

## Digital Image Processing

### Unit I Fundamentals

## What is an image?

*An image is a two-dimensional function  $f(x,y)$ , where  $x$  and  $y$  are the spatial (plane) coordinates, and the amplitude of  $f$  at any pair of coordinates  $(x,y)$  is called the intensity of the image at that level.*

## What is a digital image?

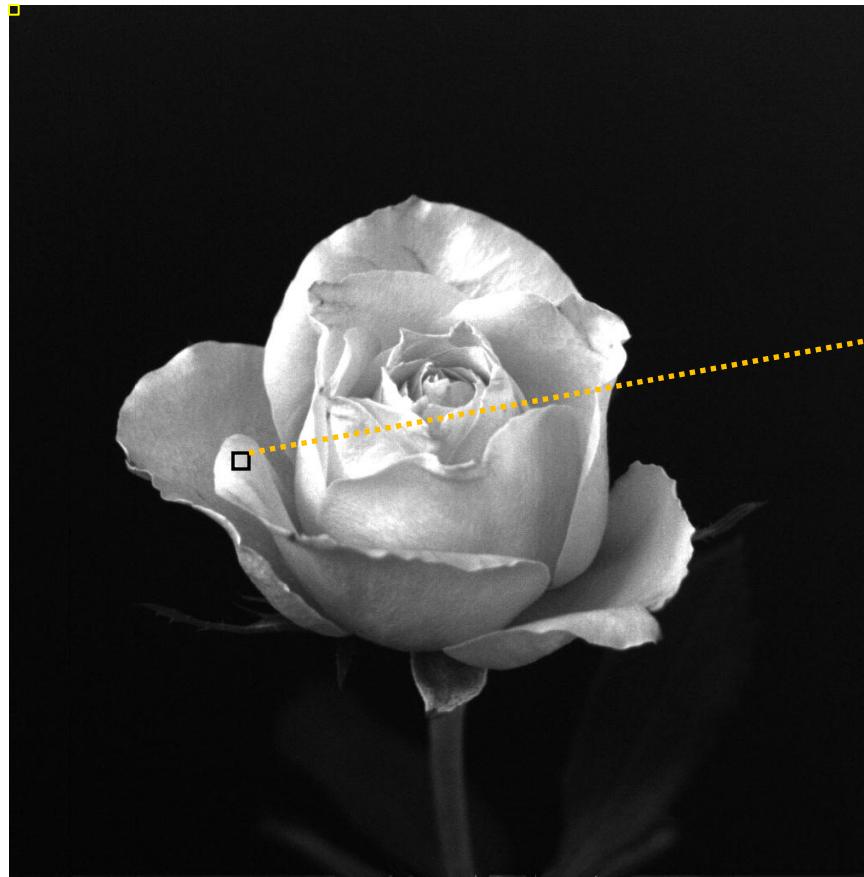
*If  $x,y$  and the amplitude values of  $f$  are finite and discrete quantities, we call the image a digital image. A digital image is composed of a finite number of elements called pixels, each of which has a particular location and value.*



*First Digital Photograph Ever: Russell Kirsch in 1957 made a 176×176 pixel digital image by scanning a photograph of his three-month-old son.*

In 8-bit representation  
Pixel intensity values change between 0 (Black) and 255 (White)

Pixel location  
Pixel intensity value  
 $f(1,1) = 21$



Consider the following image ( $1024 \times 1024$  pixels) to be 2D function or a **matrix** with **rows** and **columns**

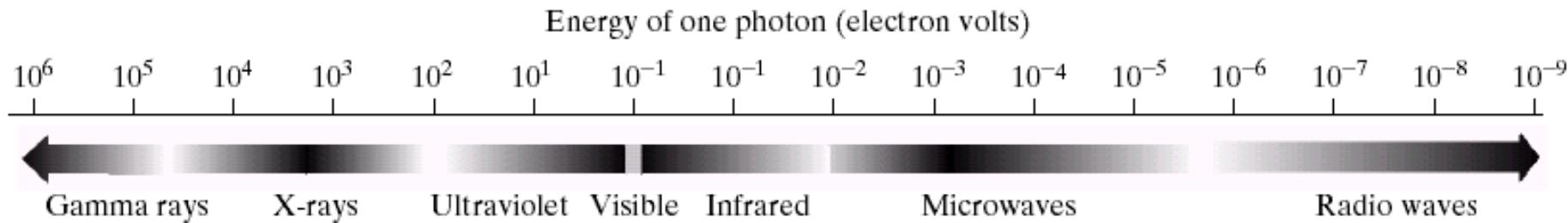
rows      columns  
 $f(520:525,375:380) =$   
152 148 144 152 181 203  
144 138 156 152 184 208  
141 141 138 156 181 203  
136 138 144 158 177 196  
144 138 148 154 177 208  
149 138 152 160 188 205  
 $f(1024,1024) = 15$

# What is Digital Image Processing?

**Definition:** *Digital image processing refers to processing of digital images by using digital computers.*

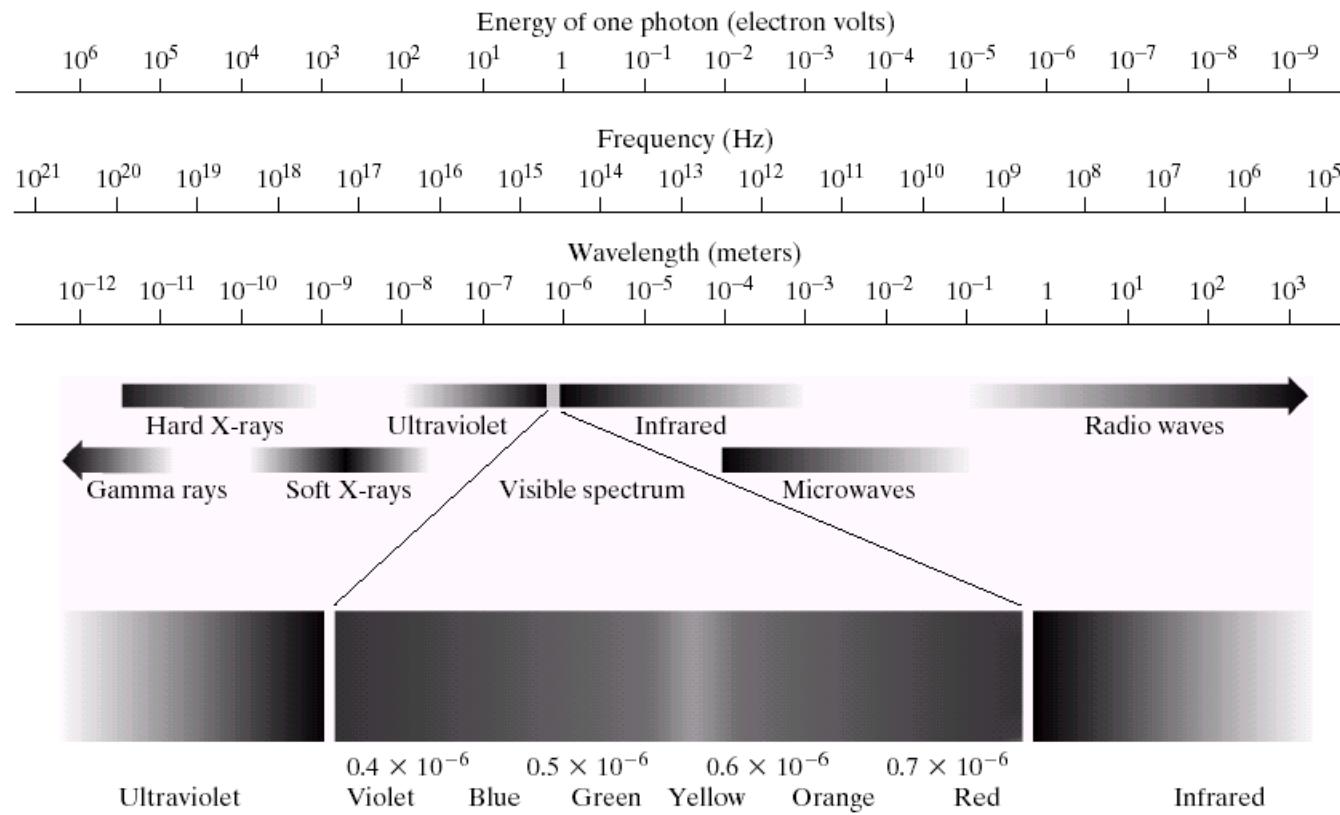
## Sources of Digital Images

- *The principal source for the images is the electromagnetic (EM) energy spectrum.*
- *The spectral bands are grouped according to energy per photon ranging from the gamma rays (highest energy) to the radio waves (lowest energy).*



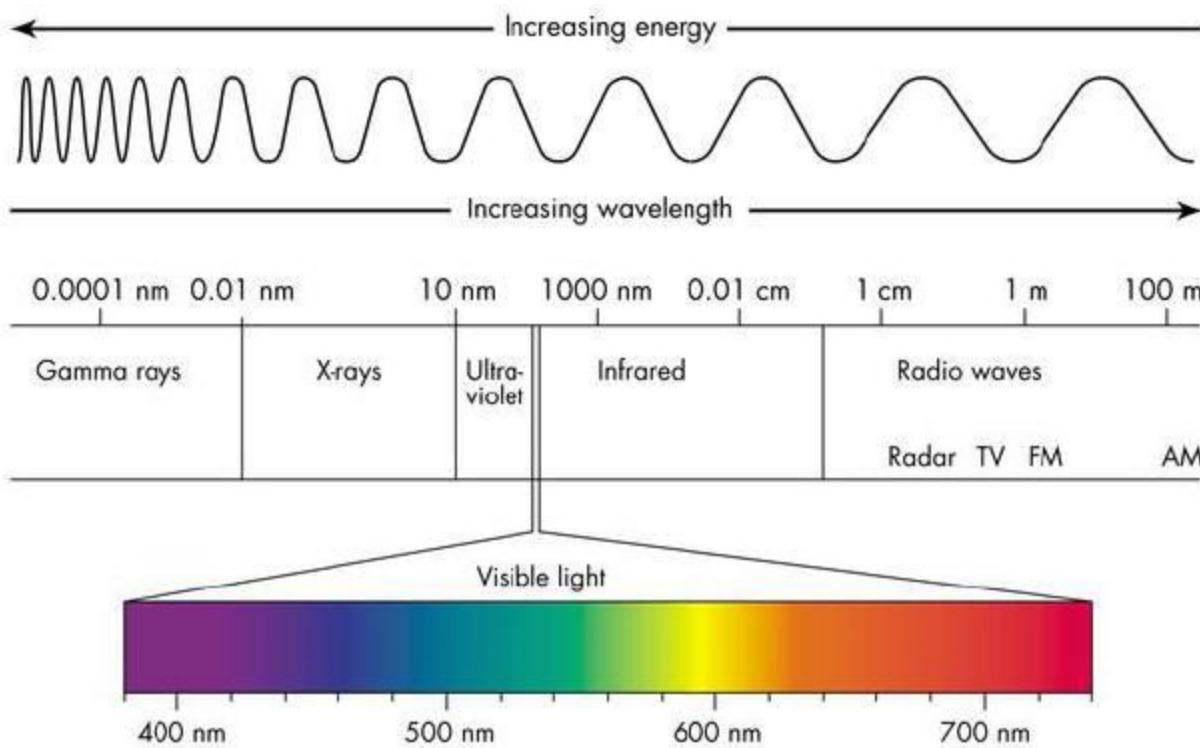
**FIGURE 1.5** The electromagnetic spectrum arranged according to energy per photon.

# The Electromagnetic Spectrum



**FIGURE 2.10** The electromagnetic spectrum. The visible spectrum is shown zoomed to facilitate explanation, but note that the visible spectrum is a rather narrow portion of the EM spectrum.

# The Electromagnetic Spectrum

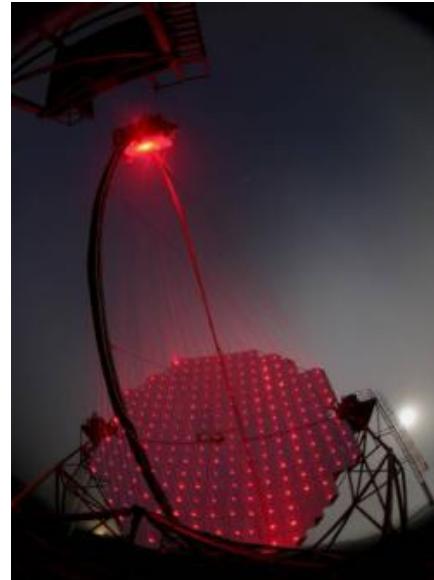


# Digital Images based on the EM Spectrum

**Gamma Ray Imaging:** Used in *nuclear medicine* and *astronomical observations*.



Gamma-Ray Imaging  
in nuclear medicine



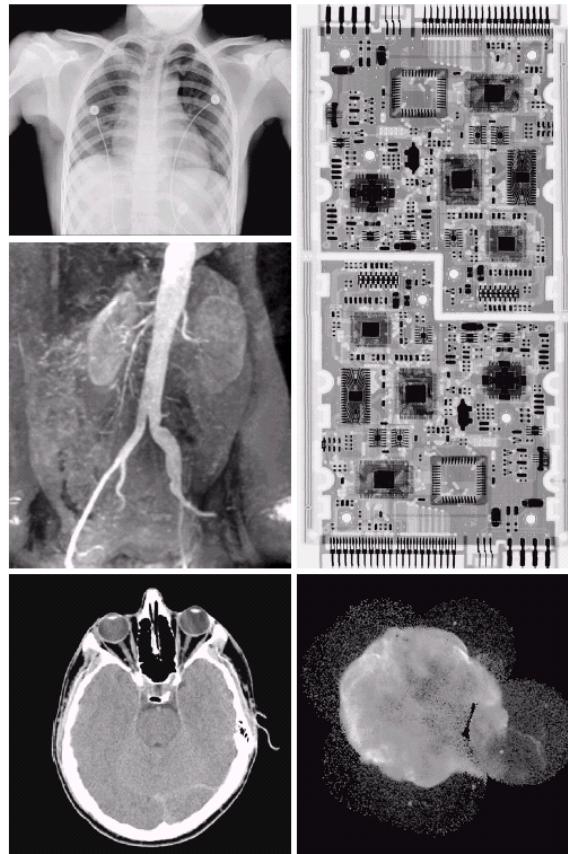
Gamma-Ray Imaging  
Cherenkov Telescope



Gamma-Ray imaging of  
A starburst galaxy about 12  
million light-years away

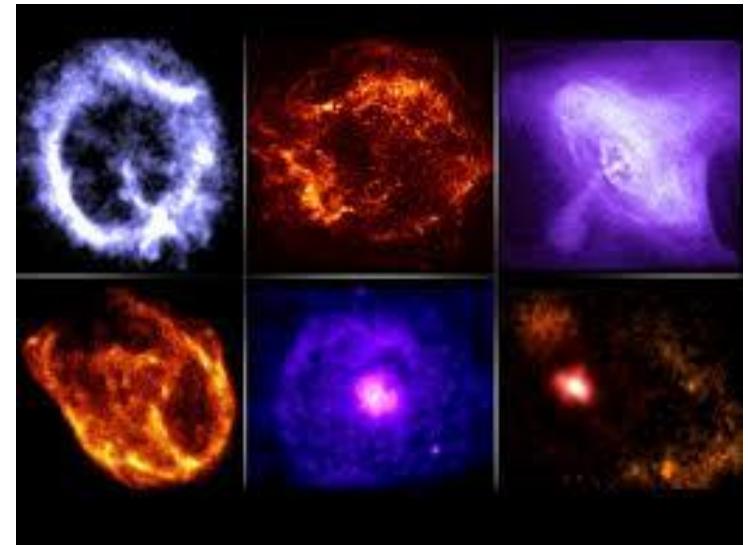
# Digital Images based on the EM Spectrum

X-ray Imaging: Used in *medical diagnostic, industrial applications and astronomy.*



a  
b  
c  
d  
e

**FIGURE 1.7** Examples of X-ray imaging. (a) Chest X-ray. (b) Aortic angiogram. (c) Head CT. (d) Circuit boards. (e) Cygnus Loop. (Images courtesy of (a) and (c) Dr. David R. Pickens, Dept. of Radiology & Radiological Sciences, Vanderbilt University Medical Center, (b) Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School, (d) Mr. Joseph E. Pascente, Lixi, Inc., and (e) NASA.)



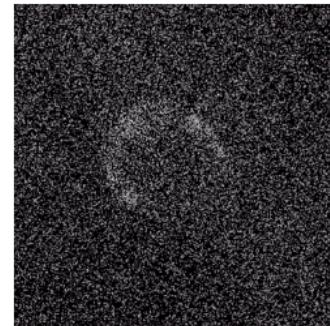
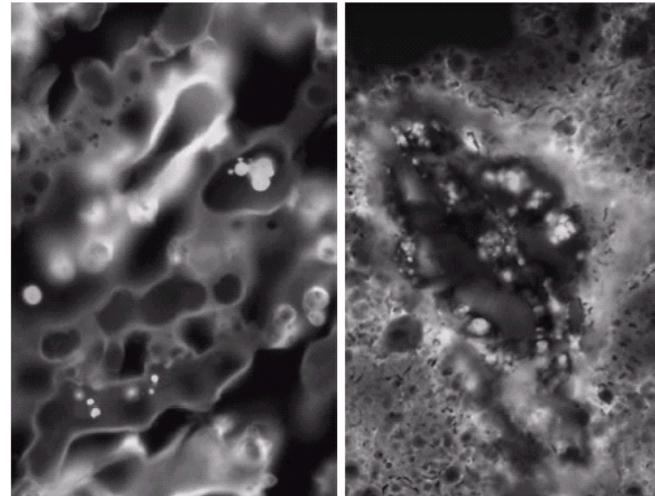
**X-ray images from the space  
The Chandra X-Ray Observatory**

# Digital Images based on the EM Spectrum

**Ultraviolet Band Imaging:** Applications of ultraviolet light includes microscopy, lasers, biological imaging and astronomy.

a  
b  
c

**FIGURE 1.8**  
Examples of ultraviolet imaging.  
(a) Normal corn.  
(b) Smut corn.  
(c) Cygnus Loop.  
(Images courtesy of (a) and (b) Dr. Michael W. Davidson, Florida State University, (c) NASA.)



# Digital Images based on the EM Spectrum

**Visible Light** and **Infrared Band Imaging**: Applications include all the images acquired by our cameras, electron microscope, and monitoring environmental conditions.



Binary image  
(1-bit)



Grayscale  
(Monochrome) image  
(8-bit)



Color image  
(24-bit)



R



B



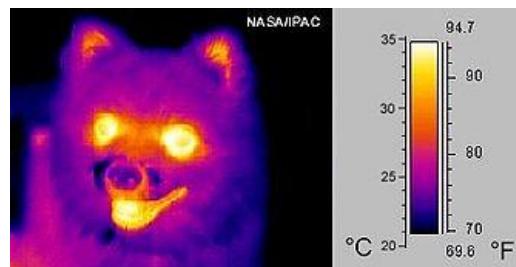
Color Image  
Respective **RGB** components  
of a color image.



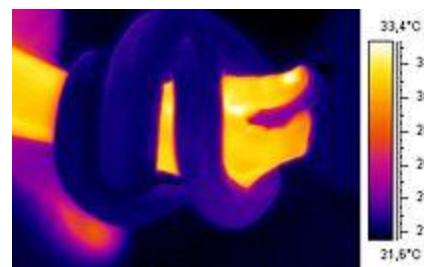
Some visible light image examples

# Digital Images based on the EM Spectrum

Visible Light and *Infrared Band Imaging*: Applications include all the images acquired by our cameras, electron microscope, and monitoring environmental conditions.



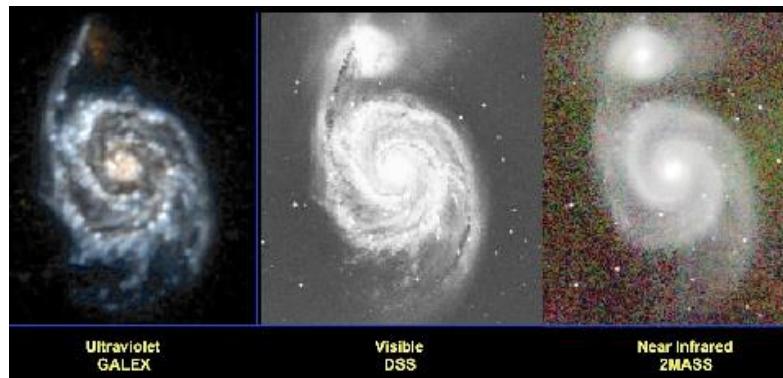
infrared ("thermal") image



Snake around the arm.



A soldier with a rifle.



Messier 51 in ultraviolet (GALEX), visible (DSS), and near infrared (2MASS).  
Courtesy of James Fanson.

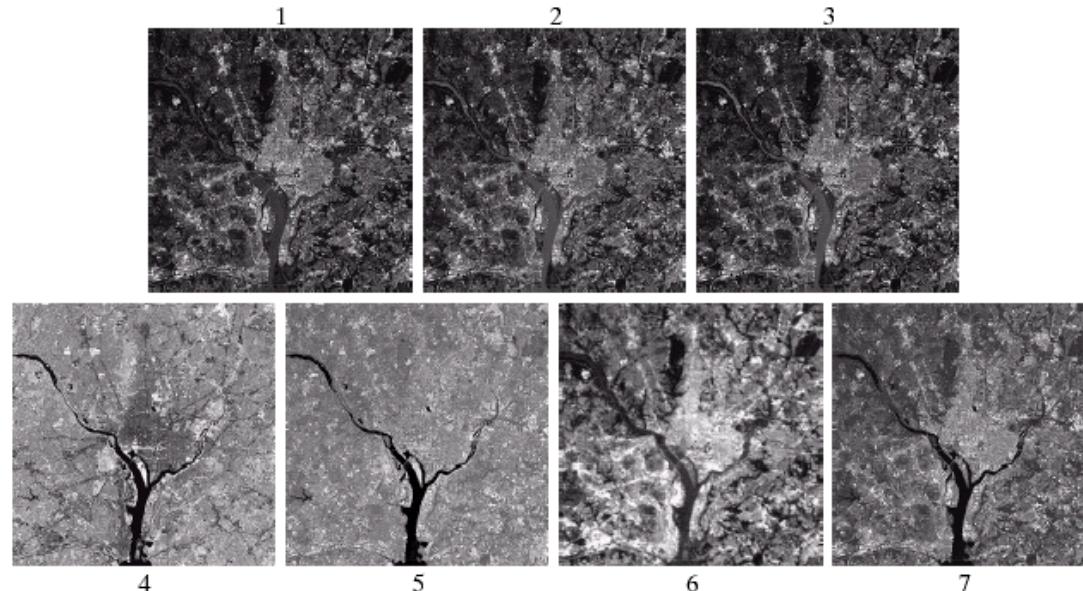
# Digital Images based on the EM Spectrum

## Visible Light and Infrared Band Imaging

**TABLE 1.1**  
Thematic bands  
in NASA's  
LANDSAT  
satellite.

Band No.	Name	Wavelength ( $\mu\text{m}$ )	Characteristics and Uses
1	Visible blue	0.45–0.52	Maximum water penetration
2	Visible green	0.52–0.60	Good for measuring plant vigor
3	Visible red	0.63–0.69	Vegetation discrimination
4	Near infrared	0.76–0.90	Biomass and shoreline mapping
5	Middle infrared	1.55–1.75	Moisture content of soil and vegetation
6	Thermal infrared	10.4–12.5	Soil moisture; thermal mapping
7	Middle infrared	2.08–2.35	Mineral mapping

**Visible and Infrared Bands  
used in Satellite imaging**

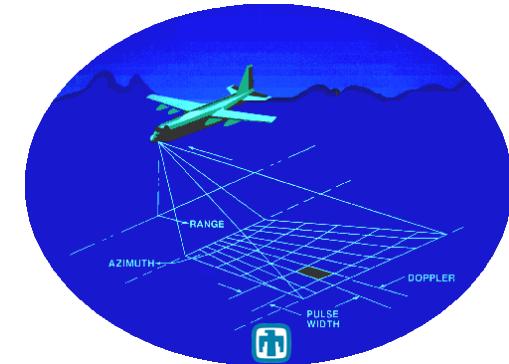
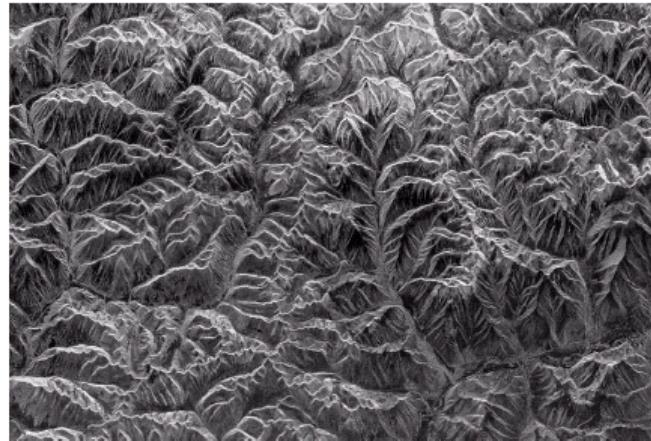


**FIGURE 1.10** LANDSAT satellite images of the Washington, D.C. area. The numbers refer to the thematic bands in Table 1.1. (Images courtesy of NASA.)

# Digital Images based on the EM Spectrum

**Microwave Band Imaging**: Applications include all the **radar** applications including **military applications** and **environmental applications**.

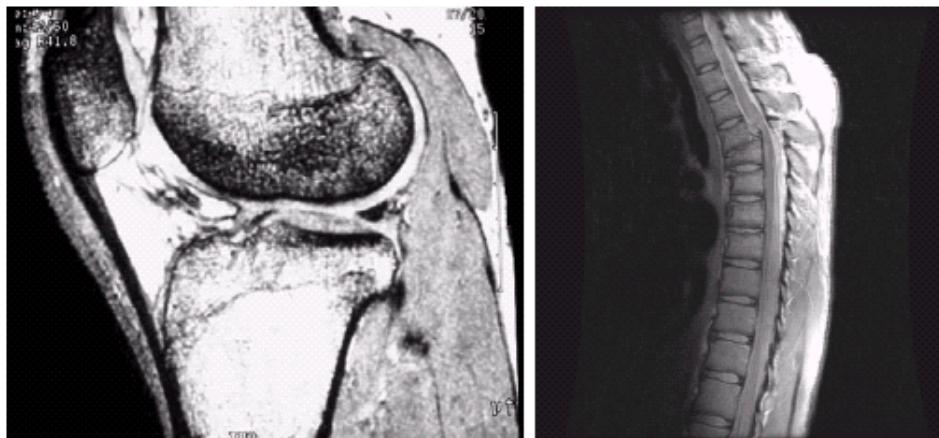
**FIGURE 1.16**  
Spaceborne radar  
image of  
mountains in  
southeast Tibet.  
(Courtesy of  
NASA.)



**Synthetic Aperture Radar System**

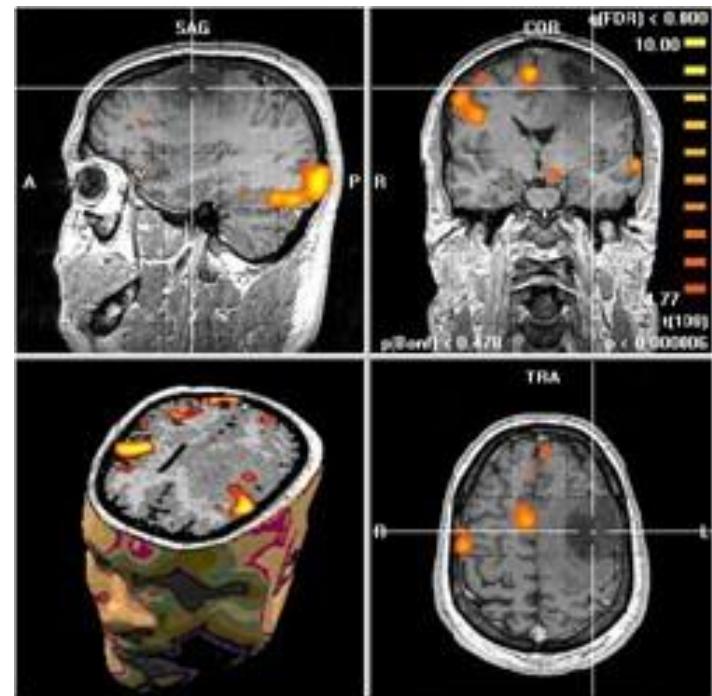
# Digital Images based on the EM Spectrum

**Radio Band Imaging** : Applications include medical imaging (i.e. Magnetic Resonance Imaging- MRI) and astronomy.



a b

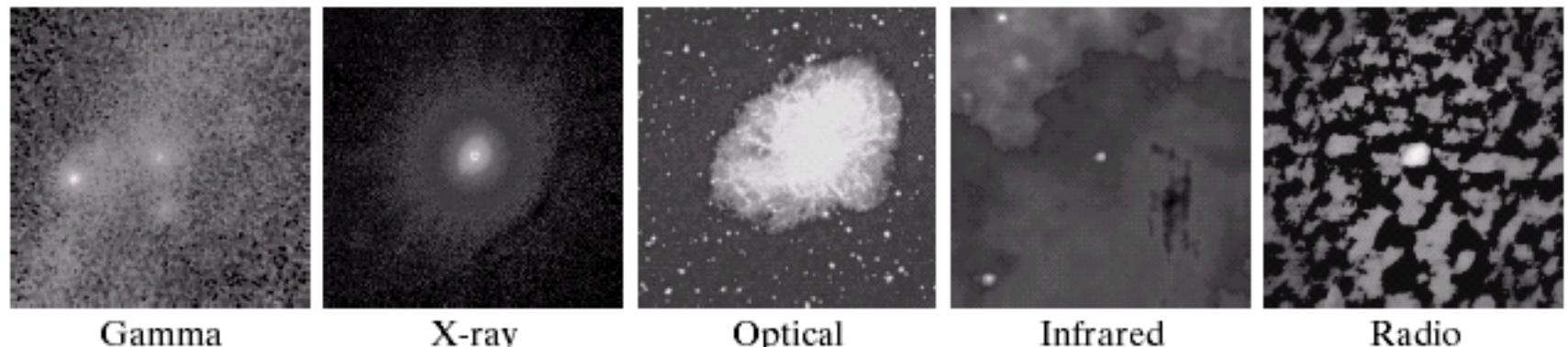
**FIGURE 1.17** MRI images of a human (a) knee, and (b) spine. (Image (a) courtesy of Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School, and (b) Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)



MRI image slices from the brain

# Digital Images based on the EM Spectrum

An example showing Imaging in all of the bands



Gamma

X-ray

Optical

Infrared

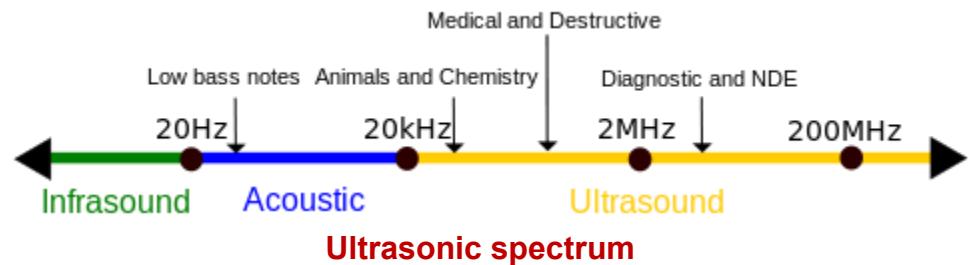
Radio

**FIGURE 1.18** Images of the Crab Pulsar (in the center of images) covering the electromagnetic spectrum.  
(Courtesy of NASA.)

Visible light

# Digital Images based on the Ultrasound

**Ultrasound Imaging:** Ultrasound is a cyclic sound pressure wave with a frequency greater than the upper limit of human hearing. The most well-known application of ultrasound is its use in **sonography** to produce pictures of fetuses in the human womb.



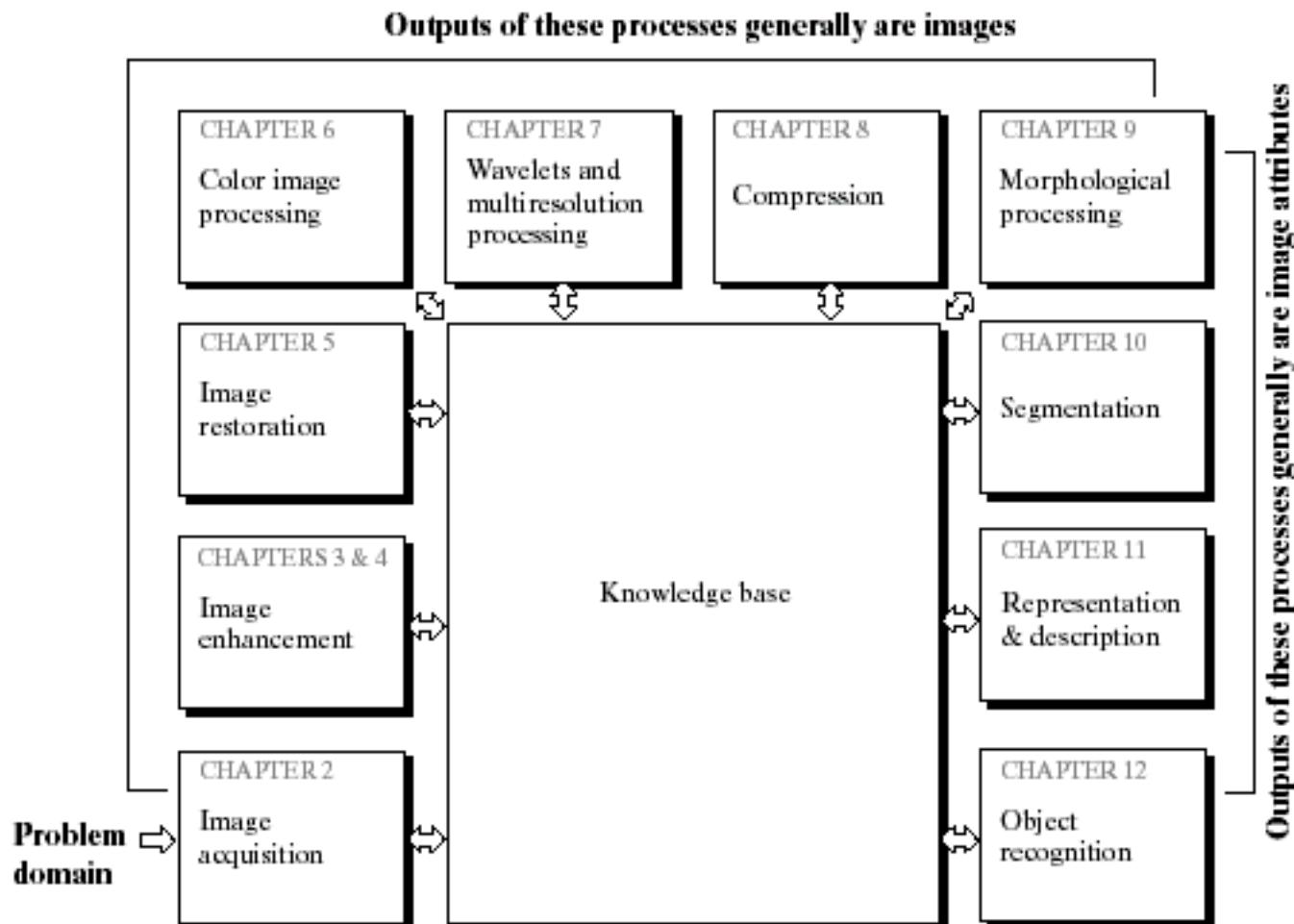
Ultrasound image acquisition device



Ultrasonic Baby image during pregnancy

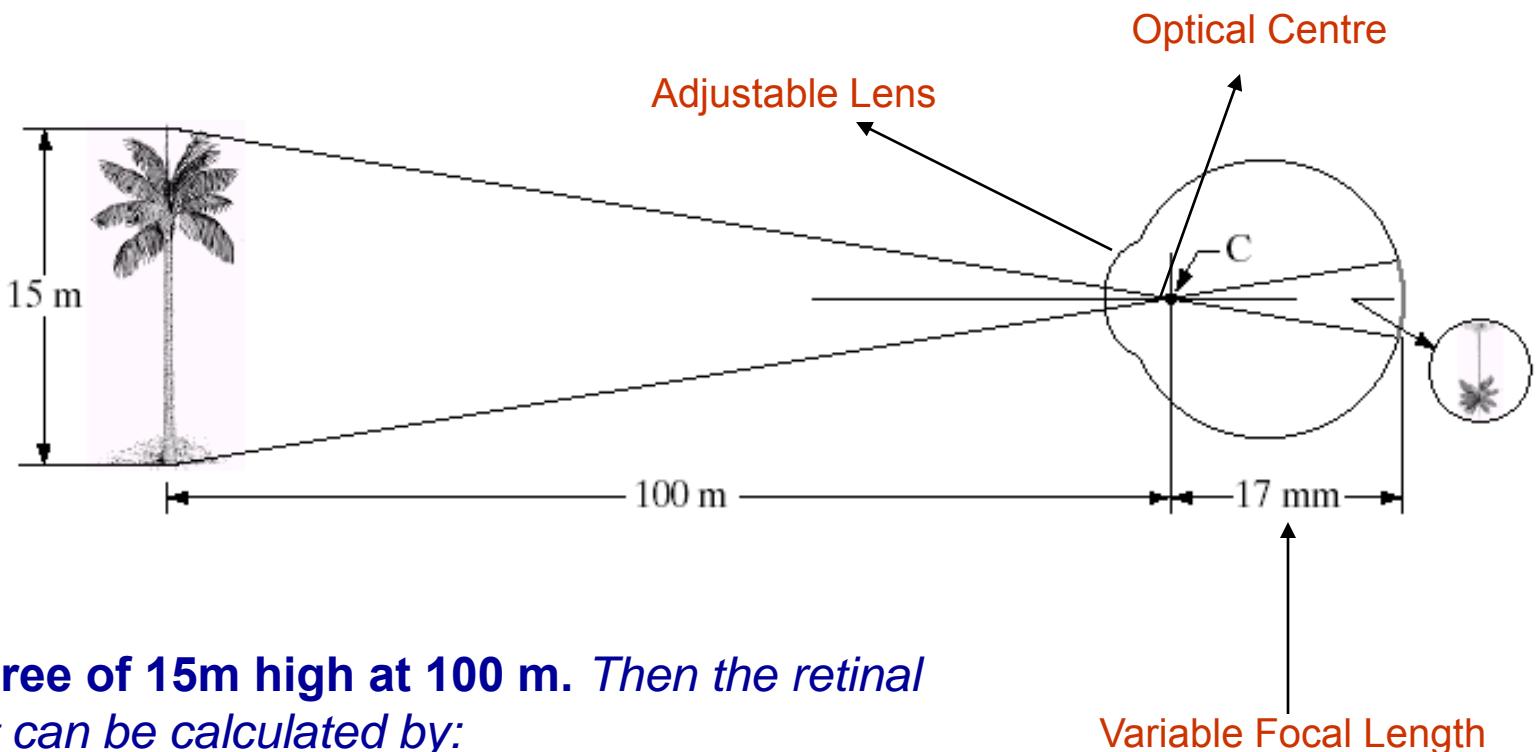
# Fundamental Steps in Image Processing

**FIGURE 1.23**  
Fundamental  
steps in digital  
image processing.



# The Human Eye & Image Formation

**FIGURE 2.3**  
Graphical representation of the eye looking at a palm tree. Point C is the optical center of the lens.



Consider a tree of 15m high at 100 m. Then the retinal image height can be calculated by:

$$\frac{15}{100} = \frac{h}{17} \Rightarrow h = 2.55\text{mm}$$

---

# Acquisition of Images.

The images are generated by the combination of an *illumination source* and the reflection or absorption of energy from that source by the elements of the *scene* being imaged.

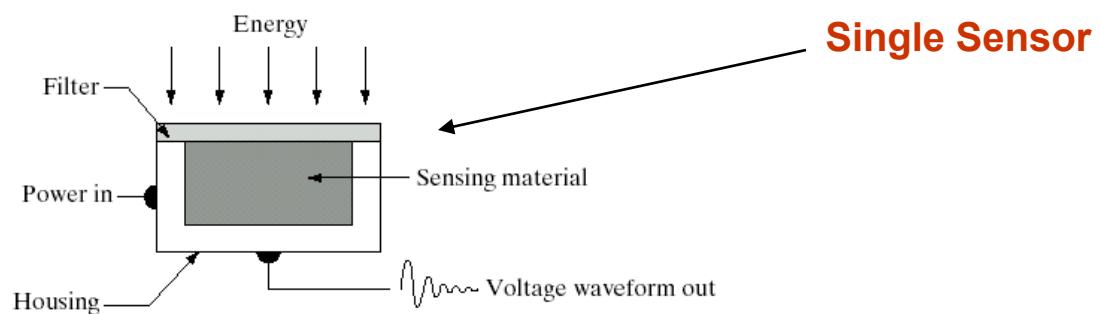
*Imaging sensors* are used to transform the *illumination energy* into digital images.

Each *sensor* transforms the incoming energy into *voltage* by the combination of the input electrical power and the sensor material that is responsive to the particular type of energy being detected.

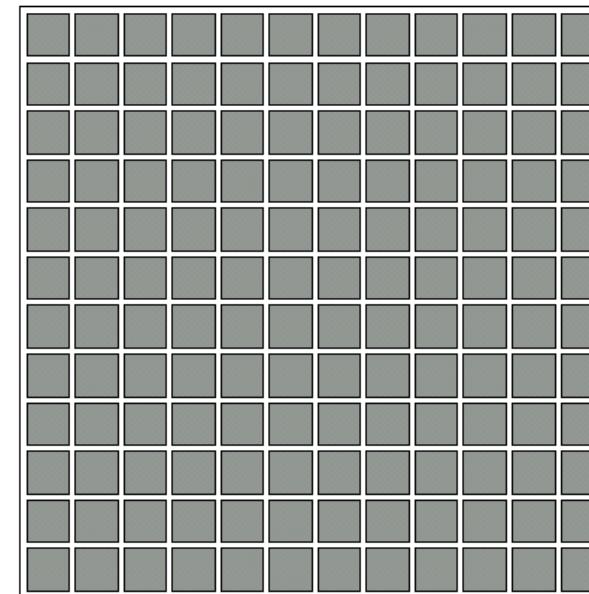
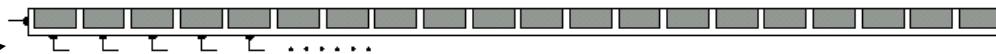
# Types of Image Sensors

a  
b  
c

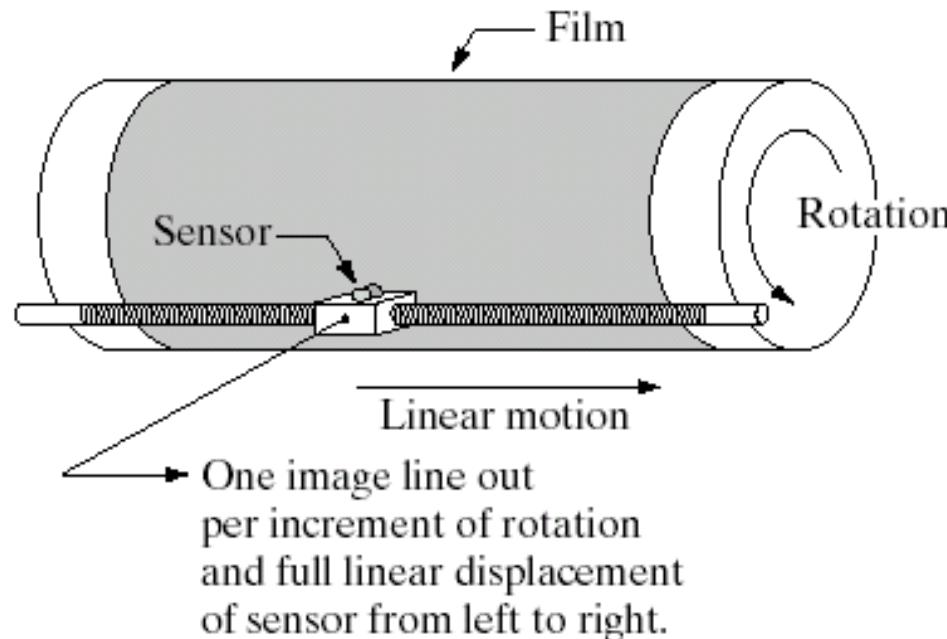
**FIGURE 2.12**  
(a) Single imaging sensor.  
(b) Line sensor.  
(c) Array sensor.



**Line Sensor**

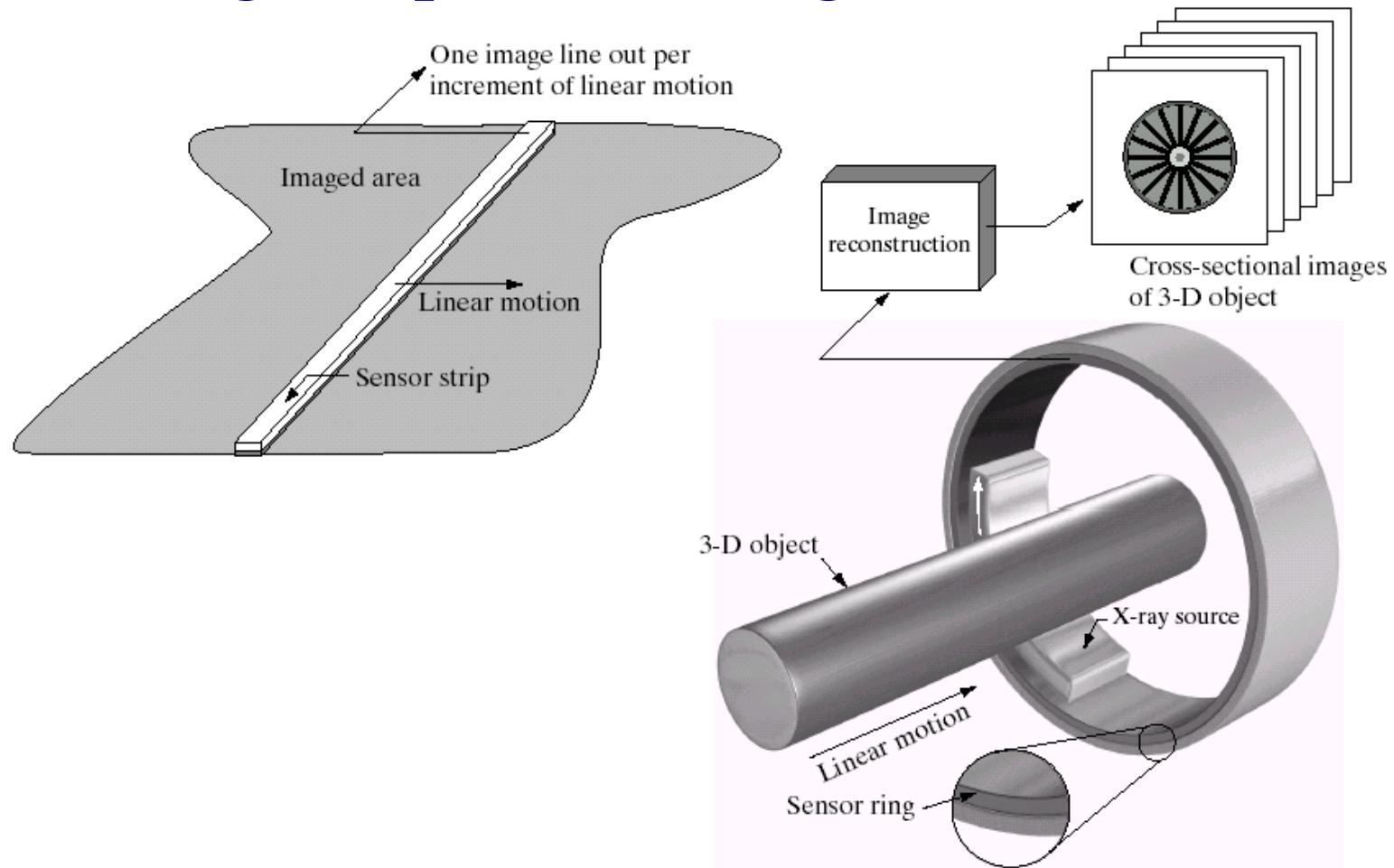


## Image Acquisition using Single Sensor



**FIGURE 2.13** Combining a single sensor with motion to generate a 2-D image.

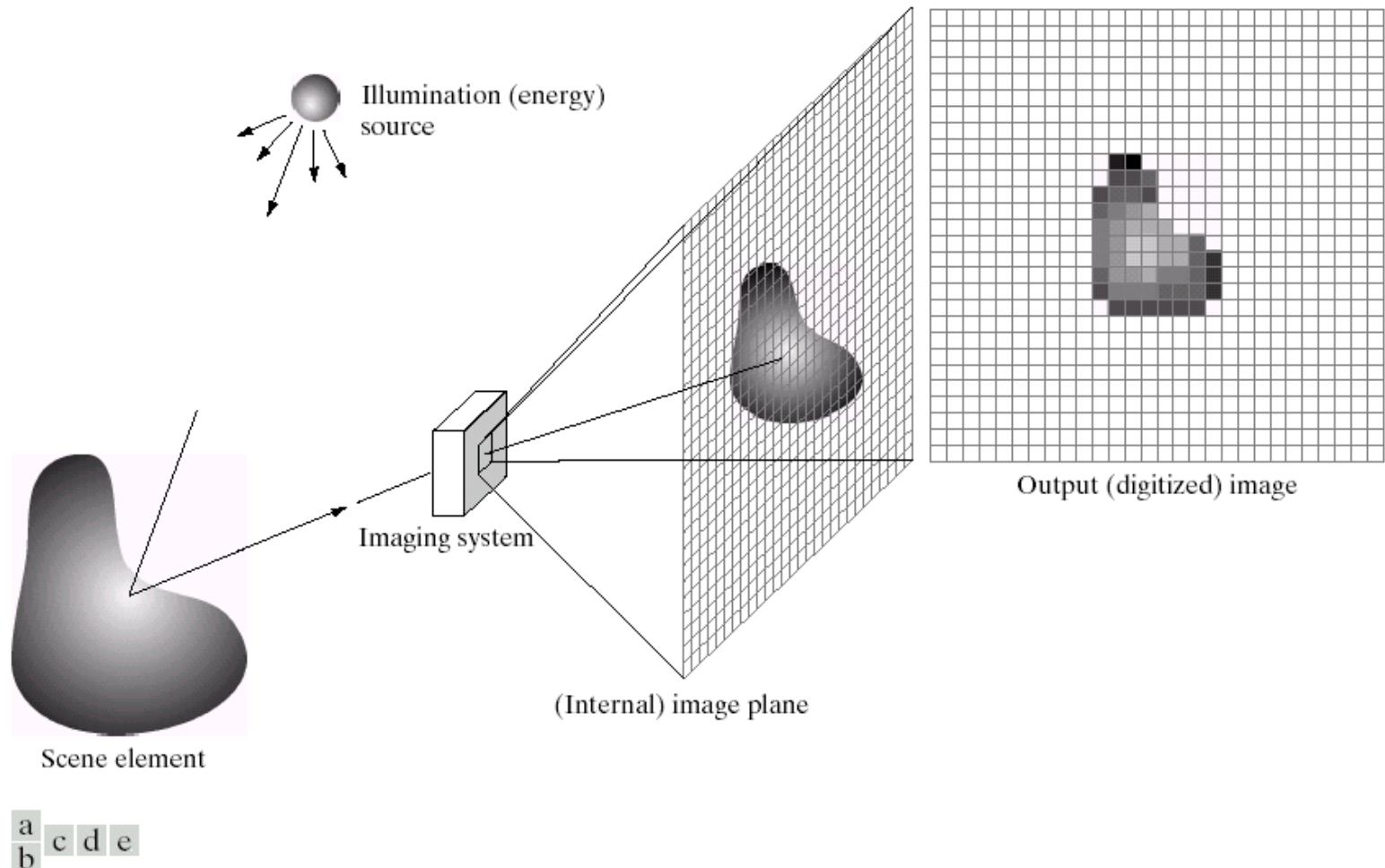
# Image Acquisition using Line Sensor



a b

**FIGURE 2.14** (a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

# Image Acquisition using Sensor Array



**FIGURE 2.15** An example of the digital image acquisition process. (a) Energy (“illumination”) source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

---

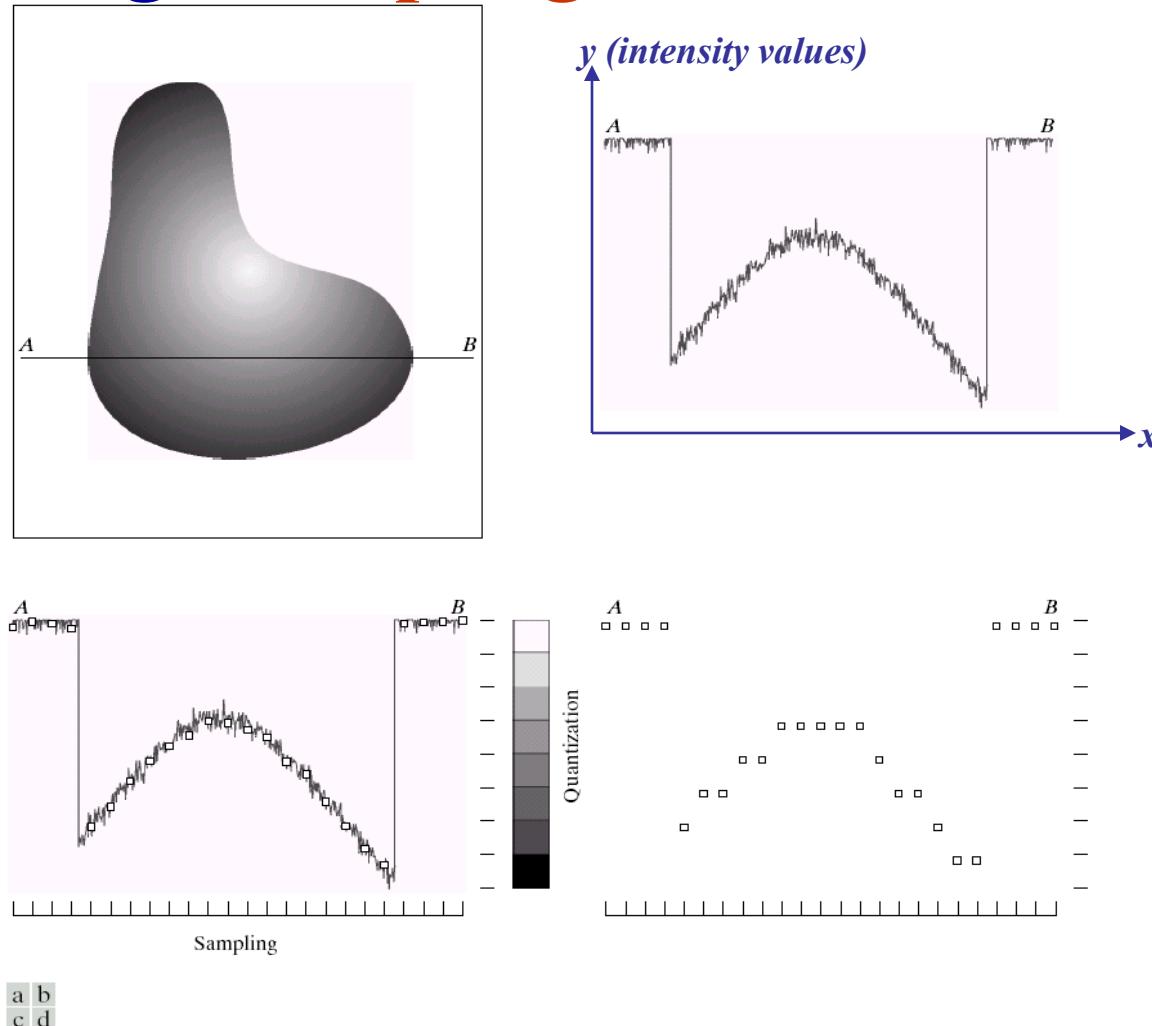
# Image Sampling and Quantization

*A digital image can be obtained by converting a continuous/analog image in a digital form by:*

*Sampling and  
Quantization.*

*Given a continuous image,  $f(x,y)$ , digitizing the coordinate values is called sampling and digitizing the amplitude (intensity) values is called quantization.*

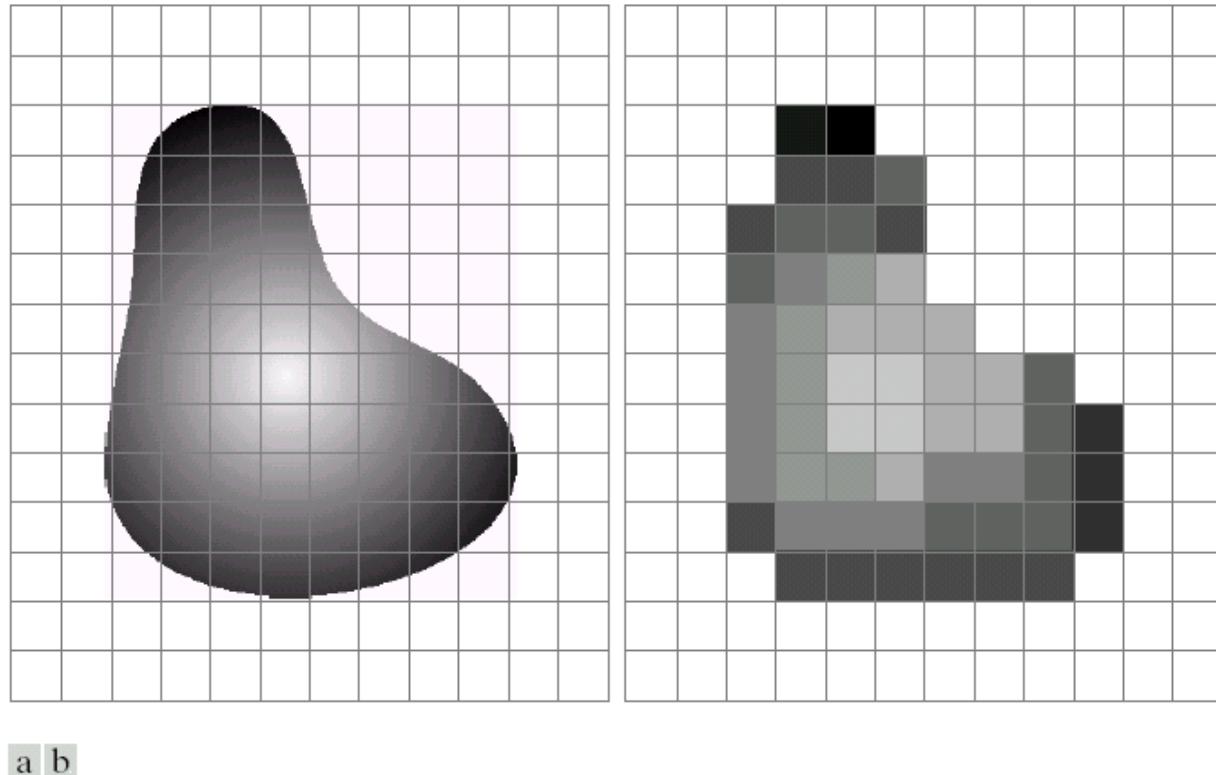
# Image Sampling and Quantization



**FIGURE 2.16** Generating a digital image. (a) Continuous image. (b) A scan line from *A* to *B* in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

---

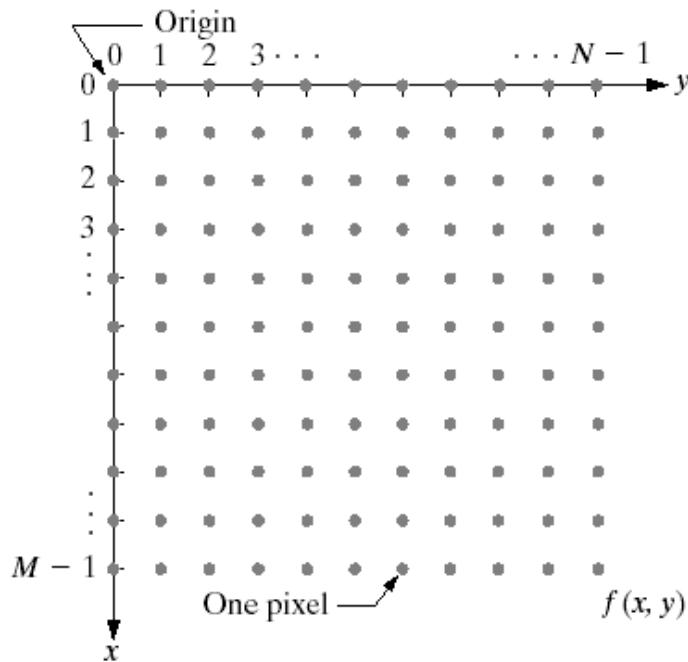
# Image Sampling and Quantization



**FIGURE 2.17** (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.

---

# Matrix Representation of Images



A matrix representing an  $N \times M$  image.

**FIGURE 2.18**

Coordinate convention used in this book to represent digital images.

- $M$  and  $N$  can be any positive integers.
- The number of gray levels,  $L$ , is an integer power of 2.
- Number of bits required to store a digitized image:  
$$b = N \times M \times k$$

# Number of bits used to represent an image

Assume that  $M=N$ . Therefore  $b=N^2 k$

**TABLE 2.1**

Number of storage bits for various values of  $N$  and  $k$ .

$N/k$	1 ( $L = 2$ )	2 ( $L = 4$ )	3 ( $L = 8$ )	4 ( $L = 16$ )	5 ( $L = 32$ )	6 ( $L = 64$ )	7 ( $L = 128$ )	8 ( $L = 256$ )
32	1,024	2,048	3,072	4,096	5,120	6,144	7,168	8,192
64	4,096	8,192	12,288	16,384	20,480	24,576	28,672	32,768
128	16,384	32,768	49,152	65,536	81,920	98,304	114,688	131,072
256	65,536	131,072	196,608	262,144	327,680	393,216	458,752	524,288
512	262,144	524,288	786,432	1,048,576	1,310,720	1,572,864	1,835,008	2,097,152
1024	1,048,576	2,097,152	3,145,728	4,194,304	5,242,880	6,291,456	7,340,032	8,388,608
2048	4,194,304	8,388,608	12,582,912	16,777,216	20,971,520	25,165,824	29,369,128	33,554,432
4096	16,777,216	33,554,432	50,331,648	67,108,864	83,886,080	100,663,296	117,440,512	134,217,728
8192	67,108,864	134,217,728	201,326,592	268,435,456	335,544,320	402,653,184	469,762,048	536,870,912

# Sampling and Spatial Resolution

*Sampling is the principal factor determining the spatial resolution of an image.*



1024



512



256



128

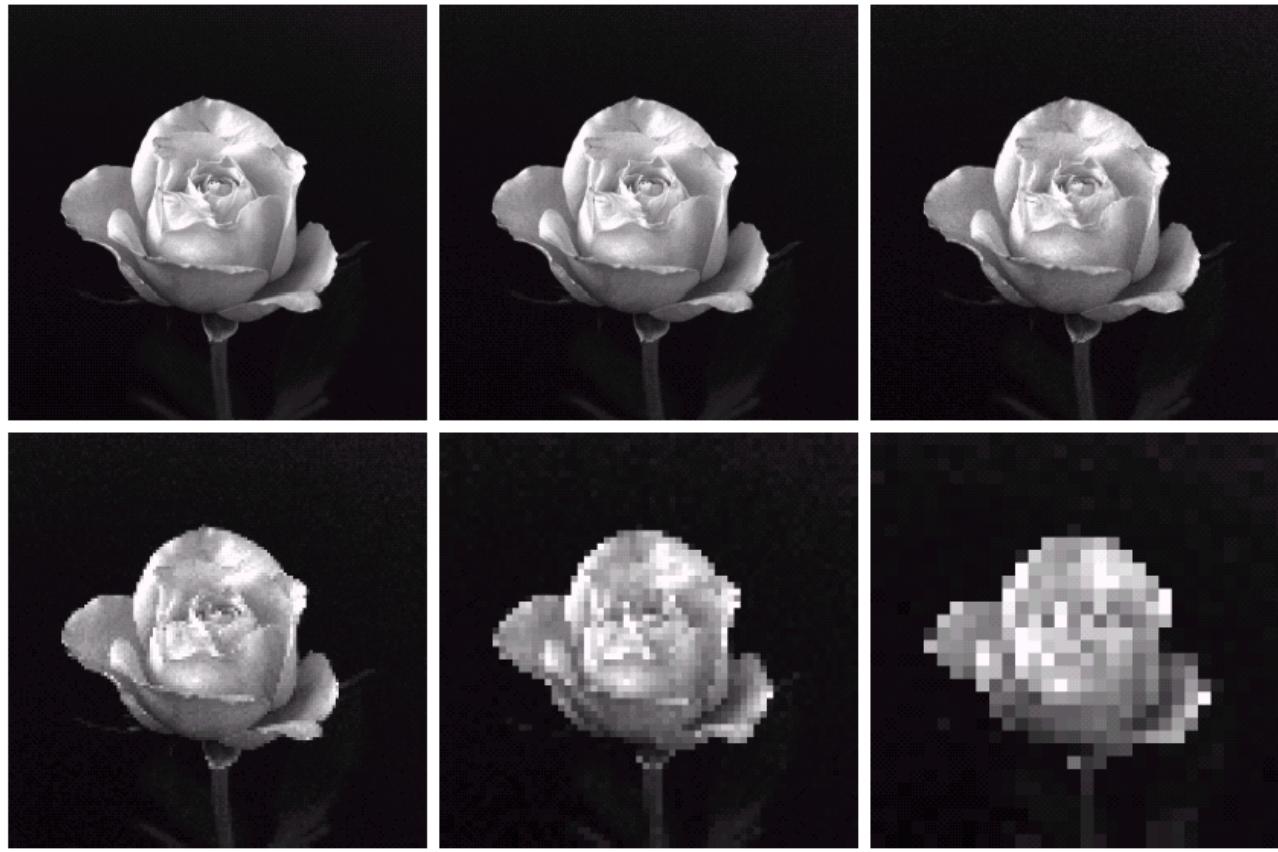
32

*Sampling determines the number of pixels of a digitized image.*

**FIGURE 2.19** A  $1024 \times 1024$ , 8-bit image subsampled down to size  $32 \times 32$  pixels. The number of allowable gray levels was kept at 256.

---

# Sampling and Spatial Resolution



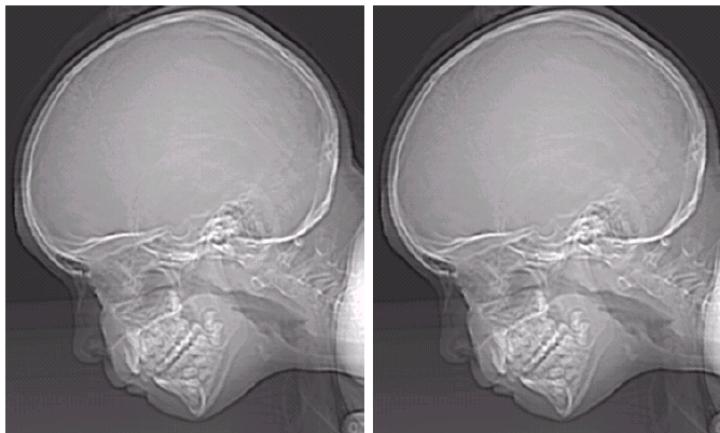
a b c  
d e f

**FIGURE 2.20** (a)  $1024 \times 1024$ , 8-bit image. (b)  $512 \times 512$  image resampled into  $1024 \times 1024$  pixels by row and column duplication. (c) through (f)  $256 \times 256$ ,  $128 \times 128$ ,  $64 \times 64$ , and  $32 \times 32$  images resampled into  $1024 \times 1024$  pixels.

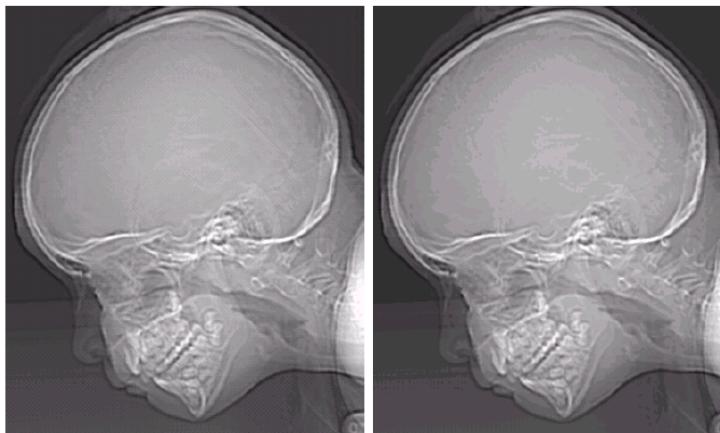
# Quantization and Gray-level Resolution

*Quantization is the most important factor determining the gray-level resolution of an image.*

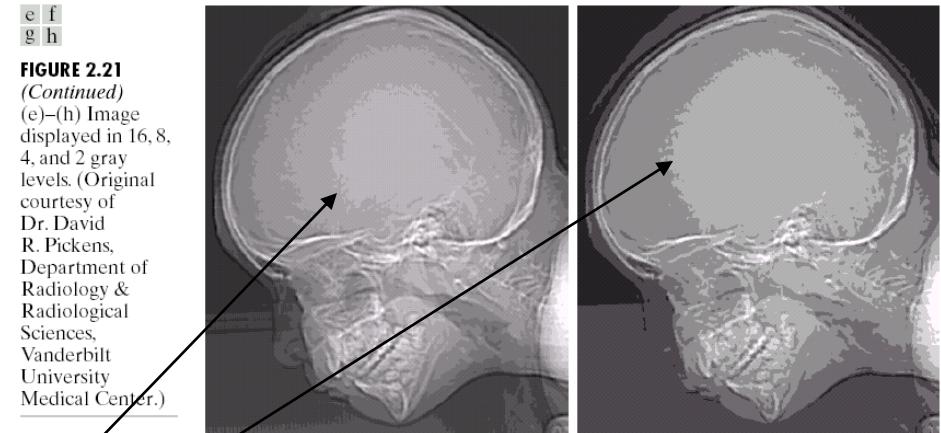
*Quantization determines the number of gray levels that each pixel can take.*



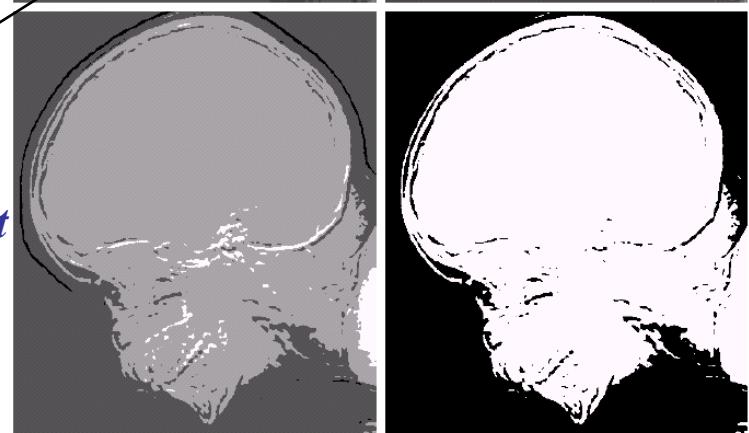
**FIGURE 2.21**  
(a) 452 × 374,  
256-level image.  
(b)–(d) Image  
displayed in 128,  
64, and 32 gray  
levels, while  
keeping the  
spatial resolution  
constant.



*False contouring effect  
is visible in 16 and less  
gray level images.*



**FIGURE 2.21**  
(Continued)  
(e)–(h) Image  
displayed in 16, 8,  
4, and 2 gray  
levels. (Original  
courtesy of  
Dr. David  
R. Pickens,  
Department of  
Radiology &  
Radiological  
Sciences,  
Vanderbilt  
University  
Medical Center.)



# Resizing Images: Zooming and Shrinking

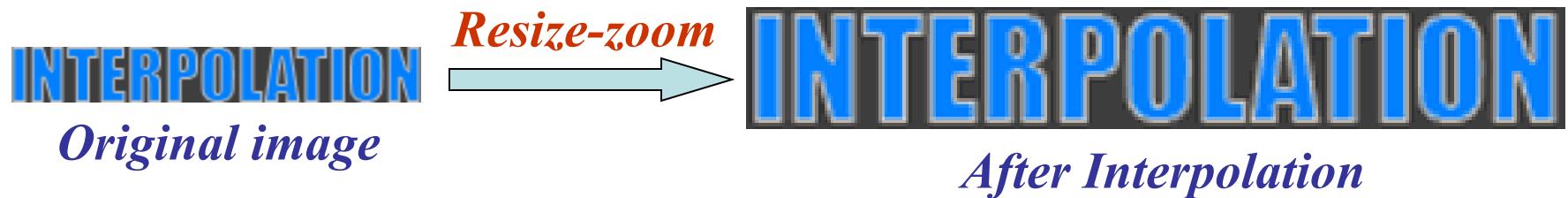
*Zooming is a method of increasing the size of a given image.*

*Zooming can be viewed as **oversampling** or **upsampling** of a given image.*

*Zooming requires **2 steps**:*

- *Creation of new pixel locations*

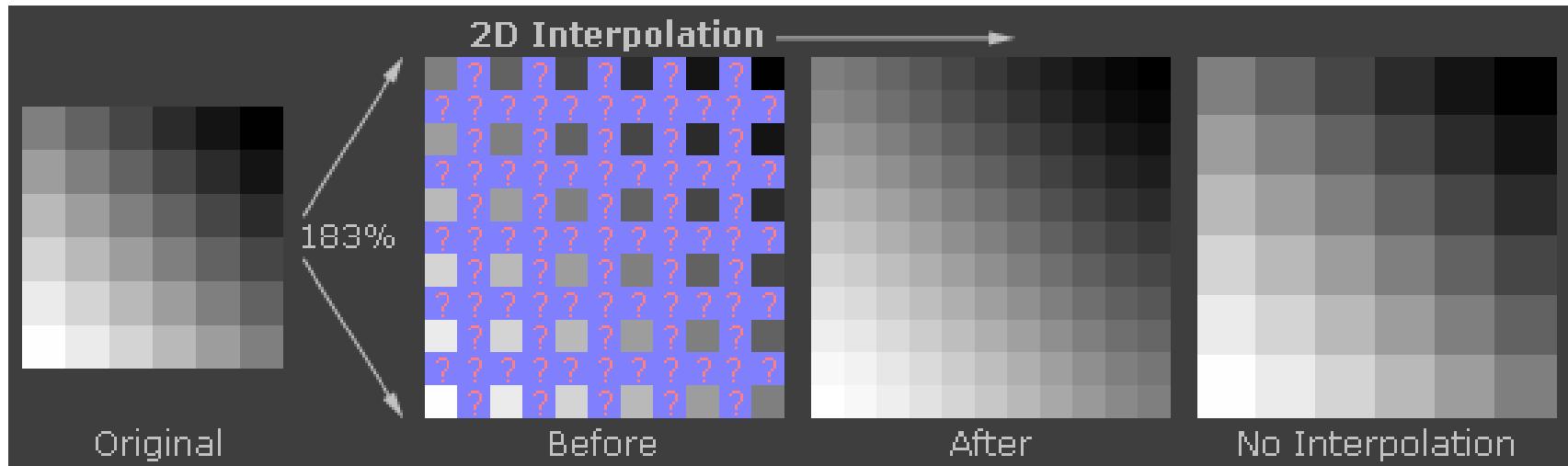
- *Assigning new gray-level values to these locations by using **interpolation**.*



*Interpolation is defined to be the estimation of the value of unknown point by using the values of known points.*

# Resizing Images: Zooming and Shrinking

## 2-D Interpolation



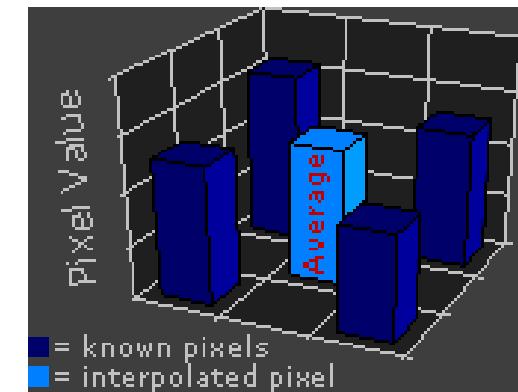
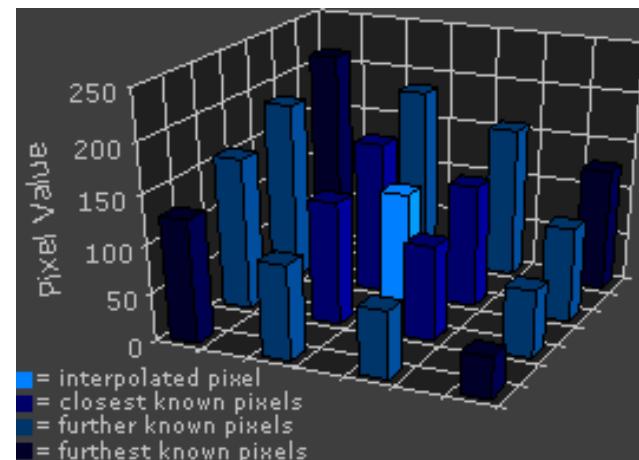
*There are 3 main types of 2-D Interpolation techniques for zooming:*

- nearest neighbor interpolation
- bilinear interpolation
- bicubic interpolation

# Zooming : Interpolation Techniques

**Nearest neighbor interpolation:** Nearest neighbor interpolation is the simplest method and basically makes the pixels bigger. The intensity of a pixel in the new image is the intensity of the nearest pixel of the original image. If you enlarge 200%, one pixel will be enlarged to a  $2 \times 2$  area of 4 pixels with the same color as the original pixel.

**Bilinear interpolation:** Bilinear interpolation considers the closest  $2 \times 2$  neighborhood of known pixel values surrounding the unknown pixel. It then takes a weighted average of these 4 pixels to arrive at its final interpolated value. This results in much smoother looking images than nearest neighbor.



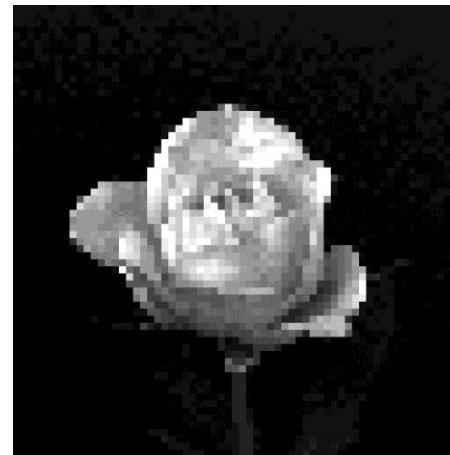
**Bicubic interpolation:** Bicubic goes one step beyond bilinear by considering the closest  $4 \times 4$  neighborhood of known pixels- for a total of 16 pixels. Since these are at various distances from the unknown pixel, closer pixels are given a higher weighting in the calculation. Bicubic interpolation produces noticeably sharper images than the previous two methods, and is perhaps the ideal combination of processing time and output quality.

# Zooming : Interpolation Techniques

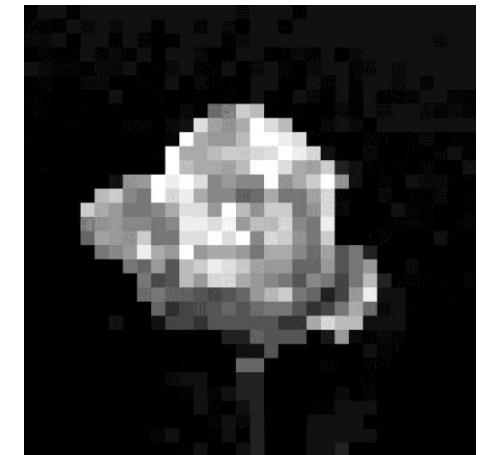
*Nearest Neighbour*



$128 \times 128 \rightarrow 1024 \times 1024$



$64 \times 64 \rightarrow 1024 \times 1024$

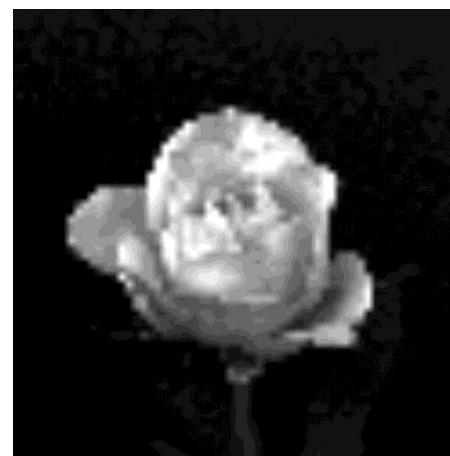


$32 \times 32 \rightarrow 1024 \times 1024$

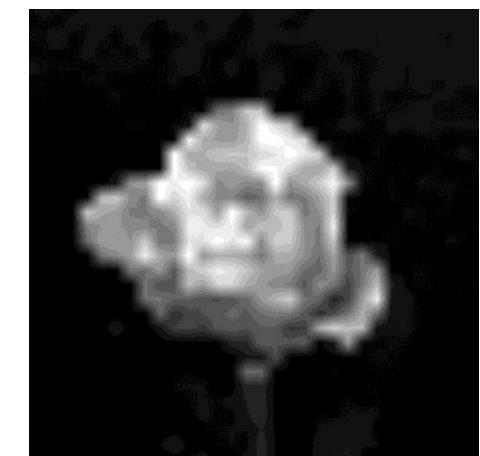
*Bilinear*



$128 \times 128 \rightarrow 1024 \times 1024$



$64 \times 64 \rightarrow 1024 \times 1024$



$32 \times 32 \rightarrow 1024 \times 1024$

---

# Zooming : Interpolation Techniques

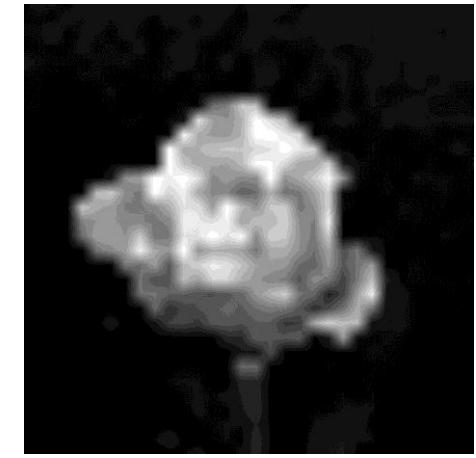
*Bilinear*



**128x128  $\Rightarrow$  1024x1024**



**64x64  $\Rightarrow$  1024x1024**

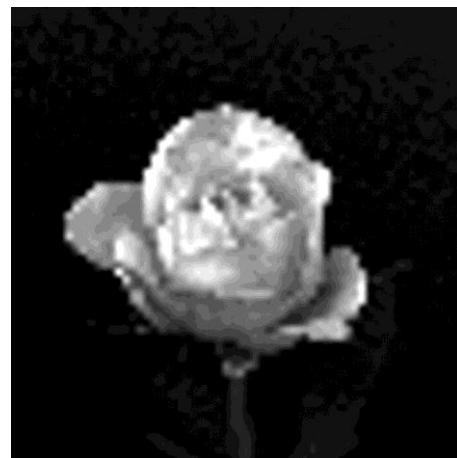


**32x32  $\Rightarrow$  1024x1024**

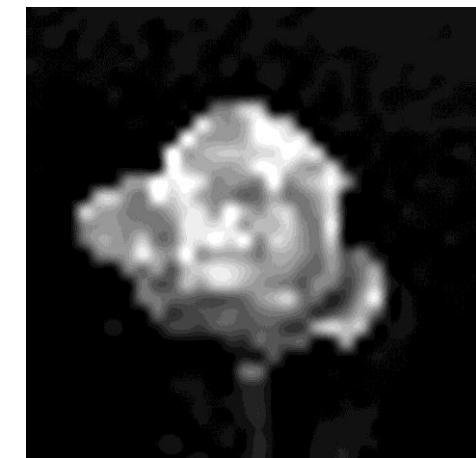
*Bicubic*



**128x128  $\Rightarrow$  1024x1024**



**64x64  $\Rightarrow$  1024x1024**



**32x32  $\Rightarrow$  1024x1024**

---

# Zooming : Interpolation Techniques

## *Nearest neighbor interpolation:*

- *Fastest Processing*
- *Produces undesired checkboard/blocking (**Aliasing**) effect*
- *May be good for rectangular images*
- *Not suitable for detailed or photographic images*

## *Bilinear interpolation:*

- *smoother looking images than nearest neighbor.*
- *has an **anti-aliasing** effect, therefore less blocking effect than nearest neighbor.*

## *Bicubic interpolation:*

- *produces noticeably **sharper** images than the previous two methods.*
- *has an **anti-aliasing** effect (**Almost no blocking**).*
- *used as a standard in many image editing programs (i.e. **Adobe Photoshop**)*

---

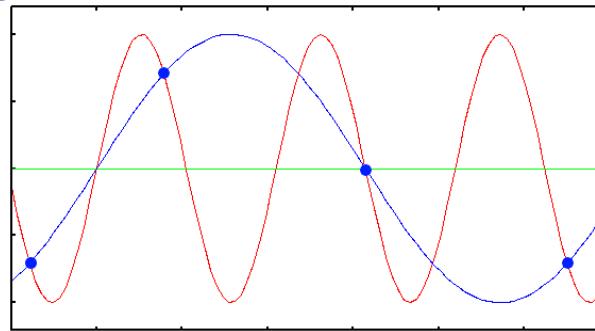
# Sampling and Spatial Resolution

**Spatial Aliasing Problem:** Spatial aliasing is insufficient sampling of data along the space axis, which occurs because of the insufficient spatial resolution of the acquired image.

- It is recommended to sample the image at a rate close to the ideal sampling rate (Nyquist Rate). Images obtained with sampling distances larger than those established by this rate suffer from undersampling.
- The Nyquist Rate is defined as twice the bandwidth of the continuous-time signal. It should be noted that the sampling frequency must be strictly greater than the Nyquist Rate of the signal to achieve unambiguous representation of the signal.
- The Critical Sampling Distance is the Sampling Distance corresponding to the Nyquist Rate.

# Sampling and Spatial Resolution

**Spatial Aliasing Problem:** In statistics, signal processing, and related disciplines, aliasing is an effect that causes different continuous signals to become indistinguishable (or aliases of one another) when sampled. When this happens, the original signal cannot be uniquely reconstructed from the sampled signal.



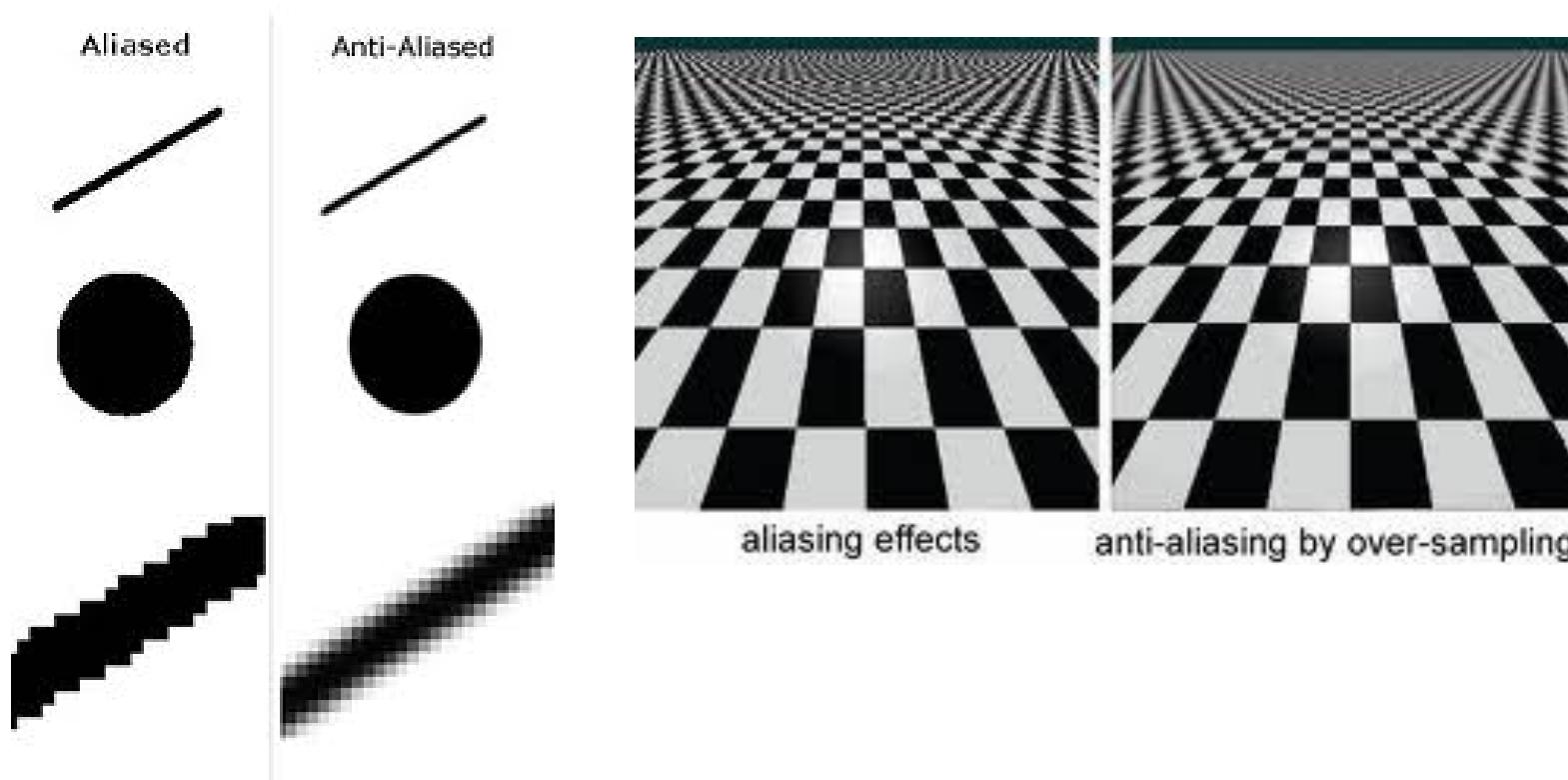
- aliasing of a undersampled 1D sinusoidal signal



- aliasing of a undersampled 2D image

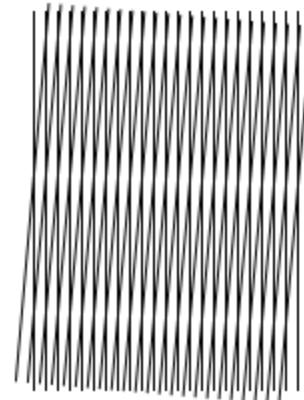
# Sampling and Spatial Resolution

*Spatial Aliasing Problem: In statistics, signal processing, and related disciplines, aliasing is an effect that causes different continuous signals to become indistinguishable (or aliases of one another) when sampled. When this happens, the original signal cannot be uniquely reconstructed from the sampled signal.*



# Sampling and Spatial Resolution

***Spatial Aliasing Problem:*** Aliasing may give rise to moiré patterns (when the original image is finely textured) or jagged outlines (when the original has sharp contrasting edges, e.g. screen fonts).



i.e. a moiré pattern, formed by two sets of parallel lines, one set inclined at an angle of 5° to the other



A moiré pattern formed by incorrectly downsampling the image

# Sampling and Spatial Resolution

*Anti-aliasing refers to the low pass filters (LPF) applied to an image before it is downsampled (downscaled) to avoid → Aliasing Artifacts. The filters remove high-frequency content from the original image reducing its Band Width, and therefore the lower Sampling Density in the downscaled image is still above the Critical Sampling Distance.*



Aliasing problem of a downsampled image



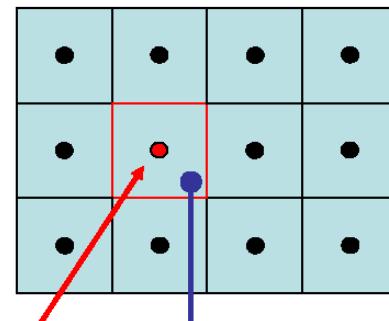
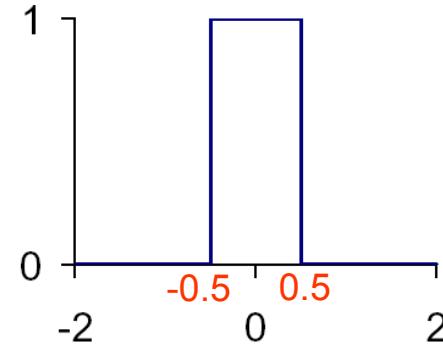
Aliasing problem avoided by using band limiting Low Pass Filter.

# Up-Sampling - Image Zooming

**Nearest Neighbor Interpolation:**

*The Interpolation kernel for the nearest neighbor interpolation is defined by:*

$$h(x) = \begin{cases} 1 & 0 \leq |x| < 0.5 \\ 0 & 0.5 \leq |x| \end{cases}$$



Interpolation  
Kernel

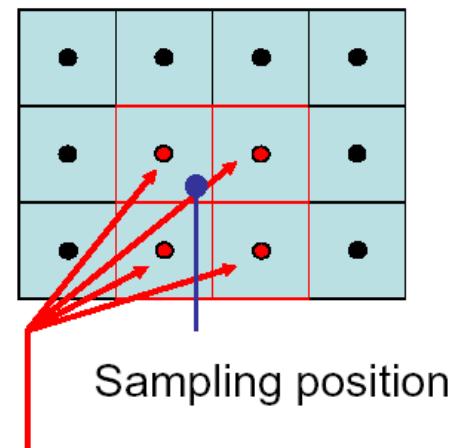
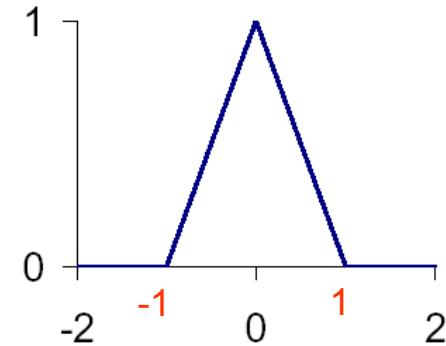
Nearest Pixel

# Up-Sampling - Image Zooming

## Bilinear Interpolation:

The Interpolation kernel for the bilinear interpolation is defined by:

$$h(x) = \begin{cases} 1 - |x| & 0 \leq |x| < 1 \\ 0 & 1 \leq |x| \end{cases}$$



Interpolation  
Kernel

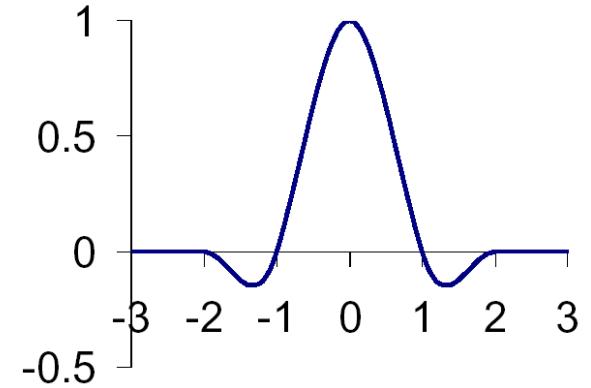
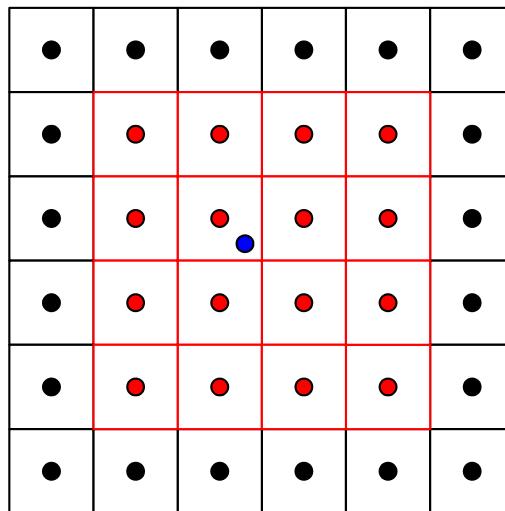
Weighted pixels

# Up-Sampling - Image Zooming

## Bicubic Interpolation:

The Interpolation kernel for the bicubic interpolation is defined by:

$$h(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & 0 \leq |x| < 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & 1 \leq |x| < 2 \\ 0 & 2 \leq |x| \end{cases}$$



- unused pixels
- weighted pixels
- sampling position

# Up-Sampling - Image Zooming

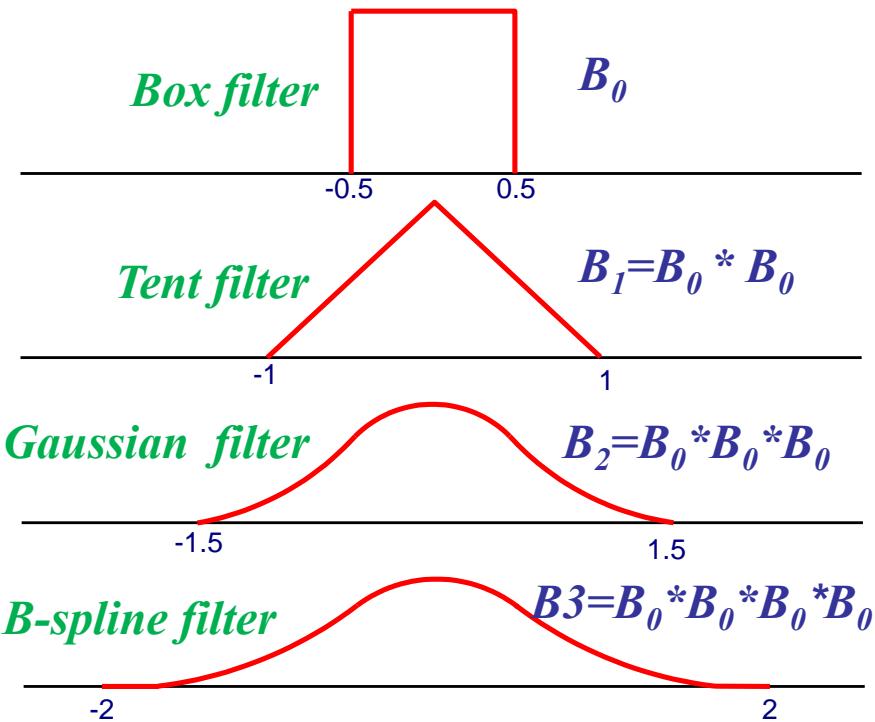
## Spline Interpolation:

*There are many spline interpolation kernels. Among them the Interpolation kernel for the B-spline interpolation is defined by and B-spline of degree n is derived through n convolutions of the **box filter**,  $B_0$ .*

$$h(x) = \frac{1}{6} \begin{cases} 3|x|^3 - 6|x|^2 + 4 & 0 \leq |x| < 1 \\ -|x|^3 + 6|x|^2 - 12|x| + 8 & 1 \leq |x| < 2 \\ 0 & 2 \leq |x| \end{cases}$$

●	●	●	●	●	●
●	●	●	●	●	●
●	●	●	●	●	●
●	●	●	●	●	●
●	●	●	●	●	●
●	●	●	●	●	●

- unused pixels
- weighted pixels
- sampling position



# Up-Sampling - Image Zooming

*Comparing Interpolation Techniques:*



Nearest Neighbor



Bilinear



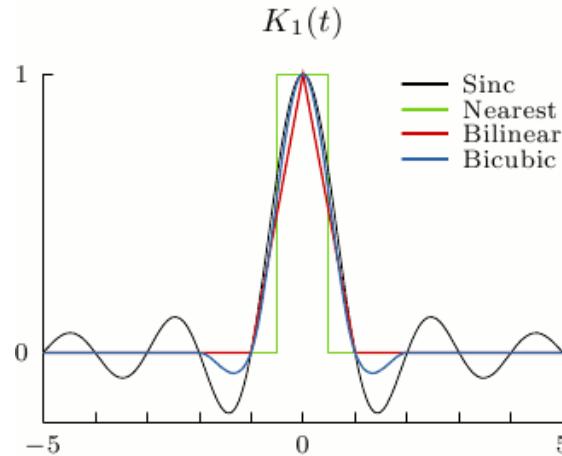
Bicubic



B-spline

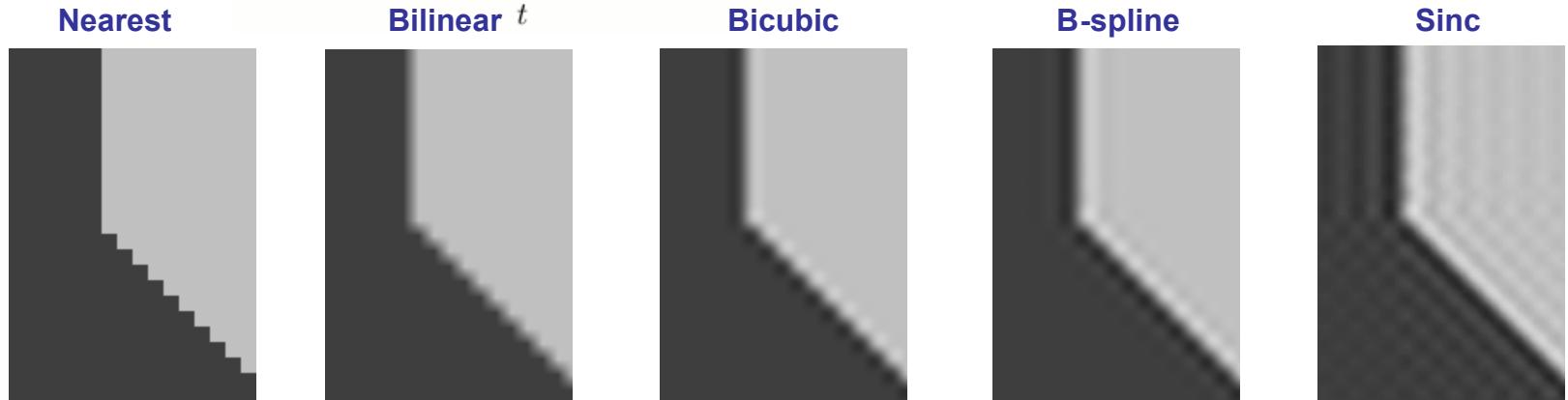
# Up-Sampling - Image Zooming

*Comparing Interpolation Techniques: The Sinc function can also be used as the interpolation kernel.*



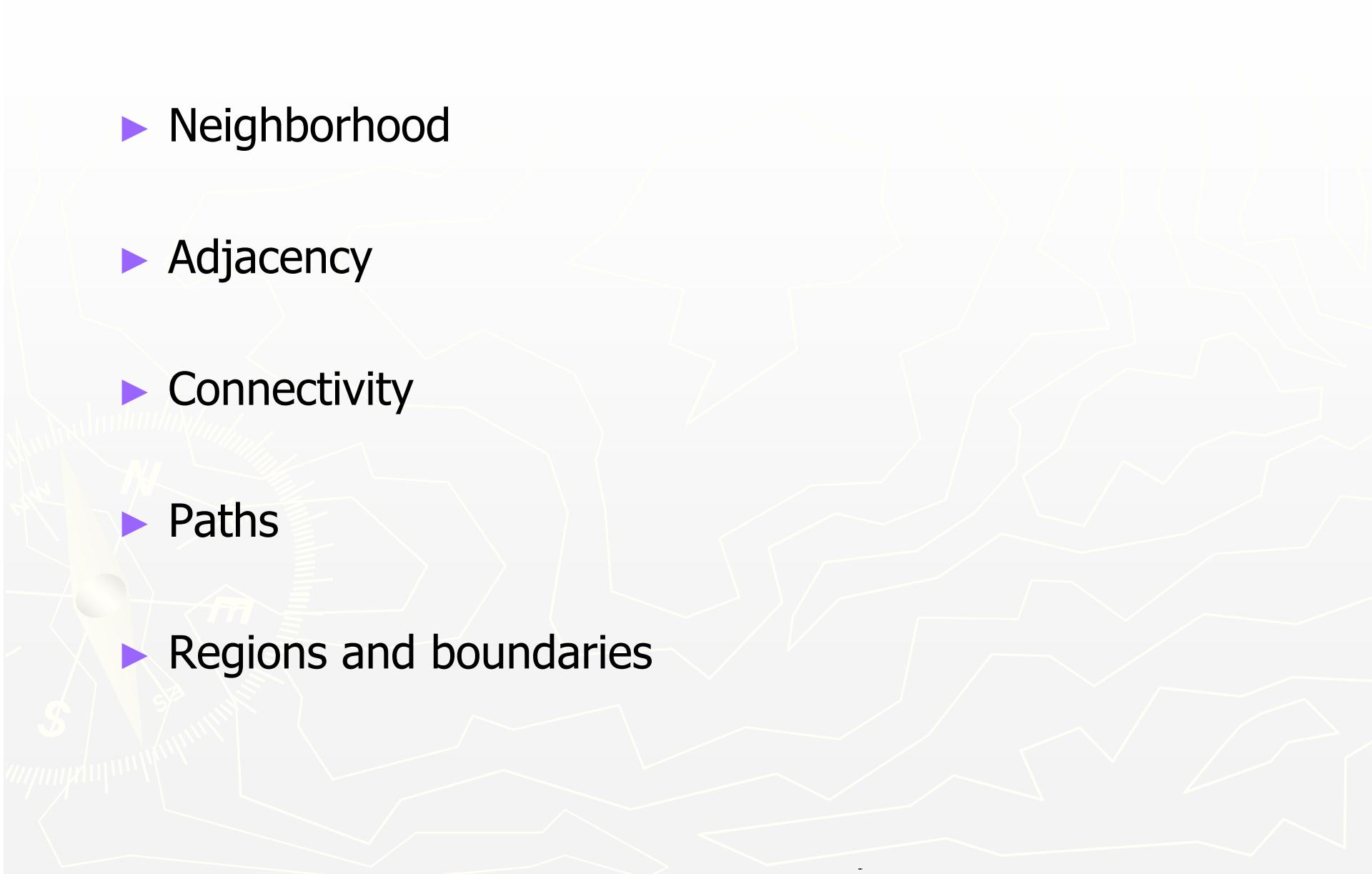
$$K_1(t) = \frac{\sin(\pi t)}{\pi t}$$

The *sinc* function



# Basic Relationships Between Pixels

- ▶ Neighborhood
- ▶ Adjacency
- ▶ Connectivity
- ▶ Paths
- ▶ Regions and boundaries



# Basic Relationships Between Pixels

- ▶ **Neighbors** of a pixel  $p$  at coordinates  $(x,y)$
- ▶ **4-neighbors of  $p$** , denoted by  $N_4(p)$ :  
 $(x-1, y)$ ,  $(x+1, y)$ ,  $(x, y-1)$ , and  $(x, y+1)$ .
- ▶ **4 diagonal neighbors of  $p$** , denoted by  $N_D(p)$ :  
 $(x-1, y-1)$ ,  $(x+1, y+1)$ ,  $(x+1, y-1)$ , and  $(x-1, y+1)$ .
- ▶ **8 neighbors of  $p$** , denoted  $N_8(p)$   
$$N_8(p) = N_4(p) \cup N_D(p)$$

# Basic Relationships Between Pixels

## ► **Adjacency**

Let  $V$  be the set of intensity values

- **4-adjacency:** Two pixels  $p$  and  $q$  with values from  $V$  are 4-adjacent if  $q$  is in the set  $N_4(p)$ .
- **8-adjacency:** Two pixels  $p$  and  $q$  with values from  $V$  are 8-adjacent if  $q$  is in the set  $N_8(p)$ .

# Basic Relationships Between Pixels

## ► **Adjacency**

Let  $V$  be the set of intensity values

► **m-adjacency:** Two pixels  $p$  and  $q$  with values from  $V$  are m-adjacent if

(i)  $q$  is in the set  $N_4(p)$ , or

(ii)  $q$  is in the set  $N_D(p)$  and the set  $N_4(p) \cap N_4(q)$  has no pixels whose values are from  $V$ .

# Basic Relationships Between Pixels

- ▶ **Path**
- A (digital) path (or curve) from pixel p with coordinates  $(x_0, y_0)$  to pixel q with coordinates  $(x_n, y_n)$  is a sequence of distinct pixels with coordinates

$$(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$$

Where  $(x_i, y_i)$  and  $(x_{i-1}, y_{i-1})$  are adjacent for  $1 \leq i \leq n$ .

- Here  $n$  is the *length* of the path.
- If  $(x_0, y_0) = (x_n, y_n)$ , the path is *closed* path.
- We can define 4-, 8-, and m-paths based on the type of adjacency used.

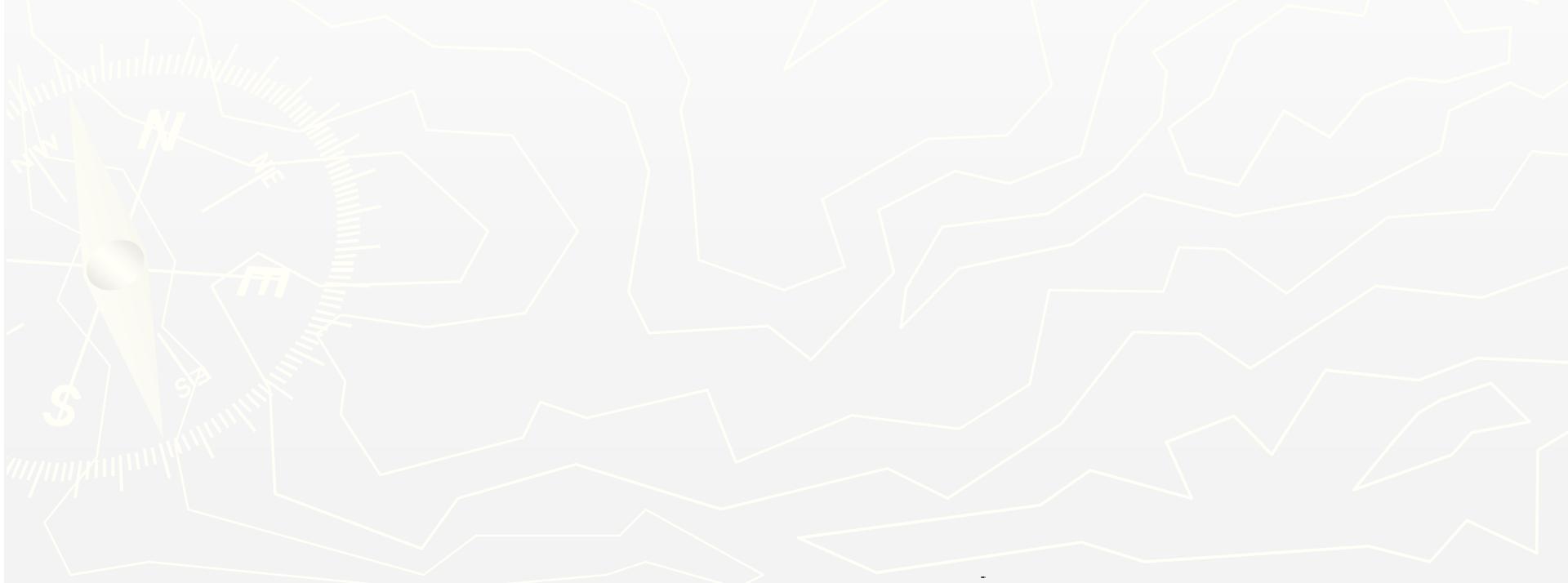
# Examples: Adjacency and Path

$$V = \{1, 2\}$$

0	1	1
0	2	0
0	0	1

0	1	1
0	2	0
0	0	1

0	1	1
0	2	0
0	0	1



# Examples: Adjacency and Path

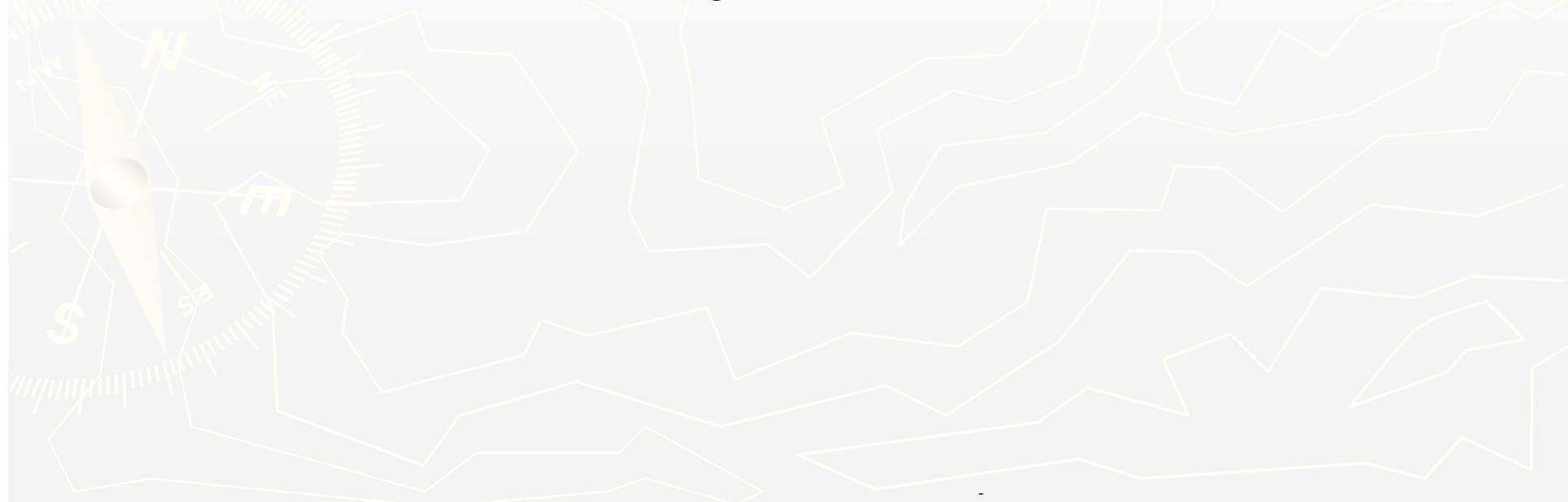
$$V = \{1, 2\}$$

0	1	1
0	2	0
0	0	1

0	1	1
0	2	0
0	0	1

0	1	1
0	2	0
0	0	1

8-adjacent



# Examples: Adjacency and Path

$$V = \{1, 2\}$$

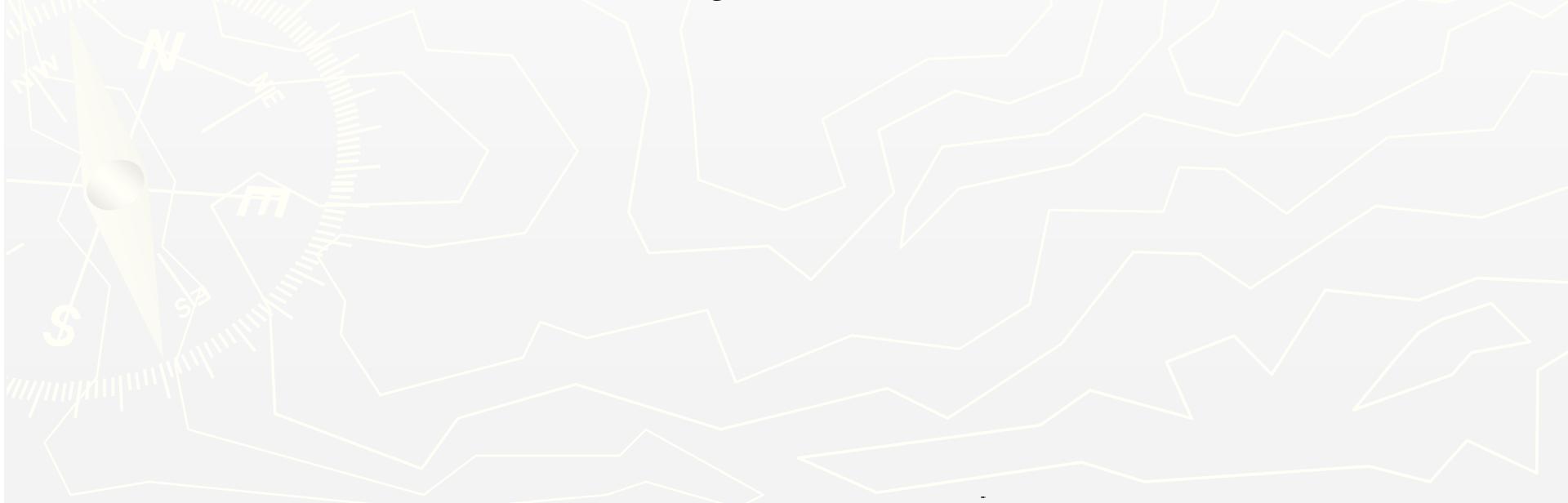
0	1	1
0	2	0
0	0	1

0	1	1
0	2	0
0	0	1

0	1	1
0	2	0
0	0	1

8-adjacent

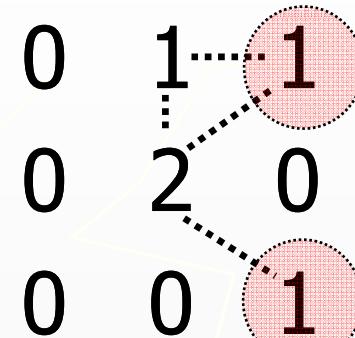
m-adjacent



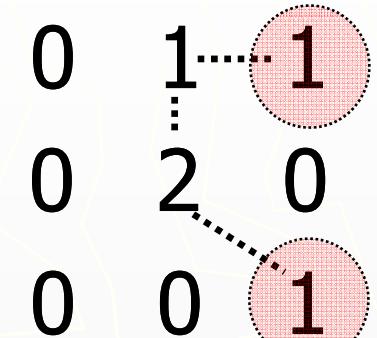
# Examples: Adjacency and Path

$$V = \{1, 2\}$$

0 <sub>1,1</sub>	1 <sub>1,2</sub>	1 <sub>1,3</sub>
0 <sub>2,1</sub>	2 <sub>2,2</sub>	0 <sub>2,3</sub>
0 <sub>3,1</sub>	0 <sub>3,2</sub>	1 <sub>3,3</sub>



**8-adjacent**



**m-adjacent**

The 8-path from (1,3) to (3,3):

- (i) (1,3), (1,2), (2,2), (3,3)
- (ii) (1,3), (2,2), (3,3)

The m-path from (1,3) to (3,3):

- (1,3), (1,2), (2,2), (3,3)

# Basic Relationships Between Pixels

## ► **Connected in S**

Let  $S$  represent a subset of pixels in an image. Two pixels  $p$  with coordinates  $(x_0, y_0)$  and  $q$  with coordinates  $(x_n, y_n)$  are said to be **connected in  $S$**  if there exists a path

$$(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$$

$$\text{Where } \forall i, 0 \leq i \leq n, (x_i, y_i) \in S$$

# Basic Relationships Between Pixels

Let  $S$  represent a subset of pixels in an image

- ▶ For every pixel  $p$  in  $S$ , the set of pixels in  $S$  that are connected to  $p$  is called a ***connected component*** of  $S$ .
- ▶ If  $S$  has only one connected component, then  $S$  is called ***Connected Set***.
- ▶ We call  $R$  a **region** of the image if  $R$  is a connected set
- ▶ Two regions,  $R_i$  and  $R_j$  are said to be ***adjacent*** if their union forms a connected set.
- ▶ Regions that are not to be adjacent are said to be ***disjoint***.

# Basic Relationships Between Pixels

- ▶ **Boundary (or border)**

- The ***boundary*** of the region  $R$  is the set of pixels in the region that have one or more neighbors that are not in  $R$ .
- If  $R$  happens to be an entire image, then its boundary is defined as the set of pixels in the first and last rows and columns of the image.

- ▶ **Foreground and background**

- An image contains  $K$  disjoint regions,  $R_k$ ,  $k = 1, 2, \dots, K$ . Let  $R_u$  denote the union of all the  $K$  regions, and let  $(R_u)^c$  denote its complement.  
All the points in  $R_u$  is called **foreground**;  
All the points in  $(R_u)^c$  is called **background**.

# Question 1

- ▶ In the following arrangement of pixels, are the two regions (of 1s) adjacent? (if 8-adjacency is used)

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

Region 1

Region 2

## Question 2

- ▶ In the following arrangement of pixels, are the two parts (of 1s) adjacent? (if 4-adjacency is used)

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

Part 1

Part 2

- ▶ In the following arrangement of pixels, the two regions (of 1s) are disjoint (if 4-adjacency is used)

1	1	1
1	0	1
0	1	0

0	0	1
1	1	1
1	1	1

Region 1

Region 2

- ▶ In the following arrangement of pixels, the two regions (of 1s) are disjoint (if 4-adjacency is used)

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

foreground

background

# Question 3

- ▶ In the following arrangement of pixels, the circled point is part of the boundary of the 1-valued pixels if 8-adjacency is used, true or false?

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

## Question 4

- ▶ In the following arrangement of pixels, the circled point is part of the boundary of the 1-valued pixels if 4-adjacency is used, true or false?

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

# Distance Measures

- Given pixels  $p$ ,  $q$  and  $z$  with coordinates  $(x, y)$ ,  $(s, t)$ ,  $(u, v)$  respectively, the distance function  $D$  has following properties:

a.  $D(p, q) \geq 0$       [ $D(p, q) = 0$ , iff  $p = q$ ]

b.  $D(p, q) = D(q, p)$

c.  $D(p, z) \leq D(p, q) + D(q, z)$

# Distance Measures

The following are the different Distance measures:

a. Euclidean Distance :

$$D_e(p, q) = [(x-s)^2 + (y-t)^2]^{1/2}$$

b. City Block Distance:

$$D_4(p, q) = |x-s| + |y-t|$$

c. Chess Board Distance:

$$D_8(p, q) = \max(|x-s|, |y-t|)$$

2	2	1	2	2
2	1	0	1	2
2	1	2	1	2
2				
2				

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

# Question 5

- ▶ In the following arrangement of pixels, what's the value of the chessboard distance between the circled two points?

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

# Question 6

- ▶ In the following arrangement of pixels, what's the value of the city-block distance between the circled two points?

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

# Question 7

- ▶ In the following arrangement of pixels, what's the value of the length of the m-path between the circled two points?

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

# Question 8

- ▶ In the following arrangement of pixels, what's the value of the length of the m-path between the circled two points?

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

# Introduction to Mathematical Operations in DIP

## ► Linear vs. Nonlinear Operation

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

$$H[a_i f_i(x, y) + a_j f_j(x, y)]$$

$$= H[a_i f_i(x, y)] + H[a_j f_j(x, y)]$$

Additivity

$$= a_i H[f_i(x, y)] + a_j H[f_j(x, y)]$$

Homogeneity

$$= a_i g_i(x, y) + a_j g_j(x, y)$$

H is said to be a **linear operator**;

H is said to be a **nonlinear operator** if it does not meet the above qualification.

# Arithmetic Operations

- ▶ Arithmetic operations between images are array operations. The four arithmetic operations are denoted as

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

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

$$p(x,y) = f(x,y) \times g(x,y)$$

$$v(x,y) = f(x,y) \div g(x,y)$$

## Example: Addition of Noisy Images for Noise Reduction

Noiseless image:  $f(x,y)$

Noise:  $n(x,y)$  (at every pair of coordinates  $(x,y)$ , the noise is uncorrelated and has zero average value)

Corrupted image:  $g(x,y)$

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

Reducing the noise by adding a set of noisy images,  $\{g_i(x,y)\}$

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

## Example: Addition of Noisy Images for Noise Reduction

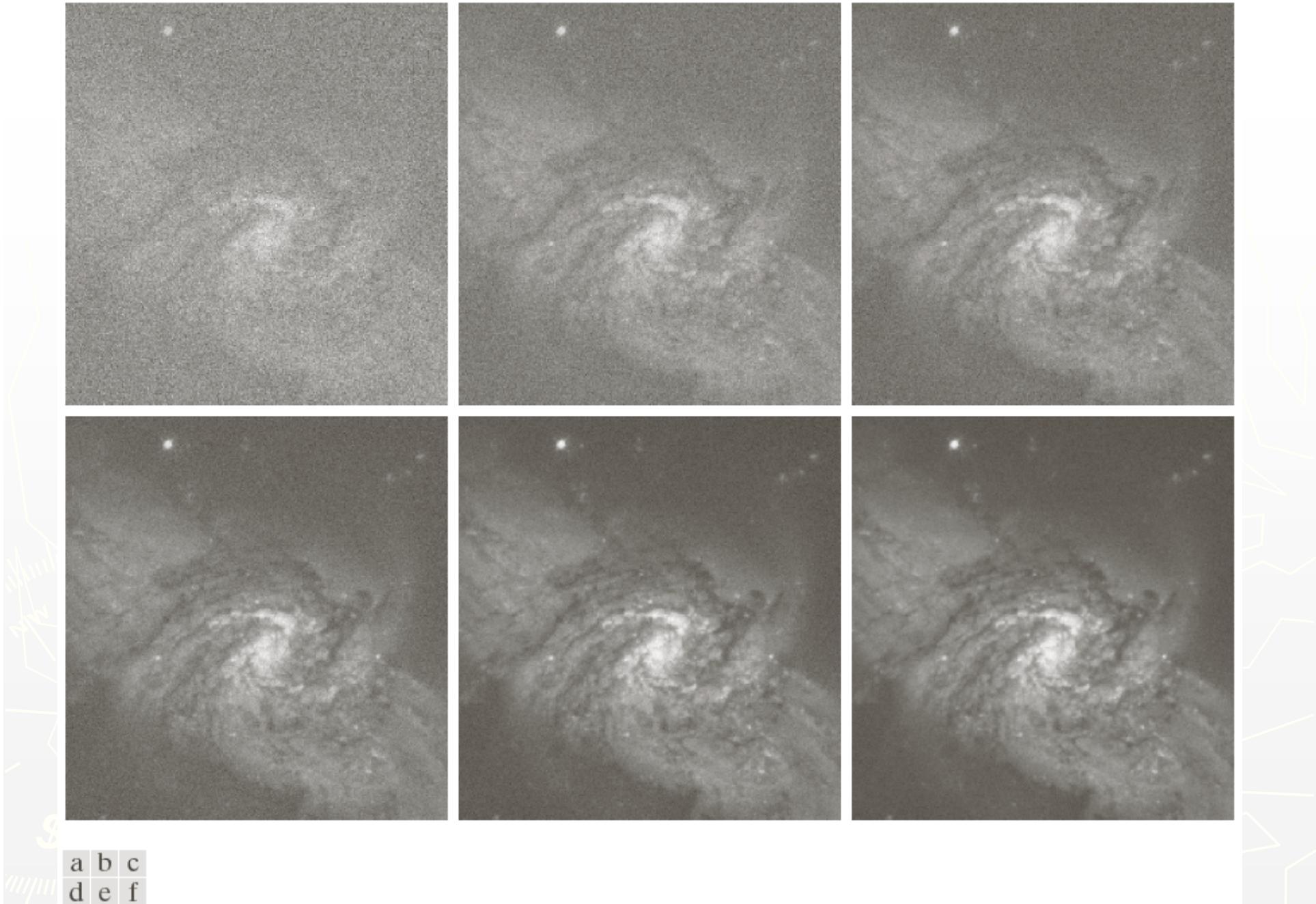
$$\bar{g}(x, y) = \frac{1}{K} \sum_{i=1}^K g_i(x, y)$$

$$\begin{aligned} E\{\bar{g}(x, y)\} &= E\left\{\frac{1}{K} \sum_{i=1}^K g_i(x, y)\right\} \\ &= E\left\{\frac{1}{K} \sum_{i=1}^K [f(x, y) + n_i(x, y)]\right\} \\ &= f(x, y) + E\left\{\frac{1}{K} \sum_{i=1}^K n_i(x, y)\right\} \\ &= f(x, y) \end{aligned}$$

$$\begin{aligned} \sigma_{\bar{g}(x, y)}^2 &= \sigma^2_{\frac{1}{K} \sum_{i=1}^K g_i(x, y)} \\ &= \sigma^2_{\frac{1}{K} \sum_{i=1}^K n_i(x, y)} \\ &= \frac{1}{K} \sigma_{n(x, y)}^2 \end{aligned}$$

## Example: Addition of Noisy Images for Noise Reduction

- ▶ In astronomy, imaging under very low light levels frequently causes sensor noise to render single images virtually useless for analysis.
- ▶ In astronomical observations, similar sensors for noise reduction by observing the same scene over long periods of time. Image averaging is then used to reduce the noise.



**FIGURE 2.26** (a) Image of Galaxy Pair NGC 3314 corrupted by additive Gaussian noise. (b)–(f) Results of averaging 5, 10, 20, 50, and 100 noisy images, respectively. (Original image courtesy of NASA.)

# An Example of Image Subtraction: Mask Mode Radiography

**Mask  $h(x,y)$ :** an X-ray image of a region of a patient's body

**Live images  $f(x,y)$ :** X-ray images captured at TV rates after injection of the contrast medium

**Enhanced detail  $g(x,y)$**

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

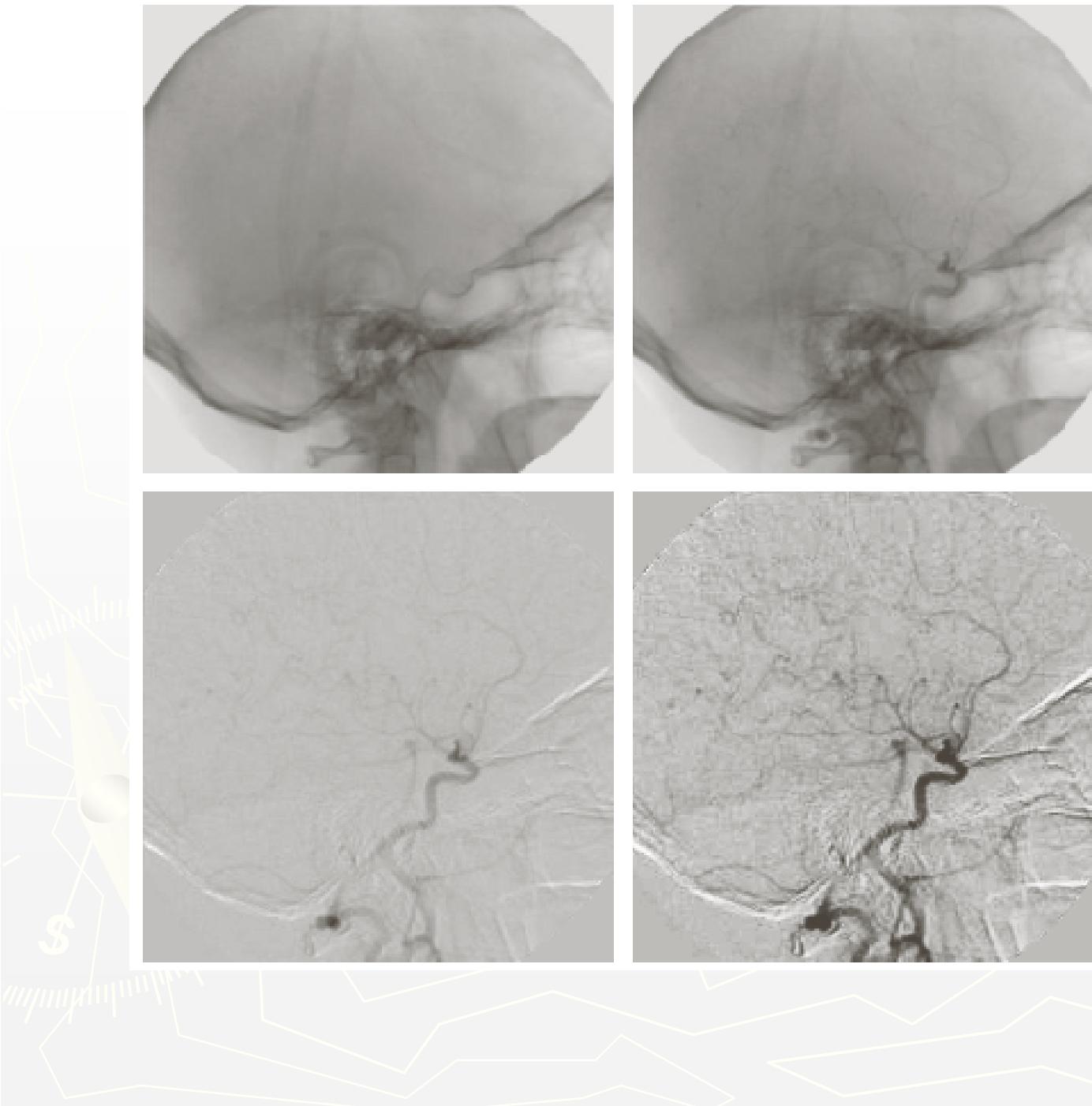
The procedure gives a movie showing how the contrast medium propagates through the various arteries in the area being observed.

a b  
c d

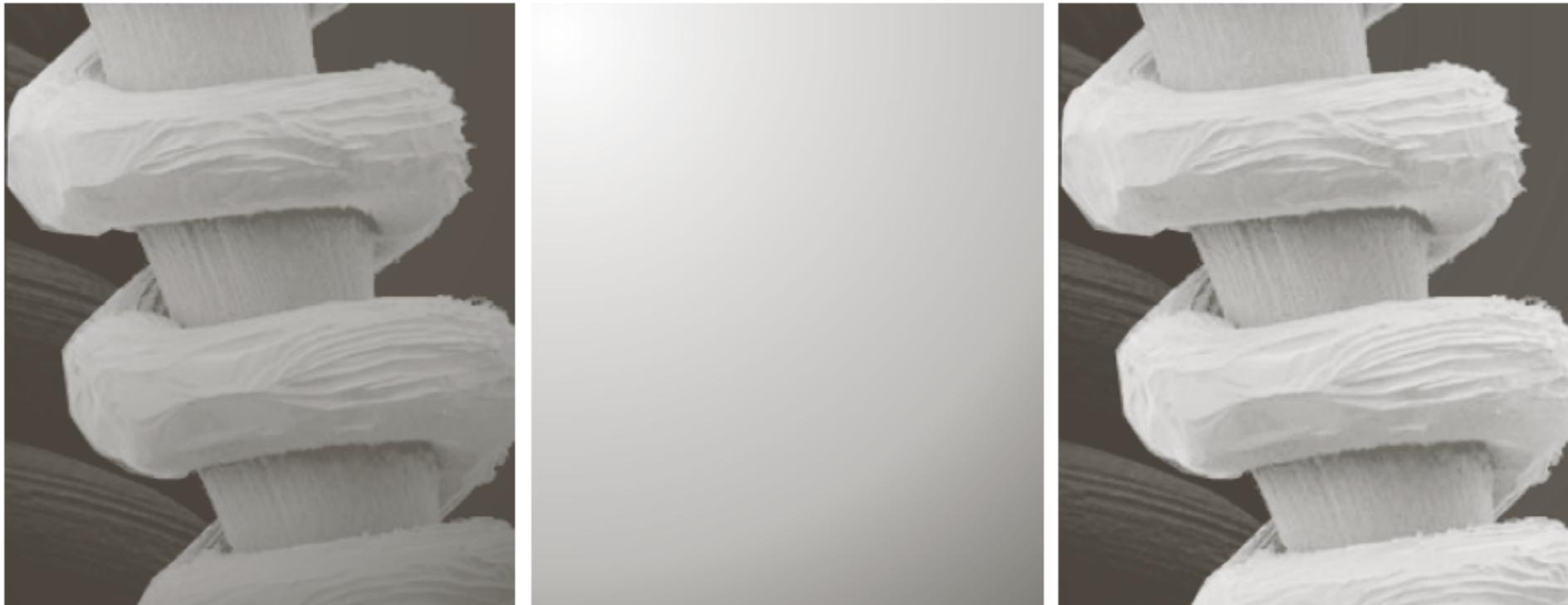
**FIGURE 2.28**

Digital subtraction angiography.

- (a) Mask image.
- (b) A live image.
- (c) Difference between (a) and (b).
- (d) Enhanced difference image.  
(Figures (a) and (b) courtesy of The Image Sciences Institute, University Medical Center, Utrecht, The Netherlands.)

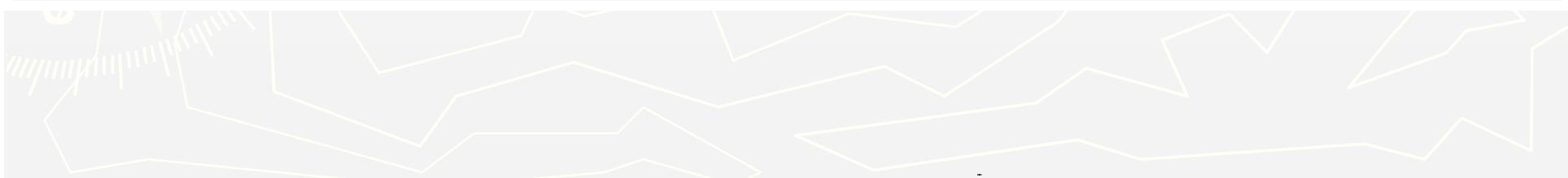


## An Example of Image Multiplication

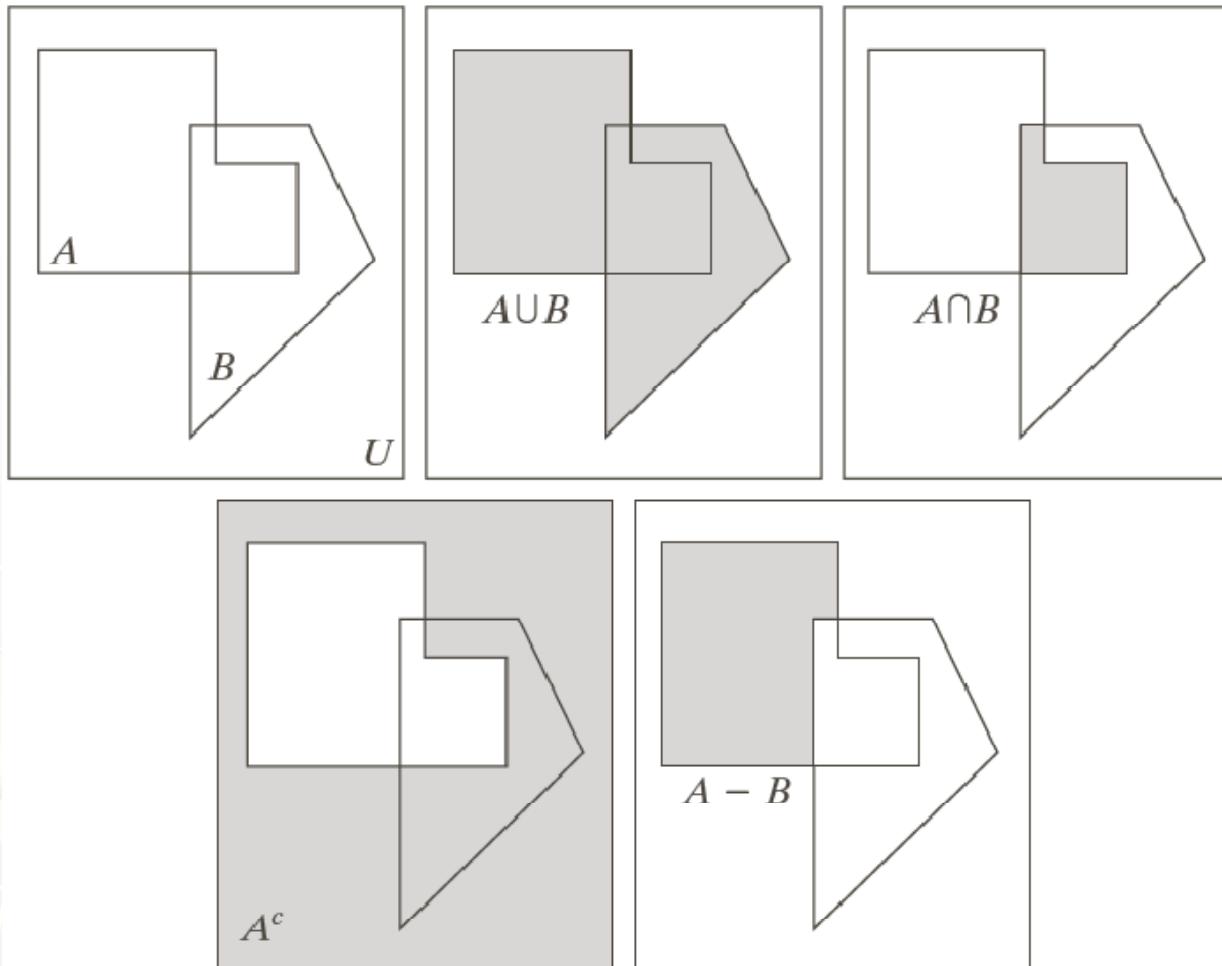


a | b | c

**FIGURE 2.29** Shading correction. (a) Shaded SEM image of a tungsten filament and support, magnified approximately 130 times. (b) The shading pattern. (c) Product of (a) by the reciprocal of (b). (Original image courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)



# Set and Logical Operations



a	b	c
d	e	

**FIGURE 2.31**

(a) Two sets of coordinates,  $A$  and  $B$ , in 2-D space. (b) The union of  $A$  and  $B$ . (c) The intersection of  $A$  and  $B$ . (d) The complement of  $A$ . (e) The difference between  $A$  and  $B$ . In (b)–(e) the shaded areas represent the member of the set operation indicated.

# Set and Logical Operations

- ▶ Let A be the elements of a gray-scale image

The elements of A are triplets of the form  $(x, y, z)$ , where x and y are spatial coordinates and z denotes the intensity at the point  $(x, y)$ .

$$A = \{(x, y, z) \mid z = f(x, y)\}$$

- ▶ The complement of A is denoted  $A^c$

$$A^c = \{(x, y, K - z) \mid (x, y, z) \in A\}$$

$K = 2^k - 1$ ; k is the number of intensity bits used to represent z

# Set and Logical Operations

- ▶ The union of two gray-scale images (sets) A and B is defined as the set

$$A \cup B = \{ \max_z(a, b) \mid a \in A, b \in B \}$$

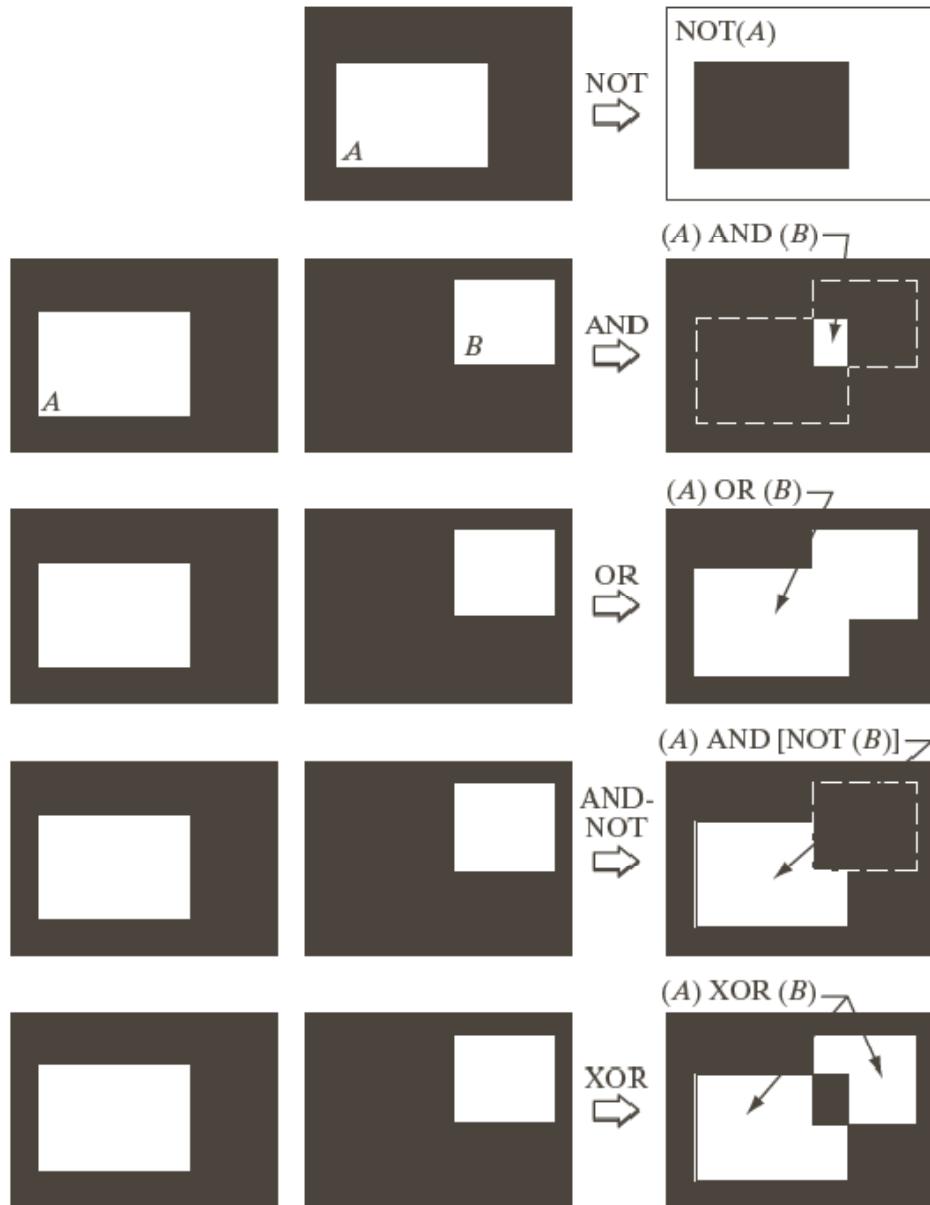
# Set and Logical Operations

a b c



**FIGURE 2.32** Set operations involving gray-scale images.  
(a) Original image. (b) Image negative obtained using set complementation. (c) The union of (a) and a constant image. (Original image courtesy of G.E. Medical Systems.)

# Set and Logical Operations



**FIGURE 2.33**  
Illustration of  
logical operations  
involving  
foreground  
(white) pixels.  
Black represents  
binary 0s and  
white binary 1s.  
The dashed lines  
are shown for  
reference only.  
They are not part  
of the result.

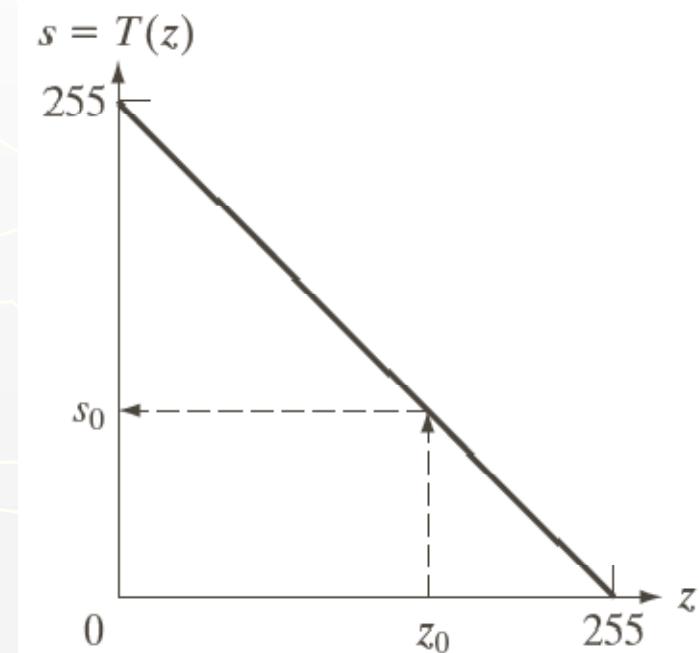
# Spatial Operations

## ► Single-pixel operations

Alter the values of an image's pixels based on the intensity.

e.g.,

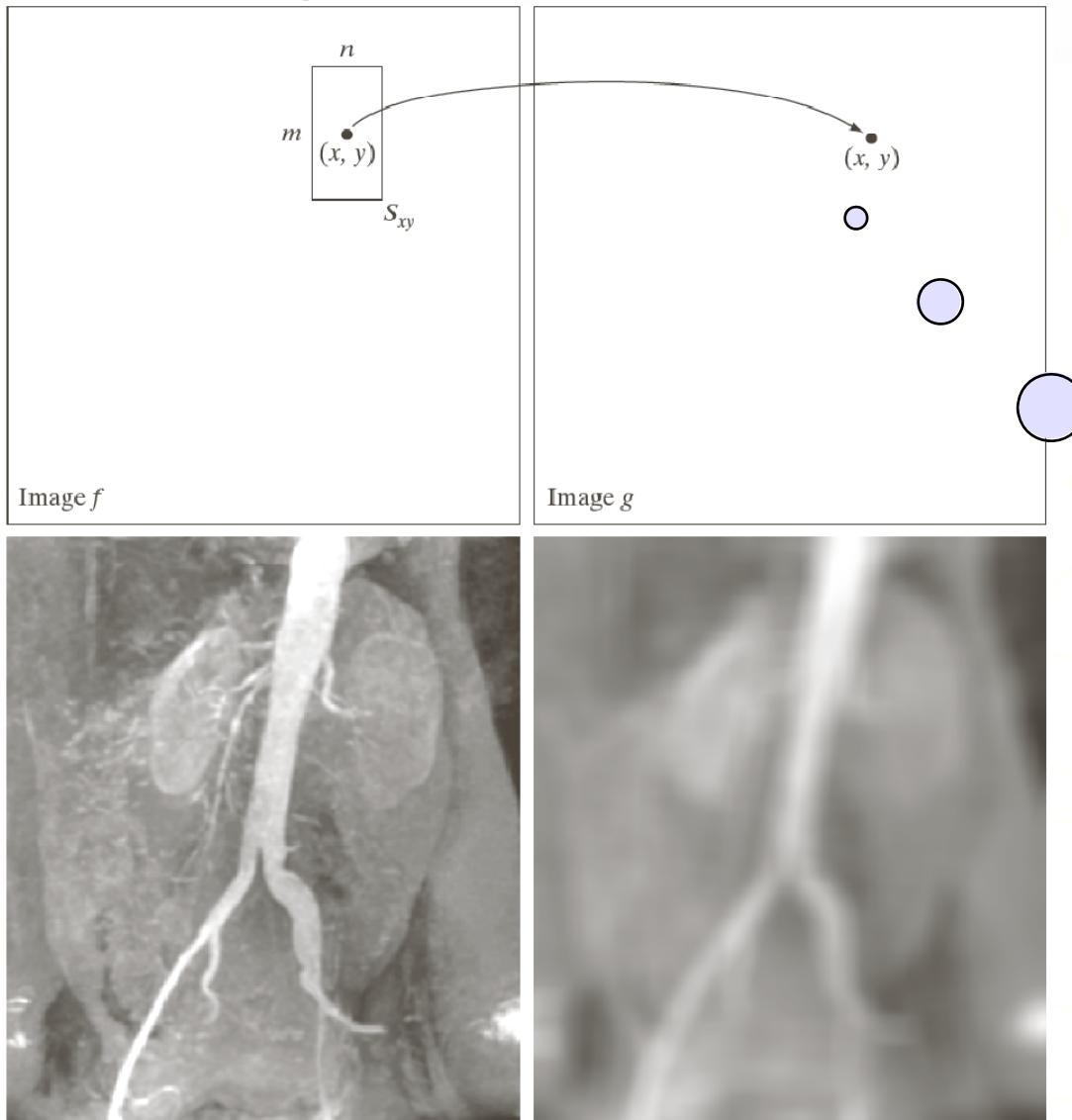
$$s = T(z)$$



**FIGURE 2.34** Intensity transformation function used to obtain the negative of an 8-bit image. The dashed arrows show transformation of an arbitrary input intensity value  $z_0$  into its corresponding output value  $s_0$ .

# Spatial Operations

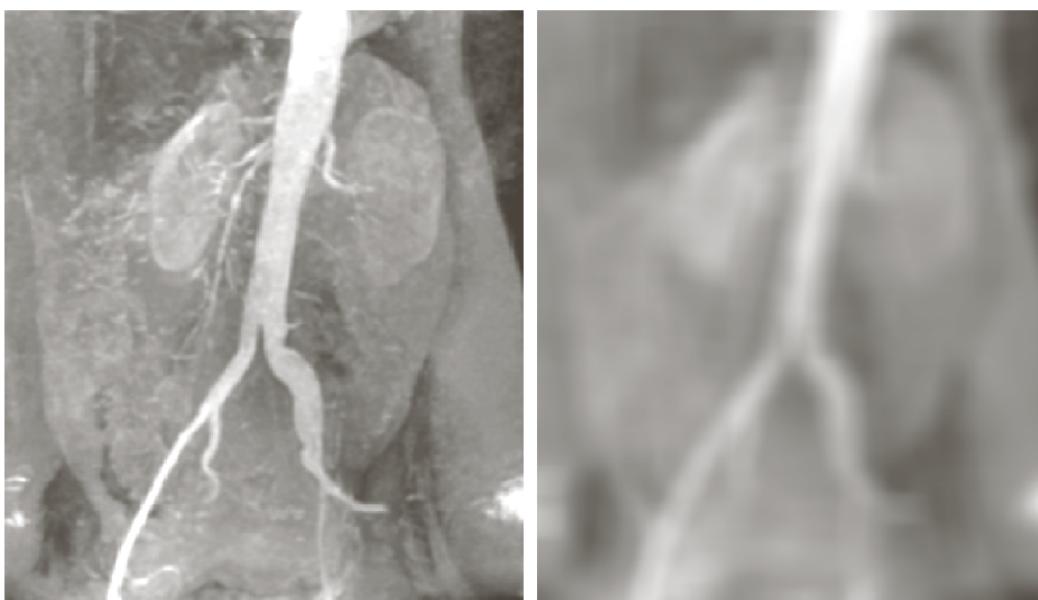
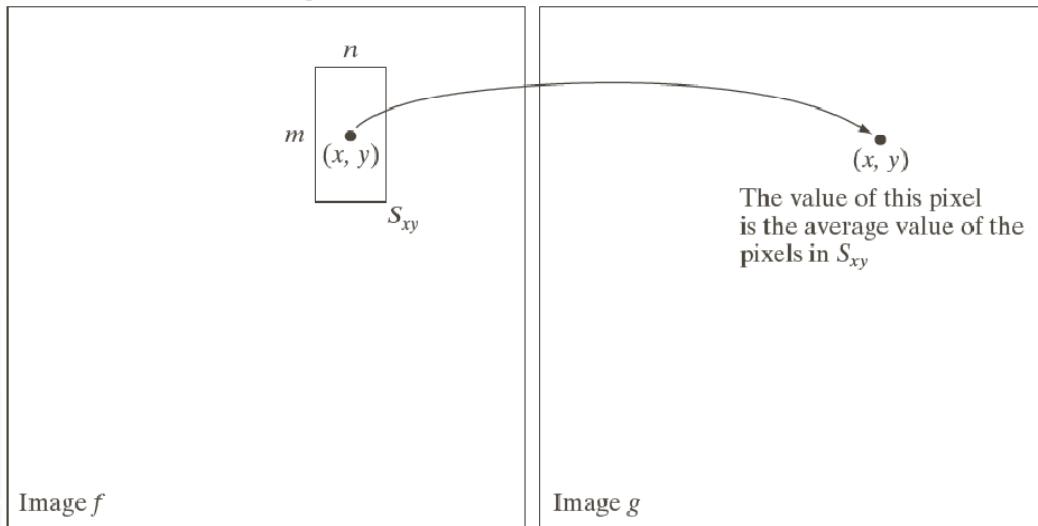
## ► Neighborhood operations



The value of this pixel is determined by a specified operation involving the pixels in the input image with coordinates in  $S_{xy}$

# Spatial Operations

## ► Neighborhood operations



# Geometric Spatial Transformations

- ▶ Geometric transformation (rubber-sheet transformation)
  - A spatial transformation of coordinates

$$(x, y) = T\{(v, w)\}$$

- intensity interpolation that assigns intensity values to the spatially transformed pixels.

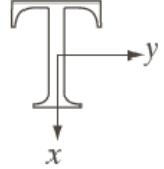
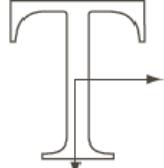
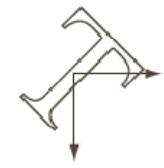
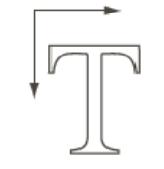
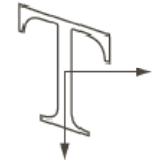
- ▶ Affine transform

$$\begin{bmatrix} x & y & 1 \end{bmatrix} =$$

$$\begin{bmatrix} v & w & 1 \end{bmatrix} \begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}$$

**TABLE 2.2**

Affine transformations based on Eq. (2.6.-23).

Transformation Name	Affine Matrix, T	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = w$	
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = c_x v$ $y = c_y w$	
Rotation	$\begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v \cos \theta - w \sin \theta$ $y = v \cos \theta + w \sin \theta$	
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	$x = v + t_x$ $y = w + t_y$	
Shear (vertical)	$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v + s_v w$ $y = w$	
Shear (horizontal)	$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = s_h v + w$	

# Intensity Assignment

- ▶ Forward Mapping

$$(x, y) = T\{(v, w)\}$$

It's possible that two or more pixels can be transformed to the same location in the output image.

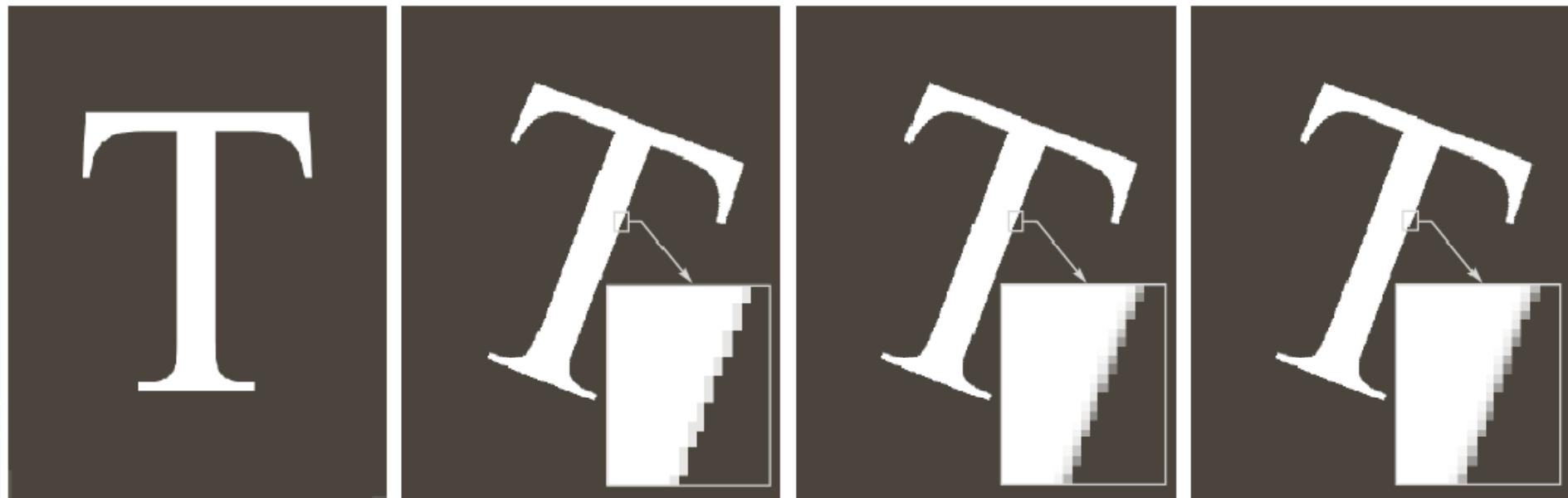
- ▶ Inverse Mapping

$$(v, w) = T^{-1}\{(x, y)\}$$

The nearest input pixels to determine the intensity of the output pixel value.

Inverse mappings are more efficient to implement than forward mappings.

# Example: Image Rotation and Intensity Interpolation



a b c d

**FIGURE 2.36** (a) A 300 dpi image of the letter T. (b) Image rotated  $21^\circ$  clockwise using nearest neighbor interpolation to assign intensity values to the spatially transformed pixels. (c) Image rotated  $21^\circ$  using bilinear interpolation. (d) Image rotated  $21^\circ$  using bicubic interpolation. The enlarged sections show edge detail for the three interpolation approaches.

# Image Registration

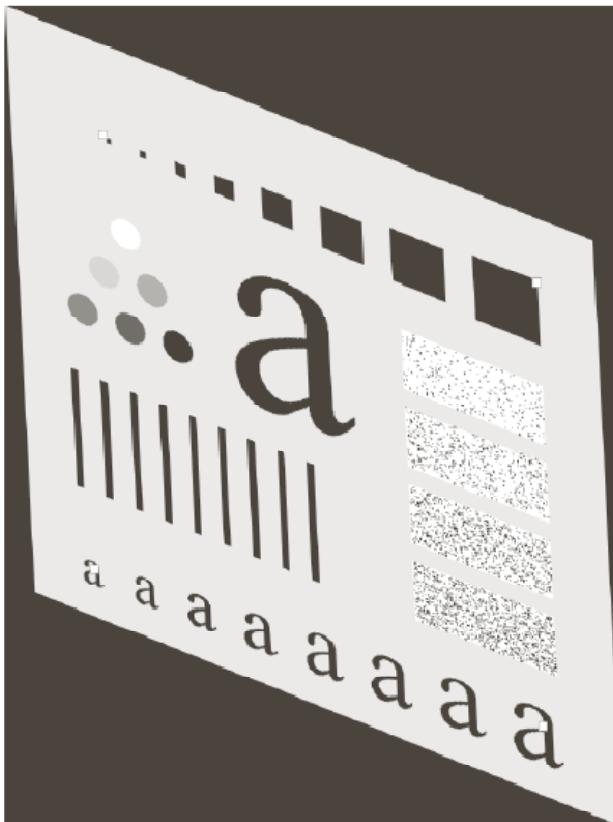
- ▶ Input and output images are available but the transformation function is unknown.  
Goal: estimate the transformation function and use it to register the two images.
- ▶ One of the principal approaches for image registration is to use *tie points* (also called *control points*)
  - The corresponding points are known precisely in the input and output (**reference**) images.

# Image Registration

- ▶ A simple model based on bilinear approximation:

$$\begin{cases} x = c_1v + c_2w + c_3vw + c_4 \\ y = c_5v + c_6w + c_7vw + c_8 \end{cases}$$

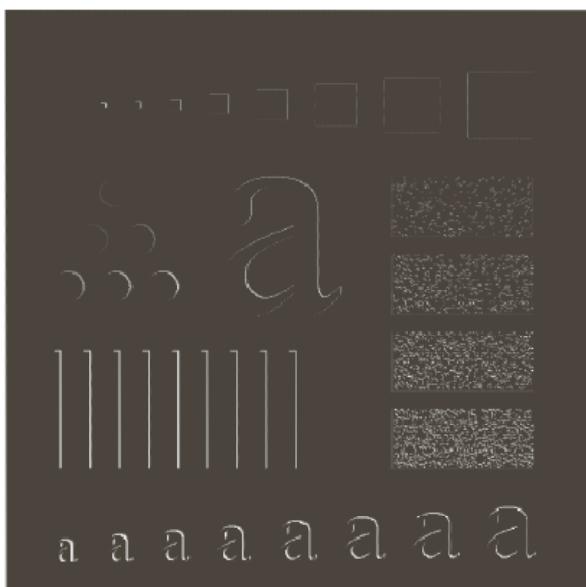
Where  $(v, w)$  and  $(x, y)$  are the coordinates of tie points in the input and reference images.



a b  
c d

**FIGURE 2.37**

Image registration.  
(a) Reference image. (b) Input (geometrically distorted image). Corresponding tie points are shown as small white squares near the corners.  
(c) Registered image (note the errors in the borders).  
(d) Difference between (a) and (c), showing more registration errors.



# Image Enhancement

**Definition:** Image enhancement is the process that improves the quality of the image for a specific application.

## Image Enhancement Methods

**Spatial Domain methods:** The term spatial domain refers to the 2-D image plane and the approaches manipulates the pixel of a given image for enhancement.

**Frequency Domain methods:** The methods in frequency domain manipulate the Fourier transform of a given image for enhancement.

---

# Image Enhancement in *Spatial Domain*

- Spatial Domain methods are procedures that *operate directly on pixels*.
- A process in Spatial Domain can be denoted by:

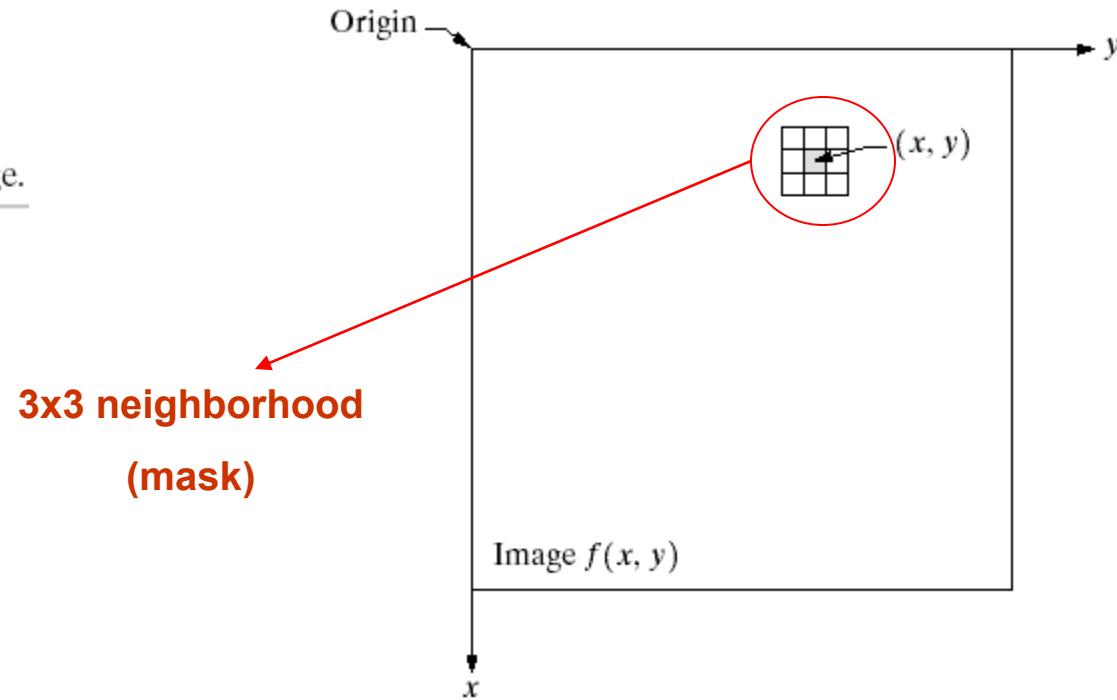
$$g(x, y) = T[f(x, y)]$$

- *f(x,y)* is the input and *g(x,y)* is the output image.
- *T* is the operator (*Transformation function*) on *f*, defined over a neighborhood of *(x,y)*.

A *neighborhood* over/about a pixel *(x,y)* uses a *square/rectangular subimage* centered at *(x,y)*.

# Image Enhancement in *Spatial Domain*

**FIGURE 3.1** A  
 $3 \times 3$   
neighborhood  
about a point  
( $x, y$ ) in an image.



- The center of the rectangular subimage is moved from the left top to the right bottom visiting all the pixels.
- $g(x,y)$  is calculated and output image is formed as each pixel is processed.

---

# **Image Enhancement in *Spatial Domain***

- *If a  $1 \times 1$  neighborhood is used for the transformation function, then the function is called the gray-level transformation function.*

$$s = T(r), \quad \text{where, } s = g(x, y) \quad \text{and} \quad r = f(x, y)$$

## **Point Processing**

- *Refers to the processing techniques where the enhancement at any point in an image depends only the gray-level at that point.*

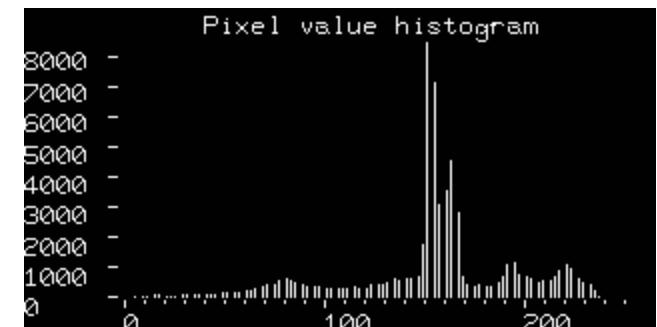
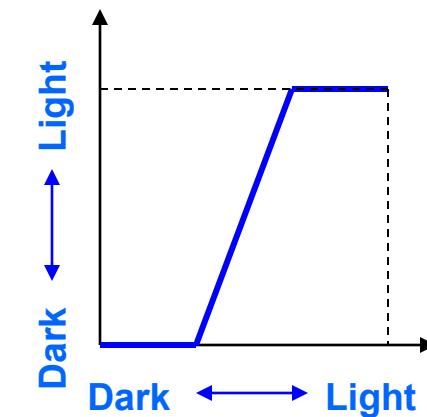
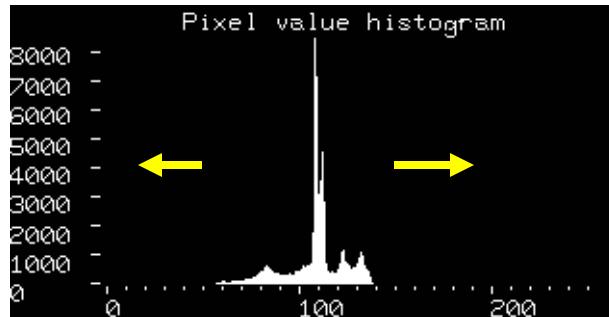
## **Mask Processing / Filtering**

- *Mask (also referred to as filter/kernel/template/window) is a 2-D array (i.e.  $3 \times 3$ ) defined around a pixel, where the values of the mask coefficients determine the nature of the process.*
- *Image enhancement methods using mask approach is called Mask Processing / Filtering.*

# Image Enhancement in *Spatial Domain*

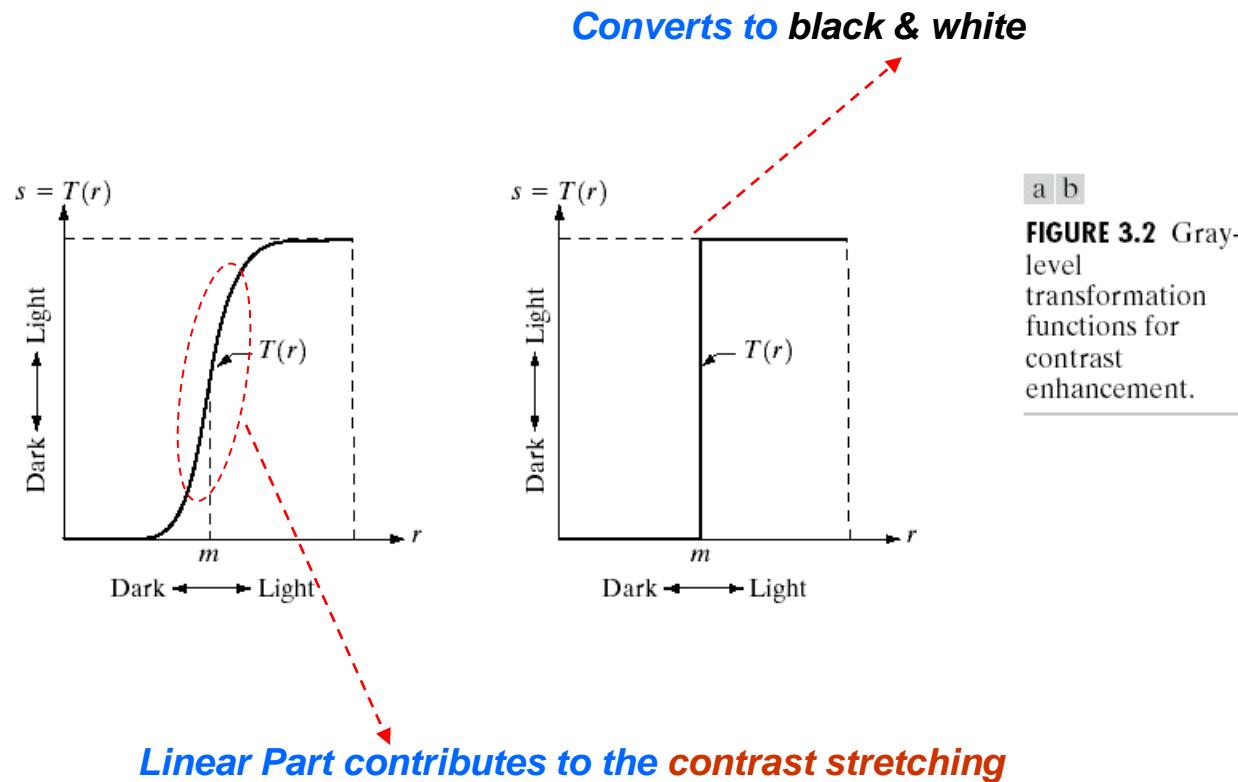
## Contrast Stretching

- *Contrast stretching (normalization) is a simple image enhancement technique that improves the contrast in an image by 'stretching' the range of intensity values it contains to span a desired range of values. Typically, it uses a Linear Scaling Function.*



# Image Enhancement in *Spatial Domain*

## Contrast Stretching

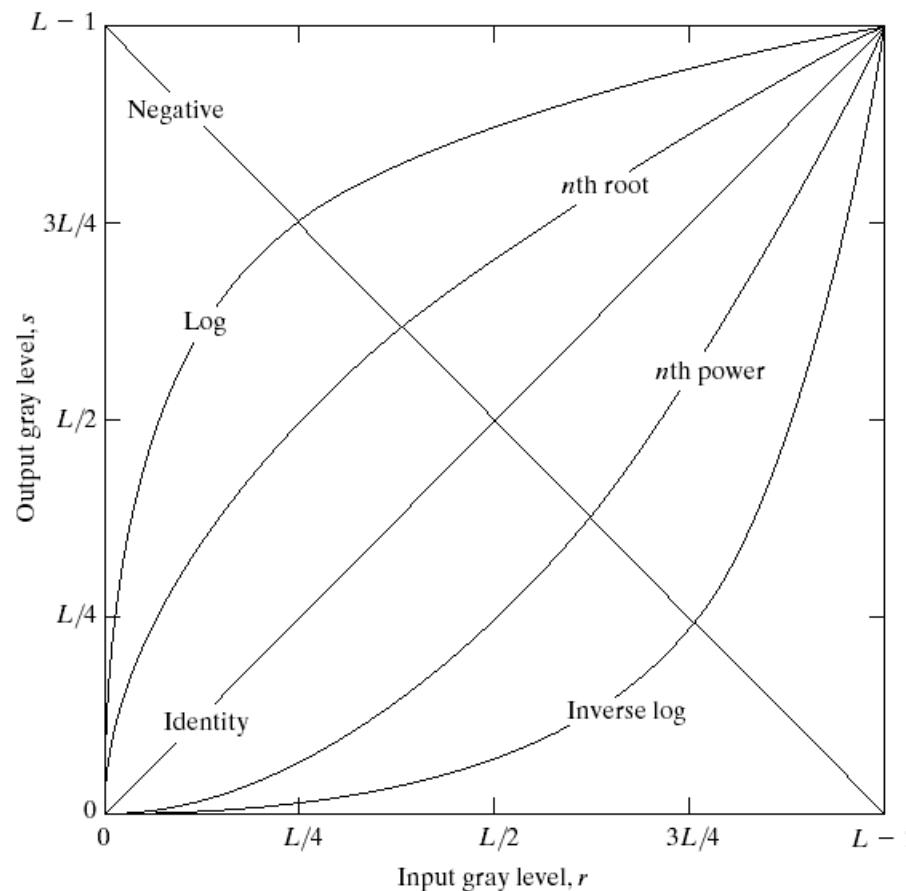


# Image Enhancement in *Spatial Domain*

## Basic Gray Level Transformations

- Given a gray-level transformation function  $T(r)$  where each pixel value  $r$  is mapped to pixel value  $s$ .

**FIGURE 3.3** Some basic gray-level transformation functions used for image enhancement.



*The transformation functions:*

- linear
- logarithmic (log/inverse-log)
- power-law ( $n^{\text{th}}$  power/ $n^{\text{th}}$  root)

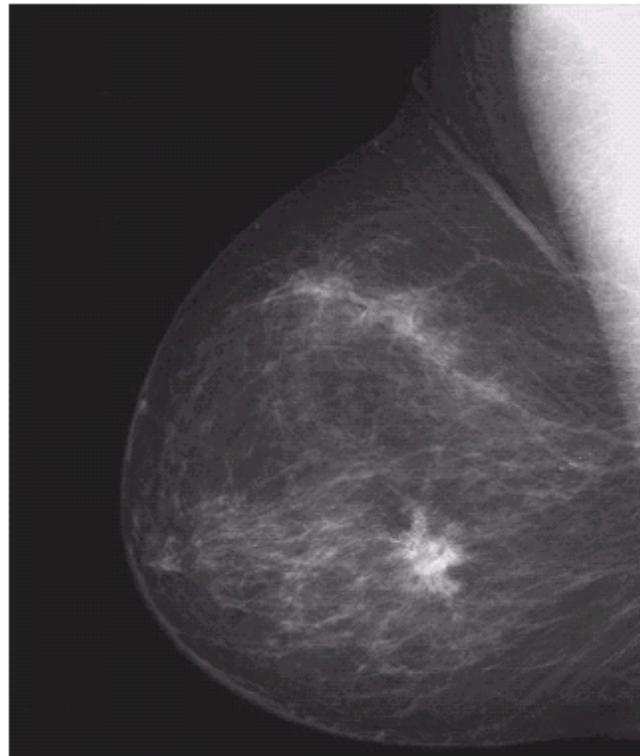
[0,  $L-1$ ] is the gray-level range

# Image Enhancement in *Spatial Domain*

## Image Negatives

- The negative of an image with gray levels in the range  $[0, L-1]$  can be obtained by

$$s = L - 1 - r$$



a | b

**FIGURE 3.4**  
(a) Original digital mammogram.  
(b) Negative image obtained using the negative transformation in Eq. (3.2-1).  
(Courtesy of G.E. Medical Systems.)

# Image Enhancement in *Spatial Domain*

## Logarithmic Transformations

- *The general form of the log transformations:*

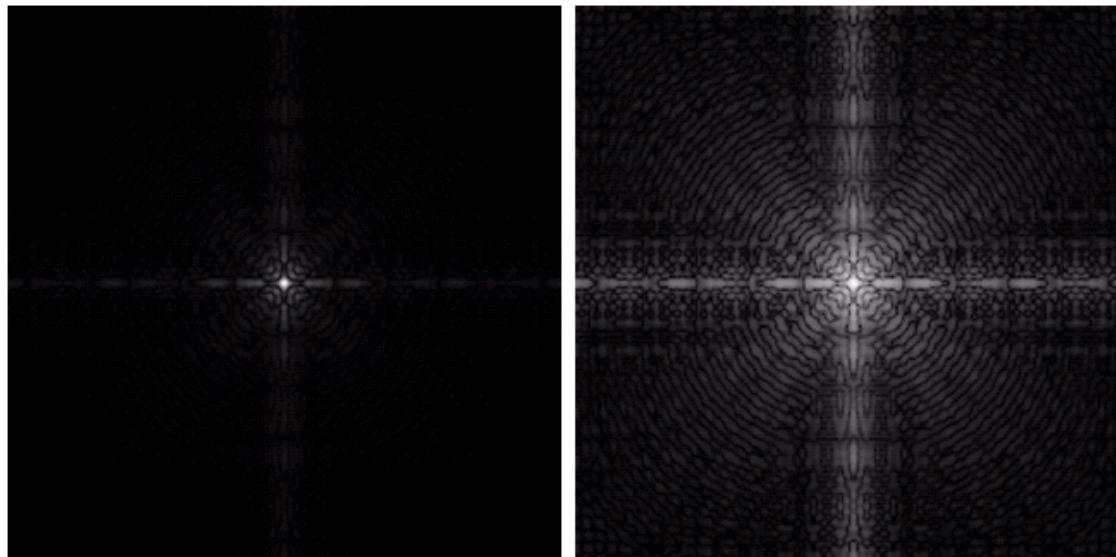
$$s = c \log(1 + r), \quad c \text{ is a const. and } r \geq 0$$

- *The Log Transforms compresses the dynamic range of images with large variations in pixel values.*
- *Typical application is to change the large dynamic range of a Fourier Spectrum to a smaller range for a clearer inspection.*

a b

**FIGURE 3.5**

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



---

# Image Enhancement in *Spatial Domain*

## Power-Law Transformations

- *The general form of the Power-law transformation:*

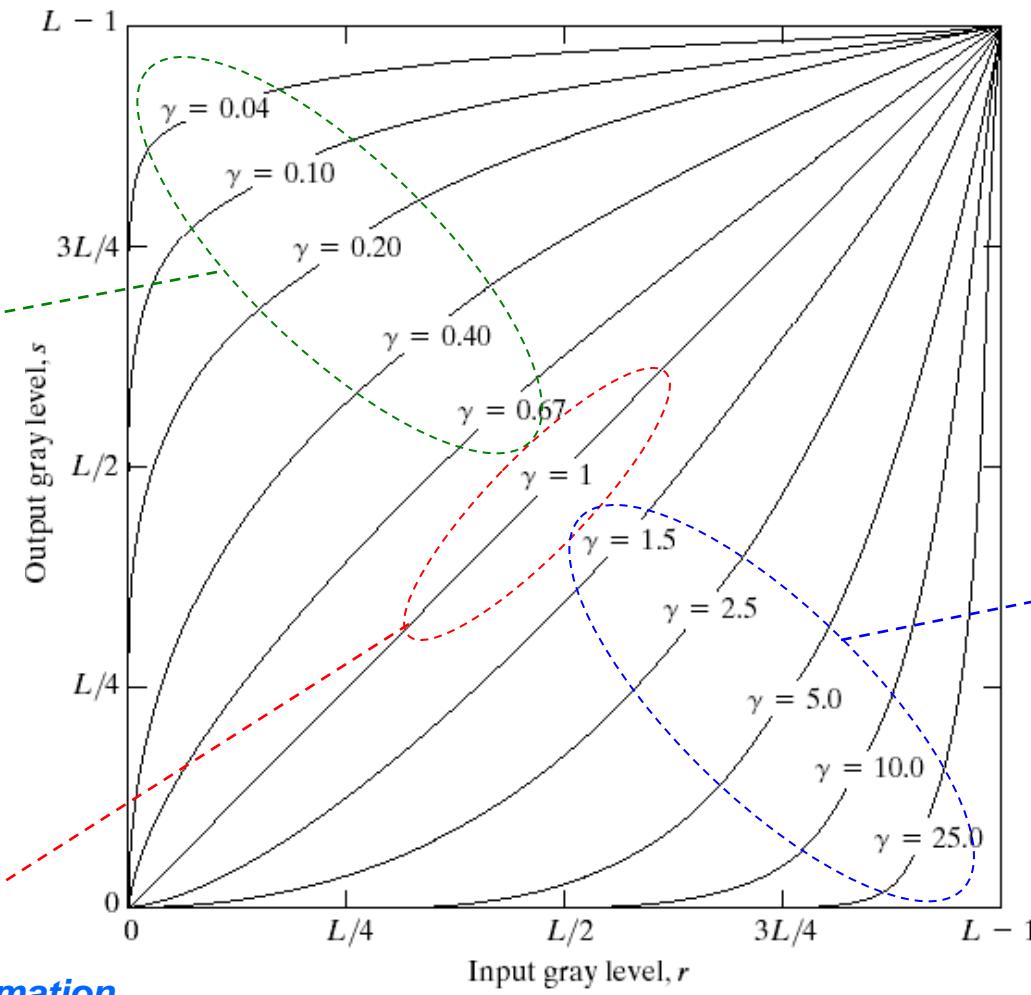
$$S = cr^\gamma, \text{ where } c \text{ and } \gamma \geq 0$$

- *Different transformation curves are obtained by varying  $\gamma$  (gamma).*
- *Many image capturing, printing and display devices use gamma correction which enhances the given images by power-law response phenomena.*

# Image Enhancement in *Spatial Domain*

## Power-Law Transformations – Gamma Correction

brighter images



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

Darker images

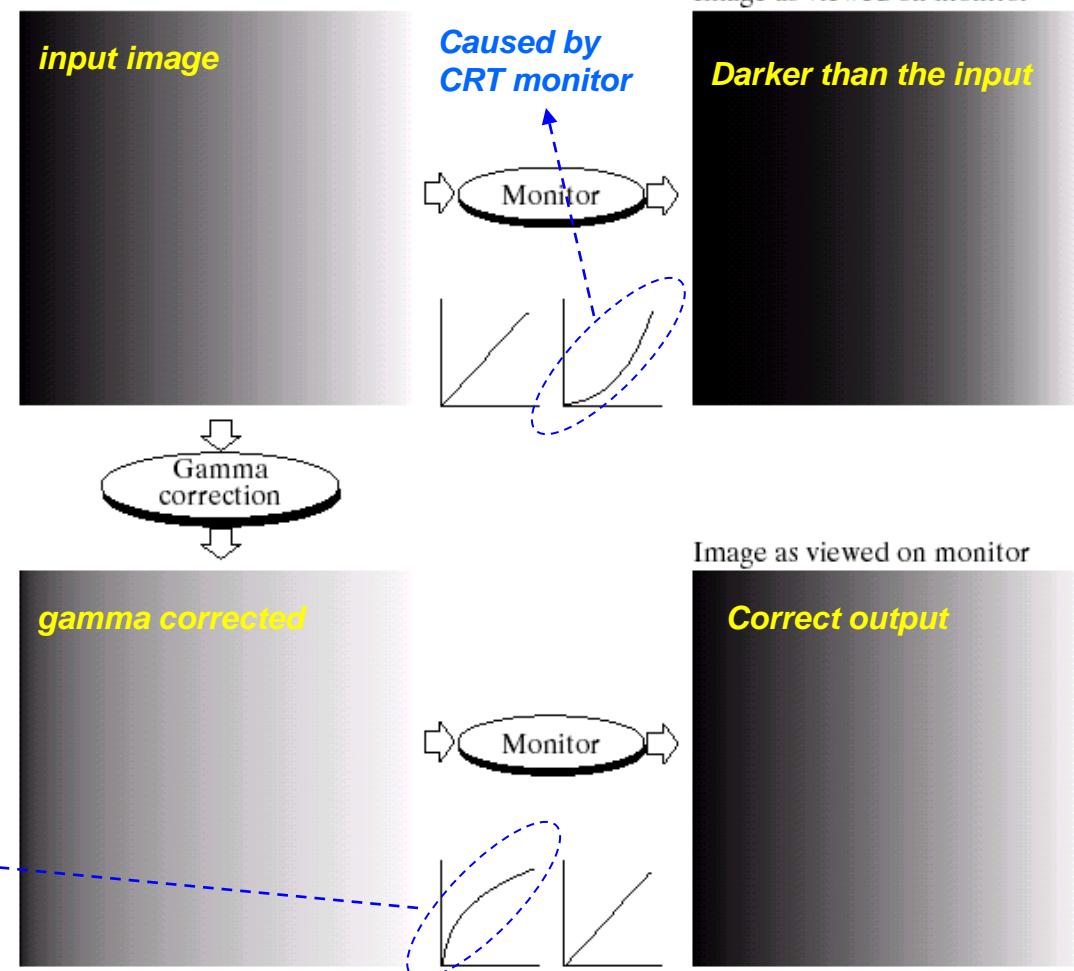
Identity transformation

# Image Enhancement in *Spatial Domain*

## Power-Law Transformation – Gamma Correction

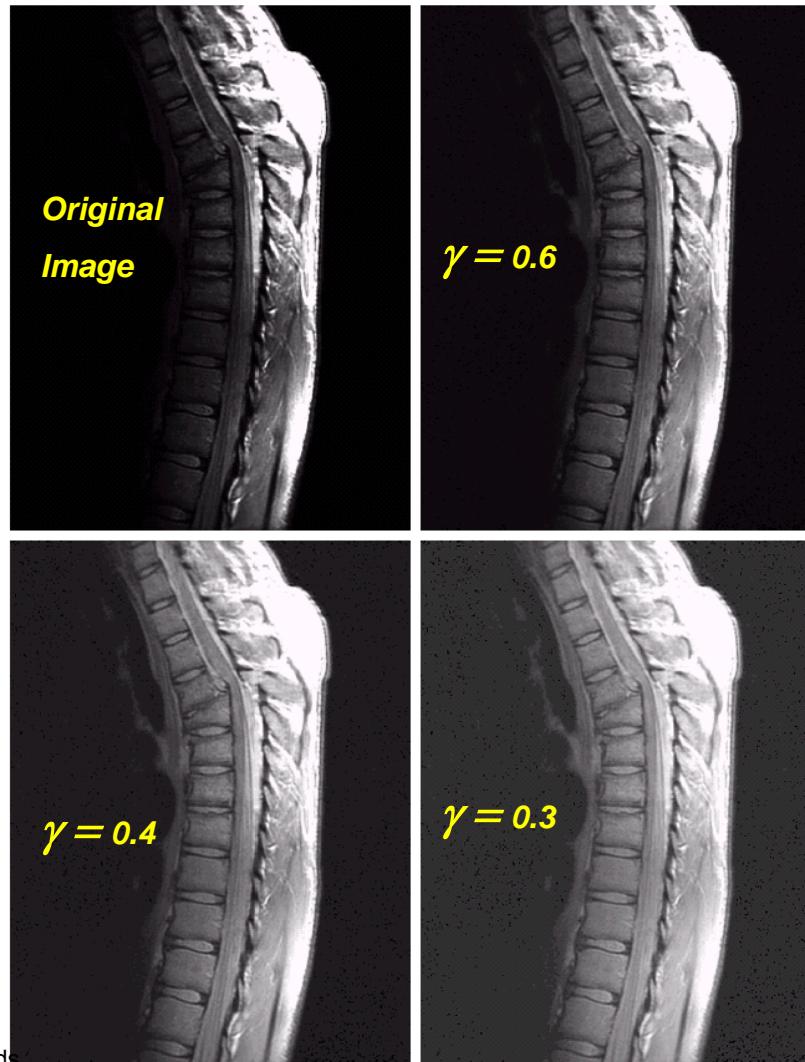
a b  
c d

**FIGURE 3.7**  
(a) Linear-wedge gray-scale image.  
(b) Response of monitor to linear wedge.  
(c) Gamma-corrected wedge.  
(d) Output of monitor.



# Image Enhancement in *Spatial Domain*

## Power-Law Transformation – Gamma Correction



a | b  
c | d

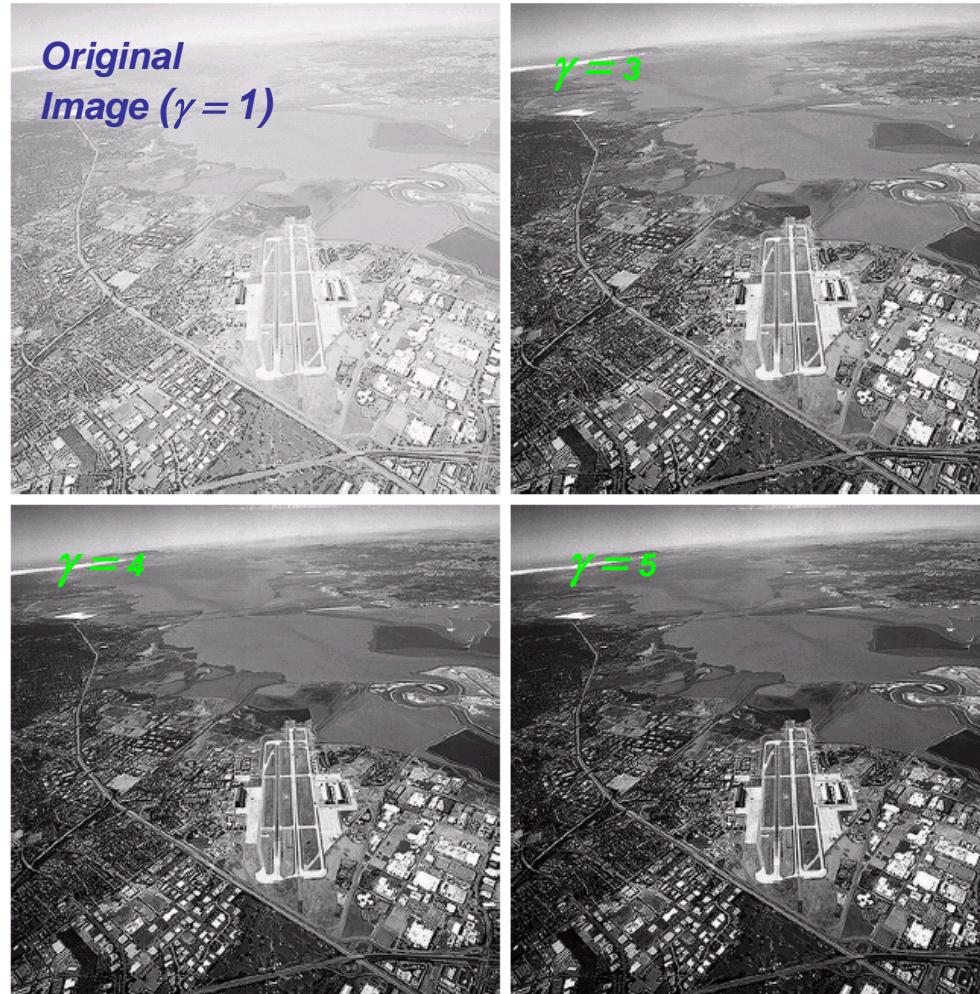
**FIGURE 3.8**  
(a) Magnetic resonance (MR) image of a fractured human spine.  
(b)–(d) Results of applying the transformation in Eq. (3.2-3) with  $c = 1$  and  $\gamma = 0.6, 0.4$ , and  $0.3$ , respectively.  
(Original image for this example courtesy of Dr. David R. Pickens, Department of Radiology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

# Image Enhancement in *Spatial Domain*

## Power-Law Transformation – Gamma Correction

a b  
c d

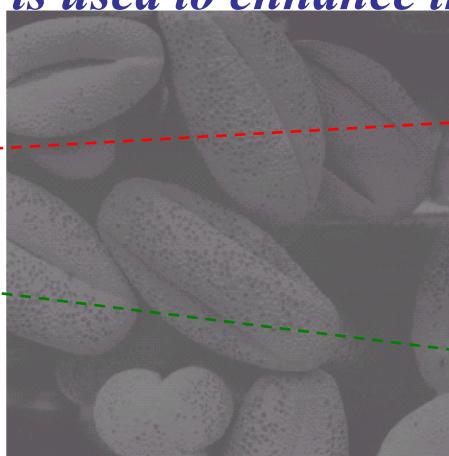
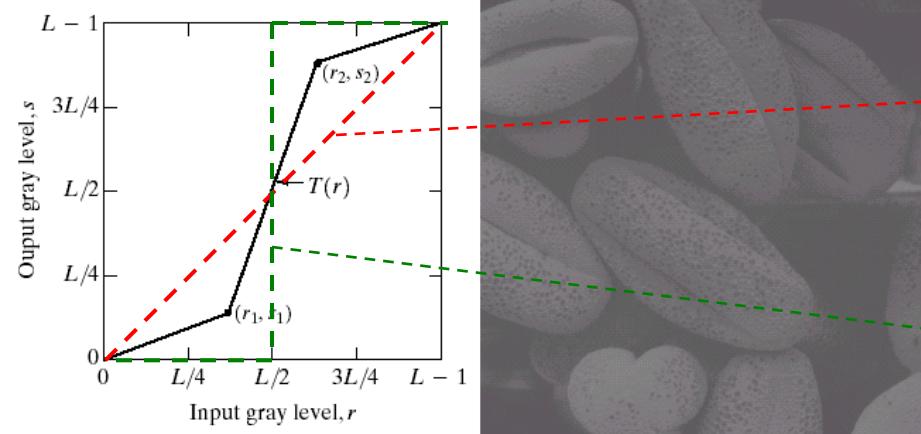
**FIGURE 3.9**  
(a) Aerial image.  
(b)–(d) Results of  
applying the  
transformation in  
Eq. (3.2-3) with  
 $c = 1$  and  
 $\gamma = 3.0, 4.0,$  and  
 $5.0,$  respectively.  
(Original image  
for this example  
courtesy of  
NASA.)



# Image Enhancement in *Spatial Domain*

## Piecewise-Linear Transformations – Contrast Stretching

- One of the simplest piecewise functions is the **contract stretching**, which is used to enhance the low contrast images.



a  
b  
c  
d

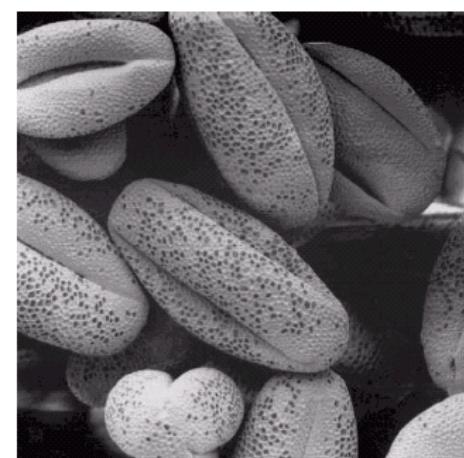
FIGURE 3.10

Contrast stretching.  
(a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching.

(d) Result of thresholding.  
(Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

No Change in the image

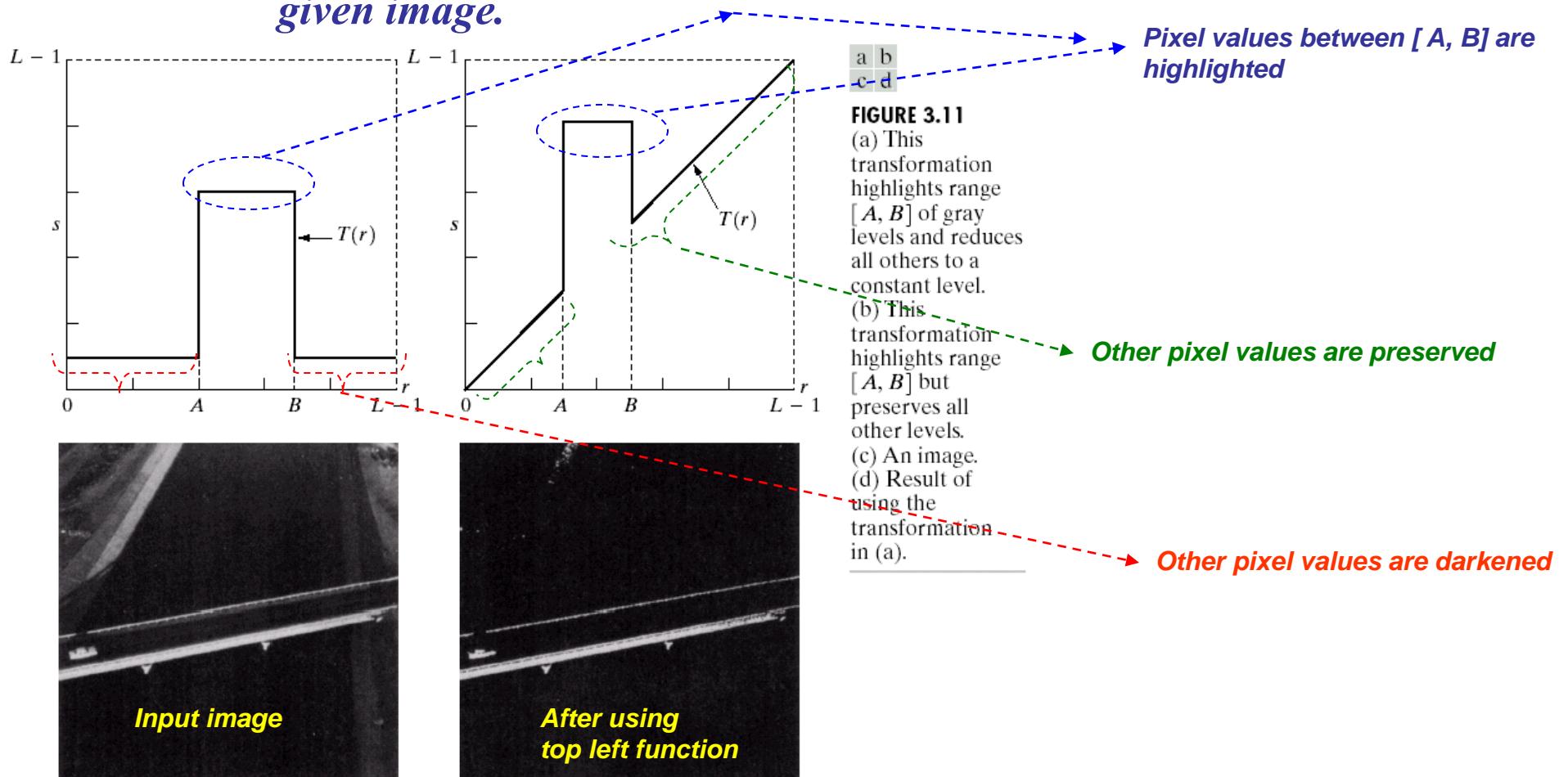
Image converted to black & white



# Image Enhancement in *Spatial Domain*

## Piecewise-Linear Transformations – Gray-level Slicing

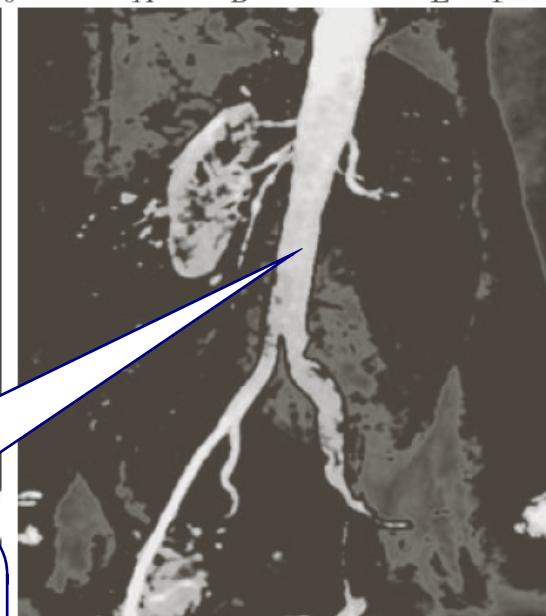
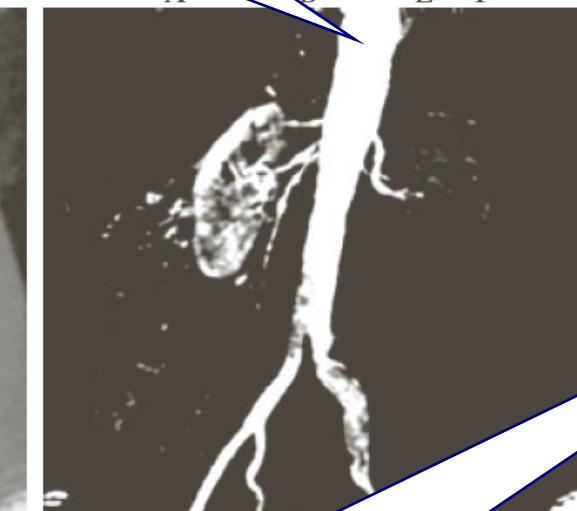
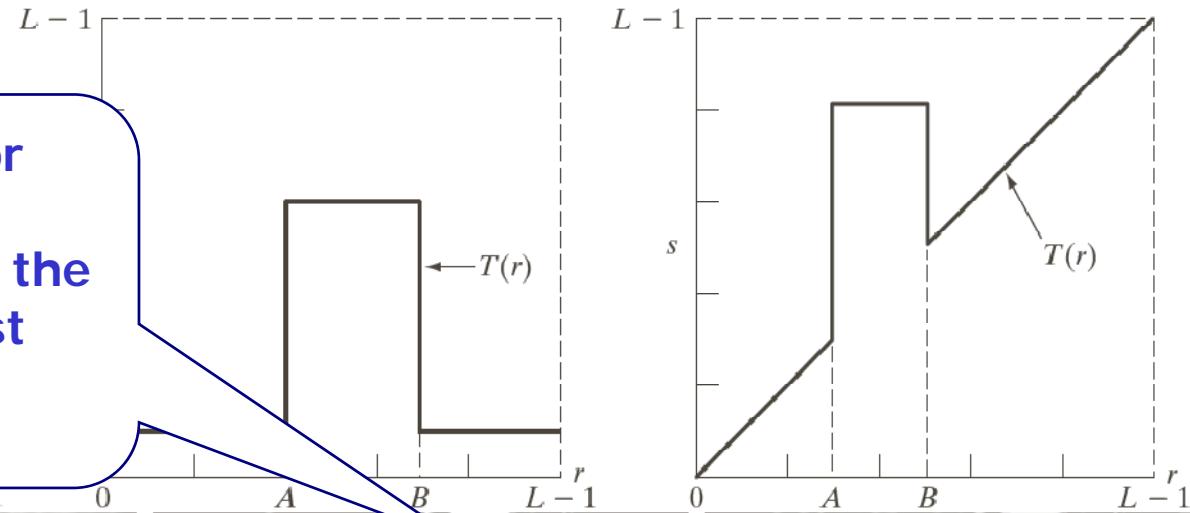
- *This technique is used to highlight a specific range of gray levels in a given image.*



a b

FIGURE 3.11 (a) This

Highlight the major blood vessels and study the shape of the flow of the contrast medium (to detect blockages, etc.)



a b c

FIGURE 3.12 (a) Aortic angiogram of the type illustrated in Fig. 3.11(a), with the range of interest indicated by points  $A$  and  $B$ . (b) Result of using the transformation in Fig. 3.11(b) to highlight the region of interest. (c) Result of applying the transformation  $T(r)$  to the image in (a). In (b) and (c), the blood vessels and kidneys were preserved. (Original image courtesy of Dr. Thomas R. Gest, University of Michigan Medical School.)

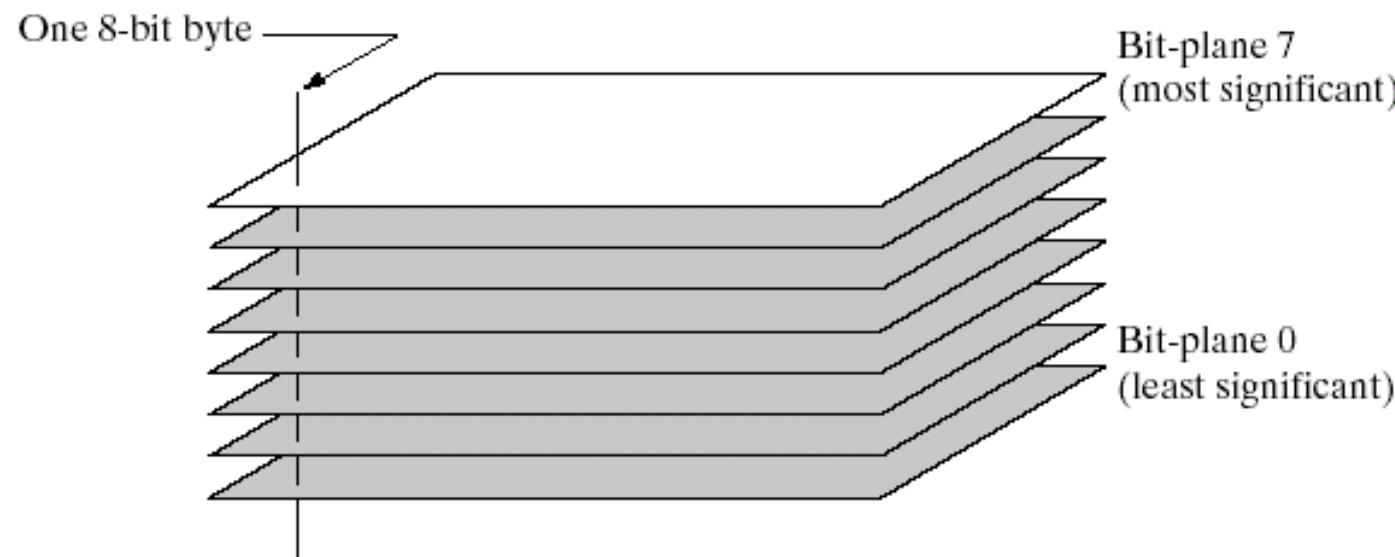
Measuring the actual flow of the contrast medium as a function of time in a series of images

mation of the type illustrated in Fig. 3.11(b) to highlight the region of interest end of the gray scale. (c) Result of applying the transformation  $T(r)$  to the image in (a). In (b) and (c), the blood vessels and kidneys were preserved. (Original image courtesy of Dr. Thomas R. Gest, University of Michigan Medical School.)

# Image Enhancement in *Spatial Domain*

## Piecewise-Linear Transformations – Bit Plane Slicing

- Instead of highlighting gray-level ranges, highlighting the contribution made by each bit is useful and used in image compression.
- Most significant bits contains the majority of the visually significant data.

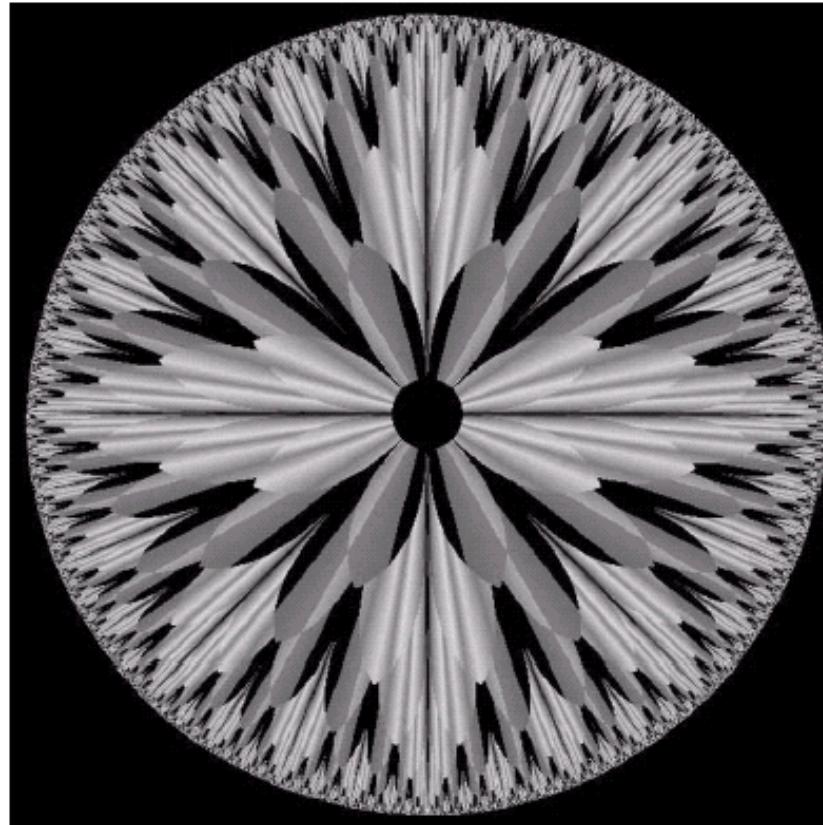


**FIGURE 3.12**  
Bit-plane representation of an 8-bit image.

# **Image Enhancement in *Spatial Domain***

## **Piecewise-Linear Transformations – Bit Plane Slicing**

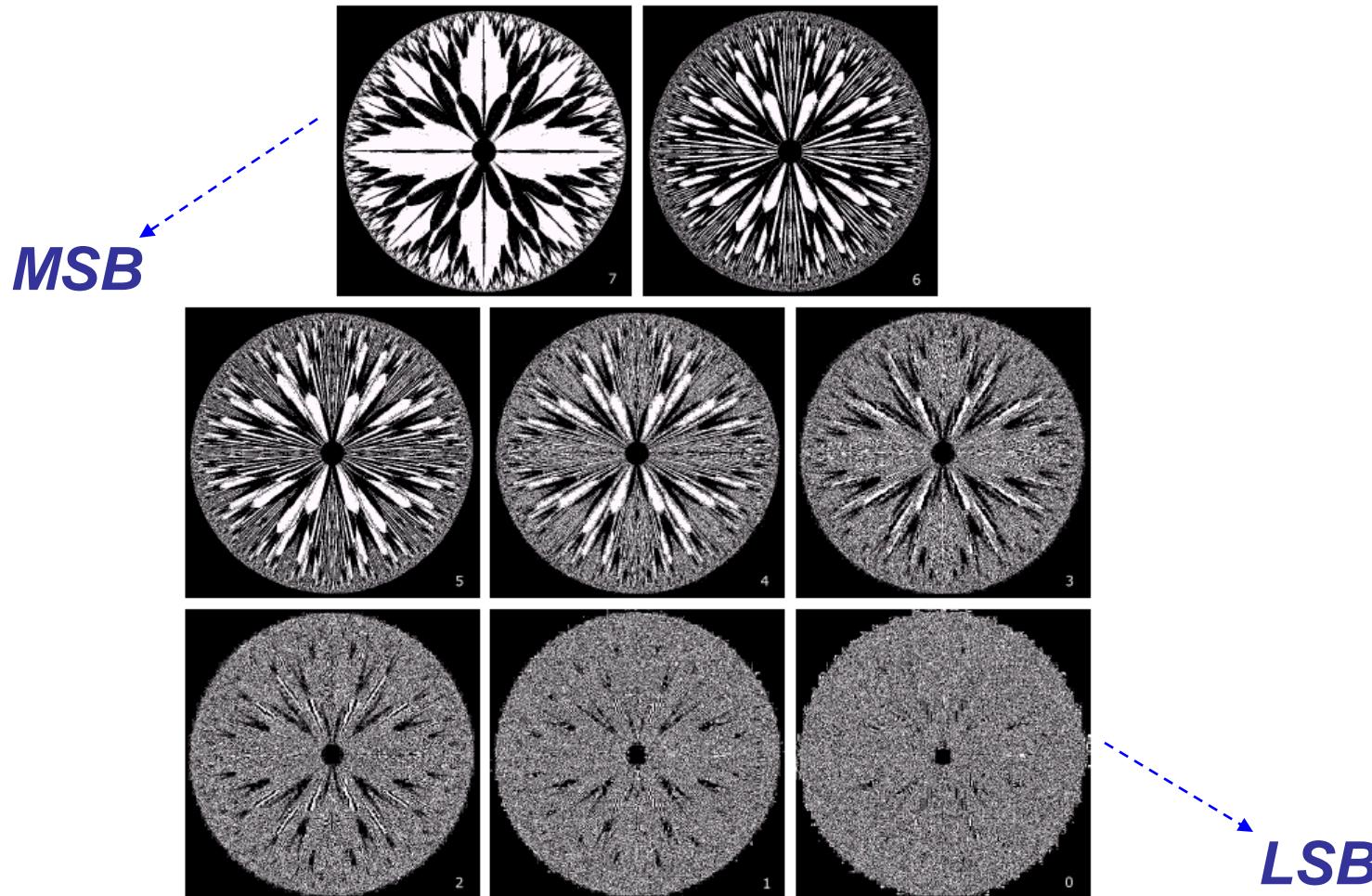
***Consider this 8-bit Fractal Image***



**FIGURE 3.13** An 8-bit fractal image. (A fractal is an image generated from mathematical expressions). (Courtesy of Ms. Melissa D. Binde, Swarthmore College, Swarthmore, PA.)

# Image Enhancement in *Spatial Domain*

## Piecewise-Linear Transformations – Bit Plane Slicing



**FIGURE 3.14** The eight bit planes of the image in Fig. 3.13. The number at the bottom, right of each image identifies the bit plane.

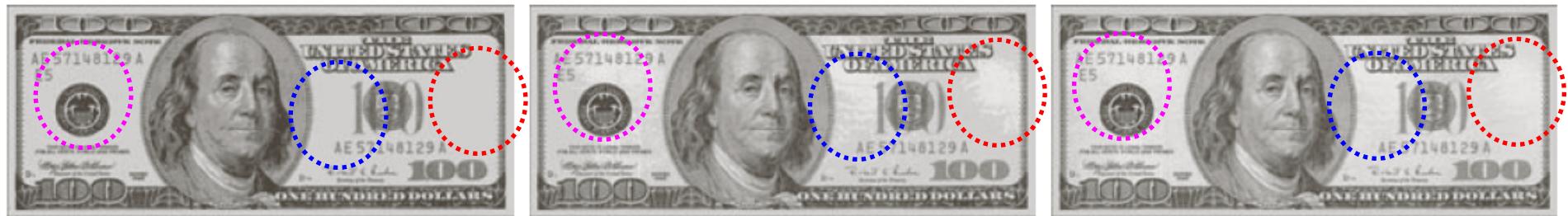
# Bit-plane Slicing



a	b	c
d	e	f
g	h	i

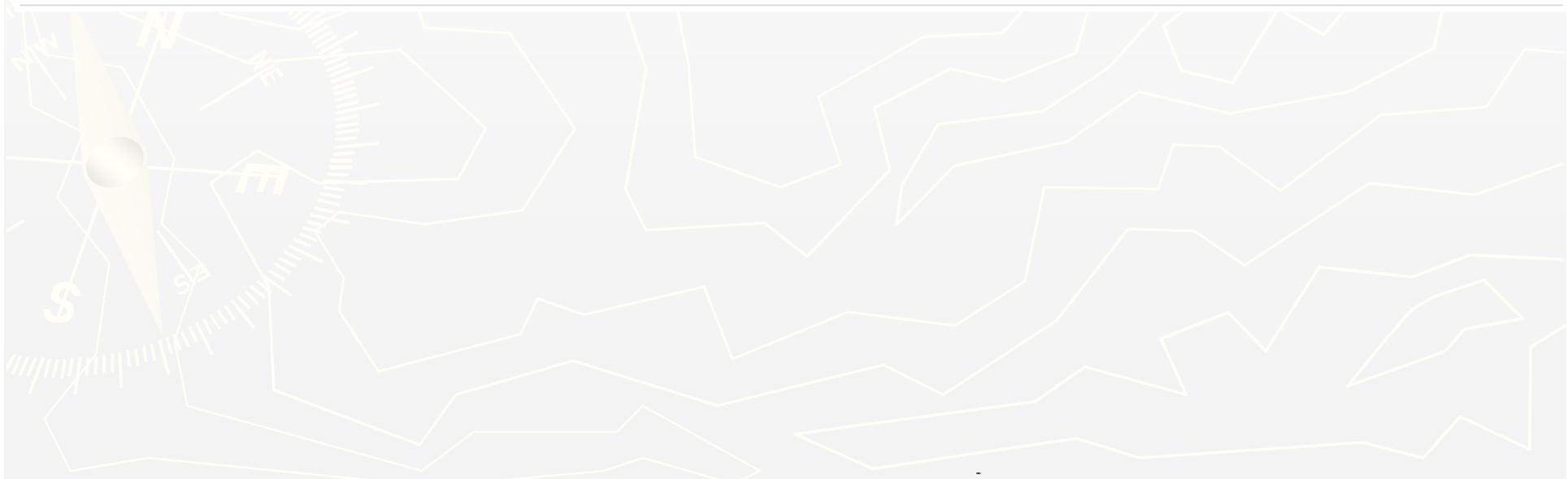
**FIGURE 3.14** (a) An 8-bit gray-scale image of size  $500 \times 1192$  pixels. (b) through (i) Bit planes 1 through 8, with bit plane 1 corresponding to the least significant bit. Each bit plane is a binary image.

# Bit-plane Slicing



a b c

**FIGURE 3.15** Images reconstructed using (a) bit planes 8 and 7; (b) bit planes 8, 7, and 6; and (c) bit planes 8, 7, 6, and 5. Compare (c) with Fig. 3.14(a).



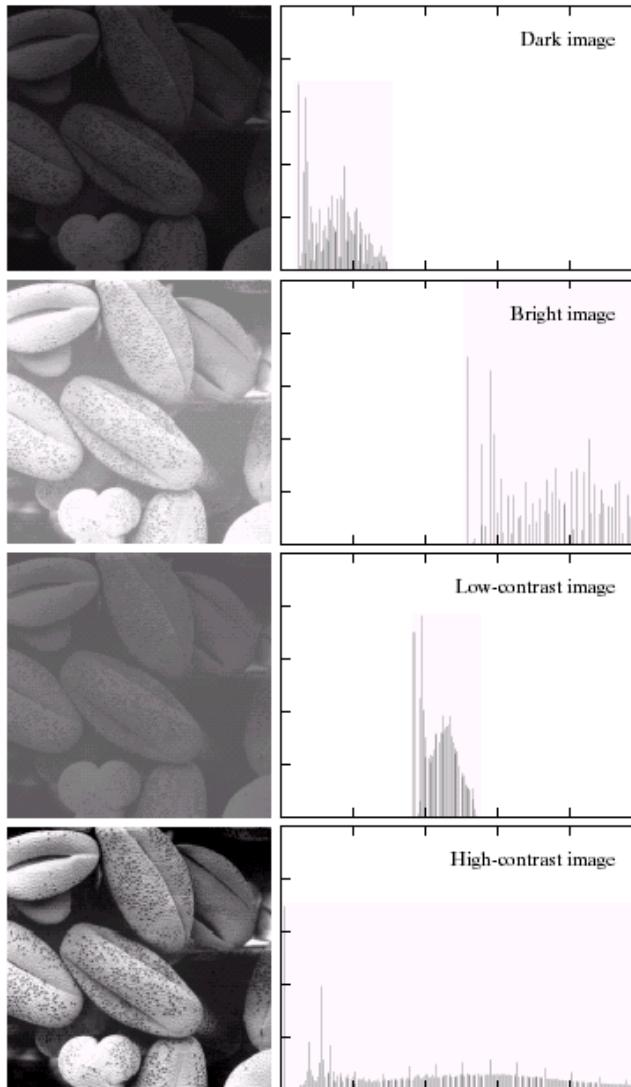
# Image Enhancement in *Spatial Domain*

## Histogram Processing

- *Histogram of a gray-level image in the range of [0, L-1] is the discrete function  $h(r_k)=n_k$ , where  $r_k$  is the  $k^{\text{th}}$  gray level and  $n_k$  is the number of pixels having gray level  $r_k$ .*
- *The histogram  $h(r_k)$  can be normalized to  $p(r_k)=n_k/n$ , for  $k=0,1,\dots,L-1$ .*
- *$p(r_k)$  can be considered to give an estimate of the probability of occurrence of ray level  $r_k$ .*
- *The cumulative of all components/probabilities of the normalized histogram is equal to 1.*

# Image Enhancement in *Spatial Domain*

## Histogram Processing

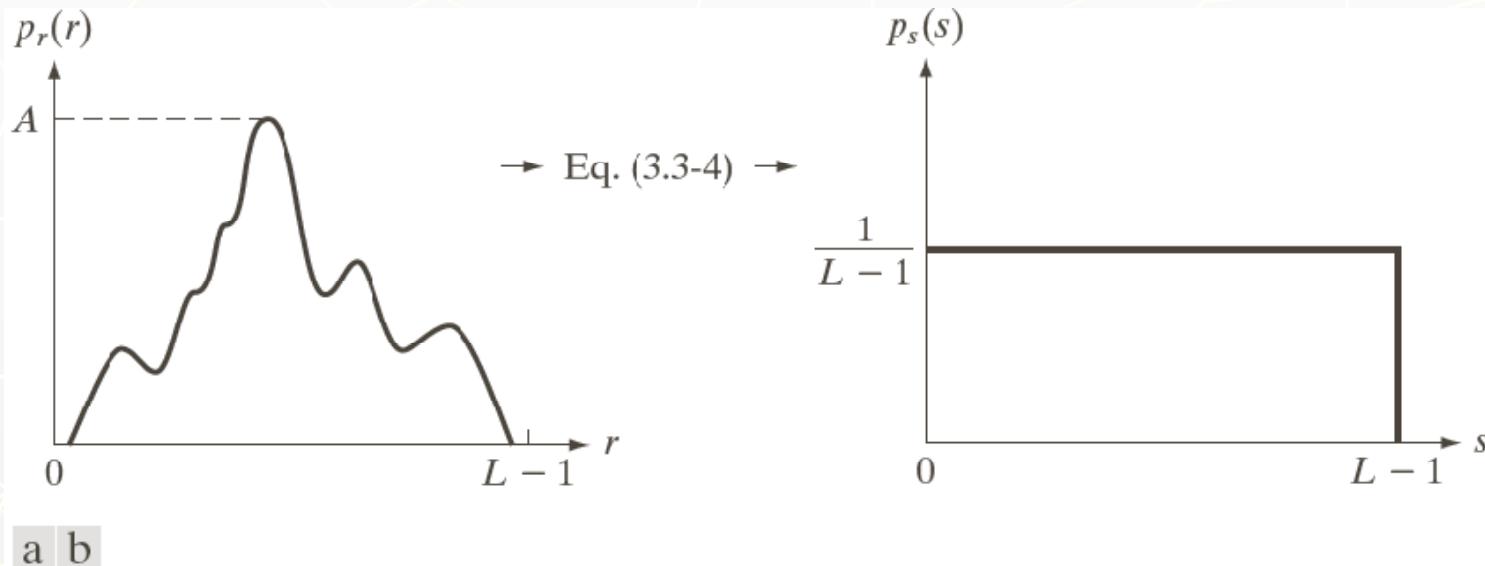


- *Histograms of four basic gray-level characteristics.*
- *The x axis shows the full range of the gray levels in the given image and the y axis corresponds to the number of occurrences of each gray  $h(r_k)$  or the probability of a pixel to have that gray-level value,  $p(r_k)$ .*

# Histogram Equalization

The intensity levels in an image may be viewed as random variables in the interval  $[0, L-1]$ .

Let  $p_r(r)$  and  $p_s(s)$  denote the probability density function (PDF) of random variables  $r$  and  $s$ .



**FIGURE 3.18** (a) An arbitrary PDF. (b) Result of applying the transformation in Eq. (3.3-4) to all intensity levels,  $r$ . The resulting intensities,  $s$ , have a uniform PDF, independently of the form of the PDF of the  $r$ 's.

---

# Image Enhancement in *Spatial Domain*

## Histogram Equalization

- *Histogram equalization is a method which increases the dynamic range of the gray-levels in a low-contrast image to cover full range of gray-levels.*
- *Histogram equalization is achieved by having a transformation function  $T(r)$ , which can be defined to be the Cumulative Distribution Function (CDF) of a given probability density function (PDF) of gray-levels in a given image.*
- *The histogram of an image can be considered as the approximation of the PDF of that image. Then the transformation function can be obtained by:*

$$S = T(r) = \int_0^r p_r(\omega) d\omega$$

- *where the intensity levels are considered as the continuous quantities normalized to the range [0,1] and  $p(r_k)$  denotes the PDF of a given input image.*

---

# Image Enhancement in *Spatial Domain*

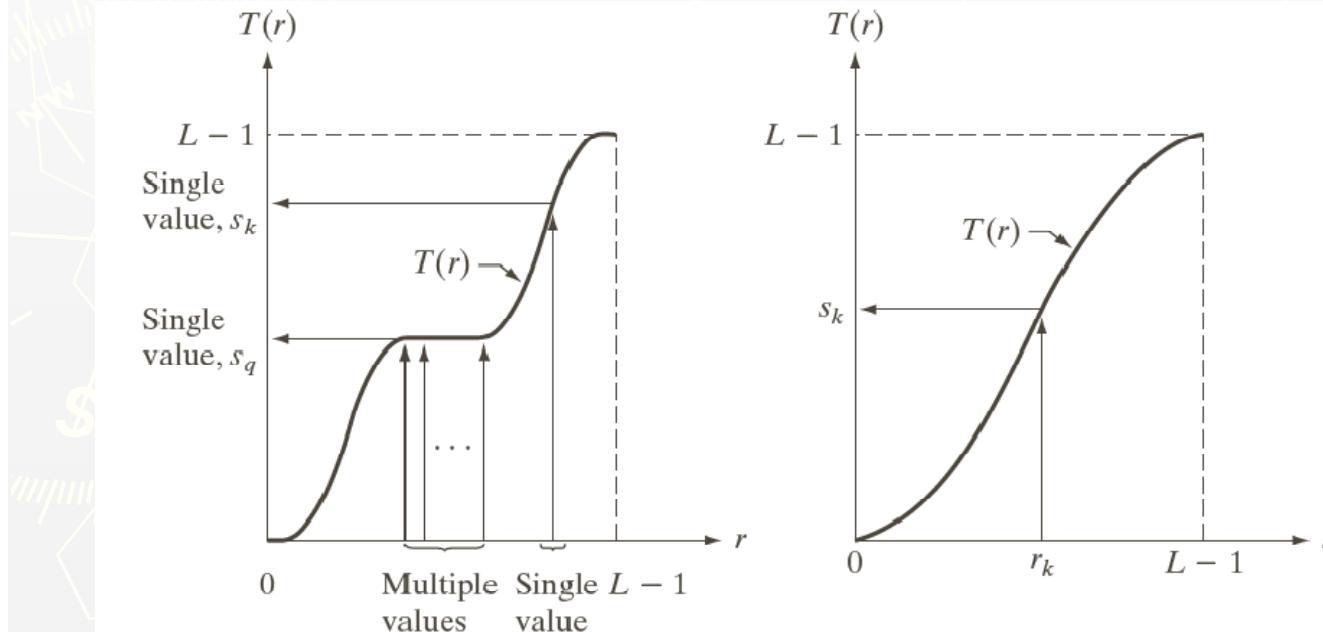
## Histogram Equalization

- *The probability function of the output levels is uniform, that is:*
- *The transformation generates an image whose intensity levels are equally likely covering the entire range [0,1].*
- *The net result is the **intensity-level equalization**. Note that the transformation function is simply the CDF .*

# Histogram Equalization

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

- a.  $T(r)$  is a strictly monotonically increasing function in the interval  $0 \leq r \leq L - 1$ ;
- b.  $0 \leq T(r) \leq L - 1$  for  $0 \leq r \leq L - 1$ .



a b

**FIGURE 3.17**

(a) Monotonically increasing function, showing how multiple values can map to a single value.  
(b) Strictly monotonically increasing function. This is a one-to-one mapping, both ways.

# Histogram Equalization

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

- a.  $T(r)$  is a strictly monotonically increasing function in the interval  $0 \leq r \leq L-1$ ;
- b.  $0 \leq T(r) \leq L-1$  for  $0 \leq r \leq L-1$ .

$T(r)$  is continuous and differentiable.

$$p_s(s)ds = p_r(r)dr$$

# Histogram Equalization

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

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

$$= (L-1) p_r(r)$$

$$p_s(s) = \frac{p_r(r) dr}{ds} = p_r(r) \cancel{\left( \frac{ds}{dr} \right)} = p_r(r) \cancel{\left( (L-1) p_r(r) \right)} = \frac{1}{L-1}$$

# Example

Suppose that the (continuous) intensity values in an image have the PDF

$$p_r(r) = \begin{cases} \frac{2r}{(L-1)^2}, & \text{for } 0 \leq r \leq L-1 \\ 0, & \text{otherwise} \end{cases}$$

Find the transformation function for equalizing the image histogram.

# Example

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

$$= (-1) \int_0^r \frac{2}{(L-1)^2}$$

$$= \frac{r^2}{L-1}$$

# Histogram Equalization

Continuous case:

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

Discrete values:

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

---

# Image Enhancement in *Spatial Domain*

## Histogram Equalization

- *Histograms are discrete quantities and we deal with probabilities and summations instead of PDFs and integrals.*
- *Let  $p(r_k)$  be the probability of occurrence of gray-level  $r_k$  in a given image:*

$$p(r_k) = \frac{n_k}{n} \quad k = 0, 1, 2, \dots, L-1$$

• *Then,*

$$\begin{aligned}s_k &= T(r_k) = \sum_{j=0}^k p(r_j) \\ &= \sum_{j=0}^k \frac{n_j}{n} \quad k = 0, 1, 2, \dots, L-1\end{aligned}$$

# Example: Histogram Equalization

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

# Example: Histogram Equalization

$r_k$	$n_k$	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02

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

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

$$s_2 = 4.55 \rightarrow 5$$

$$s_4 = 6.23 \rightarrow 6$$

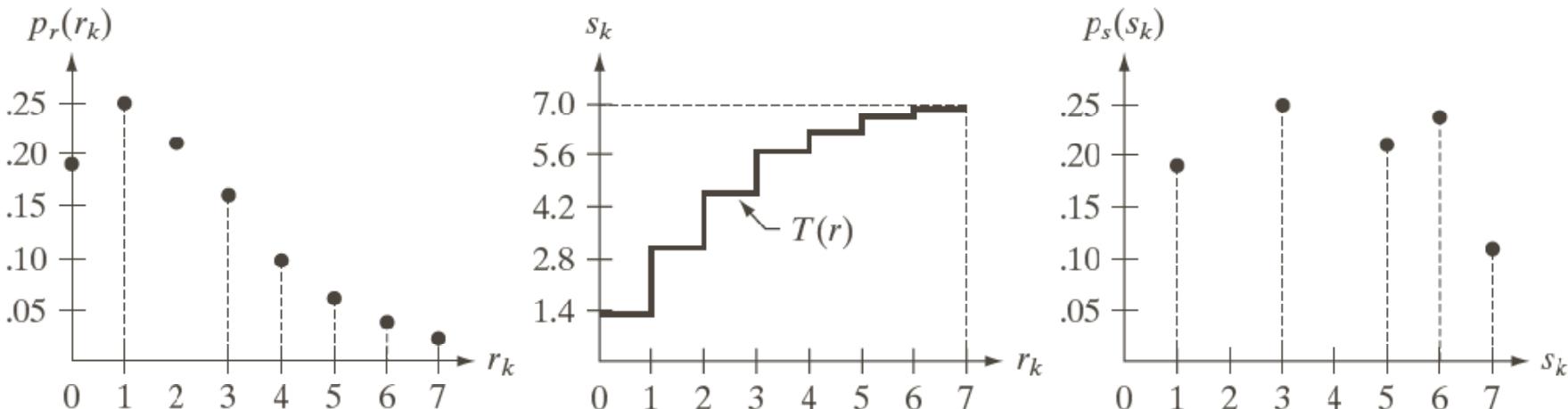
$$s_6 = 6.86 \rightarrow 7$$

$$s_3 = 5.67 \rightarrow 6$$

$$s_5 = 6.65 \rightarrow 7$$

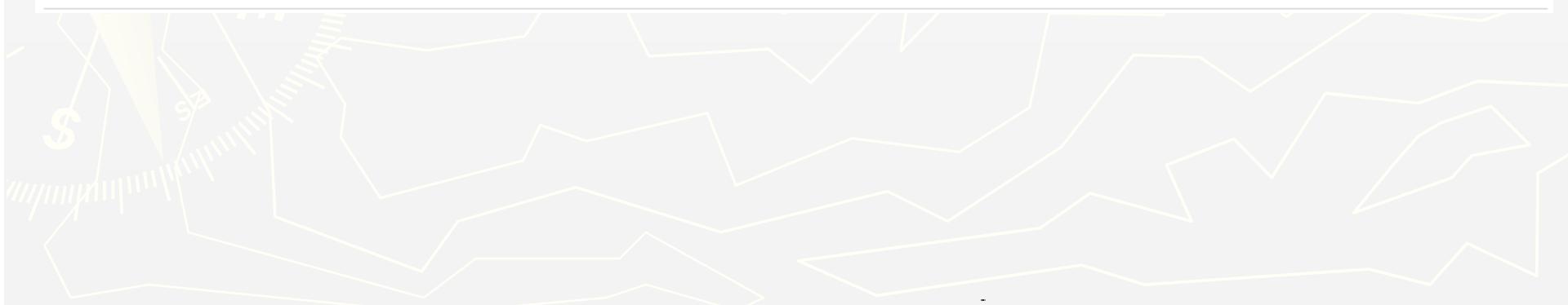
$$s_7 = 7.00 \rightarrow 7$$

# Example: Histogram Equalization



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.

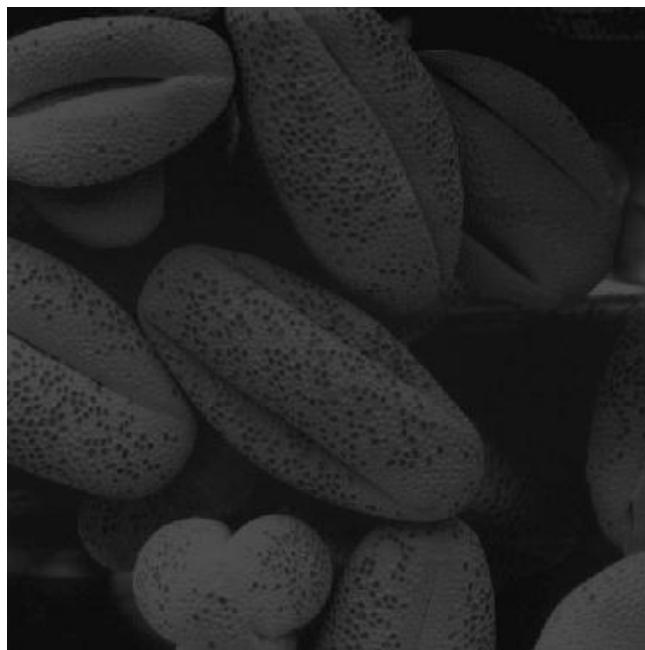


---

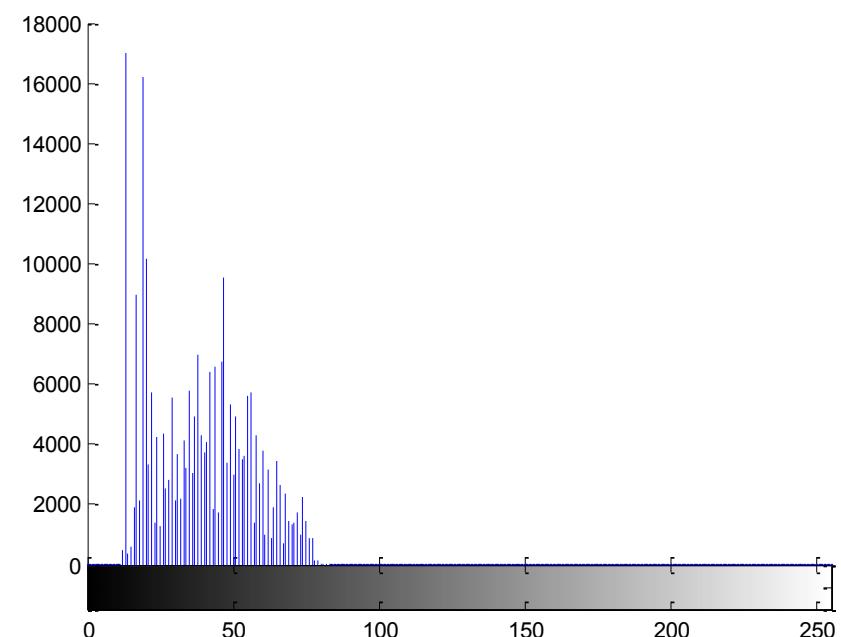
# Image Enhancement in *Spatial Domain*

## Histogram Equalization

Low-contrast image



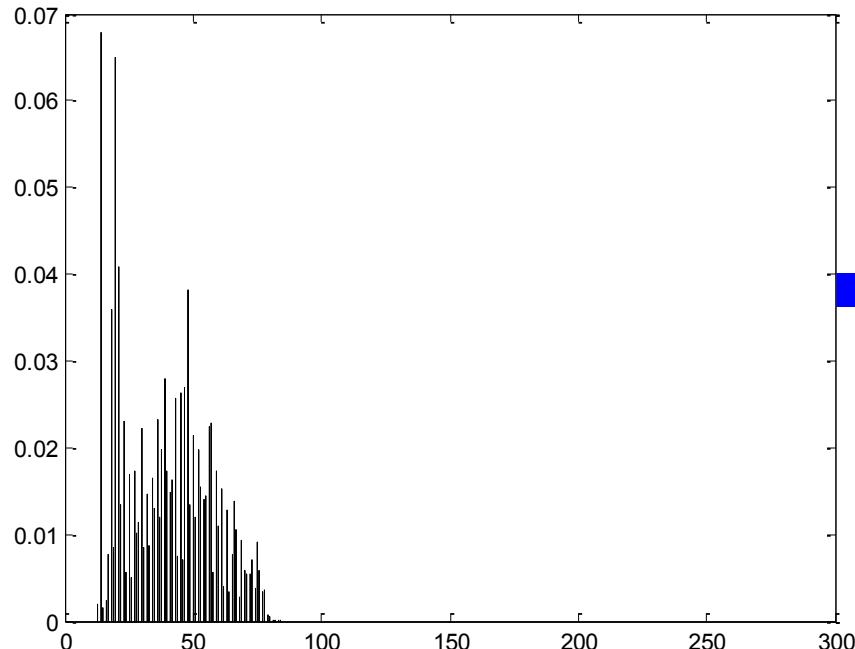
Corresponding histogram



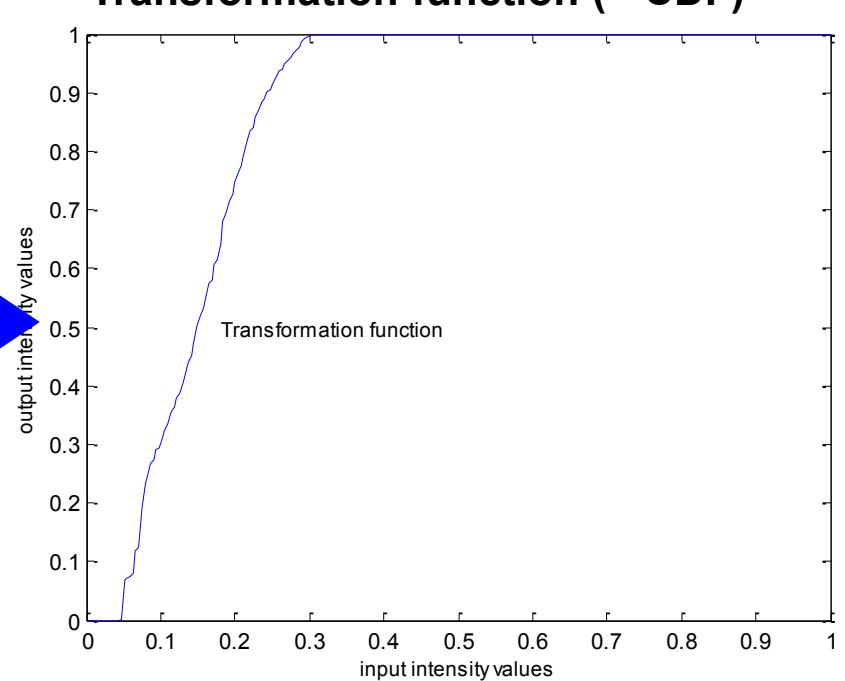
# Image Enhancement in *Spatial Domain*

## Histogram Equalization

Normalized Histogram of the input image (~ PDF)



Transformation function (~ CDF)

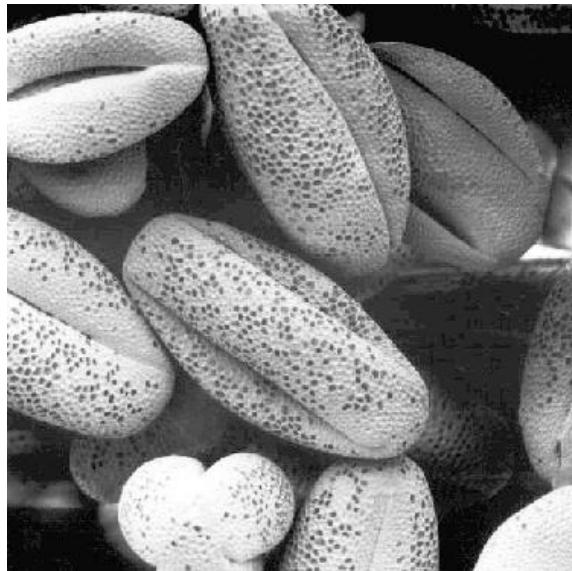


---

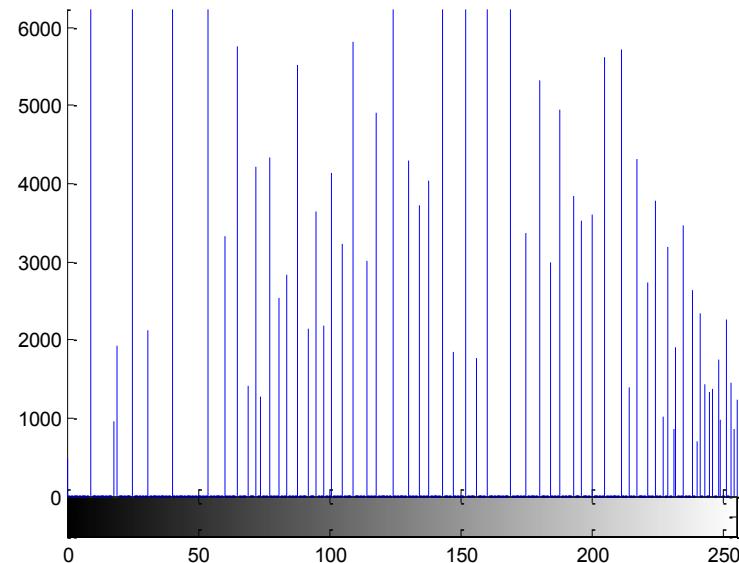
# Image Enhancement in *Spatial Domain*

## Histogram Equalization

Enhanced high-contrast image

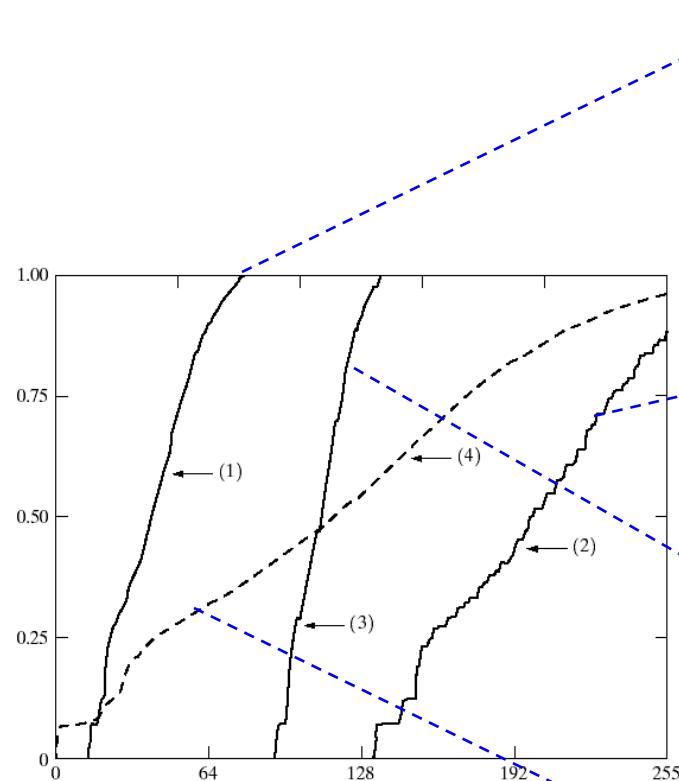


Corresponding histogram

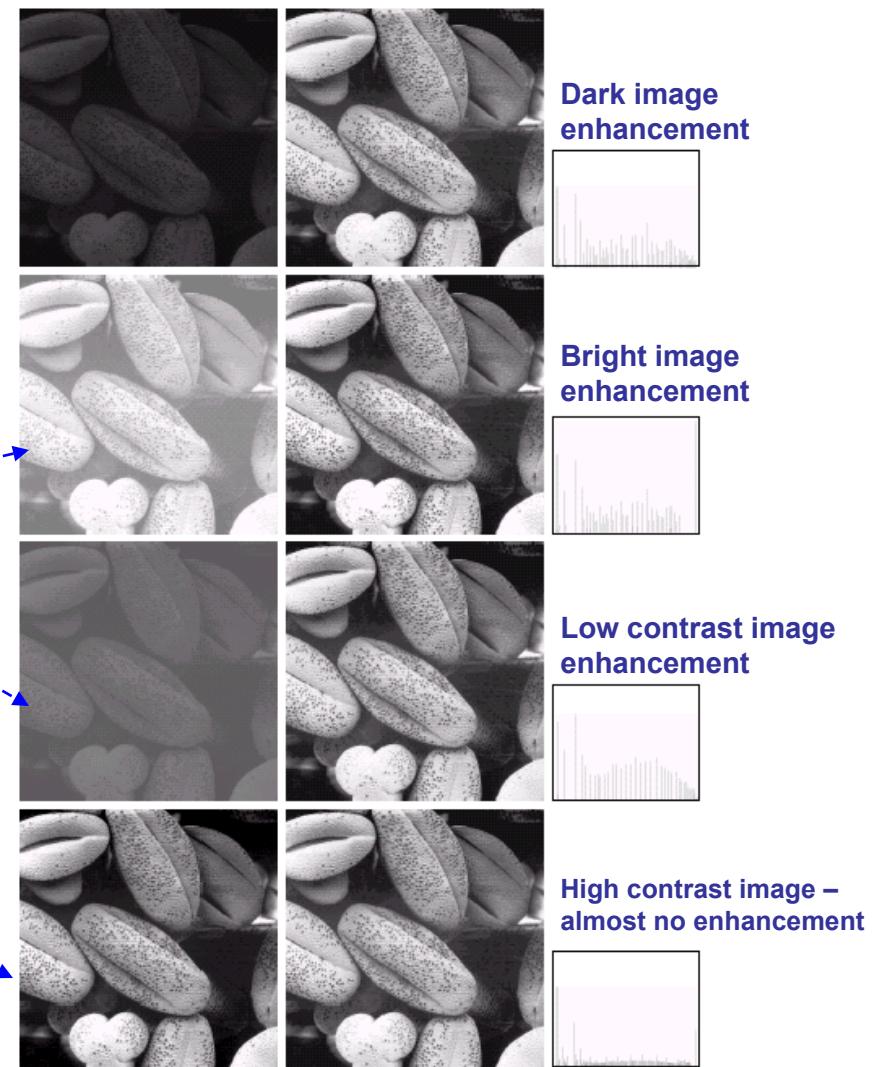


# Image Enhancement in *Spatial Domain*

## Histogram Equalization



**FIGURE 3.18**  
Transformation  
functions (1)  
through (4) were  
obtained from the  
histograms of the  
images in  
Fig.3.17(a), using  
Eq. (3.3-8).



# Question

Is histogram equalization always good?

No



# Histogram Matching

## Histogram matching (histogram specification)

— generate a processed image that has a specified histogram

Let  $p_r(r)$  and  $p_z(z)$  denote the continuous probability density functions of the variables  $r$  and  $z$ .  $p_z(z)$  is the specified probability density function.

Let  $s$  be the random variable with the probability

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

Define a random variable  $z$  with the probability

$$G(z) = (L - 1) \int_0^z p_z(t) dt = s$$

# Histogram Matching

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

$$G(z) = (L-1) \int_0^z p_z(t) dt = s$$

$$z = G^{-1}(s) = G^{-1}[T(r)]$$

# Histogram Matching: Procedure

- ▶ Obtain  $p_r(r)$  from the input image and then obtain the values of  $s$

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

- ▶ Use the specified PDF and obtain the transformation function  $G(z)$

$$G(z) = (L-1) \int_0^z p_z(t) dt = s$$

- ▶ Mapping from  $s$  to  $z$

$$z = G^{-1}(s)$$

# Histogram Matching: Example

Assuming continuous intensity values, suppose that an image has the intensity PDF

$$p_r(r) = \begin{cases} \frac{2r}{(L-1)^2}, & \text{for } 0 \leq r \leq L-1 \\ 0, & \text{otherwise} \end{cases}$$

Find the transformation function that will produce an image whose intensity PDF is

$$p_z(z) = \begin{cases} \frac{3z^2}{(L-1)^3}, & \text{for } 0 \leq z \leq (L-1) \\ 0, & \text{otherwise} \end{cases}$$

# Histogram Matching: Example

Find the histogram equalization transformation for the input image

$$s = T(r) = (L-1) \int_0^r p_r(w) dw = (L-1) \int_0^r \frac{2w}{(L-1)^2} dw = \frac{r^2}{L-1}$$

Find the histogram equalization transformation for the specified histogram

$$G(z) = (L-1) \int_0^z p_z(t) dt = (L-1) \int_0^z \frac{3t^2}{(L-1)^3} dt = \frac{z^3}{(L-1)^2} = s$$

The transformation function

$$z = \left[ (L-1)^2 s \right]^{1/3} = \left[ (L-1)^2 \frac{r^2}{L-1} \right]^{1/3} = \left[ (L-1)r^2 \right]^{1/3}$$

# Histogram Matching: Discrete Cases

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

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

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

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

- ▶ Mapping from  $s_k$  to  $z_q$

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

# Example: Histogram Matching

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

$z_q$	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

# Example: Histogram Matching

Obtain the scaled histogram-equalized values,

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

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

Compute all the values of the transformation function  $G$ ,

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

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

$$G(z_3) = 1.05 \rightarrow 1$$

$$\rightarrow 5$$

$$\rightarrow 7$$

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

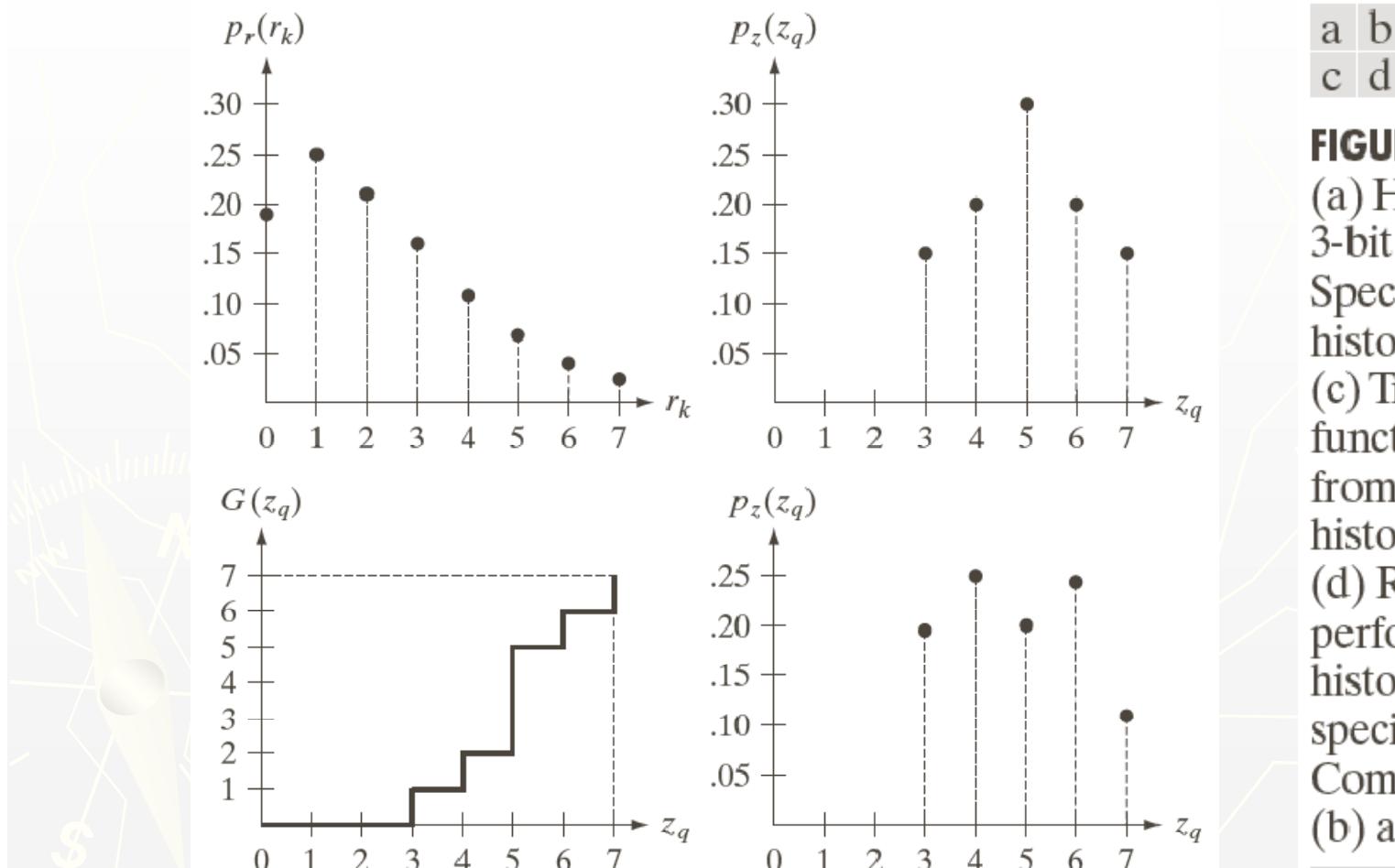
$$G(z_4) = 2.45 \rightarrow 2$$

$$G(z_6) = 5.95 \rightarrow 6$$

$r_k$	$n_k$	$G(z_5) = \frac{p_r(r_k)}{M} = \frac{n_k/MN}{0.19}$
$r_0 = 0$	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

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

# Example: Histogram Matching



**FIGURE 3.22**

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

# Example: Histogram Matching

Obtain the scaled histogram-equalized values,

$$s_0 = 1, s_1 = 3, s_2 = 5, s_3 = 6, s_4 = 7,$$
$$s_5 = 7, s_6 = 7, s_7 = 7.$$

Compute all the values of the transformation function  $G$ ,

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

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

$$G(z_3) = 1.05 \rightarrow 1 \quad s_0$$

$$G(z_5) = 4.55 \rightarrow 5 \quad s_2$$

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

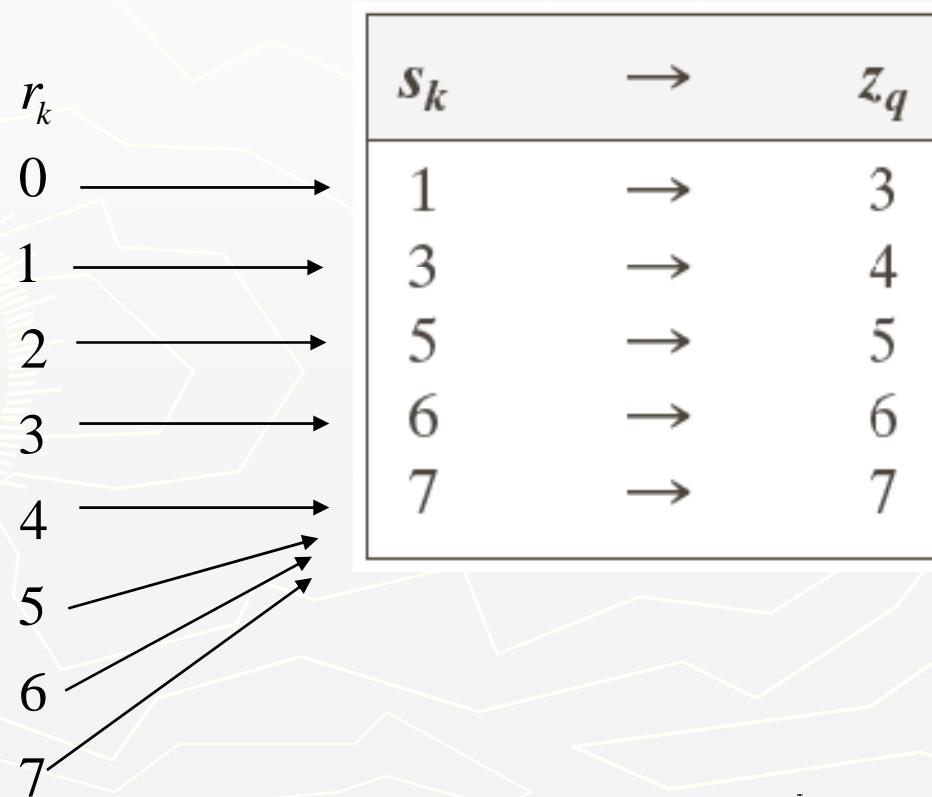
$$G(z_2) = 0.00 \rightarrow 0$$

$$G(z_4) = 2.45 \rightarrow 2 \quad s_1$$

$$G(z_6) = 5.95 \rightarrow 6 \quad s_3$$

# Example: Histogram Matching

$s_0 = 1, s_1 = 3, s_2 = 5, s_3 = 6, s_4 = 7,$   
 $s_5 = 7, s_6 = 7, s_7 = 7.$



# Example: Histogram Matching

$$r_k \rightarrow z_q$$

$$0 \rightarrow 3$$

$$1 \rightarrow 4$$

$$2 \rightarrow 5$$

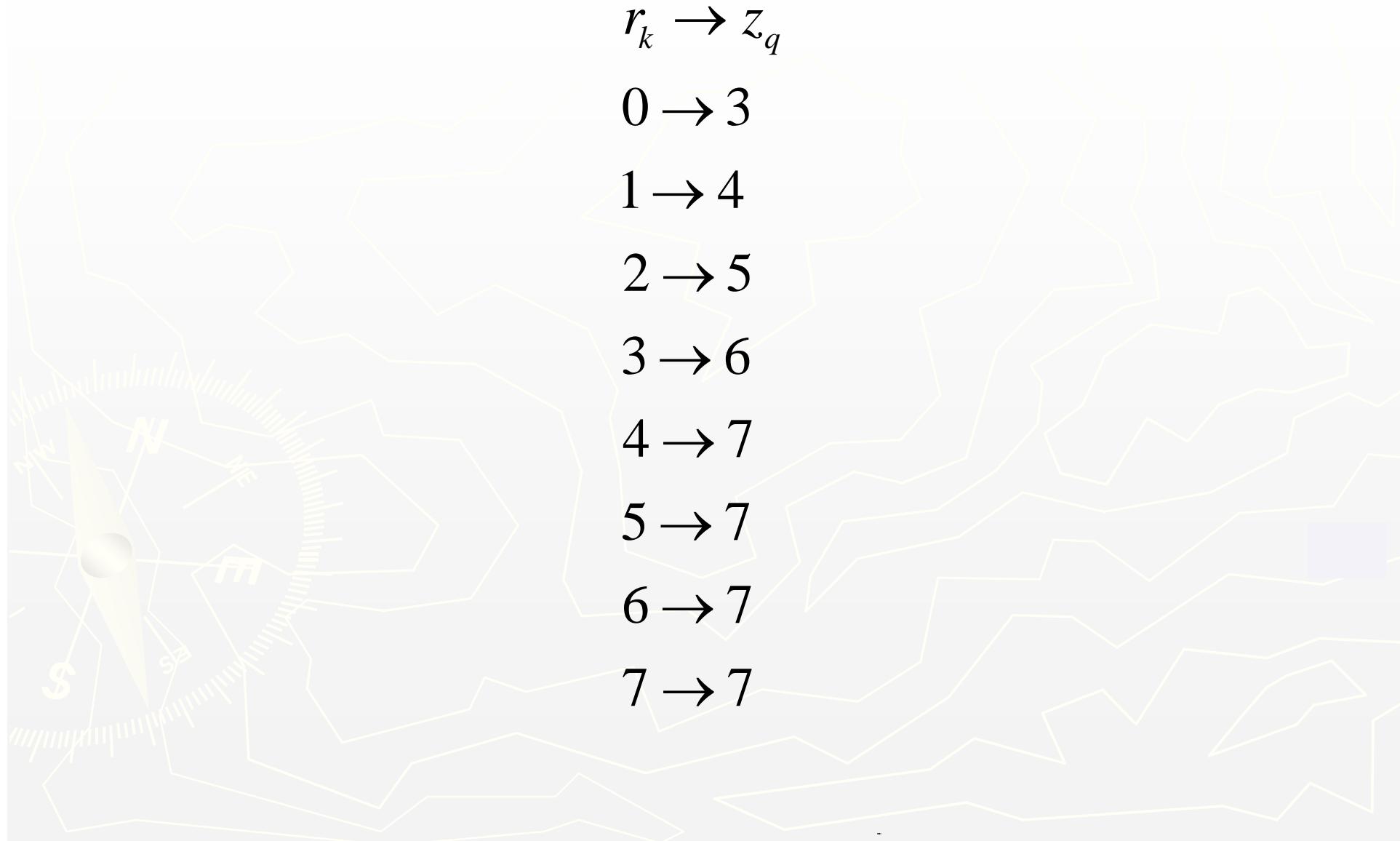
$$3 \rightarrow 6$$

$$4 \rightarrow 7$$

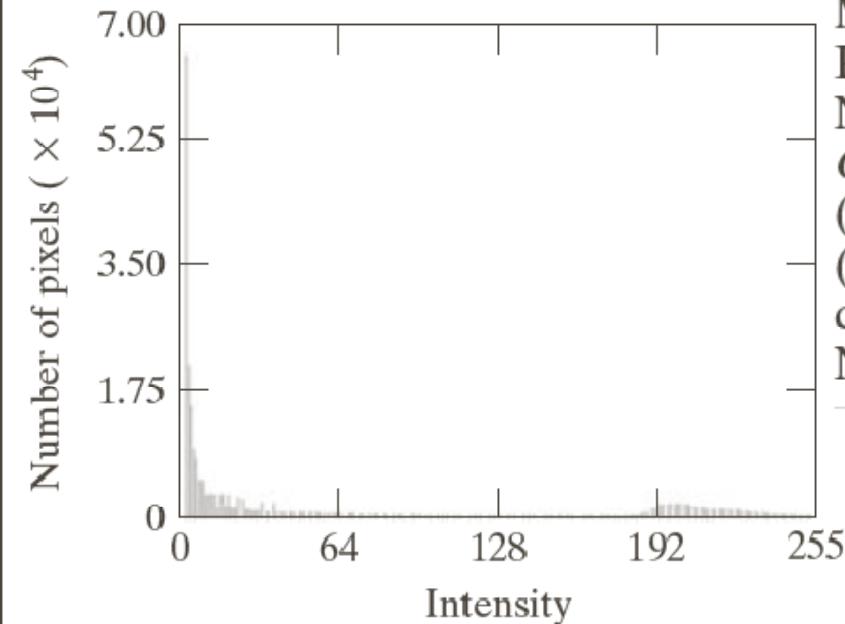
$$5 \rightarrow 7$$

$$6 \rightarrow 7$$

$$7 \rightarrow 7$$



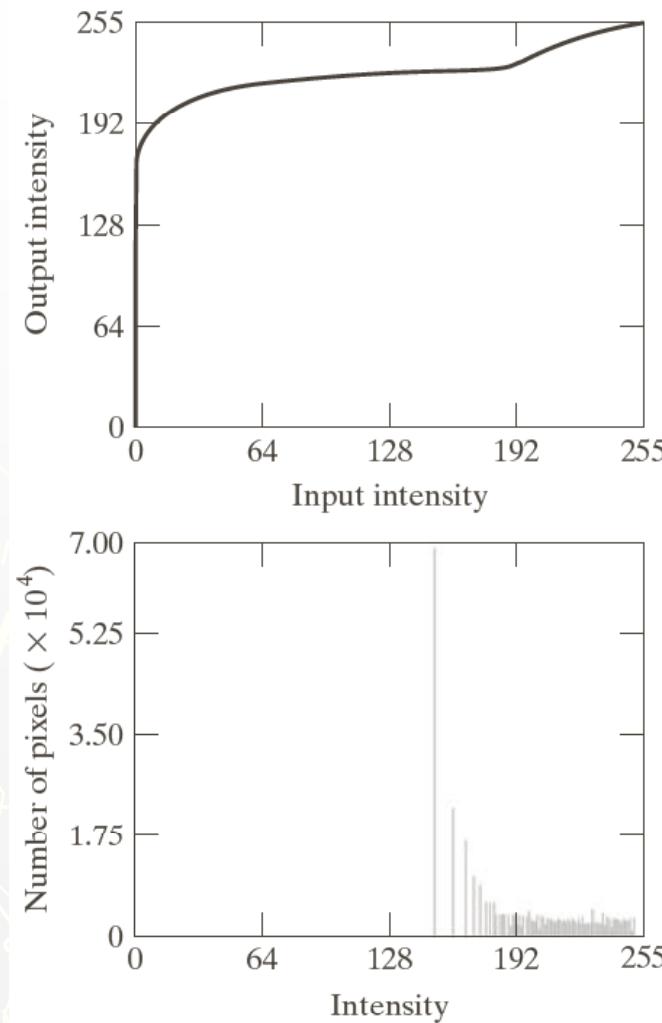
# Example: Histogram Matching



a b

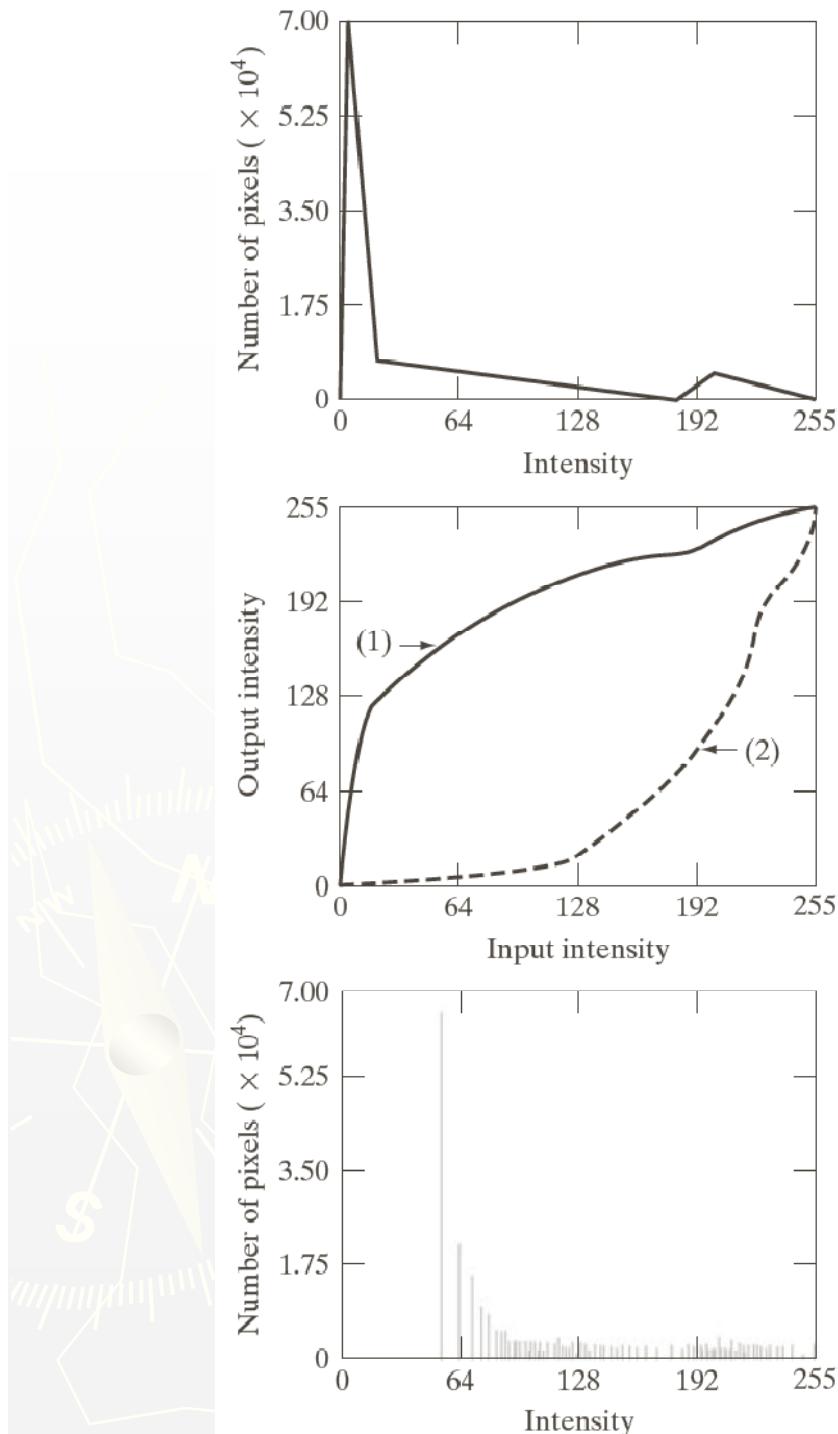
**FIGURE 3.23**  
(a) Image of the  
Mars moon  
Phobos taken by  
NASA's *Mars  
Global Surveyor*.  
(b) Histogram.  
(Original image  
courtesy of  
NASA.)

# Example: Histogram Matching



a  
b  
c

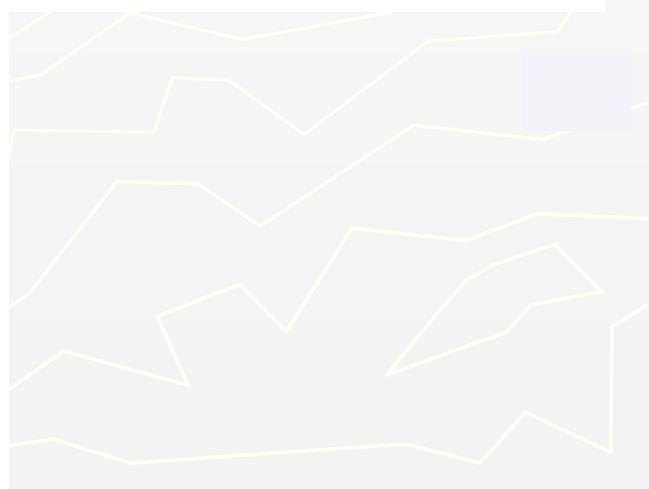
**FIGURE 3.24**  
(a) Transformation function for histogram equalization.  
(b) Histogram-equalized image (note the washed-out appearance).  
(c) Histogram of (b).



a	c
b	d

**FIGURE 3.25**

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



# Local Histogram Processing

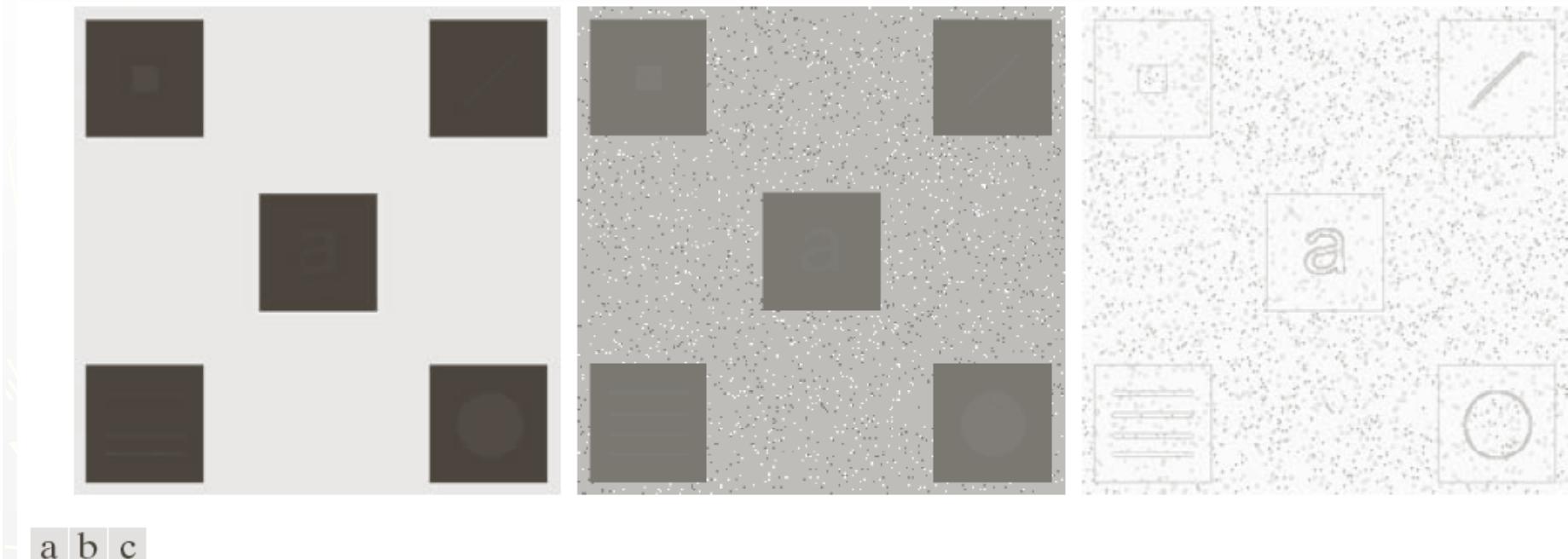
Define a neighborhood and move its center from pixel to pixel

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

Map the intensity of the pixel centered in the neighborhood

Move to the next location and repeat the procedure

# Local Histogram Processing: Example



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



# Using Histogram Statistics for Image Enhancement

Average Intensity

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

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

Variance

$$\sigma^2 = u_2(r) = \sum_{i=0}^{L-1} (r_i - m)^2 p(r_i) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y) - m]^2$$

# Using Histogram Statistics for Image Enhancement

Local average intensity

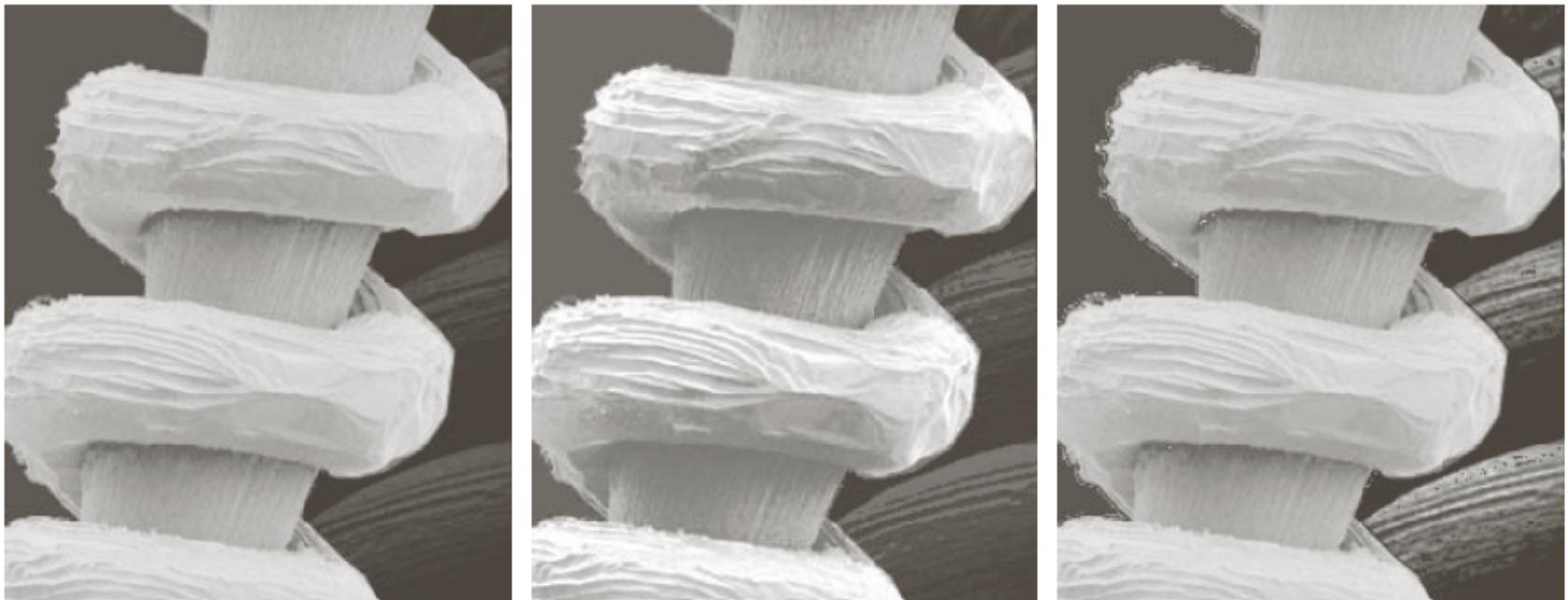
$$m_{s_{xy}} = \sum_{i=0}^{L-1} r_i p_{s_{xy}}(r_i)$$

$s_{xy}$  denotes a neighborhood

Local variance

$$\sigma_{s_{xy}}^2 = \sum_{i=0}^{L-1} (r_i - m_{s_{xy}})^2 p_{s_{xy}}(r_i)$$

# Using Histogram Statistics for Image Enhancement: Example



# **Image Enhancement in *Spatial Domain***

## **Arithmetical and Logic Operations**

- *Arithmetic/logic operations are performed on a pixel by pixel basis between two or more images.*
- *Basic arithmetic operations are: Gray-value point operations are used including the following functions*

<i>Operation</i>	<i>Definition</i>	<i>preferred data type</i>
<b>ADD</b>	$c = a + b$	integer
<b>SUB</b>	$c = a - b$	integer
<b>MUL</b>	$c = a * b$	integer or floating point
<b>DIV</b>	$c = a / b$	floating point
<b>LOG</b>	$c = \log(a)$	floating point
<b>EXP</b>	$c = \exp(a)$	floating point
<b>SQRT</b>	$c = \sqrt{a}$	floating point
<b>TRIG.</b>	$c = \sin/\cos/\tan(a)$	floating point
<b>INVERT</b>	$c = (2^B - 1) - a$	integer

# Image Enhancement in *Spatial Domain*

## Arithmetical and Logic Operations

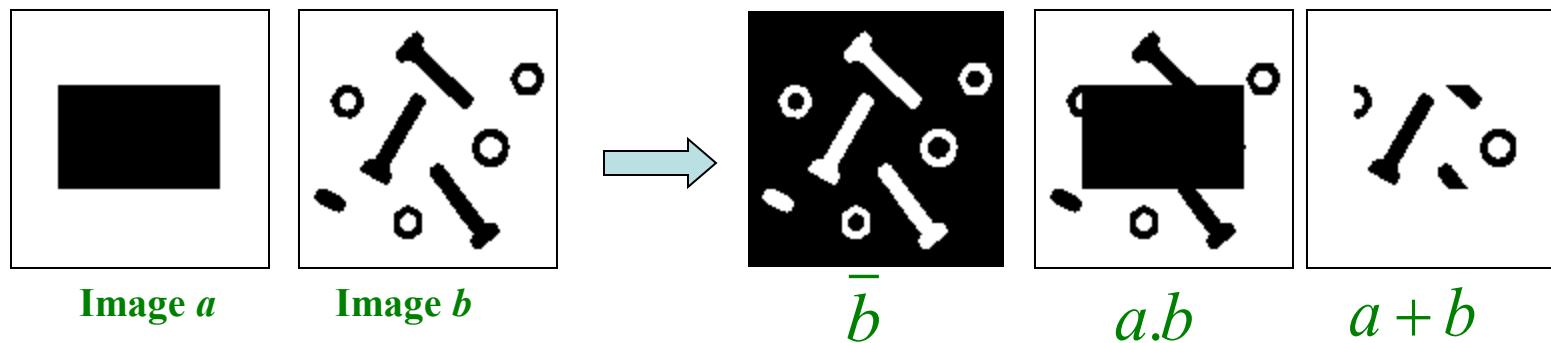
- **Basic logic operations are:** Binary operations are used including the following functions

$$NOT \quad c = \bar{a}$$

$$OR \quad c = a + b$$

$$AND \quad c = a \cdot b$$

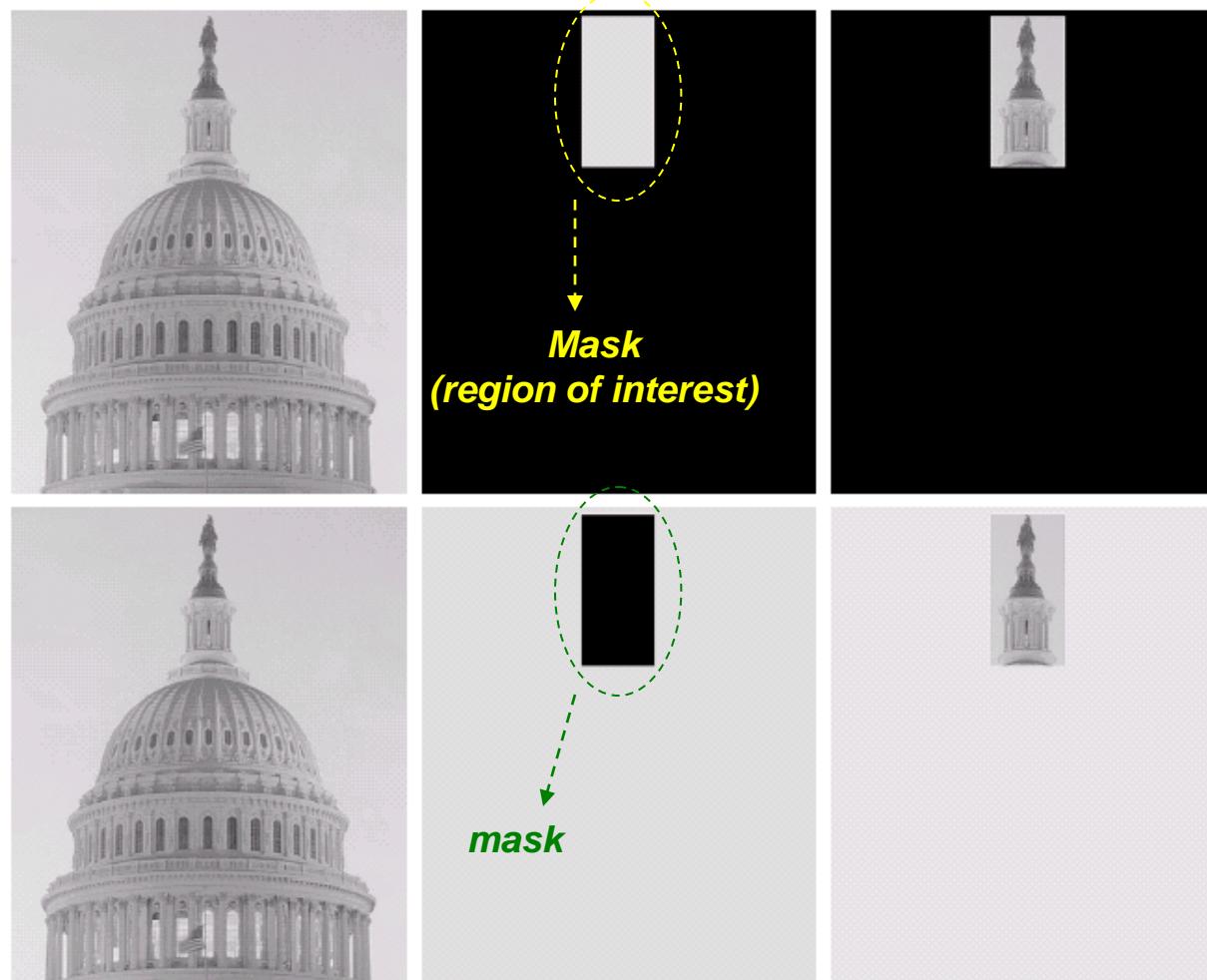
$$XOR \quad c = a \oplus b = a \cdot \bar{b} + \bar{a} \cdot b$$



Note: The images can be **binary (bi-level)** images. Each pixel is **1 (True)** or **0 (False)**.

# Image Enhancement in *Spatial Domain*

## Arithmetical and Logic Operations



a	b	c
d	e	f

**FIGURE 3.27**  
(a) Original image. (b) AND image mask.  
(c) Result of the AND operation on images (a) and (b). (d) Original image. (e) OR image mask.  
(f) Result of operation OR on images (d) and (e).

Note: The images can be *gray-level* images. Each pixel is an 8-bit binary number. Bit by bit operation is used.

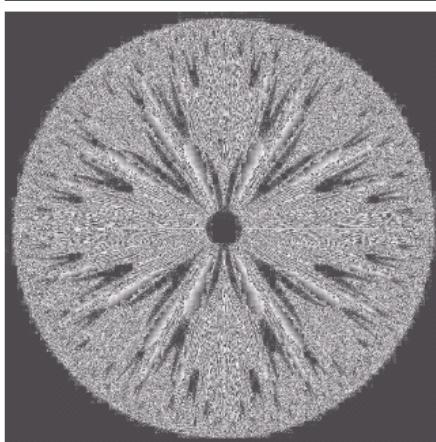
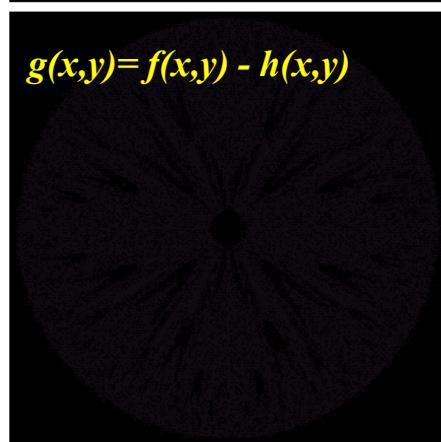
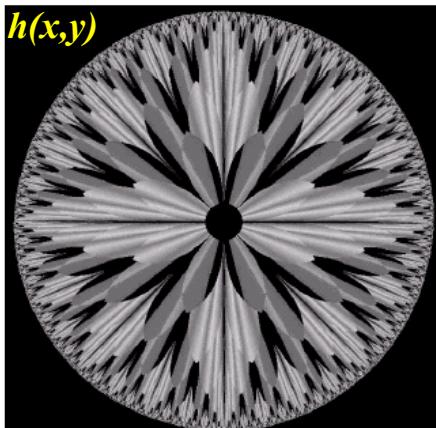
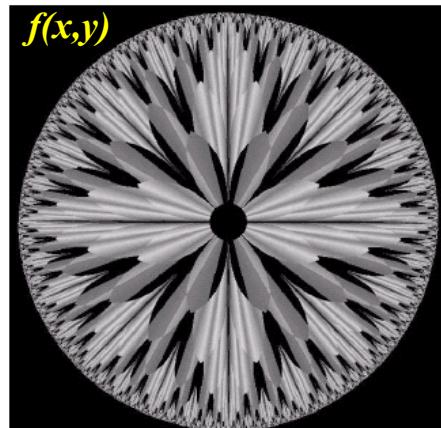
# Image Enhancement in *Spatial Domain*

## Image Subtraction

- *The difference image between two images  $f(x,y)$  and  $h(x,y)$  can be expressed by:* 
$$g(x,y) = f(x,y) - h(x,y)$$

a  
b  
c  
d

**FIGURE 3.28**  
(a) Original fractal image.  
(b) Result of setting the four lower-order bit planes to zero.  
(c) Difference between (a) and (b).  
(d) Histogram-equalized difference image.  
(Original image courtesy of Ms. Melissa D. Binde, Swarthmore College, Swarthmore, PA).

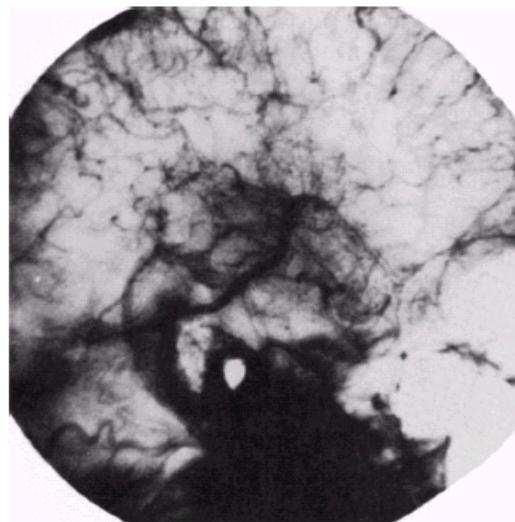


→ *Contrast stretched  $g(x,y)$*

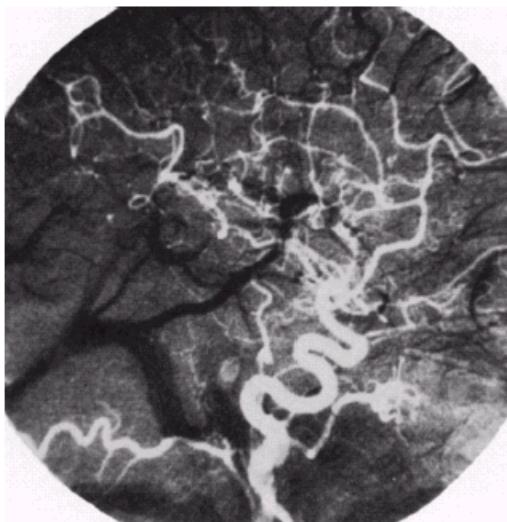
# **Image Enhancement in *Spatial Domain***

## Image Subtraction

- *Image subtraction is used in medical imagining called mask mode radiography.*
- *The initial image is captured and used as the mask image,  $h(x,y)$ . Then after injecting a contrast material into the bloodstream the mask image is subtracted from the resulting image  $f(x,y)$  to give an enhanced output image  $g(x,y)$ .*



$h(x,y)$



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

a b

**FIGURE 3.29**  
Enhancement by  
image subtraction.  
(a) Mask image.  
(b) An image  
(taken after  
injection of a  
contrast medium  
into the  
bloodstream) with  
mask subtracted  
out.

---

# Image Enhancement in *Spatial Domain*

## Image Averaging

- Consider a noisy image,  $g(x, y)$ , formed by the addition of noise  $\eta(x, y)$  to an original image  $f(x, y)$ :

$$g(x, y) = f(x, y) + \eta(x, y)$$

- Consider an uncorrelated noise with zero average value.
- An enhanced image,  $\bar{g}(x, y)$ , can be formed by adding  $K$  different noisy images.

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

- The expected value of  $\bar{g}$ :

$$E\{\bar{g}(x, y)\} = f(x, y)$$

---

# Image Enhancement in *Spatial Domain*

## Image Averaging

- Then variances:

$$\sigma_{\bar{g}(x,y)}^2 = \frac{1}{K} \sigma_{\eta(x,y)}^2$$

- Standard deviations in the average image:

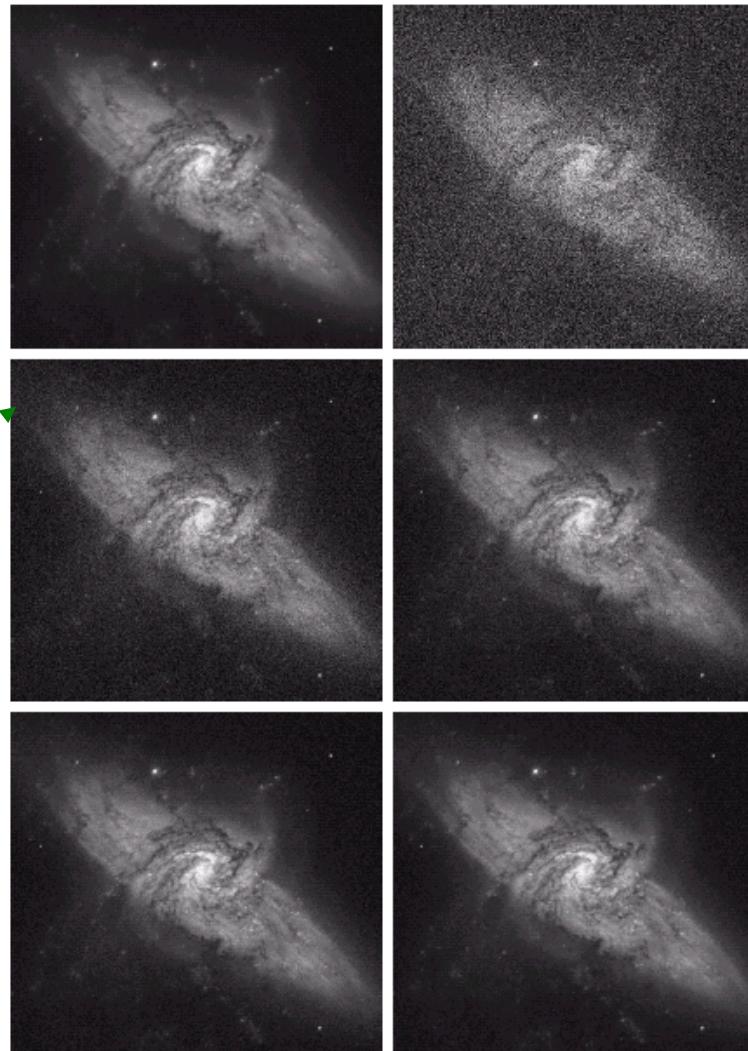
$$\sigma_{\bar{g}(x,y)} = \frac{1}{\sqrt{K}} \sigma_{\eta(x,y)}$$

- As  $K$  increases the variability (noise) of the pixel values at each location  $(x,y)$  decreases.

- In other words, the average image,  $\bar{g}(x,y)$ , approaches the input image  $f(x,y)$  as the number of noisy images used in the averaging operation increases.

# Image Enhancement in *Spatial Domain*

## Image Averaging



a  
b  
c  
d  
e  
f

**FIGURE 3.30** (a) Image of Galaxy Pair NGC 3314. (b) Image corrupted by additive Gaussian noise with zero mean and a standard deviation of 64 gray levels. (c)–(f) Results of averaging  $K = 8, 16, 32, \text{ and } 64$  noisy images. (Original image courtesy of NASA.)

*One of the noisy images*

*Average of  $K=16$  noisy images*

*Average of  $K=128$  noisy images*

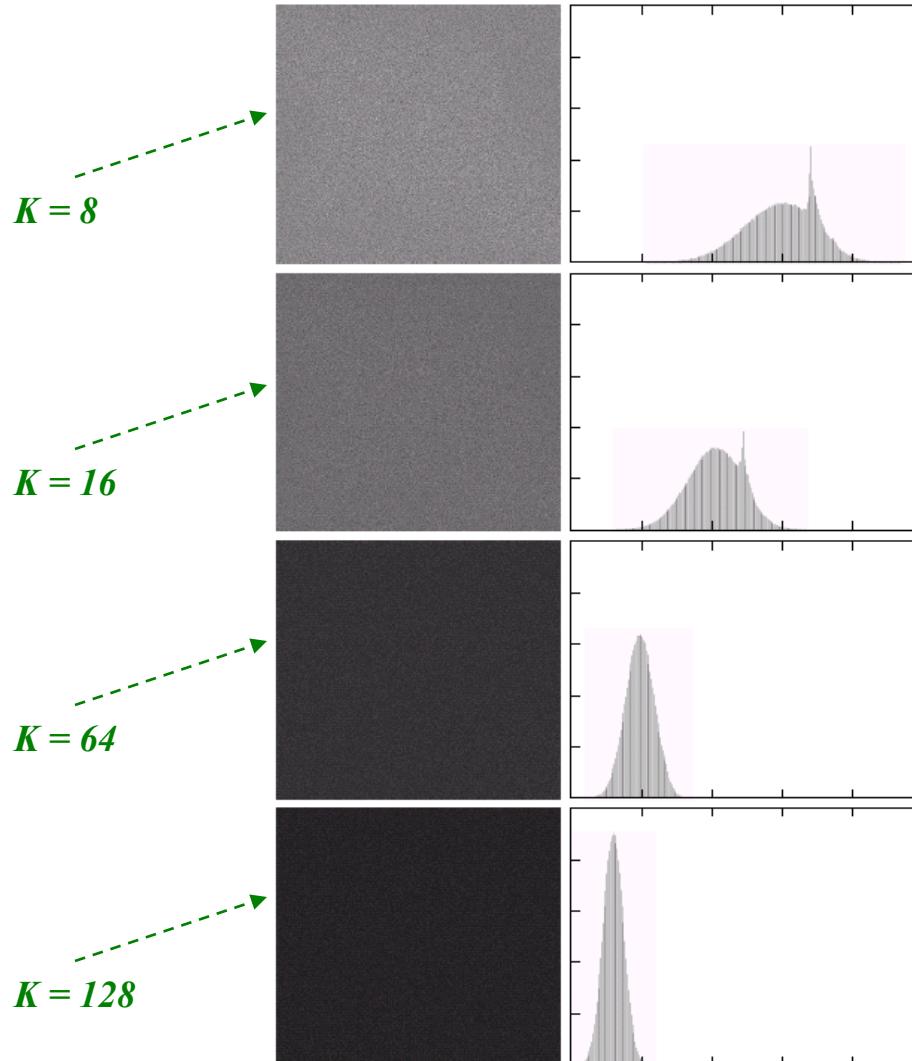
*Original im.*

*$K=8$*

*$K=32$*

# Image Enhancement in *Spatial Domain*

## Image Averaging



a

b

**FIGURE 3.31**  
(a) From top to bottom:  
Difference images  
between  
Fig. 3.30(a) and  
the four images in  
Figs. 3.30(c)  
through (f),  
respectively.  
(b) Corresponding  
histograms.

*difference = original - averaged*

$$d(x, y) = f(x, y) - \bar{g}(x, y)$$

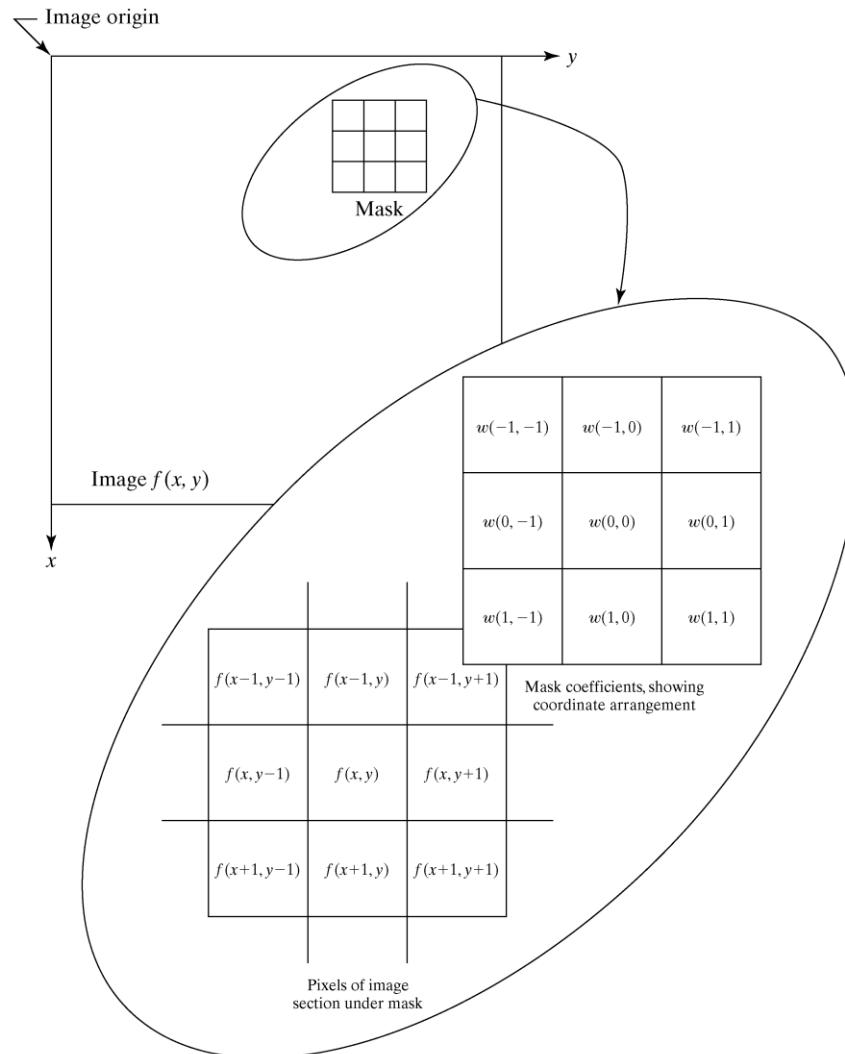
# Image Enhancement in *Spatial Domain*

## Spatial Filtering

- *Spatial filtering refers to some neighborhood operations working with the values of the image pixels in the neighborhood and the corresponding values of a subimage that has the same dimensions as the neighborhood.*
- *This subimage is called, a filter, mask, kernel, template or a window. The values in a filter is referred to as coefficients.*
- *The filtering can be performed in
  - spatial domain.
  - frequency domain (we will study later) and*
- *There are two main types of spatial domain filtering
  - linear spatial filtering (convolution filter/mask/kernel) and
  - nonlinear spatial filtering .*

# Image Enhancement in *Spatial Domain*

## Spatial Filtering



**FIGURE 3.12** The mechanics of linear spatial filtering. The magnified drawing shows a  $3 \times 3$  mask and the corresponding image neighborhood directly under it. The neighborhood is shown displaced out from under the mask for ease of readability.

---

# Image Enhancement in *Spatial Domain*

## Linear Spatial Filtering

- Using a **3 x 3** mask shown in the previous slide the response,  $R$ , of a linear filtering with the filter mask at point  $(x,y)$  in the image is:

$$R = \omega(-1, -1)f(x-1, y-1) + \omega(-1, 0)f(x-1, y) + \dots \\ + \omega(0, 0)f(x, y) + \dots + \omega(1, 0)f(x+1, y) + \omega(1, 1)f(x+1, y+1)$$

- In general, linear filtering of an image of size  **$M \times N$**  with a filter mask of size  **$m \times n$**  is given by:

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

- where  $a = (m - 1)/2$ ,  $b = (n - 1)/2$

# **Image Enhancement in *Spatial Domain***

## **Linear Spatial Filtering**

• *Linear spatial filtering is often called convolution operation and the filter mask is also referred to as convolution mask.*

• *Response, R, of a  $m \times n$  mask at any point  $(x,y)$  in the image can be formulated by:*

$$R = \omega_1 z_1 + \omega_2 z_2 + \dots + \omega_{mn} z_{mn}$$

$$= \sum_{i=1}^{mn} \omega_i z_i$$

• *Where,  $\omega$ 's are the mask coefficients and  $z$ 's are the image pixel values.*

• *Given a  $3 \times 3$  mask below the response at any point  $(x,y)$  is:*

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$

$$R = \sum_{i=1}^9 \omega_i z_i$$

**FIGURE 3.33**  
Another representation of a general  $3 \times 3$  spatial filter mask.

---

# **Image Enhancement in *Spatial Domain***

## **Smoothing Spatial Filters**

- *Smoothing filters are used for noise reduction and blurring operations. Blurring can be used as a preprocessing step for other image processing operations.*
- *There are two main types of Smoothing filters:*
  - *Smoothing Linear Filters*
  - *Smoothing Nonlinear Filters*

## **Smoothing Linear Filters/ Averaging Filters**

- *The response of a smoothing linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask.*
- *These kind of filters are called averaging filters or lowpass filters.*

---

# Image Enhancement in *Spatial Domain*

## Convolution and Correlation

• Convolution involves calculating the weighted sum of a neighborhood of pixels. The weights are taken from a convolution kernel. Each value from the neighborhood of pixels is multiplied with its opposite on the matrix. For example, the top-left of the neighbor is multiplied by the bottom-right of the kernel. All these values are summed up to calculate the result of the convolution.

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

Consider a  $3 \times 3$  neighborhood. Given a convolution kernel (mask)  $\omega$ , you need to rotate the mask with  $180^\circ$  as follows,

# Image Enhancement in *Spatial Domain*

## Convolution operation

K

Convolution kernel,  $\omega$

1	-1	-1
1	2	-1
1	1	1

Rotate  
180°

1	1	1
-1	2	1
-1	-1	1

Input Image  $f$

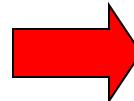
2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

*Convolution:* Step 1

1	1	1		
-1	4	2	2	3
-1	-2	1	3	3
	2	2	1	2
1	3	2	2	



5			

Input Image,  $f$

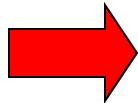
output Image,  $g$

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

***Convolution: Step 2***

1	1	1
-2	4	2
-2	-1	3
2	2	1
1	3	2

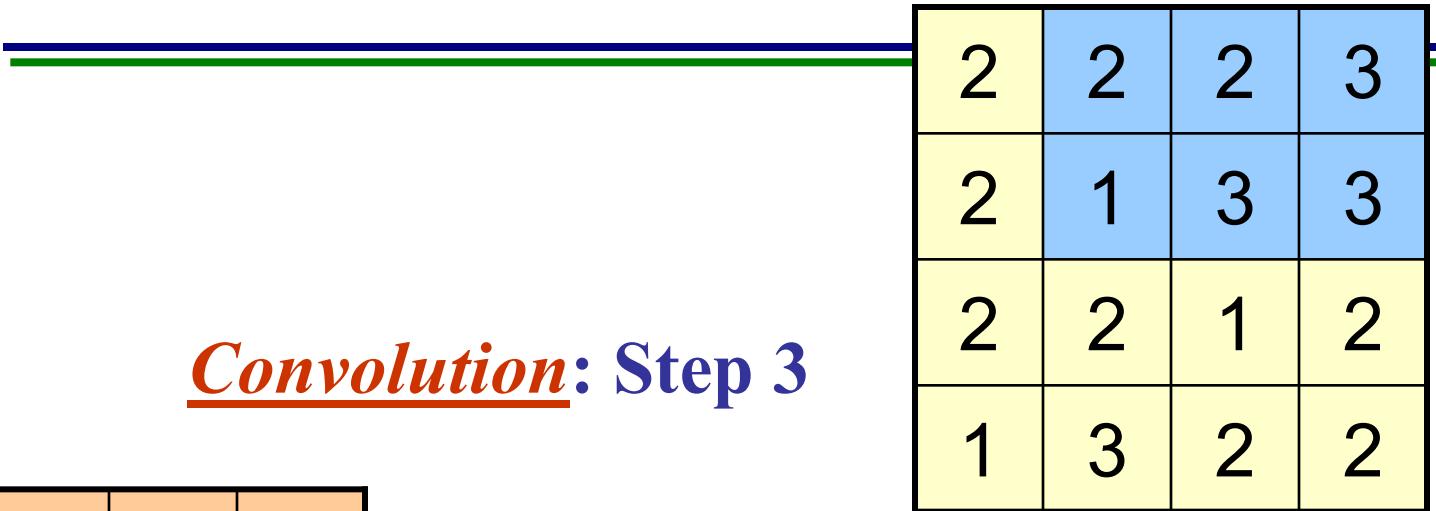


5	4	

Input Image,  $f$

output Image,  $g$

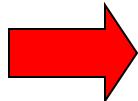
1	1	1
-1	2	1
-1	-1	1



	1	1	1
2	-2	4	3
2	-1	-3	3
2	2	1	2

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



5	4	4	

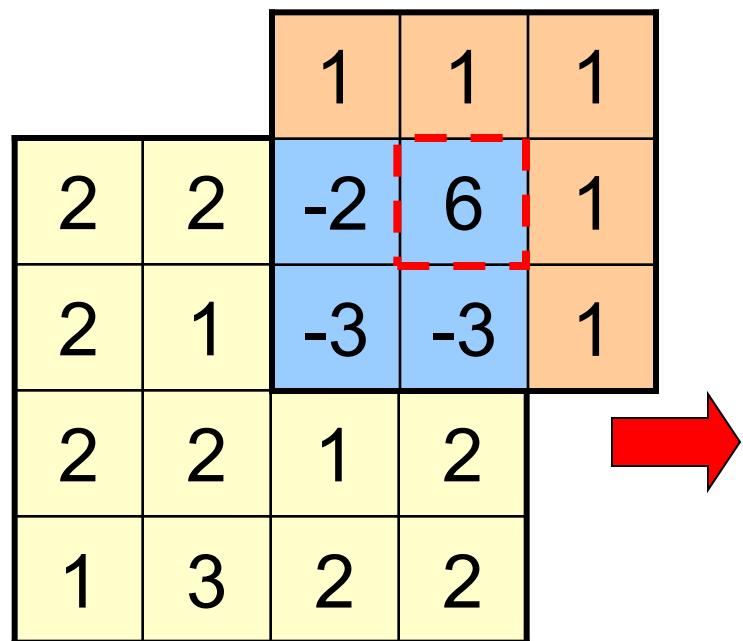
Input Image,  $f$

output Image,  $g$

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

***Convolution:*** Step 4

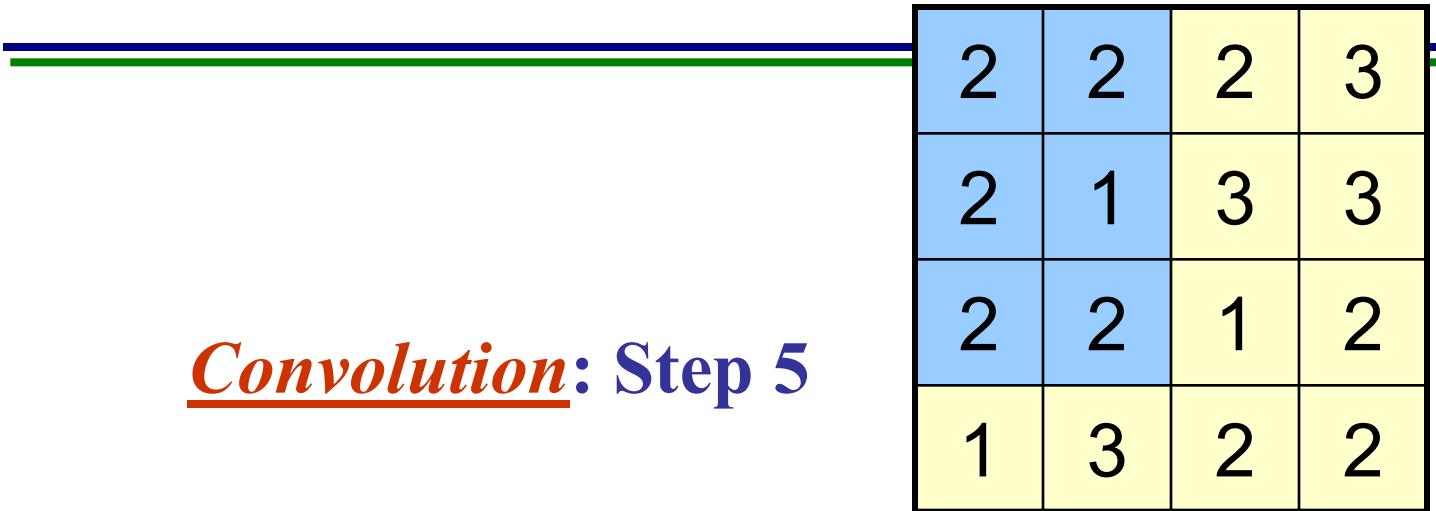


Input Image,  $f$

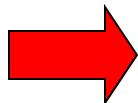
5	4	4	-2

output Image,  $g$

1	1	1
-1	2	1
-1	-1	1



1	2	2	2	3
-1	4	1	3	3
-1	-2	2	1	2
1	3	2	2	



5	4	4	-2
9			

Input Image,  $f$

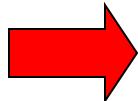
output Image,  $g$

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

***Convolution:*** Step 6

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



5	4	4	-2
9	6	0	0
0	0	0	0
0	0	0	0

Input Image,  $f$

output Image,  $g$

---

## Convolution: Final Result

5	4	4	-2
9	6	14	5
11	7	6	5
9	12	8	5

Final output Image,  $g$

---

# Image Enhancement in *Spatial Domain*

## Convolution and Correlation

- Correlation is nearly identical to convolution with only a minor difference, where instead of multiplying the pixel by the opposite in the kernel, you multiply it by the equivalent (i.e. top-left multiplied by top-left).

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

Consider a  $3 \times 3$  neighborhood. Given a correlation kernel (mask)  $\omega$ , and input image  $f$ ,

# Image Enhancement in *Spatial Domain*

## Correlation operation

K

orrelation kernel,  $\omega$

1	-1	-1
1	2	-1
1	1	1

Don't rotate use it directly

Input Image  $f$

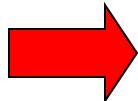
2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

*Correlation: Step 1*

1	-1	-1		
1	4	-2	2	3
1	2	1	3	3
2	2	1	2	
1	3	2	2	



5			

Input Image,  $f$

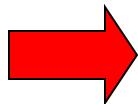
output Image,  $g$

1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

Correlation: Step 2

1	-1	-1	
2	4	-2	3
2	1	3	3
2	2	1	2
1	3	2	2



5	10		

Input Image,  $f$

output Image,  $g$

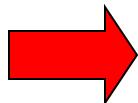
1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

Correlation: Step 3

	1	-1	-1
2	2	4	-3
2	1	3	3
2	2	1	2
1	3	2	2

Input Image,  $f$



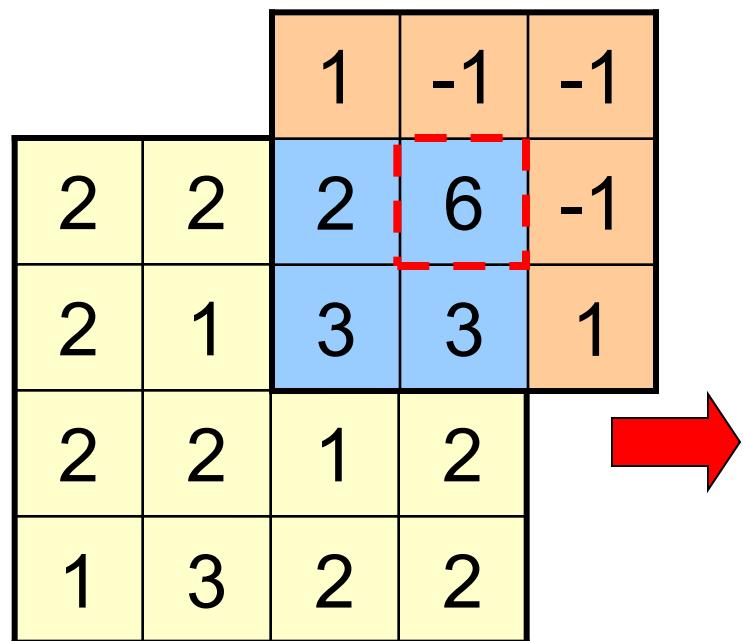
5	10	10	

output Image,  $g$

1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

Correlation: Step 4



Input Image,  $f$

5	10	10	15

output Image,  $g$

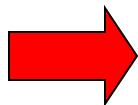
1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

***Correlation: Step 5***

1	-2	-2	2	3
1	4	-1	3	3
1	2	2	1	2
1	3	2	2	

Input Image,  $f$



5	10	10	15
3			

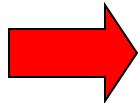
output Image,  $g$

1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

Correlation: Step 6

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



5	10	10	15
3	4		

Input Image,  $f$

output Image,  $g$

---

## Correlation: Final Result

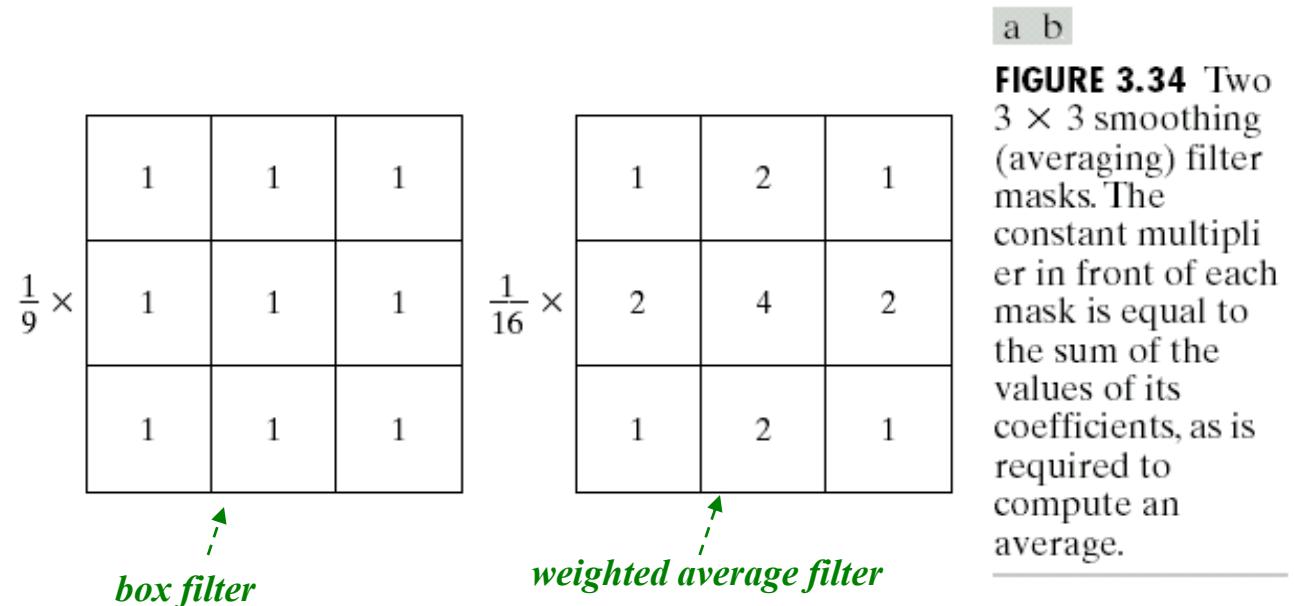
5	10	10	15
3	4	6	11
7	11	4	9
-5	4	4	5

Final output Image,  $g$

# Image Enhancement in *Spatial Domain*

## Smoothing Linear Filters/ Averaging Filters

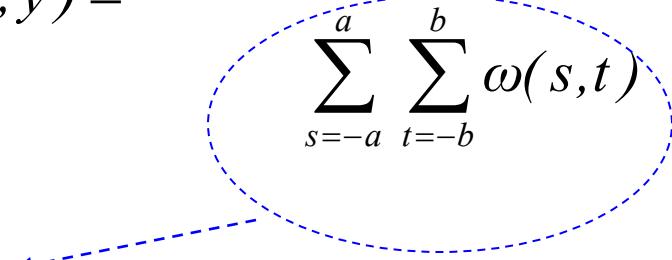
- The idea behind smoothing filters is to replace the value of every pixel in an image by the average of the gray levels defined by the filter mask.
- Random noise consists of sharp transitions in gray levels. So, the most obvious application is **noise reduction**.
- The undesirable effects of the averaging filters is the **blurring of edges**.



# Image Enhancement in *Spatial Domain*

## Smoothing Linear Filters/ Averaging Filters

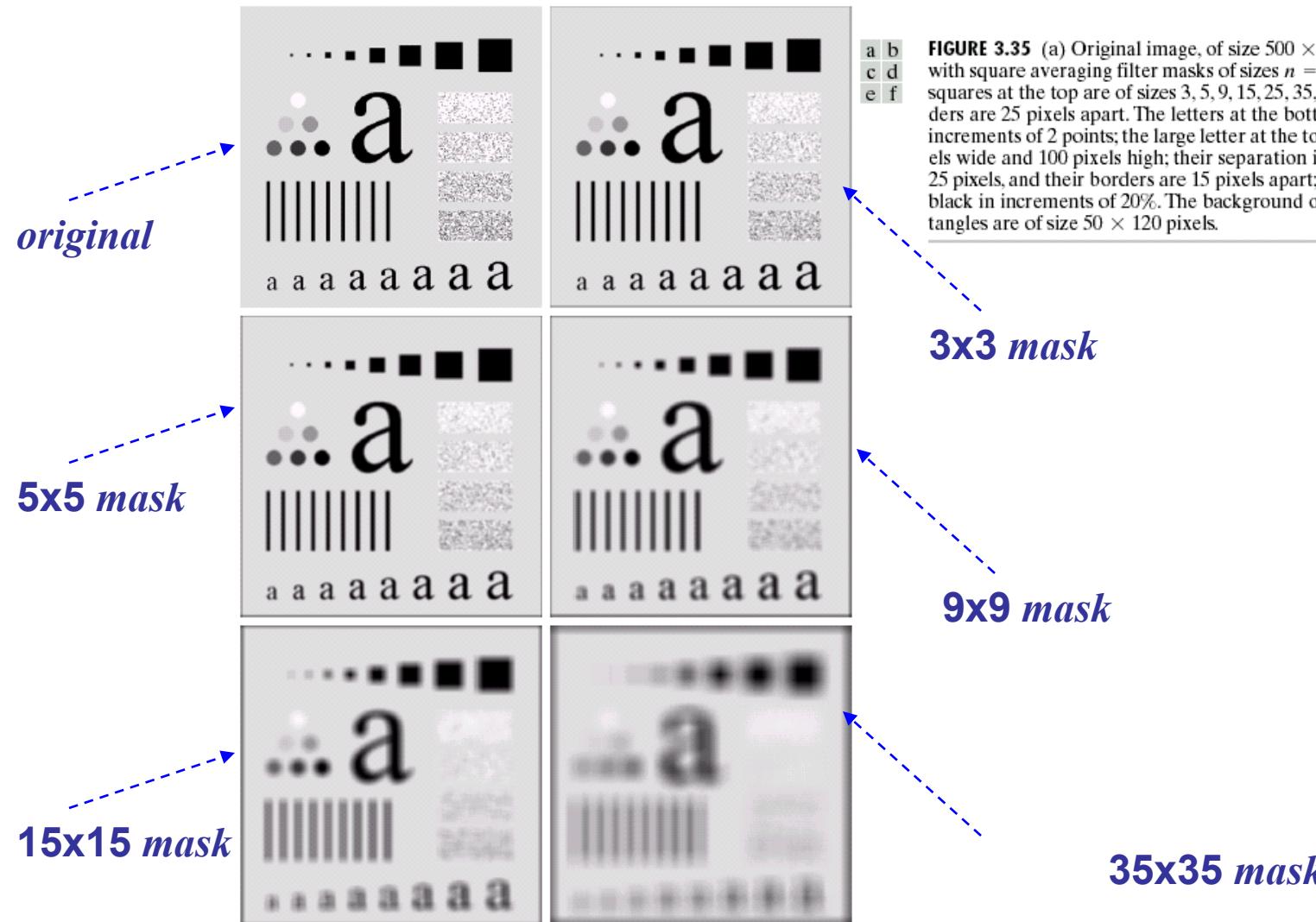
- The general implementation of an  $M \times N$  image with a weighted averaging filter of size  $m \times n$  is given by:

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


Sum of the mask coefficients, which is constant.

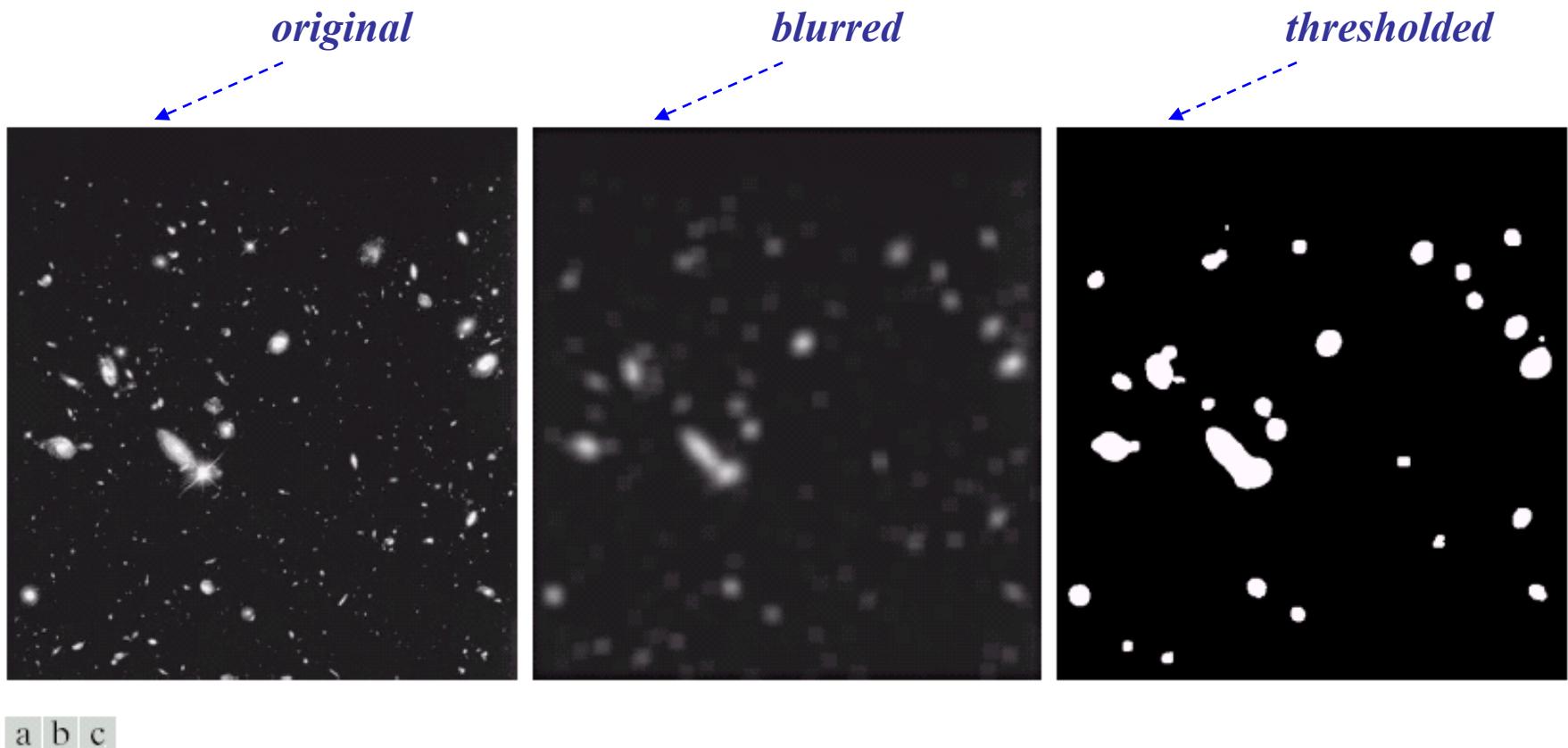
# Image Enhancement in *Spatial Domain*

## Smoothing Linear Filters/ Averaging Filters



# Image Enhancement in *Spatial Domain*

## Smoothing Linear Filters/ Averaging Filters



**FIGURE 3.36** (a) Image from the Hubble Space Telescope. (b) Image processed by a  $15 \times 15$  averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

---

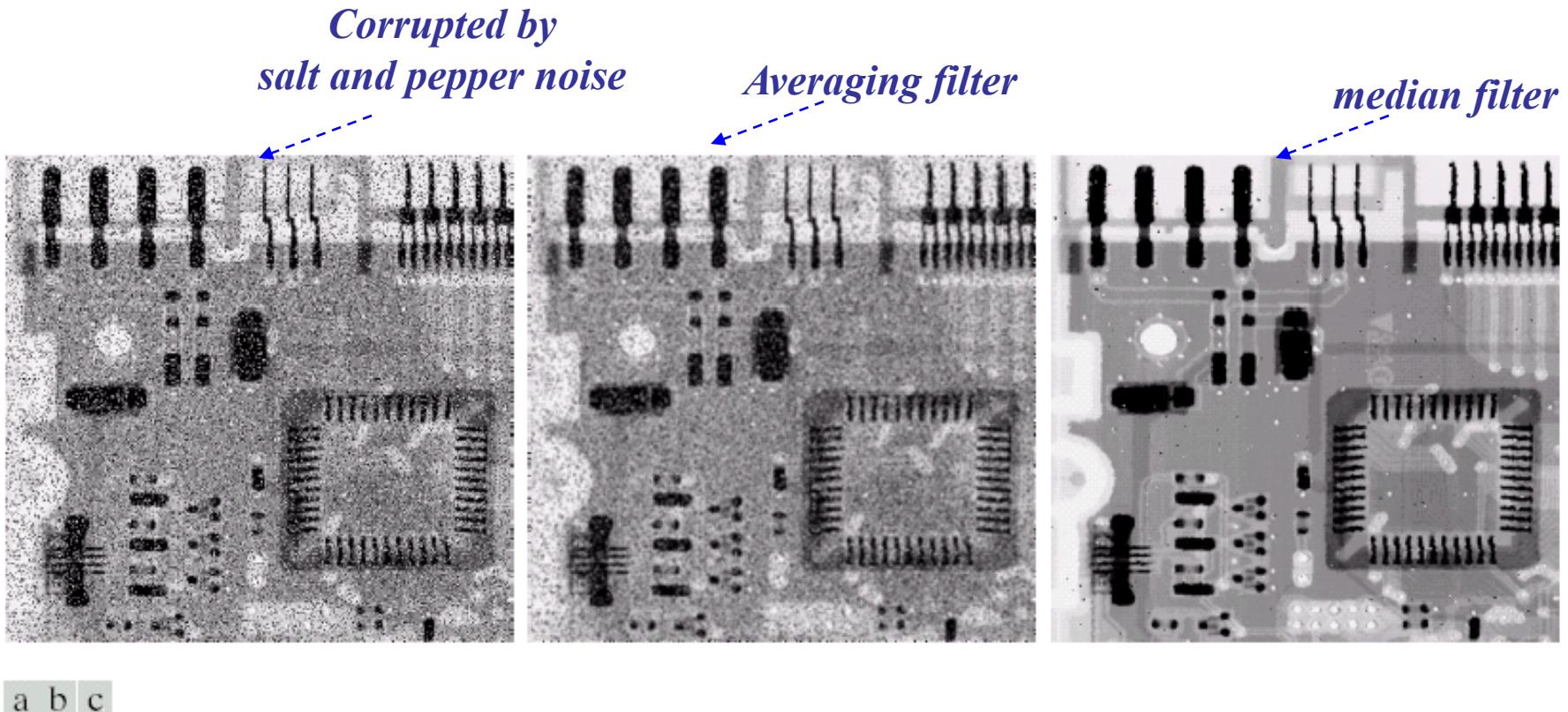
# **Image Enhancement in *Spatial Domain***

## **Order-Statistics Filters**

- *The order-statistics filters are nonlinear spatial filters whose response is based on the ordering/ranking of the pixels contained image area encompassed by the filter.*
- *The center pixel is replaced with the value determined by the ranking result.*
- *The best known ordered –statistics filter is the median filter.*
- *The median filter is excellent for random noise reduction with considerably less blurring than the linear smoothing filters.*
- *Median filters is very effective for impulsive noise which is also called salt-and-pepper noise (noise introducing white and black dots on the image)*
- *Given a 3x3 neighborhood having (10, 20 ,20 ,20 ,100 ,20 ,20 ,25 ,15) gray level values. The sorted values of the neighborhood will be: (10, 15 ,20 ,20 ,20 ,20 ,20 ,25 ,100) and the center pixel will be forced to the median value which is 20.*

# Image Enhancement in *Spatial Domain*

## Order-Statistics Filters



**FIGURE 3.37** (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a  $3 \times 3$  averaging mask. (c) Noise reduction with a  $3 \times 3$  median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

# Image Enhancement in *Spatial Domain*

## Sharpening Spatial Filters

- *Sharpening is the operation to highlight fine details or enhance the details that has been blurred.*
- *Blurring is based on the averaging in a neighborhood which is analogous to integration. Therefore the sharpening could be accomplish by differentiation.*
- *The derivative of a digital function is defined in terms of differences, where a first order derivative of a one dimensional function  $f(x)$  is:*

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

- *Second order derivative can be defined by:*

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

# Image Enhancement in *Spatial Domain*

## Sharpening Spatial Filters

- *The effect of the first and second-order derivatives on an image are:*

- **First-order derivative**

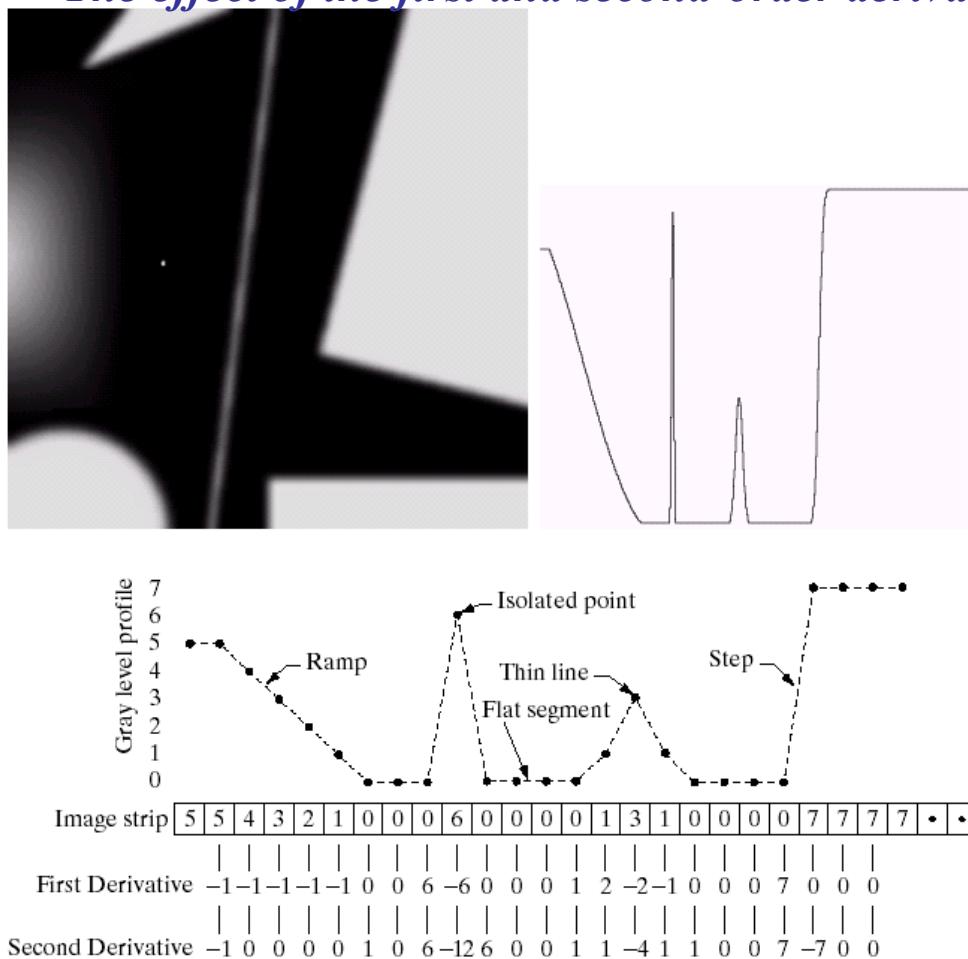
- **Zero in flat segments.**
  - **Nonzero at ramps.**

- **Second-order derivative**

- **Zero in Flat segments.**
  - **Nonzero at the beginning and the end of ramps.**
  - **Zero in along ramps.**

a  
b  
c

**FIGURE 3.38**  
(a) A simple image. (b) 1-D horizontal gray-level profile along the center of the image and including the isolated noise point. (c) Simplified profile (the points are joined by dashed lines to simplify interpretation).



# **Image Enhancement in *Spatial Domain***

## **Second-order Derivatives for Enhancement - The Laplacian**

- *Second-order derivative is used to construct a Laplacian filter mask. Laplacian is an isotropic filter where the response of the filter is independent of the direction of the discontinuities in the image.*
- *Isotropic filters are rotation invariant, which means that if you rotate and filter the image or if you filter and then rotate the image you get the same result.*
- *Given an image  $f(x,y)$  the Laplacian operator is defined by:*

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

- *The operator is a linear operator and can be expressed in discrete form in  $x$ -direction by:*

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

---

# Image Enhancement in *Spatial Domain*

## Second-order Derivatives for Enhancement - The Laplacian

- *The operator can be expressed in discrete form in y-direction by:*

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

- *Then, 2-D Laplacian is:*

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

# Image Enhancement in *Spatial Domain*

## Second-order Derivatives for Enhancement - The Laplacian

- *The filter masks used to implement the digital Laplacian:*

$\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix}$	$\begin{matrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{matrix}$	$\begin{matrix} a & b \\ c & d \end{matrix}$
$\begin{matrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{matrix}$	$\begin{matrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{matrix}$	

considers x and y coordinates  
Isotropic results for  $90^\circ$

considers x, y and two diagonal coordinates.  
Isotropic for  $45^\circ$

**FIGURE 3.39**  
(a) Filter mask used to implement the digital Laplacian, as defined in Eq. (3.7-4).  
(b) Mask used to implement an extension of this equation that includes the diagonal neighbors. (c) and (d) Two other implementations of the Laplacian.

# **Image Enhancement in *Spatial Domain***

## **Second-order Derivatives for Enhancement - The Laplacian**

- *The Laplacian operator highlights gray level discontinuities and de-emphasizes the slowly varying gray-levels.*
- *The result of Laplacian operator will give edge lines and other discontinuities on a dark and featureless background.*
- *The background features can be recovered and sharpening effect can be preserved by adding the Laplacian image to the original image.*
- *Depending on the choice of the Laplacian coefficients the following criteria is used for enhancement:*

$$g(x, y) = \begin{cases} f(x, y) - \nabla^2 f(x, y) \\ f(x, y) + \nabla^2 f(x, y) \end{cases}$$

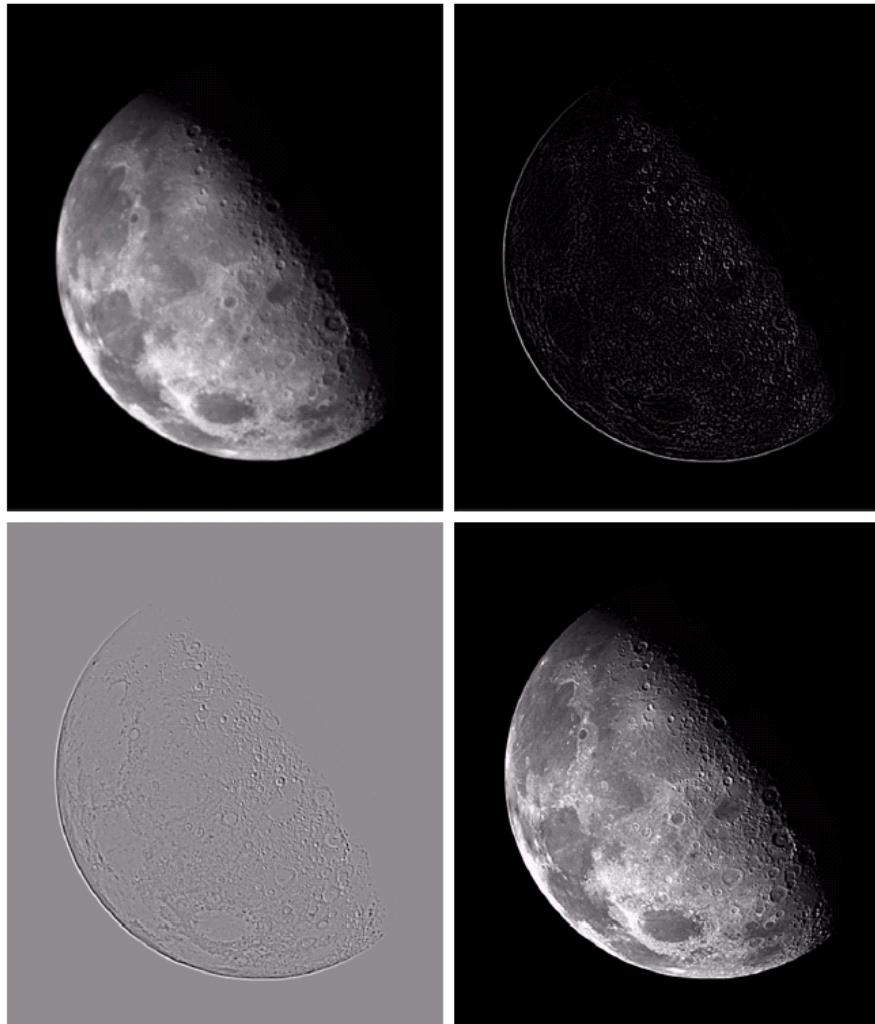
- *If the center coefficient of the Laplacian mask is negative*
- *If the center coefficient of the Laplacian mask is positive*

# Image Enhancement in *Spatial Domain*

## Second-order Derivatives for Enhancement - The Laplacian

a  
b  
c  
d

**FIGURE 3.40**  
(a) Image of the North Pole of the moon.  
(b) Laplacian-filtered image.  
(c) Laplacian image scaled for display purposes.  
(d) Image enhanced by using Eq. (3.7-5).  
(Original image courtesy of NASA.)



Laplacian image

$$\nabla^2 f(x, y)$$

Original image  
 $f(x, y)$

Enhanced image

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

# Image Enhancement in *Spatial Domain*

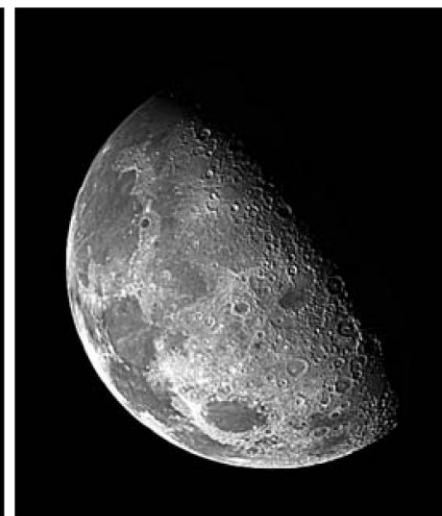
## Second-order Derivatives for Enhancement - The Laplacian



Original image



Enhanced image  
using the mask with  
Center coefficient -4



Enhanced image  
using the mask with  
Center coefficient -8

# **Image Enhancement in *Spatial Domain***

## The First Derivatives for Enhancement - The Gradient

- *The first derivatives in image processing are implemented by using the magnitude of the gradient.*
- *The gradient of  $f$  at coordinates  $(x,y)$  is defined by the two-dimensional column vector:*

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- *The magnitude of this vector, is referred to as the gradient, which is :*

$$\nabla f = \text{mag}(\nabla \mathbf{f}) = \left[ G_x^2 + G_y^2 \right]^{1/2} = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2}$$

# **Image Enhancement in *Spatial Domain***

## The First Derivatives for Enhancement - The Gradient

- *The magnitude of the gradient can be approximated by using the absolute values instead of squares and square roots, which is cheaper to compute and still preserves changes in the gray levels.*

$$\nabla f \approx |G_x| + |G_y|$$

- *If we consider a 3x3 filter mask then an approximation around the center pixel will be as follows:*

$z_1$	$z_2$	$z_3$
$z_4$	$z_5$	$z_6$
$z_7$	$z_8$	$z_9$

$$\begin{aligned}\nabla f \approx |G_x| + |G_y| &= |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| \\ &\quad + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|\end{aligned}$$

# **Image Enhancement in *Spatial Domain***

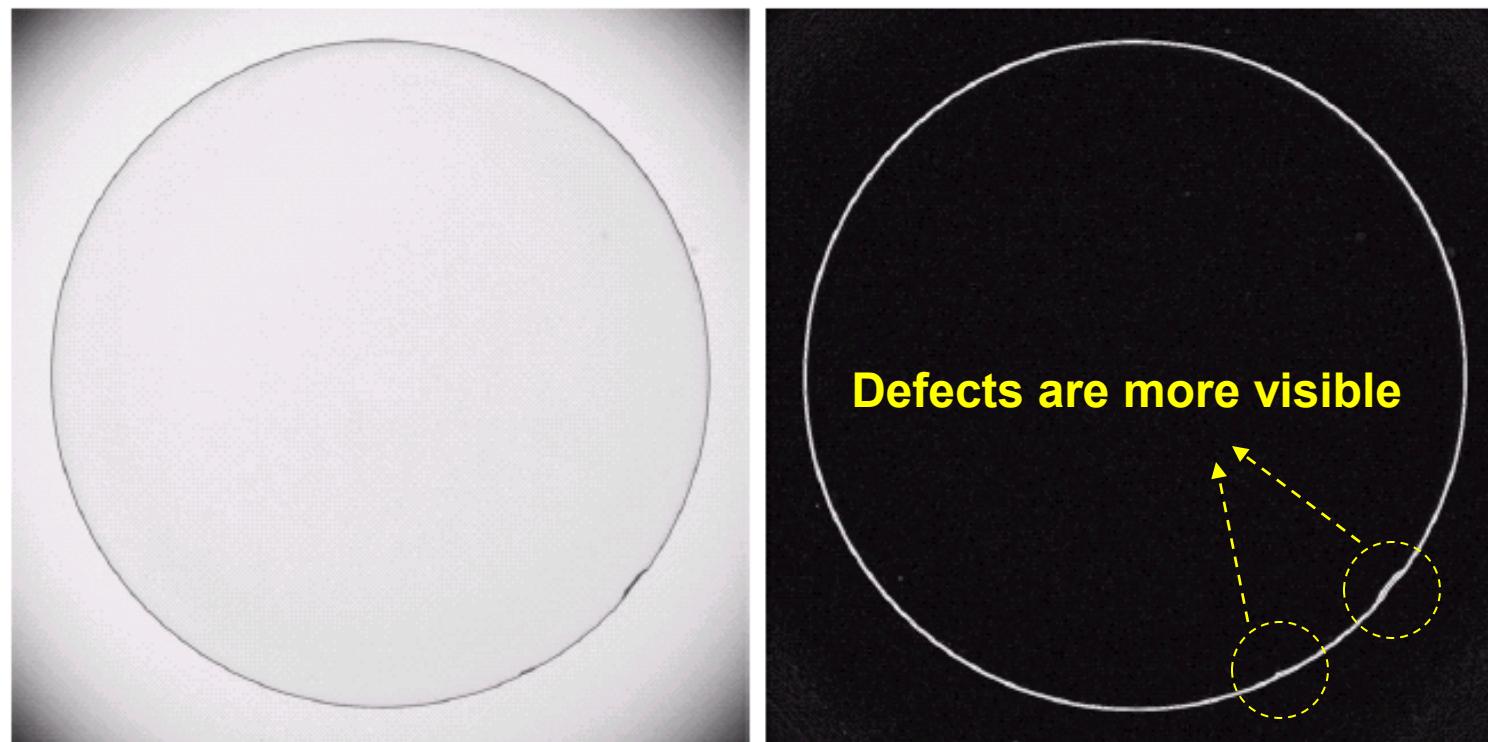
## The First Derivatives for Enhancement - The Gradient

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

- The above masks are called the **Sobel operators** and can be used to implement gradient operation.
- The idea behind using weight value of 2 is to achieve some smoothing by giving more importance to the center point.
- The mask on the left approximates the derivative in **x-direction** (row 3- row 1).
- The mask on the right approximates the derivative in **y-direction** (col 3- col 1).

# Image Enhancement in *Spatial Domain*

## The First Derivatives for Enhancement - The Gradient



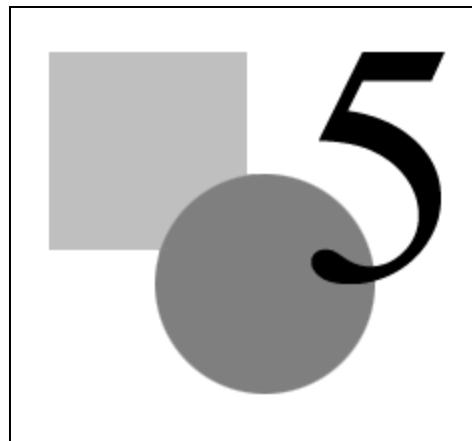
a b

**FIGURE 3.45**  
Optical image of contact lens (note defects on the boundary at 4 and 5 o'clock).  
(b) Sobel gradient.  
(Original image courtesy of Mr. Pete Sites, Perceptics Corporation.)

# Image Enhancement in *Spatial Domain*

## The First Derivatives for Enhancement - The Gradient

• Edge Detection: Consider the following image and the respective *sobel* operators



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

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

Horizontal Operator

Vertical Operator

• If the image is convolved(or correlated) by using the sobel operators given above. Then,

# Image Enhancement in *Spatial Domain*

## The First Derivatives for Enhancement - The Gradient

- Edge Detection:



$g_h$

*Horizontal Sobel Operator  
highlights the horizontal  
edges*

$g_v$

*Vertical Sobel Operator  
highlights the vertical  
edges*

# **Image Enhancement in *Spatial Domain***

## The First Derivatives for Enhancement - The Gradient

### **•Edge Detection:**

- The Gradient for each pixel can be defined to extract edges.*



$$g(x, y) = \sqrt{g(x, y)_h^2 + g(x, y)_v^2}$$

- Edges are detected by combining horizontal and vertical images obtained using respective Sobel operators.*

# Unsharp Masking and Highboost Filtering

## ► Unsharp masking

Sharpen images consists of subtracting an unsharp (smoothed) version of an image from the original image  
e.g., printing and publishing industry

## ► Steps

1. Blur the original image
2. Subtract the blurred image from the original
3. Add the mask to the original

# Unsharp Masking and Highboost Filtering

Let  $\bar{f}(x, y)$  denote the blurred image, unsharp masking is

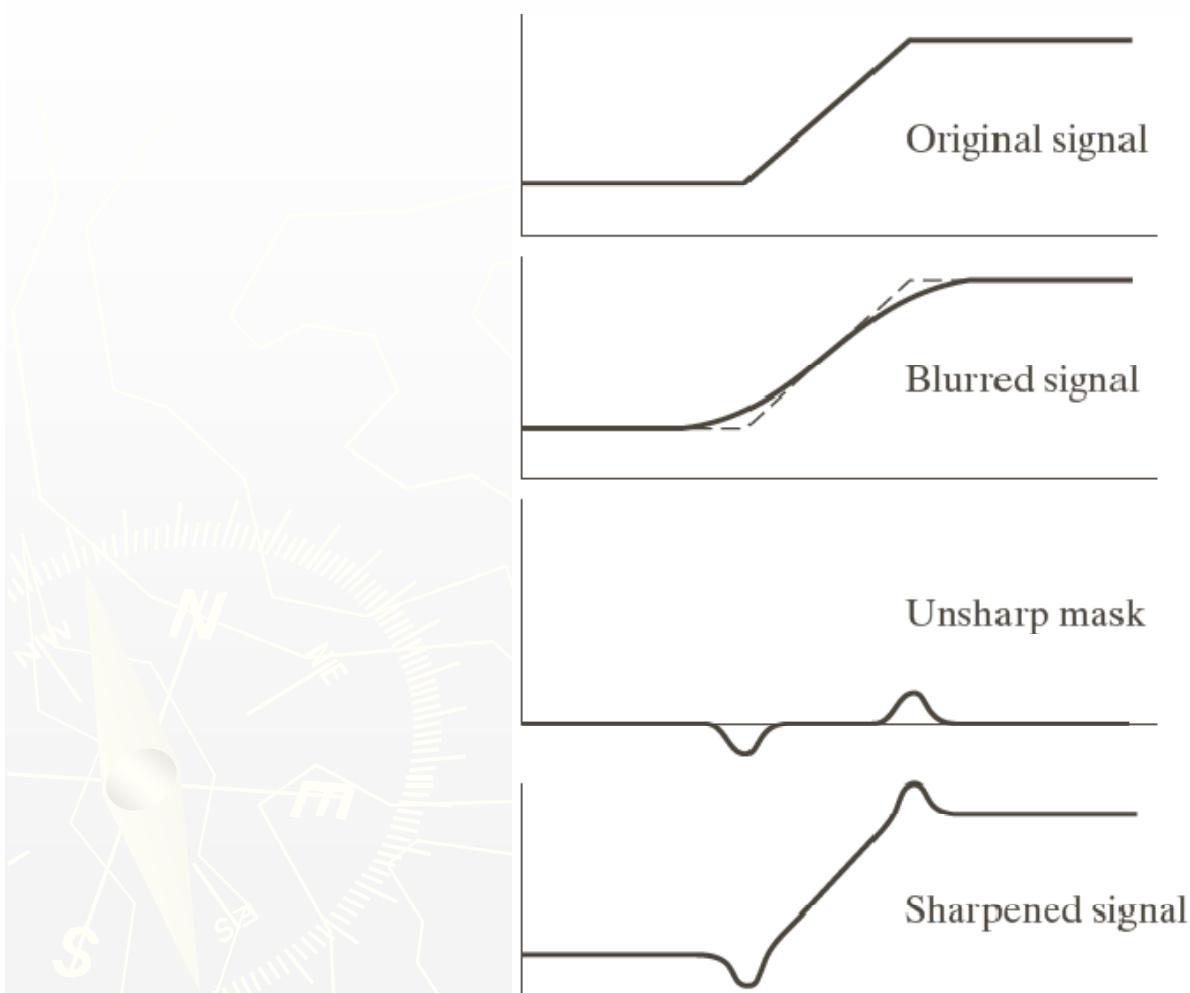
$$g_{mask}(x, y) = f(x, y) - \bar{f}(x, y)$$

Then add a weighted portion of the mask back to the original

$$g(x, y) = f(x, y) + k * g_{mask}(x, y) \quad k \geq 0$$

when  $k > 1$ , the process is referred to as highboost filtering.

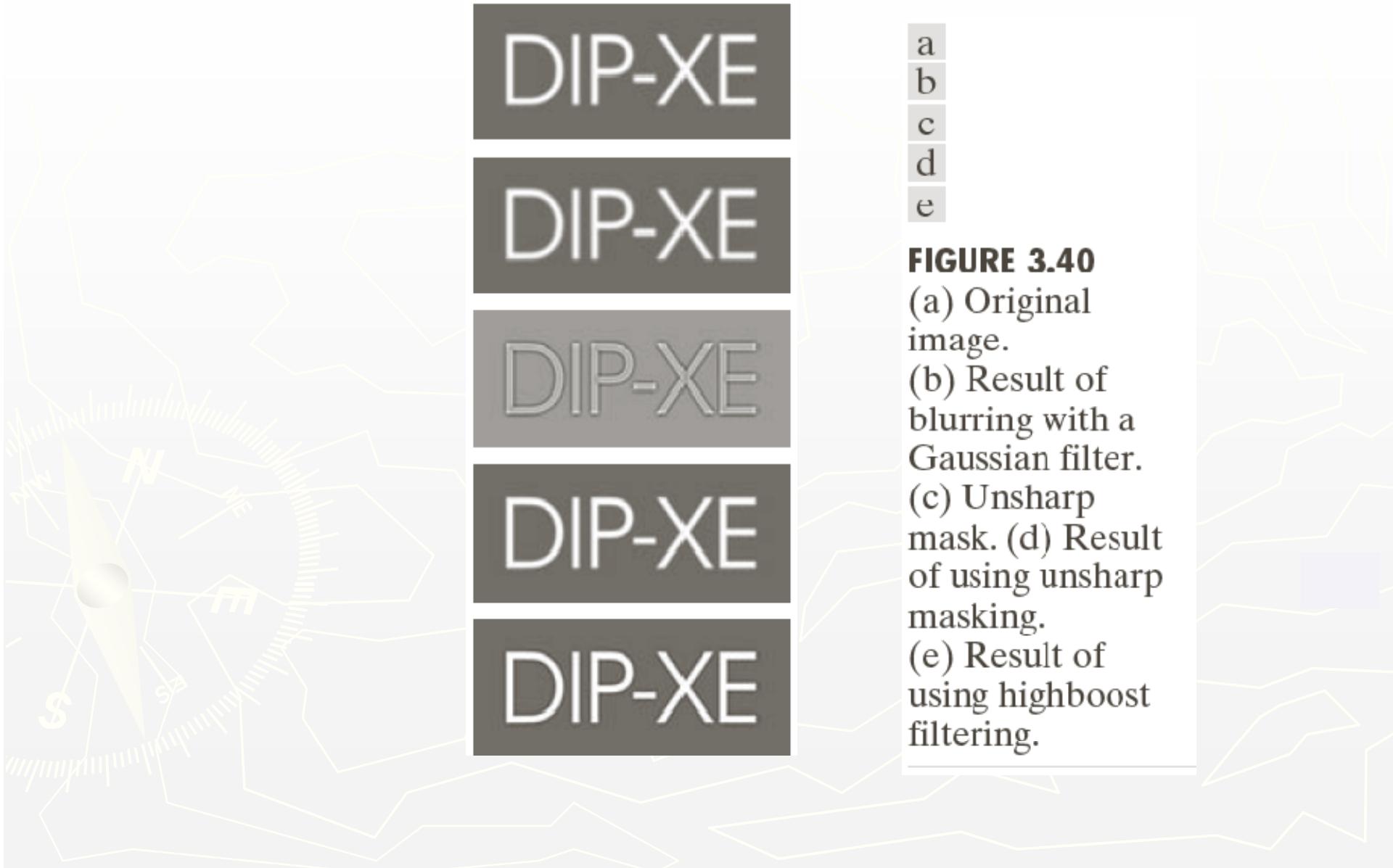
# Unsharp Masking: Demo



a  
b  
c  
d

**FIGURE 3.39** 1-D illustration of the mechanics of unsharp masking.  
(a) Original signal. (b) Blurred signal with original shown dashed for reference. (c) Unsharp mask. (d) Sharpened signal, obtained by adding (c) to (a).

# Unsharp Masking and Highboost Filtering: Example

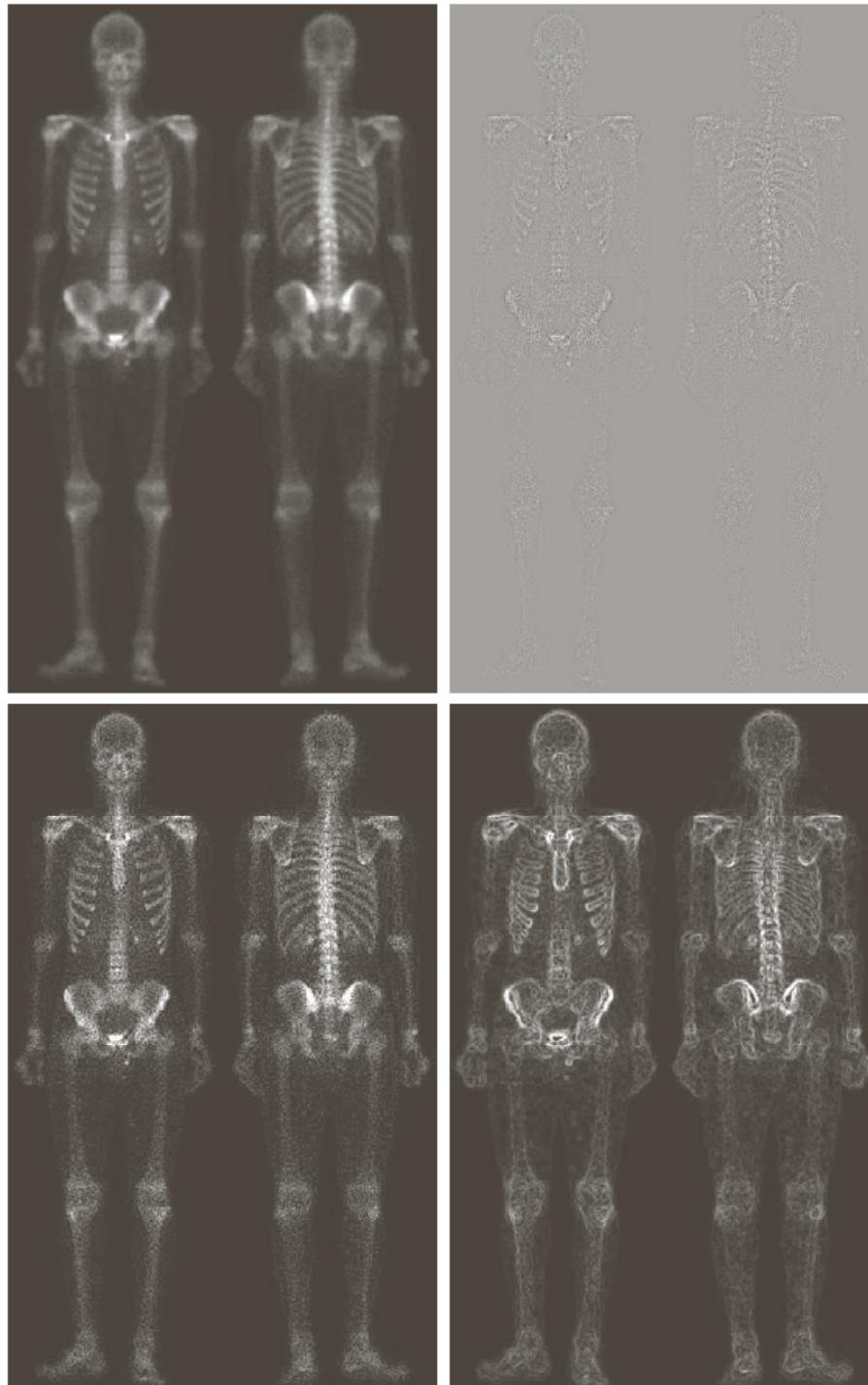


Example:

Combining  
Spatial  
Enhancement  
Methods

Goal:

Enhance the  
image by  
sharpening it  
and by bringing  
out more of the  
skeletal detail



a  
b  
c  
d

**FIGURE 3.43**

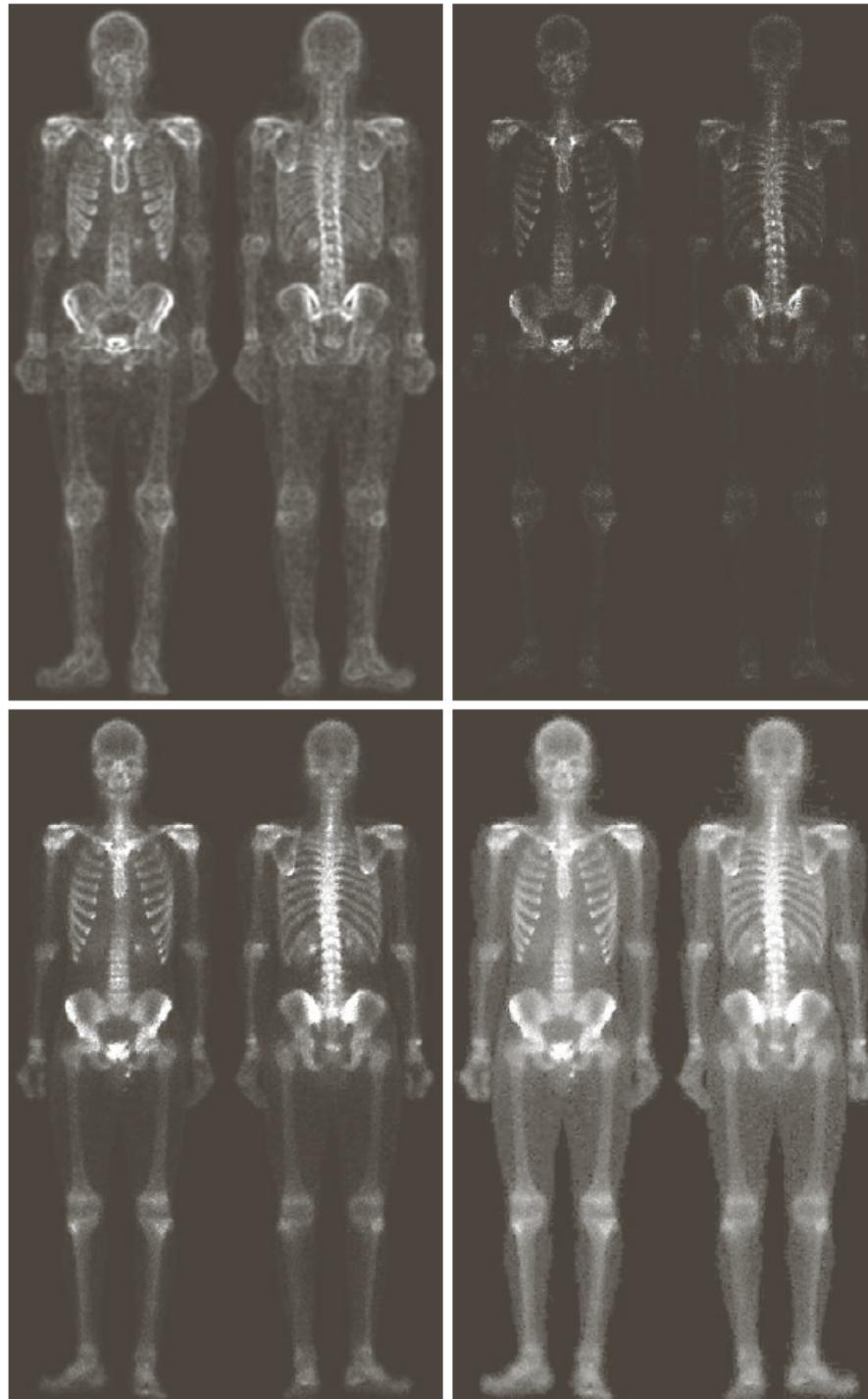
(a) Image of whole body bone scan.  
(b) Laplacian of (a).  
(c) Sharpened image obtained by adding (a) and (b).  
(d) Sobel gradient of (a).

Example:

## Combining Spatial Enhancement Methods

Goal:

Enhance the  
image by  
sharpening it  
and by bringing  
out more of the  
skeletal detail



e | f  
g | h

**FIGURE 3.43**

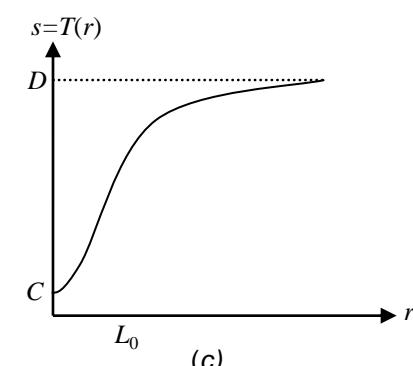
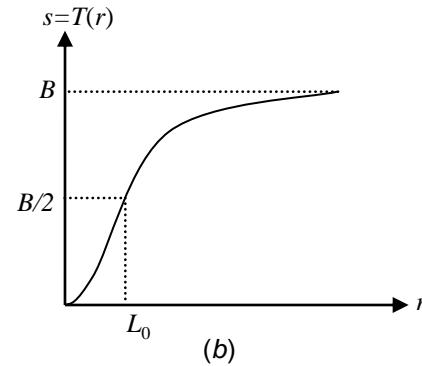
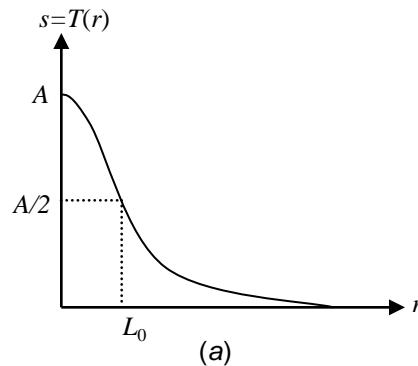
(Continued)

(e) Sobel image smoothed with a  $5 \times 5$  averaging filter. (f) Mask image formed by the product of (c) and (e).

(g) Sharpened image obtained by the sum of (a) and (f). (h) Final result obtained by applying a power-law transformation to (g). Compare (g) and (h) with (a). (Original image courtesy of G.E. Medical Systems.)

# Image Enhancement in the Spatial Domain

- **Example 1(PR3.1):** *Exponentials of the form  $e^{-\alpha r^2}$ ,  $\alpha$  a positive constant, are useful for constructing smooth gray-level transformation functions. Construct the transformation functions having the general shapes shown in the following figures. The constants shown are input parameters, and your proposed transformations must include them in their specifications.*



(a) **General form of the function:**  $s = T(r) = Ae^{-\alpha r^2}$

In Figure (a):  $Ae^{-\alpha L_0^2} = A/2$  solving for  $\alpha$ :  $-\alpha L_0^2 = \ln(0.5) = -0.693$

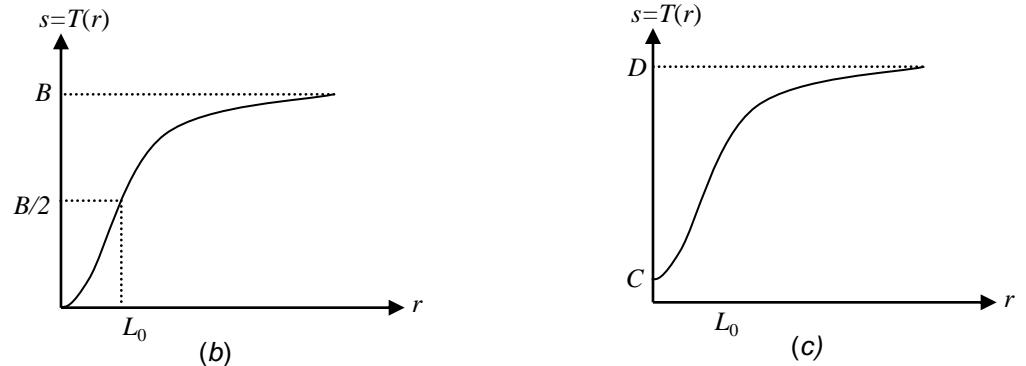
$$\alpha = 0.693 / L_0^2$$

Then:

$$s = T(r) = Ae^{-\frac{0.693}{L_0^2}r^2}$$

# Image Enhancement in the Spatial Domain

- Example 1(PR3.1):



(b) General form of the function:  $s = T(r) = B - Be^{-\alpha r^2} = B(1 - e^{-\alpha r^2})$

In Figure (b):  $B(1 - e^{-\alpha L_0^2}) = B/2$  Then:  $-\alpha L_0^2 = \ln(0.5) = -0.693$

$$s = T(r) = B(1 - e^{-\frac{0.693}{L_0^2}r^2})$$

$$\alpha = 0.693 / L_0^2$$

(c) General form of the function:

$$s = T(r) = (D - C)(1 - e^{-\frac{0.693}{L_0^2}r^2}) + C$$

# Image Enhancement in the Spatial Domain

- **Example 2(PR3.3):** Propose a set of gray-level-slicing transformations capable of producing all the individual bit planes of an 8-bit monochrome image.

Consider  $\text{mod}(x,y)$  to be the modular division resulting the remainder of the integer division  $x/y$ .

For Bit plane 7 (MSB): the following function can be written

$$T(r) = \begin{cases} 255 & \text{for } 2^7 \leq r \\ 0 & \text{otherwise} \end{cases}$$

For Bit plane 6:

$$T(r) = \begin{cases} 255 & \text{for } 2^6 \leq \text{mod}(r, 2^7) \\ 0 & \text{otherwise} \end{cases}$$

For Bit plane 5:

$$T(r) = \begin{cases} 255 & \text{for } 2^5 \leq \text{mod}(r, 2^6) \\ 0 & \text{otherwise} \end{cases}$$

# Image Enhancement in the Spatial Domain

- Example 2(PR3.3):

*For Bit plane 4:*

$$T(r) = \begin{cases} 255 & \text{for } 2^4 \leq \text{mod}(r, 2^5) \\ 0 & \text{otherwise} \end{cases}$$

*For Bit plane 3:*

$$T(r) = \begin{cases} 255 & \text{for } 2^3 \leq \text{mod}(r, 2^4) \\ 0 & \text{otherwise} \end{cases}$$

*For Bit plane 2:*

$$T(r) = \begin{cases} 255 & \text{for } 2^2 \leq \text{mod}(r, 2^3) \\ 0 & \text{otherwise} \end{cases}$$

*For Bit plane 1:*

$$T(r) = \begin{cases} 255 & \text{for } 2^1 \leq \text{mod}(r, 2^2) \\ 0 & \text{otherwise} \end{cases}$$

*For Bit plane 0:*

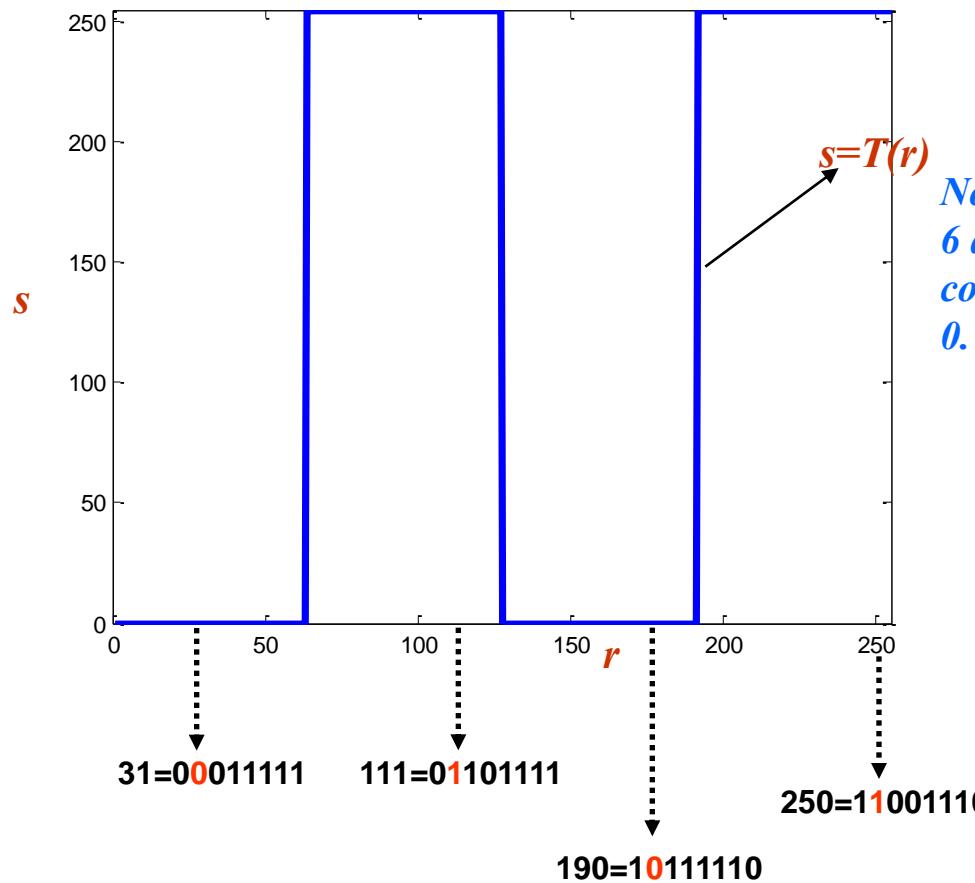
$$T(r) = \begin{cases} 255 & \text{for } 2^0 \leq \text{mod}(r, 2^1) \\ 0 & \text{otherwise} \end{cases}$$

# Image Enhancement in the Spatial Domain

- Example 2(PR3.3):

*Consider Bit plane 6:*

$$T(r) = \begin{cases} 255 & \text{for } 2^6 \leq \text{mod}(r, 2^7) \\ 0 & \text{otherwise} \end{cases}$$



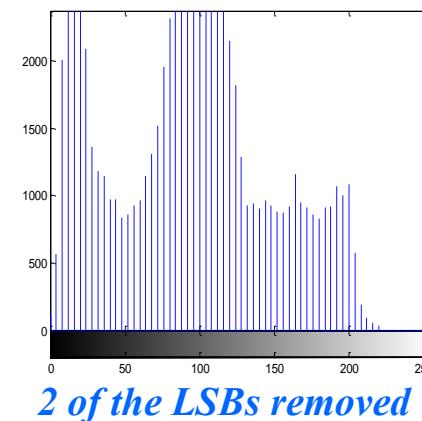
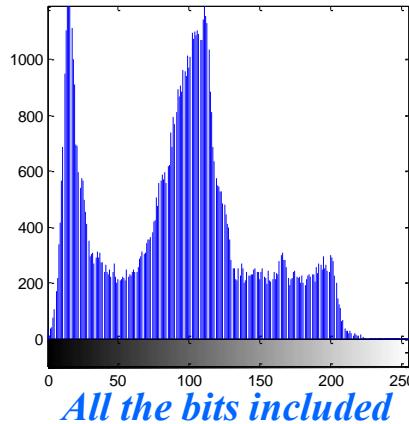
Note that all the 1s corresponding to bit 6 are scaled to 255 and all the bits corresponding to 0s are scaled down to 0.

# Image Enhancement in the Spatial Domain

- **Example 3(PR3.4):** a) What effect would setting to zero the lower-order bit planes have on the histogram of an image in general?  
b) What would be the effect on the histogram if we set to zero the higher-order bit planes instead?

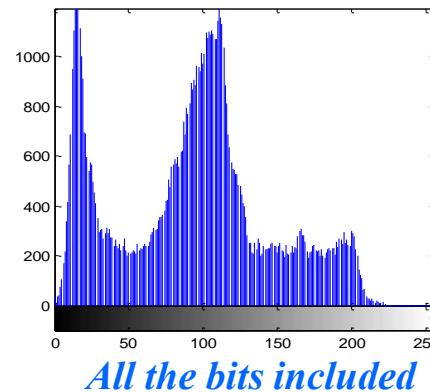
a) Removing the low order bit planes would mean the loss of some high frequency details. Furthermore the image histogram will be more sparse as compared with the all 8-bit plane case.

This is because, there will be no component representing intermediate pixel values such as 1,2,3,4, 5, 6,7 and 9,10,11,12,13,14,15 etc. Instead there will be 0 and 8 and 16 etc. This would cause the height some of the remaining histogram peaks to increase in general. Typically, less variability in gray level values will reduce contrast.

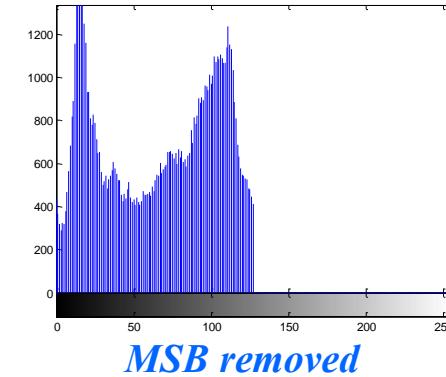


# Image Enhancement in the Spatial Domain

- **Example 3(PR3.4): b)** *What would be the effect on the histogram if we set to zero the higher-order bit planes instead?*  
*b) Removing the high order bit planes would mean the loss of some very important DC components away from the image.*  
*The meaning of this is that the image is much darker and a lot of the low frequency components will be lost.*



*All the bits included*



*MSB removed*

# Image Enhancement in the Spatial Domain

- **Example 4(PR3.6):** Suppose that a digital image is subjected to histogram equalization . Show that a second pass of histogram equalization will produce exactly the same result as the first pass?

Let  $n$  be the total number of pixels and let  $n_{r_j}$  be the number of pixels in the input image with intensity value  $r_j$ . Then, the histogram equalization transformation is:

$$s_k = T(r_k) = \sum_{j=0}^k \frac{n_{r_j}}{n} = \frac{1}{n} \sum_{j=0}^k n_{r_j}$$

Since every pixel (and no others) with value  $r_k$  is mapped to value  $s_k$ , it follows that  $n_{s_k} = n_{r_k}$ . A second pass of histogram equalization would produce values  $v_k$  according to the transformation:

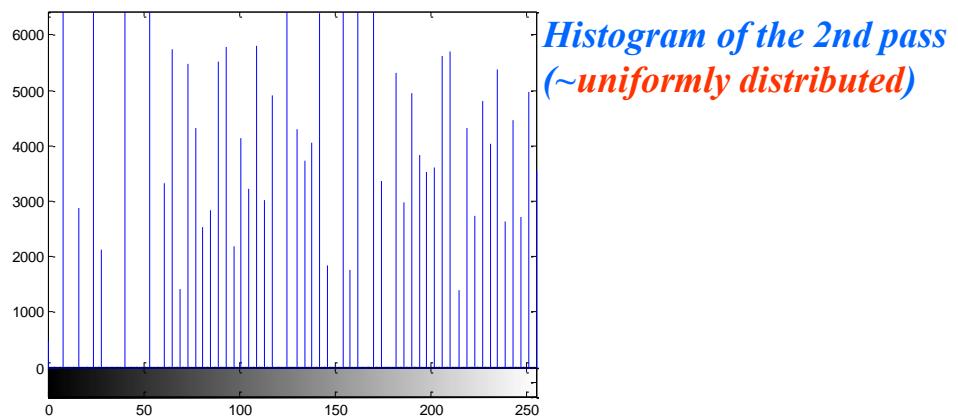
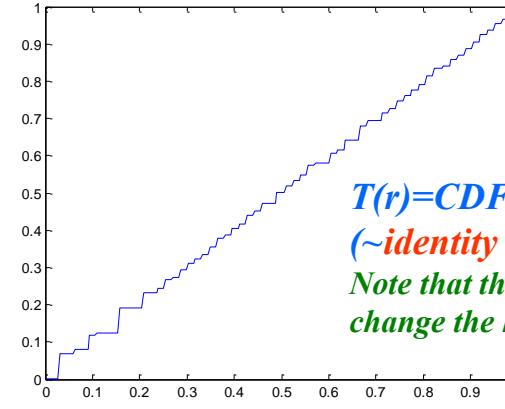
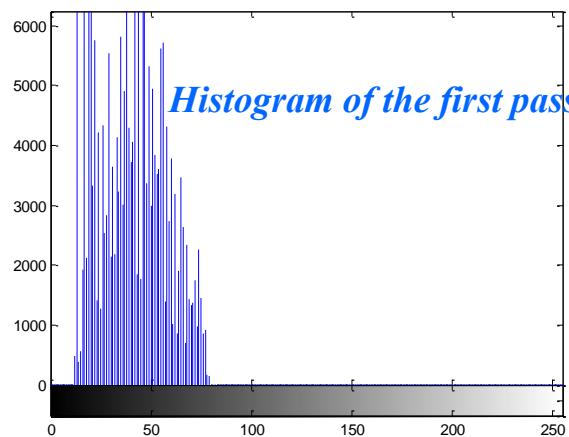
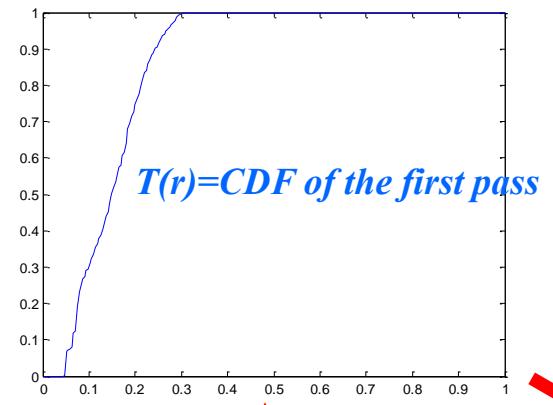
$$v_k = T(s_k) = \sum_{j=0}^k \frac{n_{s_j}}{n}, \text{ but } n_{s_j} = n_{r_j}, \text{ then}$$

$$v_k = T(s_k) = \sum_{j=0}^k \frac{n_{r_j}}{n} = s_k$$

Which shows that a second pass of histogram equalization would yield the same result as the first pass.

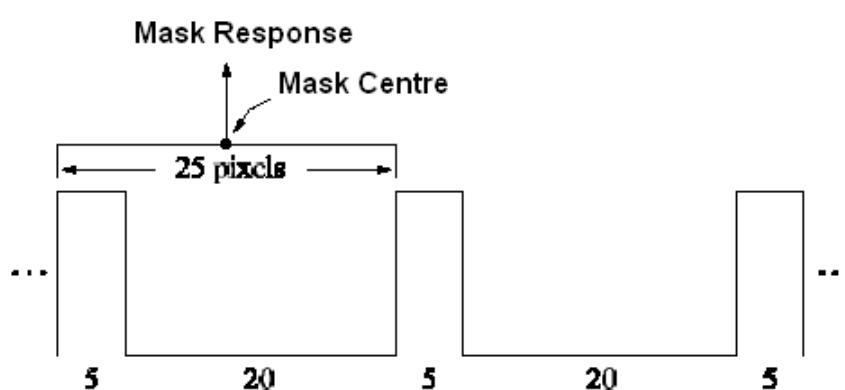
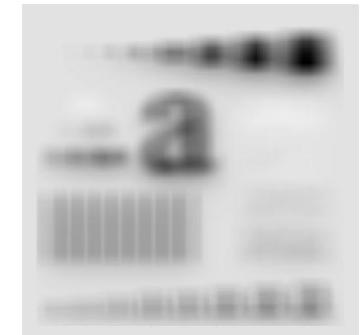
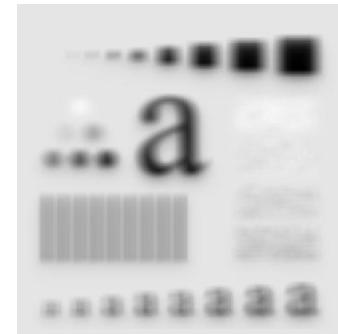
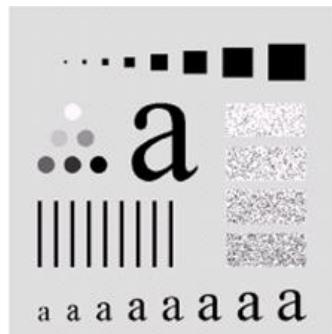
# Image Enhancement in the Spatial Domain

- **Example 4(PR3.6):** Given an image, the following histograms and CDF ( $T(r)$ ) graphs can be obtained.



# Image Enhancement in the Spatial Domain

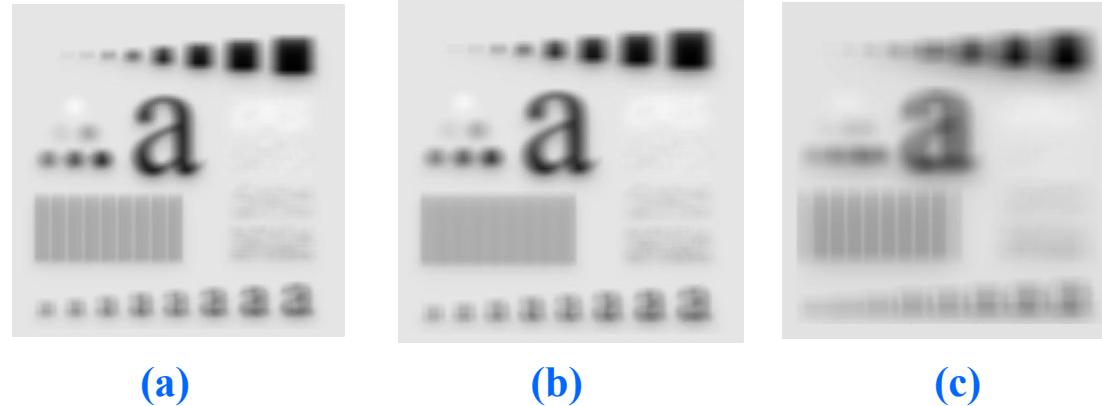
- **Example 6(PR3.22):** *The three images shown below were blurred using the square averaging masks of sizes  $n=23$ ,  $25$  and  $45$  respectively. The vertical bars on the lower part of (a) and (c) are blurred, but clear separation exists between them. However, the bars have merged in in image (b), in spite of the fact that the mask that produced the image is significantly smaller than the mask produced image (c) .Explain this.*
- *(Note that the vertical bars are 5 pixels wide and 20 pixels apart.)*



# Image Enhancement in the Spatial Domain

- Example 6(PR3.22):

*Note that the vertical bars are 5 pixels wide and 20 pixels apart.*



*The reason why the mask with size  $n=25$  producing merged uniform region around the bars is because of the sizes of the bars and the separation of the bars in the horizontal direction. The width of each bar is 5 pixels and each bar is separated by 20 pixels.*

*In such an environment as the mask moves in the horizontal direction there will be 5 black and 20 light gray pixels in each row at a time. This will provide the same average value for each pixel in the region producing a merged uniform gray level.*

*However this will not be the same in 23 and 45 pixel masks!!*

# Image Enhancement in the Spatial Domain

- **Example 7(PR3.24):** In a given application an averaging mask is applied to input images to reduce noise, and then a Laplacian mask is applied to enhance small details. Would the result be the same if the order of these operations were reversed?

Laplacian operation and averaging can be expressed by the following 3x3 masks:

$$\text{Laplacian} \rightarrow g(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

$$\text{Averaging} \rightarrow h(x, y) = \frac{1}{9} \sum_{i=1}^9 f_i$$

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Averaging Mask

0	1	0
1	-4	1
0	1	0

Laplacian Mask

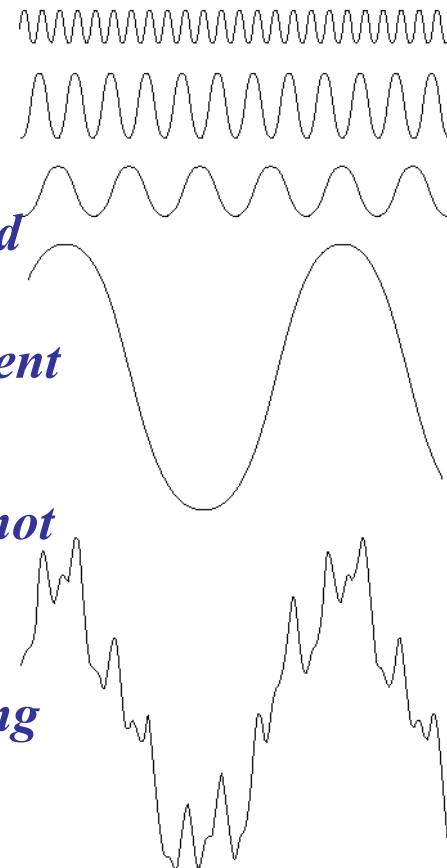
Both of the two operators are multiplying the pixels in a 3x3 neighborhood with constant numbers and perform the addition. Therefore, these operations are linear operations.

The order of two linear operations does not matter. The result would be same in any order.

# Image Enhancement in the *Frequency Domain*

## Fourier Transform

- **Fourier Series:** Any function that periodically repeats itself can be expressed as the sum of sines/cosines of different frequencies, each multiplied with a different coefficient.
- **Fourier Transform:** Functions that are not periodic, whose area under the curve is finite, can be expressed as the integral of sines and cosines multiplied by a weighting function.



**FIGURE 4.1** The function at the bottom is the sum of the four functions above it. Fourier's idea in 1807 that periodic functions could be represented as a weighted sum of sines and cosines was met with skepticism.

---

# Image Enhancement in the *Frequency Domain*

## 1D Continuous Fourier Transform

- *The Fourier Transform is an important tool in Image Processing, and is directly related to filter theory, since a filter, which is a convolution in the spatial domain, is a simple multiplication in the frequency domain.*

### *• 1-D Continuous Fourier Transform*

*The Fourier transform,  $F(u)$ , of a single variable continuous function,  $f(x)$ , is defined by:*

$$F(u) = \int_{-\infty}^{\infty} f(x)e^{-j2\pi ux} dx$$

*Given Fourier transform of a function  $F(u)$ , the inverse Fourier transform can be used to obtain,  $f(x)$ , by:*

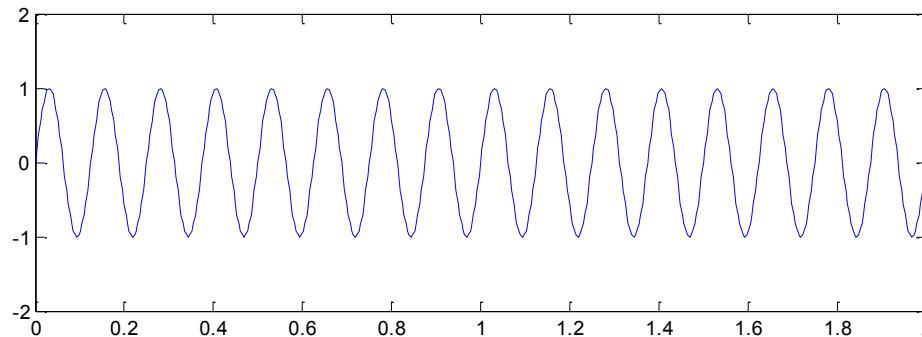
$$f(x) = \int_{-\infty}^{\infty} F(u)e^{j2\pi ux} du$$

*Note:  $F(u)$ , is the frequency spectrum, where,  $u$  represents the frequency, and  $f(x)$  is the signal where  $x$  represents time/space.       $j = \sqrt{-1}$*

# Image Enhancement in the *Frequency Domain*

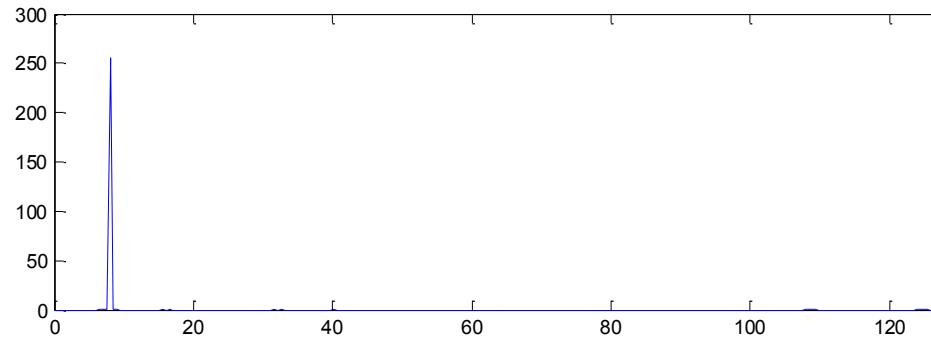
## 1D Continuous Fourier Transform

Time/Space domain



*Sine wave*

Frequency domain

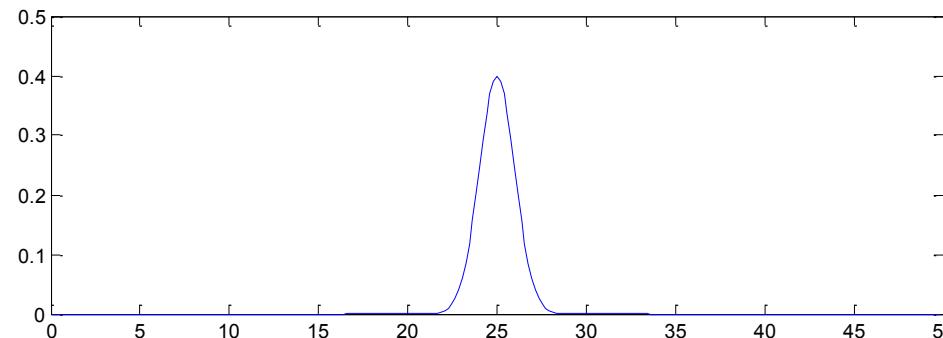


*Delta function*

# Image Enhancement in the *Frequency Domain*

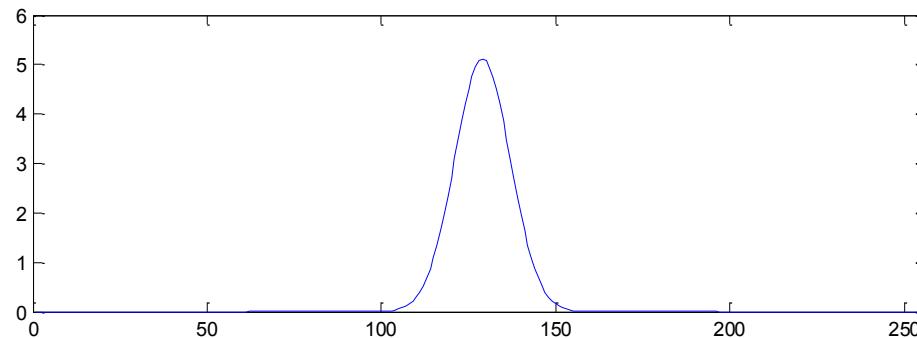
## 1D Continuous Fourier Transform

Time/Space domain



Gaussian

Frequency domain

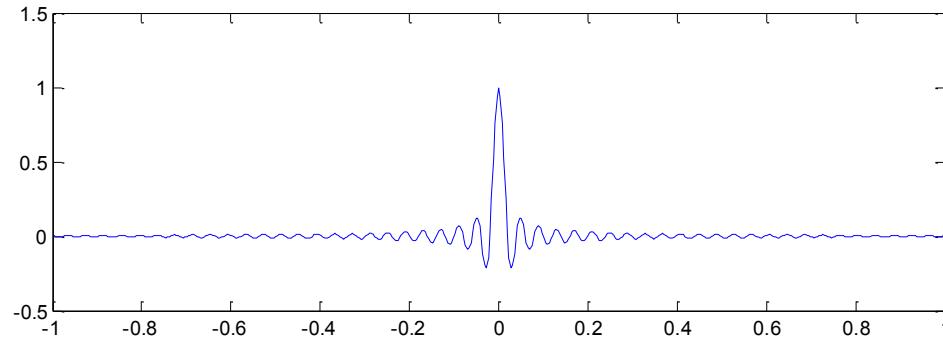


Gaussian

# Image Enhancement in the *Frequency Domain*

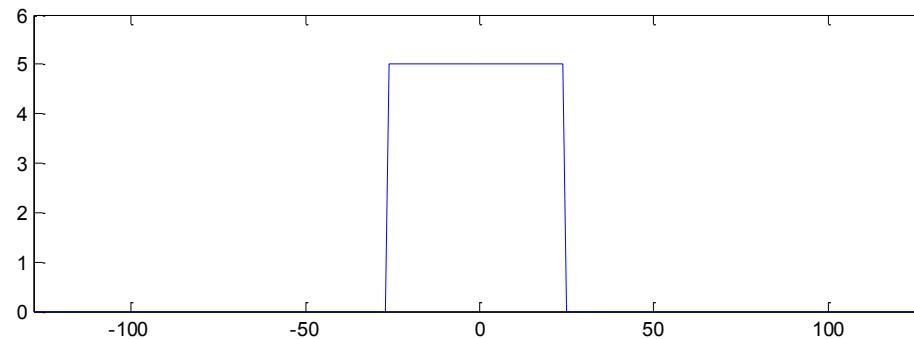
## 1D Continuous Fourier Transform

*Time/Space domain*



*Sinc function*

*Frequency domain*



*Square wave*

# Image Enhancement in the *Frequency Domain*

## Fourier Transform pairs (spatial versus Frequency)

Spatial domain

$$f(x)$$

↑ box( $x$ )

Frequency domain

$$F(s) = \int_{-\infty}^{\infty} f(x)e^{-i2\pi sx} dx$$

↑ sinc( $s$ )

...

...

$$\uparrow \text{gauss}(x; \sigma)$$

...

$$\uparrow \text{gauss}(s; 1/\sigma)$$

...

...

$$\uparrow \text{sinc}(s)$$

...

$$\uparrow \text{box}(x)$$

...

---

# Image Enhancement in the *Frequency Domain*

## 1D Discrete Fourier Transform

### • *1-D Discrete Fourier Transform*

The Fourier transform,  $F(u)$ , of a discrete function of one variable,  $f(x)$ ,  $x=0, 1, 2, \dots, M-1$ , is given by:

$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M}$$

The inverse Discrete Fourier Transform can be used to calculate  $f(x)$ , by:

$$f(x) = \sum_{u=0}^{M-1} F(u) e^{j2\pi ux/M}$$

Note:  $F(u)$ , which is the Fourier transform of  $f(x)$  contains discrete **complex** quantities and it has the same number of components as  $f(x)$ .

$$e^{j\theta} = \cos \theta + j \sin \theta$$

---

# Image Enhancement in the *Frequency Domain*

## 1D Discrete Fourier Transform

### • *1-D Discrete Fourier Transform*

Then;

$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) [\cos 2\pi ux / M - j \sin 2\pi ux / M]$$

• *The Fourier Transform generates complex quantities. The magnitude or the spectrum of the Fourier transform is given by:*

$$|F(u)| = [R^2(u) + I^2(u)]^{1/2} \quad \begin{aligned} R(u) &\text{ is the Real Part and} \\ I(u) &\text{ is the Imaginary Part} \end{aligned}$$

• *The Phase Spectrum of the transform refers to the angles between the real and imaginary components and it is denoted by:*

$$\phi(u) = \tan^{-1} \left[ \frac{I(u)}{R(u)} \right]$$

---

# Image Enhancement in the *Frequency Domain*

## 1D Discrete Fourier Transform

- **1-D Discrete Fourier Transform**

- **The Power Spectrum/spectral density is defined as the square of the Fourier spectrum and denoted by**

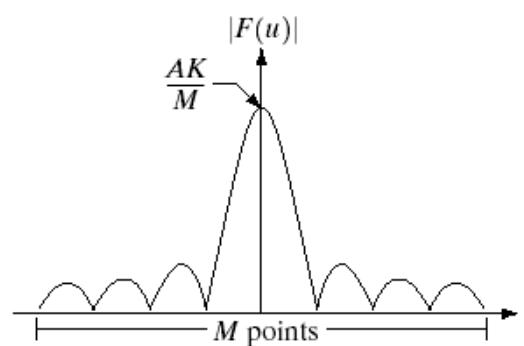
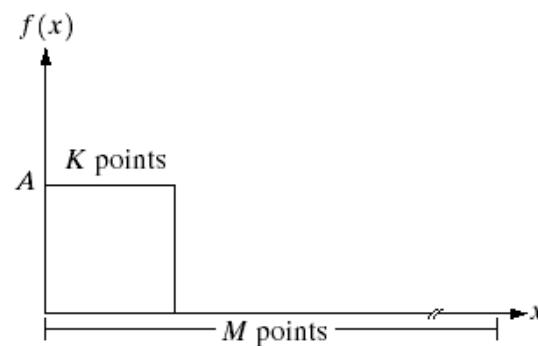
$$P(u) = |F(u)|^2 = R^2(u) + I^2(u)$$

- **The power spectrum can be used, for example to separate a portion of a specified frequency (i.e. low frequency) power from the power spectrum and monitor the effect. Typically used to define the cut off frequencies used in lowpass and highpass filtering.**

- **We primarily use the Fourier Spectrum for image enhancement applications.**

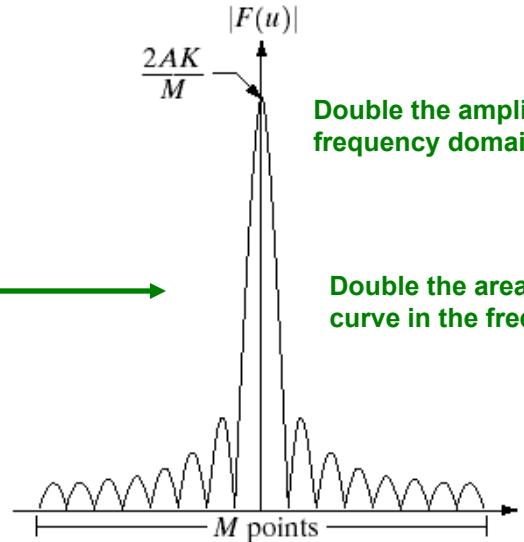
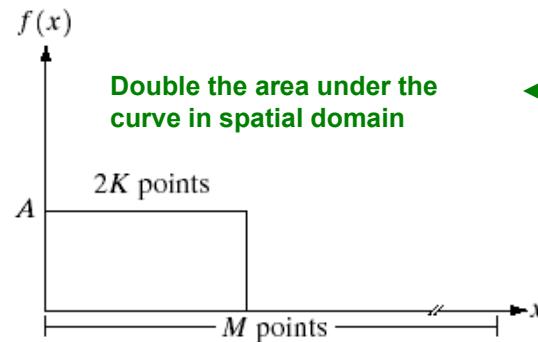
# Image Enhancement in the *Frequency Domain*

## 1D Discrete Fourier Transform



a	b
c	d

**FIGURE 4.2** (a) A discrete function of  $M$  points, and (b) its Fourier spectrum. (c) A discrete function with twice the number of nonzero points, and (d) its Fourier spectrum.



Double the amplitude in the frequency domain

Double the area under the curve in the frequency domain

---

## • 2D Orthogonal and Unitary Transform:

- $v(m,n)$ : Transformed coefficients
- $\mathbf{V}=\{v(m,n)\}$ : Transformed Image
- Orthonormality requires:

$$u_{P,Q}(m,n) = \sum_{k=0}^{P-1} \sum_{l=0}^{Q-1} v(k,l) a_{k,l}^*(m,n) \quad P \leq N, Q \leq N$$

$$u_{P,Q}(m,n) = \arg \min_{\hat{u}} \left\{ \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} [u(m,n) - \hat{u}(m,n)]^2 \right\}$$

---

- Separable Unitary Transform:

- Computational Complexity of former:  $O(N^4)$
- With Separable Transform:  $O(N^3)$

$$a_{k,l}(m,n) = a_k(m)b_l(n) \triangleq a(k,m)b(l,n)$$

$a_k(m), b_l(n)$ : One Dimensional Complete Orthonormal Basis

- Orthonormality and Completeness:

- $\mathbf{A}=\{a(k,m)\}$  and  $\mathbf{B}=\{b(l,n)\}$  are unitary:

- Usually  $\mathbf{B}$  is selected same as  $\mathbf{A}$  ( $\mathbf{A}=\mathbf{B}$ ):

$$A(A^*)^T = A^T A^* = I$$

- Unitary Transform!

---

## • 2D Orthogonal and Unitary Transform:

- $v(m,n)$ : Transformed coefficients
- $\mathbf{V}=\{v(m,n)\}$ : Transformed Image
- Orthonormality requires:

$$u_{P,Q}(m,n) = \sum_{k=0}^{P-1} \sum_{l=0}^{Q-1} v(k,l) a_{k,l}^*(m,n) \quad P \leq N, Q \leq N$$

$$u_{P,Q}(m,n) = \arg \min_{\hat{u}} \left\{ \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} [u(m,n) - \hat{u}(m,n)]^2 \right\}$$

---

- 2D Orthogonal and Unitary Transform:

- Orthogonal Series Expansion:

$$v(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} u(m,n) a_{k,l}(m,n) \quad 0 \leq k, l \leq N-1$$

$$u(m,n) = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} v(k,l) a_{k,l}^*(m,n) \quad 0 \leq m, n \leq N-1$$

- $\{a_{k,l}(m,n)\}$ : a set of complete orthonormal basis:

- Orthonormality:  $\sum_{m=0}^{N-1} \sum_{n=0}^{N-1} a_{k,l}(m,n) a_{k',l'}^*(m,n) = \delta(k-k', l-l')$

- Completeness:  $\sum_{k=0}^{N-1} \sum_{l=0}^{N-1} a_{k,l}(m,n) a_{k,l}^*(m',n') = \delta(m-m', n-n')$

---

- Two Dimensional Fourier Transform:

$$\begin{aligned}
 v(k, l) &= \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} u(m, n) W_N^{km} W_N^{ln}, \quad W_N \triangleq \exp\left(\frac{-j2\pi}{N}\right) \\
 u(m, n) &= \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} v(k, l) W_N^{-km} W_N^{-ln} \\
 v(k, l) &= \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} u(m, n) W_N^{km} W_N^{ln} \\
 u(m, n) &= \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} v(k, l) W_N^{-km} W_N^{-ln}
 \end{aligned}
 \left. \right\} \text{Unitary DFT Pair}$$

- Matrix Notation:  $\mathbf{V} = \mathbf{F} \mathbf{U} \mathbf{F}^* \Leftrightarrow \mathbf{U} = \mathbf{F}^* \mathbf{V} \mathbf{F}^*$

$$\mathbf{F} = \left\{ \frac{1}{\sqrt{N}} W_N^{kn} \right\}_{k,n=0}^{N-1}$$

---

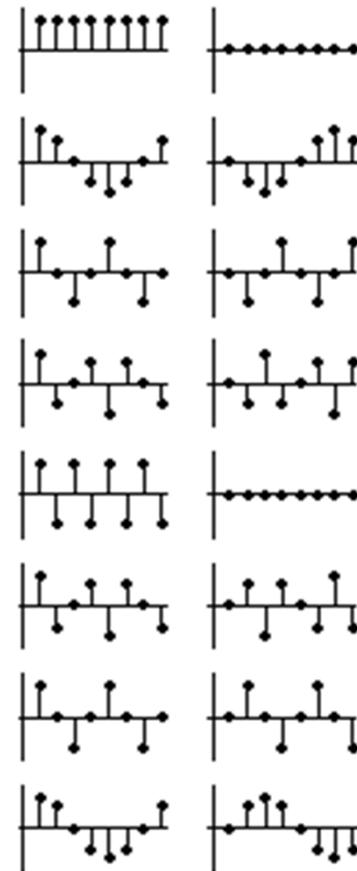
## • 2D Orthogonal and Unitary Transform:

- $v(m,n)$ : Transformed coefficients
- $\mathbf{V}=\{v(m,n)\}$ : Transformed Image
- Orthonormality requires:

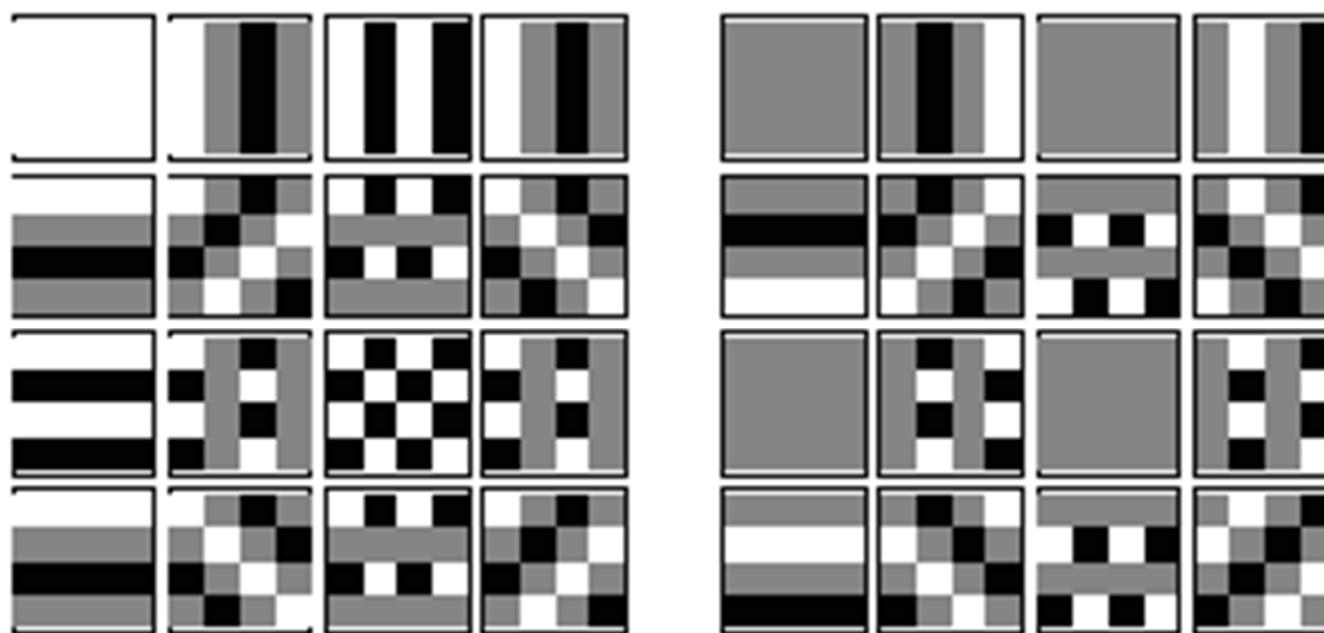
$$u_{P,Q}(m,n) = \sum_{k=0}^{P-1} \sum_{l=0}^{Q-1} v(k,l) a_{k,l}^*(m,n) \quad P \leq N, Q \leq N$$

$$u_{P,Q}(m,n) = \arg \min_{\hat{u}} \left\{ \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} [u(m,n) - \hat{u}(m,n)]^2 \right\}$$

- 
- Basis of DFT (Real and Imaginary):



- 
- Basis of DFT (Real and Imaginary):



---

# Image Enhancement in the *Frequency Domain*

## 2D Discrete Fourier Transform

The Fourier transform of a 2D discrete function (image)  $f(x,y)$  of size  $M \times N$  is given by:

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}$$

$u=0, 1, 2, \dots, M-1$ , and  $v=0, 1, 2, \dots, N-1$  and the inverse 2D Discrete Fourier Transform can be calculated by:

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)}$$

$x=0, 1, 2, \dots, M-1$ , and  $y=0, 1, 2, \dots, N-1$ .

---

# Image Enhancement in the *Frequency Domain*

## 2D Discrete Fourier Transform

The 2D **Fourier Spectrum**, **Phase Spectrum** and **Power Spectrum** can be respectively denoted by:

$$|F(u, v)| = [R^2(u, v) + I^2(u, v)]^{1/2} \quad \text{Magnitude/Fourier Spectrum}$$

$$\phi(u, v) = \tan^{-1} \left[ \frac{I(u, v)}{R(u, v)} \right] \quad \text{Phase Spectrum}$$

$$P(u, v) = |F(u, v)|^2 = R^2(u, v) + I^2(u, v) \quad \text{Power Spectrum}$$

# Properties of the 2-D DFT

## translation and rotation

$$f(x, y)e^{j2\pi(\mu_0x/M + \nu_0y/N)} \Leftrightarrow F(\mu - \mu_0, \nu - \nu_0)$$

and

$$f(x - x_0, y - y_0) \Leftrightarrow F(\mu, \nu)e^{-j2\pi(\mu x_0/M + \nu y_0/N)}$$

Using the polar coordinates

$$x = r \cos \theta \quad y = r \sin \theta \quad \mu = \omega \cos \varphi \quad \nu = \omega \sin \varphi$$

results in the following transform pair:

$$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0)$$

# Properties of the 2-D DFT

## periodicity

2-D Fourier transform and its inverse are infinitely periodic

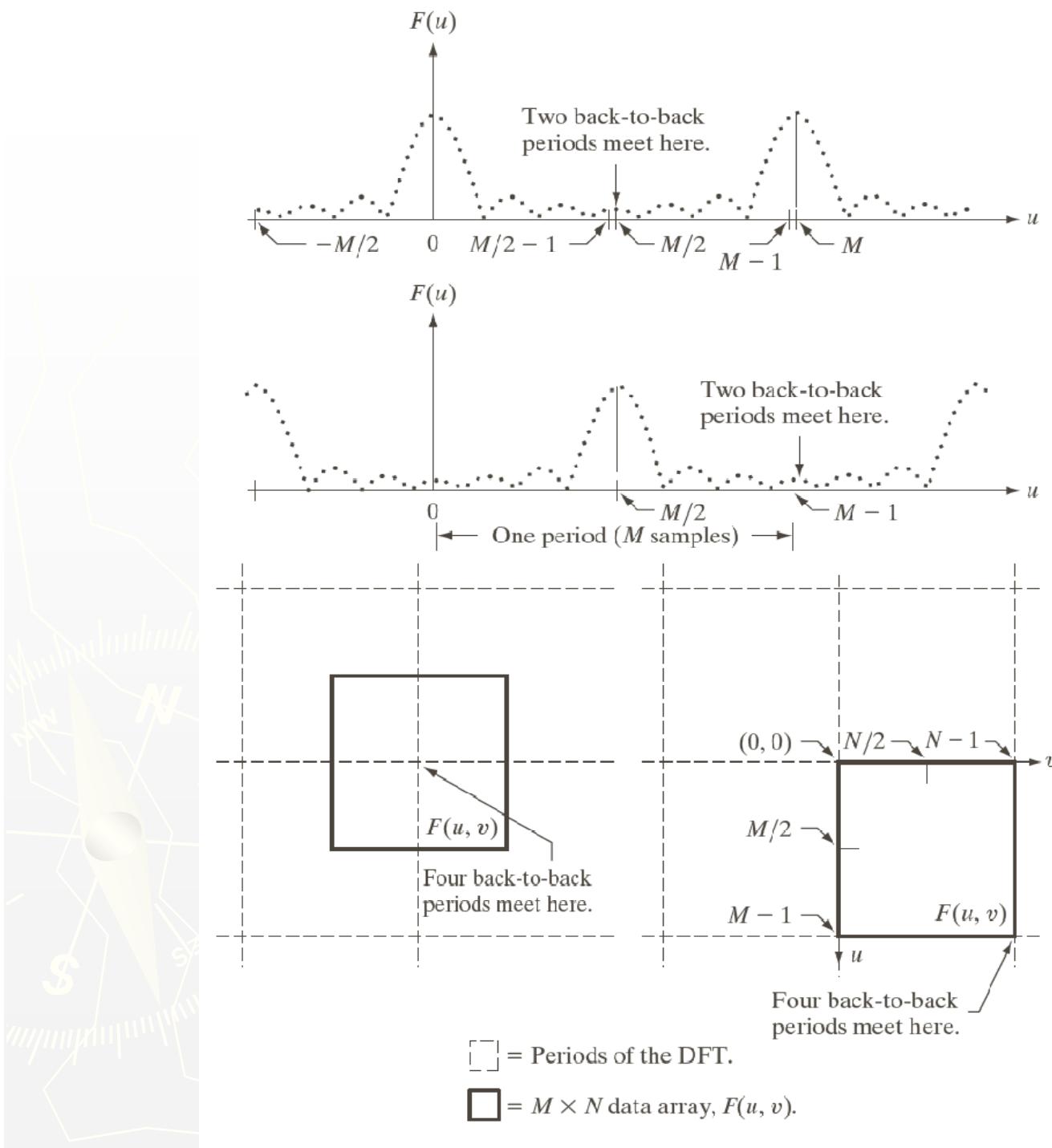
$$F(\mu, \nu) = F(\mu + k_1 M, \nu) = F(\mu, \nu + k_2 N) = F(\mu + k_1 M, \nu + k_2 N)$$

$$f(x, y) = f(x + k_1 M, y) = f(x, y + k_2 N) = f(x + k_1 M, y + k_2 N)$$

$$f(x)e^{j2\pi(\mu_0 x/M)} \Leftrightarrow F(\mu - \mu_0)$$

$$\mu_0 = M/2, \quad f(x)(-1)^x \Leftrightarrow F(\mu - M/2)$$

$$f(x, y)(-1)^{x+y} \Leftrightarrow F(\mu - M/2, \nu - N/2)$$



a  
b  
c d

**FIGURE 4.23**

Centering the Fourier transform.

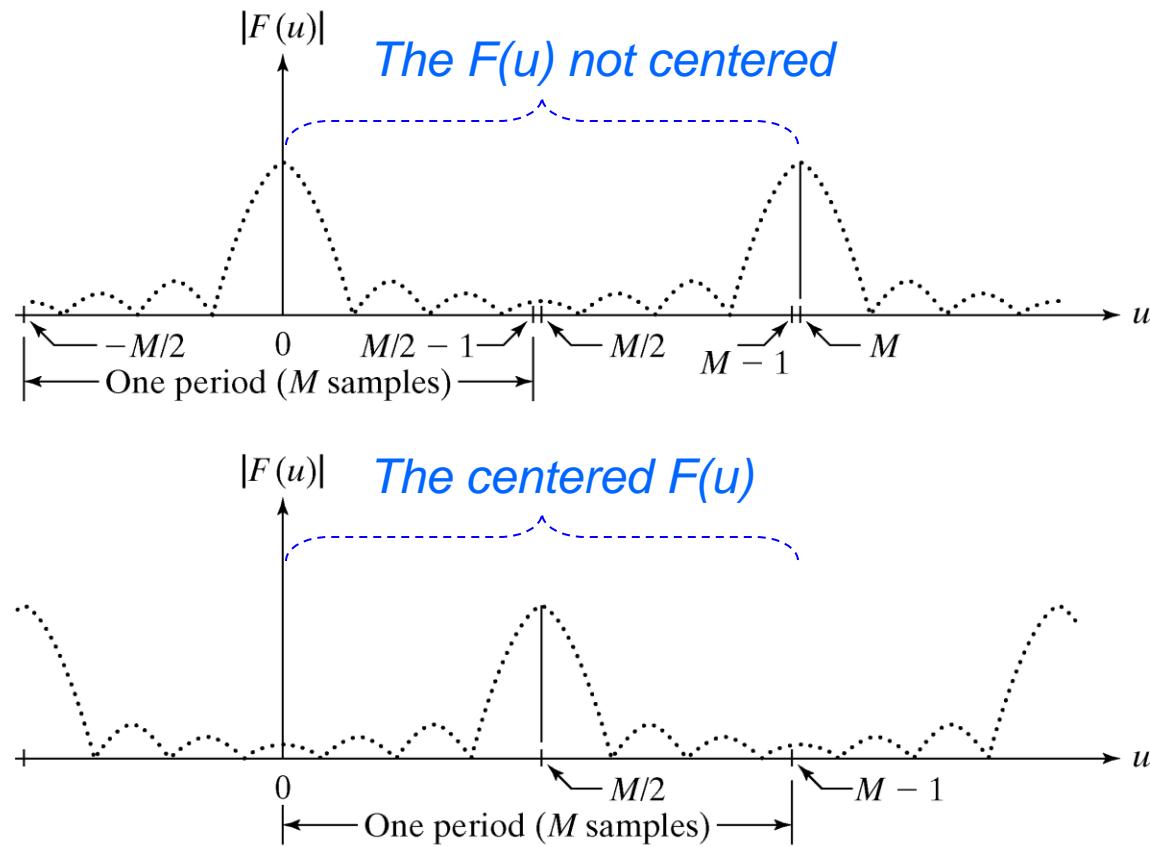
- (a) A 1-D DFT showing an infinite number of periods.
- (b) Shifted DFT obtained by multiplying  $f(x)$  by  $(-1)^x$  before computing  $F(u)$ .
- (c) A 2-D DFT showing an infinite number of periods. The solid area is the  $M \times N$  data array,  $F(u, v)$ , obtained with Eq. (4.5-15). This array consists of four quarter periods.
- (d) A Shifted DFT obtained by multiplying  $f(x, y)$  by  $(-1)^{x+y}$  before computing  $F(u, v)$ . The data now contains one complete, centered period, as in (b).

# Image Enhancement in the *Frequency Domain*

## 2D Discrete Fourier Transform

*The Periodicity property:  $F(u)$  in 1D DFT has a period of  $M$*

*The Symmetry property: The magnitude of the transform is centered on the origin.*



a  
b

**FIGURE 4.1**

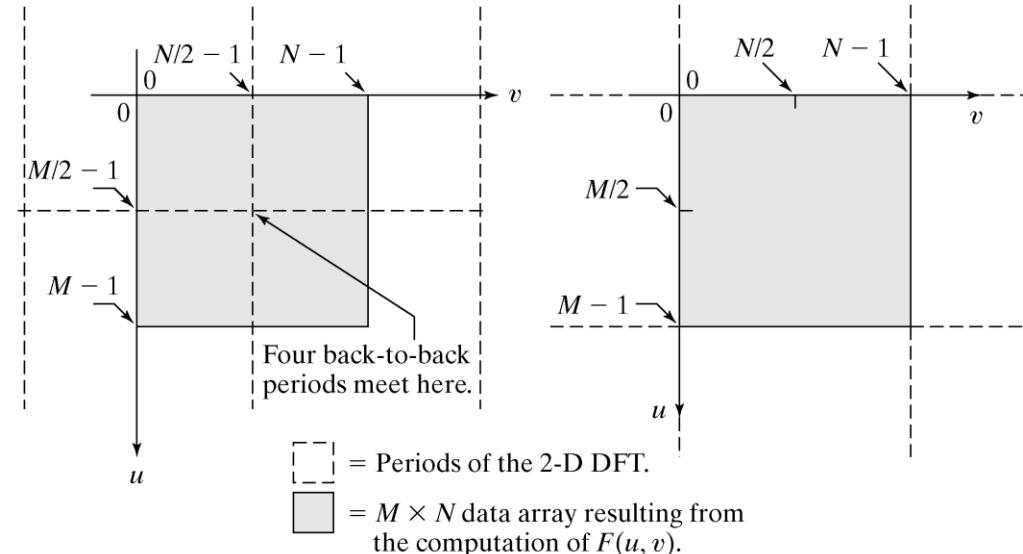
(a) Fourier spectrum showing back-to-back half periods in the interval  $[0, M - 1]$ .  
(b) Centered spectrum in the same interval, obtained by multiplying  $f(x)$  by  $(-1)^x$  prior to computing the Fourier transform.

# Image Enhancement in the *Frequency Domain*

## 2D Discrete Fourier Transform

*The Periodicity property:  $F(u,v)$  in 2D DFT has a period of  $N$  in horizontal and  $M$  in vertical directions*

*The Symmetry property: The magnitude of the transform is centered on the origin.*

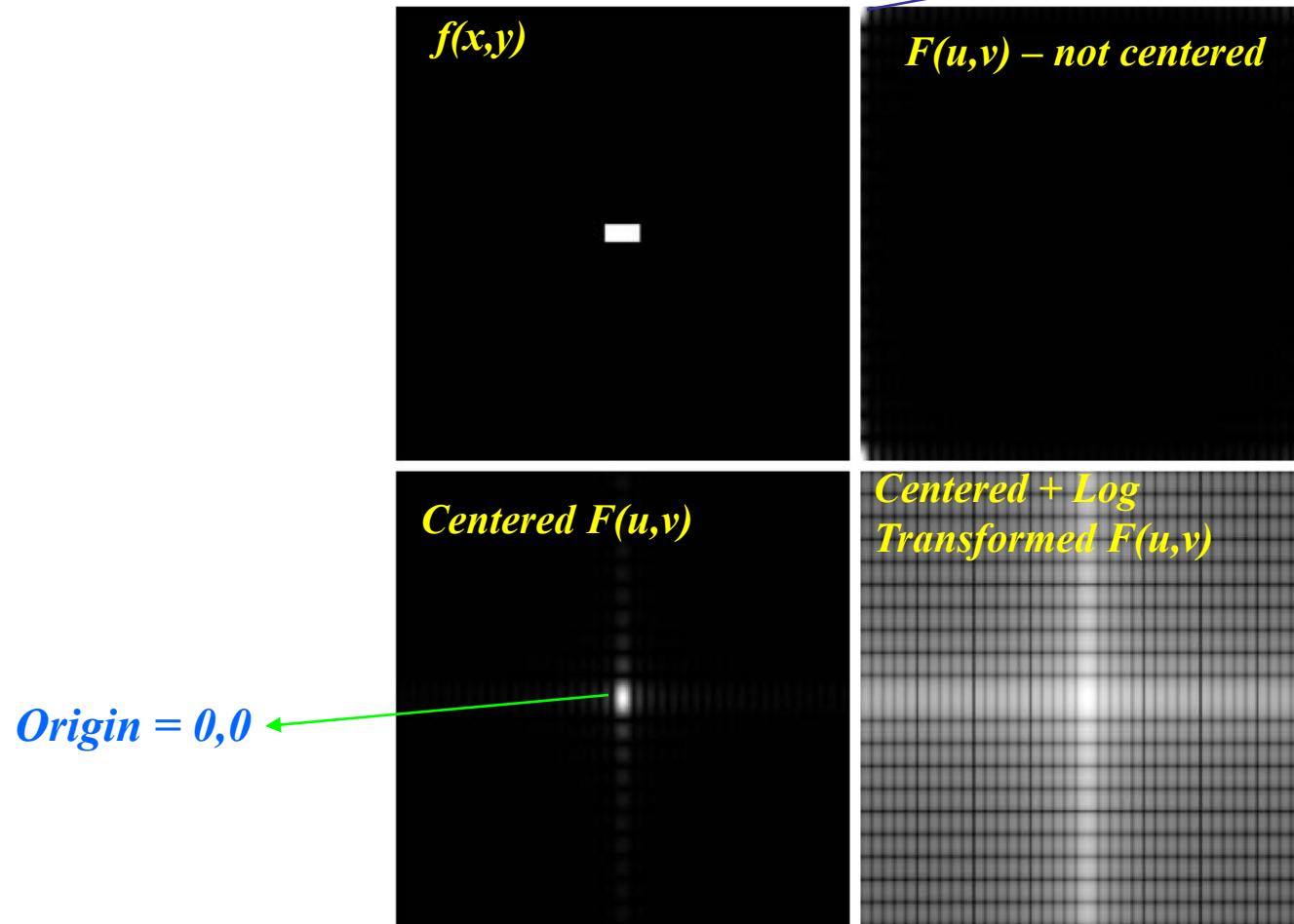


a b

**FIGURE 4.2** (a)  $M \times N$  Fourier spectrum (shaded), showing four back-to-back quarter periods contained in the spectrum data. (b) Spectrum obtained by multiplying  $f(x, y)$  by  $(-1)^{x+y}$  prior to computing the Fourier transform. Only one period is shown shaded because this is the data that would be obtained by an implementation of the equation for  $F(u, v)$ .

# Image Enhancement in the *Frequency Domain*

## 2D Discrete Fourier Transform



**FIGURE 4.3**

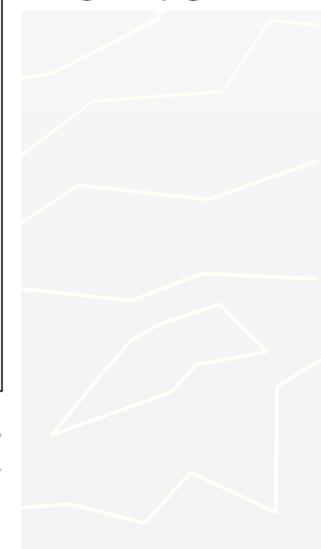
- (a) A simple image.
- (b) Fourier spectrum.
- (c) Centered spectrum.
- (d) Spectrum visually enhanced by a log transformation.

# Properties of the 2-D DFT

## Symmetry

	<b>Spatial Domain<sup>†</sup></b>		<b>Frequency Domain<sup>†</sup></b>
1)	$f(x, y)$ real	$\Leftrightarrow$	$F^*(u, v) = F(-u, -v)$
2)	$f(x, y)$ imaginary	$\Leftrightarrow$	$F^*(-u, -v) = -F(u, v)$
3)	$f(x, y)$ real	$\Leftrightarrow$	$R(u, v)$ even; $I(u, v)$ odd
4)	$f(x, y)$ imaginary	$\Leftrightarrow$	$R(u, v)$ odd; $I(u, v)$ even
5)	$f(-x, -y)$ real	$\Leftrightarrow$	$F^*(u, v)$ complex
6)	$f(-x, -y)$ complex	$\Leftrightarrow$	$F(-u, -v)$ complex
7)	$f^*(x, y)$ complex	$\Leftrightarrow$	$F^*(-u - v)$ complex
8)	$f(x, y)$ real and even	$\Leftrightarrow$	$F(u, v)$ real and even
9)	$f(x, y)$ real and odd	$\Leftrightarrow$	$F(u, v)$ imaginary and odd
10)	$f(x, y)$ imaginary and even	$\Leftrightarrow$	$F(u, v)$ imaginary and even
11)	$f(x, y)$ imaginary and odd	$\Leftrightarrow$	$F(u, v)$ real and odd
12)	$f(x, y)$ complex and even	$\Leftrightarrow$	$F(u, v)$ complex and even
13)	$f(x, y)$ complex and odd	$\Leftrightarrow$	$F(u, v)$ complex and odd

**TABLE 4.1** Some symmetry properties of the 2-D DFT and its inverse.  $R(u, v)$  and  $I(u, v)$  are the real and imaginary parts of  $F(u, v)$ , respectively. The term *complex* indicates that a function has nonzero real and imaginary parts.



<sup>†</sup>Recall that  $x, y, u$ , and  $v$  are *discrete* (integer) variables, with  $x$  and  $u$  in the range  $[0, M - 1]$ , and  $y$ , and  $v$  in the range  $[0, N - 1]$ . To say that a complex function is *even* means that its real *and* imaginary parts are even, and similarly for an odd complex function.

# Properties of the 2-D DFT

## Fourier Spectrum and Phase Angle

2-D DFT in polar form

$$F(u, v) = |F(u, v)| e^{j\phi(u, v)}$$

Fourier spectrum

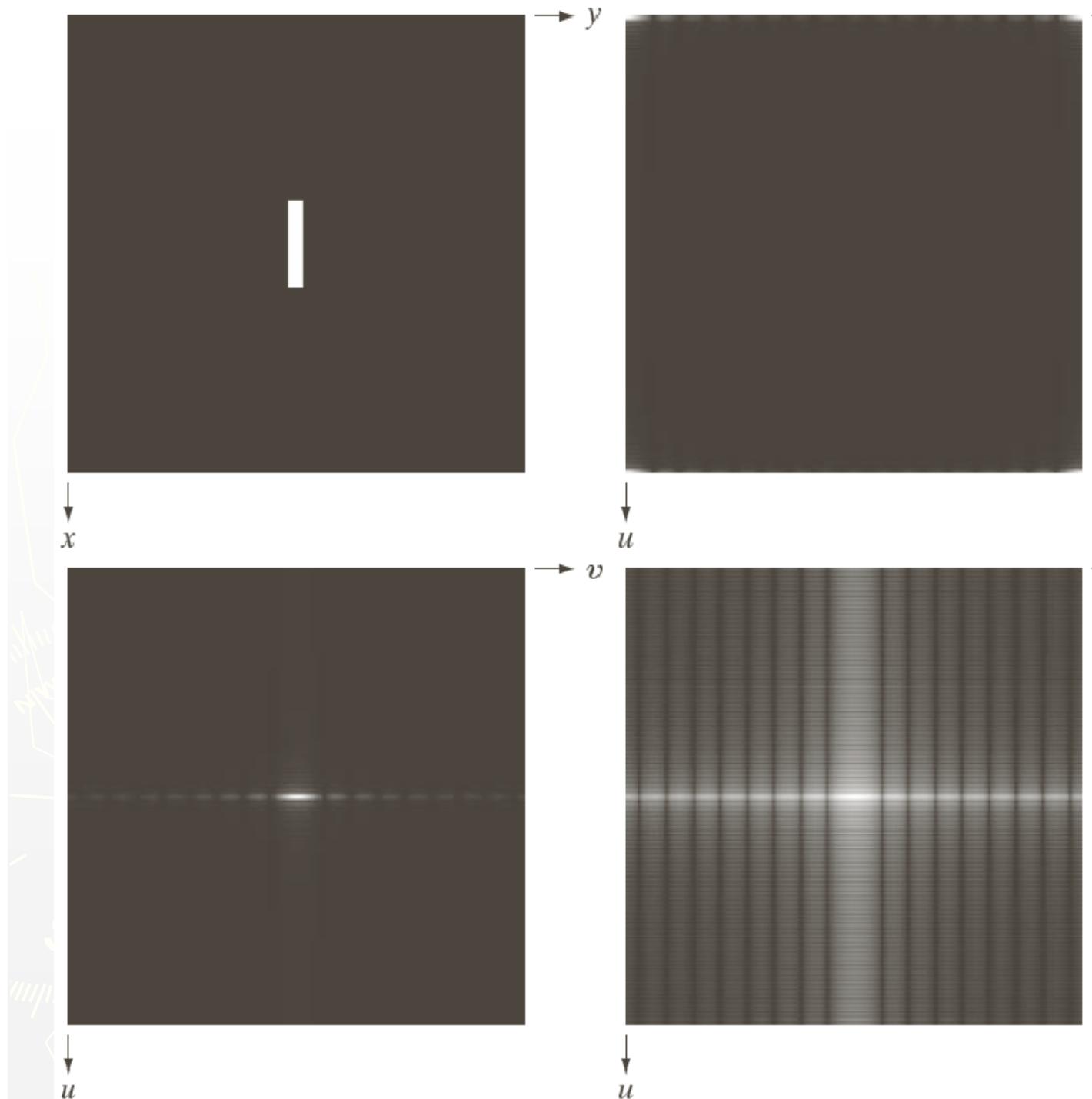
$$|F(u, v)| = \left[ R^2(u, v) + I^2(u, v) \right]^{1/2}$$

Power spectrum

$$P(u, v) = |F(u, v)|^2 = R^2(u, v) + I^2(u, v)$$

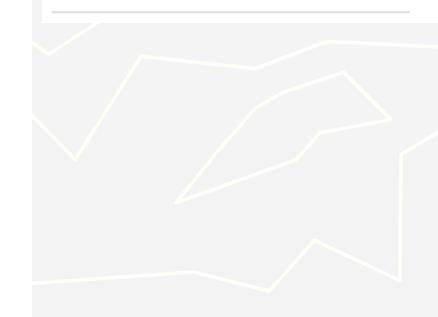
Phase angle

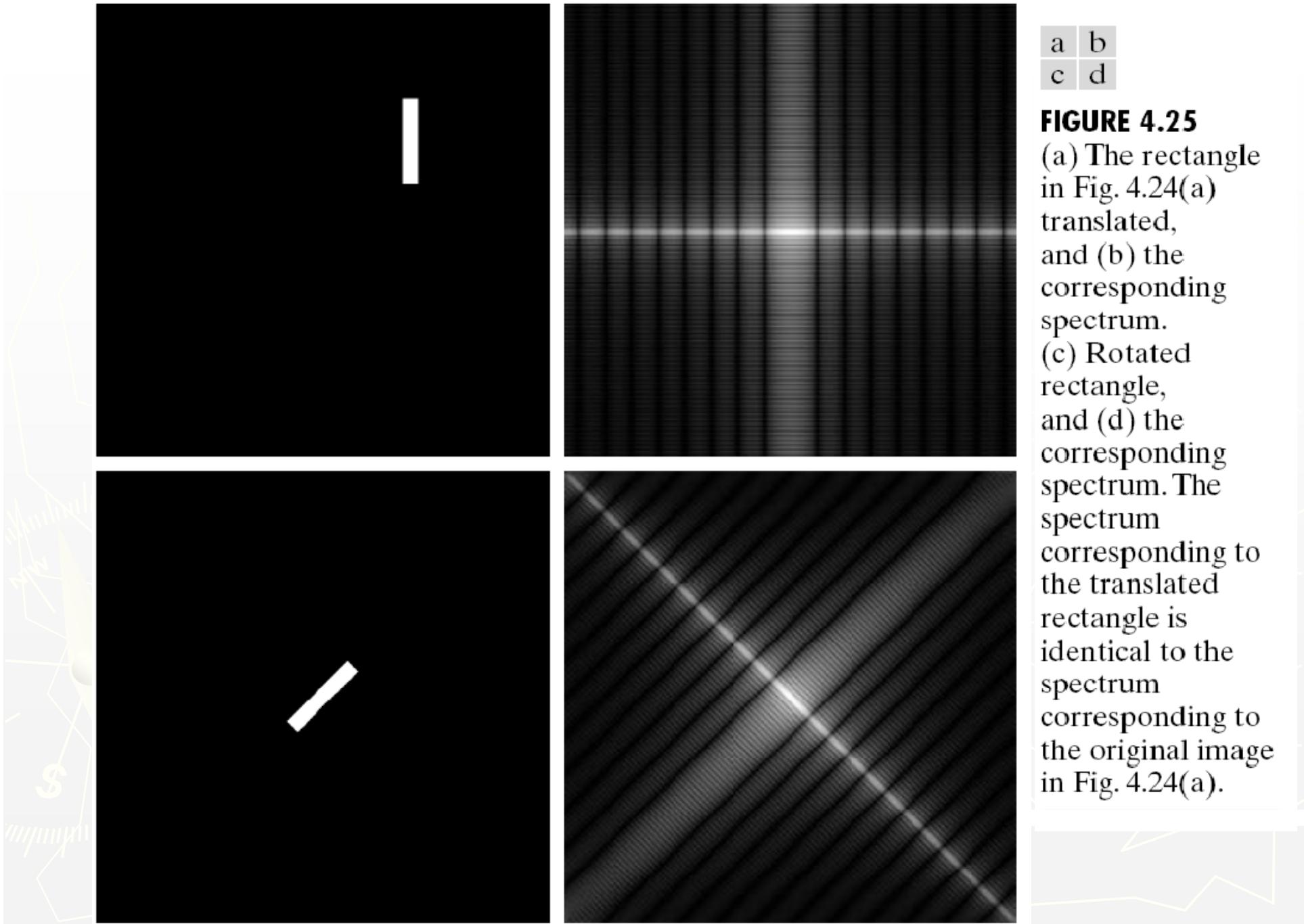
$$\phi(u, v) = \arctan \left[ \frac{I(u, v)}{R(u, v)} \right]$$



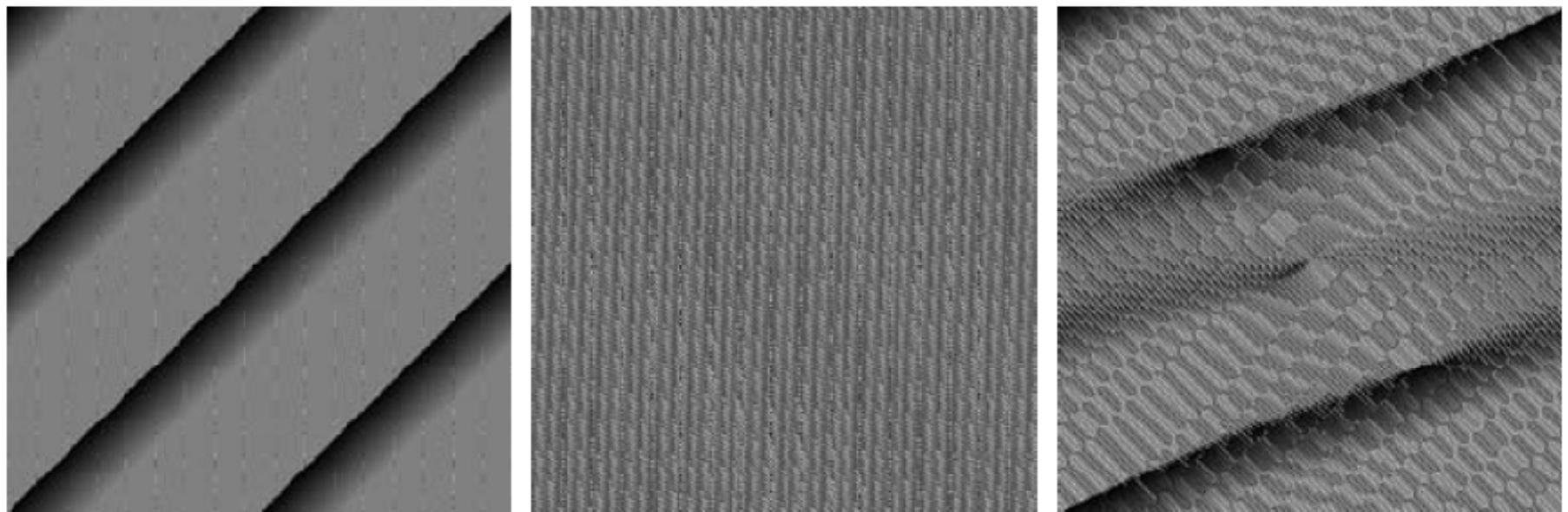
**FIGURE 4.24**

(a) Image.  
 (b) Spectrum  
 showing bright spots  
 in the four corners.  
 (c) Centered  
 spectrum. (d) Result  
 showing increased  
 detail after a log  
 transformation. The  
 zero crossings of the  
 spectrum are closer in  
 the vertical direction  
 because the rectangle  
 in (a) is longer in that  
 direction. The  
 coordinate  
 convention used  
 throughout the book  
 places the origin of  
 the spatial and  
 frequency domains at  
 the top left.



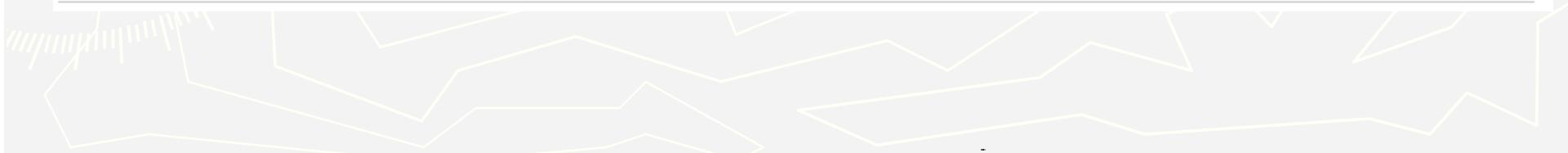


# Example: Phase Angles

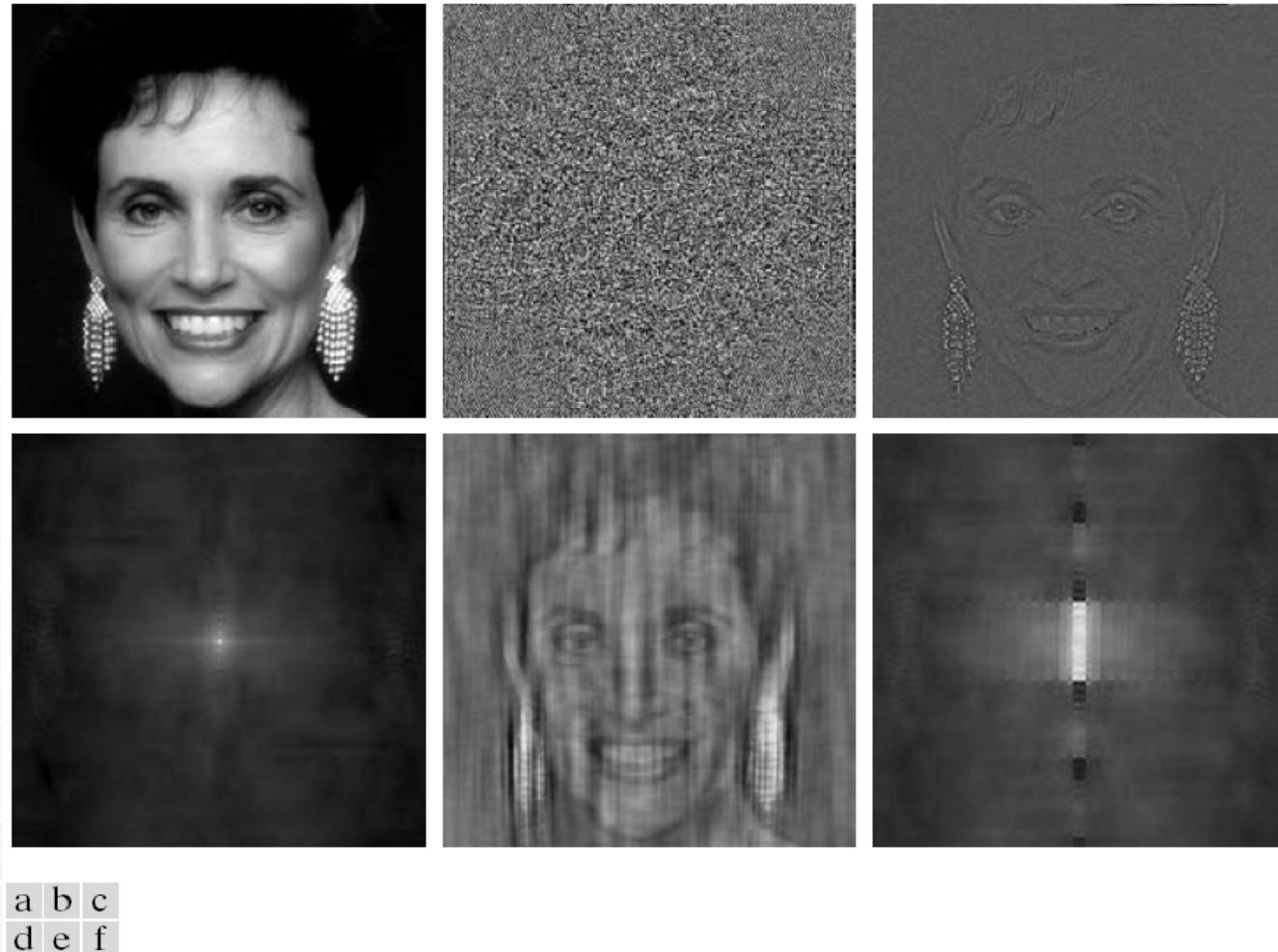


a b c

**FIGURE 4.26** Phase angle array corresponding (a) to the image of the centered rectangle in Fig. 4.24(a), (b) to the translated image in Fig. 4.25(a), and (c) to the rotated image in Fig. 4.25(c).



# Example: Phase Angles and The Reconstructed



**FIGURE 4.27** (a) Woman. (b) Phase angle. (c) Woman reconstructed using only the phase angle. (d) Woman reconstructed using only the spectrum. (e) Reconstruction using the phase angle corresponding to the woman and the spectrum corresponding to the rectangle in Fig. 4.24(a). (f) Reconstruction using the phase of the rectangle and the spectrum of the woman.

# Image Enhancement in the *Frequency Domain*

## Periodicity and the need for Padding:

- **Periodicity property of the DFT:** *The discrete Fourier Transform and the inverse Fourier transforms are periodic. Hence:*

$$F(u, v) = F(u + M, v) = F(u, v + N) = F(u + M, v + N)$$

$$f(x, y) = f(x + M, y) = f(x, y + N) = f(x + M, y + N)$$

- **Conjugate Symmetry property :** *The DFT is conjugate symmetric. The `\*' indicates the conjugate operation on a complex number.*

$$F(u, v) = F^*(-u, -v)$$

$$|F(u, v)| = |F(-u, -v)|$$

# Image Enhancement in the *Frequency Domain*

## Periodicity and the need for Padding:

a b  
c d

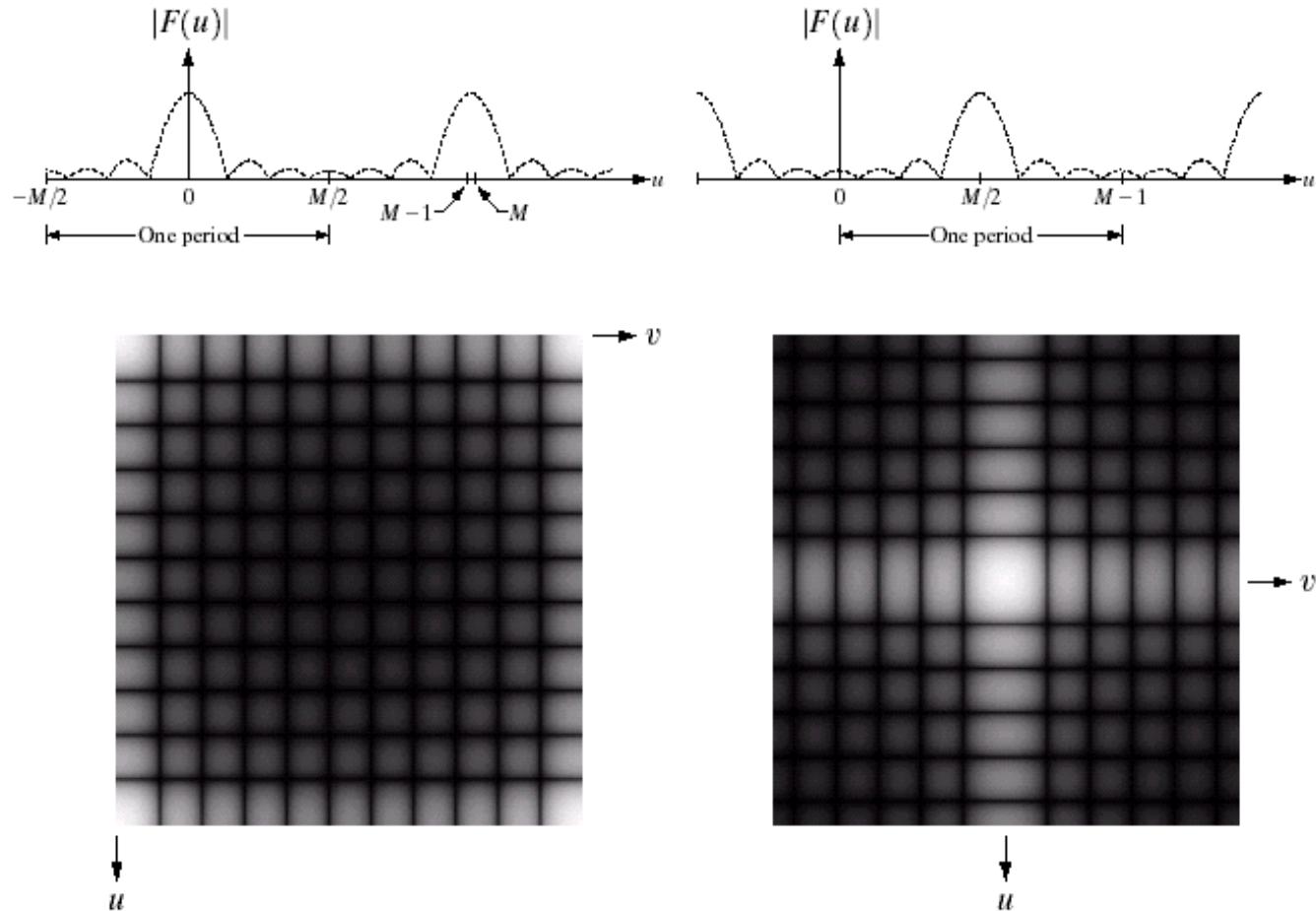
**FIGURE 4.34**

(a) Fourier spectrum showing back-to-back half periods in the interval  $[0, M - 1]$ .

(b) Shifted spectrum showing a full period in the same interval.

(c) Fourier spectrum of an image, showing the same back-to-back properties as (a), but in two dimensions.

(d) Centered Fourier spectrum.

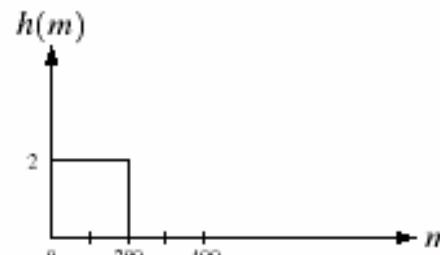


# Image Enhancement in the *Frequency Domain*

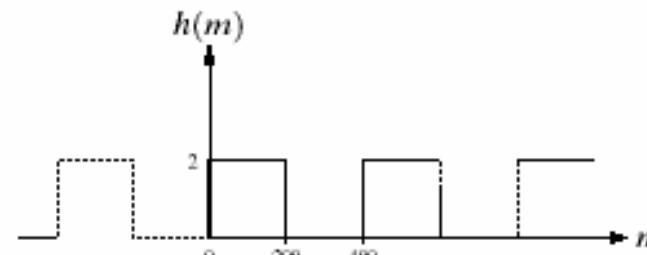
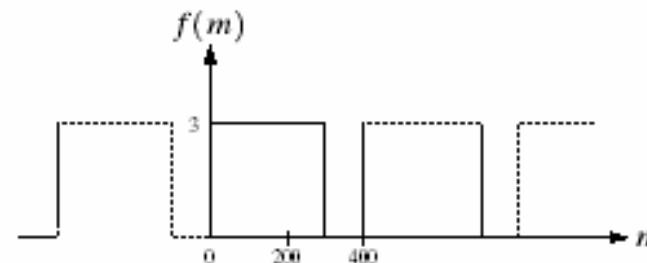
## Periodicity and the need for Padding:

- According to the convolution theorem, the multiplication in the frequency domain is the convolution in the spatial domain.
- Consider the 1D convolution of  $f(x)$  and  $h(x)$ ,

$$f(x) * h(x) = \frac{1}{M} \sum_{m=0}^{M-1} f(m)h(x-m)$$



*discrete functions*



*with periodicity property*

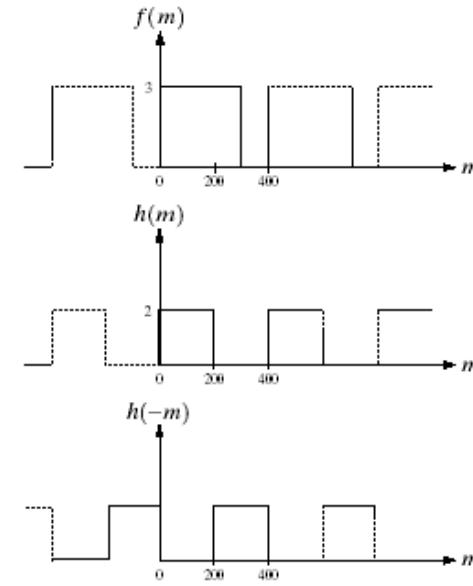
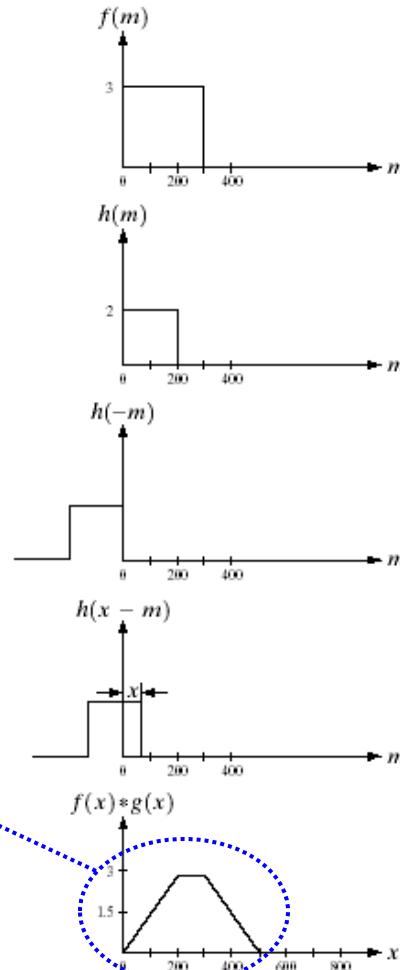
# Image Enhancement in the *Frequency Domain*

## Periodicity and the need for Padding:

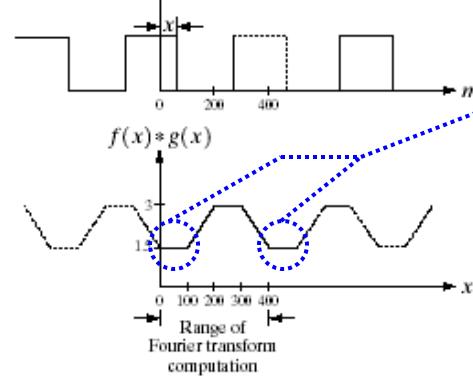
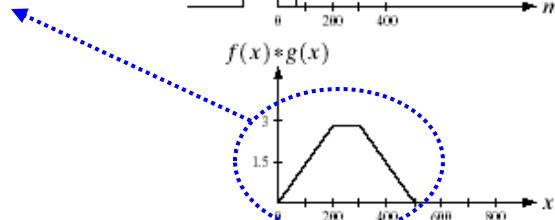
- *The illustration of the 1D convolution:*

a	f
b	g
c	h
d	i
e	j

**FIGURE 4.36** Left: convolution of two discrete functions. Right: convolution of the same functions, taking into account the implied periodicity of the DFT. Note in (j) how data from adjacent periods corrupt the result of convolution.



*Correct convolution*



*Wraparound error  
Due to periodicity*

# Image Enhancement in the *Frequency Domain*

## Periodicity and the need for Padding:

- **The solution of the wraparound error:** Given  $f$  and  $h$  consist of  $A$  and  $B$  points. Zeros are appended to both functions so that both of the functions have identical periods, denoted by  $P$ .
- The 1D extended/padded functions are given by:

$$f_e(x) = \begin{cases} f(x) & 0 \leq x \leq A-1 \\ 0 & A \leq x \leq P \end{cases}$$

$$h_e(x) = \begin{cases} h(x) & 0 \leq x \leq B-1 \\ 0 & B \leq x \leq P \end{cases}$$

$$P \geq A + B - 1$$

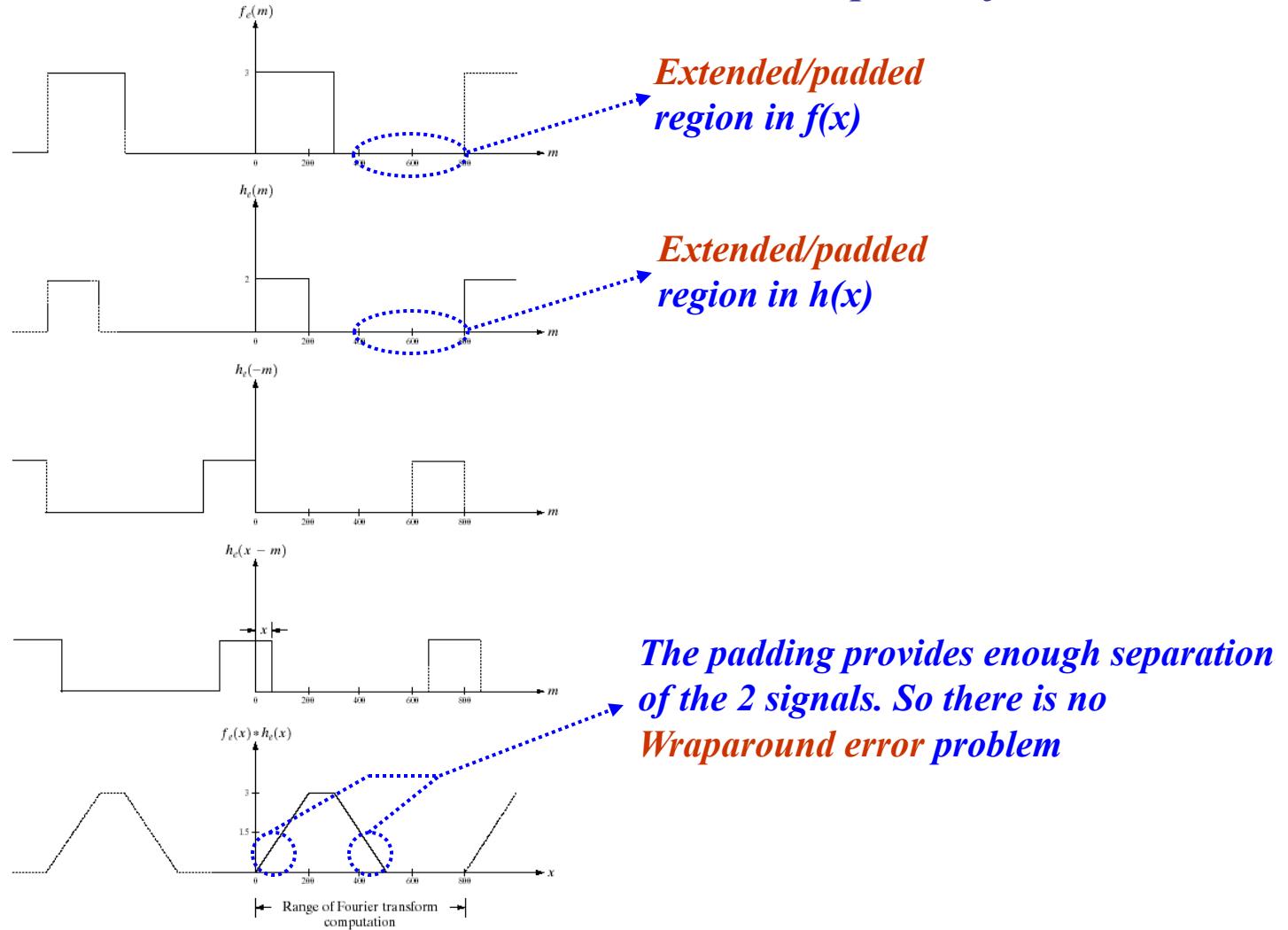
# Image Enhancement in the *Frequency Domain*

## Periodicity and the need for Padding:

- The solution of the wraparound error: *extended/padded functions.*

a  
b  
c  
d  
e

**FIGURE 4.37**  
Result of performing convolution with extended functions. Compare Figs. 4.37(e) and 4.36(e).



# Image Enhancement in the *Frequency Domain*

## Periodicity and the need for Padding:

- **The solution of the wraparound error:** *The 2D extended/padded functions  $f(x,y)$  and  $h(x,y)$  with sizes  $A \times B$  and  $C \times D$  respectively can be defined by.*

$$f_e(x, y) = \begin{cases} f(x, y) & 0 \leq x \leq A-1 \text{ and } 0 \leq y \leq B-1 \\ 0 & A \leq x \leq P \text{ or } B \leq y \leq Q \end{cases}$$

$$h_e(x, y) = \begin{cases} h(x, y) & 0 \leq x \leq C-1 \text{ and } 0 \leq y \leq D-1 \\ 0 & C \leq x \leq P \text{ or } D \leq y \leq Q \end{cases}$$

# Image Enhancement in the *Frequency Domain*

## Periodicity and the need for Padding:

- The solution of the wraparound error: *The 2D extended/padded functions  $f(x,y)$  and  $h(x,y)$  with sizes  $A \times B$  and  $C \times D$  respectively can be defined by.*

256x256 Image



Convolving Function



256x256 DFT Convolution



*Wraparound Error*

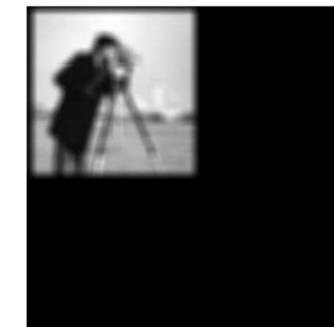
512x512 Zero padded



Convolving Function



512x512 DFT Convolution



*No Wraparound  
Error*

256x256 Top Left Corner

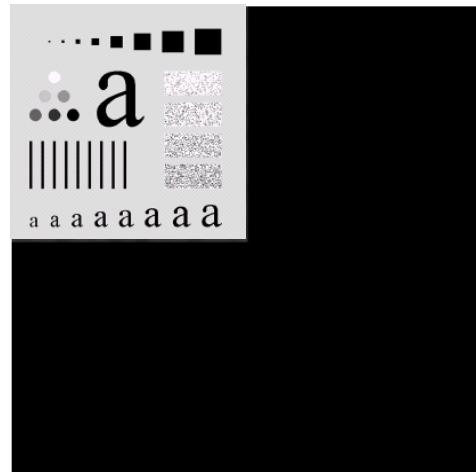


# Image Enhancement in the *Frequency Domain*

## Periodicity and the need for Padding:

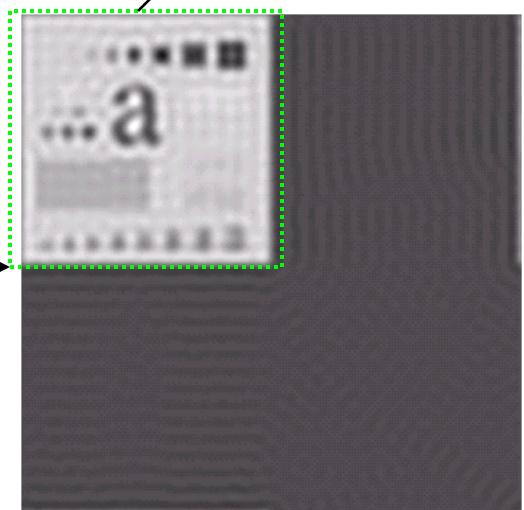
- The solution of the wraparound error:

Padded input  
image  $f(x,y)$

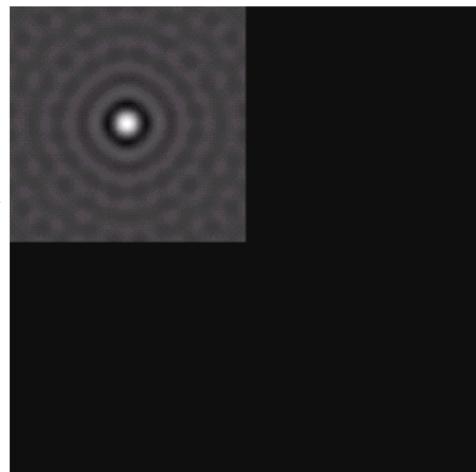


\* (Convolution)

Needs to be cropped for the final answer



Padded spatial  
Filter  $h(x,y)$



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

# **Image Enhancement in the *Frequency Domain***

## **Convolution and Correlation:**

- Convolution:** *Discrete convolution of two functions  $f(x,y)$  and  $h(x,y)$  with sizes  $M \times N$  is defined by.*

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

$$f(x, y) * h(x, y) \Leftrightarrow F(u, v)H(u, v)$$

$$f(x, y)h(x, y) \Leftrightarrow F(u, v) * H(u, v)$$

- Correlation:** *Discrete correlation of two functions  $f(x,y)$  and  $h(x,y)$  with sizes  $M \times N$  is defined by.*

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

$$f(x, y) \circ h(x, y) \Leftrightarrow F^*(u, v)H(u, v)$$

$$f^*(x, y)h(x, y) \Leftrightarrow F(u, v) \circ H(u, v)$$

---

# **Image Enhancement in the *Frequency Domain***

## **Convolution and Correlation:**

- **Convolution:** *The most important application of the convolution is the filtering in the spatial and frequency domains.*
- **Correlation:** *The principal application of the correlation is matching.*
- *In matching,  $f(x,y)$  is the image containing objects/regions and the  $h(x,y)$  is the object/reion that we are trying to locate.*
- *$h(x,y)$  is called the template.*
- *If there is a match the correlation of the two functions will be maximum at the location where the template  $h(x,y)$  finds the highest similarity in function  $f$ .*
- **Note:** *Image padding is also required in Correlation as well as in convolution.*

---

# **Image Enhancement in the *Frequency Domain***

## **Convolution and Correlation:**

- **Correlation:** *Cross correlation is the special term given to the correlation of two different images.*

- *In autocorrelation both images are identical, where:*

$$f(x, y) \circ f(x, y) \Leftrightarrow F^*(u, v)F(u, v) = |F(u, v)|^2$$

- *The spatial autocorrelation is identical to the power spectrum in the frequency domain.*

$$|f(x, y)|^2 \Leftrightarrow F(u, v) \circ F(u, v)$$

# Image Enhancement in the *Frequency Domain*

## Convolution and Correlation:

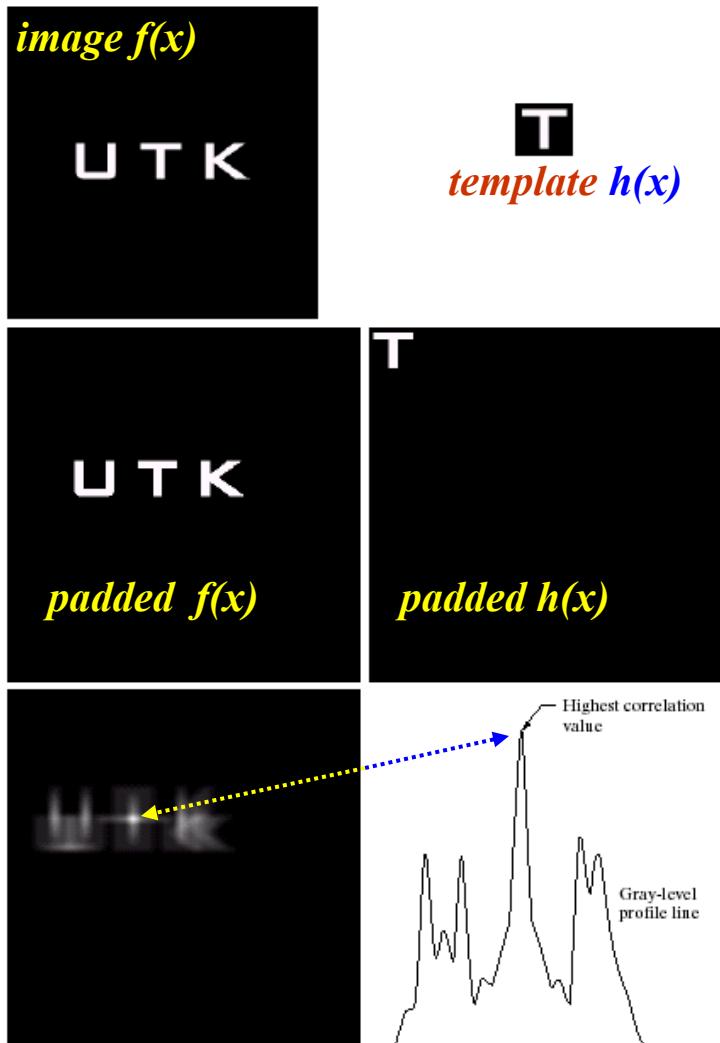


FIGURE 4.41  
(a) Image.  
(b) Template.  
(c) and  
(d) Padded  
images.  
(e) Correlation  
function displayed  
as an image.  
(f) Horizontal  
profile line  
through the  
highest value in  
(e), showing the  
point at which the  
best match took  
place.

• **Correlation:** *The cross correlation of a template  $h(x,y)$  with an input image  $f(x,y)$ . This process is also known as the template matching.*

- **Note:** *The padded images are transformed into the Frequency Domain with DFT.*
- **Complex conjugate of one of the images is multiplied with the other.**
- **The resulting function is inverse transformed.**

# Summary

Name	Expression(s)
1) Discrete Fourier transform (DFT) of $f(x, y)$	$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M+vy/N)}$
2) Inverse discrete Fourier transform (IDFT) of $F(u, v)$	$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M+vy/N)}$
3) Polar representation	$F(u, v) =  F(u, v)  e^{j\phi(u, v)}$
4) Spectrum	$ F(u, v)  = [R^2(u, v) + I^2(u, v)]^{1/2}$ $R = \text{Real}(F); \quad I = \text{Imag}(F)$
5) Phase angle	$\phi(u, v) = \tan^{-1} \left[ \frac{I(u, v)}{R(u, v)} \right]$
6) Power spectrum	$P(u, v) =  F(u, v) ^2$
7) Average value	$\bar{f}(x, y) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) = \frac{1}{MN} F(0, 0)$

(Continued)

# Summary

Name	Expression(s)
8) Periodicity ( $k_1$ and $k_2$ are integers)	$F(u, v) = F(u + k_1M, v) = F(u, v + k_2N)$ $= F(u + k_1M, v + k_2N)$ $f(x, y) = f(x + k_1M, y) = f(x, y + k_2N)$ $= f(x + k_1M, y + k_2N)$
9) Convolution	$f(x, y) \star h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n)h(x - m, y - n)$
10) Correlation	$f(x, y) \star\! h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m, n)h(x + m, y + n)$
11) Separability	The 2-D DFT can be computed by computing 1-D DFT transforms along the rows (columns) of the image, followed by 1-D transforms along the columns (rows) of the result. See Section 4.11.1.
12) Obtaining the inverse Fourier transform using a forward transform algorithm.	$MNf^*(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F^*(u, v)e^{-j2\pi(ux/M+vy/N)}$ <p>This equation indicates that inputting <math>F^*(u, v)</math> into an algorithm that computes the forward transform (right side of above equation) yields <math>MNf^*(x, y)</math>. Taking the complex conjugate and dividing by <math>MN</math> gives the desired inverse. See Section 4.11.2.</p>

# Summary

Name	DFT Pairs
1) Symmetry properties	See Table 4.1
2) Linearity	$af_1(x, y) + bf_2(x, y) \Leftrightarrow aF_1(u, v) + bF_2(u, v)$
3) Translation (general)	$f(x, y)e^{j2\pi(u_0x/M+v_0y/N)} \Leftrightarrow F(u - u_0, v - v_0)$ $f(x - x_0, y - y_0) \Leftrightarrow F(u, v)e^{-j2\pi(ux_0/M+vy_0/N)}$
4) Translation to center of the frequency rectangle, $(M/2, N/2)$	$f(x, y)(-1)^{x+y} \Leftrightarrow F(u - M/2, v - N/2)$ $f(x - M/2, y - N/2) \Leftrightarrow F(u, v)(-1)^{u+v}$
5) Rotation	$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0)$ $x = r \cos \theta \quad y = r \sin \theta \quad u = \omega \cos \varphi \quad v = \omega \sin \varphi$
6) Convolution theorem <sup>†</sup>	$f(x, y) \star h(x, y) \Leftrightarrow F(u, v)H(u, v)$ $f(x, y)h(x, y) \Leftrightarrow F(u, v) \star H(u, v)$

(Continued)

# Summary

Name	DFT Pairs
7) Correlation theorem <sup>†</sup>	$f(x, y) \star h(x, y) \Leftrightarrow F^*(u, v) H(u, v)$ $f^*(x, y) h(x, y) \Leftrightarrow F(u, v) \star H(u, v)$
8) Discrete unit impulse	$\delta(x, y) \Leftrightarrow 1$
9) Rectangle	$\text{rect}[a, b] \Leftrightarrow ab \frac{\sin(\pi ua)}{(\pi ua)} \frac{\sin(\pi vb)}{(\pi vb)} e^{-j\pi(ua+vb)}$
10) Sine	$\sin(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $j \frac{1}{2} [\delta(u + Mu_0, v + Nv_0) - \delta(u - Mu_0, v - Nv_0)]$
11) Cosine	$\cos(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $\frac{1}{2} [\delta(u + Mu_0, v + Nv_0) + \delta(u - Mu_0, v - Nv_0)]$
The following Fourier transform pairs are derivable only for continuous variables, denoted as before by $t$ and $z$ for spatial variables and by $\mu$ and $\nu$ for frequency variables. These results can be used for DFT work by sampling the continuous forms.	
12) <i>Differentiation</i> (The expressions on the right assume that $f(\pm\infty, \pm\infty) = 0$ .)	$\left(\frac{\partial}{\partial t}\right)^m \left(\frac{\partial}{\partial z}\right)^n f(t, z) \Leftrightarrow (j2\pi\mu)^m (j2\pi\nu)^n F(\mu, \nu)$ $\frac{\partial^m f(t, z)}{\partial t^m} \Leftrightarrow (j2\pi\mu)^m F(\mu, \nu); \frac{\partial^n f(t, z)}{\partial z^n} \Leftrightarrow (j2\pi\nu)^n F(\mu, \nu)$
13) <i>Gaussian</i>	$A 2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)} \Leftrightarrow A e^{-(\mu^2+\nu^2)/2\sigma^2}$ ( $A$ is a constant)

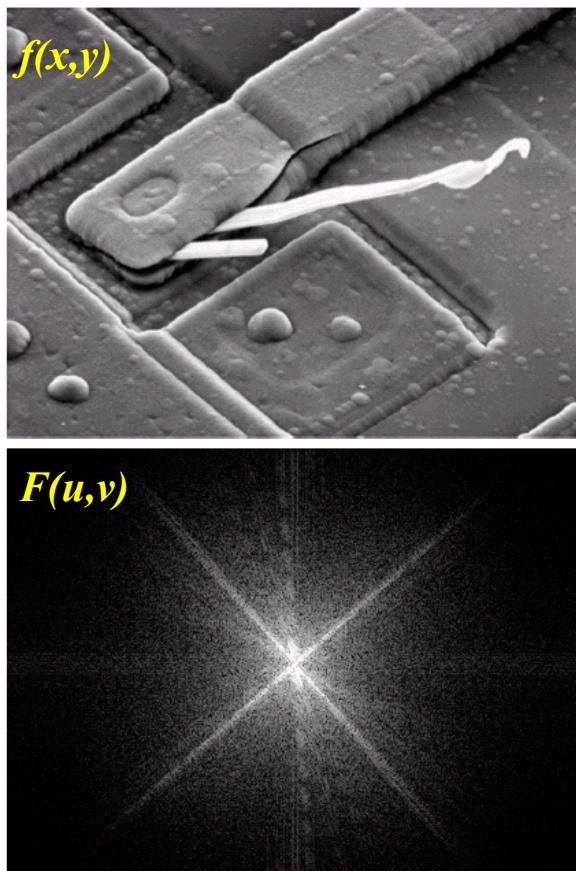
<sup>†</sup>Assumes that the functions have been extended by zero padding. Convolution and correlation are associative, commutative, and distributive.

# Image Enhancement in the *Frequency Domain*

## Filtering in the Frequency Domain

### •*Some Basic Properties of the Frequency Domain:*

- Frequency is directly related to the rate of change. Therefore, slowest varying component ( $u=v=0$ ) corresponds to the average intensity level of the image. Corresponds to the origin of the Fourier Spectrum.*



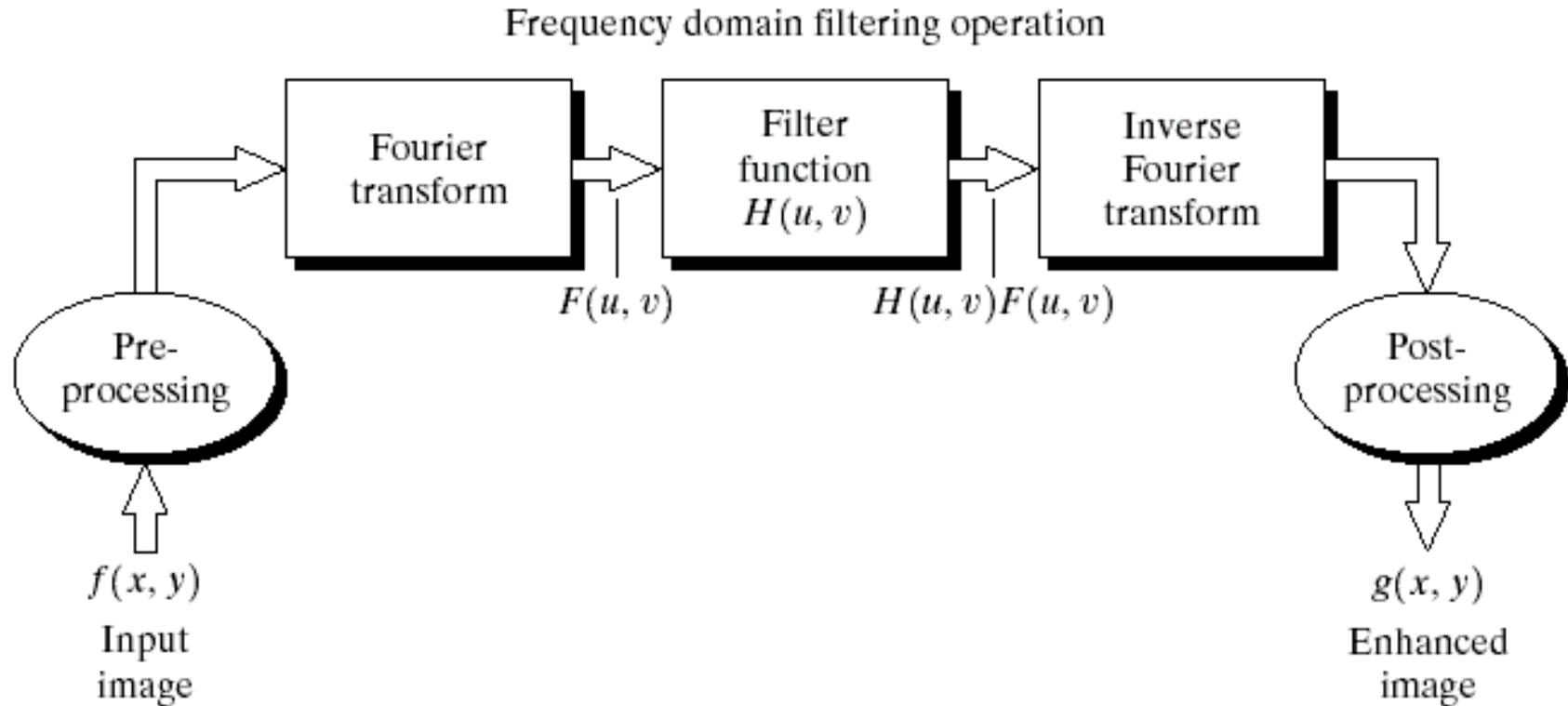
**FIGURE 4.4**  
(a) SEM image of a damaged integrated circuit.  
(b) Fourier spectrum of (a). (Original image courtesy of Dr. J. M. Hudak, Brockhouse Institute for Materials Research, McMaster University, Hamilton, Ontario, Canada.)

- Higher frequencies corresponds to the faster varying intensity level changes in the image. The edges of objects or the other components characterized by the abrupt changes in the intensity level corresponds to higher frequencies.*

# Image Enhancement in the *Frequency Domain*

## Filtering in the Frequency Domain

- *Basic Steps for Filtering in the Frequency Domain:*



**FIGURE 4.5** Basic steps for filtering in the frequency domain.

---

# **Image Enhancement in the *Frequency Domain***

## **Filtering in the Frequency Domain**

- Basic Steps for Filtering in the Frequency Domain:***

1. *Multiply the input image by  $(-1)^{x+y}$  to center the transform.*
2. *Compute  $F(u,v)$ , the DFT of the image from (1).*
3. *Multiply  $F(u,v)$  by a filter function  $H(u,v)$ .*
4. *Compute the inverse DFT of the result in (3).*
5. *Obtain the real part of the result in (4).*
6. *Multiply the result in (5) by  $(-1)^{x+y}$ .*

*Given the filter  $H(u,v)$  (filter transfer function) in the frequency domain, the Fourier transform of the output image (filtered image) is given by:*

$$G(u,v) = H(u,v)F(u,v) \quad \text{Step (3)}$$

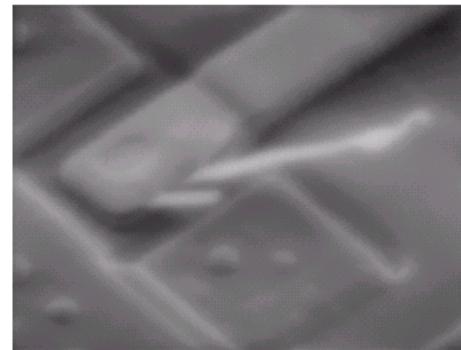
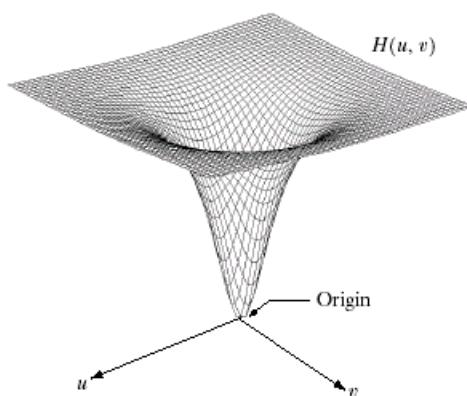
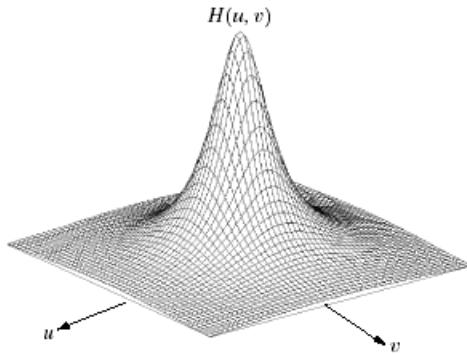
*The filtered image  $g(x,y)$  is simply the inverse Fourier transform of  $G(u,v)$ .*

$$g(x,y) = \mathfrak{I}^{-1}[G(u,v)] \quad \text{Step (4)}$$

# Image Enhancement in the *Frequency Domain*

## Filtering in the Frequency Domain

- *Basics of Low Pass Filters in the Frequency Domain:*



- **lowpass filter:** A filter that attenuates high frequencies while **passing the low frequencies.**

- **Low frequencies represent the gray-level appearance of an image over smooth areas.**

- **highpass filter:** A filter that attenuates low frequencies while **passing the high frequencies.**

- **High frequencies represents the details such as edges and noise.**

a  
b  
c  
d

**FIGURE 4.7** (a) A two-dimensional lowpass filter function. (b) Result of lowpass filtering the image in Fig. 4.4(a).  
(c) A two-dimensional highpass filter function. (d) Result of highpass filtering the image in Fig. 4.4(a).

# **Image Enhancement in the *Frequency Domain***

## **Filtering in the Frequency Domain**

- Consider the following filter transfer function:

$$H(u, v) = \begin{cases} 0 & \text{if } (u, v) = (M/2, N/2) \\ 1 & \text{otherwise} \end{cases}$$

- This filter will set  $F(0,0)$  to zero and leave all the other frequency components. Such a filter is called the **notch filter**, since it is constant function with a hole (notch) at the origin.

**FIGURE 4.6**

Result of filtering the image in Fig. 4.4(a) with a notch filter that set to 0 the  $F(0, 0)$  term in the Fourier transform.



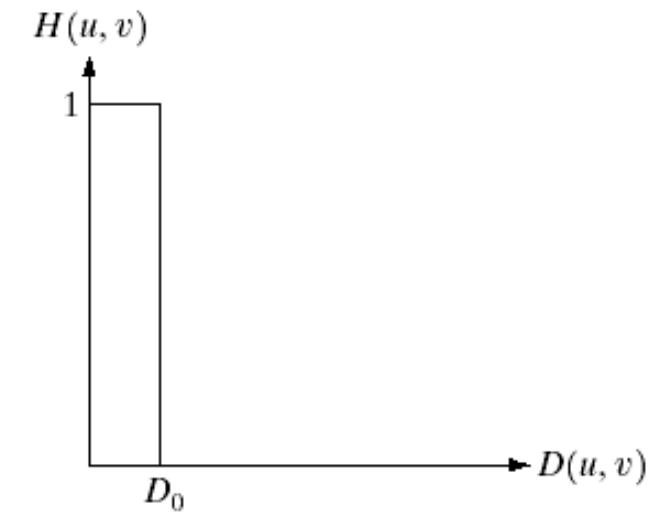
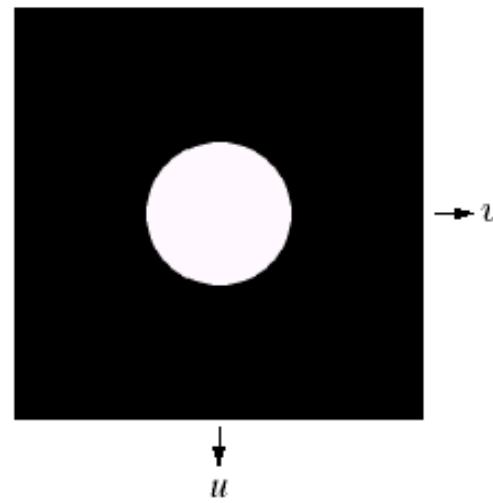
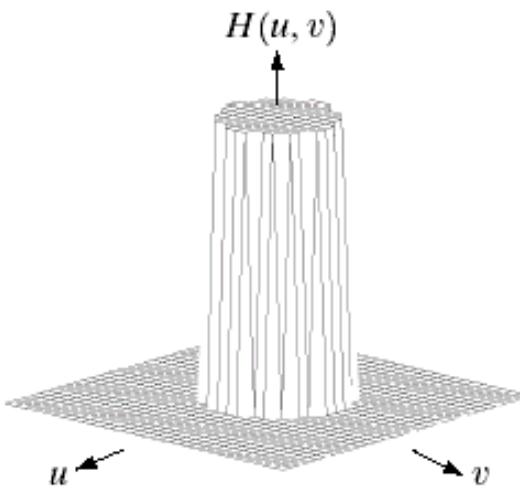
# Image Enhancement in the *Frequency Domain*

## Filtering in the Frequency Domain

- *Smoothing Frequency Domain Filters: Ideal Lowpass Filters*

$$H(u, v) = \begin{cases} 1 & \text{if } D(u, v) \leq D_0 \\ 0 & \text{if } D(u, v) > D_0 \end{cases}$$

*D(u,v) is the distance from the origin  
D<sub>0</sub> is the cutoff frequency*



a b c

**FIGURE 4.10** (a) Perspective plot of an ideal lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.

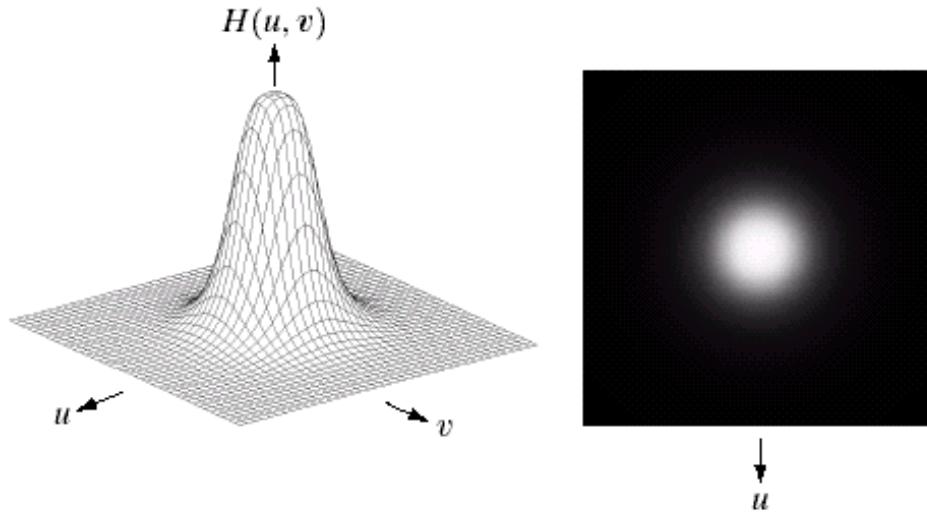
# Image Enhancement in the *Frequency Domain*

## Filtering in the Frequency Domain

- *Smoothing Frequency Domain Filters: Butterworth Lowpass Filters*

$$H(u, v) = \frac{1}{1 + [D(u, v)/D_0]^{2n}}$$

$D(u, v)$  is the distance from the origin  
 $D_0$  is the cutoff frequency.  
 $n$  is the order of the filter



a b c

**FIGURE 4.14** (a) Perspective plot of a Butterworth lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections of orders 1 through 4.

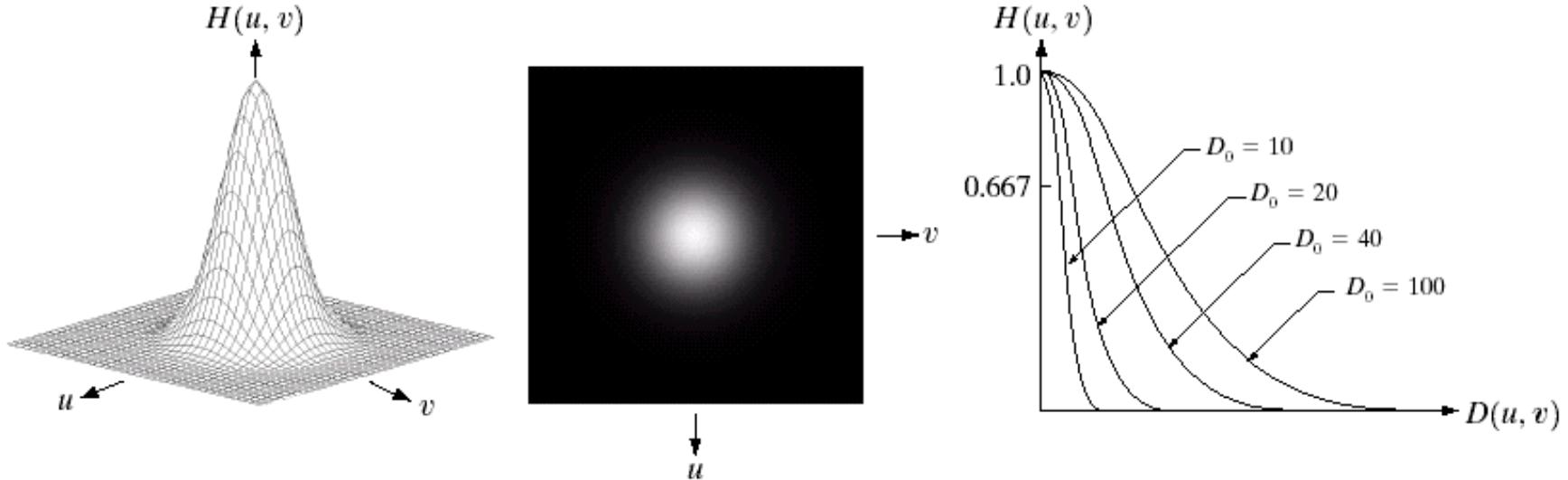
# Image Enhancement in the *Frequency Domain*

## Filtering in the Frequency Domain

- *Smoothing Frequency Domain Filters: Gaussian Lowpass Filters*

$$H(u, v) = e^{-D^2(u,v)/2D_0^2}$$

$D(u, v)$  is the distance from the origin  
 $D_0$  is the cutoff frequency.



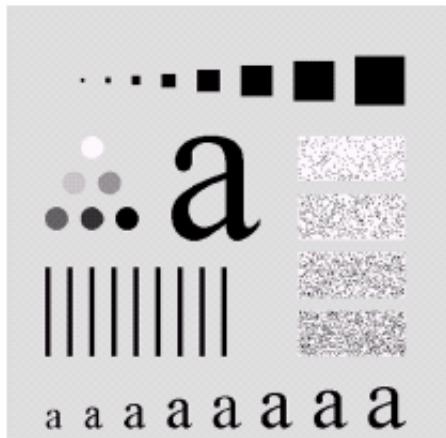
a b c

**FIGURE 4.17** (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of  $D_0$ .

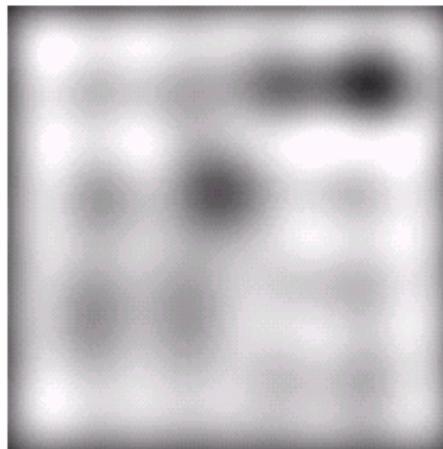
# **Image Enhancement in the *Frequency Domain***

## **Filtering in the Frequency Domain**

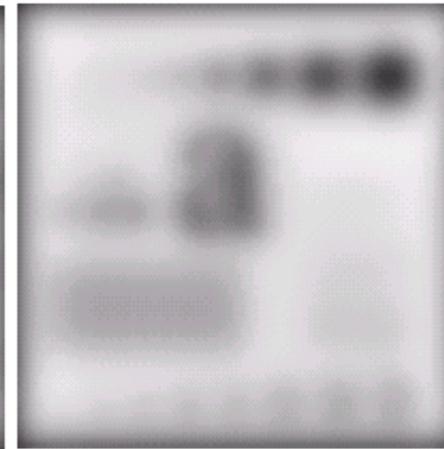
- *Comparison of Ideal, Butterworth and Gaussian Lowpass Filters having the same radii (cutoff frequency) value.*



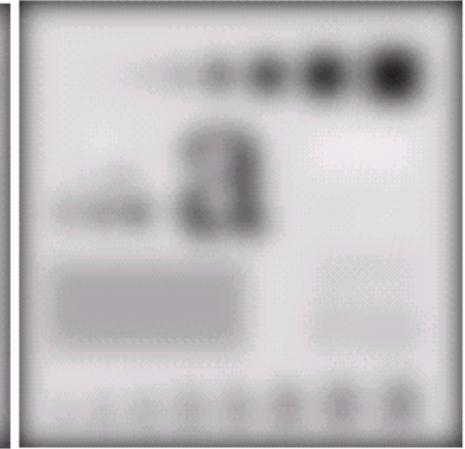
*Original image*



*Ideal LP Filtered*



*Butterworth LP Filtered*



*Gaussian LP Filtered*

---

# **Image Enhancement in the *Frequency Domain***

## **Smoothing Frequency-Domain Filters**

- *The high-frequency components are: edges and sharp transitions such as noise.*
- *Smoothing/blurring can be achieved by attenuating a specified range of high-frequency components in the frequency domain.*
- *Given the Fourier transformed image  $F(u)$ , the filtered image  $G(u)$  can be obtained by:*

$$G(u, v) = H(u, v)F(u, v)$$

- *Where  $H(u)$  is the filter transfer function.*
- *Smoothing can be achieved by lowpass filters. We will consider only 3 types of lowpass filters:*
  - *Ideal Lowpass filters,*
  - *Butterworth Lowpass filters,*
  - *Gaussian Lowpass filters*

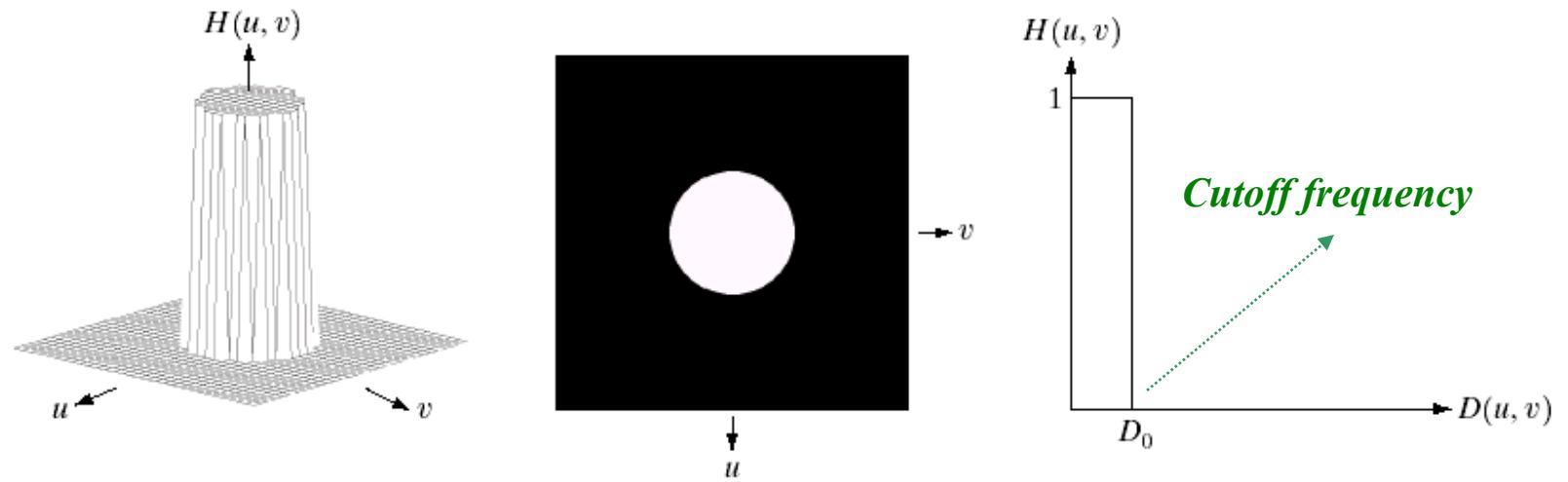
# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

- **Ideal Lowpass Filter (ILPF):** *Simply cuts off all the high frequencies higher than the specified cutoff frequency. The filter transfer function:*

$$H(u, v) = \begin{cases} 1 & \text{if } D(u, v) \leq D_0 \\ 0 & \text{if } D(u, v) > D_0 \end{cases}$$

$D(u, v)$  is the distance from the origin  
 $D_0$  is the cutoff frequency



a b c

**FIGURE 4.10** (a) Perspective plot of an ideal lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.

---

# **Image Enhancement in the *Frequency Domain***

## **Smoothing Frequency-Domain Filters**

- **Cutoff Frequency of an Ideal Lowpass Filter:** *One way of specifying the cutoff frequency is to compute circles enclosing specified amounts of total image power.*
- **Calculating  $P_T$  which is the total power of the transformed image:**

$$P_T = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} P(u, v) \quad u = 0, 1, \dots, M - 1, v = 0, 1, \dots, N - 1$$

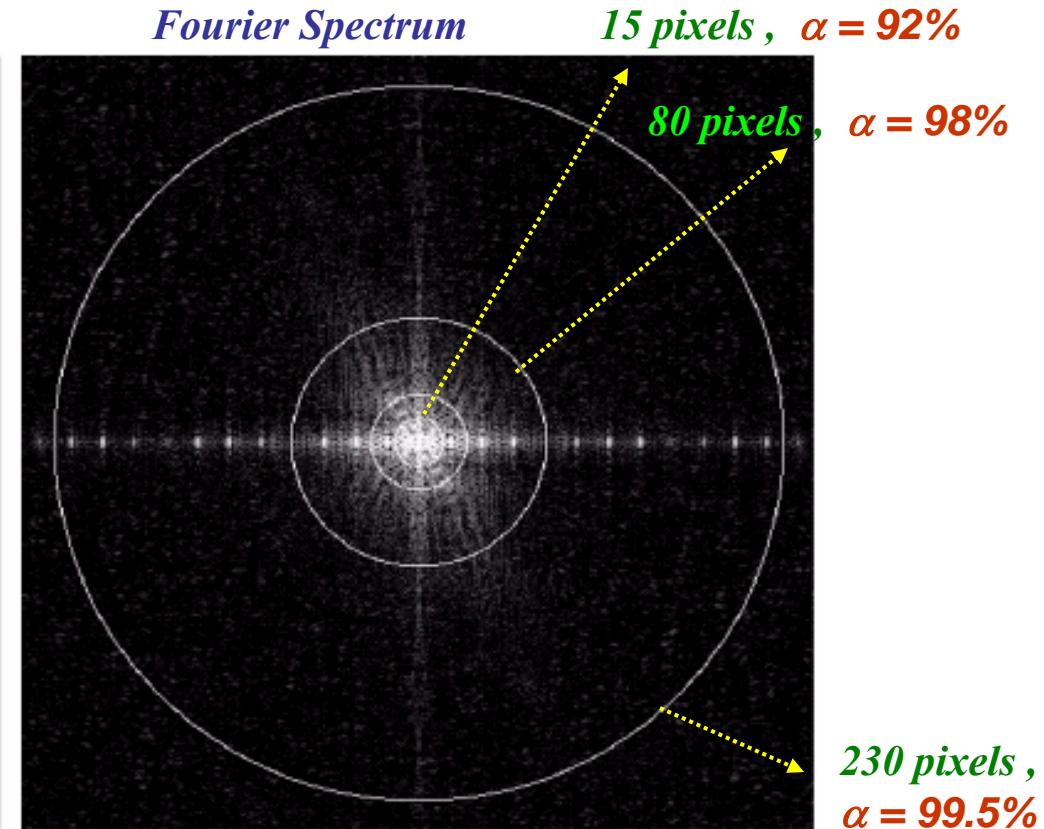
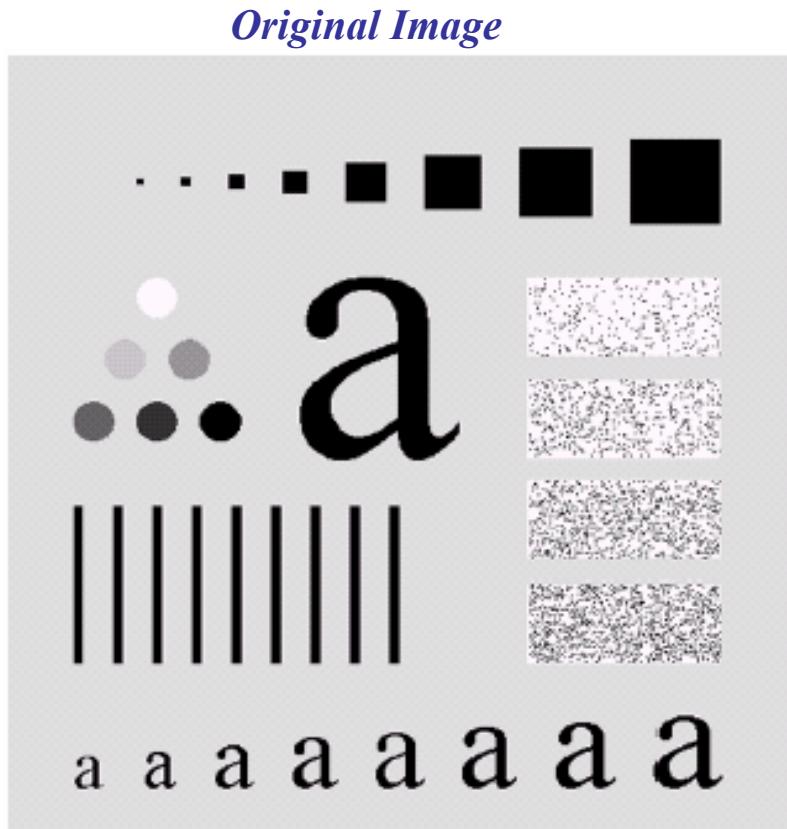
- *The cutoff frequency  $D_o$  can be determined by specifying the  $\alpha$  percent of the total power enclosed by a circle centered at the origin. The radius of the circle is the cutoff frequency  $D_o$ .*

$$\alpha = 100 \left\lfloor \sum_u \sum_v P(u, v) / P_T \right\rfloor \quad u \leq \text{radius}(D_o), v \leq \text{radius}(D_o)$$

# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

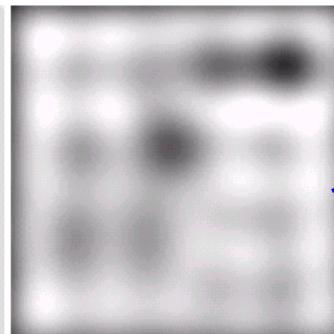
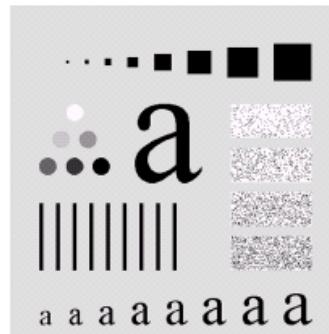
- Cutoff Frequency of an Ideal Lowpass Filter:



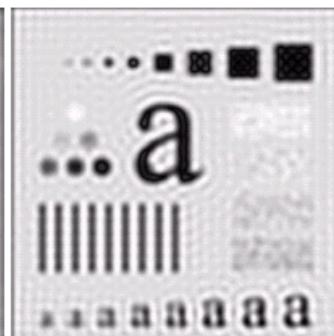
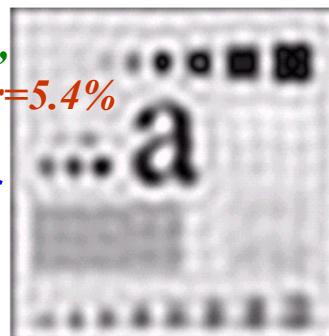
# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

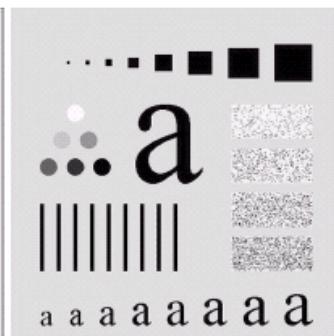
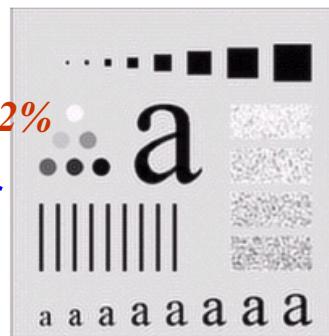
- Cutoff Frequency of an Ideal Lowpass Filter:



ILPF with cutoff=5, removed power=8%  
Blurring effect is the result of LPIF



ILPF with cutoff=30, removed power=3.6%  
Ringing effect is the problem of LPIF



ILPF with cutoff=230, removed power=0.5%

---

# **Image Enhancement in the *Frequency Domain***

## **Smoothing Frequency-Domain Filters**

- **Blurring and Ringing properties of ILPF:**
- *The blurring and ringing properties of the ILPF can be explained by the help of the convolution theorem:*
- *Given the Fourier transformed input image  $F(u)$  and output image  $G(u)$  and the filter transfer function  $H(u)$ ,*

$$G(u, v) = H(u, v)F(u, v)$$

- *The corresponding process in the spatial domain with regard to the convolution theorem is given by:*

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

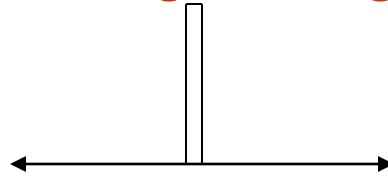
- *$h(x, y)$  in the spatial domain, is the inverse Fourier transform of the filter transfer function  $H(u, v)$ .*

- *The spatial filter  $h(x, y)$  has two major characteristics:*
  - *dominant component at the origin*
  - *Concentric circular components about center component.*

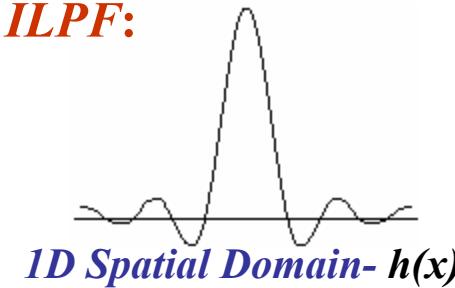
# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

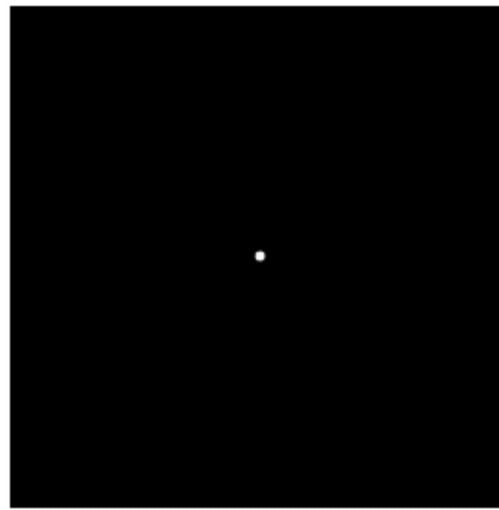
- Blurring and Ringing properties of *ILPF*:



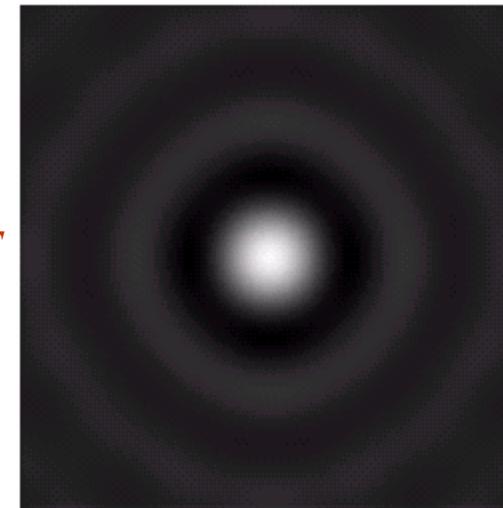
1D Frequency Domain-  $H(u)$



1D Spatial Domain-  $h(x)$



2D Frequency Domain-  $H(u,v)$



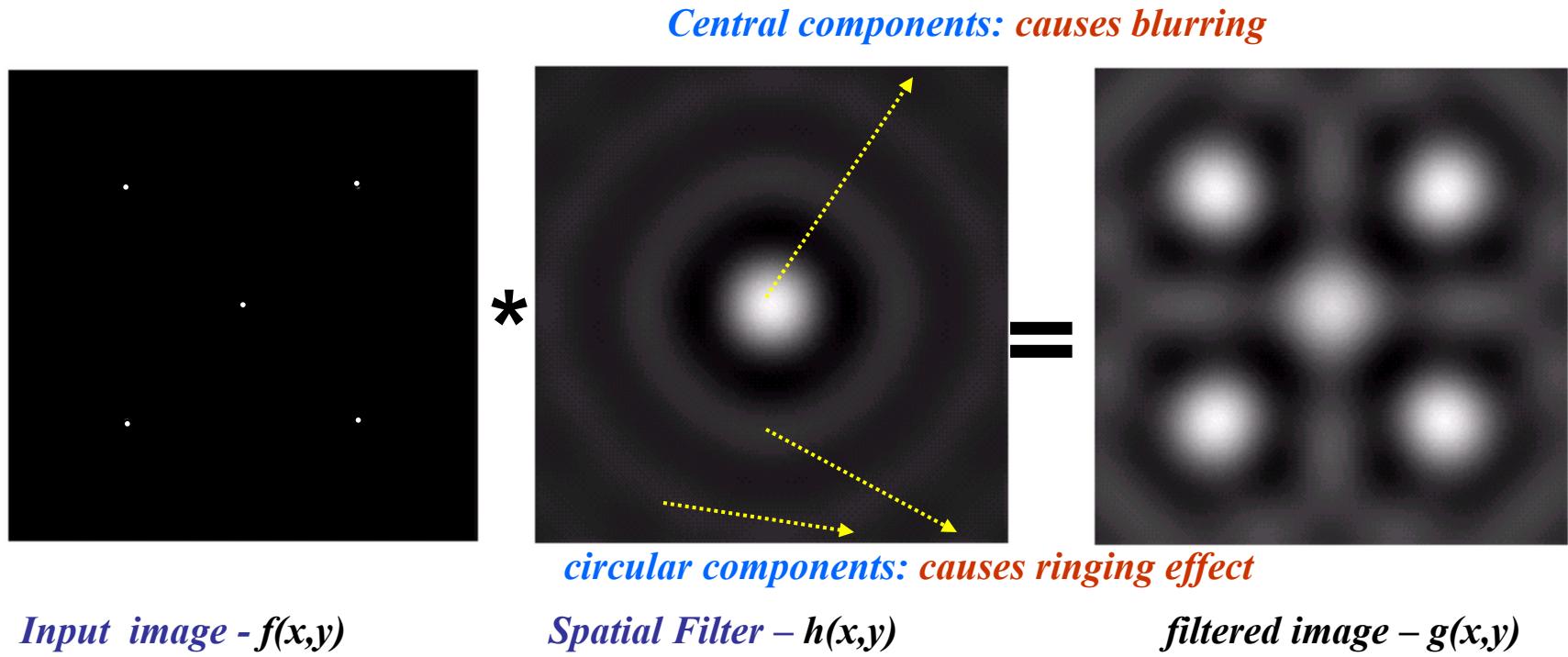
2D Spatial Domain –  $h(x,y)$

- The spatial domain filters *center component* is responsible for blurring.
- The *circular components* are responsible for the *ringing artifacts*.

# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

- Blurring and Ringing properties of ILPF: Lets consider the following convolution in the spatial domain:



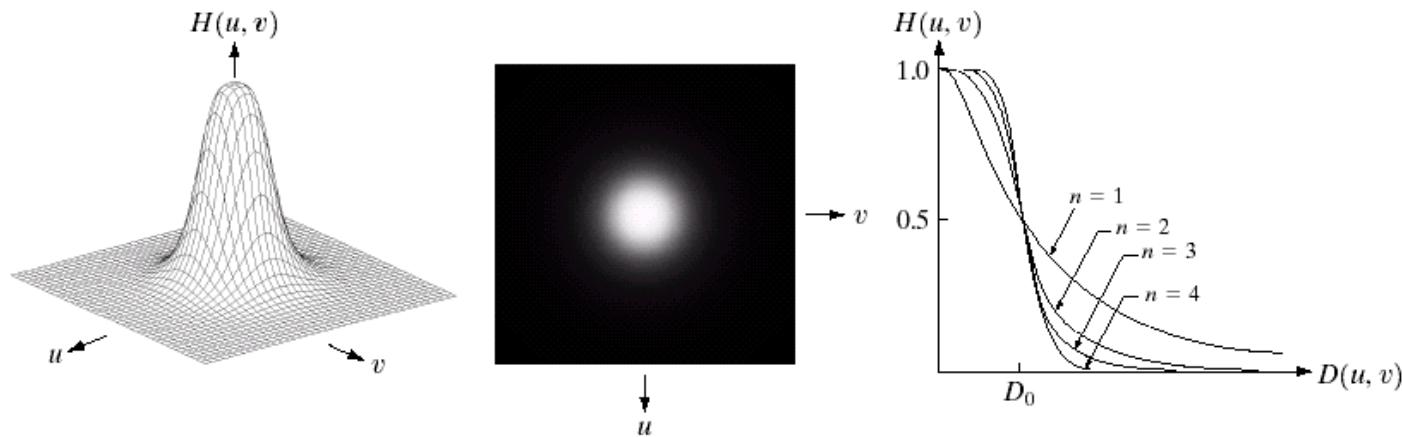
# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

- **Butterworth Lowpass Filter (BLPF):** *The transfer function of BLPF of order n and with a specified cutoff frequency is denoted by the following filter transfer function:*

$$H(u, v) = \frac{1}{1 + [D(u, v)/D_0]^{2n}}$$

$D(u, v)$  is the distance from the origin  
 $D_0$  is the cutoff frequency.  
 $n$  is the order of the filter



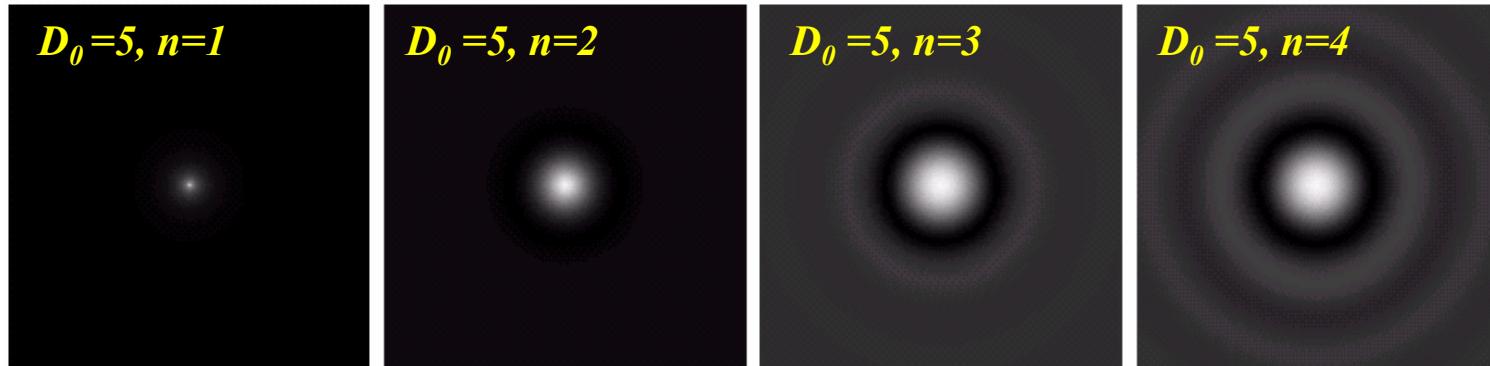
a b c

**FIGURE 4.14** (a) Perspective plot of a Butterworth lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections of orders 1 through 4.

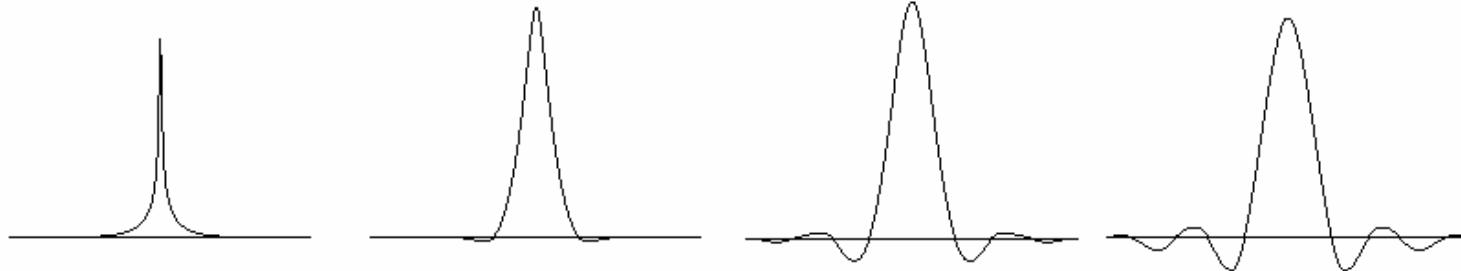
# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

- **Butterworth Lowpass Filter (BLPF):**
- *The BLPF with order of 1 does not have any ringing artifact.*
- *BLPF with orders 2 or more shows increasing ringing effects as the order increases.*



*2D - Spatial domain representation of the BLPF.*



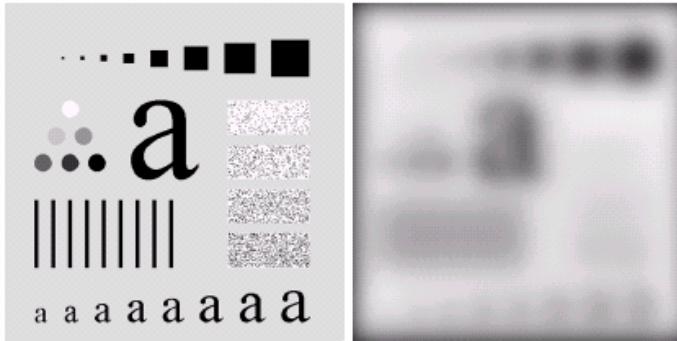
*1D - Profile through the centre of the Spatial domain filters.*

# Image Enhancement in the *Frequency Domain*

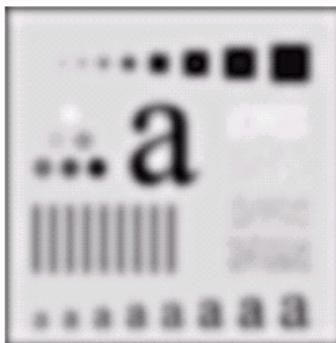
## Smoothing Frequency-Domain Filters

- Butterworth Lowpass Filter (BLPF):

Original Image



cutoff=15,  
order=2

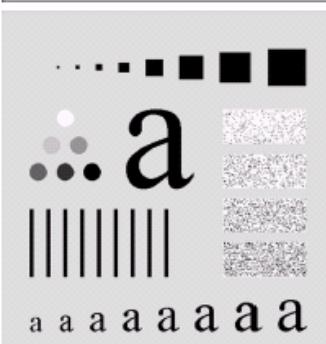
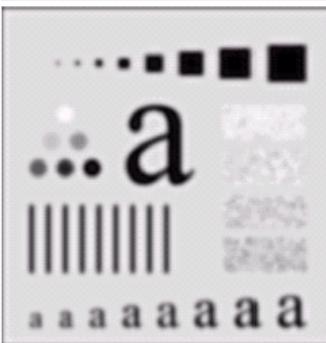
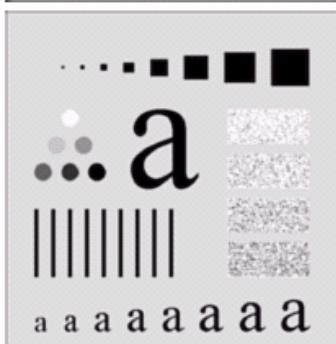


BLPF with cutoff=5, order=2

BLPF with cutoff=30, order=2

BLPF with cutoff=230, order=2

cutoff=80,  
order=2



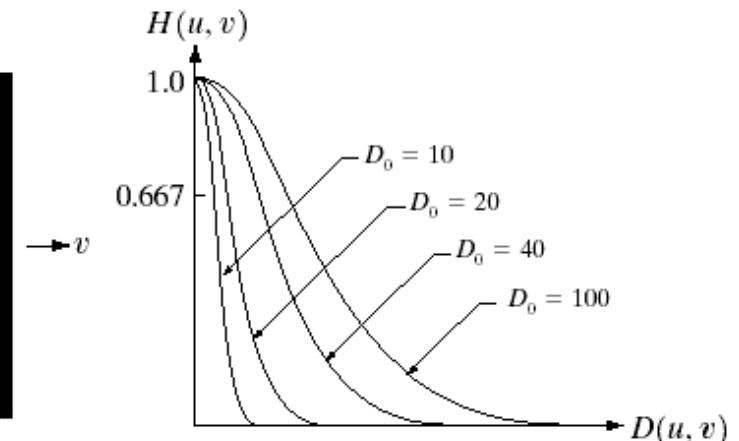
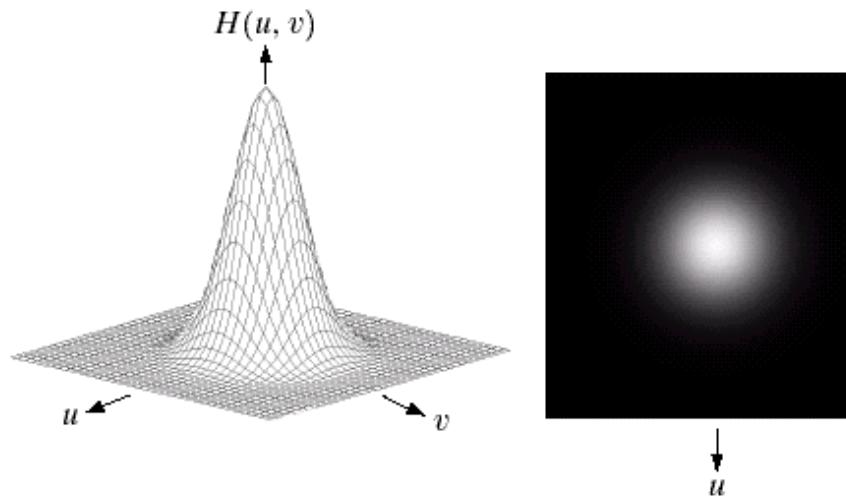
# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

- **Gaussian Lowpass Filter (GLPF):** *The transfer function of GLPF is given as follows:*

$$H(u, v) = e^{-D^2(u, v)/2D_0^2}$$

$D(u, v)$  is the distance from the origin  
 $D_0$  is the cutoff frequency.



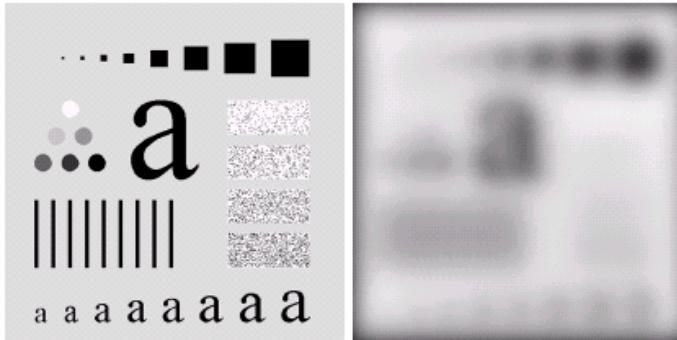
a b c

**FIGURE 4.17** (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of  $D_0$ .

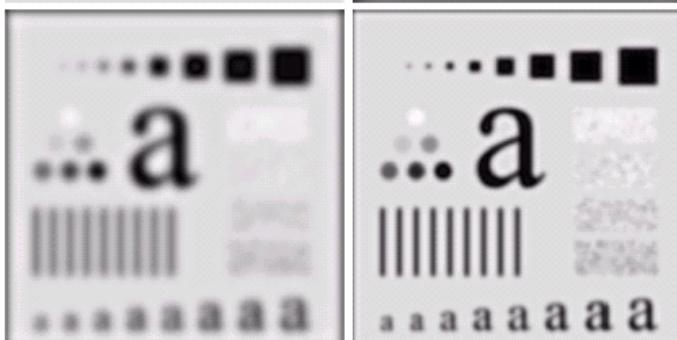
# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

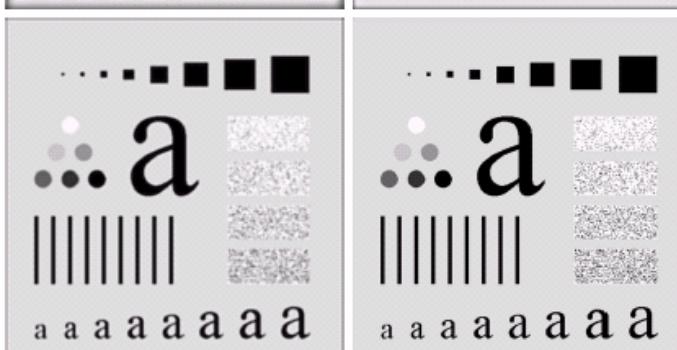
Original Image



cutoff=15



cutoff=80



cutoff=5

- (GLPF): The inverse Fourier transform of the Gaussian Lowpass filter is also Gaussian in the Spatial domain.

cutoff=30

- Therefore there is **no ringing effect** of the GLPF. Ringing artifacts are not acceptable in fields like medical imaging. Hence use Gaussian instead of the ILPF/BLPF.

cutoff=230

# **Image Enhancement in the *Frequency Domain***

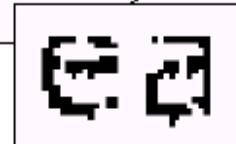
## **Smoothing Frequency-Domain Filters**

- **Gaussian Lowpass Filter (GLPF):** Refer to the improvement in the following example.

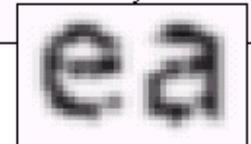
a b

**FIGURE 4.19**  
(a) Sample text of poor resolution (note broken characters in magnified view).  
(b) Result of filtering with a GLPF (broken character segments were joined).

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.



Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.



# Image Enhancement in the *Frequency Domain*

## Smoothing Frequency-Domain Filters

- **Gaussian Lowpass Filter (GLPF):** *The following example shows a lady younger. How? By using GLPF!*



a b c

**FIGURE 4.20** (a) Original image ( $1028 \times 732$  pixels). (b) Result of filtering with a GLPF with  $D_0 = 100$ . (c) Result of filtering with a GLPF with  $D_0 = 80$ . Note reduction in skin fine lines in the magnified sections of (b) and (c).

---

# **Image Enhancement in the *Frequency Domain***

## **Sharpening Frequency-Domain Filters**

- *The high-frequency components are: edges and sharp transitions such as noise.*
- *Sharpening can be achieved by highpass filtering process, which attenuates low frequency components without disturbing the high-frequency information in the frequency domain.*
- *The filter transfer function,  $H_{hp}(u,v)$ , of a highpass filter is given by:*

$$H_{hp}(u,v) = 1 - H_{lp}(u,v)$$

- *Where  $H_{lp}(u,v)$ , is the transfer function of the corresponding lowpass filter.*
- *We will consider only 3 types of sharpening highpass filters :*
  - *Ideal Highpass filters,*
  - *Butterworth Highpass filters,*
  - *Gaussian Highpass filters*

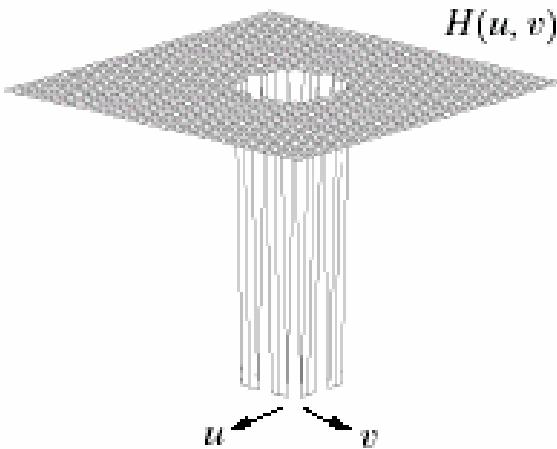
# Image Enhancement in the *Frequency Domain*

## Sharpening Frequency-Domain Filters

- **Ideal Highpass Filter (IHPF):** Simply cuts off all the low frequencies lower than the specified cutoff frequency. The filter transfer function:

$$H(u, v) = \begin{cases} 0 & \text{if } D(u, v) \leq D_0 \\ 1 & \text{if } D(u, v) > D_0 \end{cases}$$

$D(u, v)$  is the distance from the origin  
 $D_0$  is the cutoff frequency



Perspective plot  
of IHPF

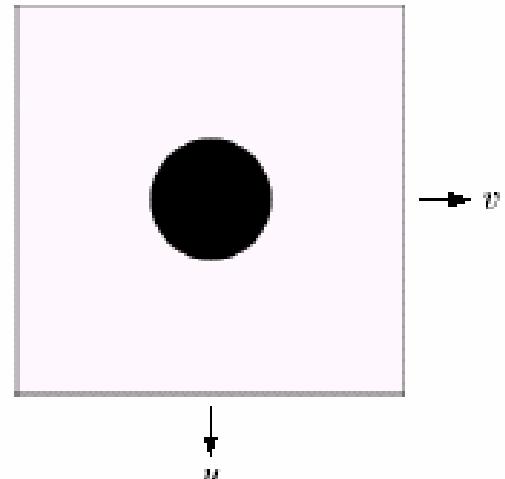
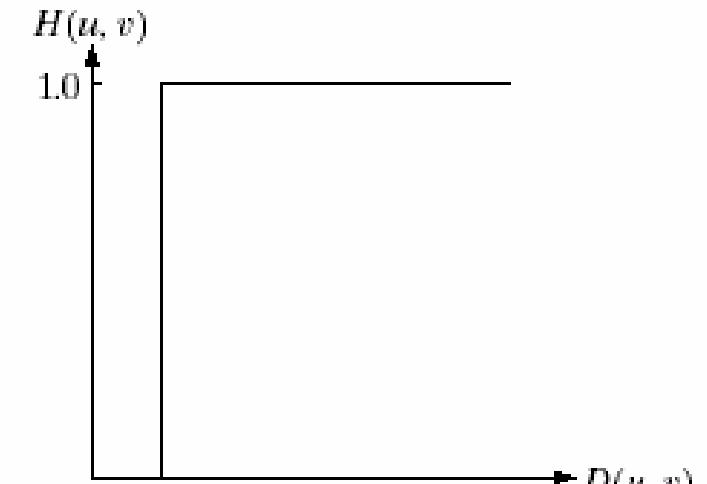


Image representation  
of IHPF

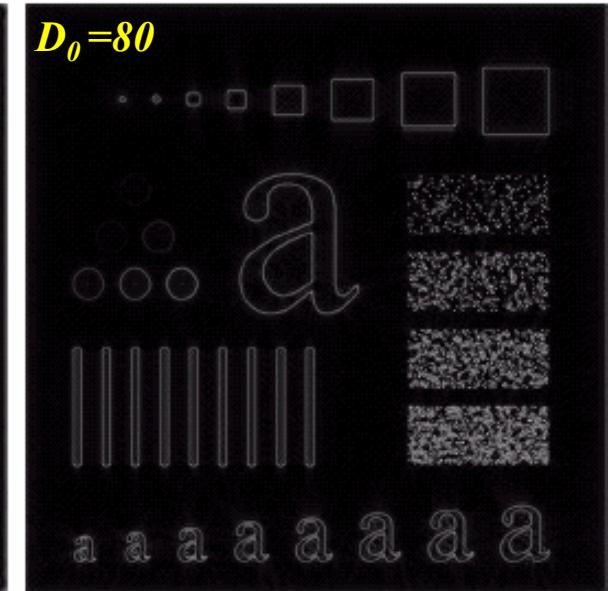
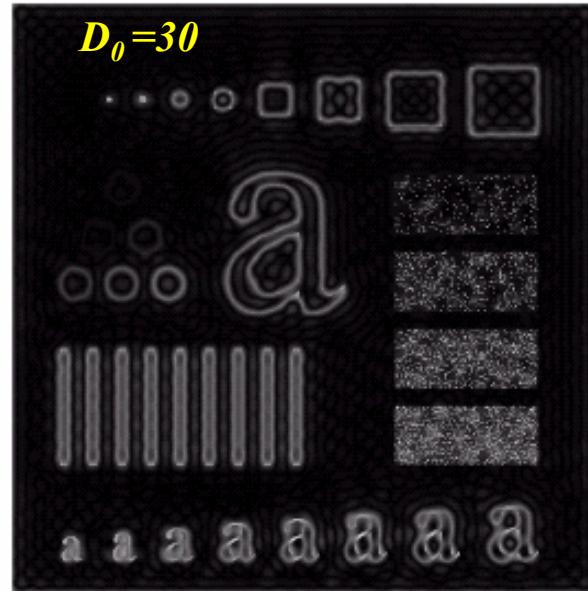
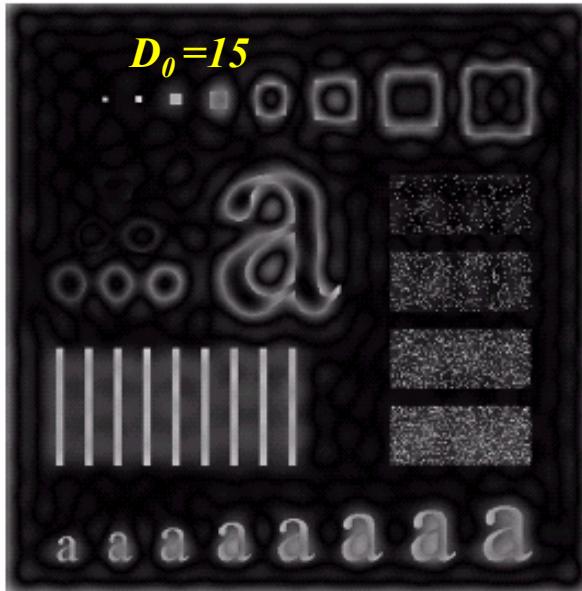


Cross section of  
IHPF

# Image Enhancement in the *Frequency Domain*

## Sharpening Frequency-Domain Filters

- Ideal Highpass Filter (IHPF): *The ringing artifacts occur at low cutoff frequencies*



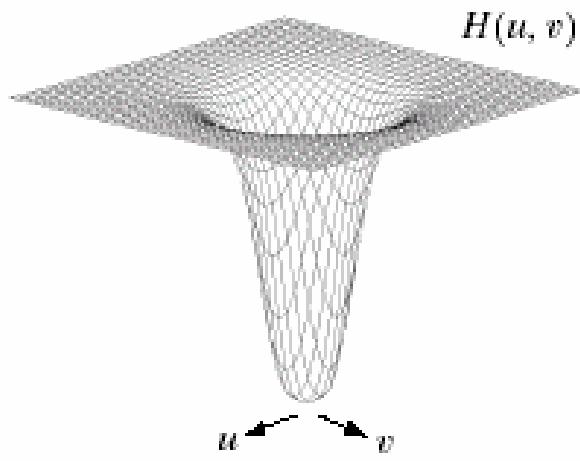
# Image Enhancement in the *Frequency Domain*

## Sharpening Frequency-Domain Filters

- **Butterworth Highpass Filter (BHPF):** *The transfer function of BHPF of order n and with a specified cutoff frequency is given by:*

$$H(u, v) = \frac{1}{1 + [D_0 / D(u, v)]^{2n}}$$

$D(u, v)$  is the distance from the origin  
 $D_0$  is the cutoff frequency.  
 $n$  is the order of the filter



Perspective plot  
of BHPF

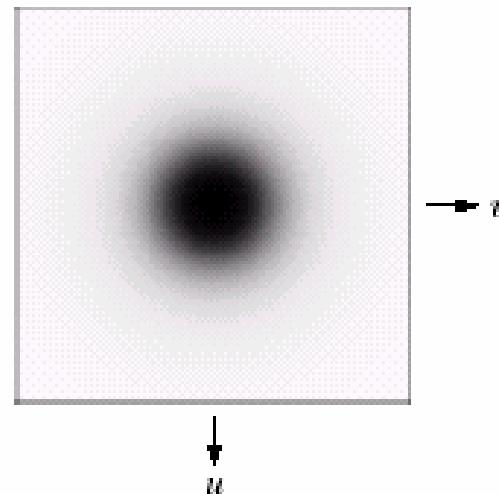
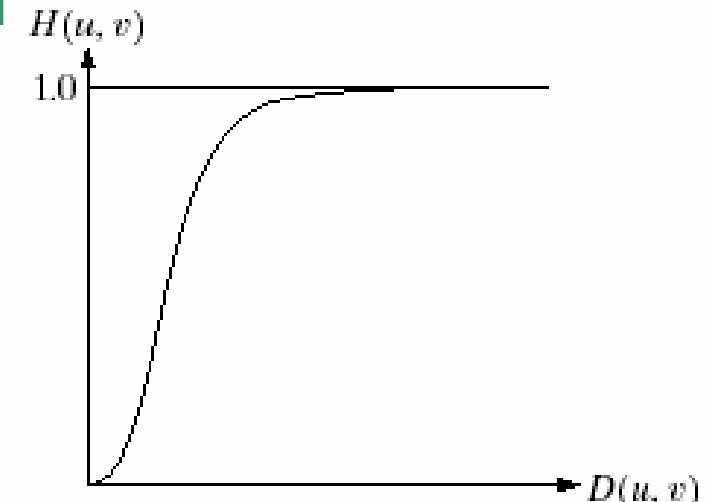


Image representation  
of BHPF



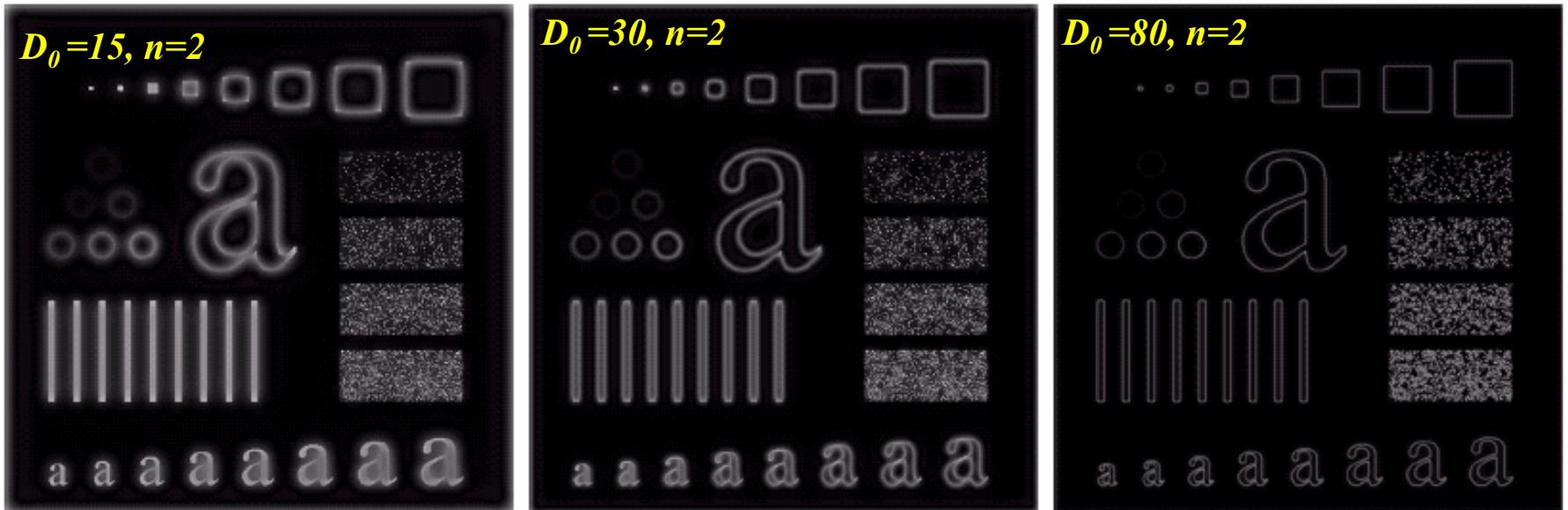
Cross section of  
BHPF

---

# Image Enhancement in the *Frequency Domain*

## Sharpening Frequency-Domain Filters

- **Butterworth Highpass Filter (BHPF):** *Smoother results are obtained in BHPF when compared IHPF. There is almost no ringing artifacts.*



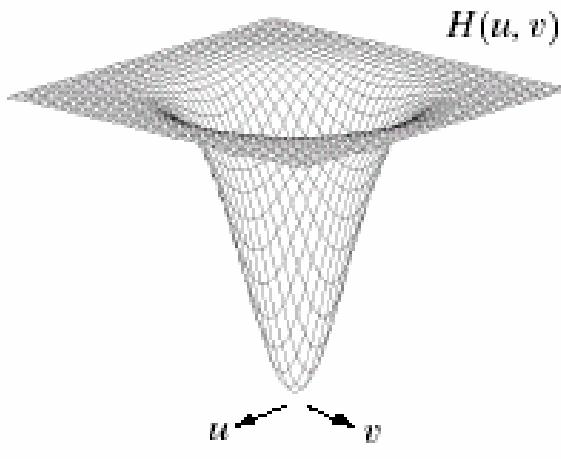
# Image Enhancement in the *Frequency Domain*

## Sharpening Frequency-Domain Filters

- **Gaussian Highpass Filter (GHPF):** *The transfer function of GHPF is given by:*

$$H(u, v) = 1 - e^{-D^2(u, v)/2D_0^2}$$

$D(u, v)$  is the distance from the origin  
 $D_0$  is the cutoff frequency.



Perspective plot  
of BHPF

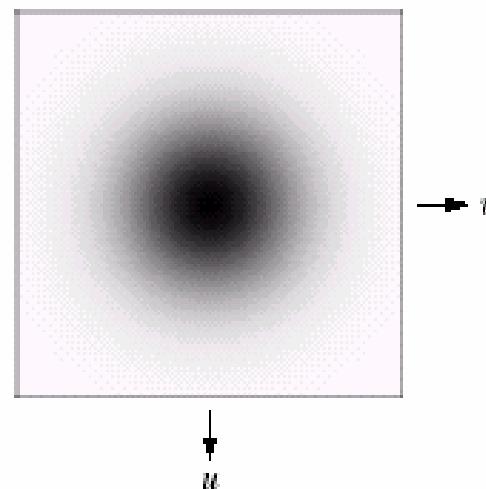
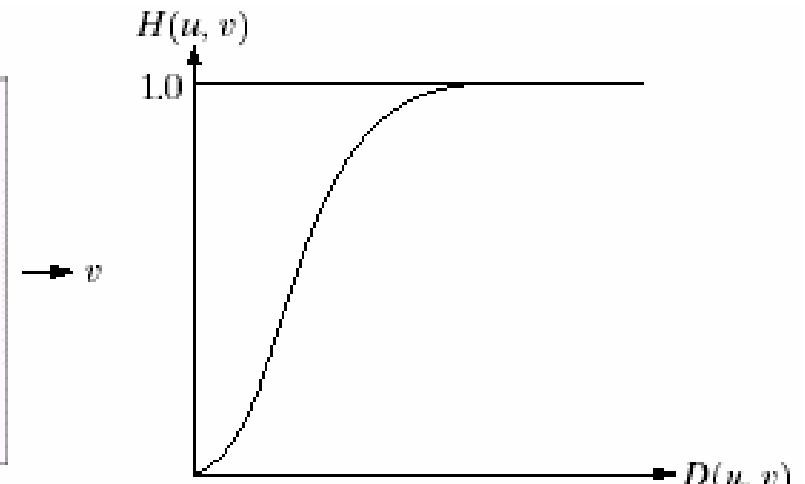


Image representation  
of BHPF

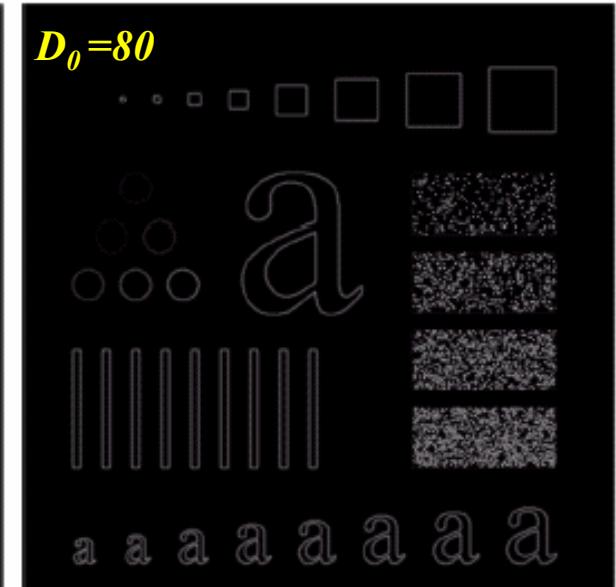
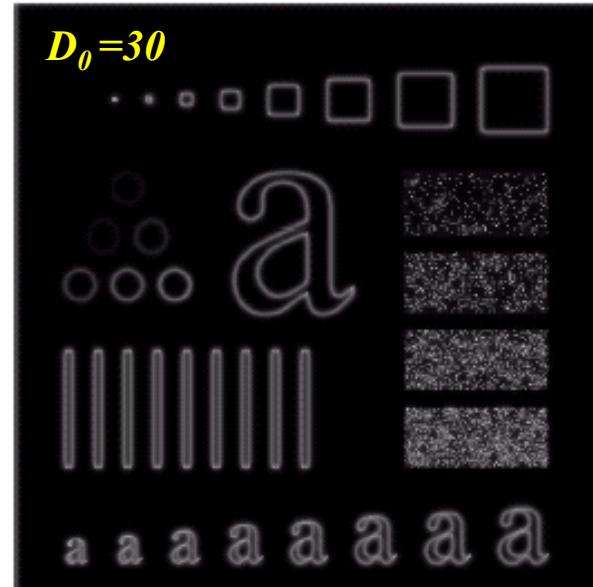
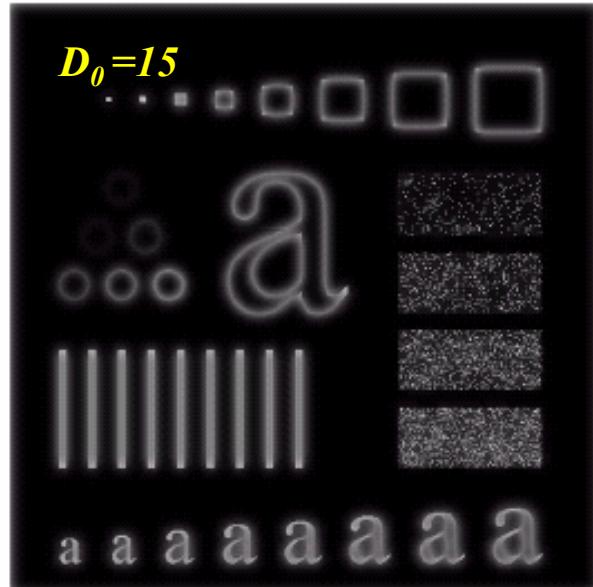


Cross section of  
BHPF

# Image Enhancement in the *Frequency Domain*

## Sharpening Frequency-Domain Filters

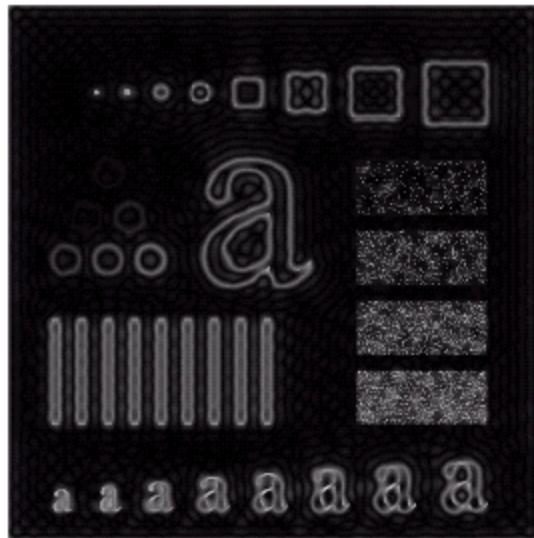
- Gaussian Highpass Filter (GHPF): *Smoother results are obtained in GHPF when compared BHPF. There is absolutely no ringing artifacts.*



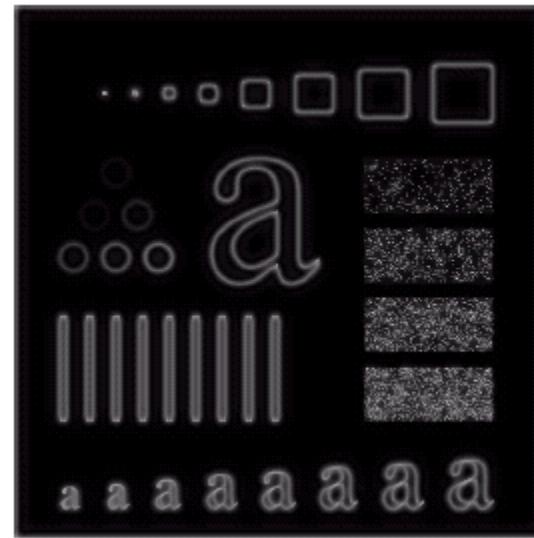
# Image Enhancement in the *Frequency Domain*

## Sharpening Frequency-Domain Filters

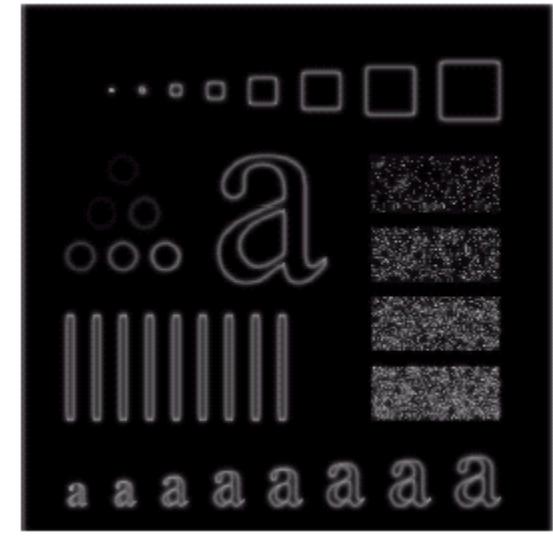
- Comparison of Ideal, Butterworth and Gaussian High pass Filters  
 $(D_o = 30)$



*Ideal High pass Filter*



*Butterworth High pass Filter*



*Gaussian High pass Filter*

# The Laplacian in the Frequency Domain

$$H(u, v) = -4\pi^2(u^2 + v^2)$$

$$\begin{aligned} H(u, v) &= -4\pi^2 \left[ (u - P/2)^2 + (v - Q/2)^2 \right] \\ &= -4\pi^2 D^2(u, v) \end{aligned}$$

The Laplacian image

$$\nabla^2 f(x, y) = \mathcal{J}^{-1} \{ H(u, v) F(u, v) \}$$

Enhancement is obtained

$$g(x, y) = f(x, y) + c \nabla^2 f(x, y) \quad c = -1$$

# The Laplacian in the Frequency Domain

The enhanced image

$$\begin{aligned}g(x, y) &= \mathcal{I}^{-1} \left\{ F(u, v) - H(u, v)F(u, v) \right\} \\&= \mathcal{I}^{-1} \left\{ [1 - H(u, v)]F(u, v) \right\} \\&= \mathcal{I}^{-1} \left\{ [1 + 4\pi^2 D^2(u, v)]F(u, v) \right\}\end{aligned}$$

# The Laplacian in the Frequency Domain



a b

**FIGURE 4.58**

(a) Original, blurry image.  
(b) Image enhanced using the Laplacian in the frequency domain. Compare with Fig. 3.38(e).

# Unsharp Masking, Highboost Filtering and High-Frequency-Emphasis Fitering

$$g_{mask}(x, y) = f(x, y) - f_{LP}(x, y)$$

$$f_{LP}(x, y) = \mathcal{F}^{-1} [H_{LP}(u, v)F(u, v)]$$

Unsharp masking and highboost filtering

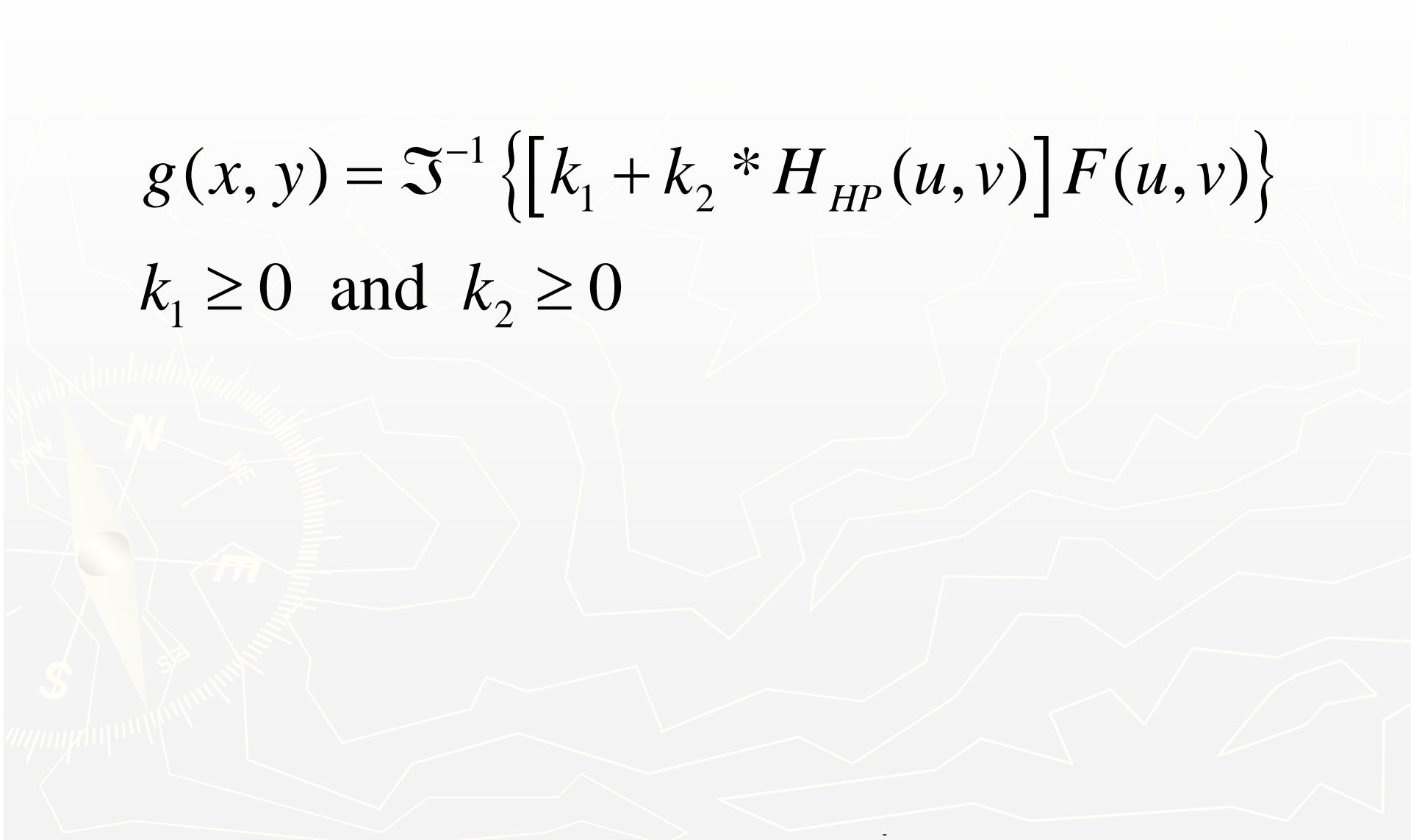
$$g(x, y) = f(x, y) + k * g_{mask}(x, y)$$

$$\begin{aligned} g(x, y) &= \mathcal{F}^{-1} \left\{ \left[ 1 + k * \left[ 1 - H_{LP}(u, v) \right] \right] F(u, v) \right\} \\ &= \mathcal{F}^{-1} \left\{ \left[ 1 + k * H_{HP}(u, v) \right] F(u, v) \right\} \end{aligned}$$

# Unsharp Masking, Highboost Filtering and High-Frequency-Emphasis Fitering

$$g(x, y) = \mathcal{I}^{-1} \left\{ [k_1 + k_2 * H_{HP}(u, v)] F(u, v) \right\}$$

$$k_1 \geq 0 \text{ and } k_2 \geq 0$$





Gaussian Filter  
 $D_0=40$



High-Frequency-Emphasis Filtering  
Gaussian Filter  
 $K1=0.5, k2=0.75$

a | b  
c | d

**FIGURE 4.59** (a) A chest X-ray image. (b) Result of highpass filtering with a Gaussian filter. (c) Result of high-frequency-emphasis filtering using the same filter. (d) Result of performing histogram equalization on (c). (Original image courtesy of Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School.)

# Homomorphic Filtering

$$f(x, y) = i(x, y)r(x, y)$$

$$\Im[f(x, y)] = \Im[i(x, y)]\Im[r(x, y)] ?$$

$$z(x, y) = \ln f(x, y) = \ln i(x, y) + \ln r(x, y)$$

$$\Im\{z(x, y)\} = \Im\{\ln f(x, y)\} = \Im\{\ln i(x, y)\} + \Im\{\ln r(x, y)\}$$

$$Z(u, v) = F_i(u, v) + F_r(u, v)$$

# Homomorphic Filtering

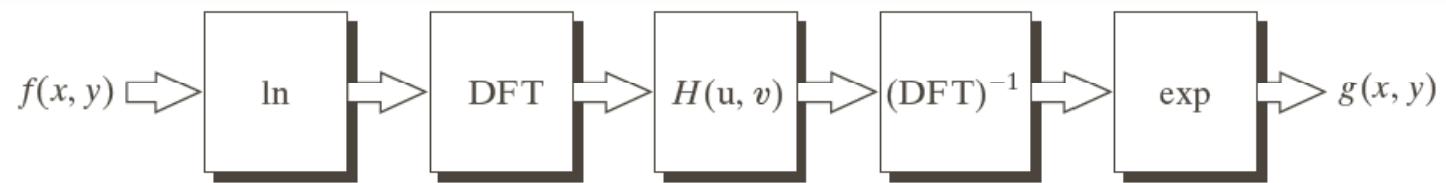
$$\begin{aligned} S(u, v) &= H(u, v)Z(u, v) \\ &= H(u, v)F_i(u, v) + H(u, v)F_r(u, v) \end{aligned}$$

$$\begin{aligned} s(x, y) &= \mathfrak{J}^{-1}\{S(u, v)\} \\ &= \mathfrak{J}^{-1}\{H(u, v)F_i(u, v) + H(u, v)F_r(u, v)\} \\ &= \mathfrak{J}^{-1}\{H(u, v)F_i(u, v)\} + \mathfrak{J}^{-1}\{H(u, v)F_r(u, v)\} \\ &= i'(x, y) + r'(x, y) \end{aligned}$$

$$g(x, y) = e^{s(x, y)} = e^{i'(x, y)}e^{r'(x, y)} = i_0(x, y)r_0(x, y)$$

# Homomorphic Filtering

**FIGURE 4.60**  
Summary of steps  
in homomorphic  
filtering.

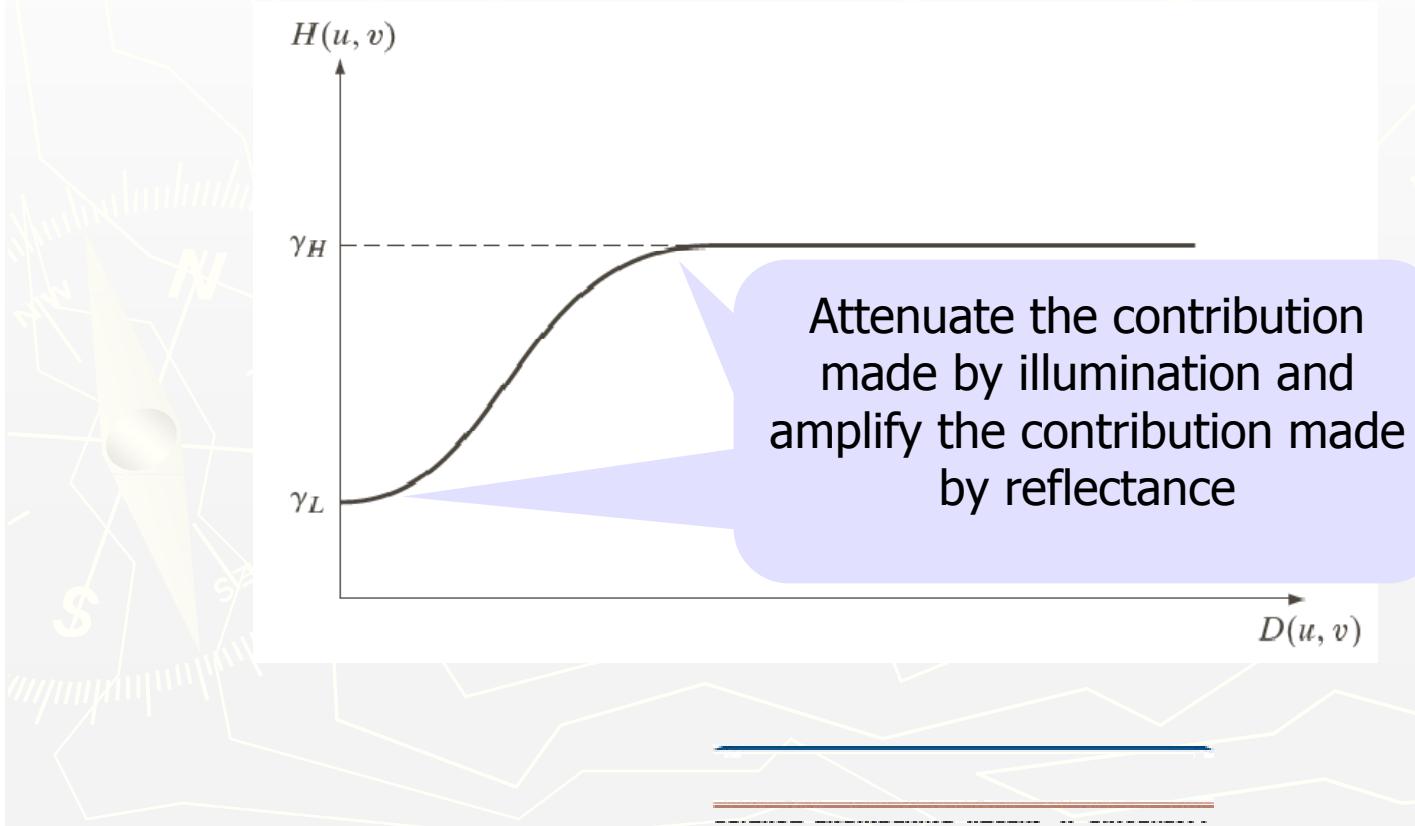


The illumination component of an image generally is characterized by slow spatial variations, while the reflectance component tends to vary abruptly

These characteristics lead to associating the low frequencies of the Fourier transform of the logarithm of an image with illumination the high frequencies with reflectance.

# Homomorphic Filtering

$$H(u, v) = (\gamma_H - \gamma_L) \left[ 1 - e^{-c[D^2(u, v)/D_0^2]} \right] + \gamma_L$$

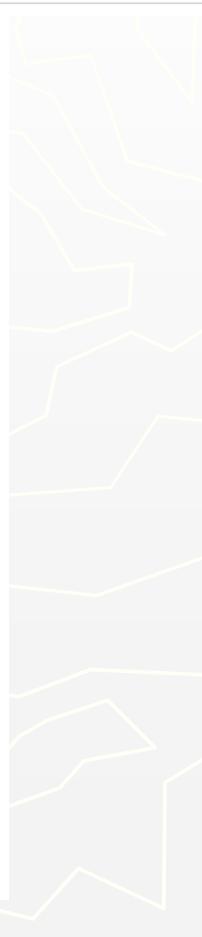
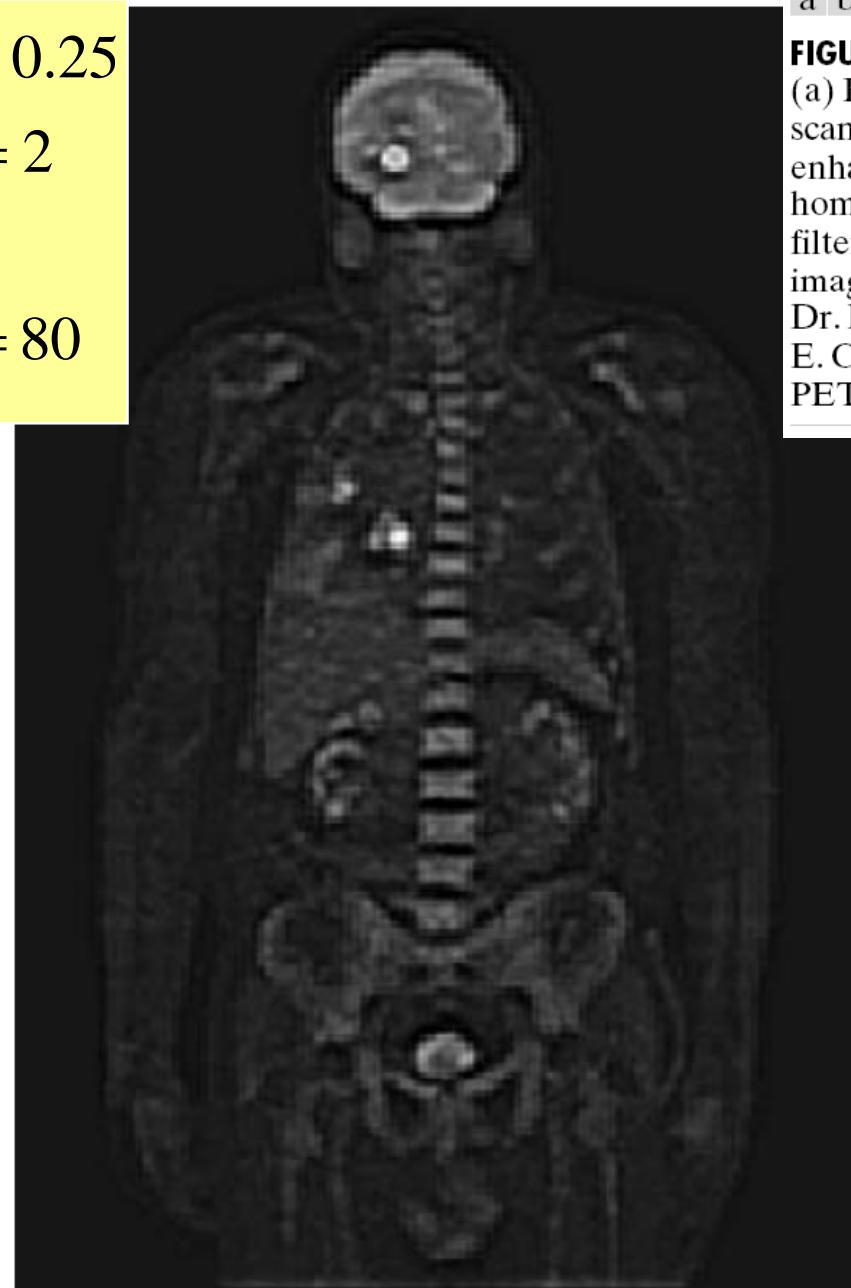
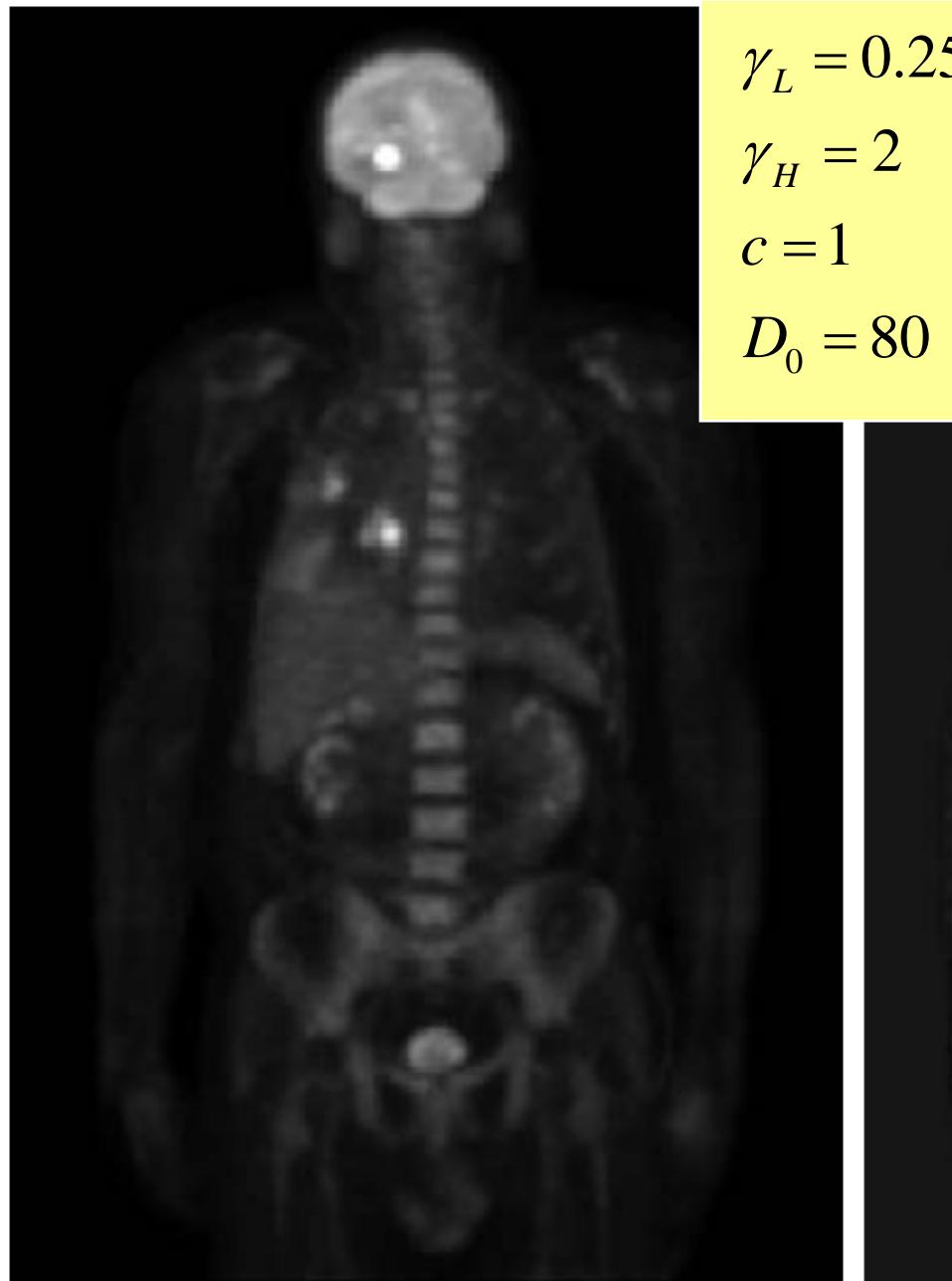


**FIGURE 4.61**  
Radial cross section of a circularly symmetric homomorphic filter function. The vertical axis is at the center of the frequency rectangle and  $D(u, v)$  is the distance from the center.

a b

**FIGURE 4.62**

(a) Full body PET scan. (b) Image enhanced using homomorphic filtering. (Original image courtesy of Dr. Michael E. Casey, CTI PET Systems.)

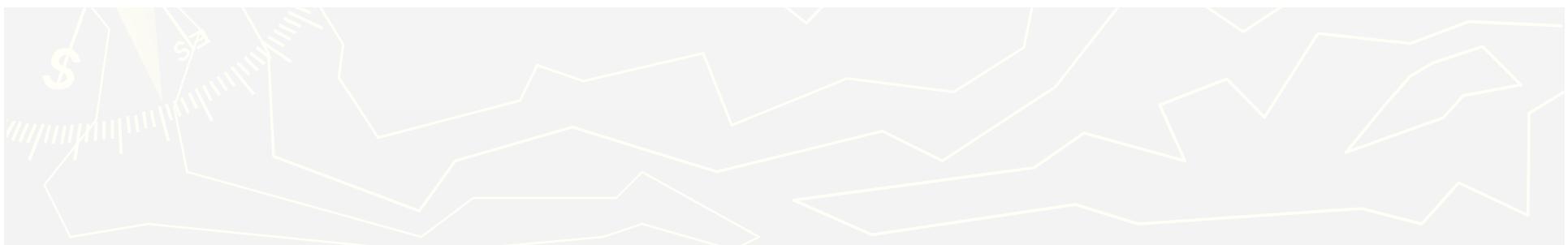
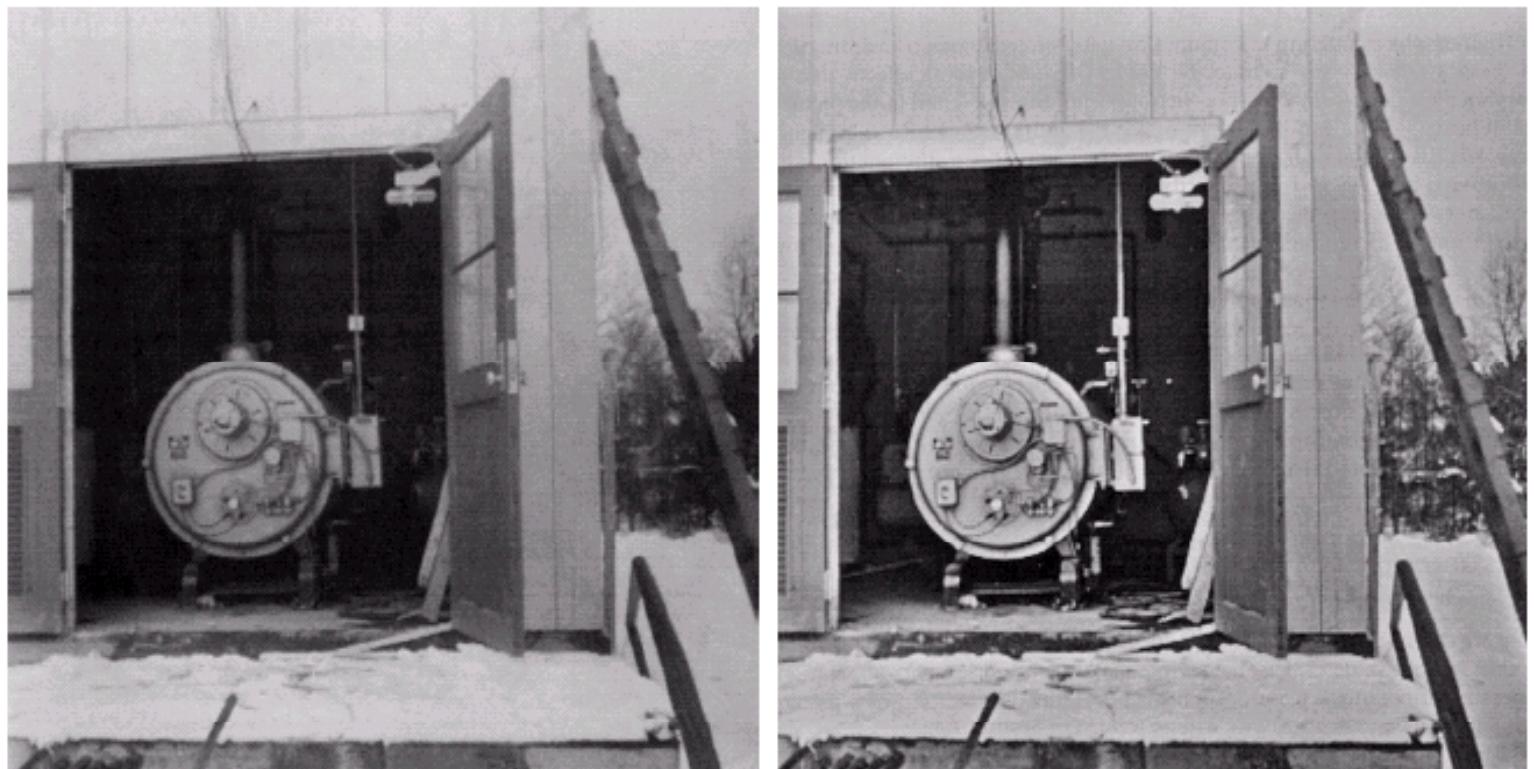


# Homomorphic Filtering

a b

**FIGURE**

(a) Original image. (b) Image processed by homomorphic filtering (note details inside shelter).  
(Stockham.)



# Selective Filtering

## Non-Selective Filters:

operate over the entire frequency rectangle

## Selective Filters

operate over some part, not entire frequency rectangle

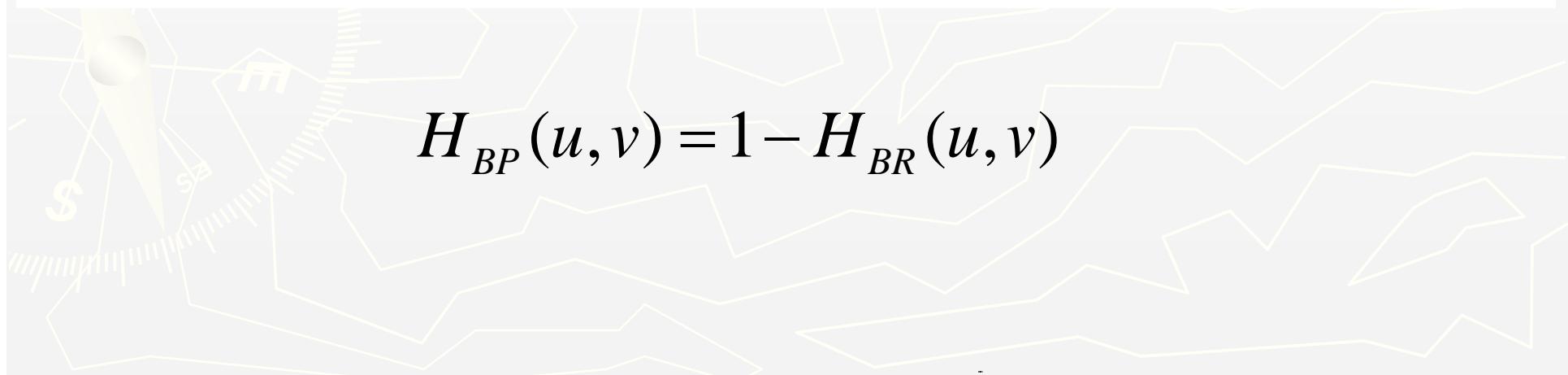
- **bandreject or bandpass:** process specific bands
- **notch filters:** process small regions of the frequency rectangle

# Selective Filtering: Bandreject and Bandpass Filters

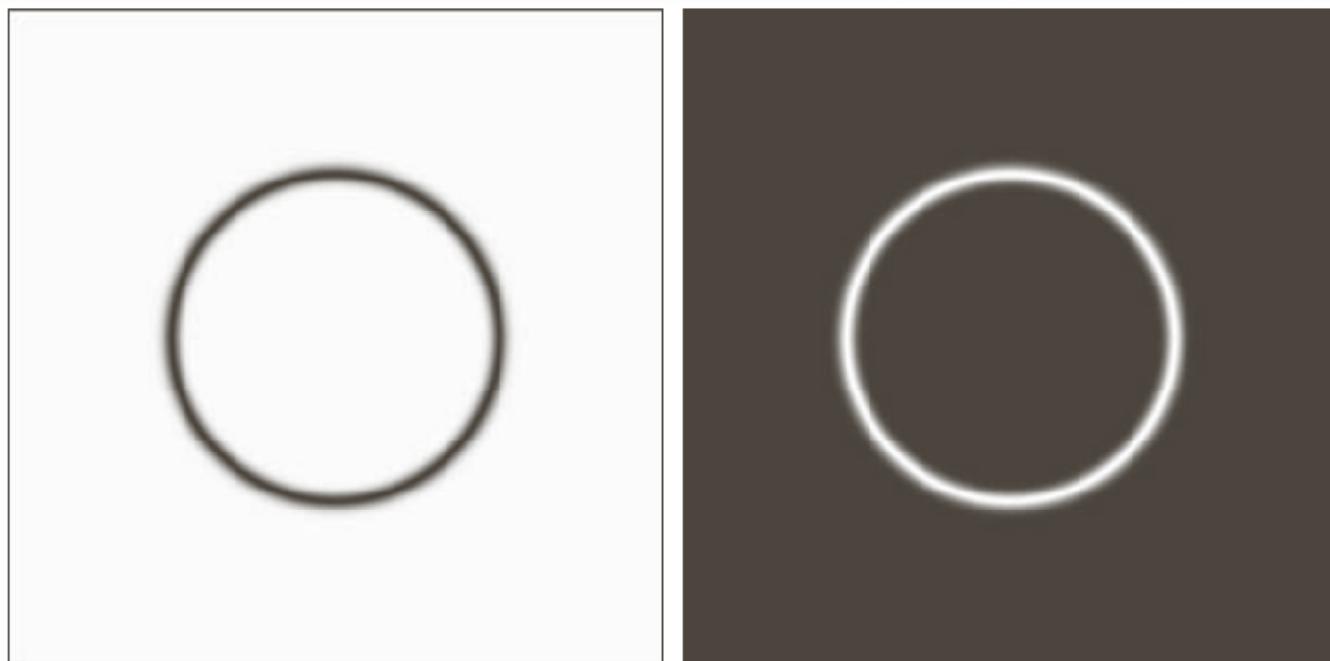
**TABLE 4.6**

Bandreject filters.  $W$  is the width of the band,  $D$  is the distance  $D(u, v)$  from the center of the filter,  $D_0$  is the cutoff frequency, and  $n$  is the order of the Butterworth filter. We show  $D$  instead of  $D(u, v)$  to simplify the notation in the table.

<b>Ideal</b>	<b>Butterworth</b>	<b>Gaussian</b>
$H(u, v) = \begin{cases} 0 & \text{if } D_0 - \frac{W}{2} \leq D \leq D_0 + \frac{W}{2} \\ 1 & \text{otherwise} \end{cases}$	$H(u, v) = \frac{1}{1 + \left[ \frac{DW}{D^2 - D_0^2} \right]^{2n}}$	$H(u, v) = 1 - e^{-\left[ \frac{D^2 - D_0^2}{DW} \right]^2}$



# Selective Filtering: Bandreject and Bandpass Filters



**FIGURE 4.63**  
(a) Bandreject Gaussian filter.  
(b) Corresponding bandpass filter.  
The thin black border in (a) was added for clarity; it is not part of the data.

# Selective Filtering: Notch Filters

Zero-phase-shift filters must be symmetric about the origin.  
A notch with center at  $(u_0, v_0)$  must have a corresponding  
notch at location  $(-u_0, -v_0)$ .

Notch reject filters are constructed as products of highpass  
filters whose centers have been translated to the centers of  
the notches.

$$H_{NR}(u, v) = \prod_{k=1}^Q H_k(u, v) H_{-k}(u, v)$$

where  $H_k(u, v)$  and  $H_{-k}(u, v)$  are highpass filters whose centers are  
at  $(u_k, v_k)$  and  $(-u_k, -v_k)$ , respectively.

# Selective Filtering: Notch Filters

$$H_{NR}(u, v) = \prod_{k=1}^Q H_k(u, v) H_{-k}(u, v)$$

where  $H_k(u, v)$  and  $H_{-k}(u, v)$  are highpass filters whose centers are at  $(u_k, v_k)$  and  $(-u_k, -v_k)$ , respectively.

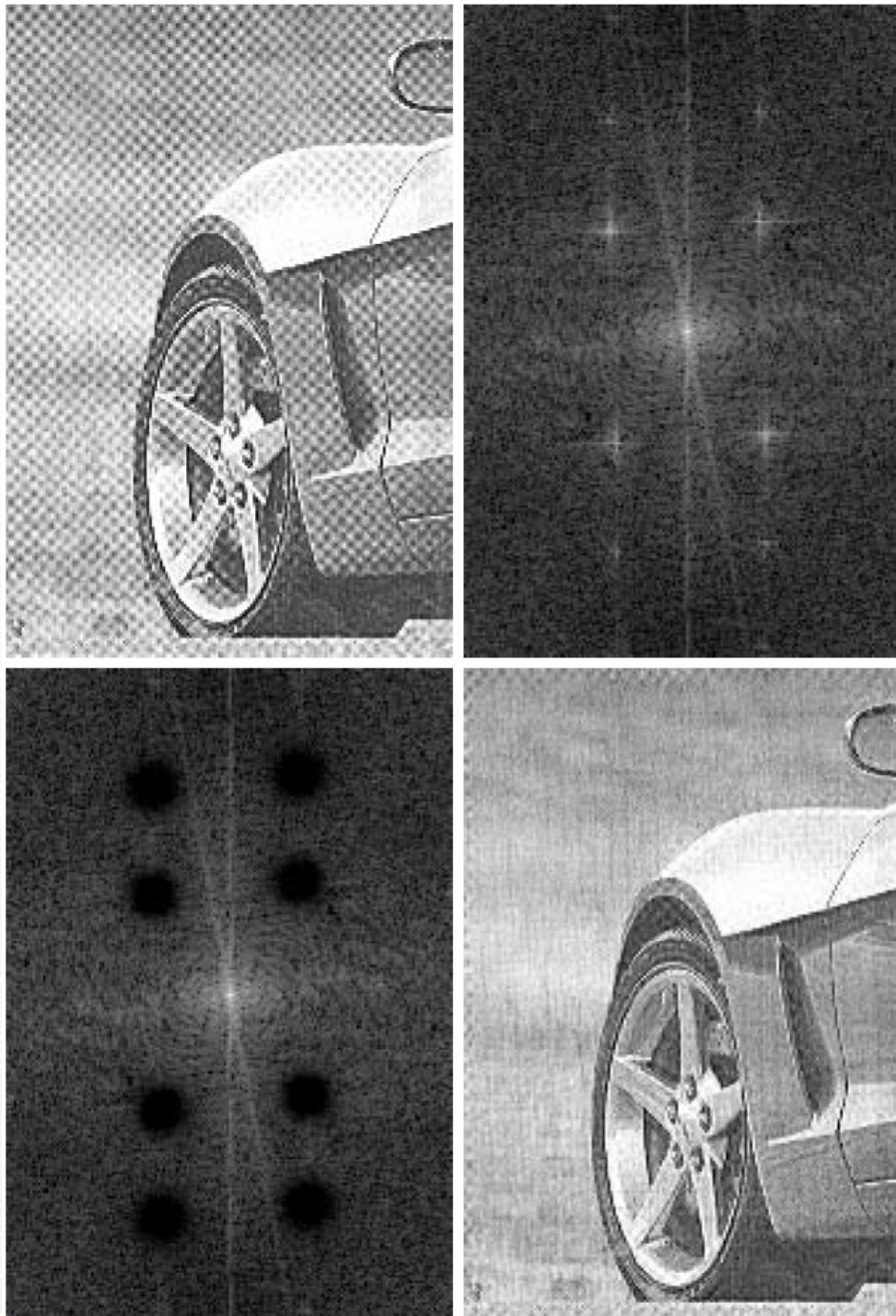
A Butterworth notch reject filter of order n

$$H_{NR}(u, v) = \prod_{k=1}^3 \left[ \frac{1}{1 + [D_{0k} / D_k(u, v)]^{2n}} \right] \left[ \frac{1}{1 + [D_{0k} / D_{-k}(u, v)]^{2n}} \right]$$

$$D_k(u, v) = \left[ (u - M/2 - u_k)^2 + (v - N/2 - v_k)^2 \right]^{1/2}$$

$$D_{-k}(u, v) = \left[ (u - M/2 + u_k)^2 + (v - N/2 + v_k)^2 \right]^{1/2}$$

## Examples: Notch Filters (1)



a  
b  
c  
d

**FIGURE 4.64**

- (a) Sampled newspaper image showing a moiré pattern.
- (b) Spectrum.
- (c) Butterworth notch reject filter multiplied by the Fourier transform.
- (d) Filtered image.

A Butterworth notch reject filter  $D_0=3$  and  $n=4$  for all notch pairs

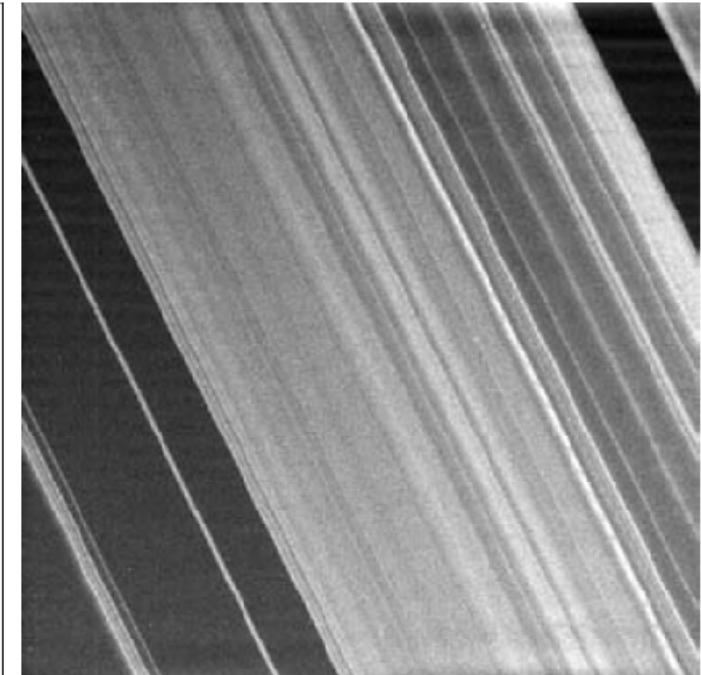
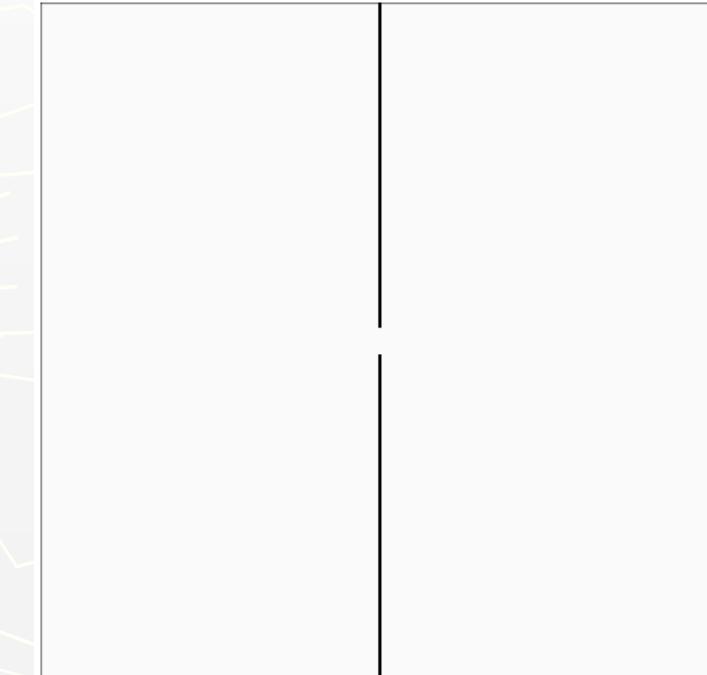
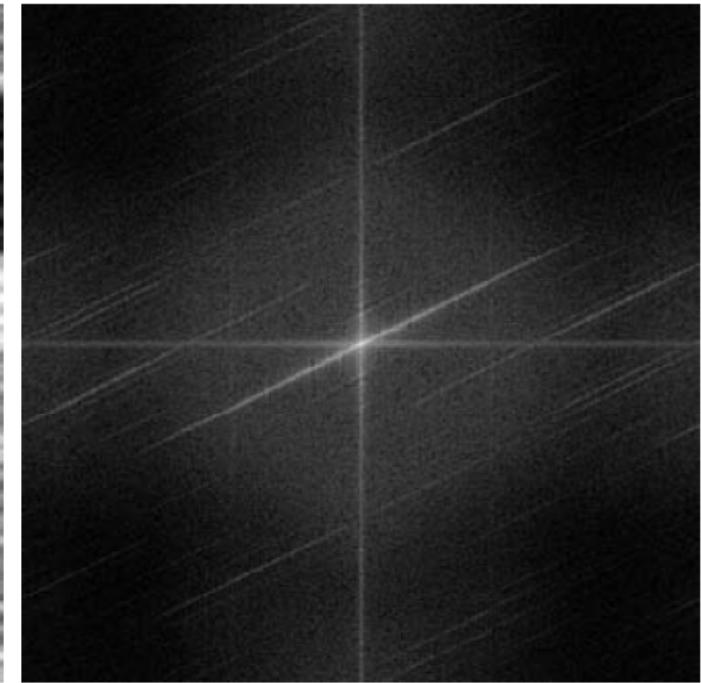
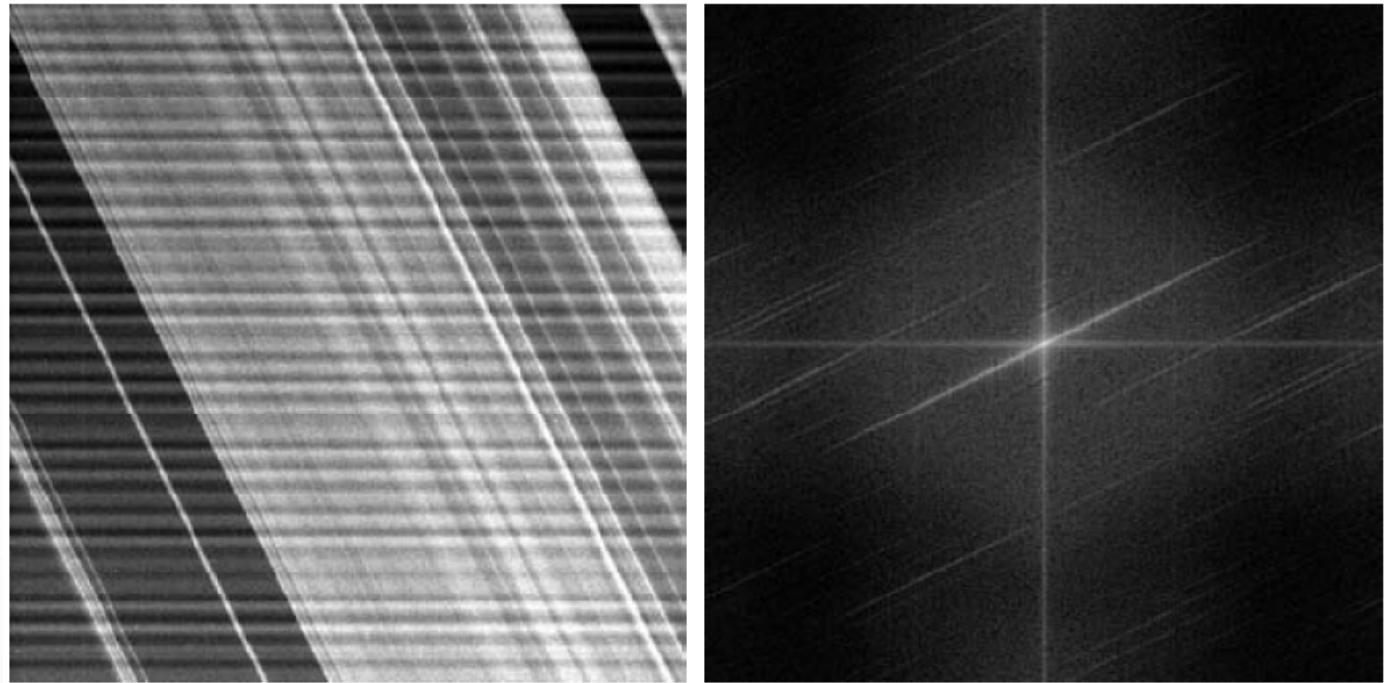
## Examples: Notch Filters (2)

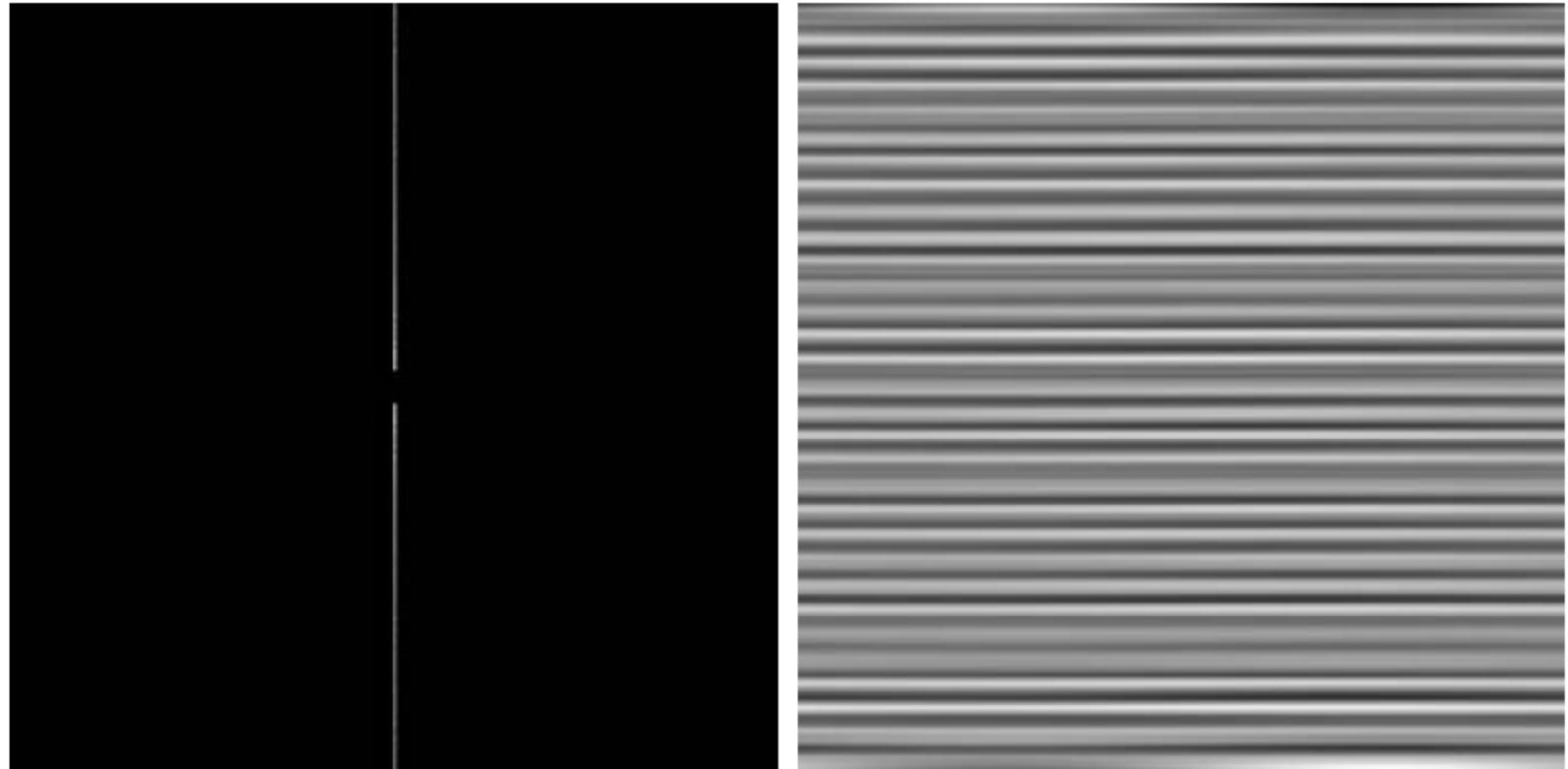
a b  
c d

**FIGURE 4.65**

- (a)  $674 \times 674$  image of the Saturn rings showing nearly periodic interference.
- (b) Spectrum: The bursts of energy in the vertical axis near the origin correspond to the interference pattern.
- (c) A vertical notch reject filter.
- (d) Result of filtering. The thin black border in (c) was added for clarity; it is not part of the data.

(Original image courtesy of Dr. Robert A. West, NASA/JPL.)





a b

**FIGURE 4.66**

(a) Result (spectrum) of applying a notch pass filter to the DFT of Fig. 4.65(a).  
(b) Spatial pattern obtained by computing the IDFT of (a).

# Image Enhancement in the Frequency Domain

- **Example 8(PR3.25) :** Show that the Laplacian operation is isotropic (invariant to rotation). You will need the following equations relating coordinates after axis rotation by an angle  $\theta$ .

$$x = x' \cos \theta - y' \sin \theta$$

$$y = x' \sin \theta + y' \cos \theta$$

Laplacian operator is defined as:

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

For the rotated Laplacian operator:  $\nabla^2 f = \frac{\partial^2 f}{\partial x'^2} + \frac{\partial^2 f}{\partial y'^2}$ .

Given that:  $x = x' \cos \theta - y' \sin \theta$  and  $y = x' \sin \theta + y' \cos \theta$

If we show that the right sides of the first 2 equations are equal than the Laplacian operation is rotation invariant.

We start with,

$$\begin{aligned}\frac{\partial f}{\partial x'} &= \frac{\partial f}{\partial x} \frac{\partial x}{\partial x'} + \frac{\partial f}{\partial y} \frac{\partial y}{\partial x'} \\ &= \frac{\partial f}{\partial x} \cos \theta + \frac{\partial f}{\partial y} \sin \theta.\end{aligned}$$

# Image Enhancement in the Frequency Domain

- **Example 8(PR3.25):**

$$x = x' \cos \theta - y' \sin \theta \quad \text{and} \quad y = x' \sin \theta + y' \cos \theta$$

*Taking the partial derivative of this expression again with respect to  $x'$  yields:*

$$\frac{\partial^2 f}{\partial x'^2} = \frac{\partial^2 f}{\partial x^2} \cos^2 \theta + \frac{\partial}{\partial x} \left( \frac{\partial f}{\partial y} \right) \sin \theta \cos \theta + \frac{\partial}{\partial y} \left( \frac{\partial f}{\partial x} \right) \cos \theta \sin \theta + \frac{\partial^2 f}{\partial y^2} \sin^2 \theta.$$

*Repeat the same operation for  $y'$ :*

$$\begin{aligned} \frac{\partial f}{\partial y'} &= \frac{\partial f}{\partial x} \frac{\partial x}{\partial y'} + \frac{\partial f}{\partial y} \frac{\partial y}{\partial y'} \\ &= -\frac{\partial f}{\partial x} \sin \theta + \frac{\partial f}{\partial y} \cos \theta. \end{aligned}$$

*Taking the partial derivative of this expression again with respect to  $y'$  yields:*

$$\frac{\partial^2 f}{\partial y'^2} = \frac{\partial^2 f}{\partial x^2} \sin^2 \theta - \frac{\partial}{\partial x} \left( \frac{\partial f}{\partial y} \right) \cos \theta \sin \theta - \frac{\partial}{\partial y} \left( \frac{\partial f}{\partial x} \right) \sin \theta \cos \theta + \frac{\partial^2 f}{\partial y^2} \cos^2 \theta$$

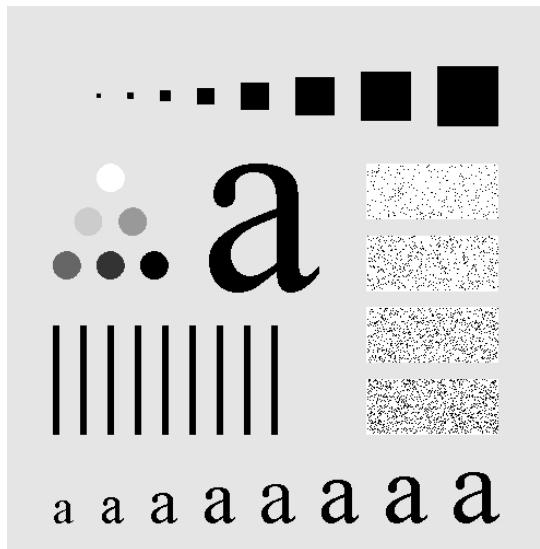
*Adding the 2 expressions for the second derivatives:*

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

*Both sides are equal, Hence Laplacian is rotational invariant.*

# Image Enhancement in the Spatial Domain

- **Example 9(PR4.7):** *What is the source of nearly periodic bright points in the horizontal axis of the spectrum in the following figure.*



*The nearly periodic bright points in the frequency spectrum corresponds to the periodic bars as well as the repeating boxes, letters and circles in the horizontal direction.*

# Image Enhancement in the Spatial Domain

- **Example 10(PR4.6) -a):** Prove the validity of the following Equation.

$\mathfrak{F}[f(x, y)(-1)^{x+y}] = F(u - M/2, v - N/2)$ ,  $\mathfrak{F}[\cdot]$  denotes the Fourier Transform

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}$$

$$F(u - M/2, v - N/2) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi((u-M/2)x/M + (v-N/2)y/N)}$$

$$= \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi((ux/M - 1/2x) + (vy/N - 1/2y))}$$

$$e^{-j2\pi((ux/M - 1/2x) + (vy/N - 1/2y))} = e^{j\pi(x+y)} e^{-j2\pi(ux/M + vy/N)} = (-1)^{x+y} e^{-j2\pi(ux/M + vy/N)}$$

$$e^{j\pi(x+y)} = (-1)^{x+y} \rightarrow e^{j\pi(x+y)} = \cos(\pi(x+y)) + j \sin(\pi(x+y))$$



always zero

1 or -1 depending on the addition of  $x+y$ . If  $(x+y)$  is even then 1, otherwise -1.

$$= \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) (-1)^{x+y} e^{-j2\pi(ux/M + vy/N)}$$

# Image Enhancement in the Spatial Domain

- **Example 10(PR4.6) -b):** Prove the validity of the following 2 Equations.

$$f(x, y)e^{j2\pi(u_0x/M+v_0y/N)} \Leftrightarrow F(u-u_0, v-v_0)$$

$$f(x-x_0, y-y_0) \Leftrightarrow F(u, v)e^{-j2\pi(ux_0/M+vy_0/N)}$$

$$\begin{aligned}\Im[f(x, y)e^{j2\pi(u_0x/M+v_0y/N)}] &= \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y)e^{j2\pi(u_0x/M+v_0y/N)}] e^{-j2\pi(ux/M+vy/N)} \\ &= \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi([u-u_0]x/M+[v-v_0]y/N)} \\ &= F(u-u_0, v-v_0)\end{aligned}$$

Similarly, it can be shown that :  $\Im[F(u, v)e^{-j2\pi(ux_0/M+vy_0/N)}] = f(x-x_0, y-y_0)$

This is the Translation property of the 2D Fourier transform:

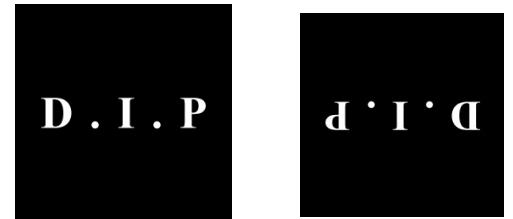
When  $x_0=u_0=M/2$  and  $y_0=v_0=N/2$ , then  $f(x, y)(-1)^{x+y} \Leftrightarrow F(u-M/2, v-N/2)$

$$f(x-M/2, y-N/2) \Leftrightarrow F(u, v)(-1)^{u+v}$$

# Image Enhancement in the Spatial Domain

- **Example 11(PR4.9):** Consider the images shown below. The image on the right is obtained by (a) multiplying the image on the left by  $(-1)^{x+y}$ ; (b) computing the DFT; (c) taking the complex conjugate of the transform; (d) computing the inverse DFT; and (e) multiplying the real part of the result by  $(-1)^{x+y}$ . Explain (mathematically) why the image on the right appears as it does.

The complex conjugate simply changes  $j$  to  $-j$  in the inverse transform, so the image on the right is given by:



$$\begin{aligned}\Im[F(u,v)^*] &= \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{-j2\pi(ux/M+vy/N)} \\ &= \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(u(-x)/M+v(-y)/N)} \\ &= f(-x,-y)\end{aligned}$$

Which simply mirrors  $f(x,y)$  about the origin, thus producing the image on the right

# Image Enhancement in the Spatial Domain

- **Example 12(PR4.14):** Suppose that you form a low pass filter that averages the four immediate neighbors of a point  $(x,y)$ , but excludes the point itself.
- (a) Find the equivalent filter  $H(u,v)$  in the frequency domain.
- (b) Show that  $H(u,v)$  is a lowpass filter.

a) The spatial average is given by:

$$g(x, y) = \frac{1}{4} [f(x, y + 1) + f(x + 1, y) + f(x - 1, y) + f(x, y - 1)]$$

Then, using the following property:

$$f(x - x_0, y - y_0) \Leftrightarrow F(u, v) e^{-j2\pi(ux_0/M + vy_0/N)}$$

$$\begin{aligned} G(u, v) &= \frac{1}{4} \left[ e^{j2\pi v/N} + e^{j2\pi u/M} + e^{-j2\pi u/M} + e^{-j2\pi v/N} \right] F(u, v) \\ &= H(u, v) F(u, v), \end{aligned}$$

Where the  $H(u,v)$  is the filter function. We get the following transfer function:

$$H(u, v) = \frac{1}{2} [\cos(2\pi u/M) + \cos(2\pi v/N)]$$

---

# Image Enhancement in the Spatial Domain

- **Example 12(PR4.14):** Suppose that you form a low pass filter that averages the four immediate neighbors of a point  $(x,y)$ , but excludes the point itself.
- (a) Find the equivalent filter  $H(u,v)$  in the frequency domain.
- (b) Show that  $H(u,v)$  is a lowpass filter.

a) The  $H(u,v)$  Filter function can be centered by:

$$H(u, v) = \frac{1}{2} [\cos(2\pi[u - M/2]/M) + \cos(2\pi[v - N/2]/N)]$$

b) Consider one variable for convenience. As  $u$  ranges from 0 to  $M$ , the value of  $\cos(2\pi[u-M/2]/M)$  starts at -1, peaks at 1 when  $u = M/2$  (the center of the filter) and then decreases to -1 again when  $u = M$ . Thus, we see that the amplitude of the filter decreases as a function of distance from the origin of the centered filter, which is the characteristic of a lowpass filter.

A similar argument is easily carried out when considering both variables simultaneously.

# Image Enhancement in the Spatial Domain

- **Example 8-PR4.19:** Derive the frequency domain filter that corresponds to the Laplacian operator in the spatial domain.

0	1	0
1	-4	1
0	1	0

Consider the Laplacian mask given. Then,

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

$$G(u, v) = H(u, v)F(u, v)$$

Where

$$\begin{aligned} H(u, v) &= \left[ e^{j2\pi u/M} + e^{-j2\pi u/M} + e^{j2\pi v/N} + e^{-j2\pi v/N} - 4 \right] \\ &= 2 [\cos(2\pi u/M) + \cos(2\pi v/N) - 2]. \end{aligned}$$

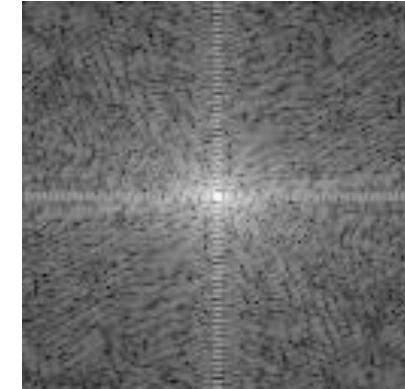
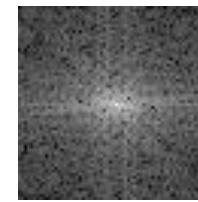
The  $H(u, v)$  Filter function can be centered by:

$$H(u, v) = 2 [\cos(2\pi [u - M/2]/M) + \cos(2\pi [v - N/2]/N) - 2]$$

# Image Enhancement in the Spatial Domain

- **Example 8-PR4.22:** *The two fourier spectra shown are of the same image. The spectrum of the left corresponds to the original image, and the spectrum on the right was obtained after the image was padded with zeros.*
- (a) *Explain the difference of the overall contrast.*
- (b) *Explain the significant increase in signal strength along the vertical and the horizontal axes of the spectrum shown on the right.*

(a) *Padding with zero increases the size but reduces the average gray level of image. The average gray level of the padded image is less than the original image.  $F(0,0)$  in the padded image is less than  $F(0,0)$  of the original image. All the others away from the origin are less in the padded image than the original image. This produces a narrower range of values hence a lower contrast spectrum in the padded image.*



(b) *Padding with 0's introduces significant discontinuities at the borders of the original image. This process introduces strong horizontal and vertical edges where the image ends abruptly and then continues with 0 values. These sharp transitions correspond to the strength of the spectrum along the horizontal and vertical axes.*

---

- Discrete Cosine Transform (DCT):

- 1D Cases:

$$\mathbf{C} = \{c(k, n)\}$$

$$c(k, n) = \begin{cases} \frac{1}{\sqrt{N}} & k = 0, 0 \leq n \leq N-1 \\ \frac{2}{\sqrt{N}} \cos\left(\frac{\pi(2n+1)k}{2N}\right) & 1 \leq k \leq N-1, 0 \leq n \leq N-1 \end{cases}$$

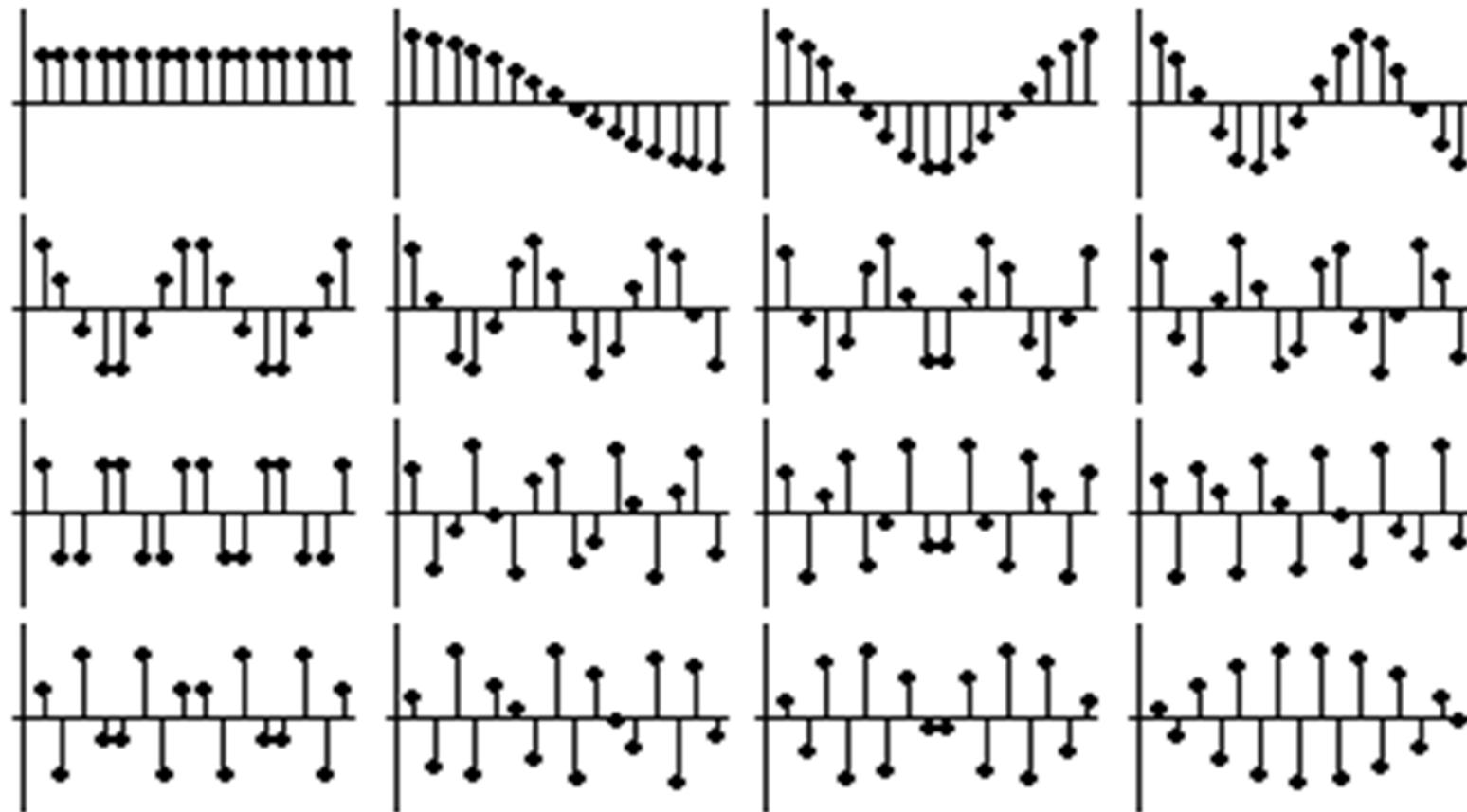
$$v(k) = \alpha(k) \sum_{n=0}^{N-1} u(n) \cos\left(\frac{\pi(2n+1)k}{2N}\right), \quad 0 \leq k \leq N-1$$

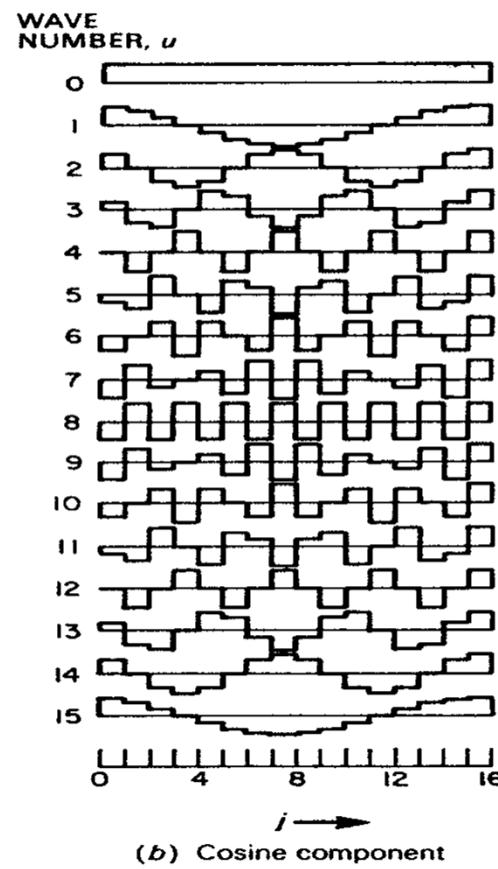
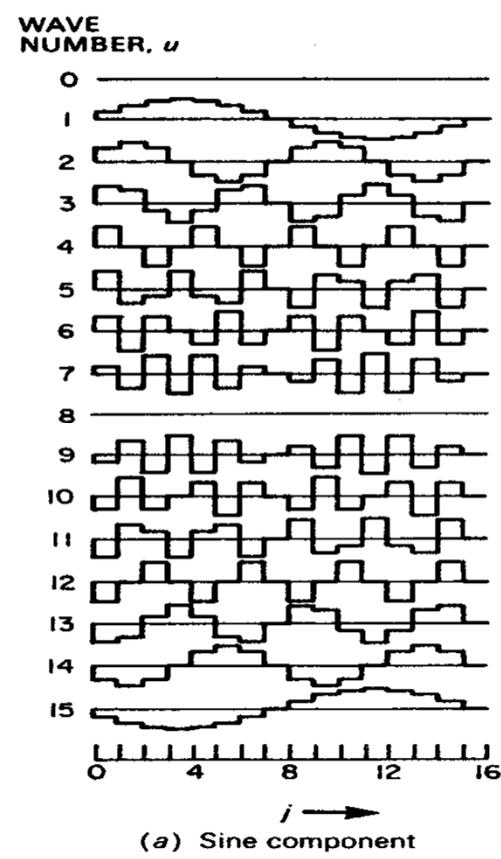
$$\alpha(0) \triangleq \frac{1}{\sqrt{N}}, \quad \alpha(k) \triangleq \frac{2}{\sqrt{N}}, \quad 1 \leq k \leq N-1$$

$$u(n) = \sum_{k=0}^{N-1} \alpha(k) v(k) \cos\left(\frac{\pi(2n+1)k}{2N}\right), \quad 0 \leq n \leq N-1$$

- 
- 
- Properties of DCT:
    - Real and Orthogonal:  $\mathbf{C}=\mathbf{C}^*$   $\rightarrow \mathbf{C}^{-1}=\mathbf{C}^T$
    - Not! Real part of DFT
    - Fast Transform
    - Excellent Energy compaction (Highly Correlated Data)
  - Two Dimensional Cases:
    - $\mathbf{A}=\mathbf{A}^*=\mathbf{C}$

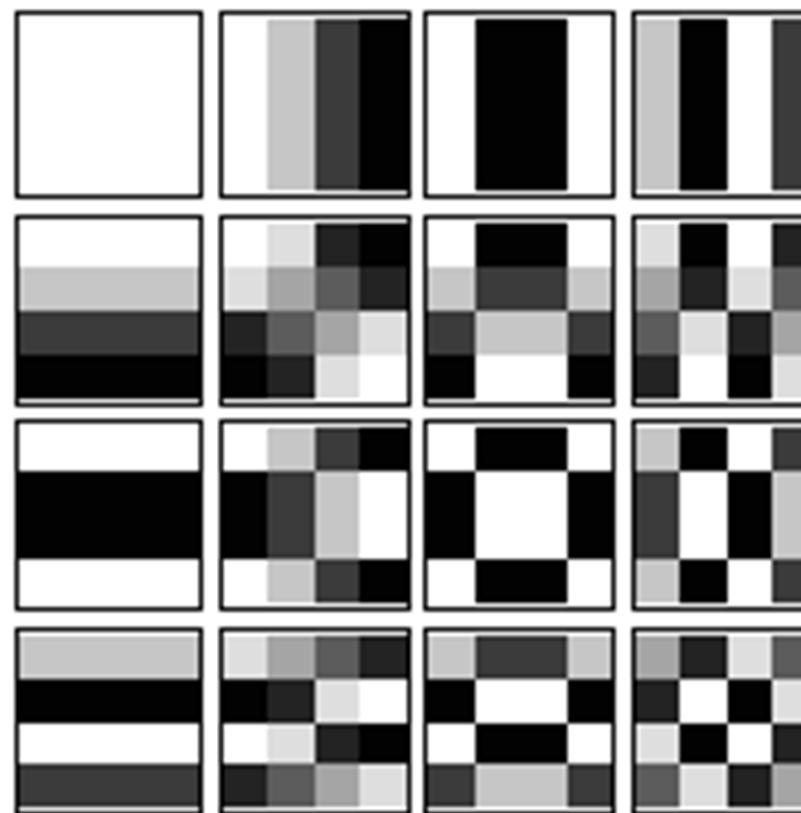
- 
- DCT Basis:





---

- DCT Basis:



---

- Discrete Sine Transform (DST):

- 1D Cases:

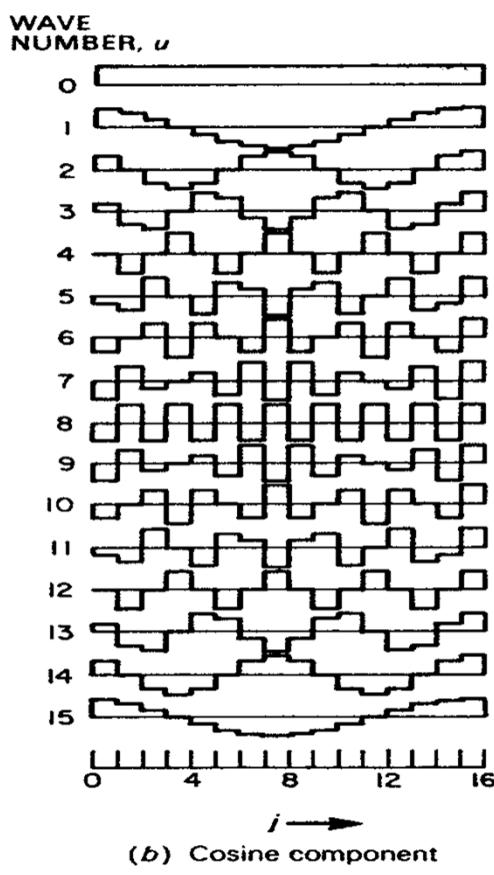
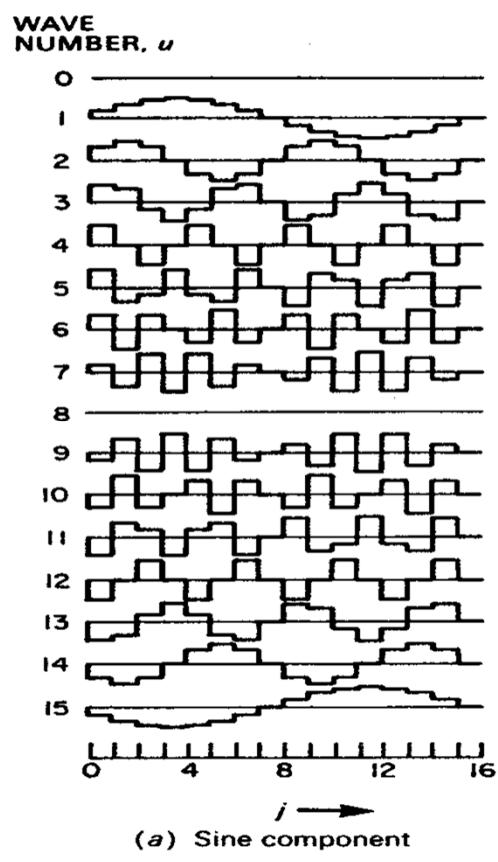
$$\Psi = \{\psi(k, n)\}$$

$$\psi(k, n) = \sqrt{\frac{2}{N+1}} \sin\left(\frac{\pi(n+1)(k+1)}{N+1}\right), \quad 0 \leq k, n \leq N-1$$

$$v(k) = \sqrt{\frac{2}{N+1}} \sum_{n=0}^{N-1} u(n) \sin\left(\frac{\pi(n+1)(k+1)}{N+1}\right), \quad 0 \leq k \leq N-1$$

$$u(n) = \sqrt{\frac{2}{N+1}} \sum_{k=0}^{N-1} v(k) \sin\left(\frac{\pi(n+1)(k+1)}{N+1}\right), \quad 0 \leq n \leq N-1$$

- 2D Case:  $\mathbf{A} = \mathbf{A}^* = \mathbf{A}^\top = \Psi$



---

---

- Properties of DST:

- Real, Symmetric and Orthogonal:  $\Psi = \Psi^* = \Psi^T = \Psi^{-1}$
- Forward and Inverse are identical
- Not! Imaginary part of DFT
- Fast Transform

---

- Welsh-Hadamard Transform (WHT):  $N=2^n$

- 1D Cases:

$$H_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H_n = H_{n-1} \otimes H_1 = H_1 \otimes H_{n-1} = \frac{1}{\sqrt{2}} \begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{bmatrix}$$

Walsh Function →

$$H_3 = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \begin{matrix} 0 \\ 7 \\ 3 \\ 4 \\ 1 \\ 6 \\ 2 \\ 5 \end{matrix}$$

---

- Welsh-Hadamard Transform (WHT):  $N=2^n$

$$\mathbf{v} = \mathbf{H}\mathbf{u} \Leftrightarrow \mathbf{u} = \mathbf{H}\mathbf{v}, \quad \mathbf{H} = \mathbf{H}_n, N = 2^n$$

$$v(k) = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} u(m) (-1)^{b(k,m)} \quad 0 \leq k \leq N-1$$

$$u(m) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} v(k) (-1)^{b(k,m)} \quad 0 \leq m \leq N-1$$

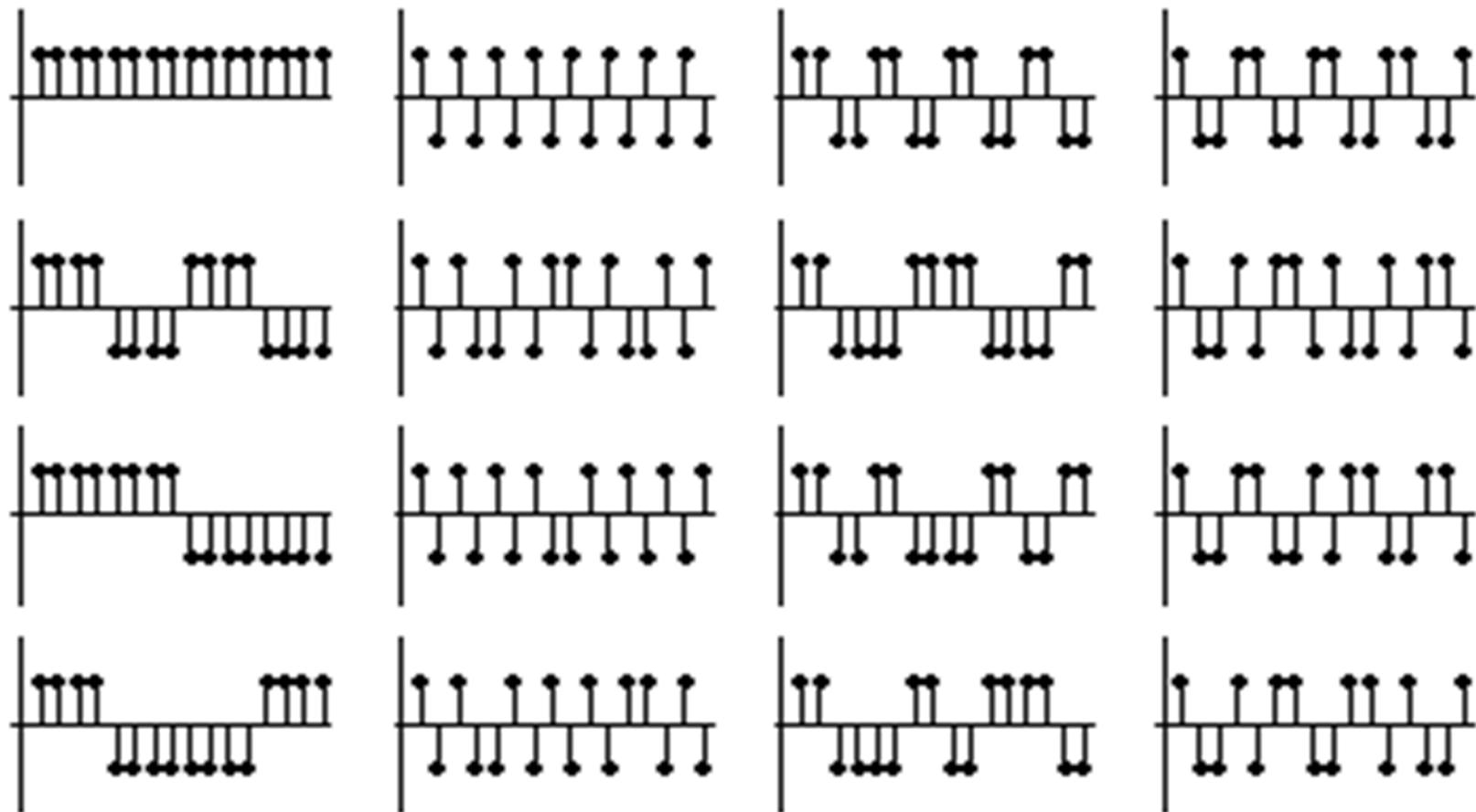
$$b(k, m) = \sum_{i=0}^{n-1} k_i m_i \quad k_i, m_i = 0, 1$$

$$k = \sum_{i=0}^{n-1} k_i 2^i, \quad m = \sum_{i=0}^{n-1} m_i 2^i$$

- 2D Cases:  $\mathbf{A}=\mathbf{A}^*=\mathbf{A}^T=\mathbf{H}$

- 
- 
- Welsh-Hadamard Transform Properties:
    - Real, Symmetric, and orthogonal:  $\mathbf{H}=\mathbf{H}^*=\mathbf{H}^T=\mathbf{H}^{-1}$
    - Ultra Fast Transform ( $\pm 1$ )
    - Good-Very Good energy compactness

- 
- Welsh-Hadamard Basis:



- Welsh-Hadamard Basis

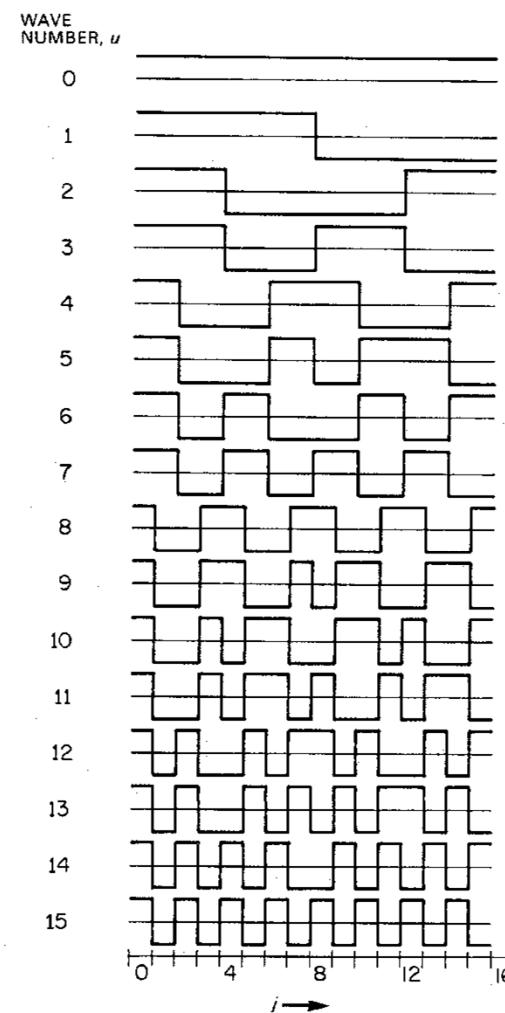
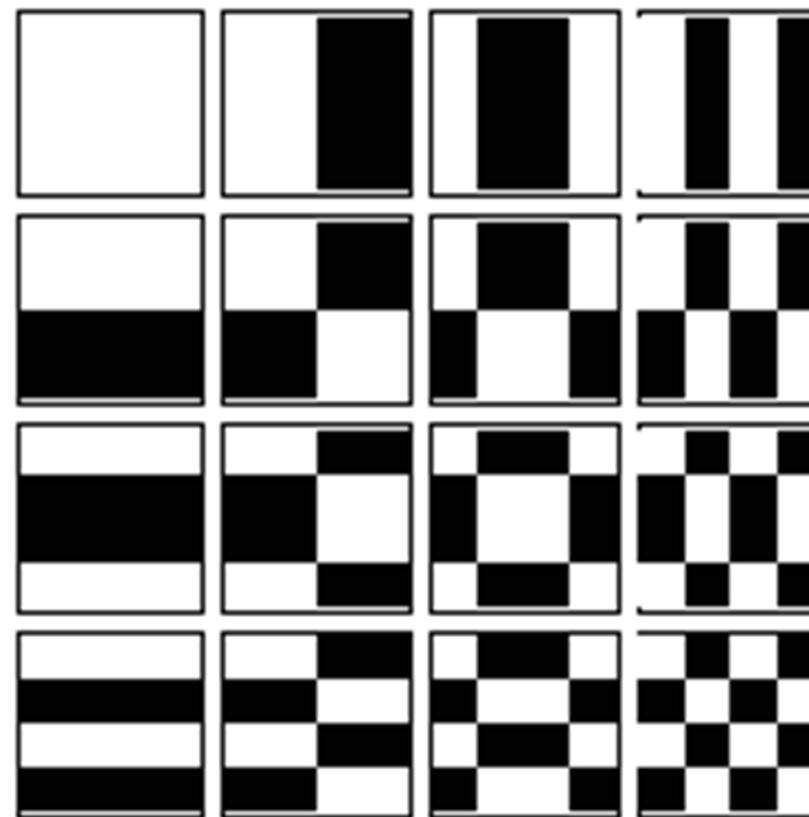


FIGURE 8.4-2. Hadamard transform basis functions,  $N = 16$ .

- 
- Welsh-Hadamard Basis



---

## • Haar Transform $N=2^n$

### – 1D Cases:

$$h_k(x), \quad x \in [0,1], \quad k = 0, \dots, N-1$$

$$k = 2^p + q - 1$$

$$\begin{cases} 0 \leq p \leq n-1; q = 0, 1 & p = 0 \\ 1 \leq q \leq 2^p & p \neq 0 \end{cases}$$

$$h_0(x) \triangleq h_{0,0}(x) = \frac{1}{\sqrt{N}}, \quad x \in [0,1]$$

$$h_k(x) \triangleq h_{p,q}(x) = \frac{1}{\sqrt{N}} \begin{cases} 2^{p/2} & \frac{q-1}{2^p} \leq x \leq \frac{q-1/2}{2^p} \\ -2^{p/2} & \frac{q-1/2}{2^p} \leq x \leq \frac{q}{2^p} \\ 0 & \text{O.W.} \end{cases}$$

$$x = \frac{m}{N}, \quad m = 0, 1, \dots, N-1$$

---

- Haar Transform  $N=2^n$

- 1D Cases:

$$\mathbf{H}_r = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix}$$

- 2D Cases:  $\mathbf{H}_r^* \mathbf{A}^* \mathbf{H}_r^T$

## • Haar Basis Function

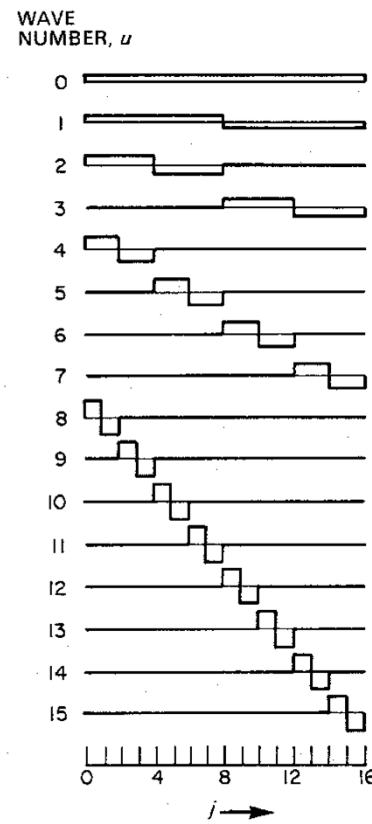


FIGURE 8.4-4. Haar transform basis functions,  $N = 16$ .

- 
- 
- Haar Transform Properties
    - Real and Orthogonal:  $\mathbf{Hr}=\mathbf{Hr}^*$  ,  $\mathbf{Hr}^{-1}=\mathbf{Hr}^T$
    - Fast Transform
    - Poor energy compactness

---

- Slant Transform ( $N=2^n$ )

---

$$S_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$S_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ a_2 & b_2 & -a_2 & b_2 \\ 0 & 1 & 0 & -1 \\ -b_2 & a_2 & b_2 & a_2 \end{bmatrix} \begin{bmatrix} S_1 & 0_2 \\ 0_2 & S_1 \end{bmatrix}$$

## • Slant Transform ( $N=2^n$ )

$$S_n = \frac{1}{2^{\frac{1}{2}}} \begin{bmatrix} 1 & 0 & & & & \\ a_n & b_n & 0 & & & \\ & & I_{(N/2)-2} & & & \\ & & & 1 & 0 & \\ & & & -a_n & b_n & 0 \\ & & & & & S_{n-1} \\ 0 & & & 0 & & 0 \\ & & & & & S_{n-1} \\ 0 & 1 & & & & \\ -b_n & a_n & 0 & & & \\ & & b_n & -1 & & \\ & & & a_n & & 0 \\ 0 & & I_{(N/2)-2} & & & \\ & & & 0 & & -I_{(N/2)-2} \end{bmatrix}$$

$$N = 2^n, \quad a_{n+1} = \left( \frac{3N^2}{4N^2 - 1} \right)^{\frac{1}{2}}, \quad b_{n+1} = \left( \frac{N^2 - 1}{4N^2 - 1} \right)^{\frac{1}{2}}$$

## • Slant Basis Function

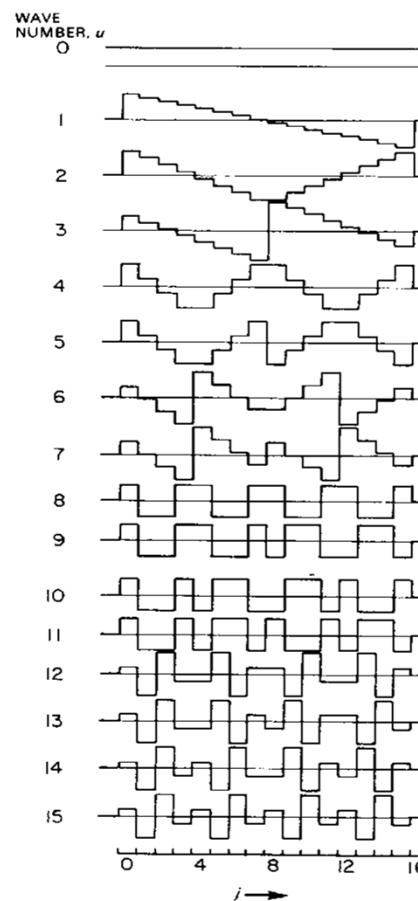


FIGURE 8.4-5. Slant transform basis functions,  $N = 16$ .

- 
- 
- Slant Transform Properties:
    - Real and Orthogonal  $S=S^*$   $S^{-1}=S^T$
    - Fast
    - Very Good Compactness

## **Image Restoration**

- *Restoration is to attempt to reconstruct or recover an image that has been degraded by using a priori knowledge of the degradation .*
- *The restoration approach involves the modeling of the degradation and applying the reverse process to recover the original image.*
- **Image Degradation Model:** *Given some knowledge about the degradation function  $h(x,y)$ , and some knowledge about the additive noise  $\eta(x,y)$ , the degraded output image  $g(x,y)$  can be obtained by:*

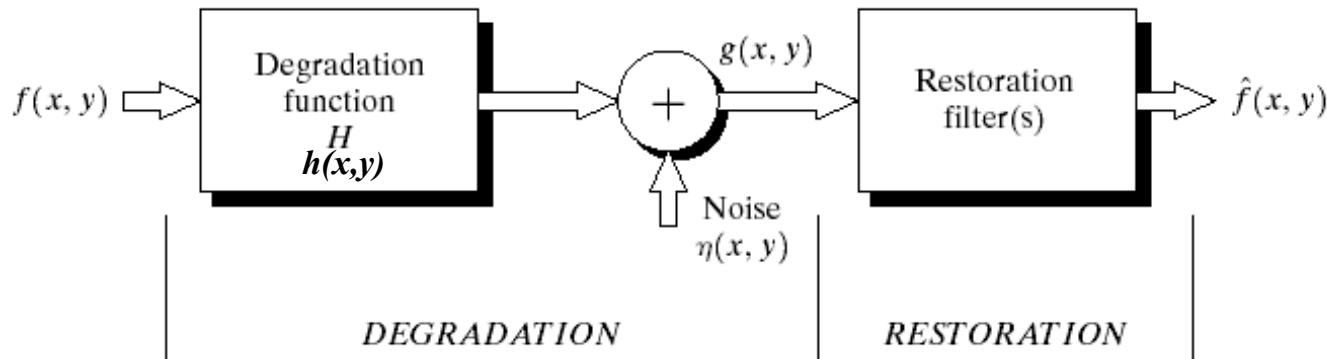
$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y)$$

- *Frequency domain representation can be modeled by:*

$$G(u, v) = H(u, v)F(u, v) + N(u, v)$$

# Image Restoration

## Image Degradation/Restoration Process



**FIGURE 5.1** A model of the image degradation/restoration process.

- **Noise Models:** *Image acquisition (digitization) and the transmission processes are the primary sources of the noise.*

*Assumption: In almost all considerations we will assume that the noise is uncorrelated with the image, which means that there is no correlation between the pixel values of the image and the noise components.*

# Image Restoration

## Noise models: Some important Noise PDFs

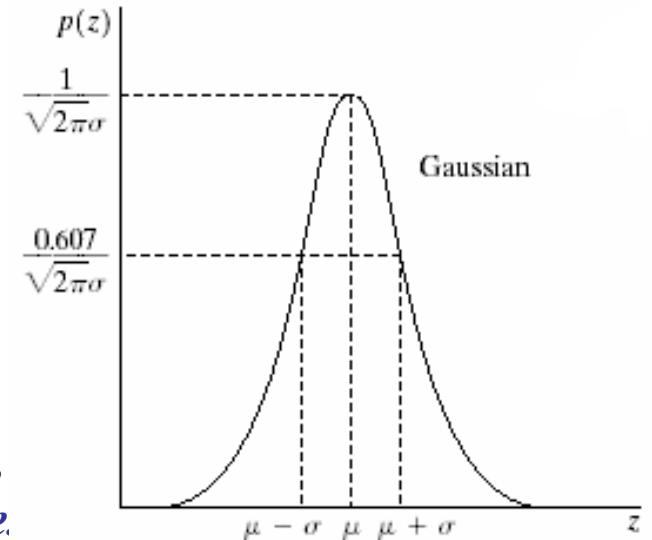
- *The statistical behaviors of the noise components can be considered as the random variables, characterized by the probability density function (PDF).*
- *The following PDFs are common for image processing applications:*

### Gaussian Noise:

- *Gaussian Noise models are frequently used in practice. Because this type of noise model is easily tractable in both spatial and frequency domains.*
- *The PDF of the Gaussian random variable,  $z$  is given by:*

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(z-\mu)^2/2\sigma^2}$$

- *Where  $z$  represents gray level,  $\mu$  is the average value of  $z$ , and the  $\sigma$  is the standard deviation and  $\sigma^2$  is the variance.*
- *Approximately 68% of the values will be in the range of  $[(\mu-\sigma), (\mu+\sigma)]$  and about 95% will be in the range of  $[(\mu-2\sigma), (\mu+2\sigma)]$ .*



# Image Restoration

## Noise models: Some important Noise PDFs

### Rayleigh Noise:

- *The PDF of Rayleigh noise is given by:*

$$p(z) = \begin{cases} \frac{2}{b}(z-a)e^{-(z-a)^2/b} & \text{for } z \geq a \\ 0 & \text{for } z < a \end{cases}$$

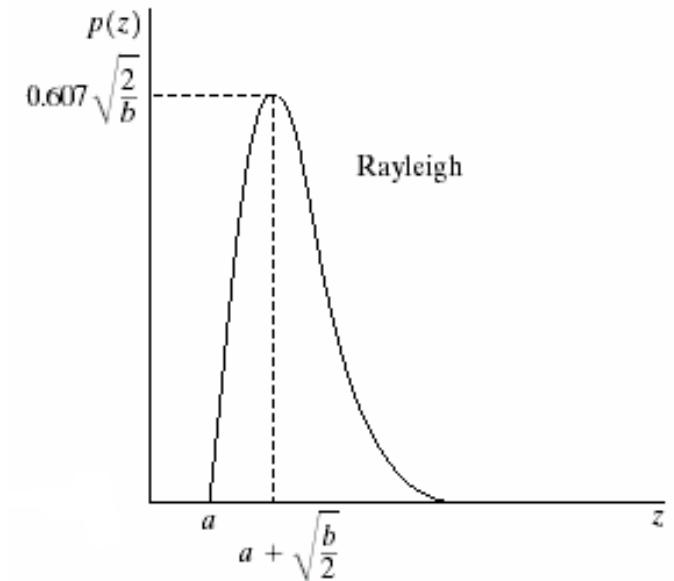
- *The mean of the density is:*

$$\mu = a + \sqrt{\pi b / 4}$$

- *The variance of the density is:*

$$\sigma^2 = \frac{b(4 - \pi)}{4}$$

- *Because of its skewed distribution, it can be useful for approximating the distribution of the images characterized by skewed histograms.*



# Image Restoration

## Noise models: Some important Noise PDFs

Erlang (Gamma) Noise:

- *The PDF of Gamma noise is given by:*

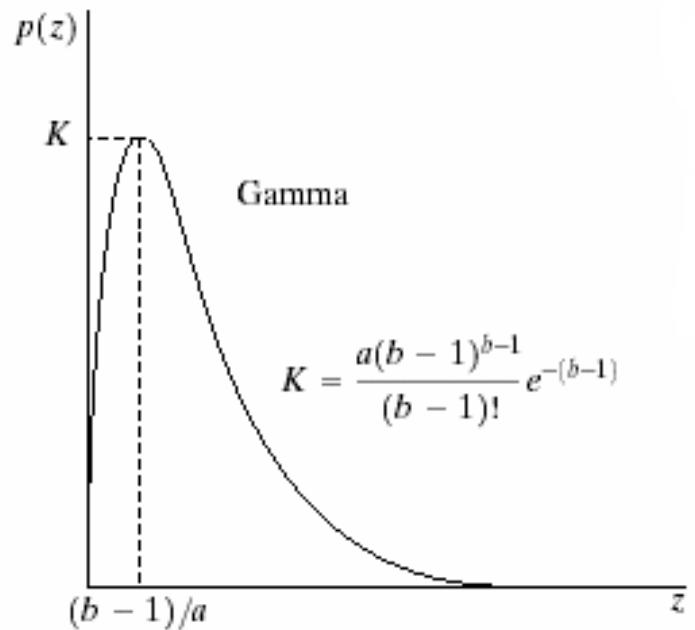
$$p(z) = \begin{cases} \frac{a^b z^{b-1}}{(b-1)!} e^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}$$

- *The mean of the density is:*

$$\mu = \frac{b}{a}$$

- *The variance of the density is:*

$$\sigma^2 = \frac{b}{a^2}$$



# Image Restoration

## Noise models: Some important Noise PDFs

### Exponential Noise:

- *The PDF of Exponential noise is given by:*

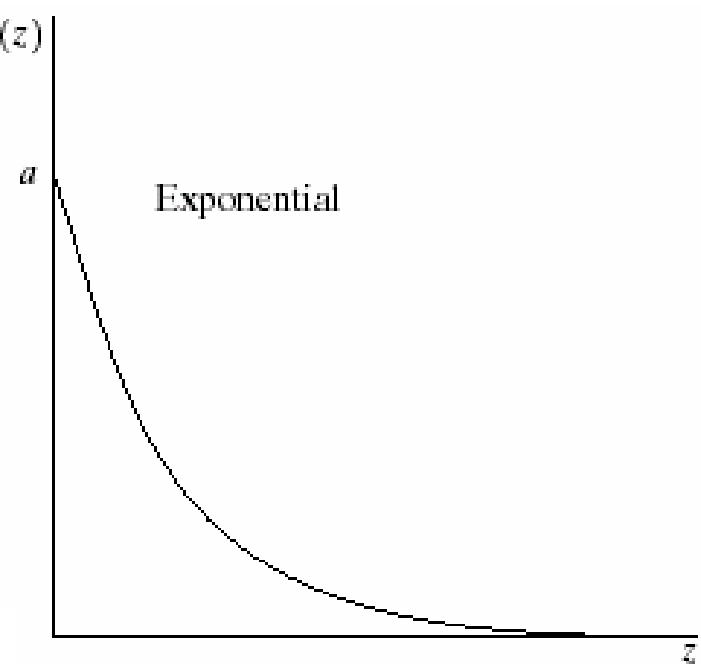
$$p(z) = \begin{cases} ae^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}$$

- *The mean of the density is:*

$$\mu = \frac{1}{a}$$

- *The variance of the density is:*

$$\sigma^2 = \frac{1}{a^2}$$



- *The PDF of the Exponential noise is the special case of the Gamma PDF, where  $b=1$ .*

# Image Restoration

## Noise models: Some important Noise PDFs

### Uniform Noise:

- *The PDF of Uniform noise is given by:*

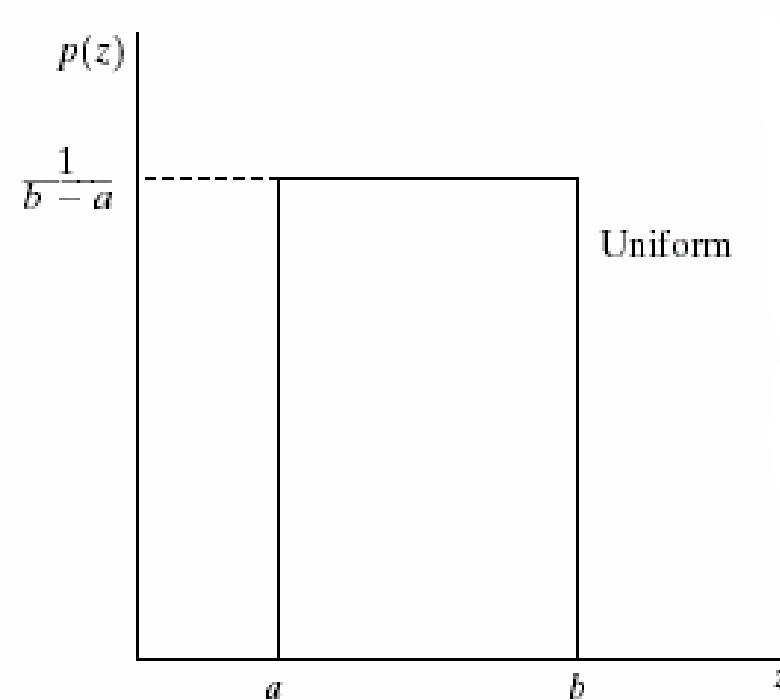
$$p(z) = \begin{cases} \frac{1}{b-a} & \text{if } a \leq z \leq b \\ 0 & \text{otherwise} \end{cases}$$

- *The mean of the density is:*

$$\mu = \frac{a+b}{2}$$

- *The variance of the density is:*

$$\sigma^2 = \frac{(b-a)^2}{12}$$



# Image Restoration

## Noise models: Some important Noise PDFs

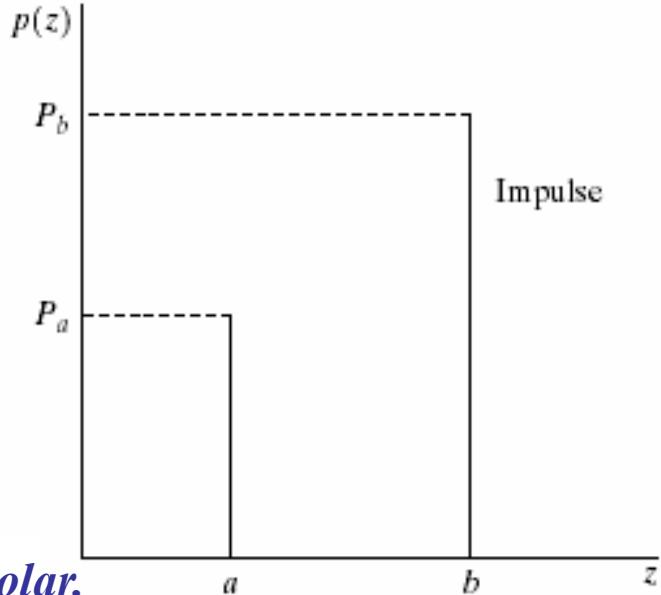
Impulse (salt-and-peper) Noise:

- *The PDF of (bipolar) impulse noise is given by:*

$$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$$

- *If  $b > a$  then, the gray-level  $b$  will appear as a light dot in the image.*
- *Otherwise the  $a$  will appear like a dark dot.*

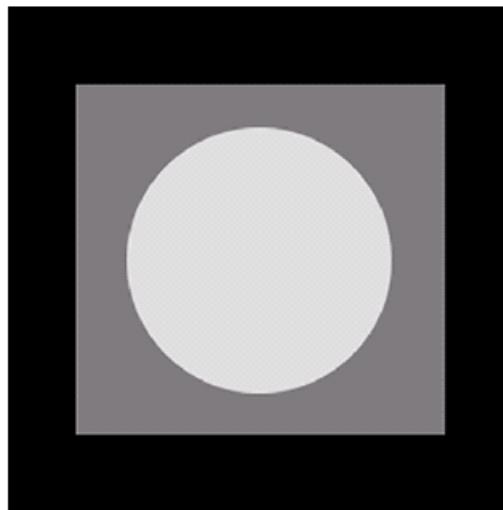
- *If  $P_a$  or  $P_b$  is zero, than impulse noise is called unipolar.*
- *Typically impulse noise can be positive or negative. The negative impulse is quantized as zero which is black (pepper) and the positive impulse is quantized as max intensity value (i.e. 255) which is white (salt).*



# Image Restoration

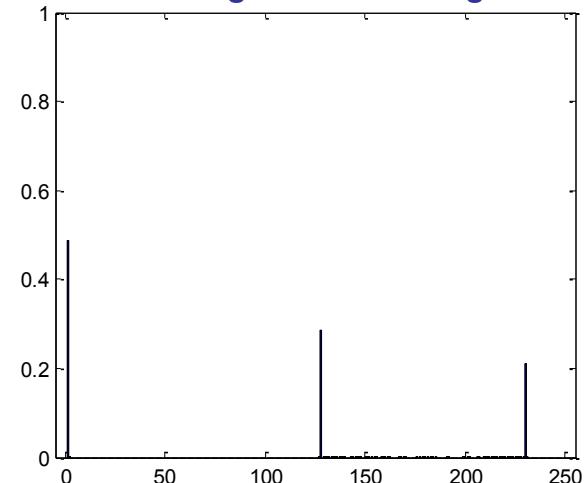
## Noise models: Some important Noise PDFs

- Consider the following image which contains 3 gray levels.



**FIGURE 5.3** Test pattern used to illustrate the characteristics of the noise PDFs shown in Fig. 5.2.

*Histogram of the image*



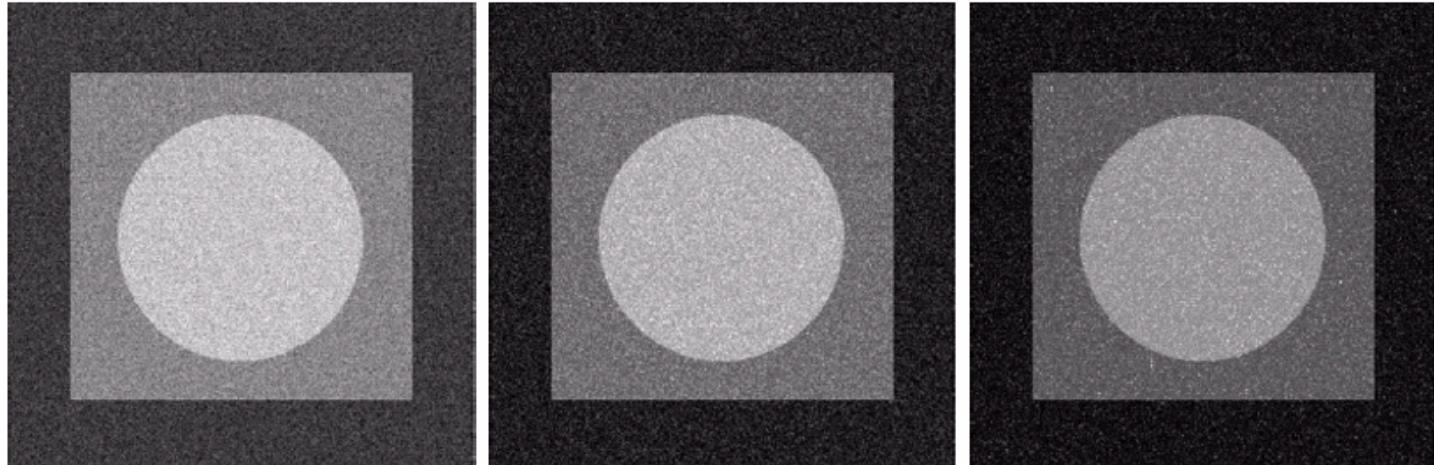
- If the only degradation in the image is additive noise then the image degraded by the additive noise is given by. Degradation function  $h(x,y)$  is ignored.

$$g(x, y) = f(x, y) + \eta(x, y)$$

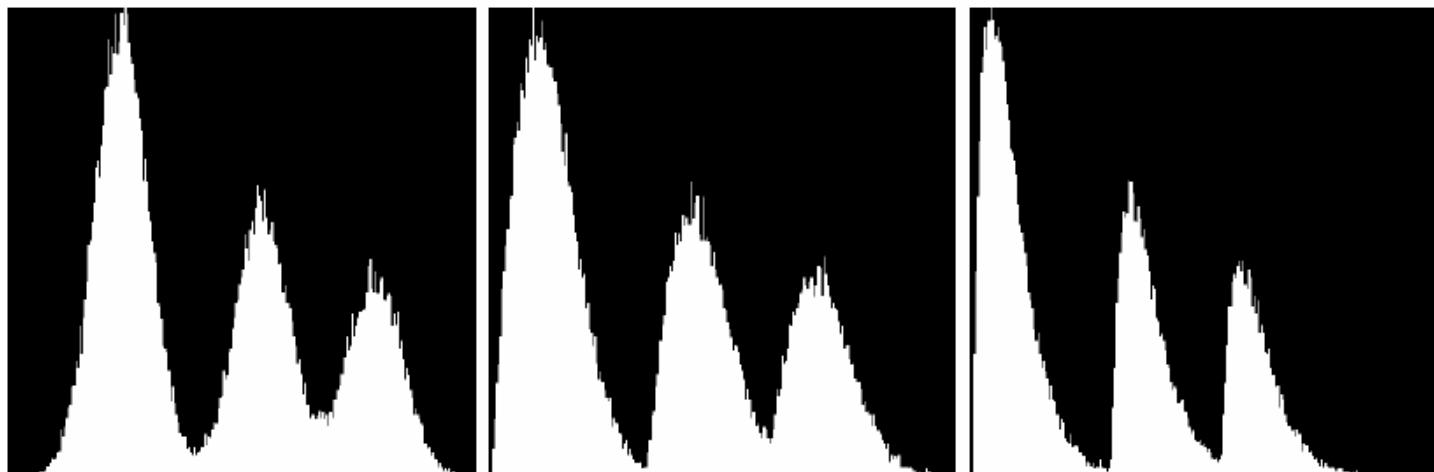
# Image Restoration

## Noise models: Some important Noise PDFs

*Noisy Image*



*Histogram of the  
Noisy Image*



Gaussian

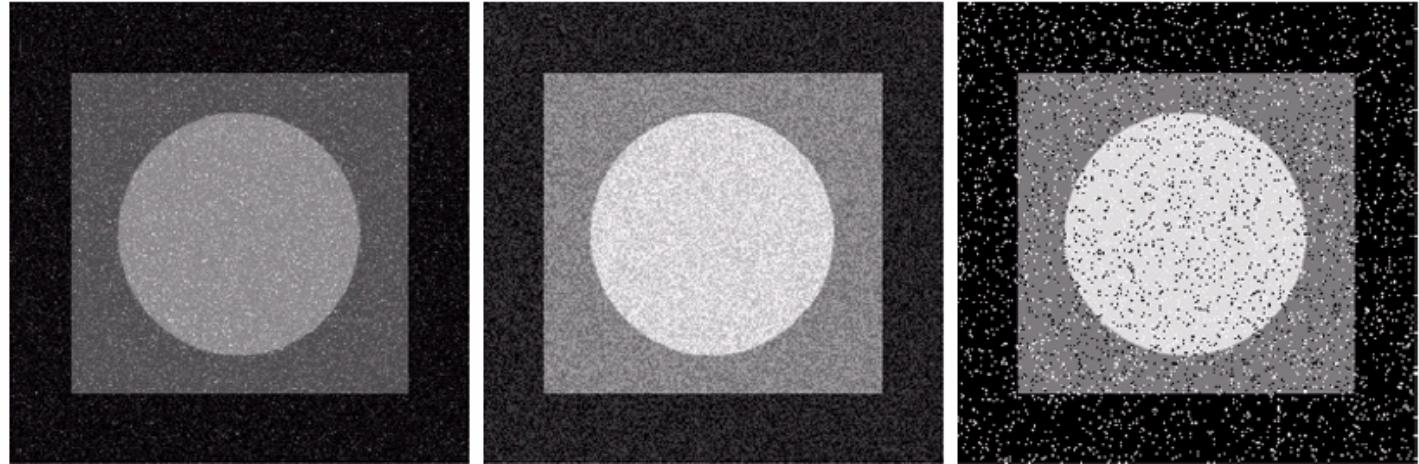
Rayleigh

Gamma

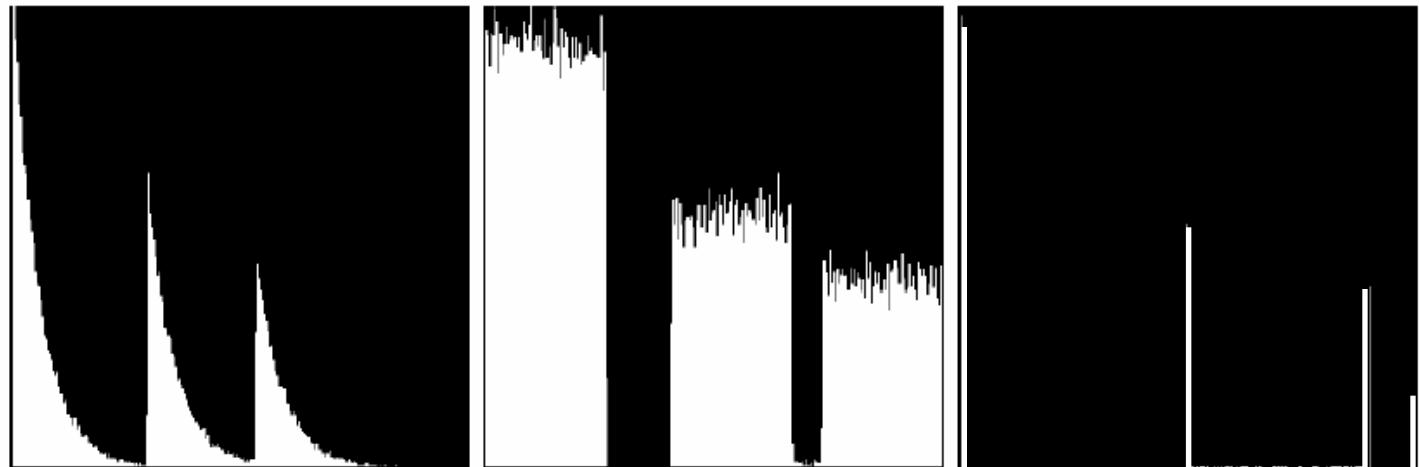
# Image Restoration

## Noise models: Some important Noise PDFs

*Noisy Image*



*Histogram of the  
Noisy Image*



**Exponential**

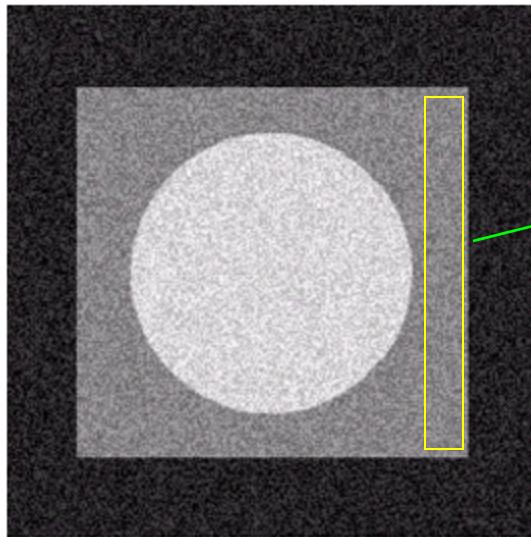
**Uniform**

**Salt & Pepper**

# Image Restoration

## Estimation of Noise Parameters models:

- *If the sensor device (i.e. camera) is available, then a solid grey area which is uniformly illuminated is acquired. Then, the noise can be estimated by analyzing the histogram of this area.*
- *If the sensor is not available and already generated images are to be considered, then PDF parameters can be obtained from small patches of reasonable constant gray level.*

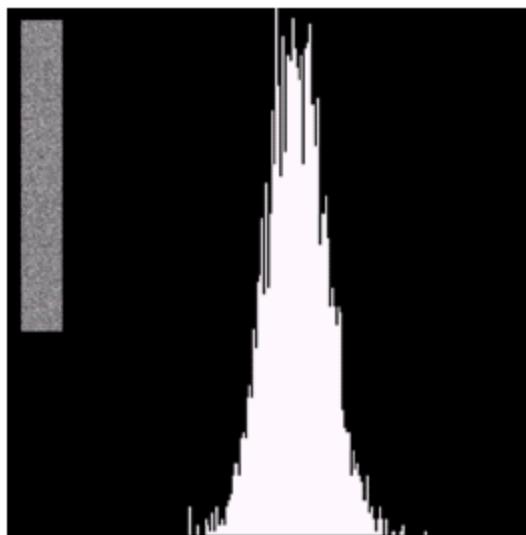


- *This image strip can be used to analyze the histogram.*
- *Then the mean and variance approximations can be extracted from this image strip.*

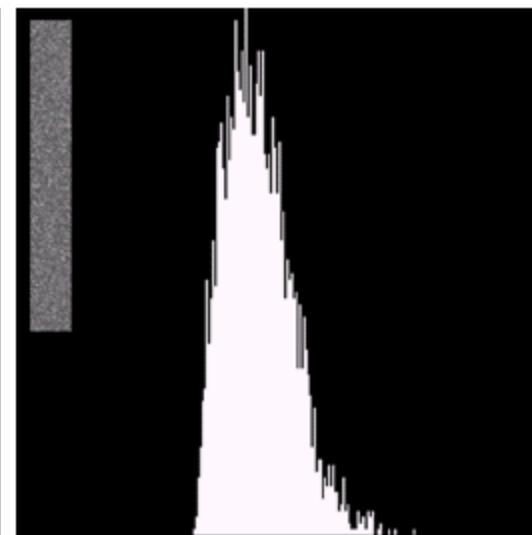
# Image Restoration

## Estimation of Noise Parameters models:

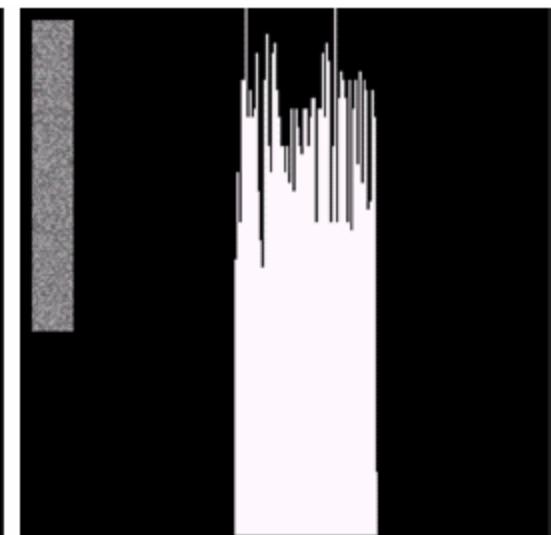
- *The following histograms are obtained from the image strips extracted from 3 different images.*
- *The shapes of the histograms gives an idea about the type of the noise distribution.*



Gaussian



Rayleigh



Uniform

- *Based on our observation, we can say that the first image strip contains Gaussian noise, the second one is Rayleigh and the third one is Uniform.*

# Image Restoration

## Estimation of Noise Parameters models:

- Once we know the PDF type, then we can estimate the mean and variance from the basic statistics by:

$$\mu = \sum_{z_i \in S} z_i p(z_i)$$

and

$$\sigma^2 = \sum_{z_i \in S} (z_i - \mu)^2 p(z_i)$$

- where,  $z_i$ 's are the gray-level values of the pixels in image strip  $S$ , and  $p(z_i)$  are the corresponding normalized histogram values.
- The PDF parameters of other noise models can be estimated by using the mean and variance approximations. The variables  $a$  and  $b$  can easily be calculated from the mean and variance.
- In the case of salt&pepper the mean and variance is not needed. The heights of the peaks corresponding to the white and black pixels are the estimates of the  $P_a$  and  $P_b$  respectively.

---

# Image Restoration

## Restoration in the presence of Noise:

- *If the only degradation in an image is the noise then:*

$$g(x, y) = f(x, y) + \eta(x, y)$$

*and*

$$G(u, v) = F(u, v) + N(u, v)$$

- *When, the distribution additive noise is estimated, we cannot simply subtract the noise terms from the noisy image.*
- *The method is to use spatial filtering for noise removal/reduction. Note that there is no point to transform into the frequency domain, as there is no convolution in the spatial domain.*
- *Only if there is a periodic noise, we can transform into the frequency domain.*

---

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

- **Mean Filters:** *There are four main types of noise reduction mean filters that can be used for image restoration/enhancement.*

- **Arithmetic Mean Filter:** *Let  $S_{xy}$  be the coordinates in a subimage window of size  $m \times n$  centered at point  $(x,y)$ . The value of the restored image at any point  $(x,y)$  is given by:*

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{s,t \in S_{xy}} g(s, t)$$

- *This operation can be implemented by a convolution mask in which all its components have a value  $1/mn$ .*
- *Local variations in the image is smoothed and noise is reduced as a result of blurring.*

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

### • Mean Filters:

- **Geometric Mean Filter:** Let  $S_{xy}$  be the coordinates in a subimage window of size  $m \times n$  centered at point  $(x,y)$ . The value of the restored image at any point  $(x,y)$  is given by:

$$\hat{f}(x, y) = \left[ \prod_{(s,t) \in S_{xy}} g(s, t) \right]^{\frac{1}{mn}}$$

- In this filter, each pixel is given by the product of the pixels in the subimage windows, raised to the power of  $1/mn$ .
- Achieves more smoothing, but loses less image details.

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

### • Mean Filters:

- **Harmonic Mean Filter:** Let  $S_{xy}$  be the coordinates in a subimage window of size  $m \times n$  centered at point  $(x,y)$ . The value of the restored image at any point  $(x,y)$  is given by:

$$\hat{f}(x, y) = \frac{mn}{\sum_{s,t \in S_{xy}} \frac{1}{g(s, t)}}$$

- Harmonic mean filter works well with the **salt noise** but fails for the **pepper noise**.
- It works well with other types of noise as well, such as **Gaussian noise**.

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

### • Mean Filters:

- *Contraharmonic Mean Filter: Let  $S_{xy}$  be the coordinates in a subimage window of size  $m \times n$  centered at point  $(x,y)$ . The value of the restored image at any point  $(x,y)$  is given by:*

$$\hat{f}(x, y) = \frac{\sum_{s,t \in S_{xy}} g(s, t)^{Q+1}}{\sum_{s,t \in S_{xy}} g(s, t)^Q}$$

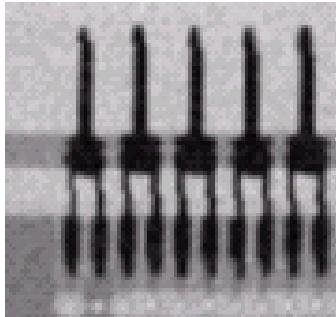
- *$Q$  is the order of the filter. This filter is well suited for reducing the effects of salt-and-pepper noise.*
- *For negative values of  $Q$ , it eliminates the salt noise and for positive values of  $Q$ , it eliminates the pepper noise.*
- *The filter becomes the arithmetic mean filter for  $Q=0$ , and harmonic mean filter for  $Q=-1$ .*

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

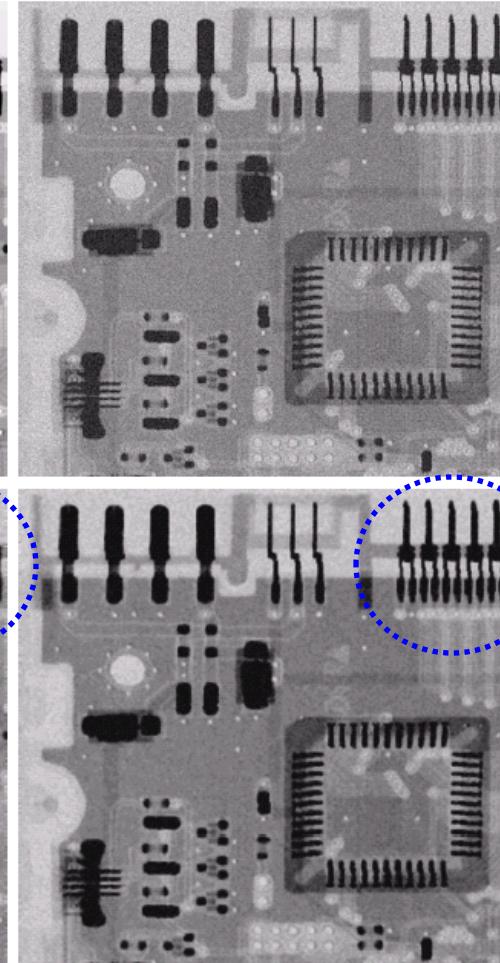
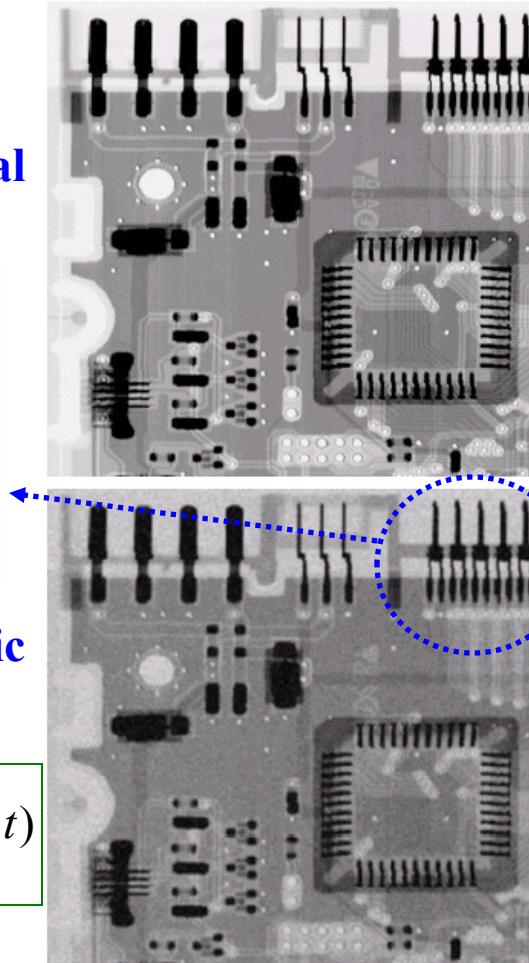
- Mean Filters:

Original



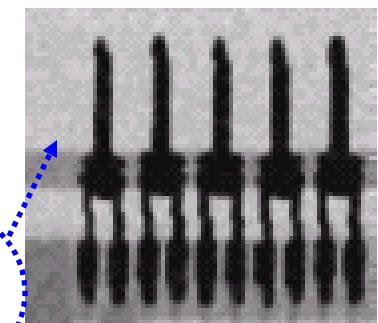
3x3 Arithmetic Mean Filter

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{s, t \in S_{xy}} g(s, t)$$



Additive Gaussian noise with

$$\mu=0, \sigma^2=400.$$



3x3 Geometric Mean Filter  
(Sharper details)

$$\hat{f}(x, y) = \left[ \prod_{s, t \in S_{xy}} g(s, t) \right]^{\frac{1}{mn}}$$

# Image Restoration

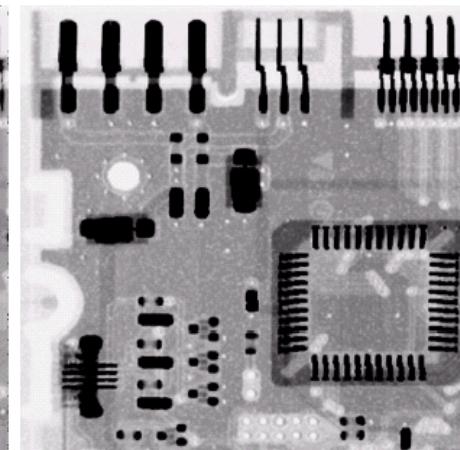
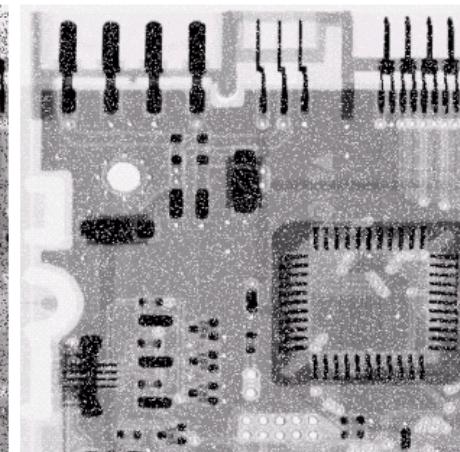
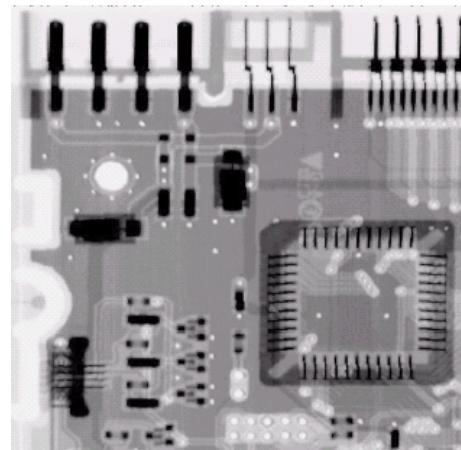
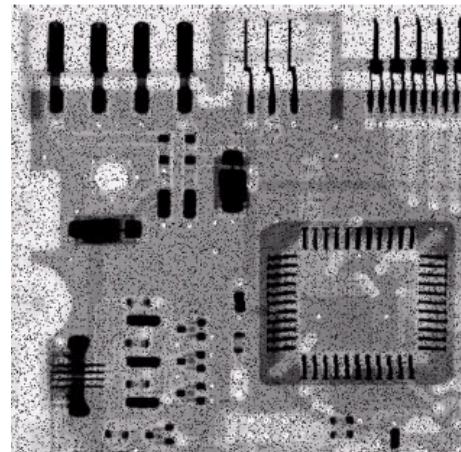
## Restoration in the presence of Noise: Only-Spatial Filtering

- Mean Filters:

Pepper Noise  
with  $P_p=0.1$

$$\hat{f}(x, y) = \frac{\sum_{s, t \in S_{xy}} g(s, t)^{Q+1}}{\sum_{s, t \in S_{xy}} g(s, t)^Q}$$

3x3  
Contraharmonic  
With Q=1.5  
(good for Pepper)



Salt Noise  
with  $P_s=0.1$

3x3  
Contraharmonic  
With Q=1.5  
(good for Salt)

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

- **Order Statistics Filters:** *The response of these spatial filters is based on the ordering/ranking the pixels in the filter mask.*

- **Median Filter:** *Let  $S_{xy}$  be the coordinates in a subimage window of size  $m \times n$  centered at point  $(x,y)$ . The value of the restored image at any point  $(x,y)$  is given by:*

$$\hat{f}(x, y) = \underset{(s,t) \in S_{xy}}{\text{median}} \{g(s, t)\}$$

- *The value of a pixel in is replaced by the median of the gray level in the neighborhood characterized by  $S_{xy}$  subimage.*
- *Provides less blurring than the other linear smoothing filters.*
- *Median filters are very effective in bipolar or unipolar impulse (**Salt&Pepper**) noise.*

---

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

### • Order Statistics Filters:

- **Max and Min Filters :** Let  $S_{xy}$  be the coordinates in a subimage window of size  $m \times n$  centered at point  $(x,y)$ . The value of the restored image at any point  $(x,y)$  is given by:

$$\hat{f}(x, y) = \max_{(s,t) \in S_{xy}} \{g(s, t)\}$$

- Useful for finding the brightest points in the image. Also reduces the pepper noise.

$$\hat{f}(x, y) = \min_{(s,t) \in S_{xy}} \{g(s, t)\}$$

- Useful for finding the darkest points in the image. Also reduces the salt noise.

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

### • Order Statistics Filters:

- *Midpoint filter : Let  $S_{xy}$  be the coordinates in a subimage window of size  $m \times n$  centered at point  $(x,y)$ . The value of the restored image at any point  $(x,y)$  is given by:*

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

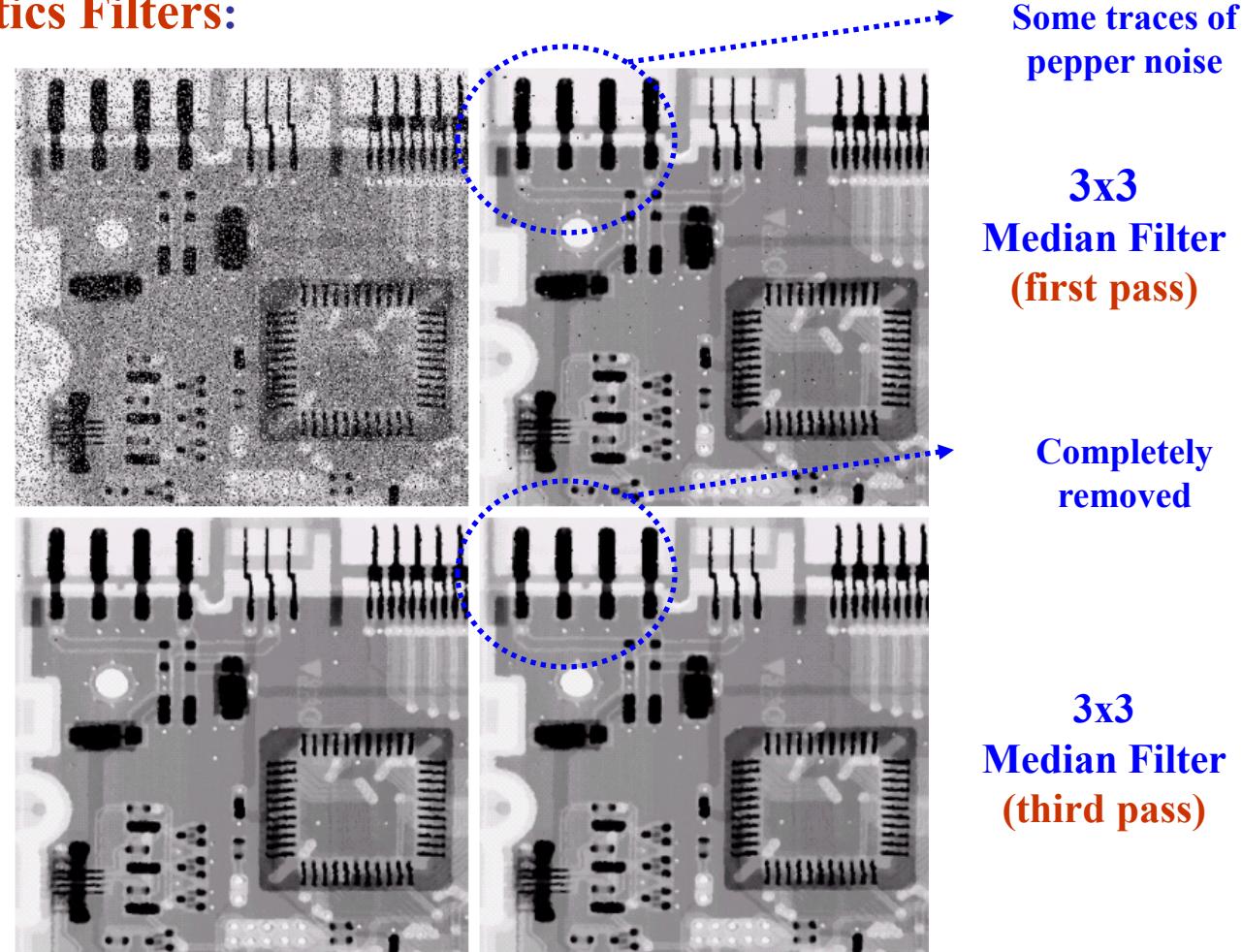
- *This filter combines the order statistics and averaging. Works best for Gaussian and uniform noise.*

# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

- Order Statistics Filters:

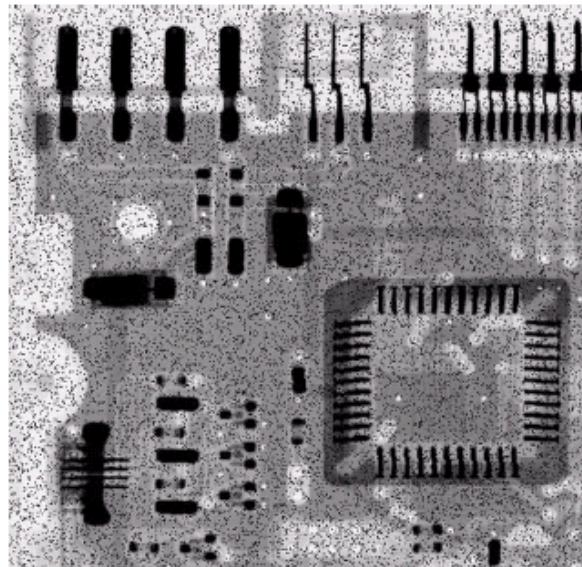
Salt & Pepper  
Noise with  
 $P_a = P_b = 0.1$



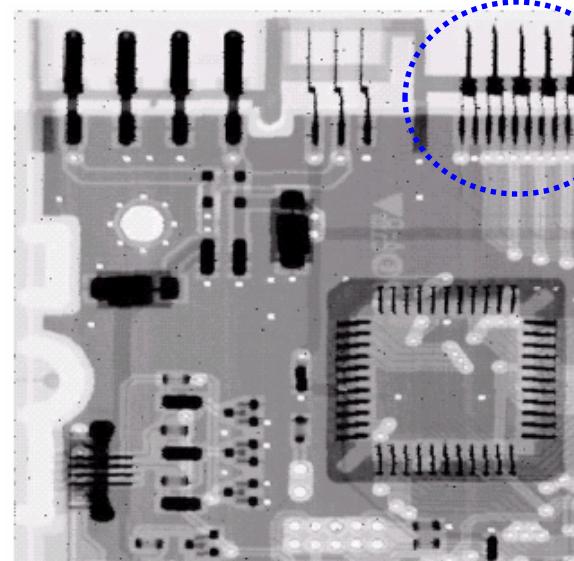
# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

- Order Statistics Filters:



Pepper Noise  
with  $P_b=0.1$



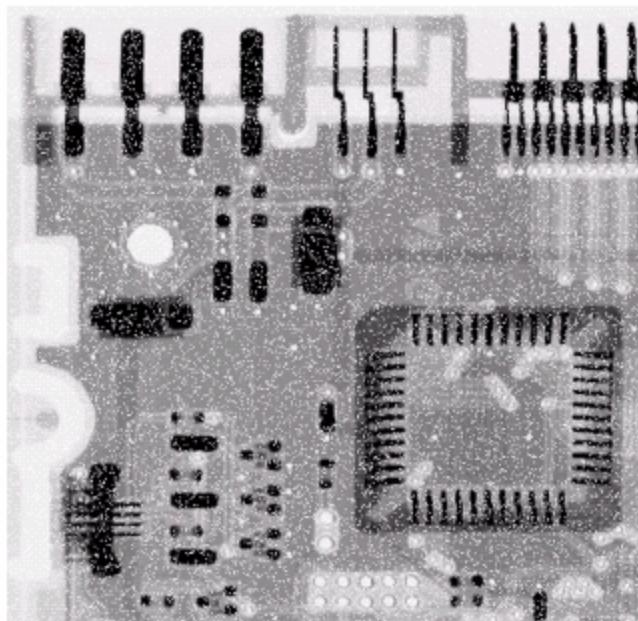
3x3  
Max Filter  
Good for pepper

Some dark border pixels  
are removed

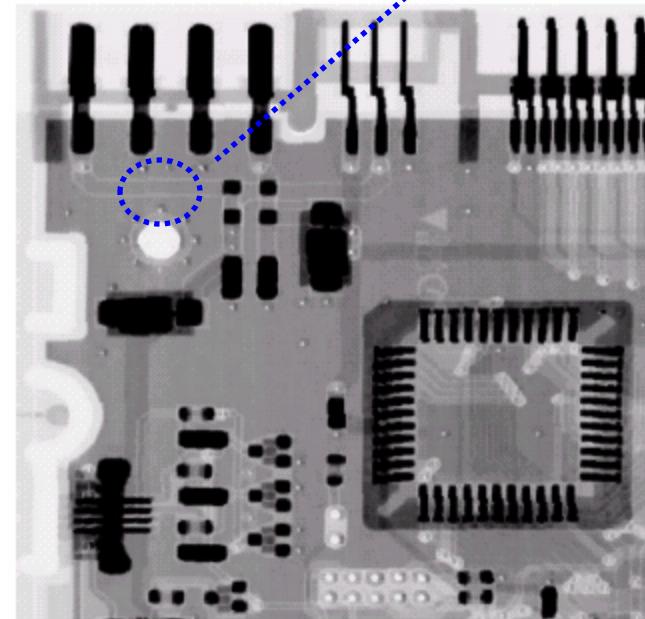
# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

- Order Statistics Filters:



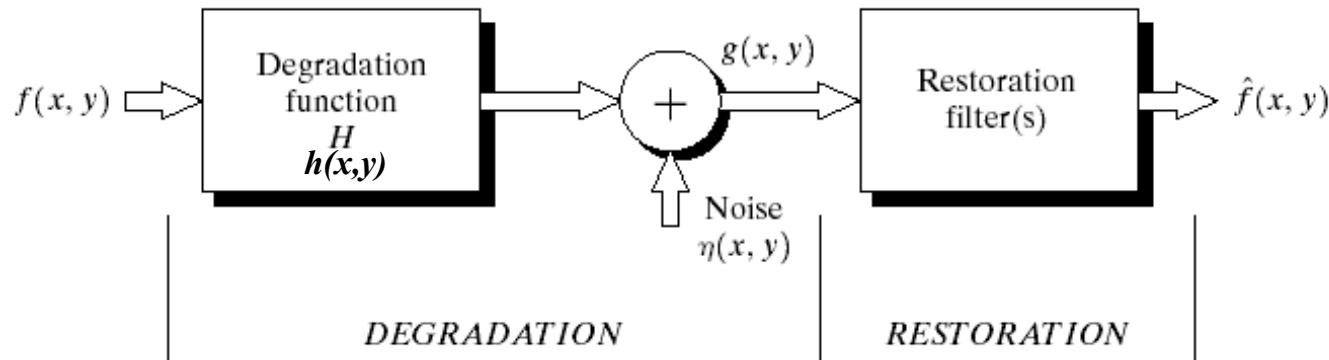
Salt Noise with  
 $P_a=0.1$



3x3  
Min Filter  
Good for Salt

# Image Restoration

## Image Degradation and Restoration



**FIGURE 5.1** A model of the image degradation/restoration process.

- **Image Degradation Model:** *Spatial domain representation can be modeled by:*

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y)$$

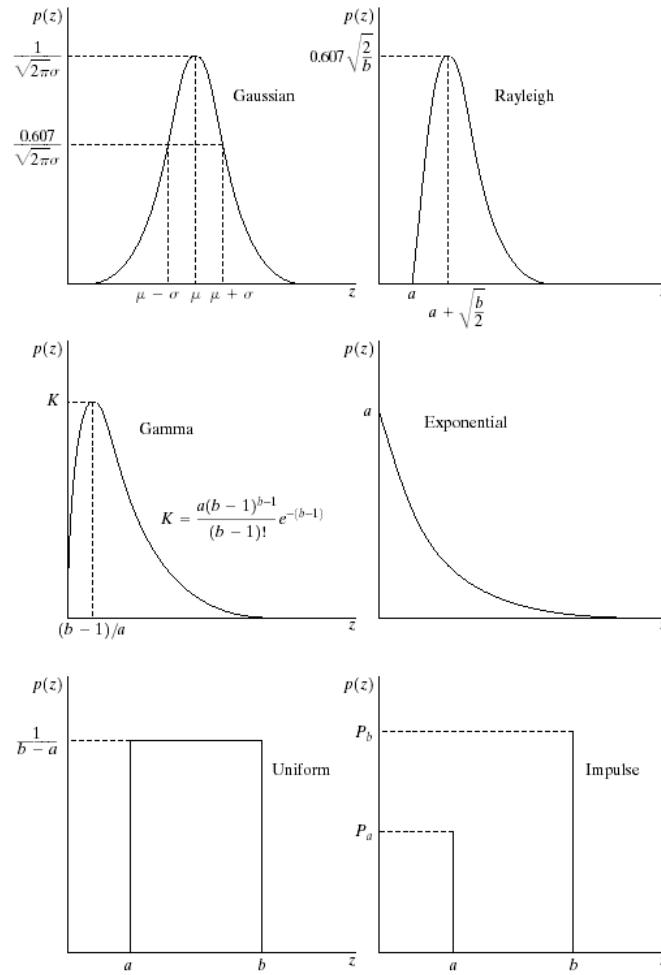
- *Frequency domain representation can be modeled by:*

$$G(u, v) = H(u, v)F(u, v) + N(u, v)$$

# Image Restoration

## Common Noise Models:

- *Most types of noise are modeled as known probability density functions*
- *Noise model is decided based on understanding of the physics of the sources of noise.*
  - Gaussian: poor illumination
  - Rayleigh: range image
  - Gamma/Exp: laser imaging
  - Impulse: faulty switch during imaging,
  - Uniform is least used.
- *Parameters can be estimated based on histogram on small flat area of an image*



# Image Restoration

**Restoration methods:** The following methods are used in the presence of noise.

- **Mean filters**
  - *Arithmetic mean filter*
  - *Geometric mean filter*
  - *Harmonic mean filter*
  - *Contra-harmonic mean filter*
- **Order statistics filters**
  - *Median filter*
  - *Max and min filters*
  - *Mid-point filter*
- **Adaptive filters**
  - *Adaptive local noise reduction filter*
  - *Adaptive median filter*

# Image Restoration

## Restoration in the presence of Noise:

Adaptive Local Noise Reduction Filter: Mean and variance are the simplest statistical measures of a random noise.

- **Mean** gives the measure of the average gray level in a local region,
- **Variance** gives the measure of the average contrast in the region.

• Consider a filter operating in a local region,  $S_{xy}$ , where the response of the filter at any point  $(x,y)$  depends on:

- a)  $g(x,y)$ , the value of the noisy image at  $(x,y)$
- b)  $\sigma^2_\eta$ , the additive noise variance,
- c)  $m_L$ , local mean of pixels in  $S_{xy}$
- d)  $\sigma^2_L$ , the local variance in  $S_{xy}$

# Image Restoration

## Restoration in the presence of Noise:

### Adaptive Local Noise Reduction Filter:

• Consider an adaptive filter where, the following conditions are satisfied:

1. If  $\sigma_\eta^2 = 0$ , the filter should return  $g(x,y)$ , zero-noise case [  $f(x,y)=g(x,y)$  ].
2. If  $\sigma_L^2 \gg \sigma_\eta^2$ , the filter should return a value close to  $g(x,y)$ . High local variance is associated with edges and should be preserved.
3. If  $\sigma_L^2 = \sigma_\eta^2$ , return arithmetic mean of  $S_{xy}$ . This occurs when local noise has the same properties of the entire image. Averaging simply reduces the noise.

• According to the preceding assumptions the filter response can be modeled as:

$$\hat{f}(x, y) = g(x, y) - \frac{\sigma_\eta^2}{\sigma_L^2} [g(x, y) - m_L]$$

• The only unknown parameter in the above adaptive filter model is  $\sigma_\eta^2$ , The other parameters can be calculated at the local neighborhood of  $S_{xy}$ .

# Image Restoration

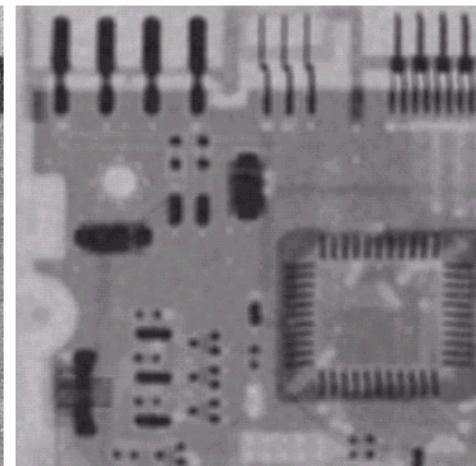
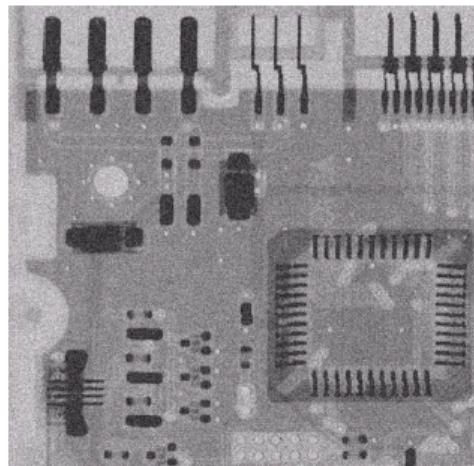
## Restoration in the presence of Noise:

### Adaptive Local Noise Reduction Filter:

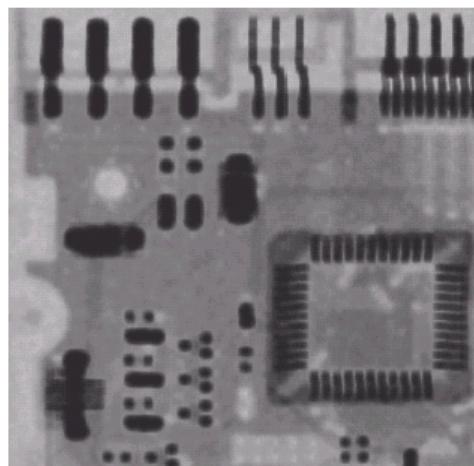
a b  
c d

Image corrupted  
by Gaussian Noise  
with

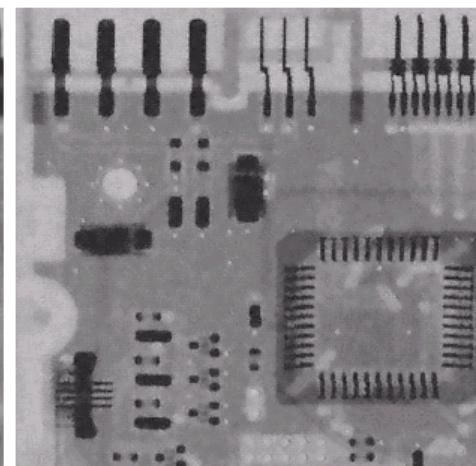
$$\sigma^2_{\eta} = 1000$$
$$\mu = 0$$



7x7  
Arithmetic Mean  
Filter



7x7  
Geometric Mean  
Filter



7x7  
Adaptive Filter

# Image Restoration

## Restoration in the presence of Noise:

**Adaptive Median Filter:** *Adaptive median filter can filter impulse noise with very high probabilities. Additionally smoothes the nonimpulse noise which is not the feature of a traditional median filter.*

- *The filter uses the following parameters in the neighborhood of  $S_{xy}$  :*

$z_{min}$  = minimum gray level value in  $S_{xy}$ ,

$z_{max}$  = maximum gray level value in  $S_{xy}$ ,

$z_{med}$  = median of gray levels in  $S_{xy}$ ,

$z_{xy}$  = gray level at coordinates (x,y).

$S_{max}$  = maximum allowed size of  $S_{xy}$ .

- *Note that unlike the other filters the size of  $S_{xy}$  increases during the filtering operation.*
- *Changing size of the filter mask does not change the fact that the output of the filter is still a single value centering the mask.*

# Image Restoration

## Restoration in the presence of Noise:

### Adaptive Median Filter:

- *The Adaptive median filtering Algorithm: Two levels exist (Levels A and B)*

#### *Level A:*

$$A1 = z_{med} - z_{min}$$

$$A2 = z_{med} - z_{max}$$

*if  $A1 > 0$  and  $A2 < 0$ , goto Level B*

*else increase the window size*

*if window size  $< S_{max}$  repeat Level A*

*else output  $z_{xy}$*

#### *Level B:*

$$B1 = z_{xy} - z_{min}$$

$$B2 = z_{xy} - z_{max}$$

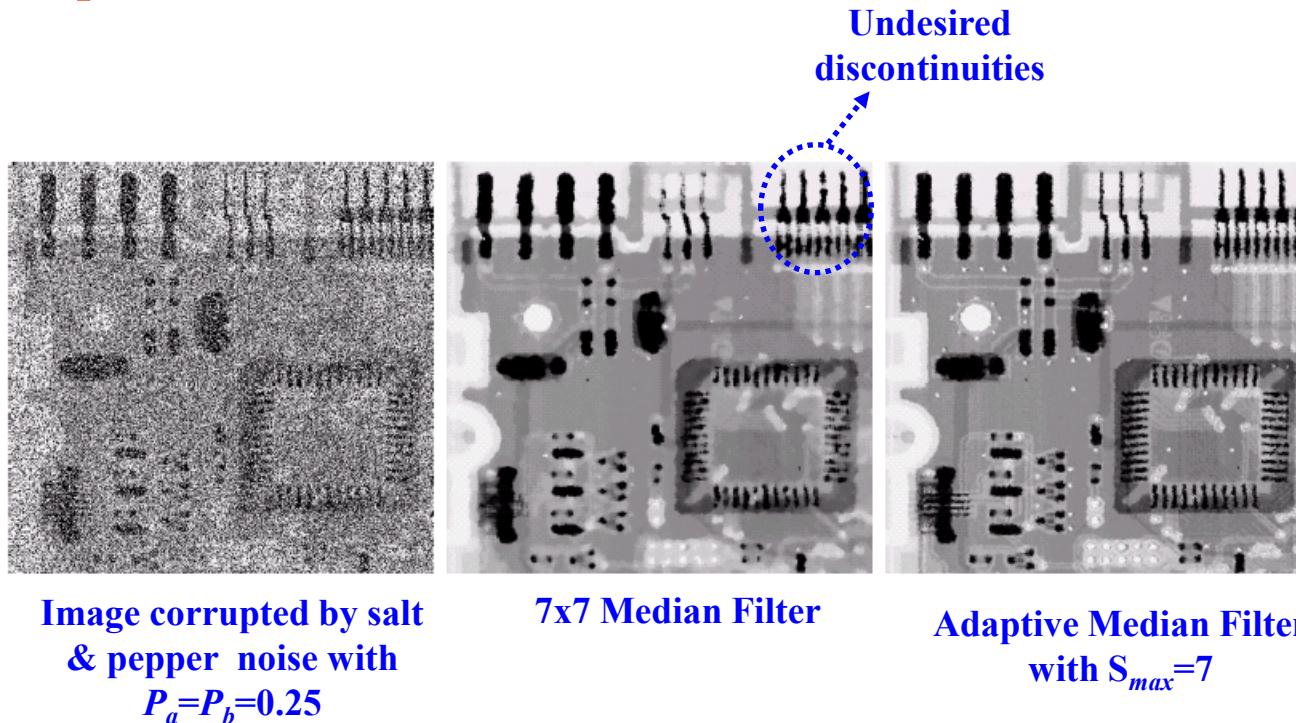
*if  $B1 > 0$  and  $B2 < 0$ , output  $z_{xy}$*

*else output  $z_{med}$*

# Image Restoration

## Restoration in the presence of Noise:

### Adaptive Median Filter:

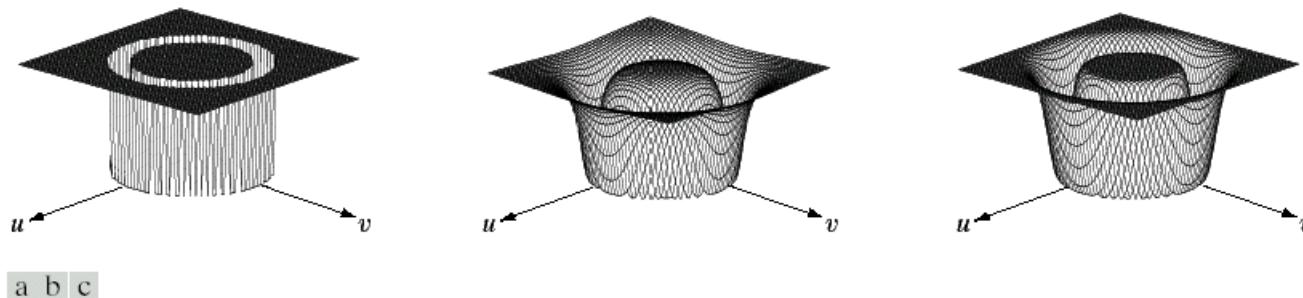


# Image Restoration

## Restoration in the presence of Noise:

### Periodic Noise Removal by Frequency Domain Filtering:

- *Bandreject, bandpass and notch filters can be used for periodic noise removal.*
- *Bandreject filters remove/attenuate a band of frequencies about the origin of the Fourier transform.*



**FIGURE 5.15** From left to right, perspective plots of ideal, Butterworth (of order 1), and Gaussian bandreject filters.

# Image Restoration

## Restoration in the presence of Noise:

### Periodic Noise Removal by Frequency Domain Filtering:

- An Ideal Bandreject filter is given by:

$$H(u, v) = \begin{cases} 1 & \text{if } D(u, v) < D_0 - \frac{W}{2} \\ 0 & \text{if } D_0 - \frac{W}{2} \leq D(u, v) \leq D_0 + \frac{W}{2} \\ 1 & \text{if } D(u, v) \geq D_0 + \frac{W}{2} \end{cases}$$

- $W$  is the width of the band
- $D_0$  is the radial center
- $D(u, v)$  distance from the origin.

- Butterworth Bandreject filter is given by:

$$H(u, v) = \frac{1}{1 + \left[ \frac{D(u, v)W}{D^2(u, v) - D_0^2} \right]^{2n}}$$

- $W$  is the width of the band
- $D_0$  is the radial center
- $D(u, v)$  distance from the origin.
- $n$  is the order of the filter

# Image Restoration

## Restoration in the presence of Noise:

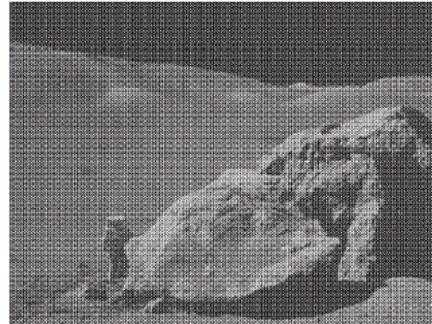
### Periodic Noise Removal by Frequency Domain Filtering:

- *Gaussian Bandreject filter is given by:*

$$H(u,v) = 1 - e^{-\frac{1}{2} \left[ \frac{D^2(u,v) - D_0^2}{D(u,v)W} \right]^2}$$

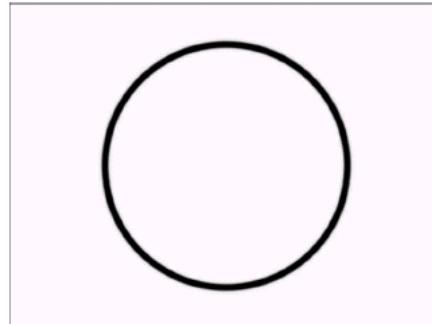
- *W is the width of the band*
- *D<sub>0</sub> is the radial center*
- *D(u,v) distance from the origin.*

Image corrupted by sinusoidal noise



Spectrum of corrupted image

Butterworth Bandreject Filter (n=4)



Filtered Image

# Image Restoration

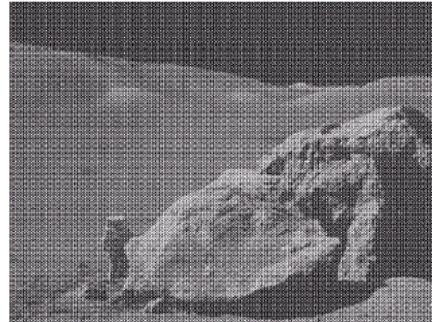
## Restoration in the presence of Noise:

### Periodic Noise Removal by Frequency Domain Filtering:

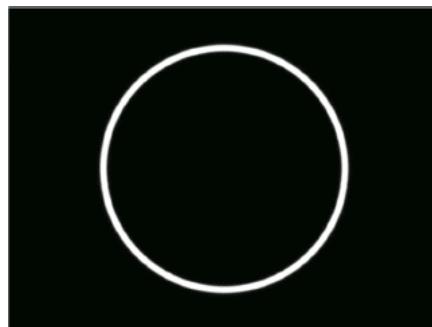
- Bandpass filters perform the opposite function of the bandreject filters and the filter transfer function of a bandpass filter is given by:

$$H(u, v)_{bp} = 1 - H(u, v)_{br}$$

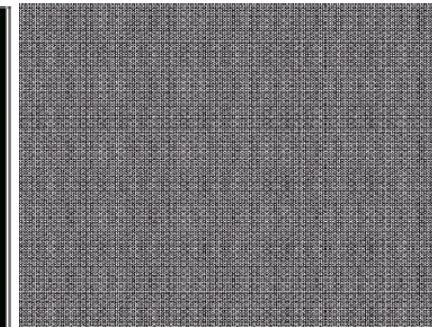
Image corrupted by sinusoidal noise



Spectrum of corrupted image



Butterworth Bandpass filter



Noise Image

# Image Restoration

## Restoration in the presence of Noise:

### Periodic Noise Removal by Frequency Domain Filtering:

- **Notch filters** rejects/passes frequencies in a predefined neighborhoods about the center frequency.
- Notch filters appear in symmetric pairs due to the symmetry of the Fourier transform.
- The transfer function of **ideal notch filter** of radius of  $D_0$ , with centers at  $(u_0, v_0)$  and by symmetry at  $(-u_0, -v_0)$ , is given by:

$$H(u, v) = \begin{cases} 0 & \text{if } D_1(u, v) \leq D_0 \quad \text{or} \quad D_2(u, v) \leq D_0 \\ 1 & \text{otherwise} \end{cases}$$

$$D_1(u, v) = \left[ (u - M/2 - u_0)^2 + (v - N/2 - v_0)^2 \right]^{1/2}$$

$$D_2(u, v) = \left[ (u - M/2 + u_0)^2 + (v - N/2 + v_0)^2 \right]^{1/2}$$

# Image Restoration

## Restoration in the presence of Noise:

### Periodic Noise Removal by Frequency Domain Filtering:

#### •Notch filters:

•The transfer function of **Butterworth notch filter** of order  $n$  and of radius of  $D_0$ , with centers at  $(u_0, v_0)$  and by symmetry at  $(-u_0, -v_0)$ , is given by:

$$H(u, v) = \frac{1}{1 + \left[ \frac{D_0^2}{D_1^2(u, v) D_2^2(u, v)} \right]^n}$$

The transfer function of **Gaussian notch filter** of radius of  $D_0$ , with centers at  $(u_0, v_0)$  and by symmetry at  $(-u_0, -v_0)$ , is given by:

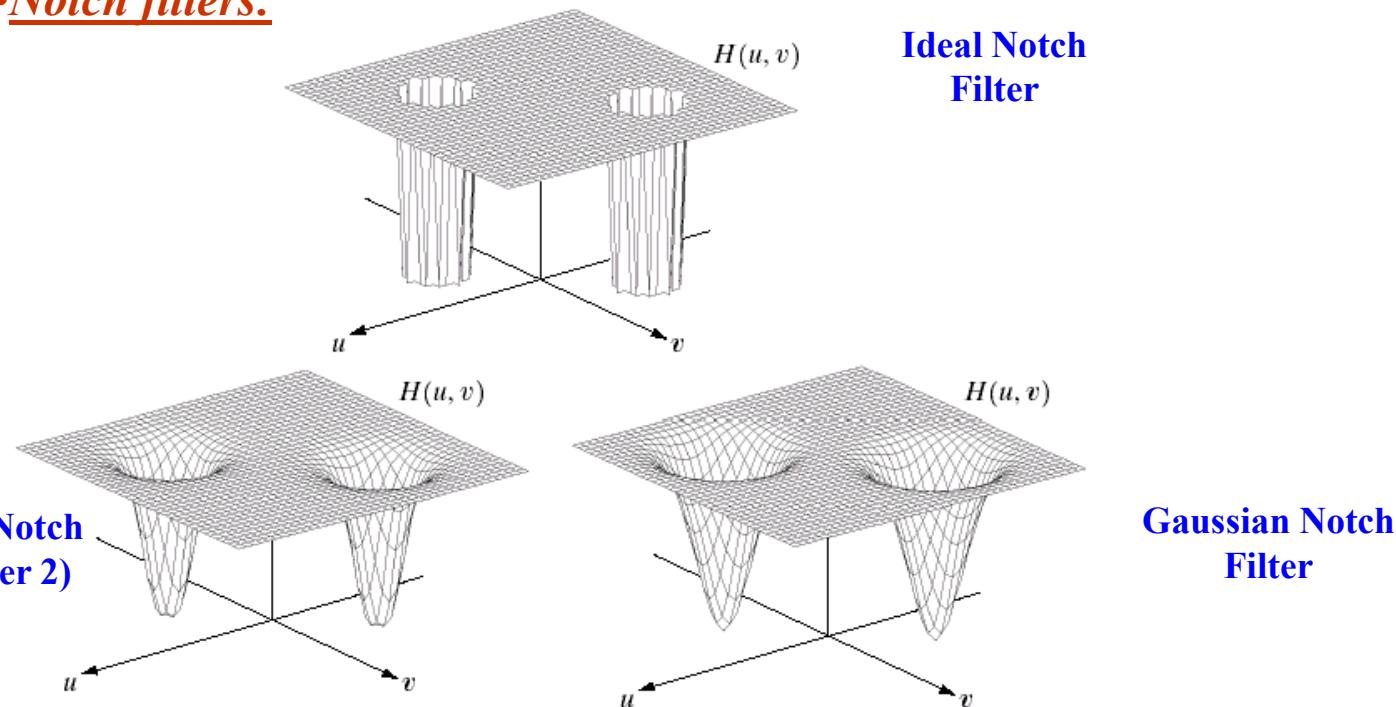
$$H(u, v) = 1 - e^{-\frac{1}{2} \left[ \frac{D_1^2(u, v) - D_2^2(u, v)}{D_0^2} \right]}$$

# Image Restoration

## Restoration in the presence of Noise:

### Periodic Noise Removal by Frequency Domain Filtering:

- Notch filters:



# Image Restoration

## Restoration in the presence of Noise:

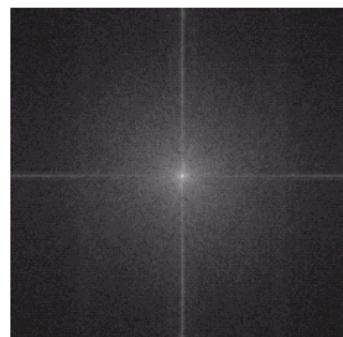
### Periodic Noise Removal by Frequency Domain Filtering:

- Notch filters:

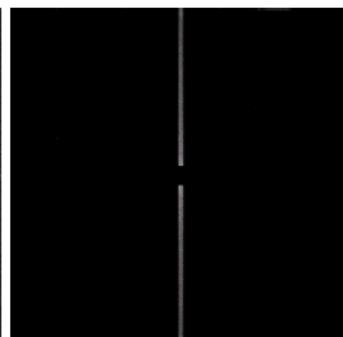


Original Noisy image with undesired horizontal scanning lines

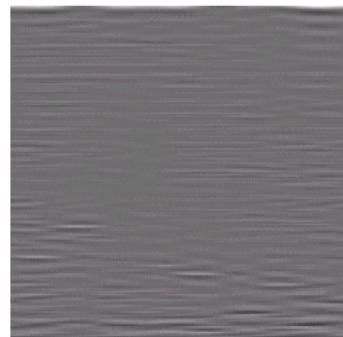
Spectrum of the image



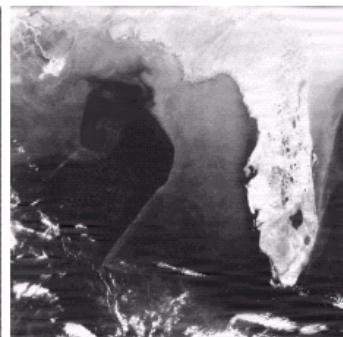
A simple ideal Notch filter along the vertical axis



filtered image free of horizontal scanning lines.



The filtered noise pattern  
Corresponding the horizontal artifacts



---

# Image Restoration

## Estimating the Degradation Function :

*There are 3 principal methods of estimating the degradation function for Image Restoration: 1) Observation, 2) Experimentation, 3) Mathematical Modeling.*

*The degradation function  $H$  can be estimated by visually looking into a small section of the image containing simple structures, with strong signal contents, like part an object and the background. Given a small subimage  $g_s(x,y)$ , we can manually (i.e. filtering) remove the degradation in that region with an estimated subimage  $\hat{f}_s(x,y)$  and assuming that the additive noise is negligible in such an area with a strong signal content.*

$$H_s(u,v) = \frac{G_s(u,v)}{\hat{F}_s(u,v)}$$

*Having  $H_s(u,v)$  estimated for such a small subimage, the shape of this degradation function can be used to get an estimation of  $H(u,v)$  for the entire image.*

# Image Restoration

## Estimating the Degradation Function :

*There are 3 principal methods of estimating the degradation function for Image Restoration: 1) Observation, 2) Experimentation, 3) Mathematical Modelling.*

### • Estimation by Image Experimentation:

- *If we have the acquisition device producing degradation on images, we can use the same device to obtain an accurate estimation of the degradation.*
- *This can be achieved by applying an impulse (bright dot) as an input image . The Fourier transform of an impulse is constant, therefore.*

$$H(u, v) = \frac{G(u, v)}{A}$$

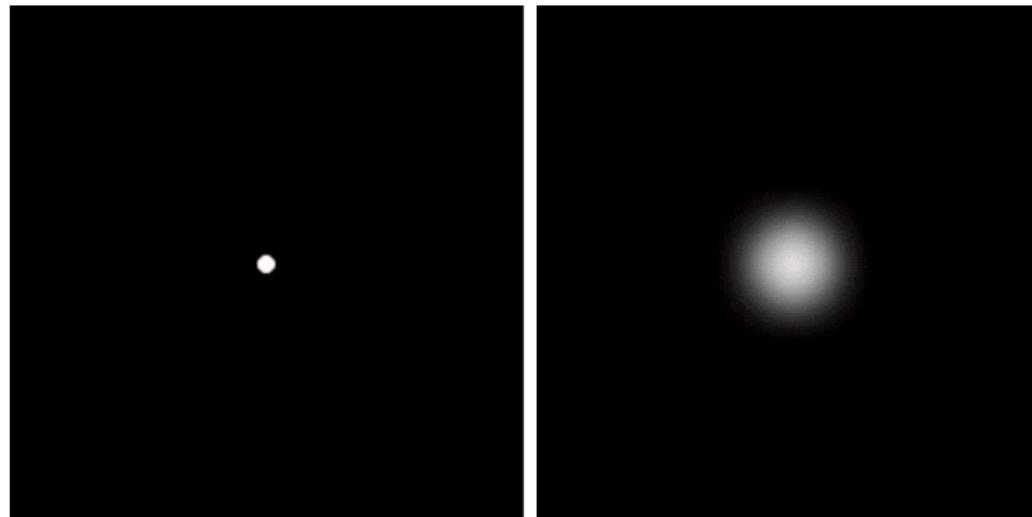
*Where, A is a constant describing the strength of the impulse. Note that the effect of noise on an impulse is negligible.*

# Image Restoration

## Estimating the Degradation Function :

*There are 3 principal methods of estimating the degradation function for Image Restoration: 1) Observation, 2) Experimentation, 3) Mathematical Modelling.*

- Estimation by Image Experimentation:



a b

**FIGURE 5.24**  
Degradation estimation by impulse characterization.  
(a) An impulse of light (shown magnified).  
(b) Imaged (degraded) impulse.

Impulse Image

Degraded Impulse Image  
consider as  $h(x,y)$

*Simply take the Fourier transform of the degraded image and after normalization by a constant A, use it as the estimate of the degradation function  $H(u,v)$ .*

# Image Restoration

## Estimating the Degradation Function :

*There are 3 principal methods of estimating the degradation function for Image Restoration: 1) Observation, 2) Experimentation, 3) Mathematical Modelling.*

- Estimation by Mathematical Modeling: Sometimes the environmental conditions that causes the degradation can be modeled by mathematical formulation. For example the atmospheric turbulence can be modeled by:

$$H(u, v) = e^{-k(u^2 + v^2)^{5/6}}$$

*k is a constant that depends on the nature of the Turbulence*

- This equation is similar to Gaussian LPF and would produce blurring in the image according to the values of k. For example if k=0.0025, the model represents severe turbulence, if k=0.001, the model represents mild turbulence and if k=0.00025, the model represents low turbulence.
- Once a reliable mathematical model is formed the effect of the degradation can be obtained easily.

# Image Restoration

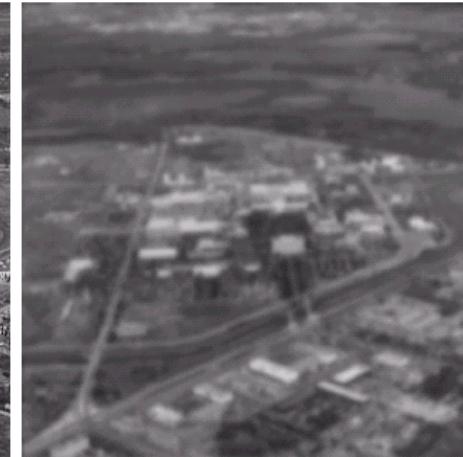
## Estimating the Degradation Function :

- *Estimation by Mathematical Modeling: Illustration of the atmospheric turbulence model*

Negligible turbulence



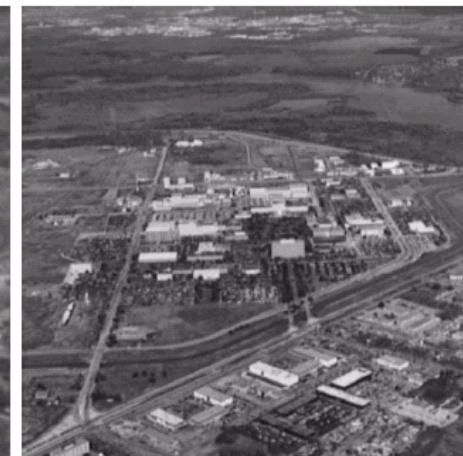
Severe turbulence  
 $k=0.0025$



Mild turbulence  
 $k=0.001$



Low turbulence  
 $k=0.00025$



# Image Restoration

## Estimating the Degradation Function :

• **Estimation by Mathematical Modeling:** In some applications the mathematical model can be derived by treating that the image is blurred by uniform linear motion between the image and the sensor during image acquisition. The motion blur can be modeled as follows:

- Let  $f(x,y)$  be subject to motion in  $x$ - and  $y$ -direction by time varying motion components  $x_0(t)$  and  $y_0(t)$ .
- The total exposure is obtained by integrating the instantaneous exposure over the time interval during the shutter of the imaging device is open.
- If  $T$  is the duration of the exposure, than

$$g(x, y) = \int_0^T f[x - x_0(t), y - y_0(t)] dt \quad g(x, y) \text{ is the blurred image}$$

$$\begin{aligned} G(u, v) &= \int_{-\infty}^{-\infty} \int_{-\infty}^{-\infty} g(x, y) e^{-j2\pi(ux+vy)} dx dy \\ &= \int_{-\infty}^{-\infty} \int_{-\infty}^{-\infty} \left[ \int_0^T f[x - x_0(t), y - y_0(t)] dt \right] e^{-j2\pi(ux+vy)} dx dy \end{aligned}$$

# Image Restoration

## Estimating the Degradation Function :

•*Estimation by Mathematical Modeling:*

•*Reversing the order of integration yields:*

$$G(u, v) = \int_0^T \left[ \int_{-\infty}^{-\infty} \int_{-\infty}^{-\infty} f[x - x_0(t), y - y_0(t)] e^{-j2\pi(ux+vy)} dx dy \right] dt$$

•*Using the translation property of the Fourier Transform, the inner part can be simplified,*

$$\begin{aligned} G(u, v) &= \int_0^T F(u, v) e^{-j2\pi(ux_0(t)+vy_0(t))} dt = \\ &= F(u, v) \int_0^T e^{-j2\pi(ux_0(t)+vy_0(t))} dt \end{aligned}$$

•*Then,*

$$H(u, v) = \int_0^T e^{-j2\pi(ux_0(t)+vy_0(t))} dt$$

---

# Image Restoration

## Estimating the Degradation Function :

- *Estimation by Mathematical Modeling:*

- *By assuming that the linear uniform motion is in x-direction only at a rate of  $x_0(t)=at/T$ , the image covers a distance , when  $t=T$ .*

$$\begin{aligned} H(u, v) &= \int_0^T e^{-j2\pi u x_0(t)} dt = \int_0^T e^{-j2\pi u a t/T} dt \\ &= \frac{T}{\pi u a} \sin(\pi u a) e^{-j\pi u a} \end{aligned}$$

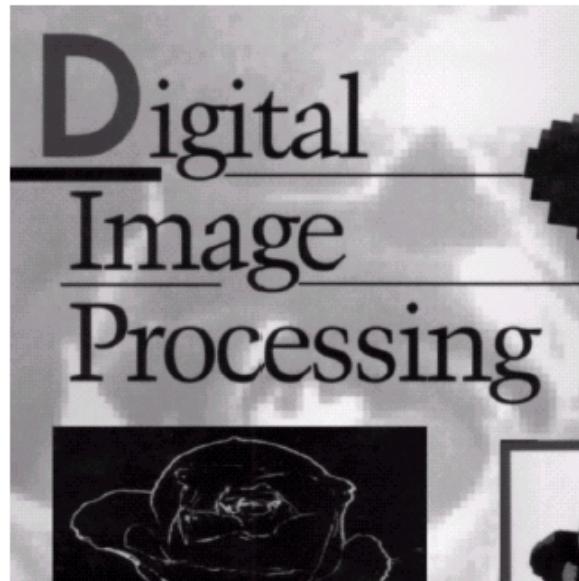
- *If we allow the motion in y-direction, with  $y_0(t)=bt/T$ , the model becomes,*

$$H(u, v) = \frac{T}{\pi(ua + vb)} \sin[\pi(ua + vb)] e^{-j\pi(ua + vb)}$$

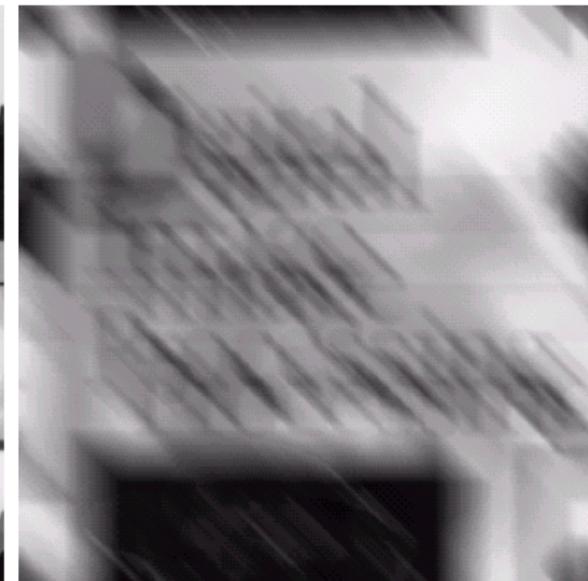
# Image Restoration

## Estimating the Degradation Function :

- **Estimation by Mathematical Modeling:** The result of the modeled motion blur is demonstrated in the following example:



Original Image



Blurred image with  $a=b=0.1$  and  $T=1$

$$H(u, v) = \frac{T}{\pi(ua + vb)} \sin[\pi(ua + vb)] e^{-j\pi(ua + vb)}$$

# Image Restoration

## Inverse Filtering:

- Until now our focus was the calculation of degradation function  $H(u,v)$ . Having  $H(u,v)$  calculated/estimated the next step is the restoration of the degraded image. The simplest way of image restoration is by using **Inverse filtering**:

$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)} \quad , \quad \hat{F}(u, v) \text{ is the Fourier transform of the restored image}$$

$$\hat{F}(u, v) = F(u, v) + \frac{N(u, v)}{H(u, v)}$$

Unknown random function

Must not be very small. Otherwise the noise dominates

- In Inverse filtering, we simply take  $H(u,v)$  such that the noise does not dominate the result. This is achieved by including only the low frequency components of  $H(u,v)$  around the origin. Note that, the origin,  $H(M/2,N/2)$ , corresponds to the highest amplitude component.

# Image Restoration

## Inverse Filtering:

- Consider the degradation function of the atmospheric turbulence for the origin of the frequency spectrum,

$$H(u, v) = e^{-k[(u-M/2)^2 + (v-N/2)^2]^{5/6}}$$

- If we consider a Butterworth Lowpass filter of  $H(u, v)$  around the origin we will only pass the low frequencies (high amplitudes of  $H(u, v)$ ).
- As we increase the cutoff frequency of the LPF more smaller amplitudes will be included. Therefore, instead of the degradation function the noise will be dominating.

$$\hat{F}(u, v) = F(u, v) + \frac{N(u, v)}{H(u, v)}$$

Must not be very small. Otherwise the noise dominates

# Image Restoration

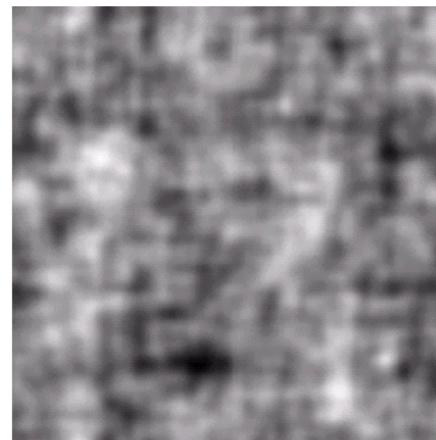
## Inverse Filtering:

- Consider the degradation function of the atmospheric turbulence for the origin of the frequency spectrum,

Result of full filter/degradation



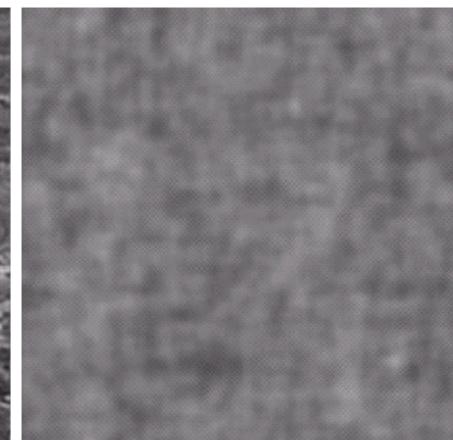
Input image with  
Severe turbulence  
 $k=0.0025$   
480x480 pixels



Cutoff outside  
of radius 40



Cutoff outside of radius 70



Cutoff outside  
of radius 85

# Image Restoration

## Wiener (Min Mean Square Error) Filtering:

*• Inverse filtering does not consider the additive noise for restoration. The Wiener Filter consider both the degradation function and the statistical characteristics of the noise in the restoration process.*

- The method tries to minimize the mean square error (MSE) between the uncorrupted image and the estimate of the image by:*

$$e^2 = E\{(f - \hat{f})^2\}$$

*E{.} is the expected value of the argument*

- The noise and the image are assumed to be uncorrelated. The minimum of the error function given above is achieved in the frequency domain by the following expression.*

$$\hat{F}(u, v) = \left[ \frac{H^*(u, v)S_f(u, v)}{S_f(u, v)|H(u, v)|^2 + S_\eta(u, v)} \right] G(u, v)$$

# Image Restoration

## Wiener (Min Mean Square Error) Filtering:

$$\begin{aligned}\hat{F}(u, v) &= \left[ \frac{H^*(u, v) S_f(u, v)}{S_f(u, v) |H(u, v)|^2 + S_\eta(u, v)} \right] G(u, v) \\ &= \left[ \frac{H^*(u, v)}{|H(u, v)|^2 + S_\eta(u, v) / S_f(u, v)} \right] G(u, v) \\ &= \left[ \frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_\eta(u, v) / S_f(u, v)} \right] G(u, v)\end{aligned}$$

$H(u, v)$  = *Degradation function.*

$H^*(u, v)$  = *Complex conjugate of  $H(u, v)$*

$$|H(u, v)|^2 = H^*(u, v) H(u, v)$$

$$S_\eta(u, v) = |N(u, v)|^2 = \text{ *Power spectrum of the noise.*}$$

$$S_f(u, v) = |F(u, v)|^2 = \text{ *Power spectrum of the undegraded image.*}$$

# Image Restoration

## Wiener (Min Mean Square Error) Filtering:

- When the power spectrum of the undegraded image and noise are not known, the ratio of the power spectrums of the noise and image is assumed to be constant.

$$\hat{F}(u, v) = \left[ \frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_\eta(u, v) / S_f(u, v)} \right] G(u, v)$$

When not known.  
Assumed to be constant

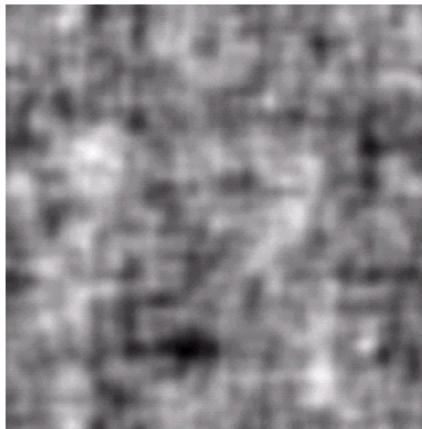
$$\hat{F}(u, v) = \left[ \frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + K} \right] G(u, v)$$

- Typically different values of  $K$  are chosen and the image quality is measured by MSE. The value of  $K$  is chosen in such a way that the MSE is minimized.

# Image Restoration

## Wiener (Min Mean Square Error) Filtering:

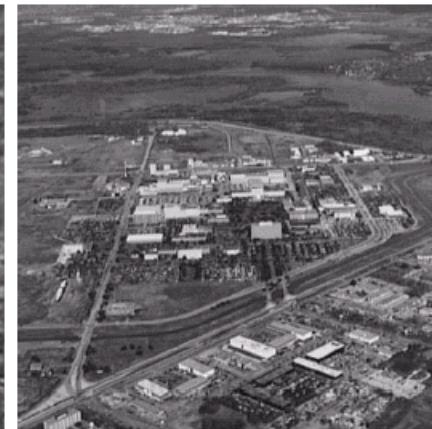
Input image with  
Severe turbulence  
blur ,  $k=0.0025$   
480x480 pixels



Result of Full inverse  
filtering



Radially limited Inverse  
Filtering with a cutoff  
radius of 70

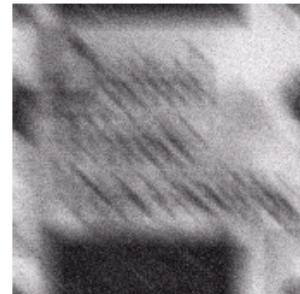


Result of Wiener filtering  
with and optimized K

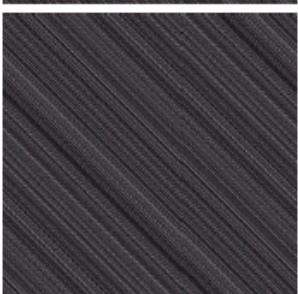
# Image Restoration

## Wiener (Min Mean Square Error) Filtering:

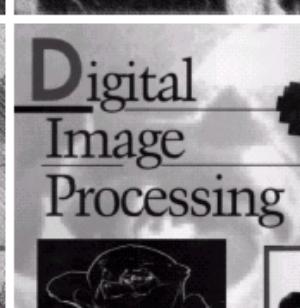
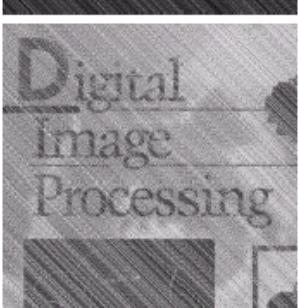
Image corrupted by motion blur and additive noise



Variance of the noise is one order of magnitude less



Variance of the noise is five order of magnitude less



Degraded Input Image

Result of Inverse filtering

Result of Wiener filtering

## Minimum Mean Square Error (Wiener) Filtering

- The inverse filtering approach discussed in the previous section makes no explicit provision for handling noise.
- In this section we discuss an approach that incorporates **both** the **degradation function** and **statistical characteristics of noise** into the restoration process.
- The object is to find an estimate  $\hat{f}$  of the uncorrupted image  $f$  such that **the mean square error(MSE)** between them is **minimized**.

$$e^2 = E\{(f - \hat{f})^2\} \quad \dots(5.8-1)$$

## Minimum Mean Square Error (Wiener) Filtering

- = The minimum of the error function in Eq.(5.8-1) is given in the frequency domain by the expression,

$$\begin{aligned}\hat{F}(u, v) &= \left[ \frac{H^*(u, v)S_f(u, v)}{S_f(u, v)|H(u, v)|^2 + S_\eta(u, v)} \right] G(u, v) \\ &= \left[ \frac{H^*(u, v)}{|H(u, v)|^2 + S_\eta(u, v)/S_f(u, v)} \right] G(u, v) \quad \dots(5.8-2) \\ &= \left[ \left( \frac{1}{H(u, v)} \right) \left( \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_\eta(u, v)/S_f(u, v)} \right) \right] G(u, v)\end{aligned}$$

$H(u, v)$  = degradation function

$H^*(u, v)$  = complex conjugate of  $H(u, v)$

$|H(u, v)|^2 = H^*(u, v)H(u, v)$

$S_\eta(u, v) = |N(u, v)|^2$  = power spectrum of the noise

$S_f(u, v) = |F(u, v)|^2$  = power spectrum of the undegraded image

## Minimum Mean Square Error (Wiener) Filtering

- A number of useful measures are based on the power spectra of noise and of the undegraded image.
- *Signal-to-noise ratio (SNR)*

$$\text{SNR} = \frac{\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |F(u, v)|^2}{\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |N(u, v)|^2} \quad (5.8-3)$$

# Minimum Mean Square Error (Wiener) Filtering

## Mean square error (MSE)

$$\text{MSE} = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y) - \hat{f}(x, y)]^2 \quad (5.8-4)$$

## Signal-to-noise ratio in the spatial domain

$$\text{SNR} = \frac{\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \hat{f}(x, y)^2}{\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} [f(x, y) - \hat{f}(x, y)]^2} \quad (5.8-5)$$

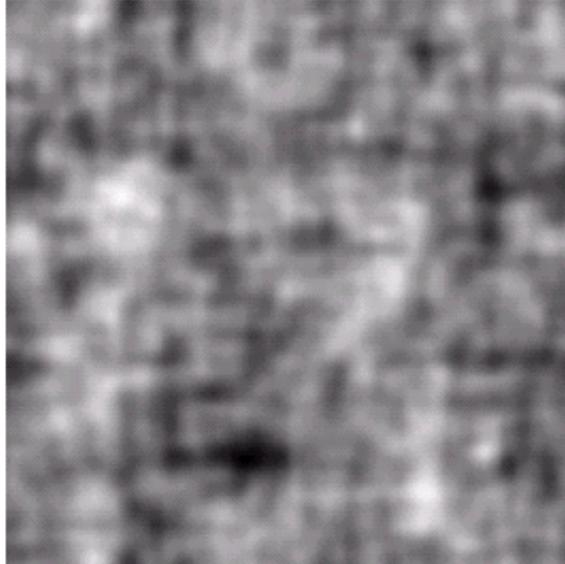
## Minimum Mean Square Error (Wiener) Filtering

- When we are dealing with spectrally white noise, the spectrum  $|N(u,v)|^2$  is a constant. However, the power spectrum of the undegraded image seldom is known. As this situation, we approximate Eq.(5.8-2) by the expression,

$$\hat{F}(u,v) = \left[ \frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + K} \right] G(u,v) \quad (5.8-6)$$

where  $K$  is a specified constant.

# Minimum Mean Square Error (Wiener) Filtering

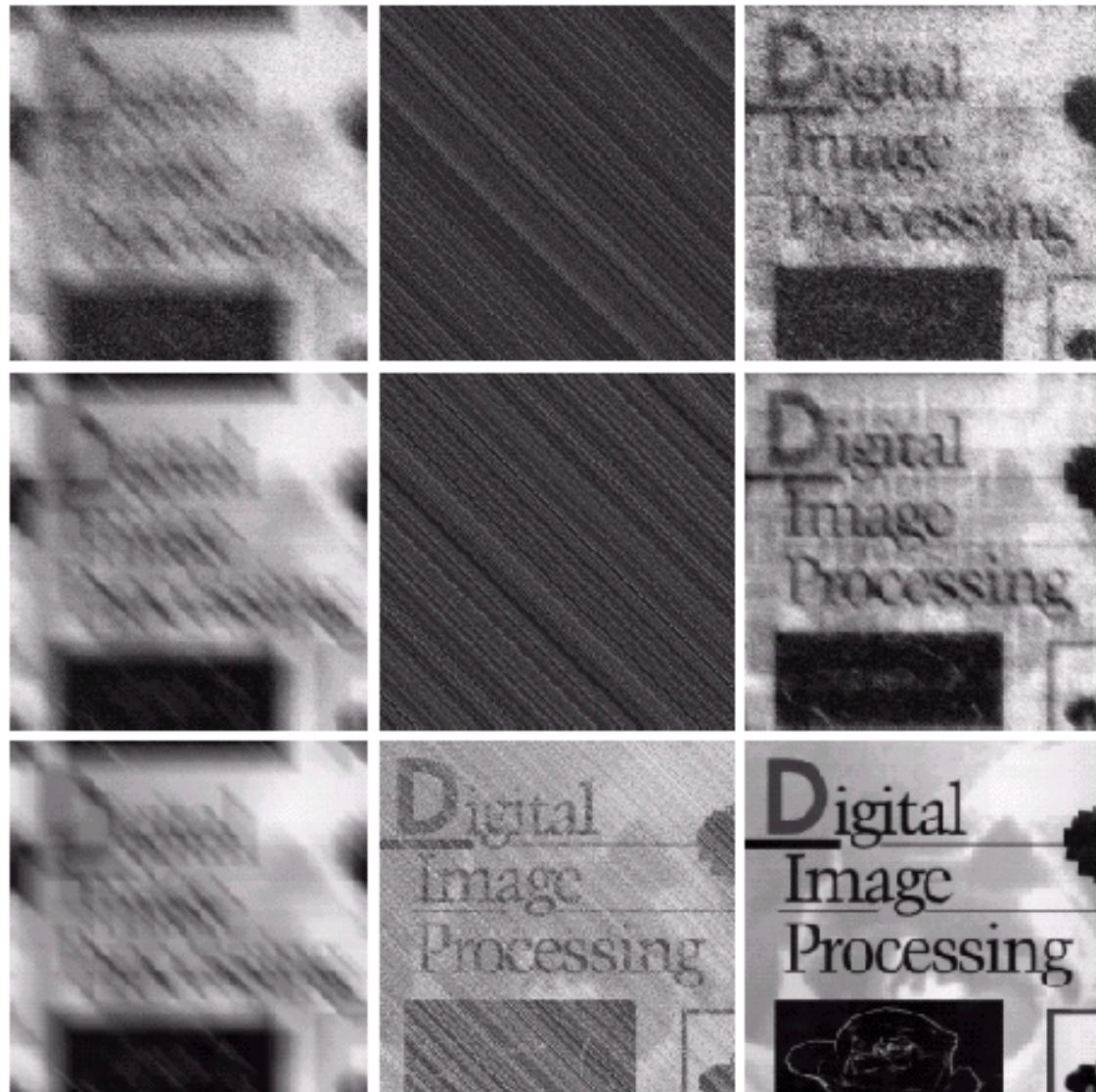


a b c



**FIGURE 5.28** Comparison of inverse- and Wiener filtering. (a) Result of full inverse filtering of Fig. 5.25(b). (b) Radially limited inverse filter result. (c) Wiener filter result.

# Minimum Mean Square Error (Wiener) Filtering



a	b	c
d	e	f
g	h	i

FIGURE 5.29

- (a) Image corrupted by motion blur and additive noise.  
(b) Result of inverse filtering.  
(c) Result of Wiener filtering.  
(d)-(f) Same sequence, but with noise variance one order of magnitude less.  
(g)-(i) Same sequence, but noise variance reduced by five orders of magnitude from (a). Note in (h) how the deblurred image is quite visible through a “curtain” of noise.

## Constrained Least Squares Filtering

- At the last section, a constant estimate(value of  $K$ ) of the ration of the power spectra is not always a suitable solution.
- In this section the method requires knowledge of only the **mean and variance** of the noise.
- By using the definition of convolution given in Eq.(4.2-30), we can express Eq.(5.5-16) in vector-matrix form, as follow:

$$g = Hf + \eta \quad \dots(5.9-1)$$

## Constrained Least Squares Filtering

- Central to the method is the issue of the **sensitivity of  $H$  to noise.**
- One way to alleviate this problem is to base optimality of restoration on **a measure of smoothness**, such as the second derivative of an image (Laplacian).

## Constrained Least Squares Filtering

- To be meaningful, the restoration must be constrained. Thus, what is desired is to **find the minimum of a criterion function,  $C$** , defined as

$$C = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\nabla^2 f(x, y)]^2 \quad \dots(5.9-2)$$

- Subject to the constraint**

$$\| g - H \hat{f} \|^2 = \| \eta \|^2 \quad \dots(5.9-3)$$

where  $\| w \|^2 \stackrel{\Delta}{=} w^T w$  is the Euclidean vector norm.

# Constrained Least Squares Filtering

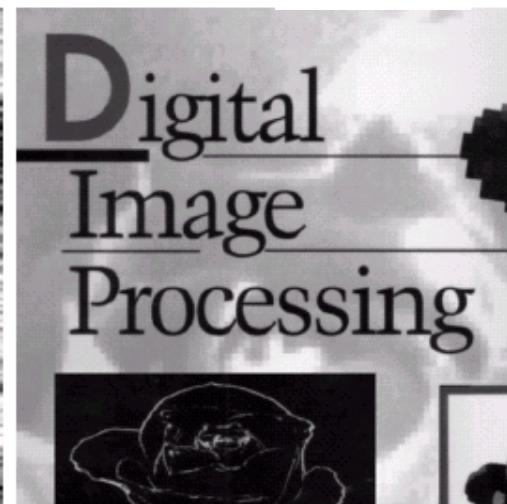
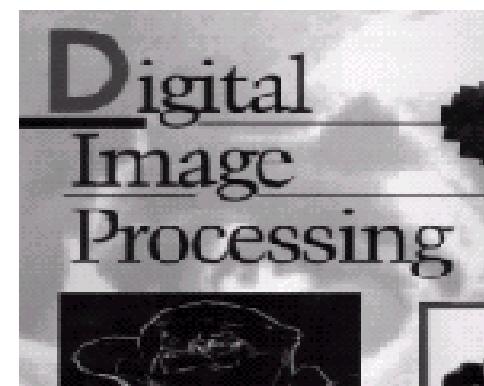
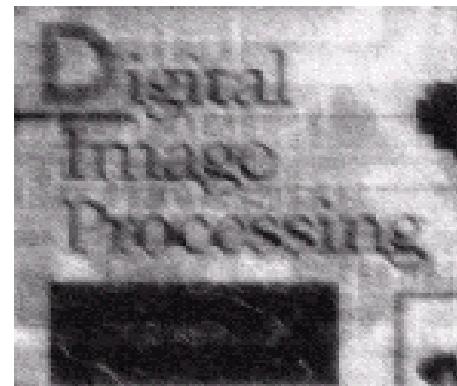
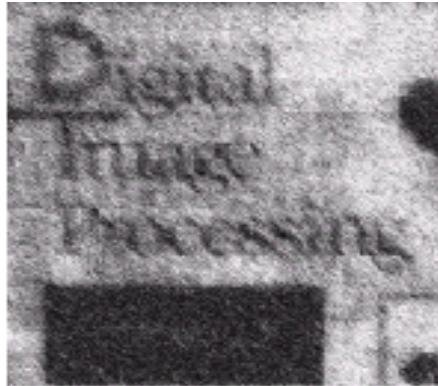
- The frequency domain solution

$$\hat{F}(u,v) = \left[ \frac{H^*(u,v)}{|H(u,v)|^2 + \gamma |P(u,v)|^2} \right] G(u,v) \quad \dots (5.9-4)$$

where  $\gamma$  is a parameter that must be adjusted so that the constraint in Eq.(5.9-3) is satisfied, and P(u,v) is the Fourier transform of the function

$$P(x,y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \dots (5.9-5)$$

# Constrained Least Squares Filtering



a b c

**FIGURE 5.30** Results of constrained least squares filtering. Compare (a), (b), and (c) with the Wiener filtering results in Figs. 5.29(c), (f), and (i), respectively.

## Constrained Least Squares Filtering

- If we are interested in optimality, however, then the parameter  $\gamma$  must be adjusted so that the constraint in Eq.(5.9-3) is satisfied. Define a “residual” vector  $r$  as

$$r = g - H \hat{f} \quad \dots(5.9-6)$$

- $r$  is a function of  $\gamma$ . It can be shown(Hunt[1973]) that

$$\phi(\gamma) = r^T r = \|r\|^2 \quad \dots(5.9-7)$$

- What we want to do is adjust gamma so that

$$\|r\|^2 = \|\eta\|^2 \pm a \quad \dots(5.9-8)$$

- Because  $\varphi(\gamma)$  is monotonic, finding the desired value of  $\gamma$  is not difficult. One approach is to
  - (1) Specify an initial value of  $\gamma$ .
  - (2) Compute  $\|r\|^2$ .
  - (3) Stop if Eq.(5.9-8) is satisfied; otherwise return to Step 2 after increasing  $\gamma$  if  $\|r\|^2 < \|\eta\|^2 - \alpha$  or decreasing  $\gamma$  if  $\|r\|^2 > \|\eta\|^2 + \alpha$ . Use the new value of  $\gamma$  in Eq.(5.9-4) to re-compute the optimum estimate  $\hat{F}(u,v)$ .
- We can use Newton-Raphson algorithm, too.

From Eq. (5.9-6)  $R(u,v) = G(u,v) - H(u,v)\hat{F}(u,v)$  (5.9-9)

In order to use this algorithm, we need the quantities

$$\|r\|^2 = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} r^2(x, y) \dots (5.9-10)$$

And then we find the

$$\sigma_\eta^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\eta(x, y) - m_\eta]^2 \dots (5.9-11)$$

where  $m_\eta = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \eta(x, y) \dots (5.9-12)$

$$\|\eta\|^2 = MN[\sigma_\eta^2 + m_\eta^2] \quad (5.9-13)$$

the use of mean & variance



a | b

**FIGURE 5.31**  
(a) Iteratively determined constrained least squares restoration of Fig. 5.16(b), using correct noise parameters.  
(b) Result obtained with wrong noise parameters.

- Generalize the Wiener filter to a **geometric mean filter**

$$\hat{F}(u, v) = \left( \frac{H^*(u, v)}{\|H(u, v)\|^2} \right)^\alpha \left( \frac{H^*(u, v)}{\|H(u, v)\|^2 + \beta [S_\eta(u, v)/S_f(u, v)]} \right)^{1-\alpha} G(u, v) \dots (5.10-1)$$

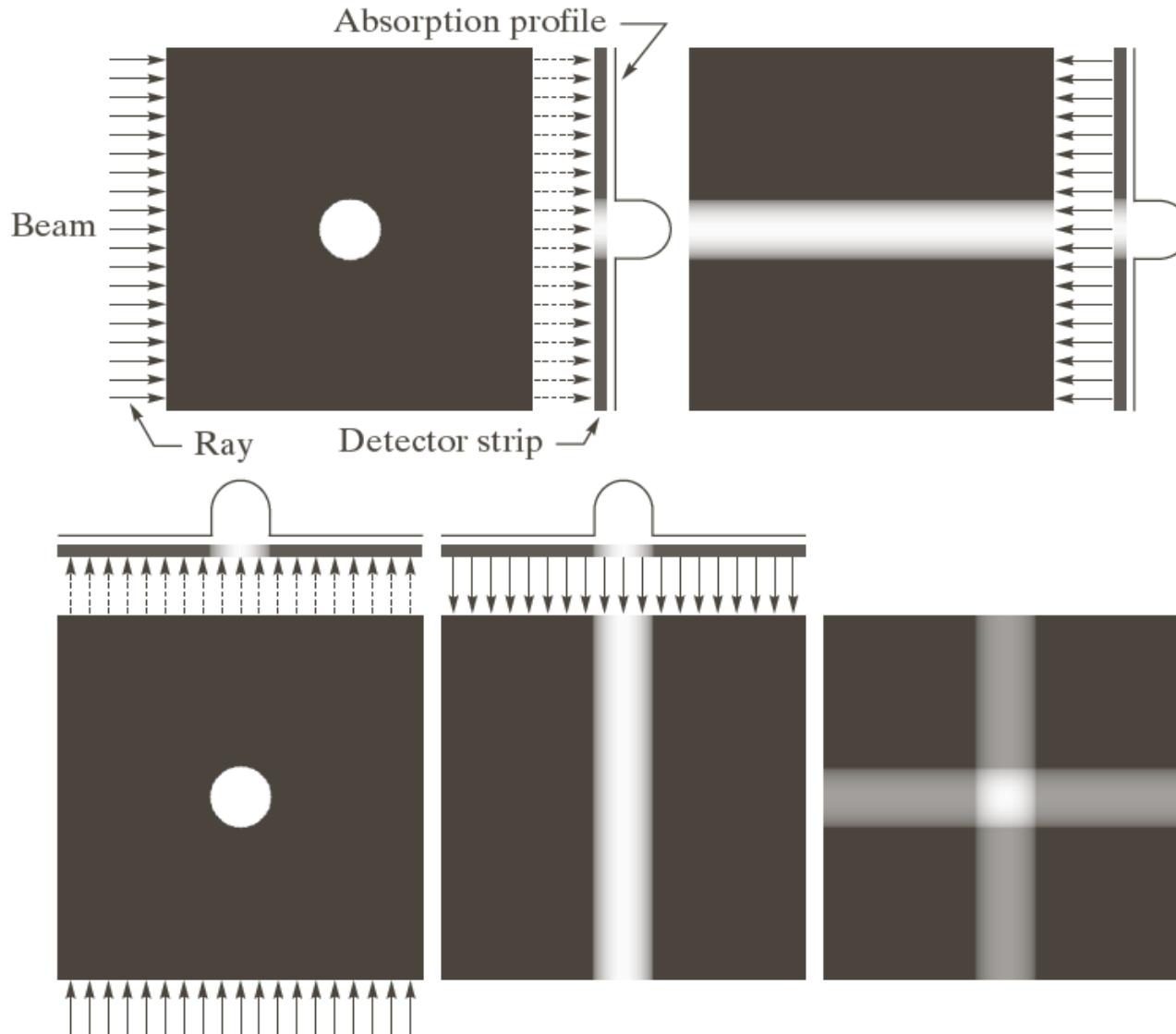
- By varying  $\alpha$  and  $\beta$  in the geometric mean filter, other filters can be realized.

With  $\alpha=1$ , this reduces to the inverse filter

With  $\alpha=0$  and  $\beta=1$ , this reduces to the Wiener filter

With  $\alpha=1/2$  and  $\beta=1$ , this is known as the spectrum equalization filter

# Image Reconstruction from Projections



a b  
c d e

**FIGURE 5.32**

- (a) Flat region showing a simple object, an input parallel beam, and a detector strip.  
**(b)** Result of back-projecting the sensed strip data (i.e., the 1-D absorption profile). (c) The beam and detectors rotated by 90°.  
**(d)** Back-projection.  
**(e)** The sum of (b) and (d). The intensity where the back-projections intersect is twice the intensity of the individual back-projections.

# Introduction

## “Halo” effect

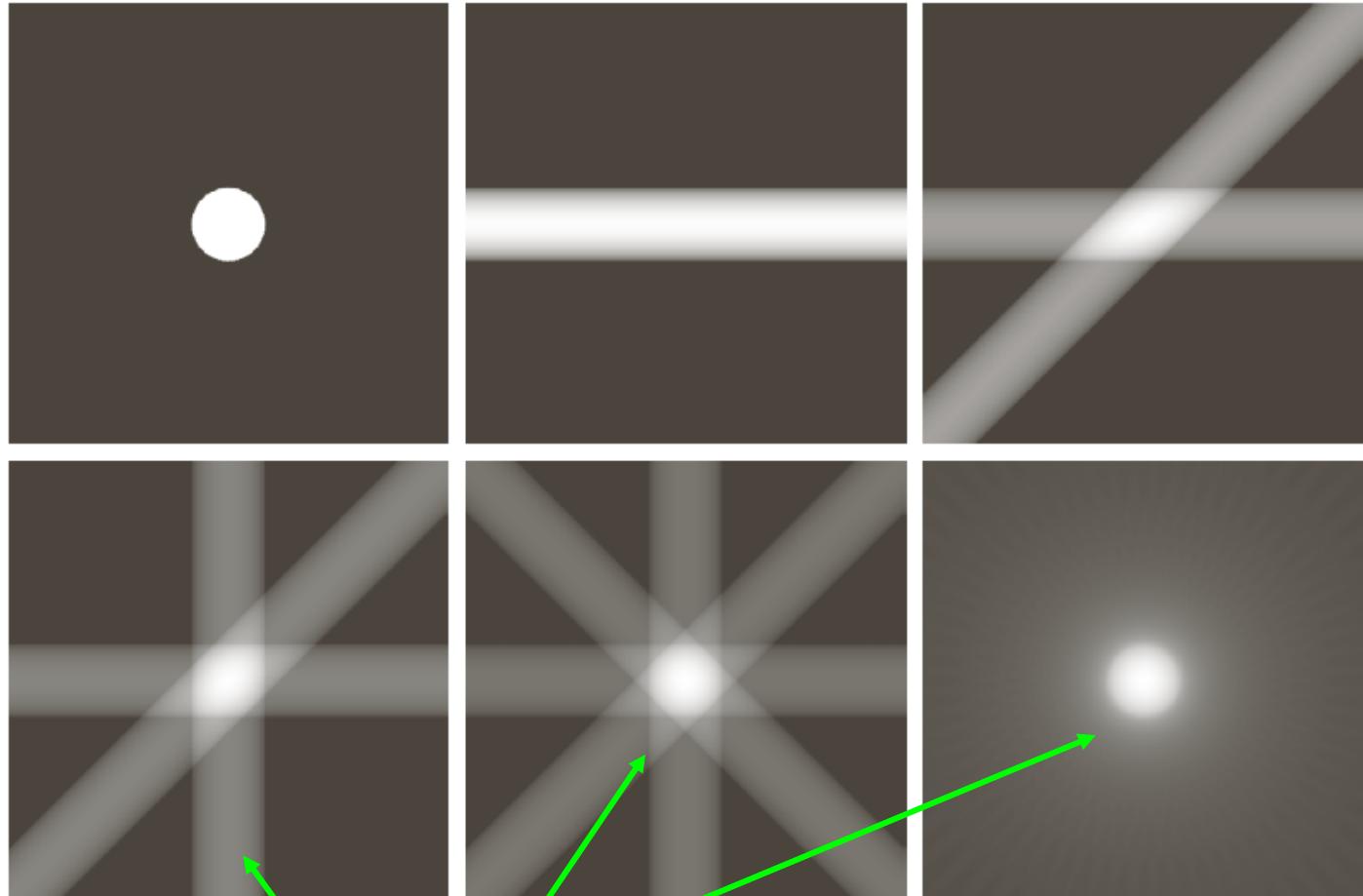
a	b	c
d	e	f

**FIGURE 5.33**

(a) Same as Fig.  
5.32(a).

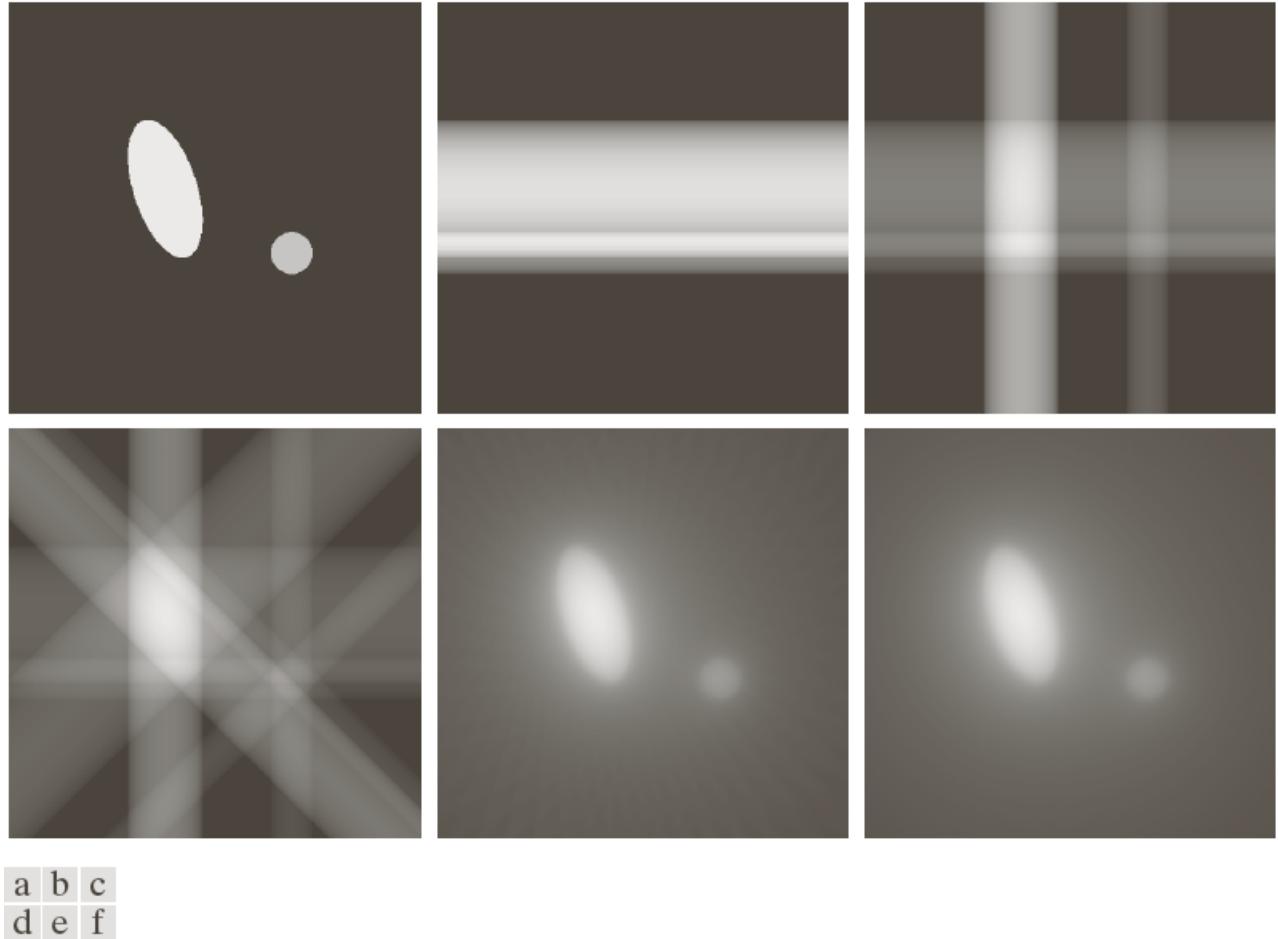
(b)–(e)  
Reconstruction  
using 1, 2, 3, and 4  
backprojections  $45^\circ$   
apart.

(f) Reconstruction  
with 32 backprojec-  
tions  $5.625^\circ$  apart  
(note the blurring).



“halo” effect

Backprojection  
of a region  
containing **two**  
objects

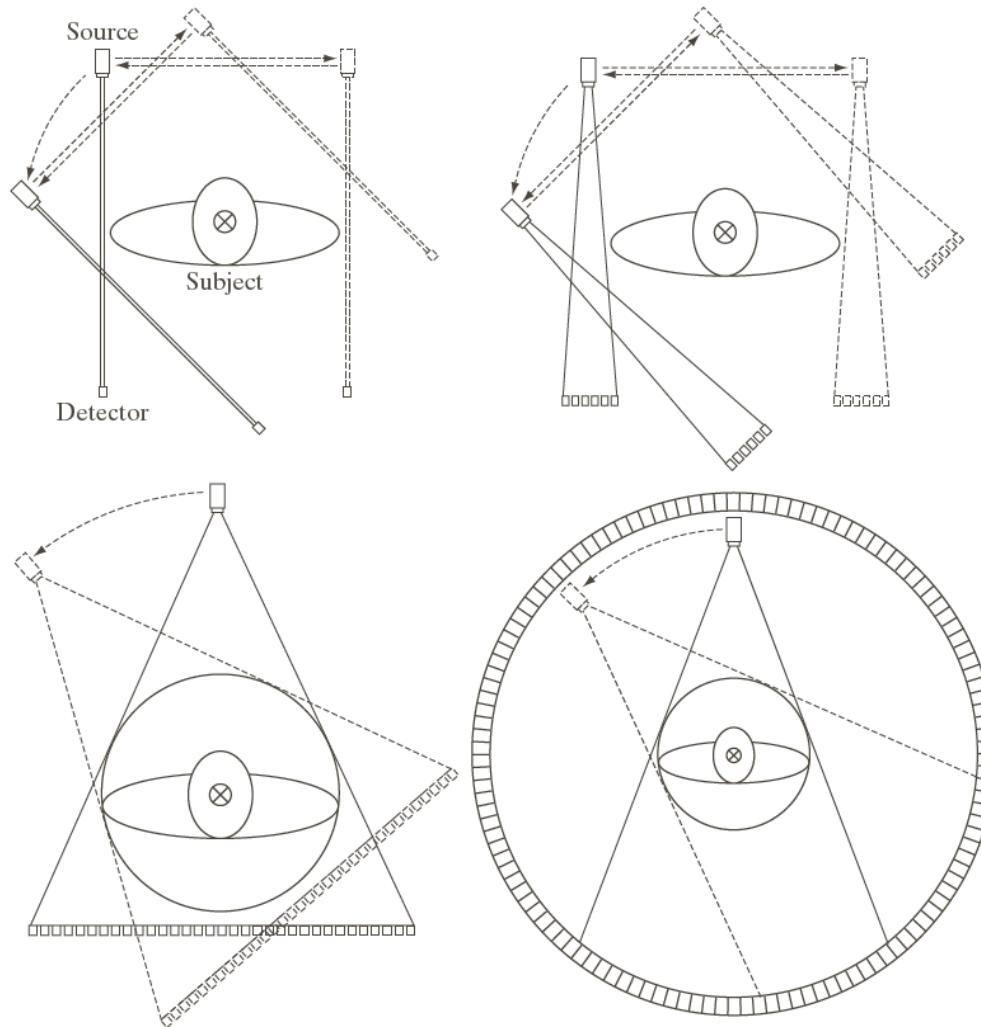


**FIGURE 5.34** (a) A region with two objects. (b)–(d) Reconstruction using 1, 2, and 4 backprojections  $45^\circ$  apart. (e) Reconstruction with 32 backprojections  $5.625^\circ$  apart. (f) Reconstruction with 64 backprojections  $2.8125^\circ$  apart.

# Principles of Computed Tomography (CT)

a  
b  
c  
d

**FIGURE 5.35** Four generations of CT scanners. The dotted arrow lines indicate incremental linear motion. The dotted arrow arcs indicate incremental rotation. The cross-mark on the subject's head indicates linear motion perpendicular to the plane of the paper. The double arrows in (a) and (b) indicate that the source/detector unit is translated and then brought back into its original position.



G1 CT scanners	G2 CT scanners
G3 CT scanners	G4 CT scanners

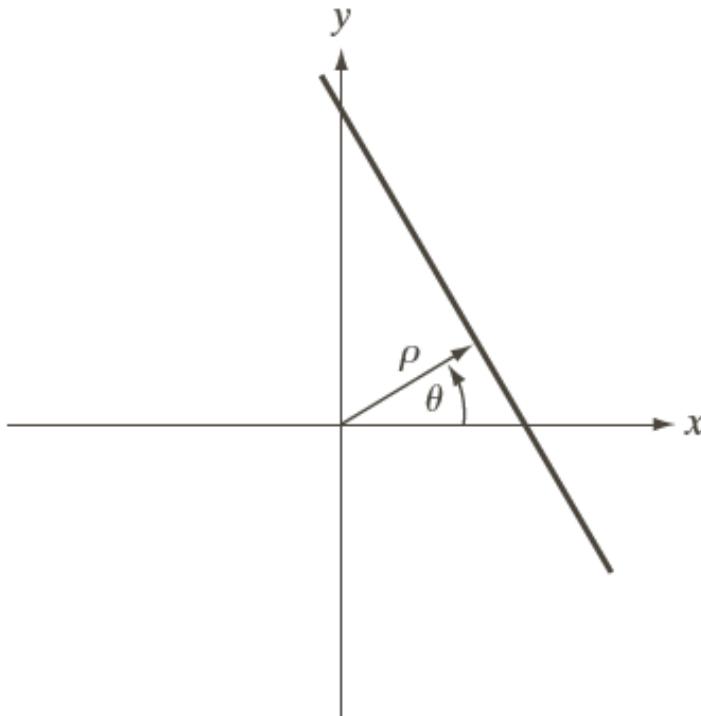
- ☛ G5 CT scanners – electron beam computed tomography (EBCT) scanners
  - Eliminate all mechanical motion by employing electron beams controlled electromagnetically.
- ☛ G6 CT scanners – helical CT
  - In this approach, a G3 or G4 scanner is configured using so-called *slip rings* that eliminate the need for electrical and signal cabling between the source/detectors and the processing unit.
- ☛ G7 CT scanners – multislice CT scanners
  - Emerge in which “thick” fan beams are used in conjunction with parallel banks of detectors to collect volumetric CT data simultaneously.

# Projections and the Radon Transform

- Normal representation of a straight line

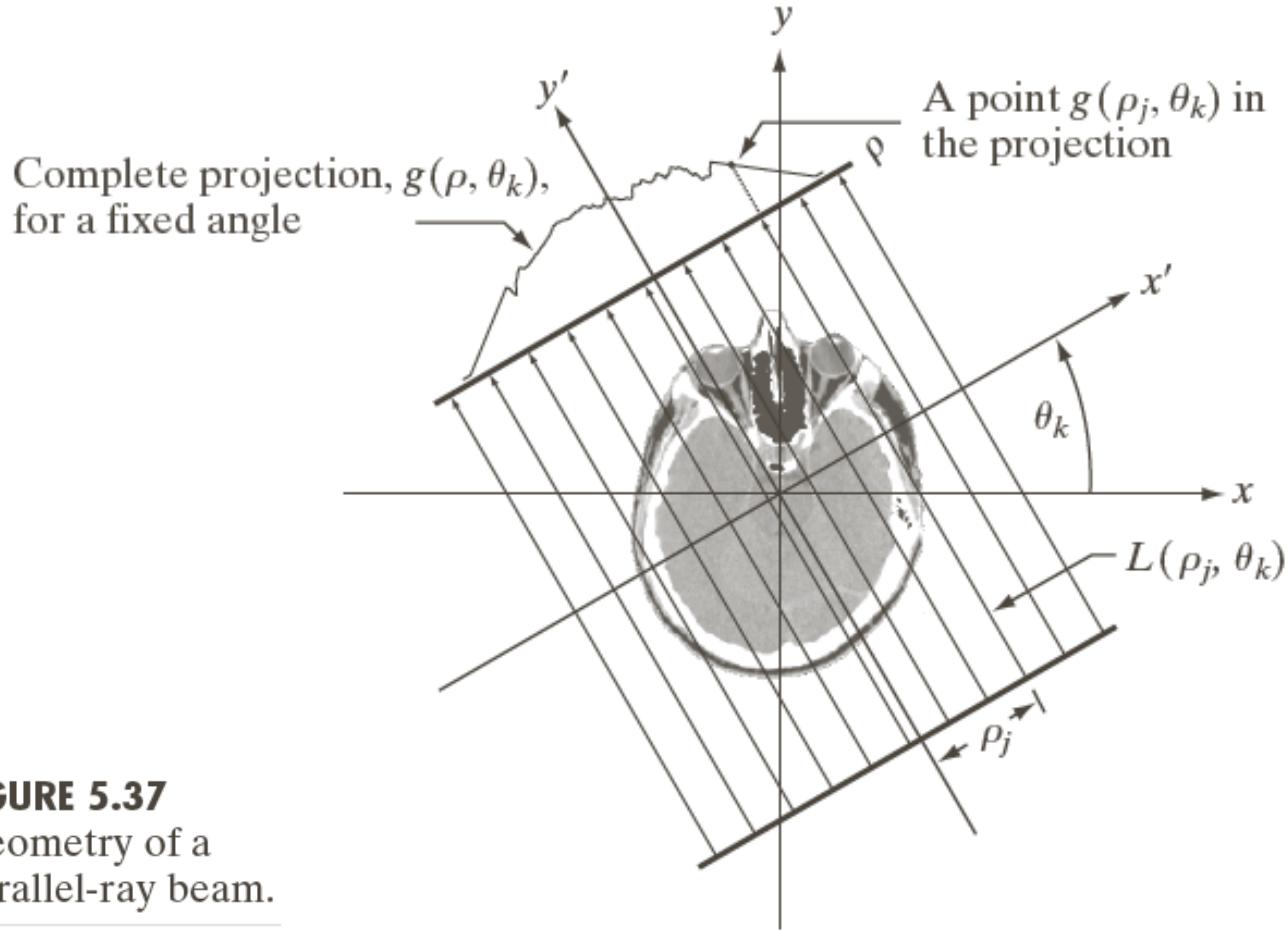
$$x \cos \theta + y \sin \theta = \rho \quad (5.11-1)$$

- The projection of a parallel-ray beam may be modeled by a set of such lines.



**FIGURE 5.36** Normal representation of a straight line.

# Projections and the Radon Transform



**FIGURE 5.37**  
Geometry of a  
parallel-ray beam.

## Projections and the Radon Transform

- ☛ An arbitrary point in the projection signal is given by the raysum along the line  $x \cos \theta_k + y \sin \theta_k = \rho_j$ .
- ☛ The raysum:

$$g(\rho_j, \theta_k) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta_k + y \sin \theta_k - \rho_j) dx dy \quad (5.11-2)$$

- ☛ Consider all values of  $\rho$  and  $\theta$ :

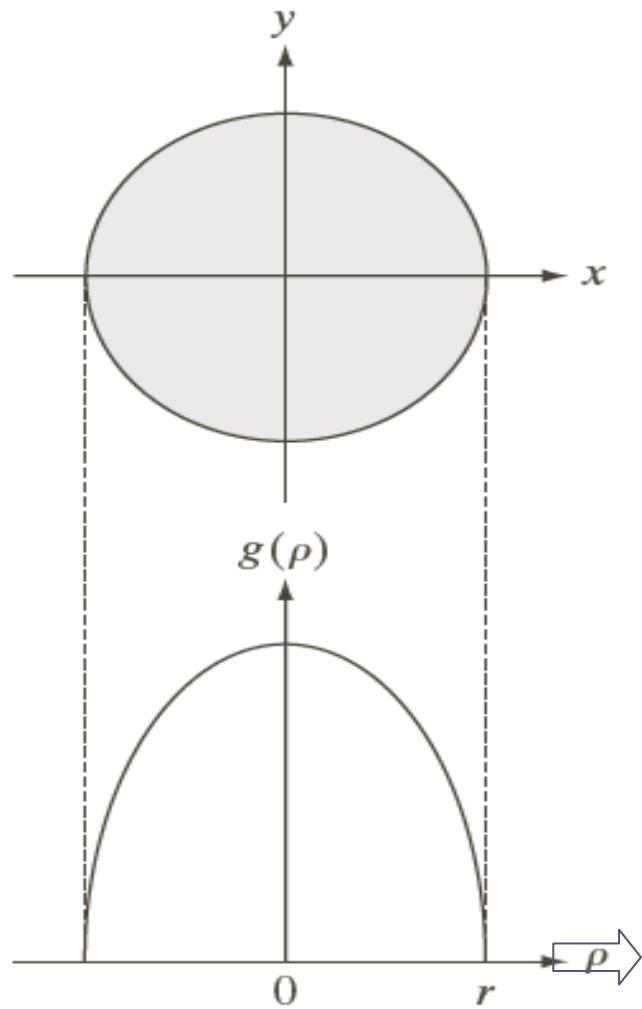
$$g(\rho, \theta) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - \rho) dx dy \quad (5.11-3)$$

- ☛ In the **discrete case**:

$$g(\rho, \theta) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \delta(x \cos \theta + y \sin \theta - \rho) \quad (5.11-4)$$

# Projections and the Radon Transform

**FIGURE 5.38** A disk and a plot of its Radon transform, derived analytically. Here we were able to plot the transform because it depends only on one variable. When  $g$  depends on both  $\rho$  and  $\theta$ , the Radon transform becomes an image whose axes are  $\rho$  and  $\theta$ , and the intensity of a pixel is proportional to the value of  $g$  at the location of that pixel.



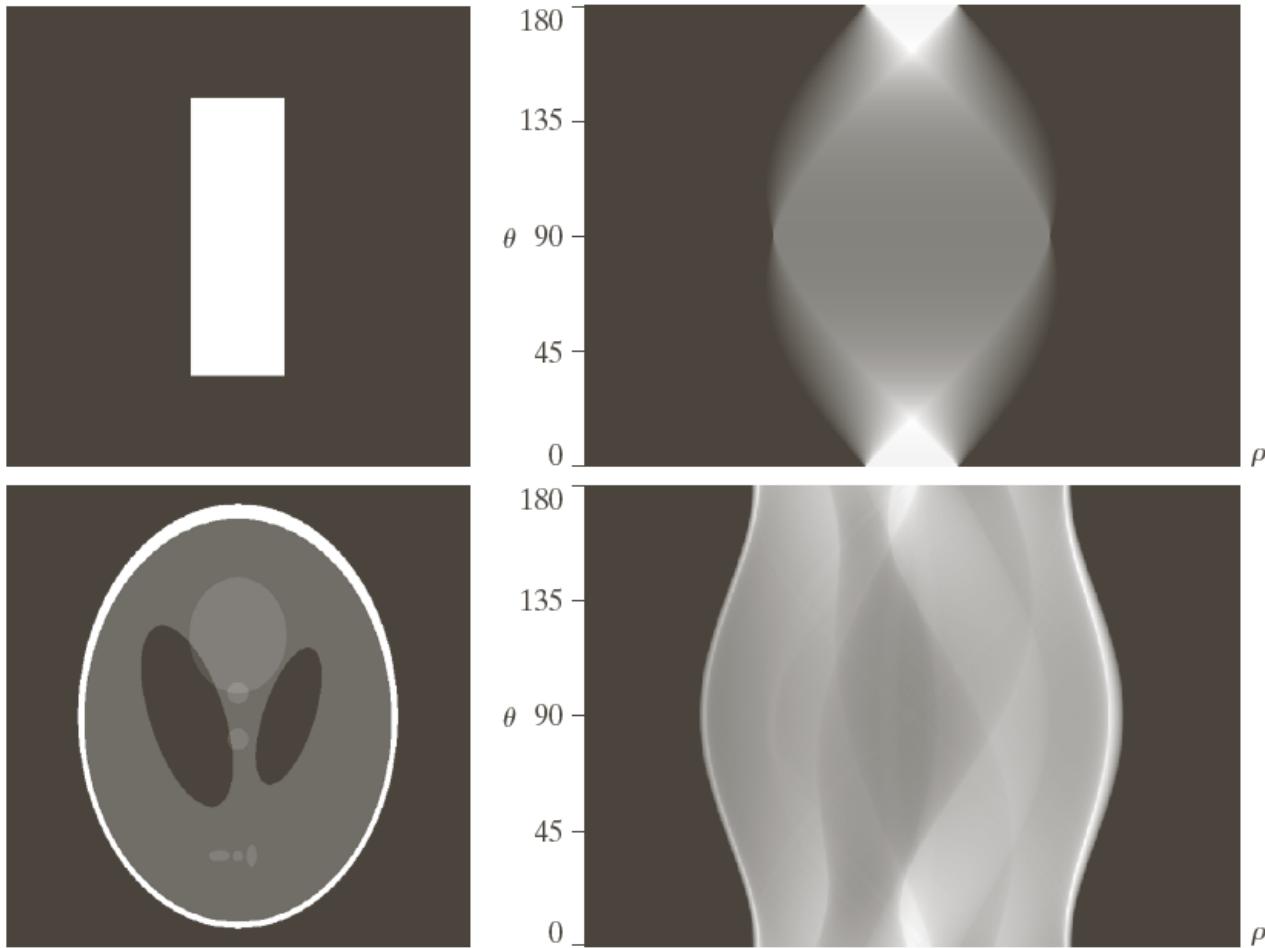
$$f(x, y) = \begin{cases} A & x^2 + y^2 \leq r^2 \\ 0 & \text{otherwise} \end{cases}$$

When  $|\rho| \leq r$

$$\begin{aligned} g(\rho, \theta) &= \int_{-\sqrt{r^2 - \rho^2}}^{\sqrt{r^2 - \rho^2}} f(\rho, y) dy \\ &= \int_{-\sqrt{r^2 - \rho^2}}^{\sqrt{r^2 - \rho^2}} A dy \end{aligned}$$

$$\begin{aligned} g(\rho, \theta) &= g(\rho) \\ &= \begin{cases} 2A\sqrt{r^2 - \rho^2} & |\rho| \leq r \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

# Projections and the Radon Transform



a	b
c	d

**FIGURE 5.39** Two images and their sinograms (Radon transforms). Each row of a sinogram is a projection along the corresponding angle on the vertical axis. Image (c) is called the *Shepp-Logan phantom*. In its original form, the contrast of the phantom is quite low. It is shown enhanced here to facilitate viewing.

- Complete projection for a fixed angle,  $\theta_k$

$$\begin{aligned}f_{\theta_k}(x, y) &= g(\rho, \theta_k) \\&= g(x \cos \theta_k + y \sin \theta_k, \theta_k)\end{aligned}$$

- Write in general:

$$f_\theta(x, y) = g(x \cos \theta + y \sin \theta, \theta) \quad (5.11-5)$$

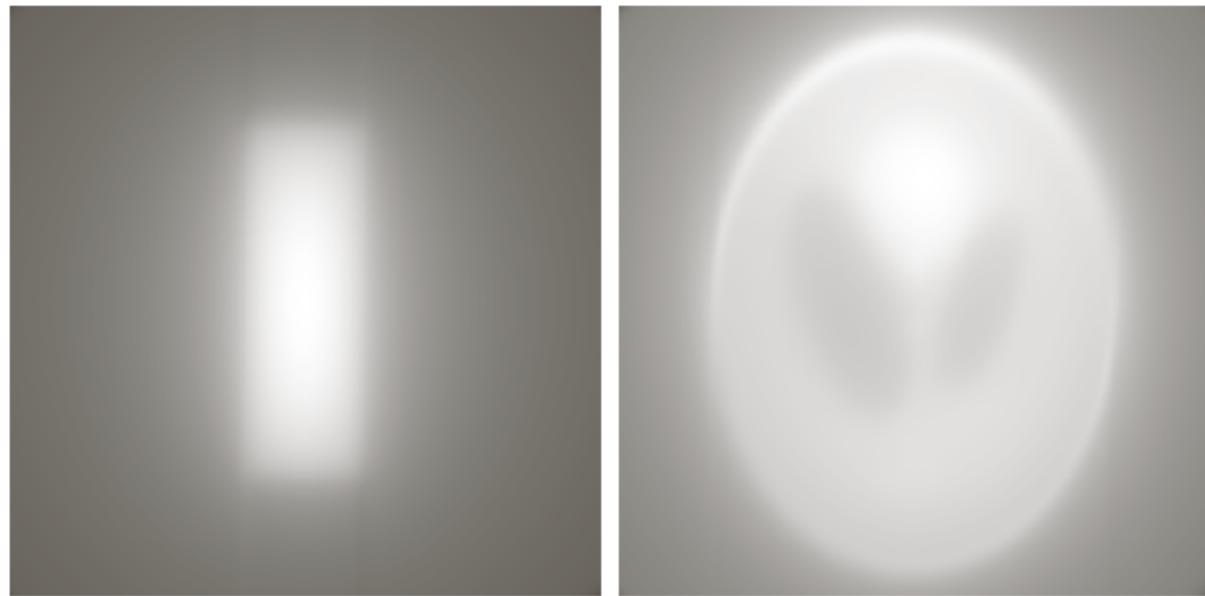
- Integrating over all the back-projected images:

$$f(x, y) = \int_0^\pi f_\theta(x, y) d\theta \quad (5.11-6)$$

- In the discrete case:

$$f(x, y) = \sum_{\theta=0}^{\pi} f_\theta(x, y) \quad (5.11-7)$$

Backprojections of the sinograms in Fig. 5.39.



## The Fourier Slice Theorem

- The 1-D Fourier transform of a projection with respect to  $\rho$ :

$$G(\omega, \theta) = \int_{-\infty}^{\infty} g(\rho, \theta) e^{-j2\pi\omega\rho} d\rho \quad (5.11-8)$$

- Substituting Eq. (5.11-3)

$$\begin{aligned} G(\omega, \theta) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - \rho) e^{-j2\pi\omega\rho} dx dy d\rho \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \left[ \int_{-\infty}^{\infty} \delta(x \cos \theta + y \sin \theta - \rho) e^{-j2\pi\omega\rho} d\rho \right] dx dy \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi\omega(x \cos \theta + y \sin \theta)} dx dy \end{aligned} \quad (5.11-9)$$

## The Fourier Slice Theorem

- Let  $u = \omega \cos \theta$  and  $v = \omega \sin \theta$

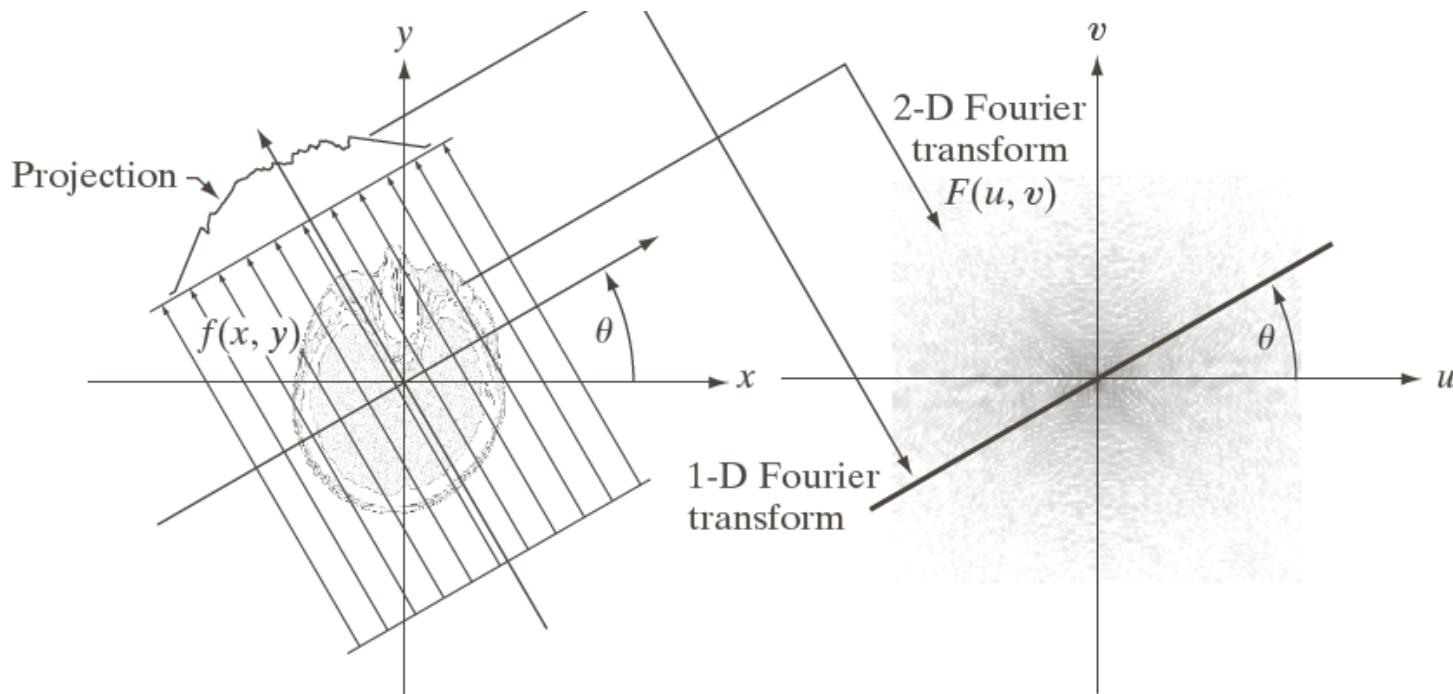
$$G(\omega, \theta) = \left[ \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(ux+vy)} dx dy \right]_{u=\omega \cos \theta; v=\omega \sin \theta} \quad (5.11-10)$$

- We recognize this expression as the 2-D Fourier transform of  $f(x, y)$  evaluated at the values of  $u$  and  $v$  indicated.

$$\begin{aligned} G(\omega, \theta) &= [F(u, v)]_{u=\omega \cos \theta; v=\omega \sin \theta} \\ &= F(\omega \cos \theta, \omega \sin \theta) \end{aligned} \quad (5.11-11)$$

- Eq. (5.11-11) is known as the *Fourier-slice theorem* (or the *projection-slice theorem*).

# The Fourier Slice Theorem



**FIGURE 5.41**  
Illustration of the Fourier-slice theorem. The 1-D Fourier transform of a projection is a slice of the 2-D Fourier transform of the region from which the projection was obtained. Note the correspondence of the angle  $\theta$ .

# Reconstruction Using Parallel Beam Filtered Backprojections

- 2-D inverse Fourier transform

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) e^{j2\pi(ux+vy)} du dv \quad (5.11-12)$$

- Let  $u = \omega \cos \theta$  and  $v = \omega \sin \theta$

$$f(x, y) = \int_0^{2\pi} \int_0^{\infty} F(\omega \cos \theta, \omega \sin \theta) e^{j2\pi\omega(x \cos \theta + y \sin \theta)} \omega d\omega d\theta \quad (5.11-13)$$

- Using the Fourier-slice theorem

$$f(x, y) = \int_0^{2\pi} \int_0^{\infty} G(\omega, \theta) e^{j2\pi\omega(x \cos \theta + y \sin \theta)} \omega d\omega d\theta \quad (5.11-14)$$

## Reconstruction Using Parallel Beam Filtered Backprojections

Given  $G(\omega, \theta + 180^\circ) = G(-\omega, \theta)$

$$\Rightarrow f(x, y) = \int_0^\pi \int_{-\infty}^{\infty} |\omega| G(\omega, \theta) e^{j2\pi\omega(x\cos\theta + y\sin\theta)} d\omega d\theta \quad (5.11-15)$$

$x\cos\theta + y\sin\theta$  is a constant

$$\Rightarrow f(x, y) = \int_0^\pi \left[ \int_{-\infty}^{\infty} |\omega| G(\omega, \theta) e^{j2\pi\omega\rho} d\omega \right]_{\rho=x\cos\theta + y\sin\theta} d\theta \quad (5.11-16)$$

$|\omega|$  is a *ramp* filter.

This function is **not integrable** because its amplitude extends to  $+\infty$  in both directions, so the inverse Fourier transform is undefined.

This is handled by methods such as using so-called *generalized delta function*.

The approach is to **window the ramp** so it becomes zero outside of a defined frequency interval.

# Reconstruction Using Parallel Beam Filtered Backprojections

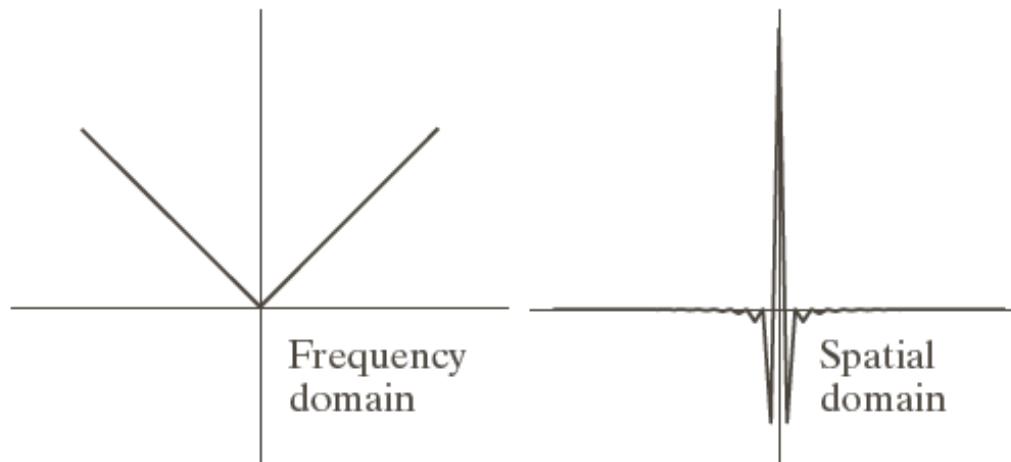
- An  $M$ -point discrete window function

$$h(\omega) = \begin{cases} c + (c-1)\cos \frac{2\pi\omega}{M-1} & 0 \leq \omega \leq (M-1) \\ 0 & \text{otherwise} \end{cases} \quad (5.11-17)$$

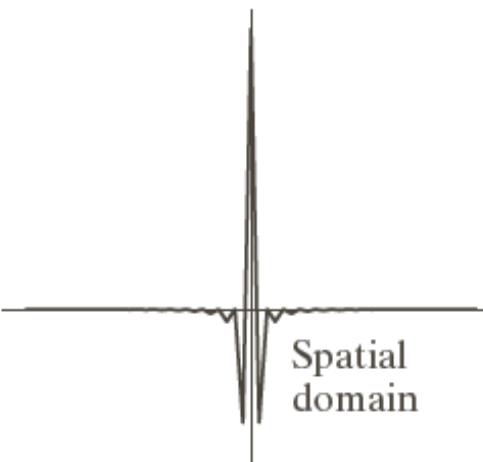
- $c = 0.54 \rightarrow \text{Hamming window}$
- $c = 0.5 \rightarrow \text{Hann window}$
- The *complete*, back-projected image  $f(x, y)$  is obtained as follows:

1. Compute the 1-D Fourier transform of each projection.
2. Multiply each Fourier transform by the filter function  $|\omega|$  which, as explained above, has been multiplied by a suitable (e.g., Hamming) window.
3. Obtain the inverse 1-D Fourier transform of each resulting filtered transform.
4. Integrate (sum) all the 1-D inverse transforms from step 3.

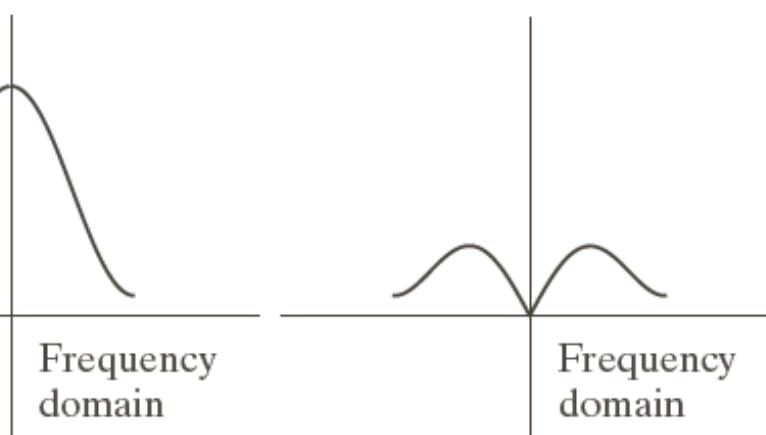
# Reconstruction Using Parallel Beam Filtered Backprojections



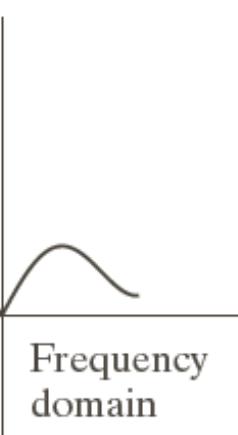
Frequency  
domain



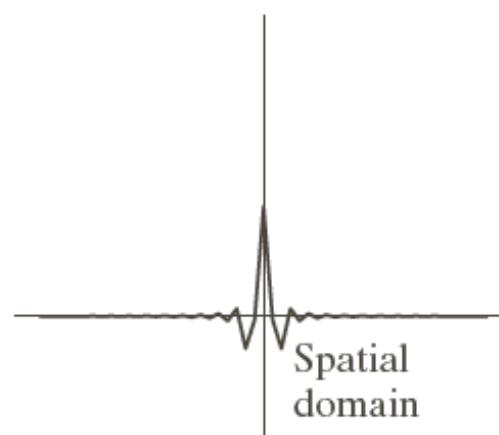
Spatial  
domain



Frequency  
domain



Frequency  
domain



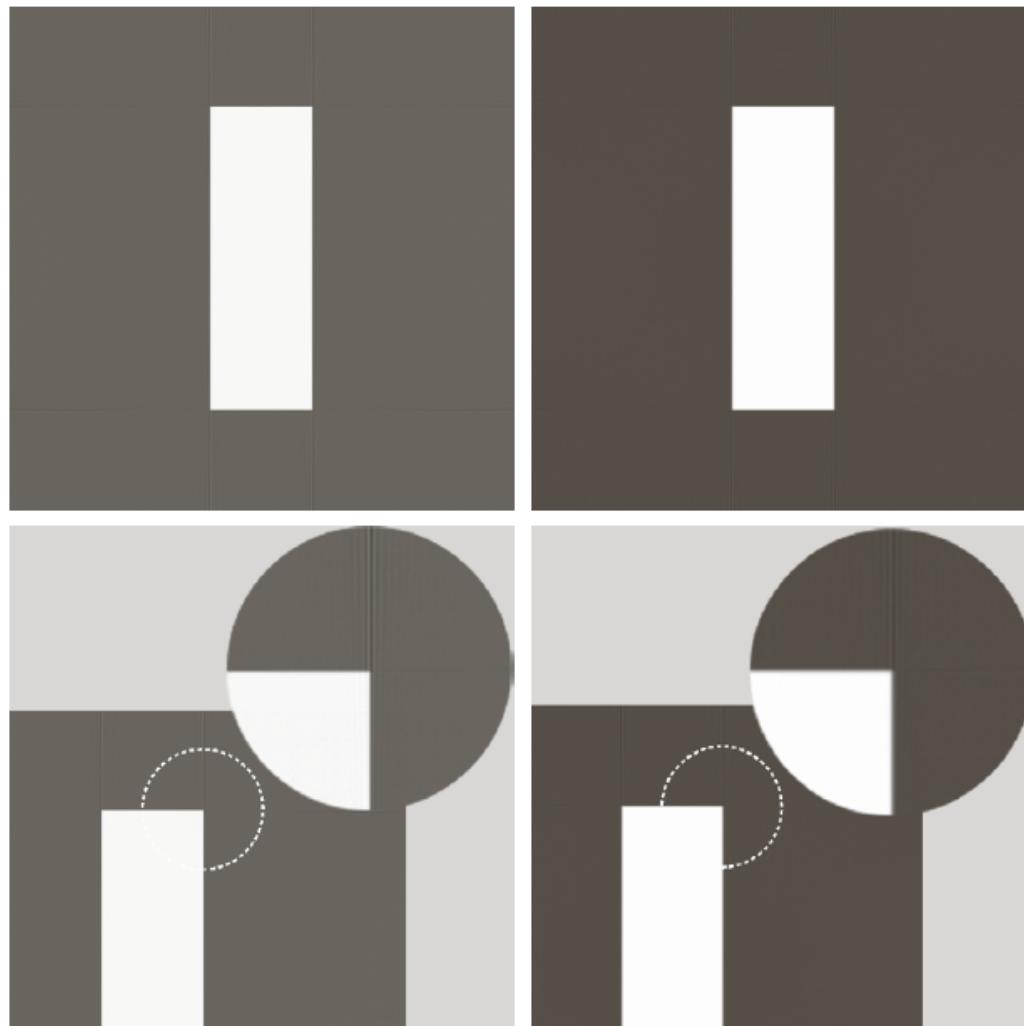
Spatial  
domain

a b  
c d e

**FIGURE 5.42**

- (a) Frequency domain plot of the filter  $|\omega|$  after band-limiting it with a box filter. (b) Spatial domain representation. (c) Hamming windowing function. (d) Windowed ramp filter, formed as the product of (a) and (c). (e) Spatial representation of the product (note the decrease in ringing).

# Reconstruction Using Parallel Beam Filtered Backprojections



a b  
c d

**FIGURE 5.43**  
Filtered back-  
projections of the  
rectangle using (a)  
a ramp filter, and (b)  
a Hamming-windowed  
ramp filter. The  
second row shows  
zoomed details of the  
images in the first  
row. Compare with  
Fig. 5.40(a).

---

# Reconstruction Using Parallel Beam Filtered Backprojections



a | b

**FIGURE 5.44**  
Filtered backprojections of the head phantom using (a) a ramp filter, and (b) a Hamming-windowed ramp filter. Compare with Fig. 5.40(b).

## Reconstruction Using Parallel Beam Filtered Backprojections

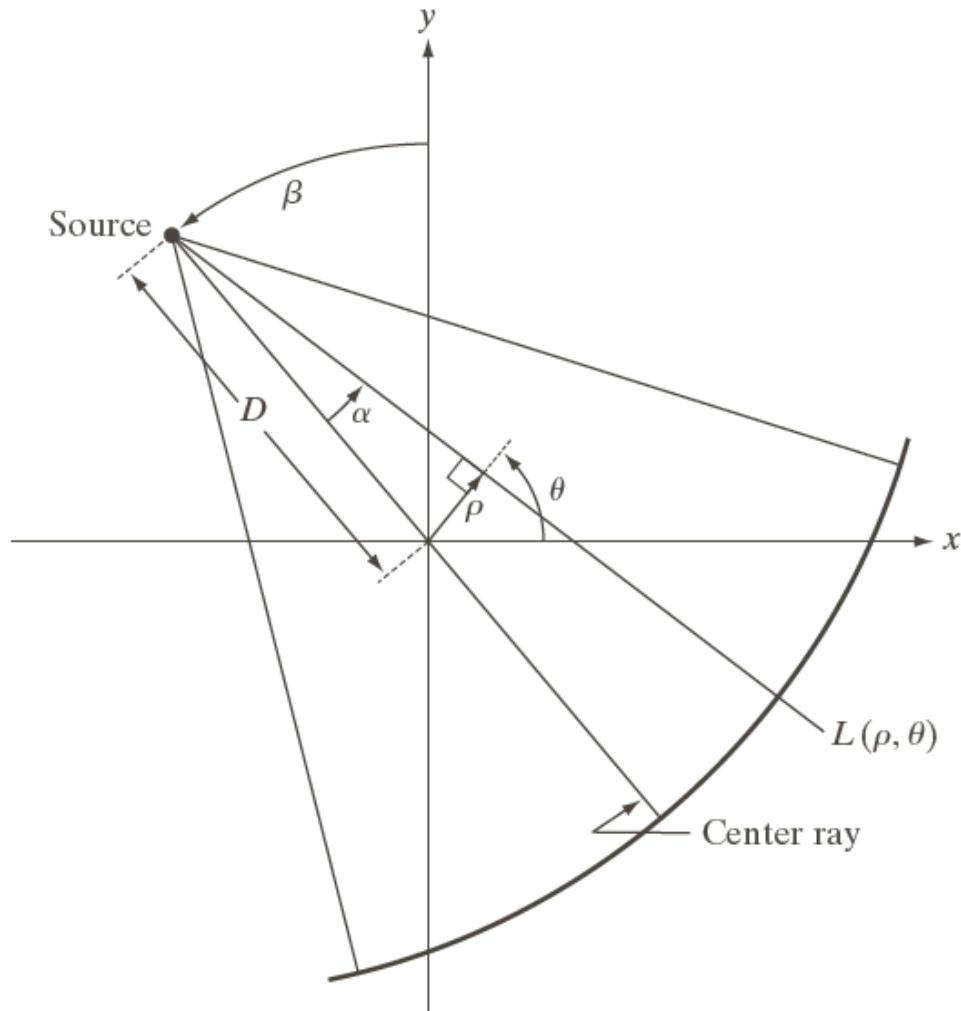
- From the convolution theorem, we know that equivalent results can be obtained using spatial convolution.
- Letting  $s(p)$  denote the inverse Fourier transform of  $| \omega |$

$$\begin{aligned} f(x, y) &= \int_0^\pi \left[ \int_{-\infty}^{\infty} |\omega| G(\omega, \theta) e^{j2\pi\omega\rho} d\omega \right]_{\rho=x\cos\theta+y\sin\theta} d\theta \\ &= \int_0^\pi [s(\rho) \star g(\rho, \theta)]_{\rho=x\cos\theta+y\sin\theta} d\theta \\ &= \int_0^\pi \left[ \int_{-\infty}^{\infty} g(\rho, \theta) s(x\cos\theta + y\sin\theta - \rho) d\rho \right] d\theta \quad (5.11-18) \end{aligned}$$

The reconstructed image is the sum of back-projected results from all projection angles.

# Reconstruction Using Parallel Beam Filtered Backprojections

## Basic fan-beam geometry



**FIGURE 5.45**  
Basic fan-beam geometry. The line passing through the center of the source and the origin (assumed here to be the center of rotation of the source) is called the *center ray*.

## Reconstruction Using Parallel Beam Filtered Backprojections

- In Fig. 5.45, the parameters of line  $L(\rho, \theta)$  are related to the parameters of a fan-beam ray by

$$\theta = \beta + \alpha \quad (5.11-19)$$

and

$$\rho = D \sin \alpha \quad (5.11-20)$$

- Suppose that we focus attention on objects that are encompassed within a circular area of radius  $T$  about the origin of the plane.
- Then  $g(\rho, \theta) = 0$  for  $|\rho| > T$  and

$$f(x, y) = \frac{1}{2} \int_0^{2\pi} \int_{-T}^T g(\rho, \theta) s(x \cos \theta + y \sin \theta - \rho) d\rho d\theta \quad (5.11-21)$$

## Reconstruction Using Parallel Beam Filtered Backprojections

- Let  $x = r \cos \varphi$  and  $y = r \sin \varphi$

$$\begin{aligned}x \cos \theta + y \sin \theta &= r \cos \varphi \cos \theta + r \sin \varphi \sin \theta \\&= r \cos(\theta - \varphi)\end{aligned}\quad (5.11-22)$$

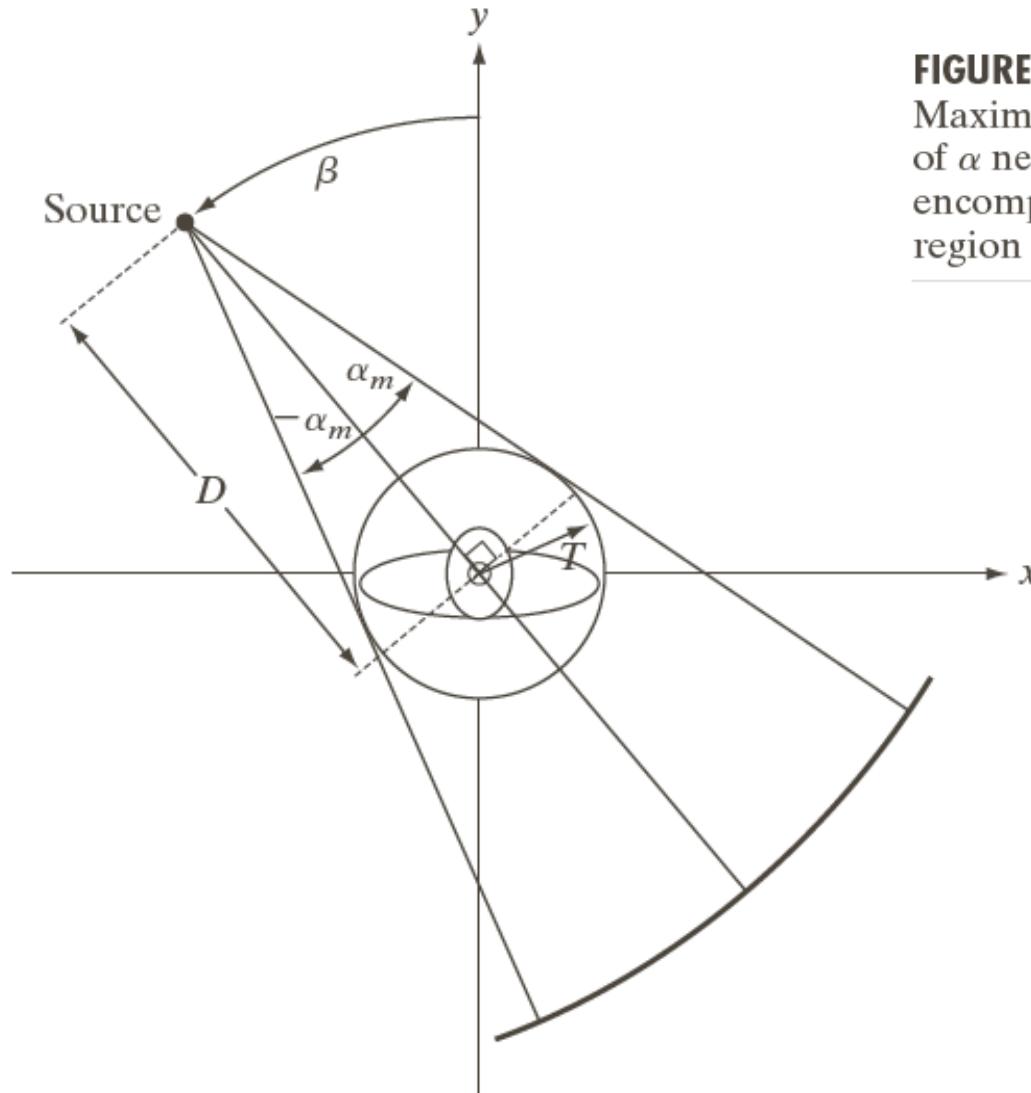
$$\Rightarrow f(x, y) = \frac{1}{2} \int_0^{2\pi} \int_{-T}^T g(\rho, \theta) s(r \cos(\theta - \alpha) - \rho) d\rho d\theta$$

- To integrate with respect to  $\alpha$  and  $\beta$  requires a transformation of coordinates using Eqs. (5.11-19) and (5.11-20)

$$\begin{aligned}f(r, \varphi) &= \frac{1}{2} \int_{-\alpha}^{2\pi - \alpha} \int_{\sin^{-1}(-T/D)}^{\sin^{-1}(T/D)} g(D \sin \alpha, \alpha + \beta) \\&\quad s[r \cos(\beta + \alpha - \varphi) - D \sin \alpha] D \cos \alpha d\alpha d\beta\end{aligned}\quad (5.11-23)$$

# Reconstruction Using Fan Beam Filtered Backprojections

- Maximum value of the term  $\sin^{-1}(T/D)$



**FIGURE 5.46**  
Maximum value  
of  $\alpha$  needed to  
encompass a  
region of interest.

## Reconstruction Using Fan Beam Filtered Backprojections

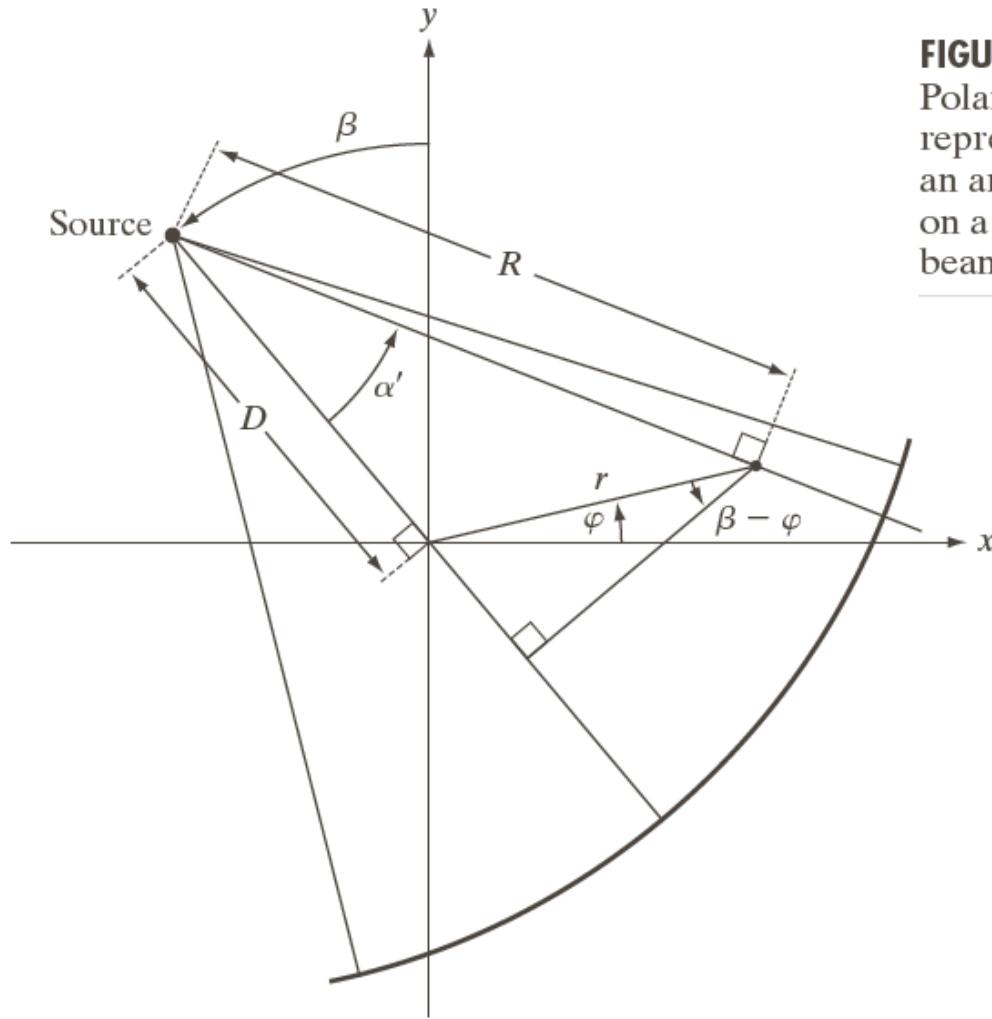
- Letting  $p(\alpha, \beta)$  denote a fan-beam projection, it follows that  
$$p(\alpha, \beta) = g(\rho, \theta)$$
- From Eqs. (5.11-19) and (5.11-20), it follows that  
$$p(\alpha, \beta) = g(D \sin \alpha, \alpha + \beta)$$
- Incorporating these observations into Eq. (5.11-23)

$$\Rightarrow f(r, \varphi) = \frac{1}{2} \int_0^{2\pi} \int_{-\alpha_m}^{\alpha_m} p(\alpha, \beta) s[r \cos(\beta + \alpha - \varphi) - D \sin \alpha] D \cos \alpha d\alpha d\beta \quad (5.11-24)$$

# Reconstruction Using Fan Beam Filtered Backprojections

- With reference to Fig. 5.47, it can be shown that

$$r \cos(\beta + \alpha - \varphi) - D \sin \alpha = R \sin(\alpha' - \alpha) \quad (5.11-25)$$



**FIGURE 5.47**

Polar representation of an arbitrary point on a ray of a fan beam.

## Reconstruction Using Fan Beam Filtered Backprojections

- Substituting Eq. (5.11-25) into Eq. (5.11-24)

$$f(r, \varphi) = \frac{1}{2} \int_0^{2\pi} \int_{-\alpha_m}^{\alpha_m} p(\alpha, \beta) s[R \sin(\alpha' - \alpha)] D \cos \alpha d\alpha d\beta \quad (5.11-26)$$

- It can be shown that

$$s(R \sin \alpha) = \left( \frac{\alpha}{R \sin \alpha} \right)^2 s(a) \quad (5.11-27)$$

- We can write Eq. (5.11-26) as

$$f(r, \varphi) = \int_0^{2\pi} \frac{1}{R^2} \left[ \int_{-\alpha_m}^{\alpha_m} q(\alpha, \beta) h(\alpha' - \alpha) d\alpha \right] d\beta \quad (5.11-28)$$

where

$$h(\alpha) = \frac{1}{2} \left( \frac{\alpha}{\sin \alpha} \right)^2 s(\alpha) \quad (5.11-29)$$

and

$$q(\alpha, \beta) = p(\alpha, \beta) D \cos \alpha \quad (5.11-30)$$

# Reconstruction Using Fan Beam Filtered Backprojections

- Instead of implementing Eq. (5.11-28) directly, an approach used often:
  - Convert a fan-beam geometry to a parallel-beam geometry.
  - Use the parallel-beam reconstruction approach.
- A fan-beam projection,  $p$ , taken at angle  $\beta$  has a corresponding parallel-beam projection,  $g$ , taken at a corresponding angle  $\theta$  and

$$\begin{aligned} p(\alpha, \beta) &= g(\rho, \theta) \\ &= g(D \sin \alpha, \alpha + \beta) \quad (5.11-31) \end{aligned}$$

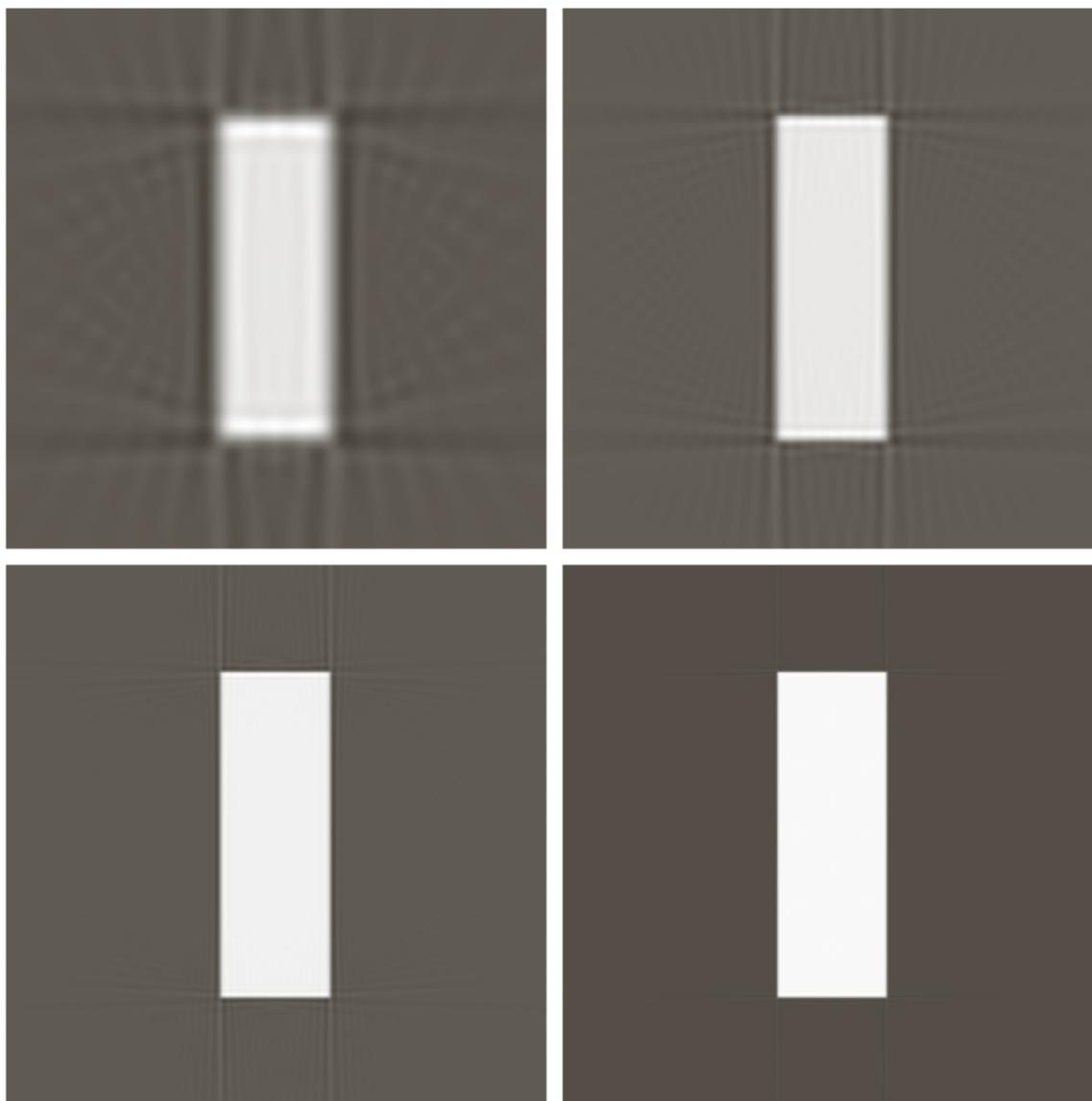
## Reconstruction Using Fan Beam Filtered Backprojections

- Let  $\Delta \beta$  denote the angular increment between successive fan-beam projections and let  $\Delta \alpha$  be the angular increment between rays

$$\Delta\beta = \Delta\alpha = \gamma \quad (5.11-32)$$

- Then  $\beta = m\gamma$  and  $\alpha = n\gamma$   
 $\Rightarrow p(n\gamma, m\gamma) = g[D \sin n\gamma, (m+n)\gamma] \quad (5.11-33)$
- This equation indicates that the  $n$ th ray in the  $m$ th radial projection is equal to the  $n$ th ray in the  $(m+n)$ th parallel projection.

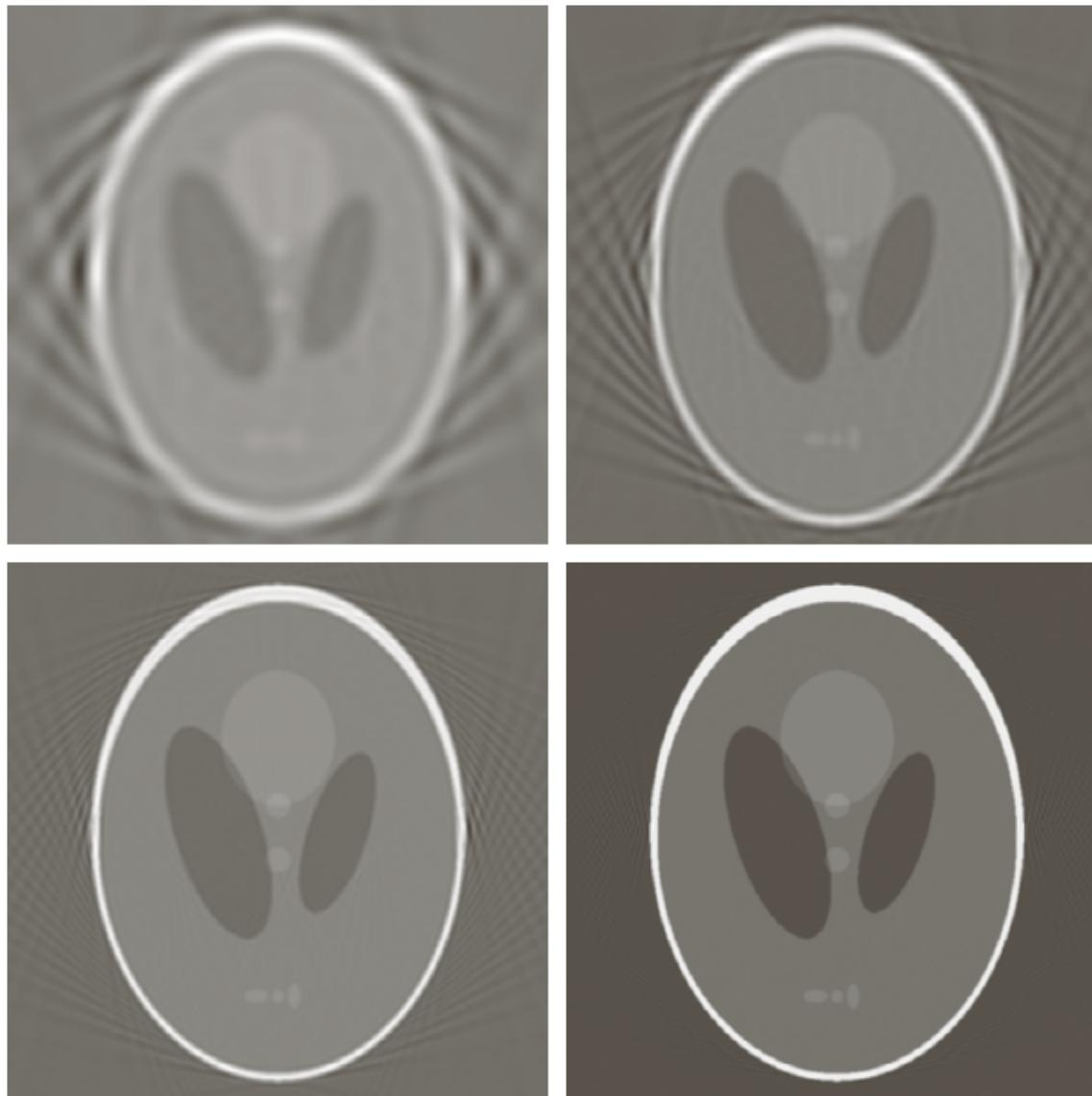
# Reconstruction Using Fan Beam Filtered Backprojections



a b  
c d

**FIGURE 5.48**  
Reconstruction of  
the rectangle  
image from  
filtered fan  
backprojections.  
(a)  $1^\circ$  increments  
of  $\alpha$  and  $\beta$ .  
(b)  $0.5^\circ$   
increments.  
(c)  $0.25^\circ$   
increments.  
(d)  $0.125^\circ$   
increments.  
Compare (d) with  
Fig. 5.43(b).

# Reconstruction Using Fan Beam Filtered Backprojections



a b  
c d

**FIGURE 5.49**

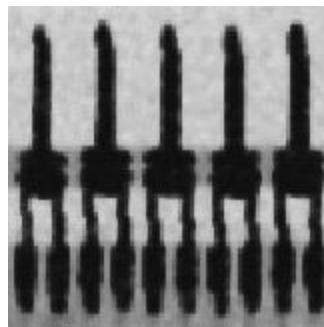
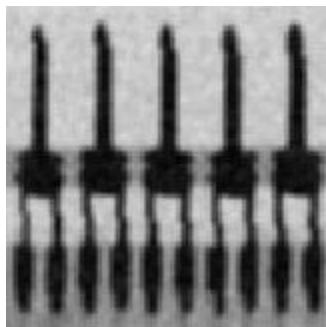
Reconstruction of the head phantom image from filtered fan backprojections.  
(a)  $1^\circ$  increments of  $\alpha$  and  $\beta$ .  
(b)  $0.5^\circ$  increments.  
(c)  $0.25^\circ$  increments.  
(d)  $0.125^\circ$  increments.  
Compare (d) with Fig. 5.44(b).



# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

- **Example 1-PR5.10:** Given the two subimages below. The sub image on the left is the result of using arithmetic mean filter of size 3x3. The other subimage is the result of using the geometric mean filter of the same size.
  - Why the subimage obtained with geometric filtering is less blurred. Hint you can start your analysis with 1-D step edge profile of the image.
  - Explain why the black components on the right image are thicker.



- Lets consider the mathematical expressions of the arithmetic and geometric mean filters:

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{s, t \in S_{xy}} g(s, t)$$

$$\hat{f}(x, y) = \left[ \prod_{s, t \in S_{xy}} g(s, t) \right]^{\frac{1}{mn}}$$

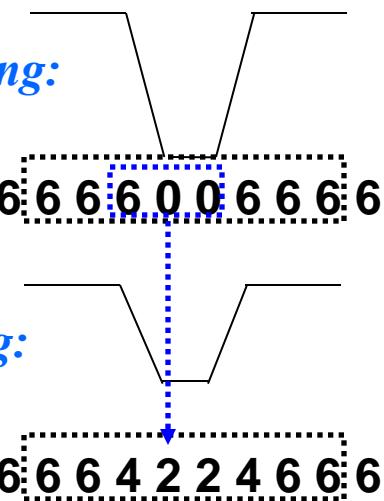
# Image Restoration

## Restoration in the presence of Noise: Only-Spatial Filtering

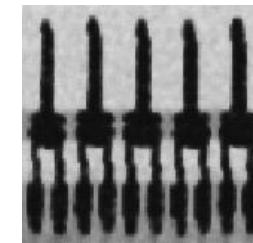
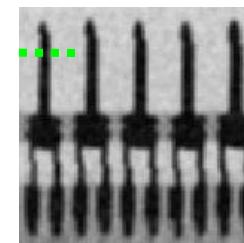
- Example 1-PR5.10 :

a) If we take a rough estimate of 1-D Step function profile before the filtering:

Arithmetic mean filtering:

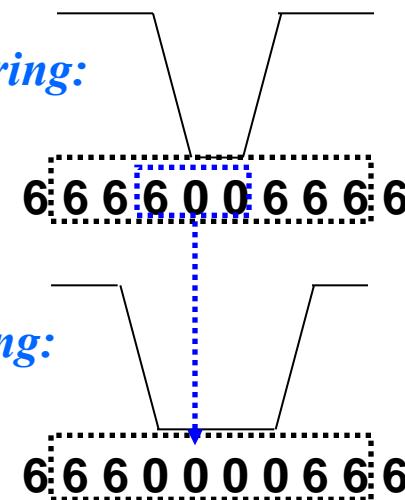


$$\hat{f}(x, y) = \frac{1}{mn} \sum_{s,t \in S_{xy}} g(s, t)$$



Geometric mean filtering:

Before filtering:



$$\hat{f}(x, y) = \left[ \prod_{s,t \in S_{xy}} g(s, t) \right]^{\frac{1}{mn}}$$

## Unit IV-Image Restoration&Reconstruction

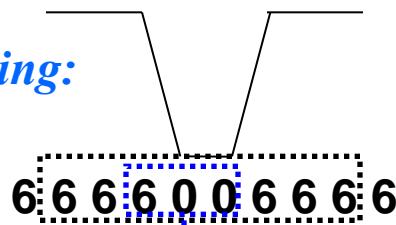
### Restoration in the presence of Noise: Only-Spatial Filtering

- **Example 1-PR5.10 :**

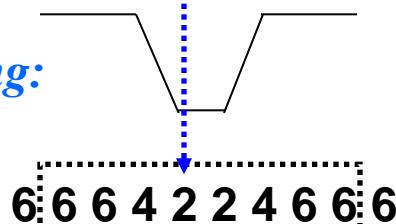
- The resulting 1-D profiles clearly indicates that the arithmetic mean filter produces smoother/blurred transition and*
- The geometric filtering increases the thickness of the black components.*

*Arithmetic mean filtering:*

*Before filtering:*

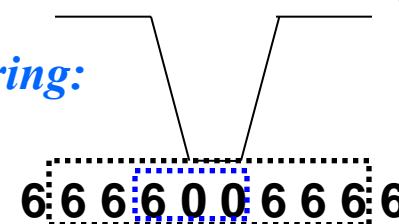


*After filtering:*

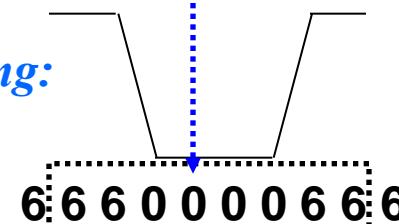


*Geometric mean filtering:*

*Before filtering:*



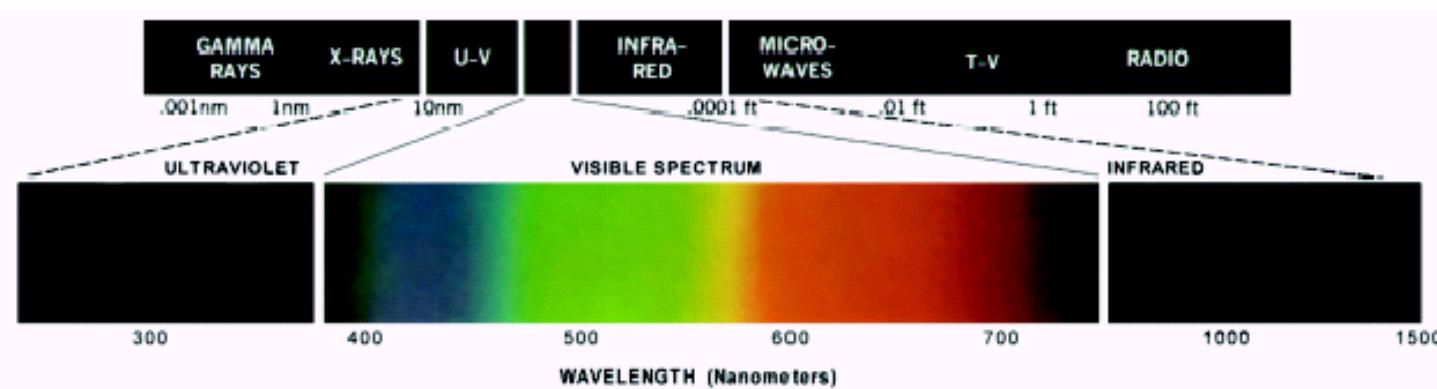
*After filtering:*



## Unit V - Color Image Processing

### Color Fundamentals

- *Visible light can be represented by the combination of different colors with changing electromagnetic wavelengths from violet to red in the visible spectrum.*

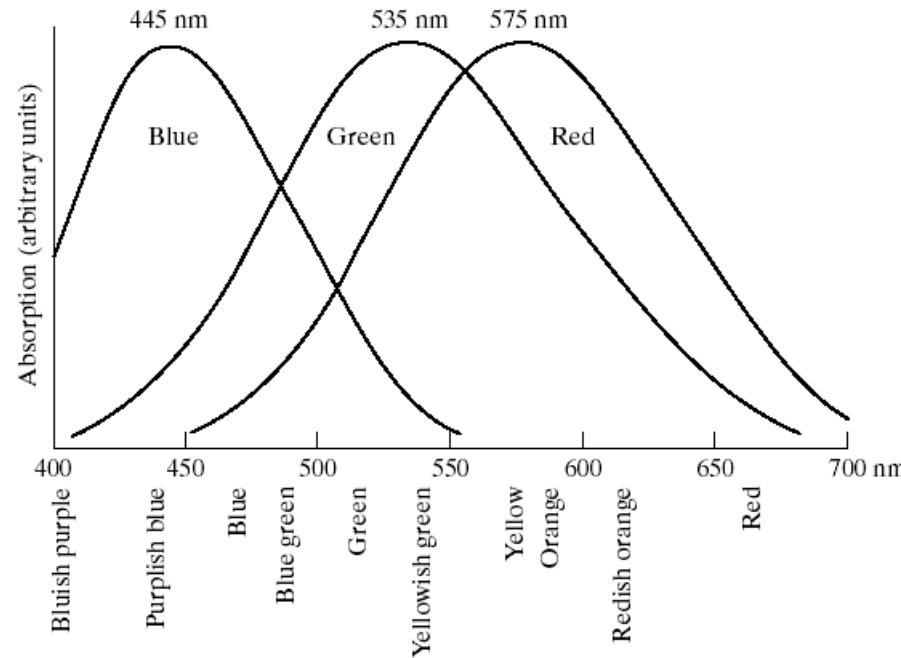


**FIGURE 6.2** Wavelengths comprising the visible range of the electromagnetic spectrum. (Courtesy of the General Electric Co., Lamp Business Division.)

# Color Image Processing

## Color Fundamentals

- *Chromatic (colored) light spans to electromagnetic waves between approximately 400 nm and 700 nm which corresponds to colors from violet to red. These colors are perceived by the human eye in the following way.*

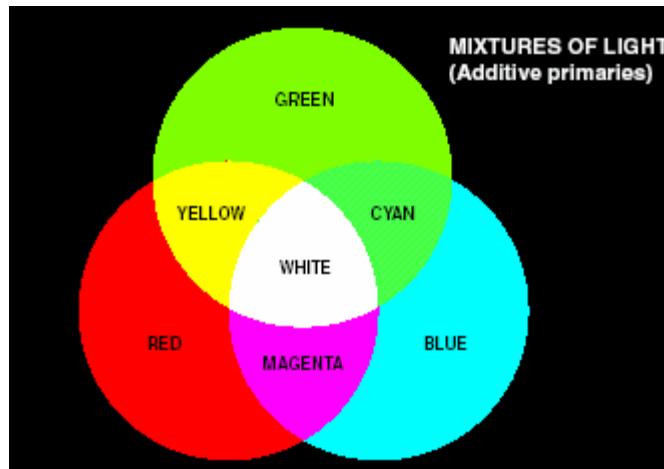


**FIGURE 6.3** Absorption of light by the red, green, and blue cones in the human eye as a function of wavelength.

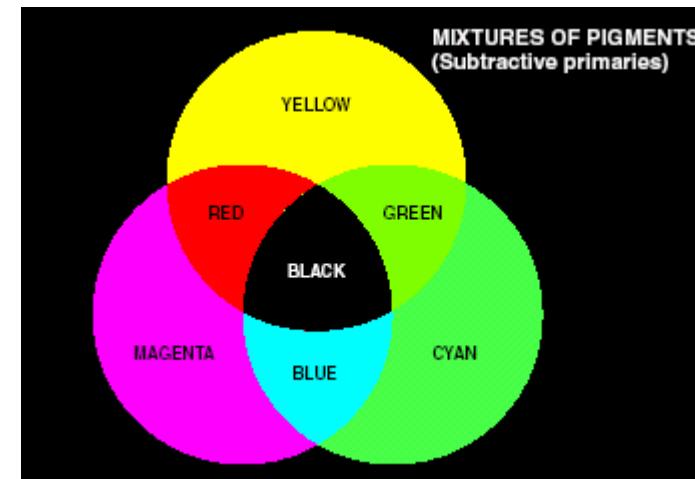
# Color Image Processing

## Color Fundamentals

- *Primary Colors: Red(R), Green(G) and Blue(B) are referred as the primary colors and when mixed with various intensity proportions, can produce all visible colors.*
- *The primary colors can be mixed to generate secondary colors such as magenda (red+blue), cyan( green + blue) and yellow (red+green).*



Primary Colors-  
RGB

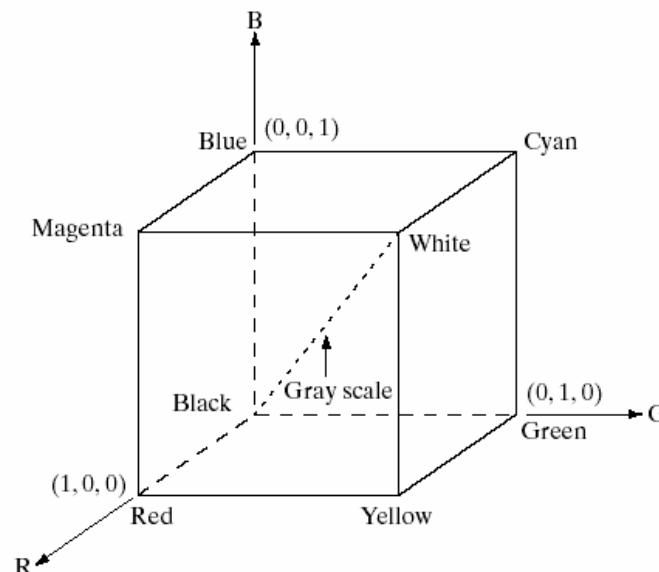


Secondary Colors -  
CMYK

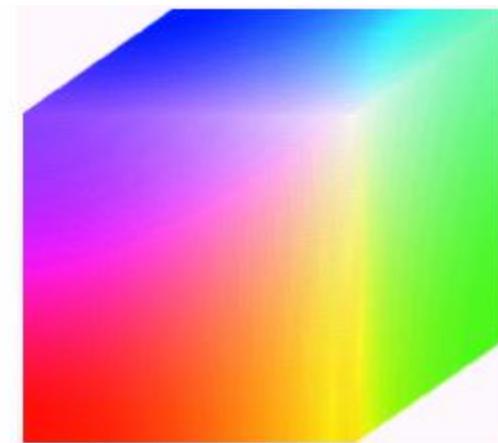
# Color Image Processing

## Color Models

- **RGB Color Model:** In this color model each color appears in its primary spectral components of Red, Green and Blue.
- The model is based on the 3D Cartesian coordinate system, where the color subspace of interest is the color cube shown below.



**Color Cube**  
 $0,0,0=\text{Black}$   
 $1,1,1=\text{White}$

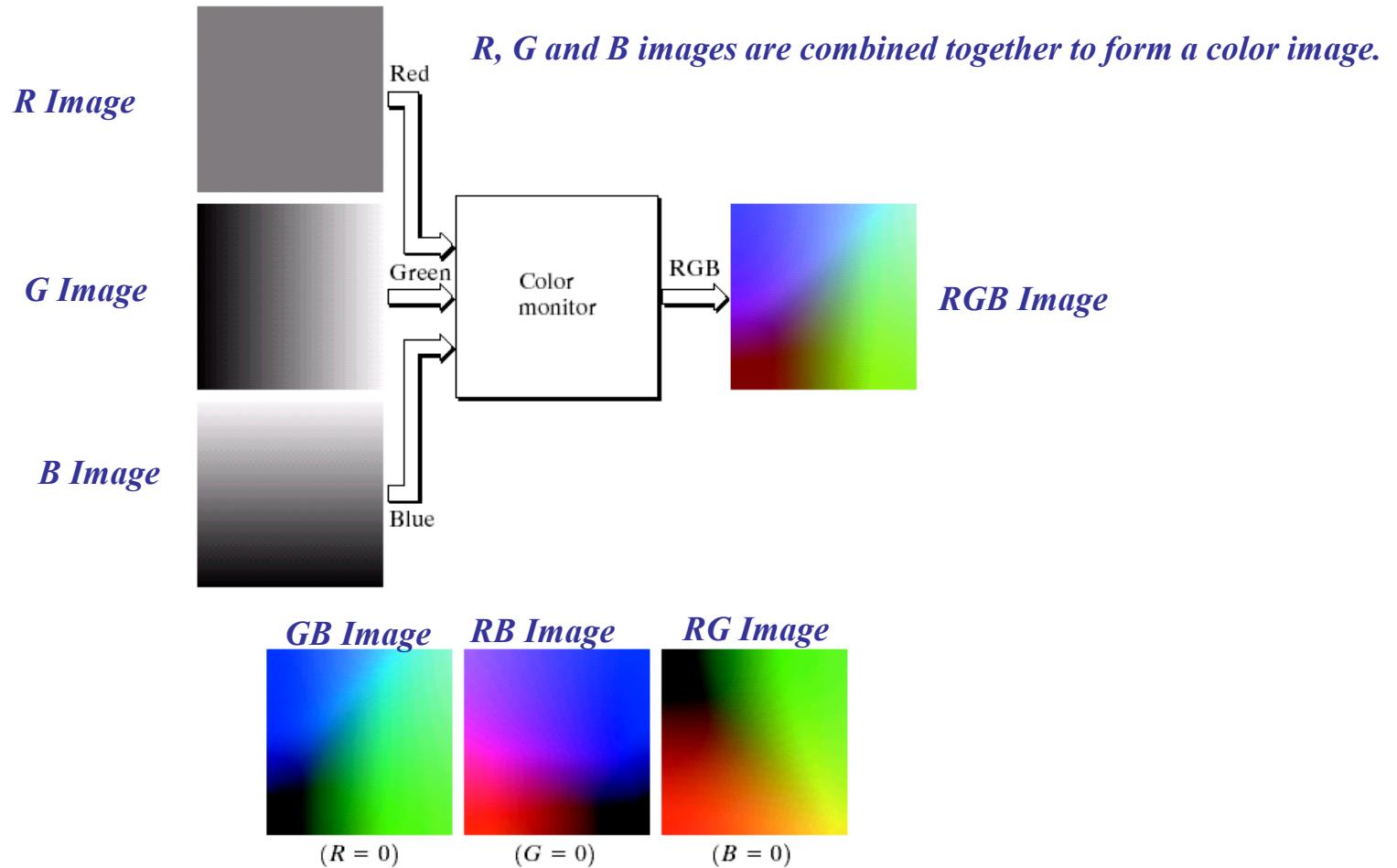


**24-bit Color Cube**

# Color Image Processing

## Color Models

- *RGB Color Model:*



---

# **Color Image Processing**

## **Color Models**

- **CMY Color Model:** *Cyan, Magenta and Yellow color model is made of secondary colors.*
- *Some printers and devices use secondary colors instead of the primary colors.*
- *The conversion from RGB to CMY can be performed as follows:*

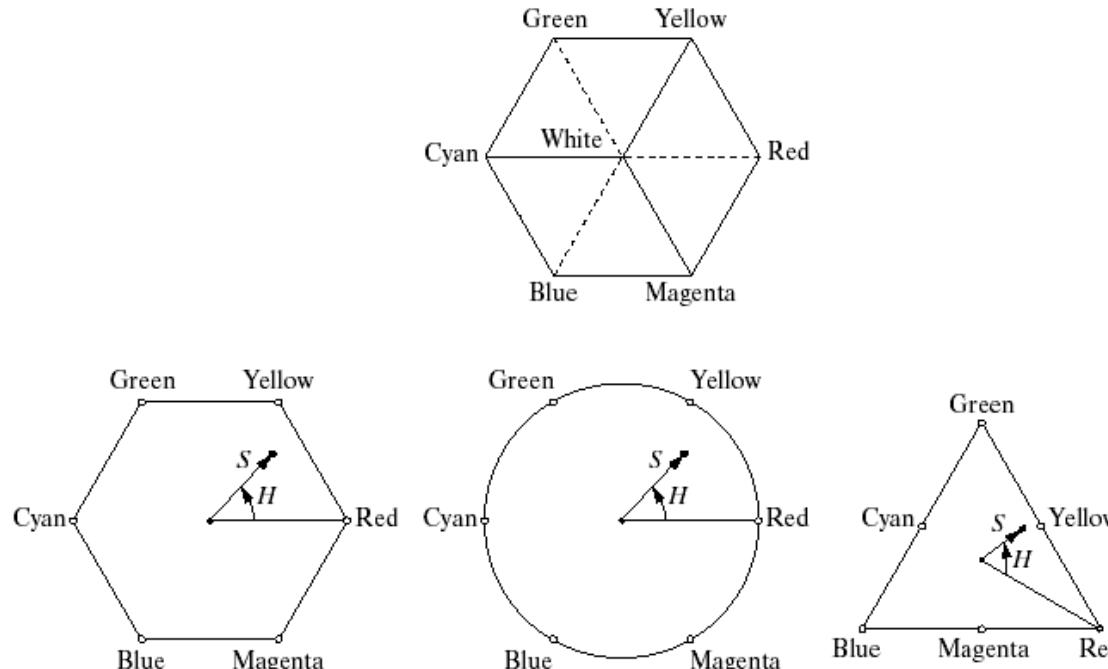
$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

- **HSI Color Model:** *Hue, Saturation and Intensity are three important descriptors used by human being in describing colors.*
- *Hue represents the purity of the color. (i.e. pure red, yellow, green).*
- *Saturation represents the measure of the degree to which a pure color is diluted by white light.*
- *Intensity is the gray level value of the color.*
- *Hue and Saturation represents the color carrying Chrominance (Chromatic) information.*
- *Intensity represents the gray-level Luminance (achromatic) information.*

# Color Image Processing

## Color Models

- **HSI Color Model:** Hue, Saturation and Intensity can be represented by:



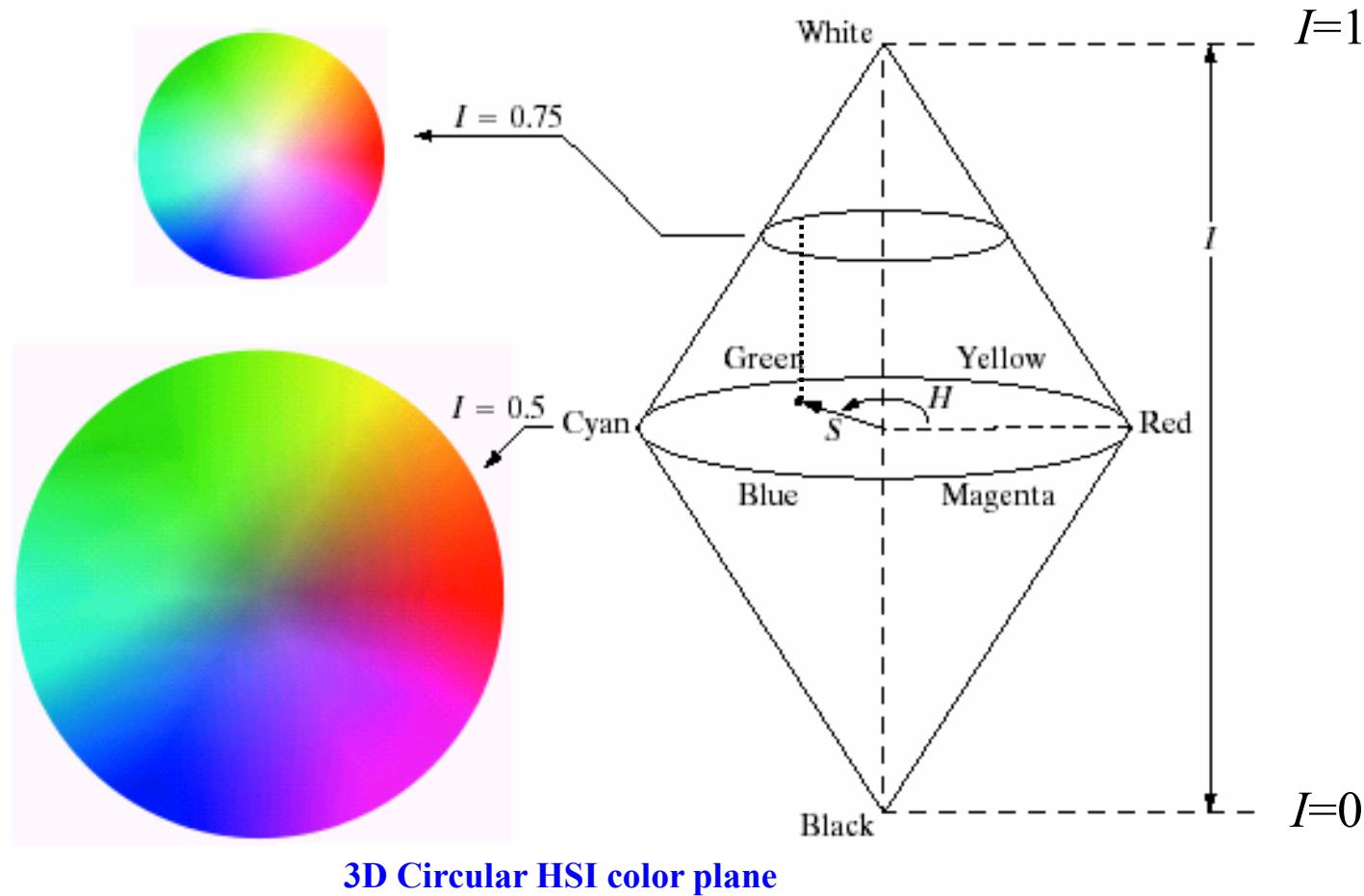
a  
b c d

**FIGURE 6.13** Hue and saturation in the HSI color model. The dot is an arbitrary color point. The angle from the red axis gives the hue, and the length of the vector is the saturation. The intensity of all colors in any of these planes is given by the position of the plane on the vertical intensity axis.

# Color Image Processing

## Color Models

- *HSI Color Model: Hue, Saturation and Intensity*



# **Color Image Processing**

## Color Models

- **Converting Color from RGB to HSI:** Given an image in RGB color format, the H component of each RGB pixel is obtained using:

$$H = \begin{cases} \theta & \text{if } B \leq G \\ 360 - \theta & \text{if } B > G \end{cases}$$

$$\theta = \cos^{-1} \left\{ \frac{\frac{1}{2}[(R-G)+(R-B)]}{\left[ (R-G)^2 + (R-B)(G-B) \right]^{1/2}} \right\}$$

$$S = 1 - \frac{3}{(R+G+B)} [\min(R, G, B)]$$

$$I = \frac{1}{3}(R+G+B)$$

- The RGB values are normalized to the range [0,1].
- The SI values are in [0,1] and the H value can be divided by  $360^\circ$  to be in the same range.

---

# **Color Image Processing**

## Color Models

- *Converting from HSI to RGB : Given the HSI values in the interval of [0,1].*
- *Multiply the H by  $360^\circ$  so that it is in the range of  $[0, 360^\circ]$ .*
- *If  $(0 \leq H < 120^\circ)$ , then color is in RG Sector and then:*

$$B = I(1 - S)$$

$$R = I \left[ 1 + \frac{S \cos H}{\cos(60^\circ - H)} \right]$$

$$G = 1 - (R + B)$$

---

# Color Image Processing

## Color Models

- *Converting from HSI to RGB : Given the HSI values in the interval of [0,1].*
- *Multiply the H by  $360^\circ$  so that it is in the range of  $[0, 360^\circ]$ .*
- *If  $(120^\circ \leq H < 240^\circ)$ , then color is in GB Sector and then. Firstly H in this sector is:*

$$H = H - 120^\circ$$

$$R = I(1 - S)$$

$$G = I \left[ 1 + \frac{S \cos H}{\cos(60^\circ - H)} \right]$$

$$B = 1 - (R + G)$$

---

# Color Image Processing

## Color Models

- *Converting from HSI to RGB : Given the HSI values in the interval of [0,1].*
- *Multiply the H by  $360^\circ$  so that it in the range of  $[0, 360^\circ]$ .*
- *If  $(240^\circ \leq H < 360^\circ)$ , then color is in BR Sector and then. Firstly H in this sector is:*

$$H = H - 240^\circ$$

$$G = I(1 - S)$$

$$B = I \left[ 1 + \frac{S \cos H}{\cos(60^\circ - H)} \right]$$

$$R = 1 - (G + B)$$

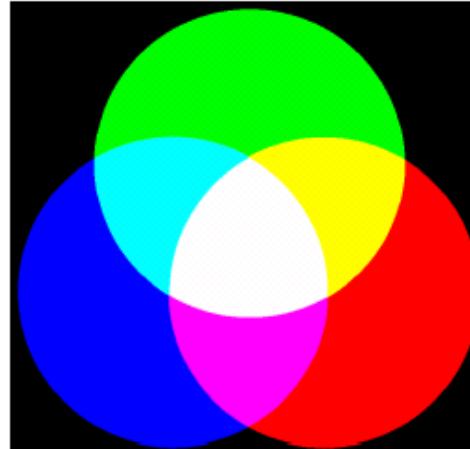
---

# Color Image Processing

## Color Models

- *Converting from HSI to RGB :*

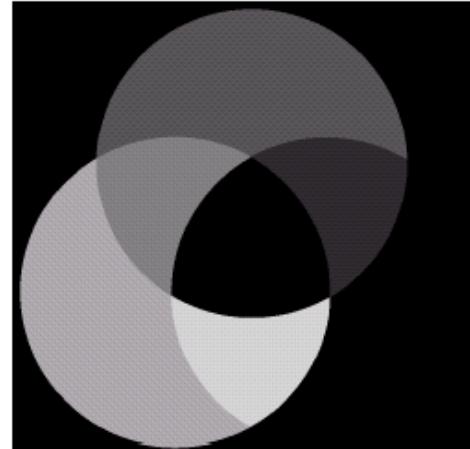
RGB Image



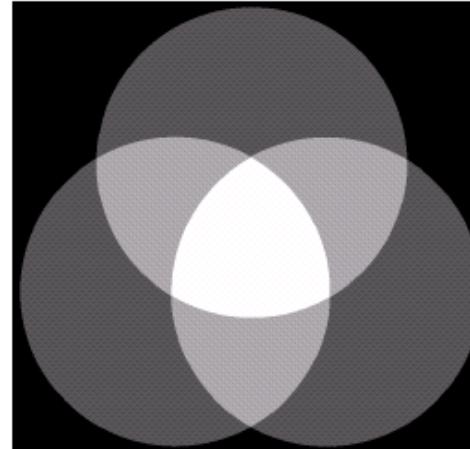
S Image



H Image



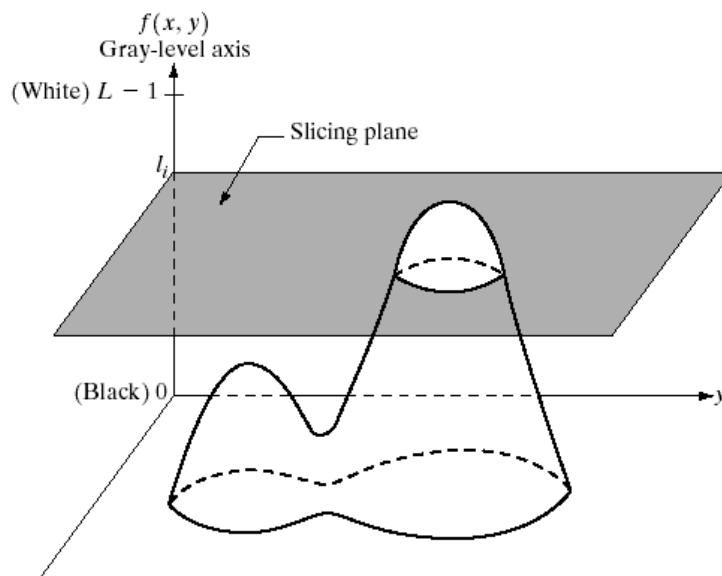
I Image



# Color Image Processing

## Pseudocolor Image Processing

- This technique is based on assigning color (false/pseudo) values to different gray levels. By converting monochrome images to color images human visualization and interpretation of the gray level images can be improved.
- Intensity/density Slicing: The image is interpreted as 3D function (intensity versus spatial coordinates), where, planes which are parallel to the coordinate planes called “slices” are considered to slice the image function into two color levels.



Intensity slicing

Let  $[0, L-1]$  represent the gray level image, and  $l_0$  represents black  $[f(x,y)=0]$  ,  $l_{L-1}$  represents white  $[f(x,y)=L-1]$  , Consider  $P$  planes defined at  $l_1, l_2, \dots, l_P$  intensity levels. Then  $P$  planes partition the gray scale into  $P+1$  intervals,  $V_1, V_2, \dots, V_{P+1}$ . Each interval is assigned to a different color and hence:

$$f(x, y) = c_k \quad , \quad \text{if } f(x, y) \in V_k$$

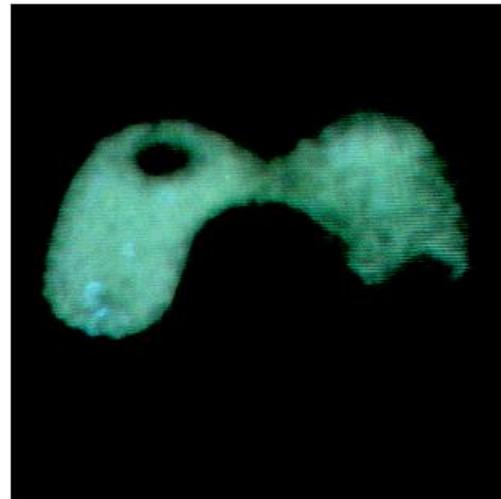
$c_k$  is the color with  $k^{\text{th}}$  intensity interval  $V_k$ .

---

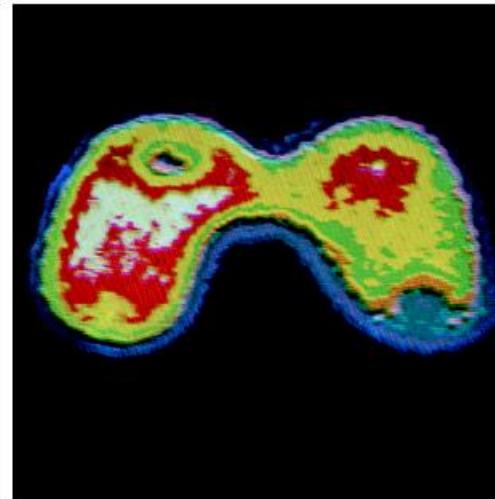
# Color Image Processing

## Pseudocolor Image Processing

- Intensity/density Slicing:



Monochrome Image



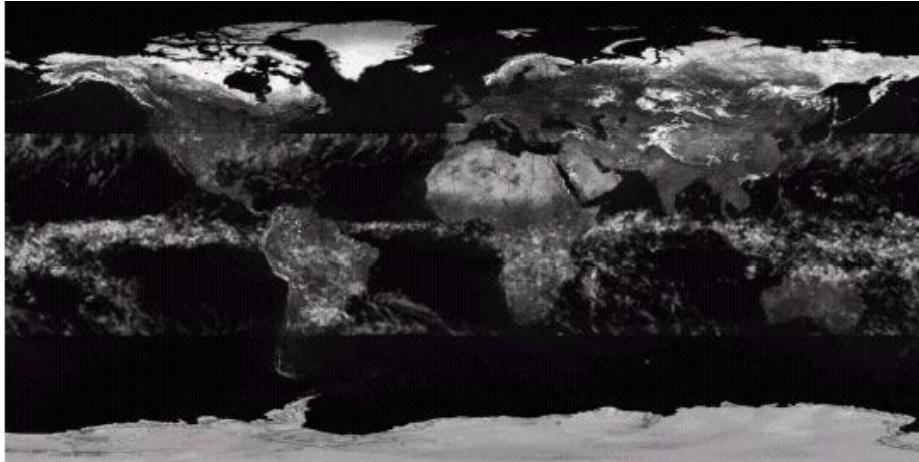
Intensity slicing into 8 colors

# Color Image Processing

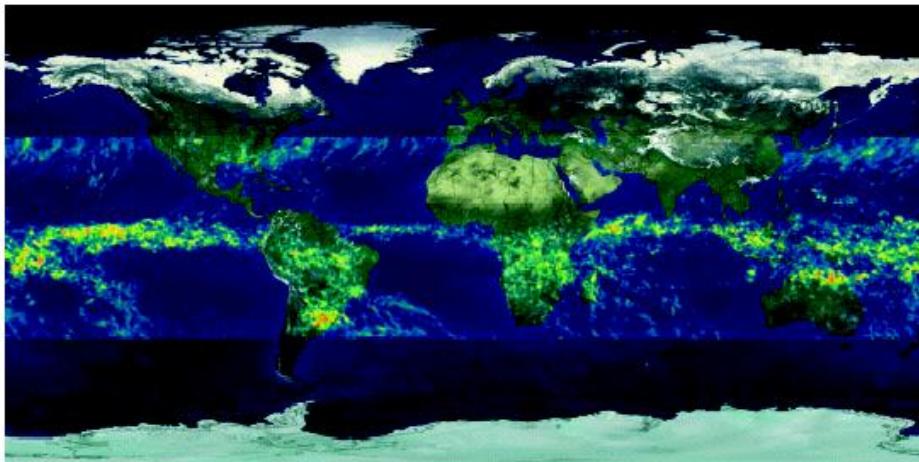
## Pseudocolor Image Processing

- Intensity/density Slicing:

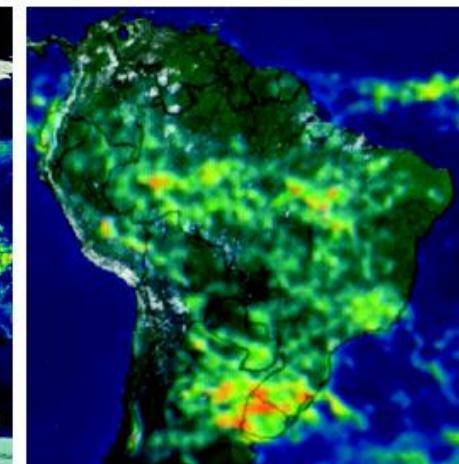
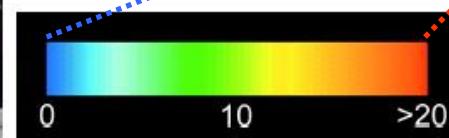
Monochrome Image



After intensity slicing applied  
Tropical regions



intensity slicing into 256 colors  
Intensity values from 0 to 255

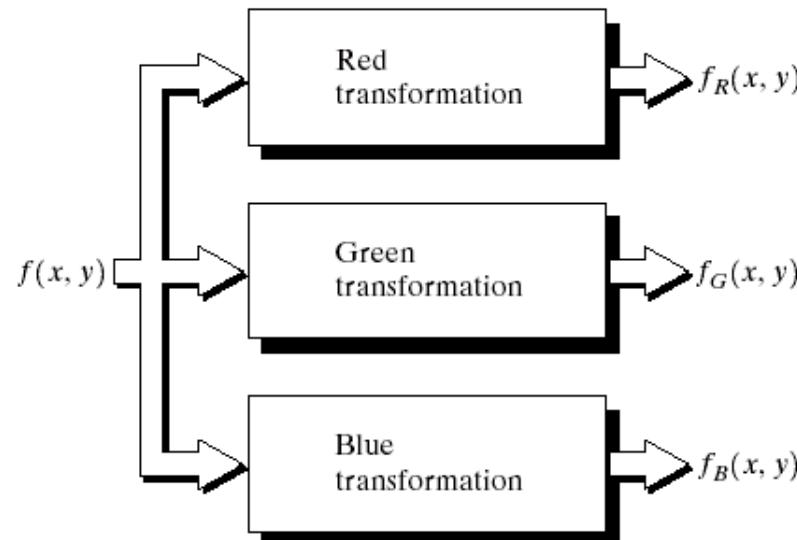


Zoom of south America region

# Color Image Processing

## Pseudocolor Image Processing

- Gray Level to Color Transformations : In this method, a given gray level image is processed by 3 different transformation functions producing 3 enhanced images in Red, Green and Blue channels respectively.
- By combining the 3 channel images we get a colored image.

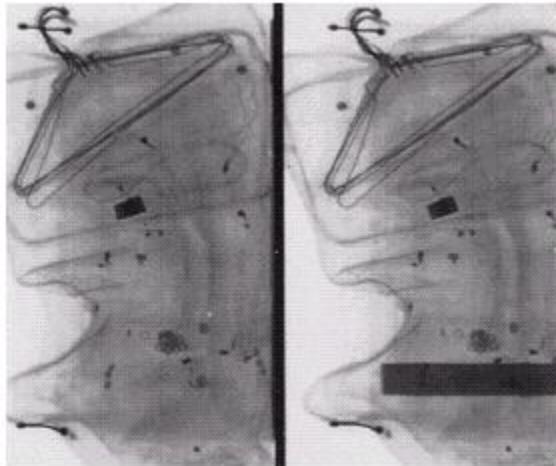


$f_R, f_G$  and  $F_B$  are used to be inputs to an RGB monitor, producing a colored image.

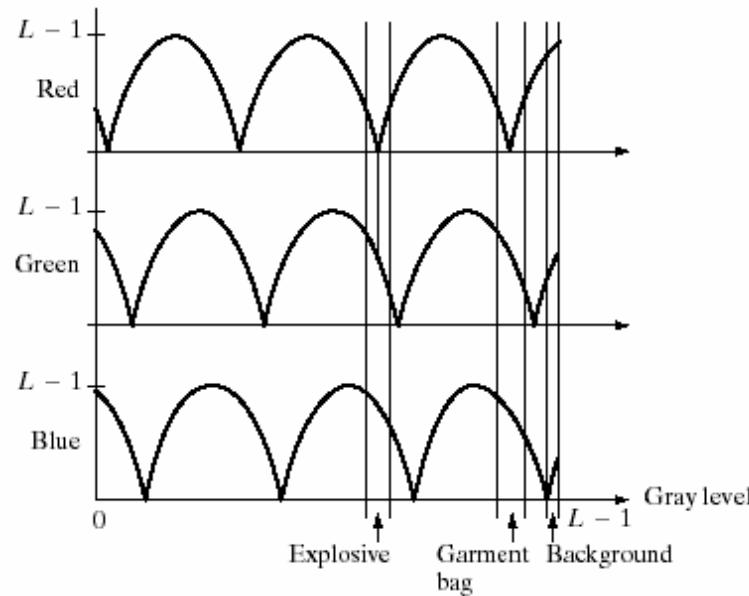
# Color Image Processing

## Pseudocolor Image Processing

- Gray Level to Color Transformations :

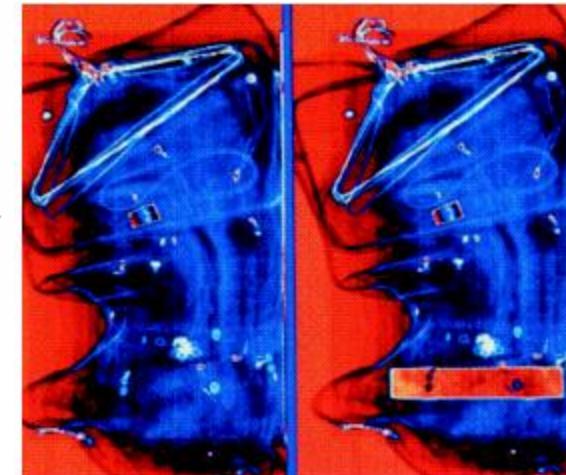


input image.



Transformation functions for R, G and B channels.

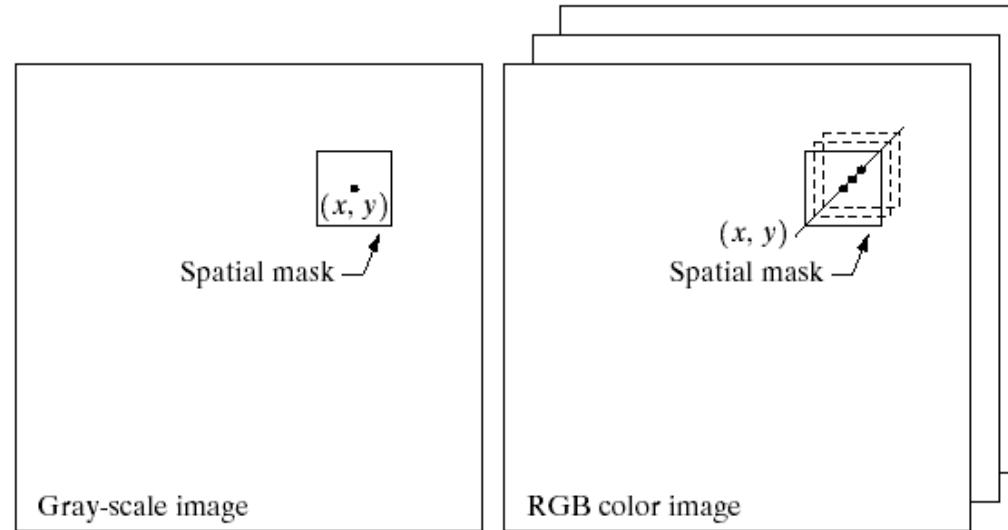
Output Pseudocolor image.



# Color Image Processing

## Full-Color Image Processing

- *There two main methods in using full color images.*
  - *In the first method each color component is processed separately to form a composite color image.*
  - *In the second approach we consider each pixel as a vector of 3 values and process each pixel.*



Each component is processed  
like a gray-level image

Each pixel is considered as a  
vector of RGB components.

# Color Image Processing

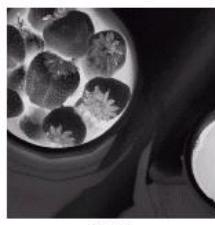
## Full-Color Image Processing

- Various Color space components: Consider each color component as a gray level image.



Full color image

Full color



Cyan



Magenta



Yellow

C,M,Y components.



Red

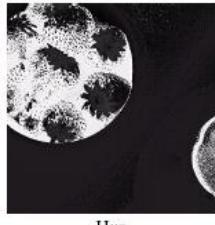


Green



Blue

R,G,B components.



Hue



Saturation



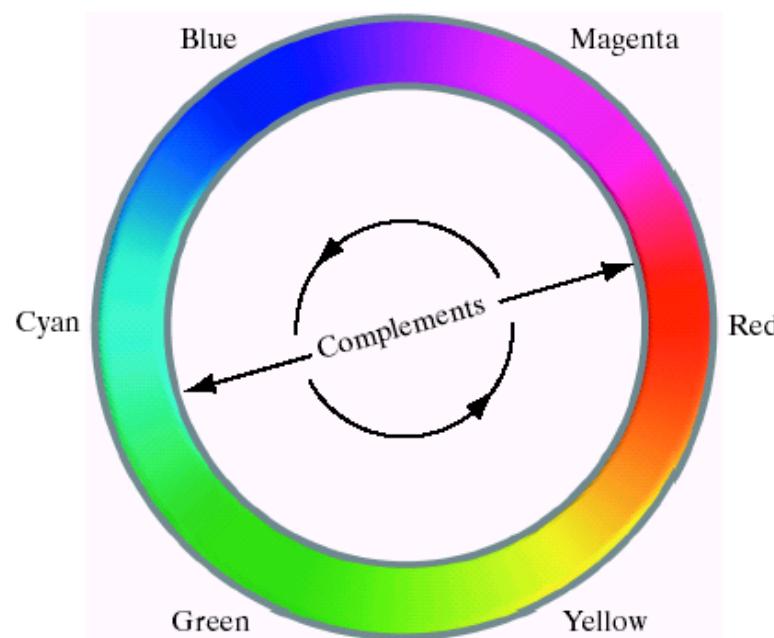
Intensity

H,S,I components.

# Color Image Processing

## Full-Color Image Processing

- Color Complements (inverse colors). Color complementing a color image is identical to gray scale negatives in monochrome images.
- Color complement transformations are performed according to the following color circle.

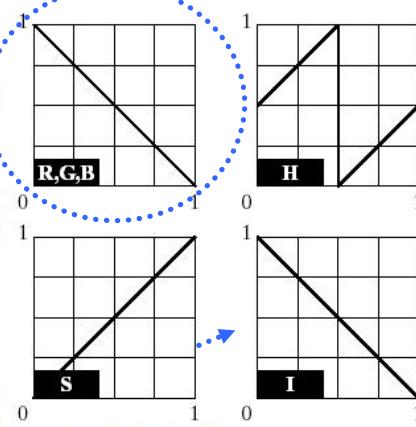


**FIGURE 6.32**  
Complements on  
the color circle.

# Color Image Processing

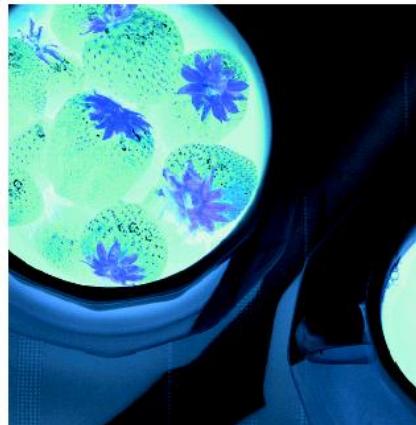
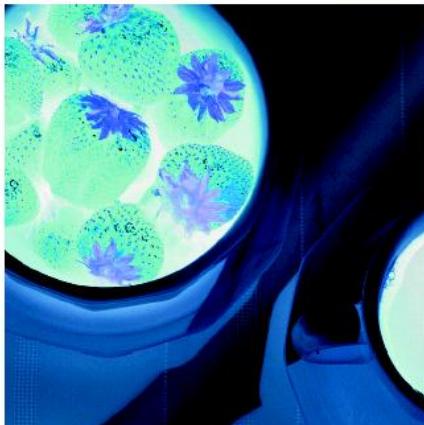
## Full-Color Image Processing

- *Color Complements (inverse colors).* If you apply respective color complement transformation to each color component, you can obtain the complement of a given color image.



Transformation functions for RGB

Transformation functions for HSI



Complementing RGB components

Complementing HSI components

# **Color Image Processing**

## **Full-Color Image Processing**

- **Color Image Smoothing and Sharpening** : The idea of gray-scale image smoothing can be extended into processing of full color images.
- Let  $S_{xy}$  denote the set of coordinates defining a neighborhood centered at  $(x,y)$  in RGB color image. **Averaging (smoothing)** and **Sharpening using Laplacian operator** of the RGB component vectors in this neighborhood is:

### **Smoothing using Averaging**

$$\bar{\mathbf{c}}(x, y) = \frac{1}{K} \sum_{(x, y) \in S_{xy}} \mathbf{c}(x, y)$$

*K is the number of pixels within the neighborhood of the averaging mask.*

$$\bar{\mathbf{c}}(x, y) = \begin{bmatrix} \frac{1}{K} \sum_{(x, y) \in S_{xy}} R(x, y) \\ \frac{1}{K} \sum_{(x, y) \in S_{xy}} G(x, y) \\ \frac{1}{K} \sum_{(x, y) \in S_{xy}} B(x, y) \end{bmatrix}$$

### **Sharpening using Laplacian**

$$\nabla^2 [\mathbf{c}(x, y)] = \begin{bmatrix} \nabla^2 R(x, y) \\ \nabla^2 G(x, y) \\ \nabla^2 B(x, y) \end{bmatrix}$$

# Color Image Processing

## Full-Color Image Processing

- Color Image Smoothing: Given a full color image with the following color components,



Full color RGB image



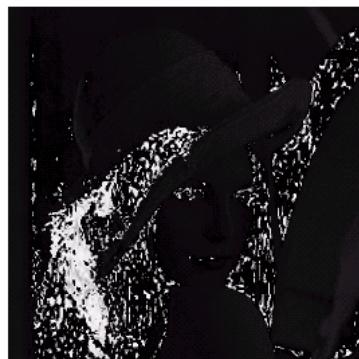
R Image



G Image



B Image



Hue image



Saturation Image



Intensity Image

# Color Image Processing

## Full-Color Image Processing

- Color Image Smoothing & Sharpening:

Smoothing by  
5x5 averaging mask

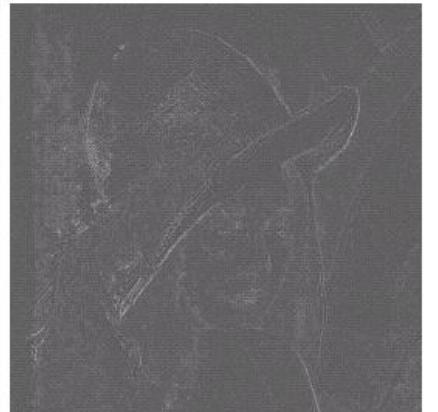


Each RGB component filtered

Only I of the HSI filtered

Difference of the 2 images

Sharpened by  
3x3 Laplacian mask



Each RGB component filtered

Only I of the HSI filtered

Difference of the 2 images

\* Original Hue and Saturation  
are maintained. Working with  
HSI image is a better idea.

---

# Color Image Processing

## Full-Color Image Processing

- Segmentation in RGB vector space : Although working with HSI space is more intuitive in most applications, in segmentation working with RGB color vectors is generally more advantages.
- Suppose that an object within a specified RGB color range is to be segmented. Assume that  $\mathbf{a}$  is the average RGB vector. Each RGB pixel is classified to have color in the specified range/distance from the average color vector.
- Let  $\mathbf{z}$  denote an arbitrary point in RGB space. Then the Euclidean Distance between  $\mathbf{z}$  and  $\mathbf{a}$  is given by:

$$\begin{aligned} D(z, a) &= \|\mathbf{z} - \mathbf{a}\| = \left[ (\mathbf{z} - \mathbf{a})^T (\mathbf{z} - \mathbf{a}) \right]^{1/2} \\ &= \left[ (z_R - a_R)^2 + (z_G - a_G)^2 + (z_B - a_B)^2 \right]^{1/2} \end{aligned}$$

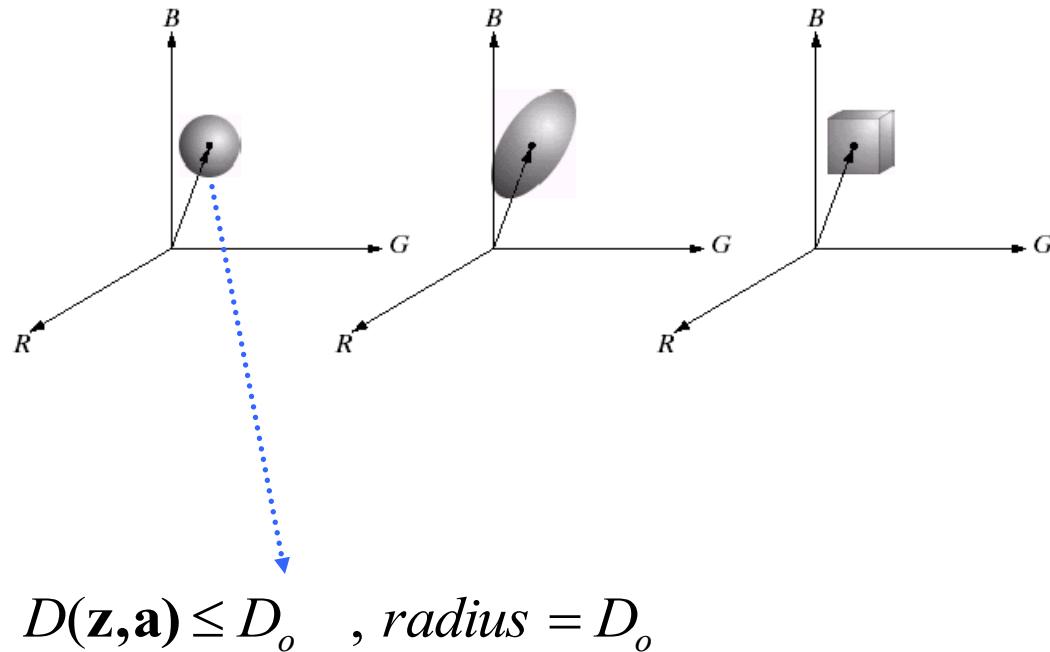
- The segmented pixels fall into the solid sphere of radius  $D_o$  where,

$$D(\mathbf{z}, \mathbf{a}) \leq D_o$$

# Color Image Processing

## Full-Color Image Processing

- Segmentation in RGB vector space :



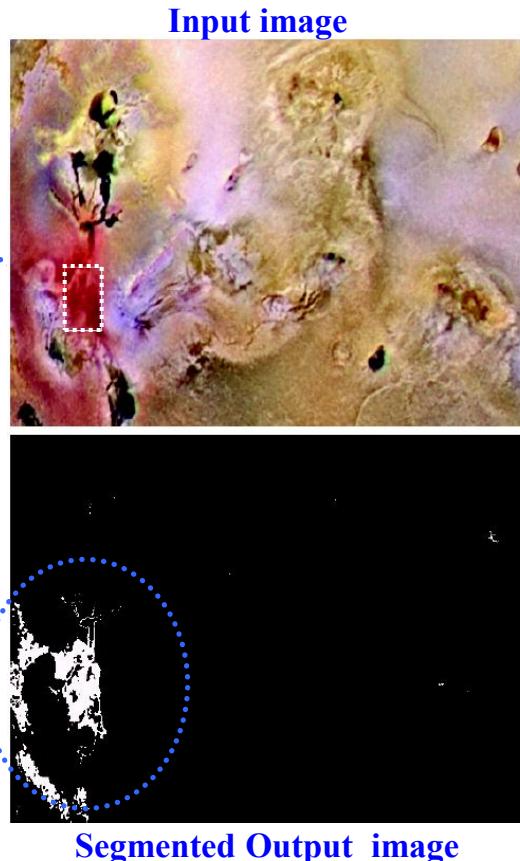
a b c

**FIGURE 6.43**  
Three approaches  
for enclosing data  
regions for RGB  
vector  
segmentation.

# Color Image Processing

## Full-Color Image Processing

- Segmentation in RGB vector space :



### Segmentation Procedure:

1. Sample data is taken from the area of interest.
2. Mean RGB components and their standard deviation values are calculated.
3.  $D_o$  is determined by using the standard deviation  $\sigma$ .  
For example in this example:  
$$D_{oR} = 1.25 \sigma_R$$
$$D_{oG} = 1.25 \sigma_G$$
$$D_{oB} = 1.25 \sigma_B$$
4. Any pixel within  $D_o$  distance from the mean is set to white color and all the other pixels are set to black color.

## Image Compression

### Compression Fundamentals

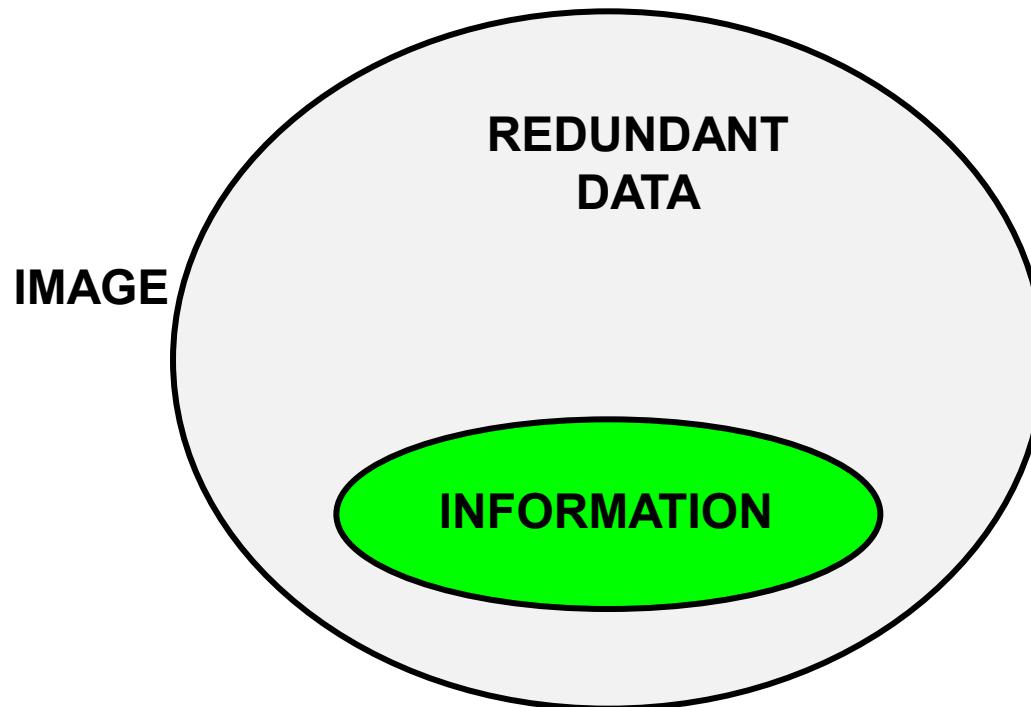
- *Data compression refers to the process of reducing the amount of **data** required to represent given quantity of **information**.*
- *Note that **data** and **information** are not the same. **Data** refers to the means by which the **information** is conveyed.*
- *Various amounts of data can represent the same amount of information.*
- *Sometimes the given data contains some data which has no relevant information, or restates/repeats the known information. It is thus said to contain **data redundancy**.*
- ***Data redundancy** is the central concept in image compression and can be mathematically defined.*

---

# **Image Compression**

## Compression Fundamentals

- *Information versus Data*



$$\text{IMAGE} = \text{INFORMATION} + \text{REDUNDANT DATA}$$

---

# **Image Compression**

## **Compression Fundamentals**

- Given  $n_1$  and  $n_2$  denoting the information-carrying units in two data sets that represent the same information/image.
- The **Relative data redundancy  $R_D$**  of the first data set,  $n_1$ , is defined by:

$$R_D = 1 - \frac{1}{C_R}$$

- $C_R$  refers to the **compression ratio** and is defined by:  $C_R = \frac{n_1}{n_2}$

- If  $n_1 = n_2$ , then  $C_R=1$  and  $R_D=0$ , indicating that the first representation of the information contains no redundant data.

- A typical compression ratio around 10 or(10:1) indicates that 90% ( $R_D=0.9$ ) of the data in the first data set is redundant.

---

# Image Compression

## Data Redundancy

- There are three main *data redundancies* used in image compression.

- *Coding redundancy*
- *Interpixel redundancy*
- *Psychovisual redundancy*

- *Coding Redundancy* : A code is a system of symbols (i.e. bytes, bits) that represents information. Each piece of information is represented by a set of code symbols.
- The gray level *histogram* of an image can be used in construction of codes to reduce the data used to represent it. Given the normalized histogram of a gray level image where,

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

- $r_k$  is the pixel values defined in the interval  $[0,1]$  and  $p_r(k)$  is the probability of occurrence of  $r_k$ .  $L$  is the number of gray levels.  $n_k$  is the number of times that  $k^{\text{th}}$  gray level appears and  $n$  is the total number of pixels.

# Image Compression

## Data Redundancy

- **Coding Redundancy**

- *Average number of bits required to represent each pixel is given by:*

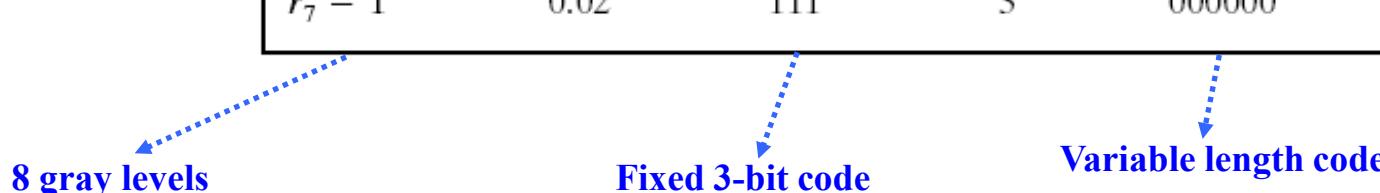
$$L_{avg} = \sum_{k=1}^{L-1} l(r_k) p_r(r_k)$$

- *Where,  $l(r_k)$  is the number of bits used to represent each value of  $r_k$*

- *An 8 gray level image has the following gray level distribution.*

$r_k$	$p_r(r_k)$	Code 1	$I_1(r_k)$	Code 2	$I_2(r_k)$
$r_0 = 0$	0.19	000	3	11	2
$r_1 = 1/7$	0.25	001	3	01	2
$r_2 = 2/7$	0.21	010	3	10	2
$r_3 = 3/7$	0.16	011	3	001	3
$r_4 = 4/7$	0.08	100	3	0001	4
$r_5 = 5/7$	0.06	101	3	00001	5
$r_6 = 6/7$	0.03	110	3	000001	6
$r_7 = 1$	0.02	111	3	000000	6

**TABLE 8.1**  
Example of  
variable-length  
coding.



# Image Compression

## Data Redundancy

- **Coding Redundancy**

- **The average number of bit used for fixed 3-bit code:**

$$L_{avg} = \sum_{k=0}^7 l_1(r_k) p_r(r_k) = 3 \sum_{k=0}^7 p_r(r_k) = 3.1 = 3 \text{ bits}$$

- **The average number of bits used for variable-length code in this particular example:**

$$\begin{aligned} L_{avg} &= \sum_{k=0}^7 l_1(r_k) p_r(r_k) = 2(0.19) + 2(0.25) + 2(0.21) + \\ &\quad 3(0.16) + 4(0.08) + 5(0.06) + 6(0.03) + 6(0.02) \\ &= 2.7 \text{ bits} \end{aligned}$$

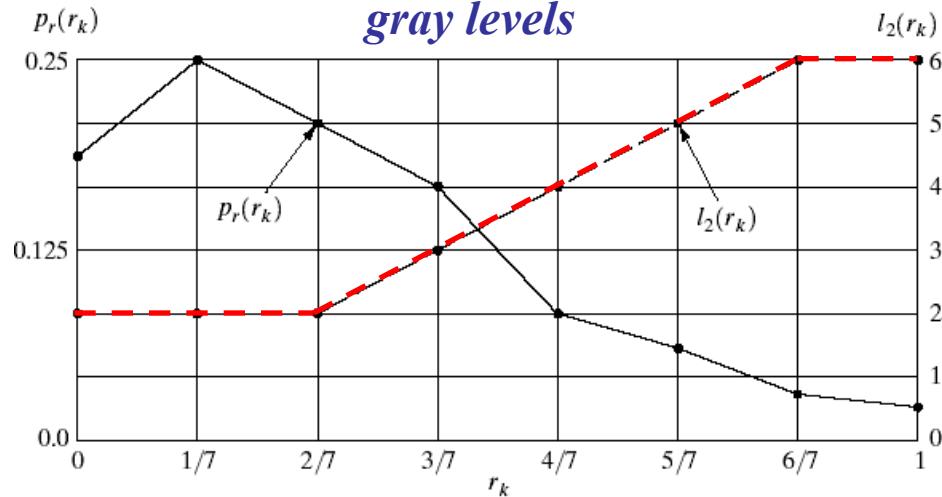
- **The compression ratio:**  $C_R = \frac{3}{2.7} = 1.11$

- **The relative Data Redundancy:**  $R_D = 1 - \frac{1}{1.11} = 0.099 \Rightarrow \sim \%10$

# Image Compression

## Data Redundancy

- **Coding Redundancy**
- In this example the suggested variable-length coding gets rid of the ~10% redundant data of the fixed 3-bit code.
- The following graph shows the relationship between the histogram of an image,  $p_r(r_k)$  and  $l_2(r_k)$  which are inversely proportional.
  - The shortest code words are assigned to the most frequent (high probability) gray levels
  - The longest code words are assigned to the least frequent (low probability) gray levels



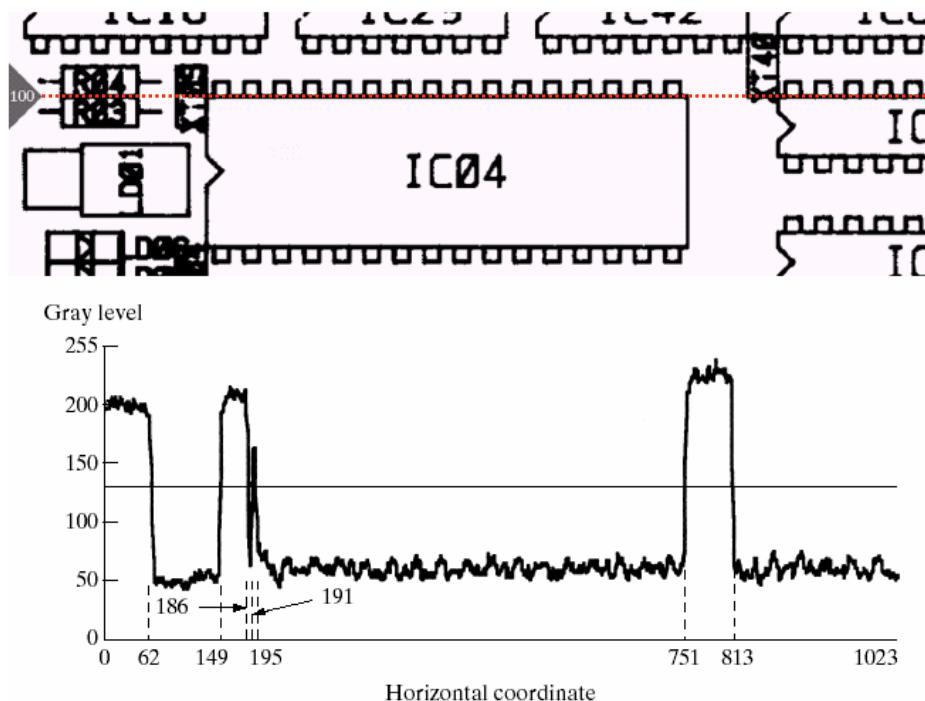
*Data compression is achieved by assigning fewer bits to more probable gray levels than the less probable gray levels.*

# Image Compression

## Data Redundancy

### *•Interpixel Redundancy*

- This type of redundancy is related with the interpixel correlations within an image.*
- Much of the visual contribution of a single pixel is redundant and can be guessed from the values of its neighbors.*



- Given a 1024x343 binary image*
- Consider a line crossing the image at line 100*
- The respective line of 1024 bits can be represented by the Run-length code given at the bottom.*
- Note that in this line there is 8 regions that are 1 or 0 with the specified run-length. Total of 11 bits (1 bit for thresholded value and 10 bit for the run length) can be used to represent each of these 8 neighborhoods.*

**Line 100: (1,63)(0,87)(1,37)(0,5)(1,4)(0,556)(1,62)(0,210)**

# Image Compression

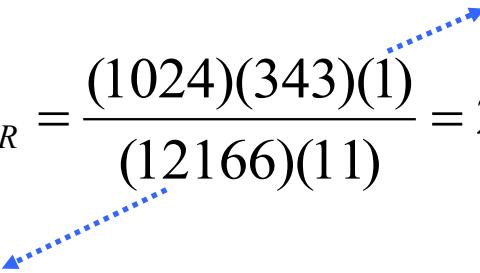
## Data Redundancy

- *Interpixel Redundancy*

- *The resulting compression ratio and respective relative redundancy is given by:*

$$C_R = \frac{(1024)(343)(1)}{(12166)(11)} = 2.63$$

1-bit for each pixel



Determined by thresholding each line and counting the run-length regions

$$R_D = 1 - \frac{1}{2.63} = 0.62$$

- *The relative redundancy is %62 obtained only by using correlation among the pixels (interpixel dependencies) in the given image.*
- *This method can be extended to gray level images.*

---

# **Image Compression**

## **Data Redundancy**

- Psychovisual Redundancy***

- Certain information has relatively less importance for the quality of image perception. This information is said to be psychovisually redundant.***

- Unlike coding and interpixel redundancies, the psychovisual redundancy is related with the real/quantifiable visual information. Its elimination results a loss of quantitative information. However psychovisually the loss is negligible.***

- Removing this type of redundancy is a lossy process and the lost information cannot be recovered.***

- The method used to remove this type of redundancy is called **quantization** which means the mapping of a broad range of input values to a limited number of output values.***

# Image Compression

## Data Redundancy

- *Psychovisual Redundancy*

- *The following example shows how an 8-bit image can be reduced to 4-bit image.*



8-bit image



4-bit image

Uniform quantization  
Undesired contouring  
effect



4-bit image

IGS quantization  
No contouring effect

# Image Compression

## Data Redundancy

- ***Psychovisual Redundancy***

- ***The improved gray-scale quantization (IGS) is one of the possible quantization procedures and summarized in the following table.***

Pixel	Gray Level	Sum	IGS Code
$i - 1$	N/A	0000 0000	N/A
$i$	0110 1100	0110 1100	0110
$i + 1$	1000 1011	1001 0111	1001
$i + 2$	1000 0111	1000 1110	1000
$i + 3$	1111 0100	1111 0100	1111

- ***The IGS Quantization Procedure:***

- ***Add the Least Significant Nibble (4 bits) of the previous sum to the current 8-bit pixel value.***
- ***If the MSN of a given 8-bit pixel is  $1111_2$  than add zero instead.***
- ***Declare the Most Significant Nibble of the sum to be the 4-bit IGS code.***

---

# **Image Compression**

## **Data Redundancy**

- ***Quality Measure of a Compressed Image (Fidelity Criteria):***
- ***The removal of psychovisual redundancy removes real/quantitative information and provides lossy image compression.***
- ***The quality of such images can be evaluated by objective and subjective methods.***
- ***The objective quality measures:***
  - ***The mean-square-error between 2 images (original versus the compressed)***

$$e_{MSE} = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2$$

- ***The root mean-square-error:***

$$e_{RMSE} = \sqrt{\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2}$$

# Image Compression

## Data Redundancy

- *Quality Measure of a Compressed Image:*

- *The objective quality measures:*

- Peak Signal to Noise Ratio (PSNR) – in decibel (dB):

$$PSNR = 10 \log_{10} \frac{(2^B - 1)^2}{e_{MSE}}$$

*B is the number of bits used for each pixel. (i.e. 8 bits)*

$$(for\ 8-bit\ images \Rightarrow PSNR = 10 \log_{10} \frac{(255)^2}{e_{MSE}} = 20 \log_{10} \frac{255}{e_{RMSE}})$$

- The mean-square signal-to-noise-ratio:

$$SNR_{ms} = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \hat{f}(x, y)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2}$$

# Image Compression

## Data Redundancy

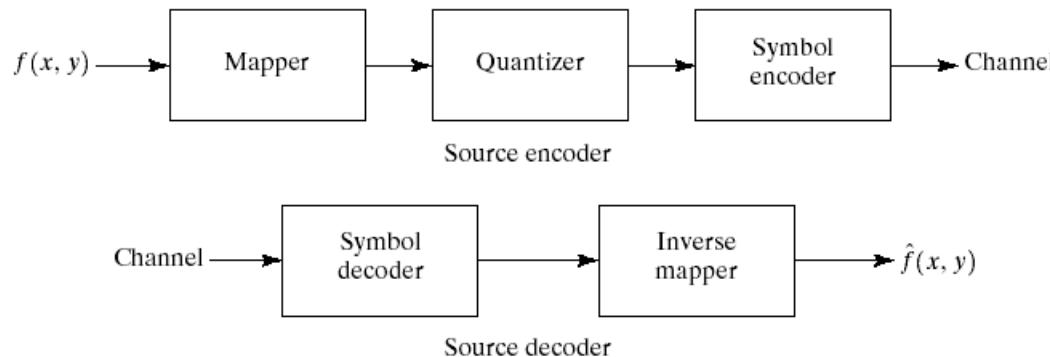
- *Subjective evaluations used for Image quality rating.*
  - *Human observers can be used to provide subjective evaluations.*

<b>Value</b>	<b>Rating</b>	<b>Description</b>
1	Excellent	An image of extremely high quality, as good as you could desire.
2	Fine	An image of high quality, providing enjoyable viewing. Interference is not objectionable.
3	Passable	An image of acceptable quality. Interference is not objectionable.
4	Marginal	An image of poor quality; you wish you could improve it. Interference is somewhat objectionable.
5	Inferior	A very poor image, but you could watch it. Objectionable interference is definitely present.
6	Unusable	An image so bad that you could not watch it.

# Image Compression

## Image Compression Models

- *The Source Encoder and Decoder:*



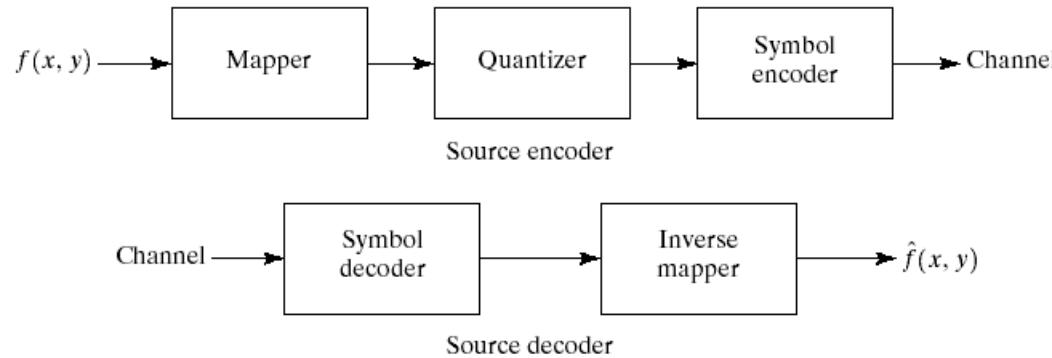
- *The Source Encoder is reduces/eliminates any coding, interpixel or psychovisual redundancies. The Source Encoder contains 3 processes:*

- **Mapper:** Transforms the image into array of coefficients reducing interpixel redundancies. This is a reversible process which is **not lossy**.
- **Quantizer:** This process reduces the accuracy and hence psychovisual redundancies of a given image. This process is irreversible and therefore **lossy**.
- **Symbol Encoder:** This is the source encoding process where fixed or variable-length code is used to represent mapped and quantized data sets. This is a reversible process. Removes coding redundancy by assigning shortest codes for the most frequently occurring output values.

# Image Compression

## Image Compression Models

- *The Source Encoder and Decoder:*



- *The Source Decoder contains two components.*
  - *Symbol Decoder: This is the inverse of the symbol encoder and reverse of the variable-length coding is applied.*
  - *Inverse Mapper : Inverse of the removal of the interpixel redundancy.*
- *The only lossy element is the Quantizer which removes the psychovisual redundancies causing irreversible loss. Every Lossy Compression methods contains the quantizer module.*
- *If error-free compression is desired the quantizer module is removed.*

---

# Image Compression

## Information Theory-Entropy

- **Measuring Information:** *The information in an image can be modeled as a probabilistic process, where we first develop a statistical model of the image generation process. The **information content (entropy)** can be estimated based on this model.*
- *The information per source (symbol or pixel), which is also referred as **entropy** is calculated by:*

$$E = - \sum_{j=1}^J P(a_j) \log P(a_j)$$

- *Where  $P(a_j)$  refers to the source symbol/pixel probabilities.  $J$  refers to the number of symbols or different pixel values.*
- *For example, given the following Image segment:*

21	21	21	95	169	243	243	243
21	21	21	95	169	243	243	243
21	21	21	95	169	243	243	243
21	21	21	95	169	243	243	243

---

# Image Compression

## Information Theory-Entropy

- **Measuring Information:** The entropy of the given 8-bit image segment can be calculated by:

Gray Level	Count	Probability
21	12	3/8
95	4	1/8
169	4	1/8
243	12	3/8

- The entropy of this image is calculated by:

$$\begin{aligned} E = -\sum_{j=1}^J P(a_j) \log P(a_j) &= -[(3/8) \log(3/8) + (1/8) \log(1/8) \\ &\quad + (1/8) \log(1/8) + (3/8) \log(3/8)] \\ &= 1.81 \text{ bits / pixel}. \end{aligned}$$

---

# **Image Compression**

## **Error-Free Compression**

- **Error-free compression** is generally composed of two relatively independent operations: (1) reduce the **interpixel redundancies** and (2) introduce a coding method to reduce the **coding redundancies**.
  - The coding redundancy can be minimized by using a **variable-length coding** method where the shortest codes are assigned to most probable gray levels.
  - The most popular variable-length coding method is the **Huffman Coding**.
- 
- **Huffman Coding:** The Huffman coding involves the following 2 steps.
    - 1) Create a series of source reductions by ordering the probabilities of the symbols and combining the lowest probability symbols into a single symbol and replace in the next source reduction.
    - 2) Code each reduced source starting with the smallest source and working back to the original source. Use 0 and 1 to code the simplest 2 symbol source.

# Image Compression

## Error-Free Compression

- **Huffman Coding:**

Original source		Source reduction			
Symbol	Probability	1	2	3	4
$a_2$	0.4	0.4	0.4	0.4	0.6
$a_6$	0.3	0.3	0.3	0.3	0.4
$a_1$	0.1	0.1	0.2	0.3	
$a_4$	0.1	0.1	0.1		
$a_3$	0.06	0.1			
$a_5$	0.04				

**1) Huffman source reductions:**

$a_i$ 's corresponds to the available gray levels in a given image.

**2) Huffman code assignments:**

The first code assignment is done for  $a_2$  with the highest probability and the last assignments are done for  $a_3$  and  $a_5$  with the lowest probabilities.

Original source			Source reduction							
Sym.	Prob.	Code	1	2	3	4	5	6	7	8
$a_2$	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
$a_6$	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
$a_1$	0.1	011	0.1	011	0.2	010	0.3	01		
$a_4$	0.1	0100	0.1	0100	0.1	011	0.1	011		
$a_3$	0.06	01010	0.1	0101						
$a_5$	0.04	01011								

First code

Last code

---

# **Image Compression**

## Error-Free Compression

• **Huffman Coding:** Note that the shortest codeword (1) is given for the symbol/pixel with the highest probability ( $a_2$ ). The longest codeword (01011) is given for the symbol/pixel with the lowest probability ( $a_5$ ).

• The average length of the code is given by:

$$\begin{aligned}L_{avg} &= (0.4)(1) + (0.3)(2) + (0.1)(3) + (0.1)(4) + (0.06)(5) + (0.04)(5) \\&= 2.2 \text{ bits / symbol}\end{aligned}$$

• The entropy of the source is given by:

$$E = -\sum_{j=1}^J P(a_j) \log P(a_j) = 2.14 \text{ bits / symbol.}$$

• The resulting Huffman coding efficiency is %97.3 ( $2.14/2.2$ ). Note that Huffman Coding is not optimal and many more efficient versions of it as well as other variable-length coding methods can be used.

---

# **Image Compression**

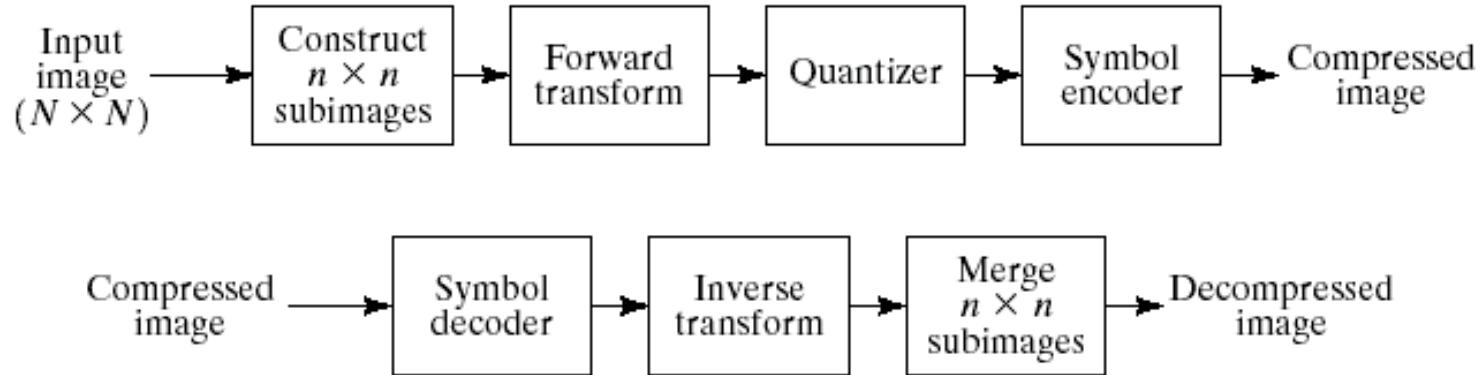
## **Lossy Compression**

- *Unlike the error-free compression, lossy encoding is based on the concept of compromising the accuracy of the reconstructed image in exchange for increased compression.*
- *The lossy compression method produces distortion which is irreversible. On the other hand, very high compression ratios ranging between 10:1 to 50:1 can be achieved with visually indistinguishable from the original. The error-free methods rarely give results more than 3:1.*
- **Transform Coding:** *Transform coding is the most popular lossy image compression method which operates directly on the pixels of an image.*
- *The method uses a reversible transform (i.e. Fourier, Cosine transform) to map the image into a set of transform coefficients which are then quantized and coded.*
- *The goal of the transformation is to decorrelate the pixels of a given image block such the most of the information is packed into smallest number of transform coefficients.*

# Image Compression

## Lossy Compression

- **Transform Coding:**



A Transform Coding System: encoder and decoder.

- **Transform Selection:** *The system is based on discrete 2D transforms. The choice of a transform in a given application depends on the amount of the reconstruction error that can be tolerated and computational resources available.*
- **Consider an  $N \times N$  image  $f(x,y)$ , where the forward discrete transform  $T(u,v)$  is given by:**

$$T(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) g(x,y,u,v)$$

- **For  $u, v=0,1,2,3,..,N-1$ .**

---

# Image Compression

## Lossy Compression

- **Transform Selection :**

- **The inverse transform is defined by:**

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) h(x, y, u, v)$$

- **The  $g(x, y, u, v)$  and  $h(x, y, u, v)$  are called the forward and inverse transformation kernels respectively.**
- **The most well known transform kernel pair is the Discrete Fourier Transform (DFT) pair:**

$$g(x, y, u, v) = \frac{1}{N^2} e^{-j2\pi(ux+vy)/N} \quad \text{and} \quad h(x, y, u, v) = e^{j2\pi(ux+vy)/N}$$

- **Another computationally simpler transformation is called the Walsh-Hadamard Transform (WHT), which is derived from the following identical kernels:**

$$g(x, y, u, v) = h(x, y, u, v) = \frac{1}{N} \left( -1 \right)^{\sum_{i=0}^{m-1} [b_i(x)p_i(u) + b_i(y)p_i(v)]}$$

---

# Image Compression

## Lossy Compression

- **Transform Selection :**

- In WHT,  $N=2^m$ . The summation in the exponent is performed in **modulo 2** arithmetic and  $b_k(z)$  is the  $k^{\text{th}}$  bit (from right to left) in the binary representation of  $z$ .

- For example if  $m=3$  and  $z=6$  ( $110_2$ ), then  $b_0(z)=0$ ,  $b_1(z)=1$ ,  $b_2(z)=1$ . Then the  $p_i(u)$  values are computed by:

$$p_0(u) = b_{m-1}(u)$$

$$p_1(u) = b_{m-1}(u) + b_{m-2}(u)$$

$$p_2(u) = b_{m-2}(u) + b_{m-3}(u)$$

$$\vdots$$

$$p_{m-1}(u) = b_1(u) + b_0(u)$$

- The sums are performed in modulo 2 arithmetic. The  $p_i(v)$  values are computed similarly.

# Image Compression

## Lossy Compression

- **Transform Selection :**

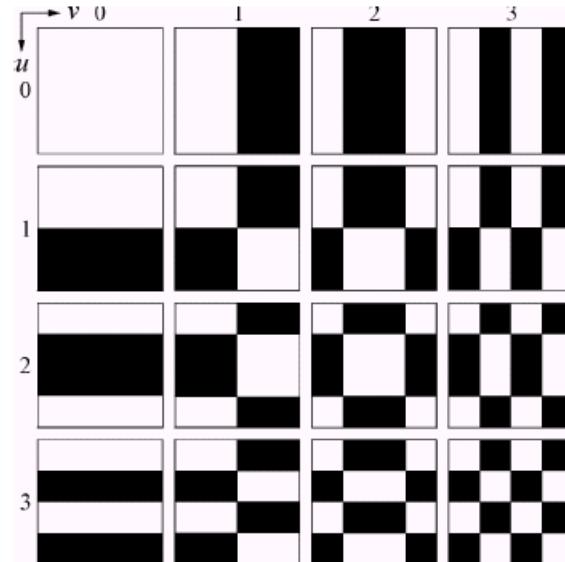
- *Unlike the kernels of DFT , which are the sums of the sines and cosines, the Walsh-Hadamard kernels consists of alternating plus and minus 1's arranged in a checkboard pattern.*

- *Each block consists of  $4 \times 4 = 16$  elements ( $n=4$ ).*

*White denotes +1 and black denotes -1.*

- *When  $u=v=0$   $g(x,y,0,0)$  for  $x,y=0,1,2,3$ . All values are +1.*

- *The importance of WHT is its simplicity of its implementation. All kernel values are +1 or -1.*



**FIGURE 8.29** Walsh-Hadamard basis functions for  $N = 4$ . The origin of each block is at its top left.

---

# Image Compression

## Lossy Compression

- **Transform Selection :**

- ***One of the most frequently used transformation for image compression is the discrete cosine transform (DCT). The kernels pairs are equal and given by:***

$$g(x, y, u, v) = h(x, y, u, v) = \alpha(u) \cos\left[\frac{(2x+1)u\pi}{2N}\right] \cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

- **Where,**

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } u = 0 \\ \sqrt{\frac{2}{N}} & \text{for } u = 1, 2, \dots, N-1 \end{cases}$$

$\alpha(v)$

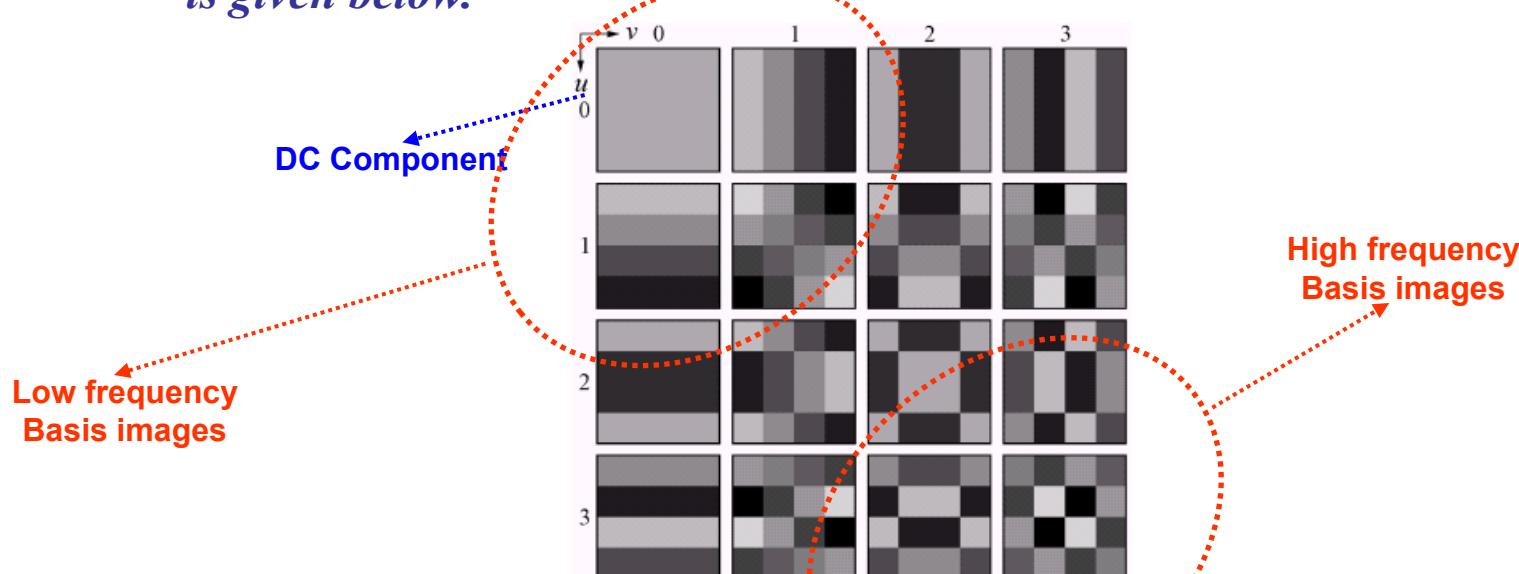
- $\alpha(v)$  is similarly determined. Unlike WHT the values of  $g$  are not integer values (-1 and +1), the DCT contains intermediate gray level values.

# Image Compression

## Lossy Compression

• *Transform Selection :*

• discrete cosine transform (DCT). The  $g(x,y,u,v)$  kernels (basis images) for  $N=4$  is given below.

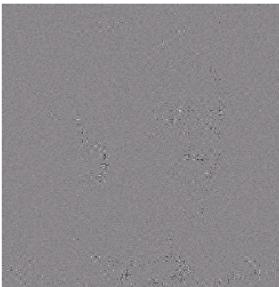
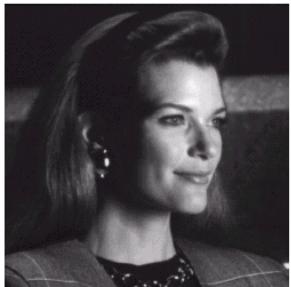


**FIGURE 8.30** Discrete-cosine basis functions for  $N = 4$ . The origin of each block is at its top left.

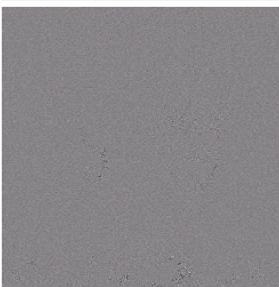
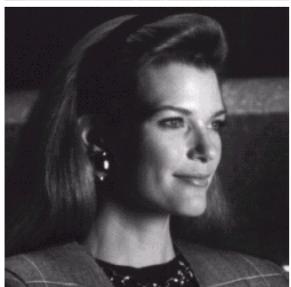
# Image Compression

## Lossy Compression

**• Transform Selection :** Given a 512x512 image . The image is divided into 8x8 sub-images and the respective transforms are applied. After truncating 50% of the transform coefficients and inverse transformation the following results are obtained for DFT, WHT and DCT.



DFT based reconstruction and error image,  $e_{rms}=1.28$



WHT based reconstruction and error image,  $e_{rms}=0.86$

DCT based reconstruction and error image,  $e_{rms}=0.68$

• Note that the first 32 coefficients with lowest magnitudes are eliminated/truncated.

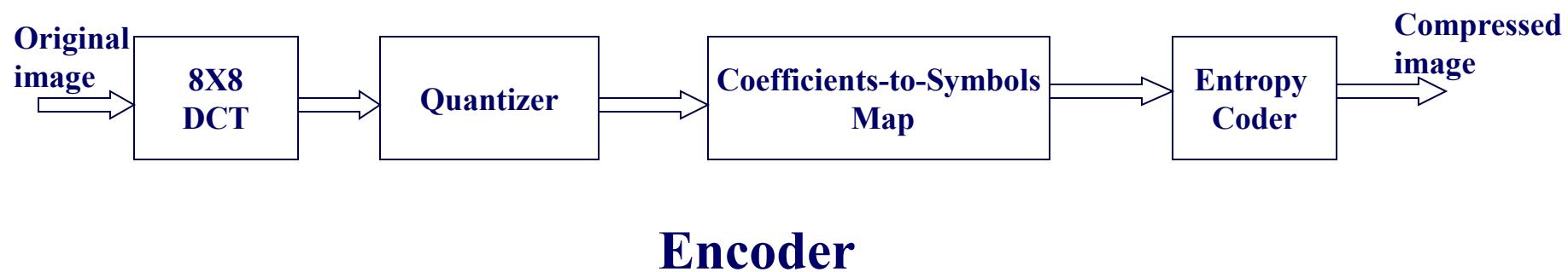
• Their contribution to the image is small and this causes some error on the reconstructed image.

• **DCT is the best among the given 3 transforms where DCT better describes the image than the WHT.**

• **DFT causes blocking artifacts at the borders which makes it worse than the DCT. Furthermore, DFT uses complex coefficients that is more computationally expensive to implement.**

# Image Compression

## DCT-based JPEG (Joint Photographic Expert Group) Standard

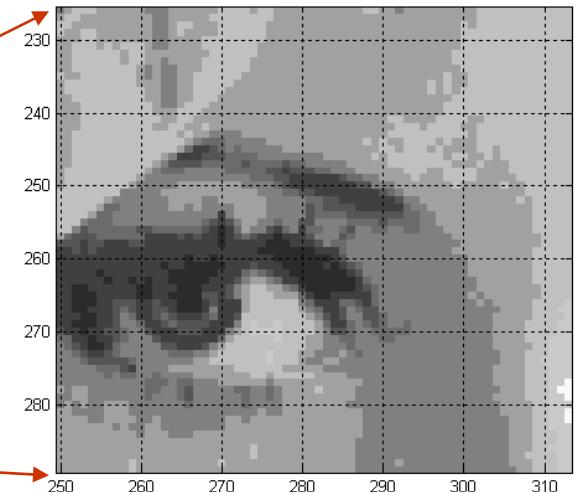


---

# Image Compression

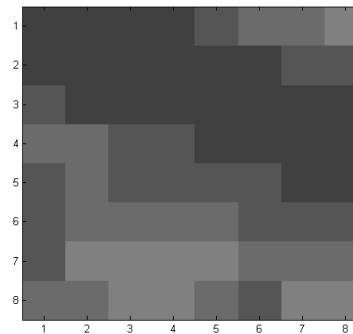
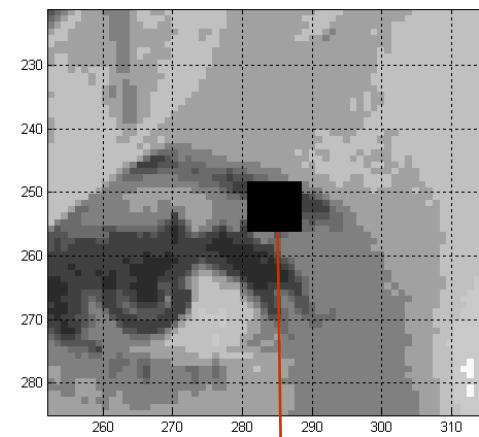
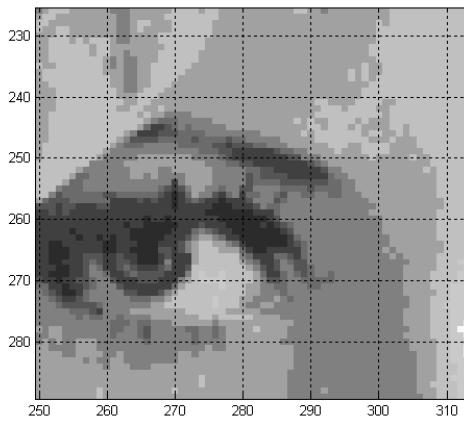
## DCT-based JPEG Standard

*Consider the following 8-bit image with size  $512 \times 512$  pixels.*



# Image Compression

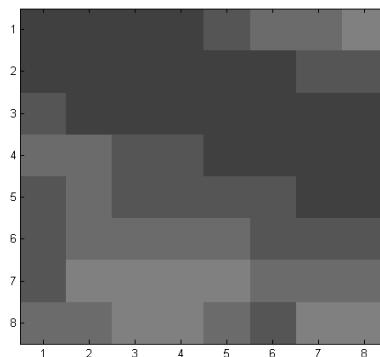
## DCT-based JPEG Standard



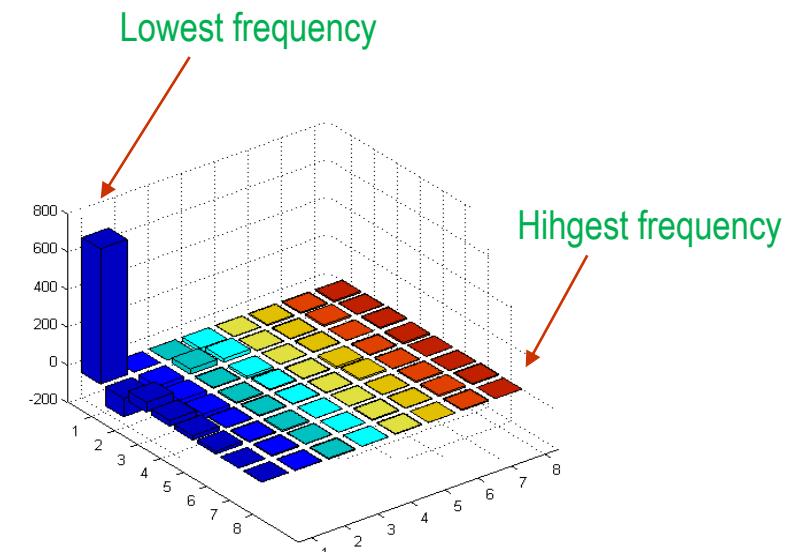
*8x8 Block to be processed*

# Image Compression

## DCT-based JPEG Standard



*DCT*



69	71	74	76	89	106	111	122
59	70	61	61	68	76	88	94
82	70	77	67	65	63	57	70
97	99	87	83	72	72	68	63
91	105	90	95	85	84	79	75
92	110	101	106	100	94	87	93
89	113	115	124	113	105	100	110
104	110	124	125	107	95	117	116

717.6	0.2	0.4	-19.8	-2.1	-6.2	-5.7	-7.6
-99.0	-35.8	27.4	19.4	-2.6	-3.8	9.0	2.7
51.8	-60.8	3.9	-11.8	1.9	4.1	1.0	6.4
30.0	-25.1	-6.7	6.2	-4.4	-10.7	-4.2	-8.0
22.6	2.7	4.9	3.4	-3.6	8.7	-2.7	0.9
15.6	4.9	-7.0	1.1	2.3	-2.2	6.6	-1.7
0.0	5.9	2.3	0.5	5.8	3.1	8.0	4.8
-0.7	-2.3	-5.2	-1.0	3.6	-0.5	5.1	-0.1

**Step 1:Discrete Cosine Transform (DCT)**

# Image Compression

## DCT-based JPEG Standard

### Step 2: Quantization Procedure

717.6	0.2	0.4	-19.8	-2.1	-6.2	-5.7	-7.6
-99.0	-35.8	27.4	19.4	-2.6	-3.8	9.0	2.7
51.8	-60.8	3.9	-11.8	1.9	4.1	1.0	6.4
30.0	-25.1	-6.7	6.2	-4.4	-10.7	-4.2	-8.0
22.6	2.7	4.9	3.4	-3.6	8.7	-2.7	0.9
15.6	4.9	-7.0	1.1	2.3	-2.2	6.6	-1.7
0.0	5.9	2.3	0.5	5.8	3.1	8.0	4.8
-0.7	-2.3	-5.2	-1.0	3.6	-0.5	5.1	-0.1

÷

Quantization Matrix

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

45	0	0	-1	0	0	0	0
-8	-3	2	1	0	0	0	0
4	-5	0	0	0	0	0	0
2	-1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

==

Quantized  
8x8 block

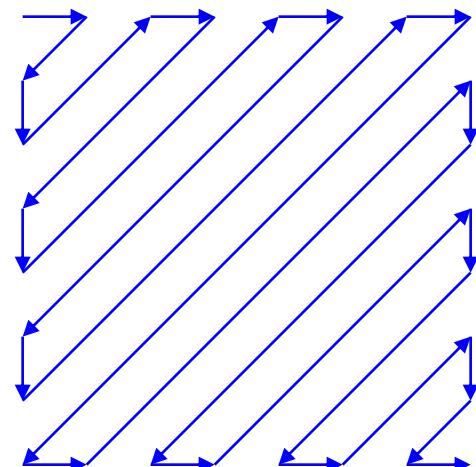
# Image Compression

## DCT-based JPEG Standard

### Step3: Coefficient-to-Symbol Mapping

45	0	0	-1	0	0	0	0
-8	-3	2	1	0	0	0	0
4	-5	0	0	0	0	0	0
2	-1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Input



Zigzag scan procedure

```
Result = 45,0,-8,4,-3,0,-1,2,-5,2,1,-1,0,1,0,0,0,0,0,1,EOB
```

**EOB symbol denotes the end-of-block condition**

---

# Image Compression

## DCT-based JPEG Standard

*Given the symbols below,*

```
Result = 45,0,-8,4,-3,0,-1,2,-5,2,1,-1,0,1,0,0,0,0,0,1,EOB
```

---

### Step 4: Entropy Coding

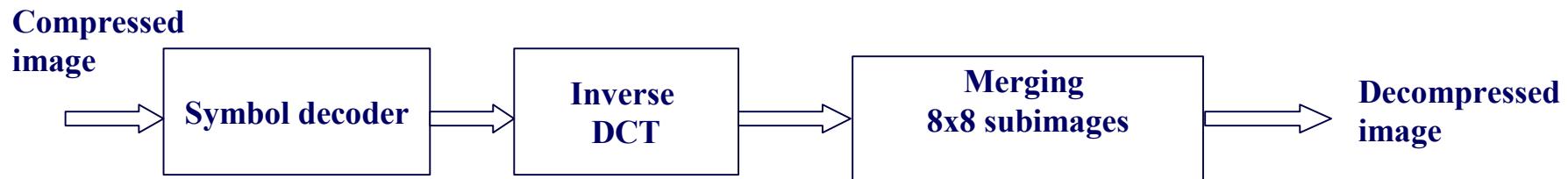
- Symbols are encoded using mostly **Huffman coding**.
- Huffman coding is a method of variable length coding in which shorter codewords are assigned to the more frequently occurring symbols.

```
1110101101 010 1011000 .....1010
```

# **Image Compression**

## DCT-based JPEG Standard

- Once the encoded file is received the decoding is the inverse process given below.



**Decoder**

# Image Compression

## DCT-based JPEG Standard

•Compressed image examples for changing compression ratios.



Size: 263224 Bytes



Size: 5728 Bytes  
 $C_R = 46$



Size: 11956 Bytes  
 $C_R = 22$



Size: 15159 Bytes  
 $C_R = 17$



Size: 18032 Bytes  
 $C_R = 15$



Size: 20922 Bytes  
 $C_R = 13$

# Morphological Image Processing

- The word **morphology** refers to the scientific branch that deals the forms and structures of animals/plants.
- Morphology in image processing is a tool for extracting image components that are useful in the representation and description of region shape, such as boundaries and skeletons.
- Furthermore, the morphological operations can be used for filtering, thinning and pruning.
- The language of the Morphology comes from the set theory, where image objects can be represented by sets. For example an image object containing black pixels can be considered a set of black pixels in 2D space of  $Z^2$ .

---

# Morphological Image Processing

## Set Theory Fundamentals

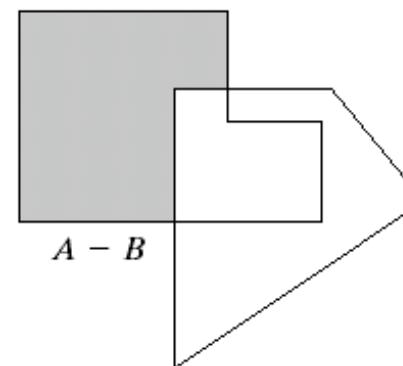
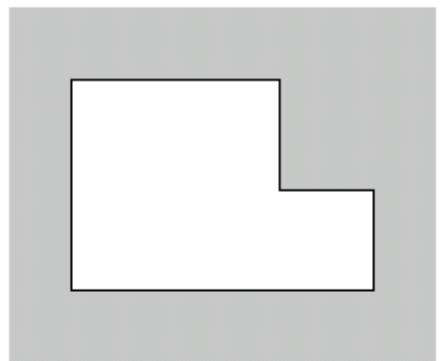
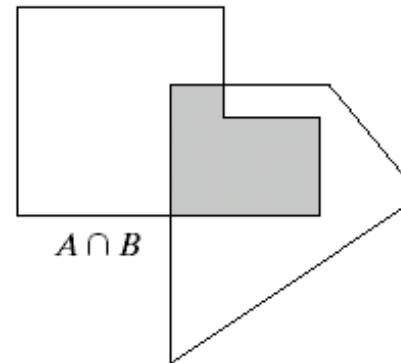
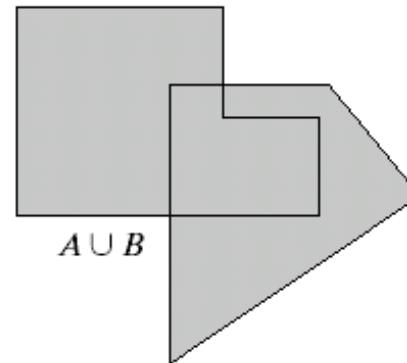
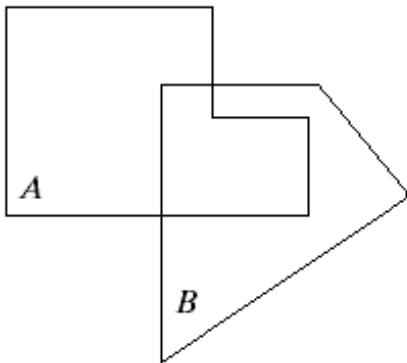
- Given that  $A$  is a set in  $Z^2$  and  $a=(a_1,a_2)$ , then
  - $a$  is an **element** in  $A$ :  $a \in A$
  - $a$  is **not an element** in  $A$ :  $a \notin A$
- given sets  $A$  and  $B$ ,  $A$  is said to be the **subset** of  $B$ :  $A \subseteq B$
- The **union** of  $A$  and  $B$  is denoted by:  $C = A \cup B$
- The **intersection** of  $A$  and  $B$  is denoted by:  $D = A \cap B$
- Two sets are **disjoint/mutually exclusive** if  $A \cap B = \emptyset$
- The **complement** of set  $A$  is the set of elements not contained in  $A$ ,  $A^c = \{\omega | \omega \notin A\}$
- The **difference** of two sets:  $A - B = \{\omega | \omega \in A, \omega \notin B\} = A \cap B^c$

---

# Morphological Image Processing

## Set Theory Fundamentals

Given 2 sets A and B



# Morphological Image Processing

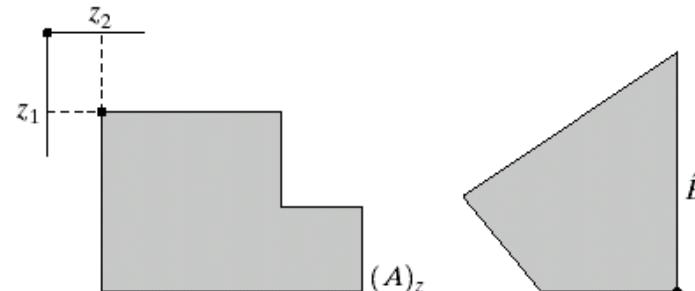
## Set Theory Fundamentals

- *The reflection of set B is defined by:*

$$\hat{B} = \{\omega | \omega = -b, \text{ for } b \in B\}$$

- *The translation of set A by point  $z=(z_1, z_2)$  is defined by:*

$$(A)_z = \{\omega | \omega = a + z, \text{ for } a \in A\}$$



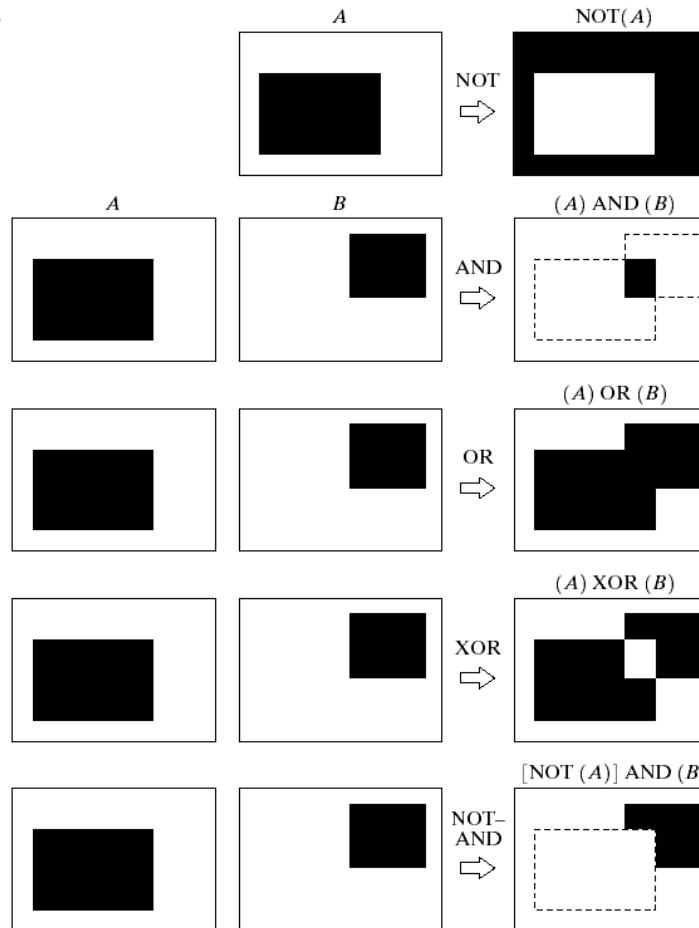
Translation of A by  $z$ .

Reflection of B

# Morphological Image Processing

## Logic operation involving Binary Images

- Given 1-bit binary images,  $A$  and  $B$ , the basic logical operations are illustrated:



- Note that the black indicates binary 1 and white indicates binary 0 here.

---

# Morphological Image Processing

## Dilation and Erosion

- *Dilation and erosion are the two fundamental operations used in morphological image processing. Almost all morphological algorithms depend on these two operations:*
- **Dilation:** Given  $A$  and  $B$  sets in  $Z^2$ , the dilation of  $A$  by  $B$ , is defined by:

$$A \oplus B = \left\{ z \mid (\hat{B})_z \cap A \neq \emptyset \right\}$$

- *The dilation of  $A$  and  $B$  is a set of all displacements,  $z$ , such that  $\hat{B}$  and  $A$  overlap by at least one element. The definition can also be written as:*

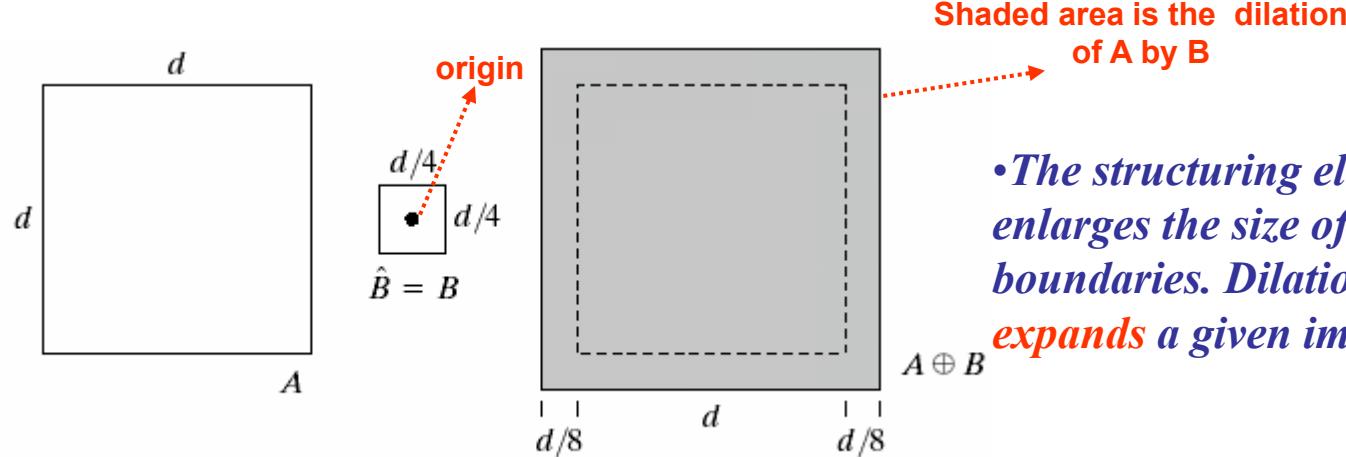
$$A \oplus B = \left\{ z \mid [(\hat{B})_z \cap A] \subseteq A \right\}$$

- *Set  $B$  is referred to as the **structuring element** and used in dilation as well as in other morphological operations. Dilation expands/dilutes a given image.*

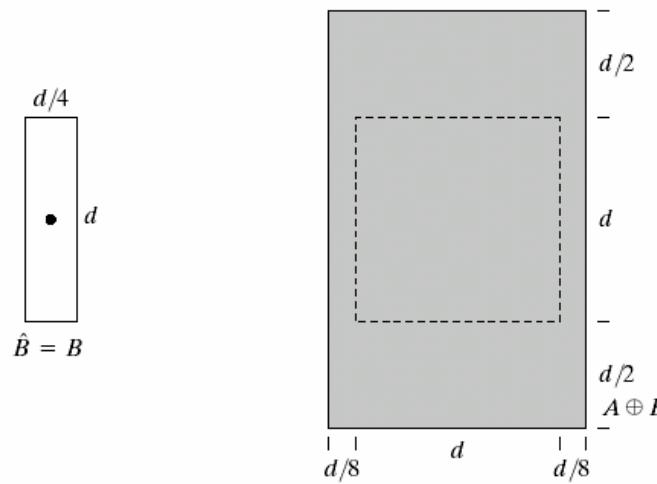
# Morphological Image Processing

## Dilation and Erosion

- **Dilation:** Given the structuring element  $B$  and set  $A$ .



• The structuring element  $B$  enlarges the size of  $A$  at its boundaries. Dilation simply expands a given image.



• The structuring element  $B$  enlarges the size of  $A$  at its boundaries, in relation to the distance from the origin of the structuring element.

# Morphological Image Processing

## Dilation and Erosion

- **Dilation:** Given the following distorted text image where the maximum length of the broken characters are 2 pixels. The image can be enhanced by bridging the gaps by using the structuring element given below:

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

$A$



0	1	0
1	1	1
0	1	0

$B$

3x3 structuring element

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

$A \oplus B$



- Note that the broken characters are joined.

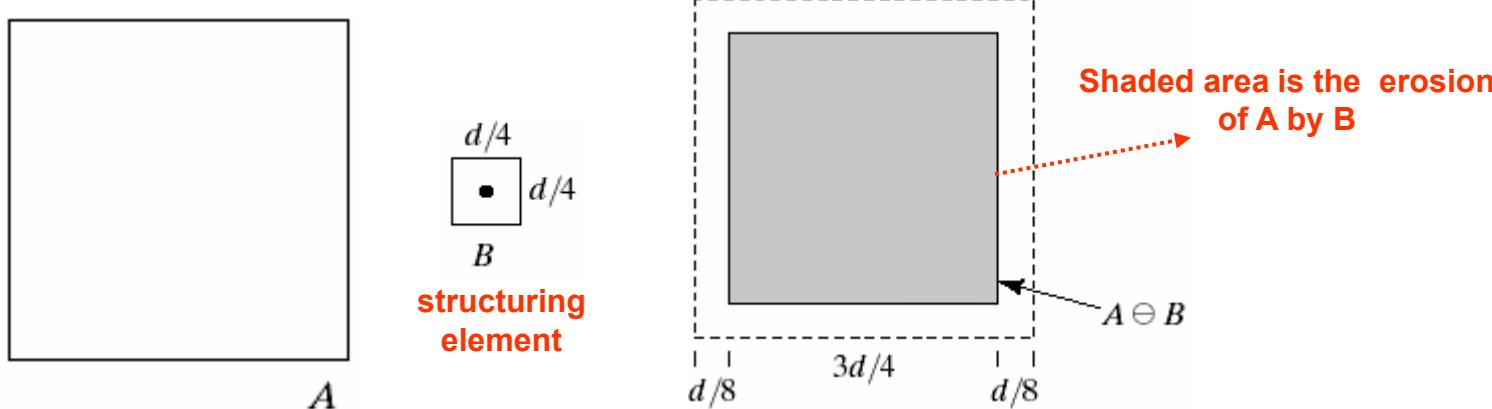
# Morphological Image Processing

## Dilation and Erosion

- **Erosion:** Given  $A$  and  $B$  sets in  $\mathbb{Z}^2$ , the **erosion** of  $A$  by structuring element  $B$ , is defined by:

$$A \ominus B = \{z \mid (B)_z \subseteq A\}$$

- The erosion of  $A$  by structuring element  $B$  is the set of all points  $z$ , such that  $B$ , translated by  $z$ , is contained in  $A$ .

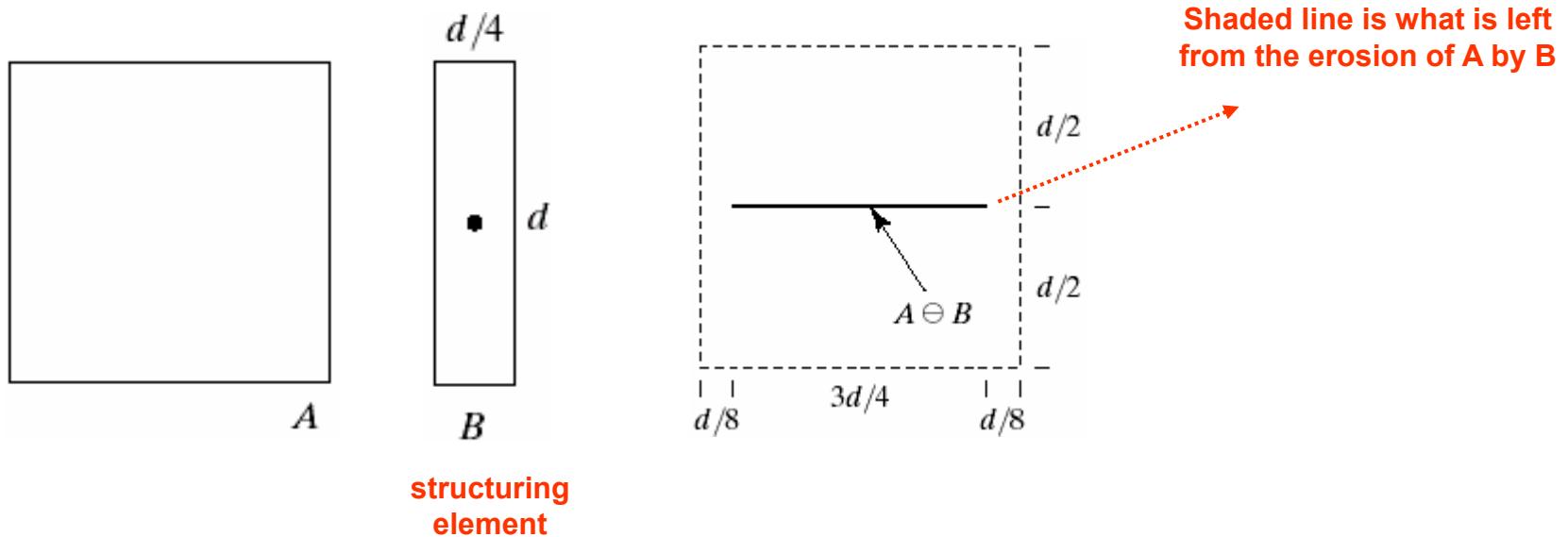


- Note that in erosion the structuring element  $B$  erodes the input image  $A$  at its boundaries. Erosion **shrinks** a given image.

# Morphological Image Processing

## Dilation and Erosion

- *Erosion: Given the structuring element  $B$  and set  $A$ .*

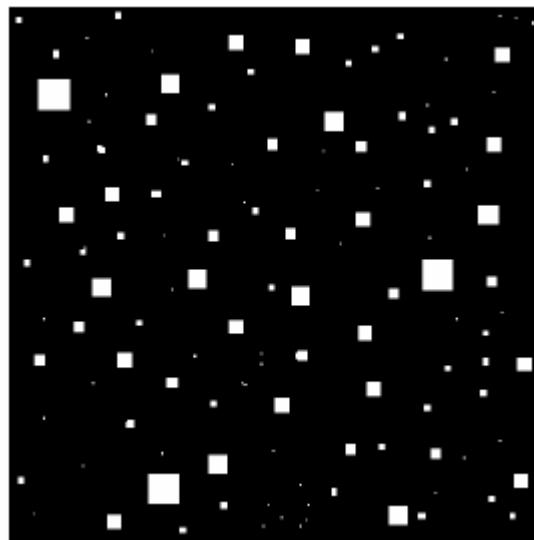


---

# Morphological Image Processing

## Dilation and Erosion

- **Erosion:** Given the following binary image with squares on size 1,3,5,7,9 and 15. You can get rid of all the squares less than size of 15 by erosion followed by dilation of a structuring element of 13x13.



$A$

$\square$   
 $B$

13x13 structuring  
element



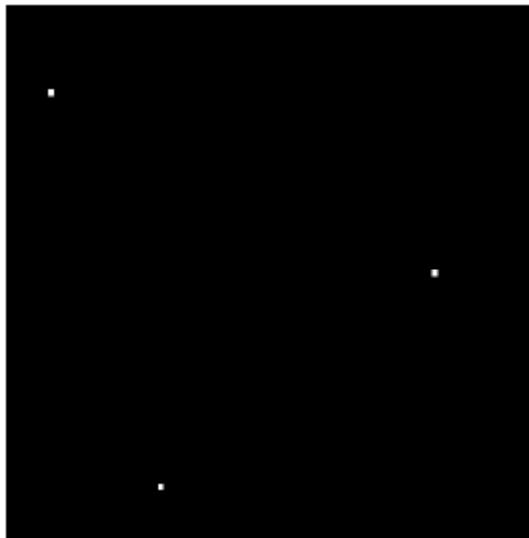
$A \ominus B$

Erosion of A by B

# Morphological Image Processing

## Dilation and Erosion

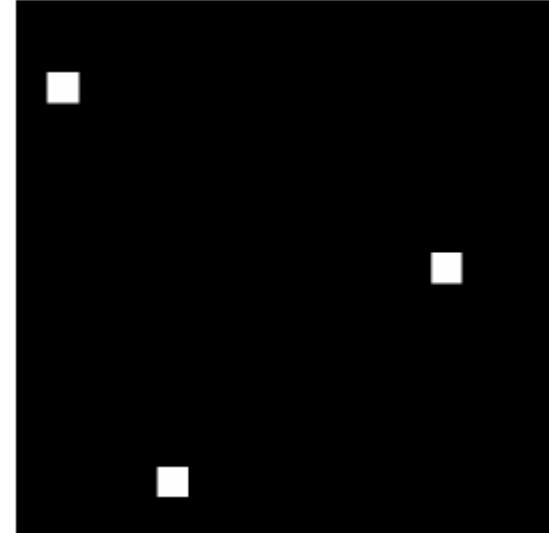
- Dilation: Cont. from the previous slide. Note that erosion followed by dilation helps to perform filtering.



$$A \ominus B$$



13x13 structuring  
element



$$(A \ominus B) \oplus B$$

dilation by B

---

# Morphological Image Processing

## Opening and closing

- **Opening:** The process of **erosion** followed by **dilation** is called **opening**. It has the effect of **eliminating small and thin objects**, **breaking the objects at thin points** and **smoothing the boundaries/contours of the objects**.
- Given set  $A$  and the structuring element  $B$ . Opening of  $A$  by structuring element  $B$  is defined by:

$$A \circ B = (A \ominus B) \oplus B$$

- **Closing:** The process of **dilation** followed by **erosion** is called **closing**. It has the effect of **filling small and thin holes**, **connecting nearby objects** and **smoothing the boundaries/contours of the objects**.
- Given set  $A$  and the structuring element  $B$ . Closing of  $A$  by structuring element  $B$  is defined by:

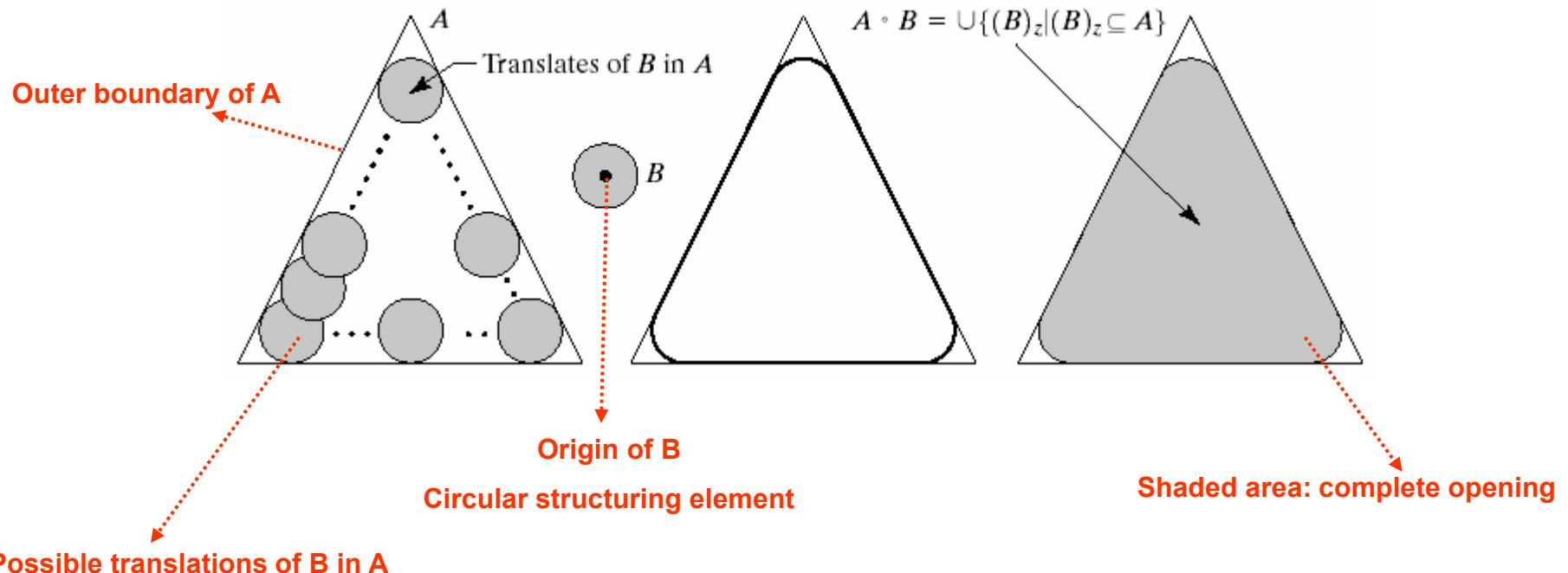
$$A \bullet B = (A \oplus B) \ominus B$$

# Morphological Image Processing

## Opening and closing

- *Opening:* The opening of  $A$  by the structuring element  $B$  is obtained by taking the union of all translates of  $B$  that fit into  $A$ .
- The opening operation can also be expressed by the following formula:

$$A \circ B = \bigcup \{B_z \mid (B_z) \subseteq A\}$$

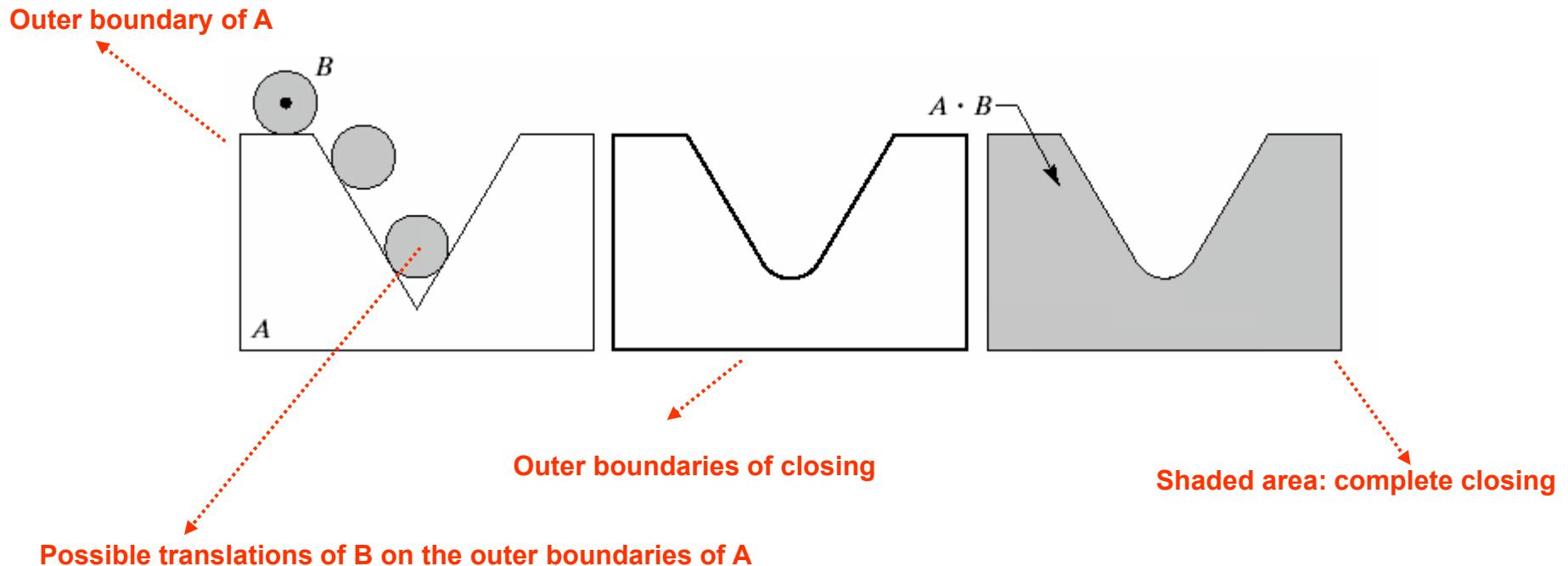


# Morphological Image Processing

## Opening and closing

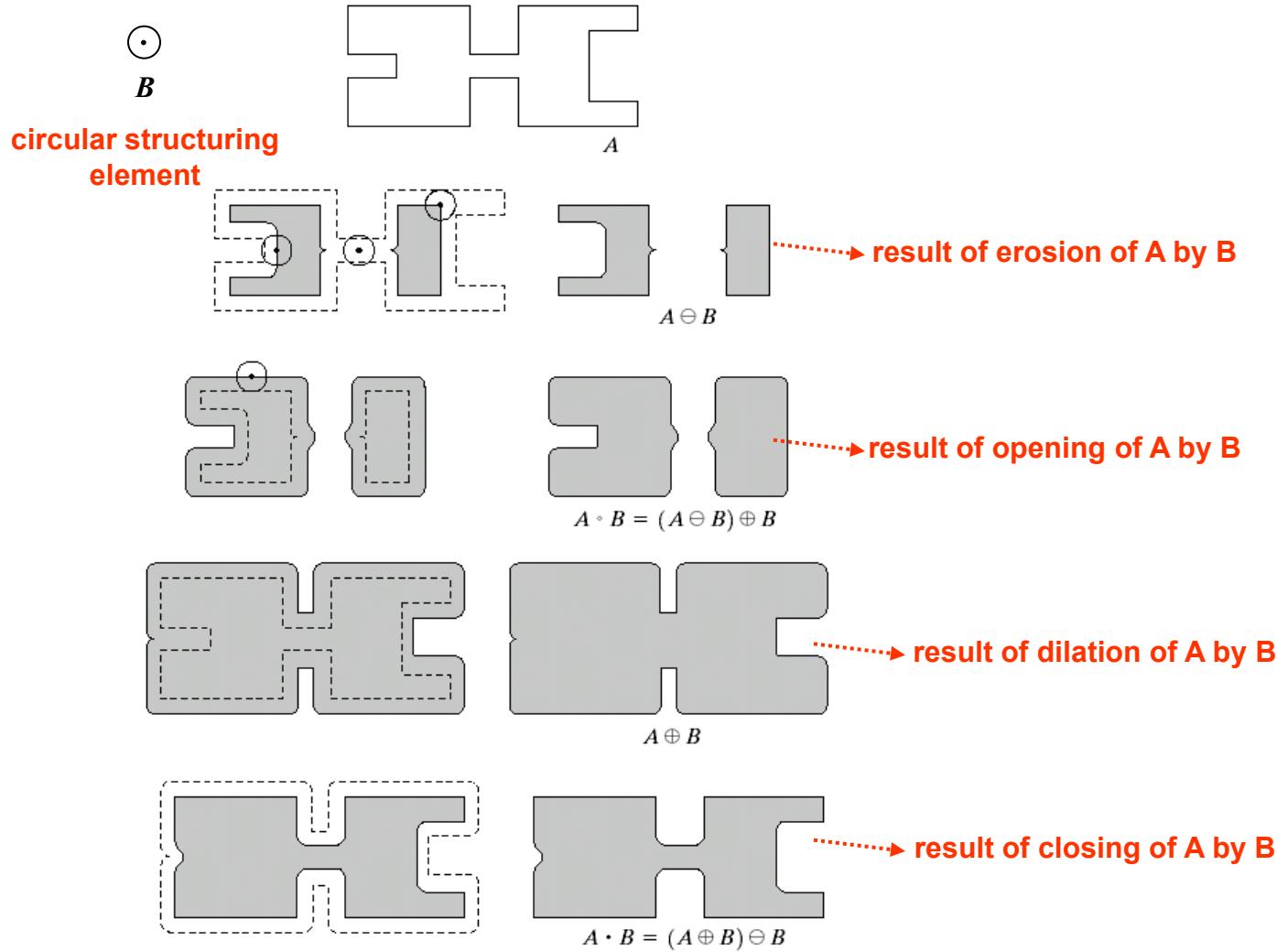
- *Closing: The closing has a similar geometric interpretation except that we roll B on the outside of the boundary.*
- *The opening operation can also be expressed by the following formula:*

$$A \bullet B = \bigcup \{(B_z) | (B_z) \cap A \neq \emptyset\}$$



# Morphological Image Processing

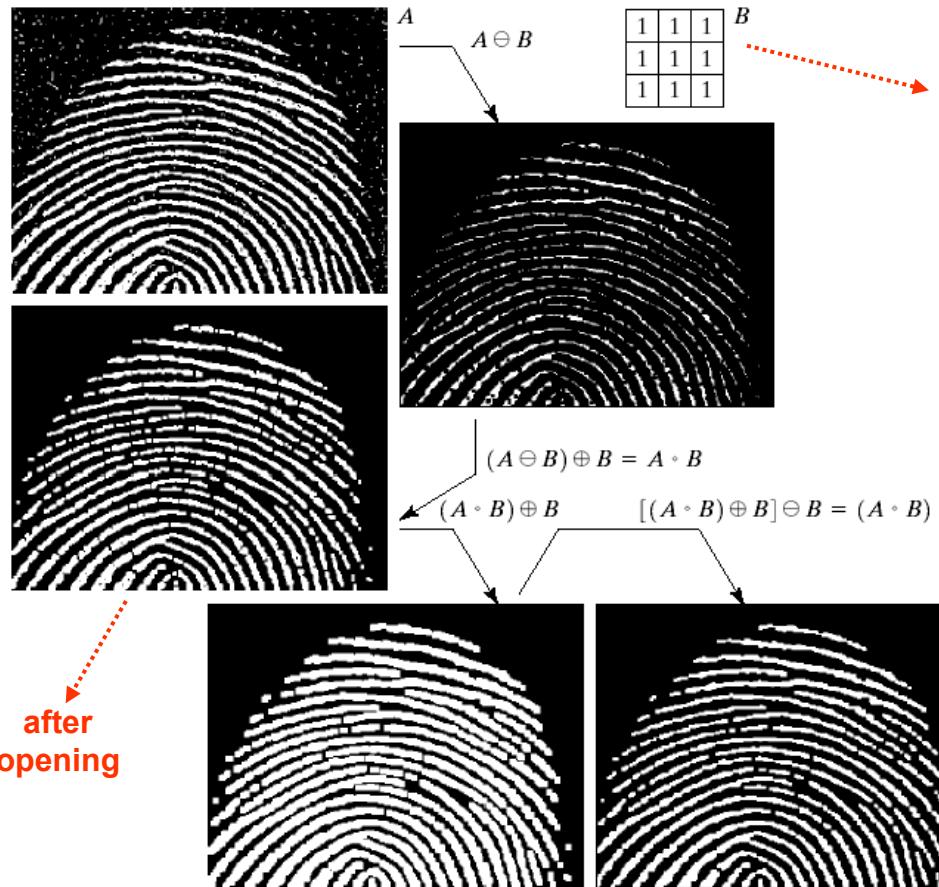
## Opening and closing



# Morphological Image Processing

## Opening and closing

- *Noise Filtering:* The morphological operations can be used to remove the noise as in the following example:



# Morphological Image Processing

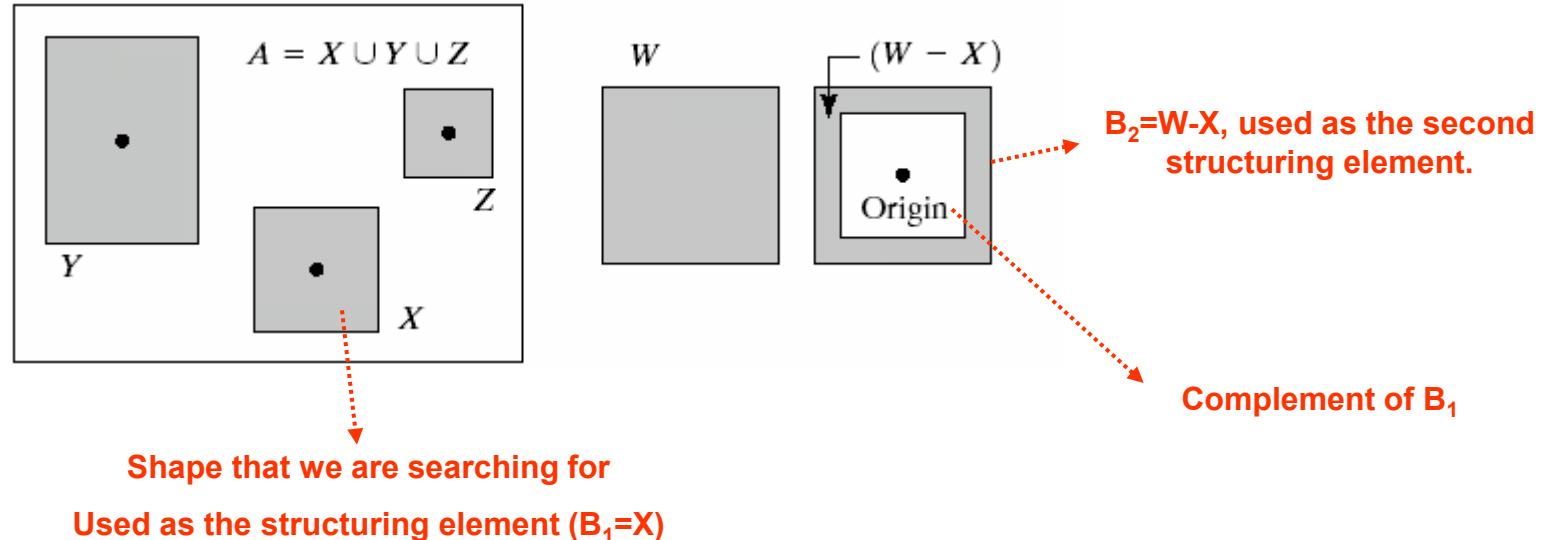
## Hit-or-Miss Transform (Template Matching)

- Hit-or-miss transform can be used for shape detection/ Template matching.

- Given the shape as the structuring element  $B_1$ , the Hit-or-miss transform is defined by:

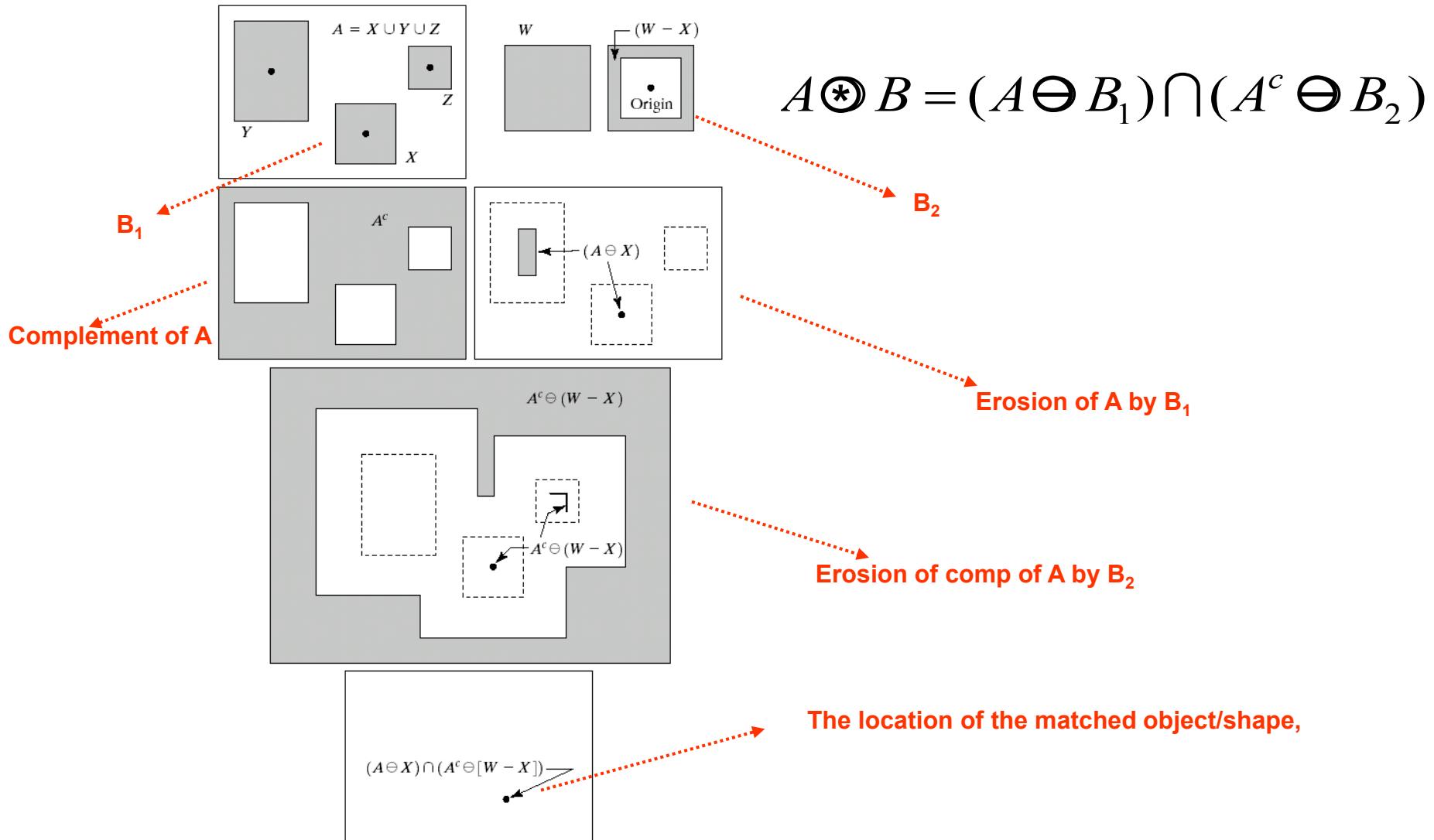
$$A \circledast B = (A \ominus B_1) \cap (A^c \ominus B_2)$$

- Where  $B_2 = W - X$  and  $B_1 = X$ .  $W$  is the window enclosing  $B_1$ . Windowing is used to isolate the structuring element/object.



# Morphological Image Processing

## Hit-or-Miss Transform

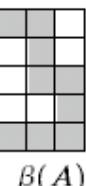
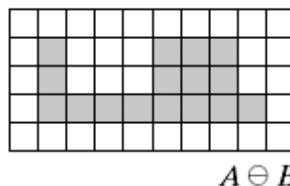
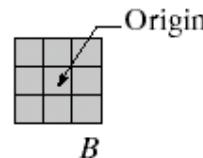
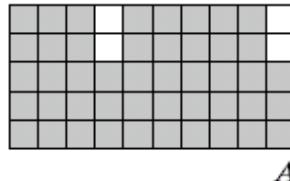


# Morphological Image Processing

## Basic Morphological Algorithms

- **Boundary Extraction:** The boundaries/edges of a region/shape can be extracted by first applying erosion on A by B and subtracting the eroded A from A.

$$\beta(A) = A - (A \ominus B)$$



**Ex 1:** 3x3 Square structuring element is used for boundary extraction.



**Ex 2:** The same structuring element in Ex1 is used.

Note that thicker boundaries can be obtained by increasing the size of structuring element.

---

# Morphological Image Processing

## Basic Morphological Algorithms

- **Region Filling:** Region filling can be performed by using the following definition. Given a symmetric structuring element  $B$ , one of the non-boundary pixels ( $X_k$ ) is consecutively diluted and its intersection with the complement of  $A$  is taken as follows:

$$X_k = (X_{k-1} \oplus B) \cap A^c \quad k = 1, 2, 3, \dots$$

terminates when  $X_k = X_{k-1}$

$$X_0 = 1 \text{ (inner pixel)}$$

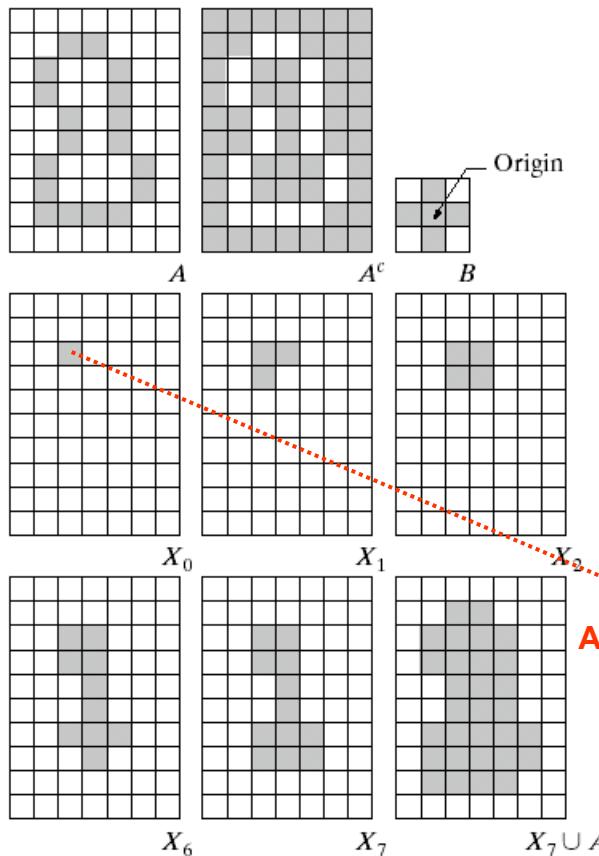
- Following consecutive dilations and their intersection with the complement of  $A$ , finally resulting set is the filled inner boundary region and its union with  $A$  gives the filled region  $F(A)$ .

$$F(A) = X_k \cup A$$

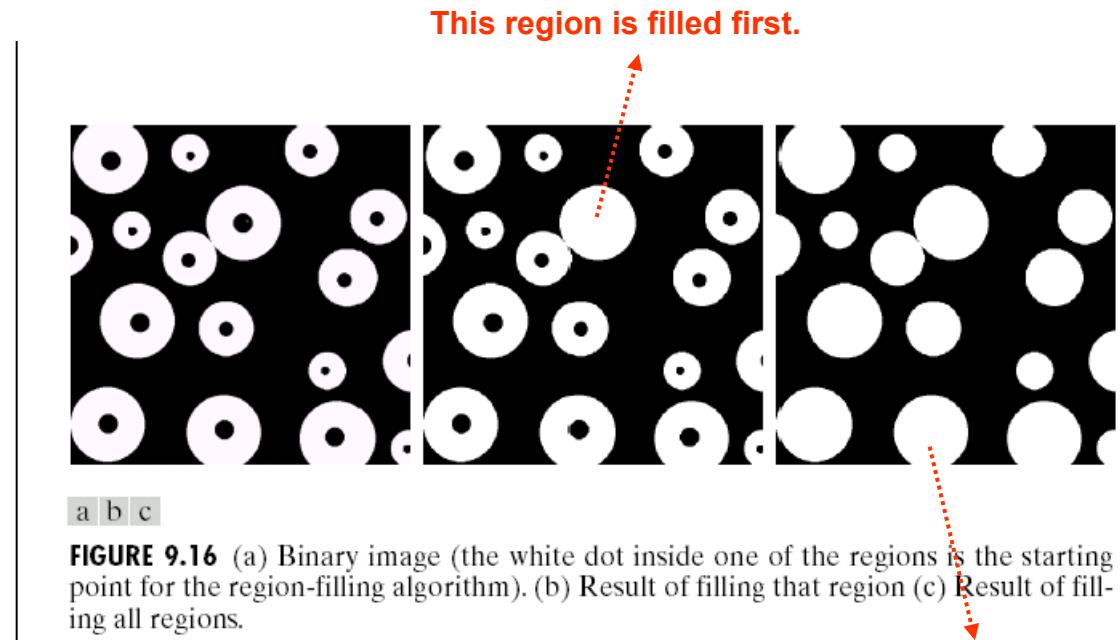
# Morphological Image Processing

## Basic Morphological Algorithms

- Region Filling:



Ex 1:  $X_0=1$  (Assume that the shaded boundary points are 1 and the white pixels are 0)



# Morphological Image Processing

## Basic Morphological Algorithms

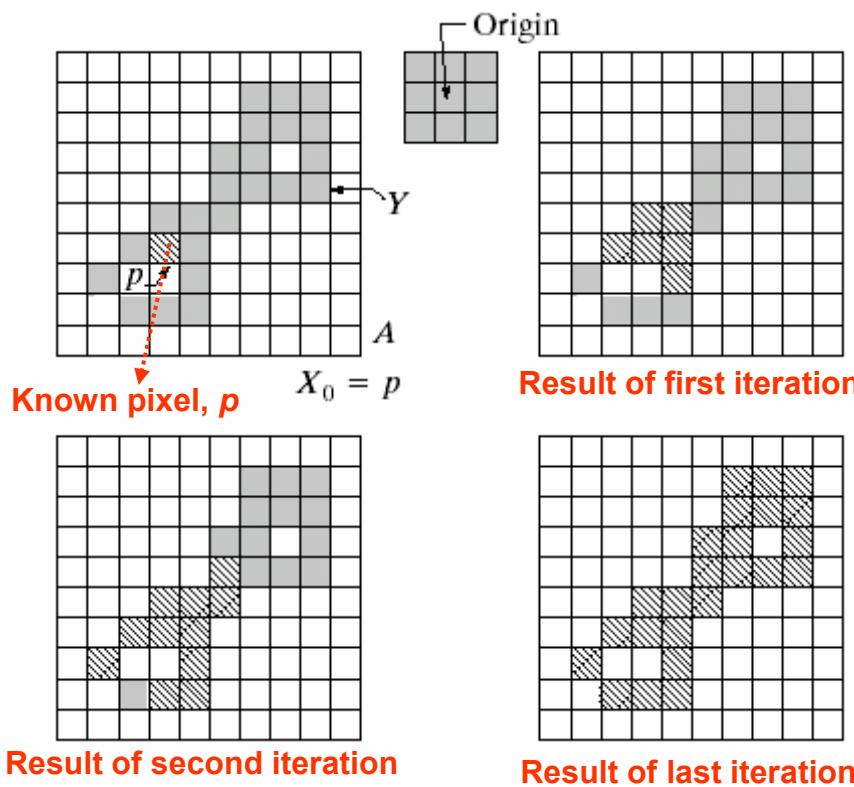
- **Connected Component Extraction:** The following iterative expression can be used to determine all the pixels in component  $Y$  which is in  $A$ .

$$X_k = (X_{k-1} \oplus B) \cap A$$

$$k = 1, 2, 3, \dots$$

terminates when  $X_k = X_{k-1}$

$$X_0 = 1 \text{ (boundary pixel)}$$

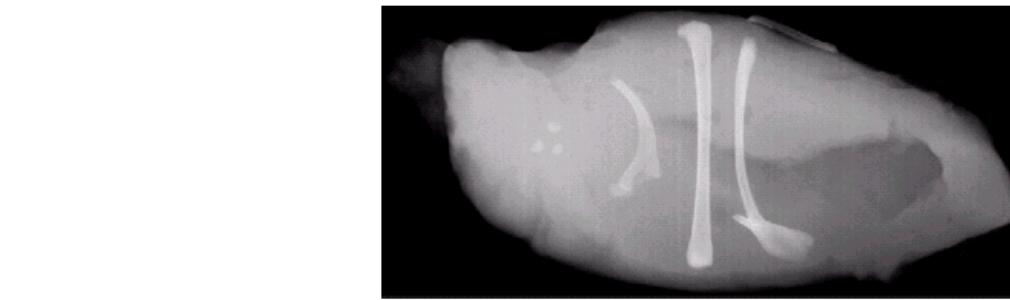


- $X_0=1$  corresponds to one of the pixels on the component  $Y$ . Note that one of the pixel locations on the component must be known.
- Consecutive dilations and their intersection with  $A$ , yields all elements of component  $Y$ .

# Morphological Image Processing

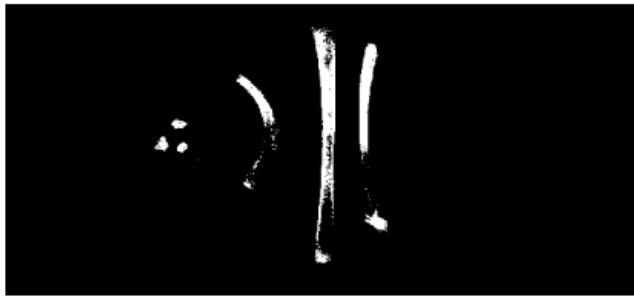
## Basic Morphological Algorithms

- Connected Component Extraction:



Input image  
(Chicken fillet)

Thresholded image



After erosion by 5x5 square structuring element of 1's



Connected component	No. of pixels in connected comp
01	11
02	9
03	9
04	39
05	133
06	1
07	1
08	743
09	7
10	11
11	11
12	9
13	9
14	674
15	85

15 connected components with different number of pixels

# Morphological Image Processing

## Basic Morphological Algorithms

- **Thinning:** Thinning of  $A$  by the structuring element  $B$  is defined by:

$$A \otimes B = A - (A \odot B)$$

hit-or-miss transform/template matching

- Note that we are only interested in pattern matching of  $B$  in  $A$ , so no background operation is required of the hit-miss-transform.

$$\{B\} = \{B^1, B^2, B^3, \dots, B^n\}$$

- The structuring element  $B$  consists of a sequence of structuring elements, where  $B^i$  is the rotated version of  $B^{i-1}$ . Each structuring elements helps thinning in one direction. If there are 4 structuring elements thinning is performed from 4 directions separated by  $90^\circ$ . If 8 structuring elements are used the thinning is performed in 8 directions separated by  $45^\circ$ .

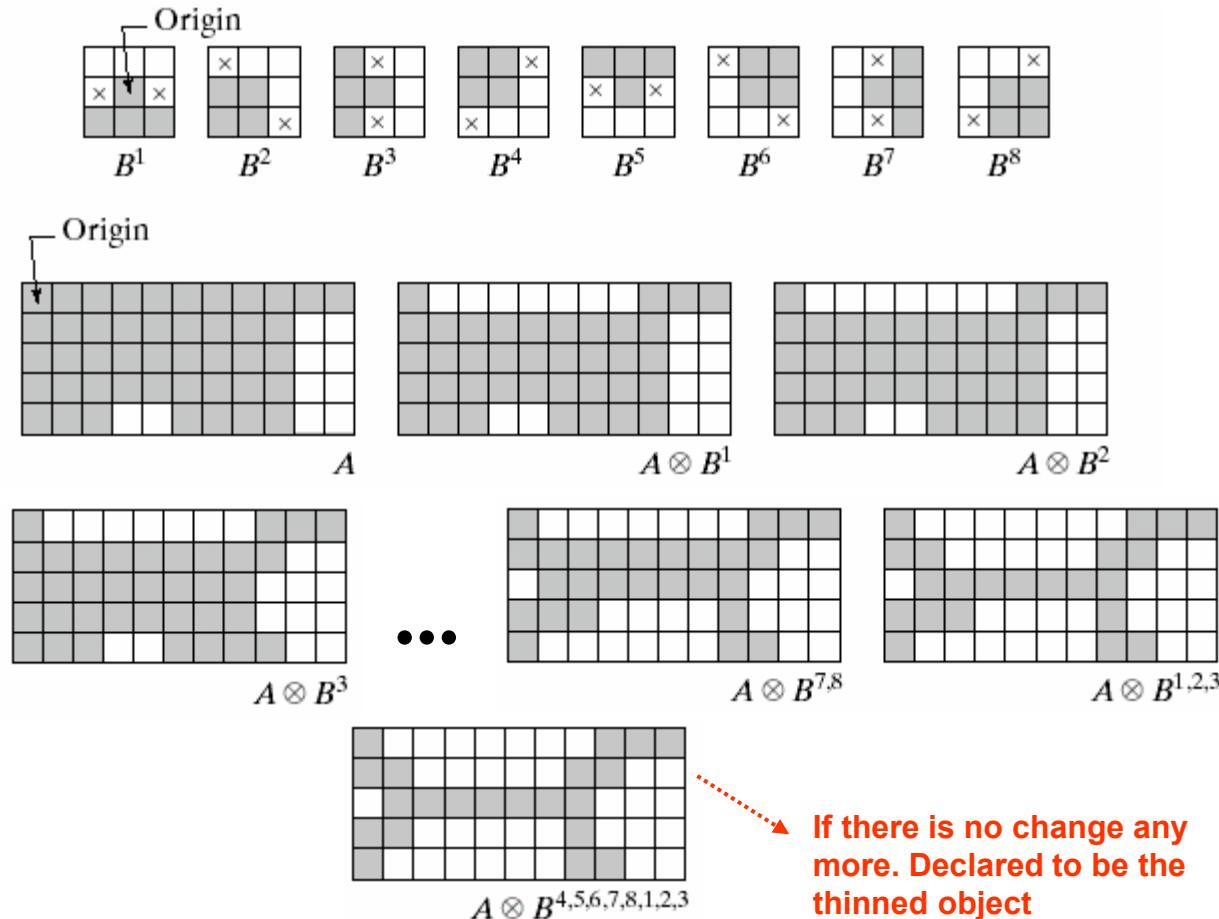
- The process is to thin  $A$  by one pass with  $B^1$ , then the result with one pass of  $B^2$ , and continue until  $A$  is thinned with one pass of  $B^n$ .

$$A \otimes \{B\} = (((((A \otimes B^1) \otimes B^2) \dots) \otimes B^n))$$

# Morphological Image Processing

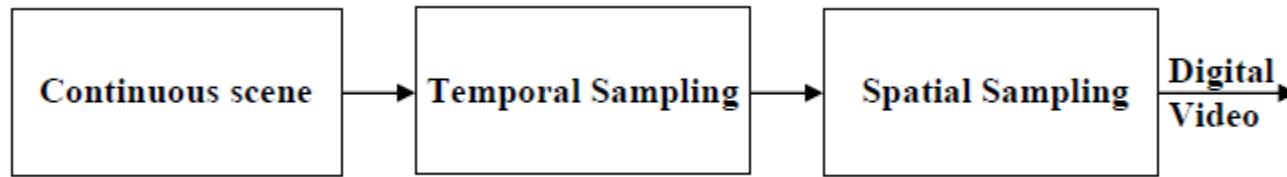
## Basic Morphological Algorithms

- **Thinning:** *The following set of structuring elements are used for thinning operation.*



## Digital Video Processing

- *Video signal is basically any sequence of time varying images.*
- *In a digital video, the picture information is digitized both spatially and temporally and the resultant pixel intensities are quantized.*



### *Spatial Sampling*

*In the digital representation of the image, the value of each pixel needs to be quantized using some finite precision. In practice, **8 bits** are used per luminance sample.*

### *Temporal sampling*

*A video consists of a sequence of images, displayed in rapid succession, to give an illusion of continuous motion. If the time gap between successive frames is too large, the viewer will observe jerky motion. In practice, most video formats use temporal sampling rates of **24 frames per second** and **above**.*

# Digital Video Processing

## Common Video Formats

- *Digital video frames that are displayed at a prescribed frame rate. For example, **frame rate of 30 frames/sec** is used in NTSC video.*
- *The **Common Intermediate Format (CIF)** has  $352 \times 288$  pixels, and the **Quarter CIF (QCIF)** format has  $176 \times 144$  pixels.*

Format	Luminance Pixel Resolution	Typical Applications
Sub-QCIF	128 X 96	Mobile Multimedia
QCIF	176 X 144	Video conferencing and Mobile Multimedia
CIF	352 X 288	Video conferencing
4CIF	704 X 576	SDTV and DVD-Video
16CIF	1408 X 1152	HDTV and DVD-Video

### ***Pixel Resolution (dots per lines)***

- *SD Resolution: **640 × 480 (720p)***
- *HD Resolution: **1280 × 720 (720p) / 1920 × 1080 (1080p)***

---

# Digital Video Processing

## Common Video Formats

- Each pixel is represented by three components: the **luminance component  $Y$** , and the two **chrominance components  $Cb$  and  $Cr$** .
- **$RGB$  to  $YCbCr$  Conversion**

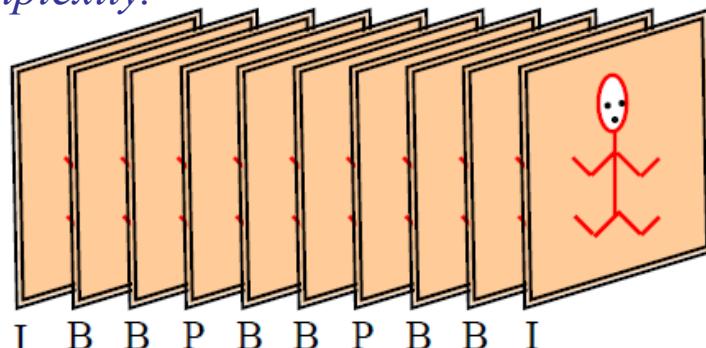
$$[Y \ Cb \ Cr] = [R \ G \ B] \begin{bmatrix} 0.299 & -0.168935 & 0.499813 \\ 0.587 & -0.331665 & -0.418531 \\ 0.114 & 0.50059 & -0.081282 \end{bmatrix}$$

- **Video quality** is commonly evaluated by using  $PSNR$  in luminance ( $Y$ ) Channel, which is referred to as the  **$Y$ - $PSNR$  (dB)**.

# Digital Video Processing

## Video Frame Types

- Three types of video frames are **I-frame**, **P-frame** and **B-frame**. ‘I’ stands for **Intra** coded frame, ‘P’ stands for **Predictive** frame and ‘B’ stands for **Bidirectional** predictive frame.
- **I’ frames** are encoded **without any motion compensation** and are used as a reference for future predicted ‘P’ and ‘B’ type frames. ‘I’ frames however require a relatively large number of bits for encoding.
- **‘P’ frames** are encoded using **motion compensated prediction** from a reference frame which can be either ‘I’ or ‘P’ frame. ‘P’ frames are more efficient in terms of number of bits required compared to ‘I’ frames, but still require more bits than ‘B’ frames. **‘B’ frames** require the lowest number of bits compared to both ‘I’ and ‘P’ frames but incur computational complexity.



---

# Digital Video Processing

## Video Coding

- **Intraframe coding**

Removing the **spatial redundancy** with a frame is generally termed as intraframe coding. The spatial redundancy within a frame is minimized by using transform. The commonly used **transform** is **DCT**.

- **Interframe coding**

The **temporal redundancy** between successive frames is removed by interframe coding. Interframe coding exploits the interdependencies of video frames.

Interframe coding relies on the fact that adjacent pictures in a video sequence have high temporal correlation.

- **Intra (I-coding)**

- **MB (Macro Block)** is encoded as is, without motion compensation.
- **DCT** followed by **Q (Quantization)**, zig-zag, run-length, **Huffman Coding**.

- **Inter (P- and B-coding)**

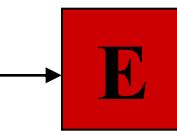
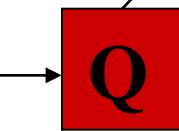
- **Block-matching - motion estimation**
- Predictive motion residue from best-match block is **DCT** encoded (similarly to intra-mode)
- Motion vector is differentially encoded .

# Digital Video Processing

## Video Coding

### Intra coding

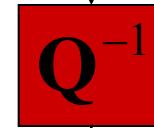
input MB



*Entropy coding*

*Quantization*

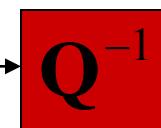
to bit-stream



to motion compensated frame

## Encoder

bit-stream

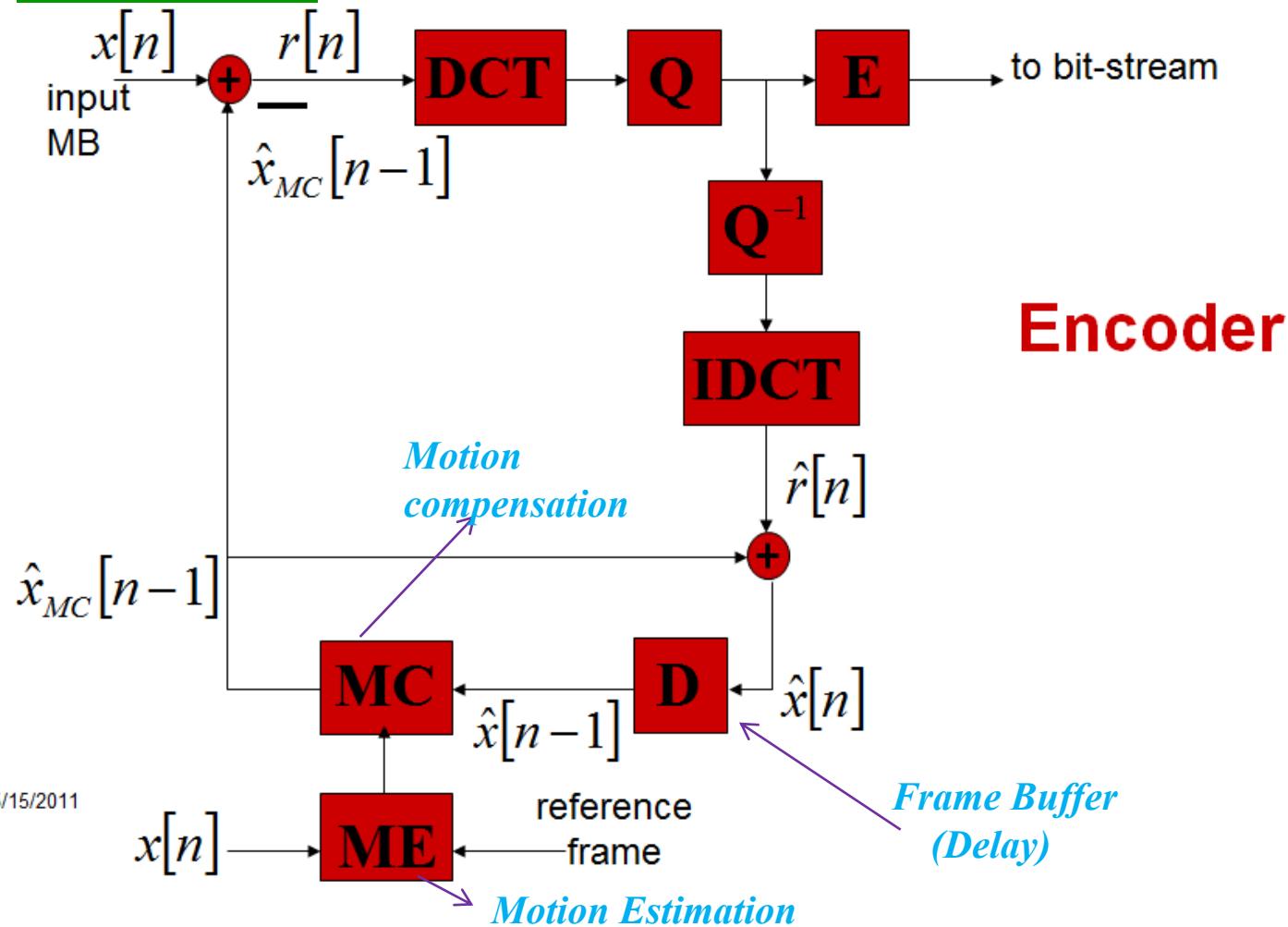


to display frame

## Decoder

# Digital Video Processing

## Video Coding *Inter coding*



# Digital Video Processing

## Video Coding

### Video Sequence and Picture



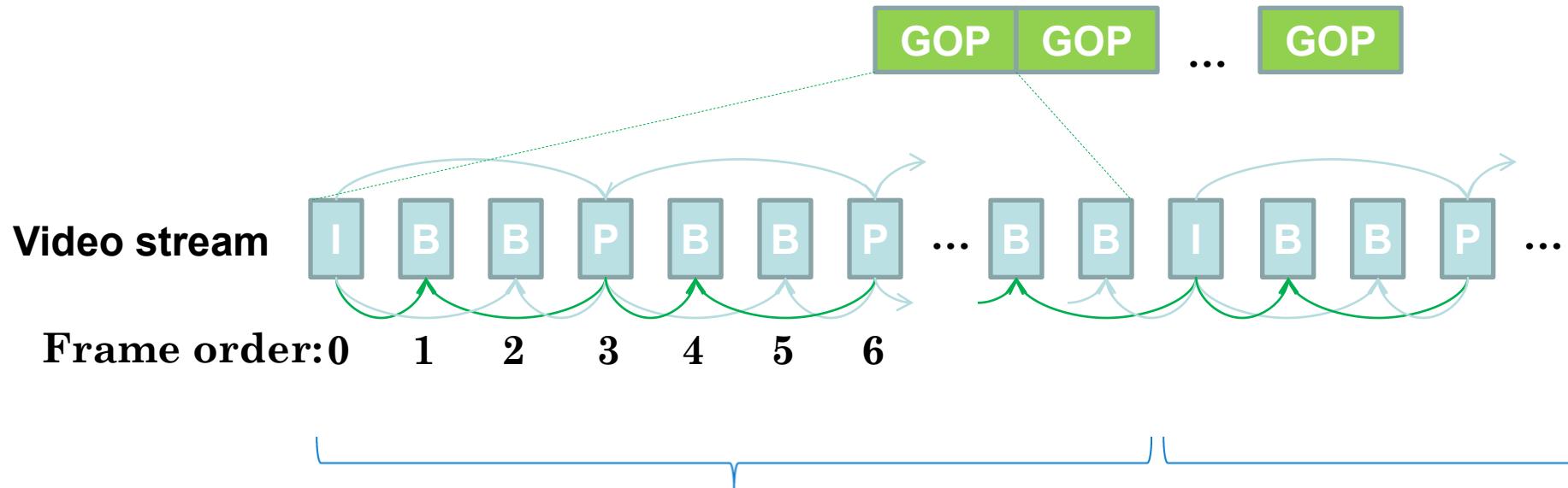
Intra 0      Inter 1      Inter 2      Inter 3      Inter 4      Inter 5

- ***Intra Picture (I-Picture)***
  - *Encoded without referencing others*
  - *All MBs are intra coded*
- ***Inter Picture (P-Picture, B-Picture)***
  - *Encoded by referencing other pictures*
  - *Some MBs are intra coded, and some are inter coded*

# Digital Video Processing

## Video Coding

### Group of Pictures (GOP)



# Digital Video Processing

## Video Coding

### Coding of I-Slice

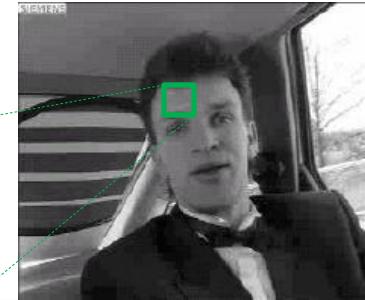
139	144	149	153	155	155	155	155
144	151	153	156	159	156	156	156
150	155	160	163	158	156	156	156
159	161	162	160	160	159	159	159
159	160	161	162	162	155	155	155
161	161	161	161	160	157	157	157
162	162	161	163	162	157	157	157
162	162	161	161	163	158	158	158

Original block

DCT

235.6	-1.0	-12.1	-5.2	2.1	-1.7	-2.7	1.3	16	11	10	16	24	40	51	61
-22.6	-17.5	-6.2	-3.2	-2.9	-0.1	0.4	-1.2	12	12	14	19	26	58	60	55
-10.9	-9.3	-1.6	1.5	0.2	-0.9	-0.6	-0.1	14	13	16	24	40	57	69	56
-7.1	-1.9	0.2	1.5	0.9	-0.1	0.0	0.3	14	17	22	29	51	87	80	62
-0.6	-0.8	1.5	1.6	-0.1	-0.7	0.6	1.3	18	22	37	56	68	109	103	77
1.8	-0.2	1.6	-0.3	-0.8	1.5	1.0	-1.0	24	35	55	64	81	104	113	92
-1.3	-0.4	-0.3	-1.5	-0.5	1.7	1.1	-0.8	49	64	78	87	103	121	120	101
-2.6	1.6	-3.8	-1.8	1.9	1.2	-0.6	-0.4	72	92	95	98	112	100	103	99

Transformed block



Quantization matrix

Bit-stream

15 0 -2 -1 -1 -1 0 ...

Entropy coding

Zig-zag scan

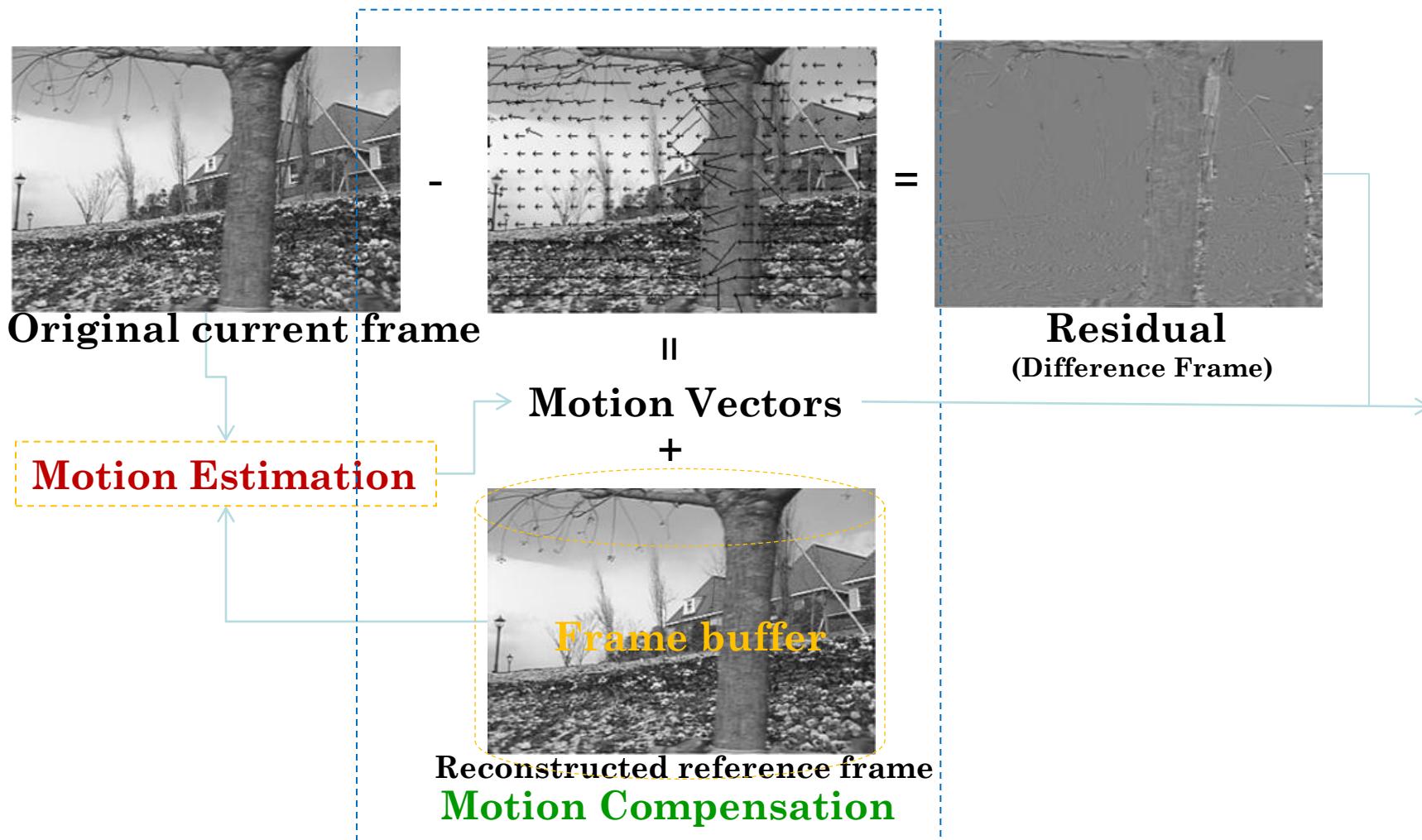
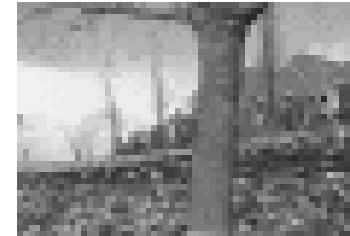
15	0	-1	0	0	0	0	0	0
-2	-1	0	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0



# Digital Video Processing

## Video Coding

### Coding of P-Slice

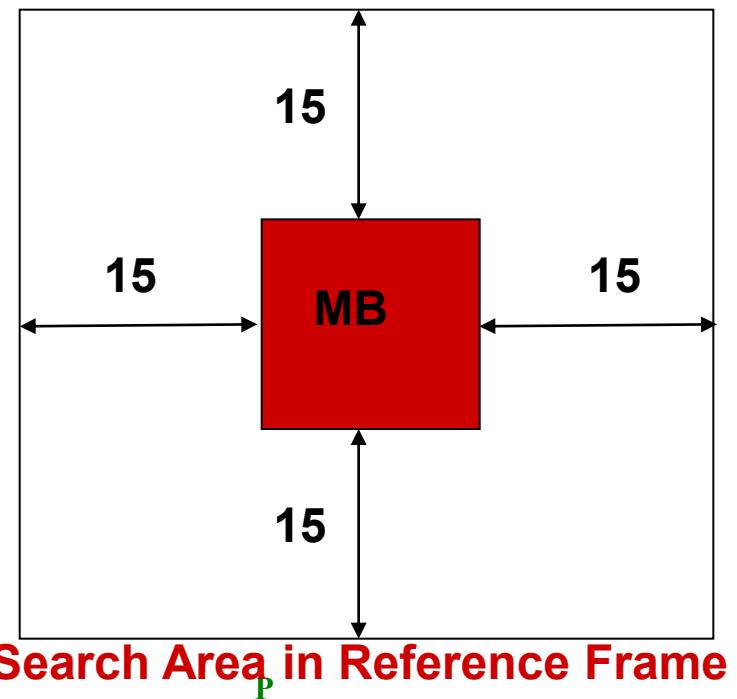
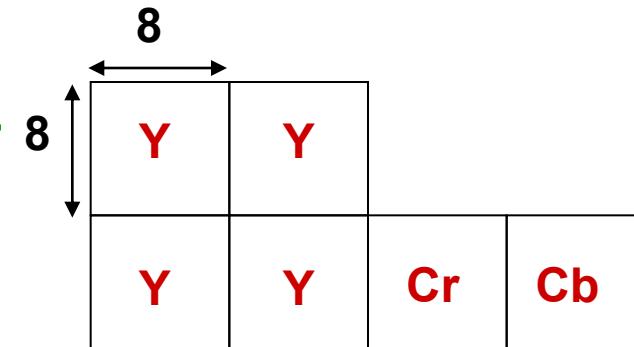


# Digital Video Processing

## Video Coding

### Motion Estimation in H.261

- *Macro-block*
  - *Luminance: 16x16, four 8x8 blocks*
  - *Chrominance: two 8x8 blocks*
  - *Motion estimation only performed for luminance component*
- *Motion vector range*
  - $[-15, 15]$



---

# Digital Video Processing

## Video Coding

### Coding of Motion Vectors

- *MV has range [-15, 15]*
- *Integer pixel ME search only*
- *Motion vectors are differentially & separably encoded*

$$MVD_x = MV_x[n] - MV_x[n-1]$$

$$MVD_y = MV_y[n] - MV_y[n-1]$$

- *11-bit VLC (Variable Length Coding) for MVD*
- *Example*

**MV = 2 2 3 5 3 1 -1...**

**MVD = 0 1 2 -2 -2 -2...**

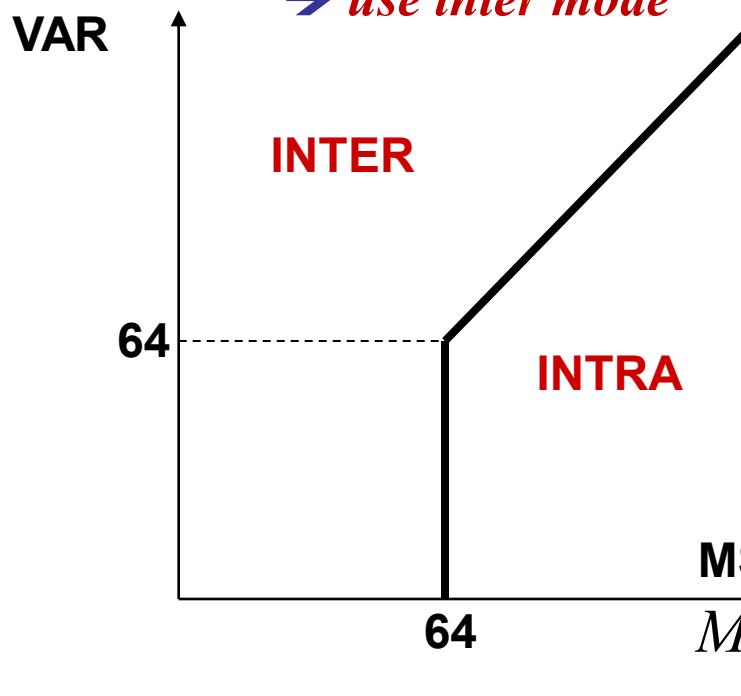
**Binary: 1 010 0010 0011 0011 0011...**

# Digital Video Processing

## Video Coding

### Inter/Intra Switching

- Based on energy of prediction error
  - **High energy**: scene change, occlusions, uncovered areas...  
→ **use intra mode**
  - **Low energy**: stationary background, translational motion ...  
→ **use inter mode**



$$VAR = \frac{1}{256} \sum_{MB} (c[x, y] - \bar{c})^2$$

$$MSE = \frac{1}{256} \sum_{MB} (c[x, y] - r[x + dx, y + dy])^2$$

---

# Digital Video Processing

## Video Coding

### H.263 Standard

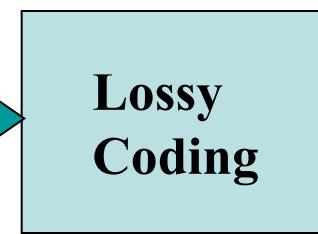
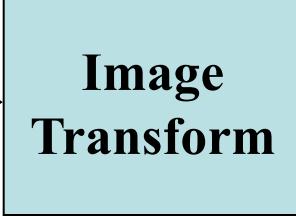
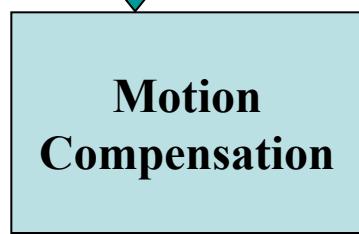
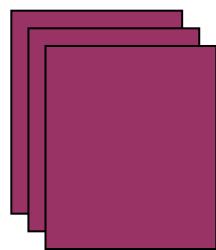
- *Standardization effort started Nov 1993*
- **Aim**
  - *low bit-rate video communications, less than 64 kbps*
  - *target PSTN and mobile network: 10-32 kbps*
- **Near-term**
  - *H.263 and H.263+: established late 1997*
- **Long-term**
  - *H.26L, H.264: still under investigation*
- **Main properties**
  - *H.261 with many MPEG features optimized for low bit rates*
  - *Performance: 3-4 dB improvements over H.261 at less than 64 kbps; 30% bit rate saving over MPEG-1.*

# Digital Video Processing

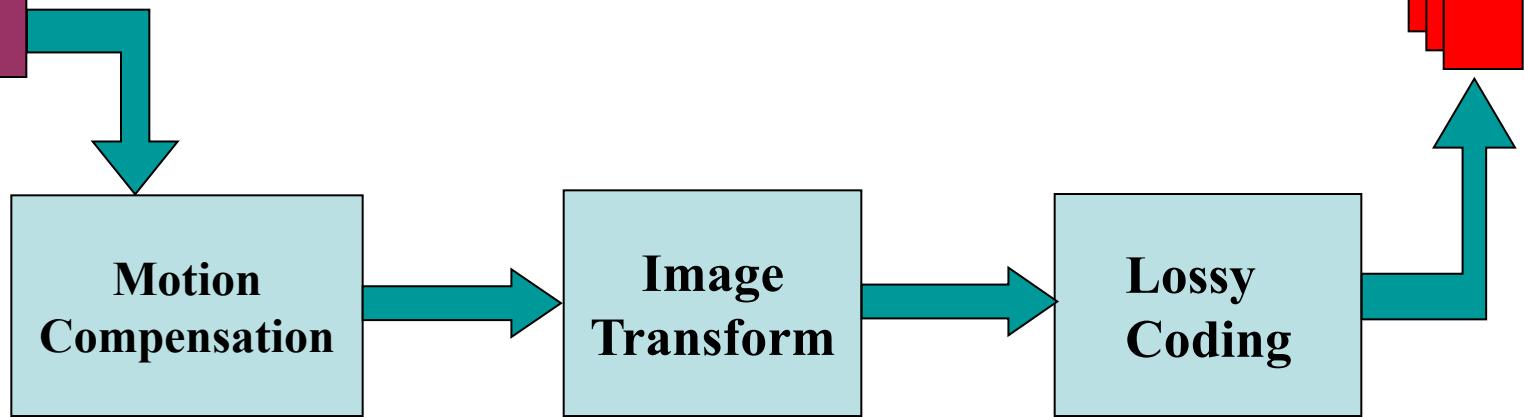
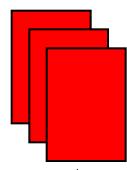
## Video Coding

### H.263 Standard Coder

original video



compressed video



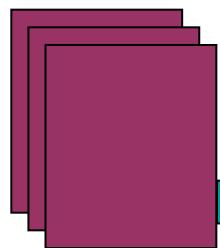
# Digital Video Processing

## Video Coding

### H.263 Standard Coder

#### H.263 Motion Compensation

original video



- Image is divided into **16x16 *macroblocks***,
- Each macroblock is matched against nearby blocks in previous frame (called *reference frame*),
  - “Nearby” = within 15-pixel horizontal/vertical range
  - Half-pixel accuracy (with bilinear pixel interpolation)
- Best match is used to *predict* the macroblock,
  - The relative displacement, or *motion vector*, is encoded and transmitted to decoder
- *Prediction error* for all blocks constitute the **residual**.

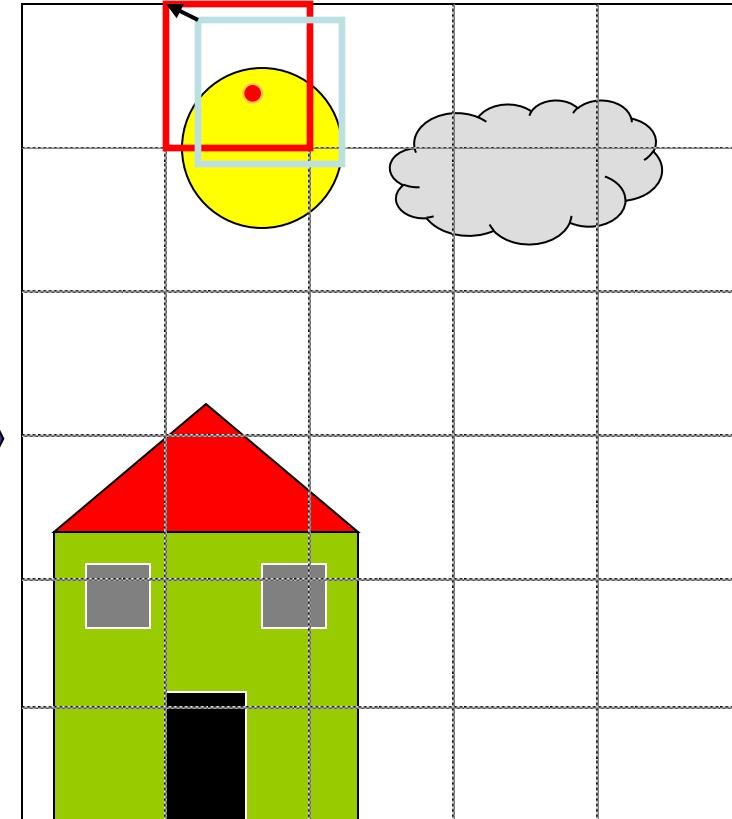
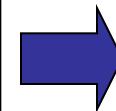
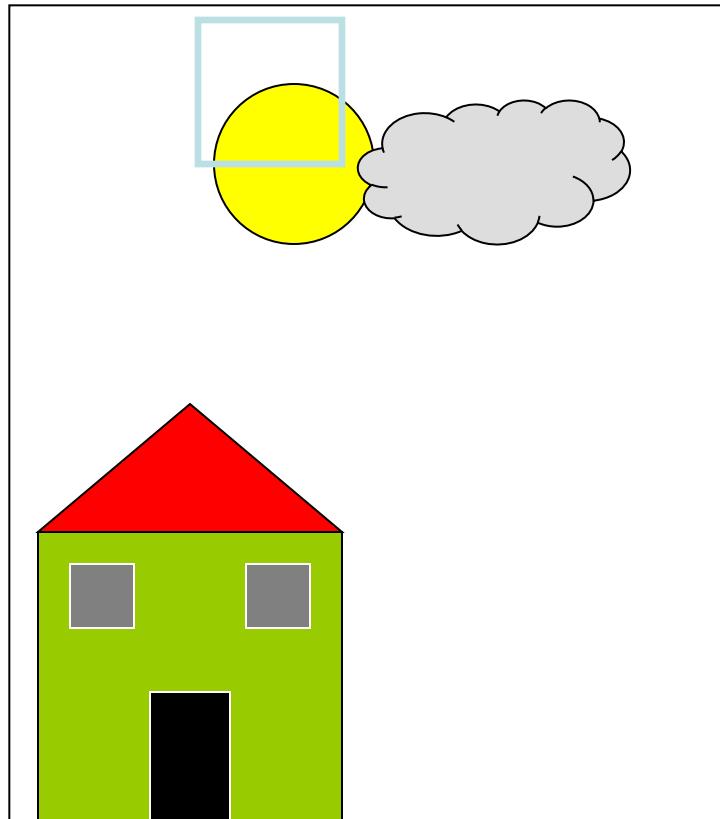
used video



# Digital Video Processing

## Video Coding

### Motion Compensation Example



**T=1 (reference)**

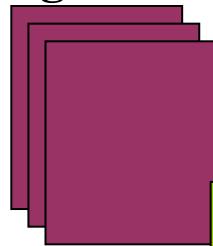
**T=2 (current)**

# Digital Video Processing

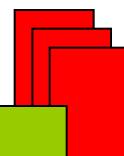
## Video Coding

### Motion Compensation Example

original video



compressed video



#### H.263 Image Transform

- Residual is divided into **8x8 blocks**,
- **8x8 2-d Discrete Cosine Transform (DCT)** is applied to each block independently
- DCT coefficients describe *spatial frequencies* in the block:
  - High frequencies correspond to small features and texture
  - Low frequencies correspond to larger features
  - Lowest frequency coefficient, called DC, corresponds to the average intensity of the block

---

# Digital Video Processing

## Video Coding

### 8x8 DCT Example

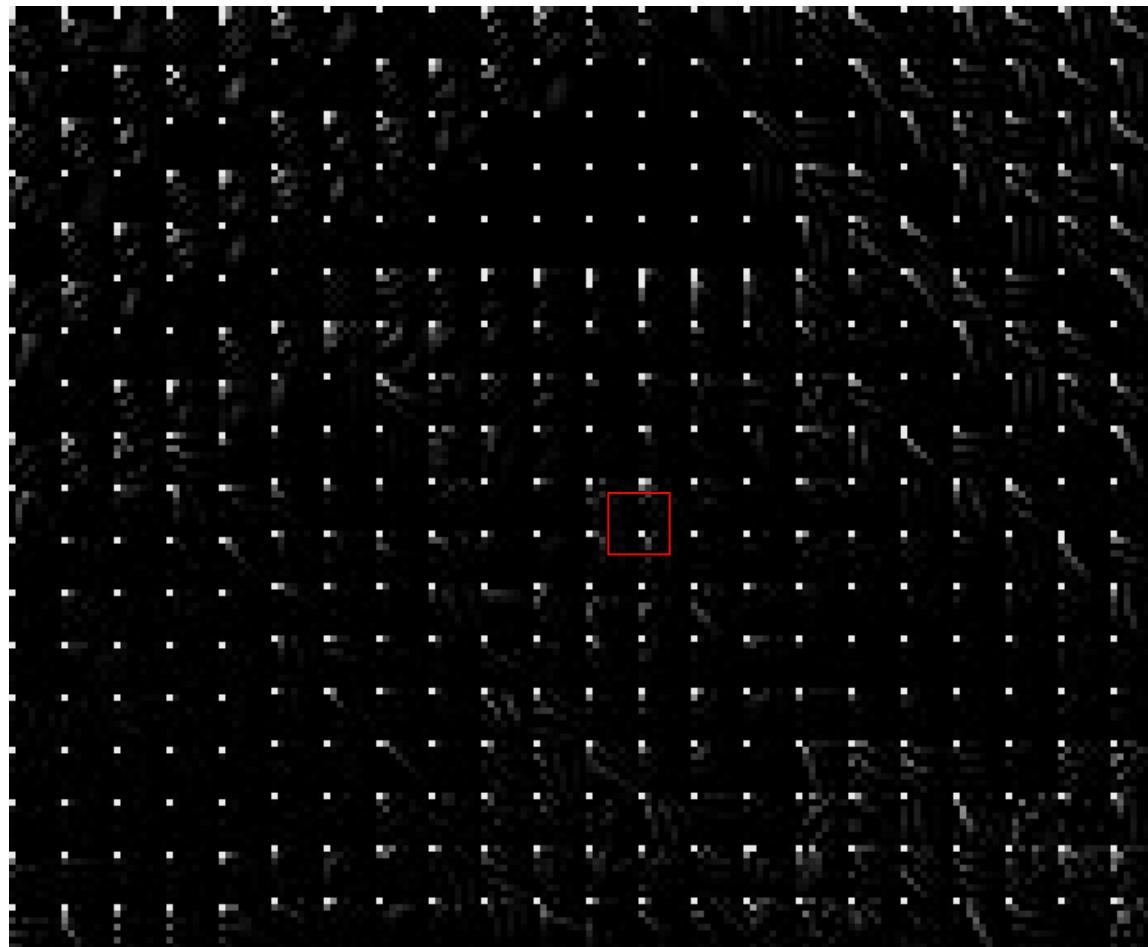


---

# Digital Video Processing

## Video Coding

### 8x8 DCT Example

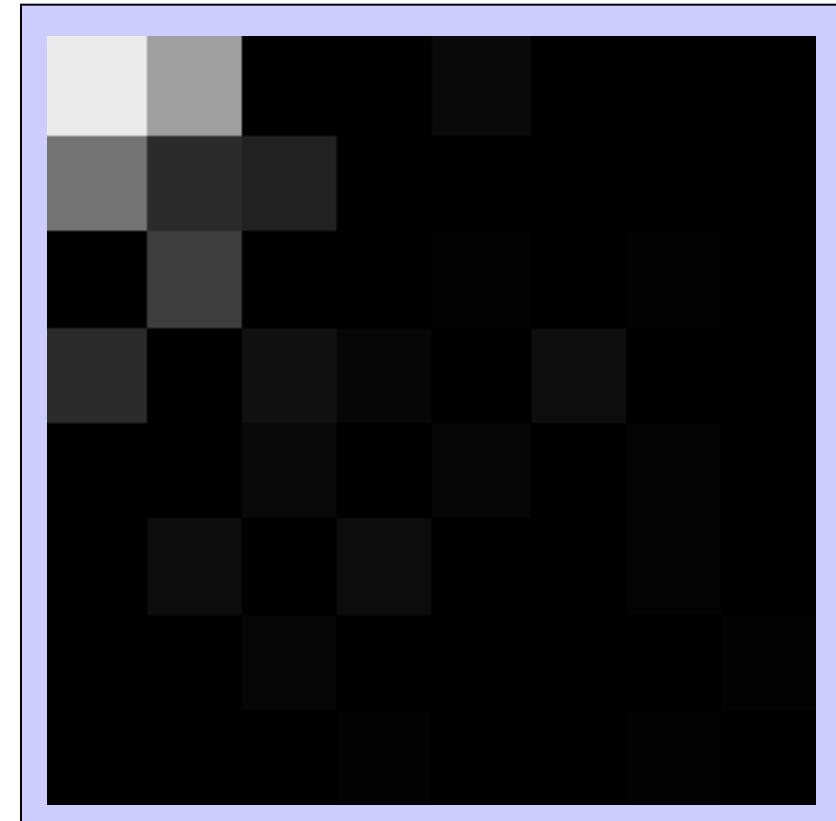
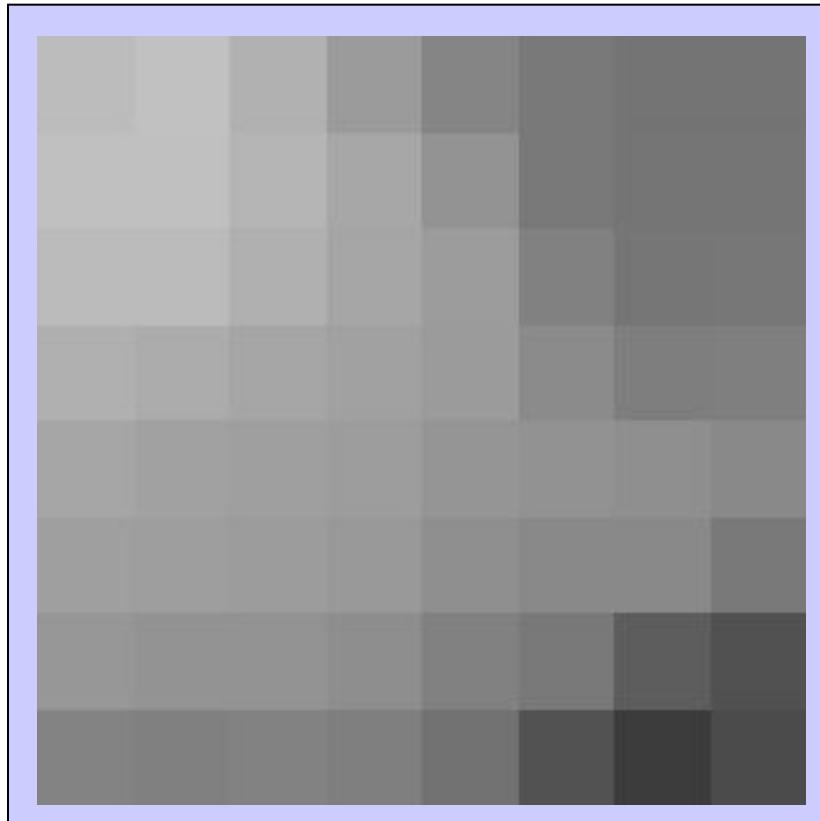


---

# Digital Video Processing

## Video Coding

### 8x8 DCT Example



# Digital Video Processing

## Video Coding

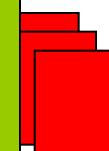
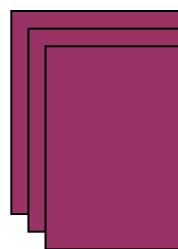
### H.263 Standard Coder

original video

#### H.263 Lossy Coding

- Transform coefficients are *quantized*:
  - Some less-significant bits are dropped
  - Only the remaining bits are encoded
- For inter-frames, all coefficients get the same number of bits, except for the DC which gets more.
- For intra-frames, lower-frequency coefficients get more bits
  - To preserve larger features better
- The actual number of bits used depends on a *quantization parameter* (QP), whose value depends on the bit-allocation policy
- Finally, bits are encoded using entropy (lossless) code
  - Traditionally Huffman-style code

compressed video

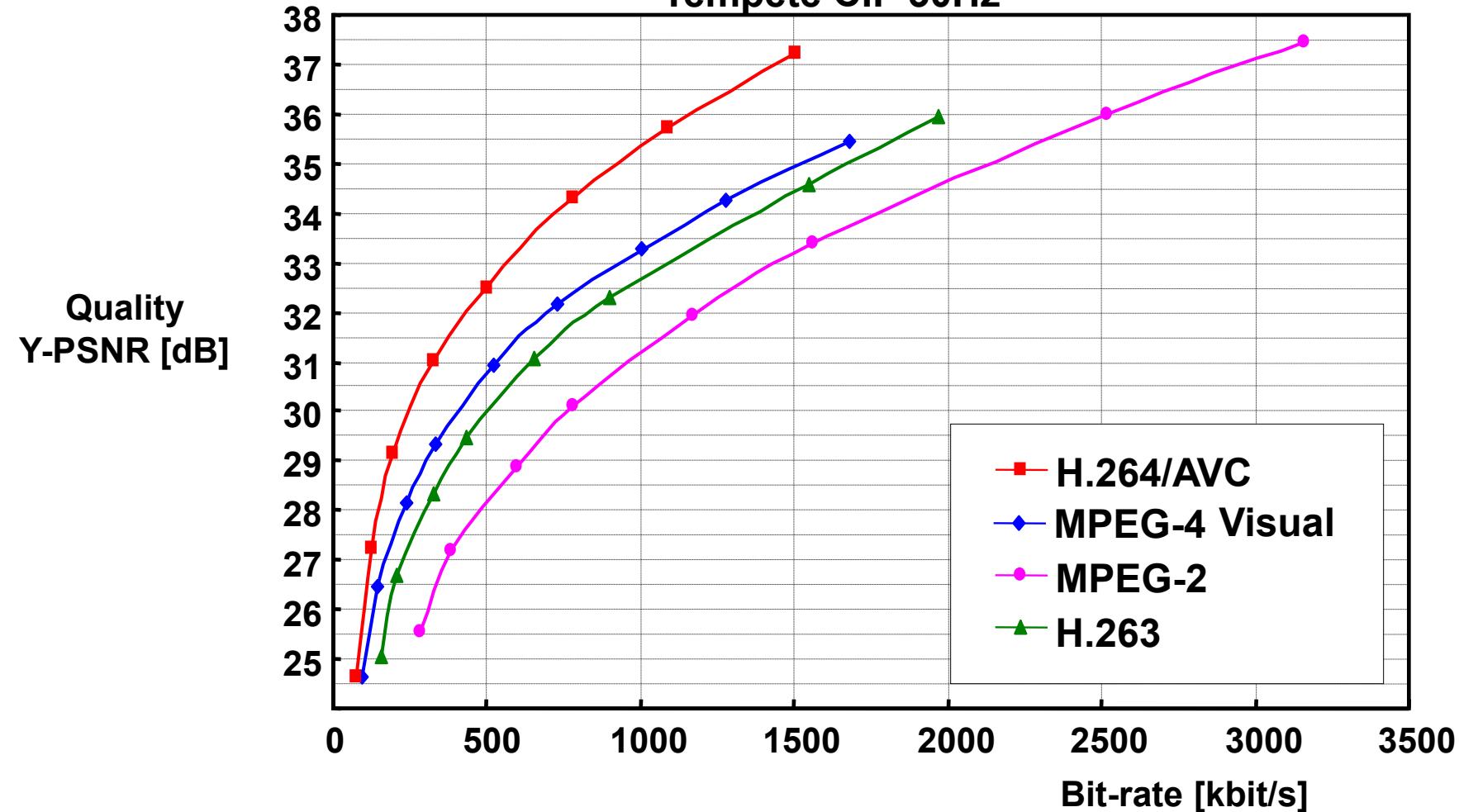


# Digital Video Processing

## Video Coding

### Comparison to MPEG-2, H.263, MPEG-4

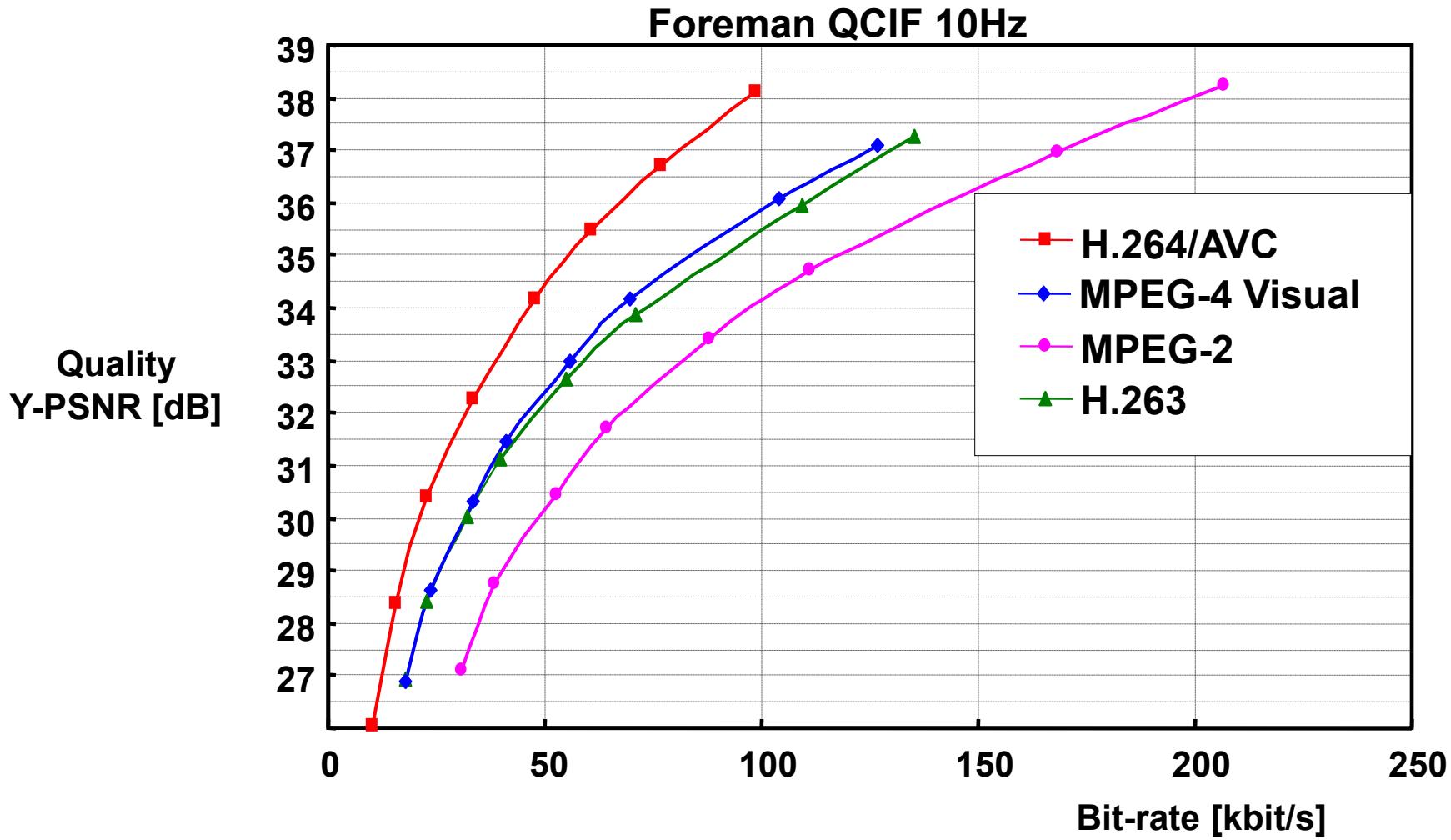
Tempete CIF 30Hz



# Digital Video Processing

## Video Coding

### Comparison to MPEG-2, H.263, MPEG-4



# Digital Video Processing

## Video Coding Standards

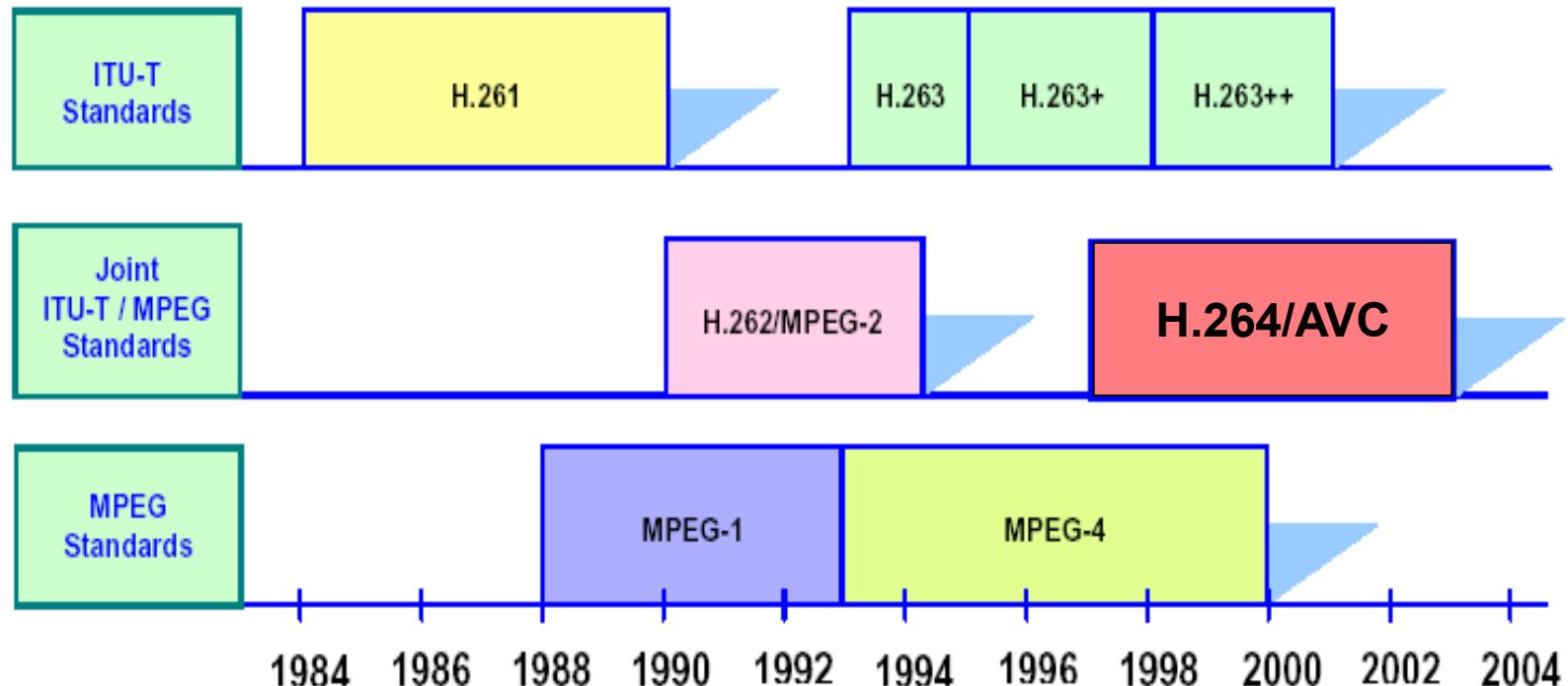


Figure 1. Progression of the ITU-T Recommendations and MPEG standards.

# Digital Video Processing

## Video Coding Standards

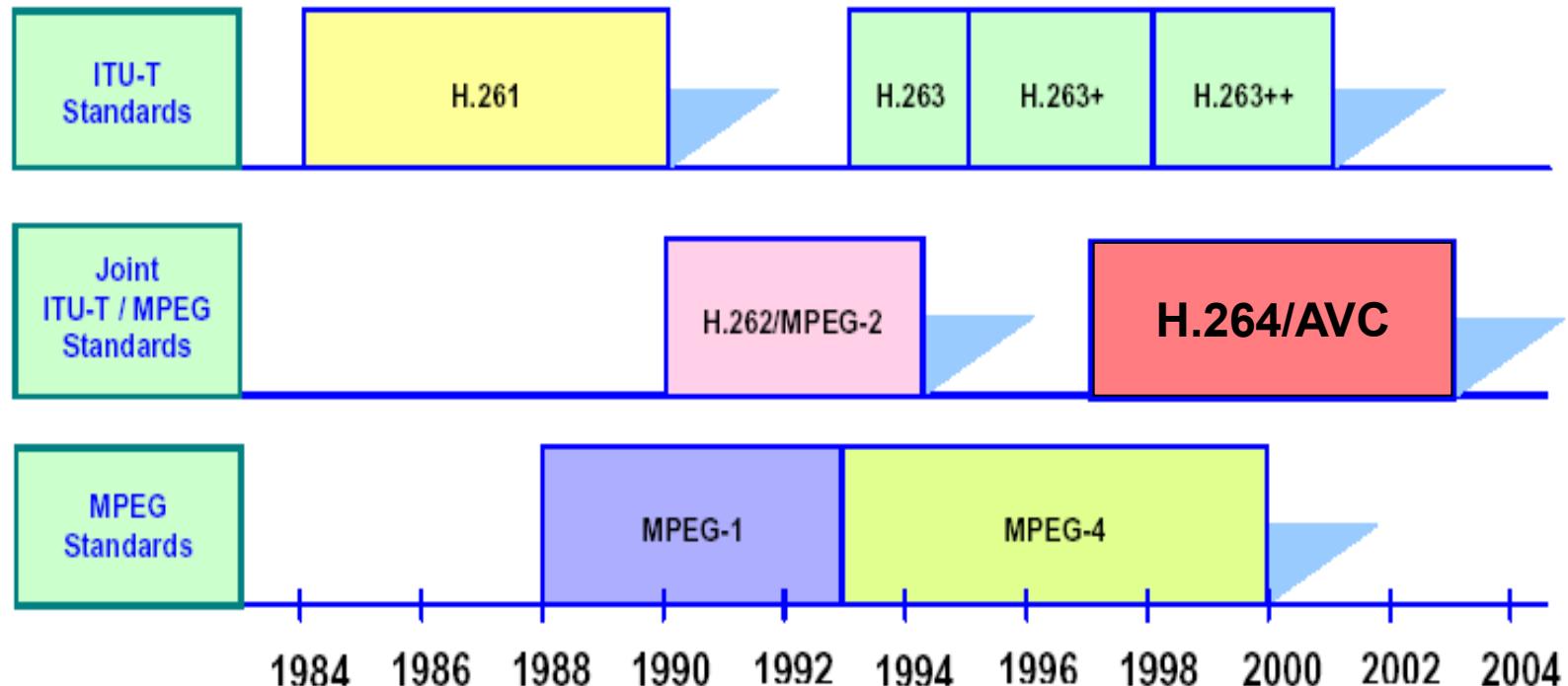


Figure 1. Progression of the ITU-T Recommendations and MPEG standards.

# Digital Video Processing

## Video Coding Standards

**ISO** (Int. Organization for Standardization)

**MPEG-1 (1992)**

**$1.5Mbps$ , VCD**

**MPEG-2 (1996)**

**$2-10Mbps$ , DVD**

**MPEG-4 (2000)**

**$8-1024Kbps$ , videophone**

**Digital cinema (ongoing)**

***windows media player(Microsoft)***

***real player(Real-Networks)***

**ITU** (Int. Telecommunication Union)

**H.261 (1990)**

**$p \times 64Kbps$**

**H.263**

**$8-64Kbps$ , videophone**

**H.263+/++**

**$8-64Kbps$ , videophone**

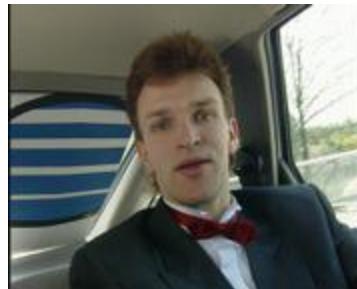
**H.264/AVC**

**Skype Video**

---

# Digital Video Processing

## Benchmark Videos



**Carphone**



**Suzie**



**Foreman**