Disclaimer

- All images/contents used in this presentation are copyright of original owners.
- The PPT has prepared for academic use only.

Image Enhancement

outline

- Introduction
- Image enhancement methods:
  - Spatial-Frequency domain enhancement methods
    - Point operations
    - Histogram operations
    - Spatial operations
    - Transform operations
- Multi-spectral image enhancement
- False color and pseudocoloring
- Color image enhancement

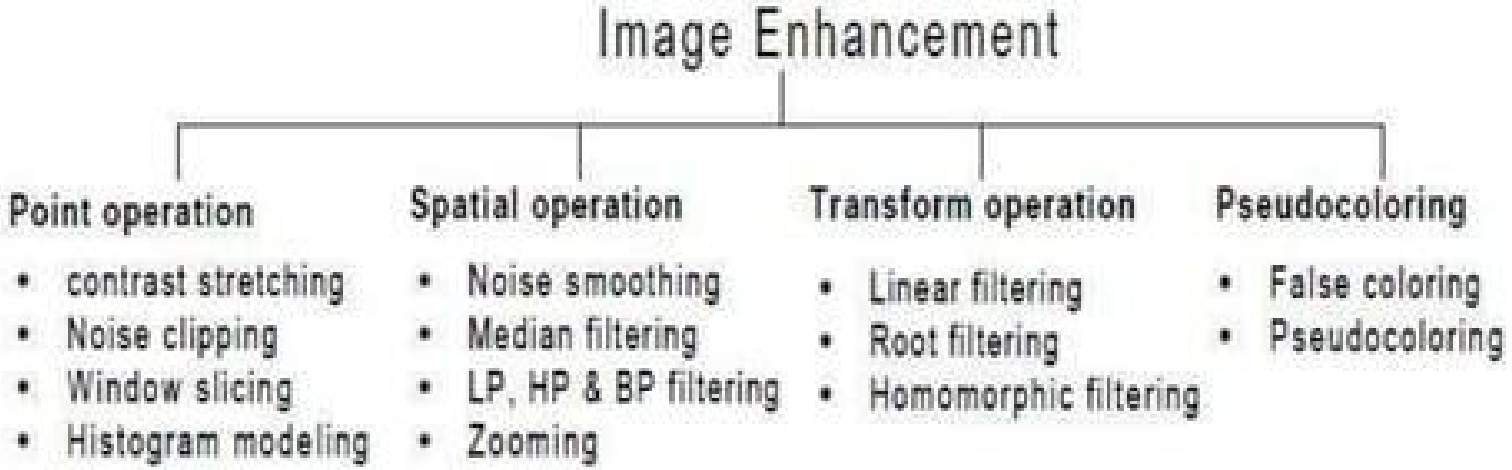
Introduction

- The principal objective of image enhancement is to process a given image so that the result is more suitable than the original image for a specific application.

- It accentuates or sharpens image features such as edges, boundaries, or contrast to make a graphic display more helpful for display and analysis.

- The enhancement doesn't increase the inherent information content of the data, but it increases the dynamic range of the chosen features so that they can be detected easily.

Cont.



- The greatest difficulty in image enhancement is quantifying the criterion for enhancement and, therefore, a large number of image enhancement techniques are empirical and require interactive procedures to obtain satisfactory results.

- Image enhancement methods can be based on either spatial or frequency domain techniques.

Spatial-Frequency domain enhancement methods

Spatial domain enhancement methods:

- Spatial domain techniques are performed to the image plane itself and they are based on direct manipulation of pixels in an image.
- The operation can be formulated as $g(x,y) = T[f(x,y)]$,

  where g is the output, f is the input image and T is an operation on *f defined over some* neighborhood of (x,y).
- According to the operations on the image pixels, it can be further divided into 2 categories: *Point operations and spatial operations*.

Frequency domain enhancement methods:

- These methods enhance an image *f(x,y) by* convoluting the image with a linear, position invariant operator.
- The 2D convolution is performed in frequency domain with DFT.

  Spatial domain: $g(x,y)=f(x,y)*h(x,y)$

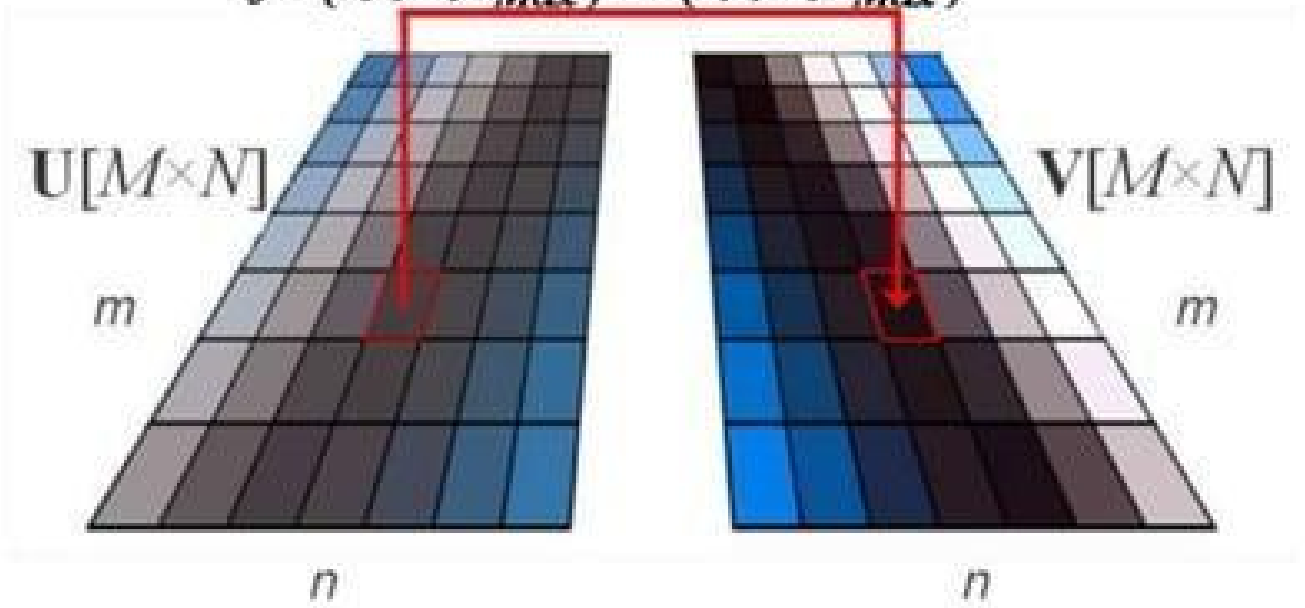
  Frequency domain: $G(w1,w2)=F(w1,w2)H(w1,w2)$

Point operations

-Zero-memory operations where a given gray level u∈[0,L] is mapped into a gray level v∈[0,L] according to a transformation.

v(m,n)=*f*(u(m,n))

$$v(m,n) = f\big(u(m,n)\big),\ \forall m = 0,1,\ldots,M-1; n = 0,1,\ldots,N-1;$$
$$f : \{0,1,\ldots,L_{Max}\} \rightarrow \{0,1,\ldots,L_{Max}\}$$



1-contrast stretching

- The idea behind contrast stretching is to increase the dynamic range of the gray levels in the image being processed.

- Low contrast images occur often due to :
  - -poor or nonuniform lightning conditions
  - -small dynamic range of imaging sensors

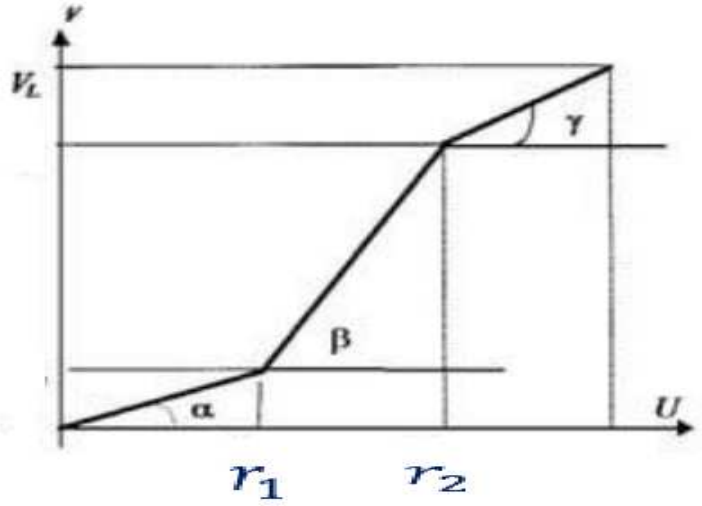
- Expressed as :

$$S = \begin{cases} \alpha r; & 0 \le r < r_1 \\ \beta(r - r_1) + S_1; & r_1 \le r < r_2 \\ \gamma(r - r_2) + S_2; & r_2 \le r < L \end{cases}$$

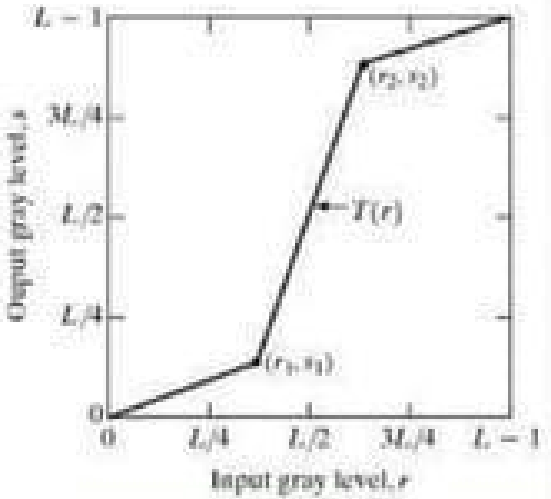
-For dark region stretch α>1

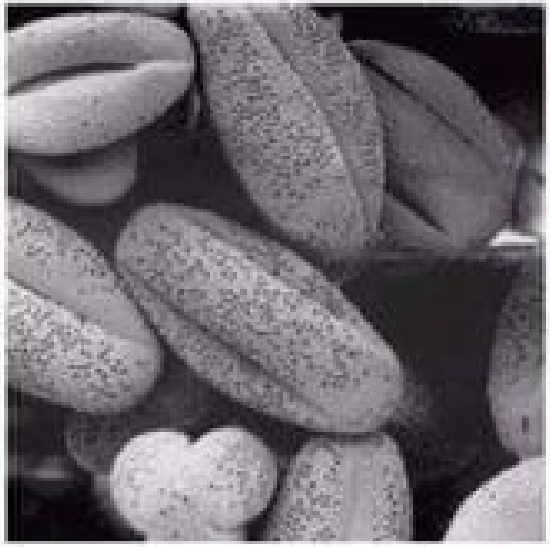
-For mid region stretch β>1

-For bright region stretch γ>1



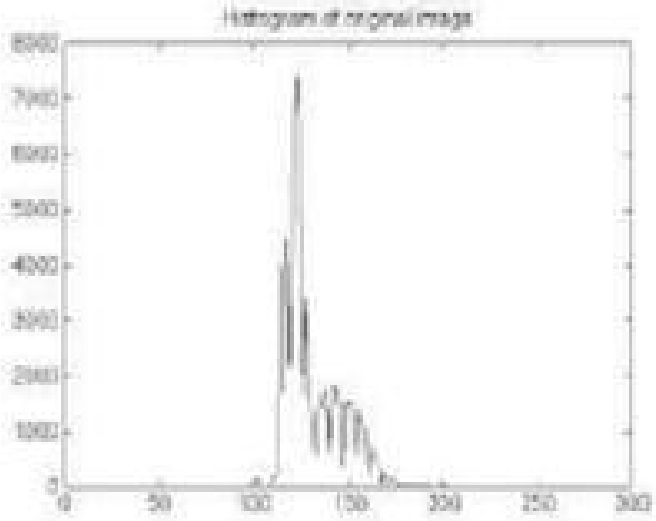
Example 1



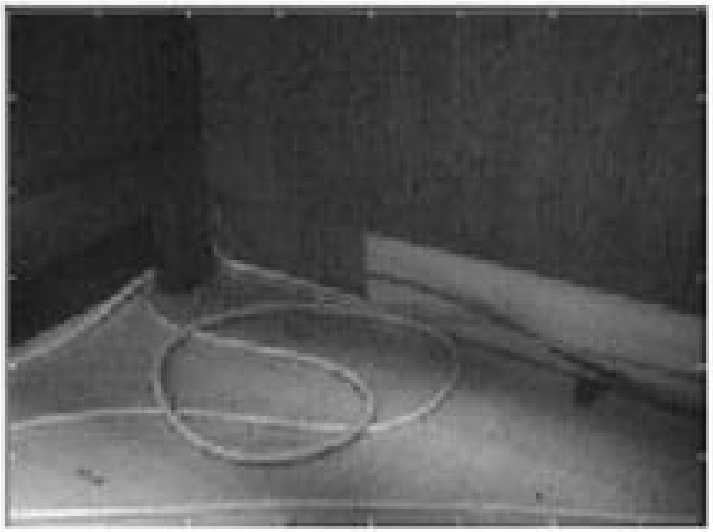


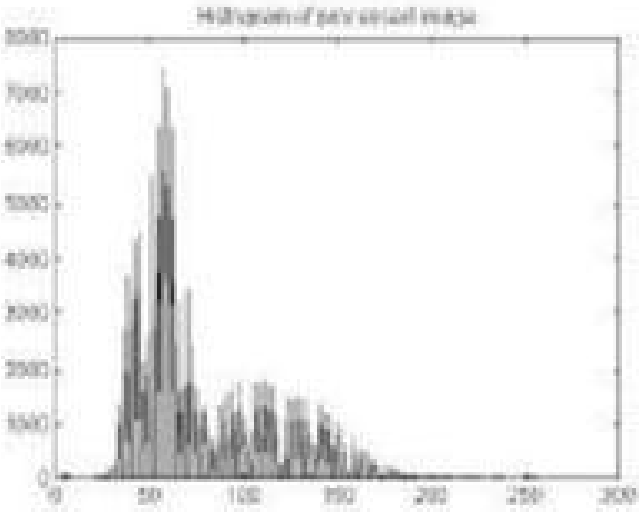
- (b) a low-contrast image : results from poor illumination, lack of dynamic range in the imaging sensor, or even wrong setting of a lens aperture of image acquisition
- (c) result of contrast stretching : (r1,s1) = (rmin,0) and (r2,s2) = (rmax,L-1)
- (d)result of thresholding

Example2



Original





Processed image

2-Clipping and Thresholding

- Expressed as :

Noise clipping and thresholding

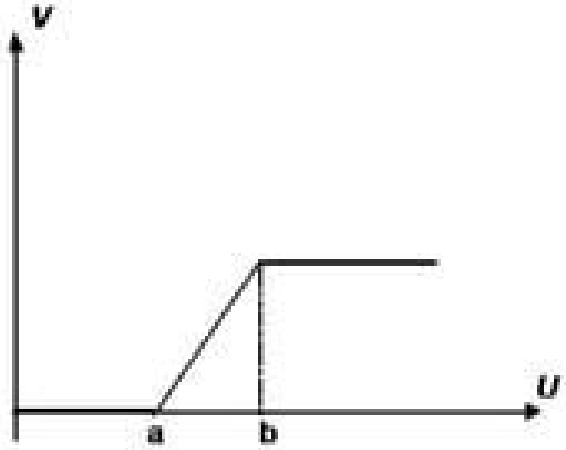
$$f(u) = \begin{cases} 0, & 0 \le u < a \\ \alpha u, & a \le u \le b \\ L, & u \ge b \end{cases}$$

Useful for binary or other images that have bimodal distribution of gray levels. The a and b define the valley between the peaks of the histogram. For $a = b = t$, this is called *thresholding*.

- **Clipping:**

-Special case of contrast stretching ,where α= γ=0

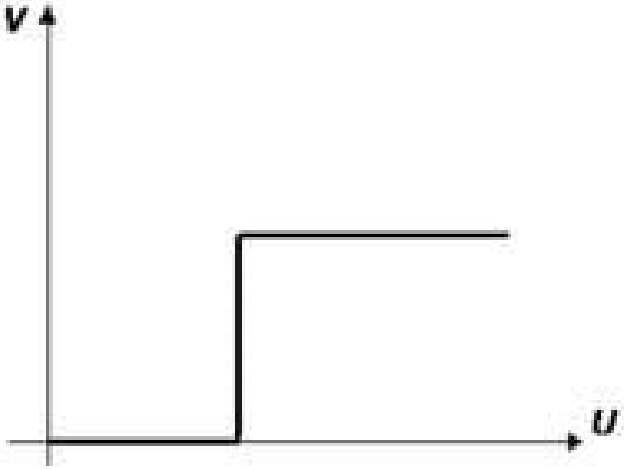
-Useful for noise reduction when the input signal is known to lie in the range [a,b].



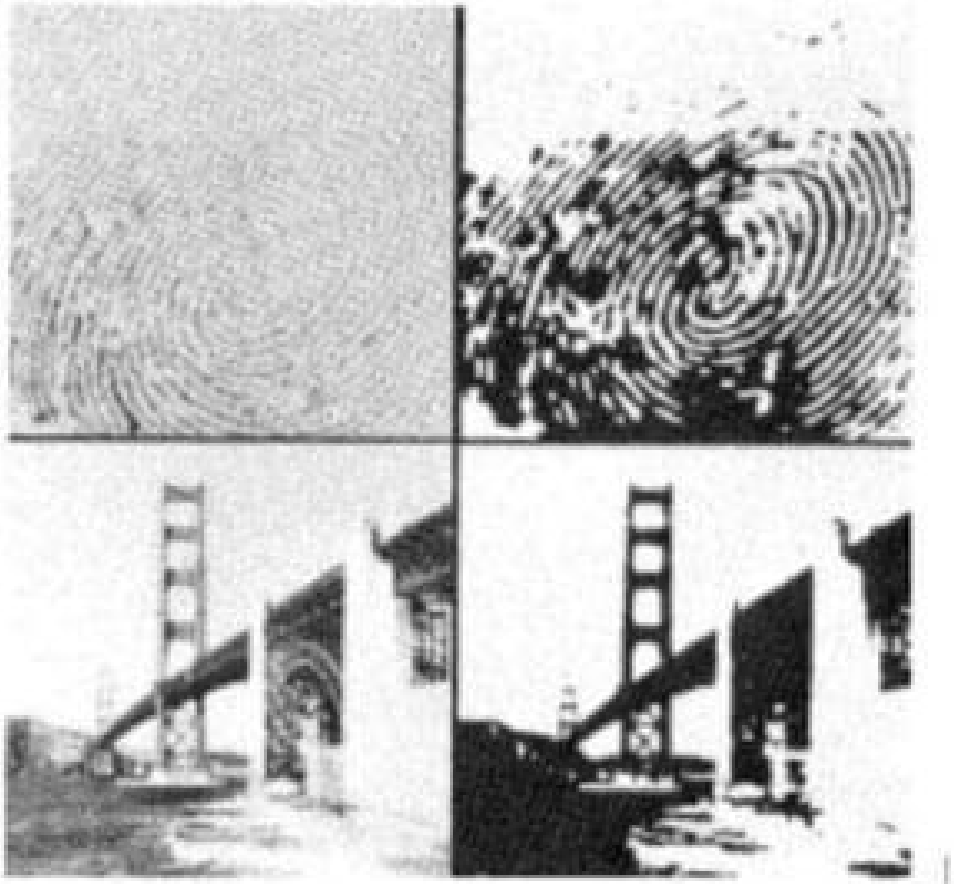
Cont.

- **Thresholding:**

- is a special case of case of clipping where a=b=t and the output comes binary.

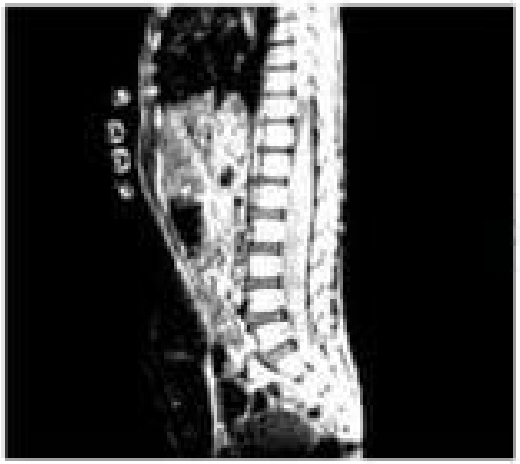


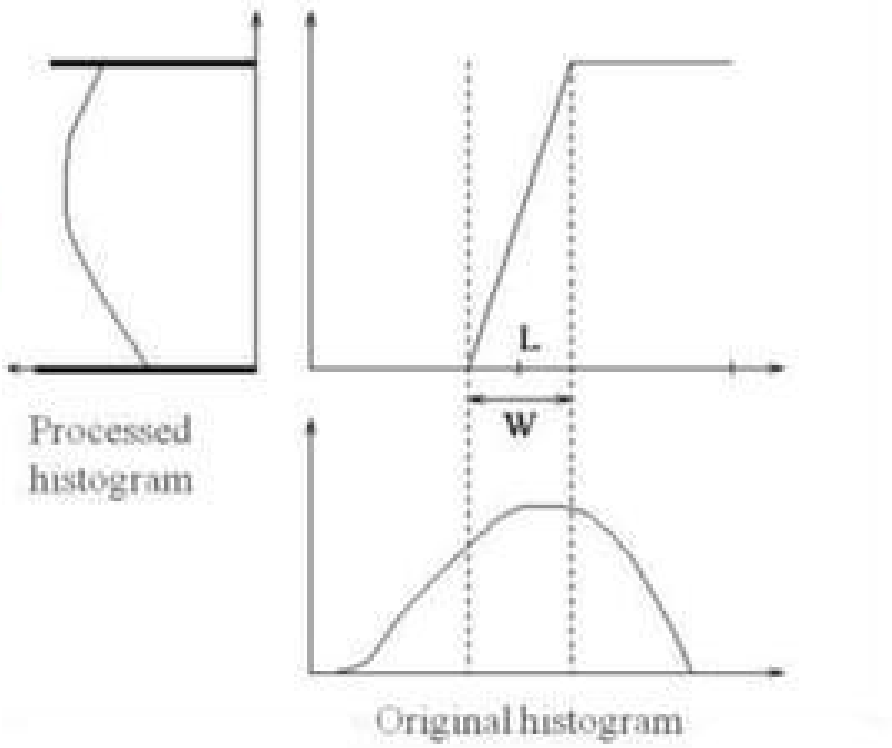
Example 1

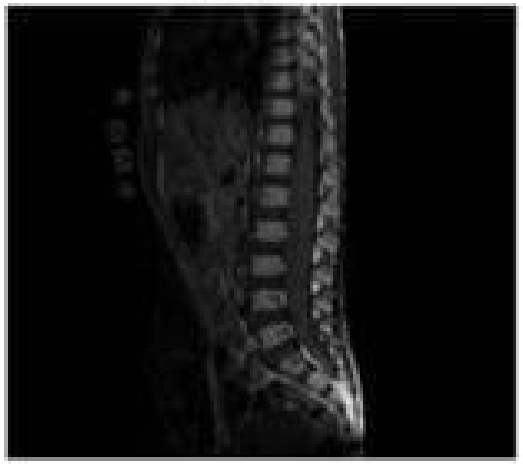


Clipping and Thresholding

Example2



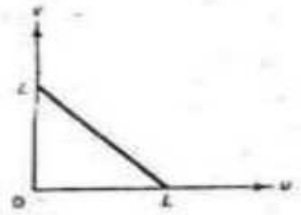




3-Digital negative

- Negative image can be obtained by reverse scaling of the gray levels according to the transformation,

$$S = L - 1 - r$$



- Useful in the display of medical images.
- Example:

4-intensity level slicing

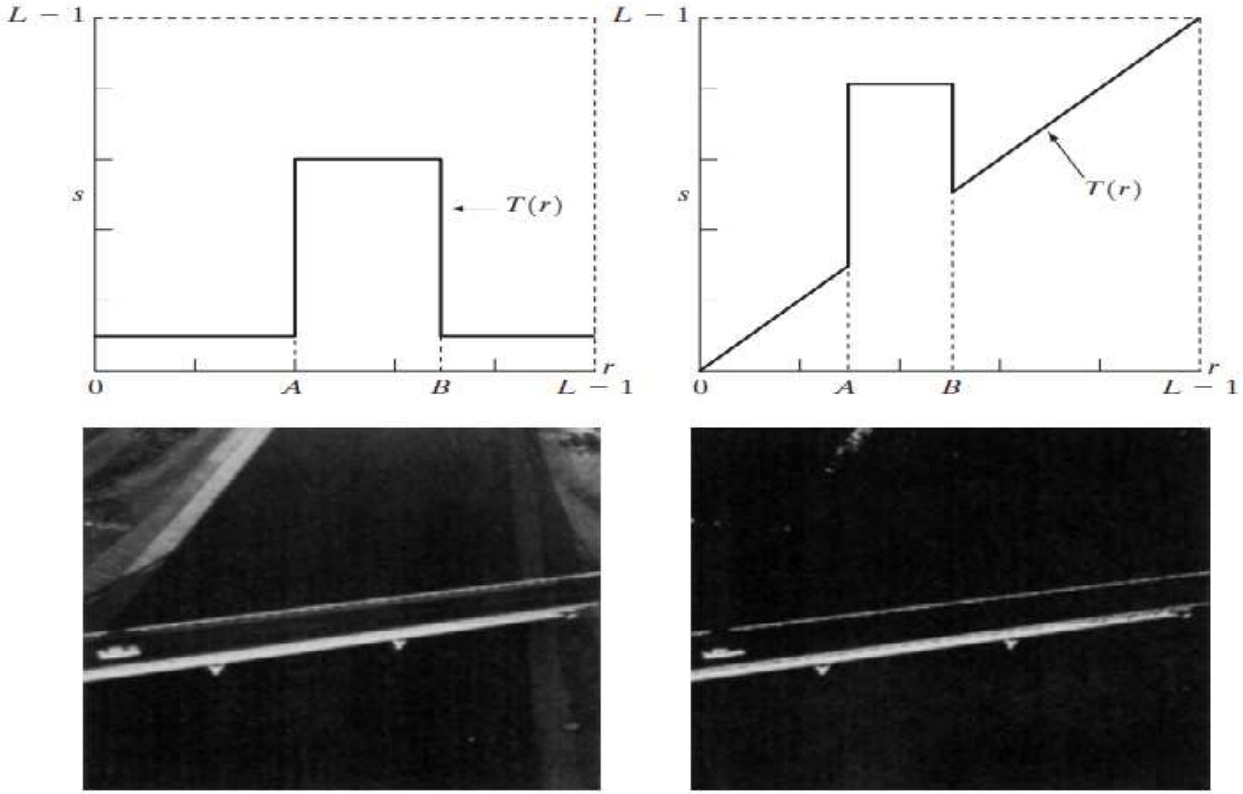
- Permit segmentation of certain gray level regions from the rest of the image.

$$\begin{cases} S = L-1; & r_1 \le r \le r_2 \\ r; & otherwise \end{cases}$$

with background

$$\begin{cases} S = L-1; & r_1 \le r \le r_2 \\ 0; & otherwise \end{cases}$$

without background



a b
c d

FIGURE 3.11
(a) This transformation highlights range $[A, B]$ of gray levels and reduces all others to a constant level. (b) This transformation highlights range $[A, B]$ but preserves all other levels. (c) An image. (d) Result of using the transformation in (a).

Question

Consider the following 3-bit input image. Perform intensity level slicing on the following image (with background and without background).

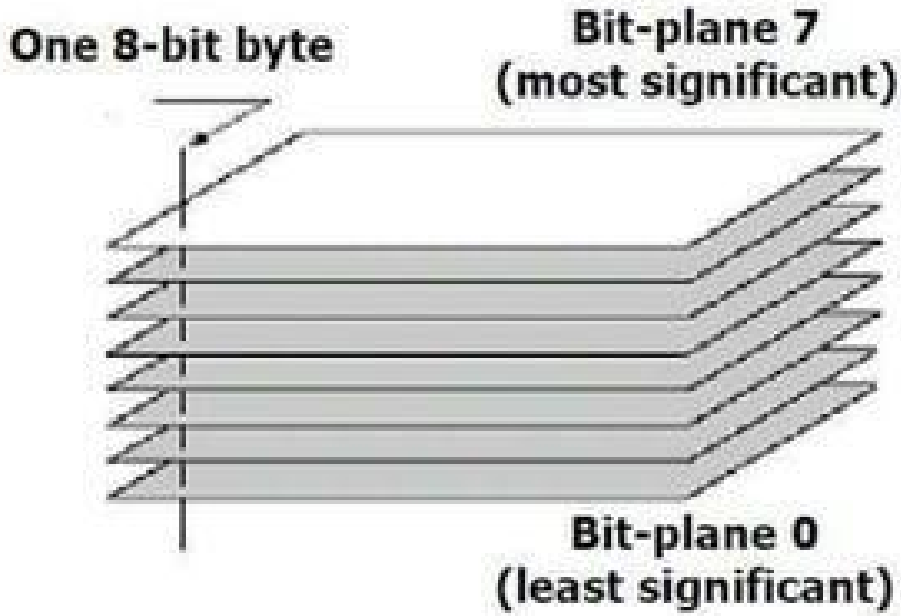
4	2	3	0
1	3	5	7
5	3	2	1
2	4	6	7

Given that: $r_1 = 2;\ r_2 = 5$

5-Bit extraction

- This transformation is useful

In determining the number of
Visually significant bits in an
Image.



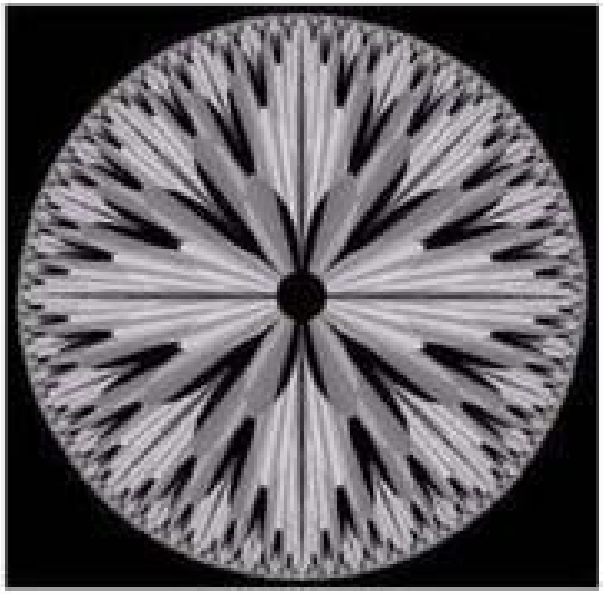
- Suppose each pixel is

represented by 8 bits it is desired
To extract the *n*th most significant bit
And display it .

- Higher-order bits contain the

majority of the visually significant data

Example



8-bit fractal image

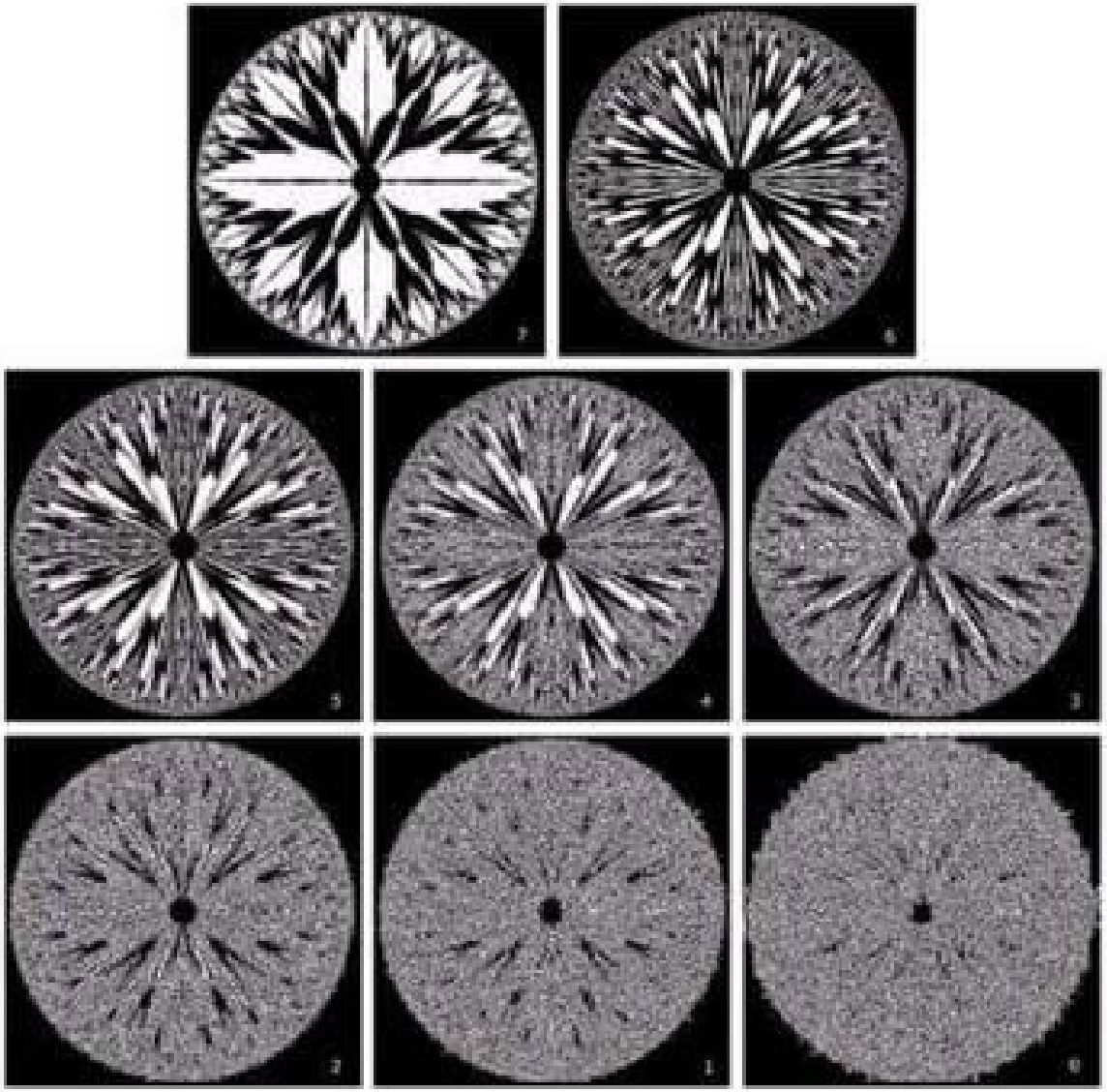
- The (binary) image for bit-plane 7 can be obtained by processing the input image with a thresholding gray-level transformation.

  -Map all levels between 0 and 127 to 0

  -Map all levels between 129 and 255 to 255

Cont.

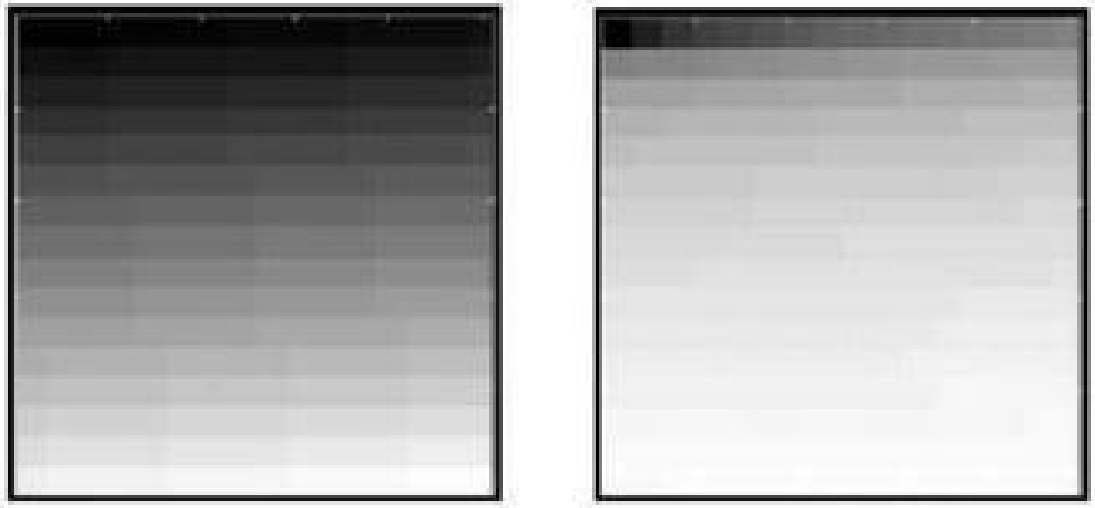
8-bit plane image



Bit-plane 7	Bit-plane 6	
Bit-plane 5	Bit-plane 4	Bit-plane 3
Bit-plane 2	Bit-plane 1	Bit-plane 0

6-Range compression

- Sometimes the dynamic range of a processed image far exceeds the capability of the display device, in which case only the brightest parts of the images are visible on the display screen.



Original Processed output

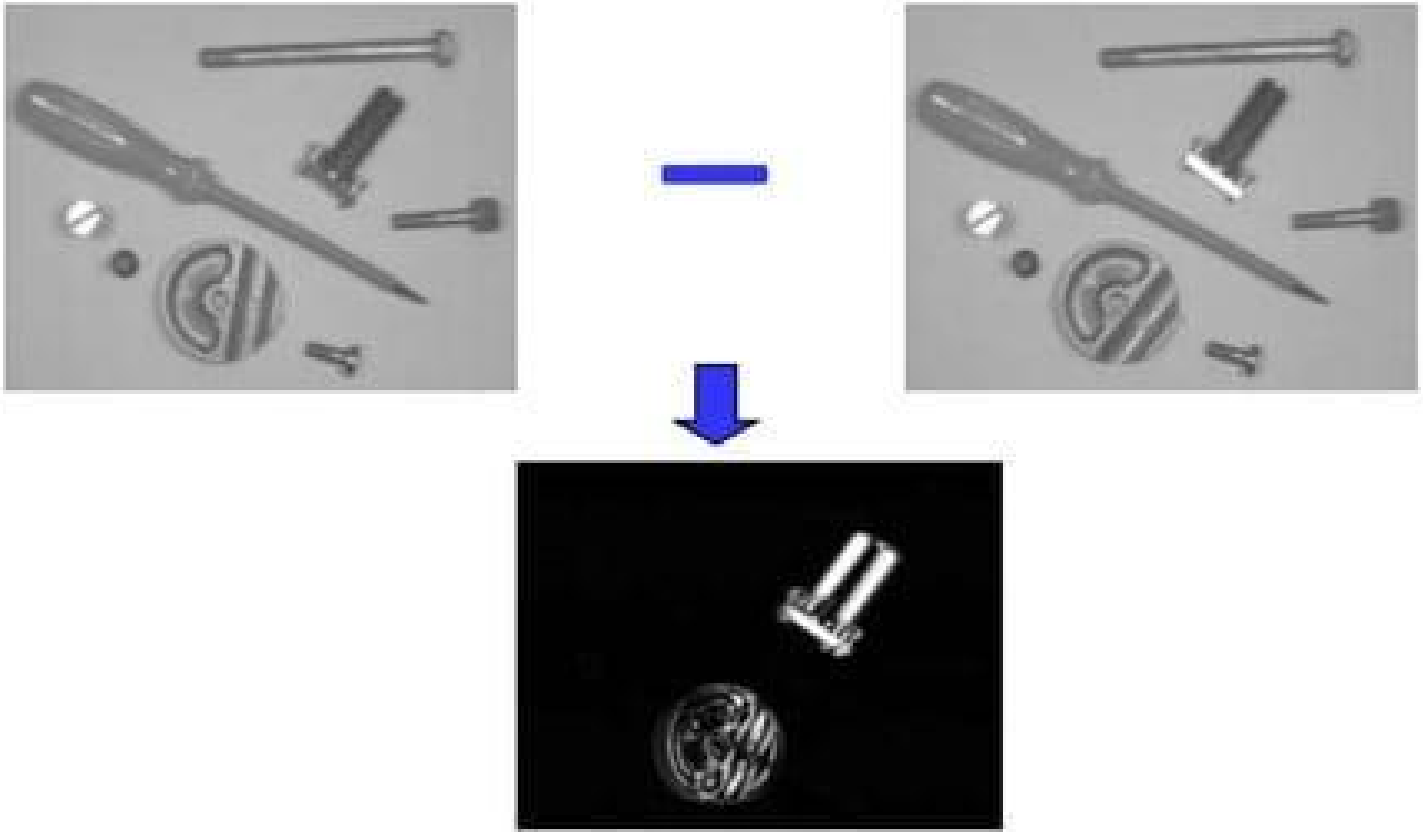
- An effective way to compress the dynamic range of pixel values is to perform the following intensity transformation function:

$$s = c \log(1+|u|)$$

where c is a scaling constant, and the logarithm function performs the desired compression.

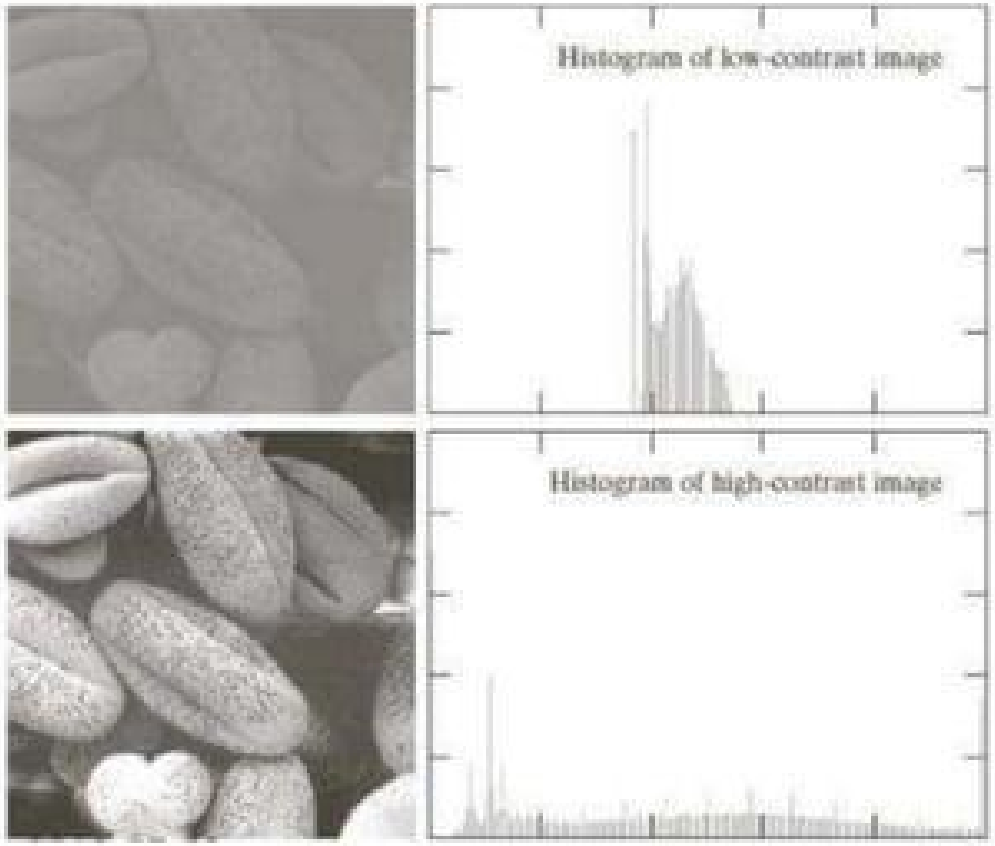
7-Image subtraction and change detection

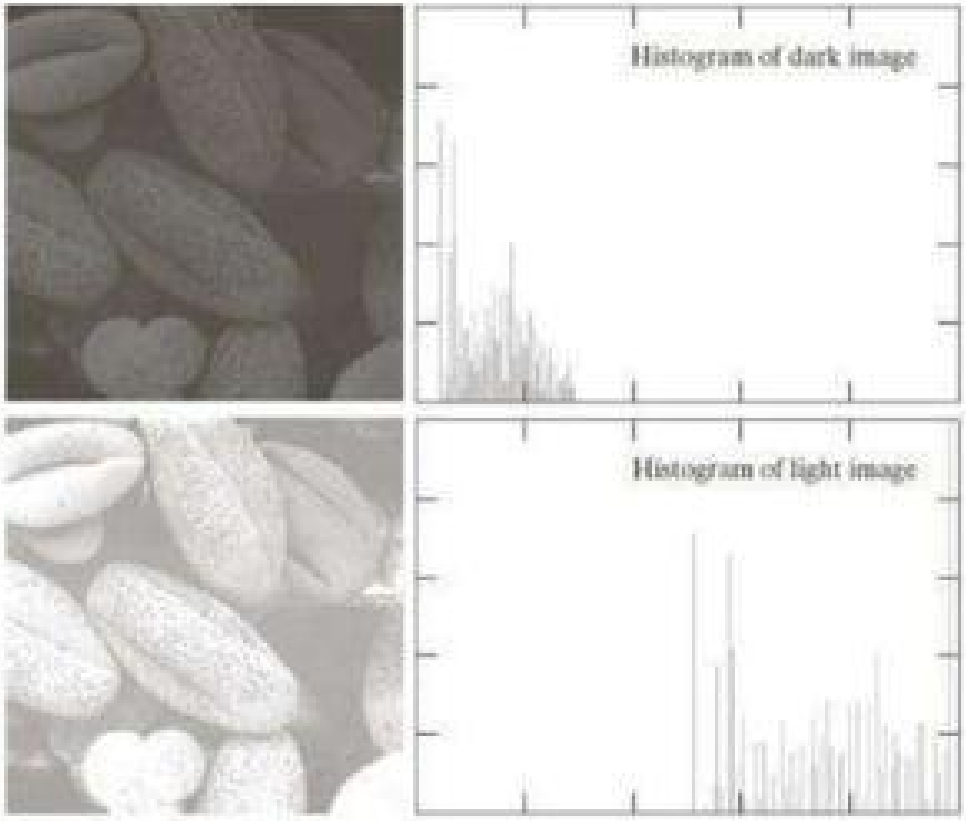
- In many imaging applications it is desired to compare two complicated or busy images .
- A simple ,but powerful method is to align the two images and subtract them .The difference image is then enhanced .
- Applications such as imaging of the blood vessils and arteries in a body , security monitoring systems .
- Example:



Histogram modeling

Histogram modeling techniques modify an image so that it's histogram has a desired shape . This is useful in stretching the low contrast levels with narrow histograms .





It is possible to develop a transformation function that can automatically achieve this effect ,based on histogram of input image .

Histogram modeling Cont.

Image Histogram

- Assume we have image with n_k pixels with intensity r_k, $k = 0, 1, ..., L-1$
- We define the image histogram as $h(r_k) = n_k$, and we define the normalized histogram as:

$$p(r_k) = h(r_k) / (M \cdot N)$$

- The normalized histogram is an estimate of the probability of occurrence of intensity level r_k in an image
- The normalized histogram sums to 1

1-Histogram equalization

- The objective is to map an input image to an output image such that its histogram is uniform after the mapping.
- Let *r* represent the gray levels in the image to be enhanced and s is the enhanced output with a transformation of the form ***s=T(r).***
- Assumptions

1. $T(r)$ is single-valued and monotonically increasing in the interval $[0,1]$, which preserves the order from black to white in the gray scale.
2. $0 \le T(r) \le 1$ for $0 \le r \le 1$, which guarantees the mapping is consistent with the allowed range of pixel values.

- Possible for multiple values of ***r*** to map to a single value of ***s.***

Histogram Equalization cont.

Example 1:

Suppose that a 3-bit image (L=8) of size 64 × 64 pixels (MN = 4096) has the intensity distribution shown in following table.
Get the histogram equalization transformation function and give the $p_s(s_k)$ for each s_k.

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

Solution

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

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

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

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

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

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

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

Solution cont.

final transform:

$r_0 \rightarrow s_0 = 1 \Rightarrow 790$ pixels map to 1

$r_1 \rightarrow s_1 = 3 \Rightarrow 1023$ pixels map to 3

$r_2 \rightarrow s_2 = 5 \Rightarrow 850$ pixels map to 5

$r_3 \rightarrow s_3 = 6 \Rightarrow 656+329 = 985$ pixels map to 6

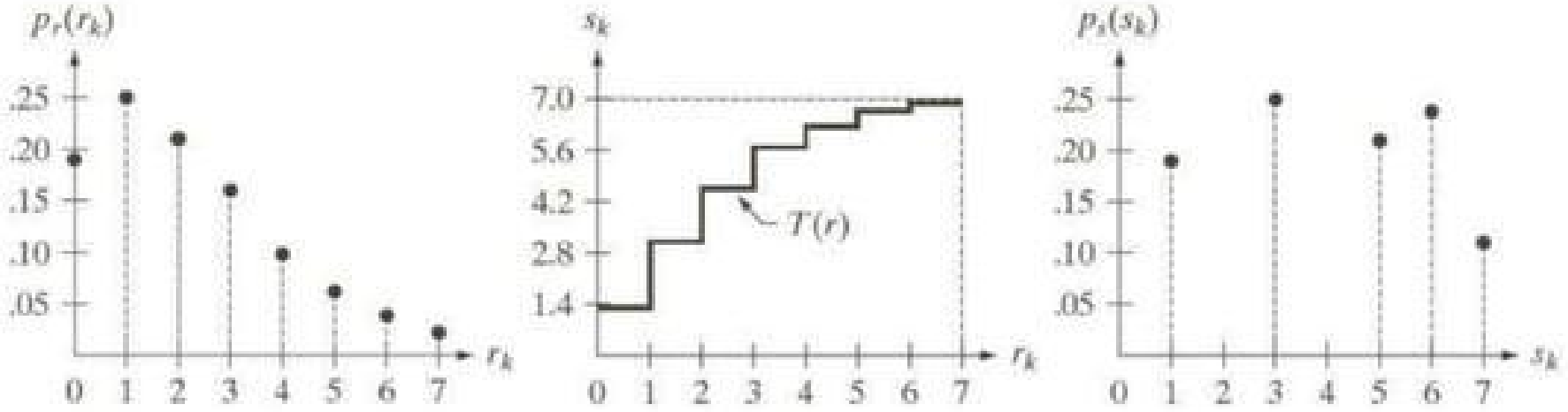
$r_4 \rightarrow s_4 = 6 \Rightarrow 656+329 = 985$ pixels map to 6

$r_5 \rightarrow s_5 = 7 \Rightarrow 245+122+81 = 458$ pixels map to 7

$r_6 \rightarrow s_6 = 7 \Rightarrow 245+122+81 = 458$ pixels map to 7

$r_7 \rightarrow s_7 = 7 \Rightarrow 245+122+81 = 458$ pixels map to 7

Solution cont.



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 2: Equalizing an image of **6** gray levels

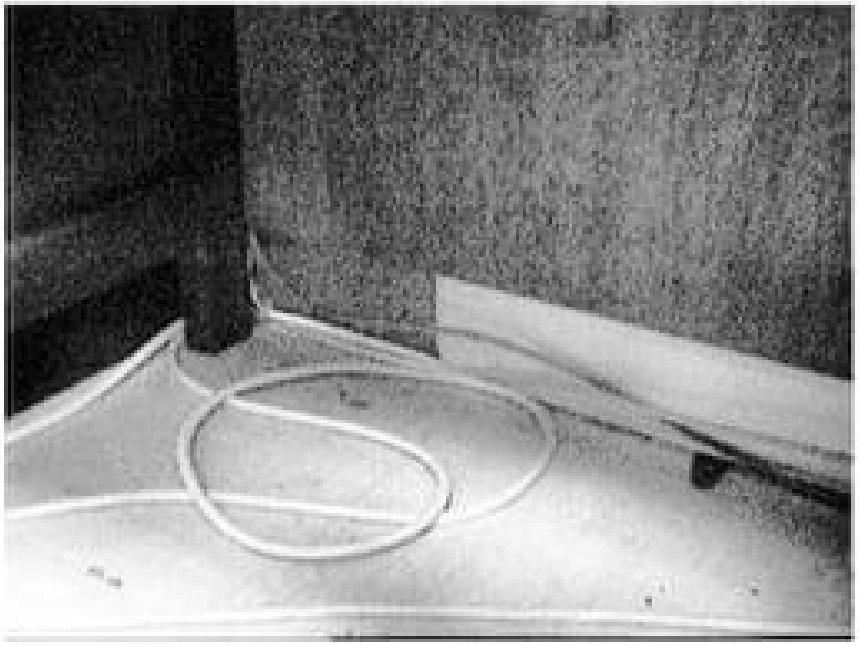
Index k	0	1	2	3	4	5
Normalized Input level. $r_k/5$	0.0	0.2	0.4	0.6	0.8	1.0
Freq. Count of r_k, n_k	**4**	**7**	**2**	**1**	**0**	**1**
Probability $P(r_k) = n_k/n$	4/15	7/15	2/15	1/15	0/15	1/15
$s_k = T(r_k) = \sum_{j=0}^{k} \frac{n_j}{n}$	4/15 **= 0.27**	11/15 **= 0.73**	13/15 **= 0.87**	14/15 **= 0.93**	14/15 **= 0.93**	15/15 **= 1.00**
Quantized s_k	0.2	0.8	0.8	1.0	1.0	1.0

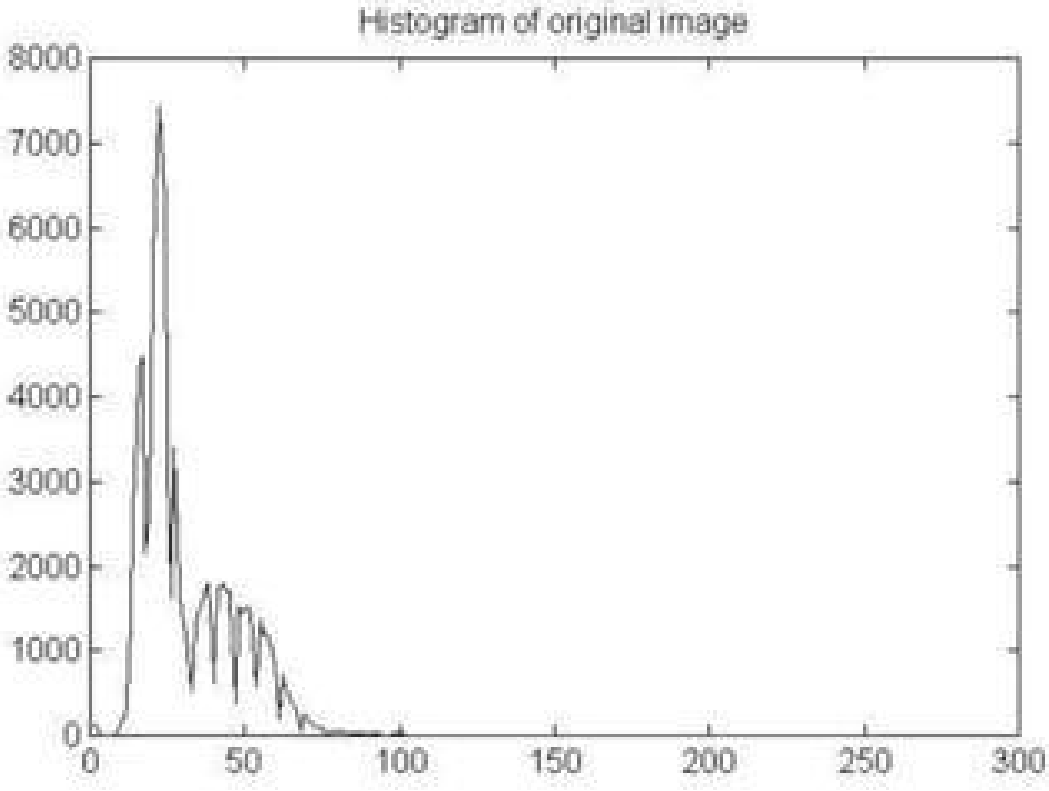
	freq
0.0:	4
0.2:	7
0.4:	2
0.6:	1
0.8:	0
1.0:	1

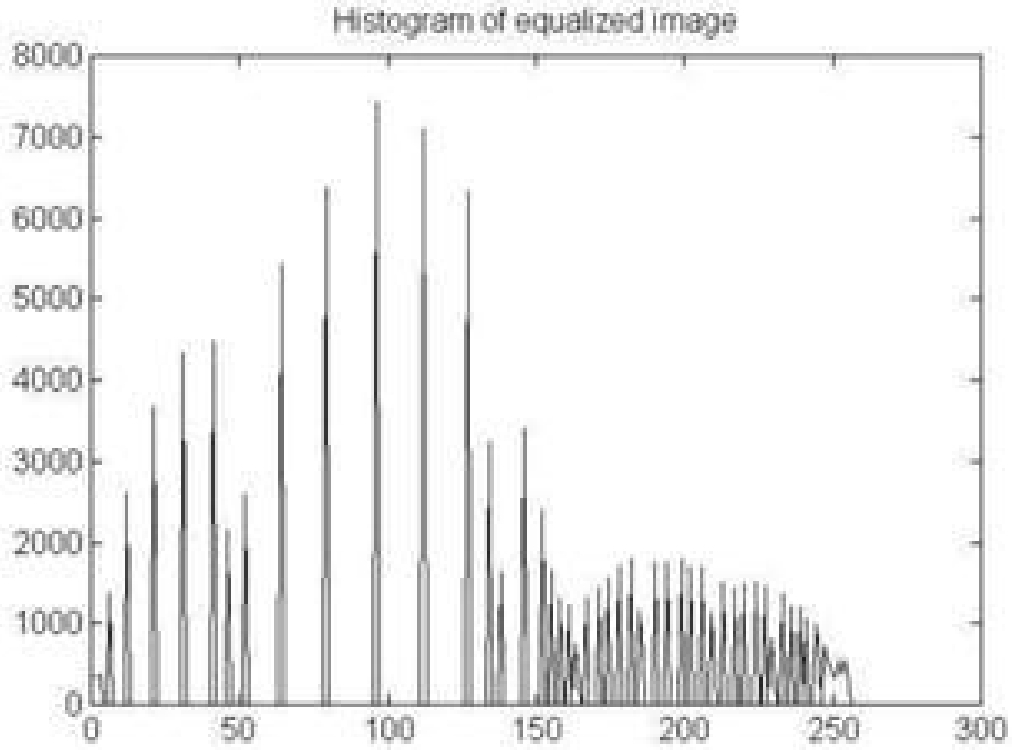
T

r_k		s_k	
0.0	→	0.2	4
0.2	→	0.8	7
0.4	→	0.8	2
0.6	→	1.0	1
0.8	→	1.0	0
1.0	→	1.0	1

	freq
0.0:	0
0.2:	4
0.4:	0
0.6:	0
0.8:	9
1.0:	2




Histogram of original image
8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250
300


Histogram of equalized image
8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250
300

Histogram specification

- Histogram equalization only generates an approximation to a uniform histogram.
- With Histogram Specification, we can specify the shape of the histogram that we wish the output image to have.
- It doesn't have to be a uniform histogram.
- The principal difficulty in applying the histogram specification method to image enhancement lies in being able to construct a meaningful histogram.

Histogram specification cont.

- histogram equalization gives:

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

$$= \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j \qquad 0 \le k \le L-1 \quad (5)$$

- given the value of s_k, we can solve for z_q as:

$$G(z_q) = (L-1)\sum_{i=0}^{q} p_z(z_i) \qquad (6)$$

- for a value of q, so that:

$$G(z_q) = s_k \qquad (7)$$

- we can find z_q as:

$$z_q = G^{-1}(s_k) \qquad (8)$$

- this is the mapping from s to z

Histogram specification cont.

Procedure :

1. compute histogram, $p_r(r)$, of image; find histogram equalization transformation:

$$s_k = T(r_k) = \frac{(L-1)}{M \cdot N} \sum_{j=0}^{k} n_j \quad 0 \le k \le L-1$$

round all values of s_k to nearest integer in $[0, L-1]$

2. compute all values of transformation G

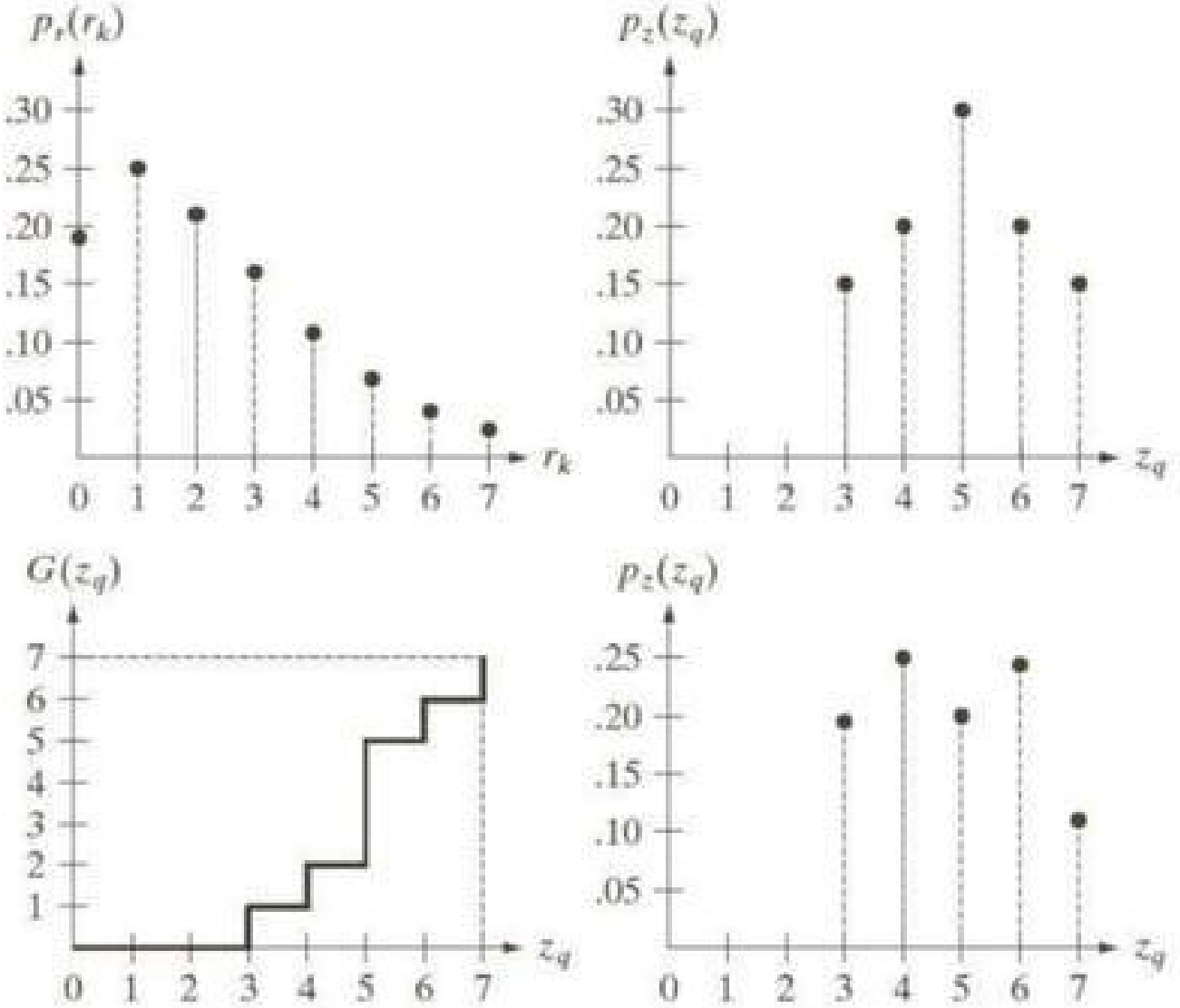
$$G(z_q) = (L-1) \sum_{i=0}^{q} p_z(z_i)$$

for $q = 0, 1, \ldots, L-1$ where $p_z(z_i)$ are values of specified histogram; round values of G to integers in range $[0, L-1]$; store values of G in table

3. for every value of s_k, $0 \le k \le L-1$, use stored values of G to find corresponding value of z_q so that $G(z_q)$ is closest to s_k, and store these mappings from s to z. For multiple z_q matches to s_k, choose smallest z_q.

4. form histogram matched image by first histogram equalizing input image and then mapping every equalized pixel value, s_k, to the corresponding value, z_q, using the mappings in Step 3.

Example



a b
c d

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

Example cont.

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

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

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

- first obtain scaled histogram equalized values:

  $s_0 = 1;\ s_1 = 3;\ s_2 = 5;\ s_3 = 6;\ s_4 = 6;\ s_{5,6,7} = 7$

- next compute and round all values of transformation G:

  $G(z_0) = 0 \rightarrow 0;\ G(z_1) = 0 \rightarrow 0;\ G(z_2) = 0 \rightarrow 0;$

  $G(z_3) = 1.05 \rightarrow 1;\ G(z_4) = 2.45 \rightarrow 2;\ G(z_5) = 4.55 \rightarrow 5;$

  $G(z_6) = 5.95 \rightarrow 6;\ G(z_7) = 7 \rightarrow 7$

- need to find smallest value of z_q so that $G(z_q)$ is closest to s_k; do this for all s_k to create required mapping:

  $s_1 \rightarrow z_3;\ s_3 \rightarrow z_4;\ s_5 \rightarrow z_5;\ s_6 \rightarrow z_6;\ s_7 \rightarrow z_7$

k	s_k	$Q[s_k]$	z_q	$G(z_q)$	$Q[G(z_q)]$
0	1.33	1	0	0	0
1	3.08	3	1	0	0
2	4.55	5	2	0	0
3	5.67	6	3	1.05	1
4	6.23	6	4	2.45	2
5	6.65	7	5	4.55	5
6	6.86	7	6	5.95	6
7	7	7	7	7	7

- Find smallest value of z_q so that $G(z_q)$ is closest to s_k

s_k	$G(z_q)$	z_q
1	1	3
3	2	4
5	5	5
6 6	6	6
7 7 7	7	7

Filtering in the spatial domain (Spatial Filtering)

refers to image operators that change the gray value at any pixel (x,y) depending on the pixel values in a square neighborhood centered at (x,y) using a fixed integer matrix of the same size. The integer matrix is called a *filter*, *mask*, *kernel* or a *window*.

The mechanism of spatial filtering, shown below, consists simply of moving the filter mask from pixel to pixel in an image. At each pixel (x,y), the response of the filter at that pixel is calculated using a predefined relationship (linear or nonlinear).

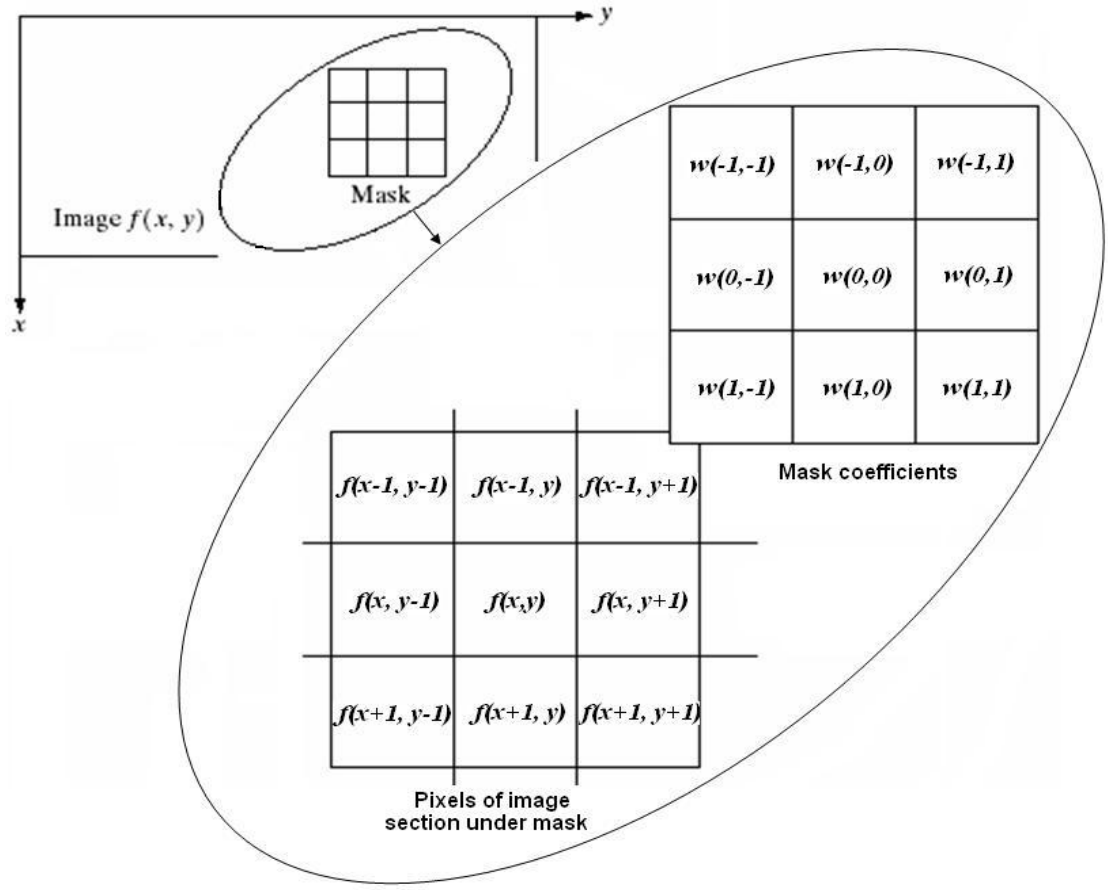


Figure 6.1 Spatial filtering

Note:

The size of mask must be odd (i.e. 3×3, 5×5, etc.) to ensure it has a center. The smallest meaningful size is 3×3.

Linear Spatial Filtering (Convolution)

The process consists of moving the filter mask from pixel to pixel in an image. At each pixel (x,y), the response is given by a sum of products of the filter coefficients and the corresponding image pixels in the area spanned by the filter mask.

For the 3×3 mask shown in the previous figure, the result (or response), R, of linear filtering is:

$$R = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \cdots$$
$$+w(0,0)f(x,y) + \cdots + w(1,0)f(x+1,y) + w(1,1)f(x+1,y+1)$$

In general, linear filtering of an image *f* of size *M*×*N* with a filter mask of size *m*×*n* is given by the expression:

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

where a = (*m* - 1)/2 and b = (*n* - l)/2. To generate a complete filtered image this equation must be applied for x = 0,1, 2,..., *M*-1 and y = 0,1, 2,..., *N*-1.

Nonlinear Spatial Filtering

The operation also consists of moving the filter mask from pixel to pixel in an image. The filtering operation is based conditionally on the values of the pixels in the neighborhood, and they do not explicitly use coefficients in the sum-of-products manner.

For example, noise reduction can be achieved effectively with a nonlinear filter whose basic function is to compute the median gray-level value in the neighborhood in which the filter is located. Computation of the median is a nonlinear operation.

Example:

Use the following 3×3mask to perform the convolution process on the shaded pixels in the 5×5 image below. Write the filtered image.

0	1/6	0
1/6	1/3	1/6
0	1/6	0

3×3 mask

30	40	50	70	90
40	50	80	60	100
35	255	70	0	120
30	45	80	100	130
40	50	90	125	140

5×5 image

Solution:

$$0\times 30+\frac{1}{6}\times 40+0\times 50+\frac{1}{6}\times 40+\frac{1}{3}\times 50+\frac{1}{6}\times 80+0\times 35+\frac{1}{6}\times 255+0\times 70=85$$

$$0\times 40+\frac{1}{6}\times 50+0\times 70+\frac{1}{6}\times 50+\frac{1}{3}\times 80+\frac{1}{6}\times 60+0\times 255+\frac{1}{6}\times 70+0\times 0=65$$

$$0\times 50+\frac{1}{6}\times 70+0\times 90+\frac{1}{6}\times 80+\frac{1}{3}\times 60+\frac{1}{6}\times 100+0\times 70+\frac{1}{6}\times 0+0\times 120=$$

$$0\times 40+\frac{1}{6}\times 50+0\times 80+\frac{1}{6}\times 35+\frac{1}{3}\times 255+\frac{1}{6}\times 70+0\times 30+\frac{1}{6}\times 45+0\times 80=118$$

and so on …

Filtered image =

30	40	50	70	90
40	**85**	**65**	**61**	100
35	**118**	**92**	**58**	120
30	**84**	**77**	**89**	130
40	50	90	125	140

Spatial Filters

Spatial filters can be classified by effect into:

1. **Smoothing Spatial Filters:** also called lowpass filters. They include:
   1.1 Averaging linear filters
   1.2 Order-statistics nonlinear filters.
2. **Sharpening Spatial Filters:** also called highpass filters. For example, the Laplacian linear filter.

Smoothing Spatial Filters

are used for blurring and for noise reduction. Blurring is used in preprocessing steps to:

- remove small details from an image prior to (large) object extraction
- bridge small gaps in lines or curves.

Noise reduction can be accomplished by blurring with a linear filter and also by nonlinear filtering.

Averaging linear filters

The response of averaging filter is simply the average of the pixels contained in the neighborhood of the filter mask.

The output of averaging filters is a smoothed image with reduced "sharp" transitions in gray levels.

Noise and edges consist of sharp transitions in gray levels. Thus smoothing filters are used for noise reduction; however, they have the undesirable side effect that they blur edges.

The figure below shows two 3×3 averaging filters.

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

1	1	1
1	1	1
1	1	1

Standard average filter

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

1	2	1
2	4	2
1	2	1

Weighted average filter

Note:

Weighted average filter has different coefficients to give more importance (weight) to some pixels at the expense of others. The idea behind that is to reduce blurring in the smoothing process.

Averaging linear filtering of an image f of size $M\times N$ with a filter mask of size $m\times n$ is given by the expression:

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

To generate a complete filtered image this equation must be applied for x = 0,1, 2,..., M-1 and y = 0,1, 2,..., N-1.

Figure below shows an example of applying the standard averaging filter.

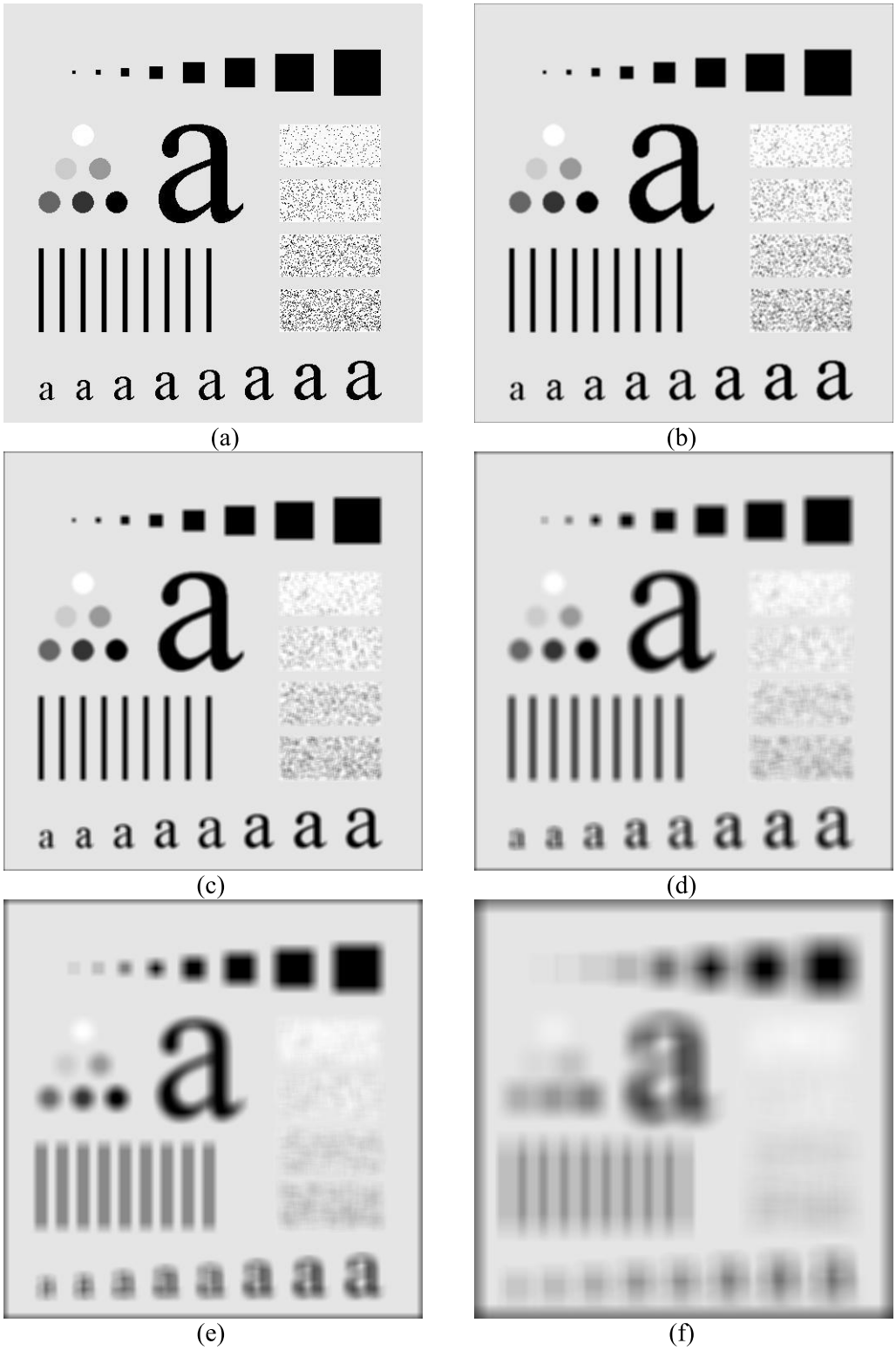


Figure 6.2 Effect of averaging filter. (a) Original image. (b)-(f) Results of smoothing with square averaging filter masks of sizes n = 3,5,9,15, and 35, respectively.

As shown in the figure, the effects of averaging linear filter are:

1. Blurring which is increased whenever the mask size increases.
2. Blending (removing) small objects with the background. The size of the mask establishes the relative size of the blended objects.
3. Black border because of padding the borders of the original image.
4. Reduced image quality.

Order-statistics filters

are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the neighborhood, and then replacing the value of the center pixel with the value determined by the ranking result.
Examples include Max, Min, and Median filters.

Median filter

It replaces the value at the center by the median pixel value in the neighborhood, (i.e. the middle element after they are sorted). Median filters are particularly useful in removing impulse noise (also known as salt-and-pepper noise). Salt = 255, pepper = 0 gray levels.
In a 3×3 neighborhood the median is the 5th largest value, in a 5×5 neighborhood the 13th largest value, and so on.

For example, suppose that a 3×3 neighborhood has gray levels (10, 20, 0, 20, 255, 20, 20, 25, 15). These values are sorted as (0,10,15,20,20,20,20,25,255), which results in a median of 20 that replaces the original pixel value 255 (salt noise).

Example:

Consider the following 5×5 image:

20	30	50	80	100
30	20	80	100	110
25	255	70	0	120
30	30	80	100	130
40	50	90	125	140

Apply a 3×3 median filter on the shaded pixels, and write the filtered image.

Solution

20	30	50	80	100
30	20	80	100	110
25	255	70	0	120
30	30	80	100	130
40	50	90	125	140

Sort:
20, 25, 30, 30, **30**, 70, 80, 80, 255

20	30	50	80	100
30	20	80	100	110
25	255	70	0	120
30	30	80	100	130
40	50	90	125	140

Sort
0, 20, 30, 70, **80**, 80, 100, 100, 255

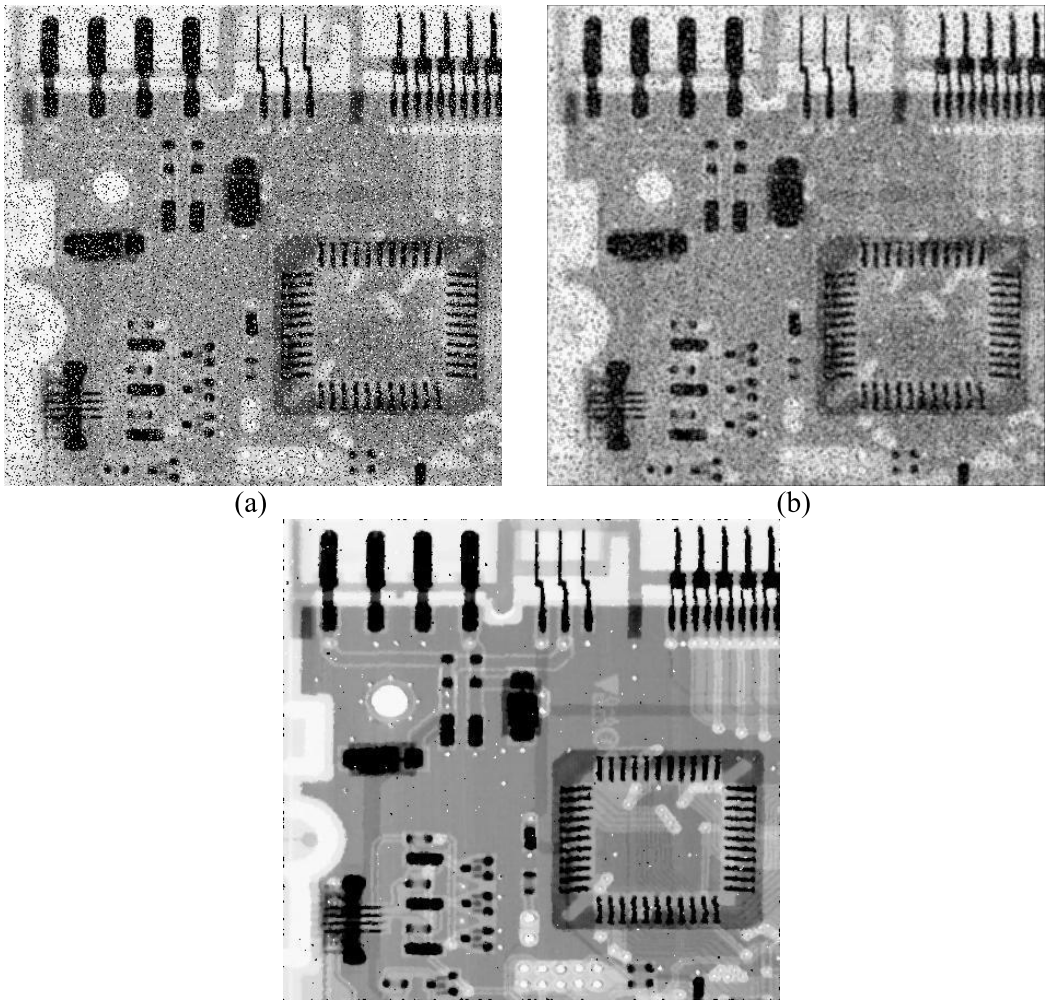
20	30	50	80	100
30	20	80	100	110
25	255	70	0	120
30	30	80	100	130
40	50	90	125	140

Sort
0, 70, 80, 80, **100**, 100, 110, 120, 130

Filtered Image =

20	30	50	80	100
30	20	80	100	110
25	**30**	**80**	**100**	120
30	30	80	100	130
40	50	90	125	140

Figure below shows an example of applying the median filter on an image corrupted with salt-and-pepper noise.



(a) (b)

(c)

Figure 6.3 Effect of median filter. (a) Image corrupted by salt & pepper noise. (b) Result of applying 3×3 standard averaging filter on (a). (c) Result of applying 3×3 median filter on (a).

As shown in the figure, the effects of median filter are:

1. Noise reduction
2. Less blurring than averaging linear filter

Sharpening Spatial Filters

Sharpening aims to highlight fine details (e.g. edges) in an image, or enhance detail that has been blurred through errors or imperfect capturing devices.

Image blurring can be achieved using averaging filters, and hence sharpening can be achieved by operators that invert averaging operators. In mathematics, averaging is equivalent to the concept of integration, and differentiation inverts integration. Thus, sharpening spatial filters can be represented by partial derivatives.

Partial derivatives of digital functions

The first order partial derivatives of the digital image *f(x,y)* are:

$$\frac{\partial f}{\partial x} = f(x+1,y) - f(x,y) \quad \text{and} \quad \frac{\partial f}{\partial y} = f(x,y+1) - f(x,y)$$

The first derivative must be:

1) zero along flat segments (i.e. constant gray values).
2) non-zero at the outset of gray level step or ramp (edges or noise)
3) non-zero along segments of continuing changes (i.e. ramps).

The second order partial derivatives of the digital image *f(x,y)* are:

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

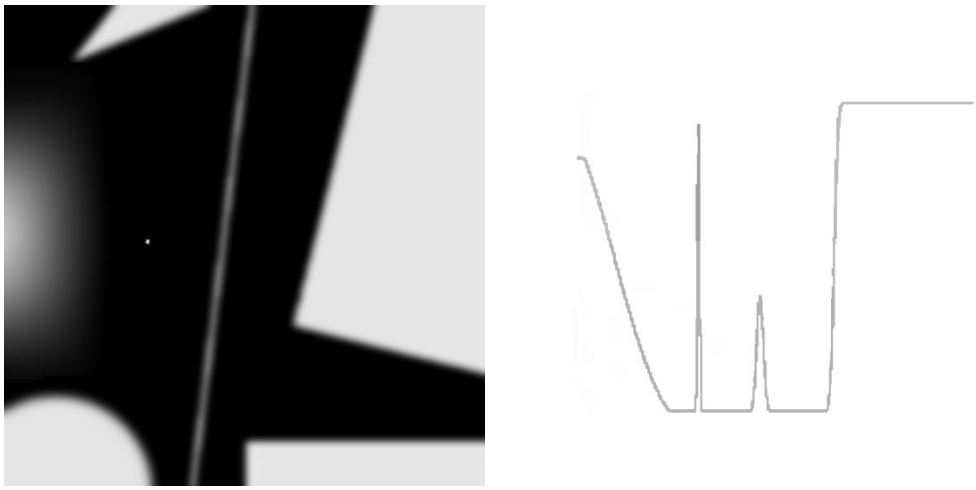
$$\frac{\partial^2 f}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

The second derivative must be:

1) zero along flat segments.
2) nonzero at the outset and end of a gray-level step or ramp;

3) zero along ramps

Consider the example below:



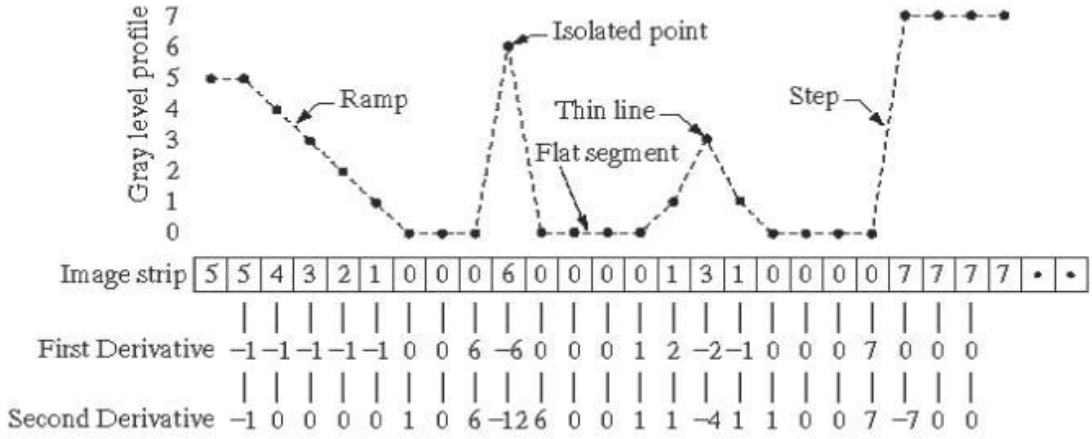


Figure 6.4 Example of partial derivatives

We conclude that:

- 1st derivative detects thick edges while 2nd derivative detects thin edges.
- 2nd derivative has much stronger response at gray-level step than 1st derivative.

Thus, we can expect a second-order derivative to enhance fine detail (thin lines, edges, including noise) much more than a first-order derivative.

The Laplacian Filter

The Laplacian operator of an image *f(x,y)* is:

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

This equation can be implemented using the 3×3 mask:

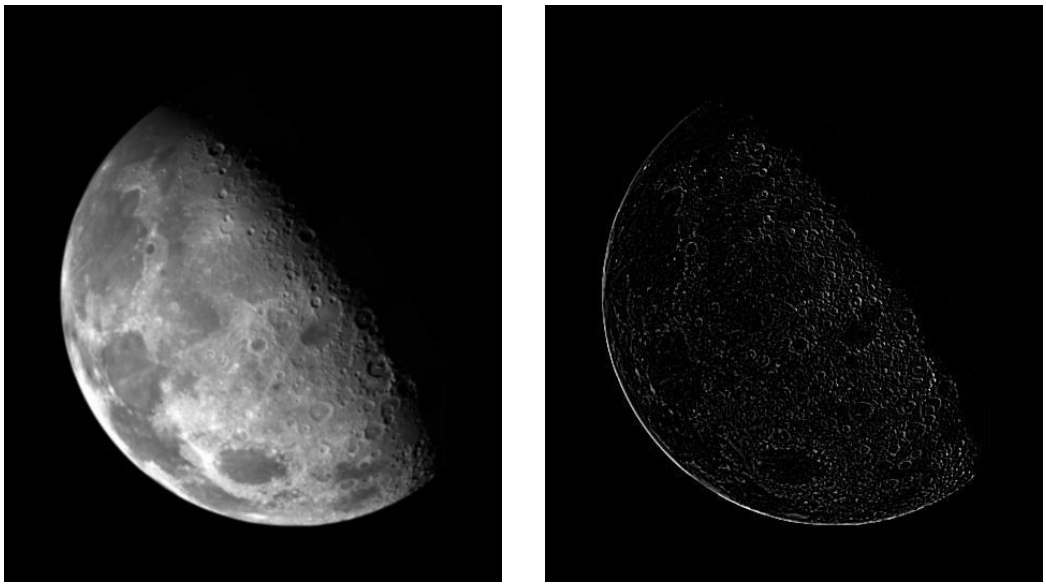
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Since the Laplacian filter is a linear spatial filter, we can apply it using the same mechanism of the convolution process. This will produce a laplacian image that has grayish edge lines and other discontinuities, all superimposed on a dark, featureless background.

Background features can be "recovered" while still preserving the sharpening effect of the Laplacian operation simply by adding the original and Laplacian images.

$$g(x,y) = f(x,y) + \nabla^2 f(x,y)$$

The figure below shows an example of using Laplacian filter to sharpen an image.



(a) (b)

(c)

Figure 6.5 Example of applying Laplacian filter. (a) Original image. (b) Laplacian image. (c) Sharpened image.