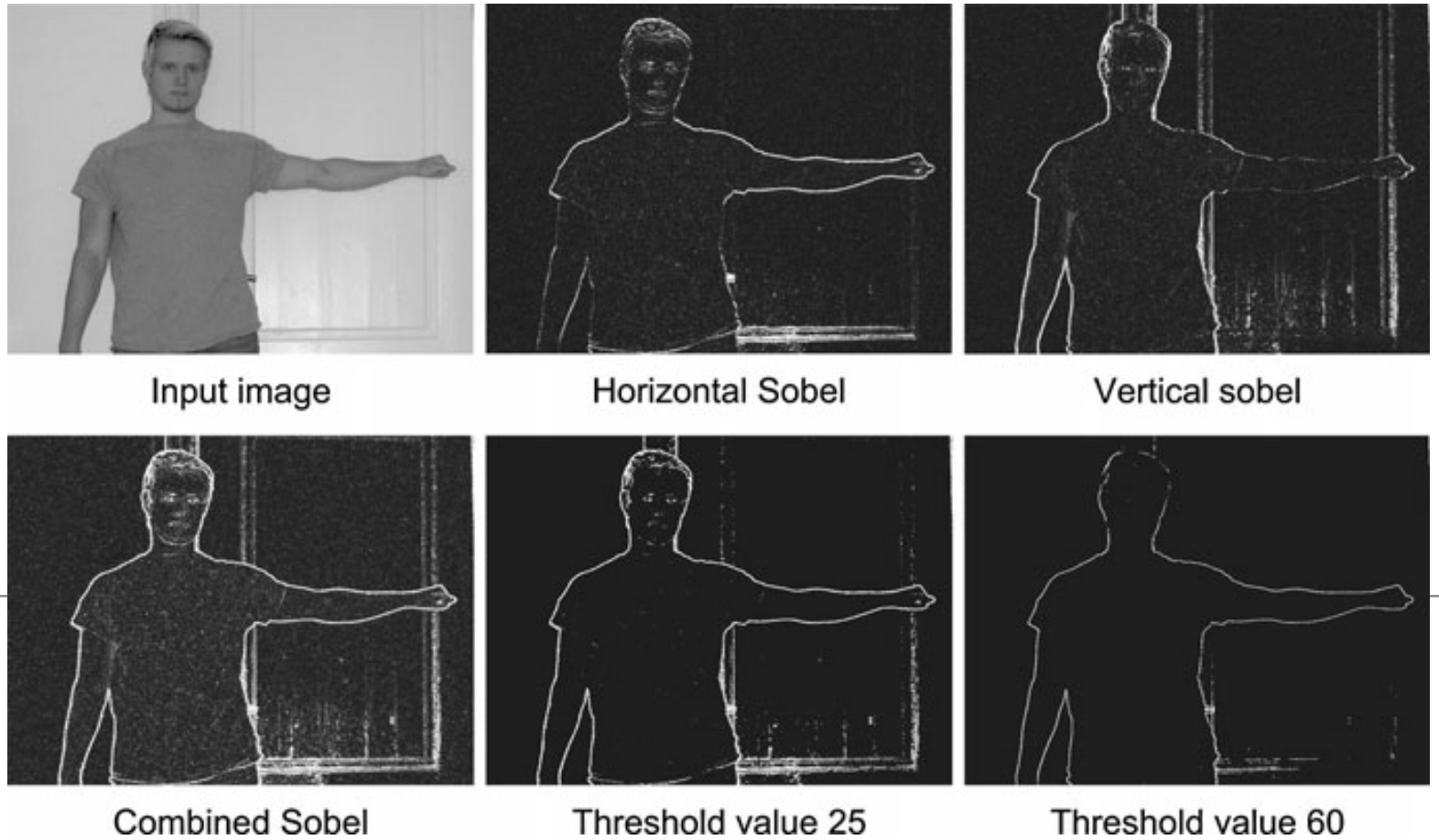


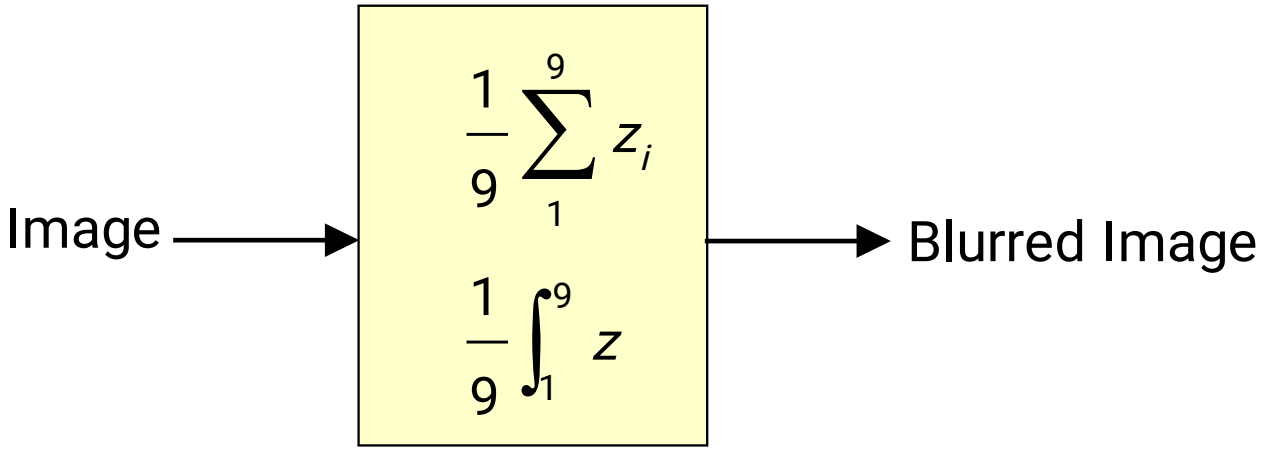
Fig. Sobel kernels applied to an image.

Image Sharpening

- **Highlight transition in intensity**
- **The principal objective of sharpening is to highlight fine detail in an image or to enhance detail that has been blurred.**

  - Electronic printing
  - Medical Imaging
  - Industrial Inspection
  - Autonomous guidance in military systems

Averaging is analogous to integration



Sharpening can be accomplished by spatial differentiation

- The Response of the derivative operator is proportional to the degree of intensity discontinuity of the image.
- Image differentiation enhances EDGES & other discontinuities (like NOISE)
- Deemphasizes area with slowly varying intensities

Requirements for digital derivative

First derivative

1) Must be zero in flat segment / constant intensity
2) Must be nonzero along ramps.
3) Must be nonzero at the onset of a gray-level step or ramp

Second derivative

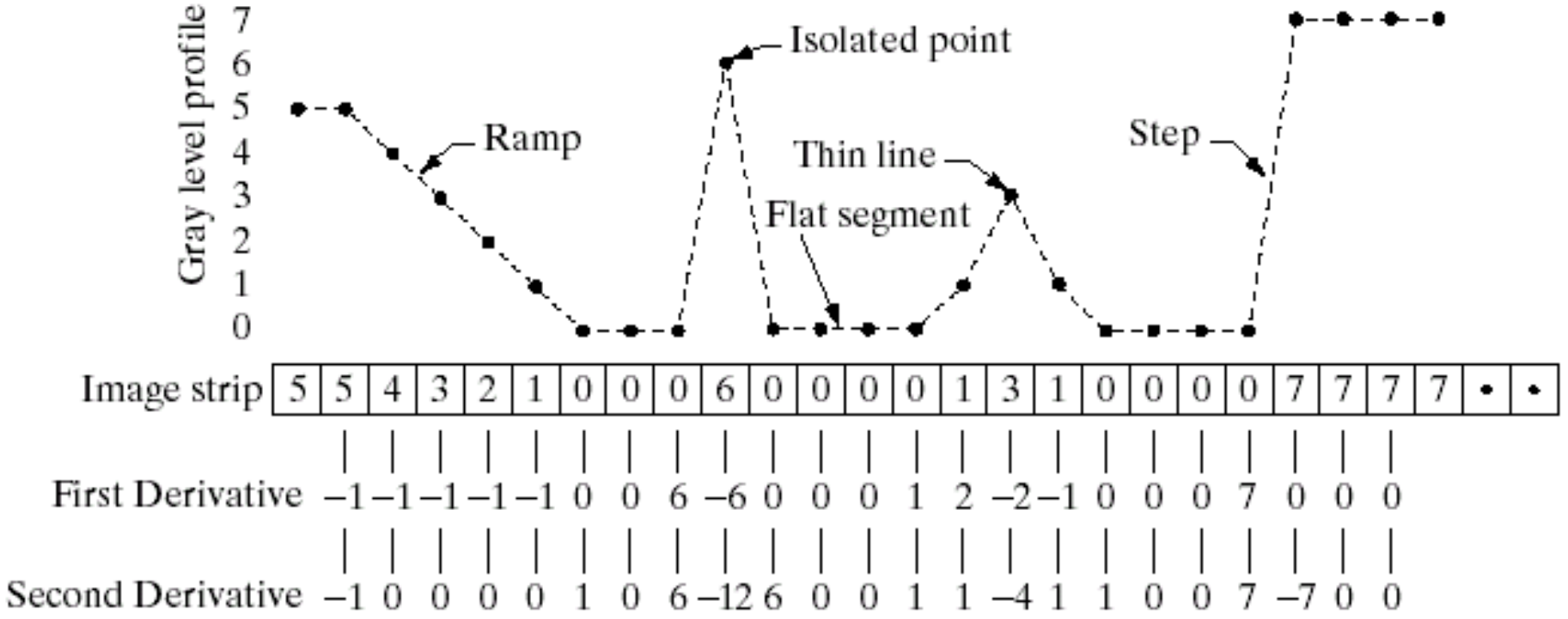
1) Must be zero in flat segment / constant intensity
2) Must be zero along ramps.
3) Must be nonzero at the onset and end of a gray-level step or ramp

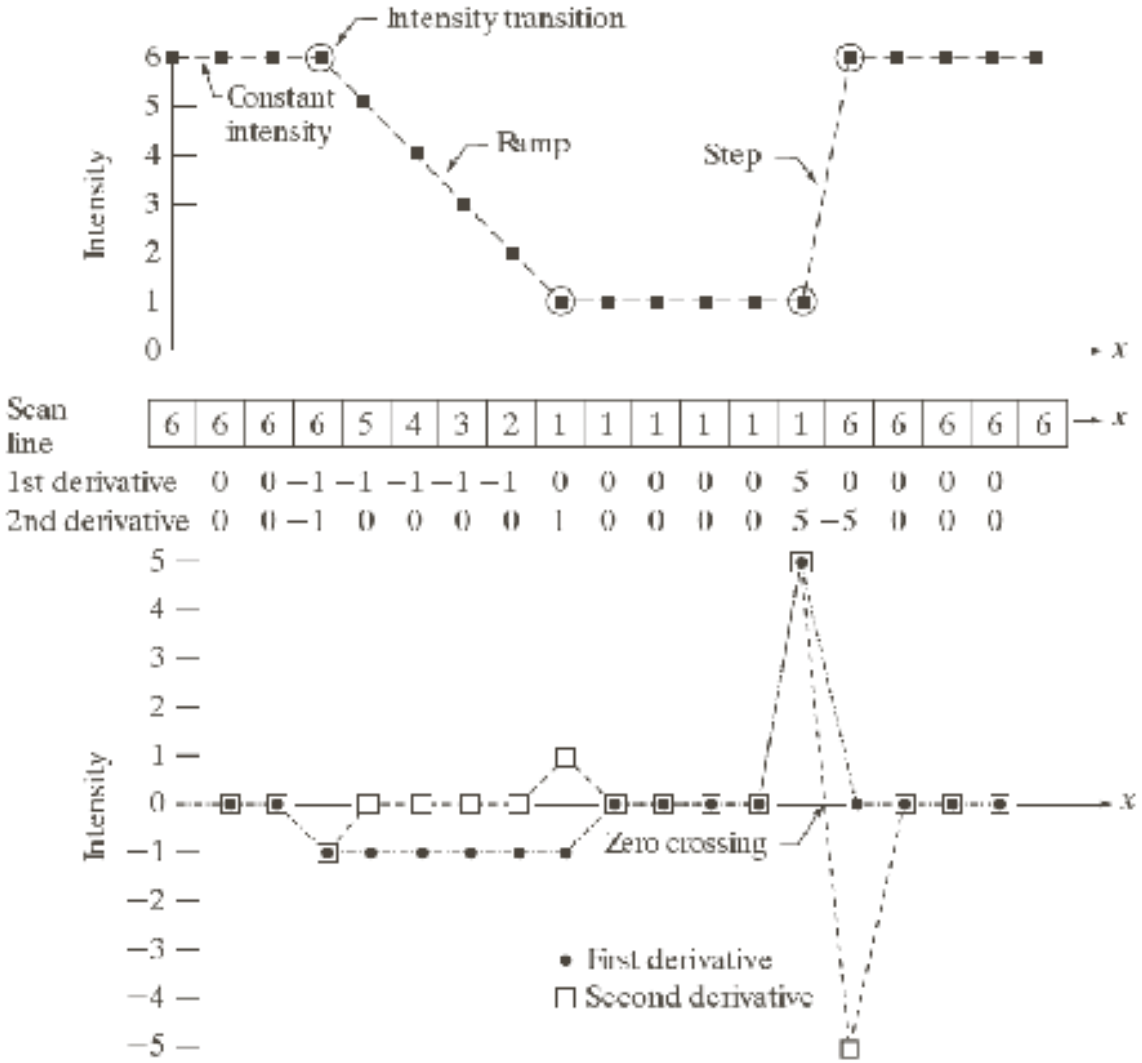
- The first-order derivative

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

- The second-order derivative

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


Gray level profile
7
6
5
4
3
2
1
0
Ramp
Isolated point
Thin line
Flat segment
Step
Image strip
5 5 4 3 2 1 0 0 0 6 0 0 0 0 1 3 1 0 0 0 0 7 7 7 7
First Derivative
−1 −1 −1 −1 −1 0 0 6 −6 0 0 0 1 2 −2 −1 0 0 0 7 0 0 0
Second Derivative
−1 0 0 0 0 1 0 6 −12 6 0 0 1 1 −4 1 1 0 0 7 −7 0 0



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.

- First derivative ------→ Thick Edge
  - ❖ Derivative is non Zero along the Ramp

- 2^{nd} Derivative ---→ Double edge of one pixel thick, separated by zeros
  - ❖ 2^{nd} derivative enhances fine details

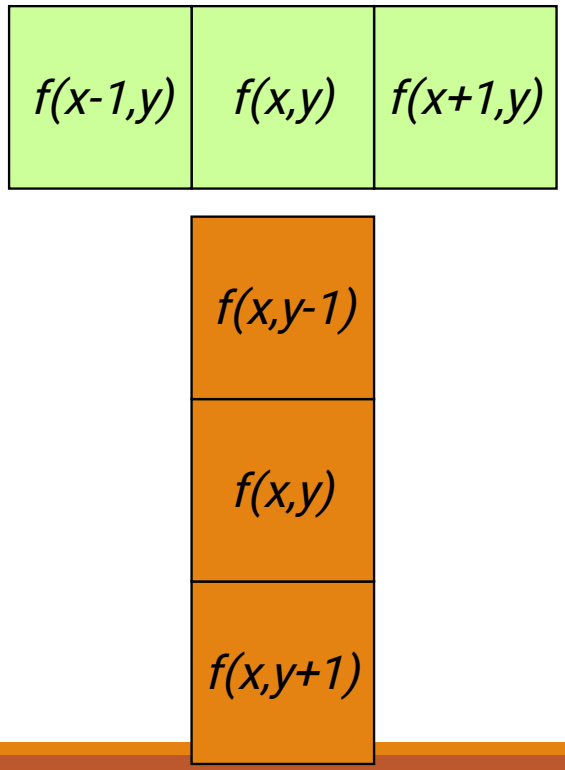
Observation	Inference
Along the ramp first derivative is non zero and second derivative is non zero at the onset and end of ramp.	First order derivative produces thick edges whereas second order derivative produces much finer ones.
At isolated noise point SOD has much more response than FOD.	SOD have strong response to fine details like noise, isolated points etc.
Ramp and step edges SOD has opposite signs. (-1 to 1 and 7 to -7)	SOD produce a double edge response at ramp and step transitions in intensity and it's sign determines edge transition from dark to light or vice versa.

The Laplacian (2nd order derivative)

Shown by Rosenfeld and Kak[1982] that the simplest isotropic derivative operator is the Laplacian is defined as

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

Discrete form of derivative



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

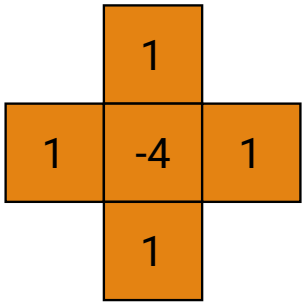
$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2 f(x, y)$$

2-Dimentional Laplacian

The digital implementation of the 2-Dimensional Laplacian is obtained by summing 2 components

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

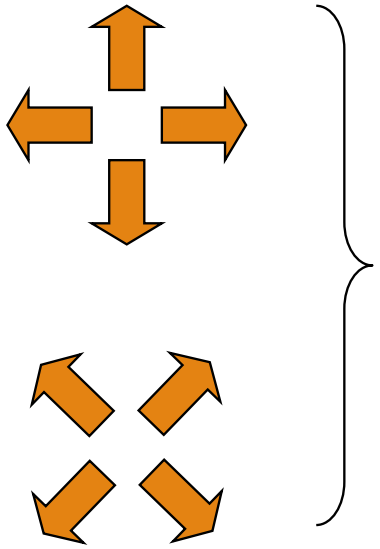
$$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4 f(x, y)$$



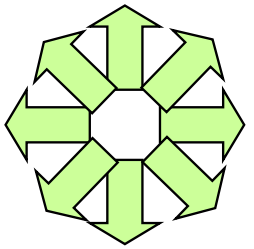
Laplacian

0	1	0
1	-4	1
0	1	0

1	0	1
0	-4	0
1	0	1

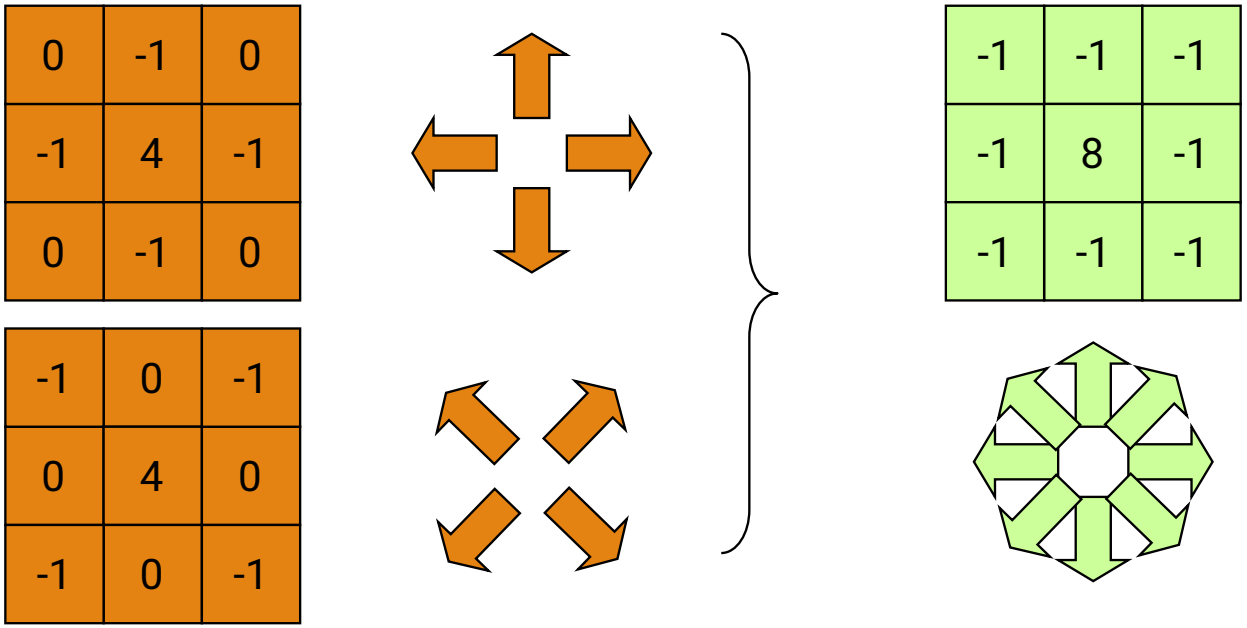


1	1	1
1	-8	1
1	1	1



Laplacian

Isotropic filter response is independent of the direction of the discontinuities in the image to which the filter is applied.



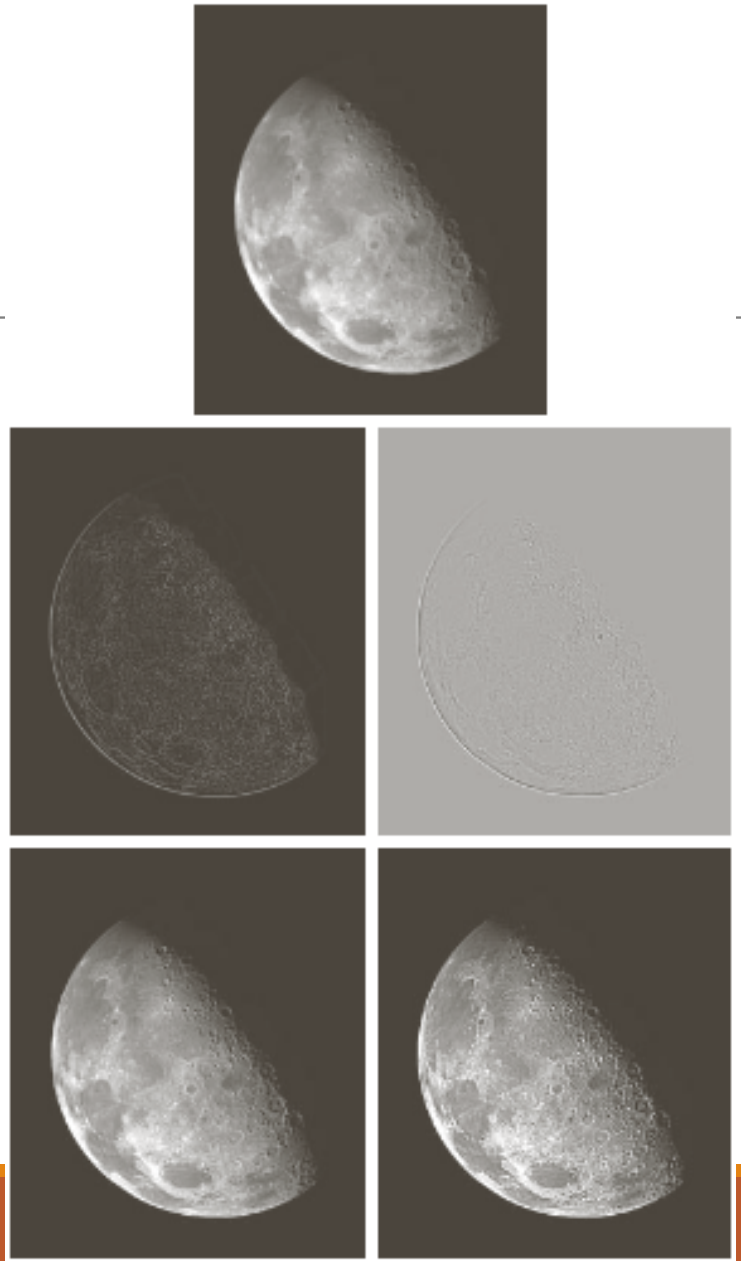
Implementation

$$g(x, y) = \begin{cases} f(x, y) - \nabla^2 f(x, y) & \textit{If the center coefficient is negative} \\ f(x, y) + \nabla^2 f(x, y) & \textit{If the center coefficient is positive} \end{cases}$$

Where f(x,y) is the original image

$\nabla^2 f(x, y)$ is Laplacian filtered image

g(x,y) is the sharpen image



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.)

Un-sharp Masking and High-boost Filtering

High-boost filtering is used when the original image is blurred and dark.

A process to sharpen images consists of subtracting a blurred version of an image from the image itself. This process, called unsharp masking,

Simplification

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

$$g(x, y) = f(x, y) + k * g_{mask}(x, y) \quad , k = 1$$

- High-boost filtering

$$g(x, y) = f(x, y) + k * g_{mask}(x, y) \qquad , k > 1$$

$$f_{hb} = Af(x, y) - \nabla^2 f(x, y) \qquad A > 1$$

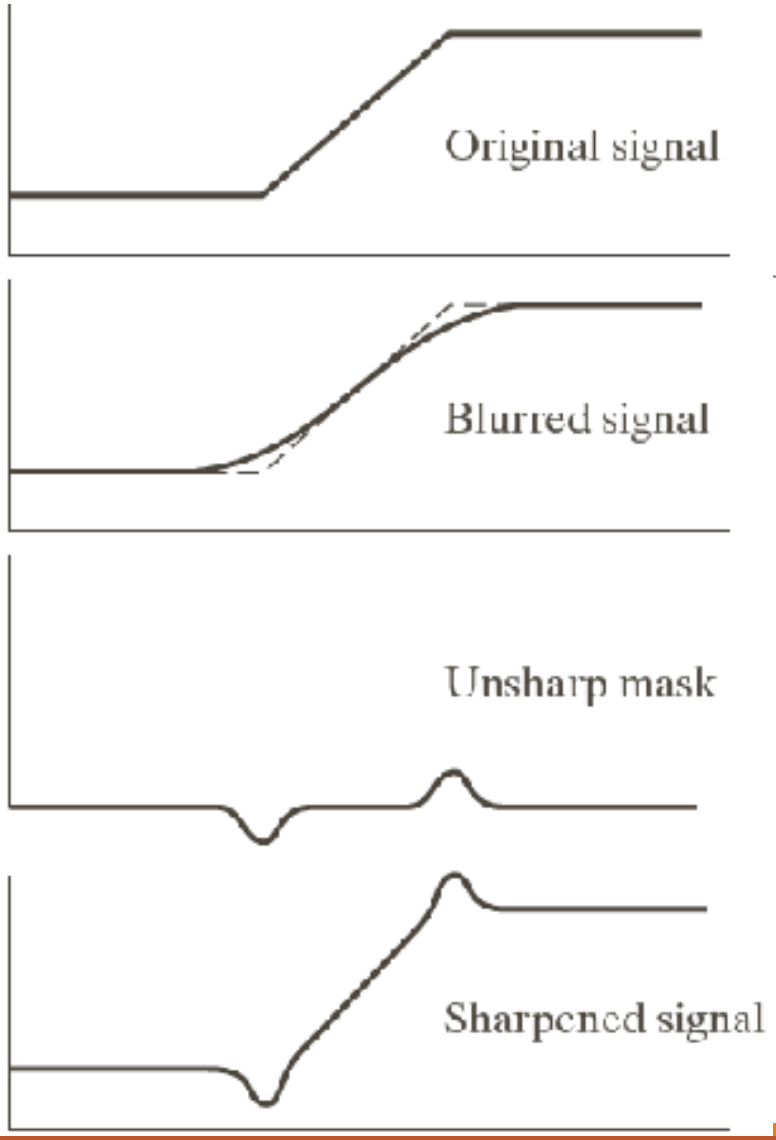
$$g(x, y) = f(x, y) - [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1)] + 4 f(x, y)$$

$$= 5 f(x, y) - [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1)]$$

0	−1	0	−1	−1	−1
−1	$A+4$	−1	−1	$A+8$	−1
0	−1	0	−1	−1	−1

a b

FIGURE 3.42 The high-boost filtering technique can be implemented with either one of these masks with $A \geq 1$.



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.

Use of First Derivative for Nonlinear Sharpening

First derivatives in image processing are implemented using the greatest rate of change (magnitude) of the gradient.

$$\nabla \mathbf{f} = \left[\frac{\partial f}{\partial x} \quad \frac{\partial f}{\partial y}\right]^t$$

$$\nabla f = mag\,(\nabla \mathbf{f}) = \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{0.5} \approx |G_x| + |G_y|$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Roberts operator

$G_x = (z_9 - z_5)$ and $G_y = (z_8 - z_6)$

Sobel operator

$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$ and

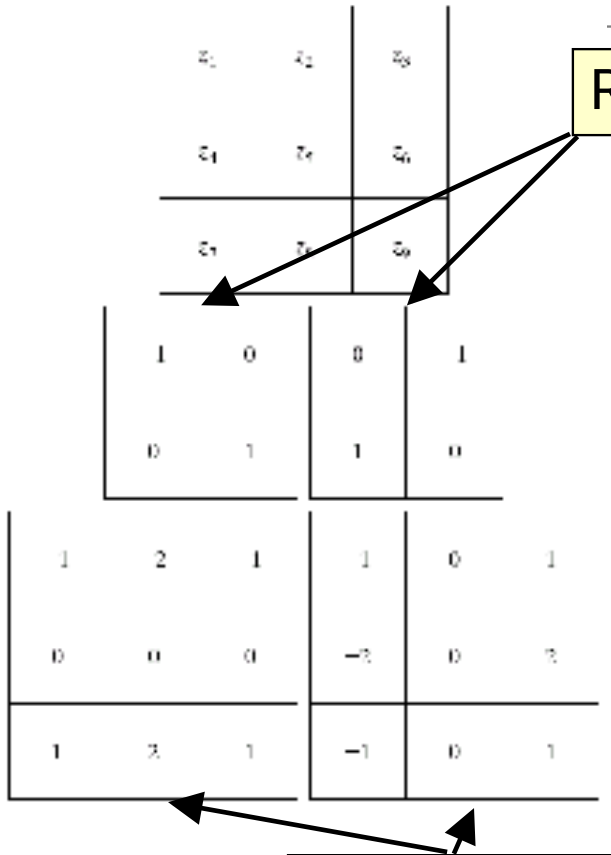
$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$

Use of First Derivative for Edge Extraction

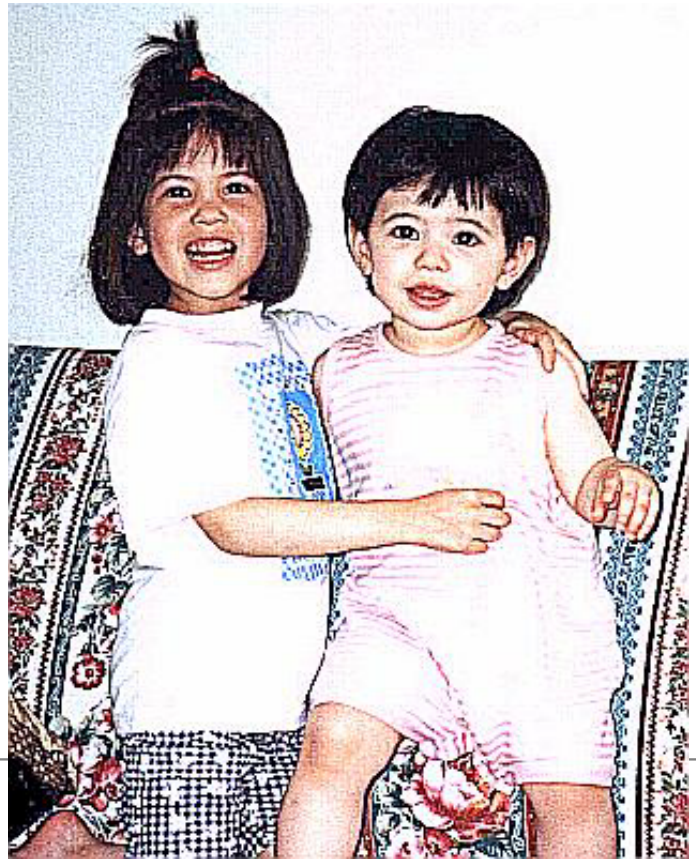
Gradient

a
b c
d e

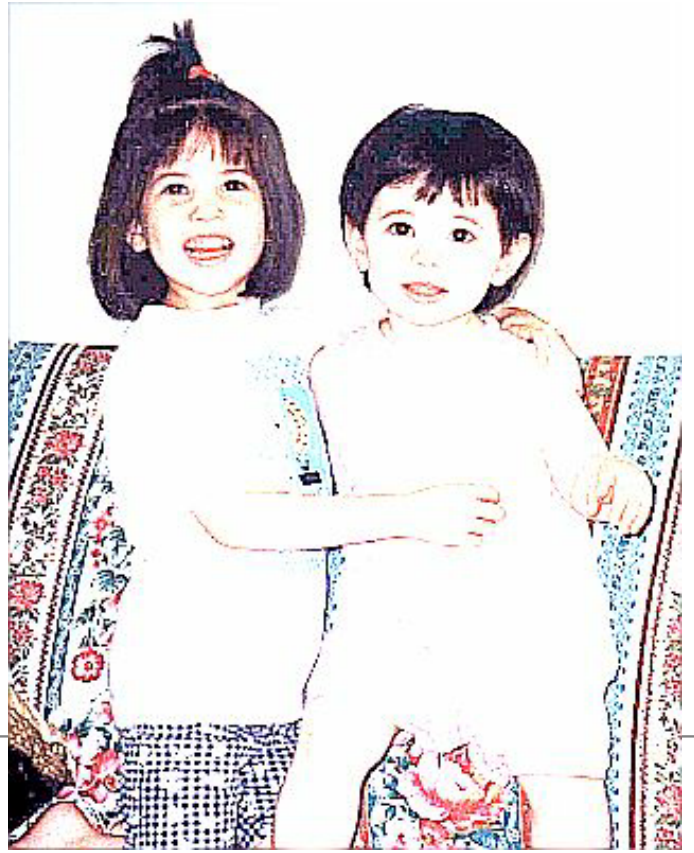
FIGURE 3.44
A 3 × 3 region of an image (the z's are gray-level values) and masks used to compute the gradient at point labeled z_5. All masks coefficients sum to zero, as expected of a derivative operator.



-1	-1	-1
-1	8	-1
-1	-1	-1



-1	-1	-1
-1	9	-1
-1	-1	-1



-1	-1	-1
-1	10	-1
-1	-1	-1

Template Matching

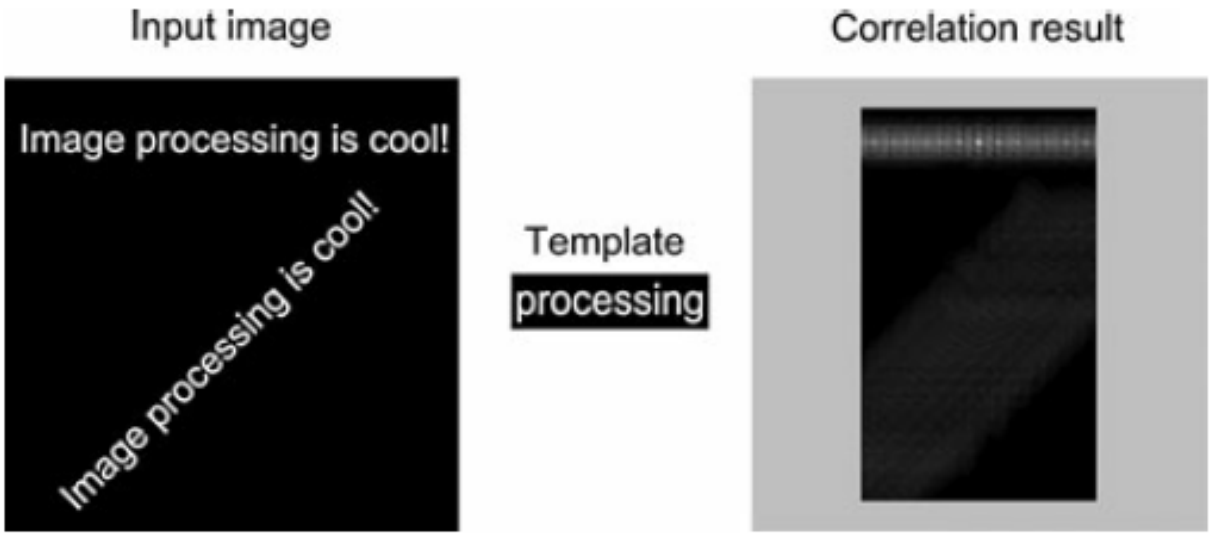


Fig. Template matching performed by correlating the input image with a template.

The result of template matching is seen to the *right.* The *gray outer region* illustrates the pixels that cannot be processed due to the border problem

- To avoid this problem we need to normalize the values in the output so they are independent of the overall level of light in the image.
- To assist us in doing so we can use a small trick. Let us denote the template *H and the image patch F. These are* both matrices, but by rearranging we can easily convert each matrix into a vector by concatenating each row (or column) in the matrix, i.e., $\vec{F}$ *and*

$$\cos\theta = \frac{\vec{H} \bullet \vec{F}}{|\vec{H}| \cdot |\vec{F}|}$$

$$\text{Length of template} = \sqrt{\sum_{j=-R}^{R} \sum_{i=-R}^{R} h(i,j) \cdot h(i,j)}$$

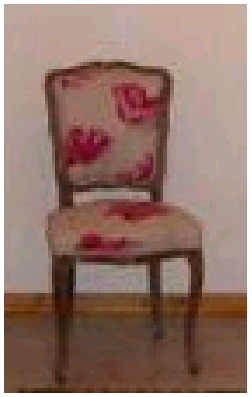
$$\text{NCC}(x,y) = \frac{\text{Correlation}}{\text{Length of image patch} \cdot \text{Length of template}} \Rightarrow \quad [0,1]$$

$$\text{NCC}(x,y) = \frac{\sum_{j=-R}^{R} \sum_{i=-R}^{R} (H \cdot F)}{\sqrt{\sum_{j=-R}^{R} \sum_{i=-R}^{R} (F \cdot F)} \cdot \sqrt{\sum_{j=-R}^{R} \sum_{i=-R}^{R} (H \cdot H)}}$$

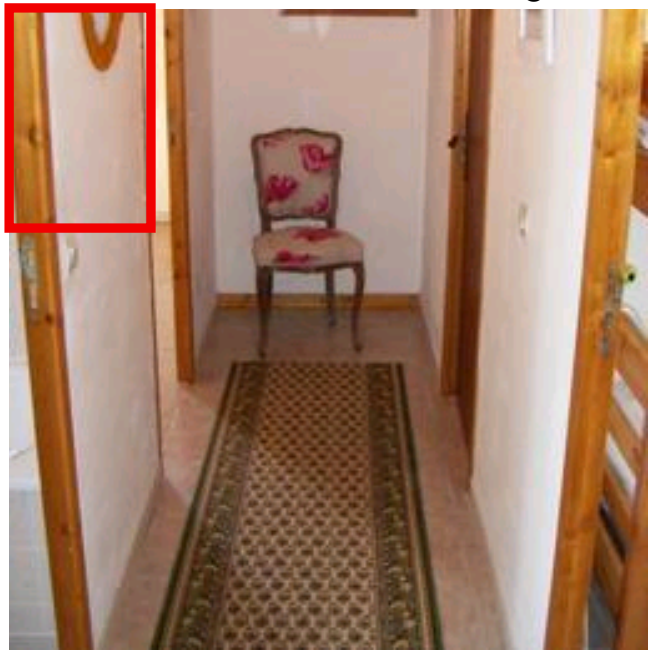
zero-mean normalized cross-correlation

Object recognition
Is it really so hard?

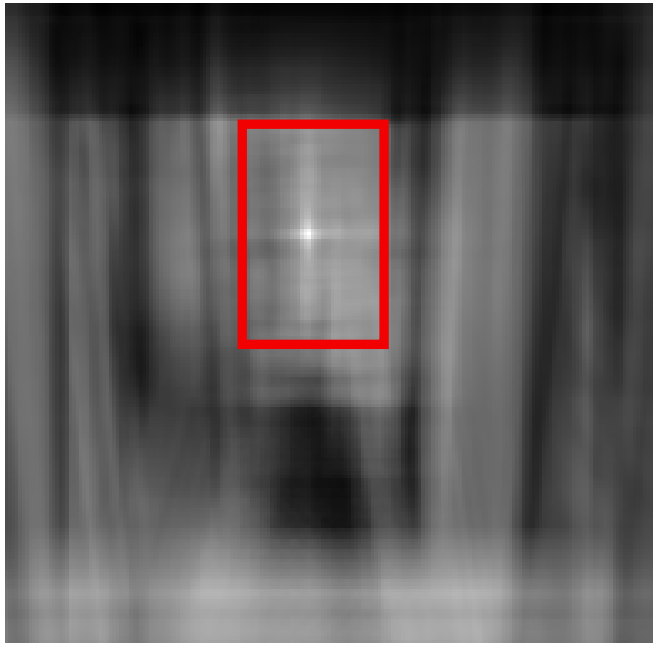
This is a chair

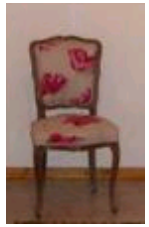


Find the chair in this image



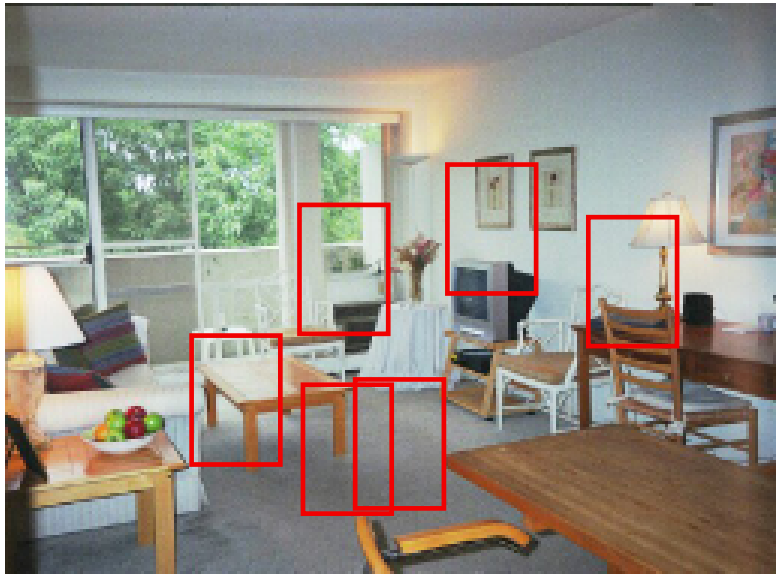
Output of normalized correlation

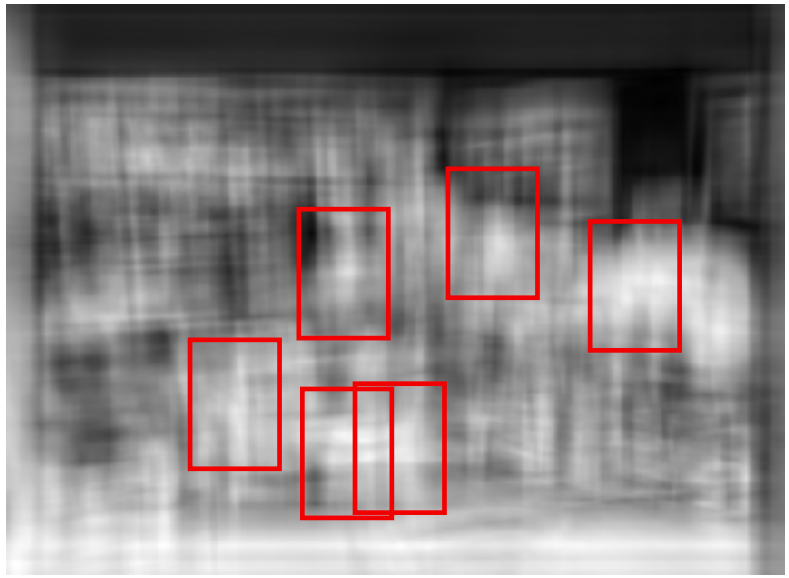




Object recognition
Is it really so hard?

Find the chair in this image





Pretty much garbage
Simple template matching is not going to make it

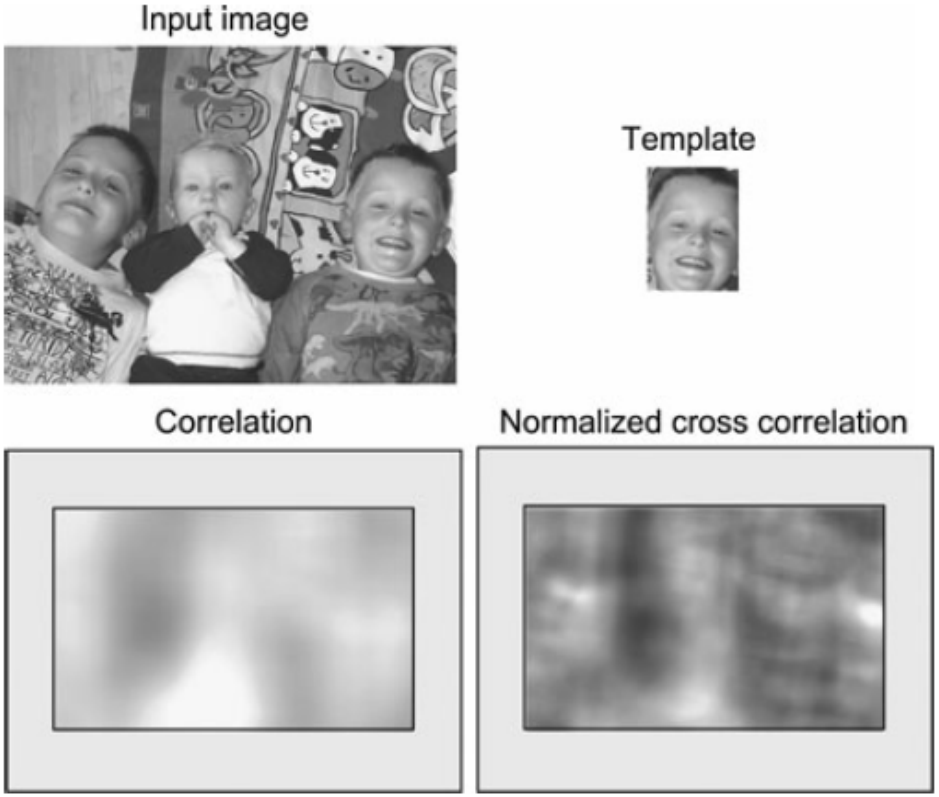


Fig. Template matching using correlation and normalized cross-correlation.

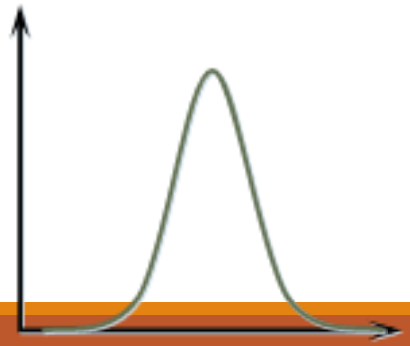
The *gray regions illustrate the pixels* that cannot be processed due to the border problem

Gaussian Filter

Very commonly used filter

Weierstrass transform.

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$$

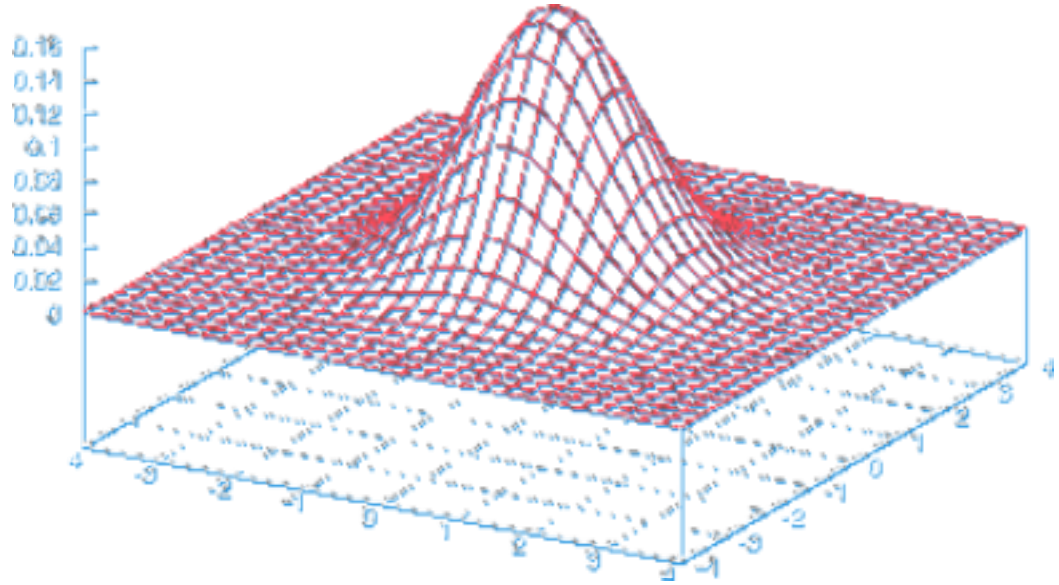


Gaussian Filters

Gaussians are used because:

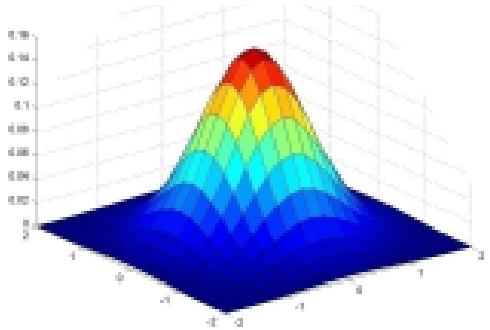
- Smooth (infinitely differentiable)
- Decay to zero rapidly
- Simple analytic formula
- Separable: multidimensional Gaussian = product of Gaussians in each dimension
- Convolution of 2 Gaussians = Gaussian

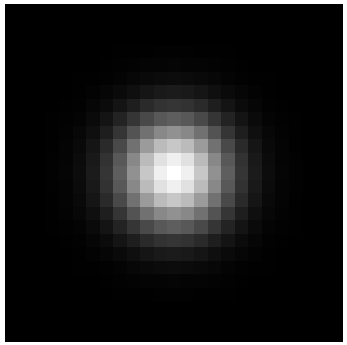
2D Gaussian Filter



Gaussian Kernel

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





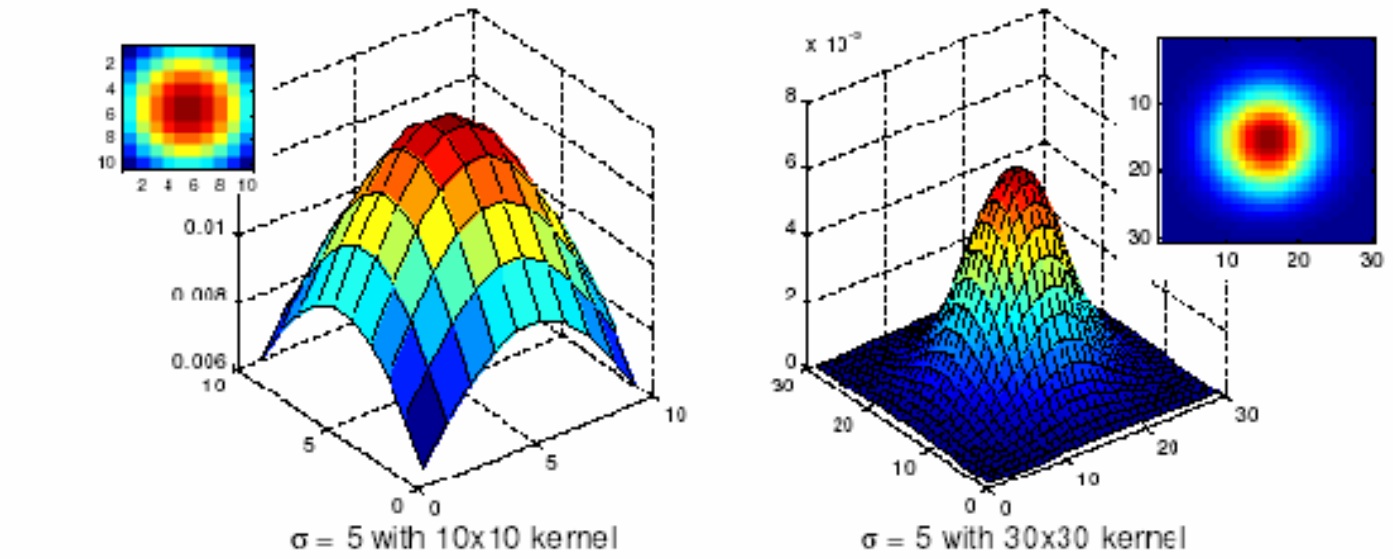
0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, $\sigma = 1$

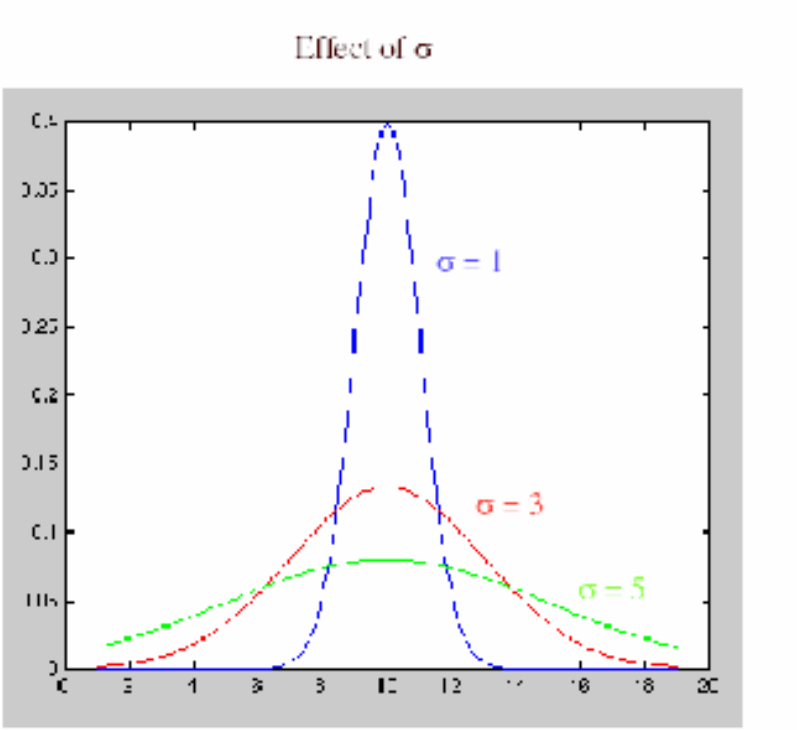
- Constant factor at front makes volume sum to 1 (can be ignored, as we should re-normalize weights to sum to 1 in any case)

Choosing kernel width

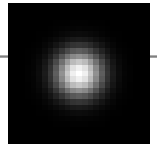
- Gaussian filters have infinite support, but discrete filters use finite kernels

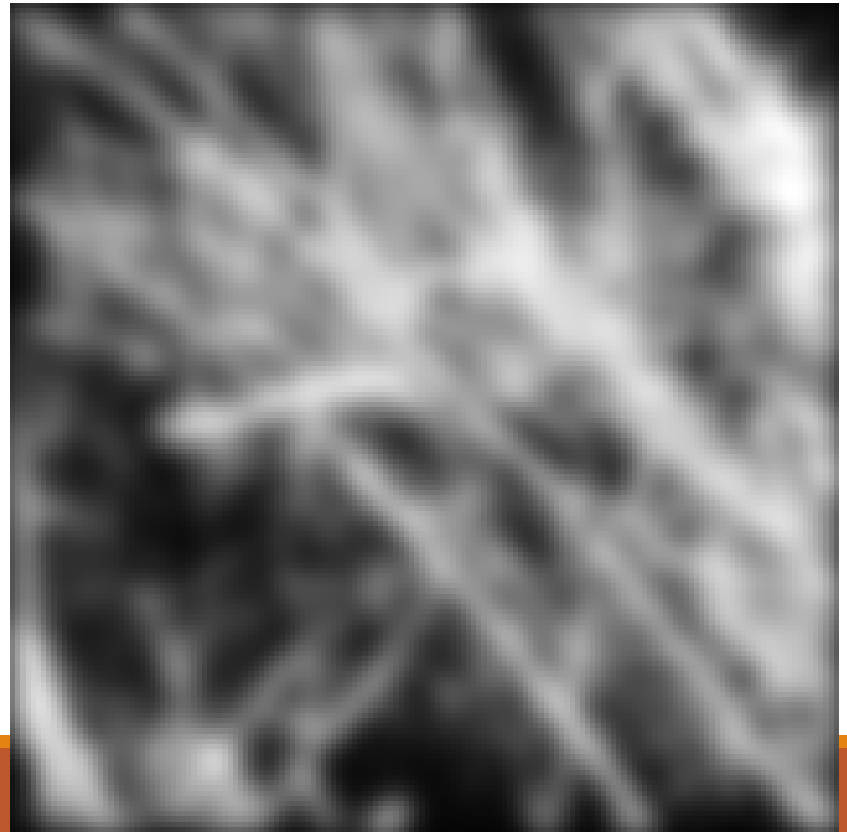


Choosing kernel width

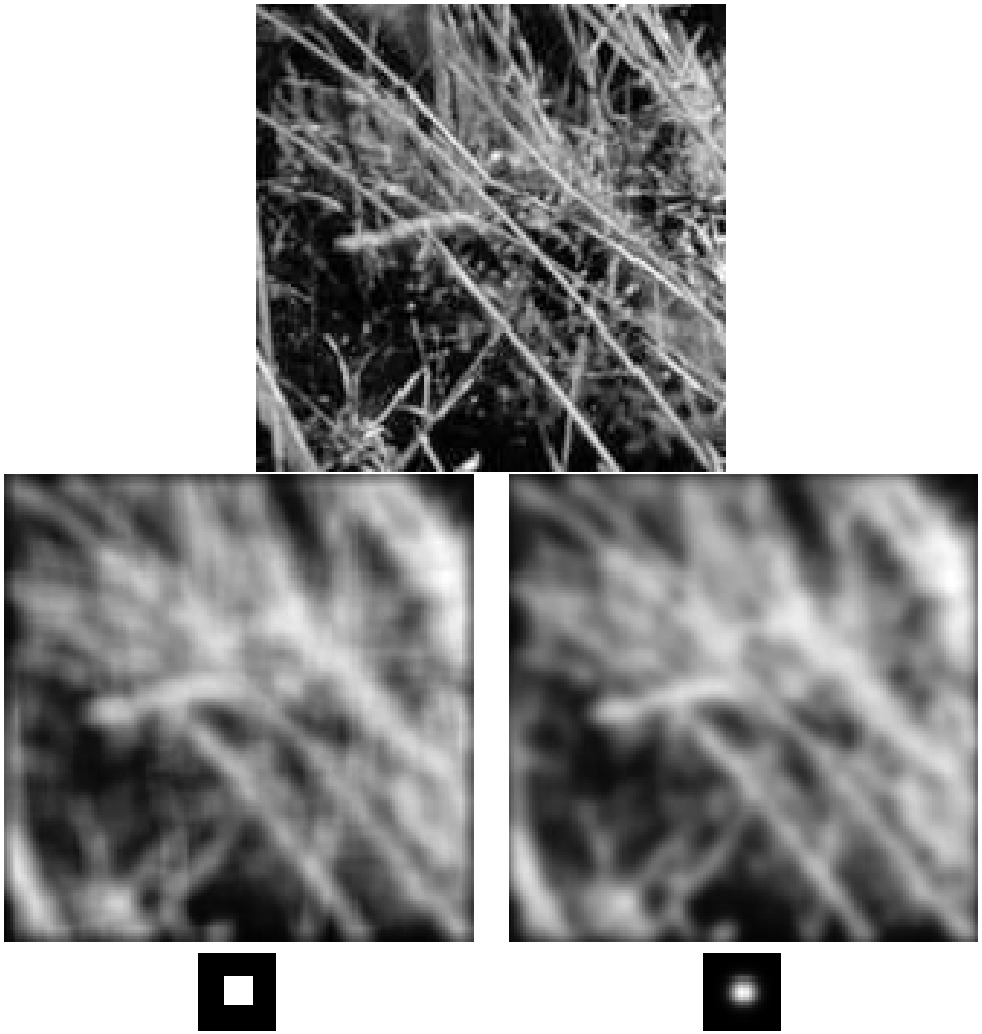


Example: Smoothing with a Gaussian





Mean vs. Gaussian filtering



Gaussian filters

- Remove “high-frequency” components from the image (low-pass filter)
- Convolution with self is another Gaussian
  - So can smooth with small-width kernel, repeat, and get same result as larger-width kernel would have
- *Separable* kernel
  - Factors into product of two 1D Gaussians

Separability of the Gaussian filter

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

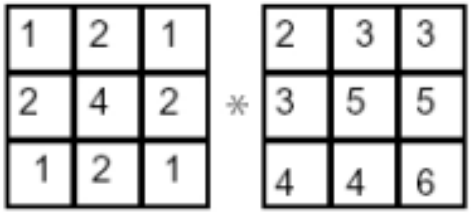
$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)$$

The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

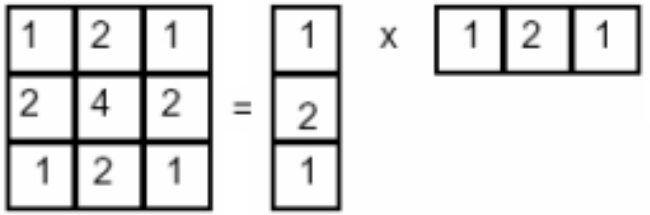
In this case, the two functions are the (identical) 1D Gaussian

Separability example

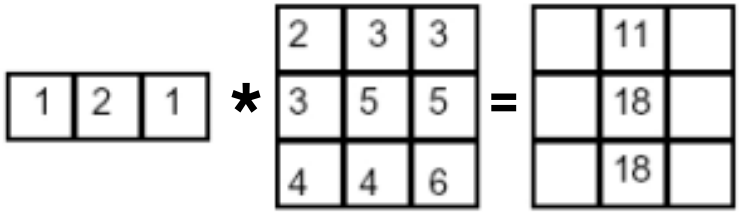
2D convolution
(center location only)



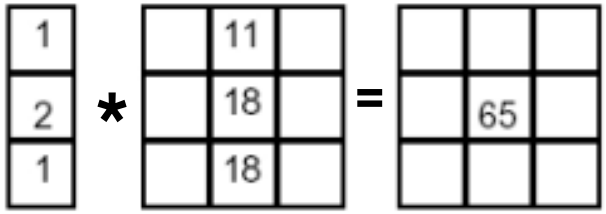
The filter factors into a product of 1D filters:



Perform convolution along rows:



Followed by convolution along the remaining column:



For MN image, PQ filter: 2D takes MNPQ add/times, while 1D takes MN(P + Q)