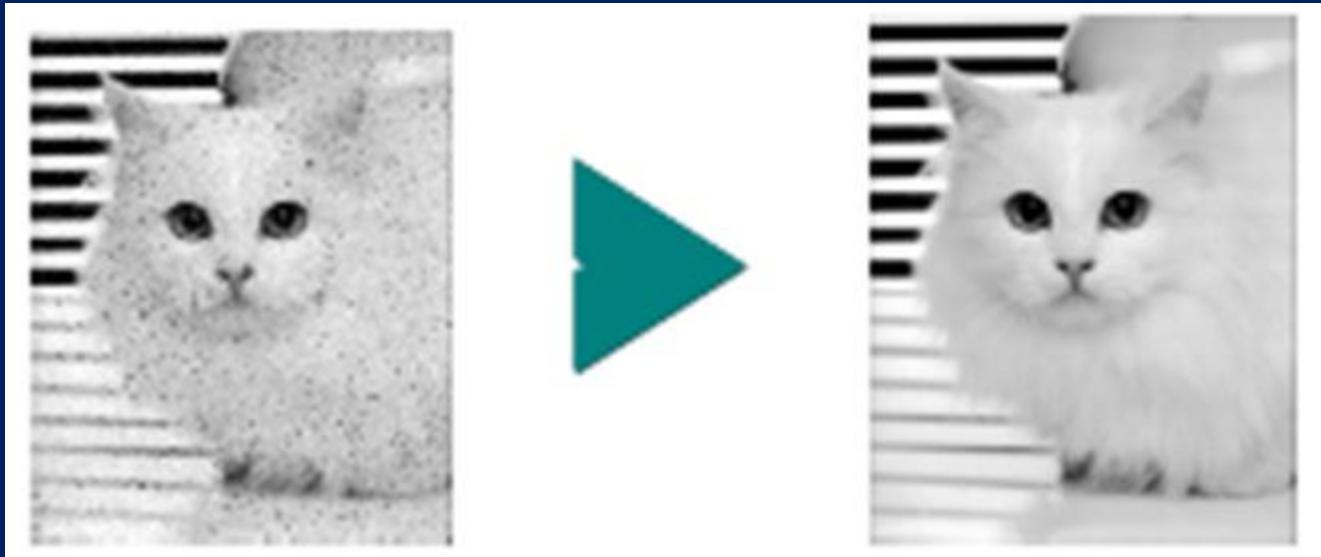




# Image Enhancement in the Spatial Domain



Dr A S Jalal



# Image is NOT Perfect Sometimes



# Why do we need Image Enhancement?

- The goal of image enhancement is to process a digital image and make it more “suitable” than the original image for a specific application
- Images may suffer from the following degradations:
  - Poor contrast due to poor illumination or finite sensitivity of the imaging device
  - Electronic sensor noise or atmospheric disturbances leading to broad band noise
  - Aliasing effects due to inadequate sampling
  - Finite aperture effects or motion

# After Image Enhancement



# Applications of Gray-scale Transformations

- Used to overcome image digitizer limitations before actual processing starts.
- Used to improve the image display process.

## 1. Contrast Enhancement

Features of interest occupy only a relatively narrow range of gray scale

Gray-scale transformation

Expand the contrast of features of interest so that they occupy a relatively larger range of the displayed gray level range

## 2. Photometric Calibration

- Desirable to have gray levels of digitized image to reflect some physical property, light intensity.
- These transformations can do this by removing the effect of image sensor nonlinearity.
- E.g.
  - Suppose the digitizer has a nonlinear response to light intensity.
  - Gray-scale transformation can transform the gray scale so that the gray levels represent equal increments in light intensity.



### 3. Display Calibration

- Some display devices have a preferred range of gray levels over which they make image features most visible.
- Lighter or darker features having the same contrast do not show up well on such a display.
- Gray-scale transformations can ensure that such features of interest fall into the maximum-visibility range of the display.

### 4. Contour Lines

- The transformations can add contour lines to an image.
- Thresholding can be accomplished that divides an image into disjoint regions on the basis of gray levels.

# Image Enhancement

- There are Two broad categories of Image Enhancement Approaches
  - **Spatial domain methods**
    - Based on direct manipulation of pixels in an image
  - **Frequency domain methods**
    - Based on modifying the Fourier transform of an image

# Spatial Domain

- **Intensity Transformations (Point processing)**
  - Operate on single pixels of an image
    - e.g., image averaging; logic operation; contrast stretching ...
- **Spatial Filtering (Mask processing)**
  - Working in a neighbourhood of every pixel in an image
    - e.g., blurring, median

# Good Images

- For human visual
  - The visual evaluation of image quality is a highly subjective process.
  - It is hard to standardize the definition of a good image.
- For machine perception
  - The evaluation task is easier.
  - A good image is one which gives the best machine recognition results.
- A certain amount of trial and error usually is required before a particular image enhancement approach is selected.

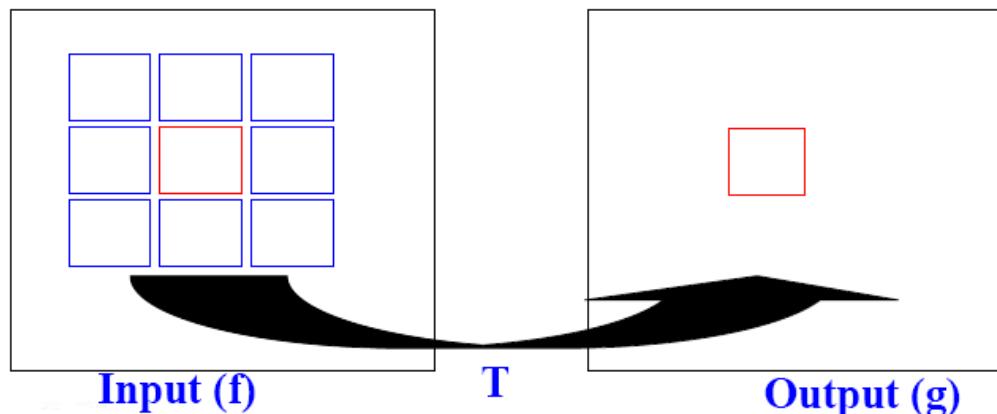
# Spatial Domain

- Spatial domain can be denoted by:

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

where

- $f(x,y)$  is the input image
- $g(x,y)$  is the processed image
- $T$  is an operator on  $f$  defined over some neighborhood of  $(x,y)$





# Intensity Transformations (Point processing)

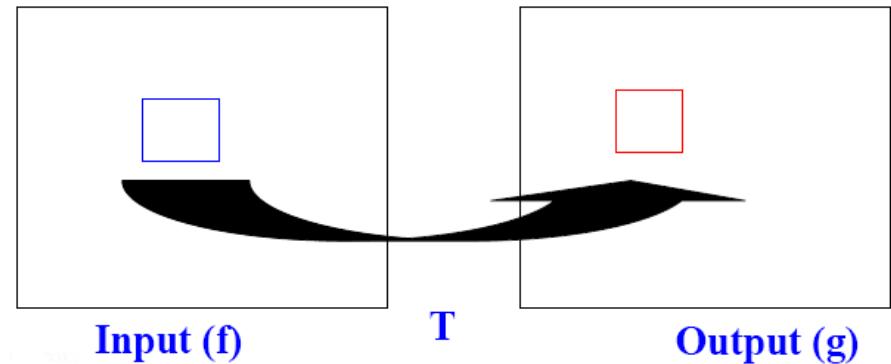
# Intensity Transformation

- Simplest **T** : operates on neighborhood of  $1 \times 1$
- Then, it becomes **intensity transformation**
- $g$  depends on only the value of  $f$  at  $(x,y)$
- $T$  = gray level (or intensity or mapping) transformation function

$$s = T(r)$$

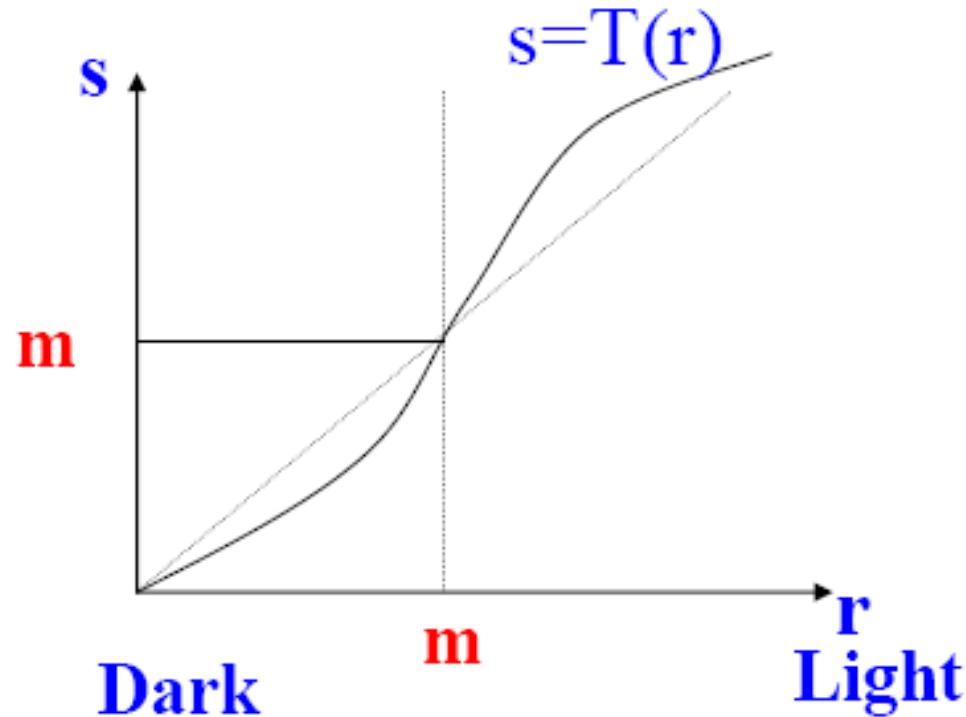
- Where

- $r$  = gray level of  $f(x,y)$
  - $s$  = gray level of  $g(x,y)$



# Intensity Transformation ...

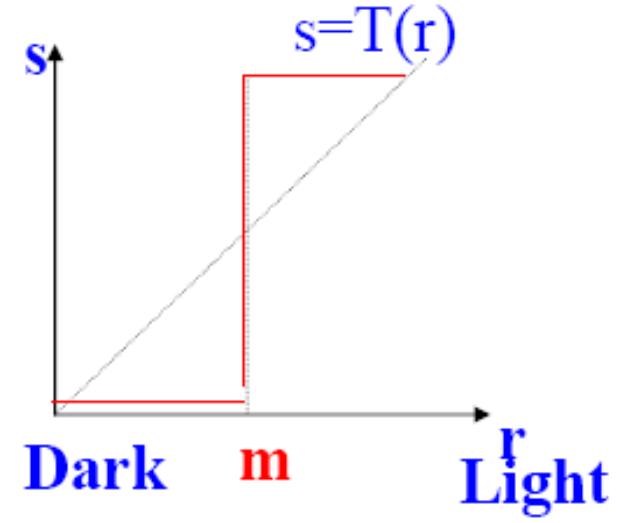
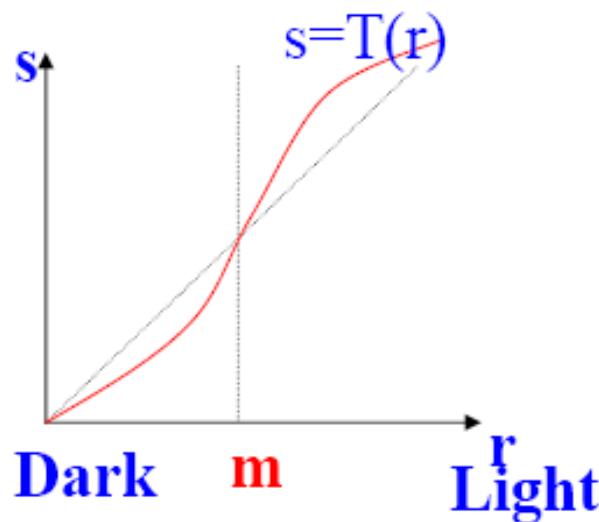
- The gray level below  $m$  are darkened and the levels above  $m$  are brightened.



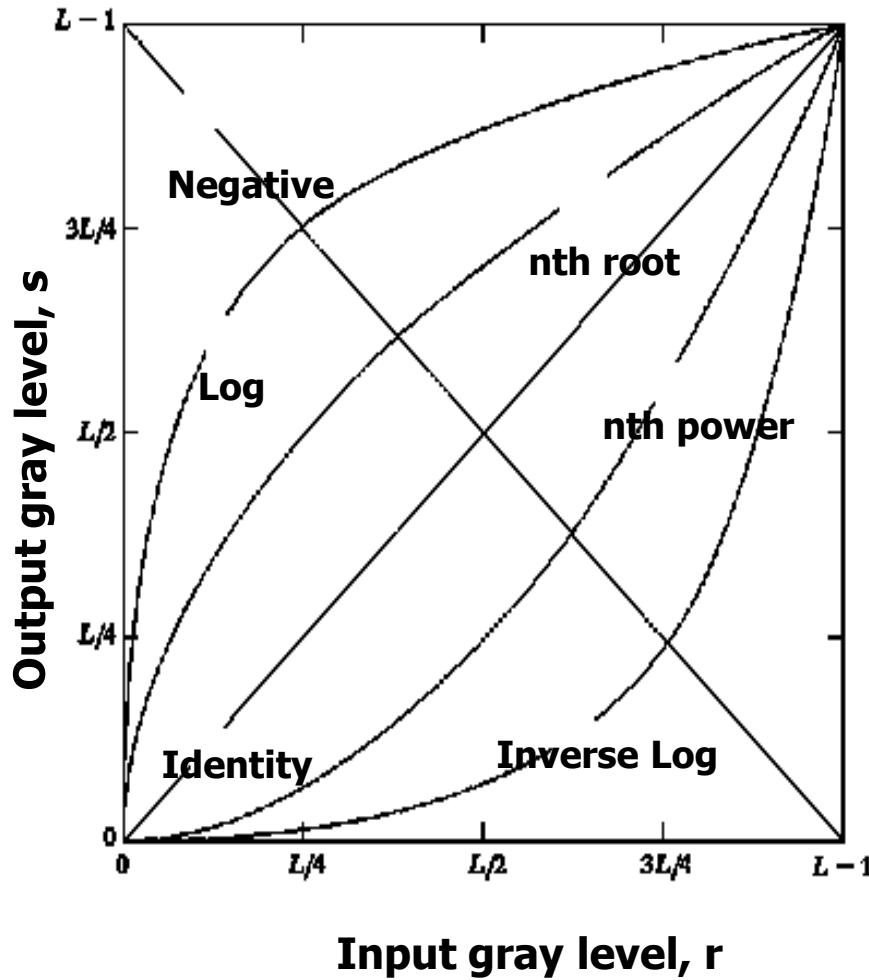
# Intensity Transformation ...

- **Limiting case:** Produces a binary image (two level) from input image and the function is known as Thresholding function

$$g(x,y) = L \quad \text{if } f(x,y) > m, \\ 0 \quad \text{otherwise}$$



# Basic Intensity Transformation Functions



## □ Linear function

- ❑ Negative and identity transformations

## □ Logarithm function

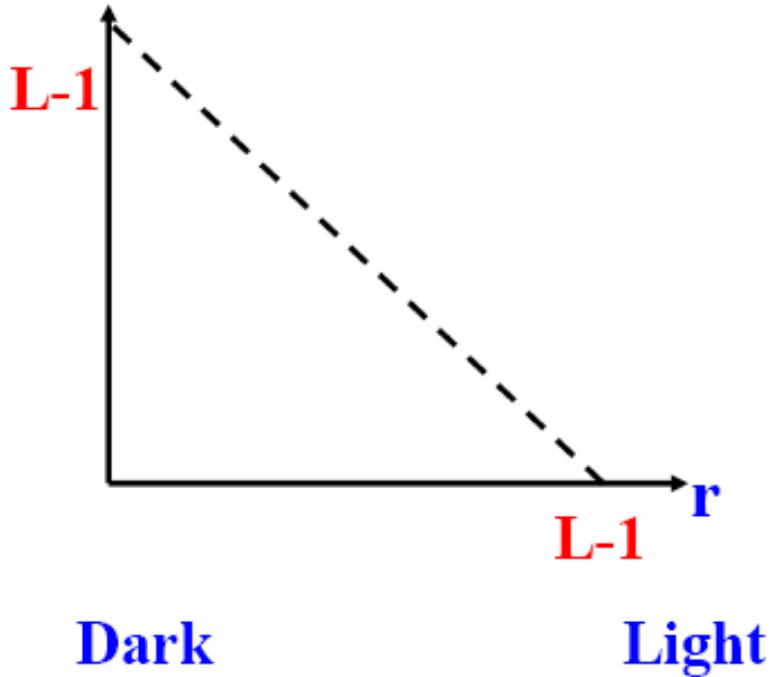
- ❑ Log and inverse-log transformation

## □ Power-law function

- ❑  $n^{\text{th}}$  power and  $n^{\text{th}}$  root transformations

Identity function: Output intensities are identical to input intensities.

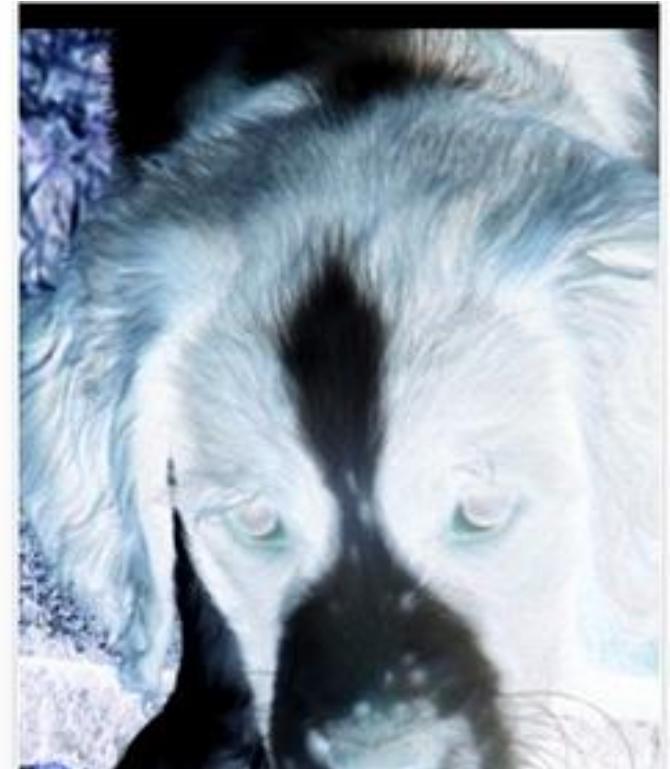
# Image Negatives



- An image with gray level in the range  $[0, L-1]$  where  $L = 2^n$ ;  $n = 1, 2\dots$
- Negative transformation :  
$$s = L - 1 - r$$
- Reversing the intensity levels of an image.
- Suitable for enhancing white or gray detail embedded in dark regions of an image, especially when the black area dominant in size.



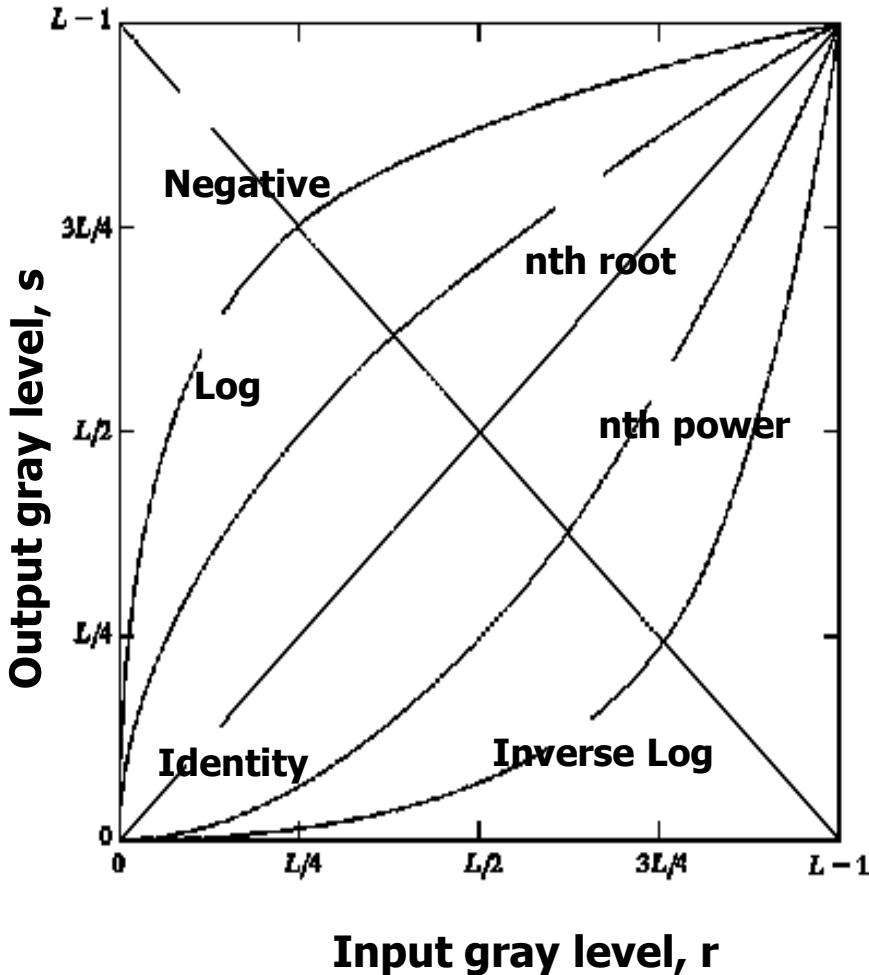
# Image Negatives



# Image Negatives: Example



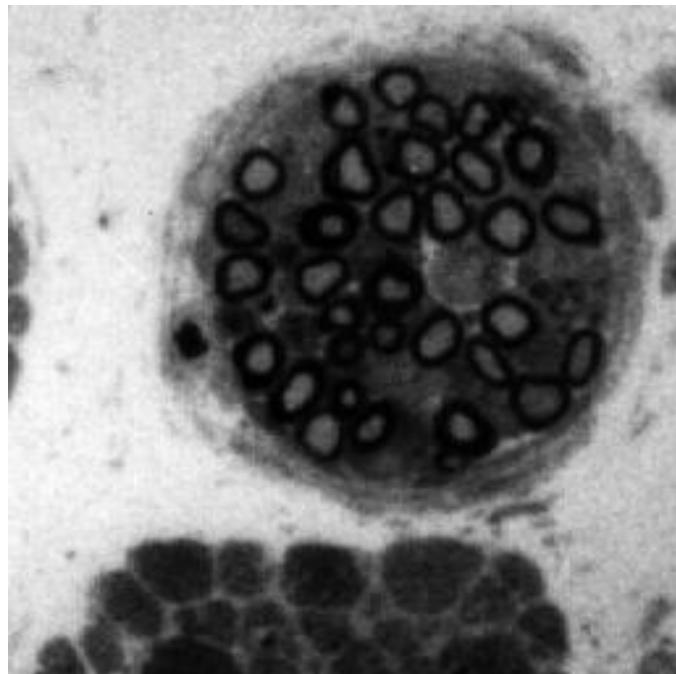
# Log Transformations



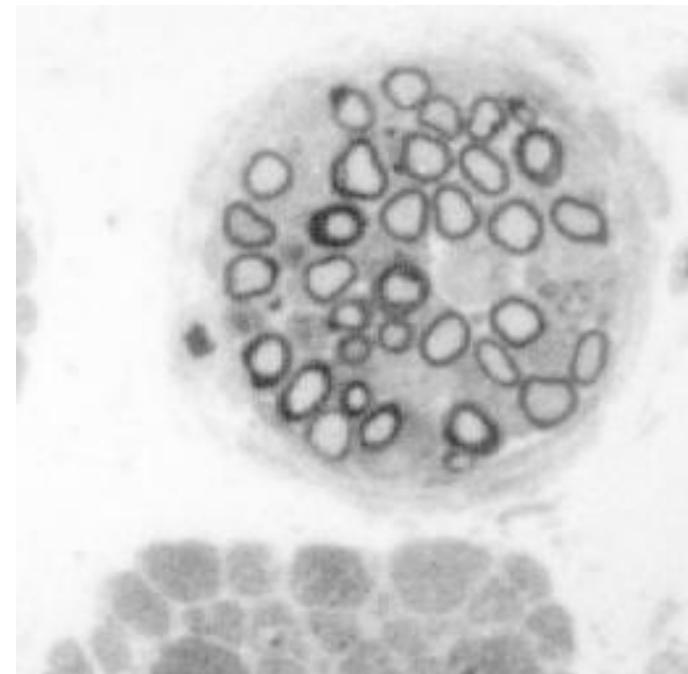
$$s = c \log (1+r)$$

- $c$  is a constant and  $r \geq 0$
- Log curve maps a narrow range of low gray-level values in the input image into a wider range of output levels.
- Used to expand the values of dark pixels in an image while compressing the higher-level values.

# Log Transformations: Example

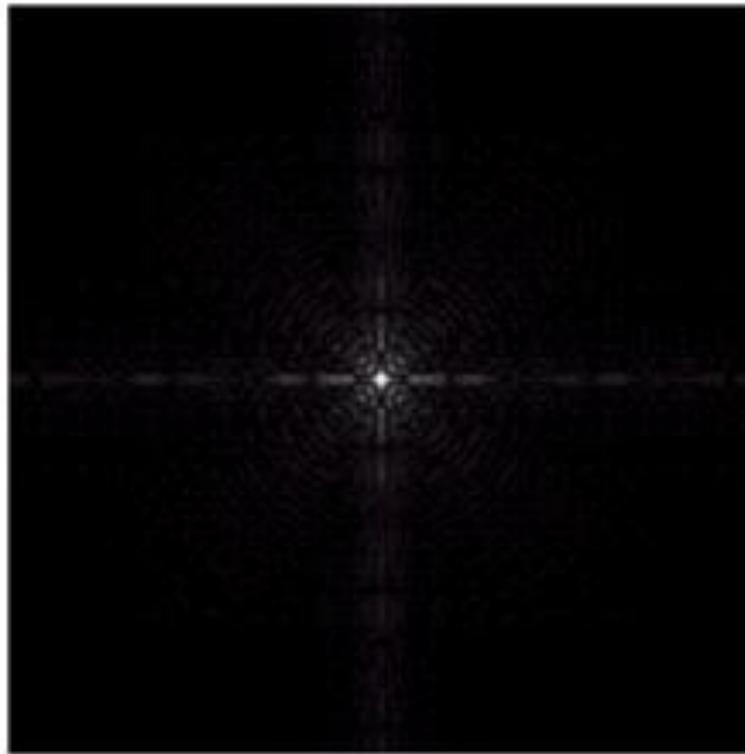


Original Image

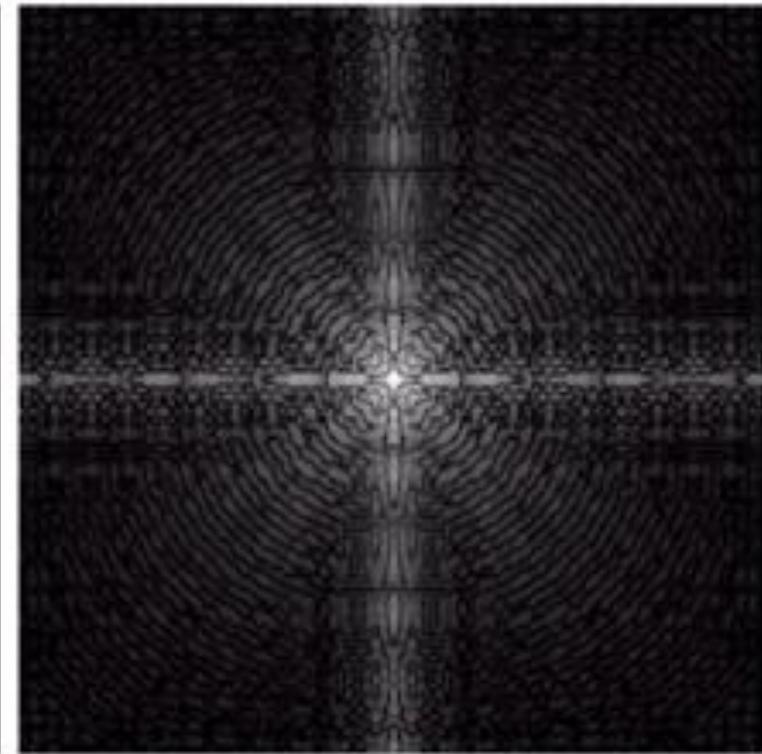


Mapped image

# Example of Logarithm Image

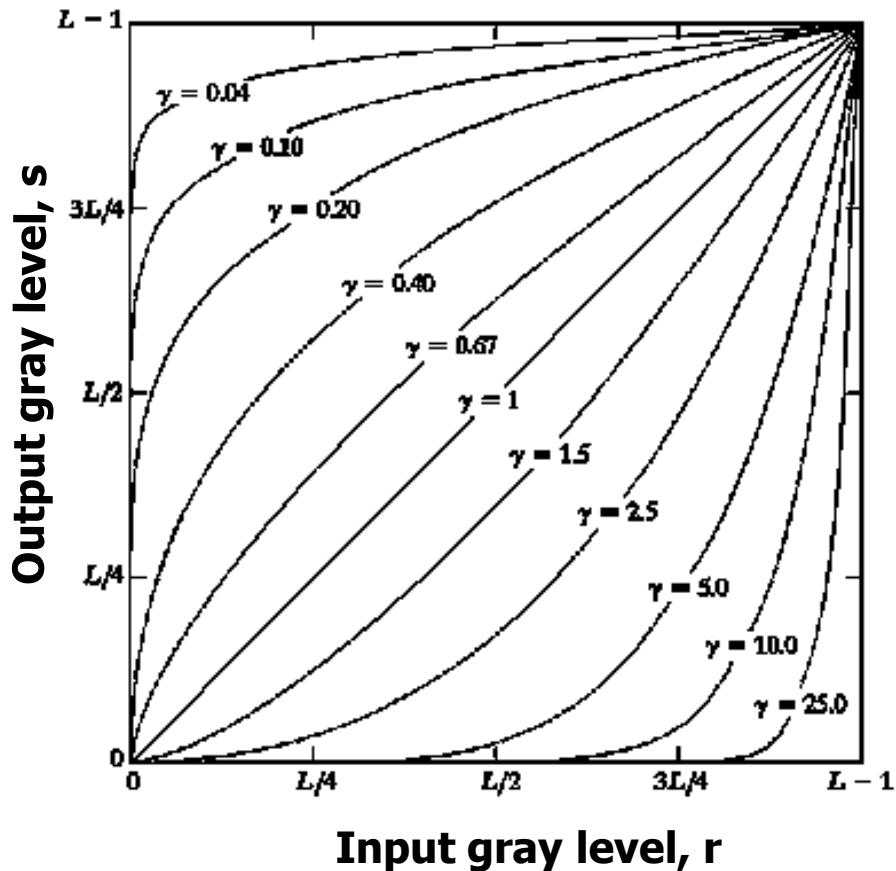


**Fourier Spectrum with  
range = 0 to  $1.5 \times 10^6$**



**Result after apply the log  
transformation with  $c = 1$ ,  
Range = 0 to 6.2**

# Power-Law Transformations

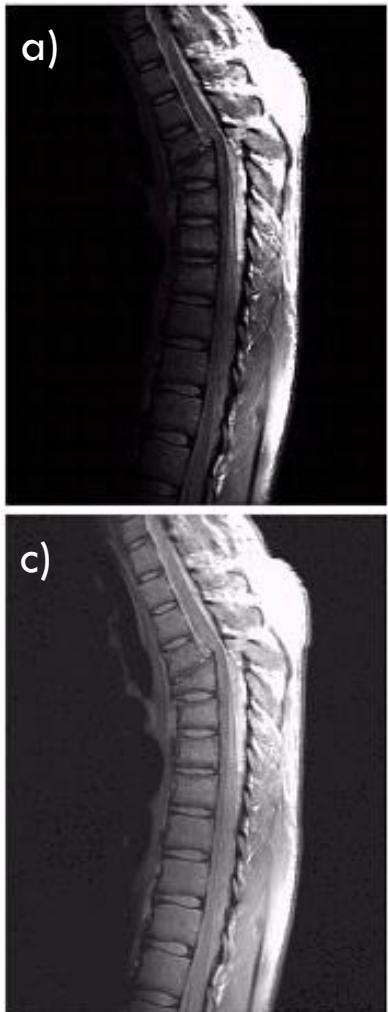


Plots of  $s = cr^\gamma$  for various values of  $\gamma$   
( $c = 1$  in all cases)

$$s = cr^\gamma$$

- $c$  and  $\gamma$  are positive constants
- Power-law curves with fractional values of  $\gamma$  map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher values of input levels.
- $c = \gamma = 1 \Rightarrow$  Identity function

# Example : MRI



(a) a magnetic resonance image of an upper thoracic human spine with a fracture dislocation and spinal cord impingement

- The picture is predominately dark
- An expansion of gray levels are desirable  $\Rightarrow$  needs  $\gamma < 1$

(b) result after power-law transformation with  $\gamma = 0.6$ ,  $c=1$

(c) transformation with  $\gamma = 0.4$   
(best result)

(d) transformation with  $\gamma = 0.3$   
(under acceptable level)

# Effect of decreasing gamma

- When the  $\gamma$  is reduced too much, the image begins to reduce contrast to the point where the image started to have very slight “wash-out” look, especially in the background



- (a) image has a washed-out appearance, it needs a compression of gray levels  $\Rightarrow$  needs  $\gamma > 1$
- (b) result after power-law transformation with  $\gamma = 3.0$  (suitable)
- (c) transformation with  $\gamma = 4.0$  (suitable)
- (d) transformation with  $\gamma = 5.0$  (high contrast, the image has areas that are too dark, some detail is lost)

# Piecewise-Linear Transformation Functions

- A complementary approach to the previous methods
- The form of piecewise functions can be arbitrarily complex
- Practical implementation of some important transformations can be formulated only as piecewise functions
- Their specification requires considerably more user input

# Contrast Stretching

- The Simplest of piecewise linear functions.
- **Basic idea** – to increase the dynamic range of the gray levels in the image being processed.
- During image acquisition, low contrast images may result due to
  - Poor illumination
  - Lack of dynamic range in image sensor
  - Wrong setting of the lens aperture

# Contrast Stretching

- Contrast transform result



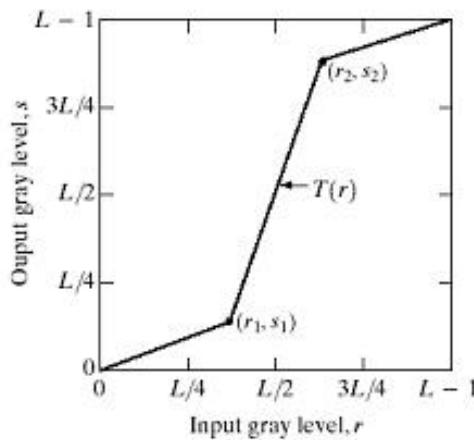
**Original**

Class Presentations on Digital Image Processing by Dr. Anand Singh Jalal



**Enhanced**

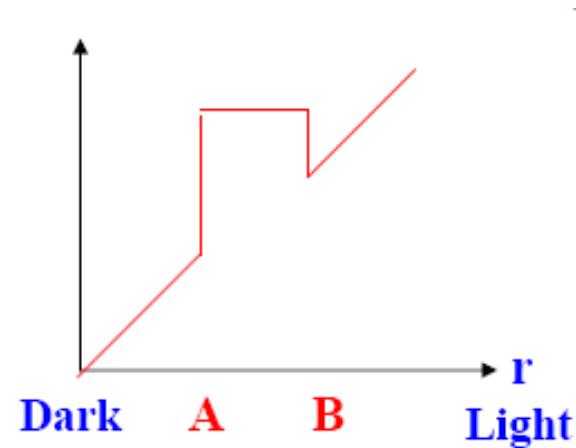
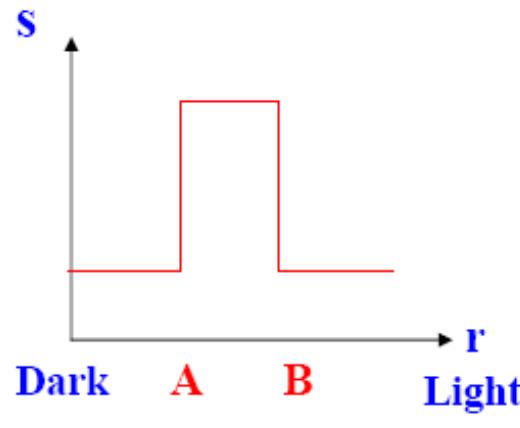
# Contrast Stretching: Example



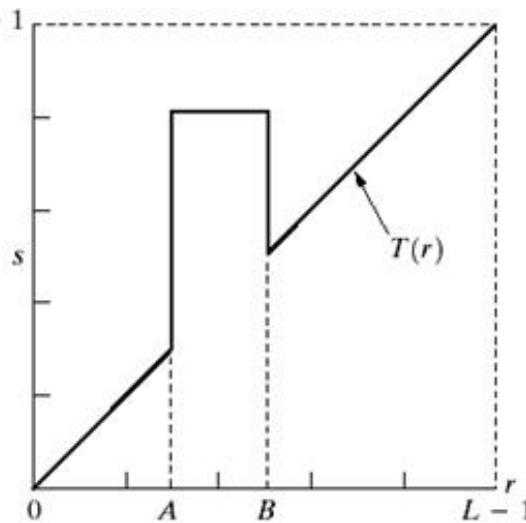
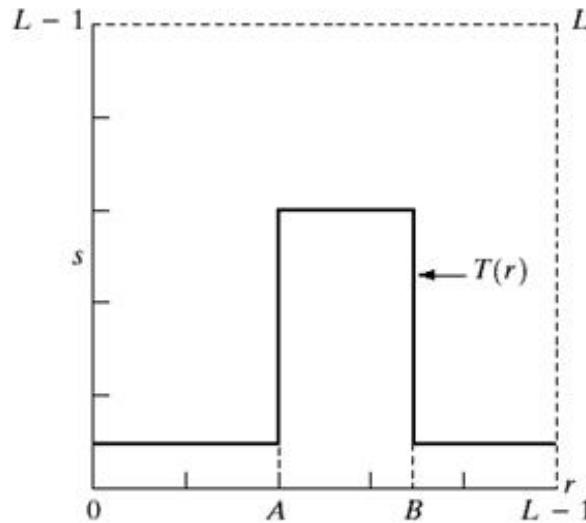
- increase the dynamic range of the gray levels in the image
- (b) a low-contrast image
- (c) result of contrast stretching:  $(r_1, s_1) = (r_{\min}, 0)$  and  $(r_2, s_2) = (r_{\max}, L-1)$
- (d) result of thresholding

# Gray-level slicing

- Highlight a specific range of gray values
- Two basic Methods
  - Display a high value for all gray levels in the range of interest and a low value for all other
  - Brighten the desired range of gray levels but preserve all other levels

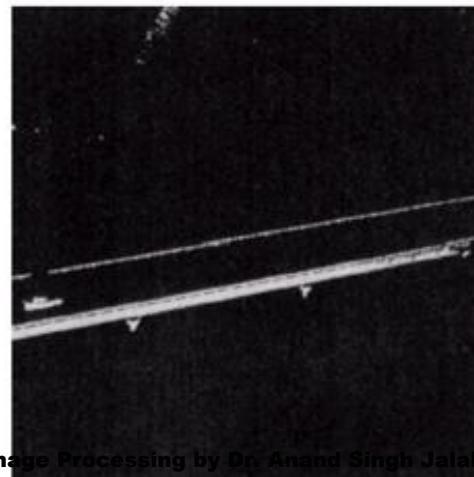


# Gray-level slicing: Example

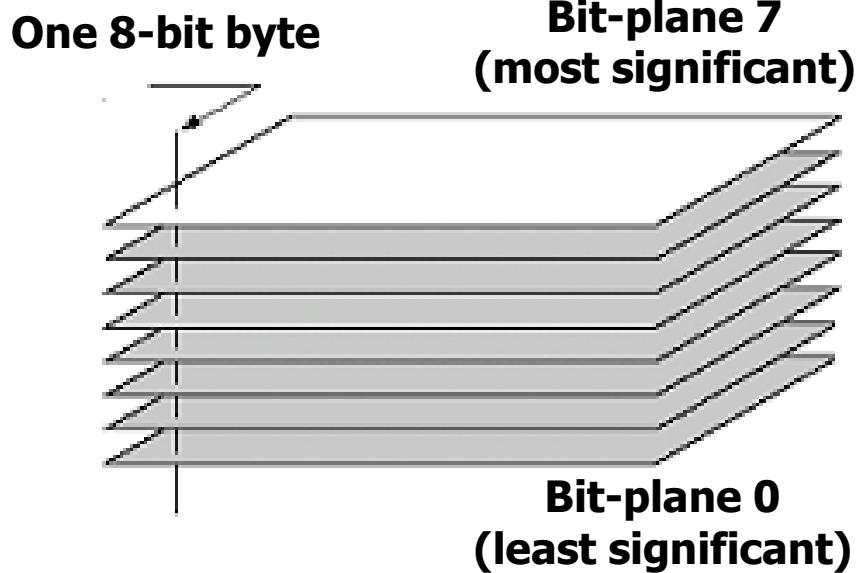


a	b
c	d

- (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).

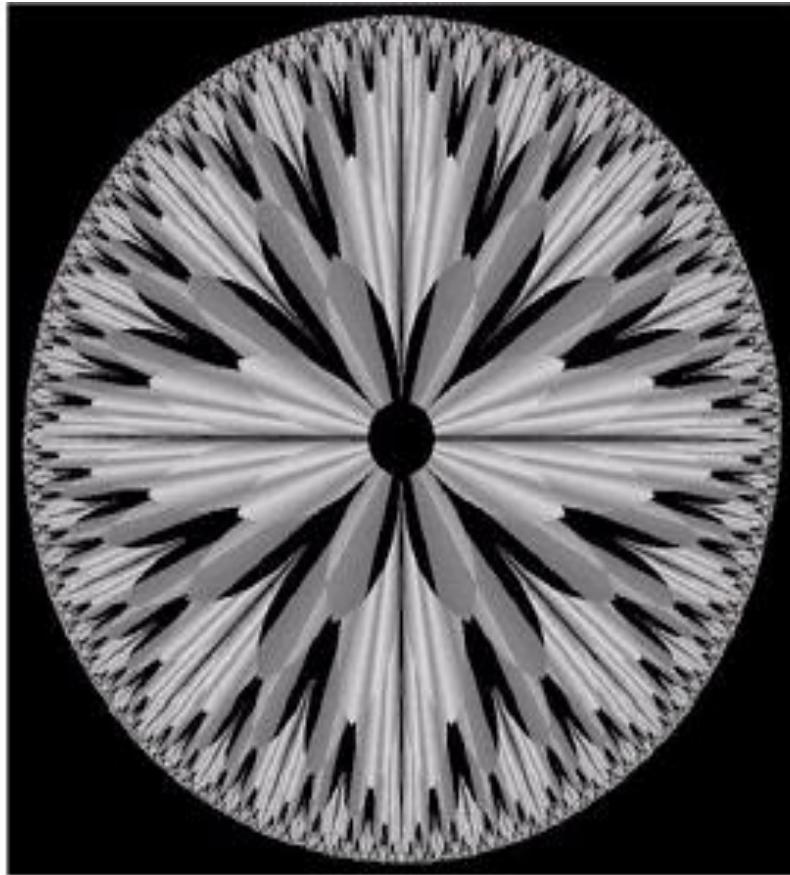


# Bit-plane slicing



- Highlighting the contribution made to total image appearance by specific bits
- Suppose each pixel is represented by 8 bits
- Higher-order bits contain the majority of the visually significant data
- Useful for analyzing the relative importance played by each bit of the image

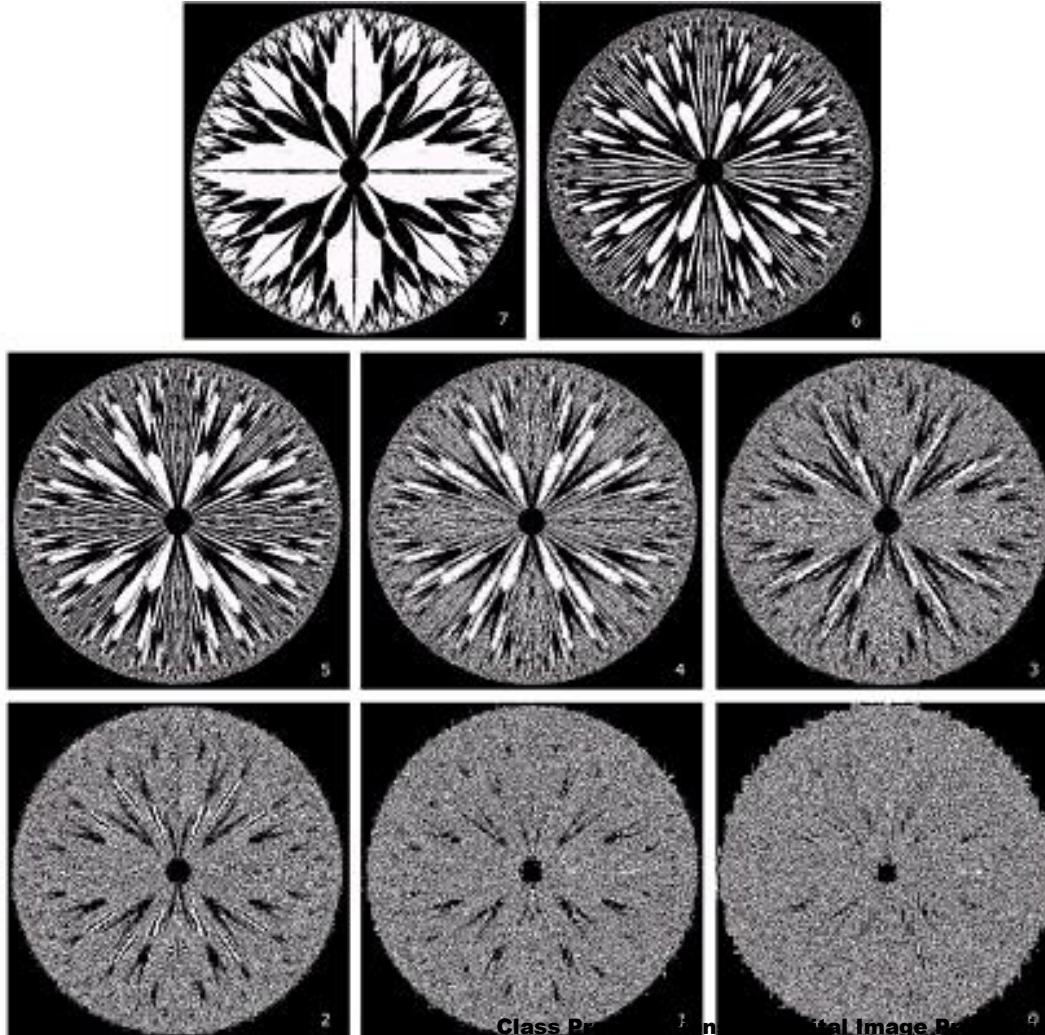
# Bit-plane slicing: Example



An 8-bit fractal image

- Higher order bit planes of an image carry a significant amount of visually relevant details.
- Lower order planes contribute more to fine (often imperceptible) details.
- 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

# Bit-plane slicing: Example (8 bit planes)



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

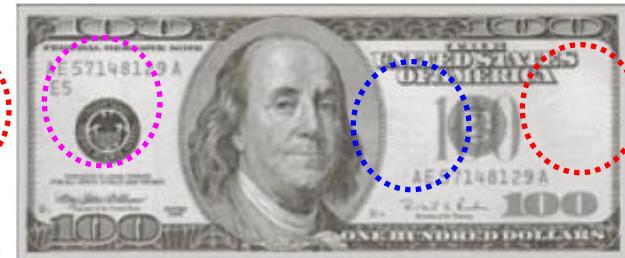
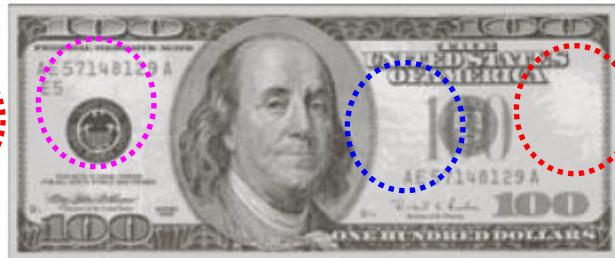
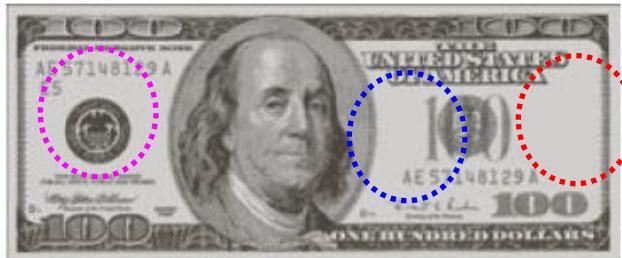
# Bit-plane slicing: Example



a	b	c
d	e	f
g	h	i

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

# Bit-plane slicing: Example



a | b | c

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).

# Histogram Processing

- Histogram is a discrete function formed by counting the number of pixels that have a certain gray level in the image .
- Histogram of a digital image with gray levels in the range [0,L-1] is a discrete function

$$h(r_k) = n_k$$

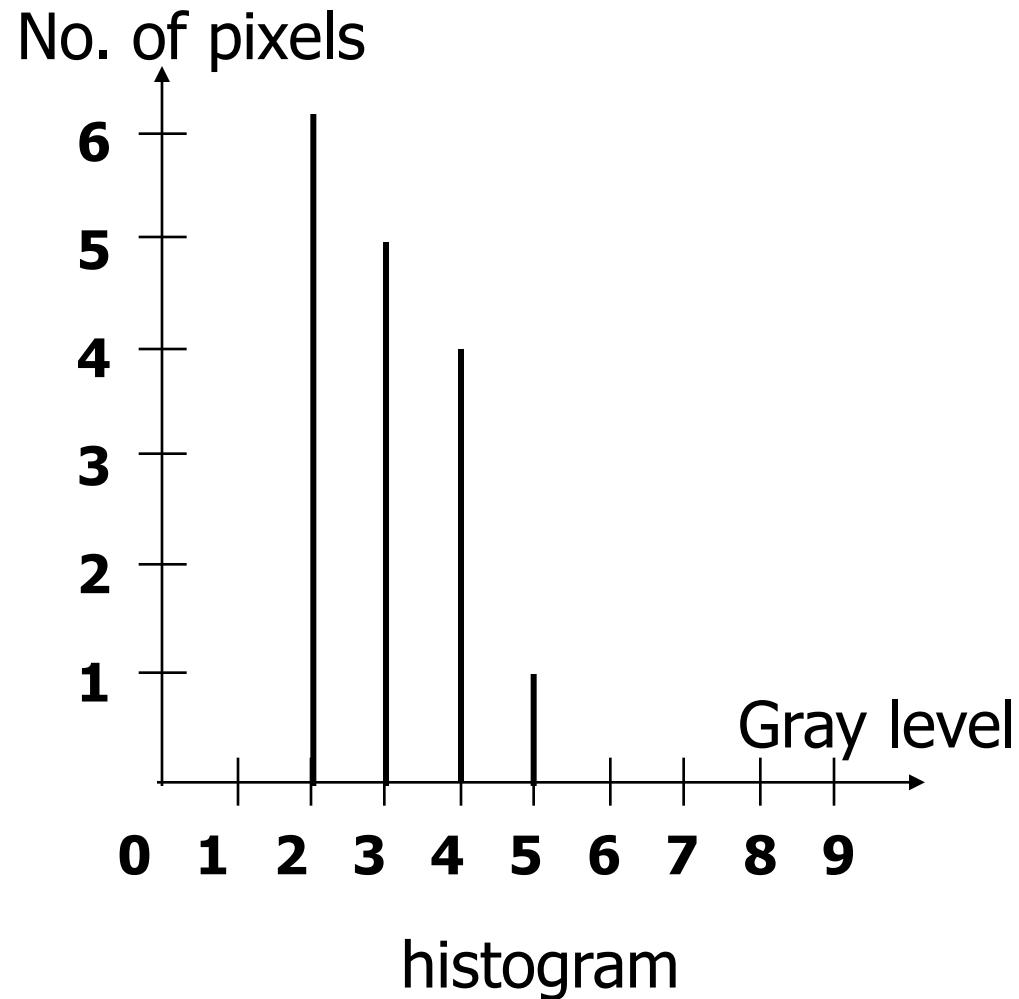
- Where
  - $r_k$  : the k<sup>th</sup> gray level
  - $n_k$  : the number of pixels in the image having gray level  $r_k$
  - $h(r_k)$  : histogram of a digital image with gray levels  $r_k$

# Histogram Processing: Example

2	3	3	2
4	2	4	3
3	2	3	5
2	4	2	4

4x4 image

Gray scale = [0,9]



# Normalized Histogram

- Dividing each of histogram at gray level  $r_k$  by the total number of pixels in the image,  $n$

$$p(r_k) = n_k / n$$

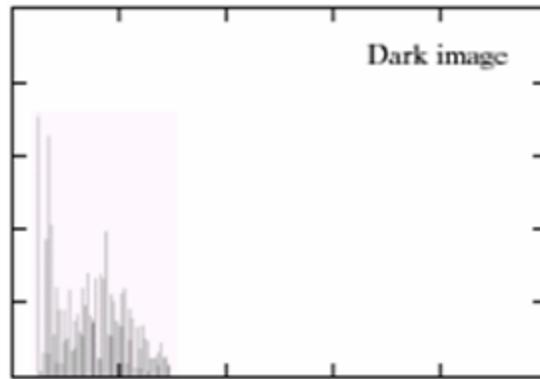
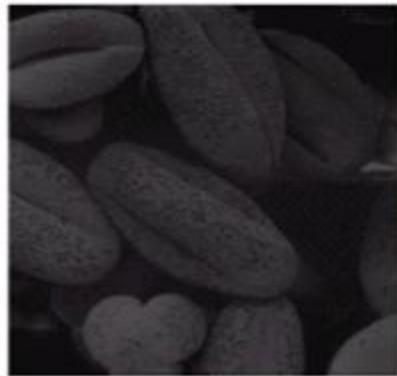
- For  $k = 0, 1, \dots, L-1$
- $p(r_k)$  gives an estimate of the probability of occurrence of gray level  $r_k$
- The sum of all components of a normalized histogram is equal to 1

# Histogram Processing: Example

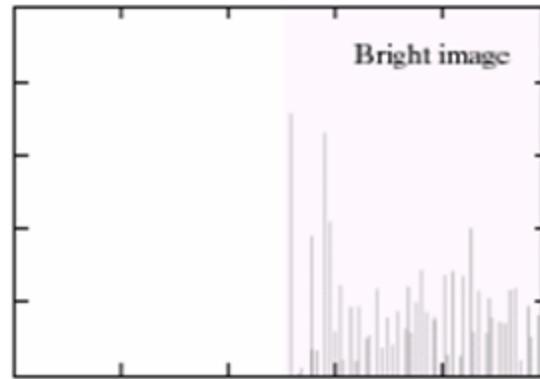
- Problem: an image with gray levels between 0 and 7 is given below. Find the histogram of the image

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

# Histogram Processing: Example

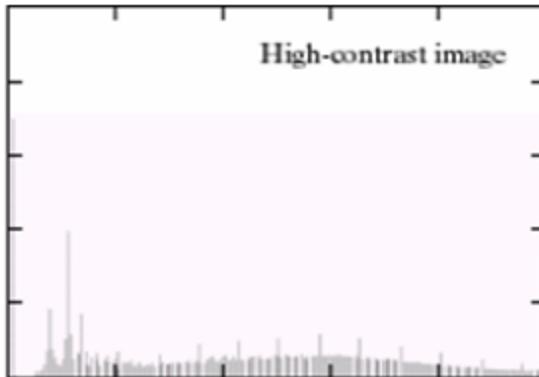
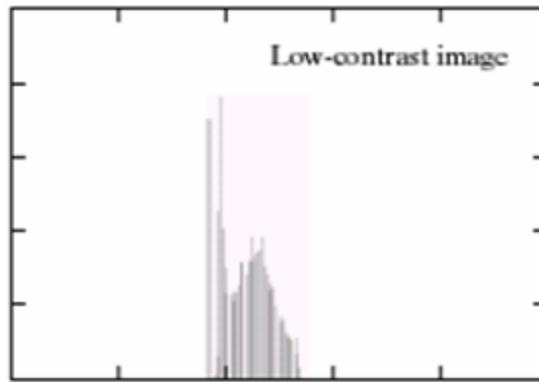
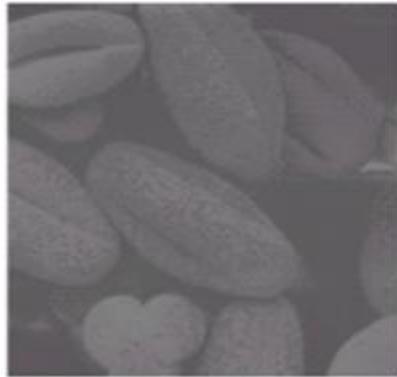


- In the **dark images**, components of the histogram are concentrated on the low (dark) side of the gray scale.



- In **bright images**, the histogram is biased towards the high side of the gray scale.

# Histogram Processing: Example



- In image with **low contrast**, the histogram will be narrow & centred towards the middle of the gray scale.
- An image with **high contrast** & a large variety of gray tones will have pixels that occupy the entire range of possible gray levels & are uniformly distributed.

# Comparing Two Histograms

The ability to compare two histograms in terms of some specific criteria for similarity.

## 1. Correlation

$$d_{\text{correl}}(H_1, H_2) = \frac{\sum_i H'_1(i) \cdot H'_2(i)}{\sqrt{\sum_i H'^2_1(i) \cdot H'^2_2(i)}}$$

Average no. of pixels  
per intensity level

where

$$H'_k(i) = H_k(i) - (1/N) \left( \sum_j H_k(j) \right)$$

and N equals the number of bins in the histogram.

For correlation, a high score represents a better match than a low score. A perfect match is 1 and a maximal mismatch is -1; a value of 0 indicates no correlation (random association).

## 2. Chi-square

$$d_{\text{chi-square}}(H_1, H_2) = \sum_i \frac{(H_1(i) - H_2(i))^2}{H_1(i) + H_2(i)}$$

For chi-square, a low score represents a better match than a high score. A perfect match is 0 and a total mismatch is unbounded (depending on the size of the histogram).

## 3. Intersection

$$d_{\text{intersection}}(H_1, H_2) = \sum_i \min(H_1(i), H_2(i))$$

For histogram intersection, high scores indicate good matches and low scores indicate bad matches. If both histograms are normalized to 1, then a perfect match is 1 and a total mismatch is 0.

## 4. Bhattacharyya distance

$$d_{\text{Bhattacharyya}}(H_1, H_2) = \sqrt{1 - \sum_i \frac{\sqrt{H_1(i) \cdot H_2(i)}}{\sqrt{\sum_i H_1(i) \cdot \sum_i H_2(i)}}}$$

For Bhattacharyya matching , low scores indicate good matches and high scores indicate bad matches. A perfect match is 0 and a total mismatch is a 1.

Normalize histograms before comparing!!!

# Histogram Equalization OR Histogram Linearization

- It is the process that transforms the intensity values so that the histogram of the output image approximately matches the flat (uniform) histogram.
- The aim is to create an image with equally distributed brightness levels over the whole brightness scale.

# Histogram Equalization OR Histogram Linearization

- As the low-contrast image's histogram is narrow and centered toward the middle of the gray scale, if we distribute the histogram to a wider range the quality of the image will be improved.
- We can do it by adjusting the probability density function of the original histogram of the image so that the probability spread equally.
- Histogram equalization (HE) results are similar to contrast stretching but offer the **advantage of full automation**, since HE automatically determines a transformation function to produce a new image with a uniform histogram.

# Histogram Equalization

- **Goal:** find a transform  $s=T(r)$  such that the transformed image has a **flat (equalized) histogram**.
- Where  $T(r)$  satisfies the following conditions:
  - $T(r)$  is single-valued and monotonically increasing in interval  $[0,1]$ ;
  - $0 \leq T(r) \leq 1$  for  $0 \leq r \leq 1$ .

# Histogram Equalization ...

For discrete values, the probability of occurrence of gray level  $r_k$  in an image is given by:

$$P_r(r_k) = n_k / n$$

where  $k = 0, 1, \dots, L-1$ ,

- $n$  is the total no. of pixels in the image
- $n_k$  is the no. of pixels with gray level  $r_k$
- $L$  is the total no. of possible gray levels in the image

# Histogram Equalization ...

The discrete transformation function is given by:

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

$$= \sum_{j=0}^k \frac{n_j}{n} \quad k = 0, 1, \dots, L - 1$$

The o/p image is obtained by mapping each pixel with level  $r_k$  in the i/p image into a corresponding pixel with level  $s_k$  in the o/p image with the help of the above equation.

# Histogram Equalization ...

The following equations bring back the gray levels in the range [0, L-1]

Discrete values:

$$\begin{aligned}s_k &= T(r_k) = (L-1) \sum_{j=0}^k p_r(r_j) \\&= (L-1) \sum_{j=0}^k \frac{n_j}{MN} = \frac{L-1}{MN} \sum_{j=0}^k n_j \quad k=0,1,\dots, L-1\end{aligned}$$

M – no. of rows

N – no. of columns

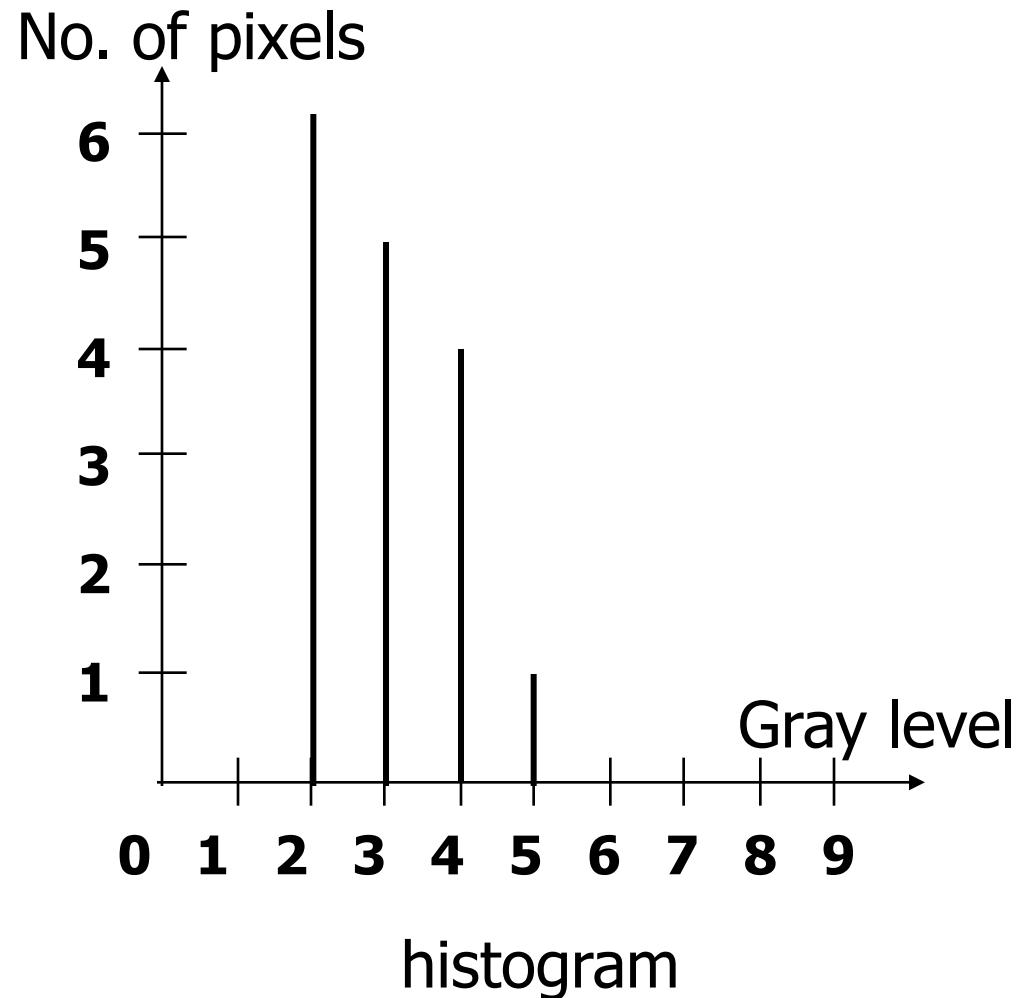
MN – total no. of pixels in the image

# Histogram Equalization: Example 1

2	3	3	2
4	2	4	3
3	2	3	5
2	4	2	4

4x4 image

Gray scale = [0,9]



# Histogram Equalization: Example 1...

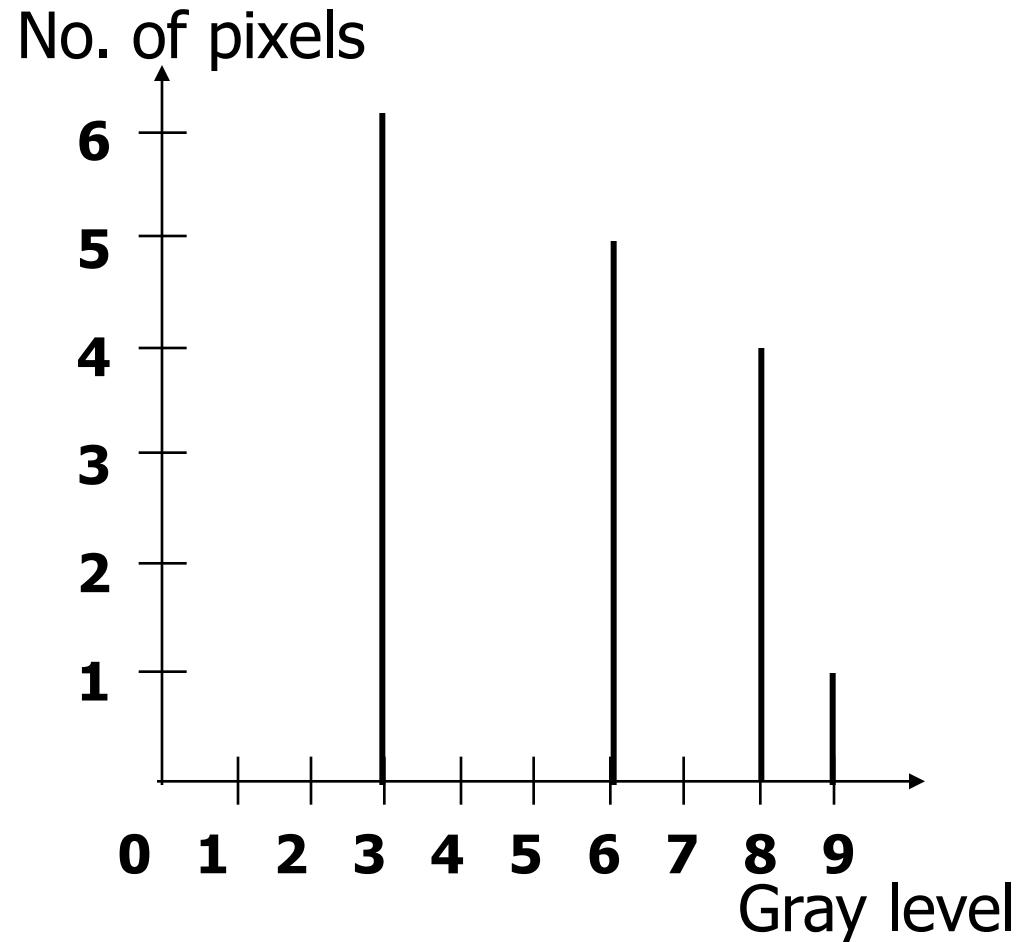
Gray Level(j)	0	1	2	3	4	5	6	7	8	9
No. of pixels	0	0	6	5	4	1	0	0	0	0
$\sum_{j=0}^k n_j$	0	0	6	11	15	16	16	16	16	16
$s = \sum_{j=0}^k \frac{n_j}{n}$	0	0	6 / 16	11 / 16	15 / 16	16 / 16	16 / 16	16 / 16	16 / 16	16 / 16
$s \times 9$	0	0	3.3 ≈ 3	6.1 ≈ 6	8.4 ≈ 8	9	9	9	9	9

# Histogram Equalization: Example 1...

3	6	6	3
8	3	8	6
6	3	6	9
3	8	3	8

Output image

Gray scale = [0,9]



# Histogram Equalization: Example 2

Suppose that a 3-bit image ( $L=8$ ) of size  $64 \times 64$  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

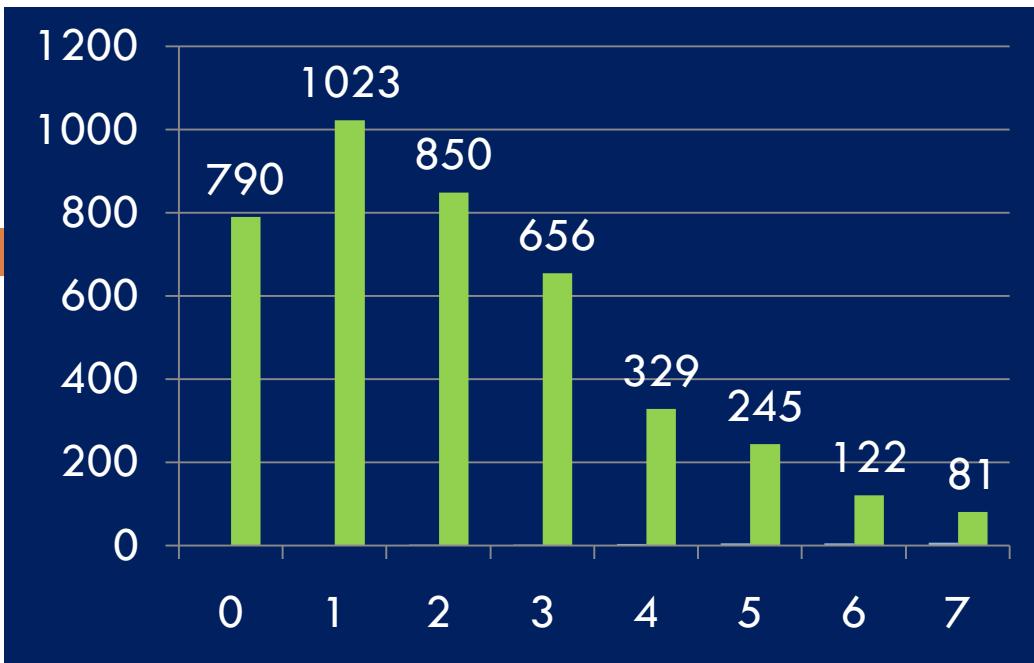
# Histogram Equalization: Example 2 ...

I/p Gray Level (r <sub>k</sub> )	no. of pixels (n <sub>k</sub> )	p(r <sub>k</sub> ) = n <sub>k</sub> /MN	$\Sigma$	(L-1) $\Sigma$	O/p Gray Level (s)
0	790	0.19			
1	1023	0.25			
2	850	0.21			
3	656	0.16			
4	329	0.08			
5	245	0.06			
6	122	0.03			
7	81	0.02			

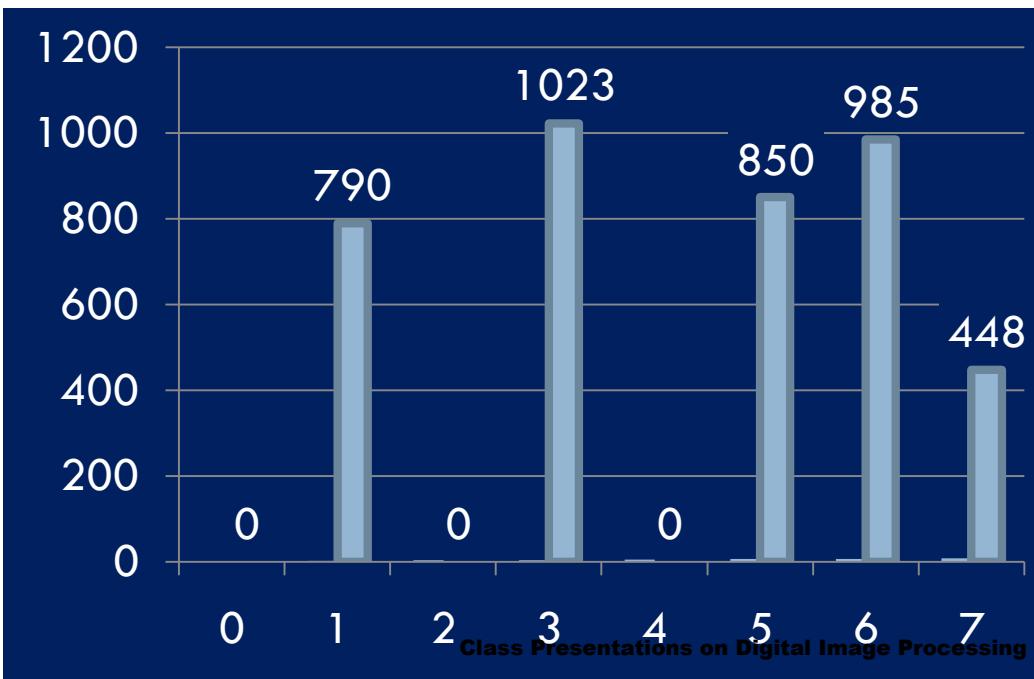
Grey level	$n_k$	$p_r(r_k)$	$s_k$
0	790	0.19	1
1	1023	0.25	3
2	850	0.21	5
3	656	0.16	6
4	329	0.08	6
5	245	0.06	7
6	122	0.03	7
7	81	0.02	7

What will be the histogram of the new image?

Equalized Grey level	$n_k$
0	0
1	790
2	0
3	1023
4	0
5	850
6	$656 + 245 = 985$
7	$245 + 122 + 81 = 448$

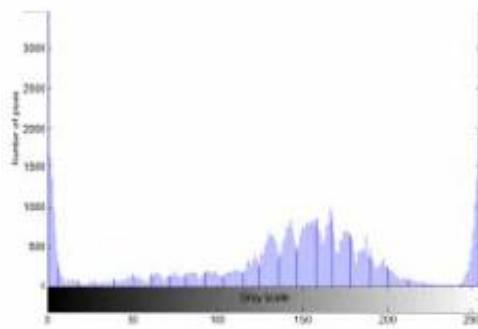
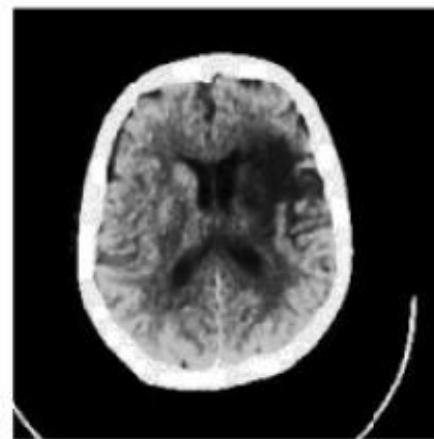
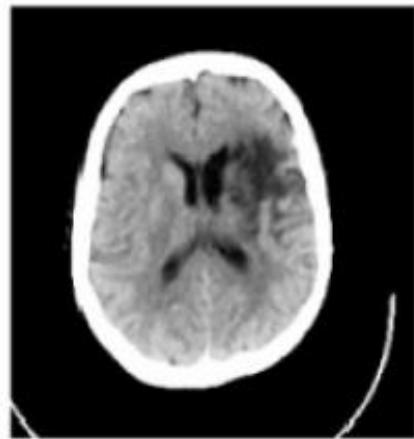


Histogram of the dark image

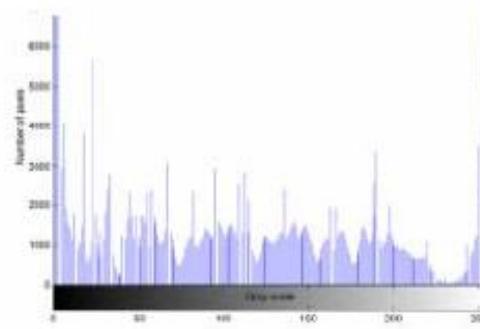


Equalized Histogram of the image

# Histogram Equalization: Example

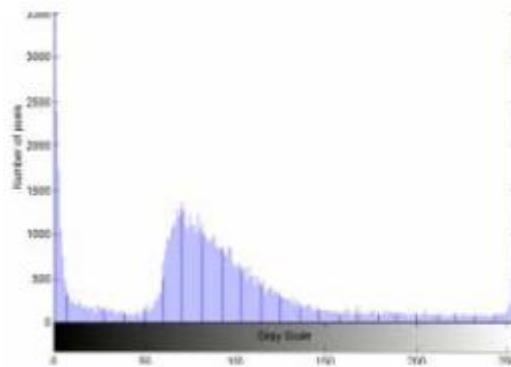
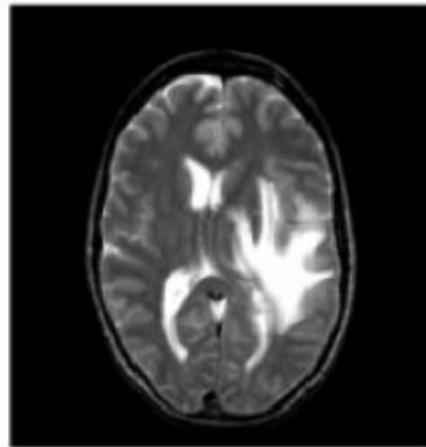


**Before HE**

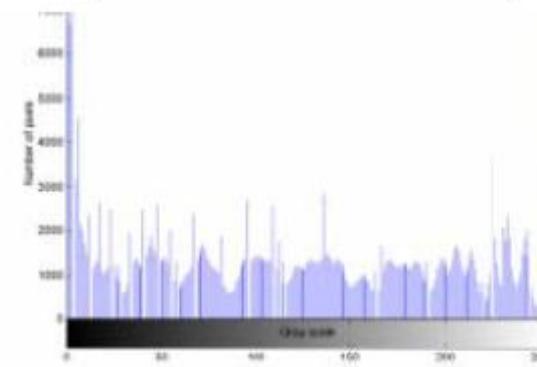
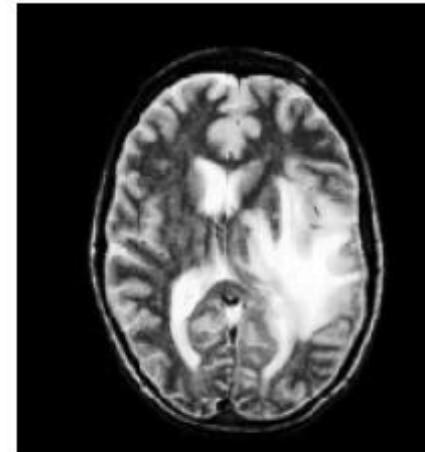


**After HE**

# Histogram Equalization: Example

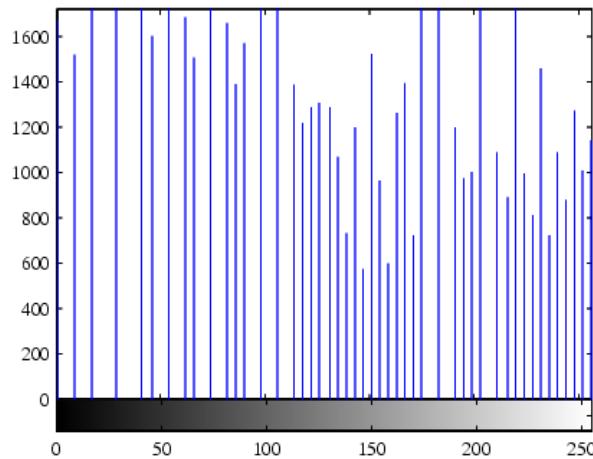
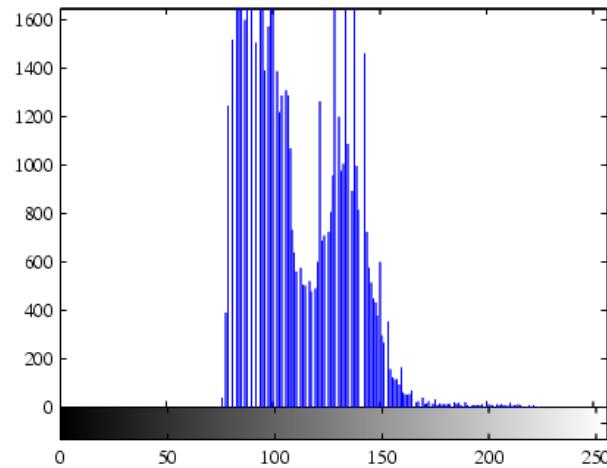


**Before HE**

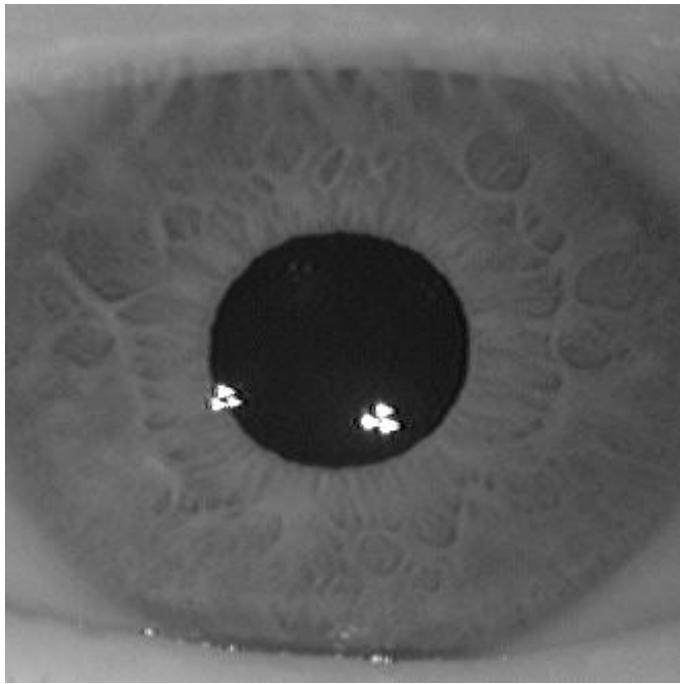


**After HE**

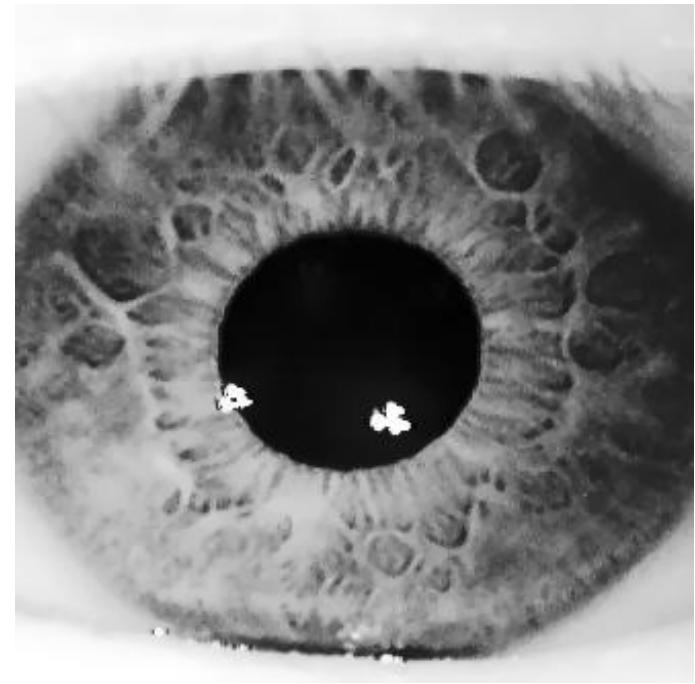
# Histogram Equalization: Example



# Histogram Equalization: Example



**before**



**after**

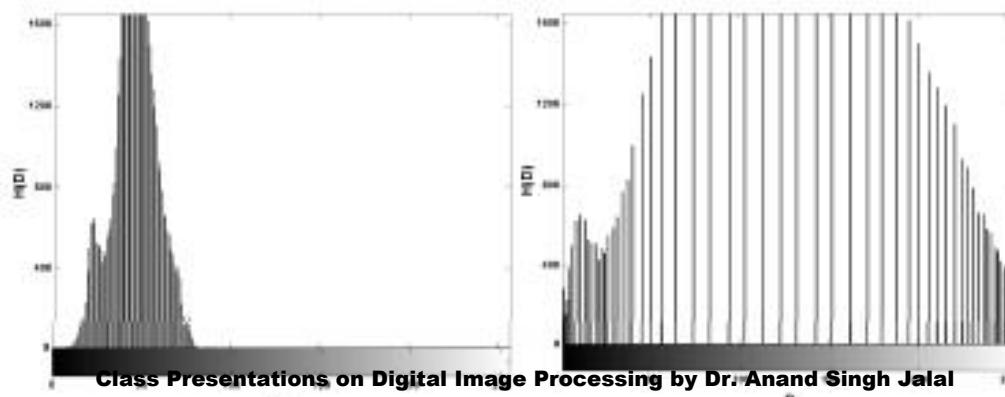
# Histogram Equalization: Example

Aerial photograph of the Pentagon.

Before



After



# Histogram Matching (Specification)

- Histogram equalization has a disadvantage which is that it can generate only one type of output image.
- Sometimes it is useful to be able to specify the shape of the histogram that we wish the processed image to have.
- 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

# Histogram Matching (Specification) ...

For an image, whose enhancement is to be done, we are given an histogram,  $G(z_k)$ , that actually shows how the processed image's histogram should look after applying the transformation function to the i/p image.

$$s_k = T(r_k)$$

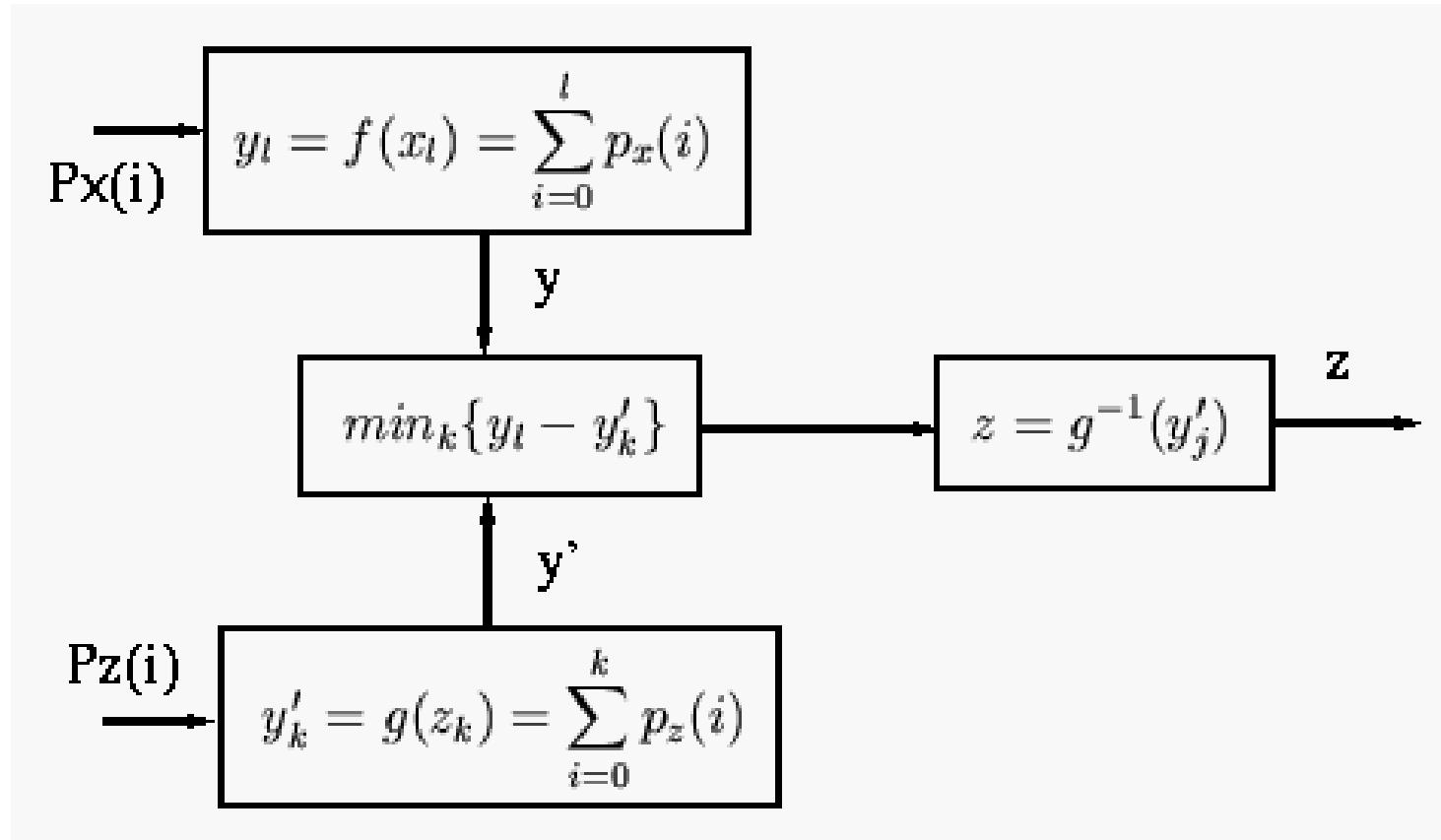
$$G(z_k) = \sum_{i=0}^k p_k(z_i) = s_k$$

$$z_k = G^{-1}(s_k) = G^{-1}[T(r_k)]$$

# Histogram Matching (Specification) ...

- Step 1: Find histogram of input image  $p_r(r_j)$  , and find histogram equalization mapping.
- Step 2: Specify the desired histogram, and find histogram equalization mapping.
- Step 3: Build lookup table:
  - For each gray level  $k$ , find  $s_k$  and then find a  $l$  level so that  $s_k$  best matches  $z_l$ :
$$\min |s_k - z_l|$$
  - setup a lookup entry  $\text{lookup}[k]=l$ .

# Histogram Matching: Discrete Cases



# Histogram Matching: Example

- Suppose that a 3-bit image ( $L=8$ ) of size  $64 \times 64$  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 $p_z(z_q)$
$z_0 = 0$	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

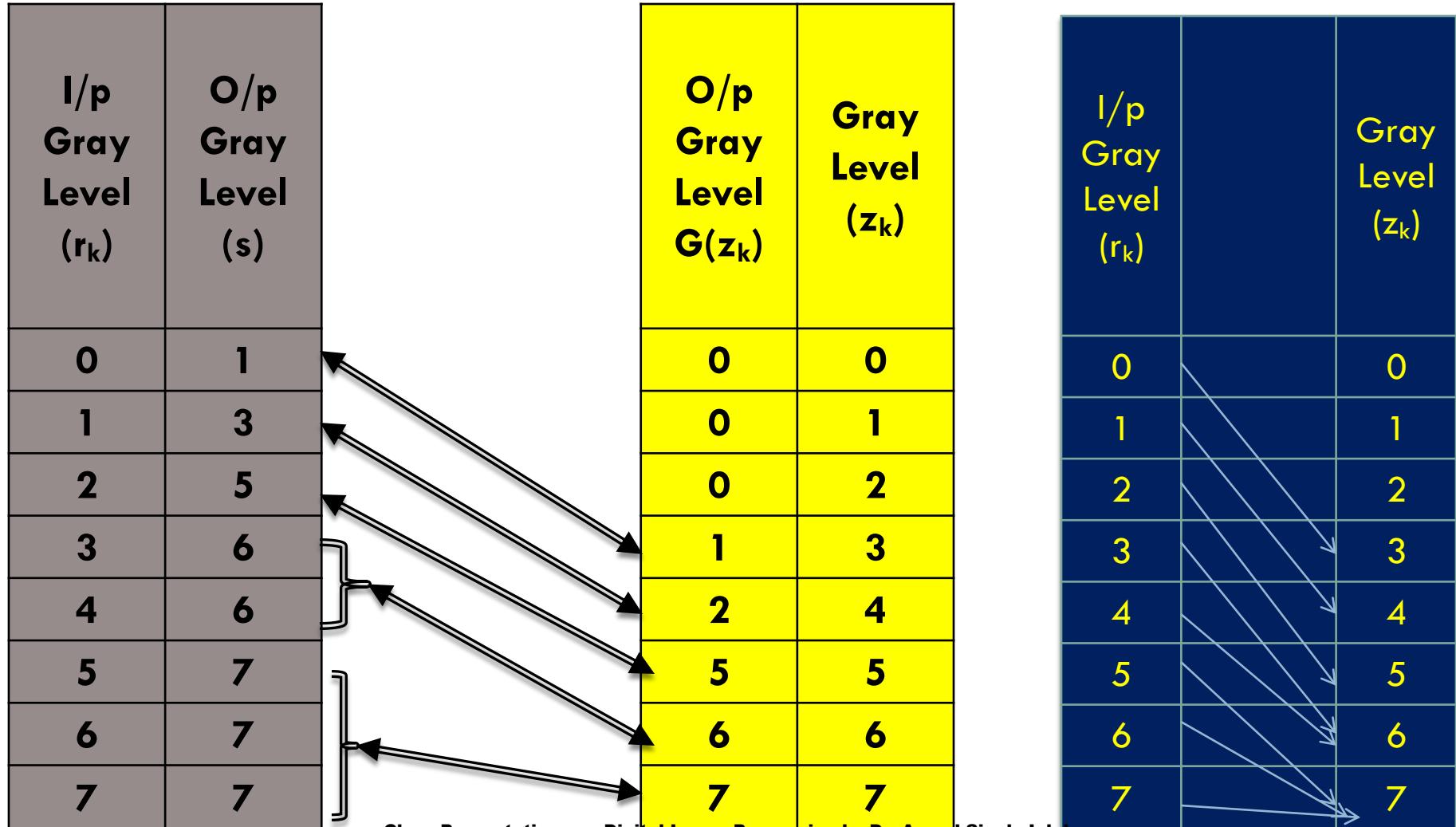
We have already equalized the first histogram.

I/p Gray Level ( $r_k$ )	no. of pixels ( $n_k$ )	$p(r_k) = n_k/MN$	$\Sigma$	$(L-1)\Sigma$	O/p Gray Level ( $s$ )
0	790	0.19	0.19	1.33	1
1	1023	0.25	0.44	3.08	3
2	850	0.21	0.65	4.55	5
3	656	0.16	0.81	5.67	6
4	329	0.08	0.89	6.23	6
5	245	0.06	0.95	6.65	7
6	122	0.03	0.98	6.86	7
7	81	0.02	1.00	7.00	7

# Now equalize the second histogram.

Gray Level (z <sub>k</sub> )	p(z <sub>k</sub> ) = n <sub>k</sub> /MN	Σ	(L-1)Σ	O/p Gray Level G(z <sub>k</sub> )
0	0.00	0	0	0
1	0.00	0	0	0
2	0.00	0	0	0
3	0.15	0.15	1.05	1
4	0.20	0.35	2.45	2
5	0.30	0.65	4.55	5
6	0.20	0.85	5.95	6
7	0.15	1	7	7

# And now we do the matching



# Histogram Matching: Example

$$r_k \rightarrow z_q$$

$$0 \rightarrow 3$$

$$1 \rightarrow 4$$

$$2 \rightarrow 5$$

$$3 \rightarrow 6$$

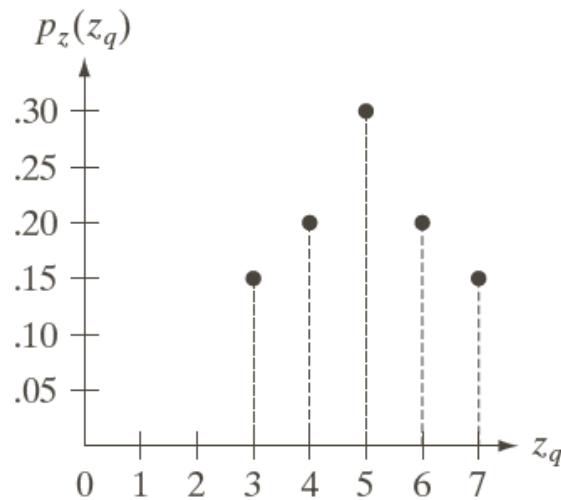
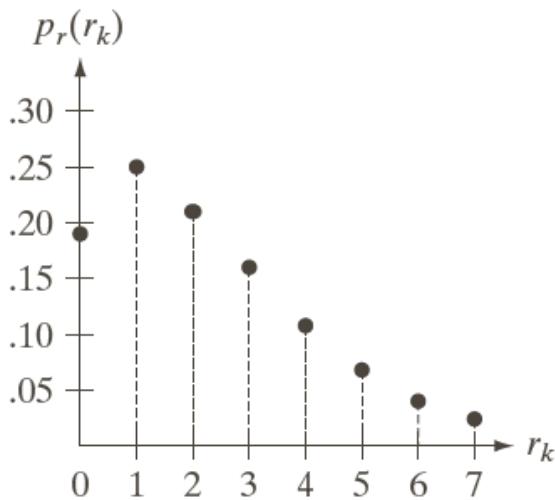
$$4 \rightarrow 6$$

$$5 \rightarrow 7$$

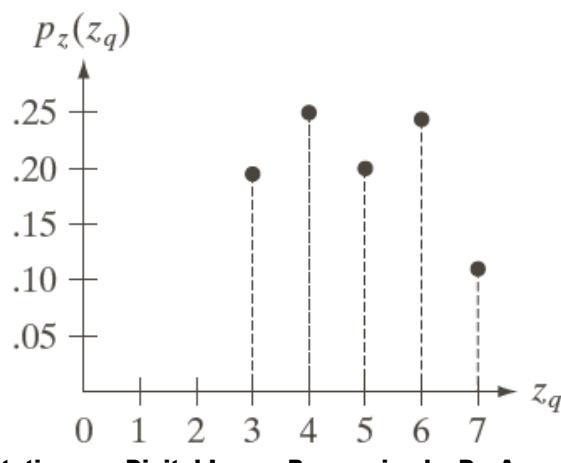
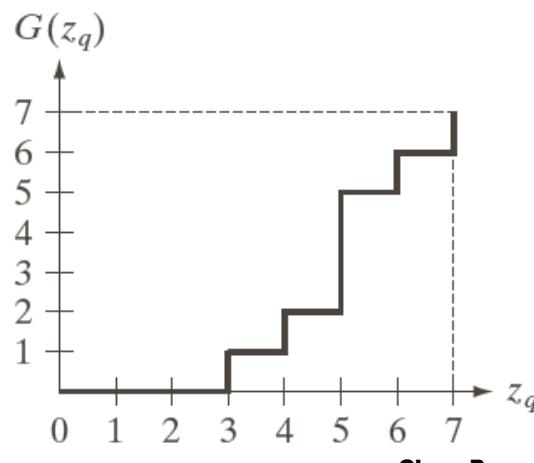
$$6 \rightarrow 7$$

$$7 \rightarrow 7$$

# Histogram Matching: Example

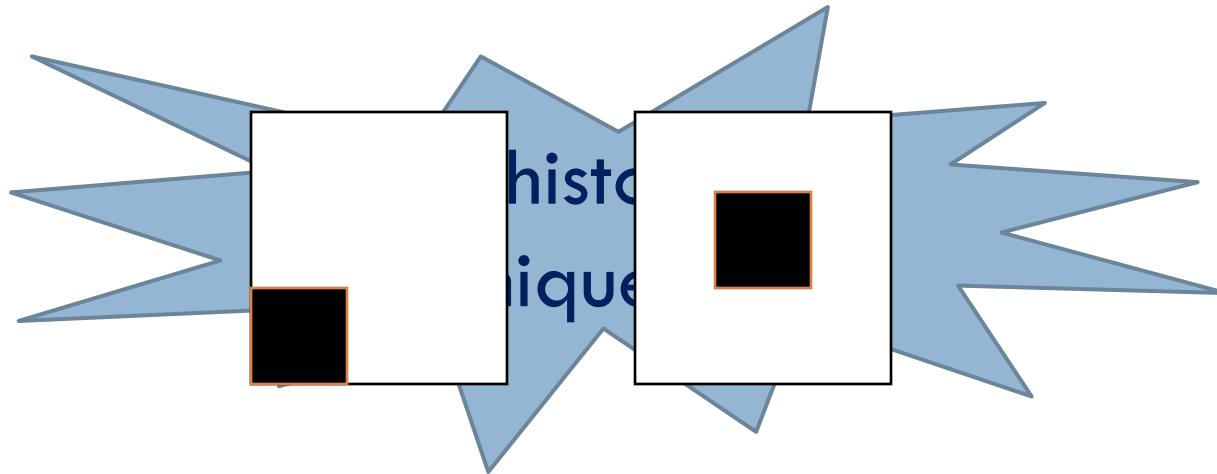


a	b
c	d



- (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).

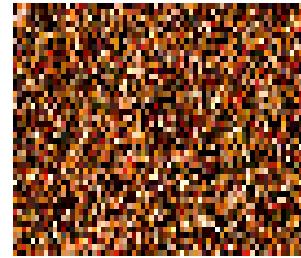
# Histogram processing yields robust image processing results



Histograms of both the images are same !!

# Local Histogram Processing ...

- Q: What happens if I reshuffle all pixels within the image?

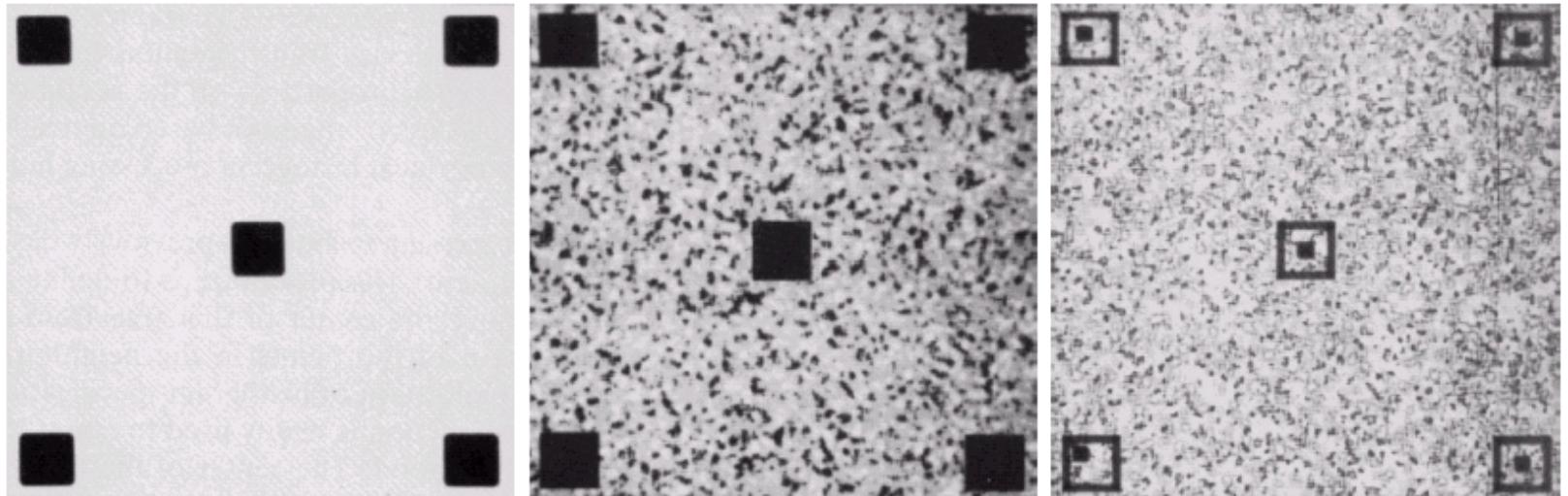


- A: Its histogram won't change. No point processing will be affected...
- Need spatial information to capture this.

# Local Histogram Processing

- Transformation should be based on gray-level distribution in the neighborhood of every pixel.
- Local histogram processing:
  - At each location the histogram of the points in the neighborhood is computed and a histogram equalization or histogram specification transformation function is obtained
  - The gray level of the pixel centered in the neighborhood is mapped
  - The center of the neighborhood is moved the next pixel and the procedure repeated

# Local Histogram Processing: Example



a b c

(a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization using a  $7 \times 7$  neighborhood about each pixel.

# Assignment

- Construct the histogram of the following image.

0	1	2	3	4	5	6	7
0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
8	9	10	11	12	13	14	15

- What will be the effect on the histogram if we set to zero the
- Higher order bit plane?
  - Lower order bit plane ?
- Take your grey scale image & construct its histogram.

# Assignment

Q. Equalize the following histogram and give the resultant histogram.

Gray levels (rk)	0	1	2	3	4	5	6	7
Number of pixels (nk)	50	100	100	300	200	200	50	0

Q. Perform histogram specification on the 8x8 image. The gray level distribution of the image is given below:

Gray levels (rk)	0	1	2	3	4	5	6	7
Number of pixels (nk)	8	10	10	2	12	16	4	2

Gray levels (rk)	0	1	2	3	4	5	6	7
Number of pixels (nk)	0	0	0	0	20	20	16	8

# Assignment

Match the histogram A to the histogram B.

**Histogram A**

Gray Level	0	1	2	3	4	5	6	7
No. of Pixels	4	17	15	18	24	12	0	10

**Histogram B**

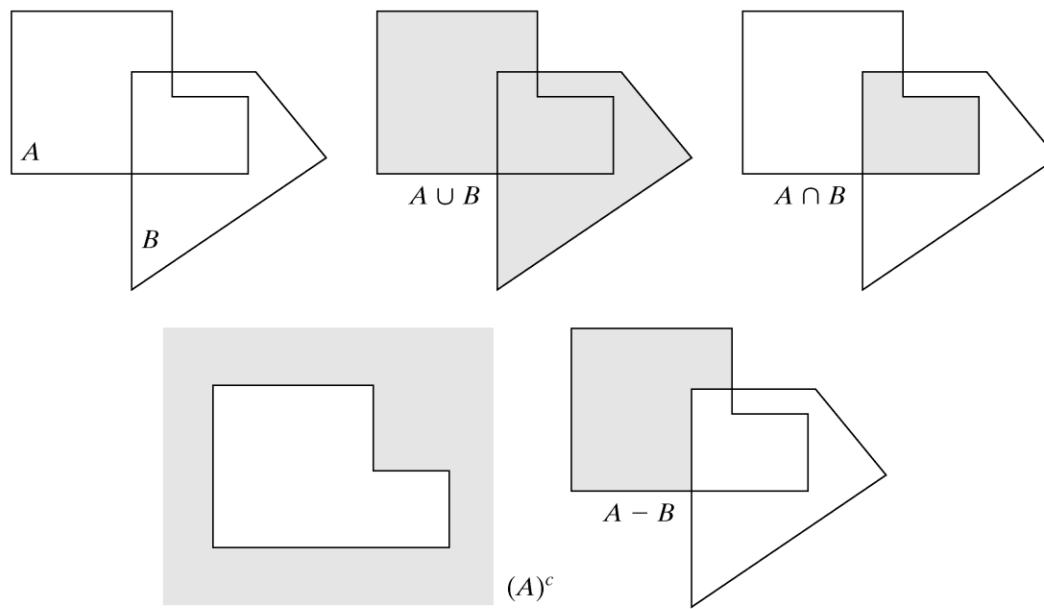
Gray Level	0	1	2	3	4	5	6	7
No. of Pixels	0	0	0	36	24	12	8	20

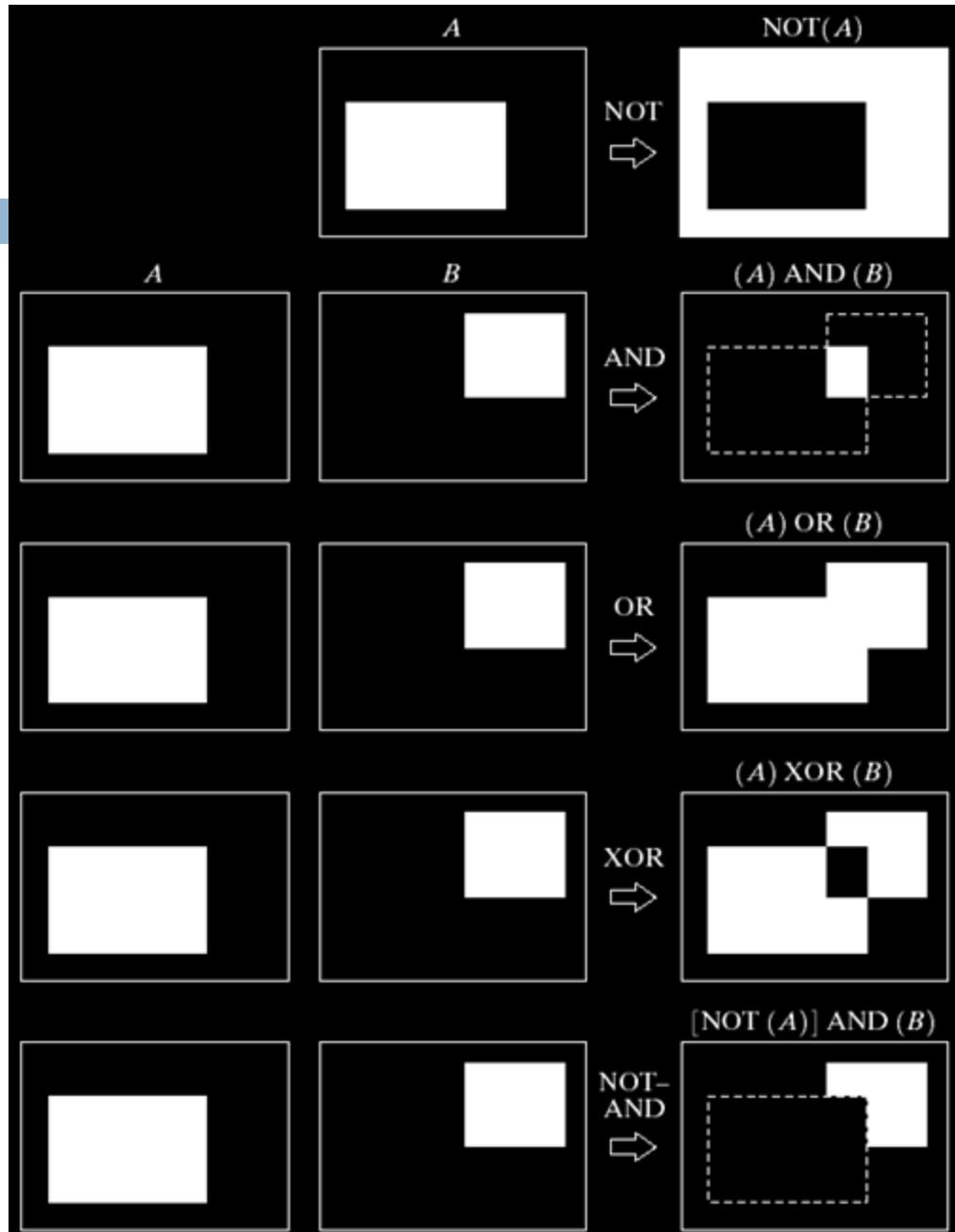
# Enhancement Using Arithmetic/Logic Operations

- Arithmetic/Logic operations are performed on the pixels of two or more images.
- **Arithmetic:** p and q are the pixel values at location  $(x,y)$  in first and second images respectively
  - Addition:  $p+q$
  - Subtraction:  $p-q$
  - Multiplication:  $p.q$
  - Division:  $p/q$

# Logic Operations

- When dealing with logic operations on gray-level images, pixel values are processed as strings of binary numbers.
- AND, OR, COMPLEMENT (NOT)

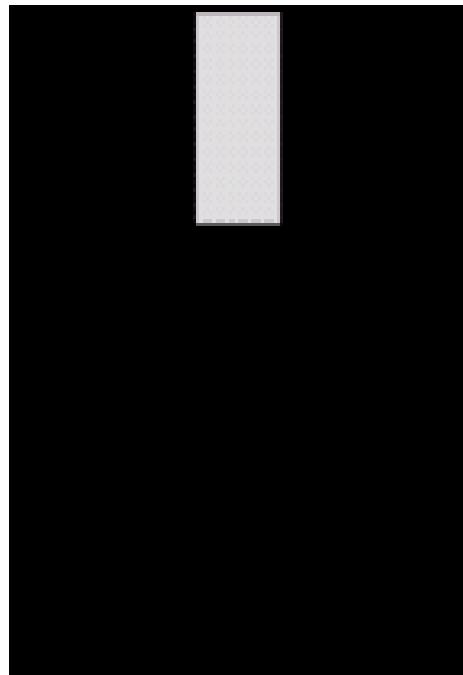




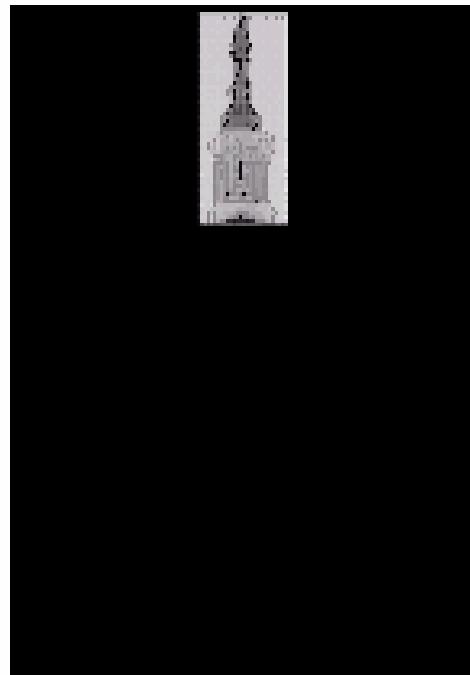
# Example of AND Operation



original image



AND image  
mask

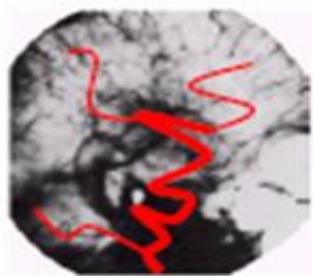


result of AND  
operation

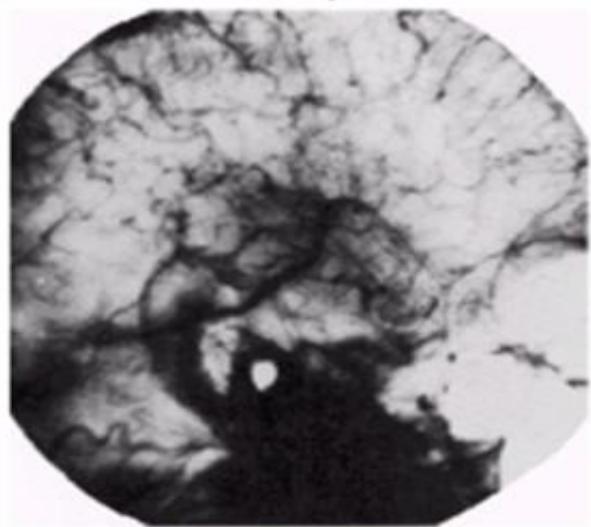
# Image Subtraction

- Obtained by computing the difference between all pairs of corresponding pixels from images  $f$  &  $h$   
$$g(x,y) = f(x,y) - h(x,y)$$
- Example: imaging blood vessels and arteries in a body. Blood stream is injected with a dye and X-ray images are taken before and after the injection
  - $f(x,y)$ : image after injecting a dye
  - $h(x,y)$ : image before injecting the dye
- The difference of the 2 images yields a clear display of the blood flow paths.

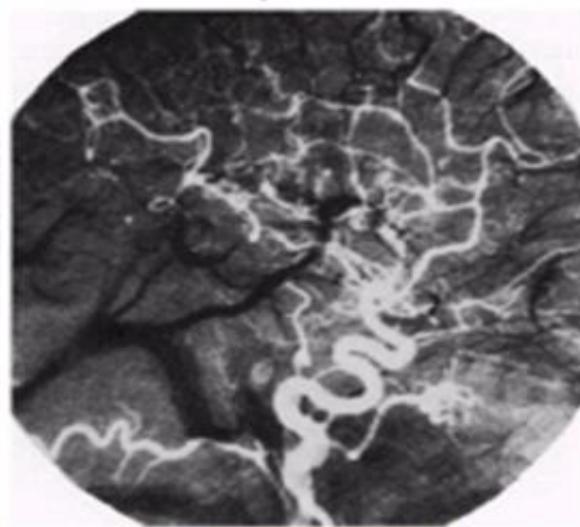
# Image Subtraction: Example



$$f(x, y) - h(x, y) = g(x, y)$$



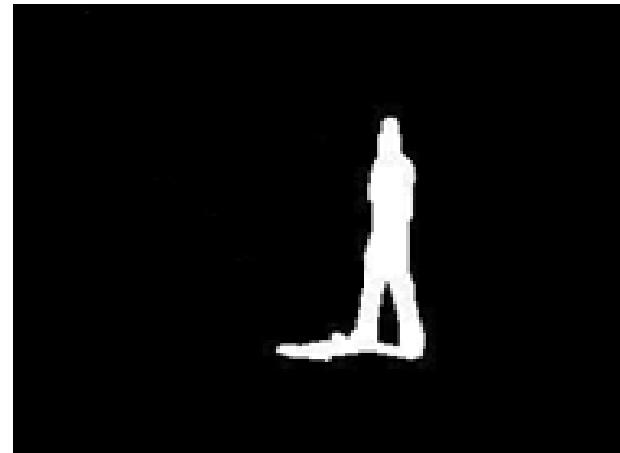
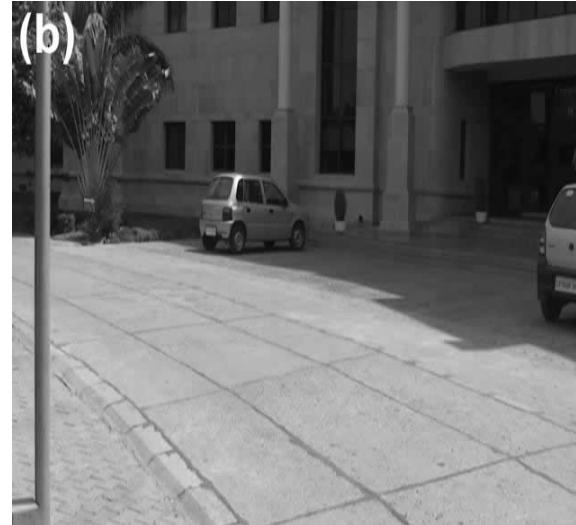
Mask



Results

Enhancement  
by image  
subtraction

# Image Subtraction: Example ...



# Image Averaging

- A noisy image:

$$g(x, y) = f(x, y) + n(x, y)$$

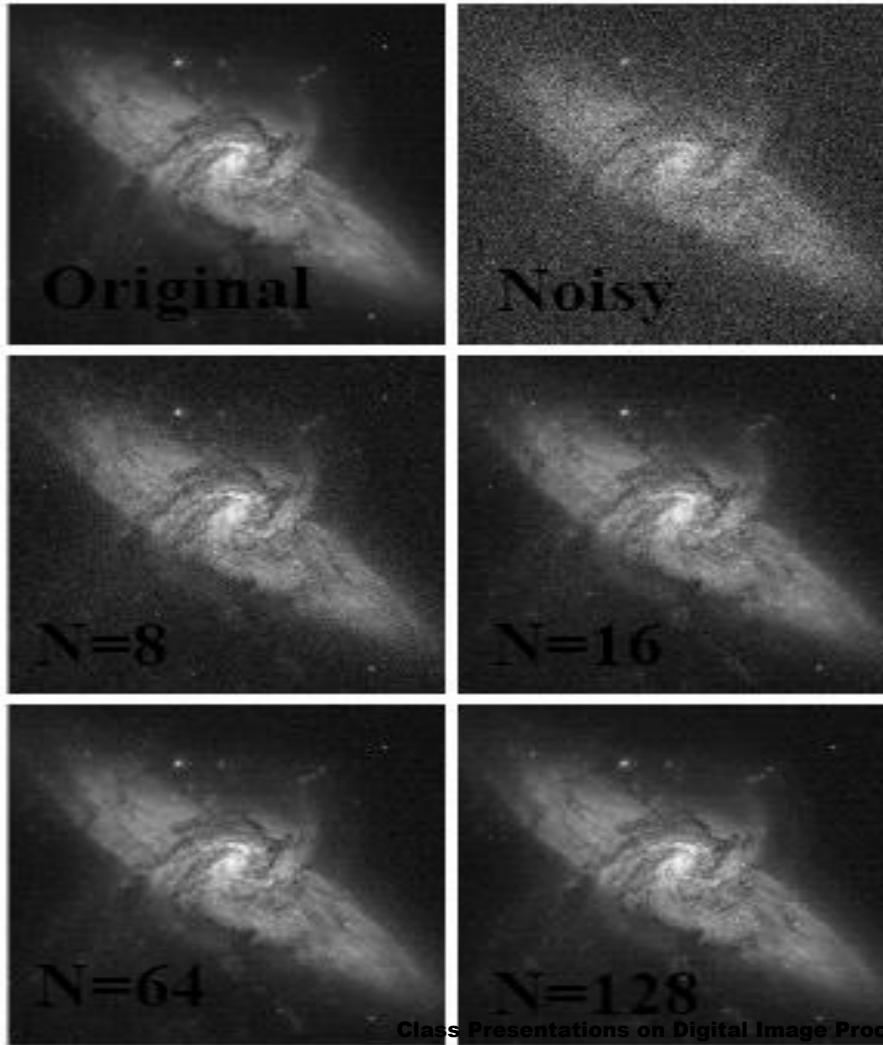
Averaging  $M$  different noisy images:

$$\bar{g}(x, y) = \frac{1}{M} \sum_{i=1}^M g_i(x, y)$$

As  $M$  increases, the variability of the pixel values at each location decreases.

This means that  $g(x, y)$  approaches  $f(x, y)$  as the number of noisy images used in the averaging process increases.

# Image Averaging: Example



- a. Image of Galaxy3314,
- b. Image corrupted by additive Gaussian Noise with zero mean and a standard deviation of 64 gray levels
- c. c-f Results of averaging K=8,16,64 and 128 noisy images (Original image courtesy of NASA)

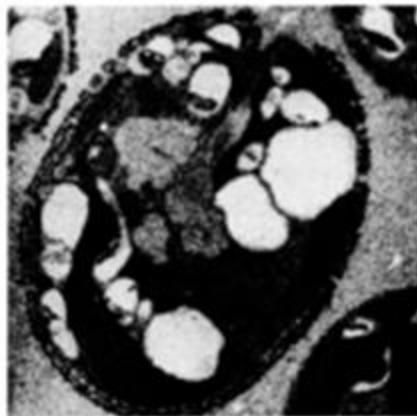
# Image Averaging: Example



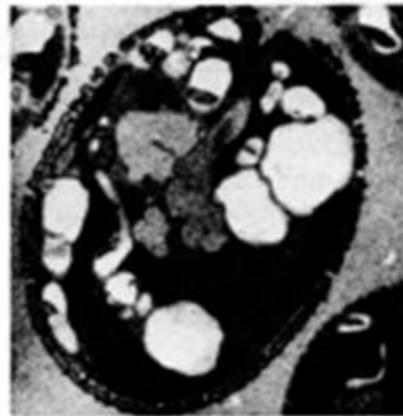
(a)



(b)



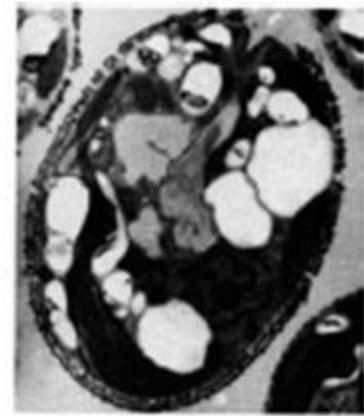
(c)



(d)



(e)

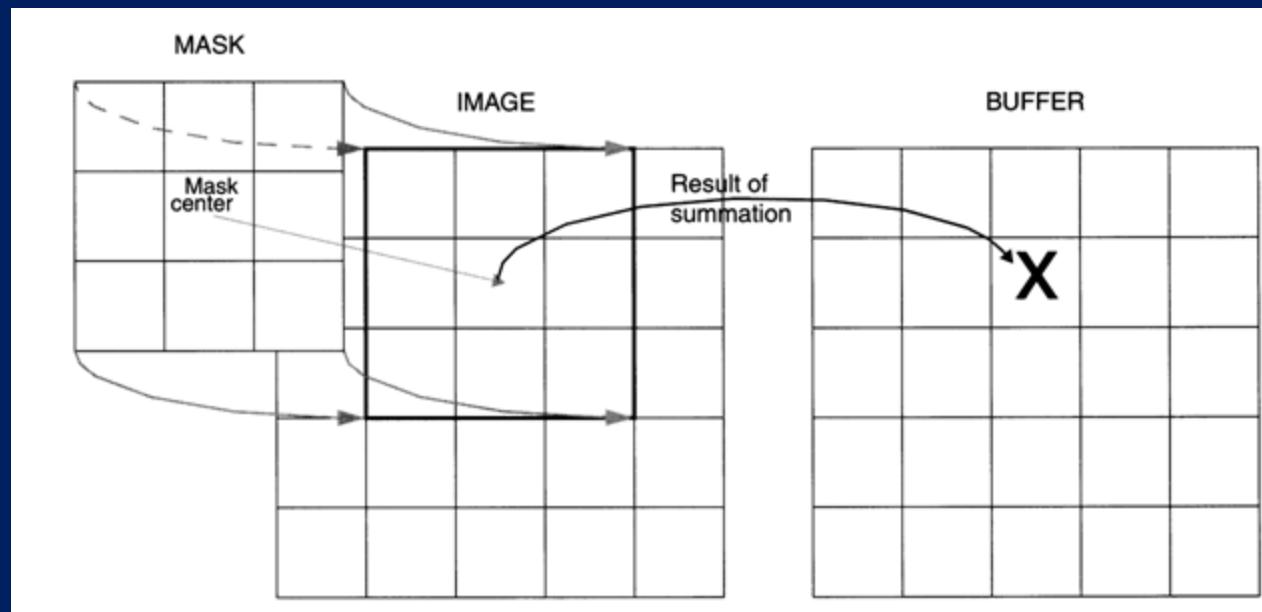


(f)

Example of noise reduction by averaging

- (a) a typical noisy image,
- (b)-(f) results of averaging 2, 8, 16, 32, and 128 noisy image.

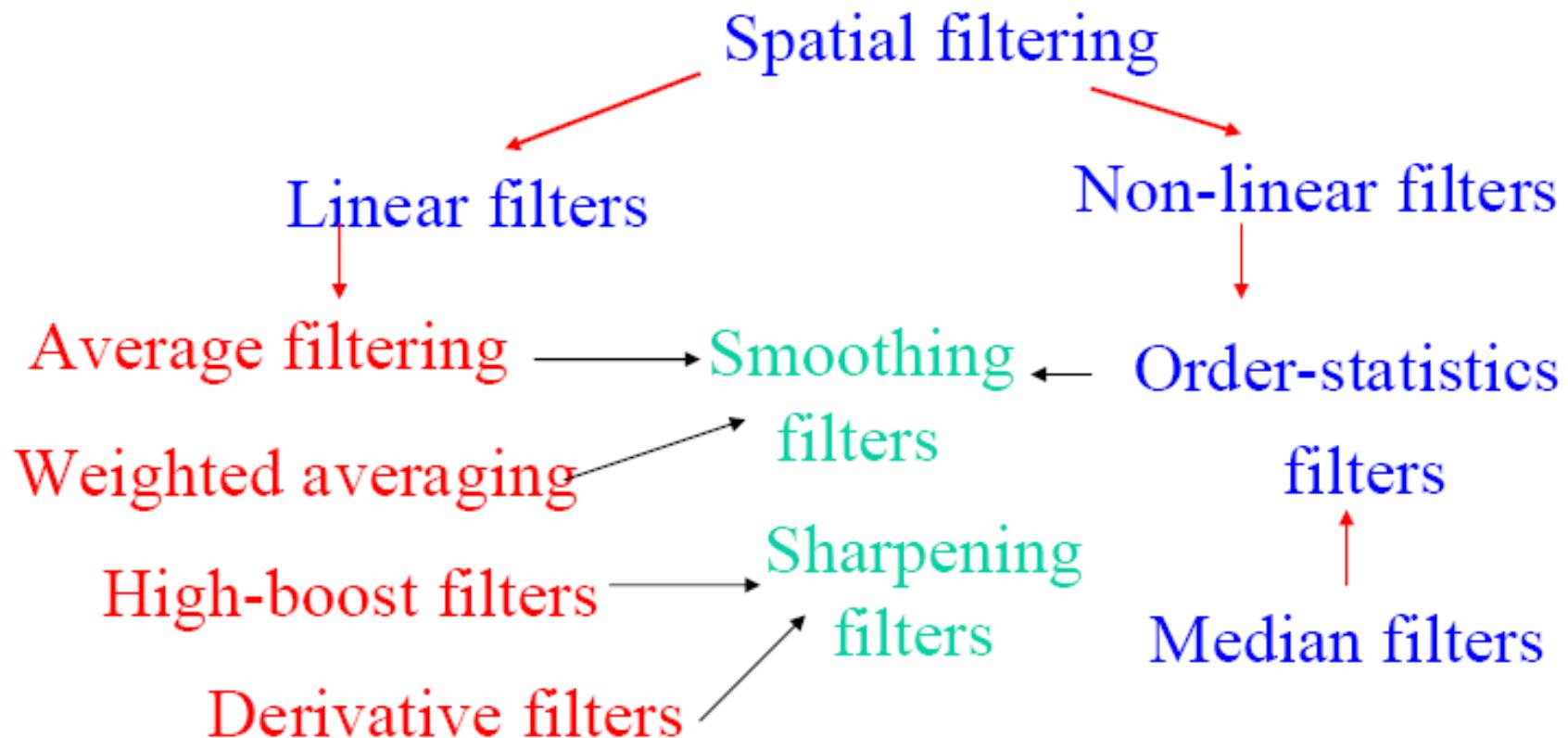
# Spatial Filtering



# Spatial Filtering

- Enhancement techniques based on the pixels of an image are often referred to as Spatial filtering
  - Working in a neighbourhood of every pixel in an image
- Filter term in “Digital image processing” is referred to the **subimage**
- There are others term to call subimage such as **mask, kernel, template, or window**
- The value in a filter subimage are referred as **coefficients, rather than pixels.**

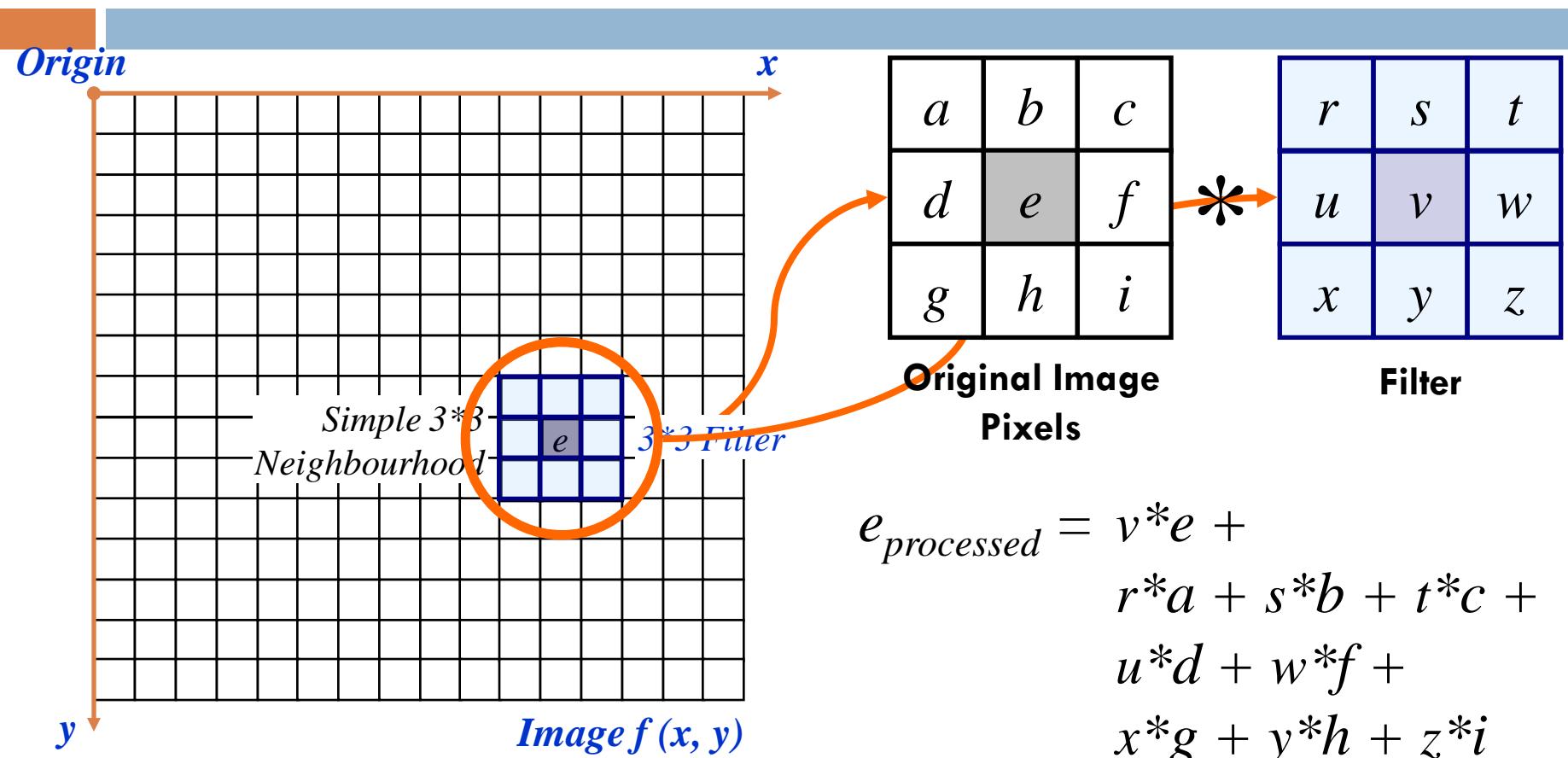
# Spatial Filtering



# Mechanics of Spatial filtering

- The process consists simply of moving the filter mask from point to point in an image.
- At each point  $(x, y)$  the response of the filter at that point is calculated using a predefined relationship.
- If the operation performed on the pixel is linear, then the filter is called **linear spatial filter**.

# Mechanics of Spatial filtering



The above is repeated for every pixel in the original image to generate the filtered image

# Spatial Filtering Process

- Simply move the filter mask from point to point in an image.
- At each point  $(x,y)$ , the response of the filter at that point is calculated using a predefined relationship.

$$\begin{aligned} R &= w_1 z_1 + w_2 z_2 + \dots + w_{mn} z_{mn} \\ &= \sum_{i=i}^{mn} w_i z_i \end{aligned}$$

In general, 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:

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

where (as per the previous discussion)

$$a = (m - 1) / 2 \quad \& \quad b = (n - 1) / 2$$

To generate a complete filtered image, the above equation must be applied for

$$x = 0, 1, \dots, M-1 \quad \& \quad y = 0, 1, \dots, N-1$$

# Correlation Vs Convolution

**Two closely related concepts**

➤ Correlation

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

➤ Convolution

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

Find the result of Correlation & Convolution for the following image  $f$  and  $1 \times 5$  mask  $w$ .

$f$

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

$w$

1	2	3	2	8
---	---	---	---	---

Origin

**Note:**

To allow every pixel in  $w$  to visit every pixel in  $f$ , pad  $f$  with enough Os on either side.

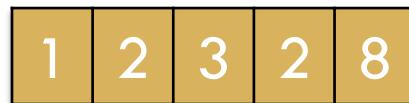
If the filter is of size  $m$ , then the no. of Os to be added are ???

$m-1$

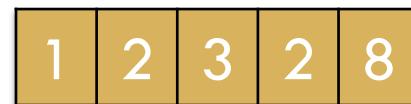
# Correlation



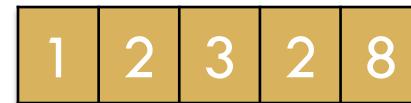
$x = 0$



$x = 1$



$x = 3$



$x = 11$



# Correlation

**Result of full correlation**

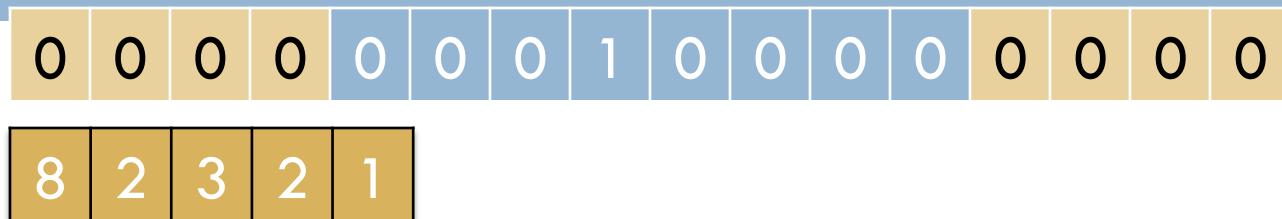


**Result of cropped correlation**

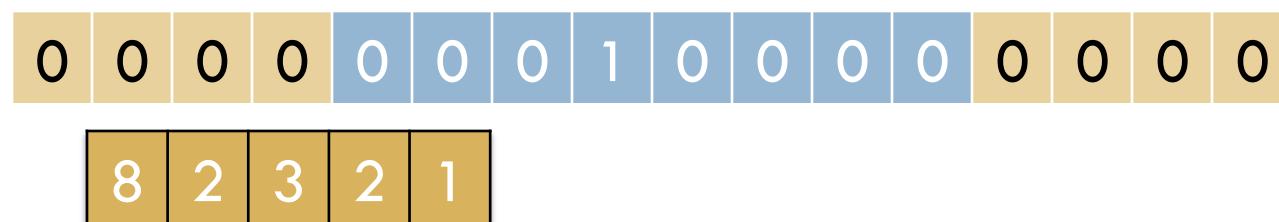


# Convolution

x = 0



x = 1



## Result of full convolution



## Result of cropped convolution



# Smoothing Spatial Filters

- Used for blurring and for noise reduction
- Blurring is used in preprocessing steps, such as
  - removal of small details from an image prior to object extraction
  - bridging of small gaps in lines or curves
- noise reduction can be accomplished by blurring with a linear filter and also by a nonlinear filter
- There are 2 types of smoothing spatial filters
  - Smoothing Linear Filters
  - Order-Statistics Filters

# Smoothing Linear Filters

- Output is simply the average of the pixels contained in the neighborhood of the filter mask.
- Also Called **Averaging filters** or **Lowpass filters**.
- Replacing the value of every pixel in an image by the average of the gray levels in the neighborhood will reduce the “sharp” transitions in gray levels.
- Sharp transitions
  - **random noise in the image**
  - **edges of objects in the image**

**Thus, smoothing can reduce noises (desirable) and blur edges (undesirable)**

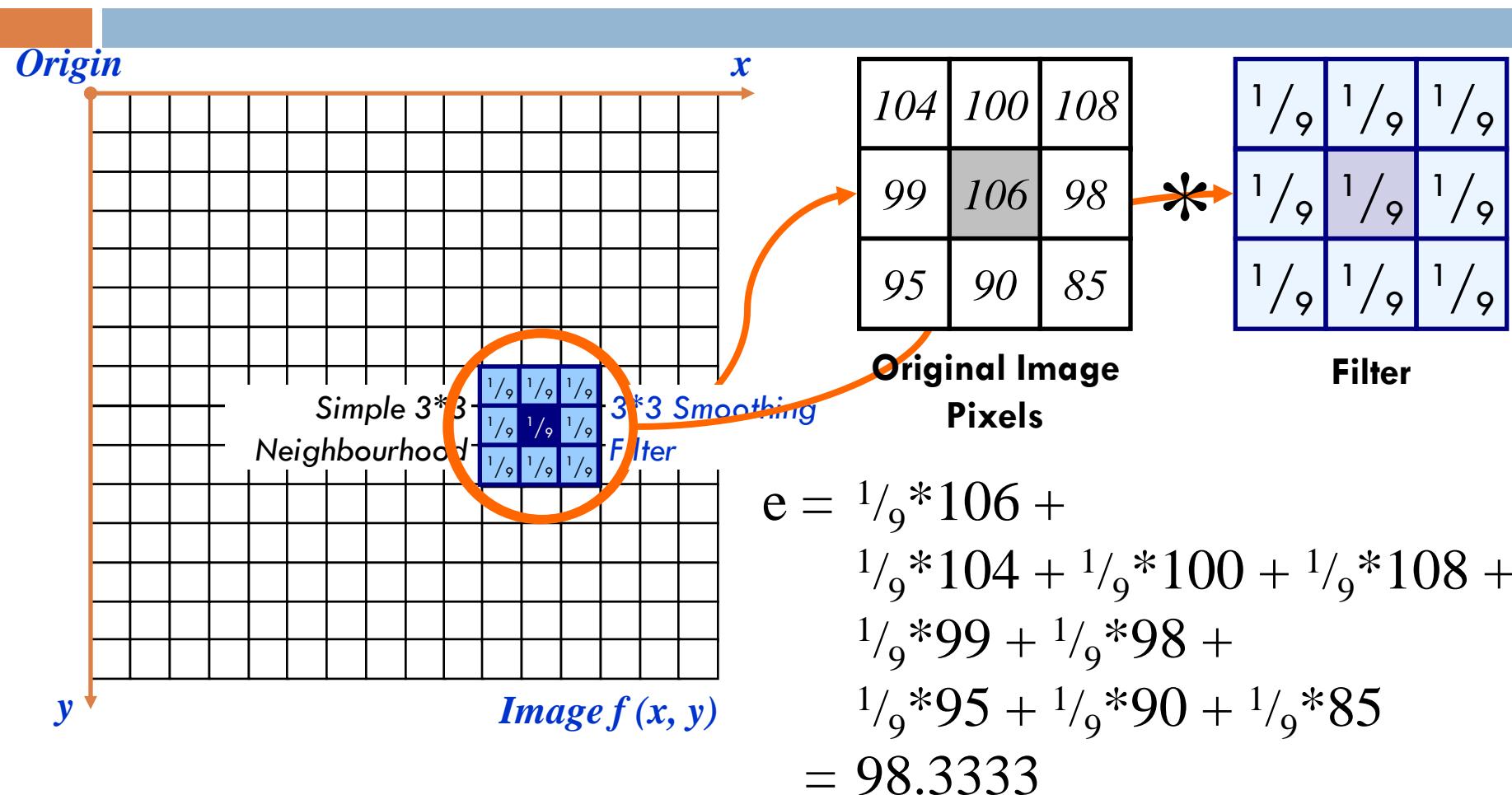
# Smoothing Linear Filters ...

- One of the simplest spatial filtering operations we can perform is a smoothing operation
  - Simply average all of the pixels in a neighbourhood around a central value
  - Especially useful in removing noise from images
  - Also useful for highlighting gross detail

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

Simple  
averaging  
filter

# Smoothing Linear Filters ...



**The above is repeated for every pixel in the original image to generate the smoothed image**

# 3x3 Smoothing Linear Filters

$$\frac{1}{9} \times \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$

Standard Average

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

weighted average

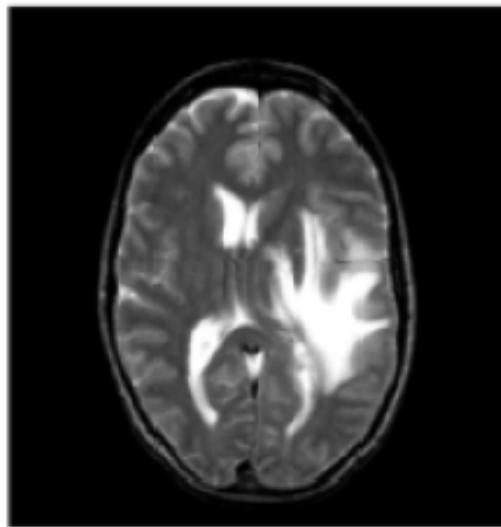
the center is the most important and other pixels are inversely weighted as a function of their distance from the center of the mask

# Smoothing Linear Filters

- The general implementation for filtering an  $M \times N$  image with a weighted averaging filter of size  $m \times n$  is given by the expression

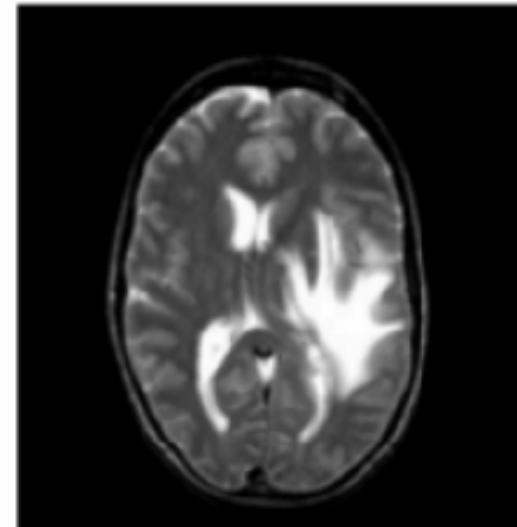
$$g(x, y) = \frac{\sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)}{\sum_{s=-a}^a \sum_{t=-b}^b w(s, t)}$$

# Smoothing Linear Filters: Example



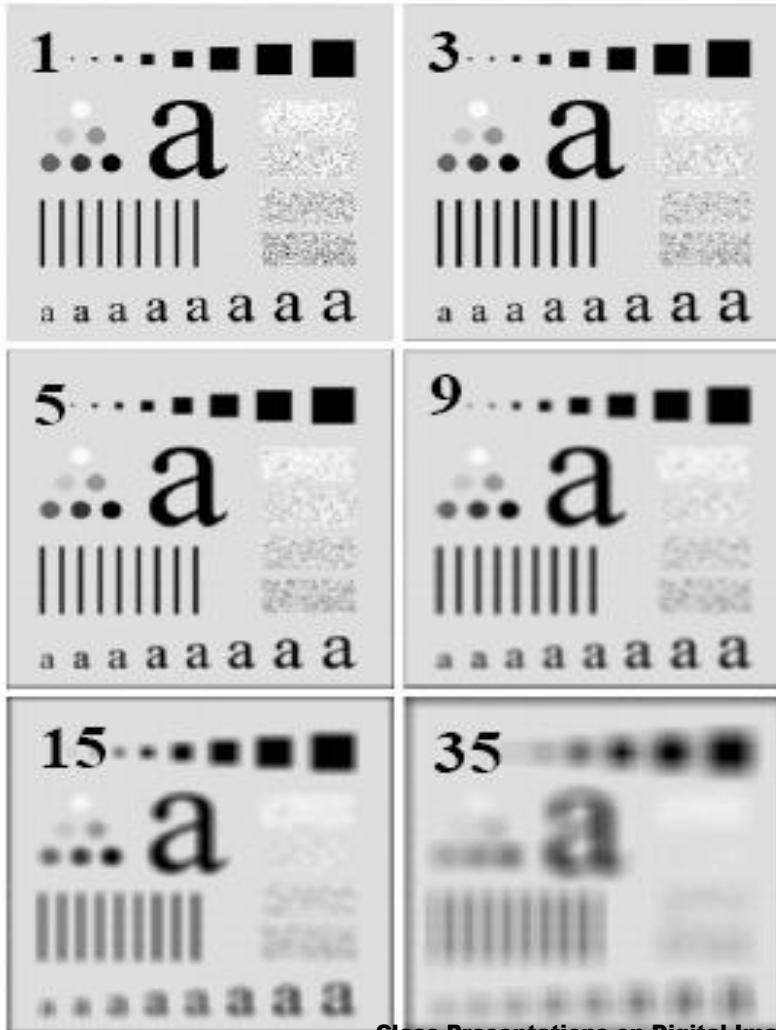
Before smoothing

1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16



After smoothing

# Smoothing Linear Filters: Example



- Original image of 500 x 500 pixels
- Results of smoothing with square averaging filter masks of size  $m=3, 5, 9, 15$  and 35 respectively
- Note:
  - **big mask is used to eliminate small objects from an image.**
  - **the size of the mask establishes the relative size of the objects that will be blended with the background.**

# Weighted Smoothing Filters

- More effective smoothing filters can be generated by allowing different pixels in the neighbourhood different weights in the averaging function

- Pixels closer to the central pixel are more important
- Often referred to as a *weighted averaging*

$1/_{16}$	$2/_{16}$	$1/_{16}$
$2/_{16}$	$4/_{16}$	$2/_{16}$
$1/_{16}$	$2/_{16}$	$1/_{16}$

Weighted averaging filter

# Order-Statistics Filters

- Order-statistics filters are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter,
- Example
  - median filter :  $R = \text{median}\{z_k \mid k = 1, 2, \dots, n \times n\}$
  - max filter :  $R = \max\{z_k \mid k = 1, 2, \dots, n \times n\}$
  - min filter :  $R = \min\{z_k \mid k = 1, 2, \dots, n \times n\}$
- note:  $n \times n$  is the size of the mask

# Process of Median filter

	10	15	20	
	20	100	20	
	20	20	25	

10, 15, 20, 20, 20, 20, 20, 25, 100

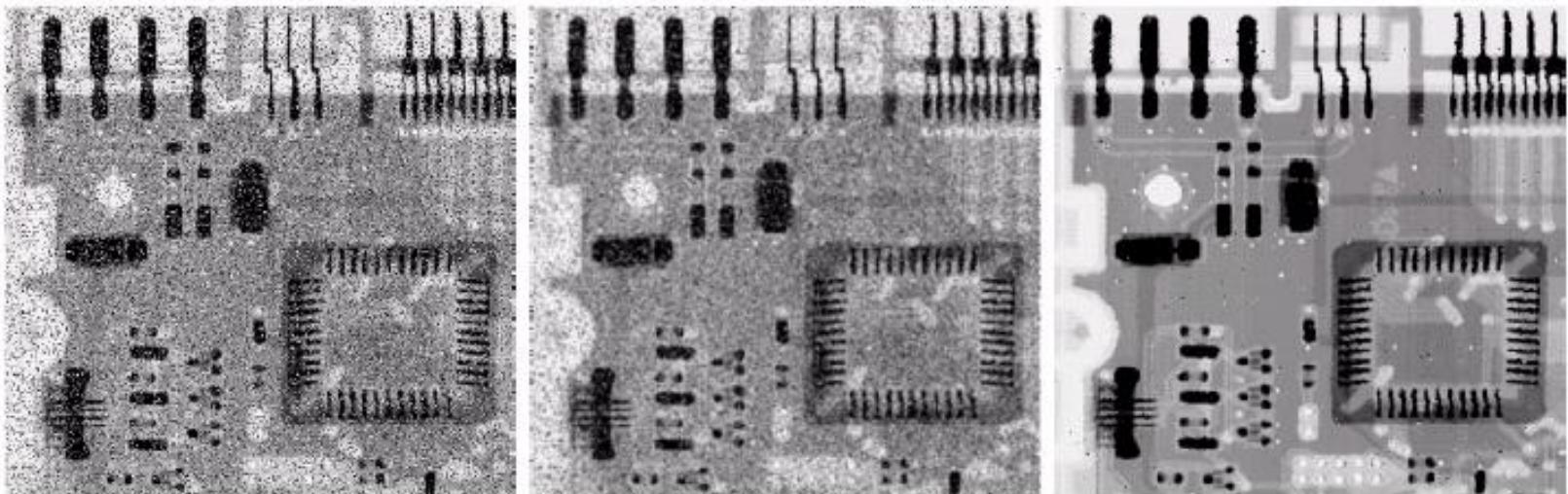
↑  
5<sup>th</sup>

- Crop region of neighborhood
- Sort the values of the pixel in the region
- In the  $M \times N$  mask the median is  $(M \times N \text{ div } 2) + 1$

# Median Filtering

- Median filtering is particularly effective in the presence of impulse noise (salt and pepper noise).
- Unlike average filtering, median filtering does not blur too much image details.
- Advantages:
  - Removes impulsive noise
  - Preserves edges
- Disadvantages:
  - performance poor when # of noise pixels in the window is greater than 1/2 # in the window
  - performs poorly with Gaussian noise

# Median Filtering: Example



- a) X-ray image of circuit board corrupted by salt-pepper noise
- b) Noise reduction with a  $3 \times 3$  averaging mask
- c) Noise reduction with a  $3 \times 3$  median filter

# Median Filtering: Example



Noise reduction with a  $3 \times 3$  median filter

# Sharpening Spatial Filters

- The principal objective of sharpening is to **highlight fine detail in an image** or **to enhance detail that has been blurred**, either in error or as a natural effect of a particular method of image acquisition.
- First and second order derivatives are commonly used for sharpening:

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

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

# Blurring vs. Sharpening

- As we know that blurring can be done in spatial domain by **pixel averaging** in a neighbors
- since **averaging is analogous to integration**
- Therefore, logically it can be concluded that the sharpening must be accomplished by **spatial differentiation.**

# Derivative operator

- The strength of the response of a derivative operator is proportional to the degree of discontinuity of the image at the point at which the operator is applied.
- thus, image differentiation
  - enhances edges and other discontinuities (noise)
  - deemphasizes area with slowly varying gray-level values.

# First-order derivative

- a basic definition of the first-order derivative of a one-dimensional function  $f(x)$  is the difference

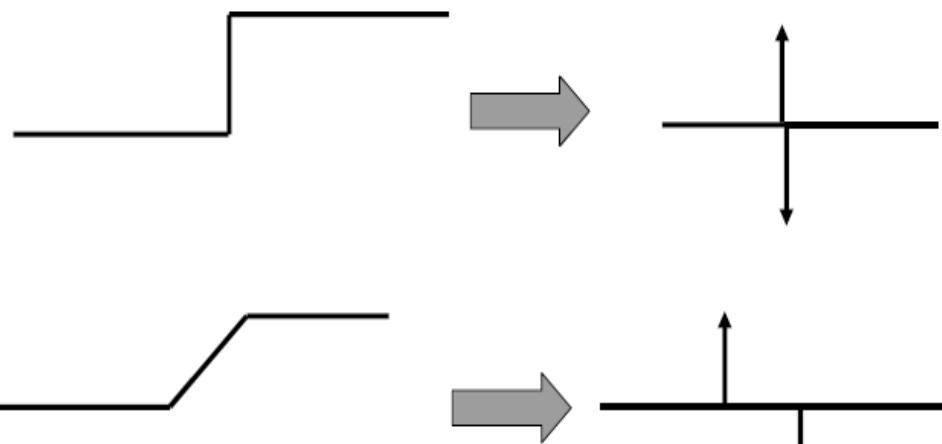
$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$



# Second-order derivative

- similarly, we define the second-order derivative of a one-dimensional function  $f(x)$  is the difference

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



# **Response of First and Second order derivatives**

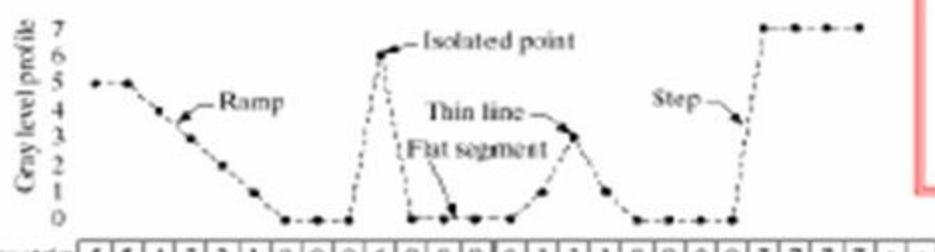
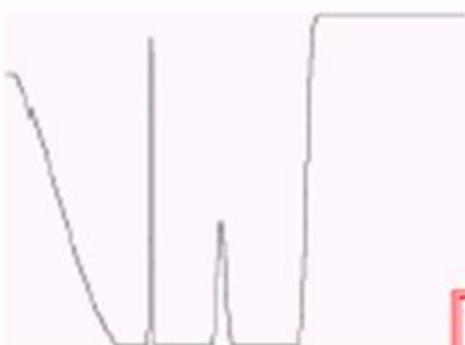
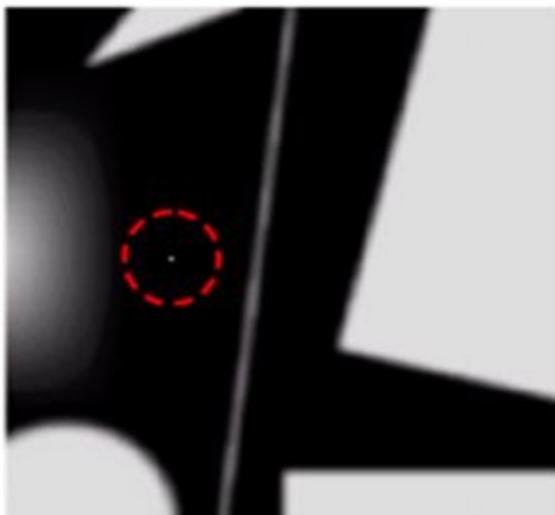
**Response of first order derivative is:**

- zero in flat segments (area of constant grey values)
- Non zero at the onset of a grey level step or ramp
- Non zero along ramps

**Response of second order derivative is:**

- Zero in flat areas
- Non zero at the onset of a grey level step or ramp
- Zero along ramps of constant slope

# First and Second Order Derivatives: Example

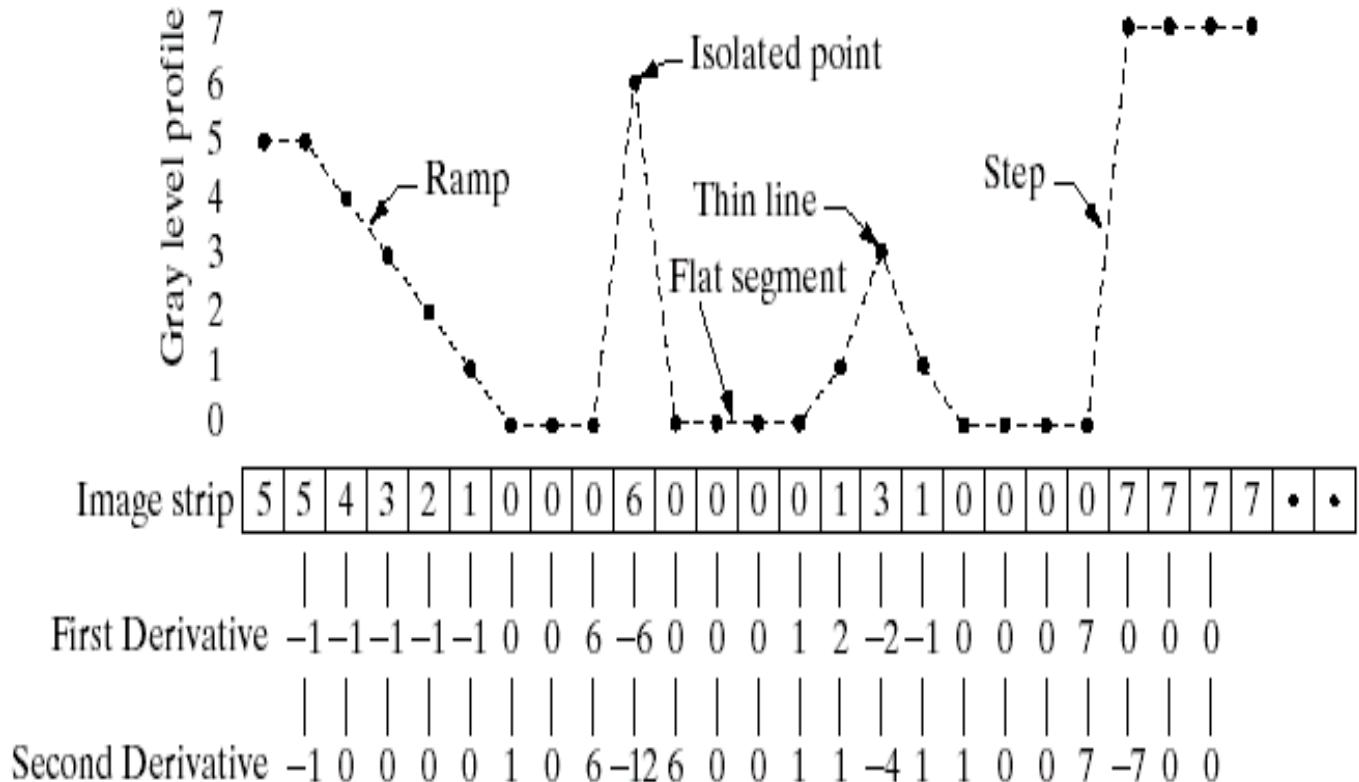


First Derivative	-1	-1	-1	-1	-1	0	0	6	-6	0	0	0	1	2	-2	-1	0	0	0	7	0	0	0
Second Derivative	-1	0	0	0	0	1	0	6	-12	6	0	0	1	1	-4	1	1	0	0	7	-7	0	0

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

$$\begin{aligned}\frac{\partial^2 f}{\partial x^2} &= f_x(x) - f_x(x-1) \\ &= (f(x+1) - f(x)) - (f(x) - f(x-1)) \\ &= f(x+1) + f(x-1) - 2 \cdot f(x)\end{aligned}$$

# First and Second Order Derivatives: Example ...



# First and Second Order Derivatives ...

## □ Analysis

- 1) The 1st-order derivative is nonzero along the entire ramp, while the 2nd-order derivative is nonzero only at the onset and end of the ramp.
- 2) Edges in an image represent this type (ramp) of transition. Therefore,  
**1<sup>st</sup> make thick edge and 2<sup>nd</sup> make thin, much finer edges**
- 3) The response at and around the point is much stronger for the 2nd- than for the 1st-order derivative.

# 1st and 2nd Derivative Comparison

- ✓ First-order derivatives generally produce thicker edges in an images.
- ✓ Second-order derivatives have a stronger response to fine detail (e.g. thin lines or isolated points).
- ✓ First-order derivatives generally have a stronger response to a gray-level step.
- ✓ Second-order derivatives produce a double response at step changes in gray level

# The Laplacian

## (Use of Second derivative for Enhancement)

- The filter is expected to be **isotropic**: response of the filter is independent of the direction of discontinuities in an image.
- Simplest 2-D isotropic second order derivative is the Laplacian:

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

$$\begin{aligned}\nabla^2 f = & f(x+1, y) + f(x-1, y) + f(x, y+1) \\ & + f(x, y-1) - 4f(x, y)\end{aligned}$$

# 2-Dimensional Laplacian

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

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

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

0	1	0
1	-4	1
0	1	0

# Laplacian Masks

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

*90° isotropic*

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

*45° isotropic*

0	1	0
1	-4	1
0	1	0

0	-1	0
-1	4	-1
0	-1	0

# Effect of Laplacian Operator

- **as it is a derivative operator,**
  - it highlights gray-level discontinuities in an image
  - it deemphasizes regions with slowly varying gray levels
- **tends to produce images that have**
  - grayish edge lines and other discontinuities, all superimposed on a dark,
  - featureless background.



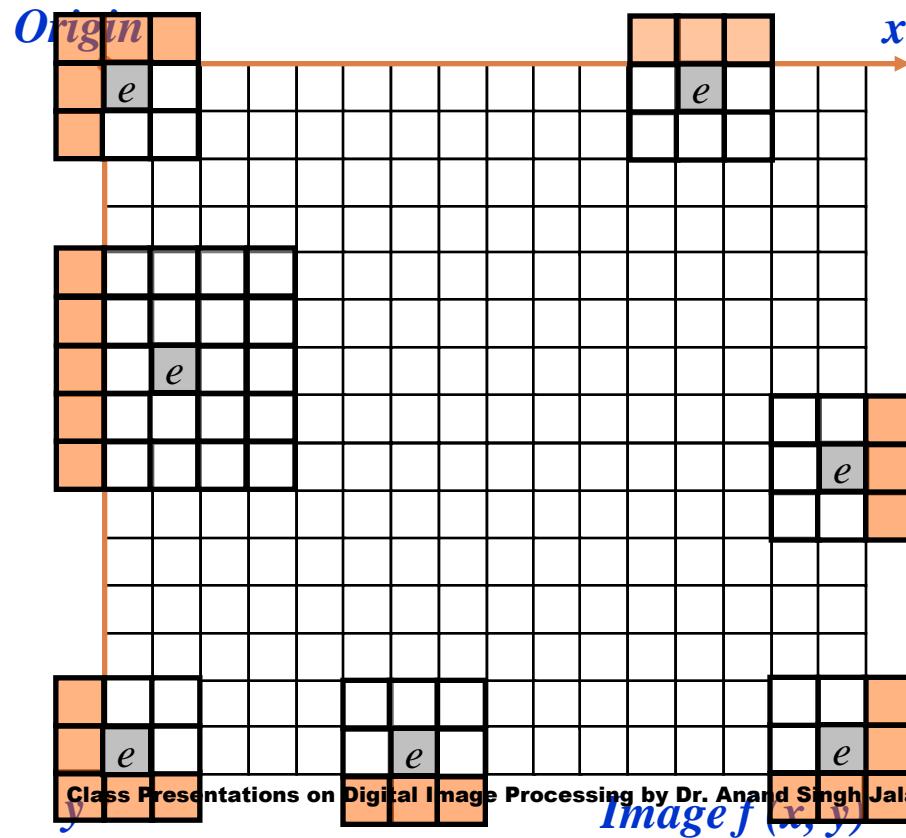
Image of the north  
pole of the moon

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



# Strange Things Happen At The Edges!

At the edges of an image we are missing pixels to form a neighbourhood



# Strange Things Happen At The Edges ...

- There are a few approaches to dealing with missing edge pixels:
  - Omit missing pixels
    - Only works with some filters
    - Can add extra code and slow down processing
  - Pad the image
    - Typically with either all white or all black pixels
  - Replicate border pixels



*Any Questions ?*