

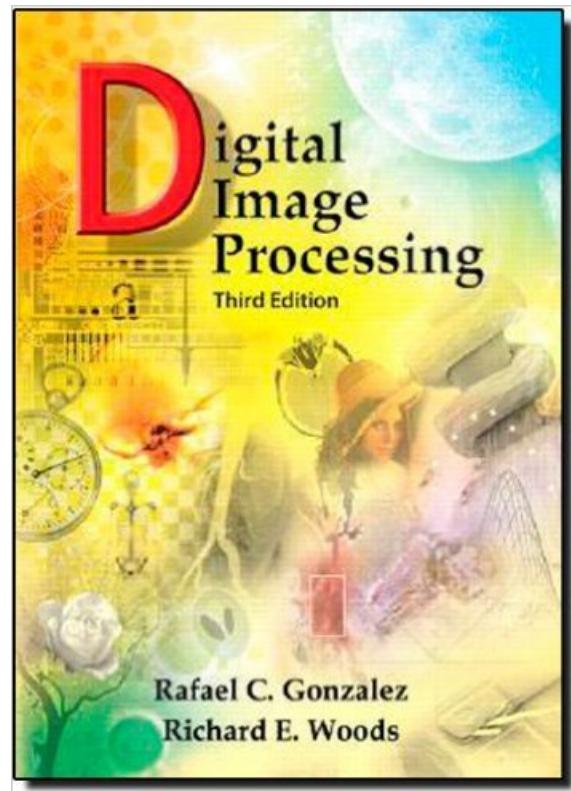
CSE 427

Digital Image Processing

Reference Book

- Digital Image Processing (*3rd Edition*)

Gonzalez and Woods

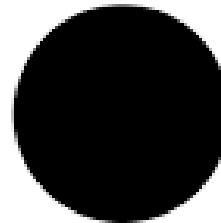




Goals

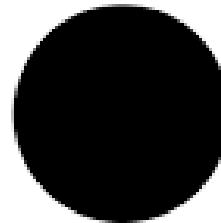
- To tell you what you can do with digital images
- To show you that doing research in image processing is fun and exciting
- To demonstrate that image processing is based on strong mathematical principles, applied to digital images via numerical schemes
- To show you that you can solve typical image processing tasks on your own

Image



Image

- A visual representation of something



Digital Image



Digital Image

- A visual representation of something captured by a digital medium

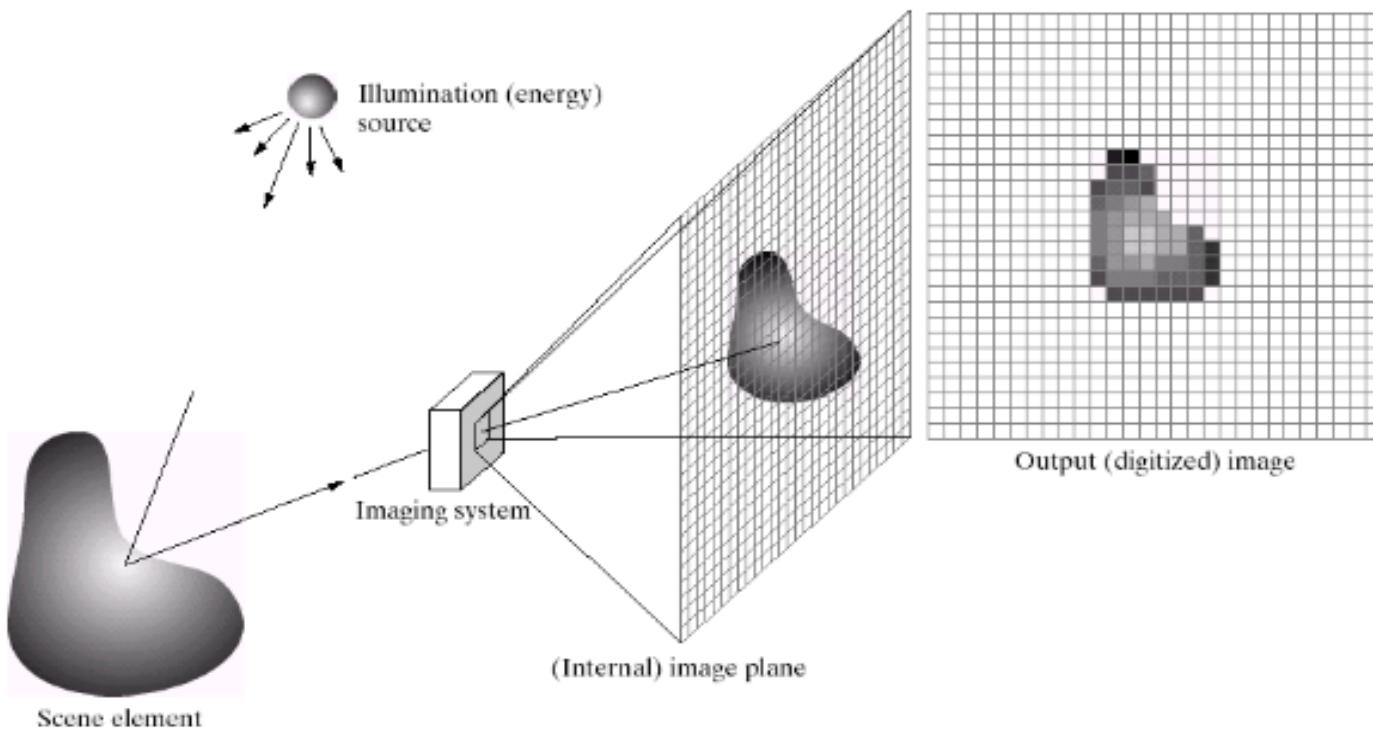


Digital Medium / Image Sensors



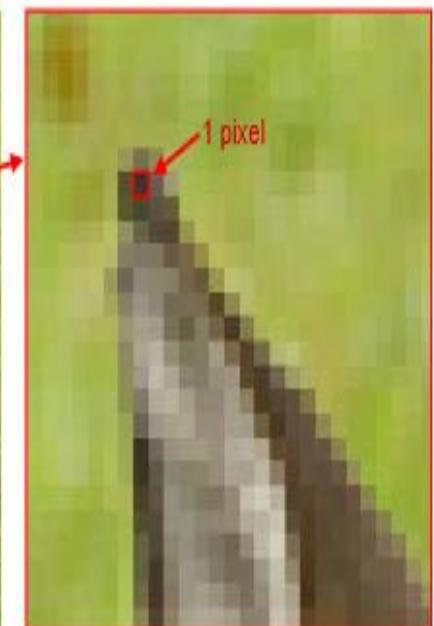
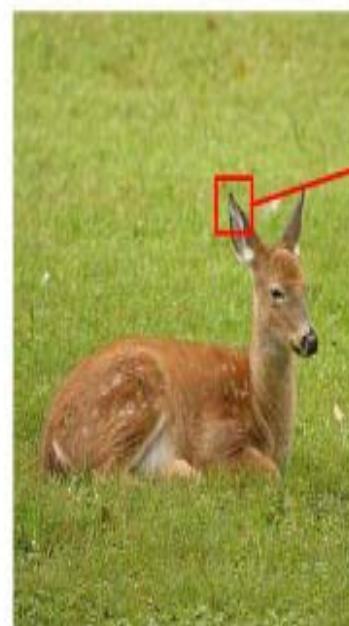
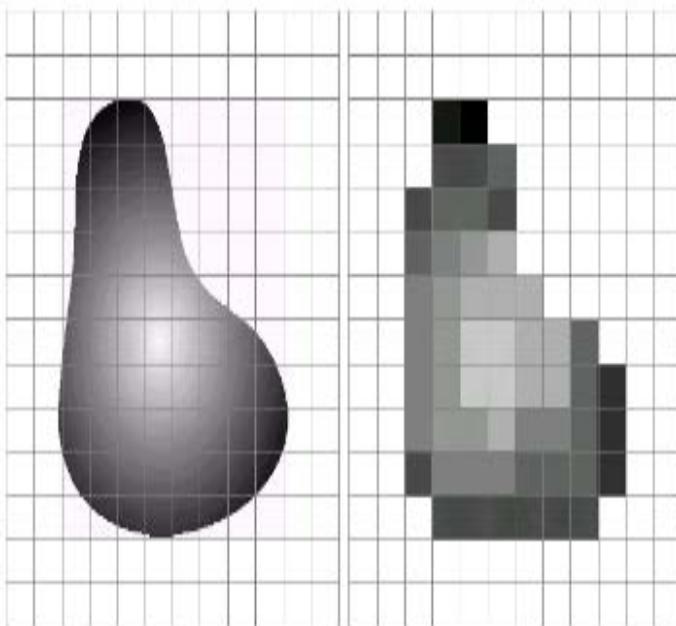
What is a Digital Image?

A digital image is a representation of a two-dimensional image as a finite set of digital values, called picture elements or pixels.



What is a Digital Image?

- Pixel values typically represent gray levels, colors.
- Digital image is an approximation of a real scene.



What is a Digital Image?

Common image formats include:

- 1 sample per point (B&W or Grayscale)
- 3 samples per point (Red, Green, and Blue)
- 4 samples per point (Red, Green, Blue, and “Alpha”).

Alpha is for transparency (0.0 fully transparent and 1.0 full opaque (not Transparent)).

What is a Digital Image?

- A fixed number of rows and columns of pixels.
- The smallest individual element in an image is pixel.
- In **digital imaging** a pixel, pel, or picture element is a physical point in a image.

What is a Digital Image?

- Each pixel is a **sample** of an original image.
- More samples typically provide more accurate representations of the original.
- The **intensity** of each pixel is variable.

What is a Digital Image?

- In color imaging systems, a color is typically represented by three or four component intensities such as

For 3 colors:
red, green, and blue, or

For 4 colors:
cyan, magenta, yellow, and black.

What is a Digital Image?

Depending on context, synonyms of pixel as follows:

- pel,
- sample,
- byte,
- bit,
- dot, and
- spot.

Pixels can be used as a unit of measure such as: 2400 pixels per inch, 640 pixels per line, or spaced 10 pixels apart.

What is a Digital Image?

- dots per inch (dpi) and
- pixels per inch (ppi)

For printer devices, dpi is a measure of the printer's density of dot (e.g. ink droplet) placement.

A high-quality photographic image may be printed with 600 ppi on a 1200 dpi inkjet printer.

What is a Digital Image?

Bits Per Pixel (bpp)

The number of distinct colors that can be represented by a pixel depends on the number of bits per pixel (bpp).

A 1 bpp image uses 1-bit for each pixel, so each pixel can be either on or off. Each additional bit doubles the number of colors available, so a 2 bpp image can have 4 colors, and a 3 bpp image can have 8 colors.

What is an Image/Digital Image?

1 bpp, $2^1 = 2$ colors (**monochrome**)

2 bpp, $2^2 = 4$ colors

3 bpp, $2^3 = 8$ colors

...

...

...

8 bpp, $2^8 = 256$ colors

16 bpp, $2^{16} = 65,536$ colors ("Highcolor")

24 bpp, $2^{24} = 16,777,216$ colors ("Truecolor")

What is an Image/Digital Image?

$2^x \rightarrow$ Here x means color depths per pixel.

Depth is normally the sum of the bits allocated to each of the red, green, and blue components.

$2^{15} \rightarrow$ Means 5 red, 5 green and 5 blue (divide by 3 channels and always red and blue are equal).

16 bpp means ???

What is an Image/Digital Image?

Highcolor: usually meaning 16 bpp, normally has

- a) Five bits for red and blue each, and
- b) Six bits for green,

As the human eye is more sensitive to errors in green than in the other two primary colors.

What is an Image/Digital Image?

For applications involving **transparency**, the 16 bits may be divided into five bits each of red, green, and blue, with one bit left for transparency.

A 24-bit depth allows 8 bits per component.

On some systems, 32-bit depth is available: this means that each 24-bit pixel has an extra 8 bits to describe its **opacity** (for purposes of combining with another image).

What is an Image/Digital Image?

A megapixel (MP) is a million pixels.

The term is used not only for the number of pixels in an image but also

- To express the number of **image sensor** elements of **digital cameras** or
- The number of display elements of **digital displays**.

What is an Image/Digital Image?

In most digital cameras, the sensor array is covered with a patterned color filter having

- Red,
- Green, and
- Blue regions

What is an Image/Digital Image?

Digital image processing focuses on two major tasks

- Improvement of pictorial information for human interpretation
- Processing of image data for storage, transmission and representation for autonomous machine perception.

What is an Image/Digital Image?

- What is the input for image processing or image analysis ???

- What is the output of image processing or image analysis???

Arithmetic Operations

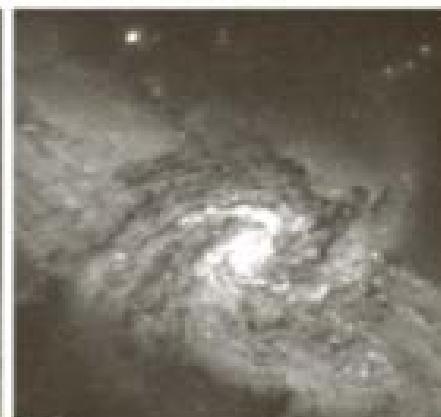
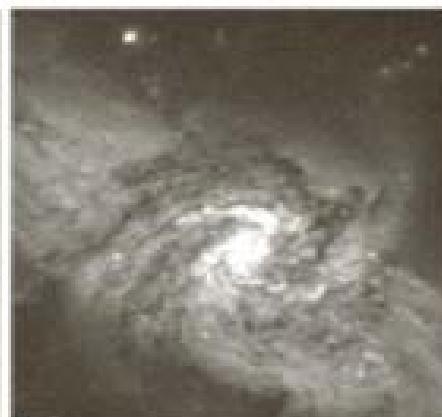
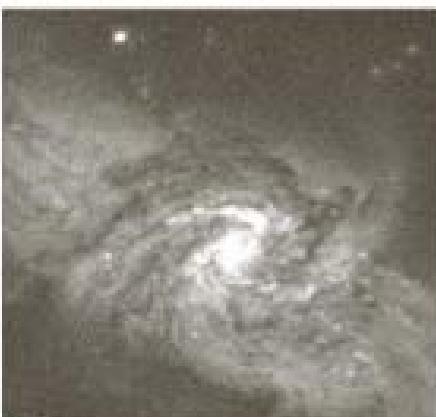
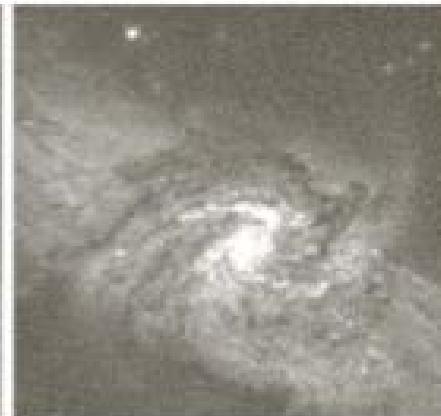
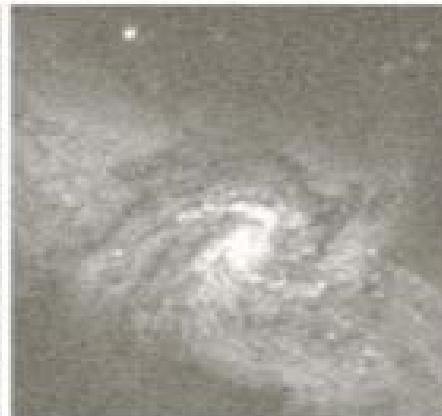
- Arithmetic operations on pixel values

Addition, subtraction, multiply, divide

$$h = f + g \Rightarrow h(i, j) = f(i, j) + g(i, j)$$

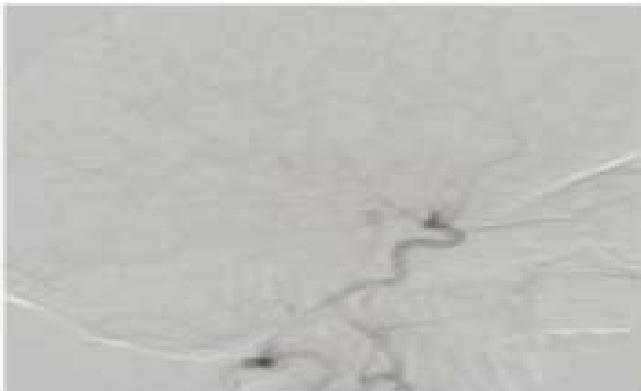
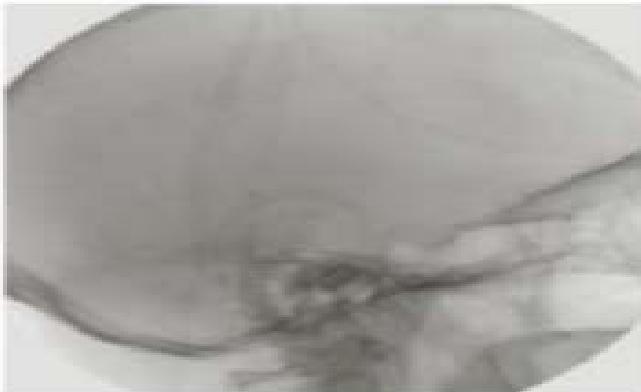
Arithmetic operations: $f + g$

- Averaging (adding) multiple images can reduce



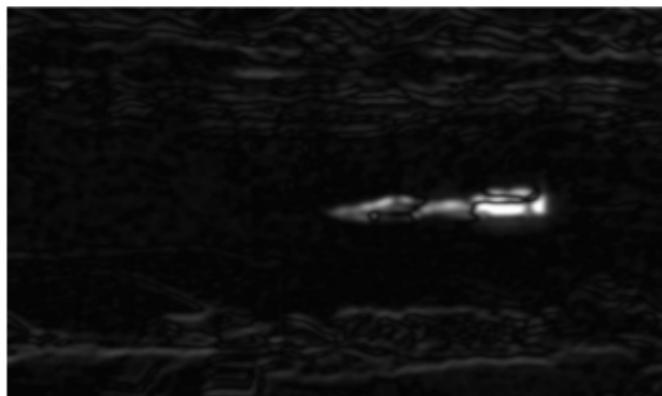
Arithmetic operations: $f - g$

- Digital Subtractive Angiography (DSA)



Arithmetic operations: $f - g$

- Image Subtraction (Motion Detection)



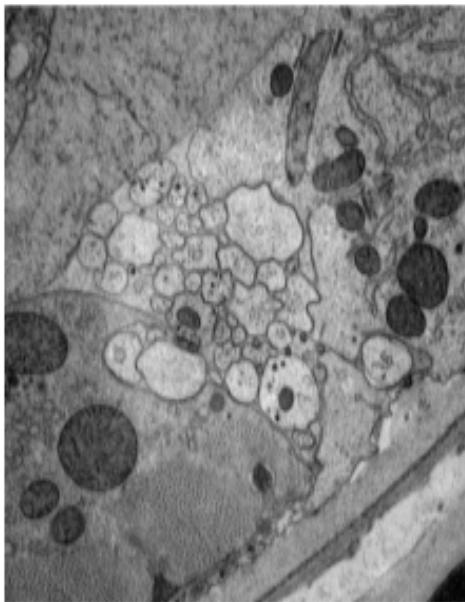
Arithmetic operations: $f \times g$



a b c

FIGURE 2.30 (a) Digital dental X-ray image. (b) ROI mask for isolating teeth with fillings (white corresponds to 1 and black corresponds to 0). (c) Product of (a) and (b).

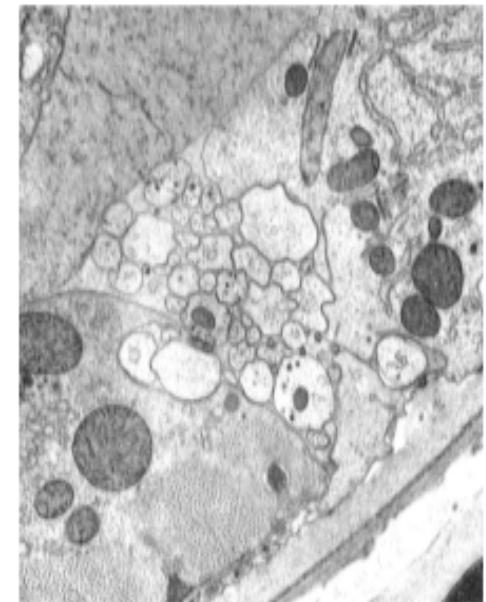
Arithmetic operations: f / g



Captured image

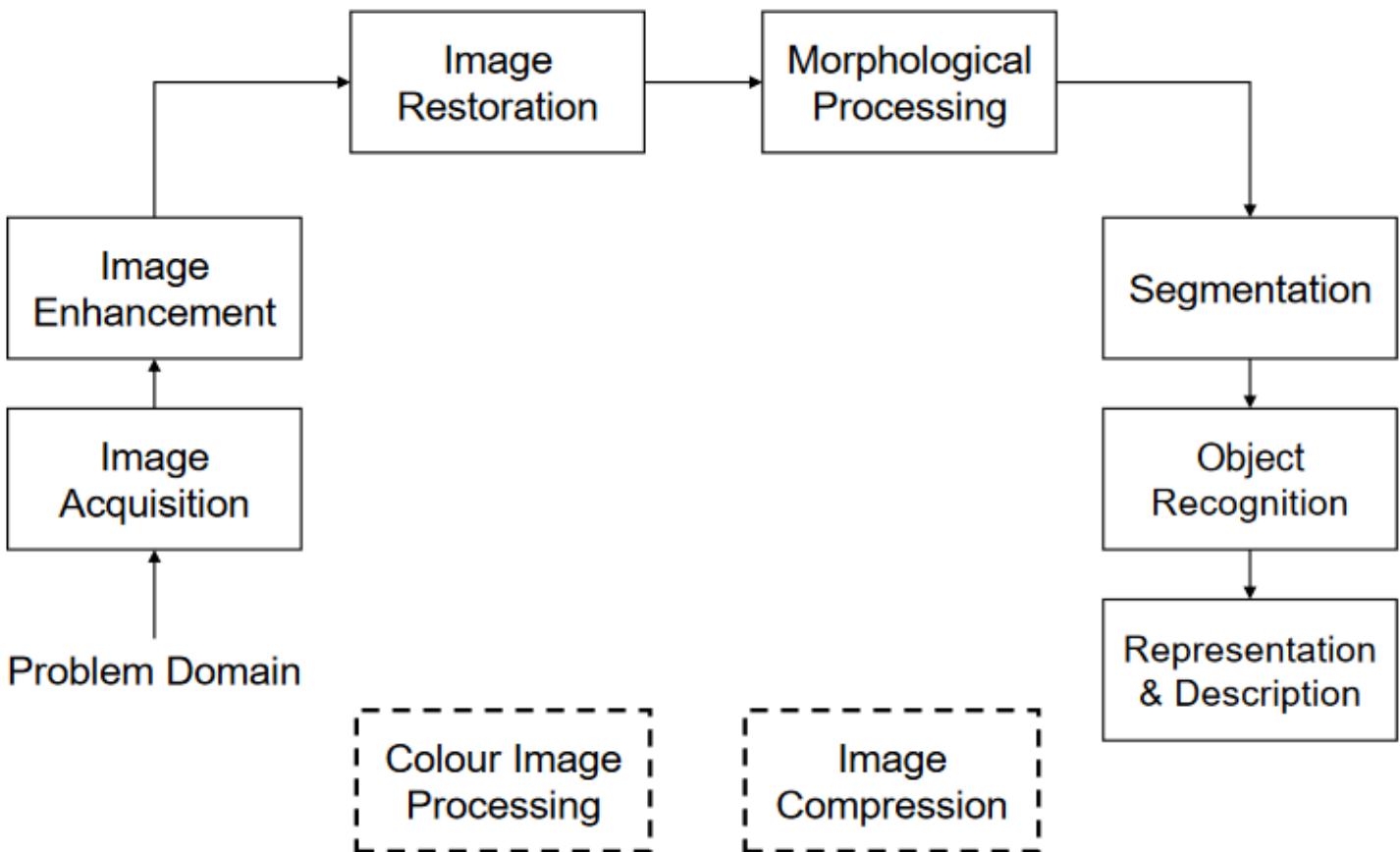


Illumination

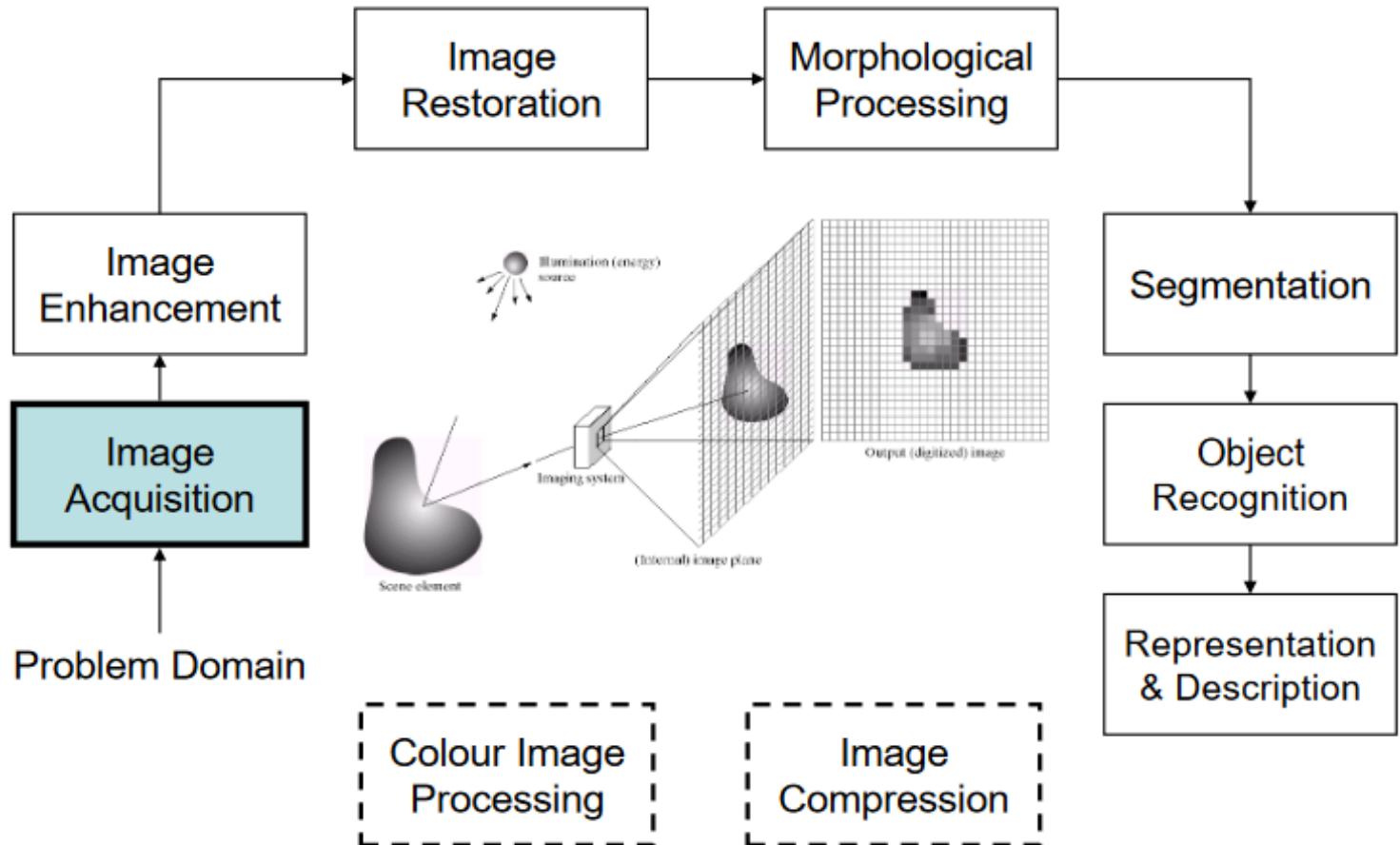


Corrected image

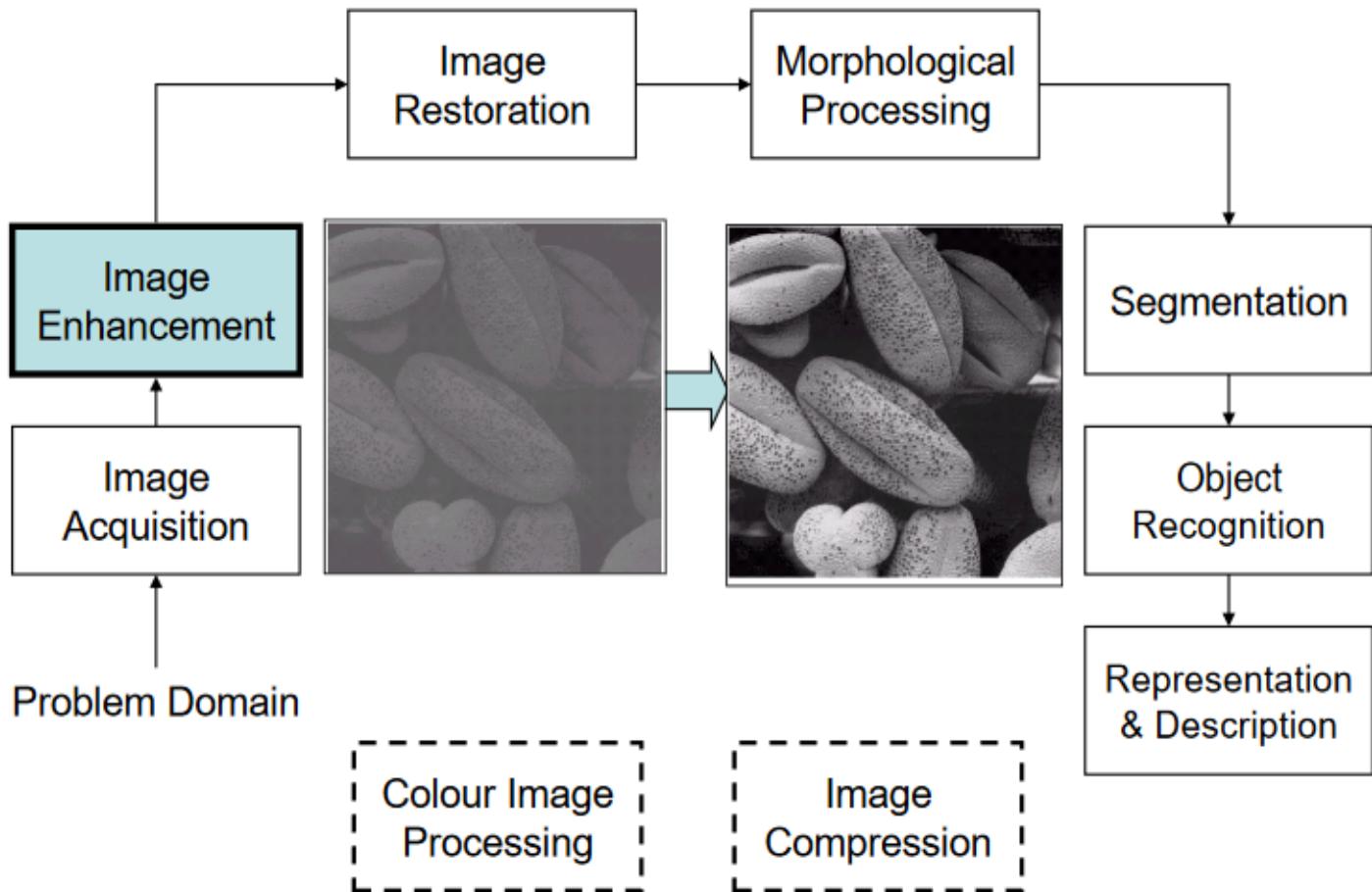
Key Stages in DIP



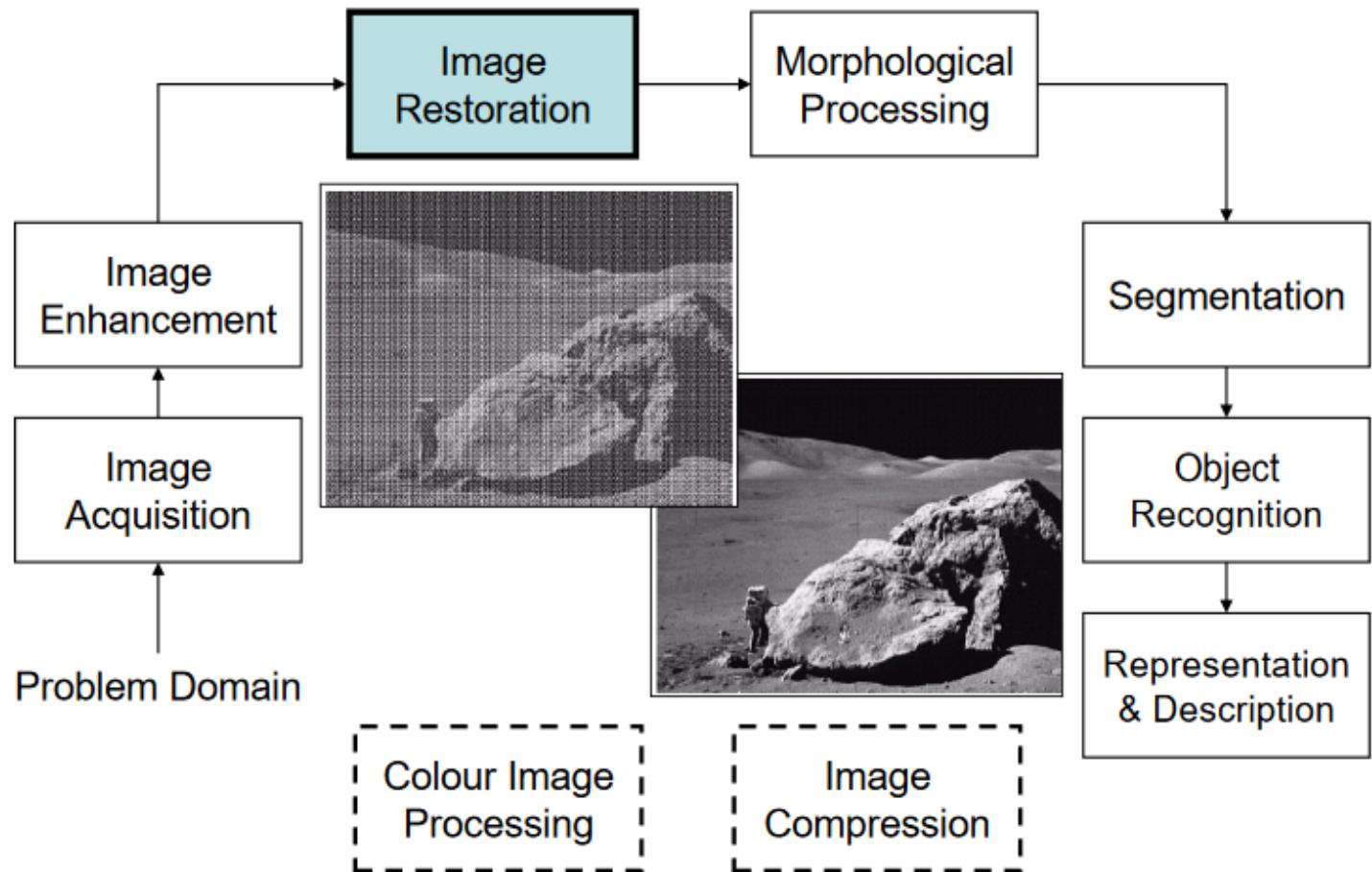
Key Stages in DIP



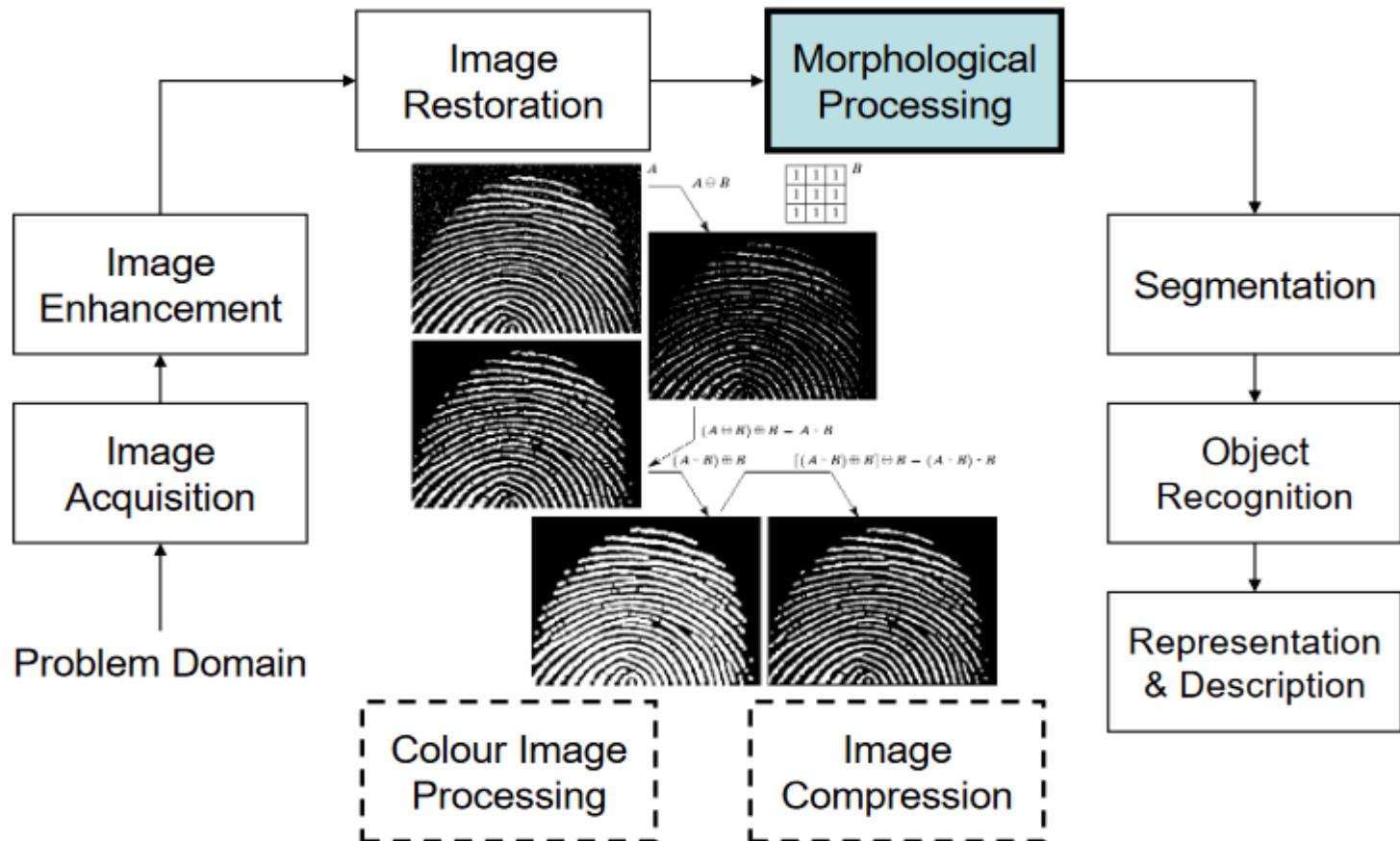
Key Stages in DIP



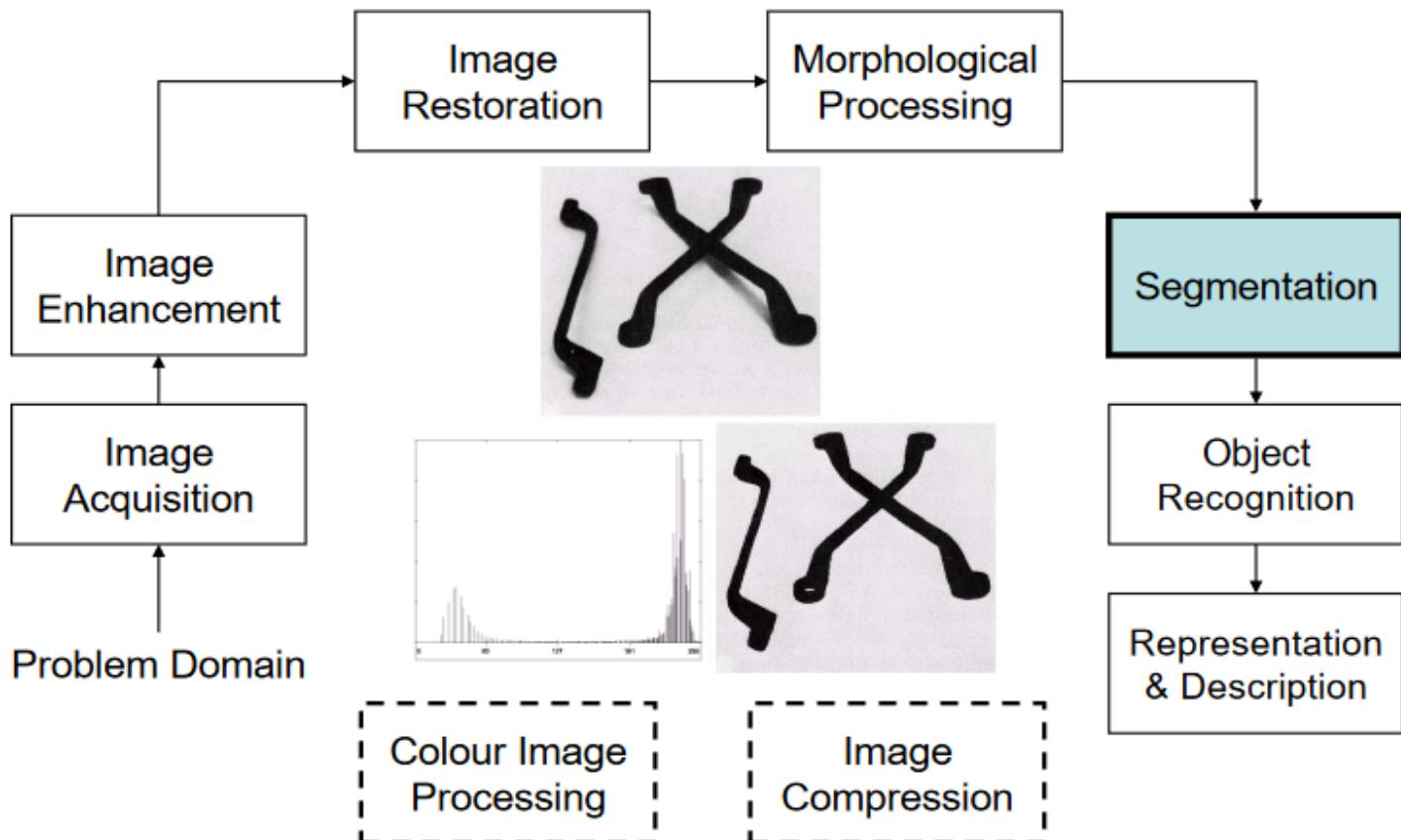
Key Stages in DIP



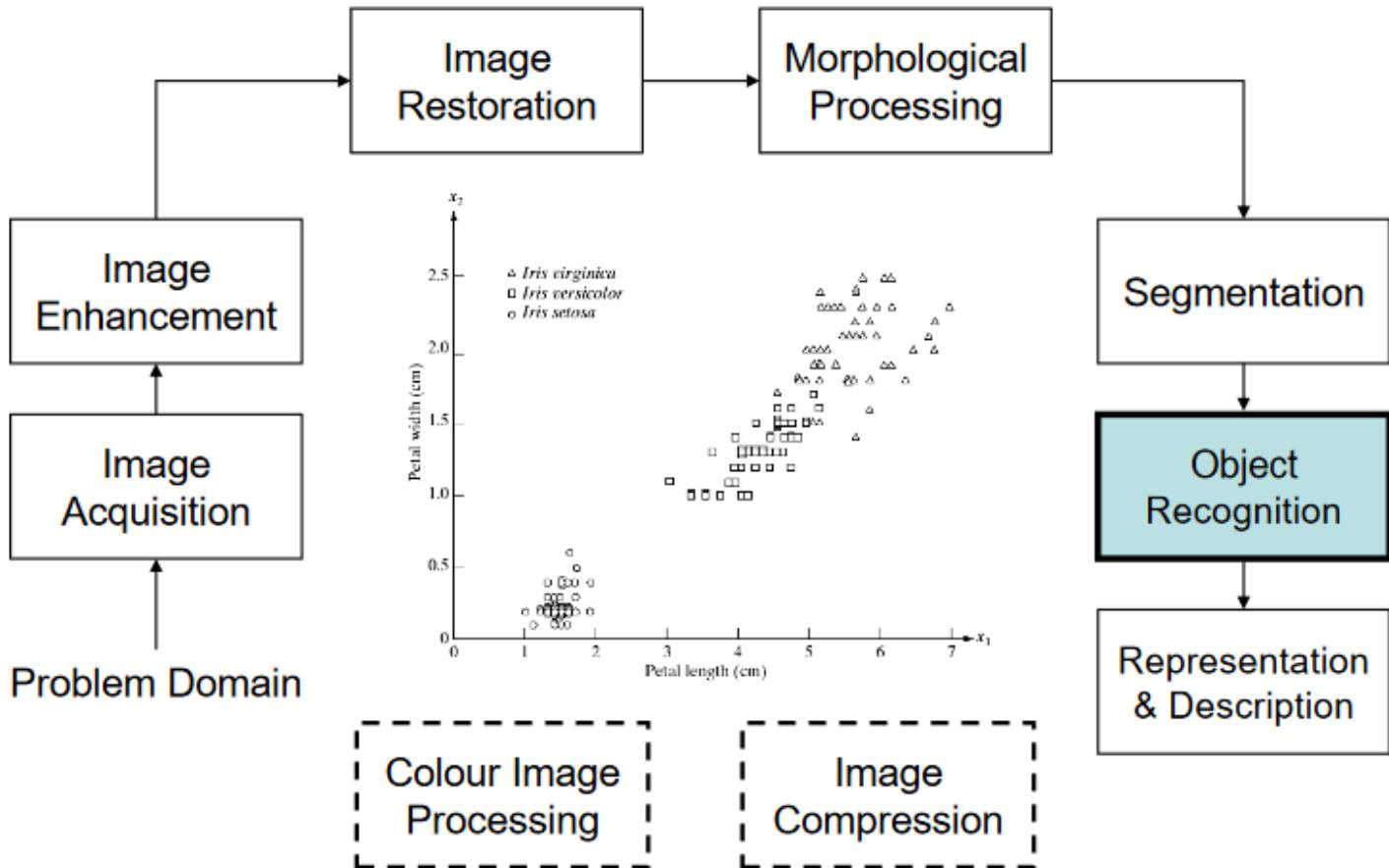
Key Stages in DIP



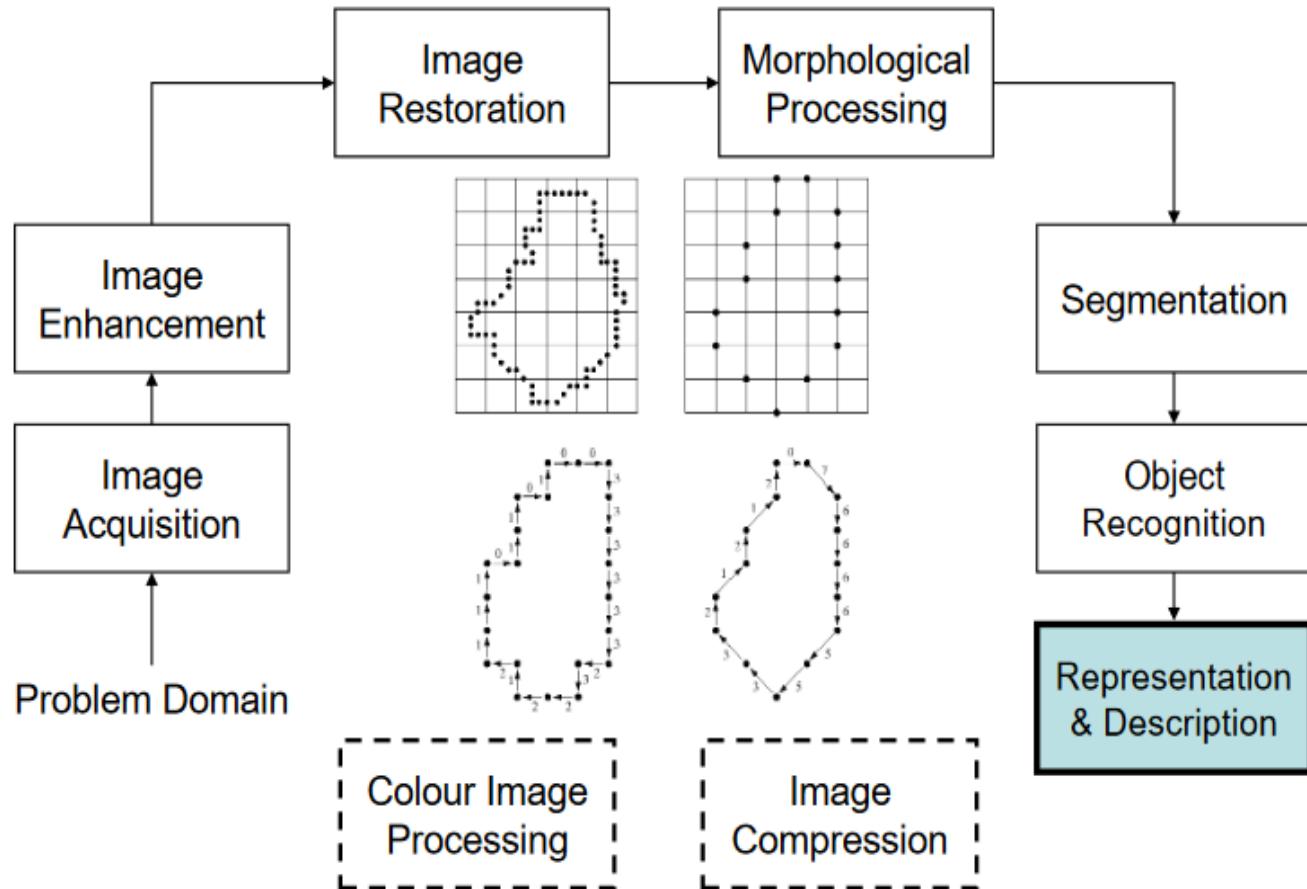
Key Stages in DIP



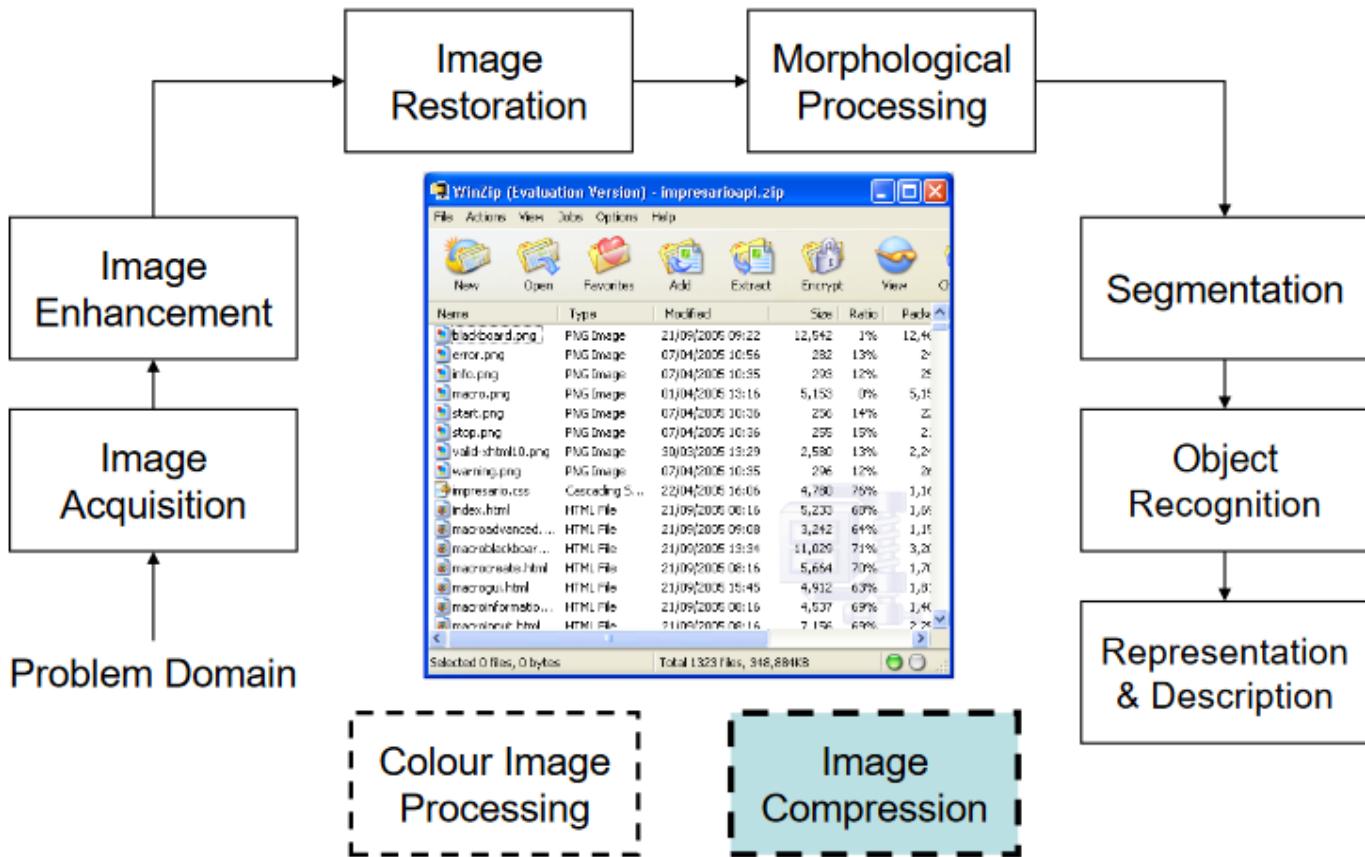
Key Stages in DIP



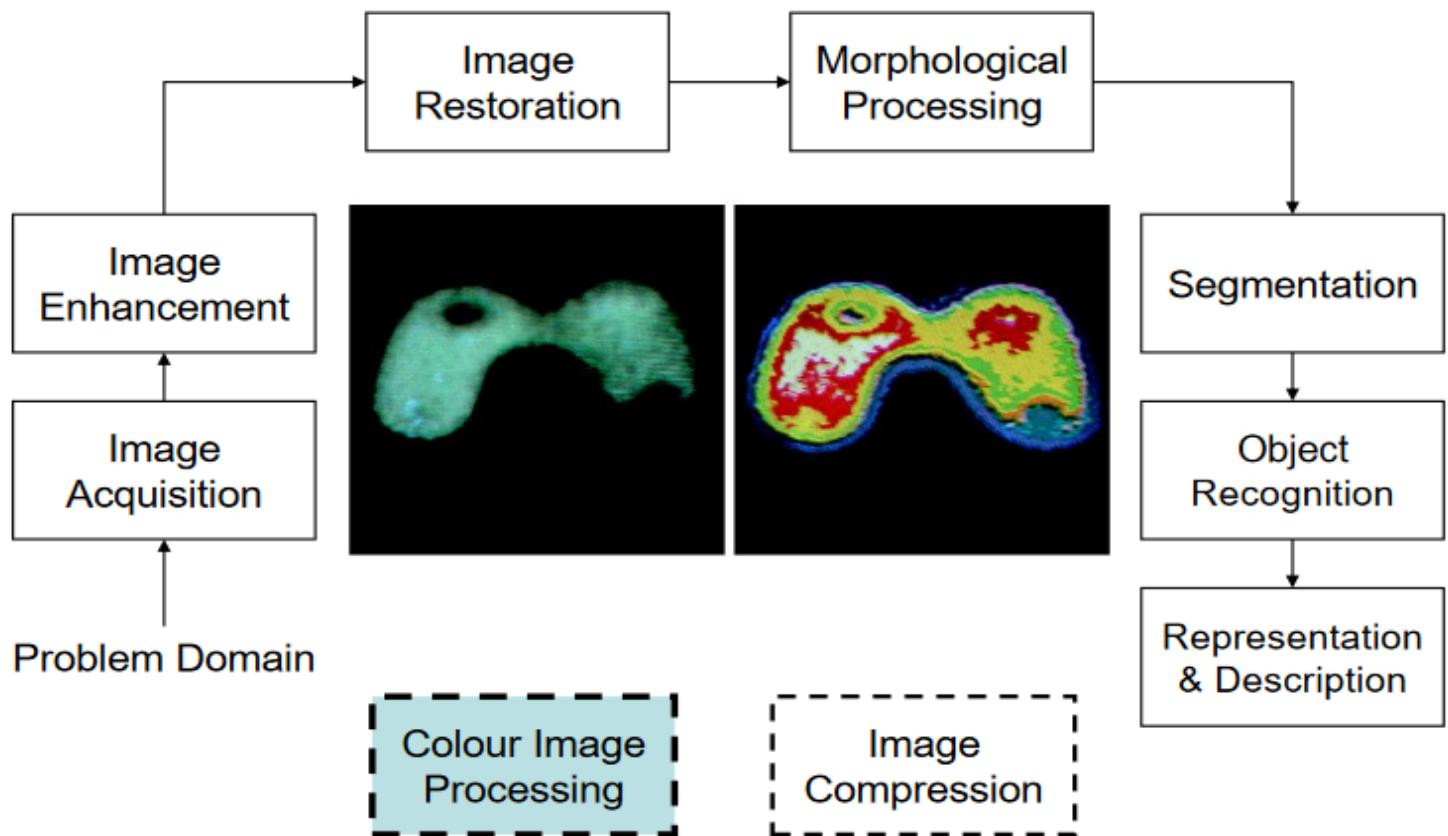
Key Stages in DIP



Key Stages in DIP

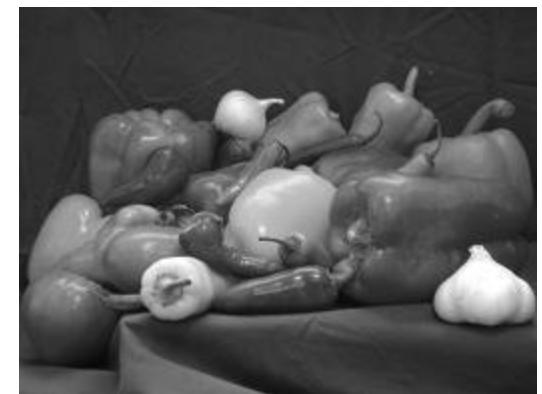


Key Stages in DIP



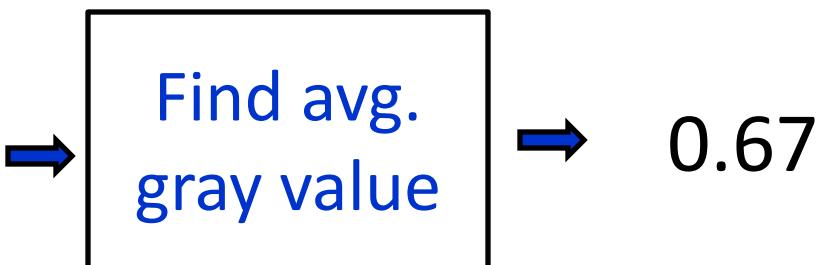
Digital Image Processing

- Processing of *digital images* using *digital devices* (*computers*)



Scope of DIP

- Is it DIP?



Why we need DIP

- DIP is not limited to image retrieval (IR)
- Thousands of others areas:
 - Core area
 - Application area

Why we need DIP

- Core area
 - Image enhancement
 - Image de-noising
 - Image segmentation
 - Image & video retrieval
 - Image Security
 - Watermarking and registration
 - Image compression

Why we need DIP

- Application area
 - Security and surveillance
 - Biometric application: face, iris, finger print, palm print recognition . .
 - Medical imaging
 - Automatic scanning and detection: X-ray, CT, MRI, PET, angiogram, ECG, echo, endoscopy,

Why we need DIP

- Application area
 - Document Classification
 - Binarization, segmentation, character recognition
 - Object and shape recognition
 - Intelligent transport system
 - Environmental monitoring and remote sensing
 - and so on, . . .

Digital Image ...

An image can be defined as

➤ a two-dimensional function, $f(x,y)$,
where

- a) x and y are spatial (plane) coordinates
- b) amplitude of f at any pair of coordinates (x,y)

is called the intensity of the image at that point.

Digital Image

The term gray level is often used to refer to the intensity of monochrome images.

A monochromatic object or image reflects colors in shades of limited **colors** or **hues**. Images using only shades of grey (with or without black or white) are called **grayscale** or **black-and-white**.

Hue is one of the main properties (called **color appearance parameters**) of a **color**.

Digital Image Processing

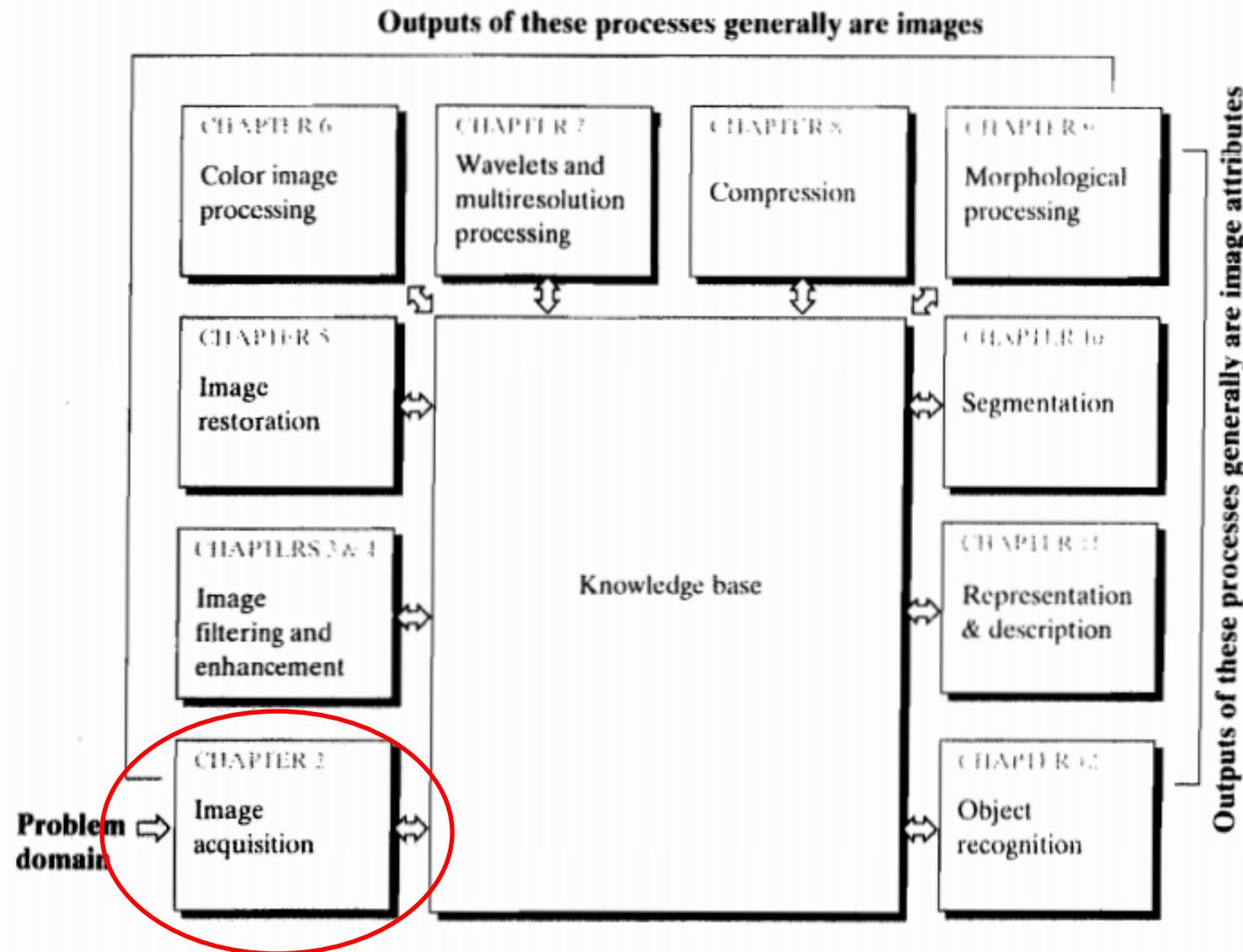
- Digital image processing refers to processing digital images by means of a digital computer.
- A digital image is composed of a finite number of elements, each of which has a particular location and value.
- These elements are referred to as picture elements, image elements, pels and pixels.
- Pixel is the term most widely used to denote the elements of a digital image.

Chapter 2

Digital Image Fundamentals

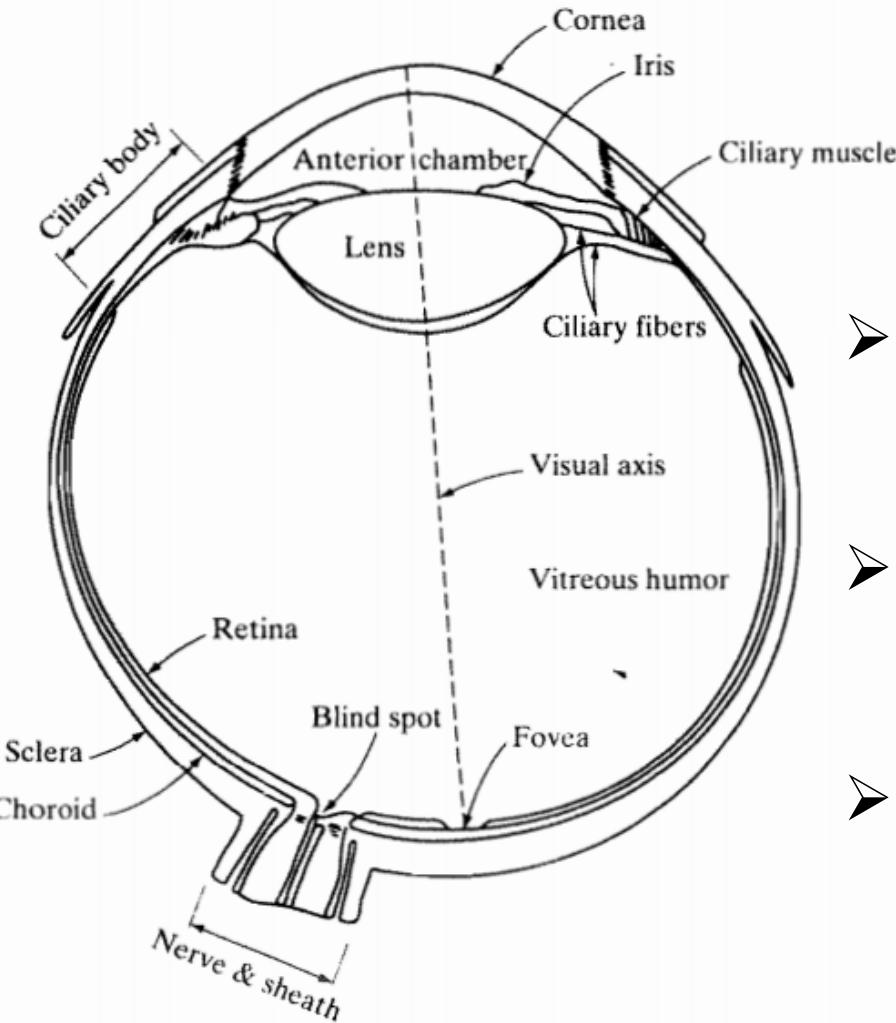
Remember?

FIGURE 1.23
Fundamental steps in digital image processing. The chapter(s) indicated in the boxes is where the material described in the box is discussed.



Outputs of these processes generally are image attributes

Structure of Human Eye



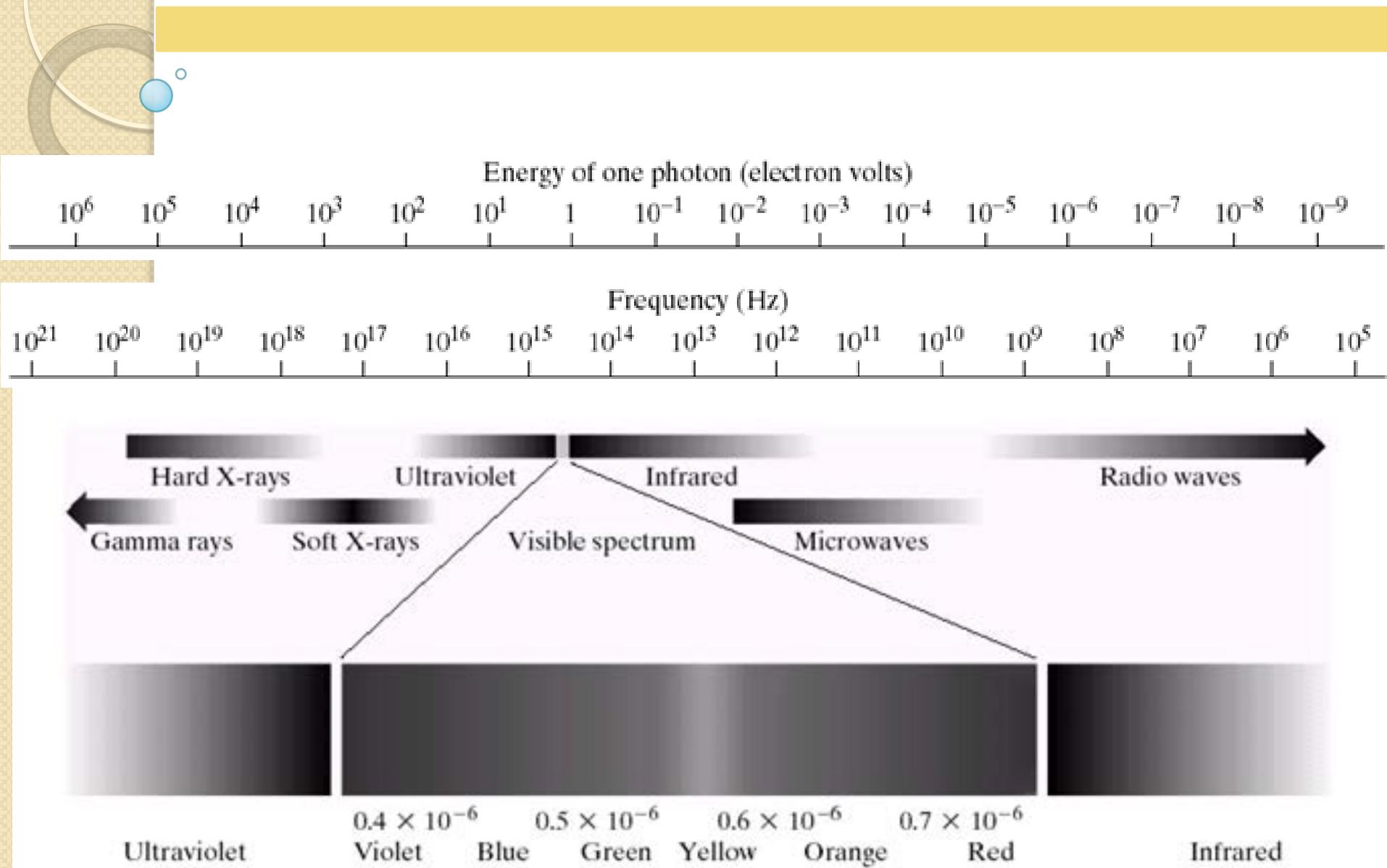
- No need to study for term final.
- You can safely skip section 2.1
- Interested readers are requested to go through it.

EM Spectrum

The **electromagnetic spectrum** is the term used by scientists to describe the entire range of light that exists.

From radio **waves** to gamma rays, most of the light in the universe is, in fact, invisible to us!

EM Spectrum



EM Spectrum

Monochromatic Light

- Light that is void of color is called **monochromatic light**. (A photograph or picture developed or executed in black and white or in varying tones of only one color.)
- The only attribute of monochromatic light is **intensity**. (can be referred to as brightness)
- The intensity of monochromatic light is perceived to vary from black to grays and finally to white.

EM Spectrum

Monochromatic Light

- The term **gray level** is used to commonly denote monochromatic intensity.
- The range of measured values of monochromatic light from black to white is usually called the **gray scale** and monochromatic images are often referred to as **gray-scale images**.

EM Spectrum

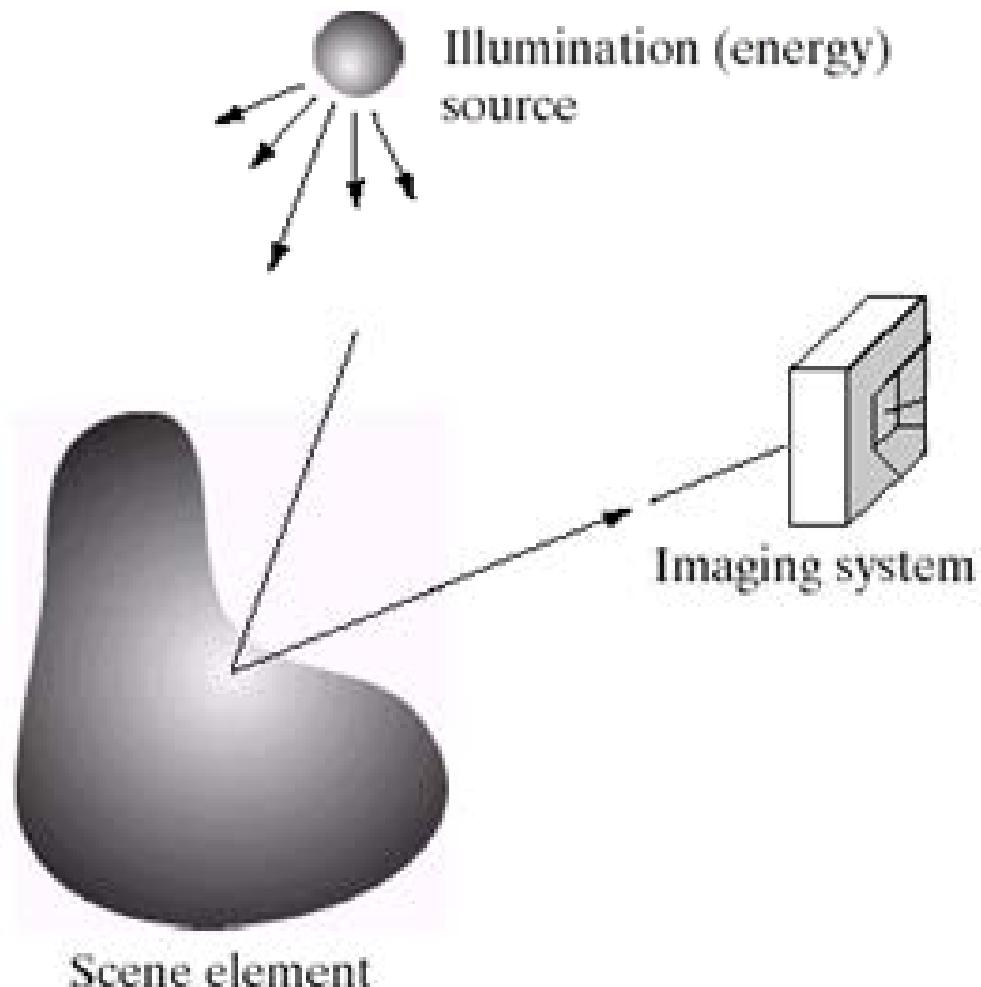
Chromatic Light

- Chromatic light spans the electromagnetic energy spectrum from approximately 0.43 to 0.79 μm

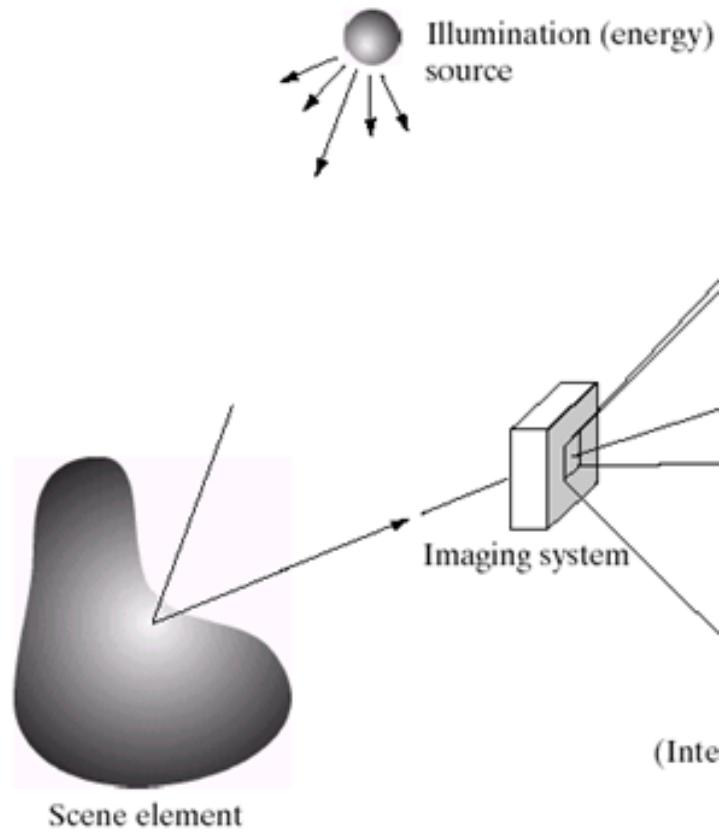
Image Acquisition

Image Acquisition

- Three main elements
- Illumination source
 - Scene
 - Sensor (imaging system)

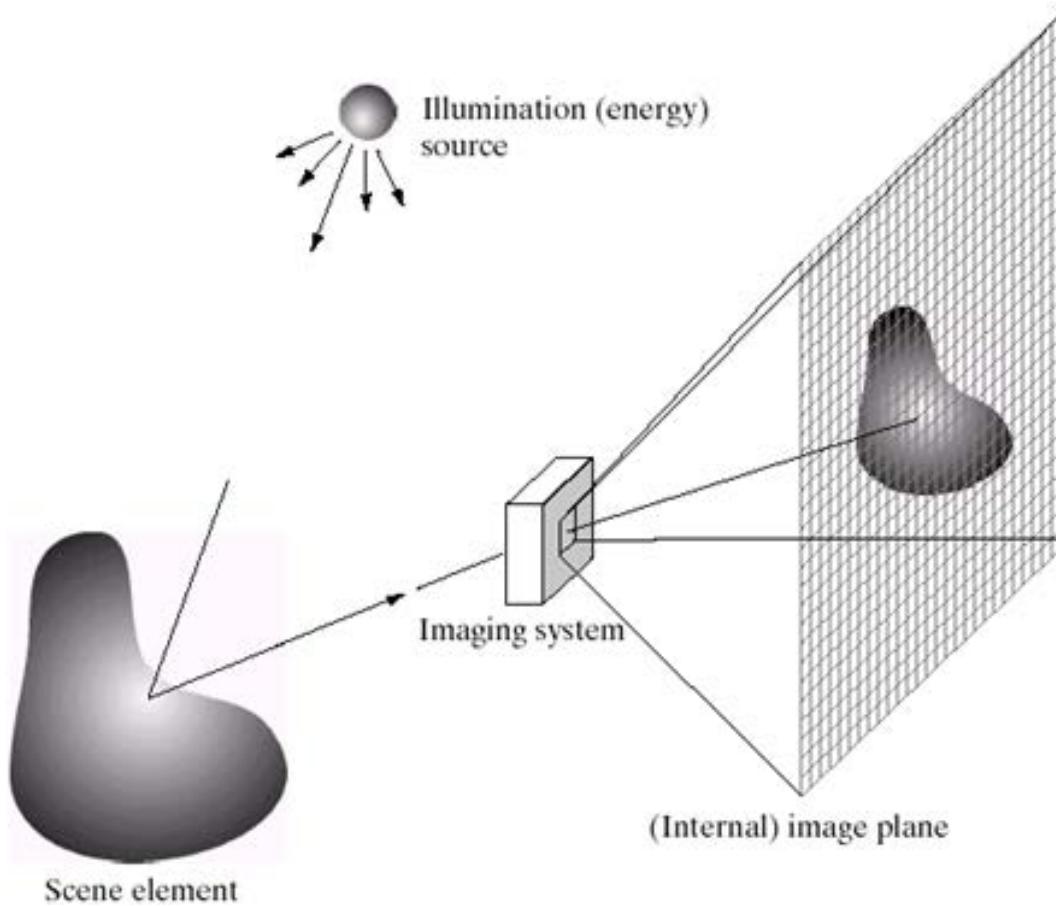


Sensor Array



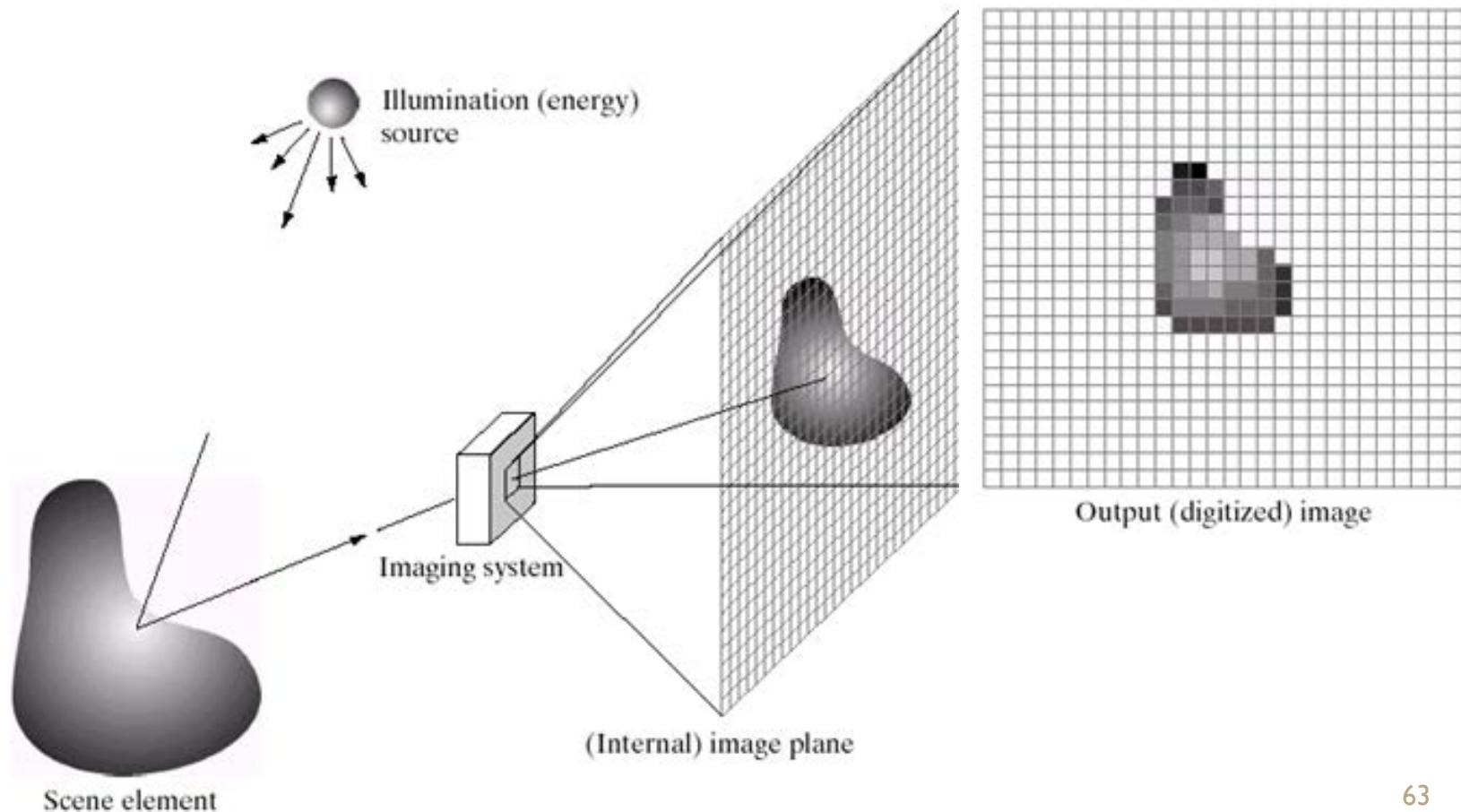
Sensor Array

The sensor array produces outputs proportional to the integral of the light received at each sensor.



Sensor Array

- Digital and analog circuitry sweep these outputs and convert them to an analog signal which is then digitized by another section of the imaging system



A Simple Image Formation Model

Recall from previous class:

An image has two basic attributes:

- A location (x, y) : picture element, **pixel**, pel, etc
- A value $f(x, y)$ at pixel (x, y) : gray scale value

$f(x, y)$ represents intensity which is proportional energy radiation.

$$0 < f(x, y) < \infty$$

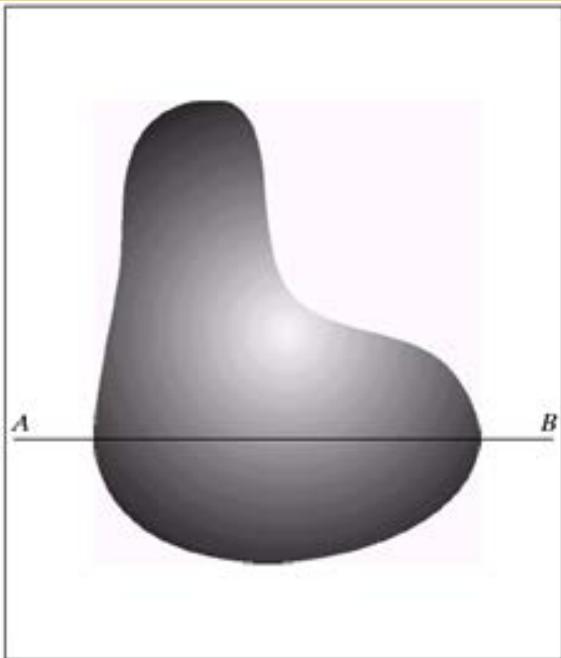
Image Digitization

Image Digitization

Two steps :

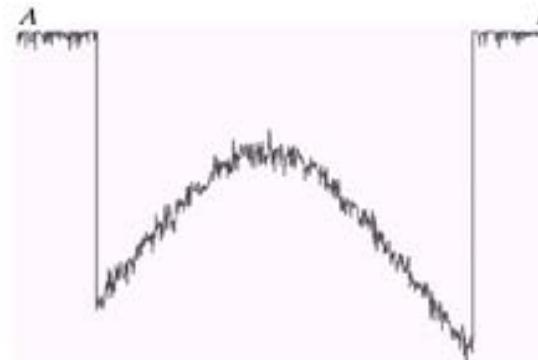
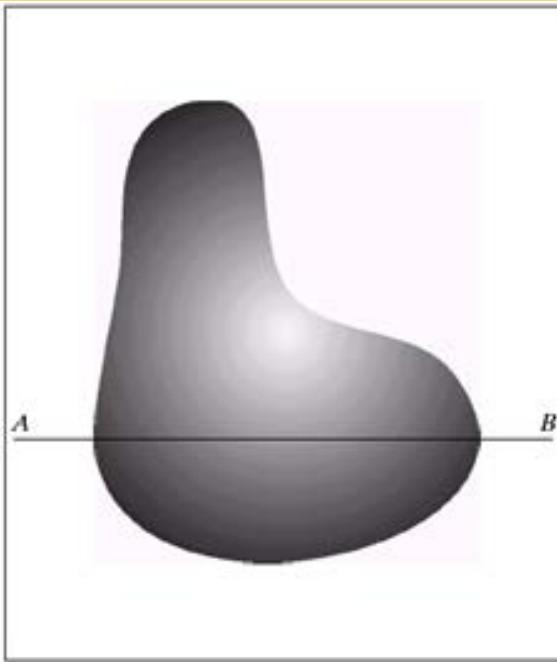
1. **Sampling** : Digitizing the coordinates values is called sampling.
2. **Quantization** : Digitizing the amplitude values is called quantization.

Image Digitization



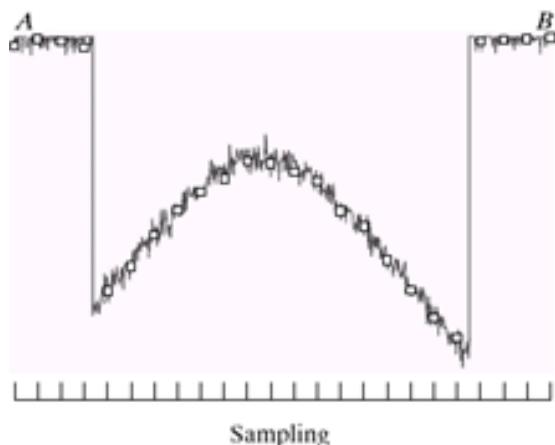
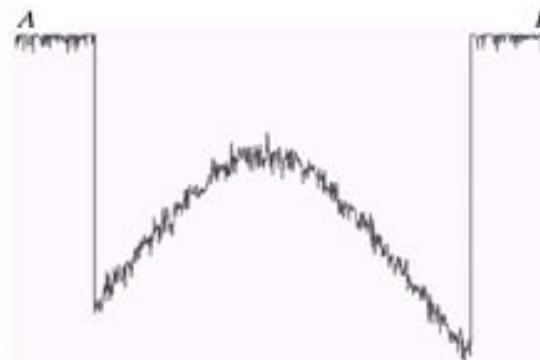
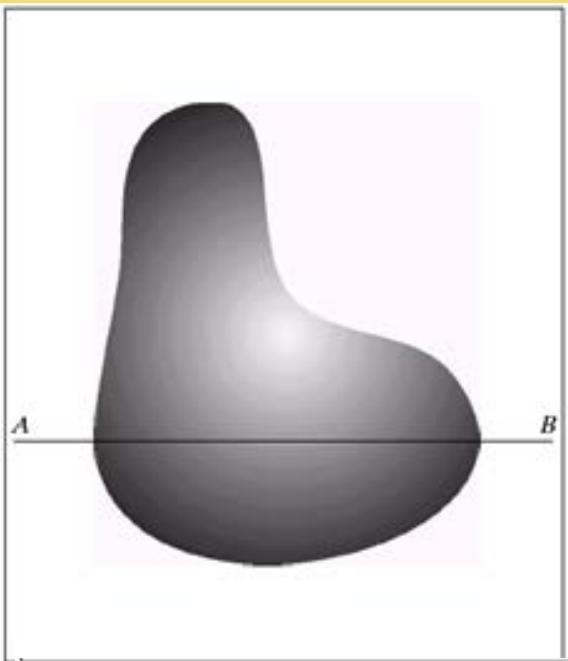
Let's start with a gray scale image.
We will try to digitize this image
along the horizontal line segment
AB

Image Digitization



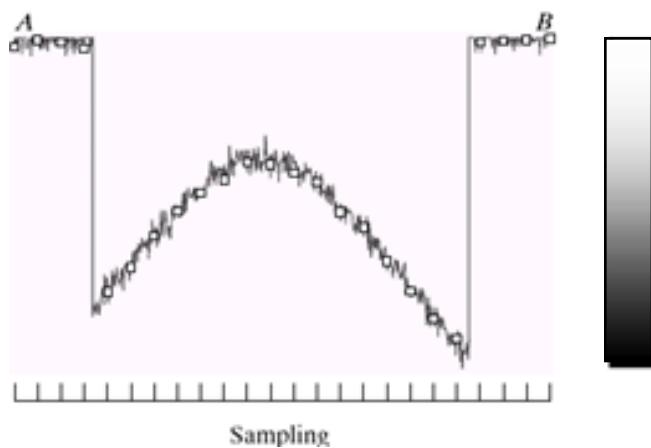
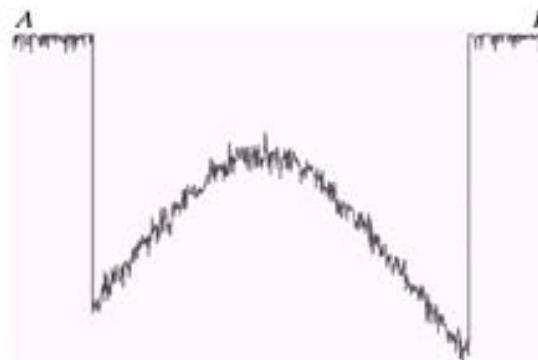
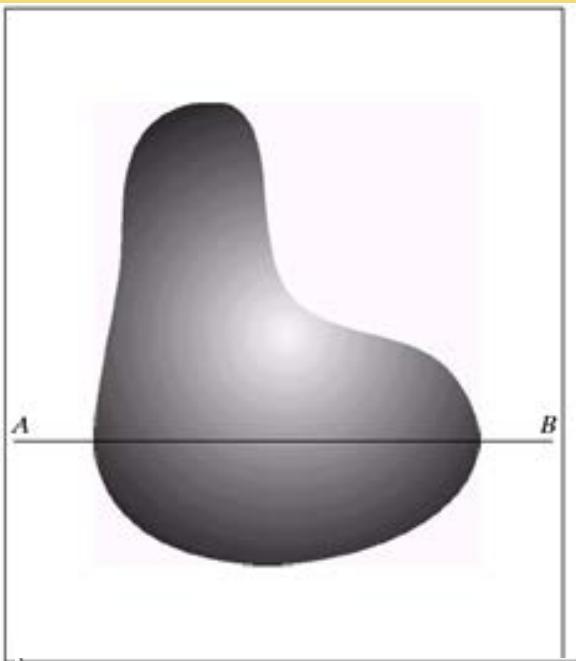
This is a plot of amplitude values of the continuous image along the line segment AB. Random variations are due to image noise.

Image Digitization



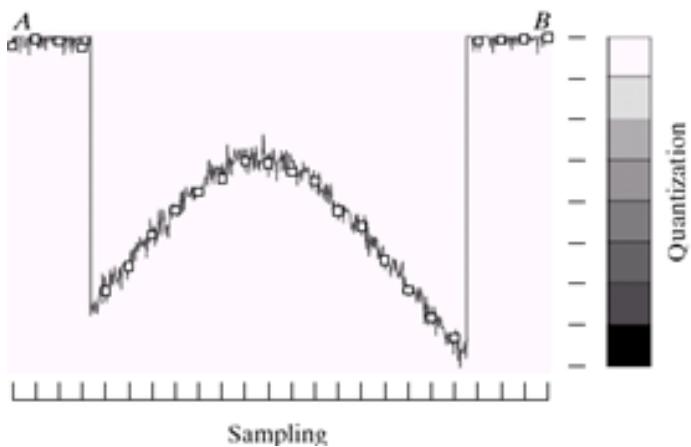
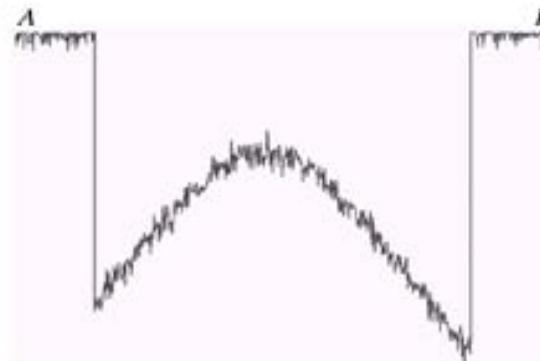
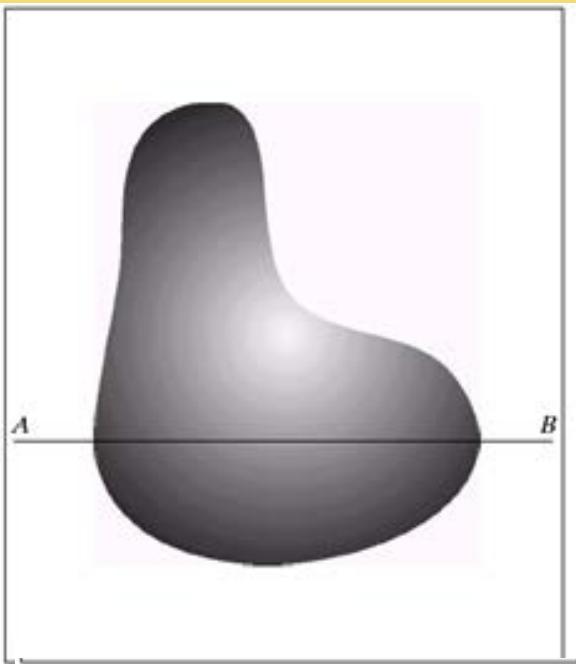
To sample this function, we take equally spaced samples along line AB

Image Digitization



The values of the samples still span a continuous range of intensity values

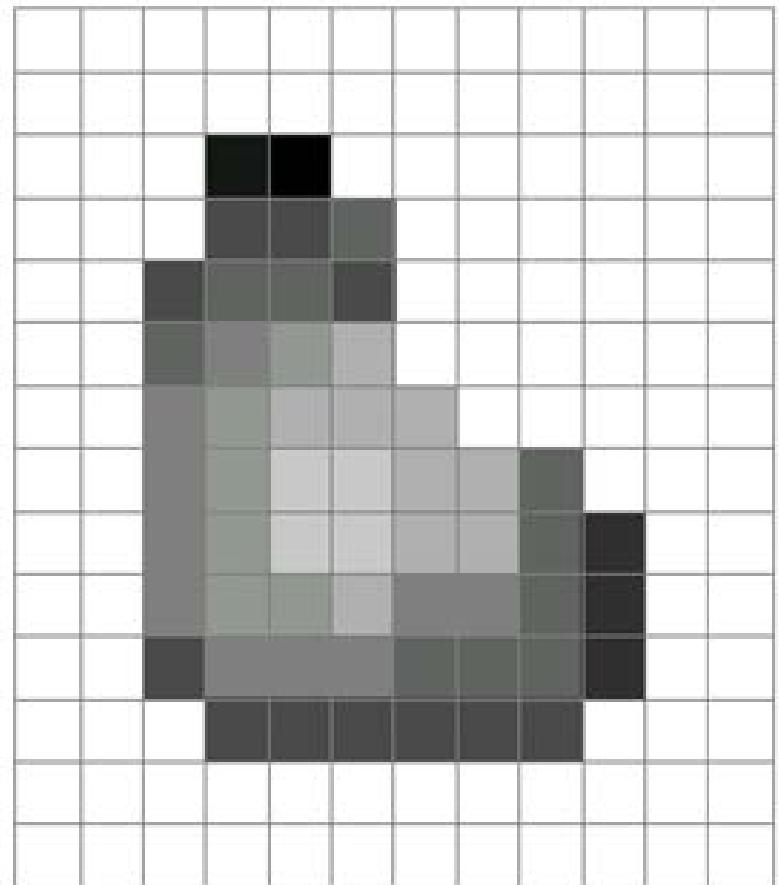
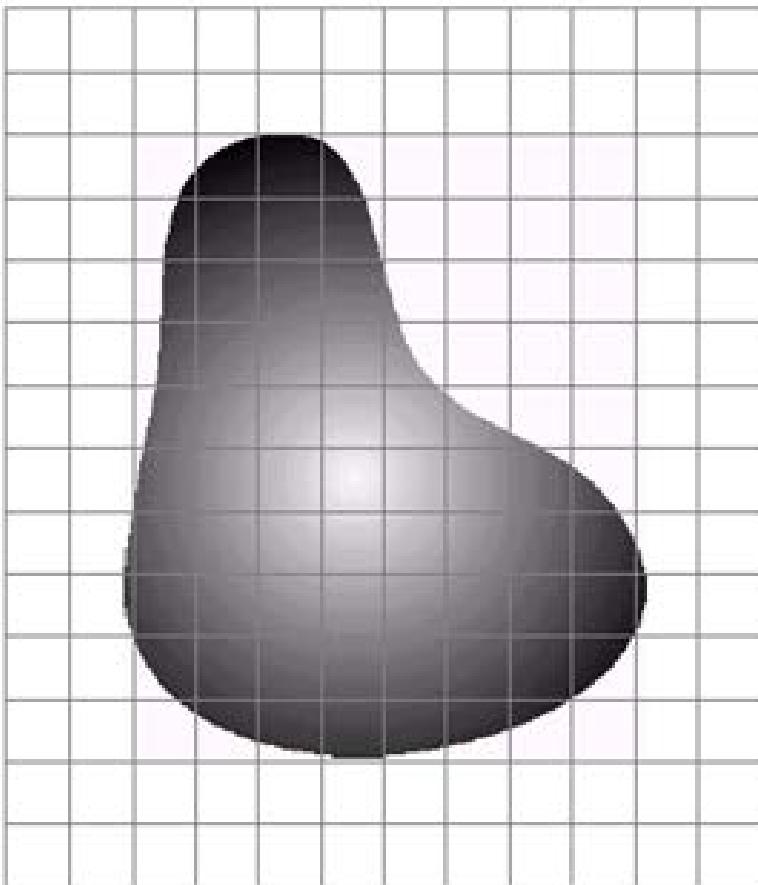
Image Digitization



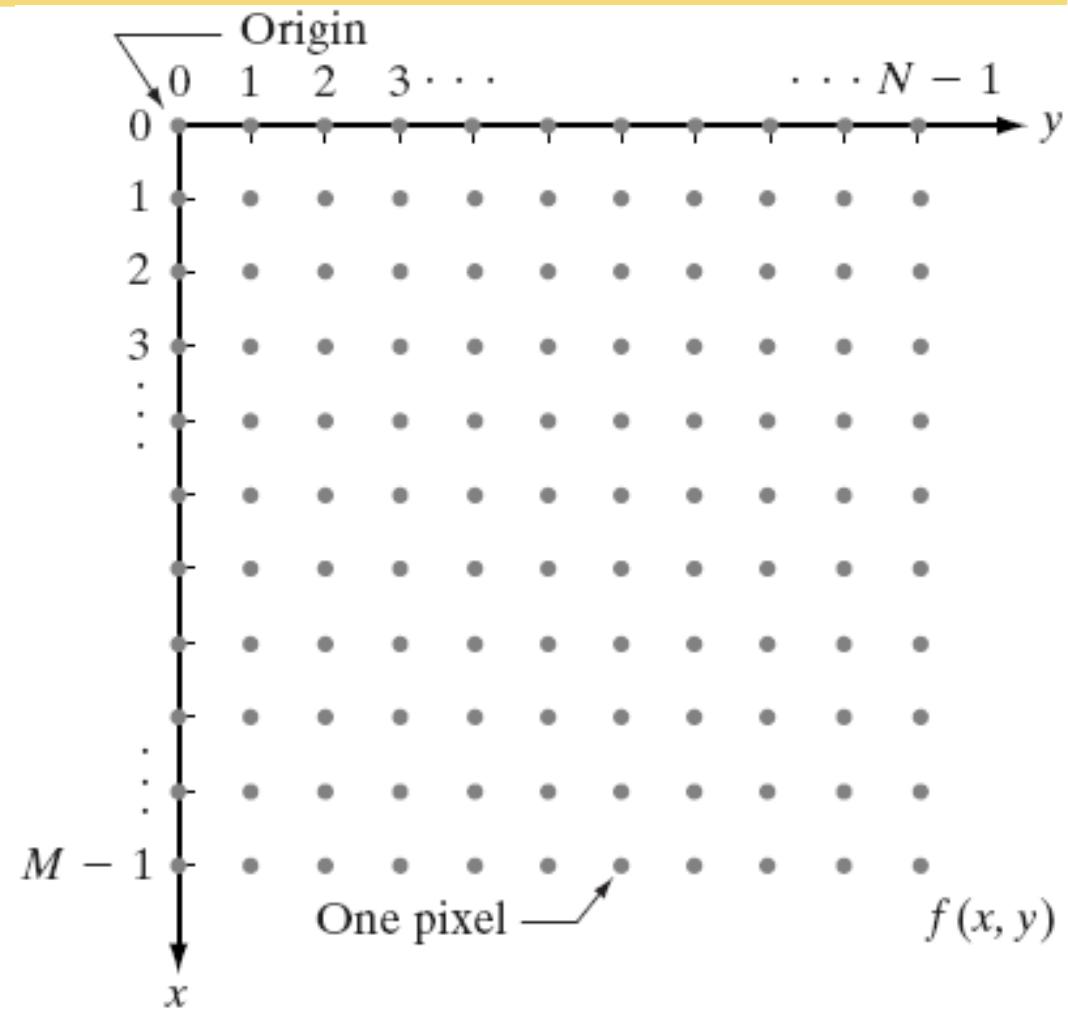
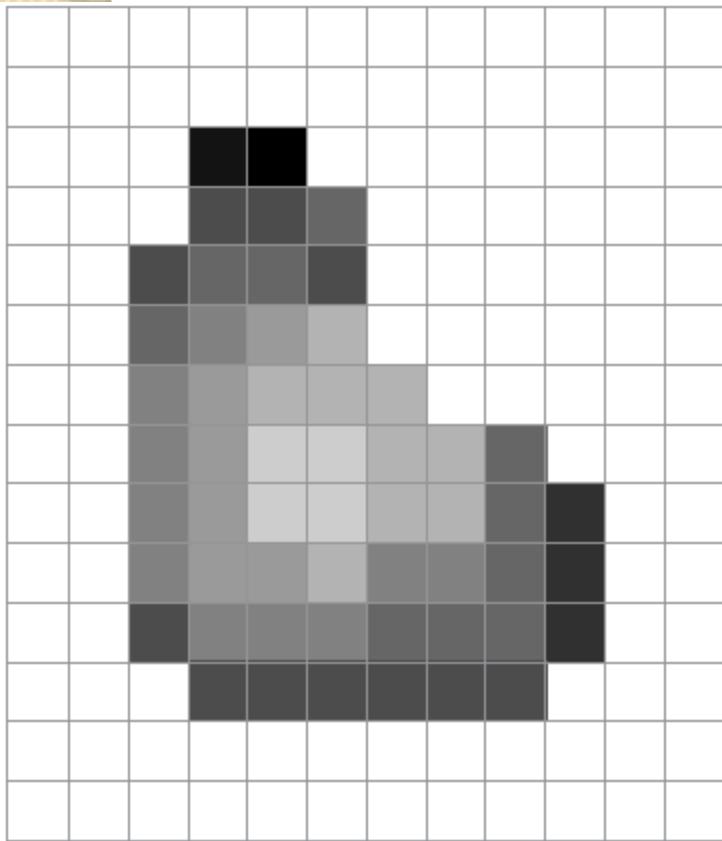
Intensity scale is divided into eight discrete intervals.

Image Digitization

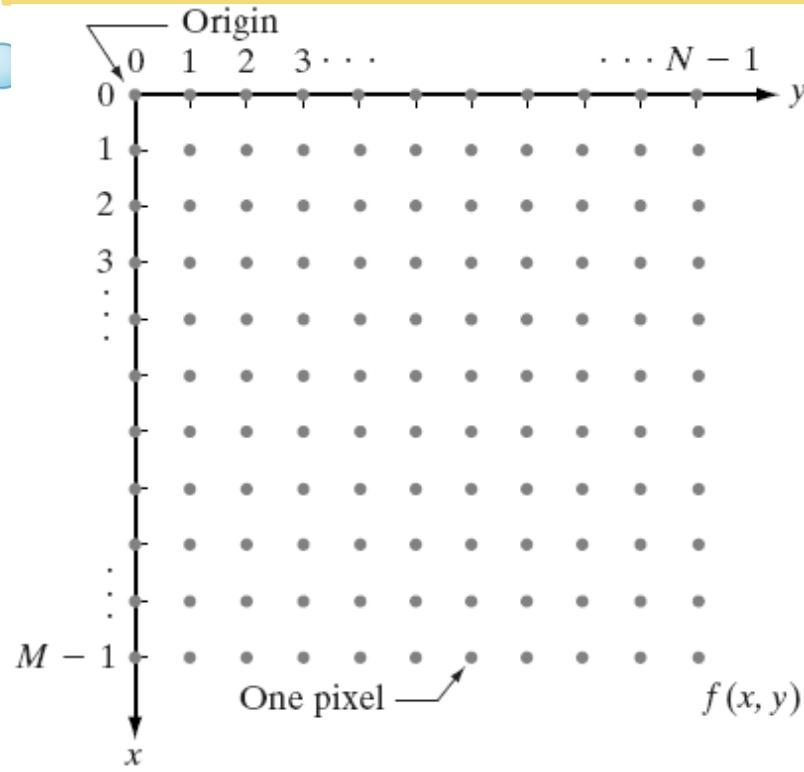
- For Sensor array:



Representing Digital Images



Representing Digital Images

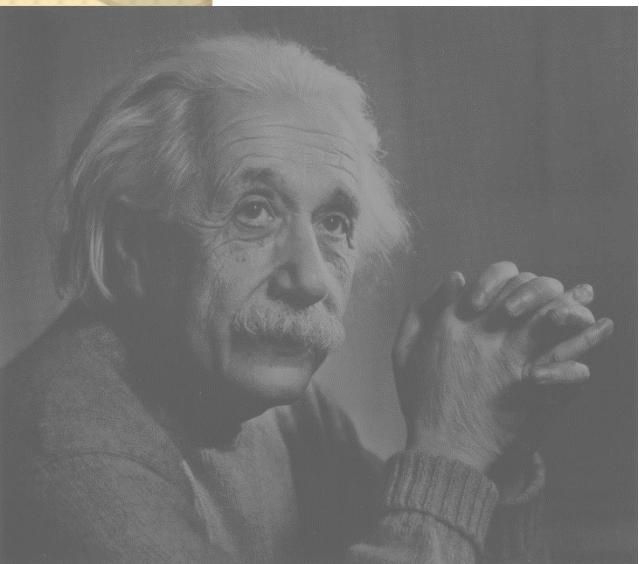


Each element of this matrix is called **pixel** or **pel**

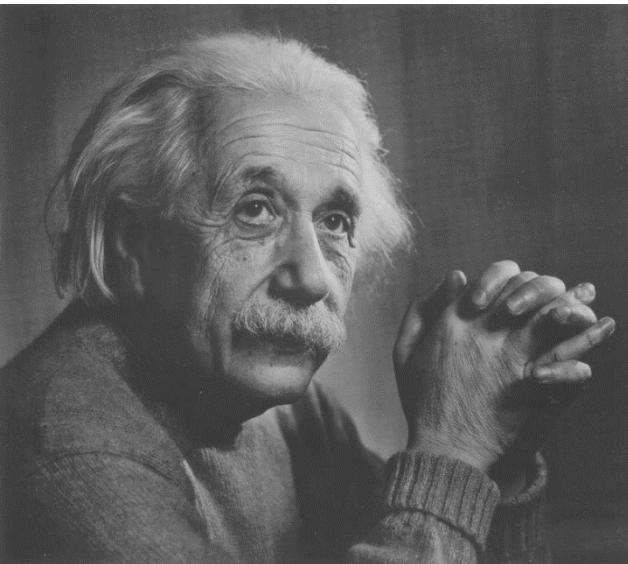
Matrix Representation

$$f(x, y) = \begin{bmatrix} f(0, 0) & f(0, 1) & \cdots & f(0, N - 1) \\ f(1, 0) & f(1, 1) & \cdots & f(1, N - 1) \\ \vdots & \vdots & & \vdots \\ f(M - 1, 0) & f(M - 1, 1) & \cdots & f(M - 1, N - 1) \end{bmatrix}$$

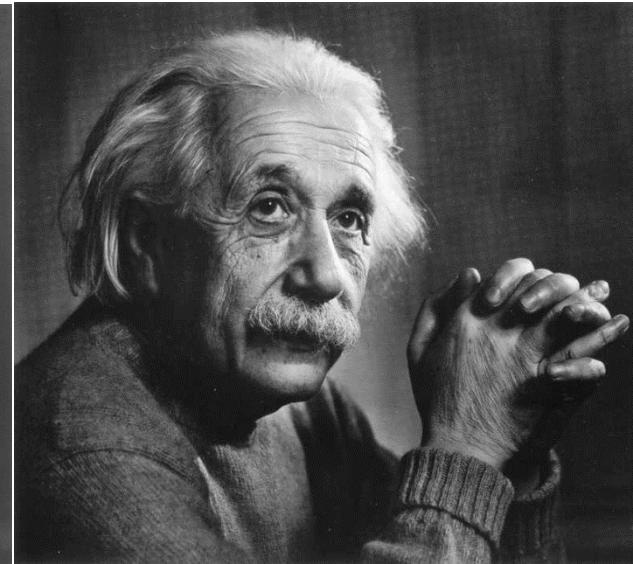
Representing Digital Images



Low contrast



Med contrast



High contrast

Representing Digital Images

- Number of Rows (M) and Columns (N) can be any integer
- How many discrete intensity levels?
 - Number of gray levels, L , is usually power of 2:

$$L = 2^k$$

- The number of bits required to store a digitized image :

$$b = M \times N \times k$$

- For squared image :

$$b = N^2 k$$

Representing Digital Images

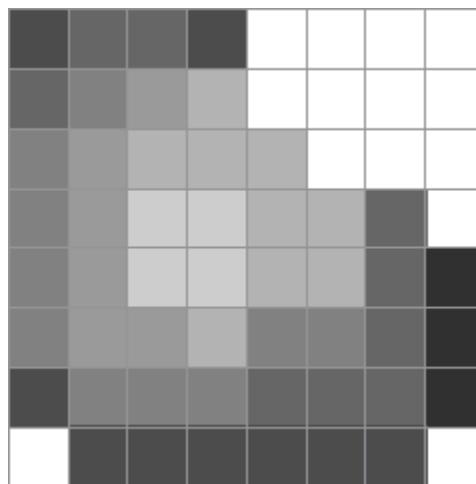
- Suppose, you want to create an image of size 1024×512 having 72 distinct intensity values. What will be the size of the image?

Image Interpolation

Image Interpolation

- 
- **Interpolation** is the process of using known data to estimate values at unknown locations.
 - Zooming (Over Sampling)
 - Shrinking (Under Sampling)

Nearest Neighbor Interpolation

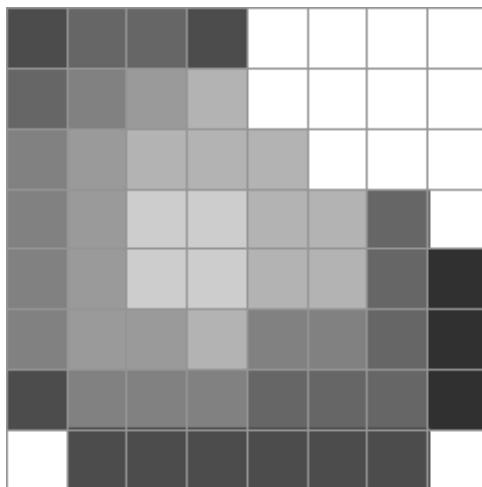


An 8X8 image

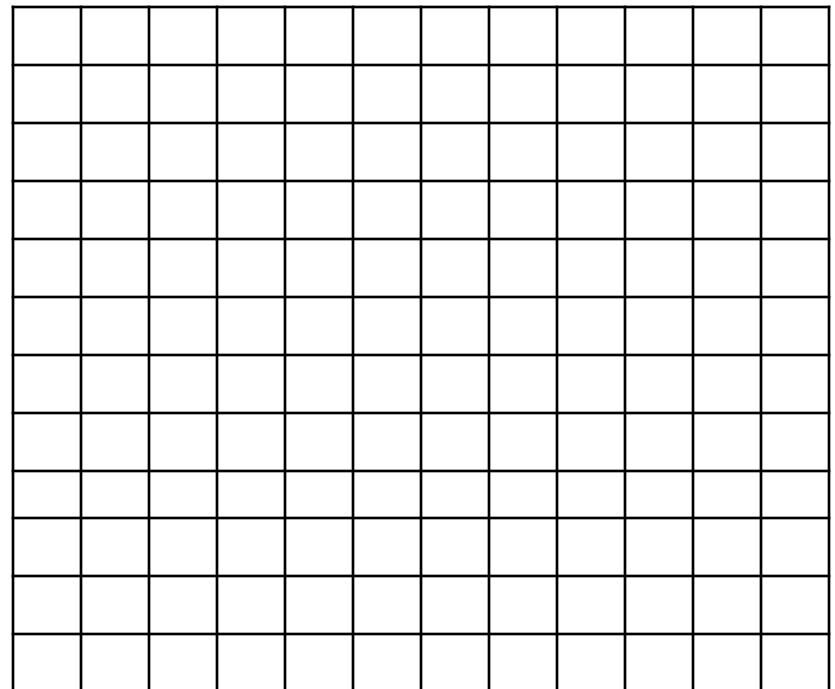
Resize this image to 12X12

Nearest Neighbor Interpolation

- >Create an imaginary 12×12 grid with the same pixel spacing as the original and then shrink it so that it fits exactly over the original image.

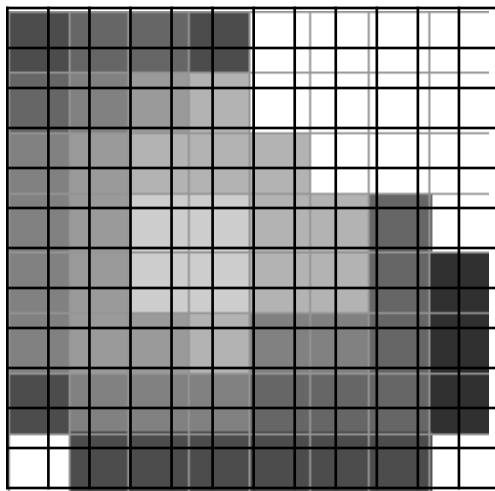


An 8X8 image
Resize this image to 12X12



Nearest Neighbor Interpolation

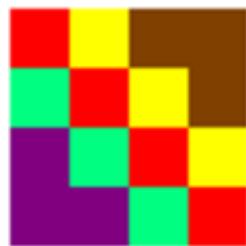
- Create an imaginary 12×12 grid with the same pixel spacing as the original and then shrink it so that it fits exactly over the original image.



- To perform intensity-level assignment for any point in the overlay, we look for its **closest pixel** in the original image and assign the intensity of that pixel to the new pixel in the 12×12 grid.
- This process is also known as pixel replication method.

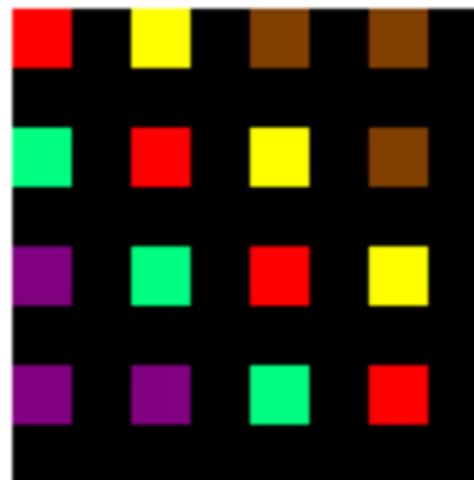
Nearest Neighbor Interpolation

$w_1 - 4 \text{ pixels}$



$h_1 - 4 \text{ pixels}$

$w_2 - 8 \text{ pixels}$



$h_2 - 8 \text{ pixels}$

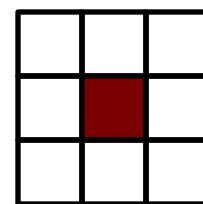
complete



Pixel and their Relationships

Neighbors of a pixel

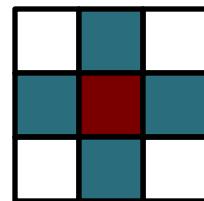
- 3 types of neighbors
 - $N_4(p)$: 4-neighbor
 - $N_D(p)$: diagonal neighbor
 - $N_8(p)$: 8-neighbor



Neighbors of a pixel

- $N_4(p)$: 4-neighbor

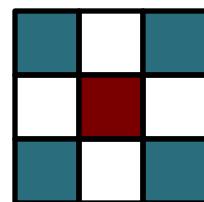
Defined as the pixels at $(x+1, y)$, $(x-1, y)$, $(x, y+1)$,
 $(x, y-1)$



Neighbors of a pixel

- $N_D(p)$: diagonal neighbor

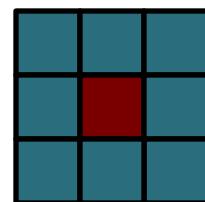
Defined as the pixels at $(x+l, y+l)$, $(x+l, y-l)$, $(x-l, y+l)$, $(x-l, y-l)$



Neighbors of a pixel

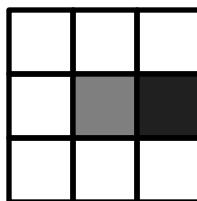
- $N_8(p)$: 8-neighbor

Defined as the pixels at $(x+1, y)$, $(x-1, y)$, $(x, y+1)$,
 $(x, y-1)$, $(x+1, y+1)$, $(x+1, y-1)$, $(x-1, y+1)$, $(x-1, y-1)$



Pixel Adjacency

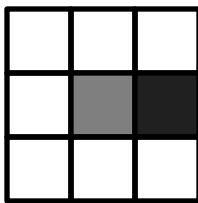
- Adjacencies depends on both
 - Neighborhood
 - Pixel gray values



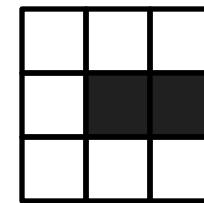
Adjacent pixels must be neighbors and have gray values from the same set, V

Pixel Adjacency

Adjacent pixels must be neighbors and have gray values from the same set, V



Can not be adjacent
if both values are
not present in V



Can be adjacent

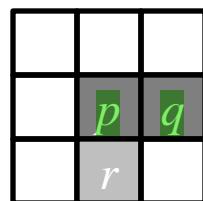
Pixel Adjacency

- 3 types of adjacencies
 - 4-adjacency
 - 8-adjacency
 - m -adjacency

4-adjacency

Two pixels p, q are 4-adjacent if

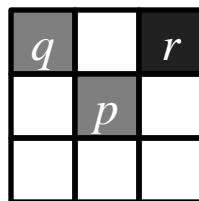
- $q \in N_4(p)$ and
- p, q have values from set V



8-adjacency

Two pixels p, q are 8-adjacent if

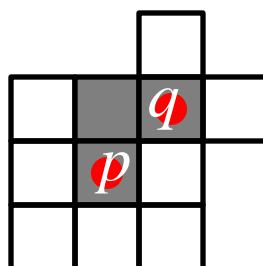
- $q \in N_8(p)$ and
- p, q have values from set V



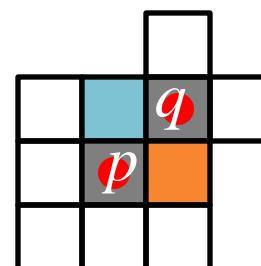
m-adjacency

Two pixels p, q are m-adjacent if

- p, q have values from set V and
 - q in $N_4(p)$, or
 - q in $N_D(p)$ and $N_4(p) \cap N_4(q)$ has no pixel with value from V



Not m-adjacent



m-adjacent



THANK YOU!!!

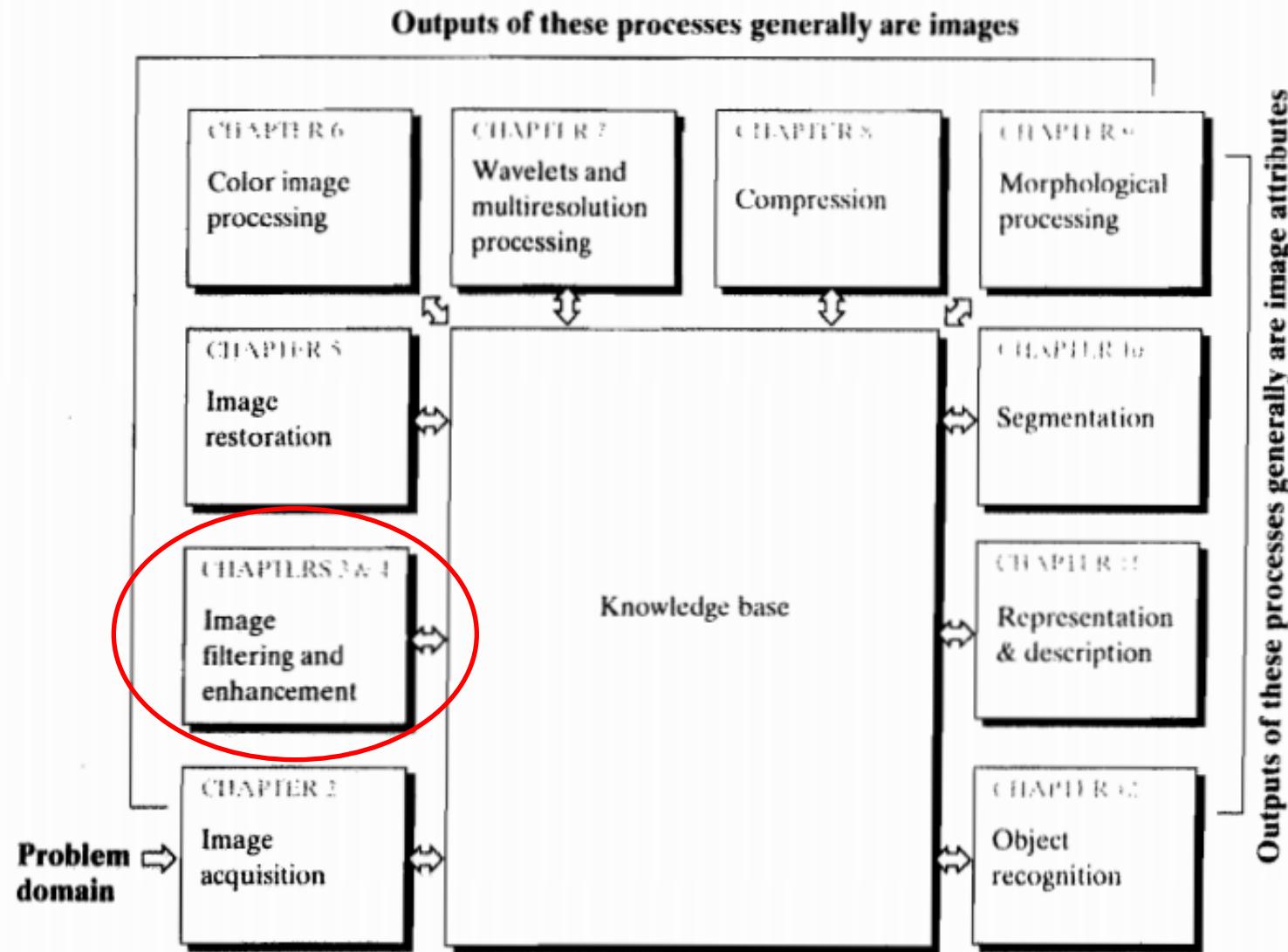


Chapter 3

Intensity Transformations and Spatial Filtering

Remember?

FIGURE 1.23
Fundamental steps in digital image processing. The chapter(s) indicated in the boxes is where the material described in the box is discussed.



Outputs of these processes generally are image attributes

Our Objective

Image Enhancement

- Enhancement is the process of manipulating an image so that the result is **more suitable** than the original for a **specific** application.
- Very subjective
- Application dependent
- Viewer is the ultimate judge

Classification of Image Enhancement

- Enhancement in spatial domain (Chapter 3)

Spatial domain is a plane where a digital image is defined by the spatial coordinates of its pixels.

- Enhancement in frequency domain (Chapter 4)

Frequency domain where a digital image is defined by its decomposition into spatial frequencies participating in its formation.

Classification of Image Enhancement

➤ Enhancement in spatial domain

- The section of the **real plane** spanned by the coordinates of an image is called the spatial domain.
- manipulates pixel by pixel basis

How does It Work?

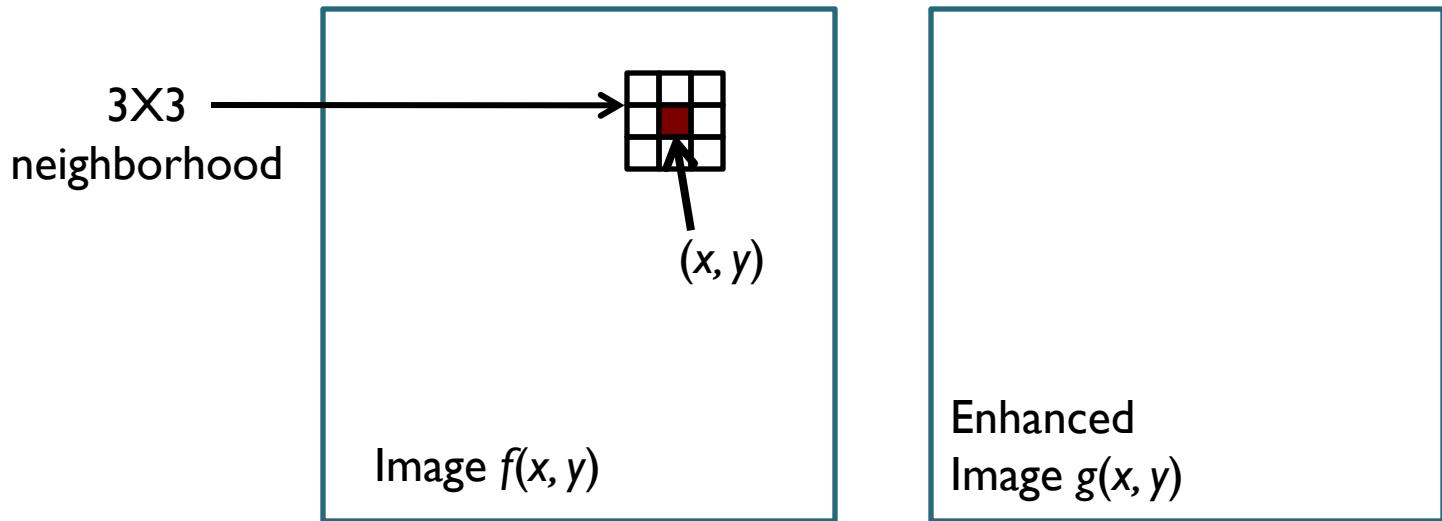
Enhancement in Spatial Domain

- The spatial domain processes can be denoted by the expression

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

- $f(x, y)$ is the **input image** and $g(x, y)$ is the **output image** and T is an operator defined over a neighborhood of point (x, y)

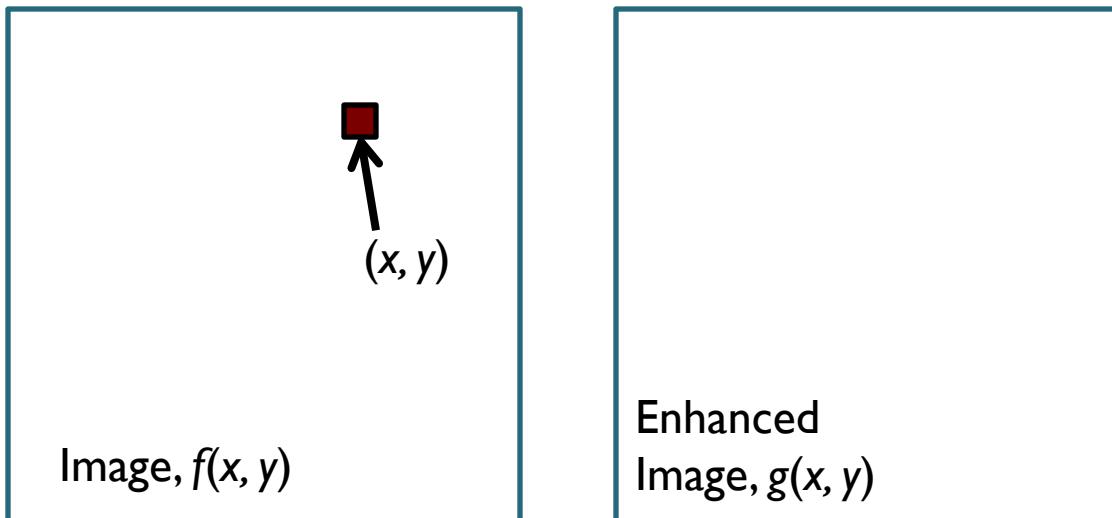
Enhancement in Spatial Domain



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

Enhancement in Spatial Domain

- Neighborhood can be as small as a single pixel



Enhancement in Spatial Domain

- Neighborhood (sub-image) rolls over the entire image, each time centering a different pixel.
- For any specific location (x, y) , the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) .



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

Enhancement in Spatial Domain

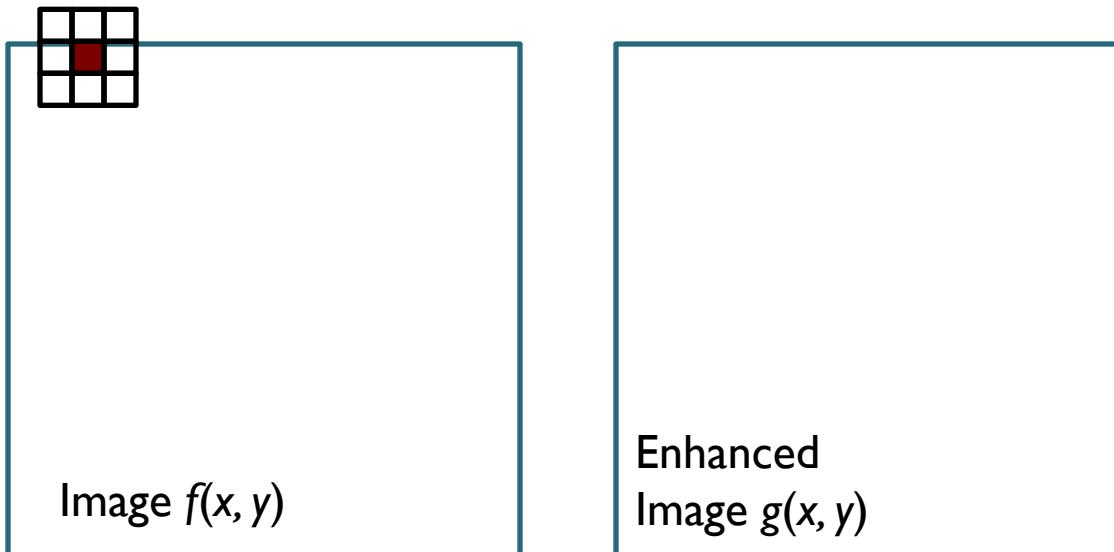
- Neighborhood (sub-image) rolls over the entire image, each time centering a different pixel.
- For any specific location (x, y) , the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) .



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

Enhancement in Spatial Domain

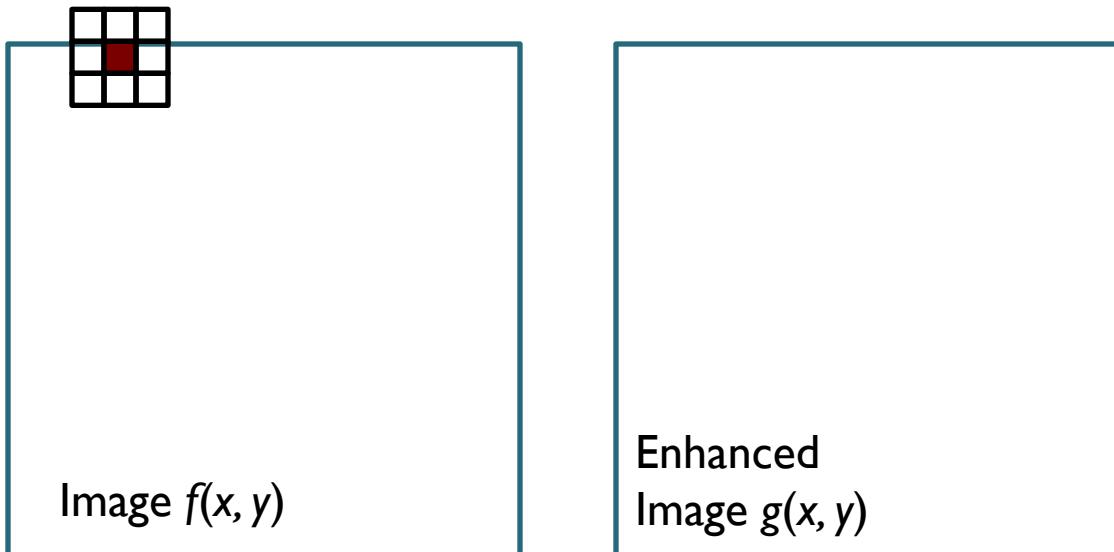
- Neighborhood (sub-image) rolls over the entire image, each time centering a different pixel.
- For any specific location (x, y) , the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) .



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

Enhancement in Spatial Domain

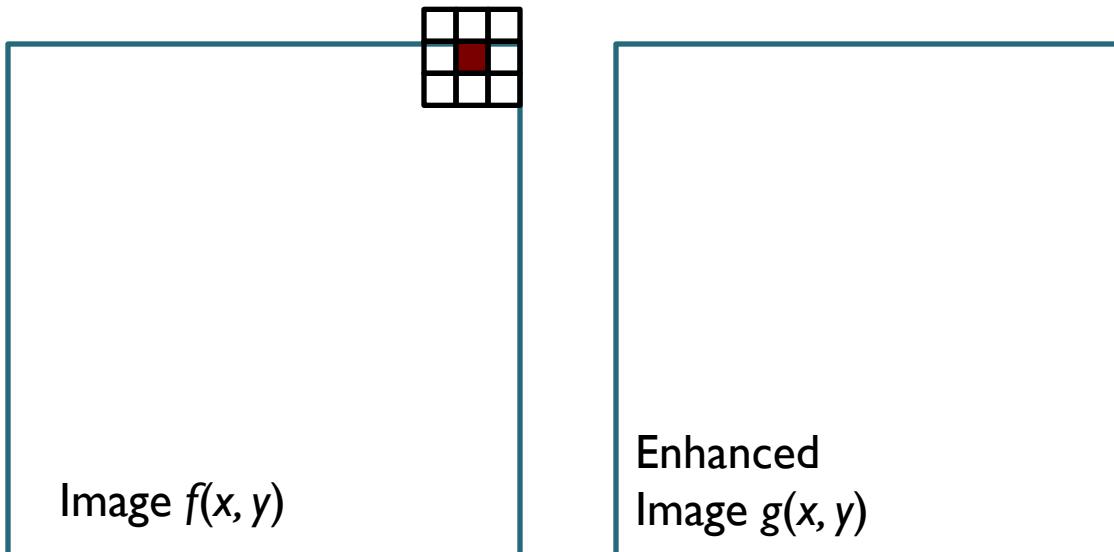
- Neighborhood (sub-image) rolls over the entire image, each time centering a different pixel.
- For any specific location (x, y) , the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) .



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

Enhancement in Spatial Domain

- Neighborhood (sub-image) rolls over the entire image, each time centering a different pixel.
- For any specific location (x, y) , the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) .



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

Enhancement in Spatial Domain

- Neighborhood (sub-image) rolls over the entire image, each time centering a different pixel.
- For any specific location (x, y) , the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) .

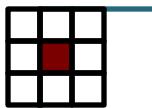
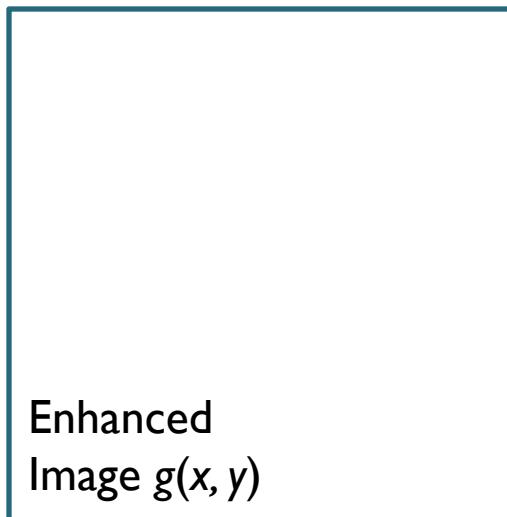


Image $f(x, y)$



Enhanced
Image $g(x, y)$

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

Enhancement in Spatial Domain

- Neighborhood (sub-image) rolls over the entire image, each time centering a different pixel.
- For any specific location (x, y) , the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) .



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

Spatial Filtering

- This procedure is called **Spatial Filtering**.
- The **neighborhood** along with a predefined **operation** is called a **spatial filter**.
- Spatial filter is also known as **spatial mask**, **kernel**, **template** or **window**. Spatial mask is the most popular.

Intensity Transformation Function

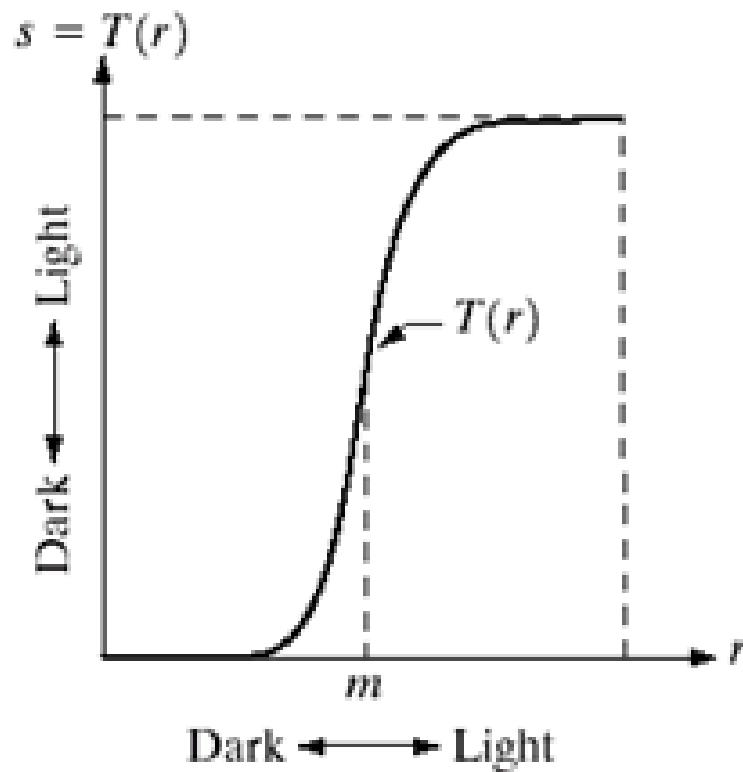
Intensity Transformation Function

- If the neighborhood is of size 1×1 , g depends only on the value of f at a single point (x, y) and T becomes intensity (gray-level) transformation function of the form

$$s = T(r)$$

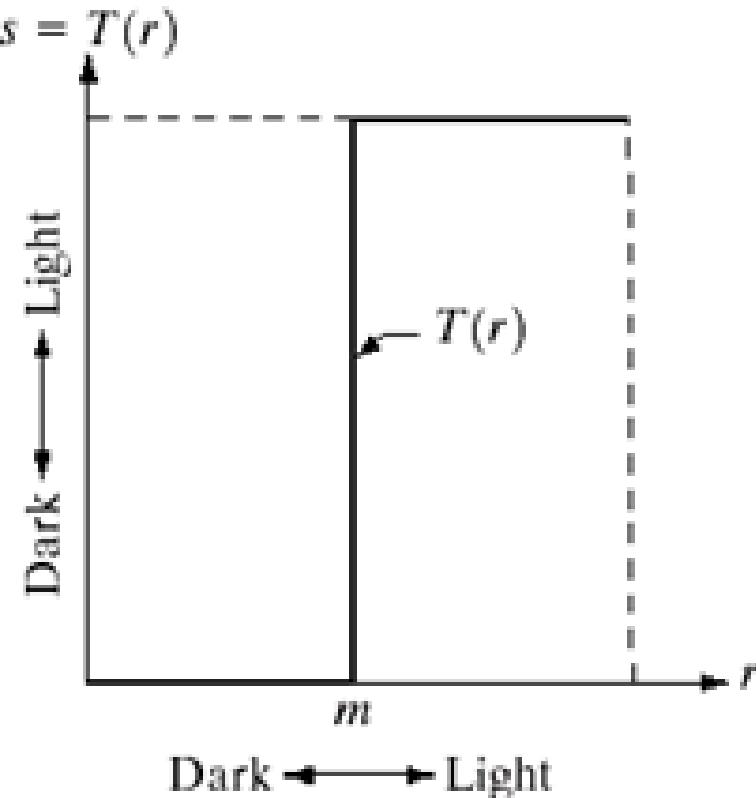
Contrast Stretching

- Input gray level R
- Produce an image of higher contrast than the original.
- Below a threshold (m), image is compressed towards black
- Values of r lower than m are compressed by the transformation function into a narrow range of s , toward black.
- Above m , image is stretched towards white

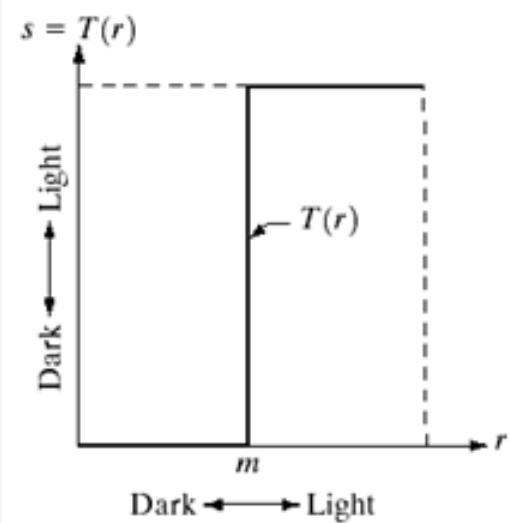


Thresholding

- Thresholding is a special case of contrast stretching.
- Produces a binary image
- Useful in image segmentation



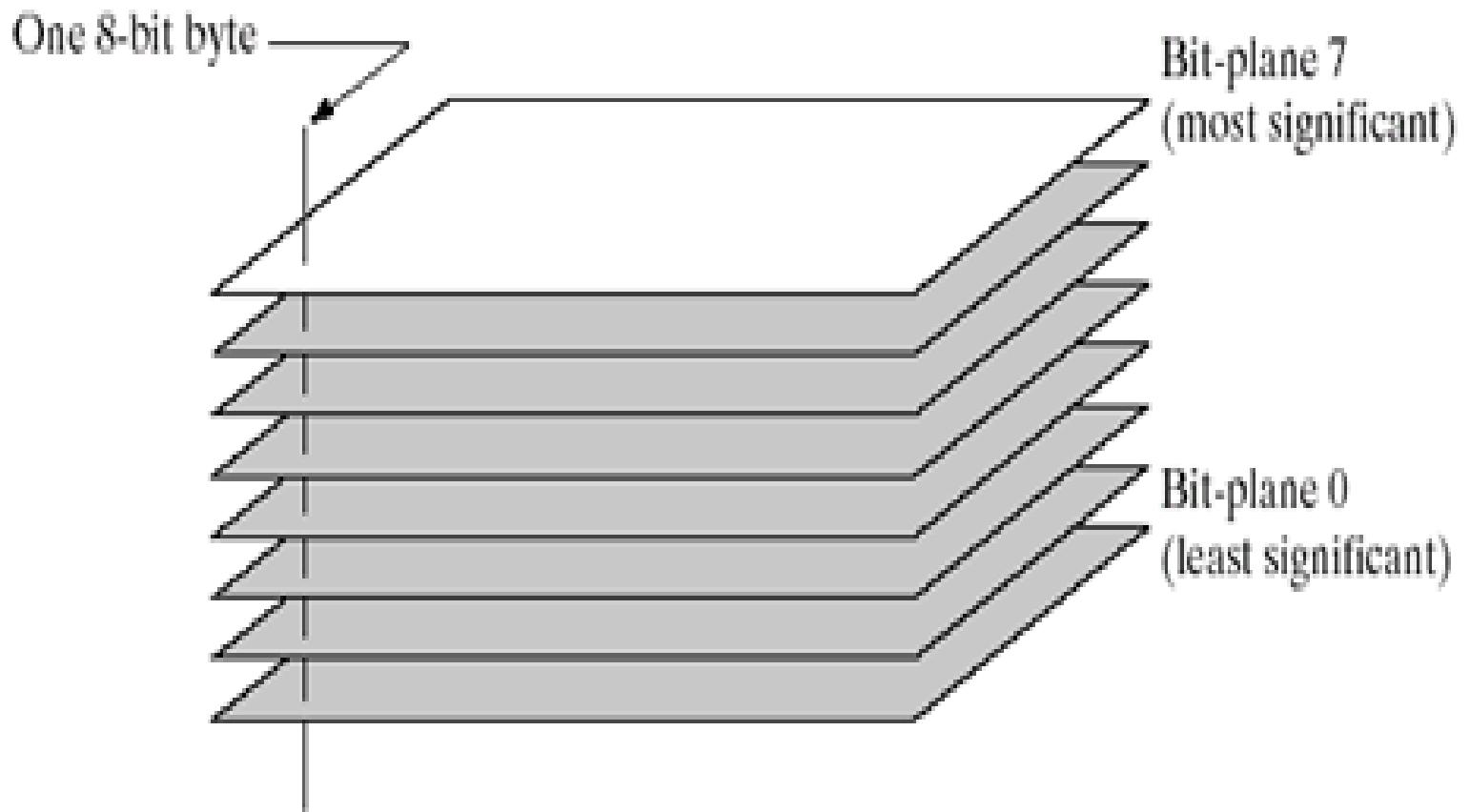
Thresholding



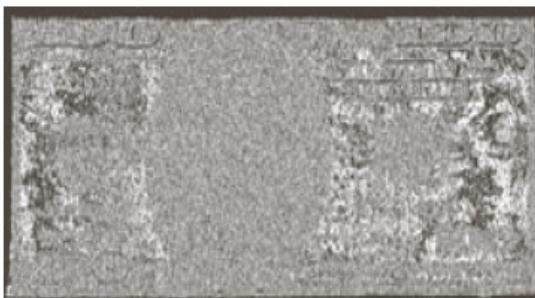
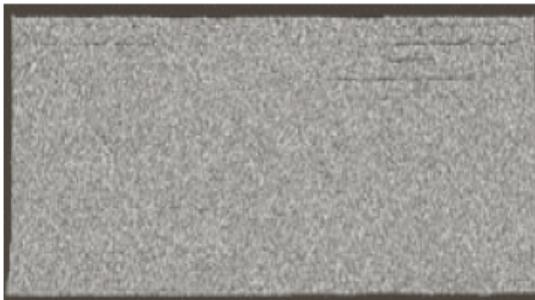
Bit Plane Slicing

What is this all about ???

Bit-plane Slicing



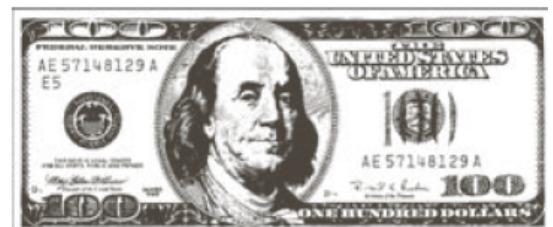
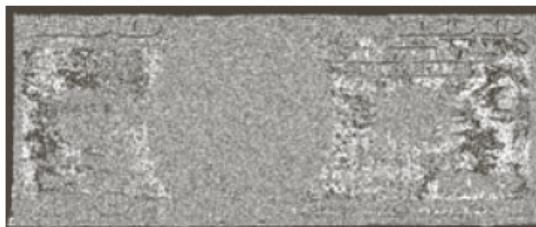
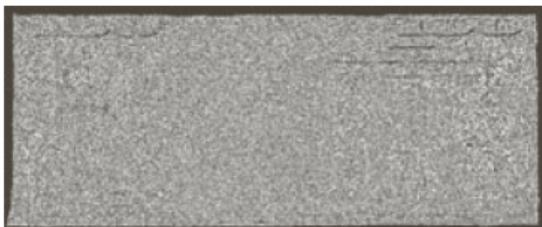
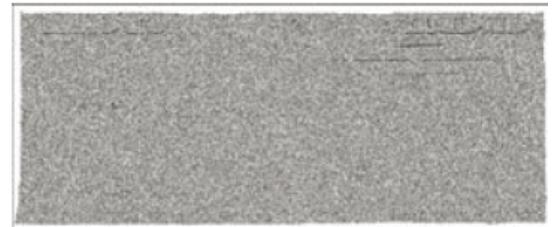
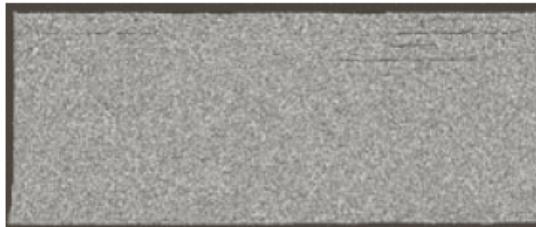
Bit-plane Slicing



Bit-plane Slicing

- The four higher order bit, especially the last two, contain a significant amount of visually significant data.
- The lower-order planes contribute to more subtle intensity details in the image.

original
image

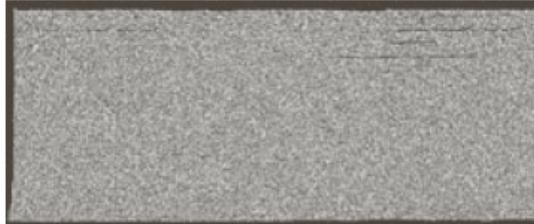


Bit-plane Slicing

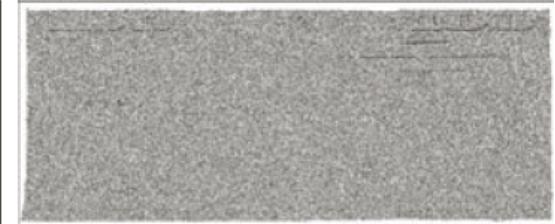
Look at this gray border.
Its decimal value is 194



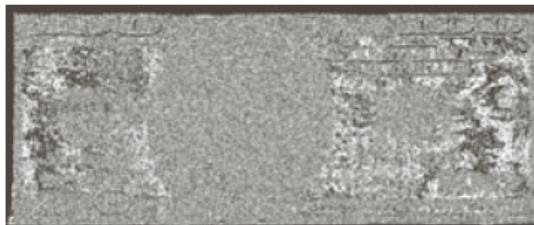
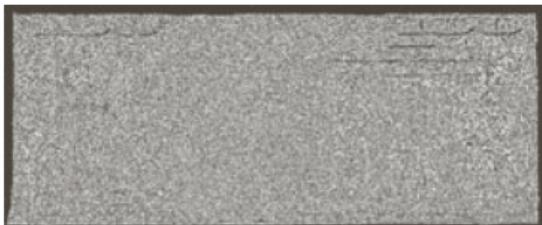
Some of the bit plane sliced images have **black** border.



Some of the bit plane sliced images have **white** border.



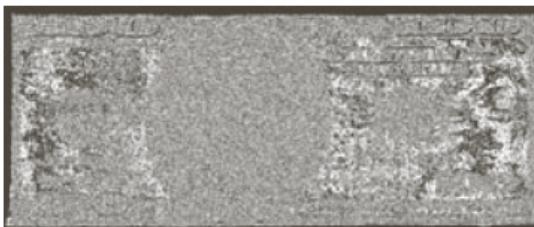
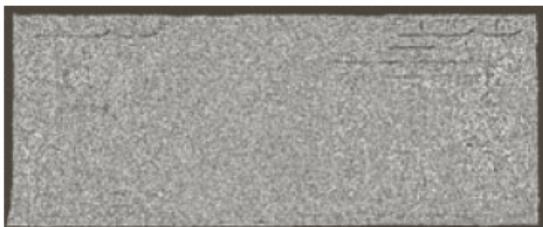
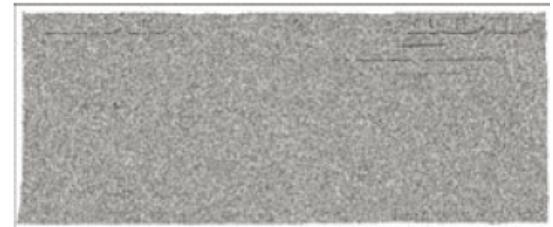
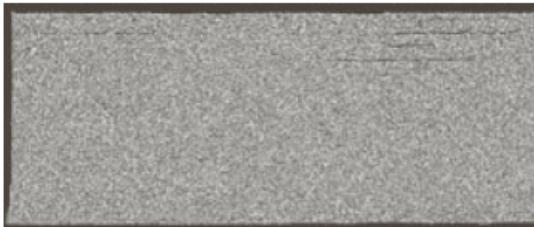
original
image



Bit-plane Slicing

Binary of 194 is 11000010

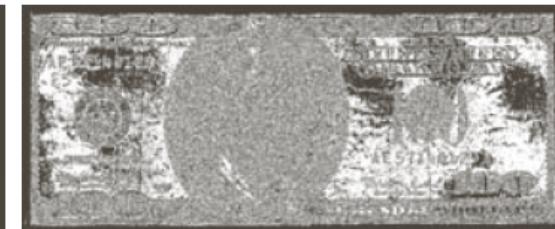
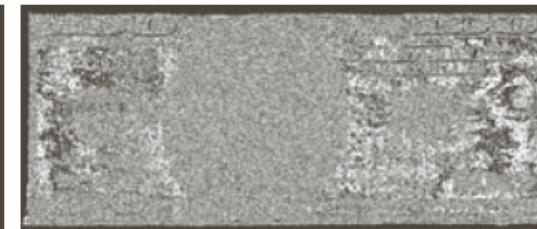
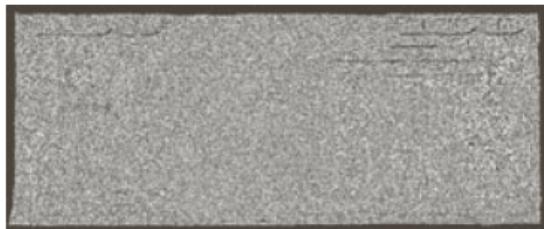
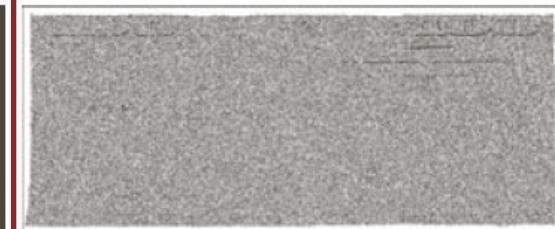
original
image



Bit-plane Slicing

Binary of 194 is 11000010

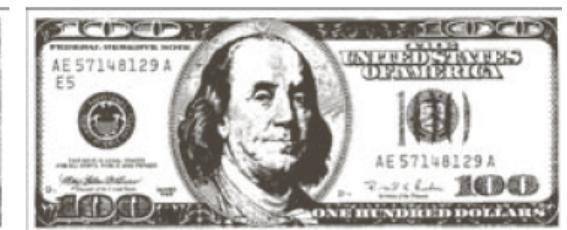
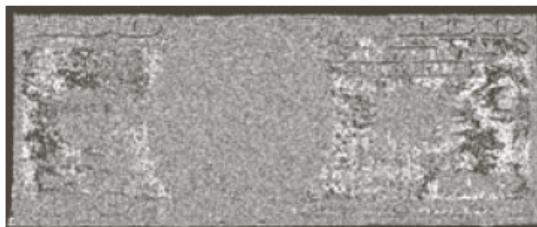
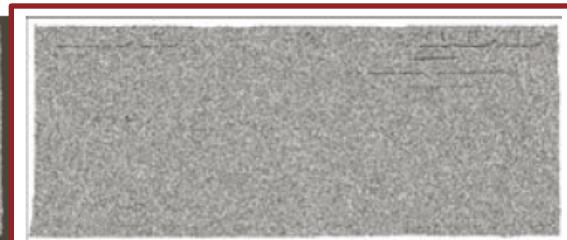
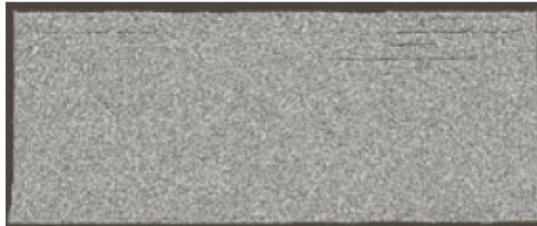
original
image



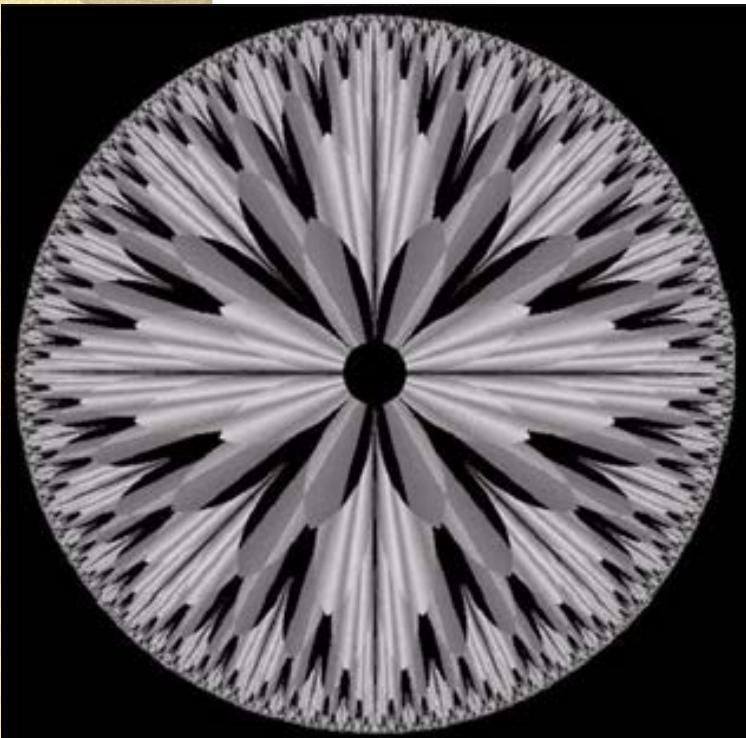
Bit-plane Slicing

Binary of 194 is 11000010

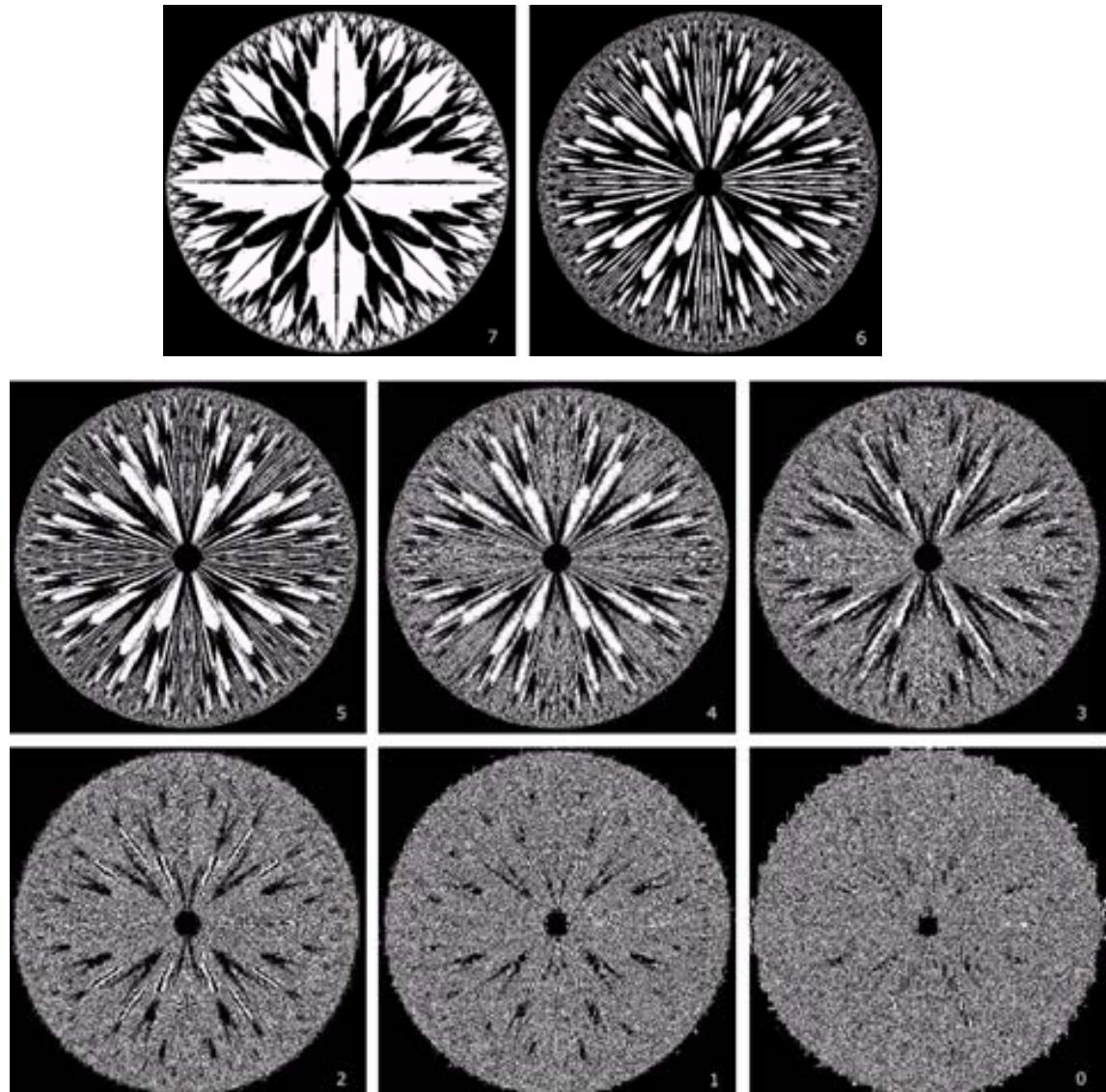
original
image



Bit-plane Slicing



Original Image



Bit-plane Slicing

- Useful for analyzing the relative importance of each bit in an image.
- This process aids in determining the adequacy of the number of bits used to quantize the image.
- Useful in image compression.

Bit-plane Slicing

Image Compression

Bit planes 8 & 7



Bit planes 8, 7, 6 & 5



Bit planes 8, 7 & 6



Original Image

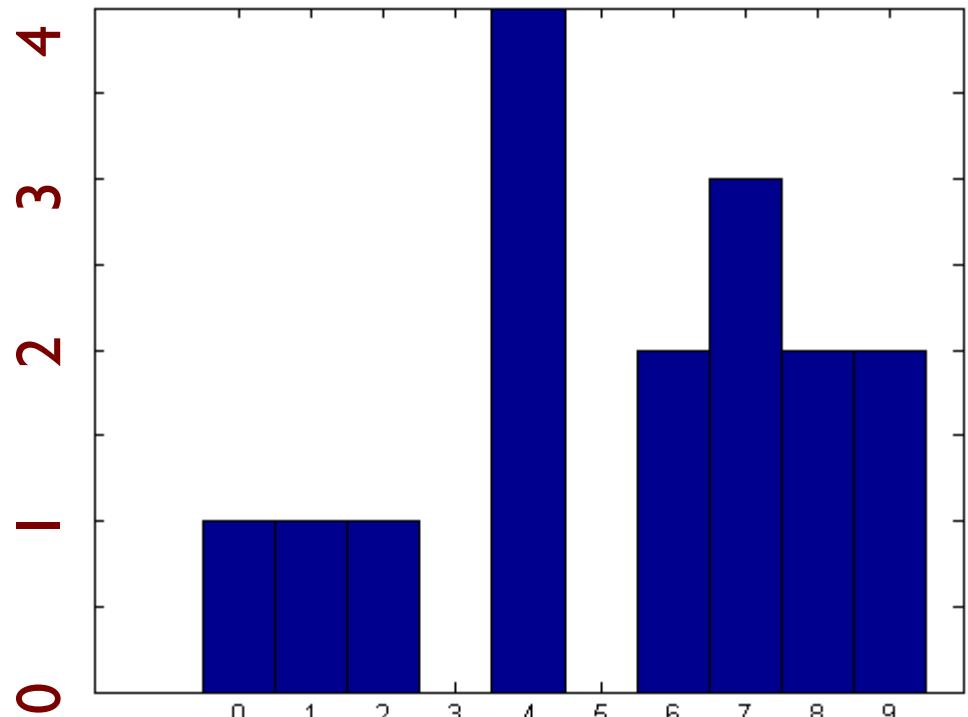


Image Enhancement using Histogram Processing

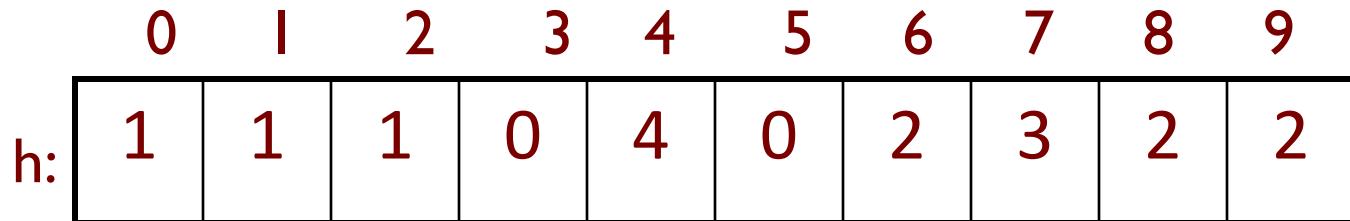
Image Histogram

9	8	9	8
2	7	4	7
6	4	6	1
4	0	7	4

An image



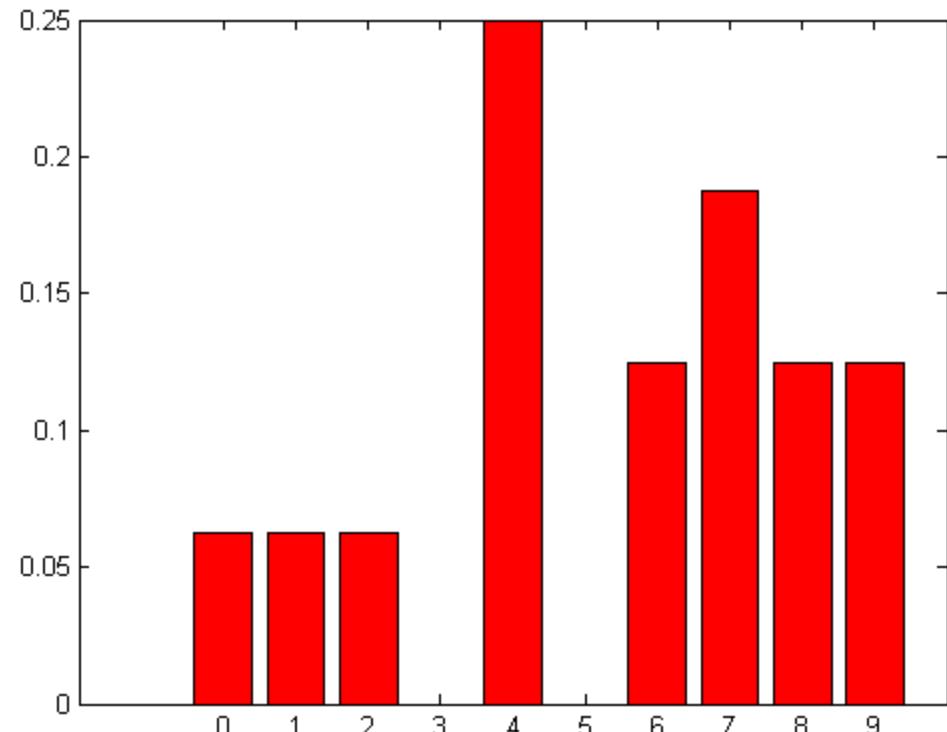
Histogram:



Normalized Image Histogram

9	8	9	8
2	7	4	7
6	4	6	1
4	0	7	4

An image



Histogram:

0 | 2 3 4 5 6 7 8 9

h:	?	?	?	0	?	0	?	?	?	?
----	---	---	---	---	---	---	---	---	---	---

Normalized Image Histogram

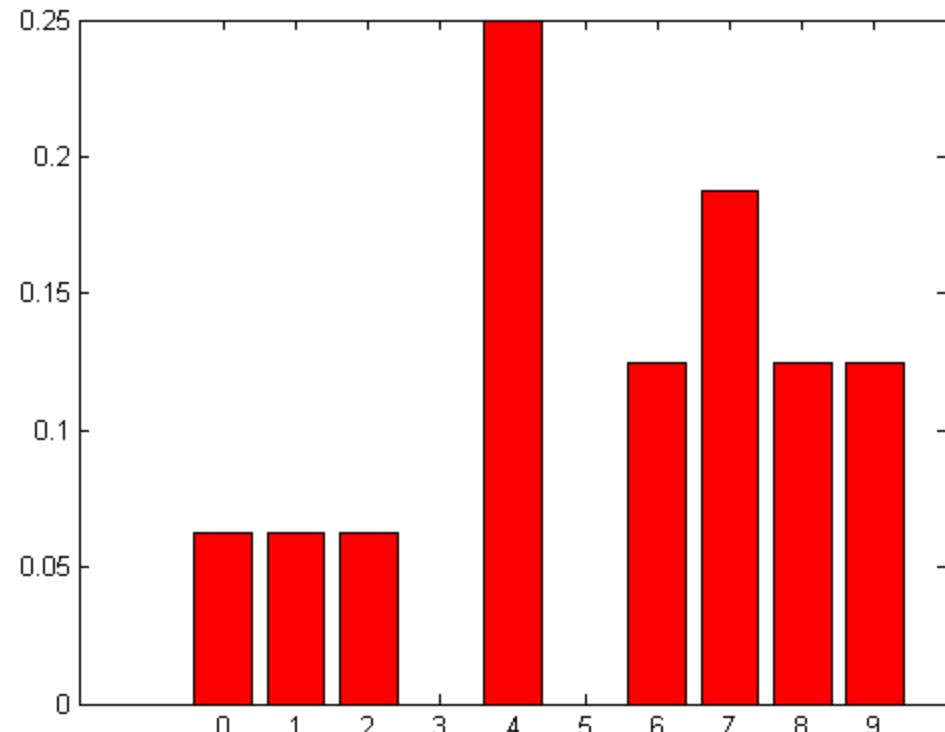
$$p(r_k) = n_k / n$$

- Normalized histogram is more like probabilities
- $p(r_k)$: how likely a pixel will have gray level r_k

Normalized Image Histogram

9	8	9	8
2	7	4	7
6	4	6	1
4	0	7	4

An image

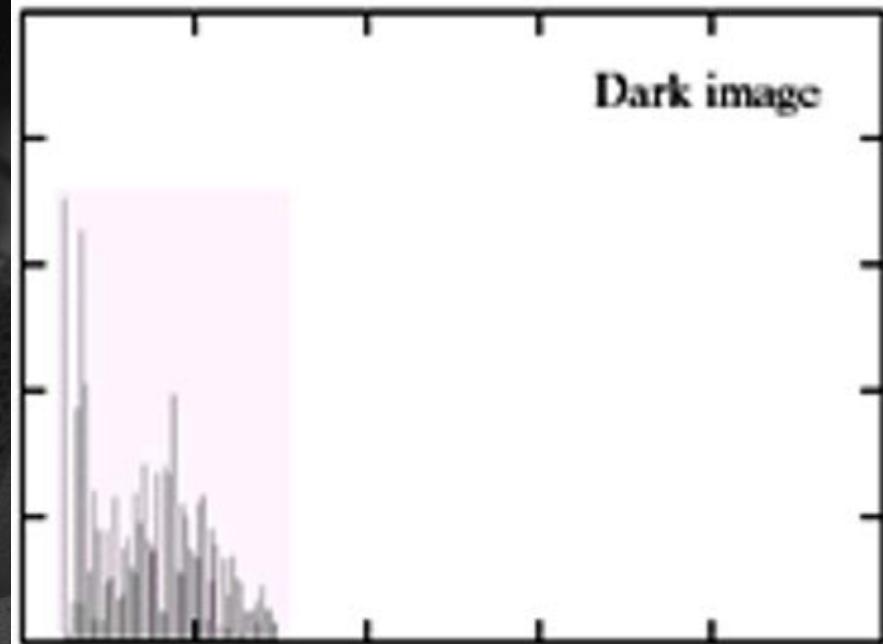
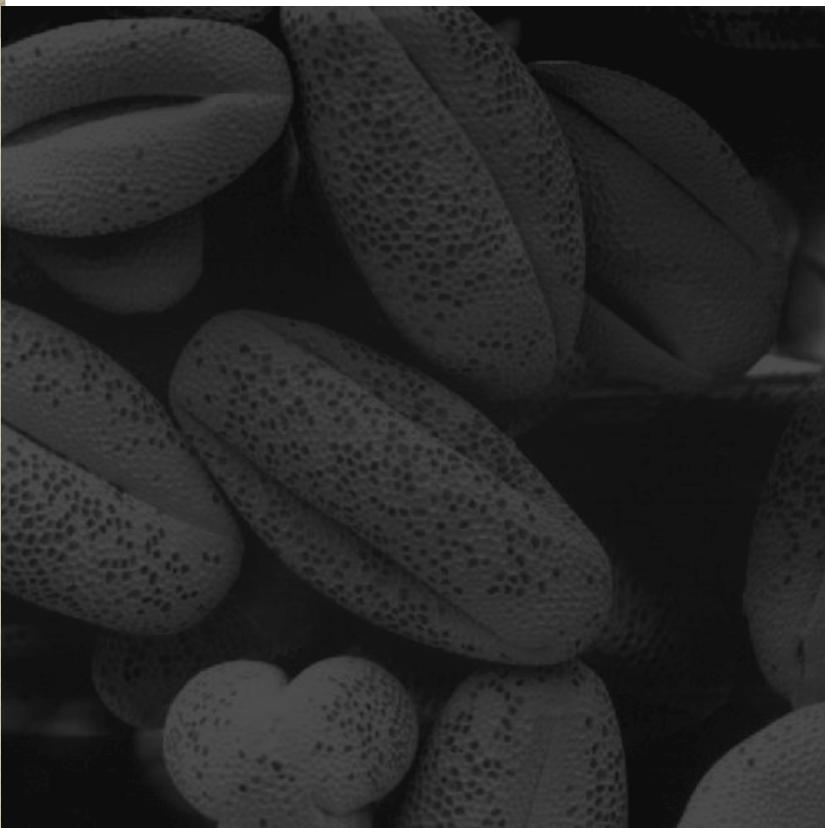


Histogram:

0 1 2 3 4 5 6 7 8 9

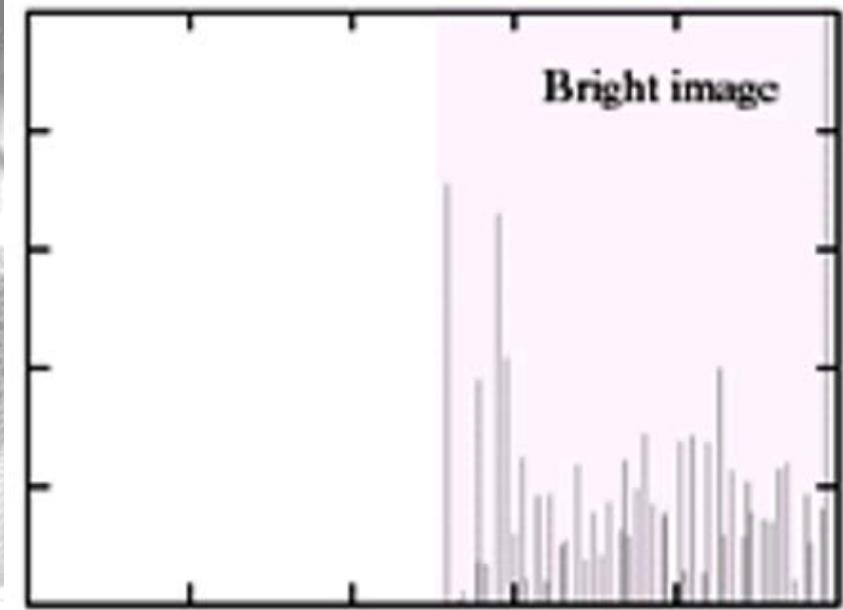
h:	1/16	1/16	1/16	0	1/4	0	1/8	3/16	1/8	1/8
----	------	------	------	---	-----	---	-----	------	-----	-----

Image Histogram



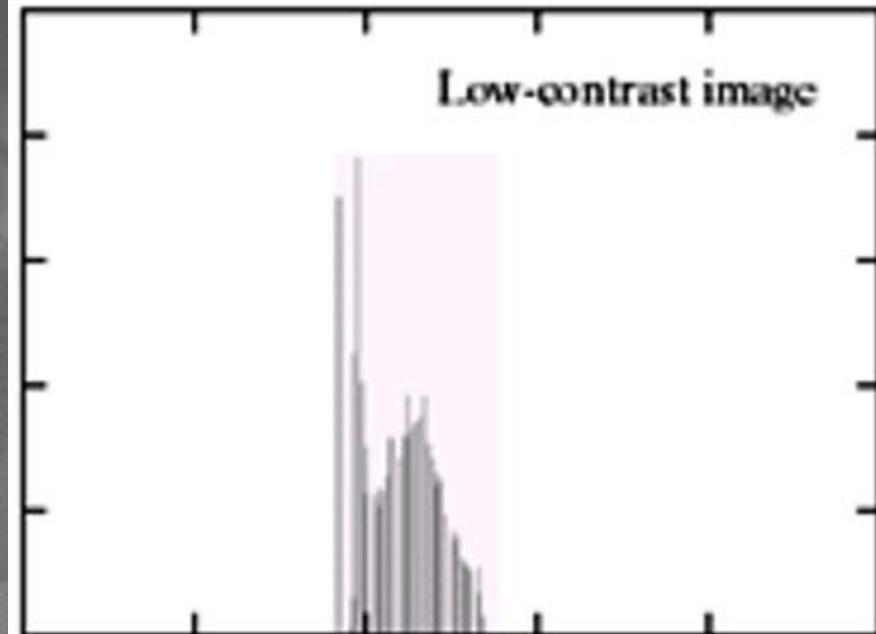
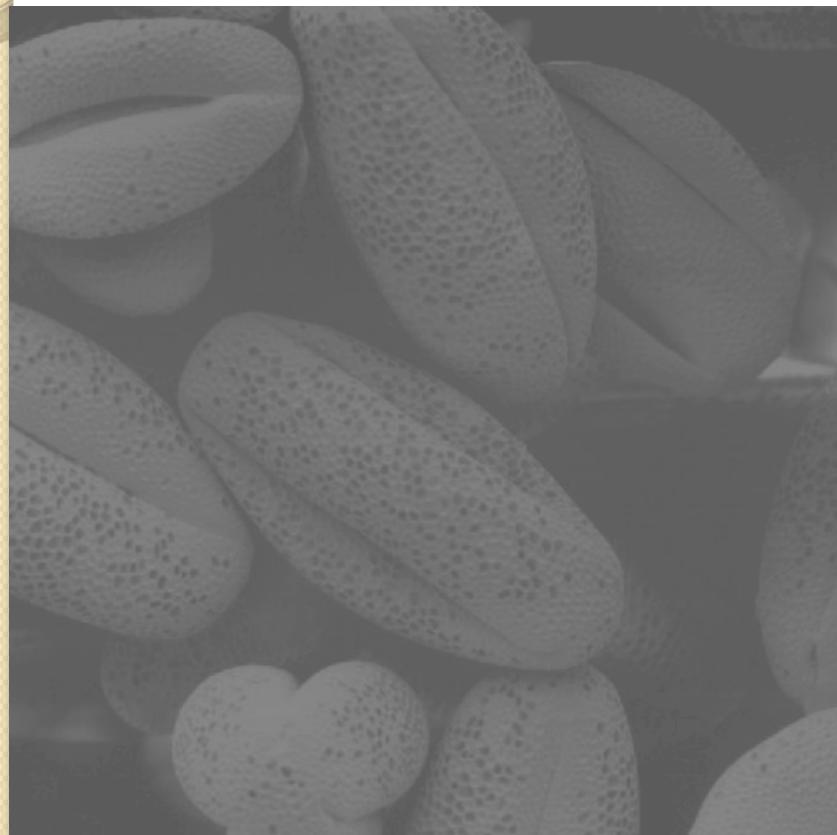
Histogram for dark image

Image Histogram



Histogram for bright image

Image Histogram



Histogram for low
contrast image

Image Histogram

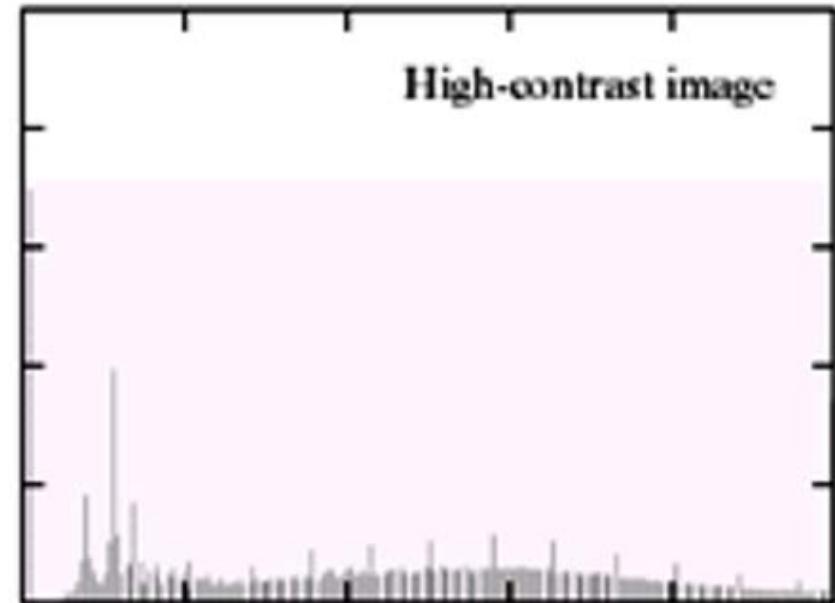
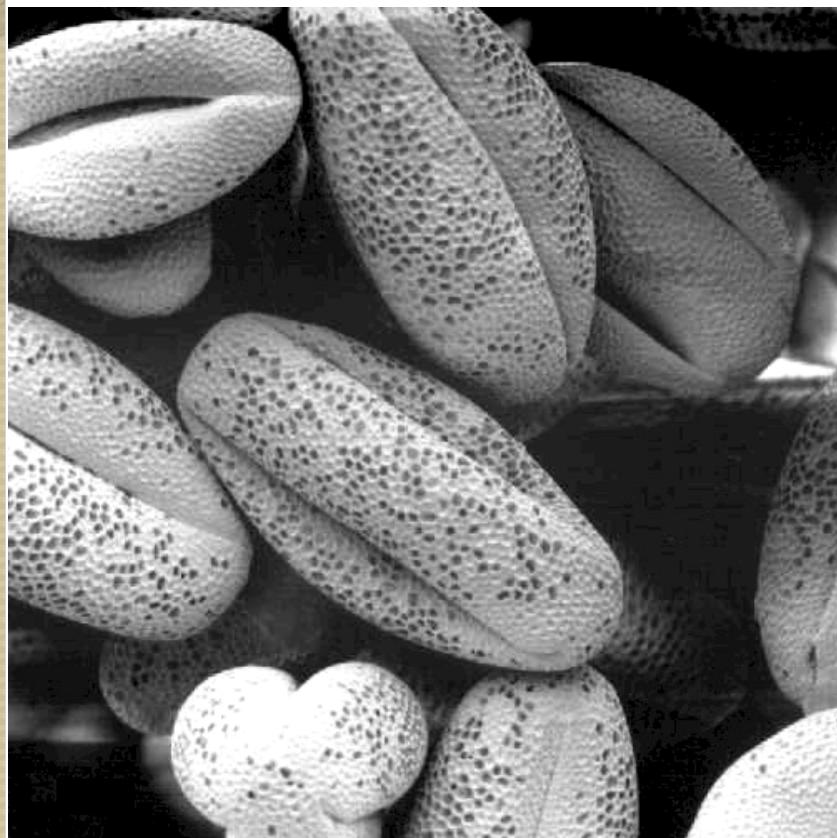
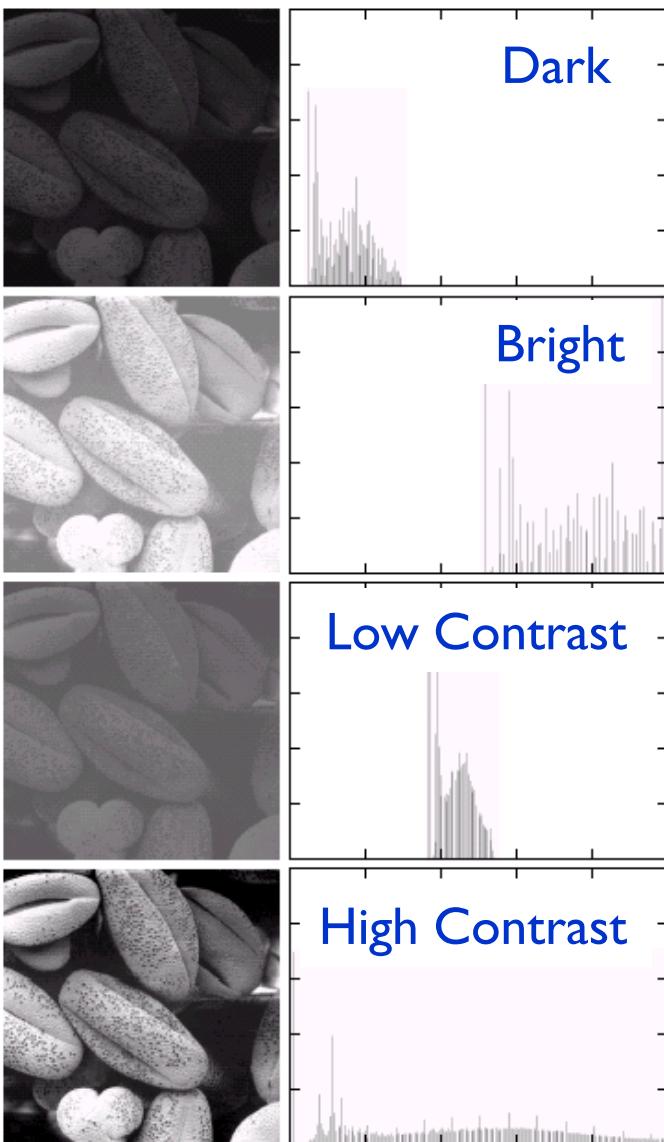


Image Histogram



- **Dark images:** bins in the low end
- **Bright images:** bins in the high end
- **Low contrast images:** bins in a narrow range
- **High contrast images:** bins are in the *entire range*

Image Histogram

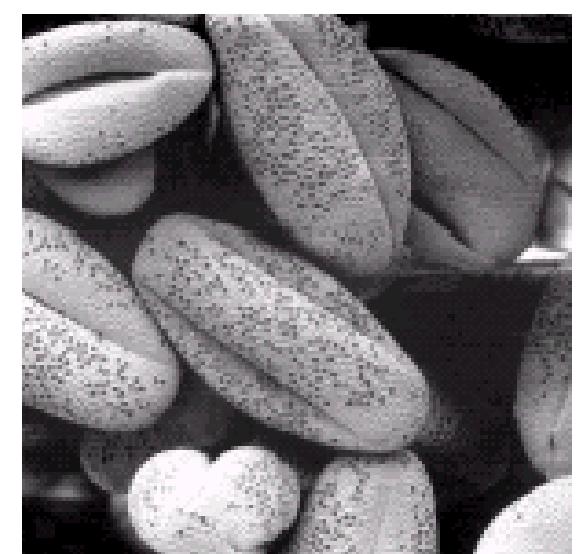
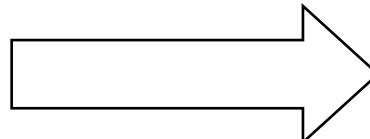
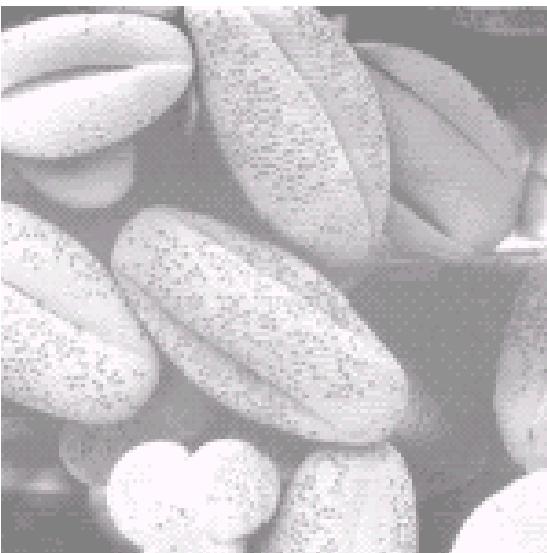
- An image whose pixels tend to occupy the **entire range** of possible intensity level, will have an appearance of high contrast.

Image Histogram

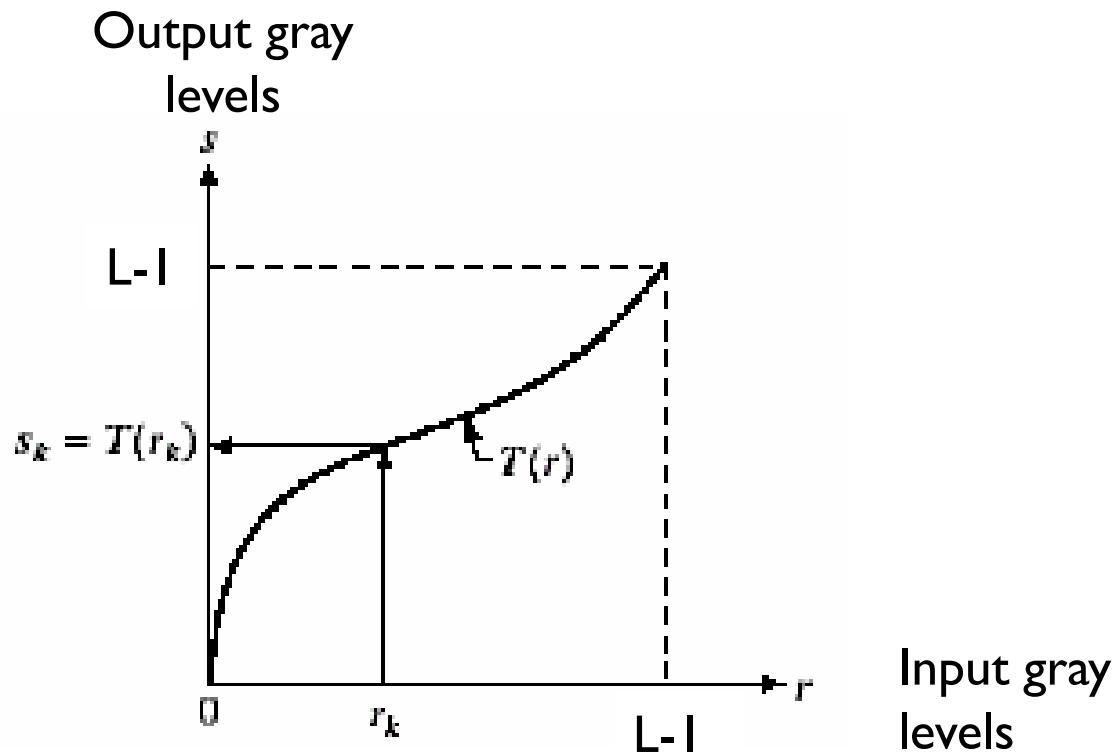
➤ `histogram(S,'Normalization','probability')`

Histogram Equalization

Histogram Equalization



Histogram Equalization



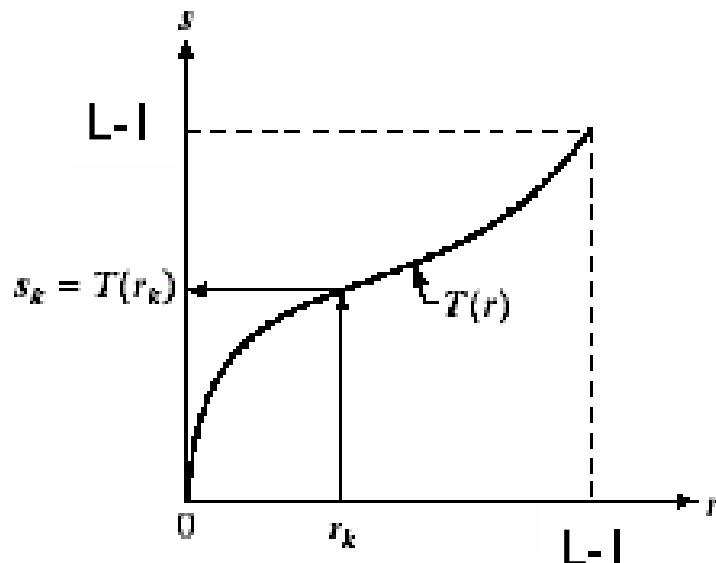
Histogram Equalization



- Let s, r : are all discrete and in $[0, L-1]$

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

- Let, 2 conditions hold:
- $T(r)$: *single valued and monotonous in* $0 \leq r \leq L-1$
- $0 \leq T(r) \leq L-1$ for $0 \leq r \leq L-1$



Condition 1 guarantees that output intensity values will never be less than corresponding input values thus preventing artifacts created by reversals of intensity.

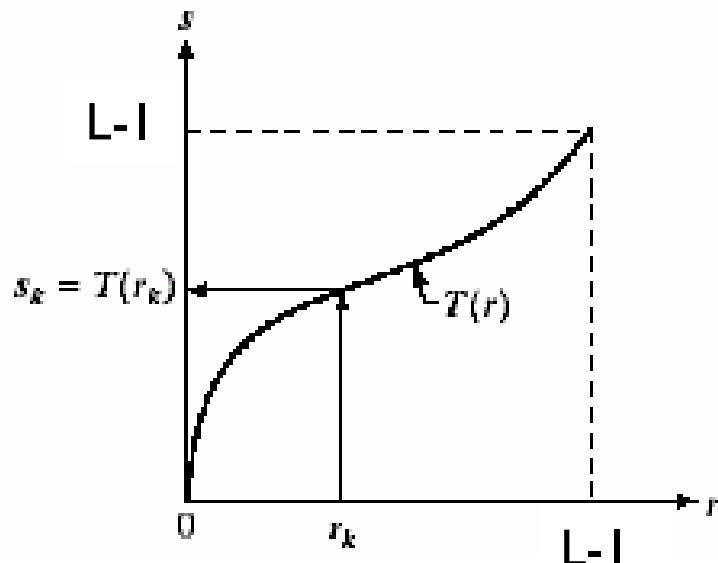
Histogram Equalization



- Let s, r : are all discrete and in $[0, L-1]$

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

- Let, 2 conditions hold:
- $T(r)$: *single valued and monotonous in* $0 \leq r \leq L-1$
- $0 \leq T(r) \leq L-1$ for $0 \leq r \leq L-1$

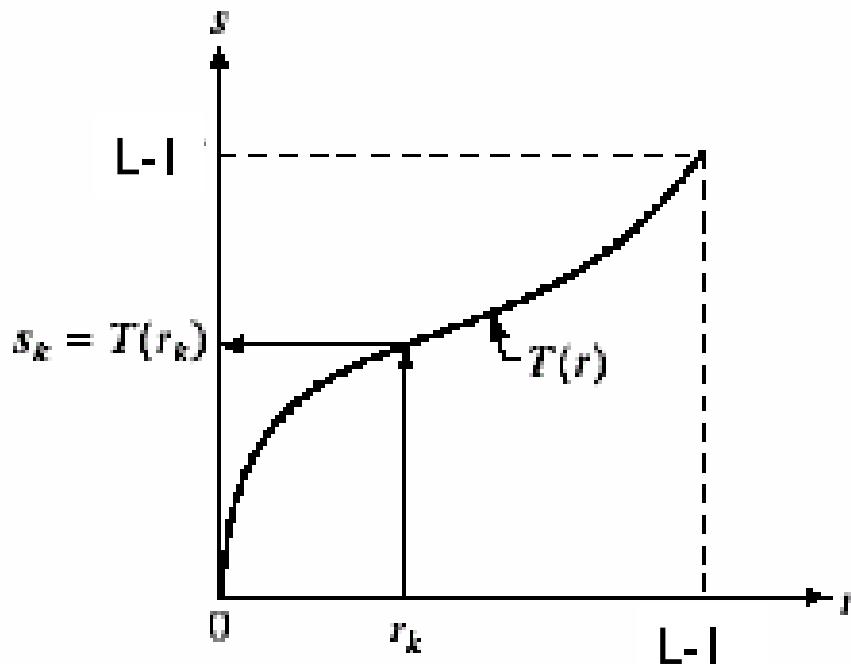


Condition 2 guarantees that the range of output intensities is the same as the input.

Histogram Equalization

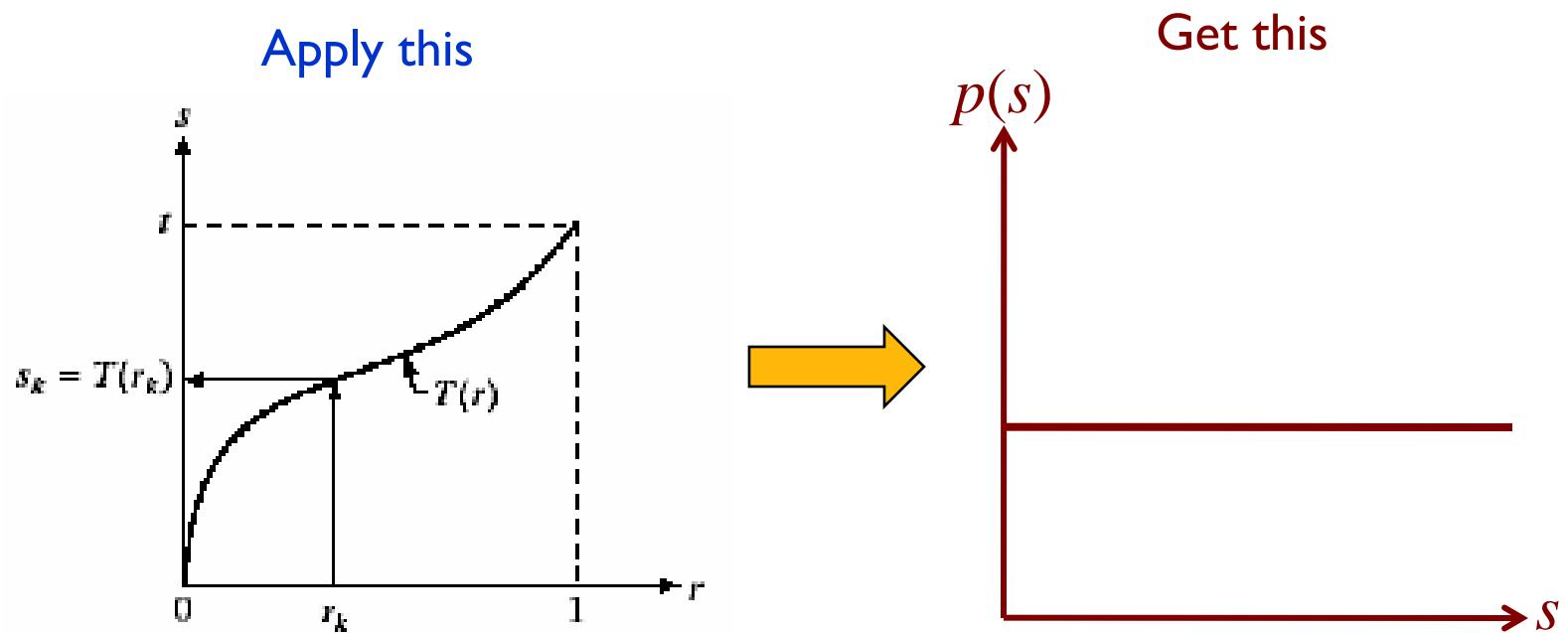
- Let,

- $p_r(r)$: the equalized histogram of the **given** image
- $p_s(s)$: the equalized histogram of the **output** image

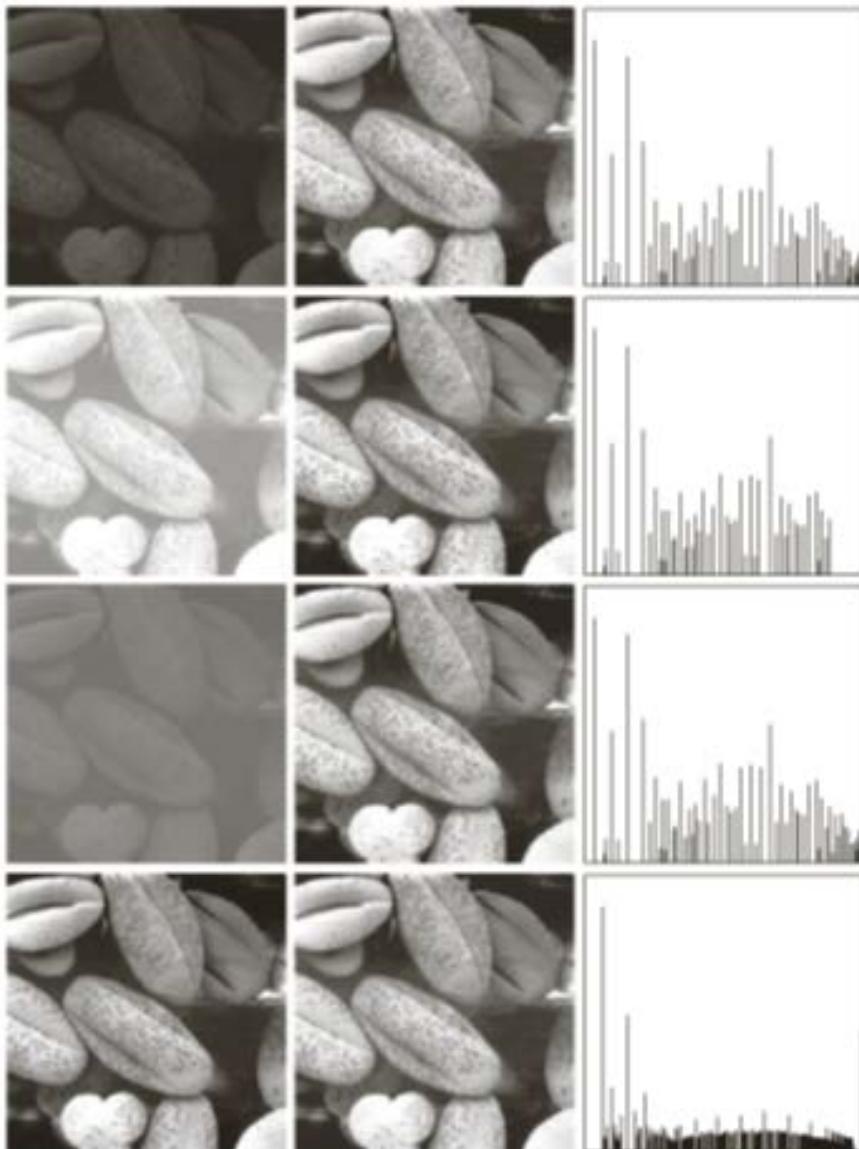


Histogram Equalization

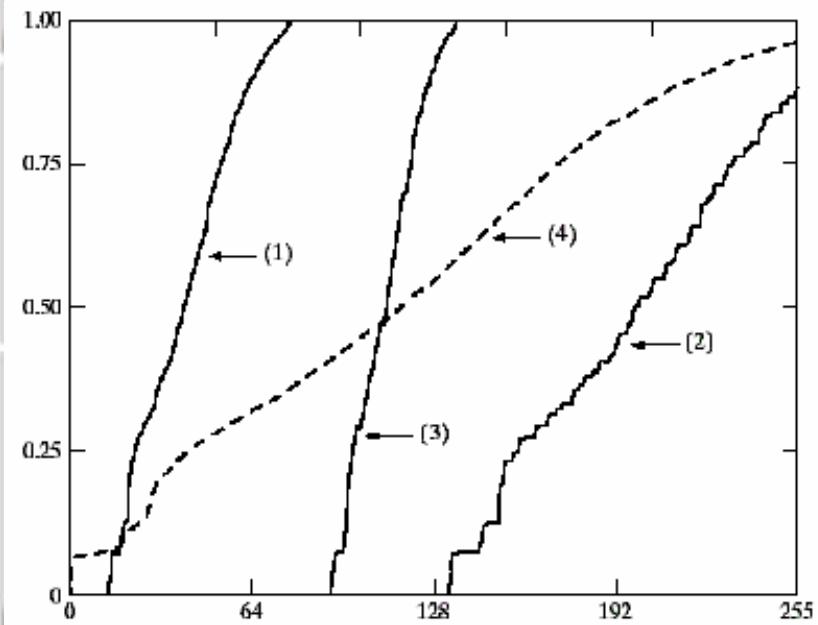
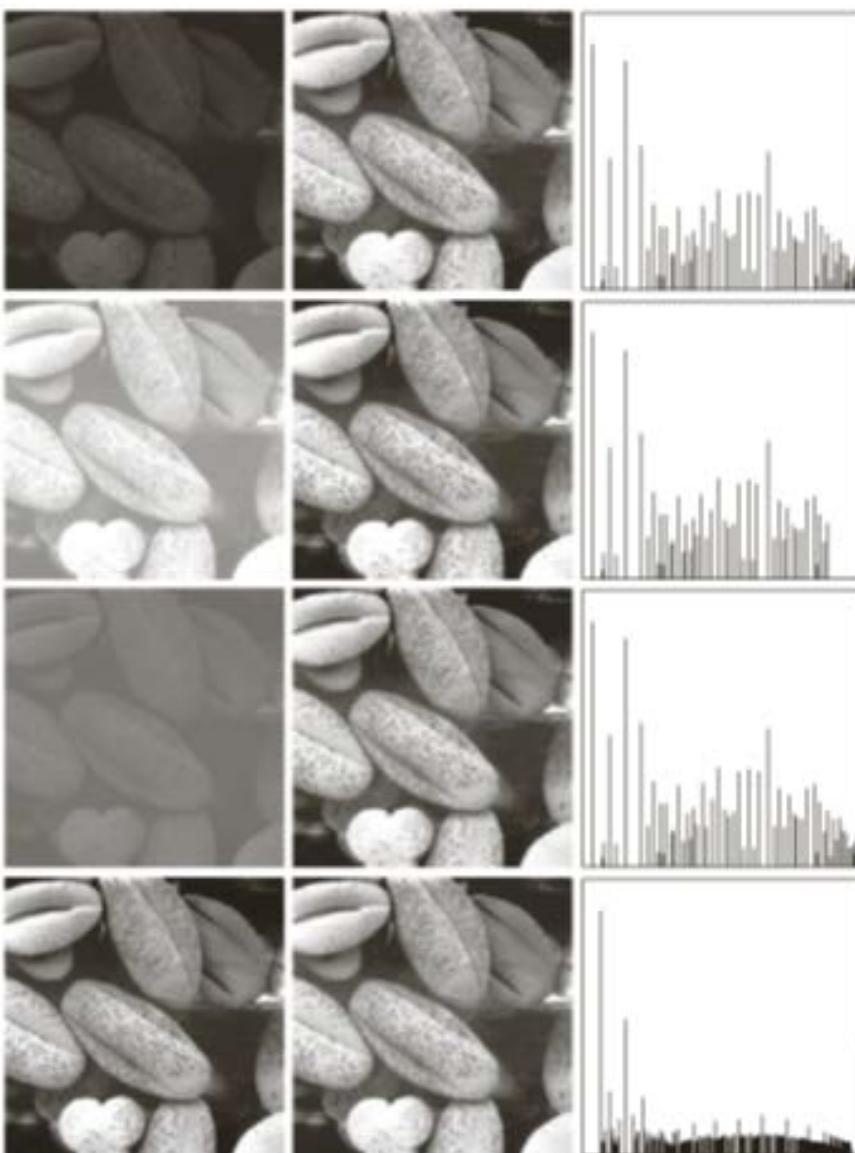
- **Objective:** to find the transformed image so that $p_s(s)$ of the transformed image is uniform



Histogram Equalization



Histogram Equalization



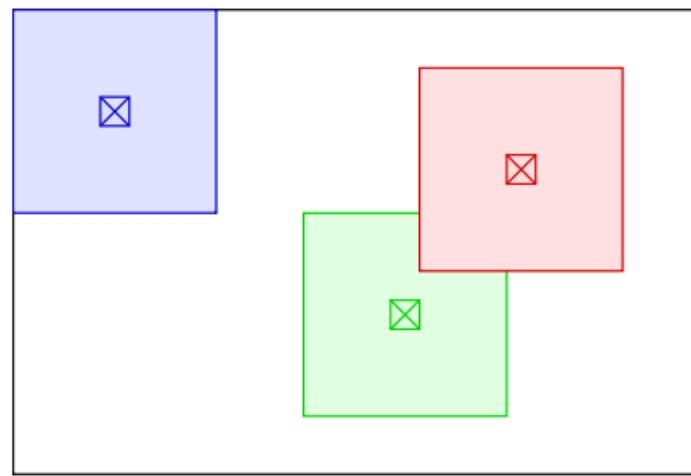
Local Histogram Processing

Local Histogram Processing

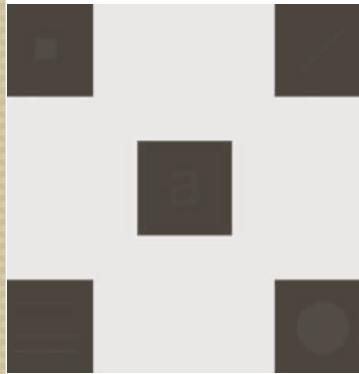
- The previous method was **global** because pixels are modified by a transformation function based on the intensity distribution of an entire image.
- In some cases, it is necessary to enhance details over small areas in an image.
- The solution is to devise transformation functions based on the intensity distribution in a neighborhood of every pixel in the image.

Local Histogram Processing

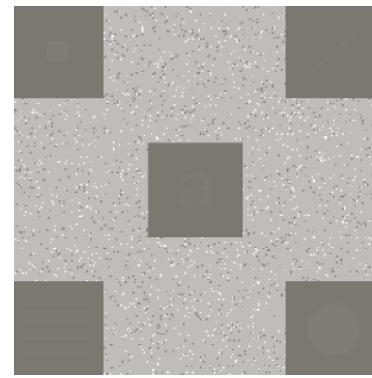
- Define a neighborhood and move its center from pixel to pixel.
- At each location, the histogram of the points in the neighborhood is computed.
- Histogram equalization is applied.



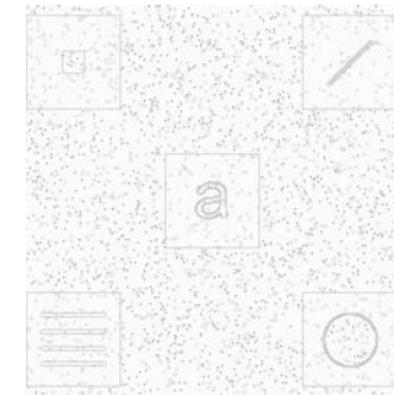
Local Histogram Processing



original



Equalized
with global
histogram



Equalized with
3X3 local
histogram

Local Histogram Processing

- Global histogram equalization increases noise
- Local histogram equalization provides more fine details than global histogram equalization
- The intensity values of the objects were too close to the intensity of the large squares and their sizes were too small to influence global histogram equalization.

Spatial Filtering

Spatial Filtering

- Neighborhood (sub-image) rolls over the entire image, each time centering a different pixel.
- For any specific location (x, y) , the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) .



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

Spatial Filtering

- This procedure is called **Spatial Filtering**.
- The **neighborhood** along with a predefined **operation** is called a **spatial filter**.
- Spatial filter is also known as **spatial mask**, **kernel**, **template** or **window**. Spatial mask is the most popular.

Spatial Filtering

- Spatial filter consists of
 - A neighborhood
 - A predefined operation
- Filtering creates a new pixel with coordinates equal to the coordinates of the center of the neighborhood and whose value is the result of the filtering operation.

Spatial Filtering

- **Linear spatial filter**

If the operation performed on the image pixels is linear, then the filter is called a **linear spatial filter**.

- **Non-linear spatial filter**

If the operation performed on the image pixels is non-linear, then the filter is called a **non-linear spatial filter**.

Spatial Filtering

- A *mask* or *filter* or *template* or *kernel* or *window* defines the neighborhood

- Mask size is usually $m \times n$

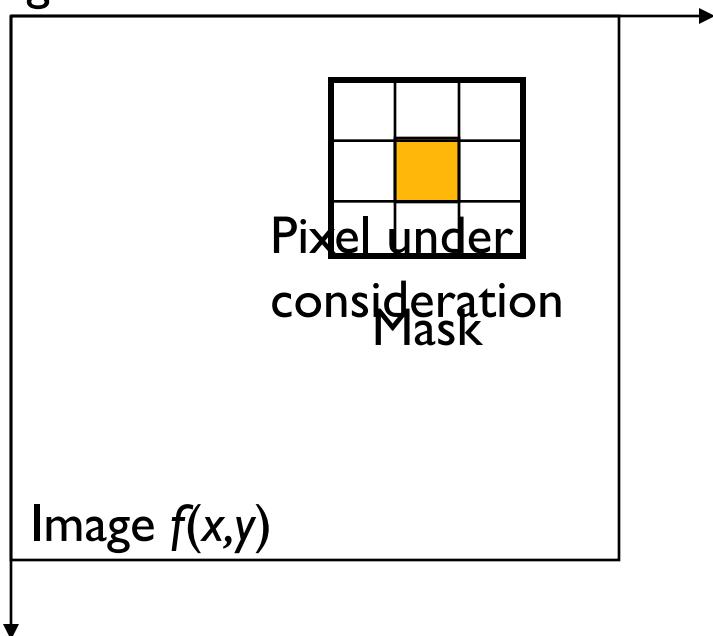
$$m = 2a + 1, n = 2b + 1$$

where a & b are positive integers.

Let's take a look how it works!!!

Spatial Filtering

Image
Origin

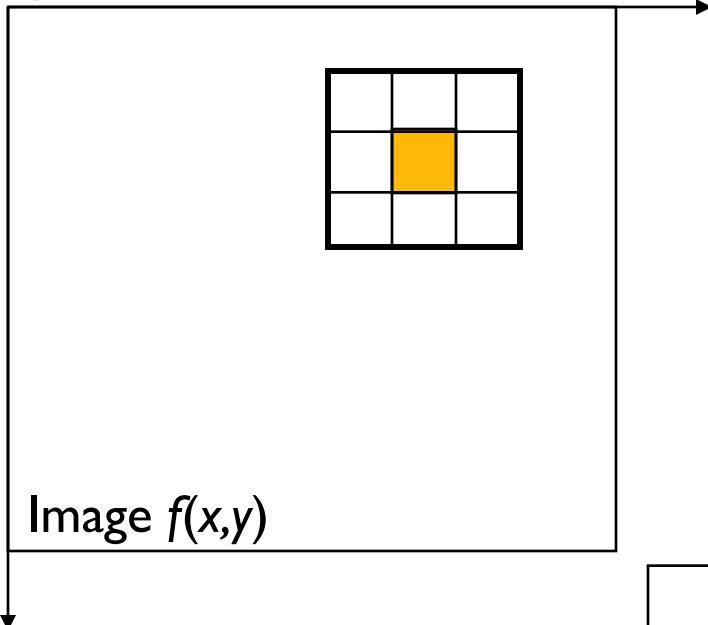


$w(-1,-1)$	$w(-1,0)$	$w(-1,1)$
$w(0,-1)$	$w(0,0)$	$w(0,1)$
$w(1,-1)$	$w(1,0)$	$w(1,1)$

Mask Coefficients
showing coordinate
arrangement

Spatial Filtering

Image
Origin



$w(-1,-1)$	$w(-1,0)$	$w(-1,1)$
$w(0,-1)$	$w(0,0)$	$w(0,1)$
$w(1,-1)$	$w(1,0)$	$w(1,1)$

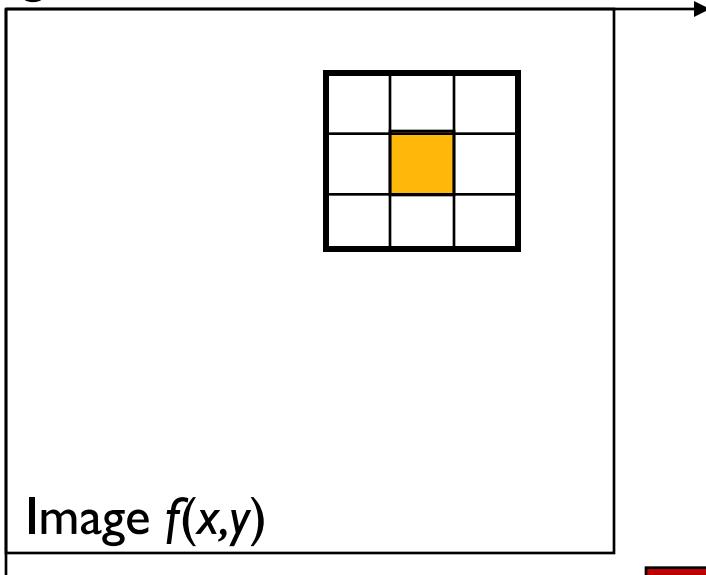
Mask Coefficients showing
coordinate arrangement

$f(x-1,y-1)$	$f(x-1,y)$	$f(x-1,y+1)$
$f(x,y-1)$	$f(x,y)$	$f(x,y+1)$
$f(x+1,y-1)$	$f(x+1,y)$	$f(x+1,y+1)$

Pixels of image section under Mask

Spatial Filtering

Image
Origin



$w(-1,-1)$	$w(-1,0)$	$w(-1,1)$
$w(0,-1)$	$w(0,0)$	$w(0,1)$
$w(1,-1)$	$w(1,0)$	$w(1,1)$

Mask Coefficients showing
coordinate arrangement

$f(x-1,y-1)$	$f(x-1,y)$	$f(x-1,y+1)$
$f(x,y-1)$	$f(x,y)$	$f(x,y+1)$
$f(x+1,y-1)$	$f(x+1,y)$	$f(x+1,y+1)$

Pixels of image section under Mask

Spatial Filtering

Response of the filter
at point (x, y) :

$$\begin{aligned} R = & w(-1,-1)f(x-1, y-1) \\ & + w(-1,0)f(x-1, y) + \dots \\ & \dots + w(1,0)f(x+1, y) \\ & + w(1,1)f(x+1, y+1) \end{aligned}$$

w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Mask Coefficients showing
coordinate arrangement

f(x-1,y-1)	f(x-1,y)	f(x-1,y+1)
f(x,y-1)	f(x,y)	f(x,y+1)
f(x+1,y-1)	f(x+1,y)	f(x+1,y+1)

Pixels of image section under Mask

Spatial Filtering

A more general equation for response:

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

Convolution operation

Convolution mask

w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Spatial Correlation and Convolution

Spatial Correlation

0 0 0 | 0 0 0 0

Discrete unit impulse function

Spatial Correlation

Correlation

$$\begin{array}{ccccccccc} \swarrow & \text{Origin} & & f & & w \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \\ & & & & & 1 & 2 & 3 & 2 & 8 \end{array}$$

Spatial Correlation

Correlation

Origin f w

0 0 0 1 0 0 0 0	1 2 3 2 8
0 0 0 1 0 0 0 0	
1 2 3 2 8	

Starting position alignment

Spatial Correlation

Correlation

$$\begin{array}{ccccccccc}
 & \swarrow & \text{Origin} & & f & & & w \\
 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 8 \\
 & & & \downarrow & & & & & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
 & & & & & & & & 1 & 2 & 3 & 2 & 8 \\
 & & & & & & & & \uparrow & & & & & & & \text{Starting position alignment}
 \end{array}$$

Spatial Correlation

Correlation

Origin f w

$$\begin{matrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ & \downarrow & & & & & & \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 3 & 2 & 8 \\ \uparrow & & & & & & \\ \text{Starting position alignment} \end{matrix}$$

Zero padding

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 3 & 2 & 8 \end{matrix}$$

Full correlation result

0 0 0 8 2 3 2 1 0 0 0 0

Spatial Correlation

Correlation

Origin f w

0 0 0 1 0 0 0 0	1 2 3 2 8
0 0 0 1 0 0 0 0	
1 2 3 2 8	

Starting position alignment

Zero padding

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8

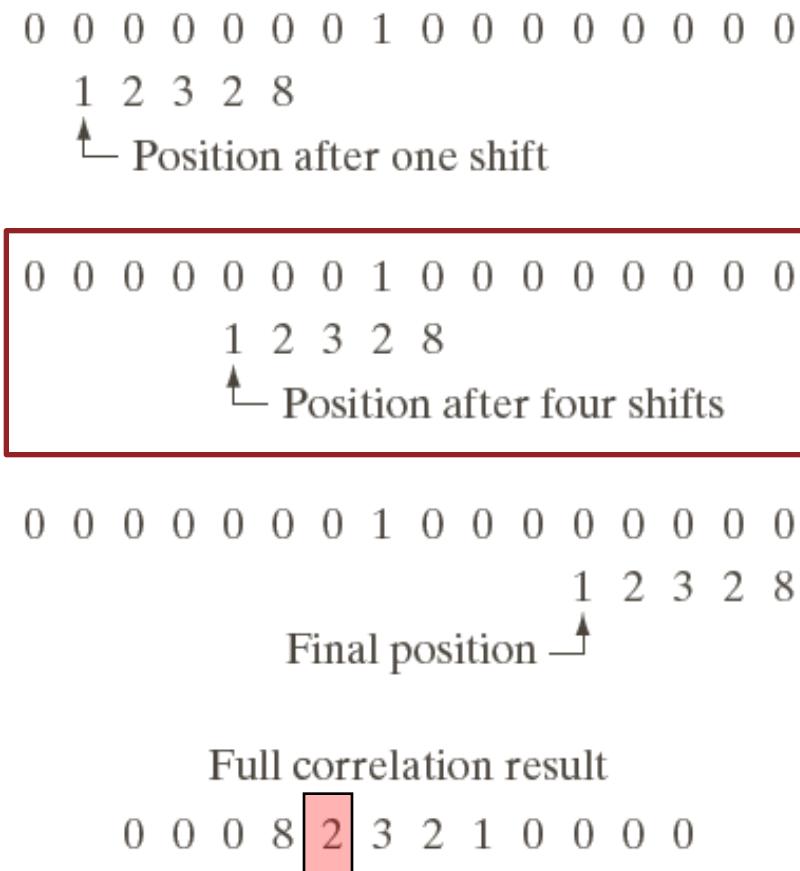
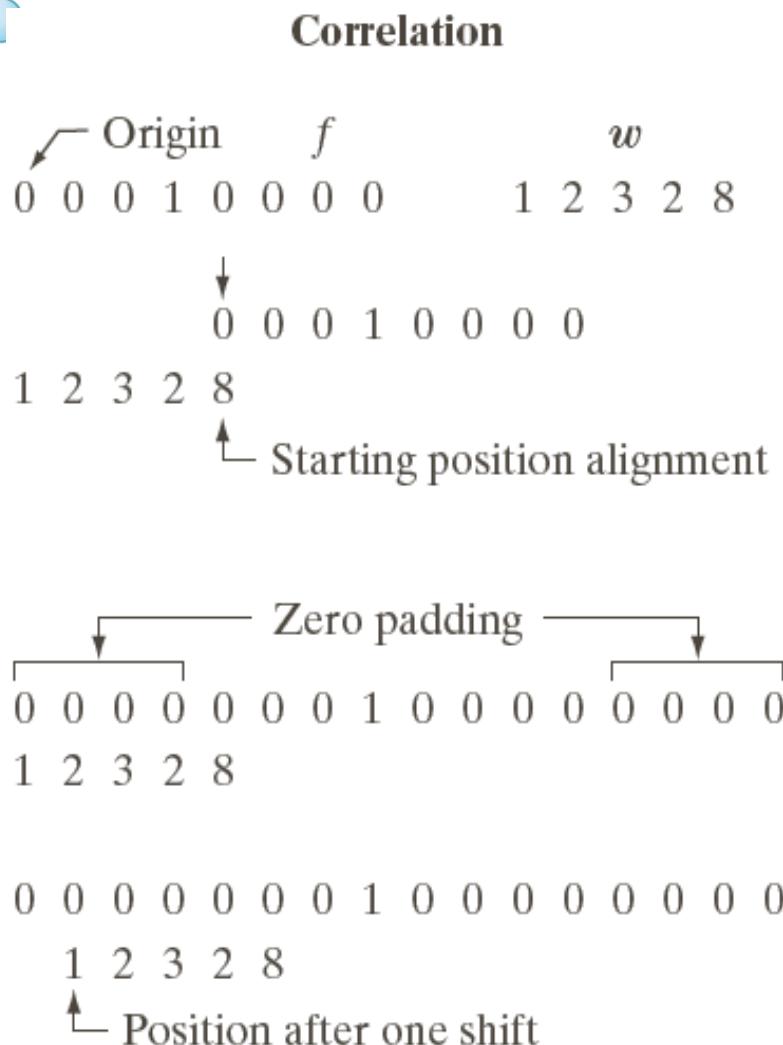
Position after one shift

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8

Full correlation result

0	0	0 8 2 3 2 1 0 0 0 0
---	---	---------------------

Spatial Correlation



Spatial Correlation

Correlation

Origin f w

0	0	0	1	0	0	0	0
1	2	3	2	8			

↓

0	0	0	1	0	0	0	0
1	2	3	2	8			

↑
Starting position alignment

Zero padding

0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	2	3	2	8										

0

0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	2	3	2	8										

↑
Position after one shift

0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
1	2	3	2	8																	

↑
Position after one shift

0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
1	2	3	2	8																	

↑
Position after four shifts

0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
1	2	3	2	8																	

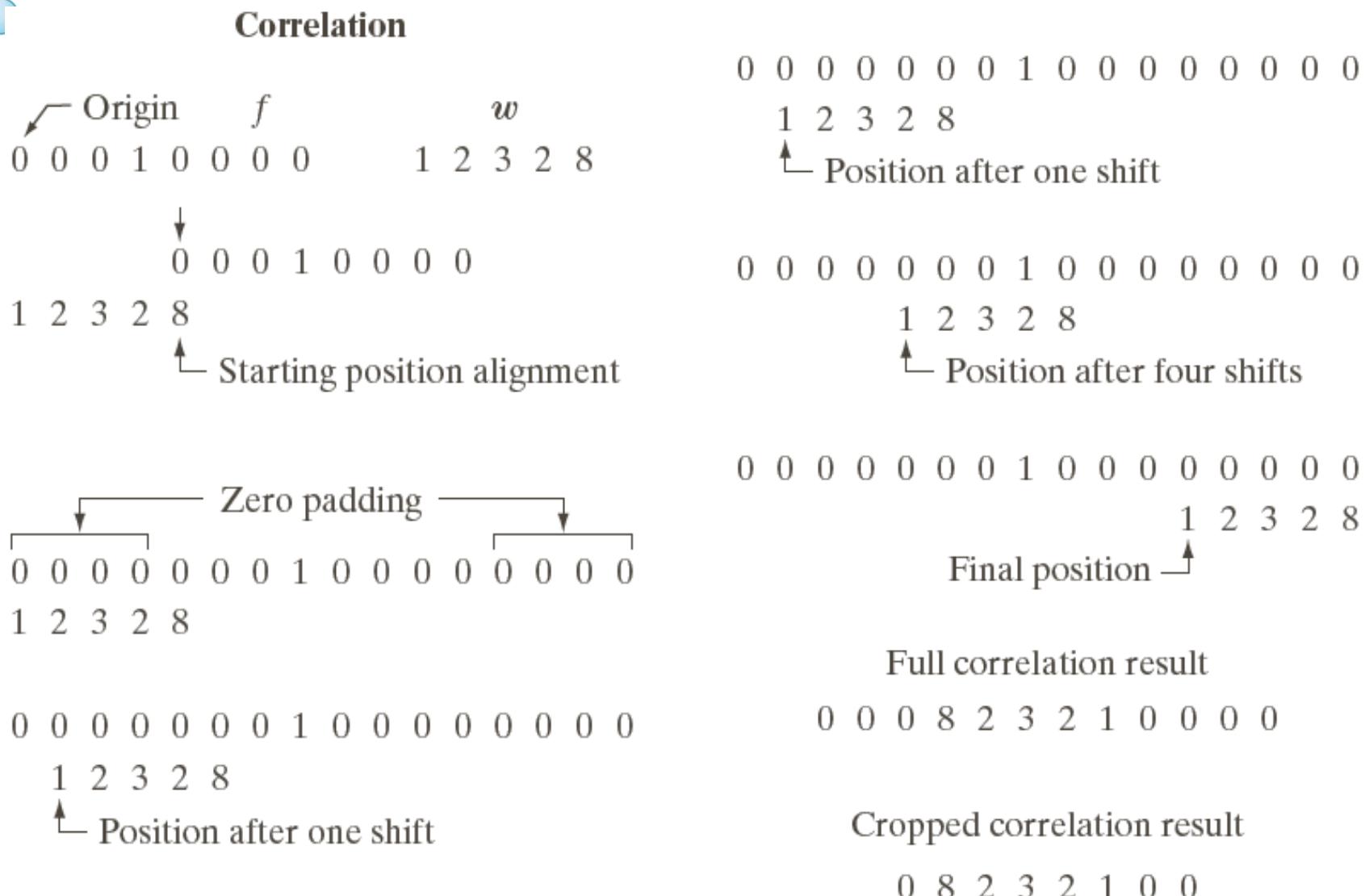
Final position ↑

Full correlation result

0	0	0	8	2	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	2	3	2	8																	



Spatial Correlation



Spatial Convolution

Now, Convolute **0 0 0 | 0 0 0 0** by **I 2 3 2 8**

Steps

- Reverse **I 2 3 2 8** to **8 2 3 2 I**
- Then apply correlation as mentioned in previous slides

Spatial Convolution

Convolution

Origin f w rotated 180°

0	0	0	1	0	0	0
8	2	3	2	1		

Starting position

0	0	0	1	0	0	0
8	2	3	2	1		

Full padded

0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	2	3	2	1										

Position after one shift

0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	2	3	2	1										

0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1 Position after one shift

0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1 Position after 4 shifts

0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1 Final position

Full convolution result

0	0	0	1	2	3	2	8	0	0	0	0

Cropped convolution result

0	1	2	3	2	8	0	0

2D Correlation and Convolution

240	225	210	190	160
220	195	170	140	115
190	160	145	125	100
150	130	115	95	75
110	85	65	50	40
70	60	45	30	15

An Image

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

Mask

Apply the mask and calculate the first row

Issues in spatial filtering

- When the mask moves closer to the border
 - Closer to that distance: some rows/columns of the mask are out of image
- Way out:
 - Ignoring the outside columns/rows
 - Padding
 - Zero padding
 - Mirror padding

Smoothing Spatial Filters

Smoothing Spatial Filters

- Smoothing filters are used for
 - blurring
 - noise reduction
 - sharp transition reduction
 - finding large objects, ignoring small details

Smoothing Linear Filters

- The output of a smoothing linear spatial filter is simply the (weighted) average of the pixels contained in the neighborhood.
- Also known as **averaging filters** or **lowpass filters**.

Smoothing Linear Filters

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

1	1	1
1	1	1
1	1	1

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

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

1	2	1
2	4	2
1	2	1

A spatial averaging filter in which all coefficients are equal is called a **box filter**.

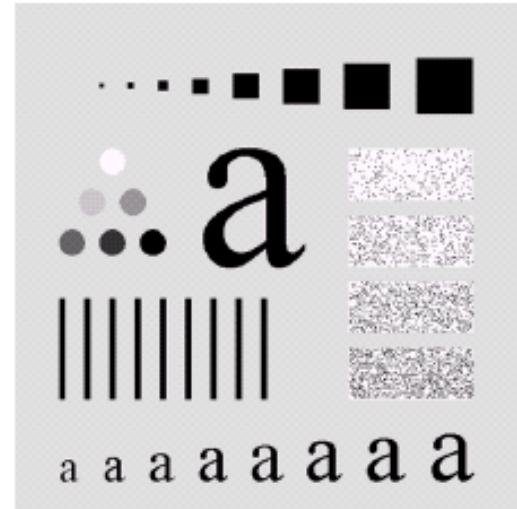
This mask yields a weighted average. It gives more importance to the center pixels and the 4-neighbors.

Smoothing Linear Filters

```
clc;  
clear all;  
close all;  
I = imread('cameraman.tif');  
figure, imshow(I);  
  
h = fspecial('average',5); % h = fspecial('average',hszie)  
localMean = imfilter(I,h,'replicate');  
imshowpair(I,localMean,'montage')
```

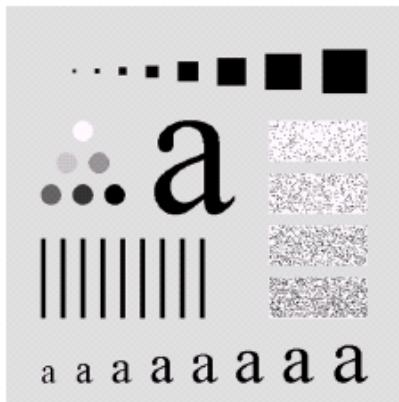
Smoothing Linear Filters

- Black square: 3, 5, 9, 15, 25, 35, 45
- Border 25 pixels apart
- Circle: 25 pixels, gray: 0, 20, ..., 100%
- Small a's: 10, 12, 24 pixels
- Large a: 60 pixels
- Bars: 5X100 pixels, Sep: 20 pix
- Noise boxes: 50X120 pixels

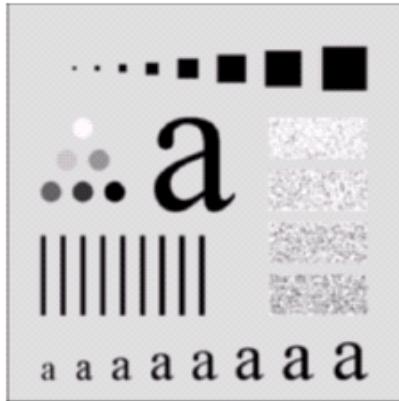


Smoothing Linear Filters

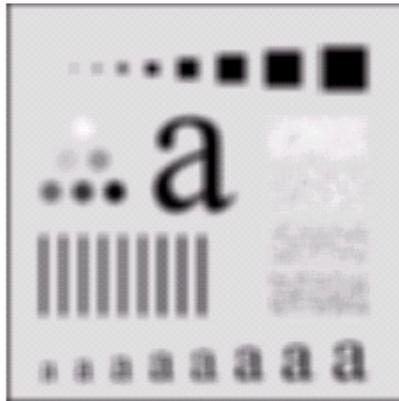
Original Image



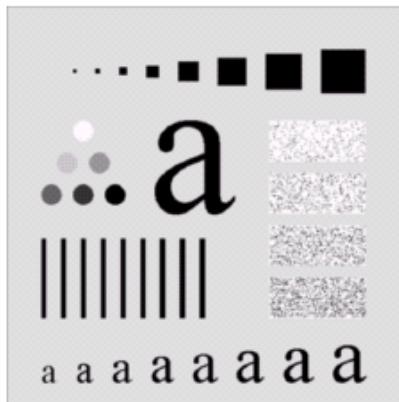
After applying
 5×5 filter



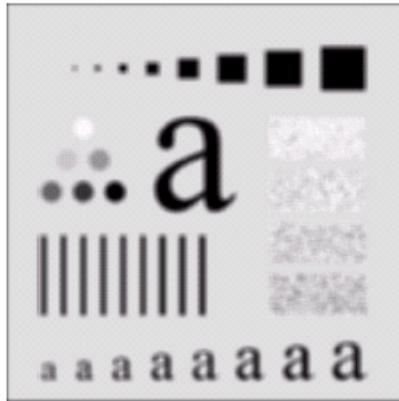
After applying
 15×15 filter



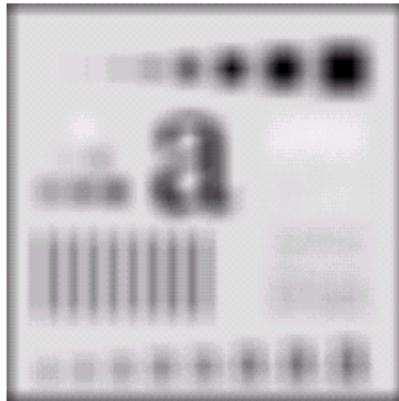
After applying
 3×3 filter



After applying
 9×9 filter

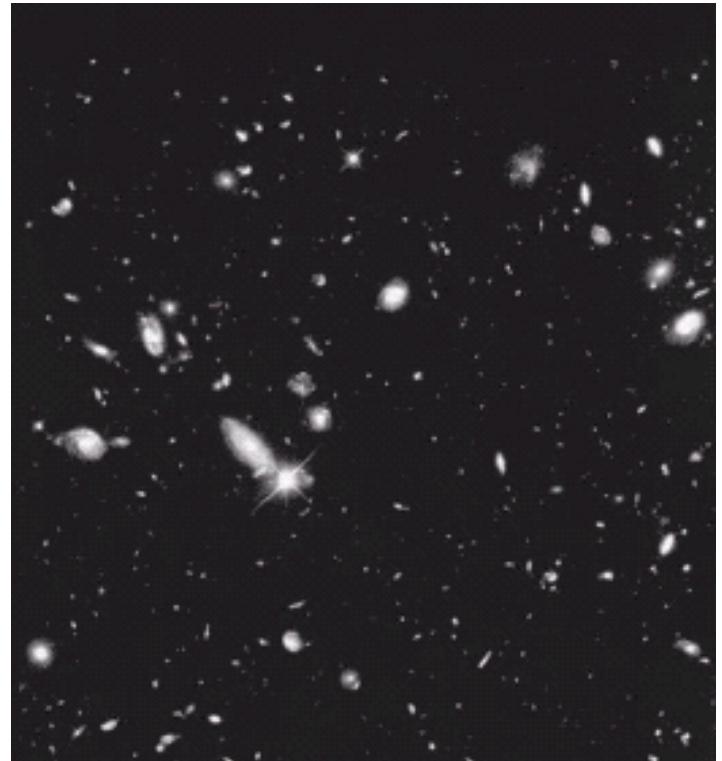


After applying
 25×25 filter



Smoothing Linear Filters

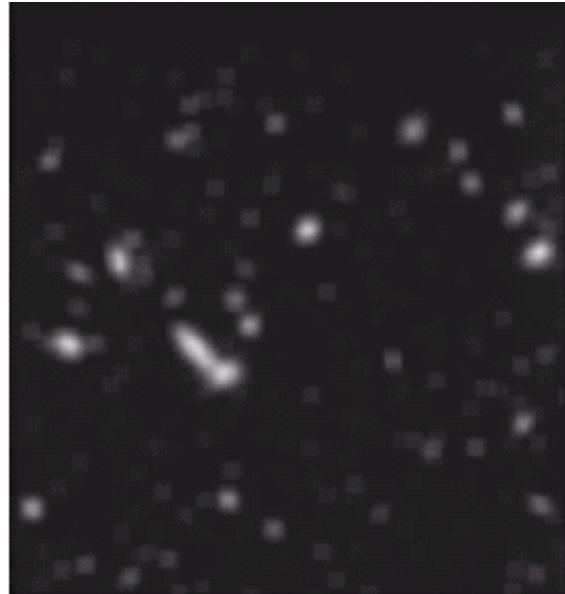
- Blurring helps to find major objects
- Removes fine details
- Original image contains lots of small objects



Smoothing Linear Filters



Original image



Smoothed by
15X15 mask



Thresholding
after smoothing

Smoothing Non-linear Filters

Order-Statistic(Nonlinear) Filters

- Nonlinear filtering
- Response is based on ordering(ranking) the pixels
- Example : Median filter

Median Filter

Advantage

- For certain type of random noise, they provide excellent noise reduction
- Considerably less blurring effect
- Particularly effective in the presence of **impulse noise**, also called **salt-and-pepper noise**.

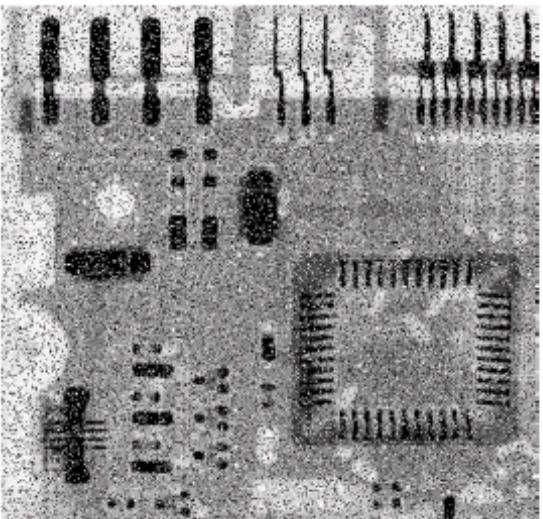
Median Filter

- Policy:
 - Sort the values of enclosed pixels
 - Select the median as output pixel level
- Example:
 - Let, 3X3 mask size
 - Sorted values: 10, 15, 20, 20, 20, 20, 20, 25, 100
 - Median is 20
 - Output pixel value is 20

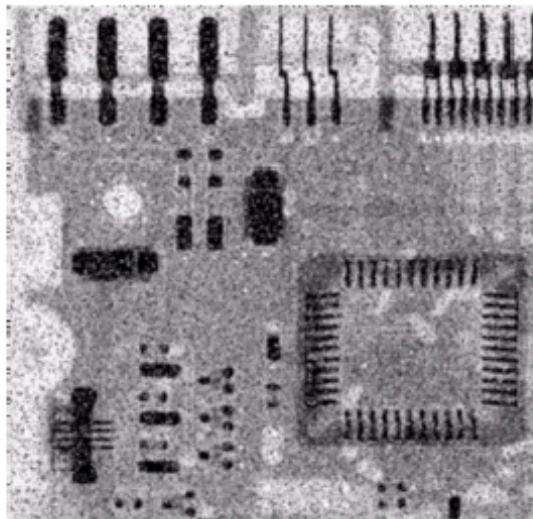
10	20	20
20	100	20
25	20	15

Image pixel values
under masks

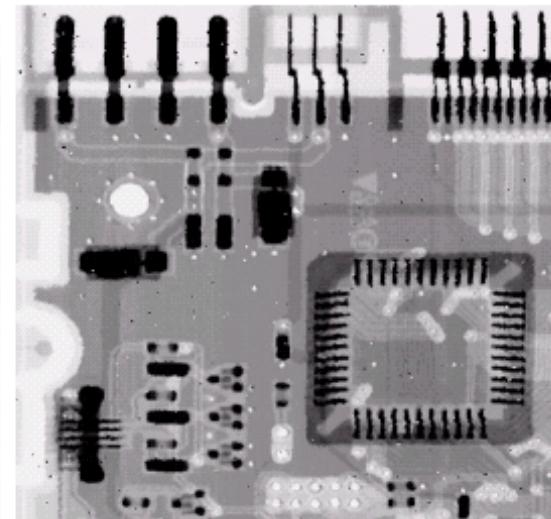
Median Filter



Noisy image



Noise reduction
by Avg. filter



Noise reduction
by median filter

Histogram processing

The **histogram** of a digital image with **L** total possible intensity levels in the range **[0,G]** is defined as the discrete function:

$$h(r_k) = n_k$$

Where

- ✓ r_k is the k_{th} intensity level in the interval **[0,G]**
- ✓ n_k is the number of pixels in the image whose intensity level is r_k
- ✓ **G**: [255 for images of class *uint8*, 65535 for images class *uint16* and 1.0 for images of class *double*].

Histogram processing

Normalized histograms: can be obtained by dividing all elements of $h(r_k)$ by the total number of pixels in the image:

$$P(r_k) = \frac{h(r_k)}{n} = \frac{n_k}{n}, \text{ for } k = 1, 2, \dots, L$$

n total number of pixels

From probability:

- $P(r_k)$ - *Estimation* of the probability of occurrence of intensity level r_k . (The sum of all components of a normalized histogram is equal =1).

Histogram processing

Histogram Equalization

- **Histogram Equalization:** is a method which increases the dynamic range of the gray-level 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).

Histogram processing

Continuous Case

- For intensity levels that are continuous quantities *normalised* to the range $[0, 1]$.

Let $P_r(r)$ – probability density function (PDF) of the intensity levels. The following transformation on the input levels to obtain output levels, s :

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

ω -dummy variable of integration.

- The PDF of the output levels is uniform:

$$P_s(s) = \begin{cases} 1 & 0 \ll s \ll 1 \\ 0 & \text{otherwise} \end{cases}$$

- Image, whose intensity levels are equally likely, and it cover the entire range $[0, 1]$.
- This transformation is called intensity-levels equalization process (and it's nothing more than the cumulative distribution function (CDF)).

Discrete case (quantities):

- The equalization transformation becomes:

$$S_k = T(r_k) = \sum_{j=1}^k P_r(r_j) = \sum_{j=1}^k \frac{n_j}{n}, \text{ for } k = 1, 2, \dots, L,$$

S_k – Intensity value of the output image corresponding to value r_k in the input image.

Histogram processing

- As noted earlier, the transformation function $T(r_k)$ is simply the *cumulative sum of normalized histogram* values, we can use the function **cumsum** to obtain the transformation function, type:
`>> hnorm = imhist (f) ./numel(f);
>> cdf = cumsum (hnorm);`

- A plot of **CDF** (for example) can be obtained using the following commands:

```
>> x= linspace (0, 1, 256);  
% Intervals for [0, 1] horiz scale  
>> plot (x, cdf) % plot cdf vs. x  
>> axis ([0 1 0 1]) % scale, setting and labels  
>> set (gca, 'xtick', 0:0.2:1)  
>> set (gca, 'ytick', 0:0.2:1)  
>> xlabel ('Input Intensity values', 'fontsize' ,9)  
>> ylabel ('Output Intensity values', 'fontsize' , 9)  
>> %specify text in the body of the graph:  
>> text (0.18, 0.5,'Transformation function',...  
'fontsize' , 9)
```

Histogram processing

CDF Manual Calculation

Example :

- Consider an 8-bit gray scale image has the following values:

52	55	61	66	70	61	64	73
63	59	55	90	109	85	69	72
62	59	68	113	144	104	66	73
63	58	71	122	154	106	70	69
67	61	68	104	126	88	68	70
79	65	60	70	77	68	58	75
85	71	64	59	55	61	65	83
87	79	69	68	65	76	78	94

8*8 sub image

- The *histogram* for this sub image is shown in the following table. Pixels values that have a zero count are *excluded*.

Histogram processing

Value	Count	Value	Count	Value	Count	Value	Count
52	1	66	2	77	1	106	1
55	3	67	1	78	1	109	1
58	2	68	5	79	2	113	1
59	3	69	3	83	1	122	1
60	1	70	4	85	2	126	1
61	4	71	2	87	1	144	1
62	1	72	1	88	1	154	1
63	2	73	2	90	1		
64	2	75	1	94	1		
65	3	76	1	104	2		

Histogram processing

- The *cumulative distribution function (cdf)* is shown below:

Value	cdf								
52	1	64	19	72	40	85	51	113	60
55	4	65	22	73	42	87	52	122	61
58	6	66	24	75	43	88	53	126	62
59	9	67	25	76	44	90	54	144	63
60	10	68	30	77	45	94	55	154	64
61	14	69	33	78	46	104	57		
62	15	70	37	79	48	106	58		
63	17	71	39	83	49	109	59		

Histogram processing

- This *cdf* table shows that the minimum value in the sub image is 52 and the maximum value is 154.

$$cdf(52) = 1, \quad cdf(154) = 64$$

- The *cdf* must be normalized to [0, 255],
 - The **general histogram equalization formula** is:

$$h(v) = \text{round} \left(\frac{cdf(v) - cdf_{min}}{(M * N) - cdf_{min}} * (L - 1) \right)$$

Where:

- *cdf_{min}*- The minimum value of the *cdf*
- *M * N*: image's number of pixels. (*M-width, N-height*)
- *L* -number of gray scale levels (in most cases, 255).
- The equalization formula for this particular example, is

$$h(v) = \text{round} \left(\frac{cdf(v) - 1}{63} * 255 \right)$$

Histogram processing

- For example: the $cdf(78) = 46$, the *normalized* value becomes:

$$h(78) = \text{round} \left(\frac{46 - 1}{63} * 255 \right) = \text{round} (0.714286 * 255) = 182$$

- We use the above formula to calculate the normalized *cdf* and the values of the equalized image are directly taken from normalized *cdf*, the following table contains the normalized *cdf*.

0	12	53	93	146	53	73	166
65	32	12	215	235	202	130	158
57	32	117	239	251	227	93	166
65	20	154	243	255	231	146	130
97	53	117	227	247	210	117	146
190	85	36	146	178	117	20	170
202	154	73	32	12	53	85	194
206	190	130	117	85	174	182	219

Sharpening Spatial Filters

**The concept of
sharpening filter**

Sharpening Spatial Filters

- To highlight fine detail in an image or to enhance detail that has been blurred, either in error or as a natural effect of a particular method of image acquisition.

Blurring vs Sharpening

- Blurring/smooth is done in spatial domain by pixel averaging in a neighbors, it is a process of integration.
- Sharpening is an inverse process, to find the difference by the neighborhood, done by spatial differentiation. spatial differentiation.

Sharpening Spatial Filters

- Sharpening filters highlights transitions in intensity
- Very useful for highlighting fine details
- Helps to remove blurring

Smoothing vs Sharpening

Smoothing



Averaging



Integration

Sharpening



Difference



Differentiation

Image Sharpening

Sharpening



Difference



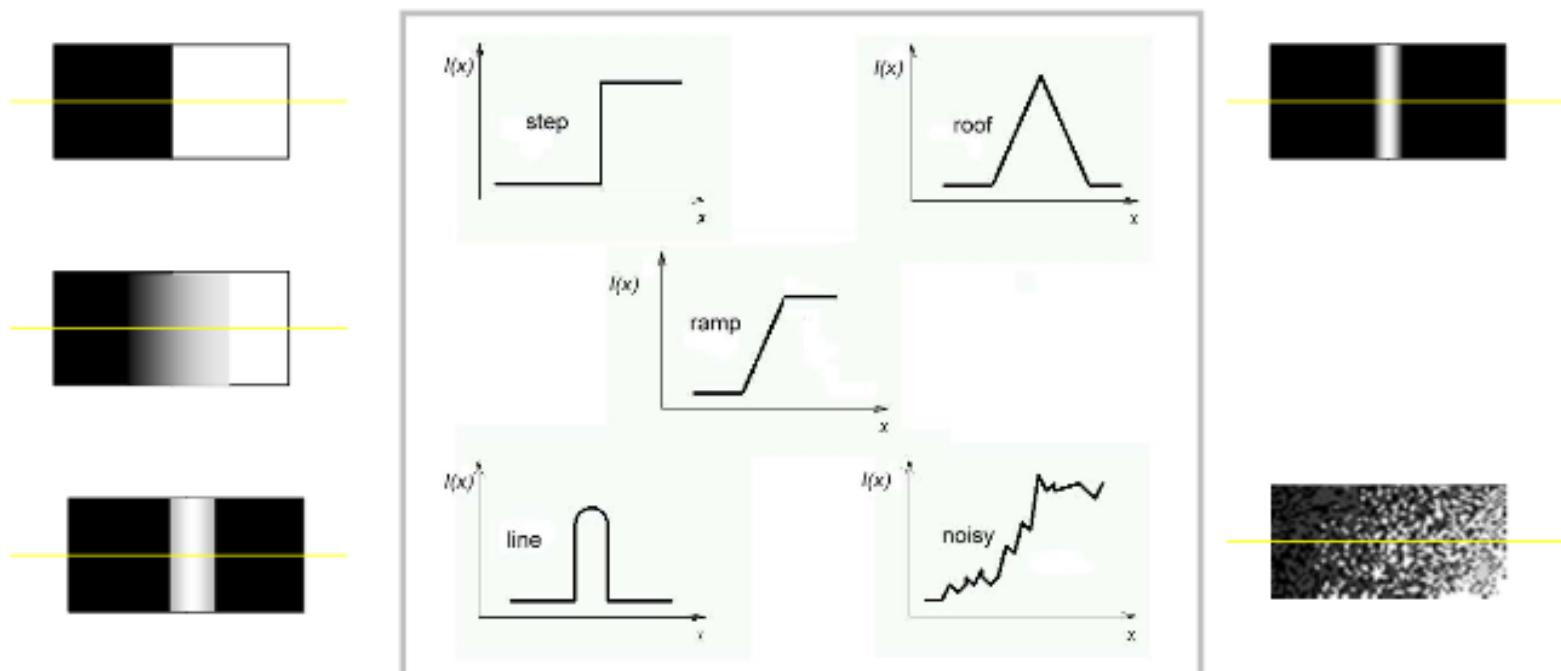
Differentiation

- Strength of response of derivative proportional to image discontinuity

- Sharp changes noise point, edges, lines, grey ramp are easily detected

Sharpening Spatial Filters

- Edge is a boundary between two regions with relatively distinct gray level properties.
- Edges are pixels where the brightness function changes abruptly.
- Edge detectors are a collection of very important local image pre-processing methods used to locate (sharp) changes in the intensity function.



Criteria for Optimal Edge Detection

(1) Good detection

- Minimize the probability of false positives (i.e., spurious edges).
- Minimize the probability of false negatives (i.e., missing real edges).

(2) Good localization

- Detected edges must be as close as possible to the true edges.

Criteria for Optimal Edge Detection

- Often, points that lie on an edge are detected by:

(1) Detecting the local maxima or minima of the first derivative.

(2) Detecting the zero-crossings of the second derivative.

Criteria for Optimal Edge Detection

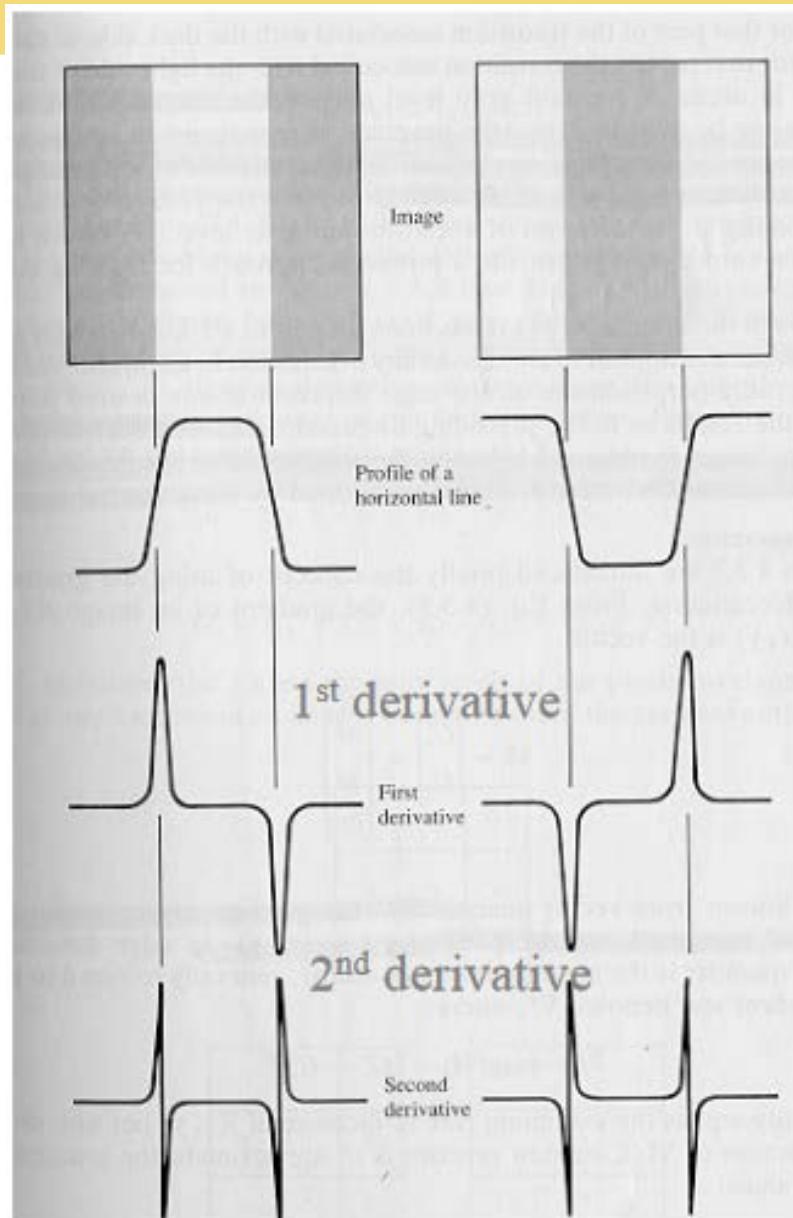


Image Sharpening

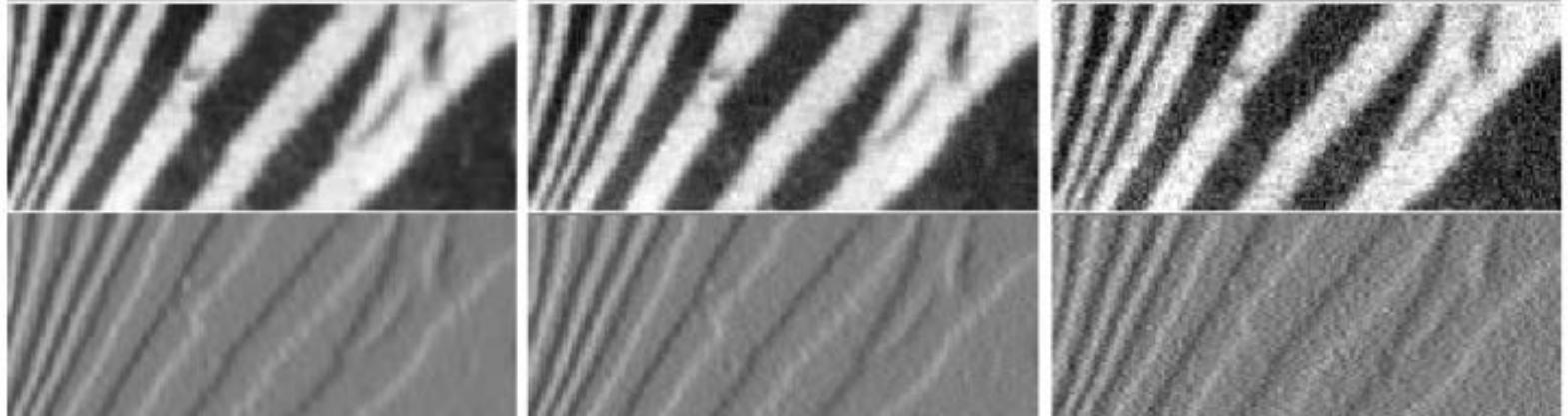
Derivative operator

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

Derivatives and Noise

- Derivatives are strongly affected by noise
 - obvious reason: image noise results in pixels that look very different from their neighbors
 - The larger the noise - the stronger the response
- What is to be done?
 - Neighboring pixels look alike
 - Pixel along an edge look alike
 - Image smoothing should help
 - Force pixels different to their neighbors (possibly noise) to look like neighbors

Derivatives and Noise



Increasing noise



- Need to perform image smoothing as a preliminary step
- Generally – use Gaussian smoothing

Image Sharpening

First and second order difference of 1D

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

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

- The second-order derivative of a one-dimensional function $f(x)$ is the difference

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

Image Sharpening

- 1st and 2nd order derivatives will be used
- Behavior to check
 - Const gray level
 - At onset and end of discontinuities (ramp and step)
 - Along gray-ramp

Properties of Derivatives

- Value of 1st order derivative will be
 - 0 at const gray level
 - Nonzero at onset of step and ramp
 - Nonzero along ramp

- Value of 2nd order derivative will be
 - 0 at const gray level
 - Nonzero at onset and end of step and ramp
 - 0 along ramp

Digital 1st Order Derivative

$$\frac{\partial f}{\partial x} = \frac{\text{change of } f}{\text{change of } x}$$

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

Digital 2nd Order Derivative

$$\frac{\partial^2 f}{\partial x^2} = \frac{\text{change of } \frac{\partial f}{\partial x}}{\text{change of } x}$$

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

1st Order & 2nd Order Derivative

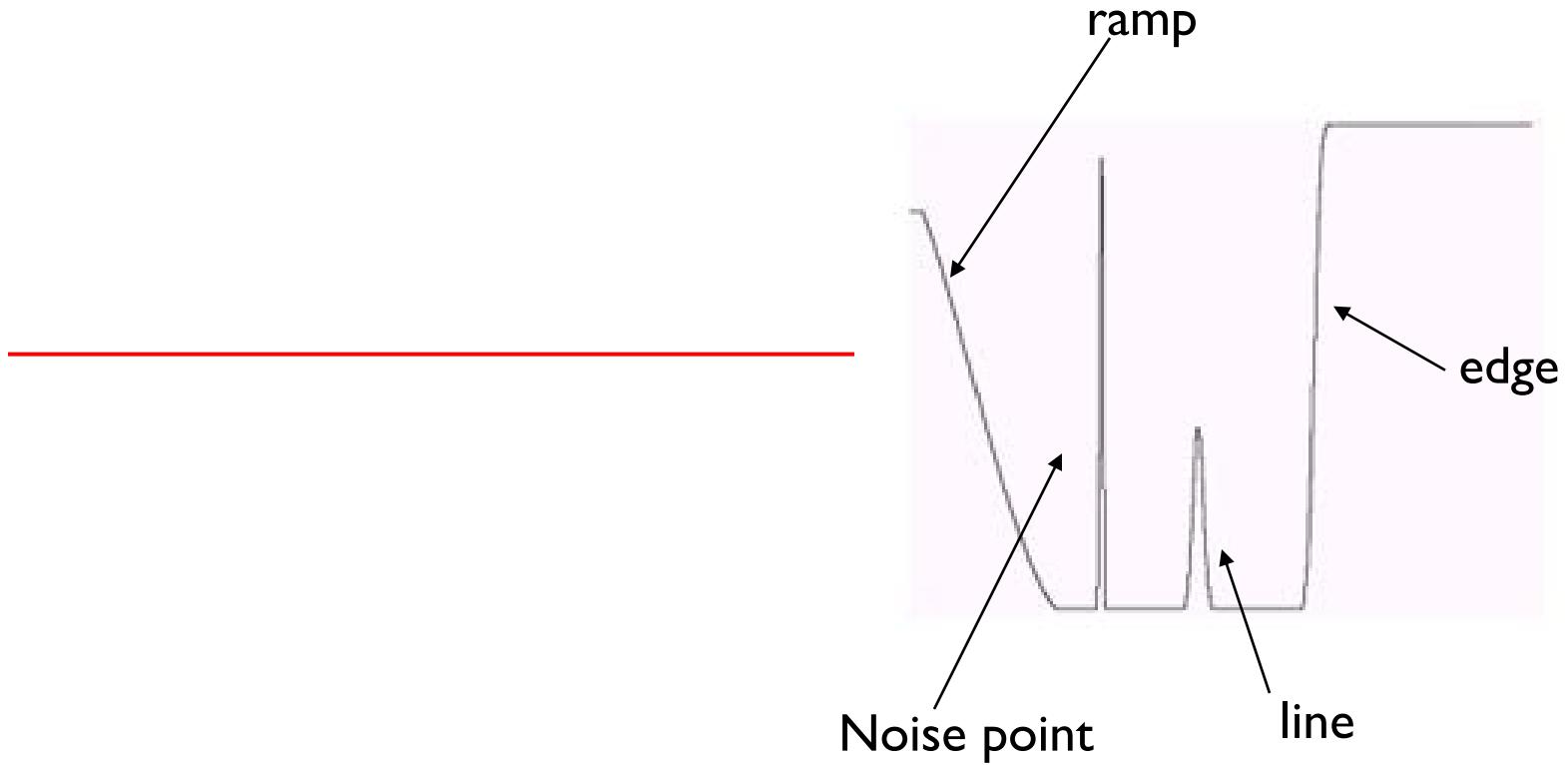


- Gray-ramp (smooth transition betn white and black)
- Isolated noise point
- Line
- Edge

1st Order & 2nd Order Derivative



1st Order & 2nd Order Derivative

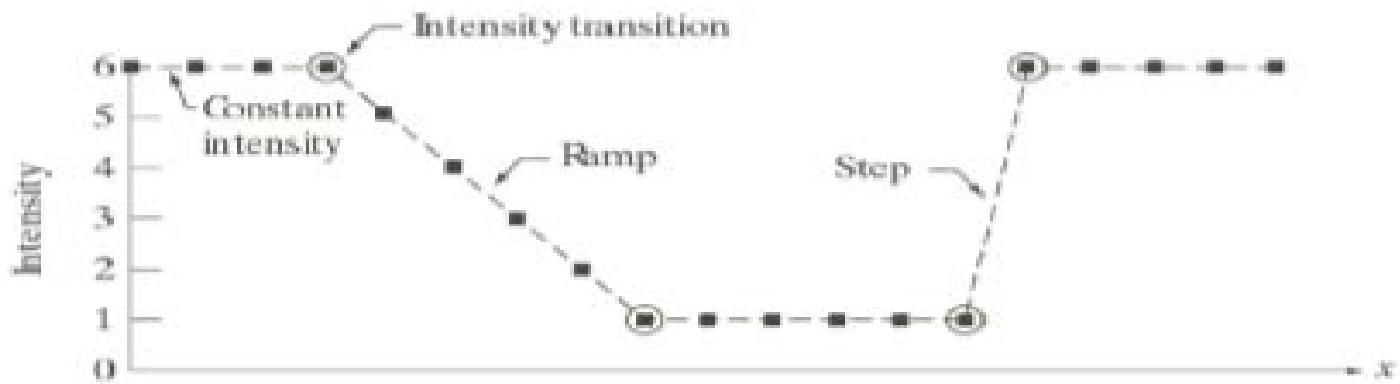


1st Order & 2nd Order Derivative

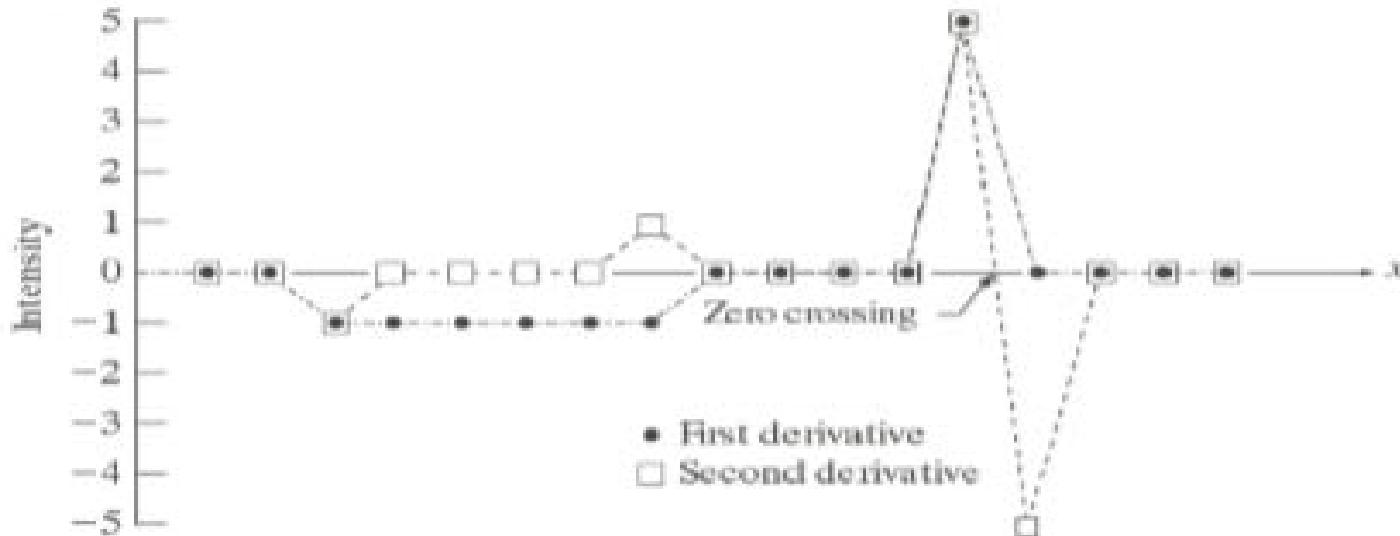
$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

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

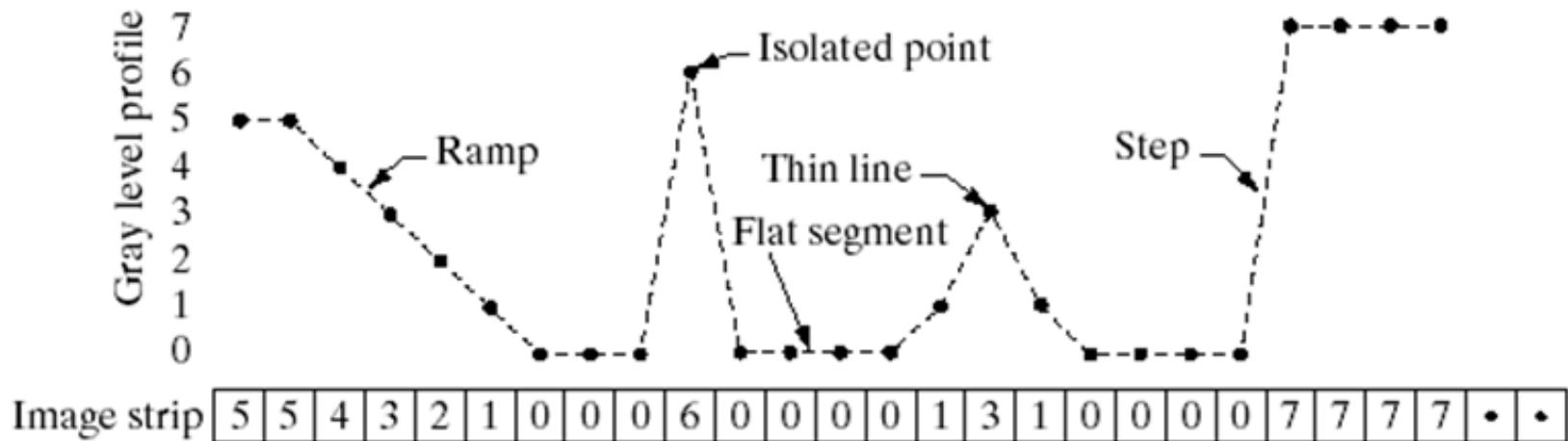
1st Order & 2nd Order Derivative



Scan line	6	6	6	6	5	4	3	2	1	1	1	1	1	1	6	6	6	6	$\rightarrow x$
1 st derivative	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0	0	5	0	0	0	
2 nd derivative	0	0	-1	0	0	0	0	0	1	0	0	0	0	0	5	-5	0	0	

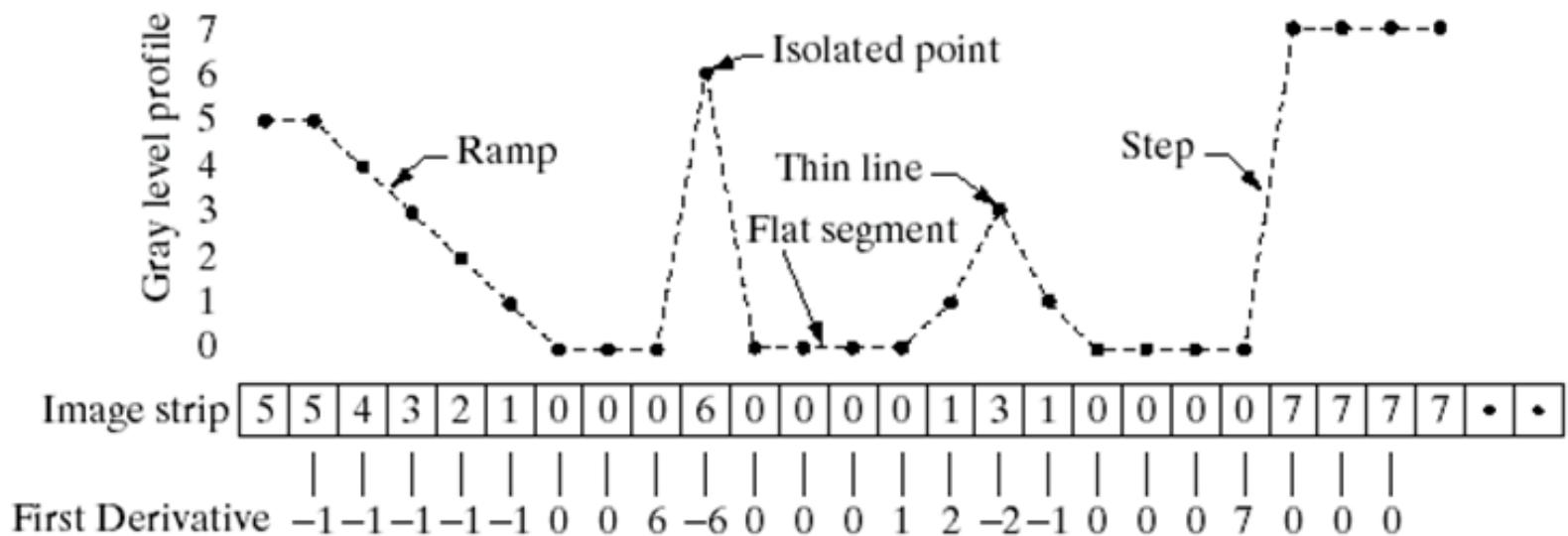


1st Order & 2nd Order Derivative



1st Order & 2nd Order Derivative

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



1st Order & 2nd Order Derivative

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

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

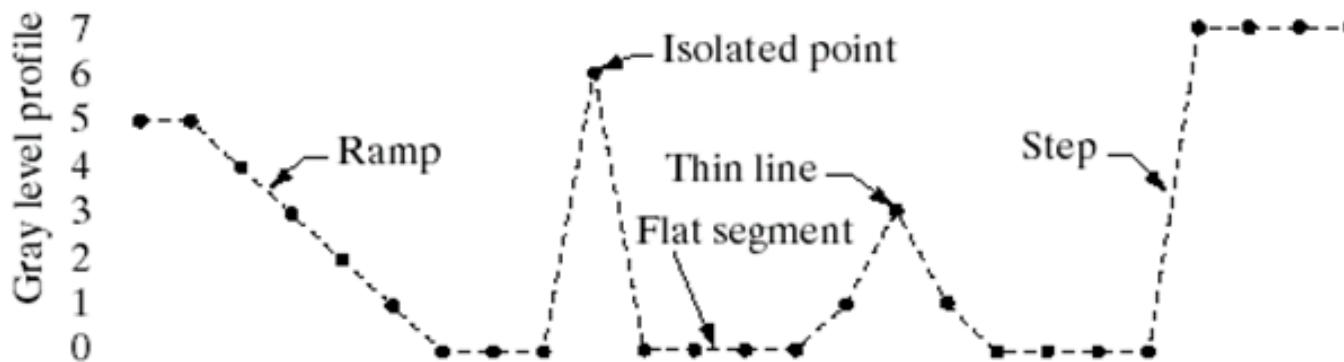


Image strip [5 | 5 | 4 | 3 | 2 | 1 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 1 | 3 | 1 | 0 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | • | •]

First Derivative | -1 | -1 | -1 | -1 | -1 | 0 | 0 | 6 | -6 | 0 | 0 | 0 | 1 | 2 | -2 | -1 | 0 | 0 | 0 | 7 | 0 | 0 | 0 |

Second Derivative | -1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 6 | -12 | 6 | 0 | 0 | 1 | 1 | -4 | 1 | 1 | 0 | 0 | 7 | -7 | 0 | 0 |



THANK YOU!!!

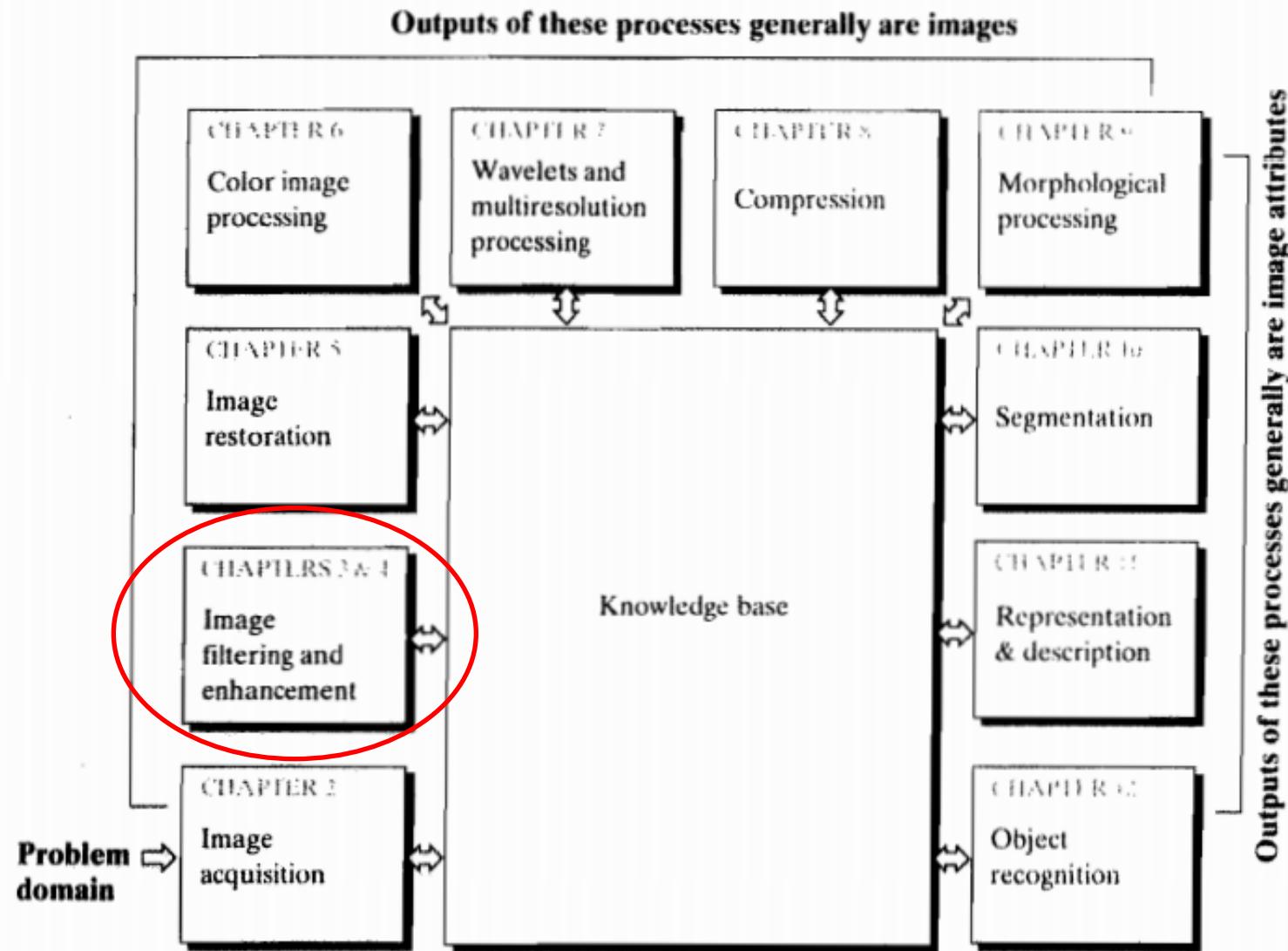


Chapter 9

Morphological Processing

Remember?

FIGURE 1.23
Fundamental steps in digital image processing. The chapter(s) indicated in the boxes is where the material described in the box is discussed.



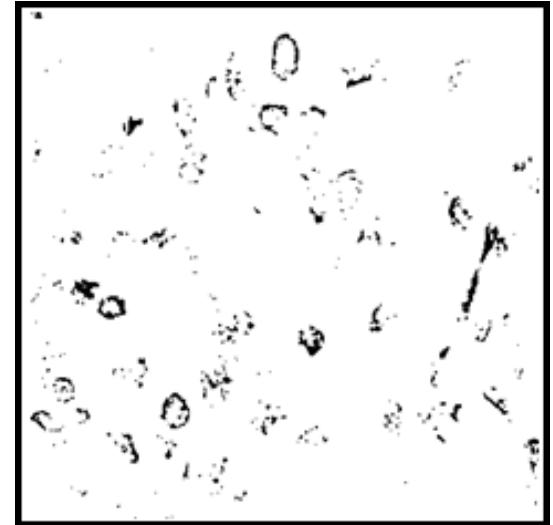
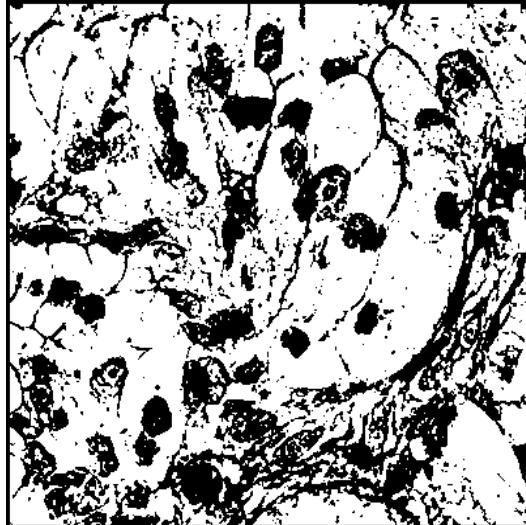
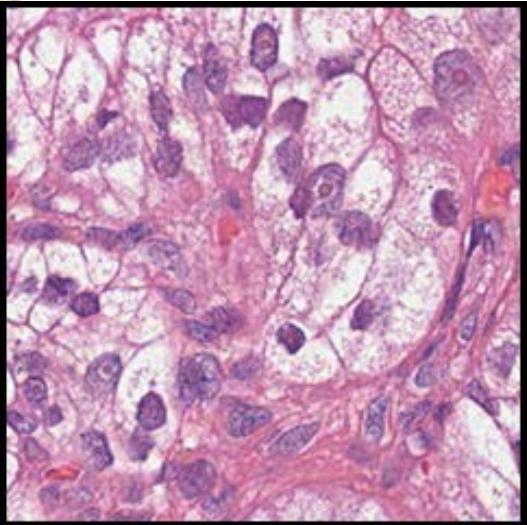
Outputs of these processes generally are image attributes

Morphological Processing

- Binary images may contain numerous imperfections.
- Morphological image processing pursues the goals of removing these imperfections by accounting for the **form and structure** of the image.

Morphological Processing

- Binary images may contain numerous imperfections. In particular, the binary regions produced by simple thresholding are distorted by noise and texture.



Basic concepts

- Morphological image processing is a collection of non-linear (non-sequential) operations related to the **shape or morphology of features** in an image.
- Morphological operations rely only on the relative ordering of pixel values and therefore are especially suited to the processing of binary images.
- Morphological techniques probe an image with a small shape or template called a **structuring element**.

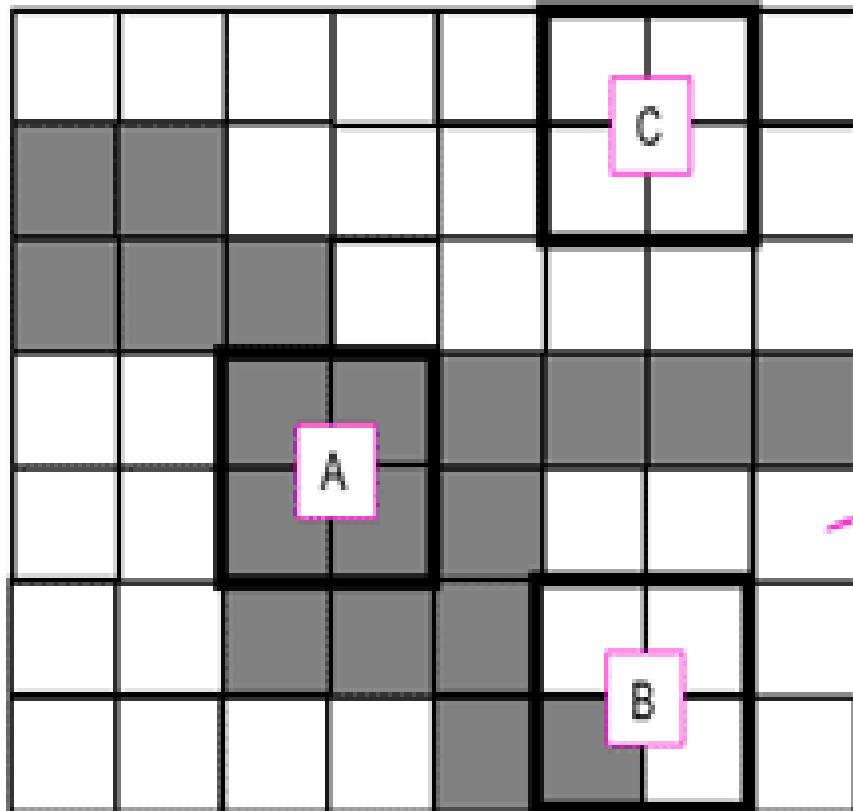
Basic concepts

- The structuring element is positioned at all possible locations in the image and it is compared with the corresponding neighborhood of pixels.
- Some operations test whether the element "**fits**" within the neighborhood, while
- others test whether it "**hits**" or intersects the neighborhood:

Morphological Operations

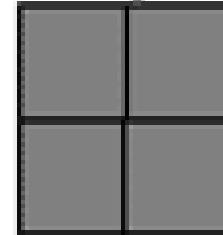
- The most basic morphological operations are **dilation** and **erosion**.
- In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbors.
- By choosing the size and shape of the neighborhood, you can construct a morphological operation that is sensitive to specific shapes in the input image

Basic concepts



- A - the structuring element fits the image
- B - the structuring element hits (intersects) the image
- C - the structuring element neither fits, nor hits the image

Structuring element



Probing of an image with a structuring element
(white and grey pixels have zero and non-zero values, respectively).

Basic concepts

- Structuring elements can be seen in 3 different ways in the test input image which are as follows:
 - 1) Structuring element **fits** the image region
 - 2) Structuring element **hits** (intersects) the image region
 - 3) Structuring element **neither fits nor hits** the image region

Structuring Element

- Morphological techniques probe an image with a small shape or template called a **structuring element**.

- The **structuring element** is a small binary image, i.e. a small matrix of pixels, each with a value of zero or one:

Structuring Element

What is matrix of pixels???

- The matrix dimensions specify the ***size*** of the structuring element.
- The pattern of ones and zeros specifies the ***shape*** of the structuring element.

Structuring Element

- An essential part of the dilation and erosion operations is the structuring element used to probe the input image.
- A structuring element is a matrix consisting of only 0's and 1's that can have any arbitrary shape and size.
- The pixels with values of 1 define the neighborhood.

Structuring Element

Structuring Element

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

Origin

Structuring Element

How to Calculate the Origin of a Structuring Element

Equation can be written as:

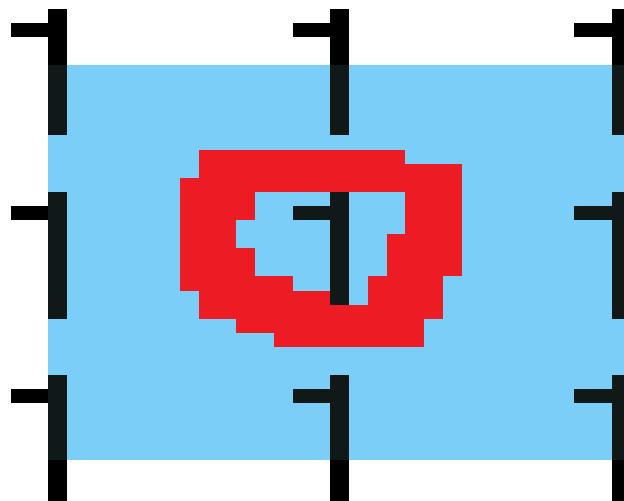
$$\text{origin} = \text{floor}((\text{size}+1)/2)$$

Lets say structuring element mask size is 3×3

Base on the equation;

$$\text{Origin} = \text{floor}((3+1)/2) = 2$$

The position will be $(2,2)$ if initial Position is considered as $(1,1)$



Structuring Element

There are two main characteristics that are directly related to structuring elements:

- Shape and
- Size

Structuring Element

Shape

- For example, the structuring element can be a circle, square, diamond etc.
- By choosing a particular structuring element, one sets a way of differentiating some objects (or parts of objects) from others, according to their shape or spatial orientation.

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

Structuring Element

Size

- For example, one structuring element can be a 3 x 3 square or a 21 x 21 square.
- Setting the size of the structuring element is similar to setting the observation scale, and setting the criterion to differentiate image objects or features according to size.

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

Square 5x5 element

Structuring Element

Structuring elements contains different shapes...

1) Square

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

Square 5x5 element

Structuring Element

2) Diamond

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

Diamond-shaped 5x5 element

Structuring Element

3) Disk

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

Structuring Element

4) Cross

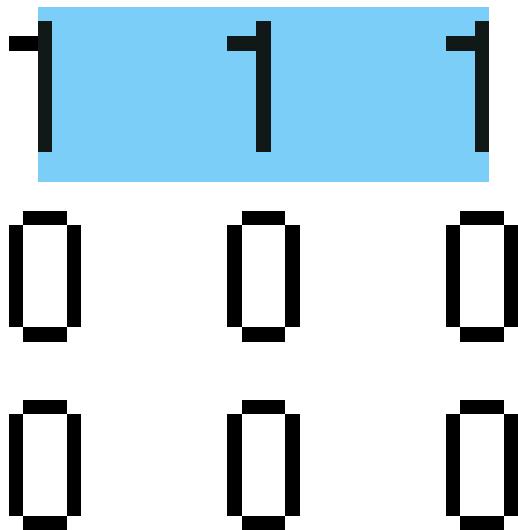
0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

Cross-shaped 5x5 element

Structuring Element

5) Rectangle

1	0	0
1	0	0
1	0	0



Structuring Element

6) Irregular

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

Morphological Operation

Dilation and Erosion

Dilation and erosion are two fundamental morphological operations.

- ***Dilation adds*** pixels to the boundaries of objects in an image, while
- ***Erosion removes*** pixels on object boundaries.

The number of pixels added or removed from the objects in an image depends on the size and shape of the ***structuring element*** used to process the image.

Morphological Operation

Characteristics of Erosion

- Erosion generally decreases the sizes of objects and removes small anomalies by subtracting objects with a radius smaller than the structuring element.

- With grayscale images, erosion reduces the brightness (and therefore the size) of bright objects on a dark background by taking the neighborhood minimum when passing the structuring element over the image.

Morphological Operation

Characteristics of Erosion

- With binary images, erosion completely removes objects smaller than the structuring element and removes perimeter pixels from larger image objects (the continuous line forming the boundary of a closed geometrical figure.)

Morphological Operation

Characteristics of Dilation

➤ Dilation generally increases the sizes of objects, filling in holes and broken areas, and connecting areas that are separated by spaces smaller than the size of the structuring element.

Morphological Operation

Characteristics of Dilation

- With grayscale images, dilation increases the brightness of objects by taking the neighborhood maximum when passing the structuring element over the image.

- With binary images, dilation connects areas that are separated by spaces smaller than the structuring element and adds pixels to the perimeter of each image object.

Erosion

The **erosion** of a binary image f by a structuring element s (denoted $f \Theta s$) produces a new binary image $g = f \Theta s$ with ones in all locations (x,y) of a structuring element's origin at which that structuring element s **fits** the input image f ,

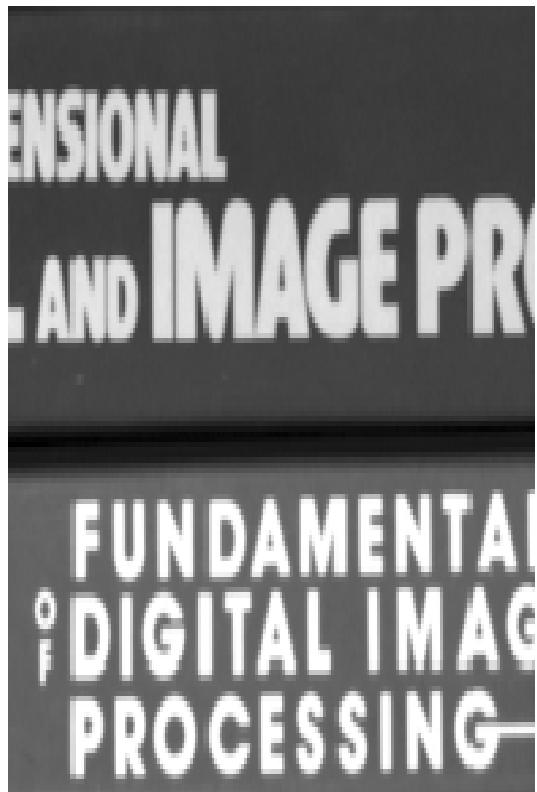
i.e. $g(x,y) = 1$ if s fits f

and

0 otherwise,

repeating for all pixel coordinates (x,y) .

Erosion



Greyscale image



Binary image by thresholding



Erosion: a 2×2 square structuring element

Erosion

- Erosion with small (e.g. 2×2 - 5×5) square structuring elements shrinks an image by stripping away a layer of pixels from both the inner and outer boundaries of regions.
- The holes and gaps between different regions become larger, and small details are eliminated.

Erosion

- The value of the output pixel is the *minimum* value of all the pixels in the input pixel's neighborhood.
- In a binary image, if any of the pixels is set to 0, the output pixel is set to 0.

Erosion

Morphological Erosion of a Binary Image

Rules to follow:

- 1) Fully match $\rightarrow 1$
- 2) Some match $\rightarrow 0$
- 3) No match $\rightarrow 0$

Let's solve this for Erosion using 3 x 3 square Structuring element

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

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

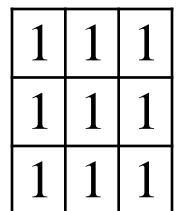
Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

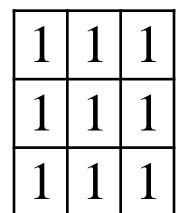
Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding



Red 0's are used as zero padding



Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

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

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

Red 0's are used as zero padding

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

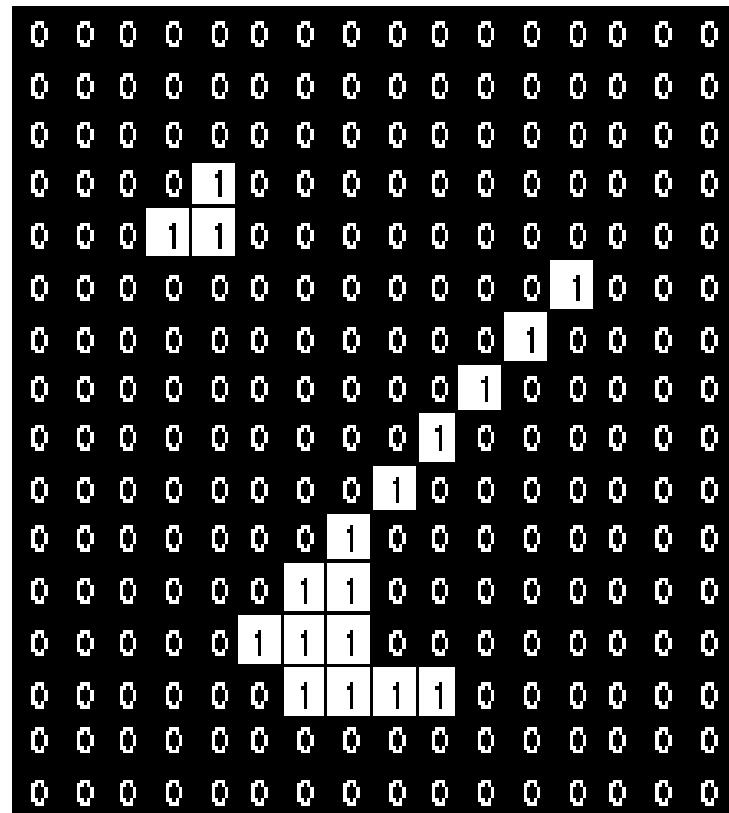
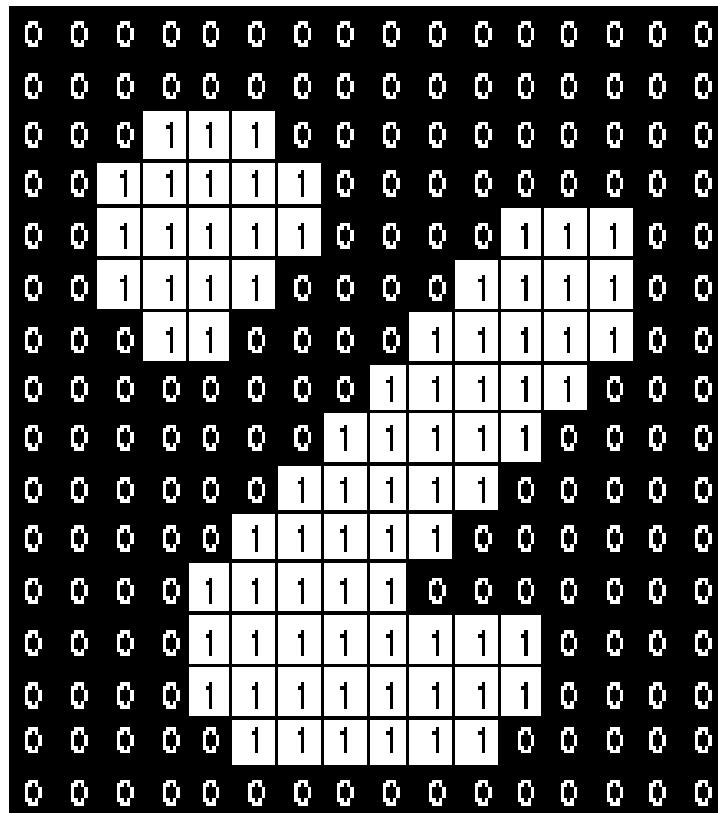
Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Erosion



Erosion: a 3×3 square structuring element

Let's check the result

Erosion



Try this example and solve if you can

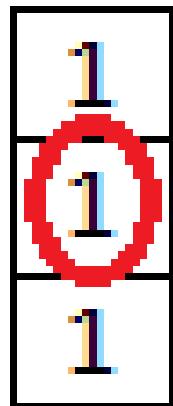
0	0	0	0	0	0
0	0	1	1	0	0
0	1	1	1	1	0
0	0	1	1	0	0
0	0	0	0	0	0

1
1
1

Structuring Element

Test Image

Erosion



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

Erosion

1					
1					
1					

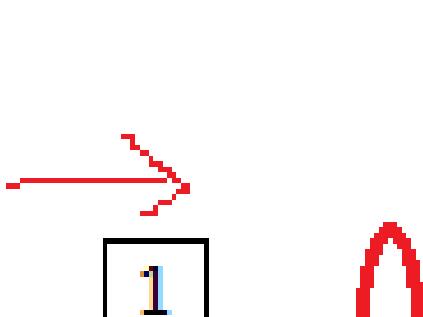


0	0	0	0	0	0
0	0	1	1	0	0
0	1	1	1	1	0
0	1	1	1	1	0
0	0	1	1	0	0
0	0	0	0	0	0

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

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

Erosion



1		
1		
1		

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

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

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

Erosion



1
1
1

A 6x6 binary image representing a shape. The shape is defined by the following values:

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

The result of applying a 3x3 erosion kernel to the input image. The kernel is defined by the following values:

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

The result of applying a larger 5x5 erosion kernel to the input image. The kernel is defined by the following values:

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

Erosion



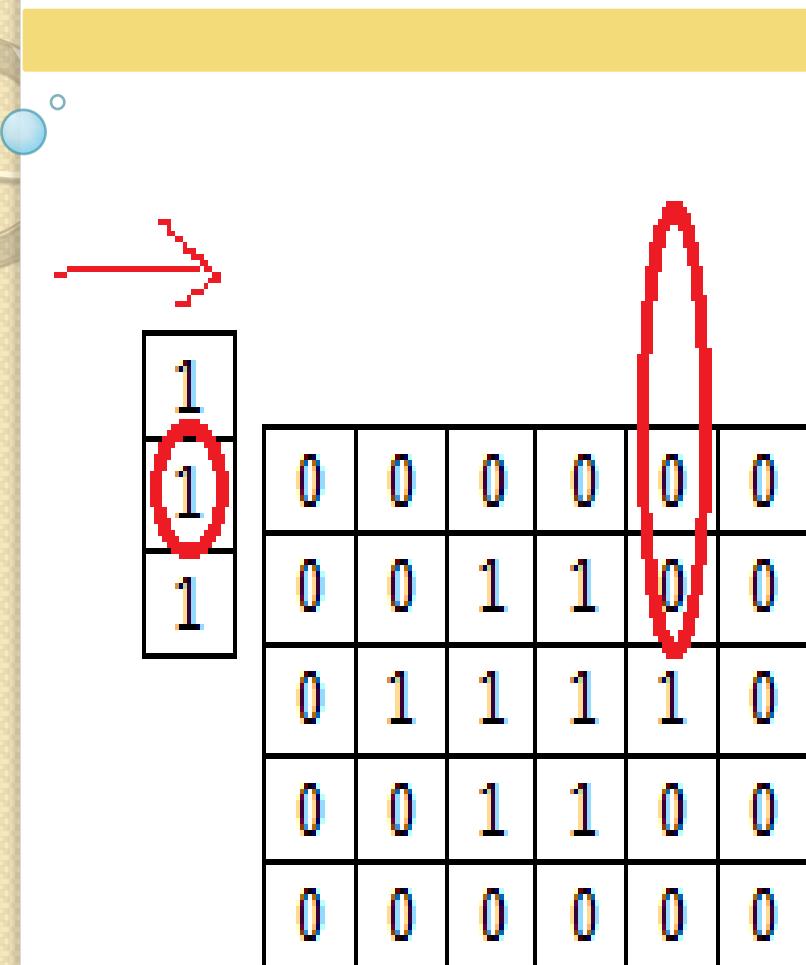
1					
1					
1					

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

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

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

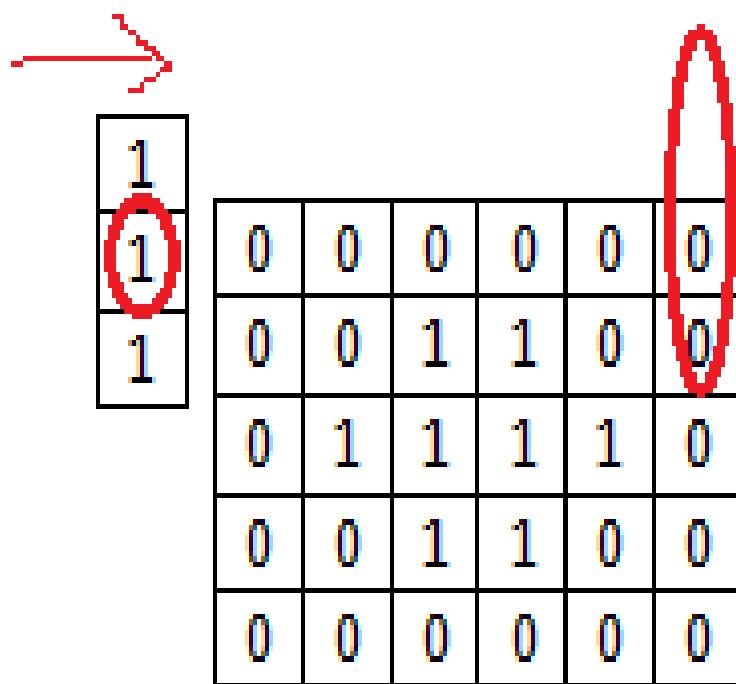
Erosion



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

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

Erosion



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

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

Erosion



1
1
1

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

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

Erosion



1	
1	
1	
1	

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

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

Erosion



1		
1		
1		

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

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

Erosion

1
1
1

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

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

Erosion

1
1
1

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



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

Erosion

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

Erosion

Simple and easy way to solve

Rules:

- 1) Use zero (0) padding for each side.
- 2) Place the Structuring element and get the result straightway Using the neighbors and erosion rule.
- 3) Make sure you only update values for origin only.

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

Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

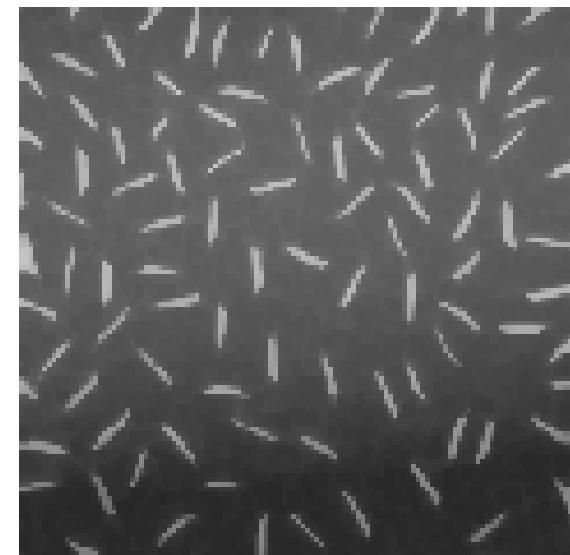
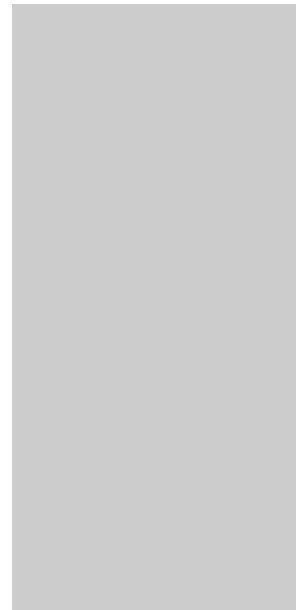
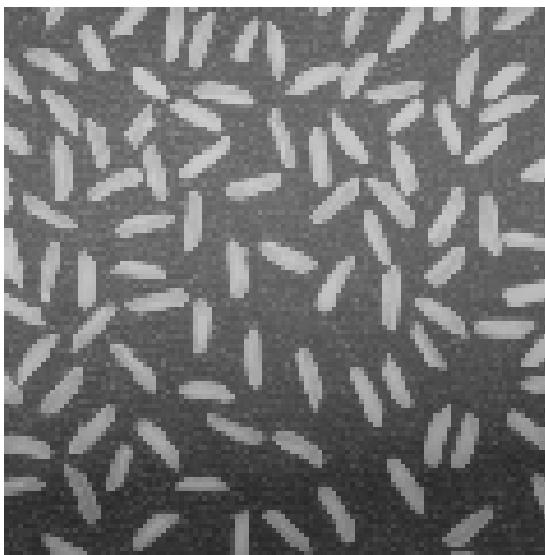
Red 0's are used as zero padding

1	1	1
1	1	1
1	1	1

Erosion

```
1) clc,  
2) close all;  
3) clear all;  
4)  
5) I = imread('D:\matlab_folder\week_9.png');  
6) figure, imshow(I);  
7) SE = strel('square',5);  
8)  
9) J = imerode(I,SE);  
10) figure, imshow(J);
```

Erosion



Erosion

Just change line number 7 for the previous slide and play with different shapes of structuring element

-> SE = strel('diamond',5);

OR -> SE = strel('disk',5);

OR -> SE = strel('octagon',5);

OR -> SE = strel('line',len,deg); %len =lenth , deg = degree

OR -> SE = strel('rectangle',[m n]);

OR -> SE = strel('cube',5);

OR -> SE = strel('square',5);

Erosion

Larger structuring elements have a more pronounced effect, the result of erosion with a large structuring element being similar to the result obtained by iterated erosion using a smaller structuring element of the same shape.

If s_1 and s_2 are a pair of structuring elements identical in shape, with s_2 twice the size of s_1 ,

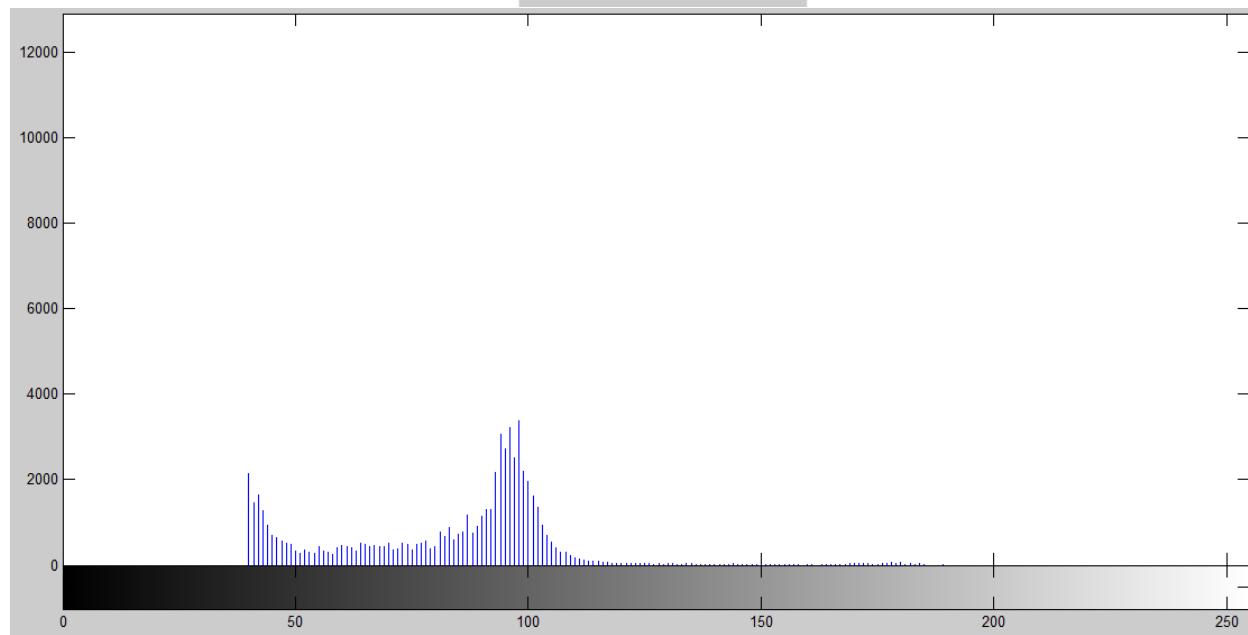
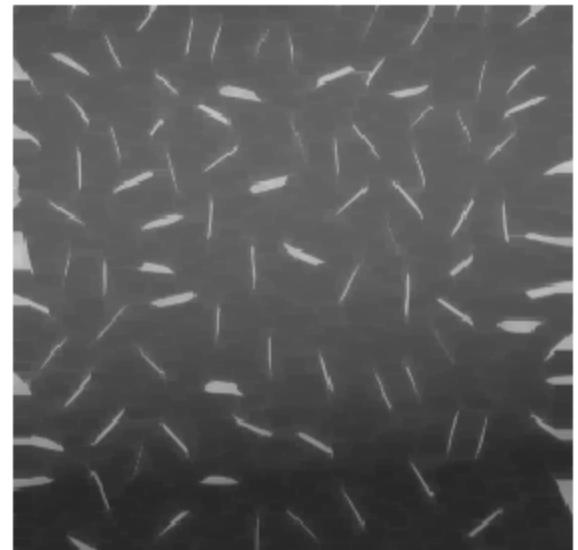
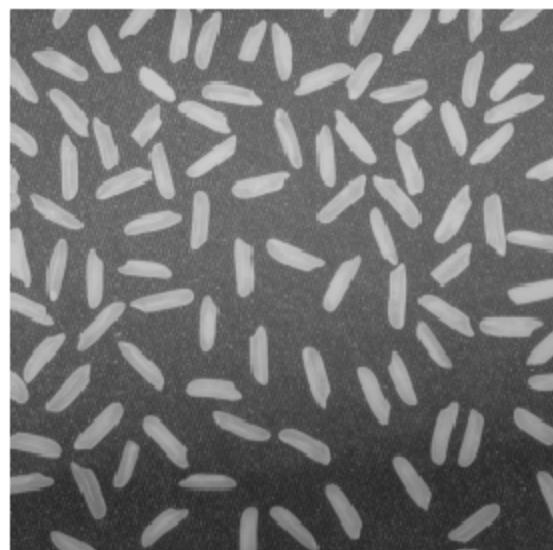
then

$$f \Theta s_2 \approx (f \Theta s_1) \Theta s_1.$$

Erosion

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
figure, imshow(I);  
SE = strel('rectangle',[4 8]);  
J = imerode(I,SE);  
%J = imerode(J,SE);  
figure, imshow(J);
```

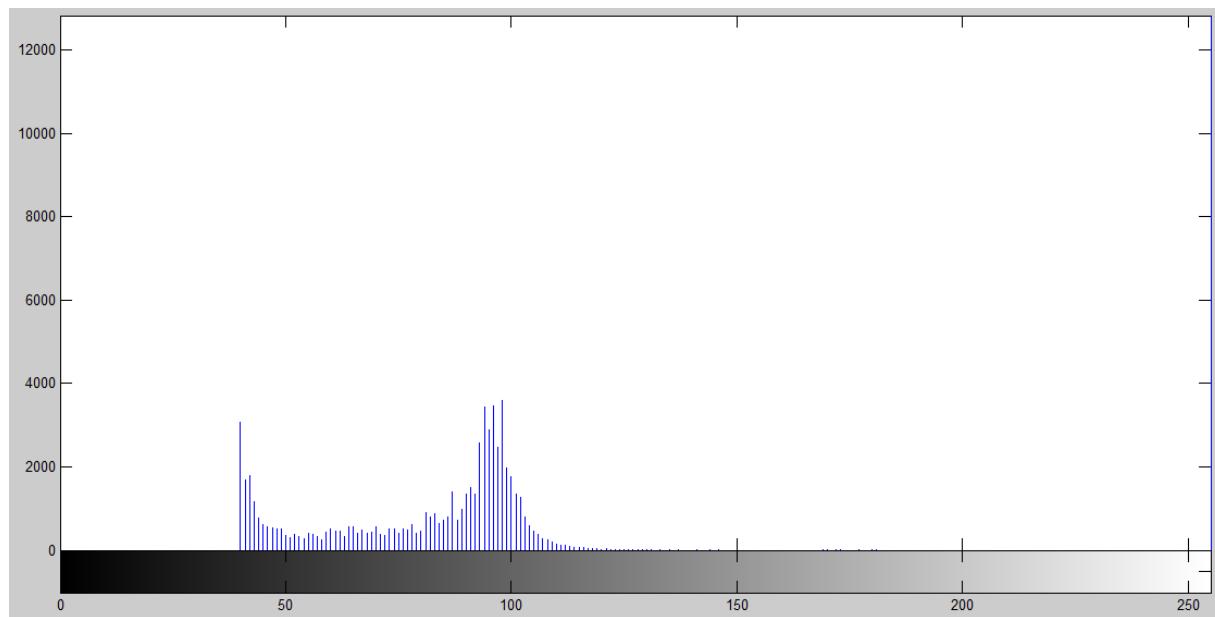
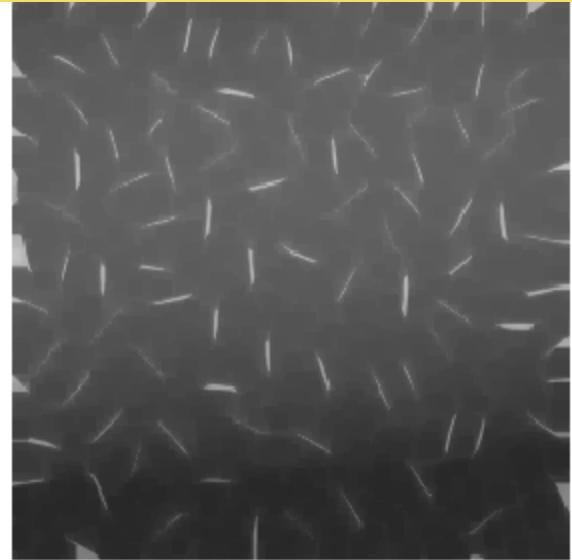
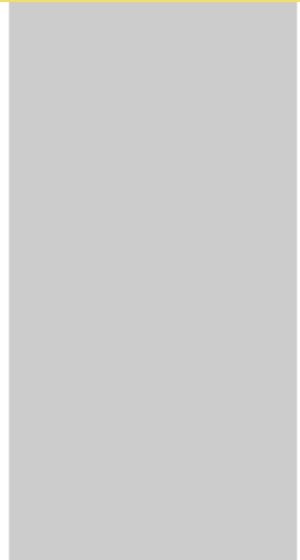
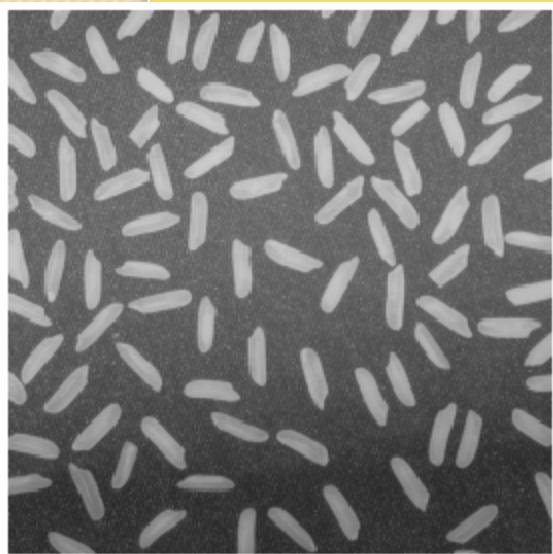
Erosion



Erosion

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
figure, imshow(I);  
SE = strel('rectangle',[4 4]);  
J = imerode(I,SE);  
J = imerode(J,SE);  
figure, imshow(J);
```

Erosion



Erosion

Erosion removes small-scale details from a binary image but simultaneously reduces the size of regions of interest, too.

By subtracting the eroded image from the original image, boundaries of each region can be found:

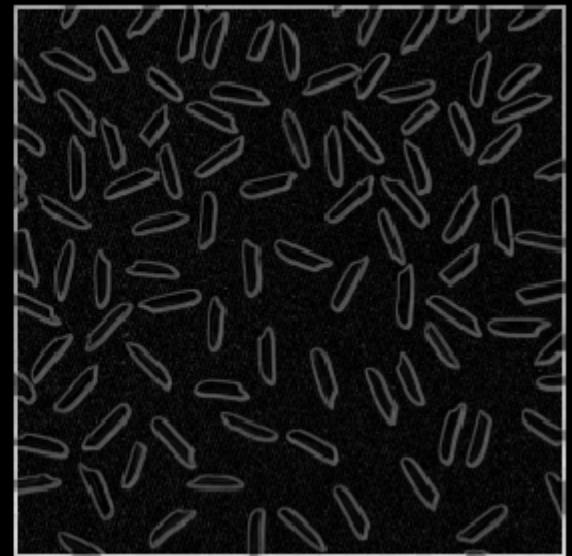
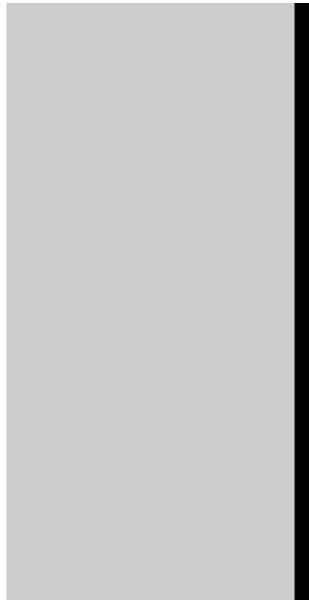
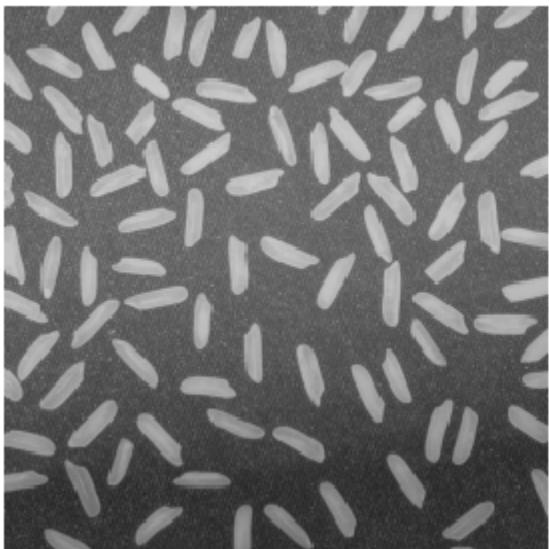
$$b = f - (f \Theta s)$$

where f is an image of the regions, s is a 3×3 structuring element, and b is an image of the region boundaries.

Erosion

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
figure, imshow(I);  
SE = strel('rectangle',[4 4]);  
J = imerode(I,SE);  
figure, imshow(J);  
K = I - J;  
figure, imshow(K);
```

Erosion



Dilation

- The **dilation** of an image f by a structuring element s (denoted $f \oplus s$) produces a new binary image $g = f \oplus s$ with ones in all locations (x,y) of a structuring element's origin at which that structuring element s hits the input image f ,
i.e. $g(x,y) = 1$ if s hits f
and
 0 otherwise, repeating for all pixel coordinates (x,y) .

Dilation has the opposite effect to erosion as it adds a layer of pixels to both the inner and outer boundaries of regions.

Dilation



Binary image



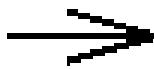
Dilation: a 2×2 square structuring element

Dilation

The value of the output pixel is the *maximum* value of all the pixels in the input pixel's neighborhood. In a binary image, if any of the pixels is set to the value 1, the output pixel is set to 1.

Dilation

- The holes enclosed by a single region and gaps between different regions become smaller, and small intrusions into boundaries of a region are filled in:



Dilation: a 3×3 square structuring element

Dilation and Erosion relation

Results of dilation or erosion are influenced both by the size and shape of a structuring element. Dilation and erosion are *dual* operations in that they have opposite effects. Let f^c denote the complement of an image f , i.e., the image produced by replacing 1 with 0 and vice versa. Formally, the duality is written as

$$f \oplus s = f^c \Theta s_{\text{rot}}$$

Dilation and Erosion relation

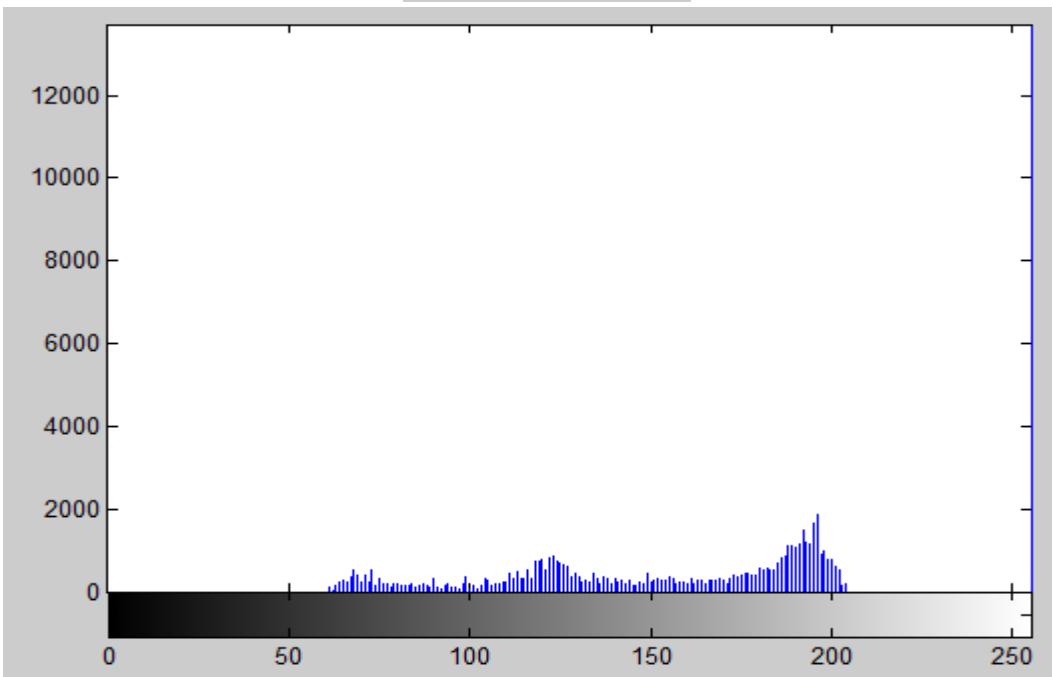
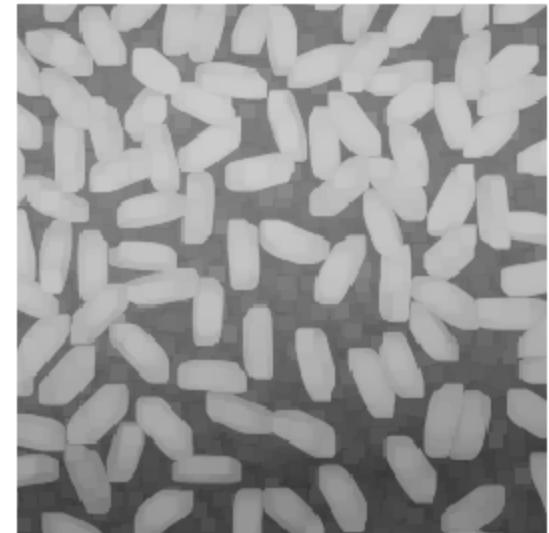
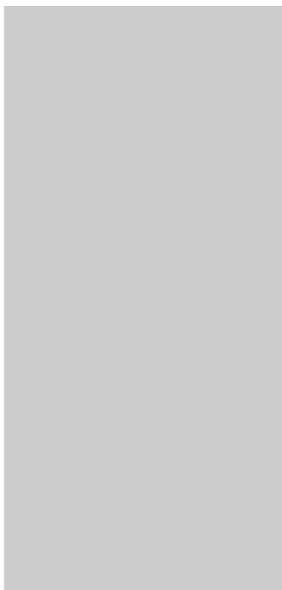
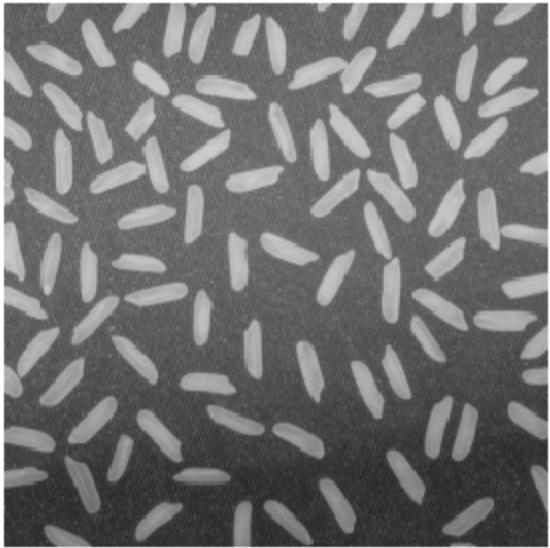
where s_{rot} is the structuring element s rotated by 180. If a structuring element is symmetrical with respect to rotation, then s_{rot} does not differ from s .

If a binary image is considered to be a collection of connected regions of pixels set to 1 on a background of pixels set to 0, then erosion is the fitting of a structuring element to these regions and dilation is the fitting of a structuring element (rotated if necessary) into the background, followed by inversion of the result.

Dilation and Erosion relation

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
figure, imshow(I);  
SE = strel('square',7);  
J = imdilate(I,SE);  
figure, imshow(J);
```

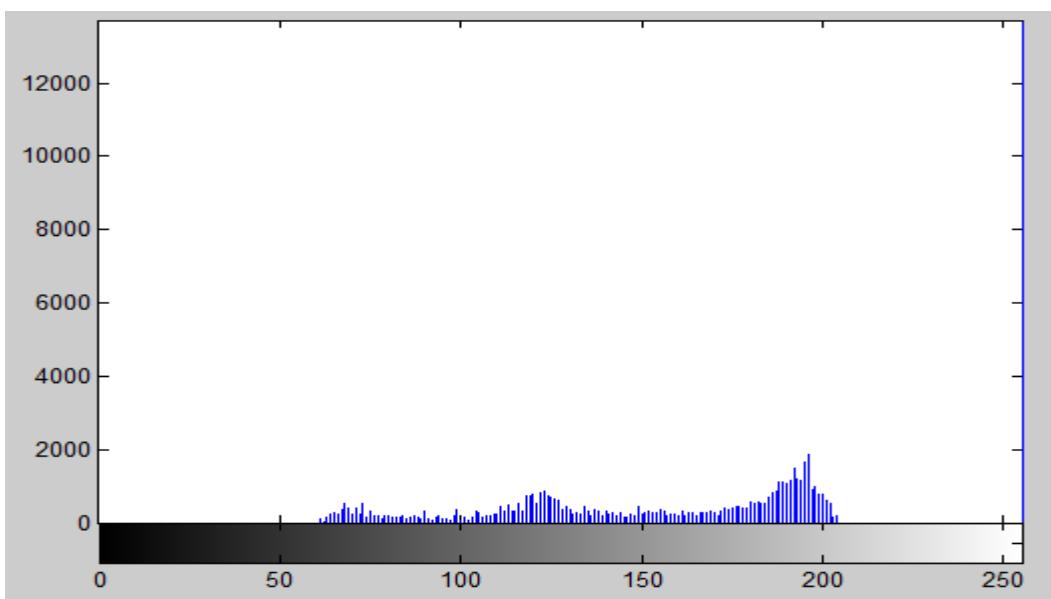
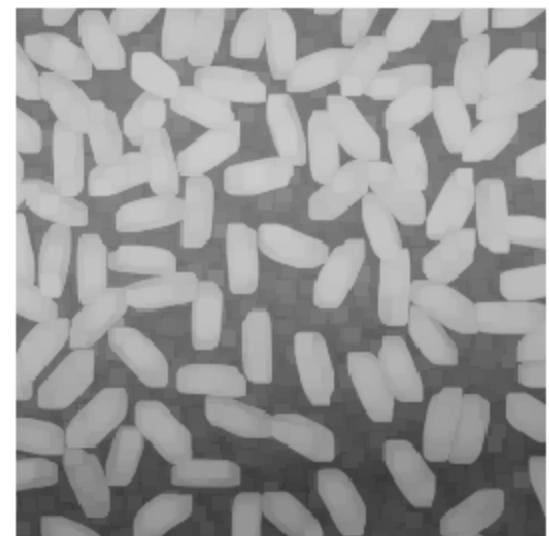
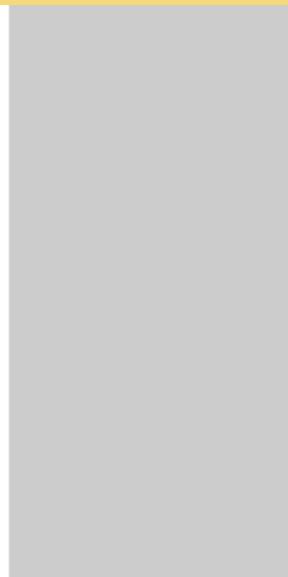
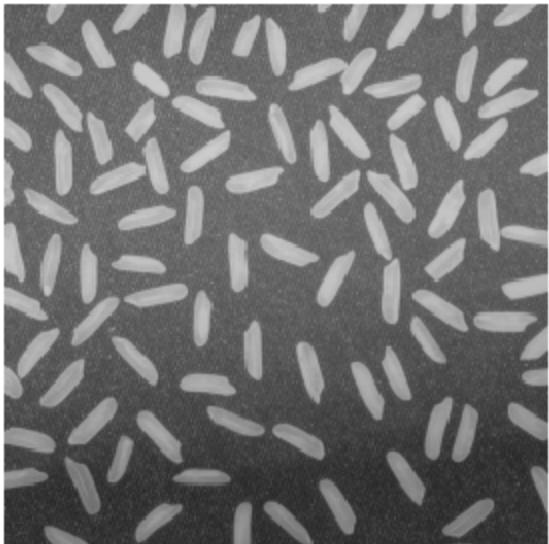
Dilation and Erosion relation



Dilation and Erosion relation

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
I1 = imcomplement(I);  
figure, imshow(I);  
SE = strel('square',7);  
J = imerode(I1,SE);  
J = imcomplement(J);  
figure, imshow(J);
```

Dilation and Erosion relation



Dilation

Morphological Dilation of a Binary Image

- 1) Fully match $\rightarrow 1$
- 2) Some match $\rightarrow 1$
- 3) No match $\rightarrow 0$

Let's try this with 3×3 matrix for Dilation

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

1	1	1
1	1	1
1	1	1

000000000000000000000000
000000000000000000000000
0001110000000000000000
0011110000000000000000
001111100000111100
001111000011111100
000110000111111100
000000001111110000

1	1	1
1	1	1
1	1	1

000000000000000000000000
000000000000000000000000
000111000000000000000000
001111100000000000000000
0011111100000111100
0011111000011111100
000110000111111100
000000001111111000

00000000000000000000
00000000000000000000
00011110000000000000
00111110000000000000
001111100000111100
001111000011111100
000110000111111100
000000001111110000

1	1	1
1	1	1
1	1	1

00000000000000000000
00111110000000000000
00011110000000000000
00111110000000000000
0011111000000000111100
001111100000000111111100
00011000011111111100
00000000111111111100

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

1	1	1
1	1	1
1	1	1

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

1	1	1
1	1	1
1	1	1

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

1	1	1
1	1	1
1	1	1

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

1	1	1
1	1	1
1	1	1

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

1	1	1
1	1	1
1	1	1

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

1	1	1
1	1	1
1	1	1

Dilation

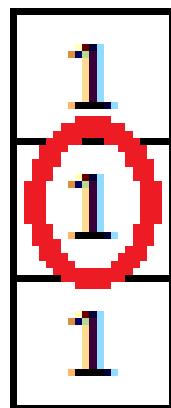


Try this and check your result

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

1
1
1

Dilation



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

Dilation

1					
1					
1					



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

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

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

Dilation



1		
1		
1		

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

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

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

Dilation



1
1
1

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

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

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

Dilation



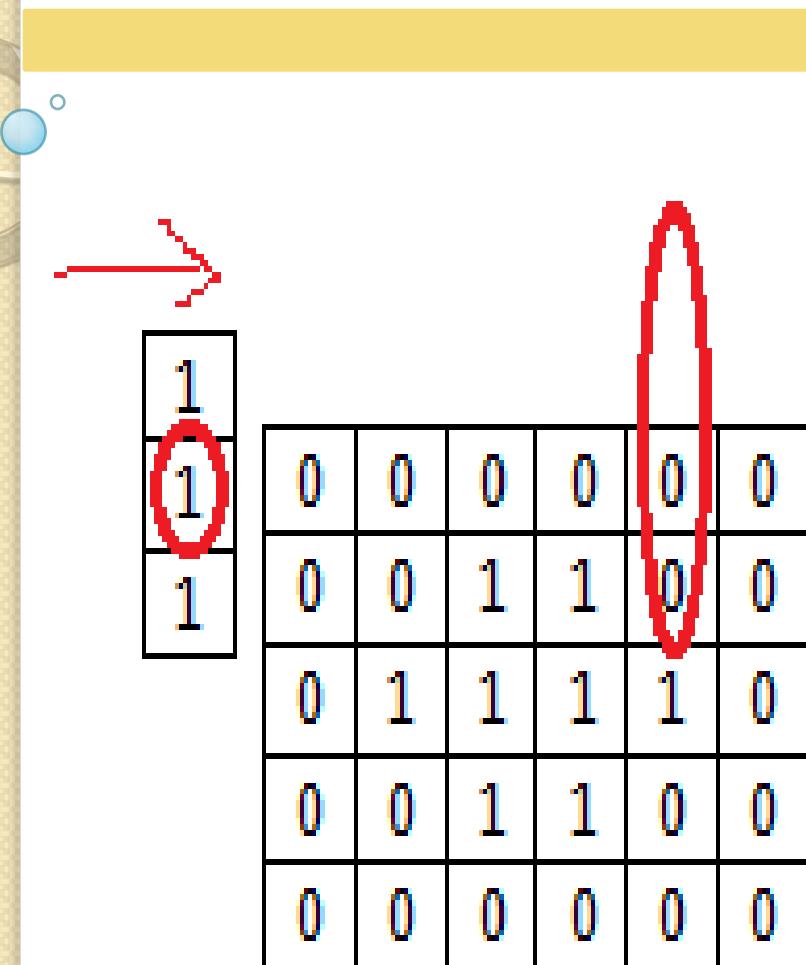
1		
1		
1		

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

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

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

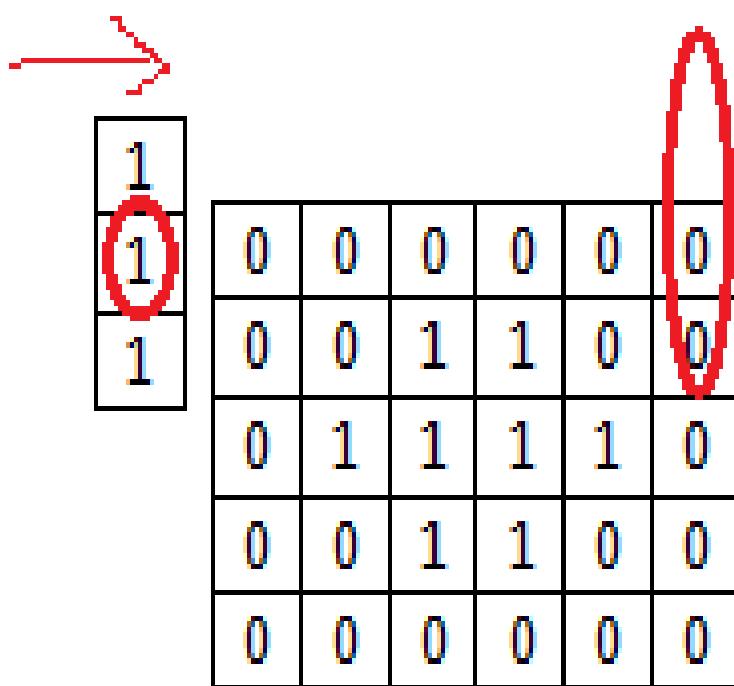
Dilation



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

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

Dilation



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

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

Dilation



1
1
1

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

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

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

Dilation



1	1	1
1	1	1
1	1	1
1	1	1

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

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

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

Dilation



1	
1	
1	

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

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

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

Dilation

1
1
1

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

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

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

Dilation

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

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

Dilation

Simple and easy way to solve

Rules:

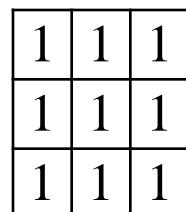
- 1) Use zero (0) padding for each side.
- 2) Place the Structuring element and get the result straightway Using the neighbors and dilation rule.
- 3) Make sure you only update values for origin only.

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



1	1	1
1	1	1
1	1	1



Morphological Opening

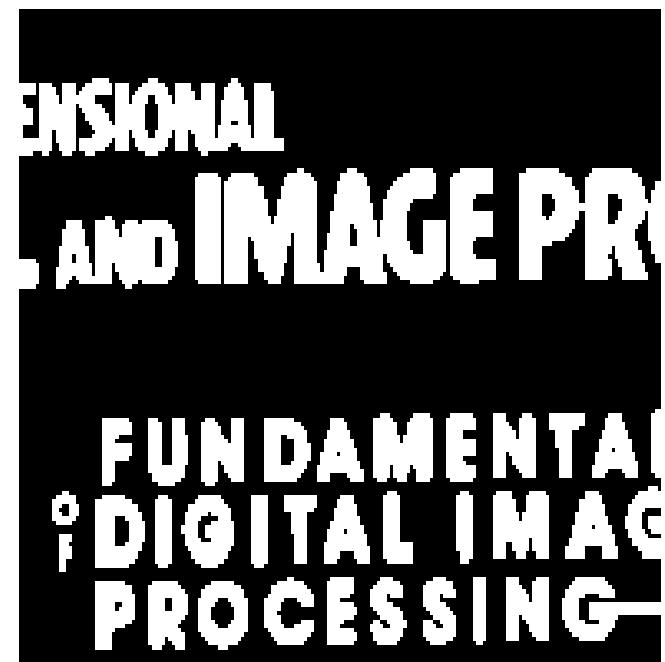
The **opening** of an image f by a structuring element s (denoted by $f \circ s$) is an erosion followed by a dilation:

$$f \circ s = (f \ominus s) \oplus s$$

Morphological Opening



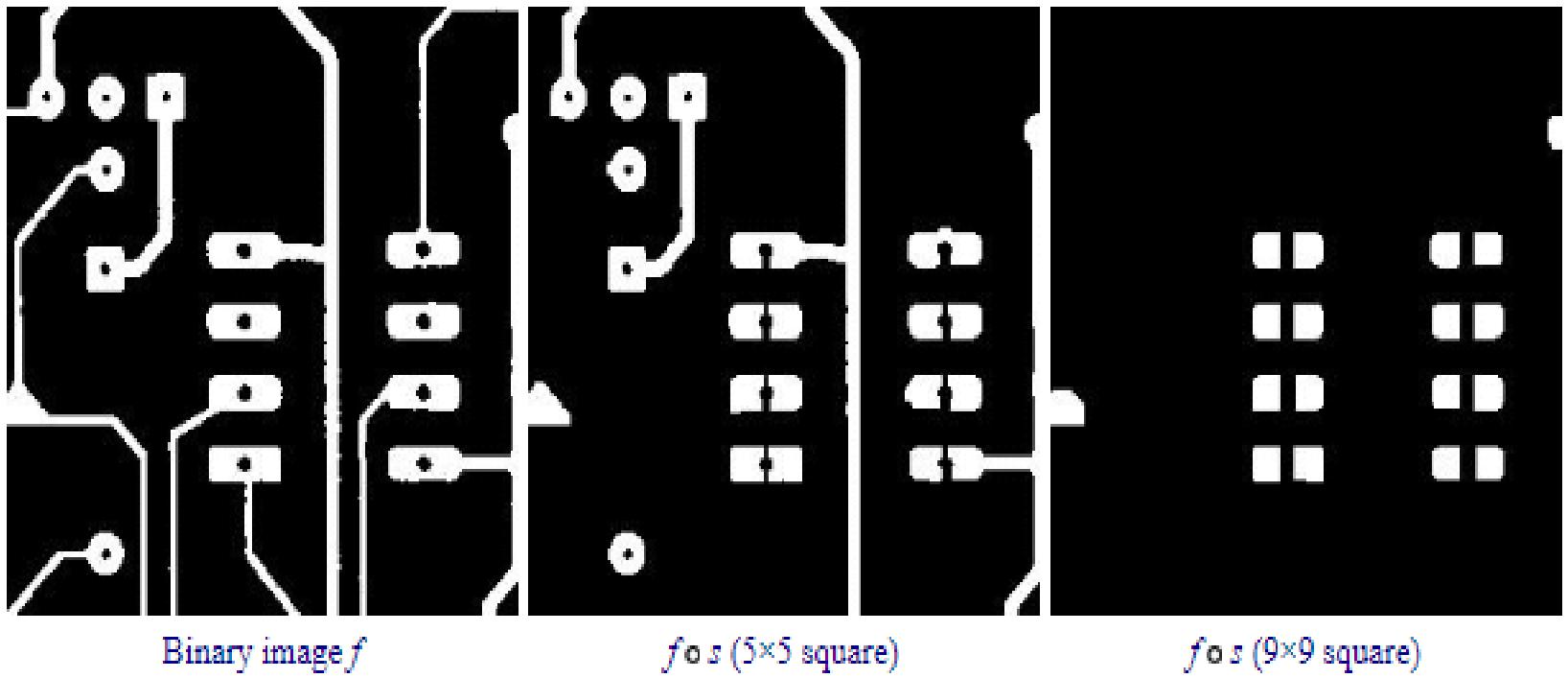
Binary image



Opening: a 2×2 square structuring element

Opening is so called because it can open up a gap between objects connected by a thin bridge of pixels. Any regions that have survived the erosion are restored to their original size by the dilation:

Morphological Opening



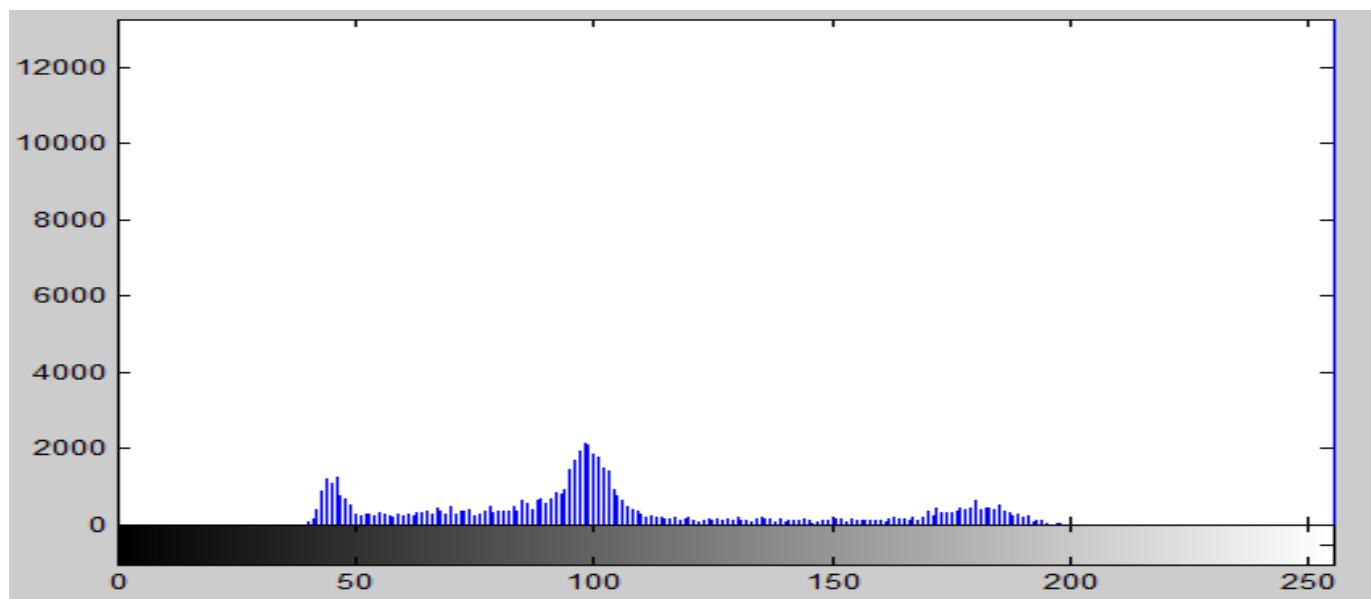
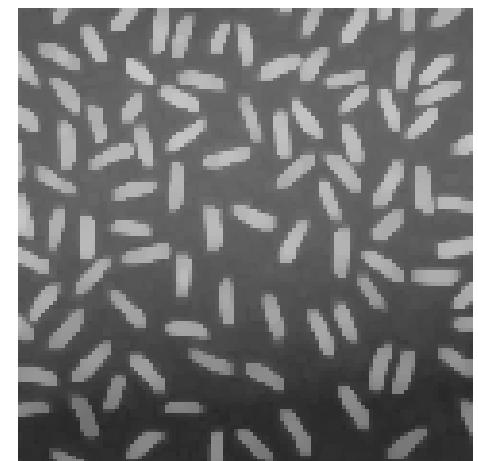
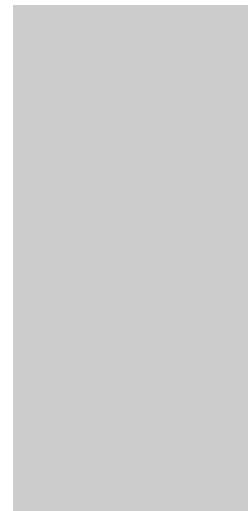
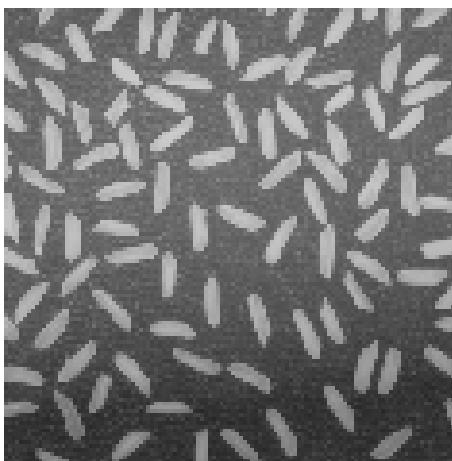
Opening is an **idempotent** operation: once an image has been opened, subsequent openings with the same structuring element have no further effect on that image:

$$(f \circ s) \circ s = f \circ s$$

Morphological Opening

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
figure, imshow(I);  
SE = strel('square',5);  
J = imopen(I,SE);  
figure, imshow(J);
```

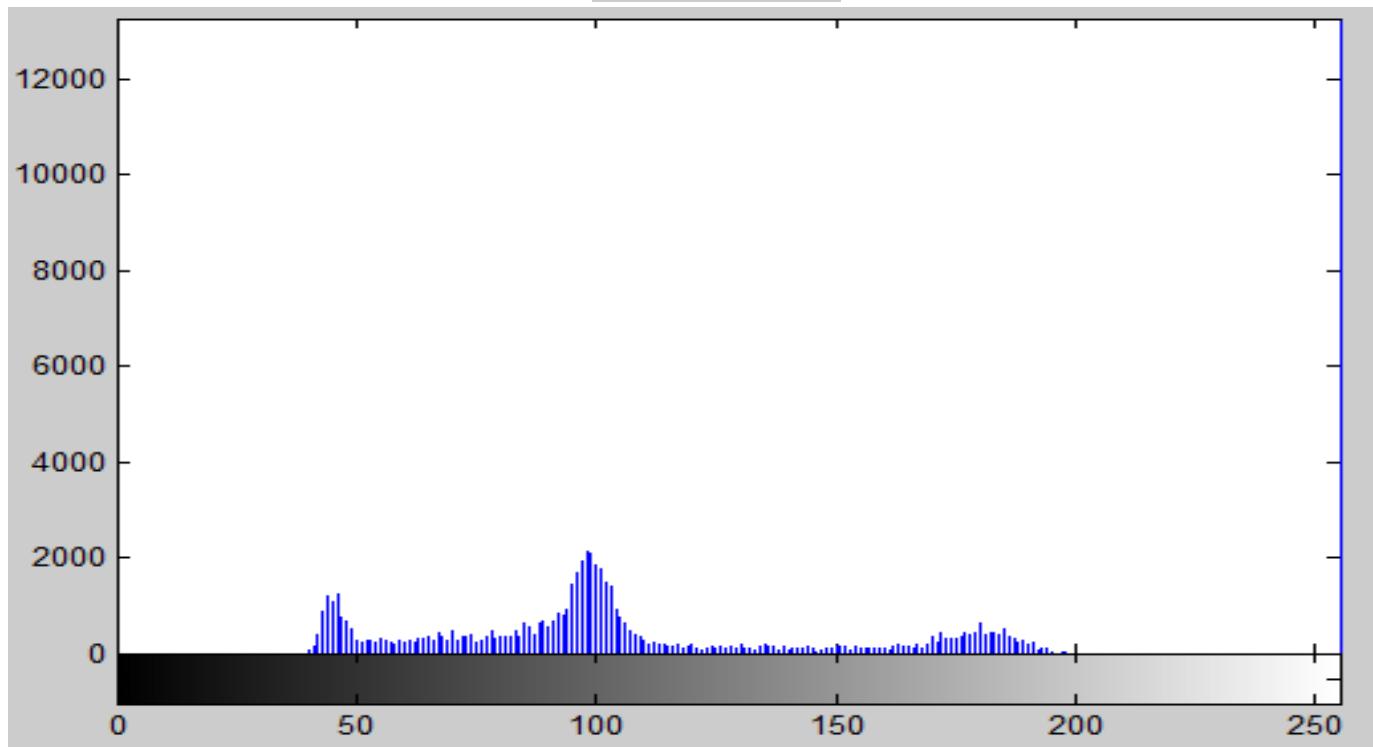
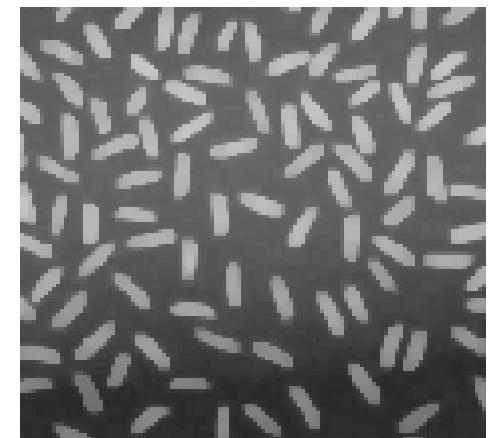
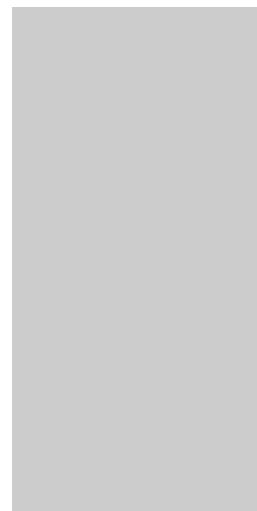
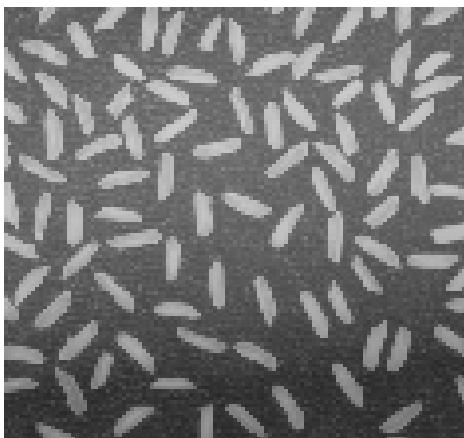
Morphological Opening



Morphological Opening

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
figure, imshow(I);  
SE = strel('square',5);  
J = imerode(I,SE);  
J = imdilate(J,SE);  
figure, imshow(J);
```

Morphological Opening



Morphological Closing

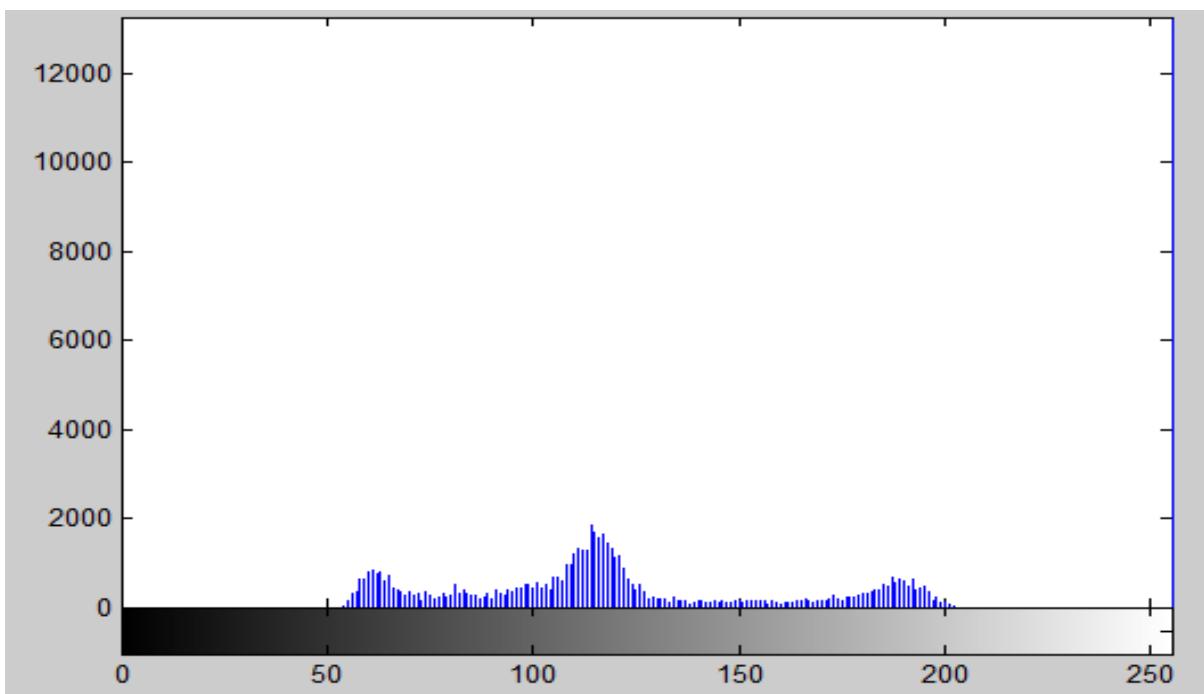
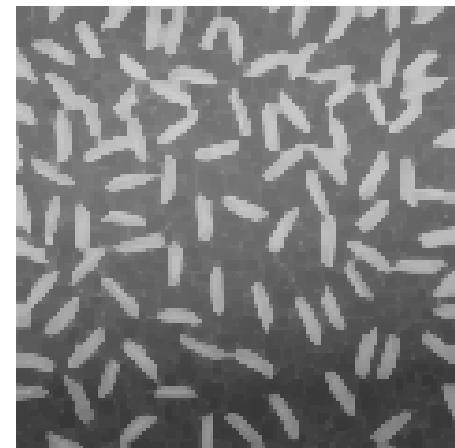
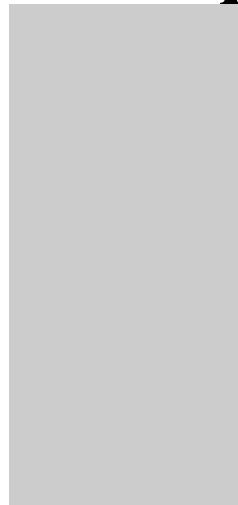
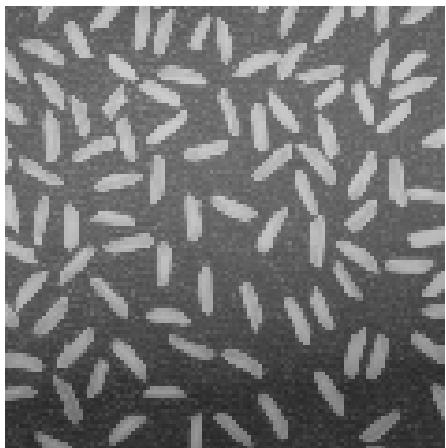
The **closing** of an image f by a structuring element s (denoted by $f \bullet s$) is a dilation followed by an erosion:

$$f \bullet s = (f \oplus s_{\text{rot}}) \ominus s_{\text{rot}}$$

Morphological Closing

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
figure, imshow(I);  
SE = strel('square',5);  
J = imclose(I,SE);  
figure, imshow(J);
```

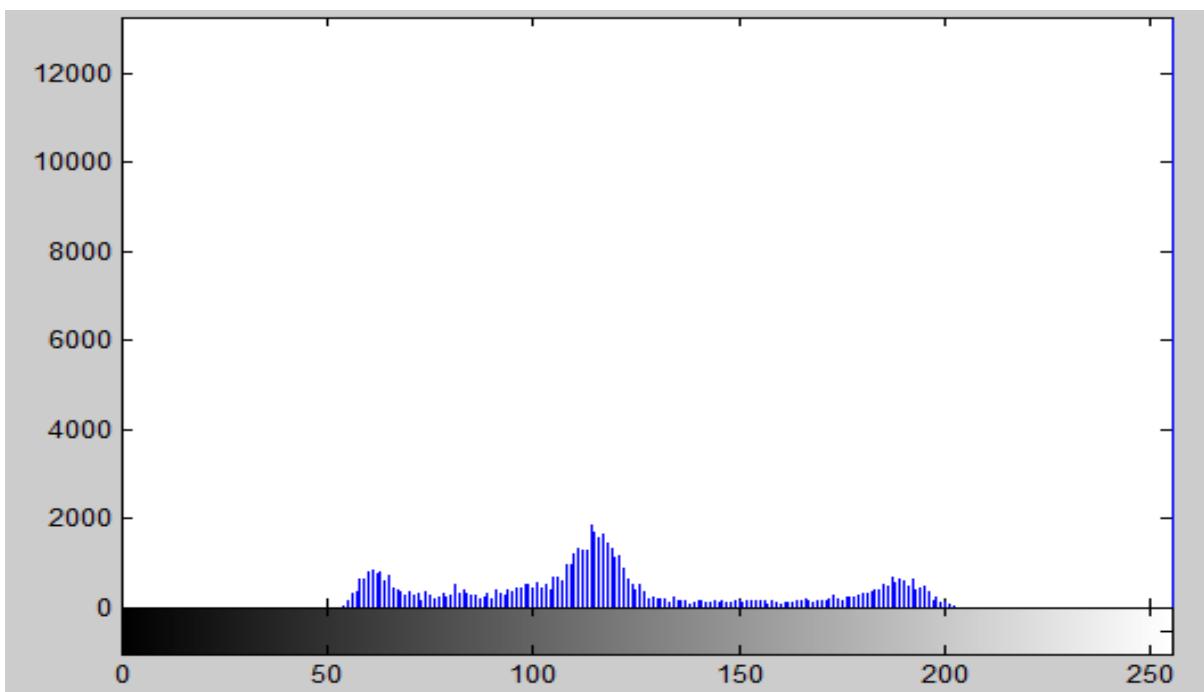
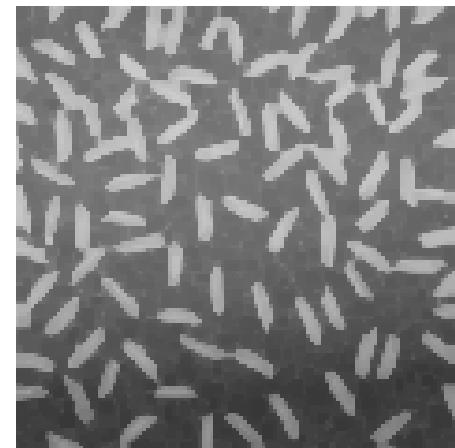
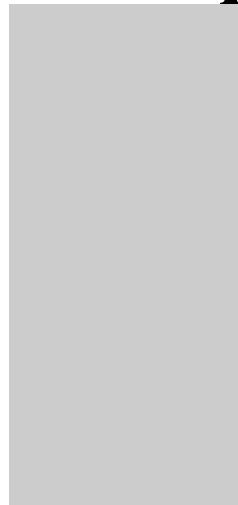
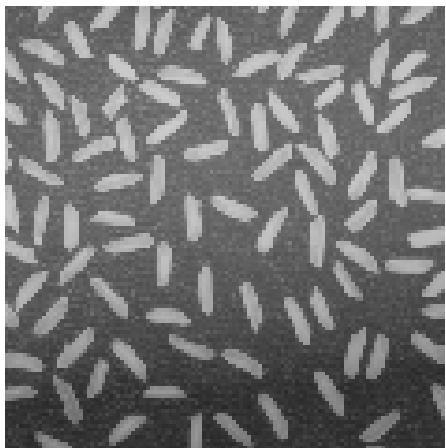
Morphological Closing



Morphological Closing

```
clc,  
%close all;  
clear all;  
I = imread('D:\matlab_folder\2.jpg');  
figure, imshow(I);  
SE = strel('square',11);  
J= imdilate(I,SE);  
J = imerode(J,SE);  
figure, imshow(J);
```

Morphological Closing



Morphological Filtering

There can be two types of morphological filtering possible which are:

- 1) Top-hat filtering
- 2) Bottom-hat filtering

Morphological Filtering

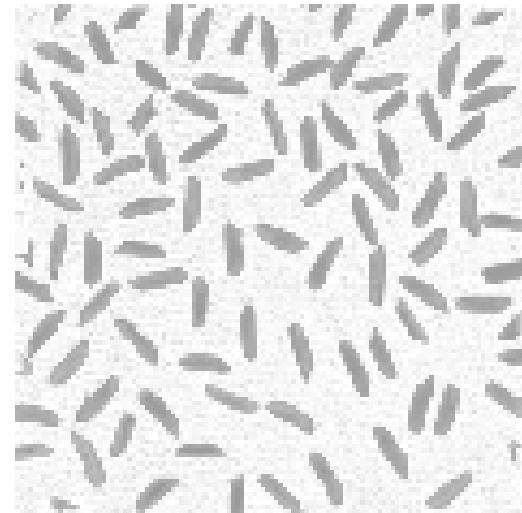
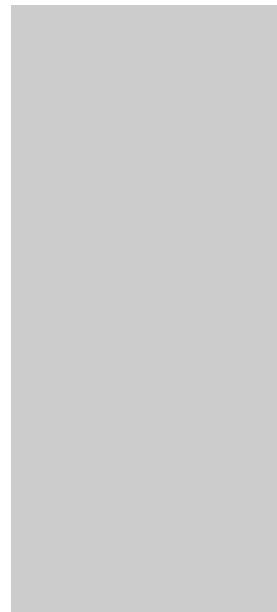
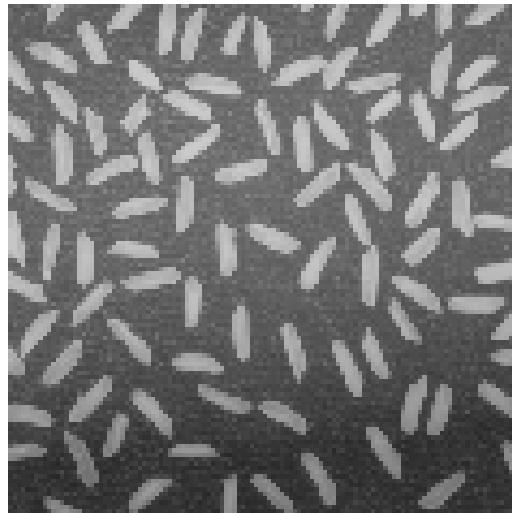
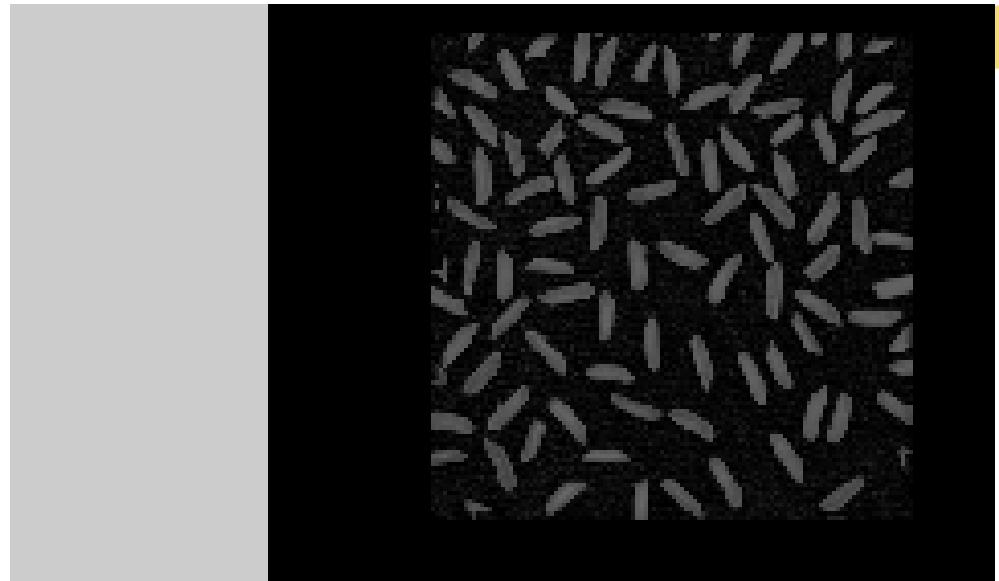
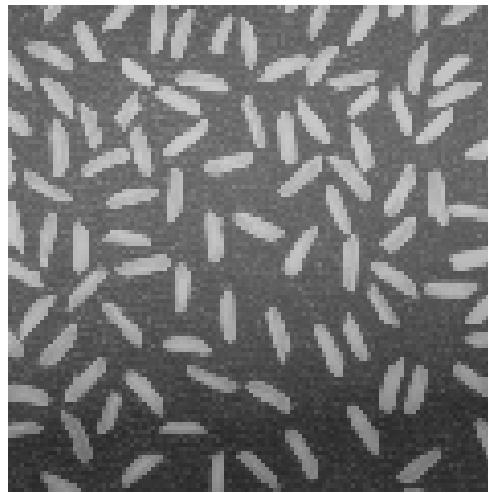
Top-hat filtering:

Morphological top-hat filtering performs on the grayscale or binary image and returns a filtered image. Top-hat filtering computes the morphological opening of the image and then subtracts the result from the original image.

Morphological Filtering

```
clc,  
close all;  
clear all;  
I = imread('D:\matlab_folder\Dig1.png');  
figure, imshow(I);  
SE = strel('square',11);  
J = imtophat(I,SE);  
figure, imshow(J);
```

Morphological Filtering



Morphological Filtering

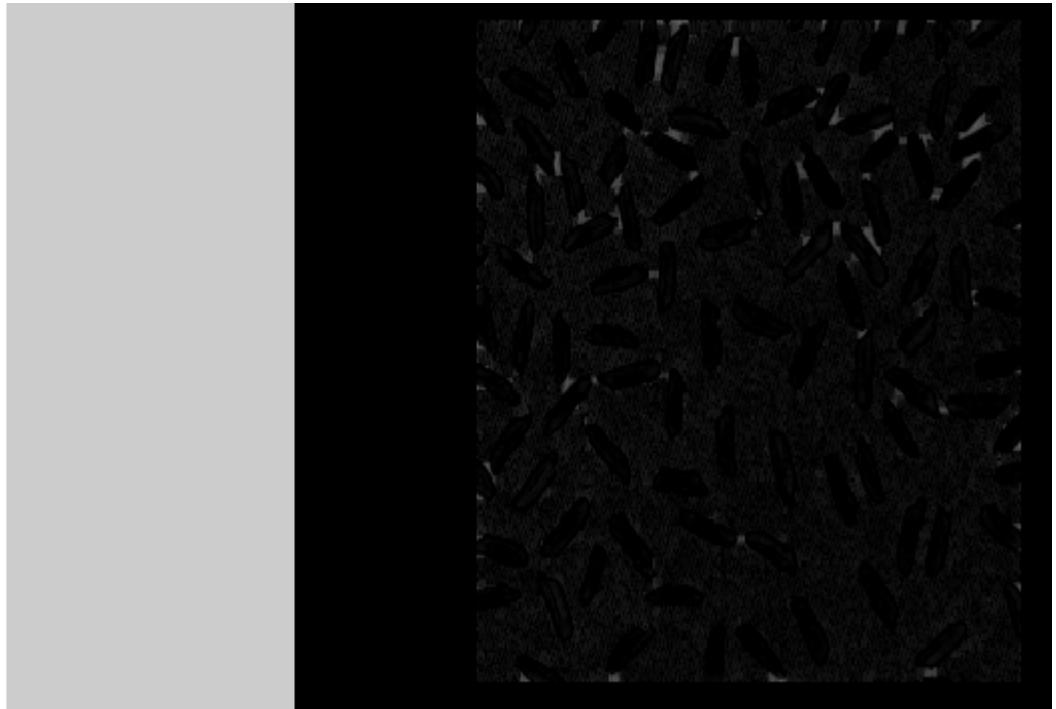
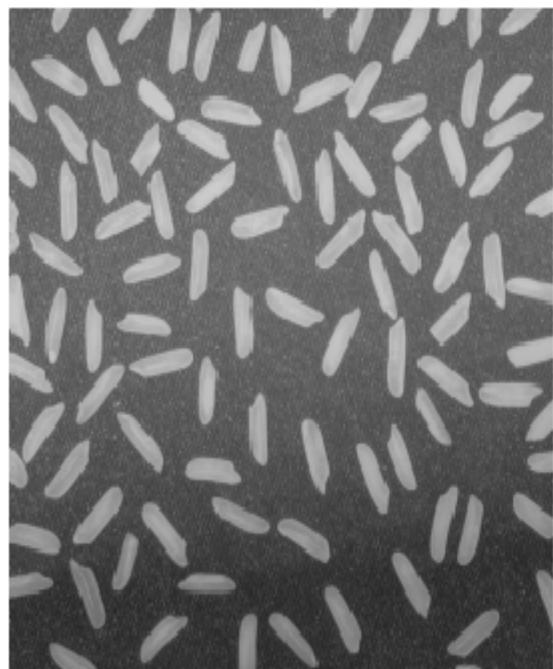
Bottom-hat filtering:

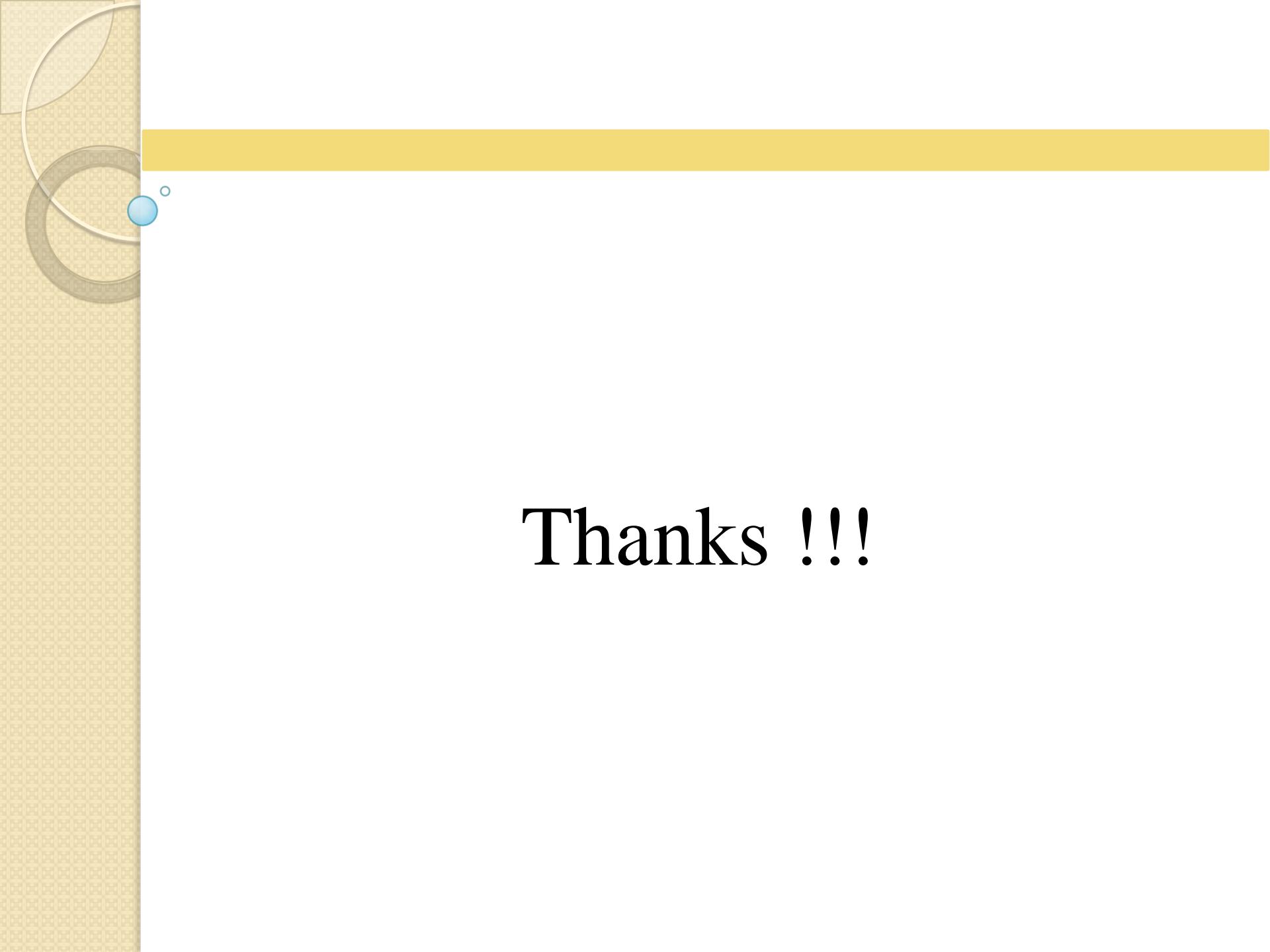
morphological bottom-hat filtering performs on the grayscale or binary image and returns a filtered image, Bottom-hat filtering computes the morphological closing of the image and then subtracts the original image from the result.

Morphological Filtering

```
clc,  
close all;  
clear all;  
I = imread('D:\matlab_folder\Dig1.png');  
figure, imshow(I);  
SE = strel('square',5);  
J = imbothat(I,SE);  
figure, imshow(J);
```

Morphological Filtering





Thanks !!!

Spatial Filters

Sharpening Spatial Filters

The principal objective of sharpening is to highlight transitions of intensity.

Transition intensity: to change (intensity) from an origin state to a destination state at time t.

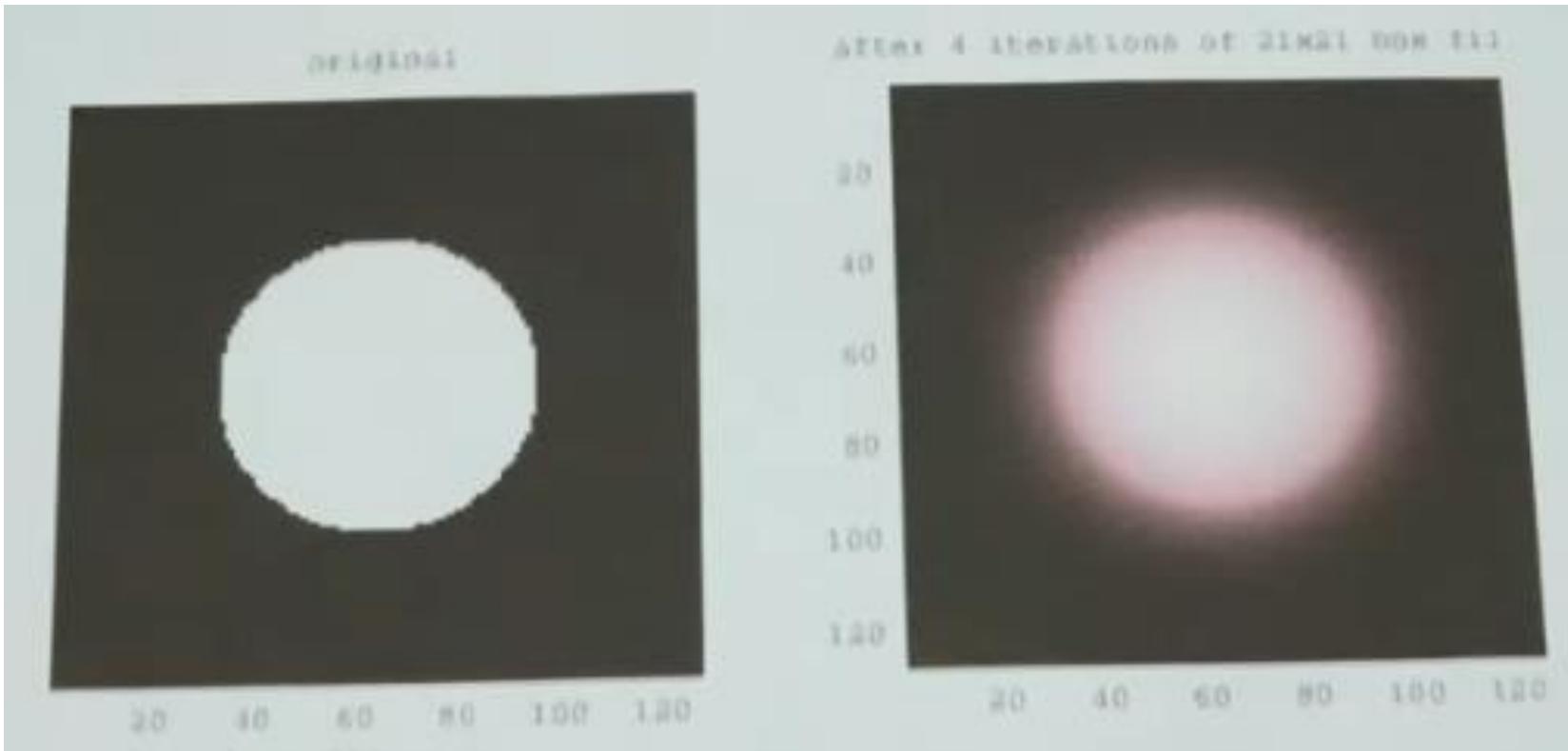
Applications of image sharpening include electronic printing, medical imaging, industrial inspection and autonomous guidance in military systems.

- Blurring – Pixel averaging
- Sharpening – spatial differentiation

The strength of the response of a derivational operator is proportional to the degree of intensity discontinuity of the image at the point at which the operator is applied.

Therefore, image differentiation enhances edges and other discontinuities (such as noise) and deemphasizes area which have slowly varying intensities.

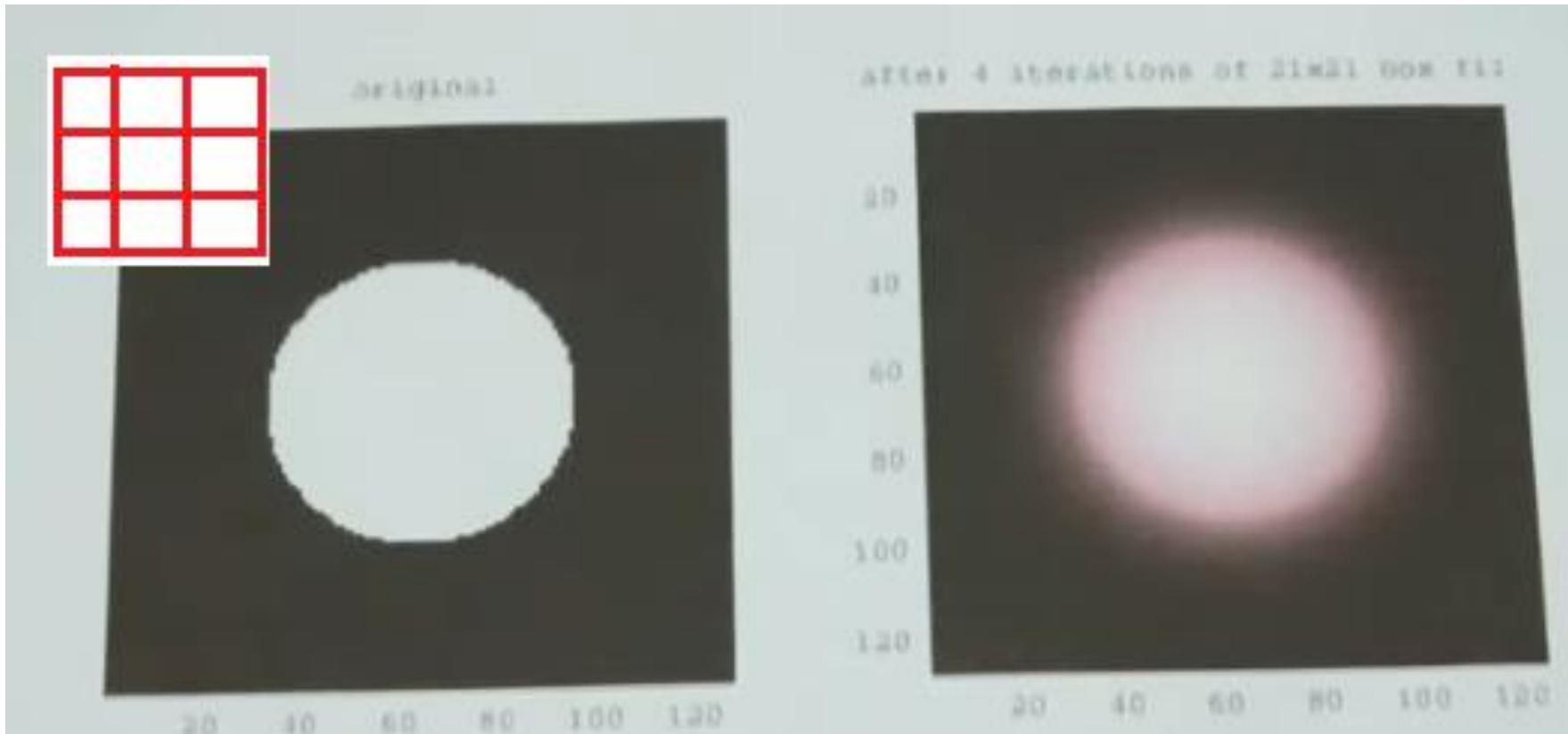
Sharpening Spatial Filters



Left – original Image
Right – Blurred Image

There will be a mask which will slide the original image and will result the blurred image (average value will be placed in the center pixel to obtain the result)

Sharpening Spatial Filters



Left – original Image
Right – Blurred Image

There will be a mask which will slide the original image and will result the blurred image (average value will be placed in the center pixel to obtain the result)
After 4 iterations of 21×21 box filter the result will be right one

Foundation of sharpening filters

- 1) First-order derivatives of a one-dimensional function

$f(x)$:

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

- 2) Second-order derivative of a one-dimensional function

$f(x)$:

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

Laplacian Filter

- It highlights gray-level discontinuities in an image
- It deemphasizes regions with slowly varying gray levels

Formula:

$f(x)$:

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

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

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

Previously we had 1 dimensional now we will consider 2 dimensional

Laplacian Filter

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

Laplacian Filter

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

$f(x-1,y-1)$	$f(x,y-1)$	$f(x+1,y-1)$
$f(x-1,y)$	$f(x,y)$	$f(x+1,y)$
$f(x-1,y+1)$	$f(x,y+1)$	$f(x+1,y+1)$

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

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

Laplacian Filter

8	5	4
0	6	2
1	3	7

Input Image

0	1	0
1	-4	1
0	1	0

Mask

$$\begin{aligned} &= (8 \times 0) + (5 \times 1) + (4 \times 0) + (0 \times 1) + (6 \times -4) + (2 \times 1) + (1 \times 0) + (3 \times 1) + (7 \times 0) \\ &= 0 + 5 + 0 + 0 - 24 + 2 + 0 + 3 + 0 \\ &= -14 \end{aligned}$$

Enhanced Laplacian Filter

Enhanced Laplacian Filter

$$\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix} \xrightarrow{\text{Enhanced}} \begin{matrix} 0 & 1 & 0 \\ 1 & -5 & 1 \\ 0 & 1 & 0 \end{matrix}$$

$$\begin{matrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{matrix} \xrightarrow{\text{Enhanced}} \begin{matrix} 1 & 1 & 1 \\ 1 & -9 & 1 \\ 1 & 1 & 1 \end{matrix}$$

- Q2. Apply enhanced laplacian filter on the given image on the center pixel.

8	5	4
0	6	2
1	3	7

Rule: will increase 1 in the center point (e.g.; if -4 then it will be -5, if +4 then it will be +5)

Enhanced Laplacian Filter

Enhanced Laplacian Filter

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \xrightarrow{\text{Enhanced}} \begin{bmatrix} 0 & 1 & 0 \\ 1 & -5 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \xrightarrow{\text{Enhanced}} \begin{bmatrix} 1 & 1 & 1 \\ 1 & -9 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- Q2. Apply enhanced laplacian filter on the given image on the center pixel.

8	5	4
0	6	2
1	3	7

$$* \begin{bmatrix} 1 & 1 & 1 \\ 1 & -9 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Enhanced Laplacian Filter

Enhanced Laplacian Filter

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

Enhanced

$$\begin{matrix} 0 & 1 & 0 \\ 1 & -5 & 1 \\ 0 & 1 & 0 \end{matrix}$$

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

Enhanced

$$\begin{matrix} 1 & 1 & 1 \\ 1 & -9 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Q2. Apply enhanced laplacian filter on the given Image
on the center pixel.

$$\begin{bmatrix} 8 & 5 & 4 \\ 0 & 6 & 2 \\ 1 & 3 & 7 \end{bmatrix}$$

*

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

$$\begin{aligned} &= 8 + 5 + 4 + 0 - 54 \\ &\quad + 2 + 1 + 3 + 7 \\ &= 30 - 54 = \underline{\underline{-24}}. \end{aligned}$$

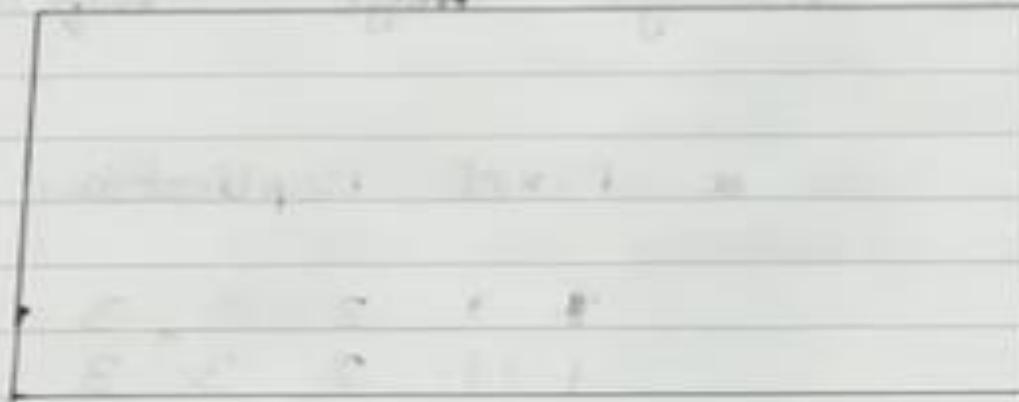
Enhanced Laplacian Filter

Q3. Apply Laplacian filter on the given image.

50	50	50	100	100	100
50	50	50	100	100	100
50	50	50	100	100	100
100	100	100	50	50	50
100	100	100	50	50	50
100	100	100	50	50	50

1	1	1
1	-8	1
1	1	1

Mask



Enhanced Laplacian Filter

Q 3. Apply Laplacian filter on the given image.

50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50

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

Mask

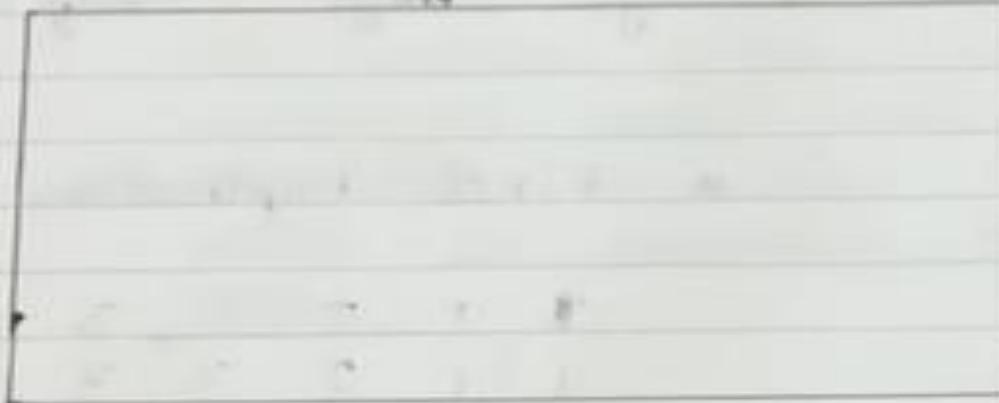
Enhanced Laplacian Filter

Q3. Apply Laplacian filter on the given image.

50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50

1	1	1
1	-8	1
1	1	1

Mask



Enhanced Laplacian Filter

Q3. Apply Laplacian filter on the given image.

50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50

Mask

$$50 \times 8 - 50 \times 8 = 0$$

Enhanced Laplacian Filter

Q 3. Apply Laplacian filter on the given image.

50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
50	50	50	50	100	100	100	100
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50
100	100	100	100	50	50	50	50

1	1	1
1	-8	1
1	1	1

Mask

$$50 \times 8 - 50 \times 8$$

$$= 450 + 300 - 400$$

$$= 150.$$

$$50 \times 3 + 100 \times 5 - 400$$

$$= 150 + 500 - 400$$

$$= -150.$$

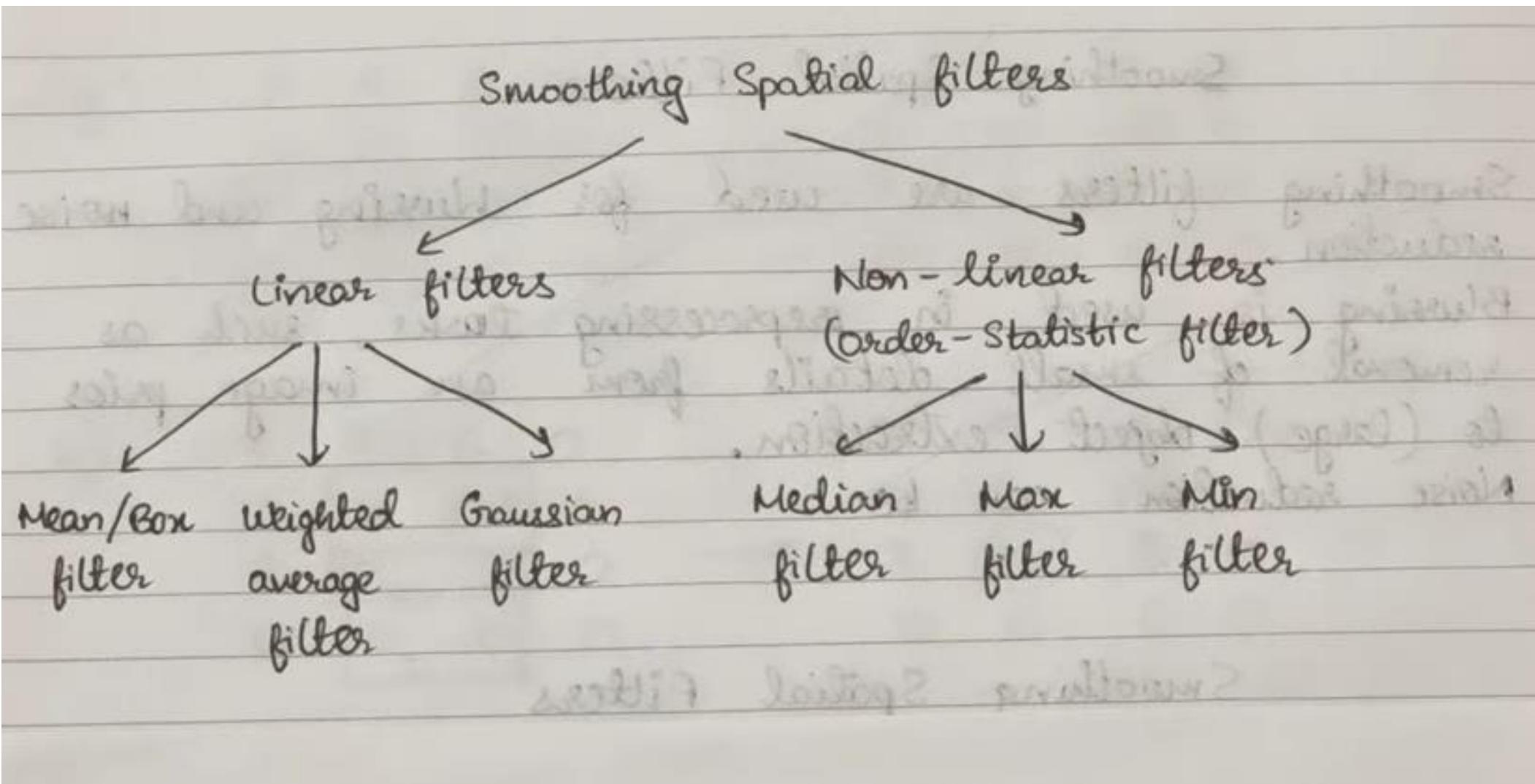
0	0	150	-150	0	0
0	0	150	-150	0	0
150	150	200	-200	-150	-150
-150	-150	-200	200	150	150
0	0	-150	150	0	0
0	0	-150	150	0	0

Smoothing Spatial Filter

Smoothing Spatial Filters

- Smoothing filters are used for blurring and noise reduction.
- Blurring is used in preprocessing tasks, such as removal of small details from an image prior to (large) object extraction.
- Noise reduction can be accomplished by blurring with a linear filter and also by non-linear filtering.

Smoothing Spatial Filter



Smoothing Spatial Filter

* Smoothing linear filters

They are also known as averaging filters (or) lowpass filters as they are simply the average of the pixels contained in the neighbourhood of the filter mask.

The process results in an image with reduced 'sharp' transitions in intensities which ultimately leads to noise reduction.

Smoothing Spatial Filter

1) Box filter - all coefficients are equal.

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \rightarrow \text{Mask}$$

2) weighted average - give more (less) weight to pixels near (away from) the output location.

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \rightarrow \text{Mask}$$

Smoothing Spatial Filter

3) Gaussian filter - the weights are samples of 2D Gaussian function:

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

(2D Gaussian function)
 σ → Standard deviation

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \rightarrow \text{Mask}$$

- Used to blur edges and reduce contrast.
- Similar to median filter but is faster.

Smoothing Spatial Filter

- * Non-linear (Order-Statistic) filters

Their response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result.

- 1) Median filter - Find the median of all the pixel values.
- 2) Min filter - Find the minimum of all the pixel values.
- 3) Max filter - Find the maximum of all the pixel values.

Smoothing Spatial Filter

Q1. Consider the image below and calculate the output of the pixel (2,2) if smoothing is done using 3×3 neighbourhood using all the filters below:

- a) Box / Mean filter
- b) Weighted average filter
- c) Median filter
- d) Min filter
- e) Max filter

1	8	8	0	7
4	7	9	5	7
5	4	6	8	6
4	2	0	1	5
0	1	0	2	0

3×3

Smoothing Spatial Filter

Q1. Consider the image below and calculate the output of the pixel (2,2) if smoothing is done using 3×3 neighbourhood using all the filters below:

- a) Box / Mean filter
- b) Weighted average filter
- c) Median filter
- d) Min filter
- e) Max filter

a) Box filter

1	8	8	0	7
4	7	9	5	7
5	4	6	8	6
4	2	0	1	5
0	1	0	2	0

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

Smoothing Spatial Filter

Q1. Consider the image below and calculate the output of the pixel (2,2) if smoothing is done using 3×3 neighbourhood using all the filters below:

- a) Box / Mean filter
- b) Weighted average filter
- c) Median filter
- d) Min filter
- e) Max filter

1	8	8	0	7
4	7	9	5	7
5	4	6	8	6
4	2	0	1	5
0	1	0	2	0

3×3

a) Box filter

$$= \frac{1}{9} \times [7+9+5+4+6+8+2+0+1] \quad \frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$= \frac{1}{9} \times [42] = 4.66 \approx \underline{\underline{5}}$$

Smoothing Spatial Filter

Consider the image below and calculate the output of the pixel (2,2) if smoothing is done using 3×3 neighbourhood using all the filters below:

- a) Box / Mean filter
- b) Weighted average filter
- c) Median filter
- d) Min filter
- e) Max filter

1	8	8	0	7
4	7	9	5	7
5	4	6	8	6
4	2	0	1	5
0	1	0	2	3×3 0

- b) Weighted average filter

$$= \frac{1}{16} [7 \times 1 + 9 \times 2 + 5 \times 1 + 4 \times 2 + \\ 4 \times 6 + 8 \times 2 + 2 \times 1 + 0 \times 2 \\ + 1 \times 1]$$

$$= \frac{1}{16} [81] = 5.0625 \approx \underline{\underline{5}}$$

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

Smoothing Spatial Filter

Consider the image below and calculate the output of the pixel (2,2) if smoothing is done using 3×3 neighbourhood using all the filters below:

- a) Box / Mean filter
- b) Weighted average filter
- c) Median filter
- d) Min filter
- e) Max filter

1	8	8	0	7
4	7	9	5	7
5	4	6	8	6
4	2	0	1	5
0	1	0	2	3×3

c) Median filter
0, 1, 2, 4, 5, 6, 7, 8, 9

$$\text{Median} = 5.$$

d) Min filter
 $= \underline{\underline{0}}$.

e) Max filter
 $= \underline{\underline{9}}$.

Smoothing Spatial Filter

First Order Derivative Filters

- 1) Roberts operators (Cross-gradient)

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

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

$$\begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix}$$

Image

- 2) Sobel operators

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

- 3) Prewitt operators

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

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

Mainly used for edge detection

Smoothing Spatial Filter

Problems with Roberts Cross :

- 1) 2×2 masks are not easy to implement.
- 2) No. of calculations are more.
- 3) No. of neighboring pixels considered in one go are less.

To solve these problems, we make the following changes:

- 1) Change in the size of the mask.
- 2) Change in the no. of neighbouring pixels considered.

Smoothing Spatial Filter

Q2. Apply Roberts, Sobel and Prewitt operators on the pixel (1,1) in the following image.

50	50	100	100
50	50	100	100
50	50	100	100
50	50	100	100

Smoothing Spatial Filter

Q2 Apply Roberts, Sobel and Prewitt operators on the pixel (1,1) in the following image.

50	50	100	100
50	50	100	100
50	50	100	100
50	50	100	100

Input image

1) Roberts operator

-1	0
0	1

Smoothing Spatial Filter

Q2. Apply Roberts, Sobel and Prewitt operators on the pixel (1,1) in the following image.

50	50	100	100
50	50	100	100
50	50	100	100
50	50	100	100
50	50	100	100

Input image

1) Roberts operator

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

$$\begin{aligned} &= 50 \times (-1) + 100 \times 1 \\ &= -50 + 100 = \underline{\underline{50}}. \end{aligned}$$

Smoothing Spatial Filter

Q2. Apply Roberts, Sobel and Prewitt operators on the pixel (1,1) in the following image.

50	50	100	100
50	50	100	100
50	50	100	100
50	50	100	100

Input image

1) Roberts operator

-1	0
0	1

$$\begin{aligned} &= 50 \times (-1) + 100 \times 1 \\ &= -50 + 100 = \underline{\underline{50}}. \end{aligned}$$

2) Sobel operator

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

$$\begin{aligned} &= 50(-1) + 50(-2) + 100(-1) + 50(1) + 50(2) + 100(1) \\ &= -50 - 100 - 100 + 50 + 100 + 100 = \underline{\underline{0}}. \end{aligned}$$

Smoothing Spatial Filter

3) Prewitt operator

$$\begin{matrix} 50 & 50 & 100 \\ 50 & \textcircled{50} & 100 \\ 50 & 50 & 100 \end{matrix} \quad \begin{matrix} 100 \\ 100 \\ 100 \end{matrix} \quad \times \quad \begin{matrix} -1 & -1 & -1 \\ 0 & \textcircled{0} & 0 \\ 1 & 1 & 1 \end{matrix}$$

Input image

$$\begin{aligned} &= -1(50 + 50 + 100) + 1(50 + 50 + 100) \\ &= \underline{\underline{0}}. \end{aligned}$$

Thanks !!!

Region Growing and Region Splitting

Region Growing and Region Splitting

Edges and thresholds sometimes do not give good results for segmentation.

Region-based segmentation is based on the connectivity of similar pixels in a region.

There are two main approaches to region-based segmentation : region-growing and region-splitting.

Region Growing and Region Splitting

Procedure for Region-Growing

- 1) Find all connected components in $S(x,y)$ and reduce each connected component to one pixel. Label all such pixels found as 1. All other pixels in S are labeled 0.
- 2) Form an image f_ϕ such that, at each point (x,y) , $f_\phi(x,y) = 1$ if the input image satisfies a given predicate, ϕ , at those coordinates, and $f_\phi(x,y) = 0$ otherwise.

Region Growing and Region Splitting

- 3) Let g be an image formed by appending to each seed point in S all the 1-valued points in f_0 that are 8-connected to that seed point.
- 4) Label each connected component in g with a different region label (Ex. integers or letters). This is the segmented image obtained by region growing.

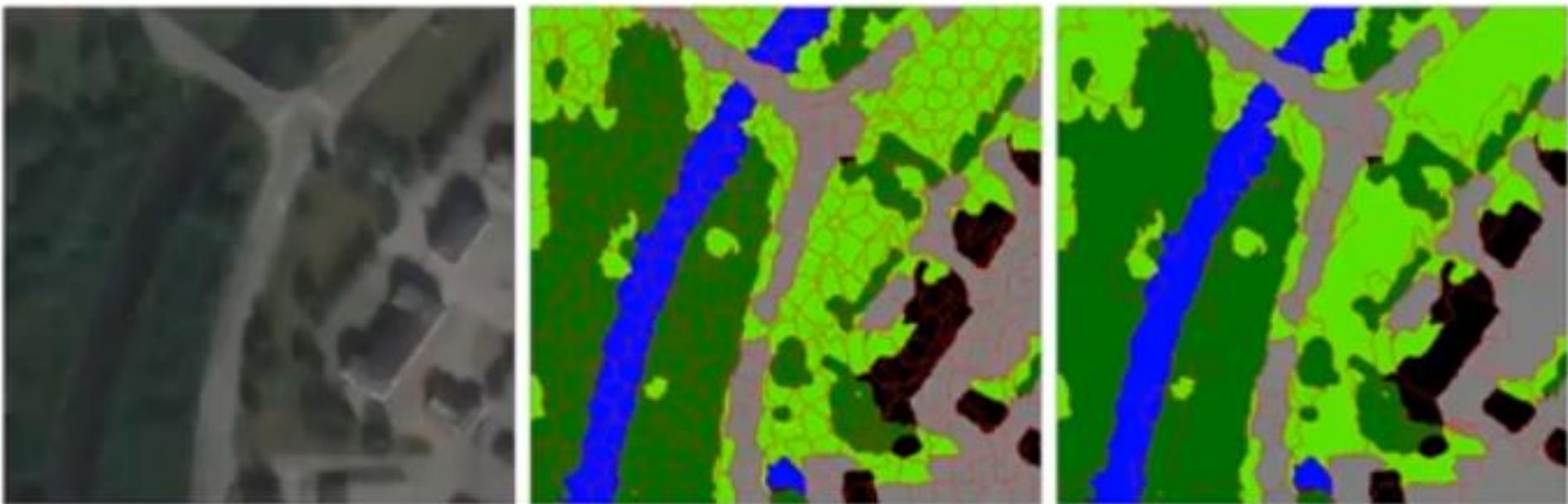
Region Growing and Region Splitting

Procedure for Region splitting and Merging

- 1) If a region R is inhomogeneous ($P(R) = \text{False}$), then R is split into four sub-regions.
 - 2) If 2 adjacent regions R_i, R_j are homogeneous ($P(R_i \cup R_j) = \text{True}$), then they are merged.
 - 3) The algorithm stops when no further splitting or merging is possible.
- # Note : Condition for region growing :
 $\text{abs}(\text{seed value} - \text{pixel value}) \leq \text{Threshold}$

Region Merging/Growing

Used to segment an image into regions of similar intensity or color.
The algorithm is used to evaluate the values within a regional span
and grouped together based on the **merging criteria**



Region Growing

Q1. Apply region growing on the following image with initial point at (2,2) and threshold value as 2. Use 4 connectivity.

$$T = 2$$

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

Ans. The segmented region is shown in the following figure.

Condition \rightarrow absolute difference ≤ 2 .
4 way connectivity

Region Growing

Q1. Apply region growing on the following image with initial point at (2,2) and threshold value as 2. Use 4 connectivity.

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

$$T = 2$$

a

$$7 - 6 = 1 \checkmark$$

$$7 - 4 = 3 \times$$

$$7 - 5 = 2 \checkmark$$

$$7 - 3 = 4 \times$$

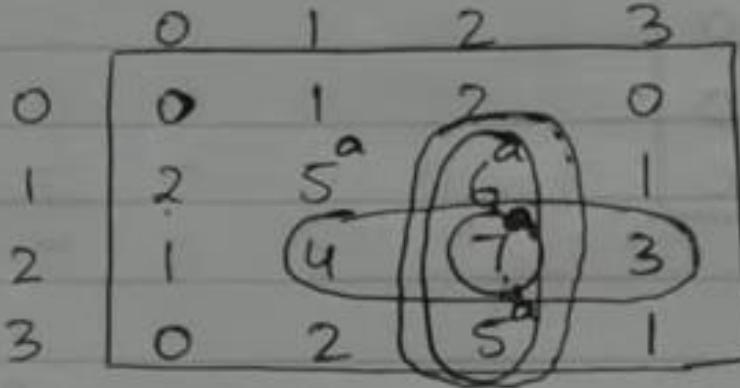
Ans. The segmented region is shown in the following figure.

Condition \rightarrow absolute difference ≤ 2
4 way connectivity

Region Growing

Q1. Apply region growing on the following image with initial point at (2,2) and threshold value as 2. Use 4 connectivity.

$$T = 2$$



$$7 - 6 = 1 \checkmark$$

$$7 - 4 = 3 \times$$

$$7 - 5 = 2 \checkmark$$

$$7 - 3 = 4 \times$$

Ans. The segmented region is shown in the following figure.

Condition \rightarrow absolute difference ≤ 2 .

4 way connectivity

Region Growing

Ans. The segmented region is shown in the following figure.

Condition \rightarrow absolute difference ≤ 2 .
4 way connectivity

Here, we will compare each element which is 4 way connected to the seed element.

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

Region Growing

Segmented image obtained from region growing:

	0	1	2	3
0	0	0	0	0
1	0	1	1	0
2	0	0	1	0
3	0	0	1	0

Region Growing

Q2. Apply region growing on the following image with seed point as 6 and threshold value as 3.

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

Seed point = 6

T = 3

Condition \rightarrow absolute difference $<= 3$.
8 way connectivity.

Region Growing

Q2. Apply region growing on the following image with seed point as 6 and threshold value as 3.

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

Seed point = 6
 $T = 3$

a

:

Condition \rightarrow absolute difference $<= 3$.
8 way connectivity.

Region Growing

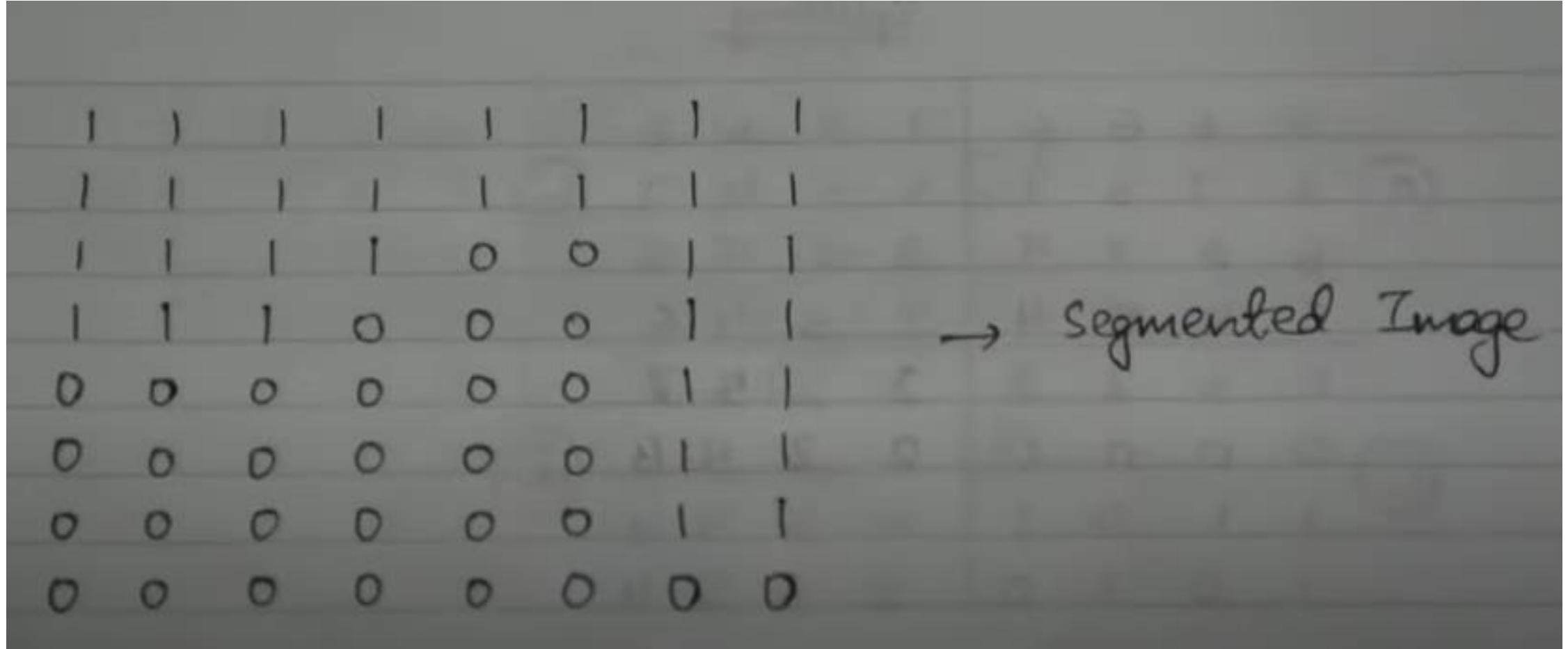
Q2. Apply region growing on the following image with seed point as 6 and threshold value as 3.

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

Seed point = 6
 $T = 3$

Condition \rightarrow absolute difference $<= 3$.
8 way connectivity.

Region Growing



Region Growing

Region Growing Segmentation

Example: Consider a 3-bit image given below. Assuming seed value of 6, segment the following image using 8-connectivity and threshold=3
How many segments are identified ?

Ans: Conditions for growing a region

8-Connectivity between seed pixel and given pixel

$$\text{abs}(\text{seed_value} - \text{pixel_value}) < 3$$

seed=	6				
2	2	7	2	1	
1	7	6	6	2	
7	6	6	5	7	
2	4	5	4	2	
1	2	5	1	1	



Applying thresholding condition

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



Region Growing

Applying thresholding condition

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

Applying 8-connectivity to label each pixel

a	a	b	c	c
a	b	b	b	c
b	b	b	b	b
d	b	b	b	e
d	d	b	e	e

Region Splitting

Q. Apply the region splitting on the following image
Assume the threshold value be $\lambda = 4$

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

Region Splitting

$$7 - 0 = 7$$

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

Region Splitting

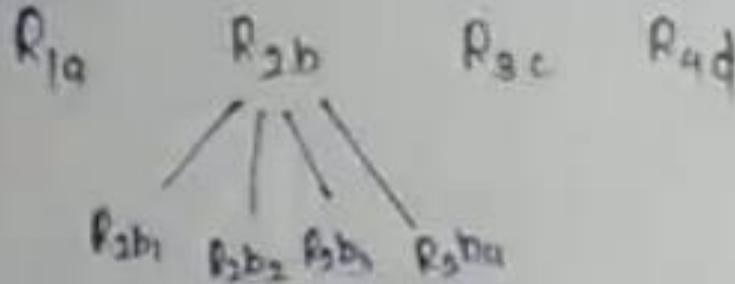
R_{1a} R_{2b} R_{3c} R_{4d}

Q-12				Q-13			
5	6	6	6	7	7	6	6
6	7	6	7	5	5	4	7
✓	6	6	4	4	3	2	5
3	4	5	4	2	3	4	6
0	3	2	3	3	2	4	7
0	0	0	0	2	2	5	6
1	1	0	1	0	3	4	4
1	0	1	0	2	3	5	4

$$7 - 0 = 7$$

7-4 = 3

Region Splitting



R _{1a}				R _{2b}			R _{3c}		
5	6	6	6	7	7	6	6	7	
6	7	6	7	5	5	4	7		
✓	6	6	4	3	2	5	6	v ₃	
5	4	5	4	2	3	4	6		
0	3	2	3	3	2	4	7		
0	0	0	0	2	2	5	6		
1	1	0	1	0	3	4	4	R _{3c}	
1	0	1	0	2	3	5	4		
								Q _{3c}	

$$7 - 0 = 7$$

$$7 - 4 = 3$$

$$7 - 2 = 5$$

$$7 - 5 = 2$$

$$7 - 4 = 3$$

$$6 - 4 = 2$$

$$3 - 2 = 1$$

Region Splitting

Q _{1,0}								Q _{2,0}		Q _{3,0}	
5	6	6	6	7	7	6	6				
6	7	6	7	5	5	4	7				
6	6	4	4	3	2	5	6				
5	4	5	4	2	3	4	6				
0	3	2	3	3	2	4	7				
0	0	0	0	2	2	5	6				
1	1	0	1	0	3	4	4				
1	0	1	0	2	3	5	4				
Q _{4,0}											

7 - 0 = 7

7 - 4 = 3

7 - 2 = 5

7 - 5 = 2

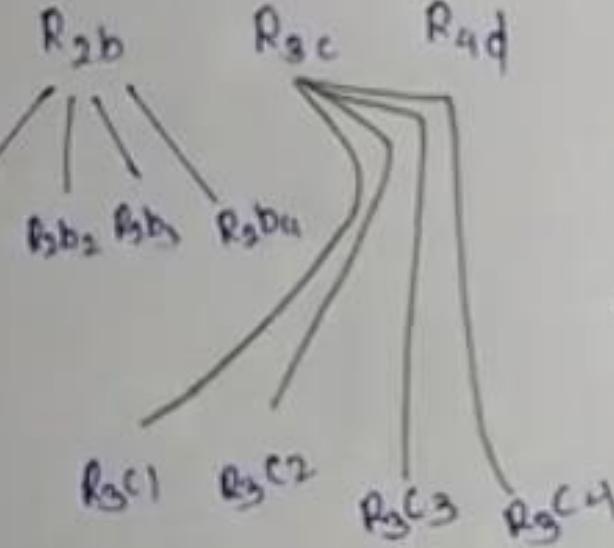
7 - 4 = 3

6 - 4 = 2

3 - 2 = 1

7 - 0 = 7

Region Splitting



R _{1a}				R _{2a}		R _{2b}	
5	6	6	6	7	7	6	6
6	7	6	7	5	5	4	7
✓	6	6	4	4	3	2	5
5	4	5	4	2	3	4	6
0	3	2	3	3	2	4	7
0	0	0	0	2	2	5	6
1	1	0	1	0	3	4	0
1	0	1	0	2	3	5	4
R _{3a}				R _{3b}		R _{4a}	

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

Region Splitting

R _{1a}				Q _{2a}		R _{2b}	
5	6	6	6	7	7	6	6
6	7	6	7	5	5	4	7
6	6	4	4	3	2	5	6
5	4	5	4	2	3	4	6
0	3	2	3	3	2	4	7
0	0	0	0	2	2	5	6
1	1	0	1	9	3	4	5
1	0	1	0	2	3	5	4

Q_{2b}

3 - 0 = 3

7 - 0 = 7

7 - 4 = 3

7 - 2 = 5

7 - 5 = 2

7 - 4 = 3

6 - 4 = 2

3 - 2 = 1

7 - 0 = 7

8 - 2 = 1

7 - 4 = 3

5 - 4 = 1

3 - 0 = 3

Region splitting and Merging

Q3. Apply splitting and merging on the following image with threshold value equal to 3.

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

Ans. Condition : absolute difference ≤ 3 .

Max value = 7

Min value = 0

$|7 - 0| = 7$ which is greater than 3.

Therefore we will split the region into 4 sub-regions.

Region splitting and Merging

Splitting

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

In region ④ :

$$\text{Max} = 7 \quad \text{Min} = 4$$

$|7-4| = 3$ which is equal to threshold.
Therefore, no need to split.

Region splitting and Merging

Splitting

	5	6	6	6		7	7	6	6		7 - 2 = 5
(a)	6	7	6	7		5	5	4	7	(b)	.
	6	6	4	4		3	2	5	6	.	.
	5	4	5	4		2	3	4	6	.	.
	0	3	2	3		3	2	5	7	.	.
(c)	0	0	0	0		2	2	5	6	(d)	.
	1	1	0	1		0	3	4	4	.	.
	1	0	1	0		2	3	5	4	.	.

Region splitting and Merging

Splitting

	5	6	6	6	7	7	6	6	$ 7 - 2 = 5$
a	6	7	6	7	5	5	4	7	b
	6	6	4	4	3	2	5	6	
	5	4	5	4	2	3	4	6	$ 3 - 0 = 3$
	0	3	2	3	3	2	5	7	
c	0	0	0	0	2	2	5	6	d
	1	1	0	1	0	3	4	4	$ 7 - 0 = 7$
	1	0	1	0	2	3	5	4	

Region splitting and Merging

In region (b) :

$$\text{Max} = 7 \quad \text{Min} = 2$$

$$|7-2| = 5 \text{ which is greater than } 3.$$

Therefore we will split the region (b) into 4 sub-regions.

In region (c) :

$$\text{Max} = 3 \quad \text{Min} = 0$$

$$|3-0| = 3.$$

So no need to split.

In region (d) :

$$\text{Max} = 7 \quad \text{Min} = 0$$

$$|7-0| = 7.$$

So we will split into 4 sub-regions.

Region splitting and Merging

	5	6	6	6	7	7	6	6	b1	b2	
a	6	7	6	7	5	5	4	7			b
	6	6	4	4	3	2	5	6	b3	b4	
	5	4	5	4	2	3	4	6			
	0	3	2	3	3	2	5	7	a1	d2	
c	0	0	0	0	2	2	5	6			d
	1	1	0	1	0	3	4	4	d3	d4	
	1	0	1	0	2	3	5	4			

Furthermore, we will check all the sub-regions. Since all of them are ≤ 3 , no further splitting is required.

Region splitting and Merging

Merging

Check adjacent regions, if they are falling within the threshold, then merge.

Consider regions \textcircled{a} and $\textcircled{b1}$

$$\text{Max} = 7 \quad \text{Min} = 4$$

$$|7-4| = 3. \checkmark \text{ Merge.}$$

Region splitting and Merging

5	6	6	6	7	7	b1	b2	
a	6	7	6	7	5	5	4	7
	6	6	4	4	3	2	5	6
	5	4	5	4	2	3	4	6
	0	3	2	3	3	2	5	7
c	0	0	0	0	2	2	5	6
	1	1	0	1	0	3	4	4
	1	0	1	0	2	3	5	4

Furthermore, we will check all the sub-regions.
Since all of them are ≤ 3 , no further
splitting is required.

Merging

Check adjacent regions, if they are falling
within the threshold, then merge.

Consider regions a and b1

$$\text{Max} = 7 \quad \text{Min} = 4$$

$$|7-4| = 3 \quad \checkmark \text{ Merge.}$$

Region splitting and Merging

Consider regions $a b_1$ and b_2

$$\text{Max} = 7 \quad \text{Min} = 4$$

$$|7-4| = 3. \quad \checkmark \text{ Merge.}$$

Consider regions $a b_1 b_2$ and b_4

$$\text{Max} = 7 \quad \text{Min} = 4$$

$$|7-4| = 3 \quad \checkmark \text{ Merge}$$

Consider regions $a b_1 b_2 b_4$ and d_2

$$\text{Max} = 7 \quad \text{Min} = 4$$

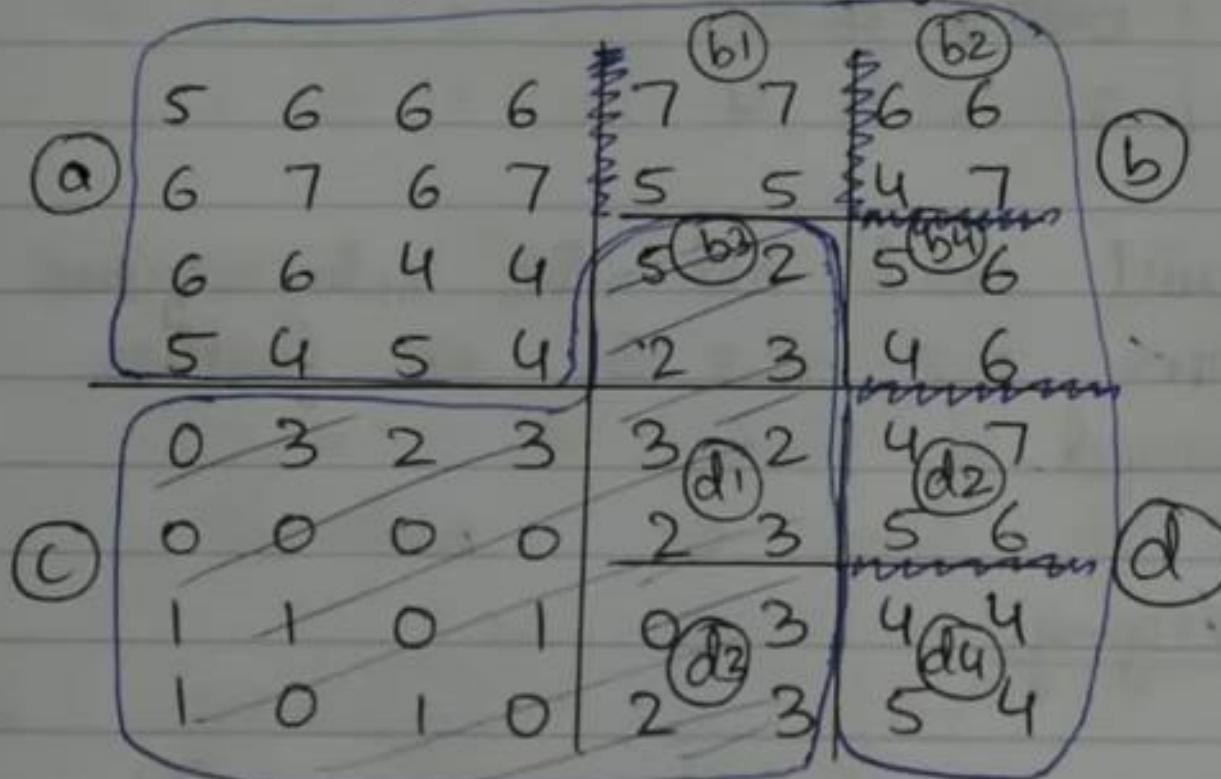
$$|7-4| = 3 \quad \checkmark \text{ Merge.}$$

Similarly, merge $a b_1 b_2 b_4 d_2$ with d_4 .

Region splitting and Merging

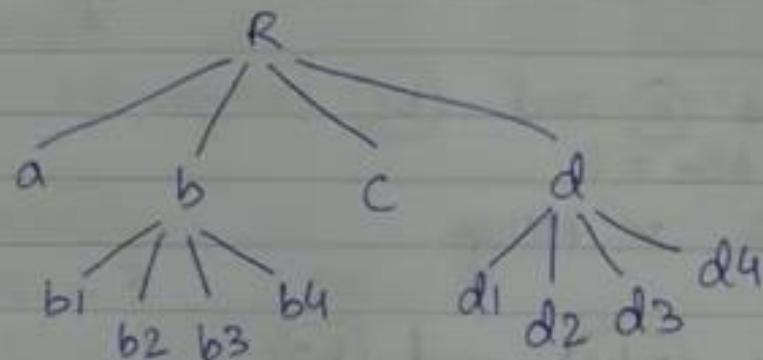
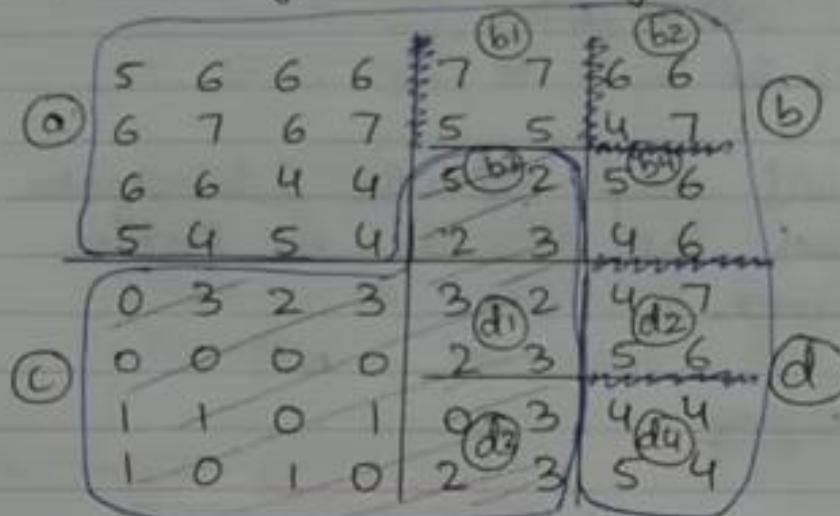
Similarly, merge \textcircled{c} , $\textcircled{d1}$, $\textcircled{b3}$ and $\textcircled{d3}$.

Final segmented image :



Region splitting and Merging

Final segmented image :



Quadtree Structure
for Splitting

Thank you

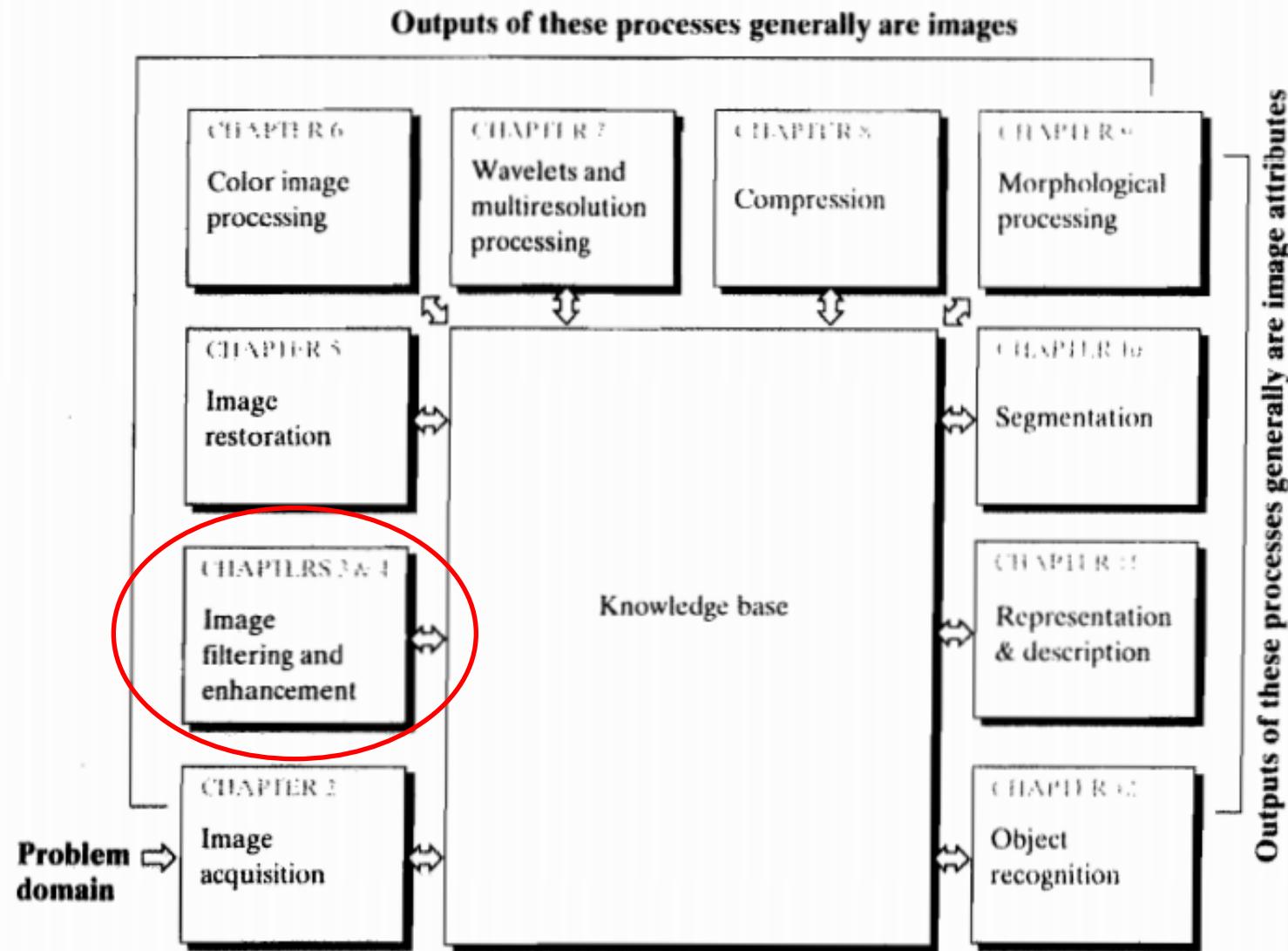


Chapter 10

Segmentation

Remember?

FIGURE 1.23
Fundamental steps in digital image processing. The chapter(s) indicated in the boxes is where the material described in the box is discussed.



Outputs of these processes generally are image attributes



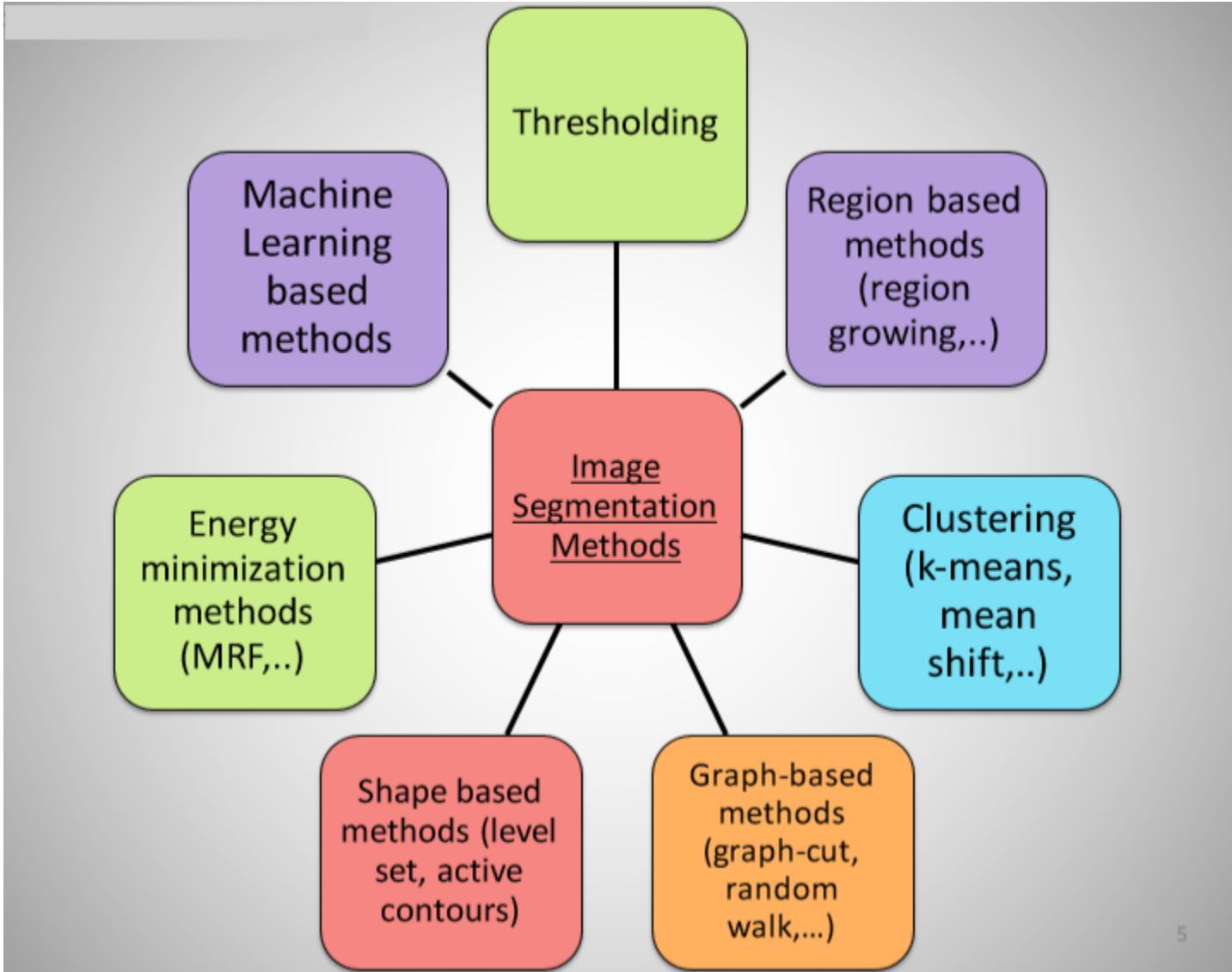
Segmentation !!!

Segmentation

- **Aim:** to partition an image into a collection of set of pixels
 - Meaningful regions (coherent objects)
 - Linear structures (line, curve, ...)
 - Shapes (circles, ellipses, ...)



Segmentation



Basics of Image Segmentation

Definition: Image segmentation partitions an image into regions called segments.

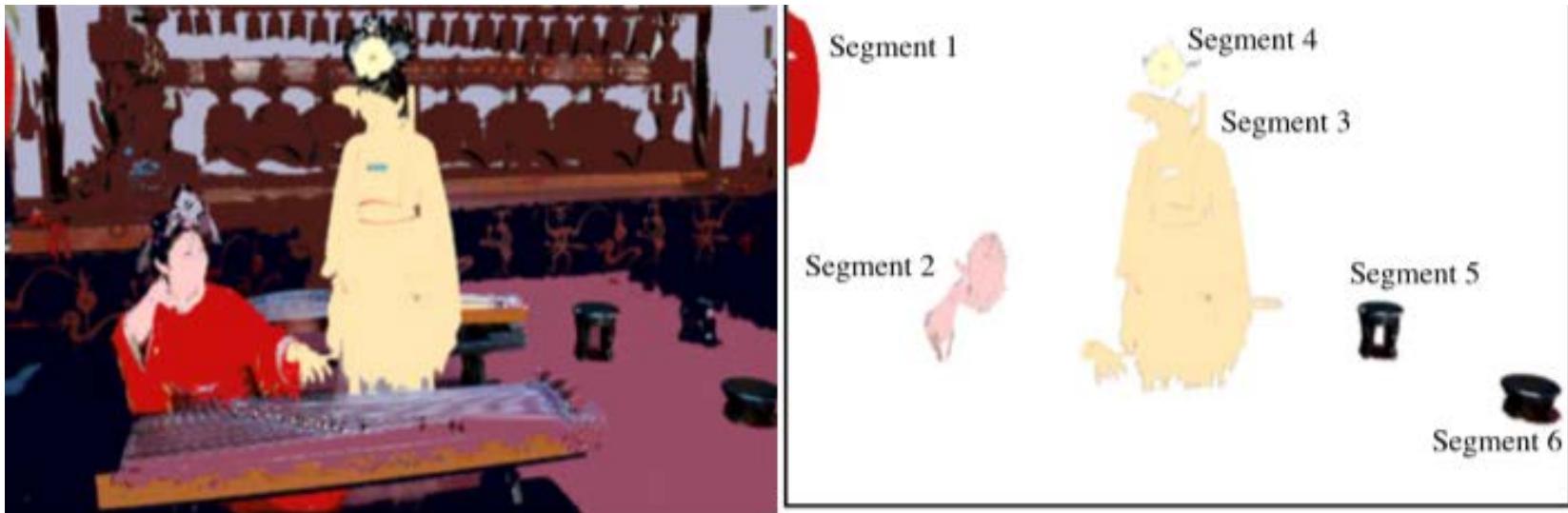


Image segmentation creates segments of connected pixels by analyzing some similarity criteria: intensity, color, texture, histogram, features, ...

Basics of Image Segmentation

- Segmentation partitions an image into distinct regions containing each pixels with similar attributes.
- To be meaningful and useful for image analysis and interpretation, the regions should strongly relate to depicted objects or features of interest.

Basics of Image Segmentation

- Meaningful segmentation is the first step from low-level image processing transforming a greyscale or color image into one or more other images to high-level image description in terms of features, objects, and scenes. The success of image analysis depends on reliability of segmentation, but an accurate partitioning of an image is generally a very challenging problem

Image Binarization

- Image binarization applies often just one global threshold T for mapping a scalar image I into a binary image.



Image Binarization

Image binarization applies often just one global threshold T for mapping a scalar image I into a binary image.

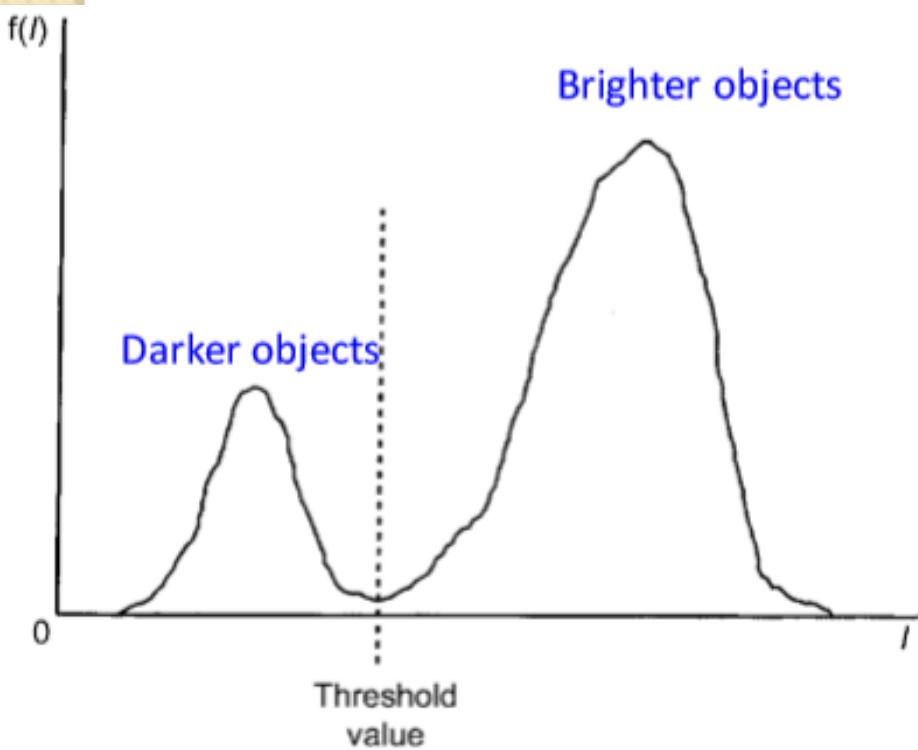
- The global threshold can be identified by an optimization strategy aiming at creating “large” connected regions and at reducing the number of small-sized regions, called artifacts

Thresholding

Thresholding:

method for
based on
intensity

Most frequently employed
determining threshold is
histogram analysis
of levels



Peak on the left of the histogram corresponds to dark objects

Peak on the right of the histogram corresponds to brighter objects

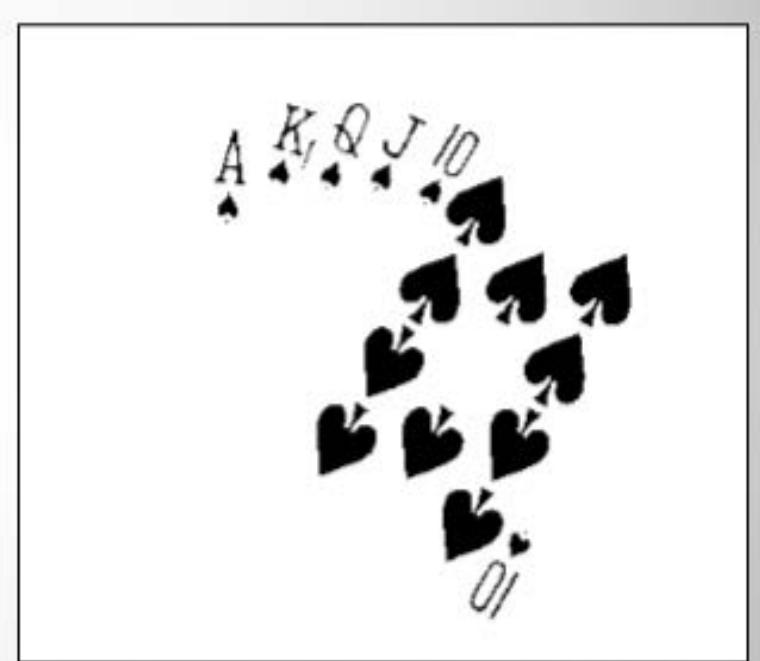
DIFFICULTIES

1. The valley may be so broad that it is difficult to locate a significant minimum
2. Number of minima due to type of details in the image
3. Noise
4. No visible valley
5. Histogram may be multi-modal

Thresholding



Original Image

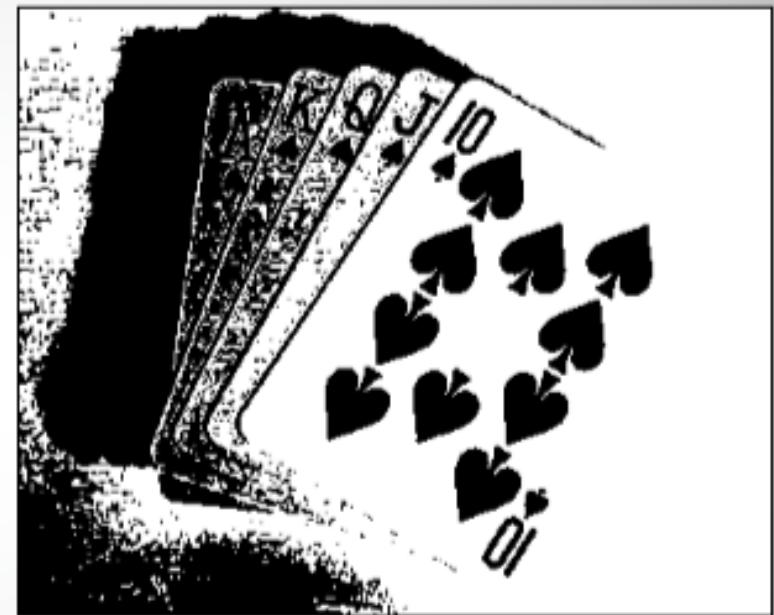


Thresholded Image

Thresholding

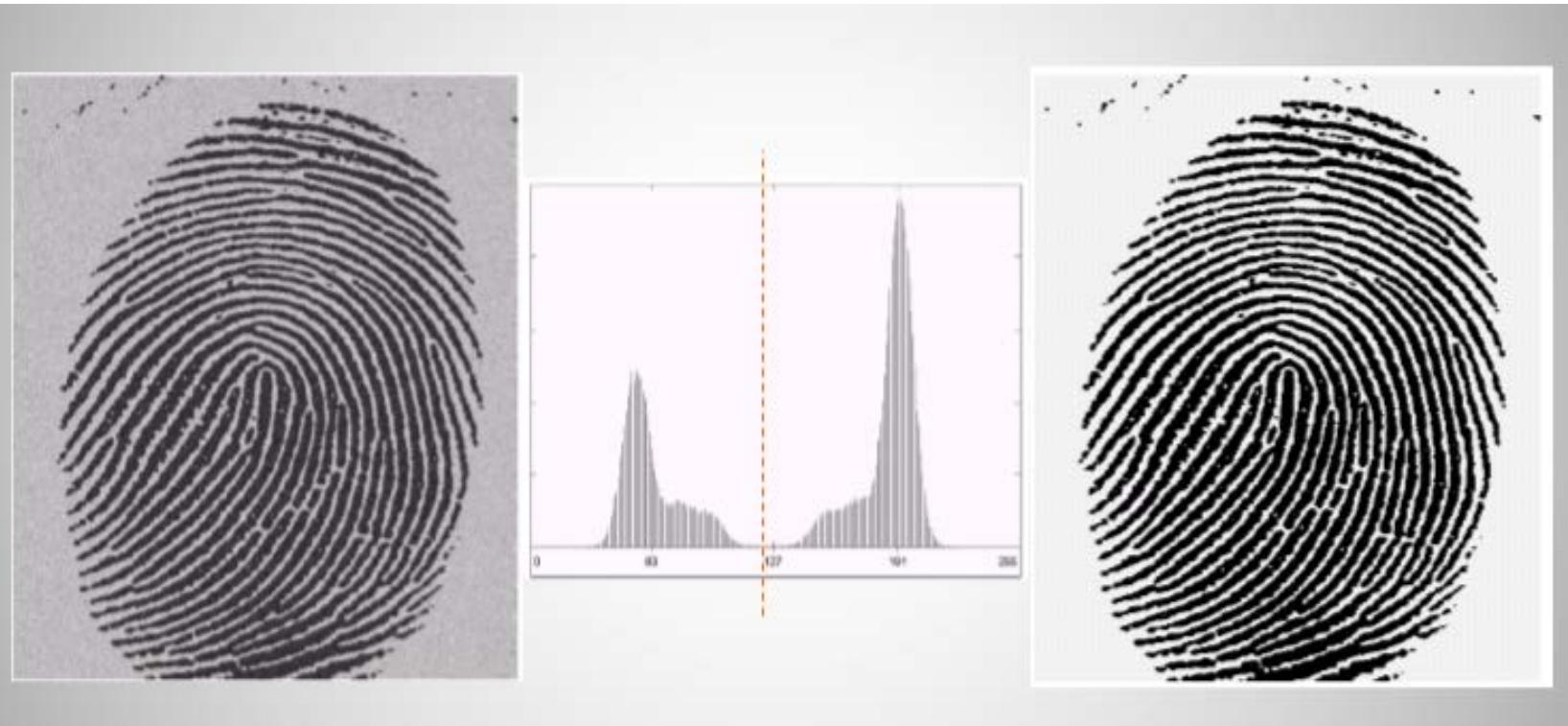


Threshold Too Low

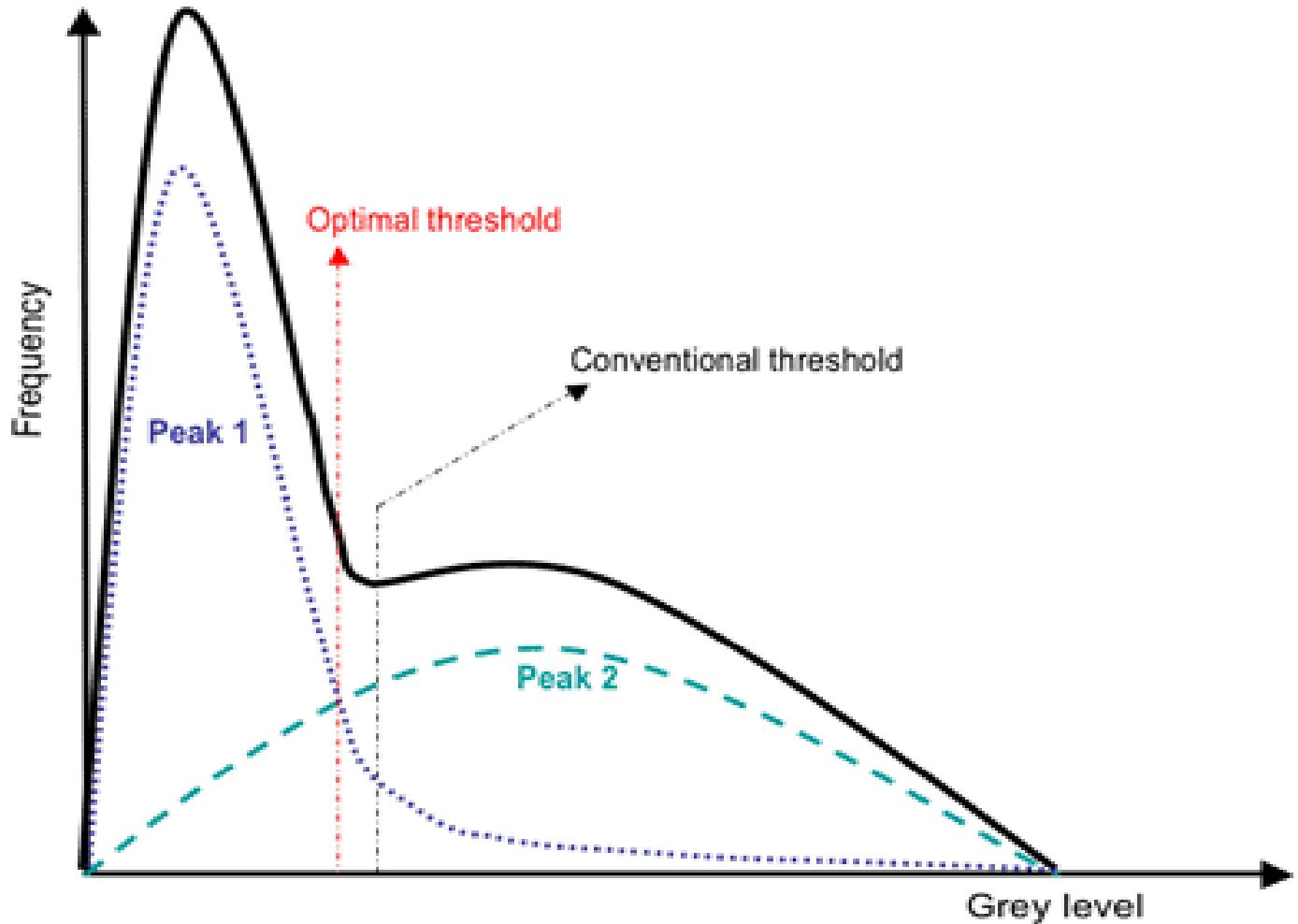


Threshold Too High

Thresholding



Thresholding



OTSU

Definition: The method uses the grey-value histogram of the given image I as input and aims at providing the best threshold in the sense that the “overlap” between two classes, set of object and background pixels, is minimized (i.e., by finding the best balance).

- Otsu’s algorithm selects a threshold that maximizes the between-class variance σ_b^2 . In the case of two classes,

$$\sigma_b^2 = P_1(\mu_1 - \mu)^2 + P_2(\mu_2 - \mu)^2 = P_1 P_2 (\mu_1 - \mu_2)^2$$

where P_1 and P_2 denote class probabilities, and μ_i the means of object and background classes.

- Let C_I be the relative cumulative histogram of an image I , then P_1 and P_2 are approximated by $c_I(u)$ and $1 - c_I(u)$, respectively.
- u is assumed to be the chosen threshold.

OTSU

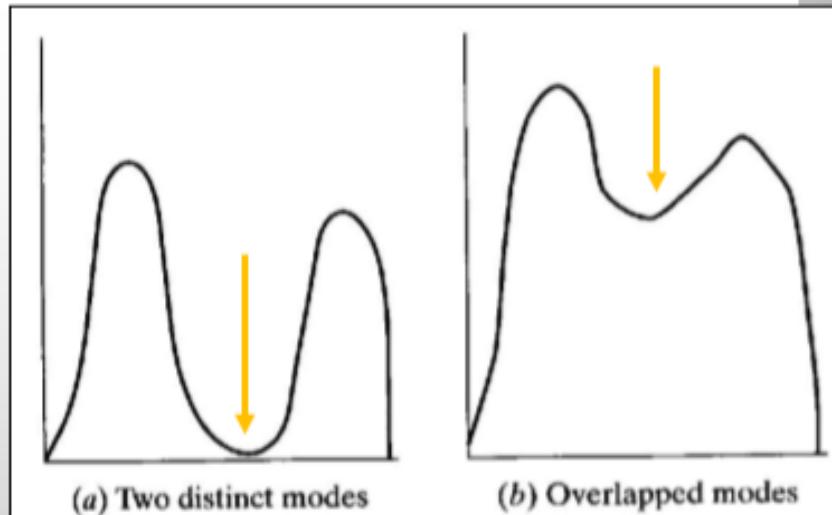
- 1: Compute histogram H_I for $u = 0, \dots, G_{\max}$;
- 2: Let T_0 be the increment for potential thresholds; $u = T_0$; $T = u$; and $S_{\max} = 0$;
- 3: **while** $u < G_{\max}$ **do**
- 4: Compute $c_I(u)$ and $\mu_i(u)$ for $i = 1, 2$;
- 5: Compute $\sigma_b^2(u) = c_I(u)[1 - c_I(u)][\mu_1(u) - \mu_2(u)]^2$;
- 6: **if** $\sigma_b^2(u) > S_{\max}$ **then**
- 7: $S_{\max} = \sigma_b^2(u)$ and $T = u$;
- 8: **end if**
- 9: Set $u = u + T_0$
- 10: **end while**

$$P_1 = \sum_{i=0}^u p(i) \quad \mu_1 = \sum_{i=0}^u ip(i)/P_1$$

$$P_2 = \sum_{i=u+1}^{G_{\max}} p(i) \quad \mu_2 = \sum_{i=u+1}^{G_{\max}} ip(i)/P_2$$

probabilities

Class means





Thanks !!!