

Computer Vision, and Digital Image Processing

Prepared by Dr. Mohamed Mostafa

AI 612

Lecture 9



Image Segmentation, Feature Extraction

Lecture 9

AI 612

Prepared by
Dr. Mohamed Mostafa

www.midocean.ae

Chapters

10- 11

Resources for this
lecture

Image Segmentation Feature Extraction



Chapter 10, 11:

Gonzalez, Rafael C. *Digital image processing*.
Pearson education, India, 2009.

The Image Segmentation

- The segmentation refers to the task of partitioning an image into multiple regions or segments based on certain characteristics or criteria.
- The goal is to group together pixels that belong to the same object or structure in the image while separating them from pixels belonging to different objects or backgrounds.
- Segmentation plays a crucial role in various computer vision tasks, including object detection, recognition, tracking, and scene understanding. It enables computers to interpret and analyze visual data by dividing the image into meaningful components.

The Image Segmentation

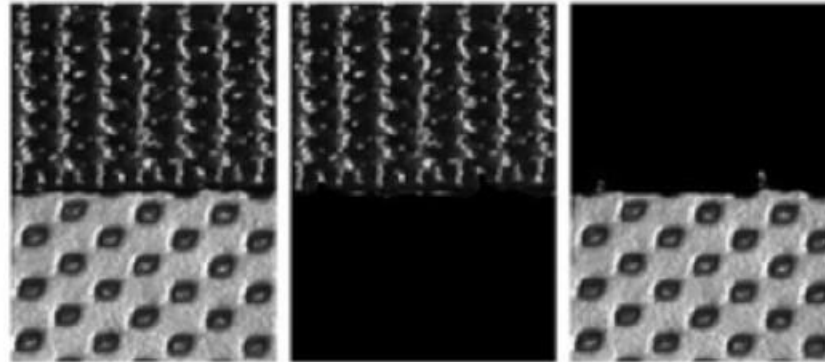
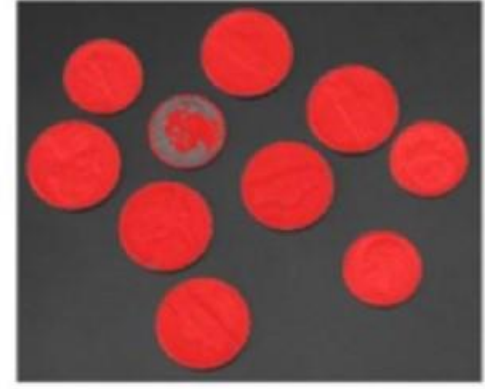


Image Source:
[mathworks](https://www.mathworks.com/)

Segmenting regions based on color values, shapes, or texture

- Several algorithms and techniques for image segmentation have been developed over the years using domain-specific knowledge to effectively solve segmentation problems in that specific application area; Including:
 - Medical Imaging
 - Autonomous Driving
 - video surveillance
 - Object Detection and Recognition
 - Biometrics
 - Satellite and Aerial Imaging
 - Augmented Reality and Virtual Reality

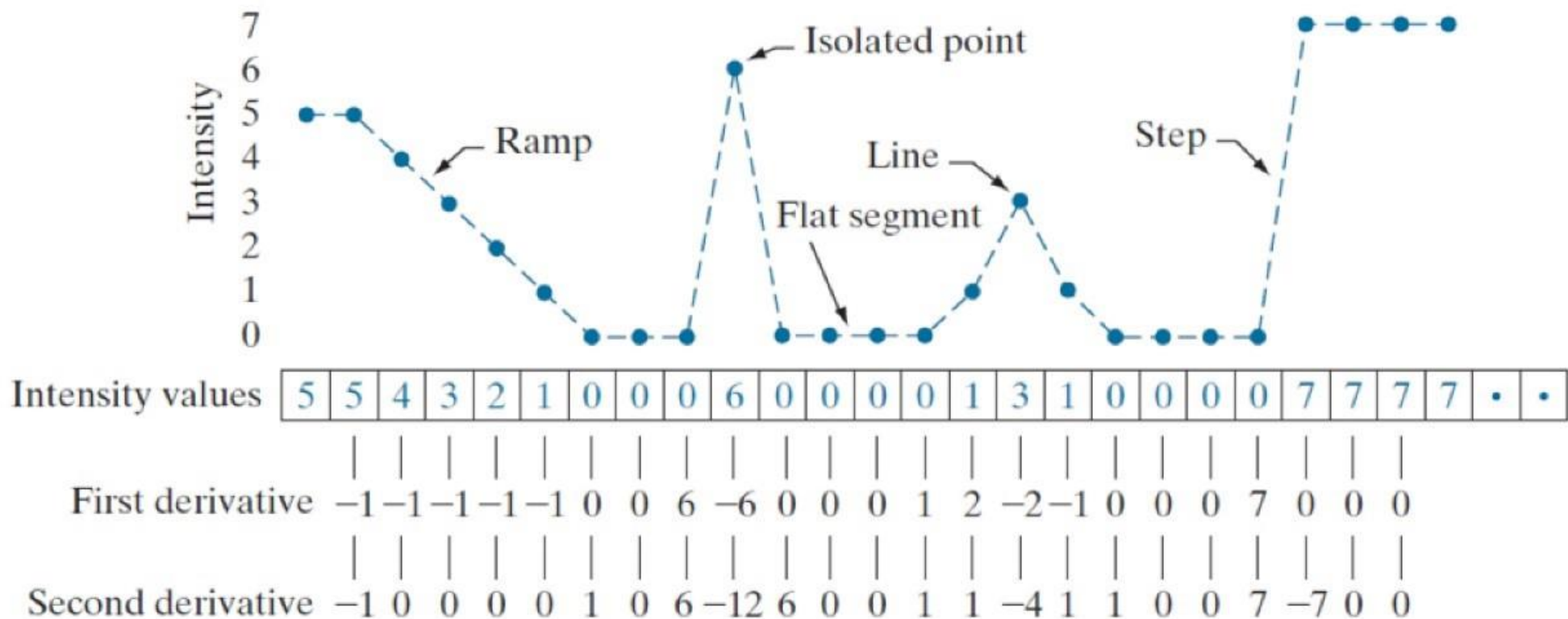
- **Discontinuity :**

- partition an image based on the sudden changes in intensity, such edges in an image.
- Discontinuity-based segmentation is effective for images containing well-defined edges or regions with significant intensity or color variations. However, it may struggle with noisy or textured images where edges are less distinct.

- **Similarity:**

- partitioning an image into regions that are similar according to a set of predefined criteria. Thresholding, region splitting and merging.
- Similarity-based segmentation is suitable for images with smooth gradients or regions with relatively uniform attributes. It is particularly effective for segmenting homogeneous regions or objects with consistent properties.

- There are three basic types of grey level discontinuities that we tend to look for in digital images:
- **–Points**
 - These represent isolated pixels in the image that exhibit a significant contrast or intensity difference compared to their surrounding pixels.
- **–Lines**
 - Lines refer to linear arrangements of pixels with a consistent intensity or color that stand out from the background.
- **–Edges**
 - Edges represent sharp transitions or boundaries between regions with different intensity levels or colors
- We typically find discontinuities using masks and correlation



A general 3 · 3 spatial filter kernel. The w 's are the kernel coefficients (weights).

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

$$\begin{aligned} Z &= w_1 z_1 + w_2 z_2 + \dots + w_9 z_9 \\ &= \sum_{k=1}^9 w_k z_k \end{aligned}$$

- Point detection can be achieved simply using the mask below:

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

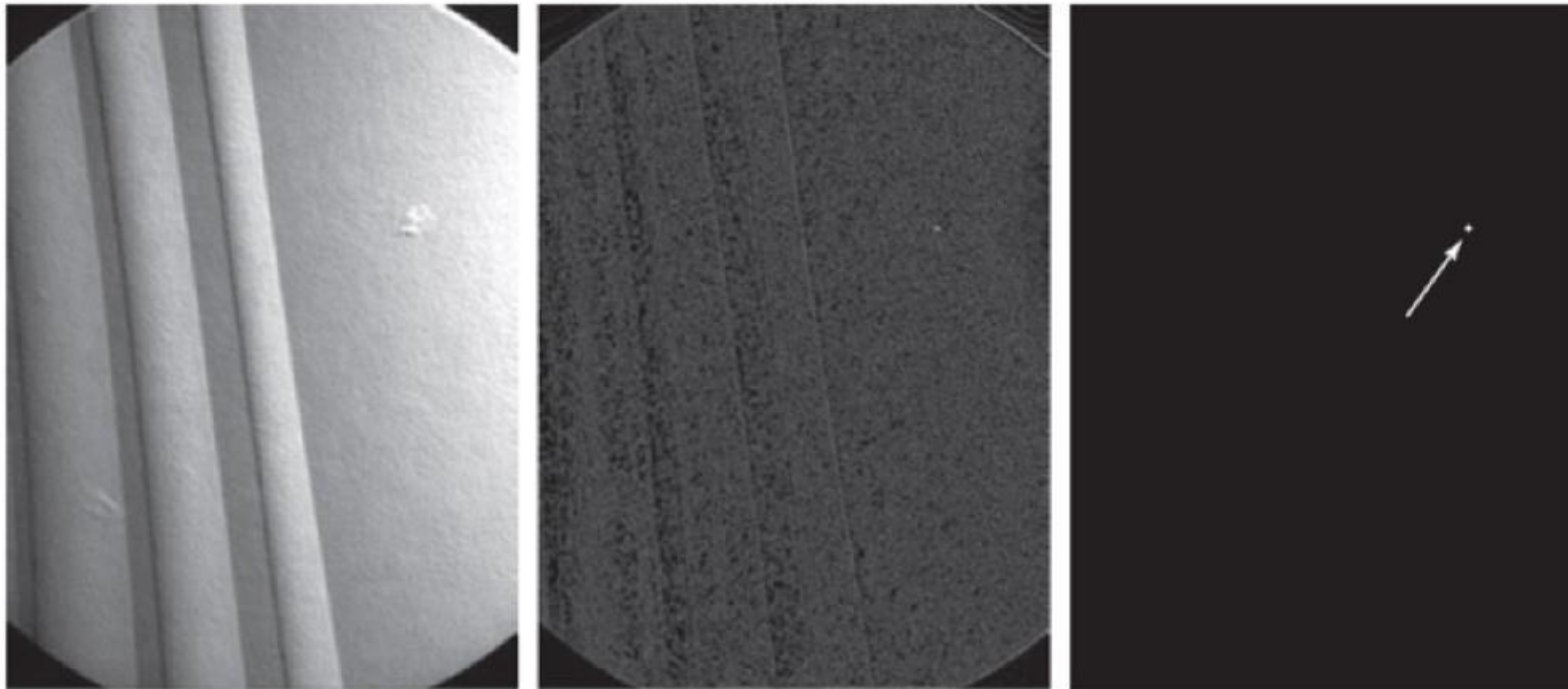
Laplacian Operator

- Points are detected at those pixels in the subsequent filtered image that are above a set threshold

Detection of Isolated Points

- Points are detected at those pixels in the subsequent filtered image that are above a set threshold

$$g(x, y) = \begin{cases} 1 & \text{if } |Z(x, y)| > T \\ 0 & \text{otherwise} \end{cases}$$



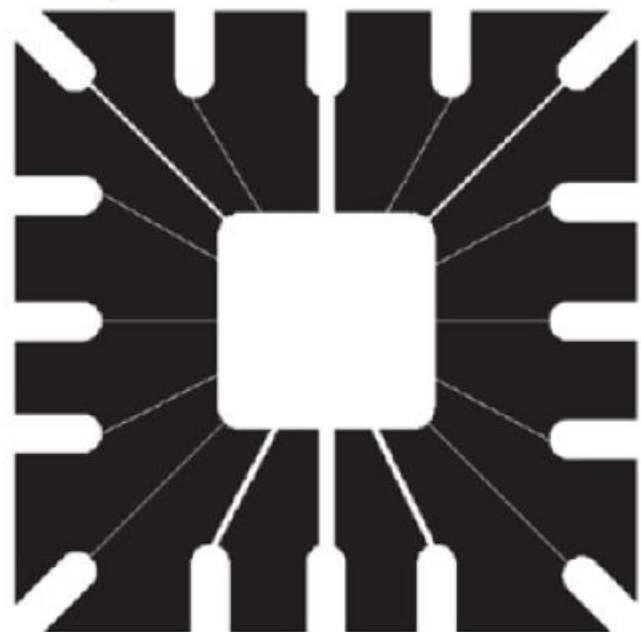
Detecting Lines in Specified Directions

- The masks below will extract lines that are one pixel thick and running in a particular direction

-1	-1	-1	2	-1	-1	-1	2	-1	-1	-1	2
2	2	2	-1	2	-1	-1	2	-1	-1	2	-1
-1	-1	-1	-1	-1	2	-1	2	-1	2	-1	-1
Horizontal			+45°			Vertical			-45°		

Detecting Lines in Specified Directions

- The masks below will extract lines that are one pixel thick and running in a particular direction



with the +45° line
detector kernel



All points (in white)
whose values satisfied
the condition $g > T$,

- An edge is a set of connected pixels that lie on the boundary between two regions.
- A **step edge** is characterized by a transition between two intensity levels occurring ideally over the distance of one pixel.
- A **ramp edge** is characterized by a gradual and smooth transition in intensity or color across adjacent regions in an image.
- A **roof edge** is a less common type of edge characterized by a combination of both step-like and ramp-like transitions in intensity or color.



A Step Edge

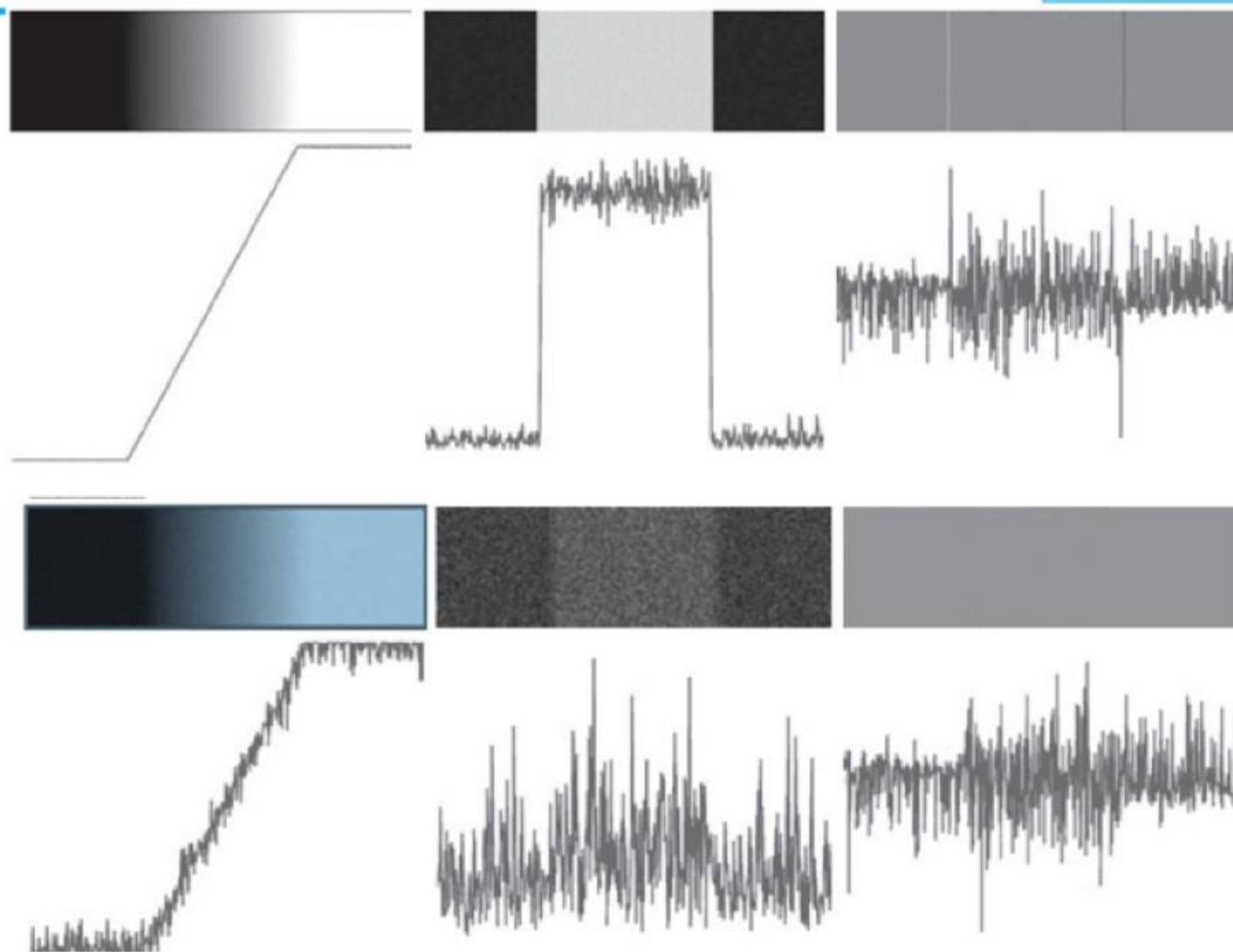


A Ramp Edge



A Roof Edge

Edge Detection and Noises



Steps for Edge Detection

The three steps performed typically for edge detection are:

- *Image smoothing for noise reduction.*
- *Detection of edge points.*
- *Edge localization.*

Filters for Edge Detection

-1	0	0	-1
0	1	1	0

Roberts

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

Prewitt

Given a 3*3 region of an image the following edge detection filters can be used

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

Sobel

Filters for Edge Detection

10	20	50	90
10	20	50	90
10	20	50	90
10	20	50	90

-1	0
0	1

Roberts

$$g(x, y) = 50 * -1 + 90 * 1 = 40$$

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

Prewitt

$$g(x, y) = 20 * -1 + 50 * -1 + 90 * -1 + 20 * 1 + 50 * 1 + 90 * 1 = 0$$

Filters for Edge Detection

10	20	50	90
10	20	50	90
10	20	50	90
10	20	50	90

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

Sobel

$$g(x, y) = 20 * -1 + 50 * -2 + 90 * -1 \\ + 20 * 1 + 50 * 2 + 90 * 1 = 0$$

- Know more about the *Kirsch compass kernels*? Why it is much better than introduced filters?

<table><tr><td>-3</td><td>-3</td><td>5</td></tr><tr><td>-3</td><td>0</td><td>5</td></tr><tr><td>-3</td><td>-3</td><td>5</td></tr></table>	-3	-3	5	-3	0	5	-3	-3	5	<table><tr><td>-3</td><td>5</td><td>5</td></tr><tr><td>-3</td><td>0</td><td>5</td></tr><tr><td>-3</td><td>-3</td><td>-3</td></tr></table>	-3	5	5	-3	0	5	-3	-3	-3	<table><tr><td>5</td><td>5</td><td>5</td></tr><tr><td>-3</td><td>0</td><td>-3</td></tr><tr><td>-3</td><td>-3</td><td>-3</td></tr></table>	5	5	5	-3	0	-3	-3	-3	-3	<table><tr><td>5</td><td>5</td><td>-3</td></tr><tr><td>5</td><td>0</td><td>-3</td></tr><tr><td>-3</td><td>-3</td><td>-3</td></tr></table>	5	5	-3	5	0	-3	-3	-3	-3
-3	-3	5																																					
-3	0	5																																					
-3	-3	5																																					
-3	5	5																																					
-3	0	5																																					
-3	-3	-3																																					
5	5	5																																					
-3	0	-3																																					
-3	-3	-3																																					
5	5	-3																																					
5	0	-3																																					
-3	-3	-3																																					
N	NW	W	SW																																				
<table><tr><td>5</td><td>-3</td><td>-3</td></tr><tr><td>5</td><td>0</td><td>-3</td></tr><tr><td>5</td><td>-3</td><td>-3</td></tr></table>	5	-3	-3	5	0	-3	5	-3	-3	<table><tr><td>-3</td><td>-3</td><td>-3</td></tr><tr><td>5</td><td>0</td><td>-3</td></tr><tr><td>5</td><td>5</td><td>-3</td></tr></table>	-3	-3	-3	5	0	-3	5	5	-3	<table><tr><td>-3</td><td>-3</td><td>-3</td></tr><tr><td>-3</td><td>0</td><td>-3</td></tr><tr><td>5</td><td>5</td><td>5</td></tr></table>	-3	-3	-3	-3	0	-3	5	5	5	<table><tr><td>-3</td><td>-3</td><td>-3</td></tr><tr><td>-3</td><td>0</td><td>5</td></tr><tr><td>-3</td><td>5</td><td>5</td></tr></table>	-3	-3	-3	-3	0	5	-3	5	5
5	-3	-3																																					
5	0	-3																																					
5	-3	-3																																					
-3	-3	-3																																					
5	0	-3																																					
5	5	-3																																					
-3	-3	-3																																					
-3	0	-3																																					
5	5	5																																					
-3	-3	-3																																					
-3	0	5																																					
-3	5	5																																					
S	SE	E	NE																																				

Edge Detection Example

Original Image



Horizontal Gradient Component



Vertical Gradient Component



Combined Edge Image

Often, problems arise in edge detection in that there are is too much detail (noise): For example, the brickwork in the previous example

- One way to overcome this is to smooth images prior to edge detection

Edge Detection Example With Smoothing

Original Image



Horizontal Gradient Component

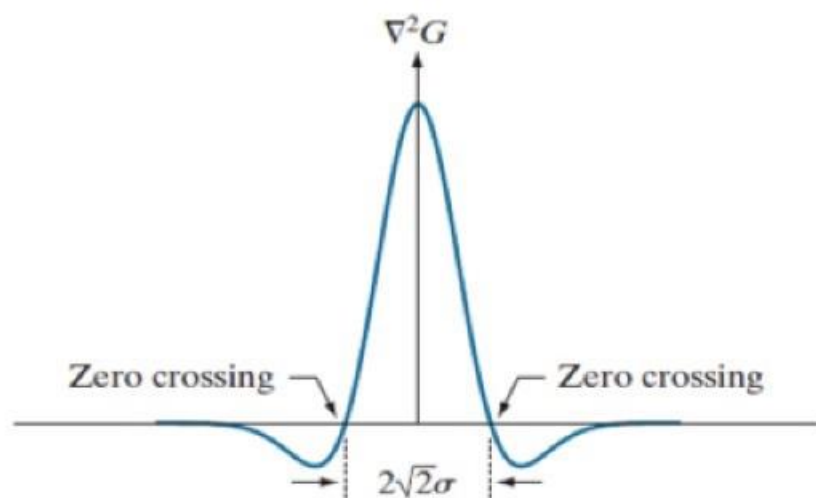
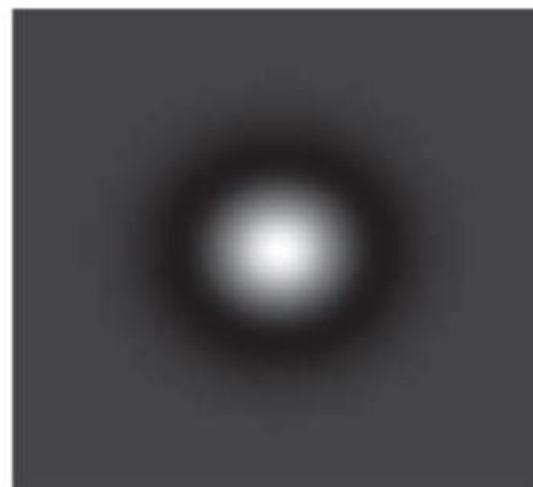
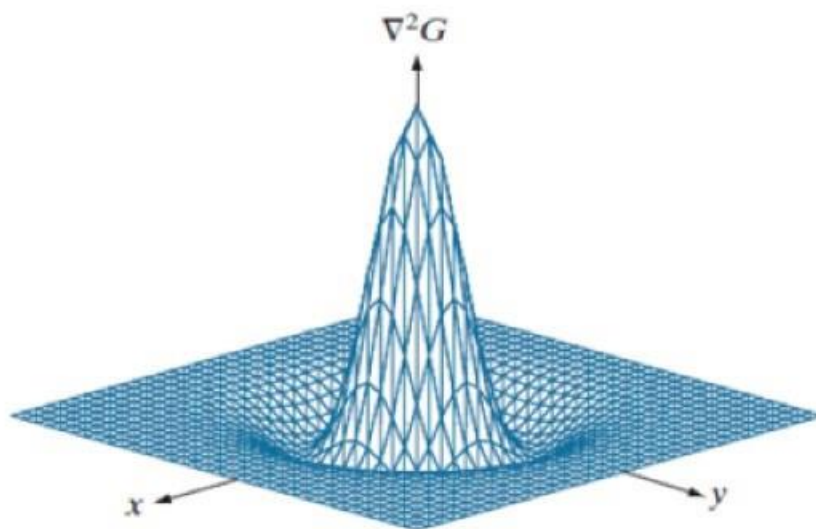


Vertical Gradient Component



Combined Edge Image

Laplacian of a Gaussian (LoG).



0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

The Marr-Hildreth algorithm

$$g(x, y) = [\nabla^2 G(x, y)] \star f(x, y)$$

$$g(x, y) = \nabla^2 [G(x, y) \star f(x, y)]$$

Laplacian of a Gaussian example



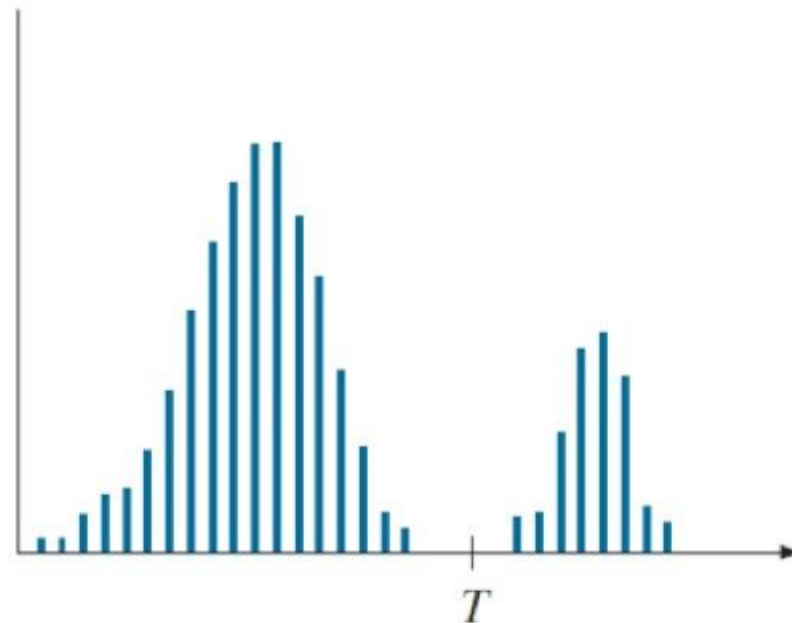
- Because of its intuitive properties, simplicity of implementation, and computational speed, image thresholding enjoys a central position in applications of image segmentation.
- way to extract the objects from the background is to select a threshold, T , that separates these modes. Then:
 - any point (x, y) in the image at which $f(x, y) > T$ is called an *object point*.
 - The rest of the points are called a *background point*.

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T \end{cases}$$

global thresholding

Multiple Thresholding

Single value thresholding only works for bimodal histograms
Images with other kinds of histograms need more than a single threshold

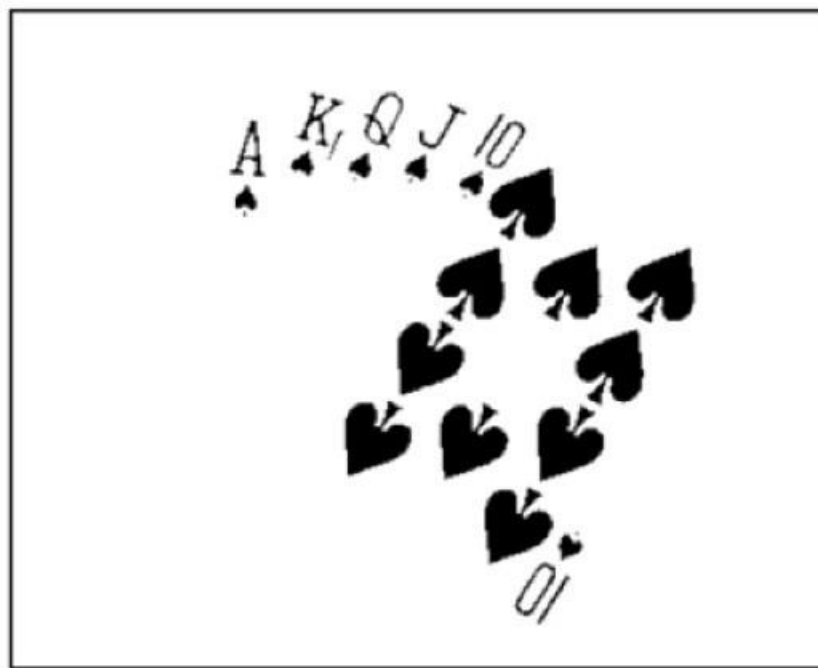


Thresholding Example

- Imagine a poker playing robot that needs to visually interpret the cards in its hand

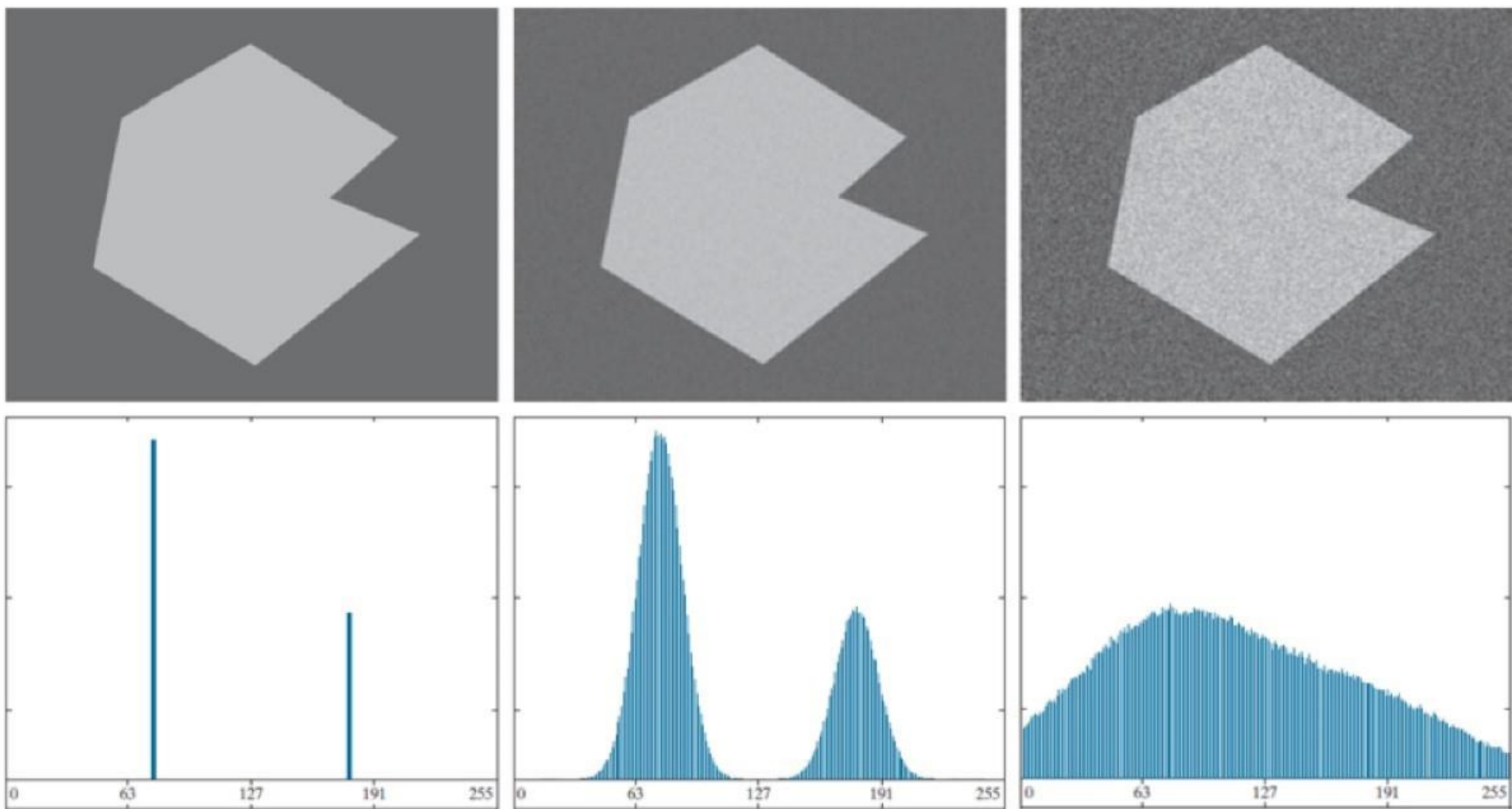


Original Image

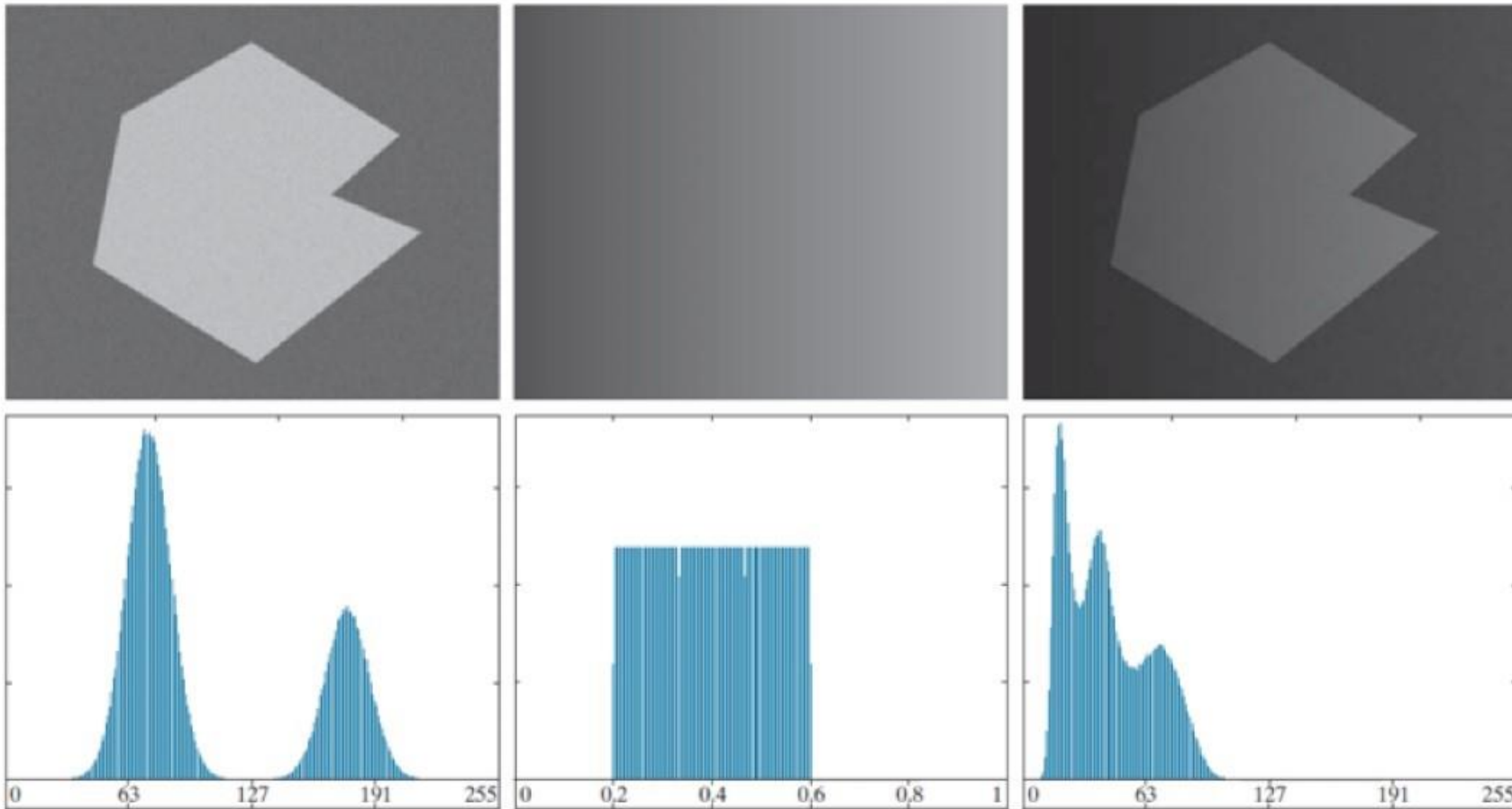


Thresholded Image

Noise and Image Thresholding



Illumination and Reflectance Effect



(a) Noisy image. (b) Intensity ramp in the range $[0.2, 0.6]$. (c) Product of (a) and (b). (d) through (f) Corresponding histograms.

Basic Global Thresholding Algorithm

1. Select an initial estimate for the global threshold, T .

2. Segment the image using T in Eq.

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{if } f(x,y) \leq T \end{cases}$$

This will produce two groups of

pixels: $G1$, consisting of pixels with intensity values $> T$; and $G2$, consisting of pixels with values $\leq T$.

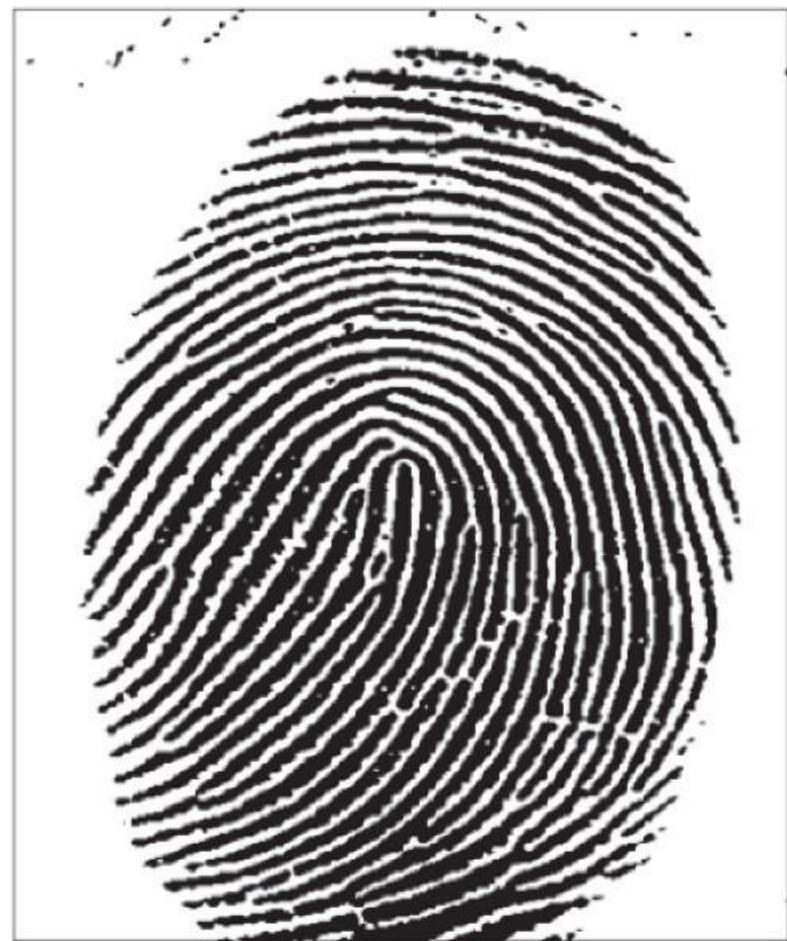
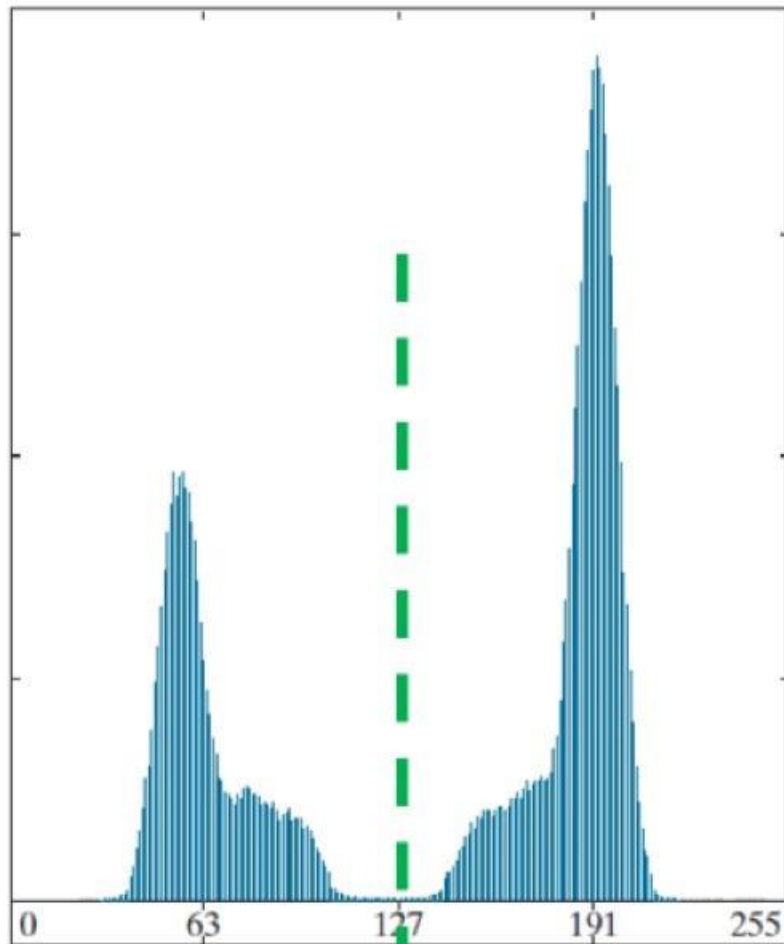
3. Compute the average (mean) intensity values $m1$ and $m2$ for the pixels in $G1$ and $G2$, respectively.

4. Compute a new threshold value midway between $m1$ and $m2$: $T = \frac{1}{2}(m_1 + m_2)$

5. Repeat Steps 2 through 4 until the difference between values of T in successive

6. iterations is smaller than a predefined value, T .

Basic Global Thresholding Example



Feature Extraction

Image descriptors

What is after Segmentation

- After an image has been segmented into regions or their boundaries, the resulting sets of segmented pixels usually have to be converted into a form suitable for further computer processing.
- The step after segmentation is *feature extraction*, which consists of:
 - ***feature detection*** : finding the features in an image, region, or boundary.
 - ***feature description***: assigns quantitative attributes to the detected features.

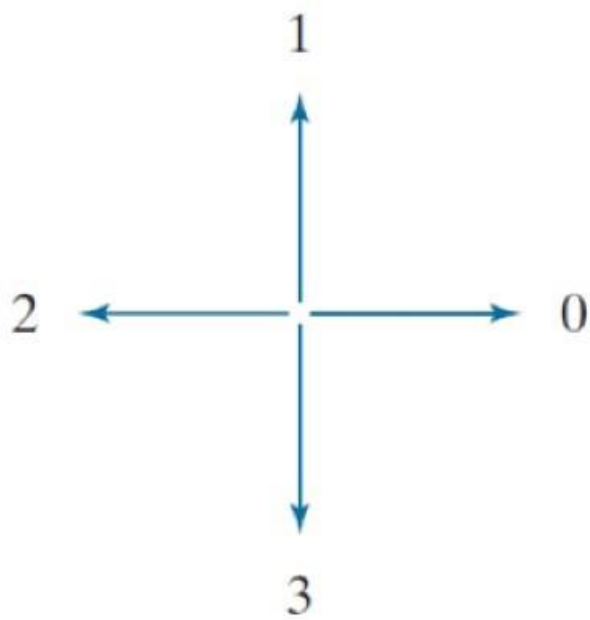
- Representations/Descriptions should be :
 - Translation (invariant)
 - Rotation (invariant)
 - Scale (covariant)
- Similar regions should have the same description regardless of their position or orientation in the image

- Generally two approaches
- **Boundary Processing (TRACING)**
 - Represent region by external characteristics (ie, the boundary)
- **Region Characteristics**
 - Represent region by internal characteristics

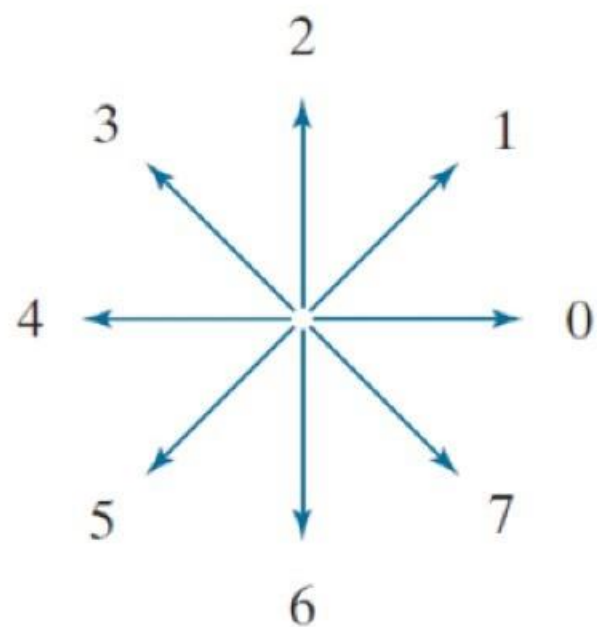
- Chain codes
- Others:
 - Polygonal Approximations
 - Signature

- Chain codes are used to represent a boundary
 - Uses a logically connected sequence of straight-line segments
 - The line segments specify direction (clockwise)
- Depending on the connectivity between neighbors the Direction coded using a number scheme either the 4- or 8-connectivity.
- A boundary code formed as a sequence of such directional numbers is referred to as a *Freeman chain code*.

Chain Codes

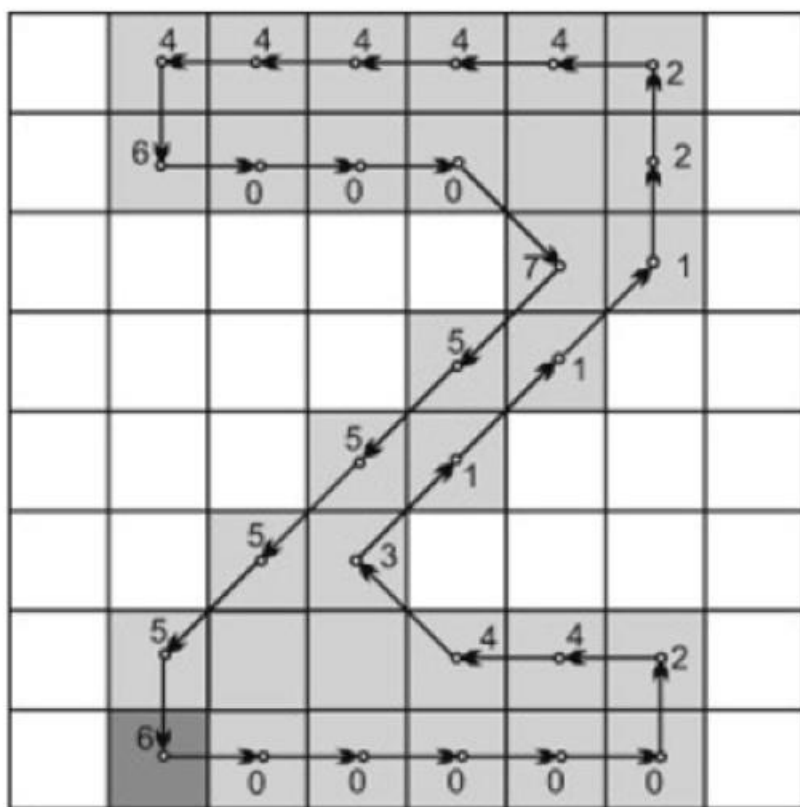


4-directional
chain code,

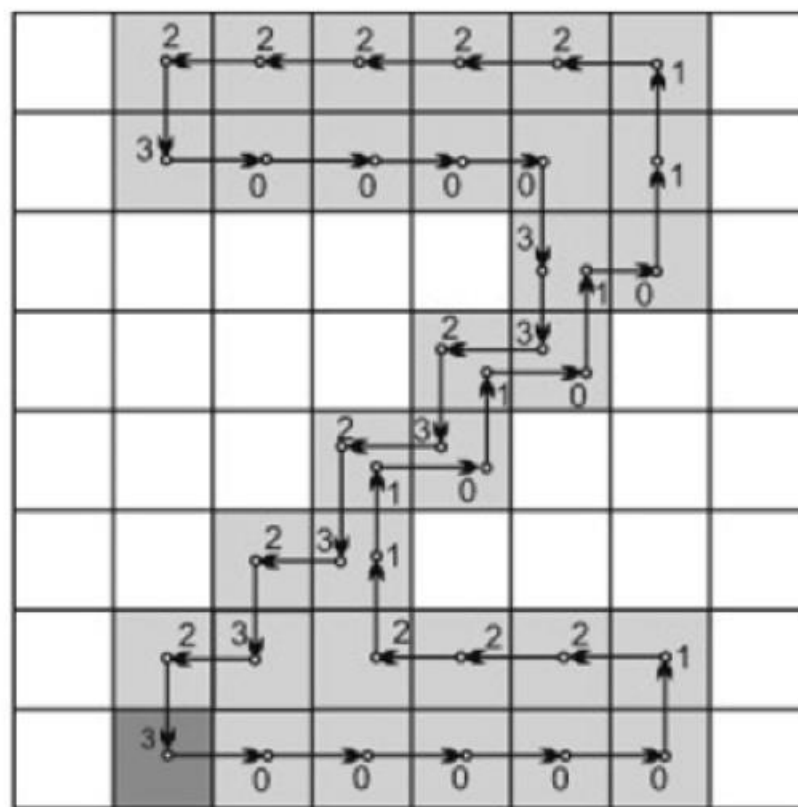


8-directional
chain code.

Chain Codes

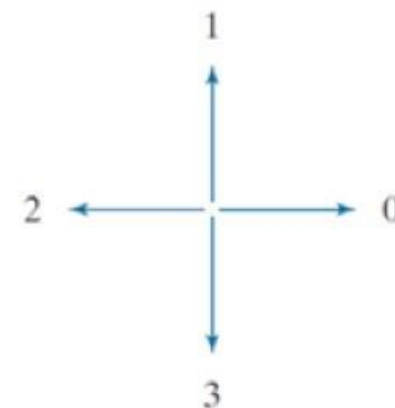


(a)



(b)

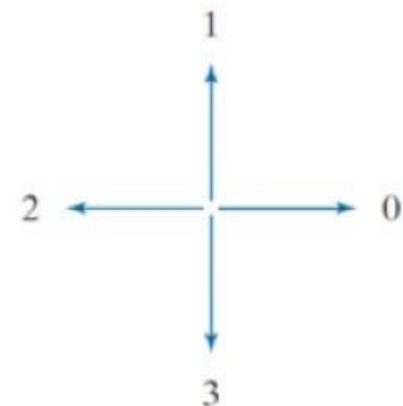
- Chain Code depends on the starting point
- We can normalize the chain code to address this problem
 - Assume the chain is a circular sequence
 - Redefine the starting point such that we generate an integer of smallest magnitude



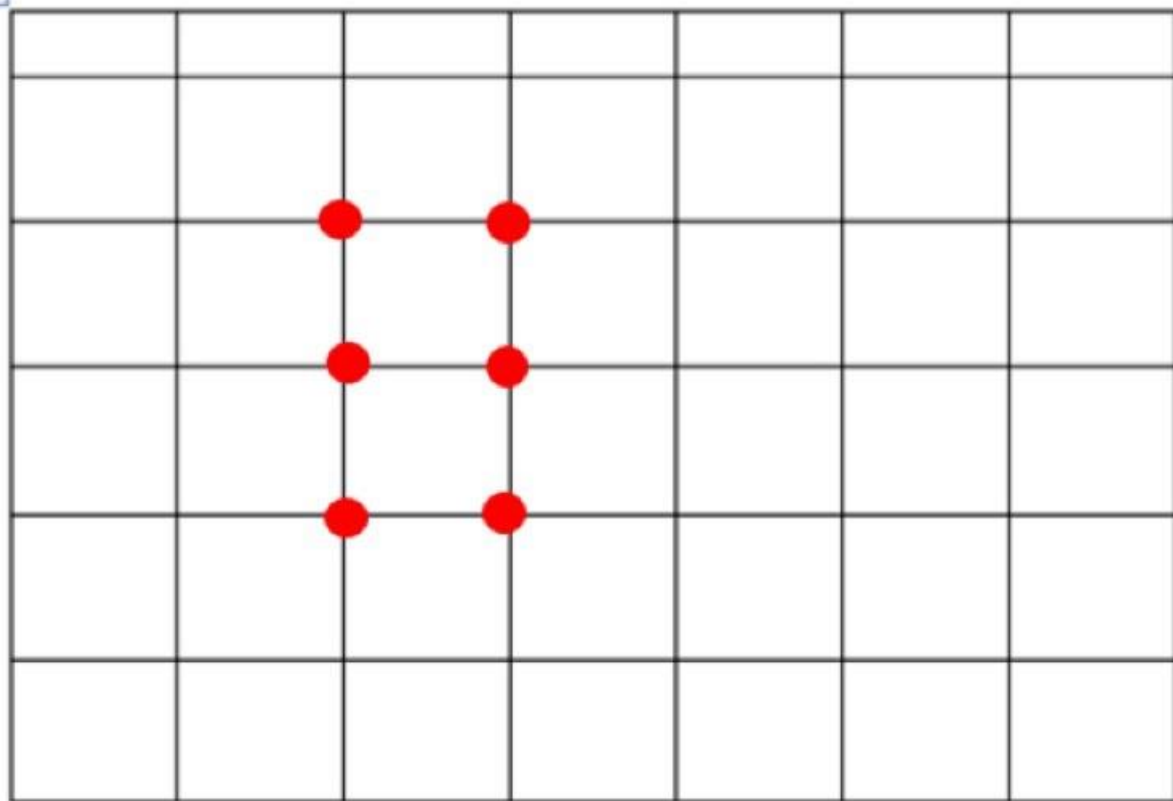
Normalize w.r.t the Starting point

- Chain code depends on orientation
 - a rotation results in a different chain code
- One solution is : Use the “first difference” of the chain code instead of the code itself
- The difference is obtained by simply counting (counter-clockwise) the number of directions that separate two adjacent elements

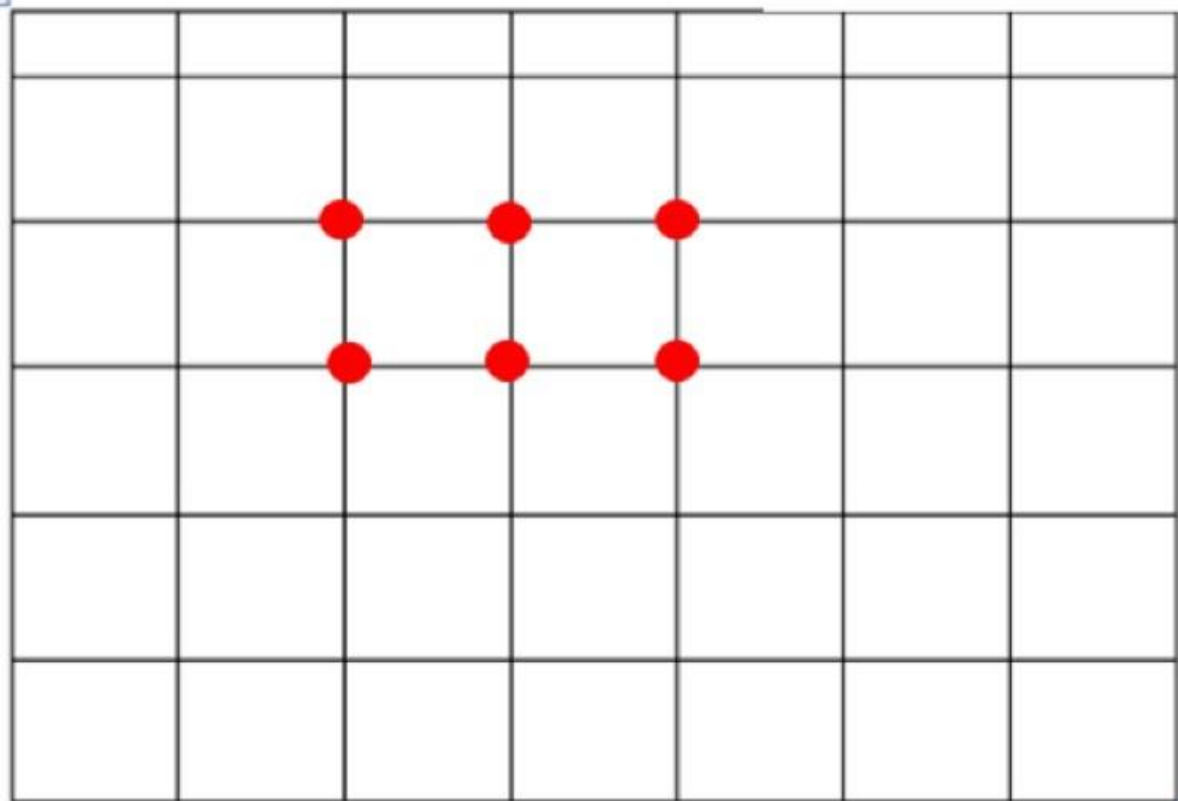
Chain Codes Example



+



+



Normalize w.r.t the Rotation



Thank You