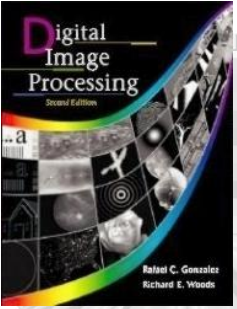
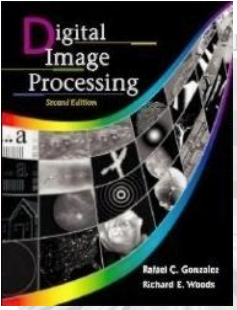
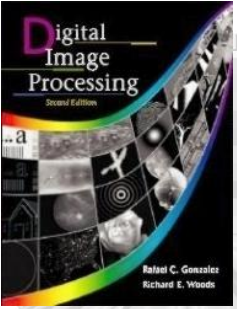


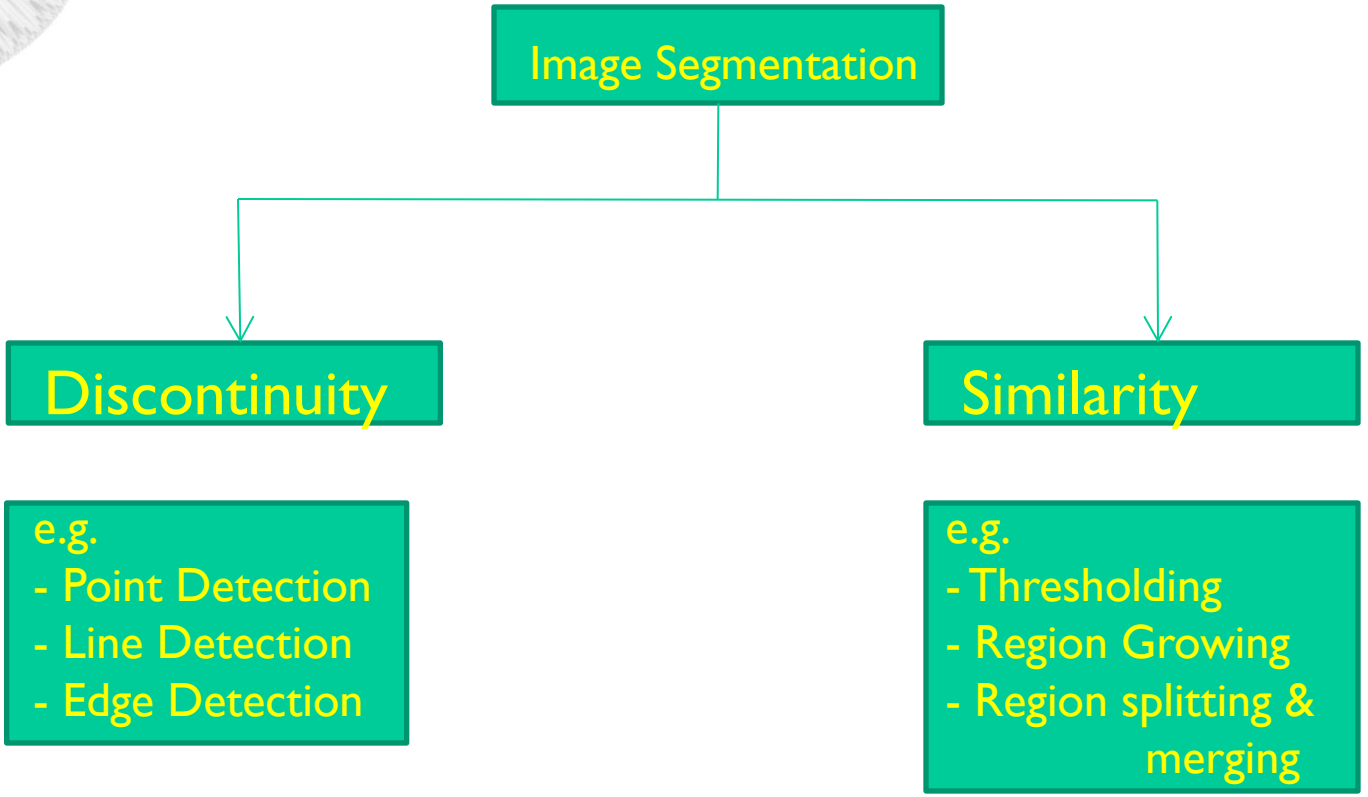
Chapter 10
Image Segmentation

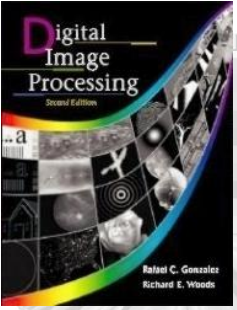


Chapter 10
Image Segmentation

- Image segmentation divides an image into regions that are connected and have some similarity within the region and some difference between adjacent regions.
- The goal is usually to find individual objects in an image.
- For the most part there are fundamentally two kinds of approaches to segmentation: discontinuity and similarity.
  - Similarity may be due to pixel intensity, color or texture.
  - Differences are sudden changes (discontinuities) in any of these, but especially sudden changes in intensity along a boundary line, which is called an edge.







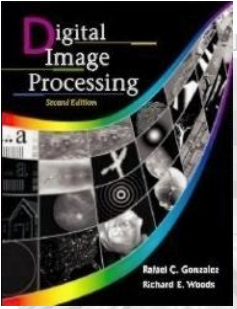
Detection of Discontinuities

- There are three kinds of discontinuities of intensity: points, lines and edges.
- The most common way to look for discontinuities is to scan a small mask over the image. The mask determines which kind of discontinuity to look for.

$$R = w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9 = \sum_{i=1}^{9} w_i z_i$$

FIGURE 10.1 A general 3×3 mask.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

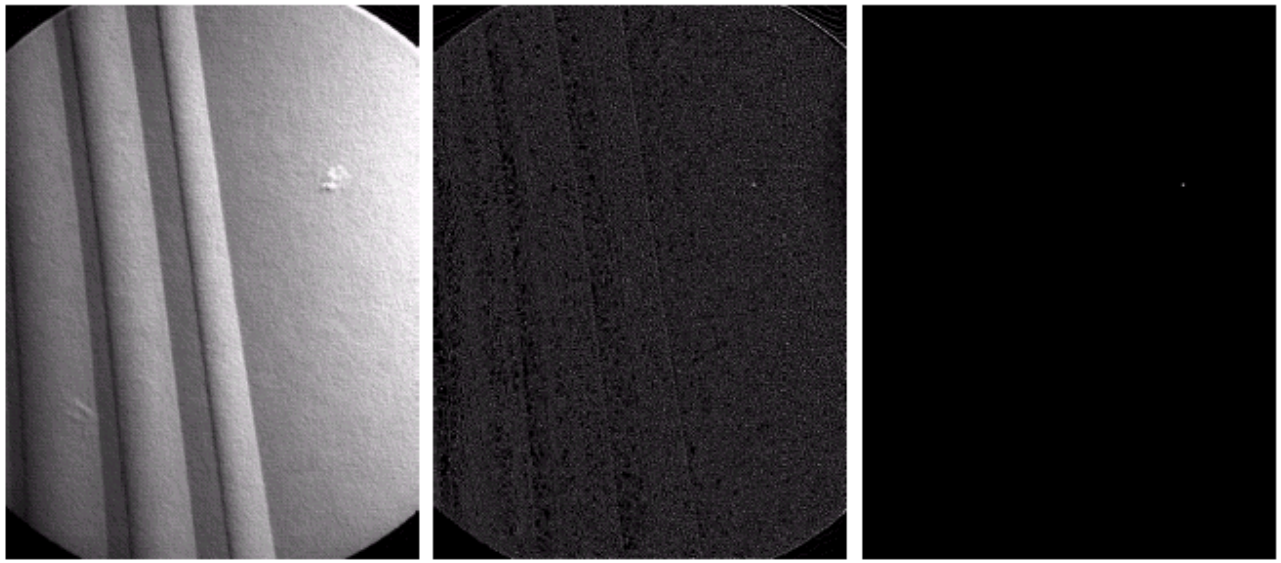


Detection of Discontinuities
Point Detection

$$|R| \geq T$$

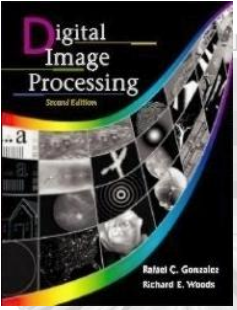
where T : a nonnegative threshold

−1	−1	−1
−1	8	−1
−1	−1	−1



a
b c d

FIGURE 10.2
(a) Point detection mask. (b) X-ray image of a turbine blade with a porosity. (c) Result of point detection. (d) Result of using Eq. (10.1-2). (Original image courtesy of X-TEK Systems Ltd.)

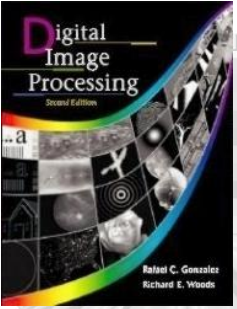


Detection of Discontinuities
Line Detection

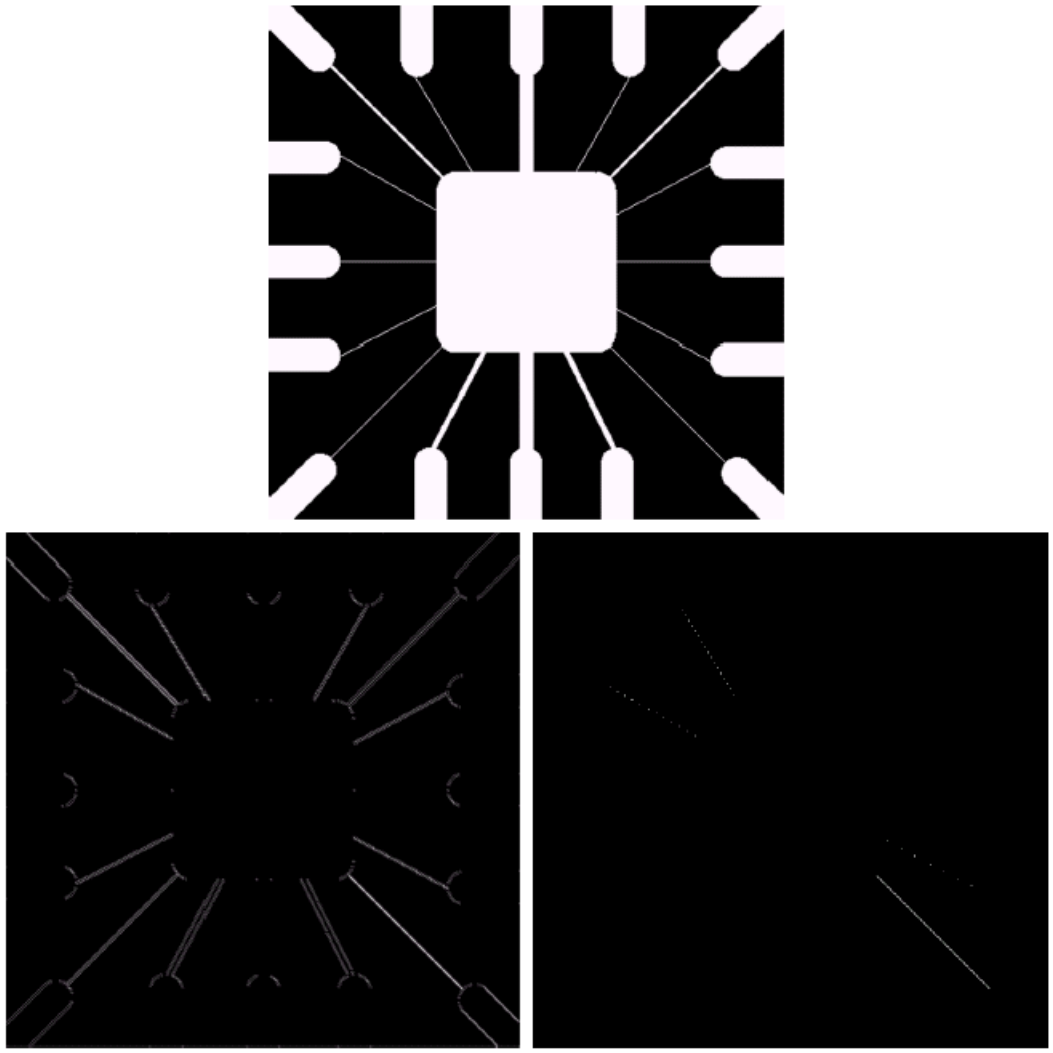
- Only slightly more common than point detection is to find a one pixel wide line in an image.
- For digital images the only three point straight lines are only horizontal, vertical, or diagonal (+ or –45°).

FIGURE 10.3 Line masks.

−1	−1	−1	−1	−1	2	−1	2	−1	2	−1	−1
2	2	2	−1	2	−1	−1	2	−1	−1	2	−1
−1	−1	−1	2	−1	−1	−1	2	−1	−1	−1	2
Horizontal			+45°			Vertical			−45°		

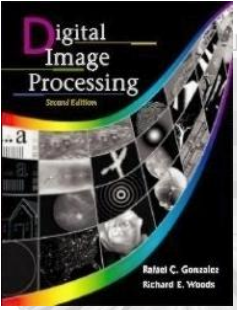


Detection of Discontinuities
Line Detection

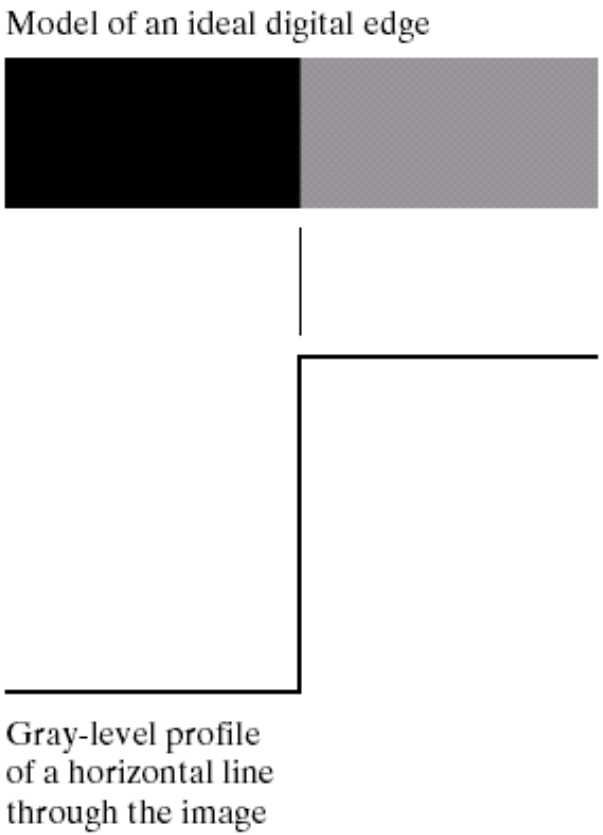


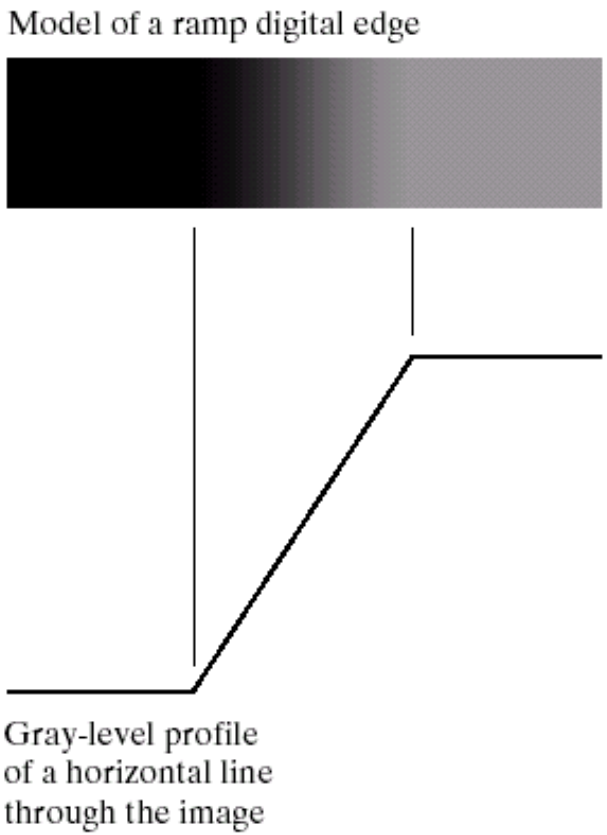
a
b c

FIGURE 10.4
Illustration of line detection.
(a) Binary wire-bond mask.
(b) Absolute value of result after processing with −45° line detector.
(c) Result of thresholding image (b).



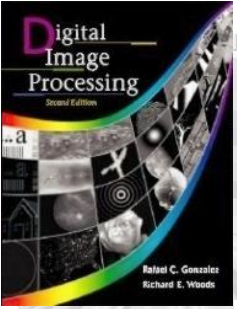
Detection of Discontinuities
Edge Detection





a b

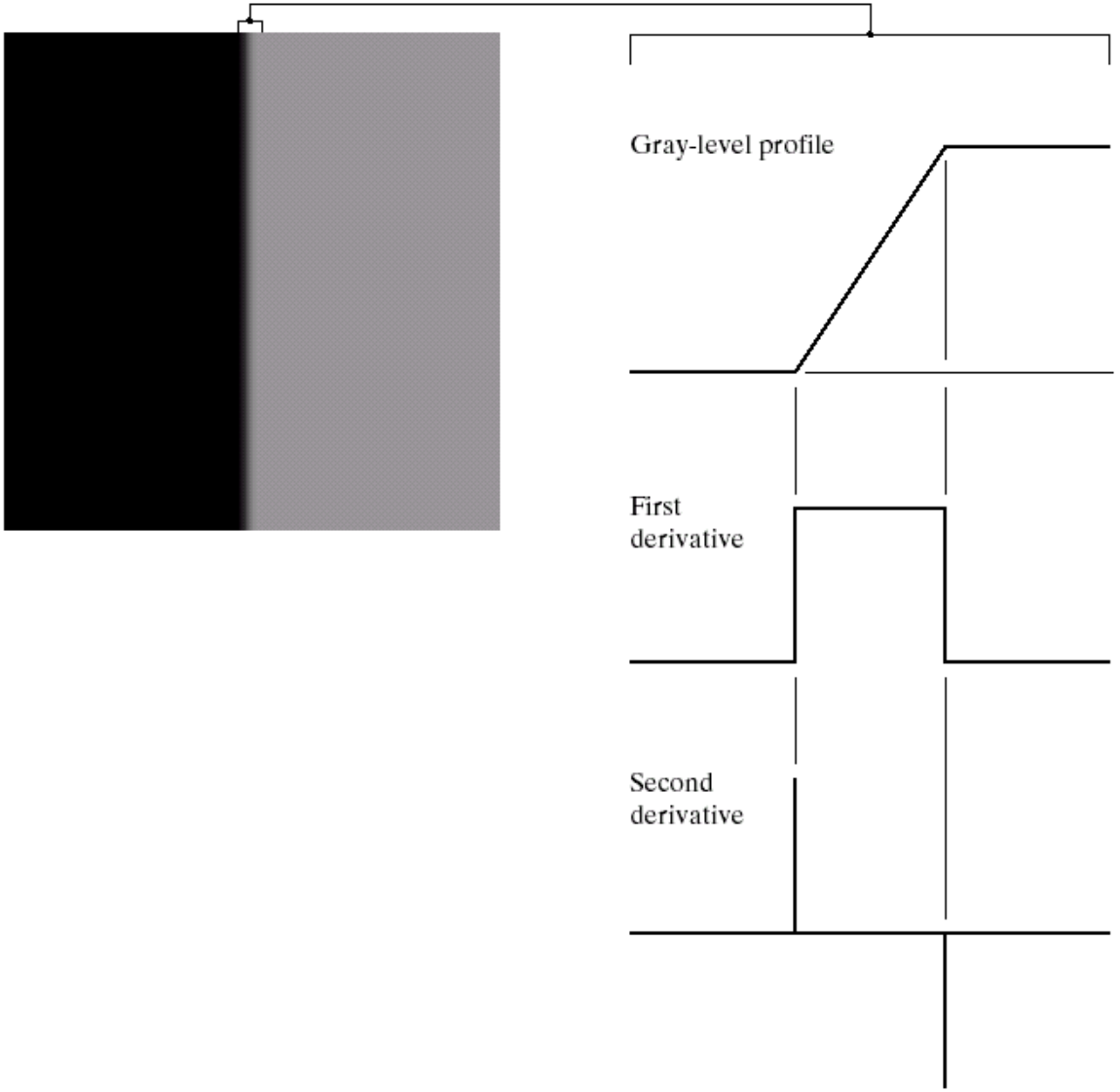
FIGURE 10.5
(a) Model of an ideal digital edge. (b) Model of a ramp edge. The slope of the ramp is proportional to the degree of blurring in the edge.

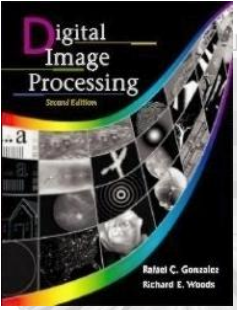


Detection of Discontinuities
Edge Detection

a b

FIGURE 10.6
(a) Two regions separated by a vertical edge. (b) Detail near the edge, showing a gray-level profile, and the first and second derivatives of the profile.





Detection of Discontinuities
Edge Detection

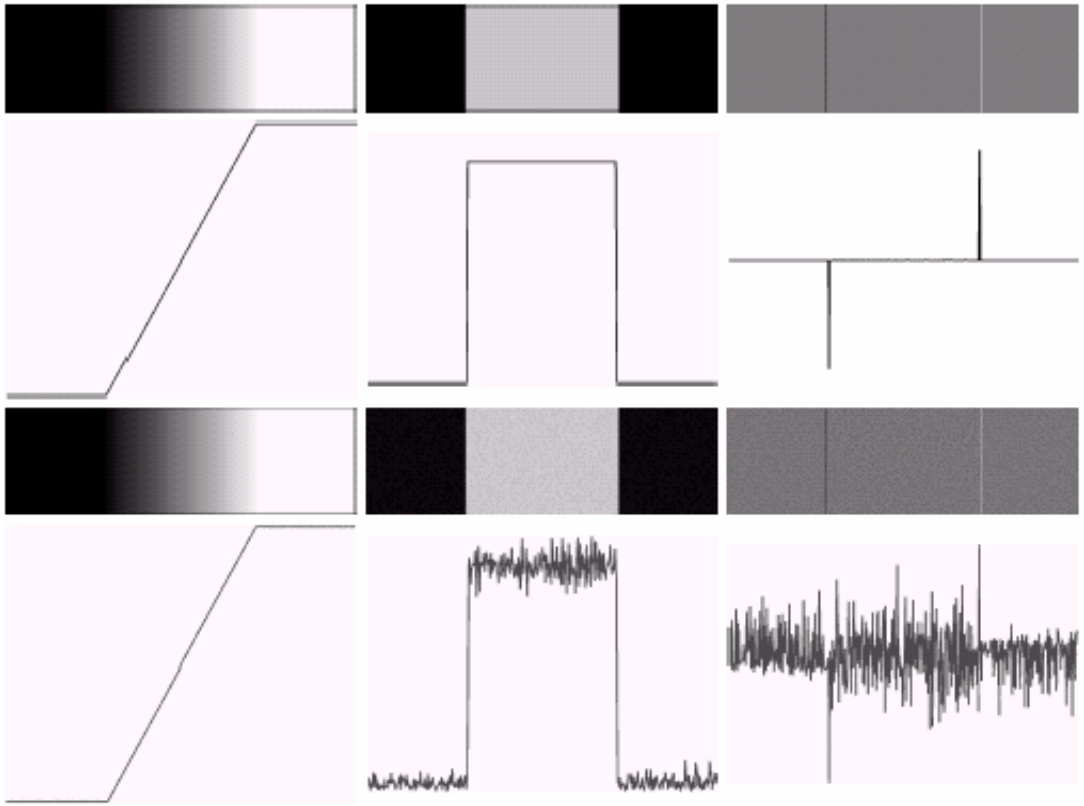
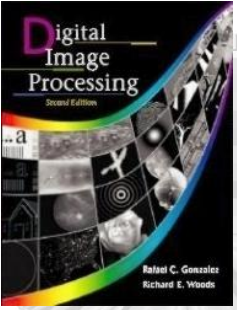


FIGURE 10.7 First column: images and gray-level profiles of a ramp edge corrupted by random Gaussian noise of mean 0 and $\sigma = 0.0, 0.1, 1.0,$ and $10.0,$ respectively. Second column: first-derivative images and gray-level profiles. Third column: second-derivative images and gray-level profiles.

a
b
c
d



Detection of Discontinuities
Edge Detection

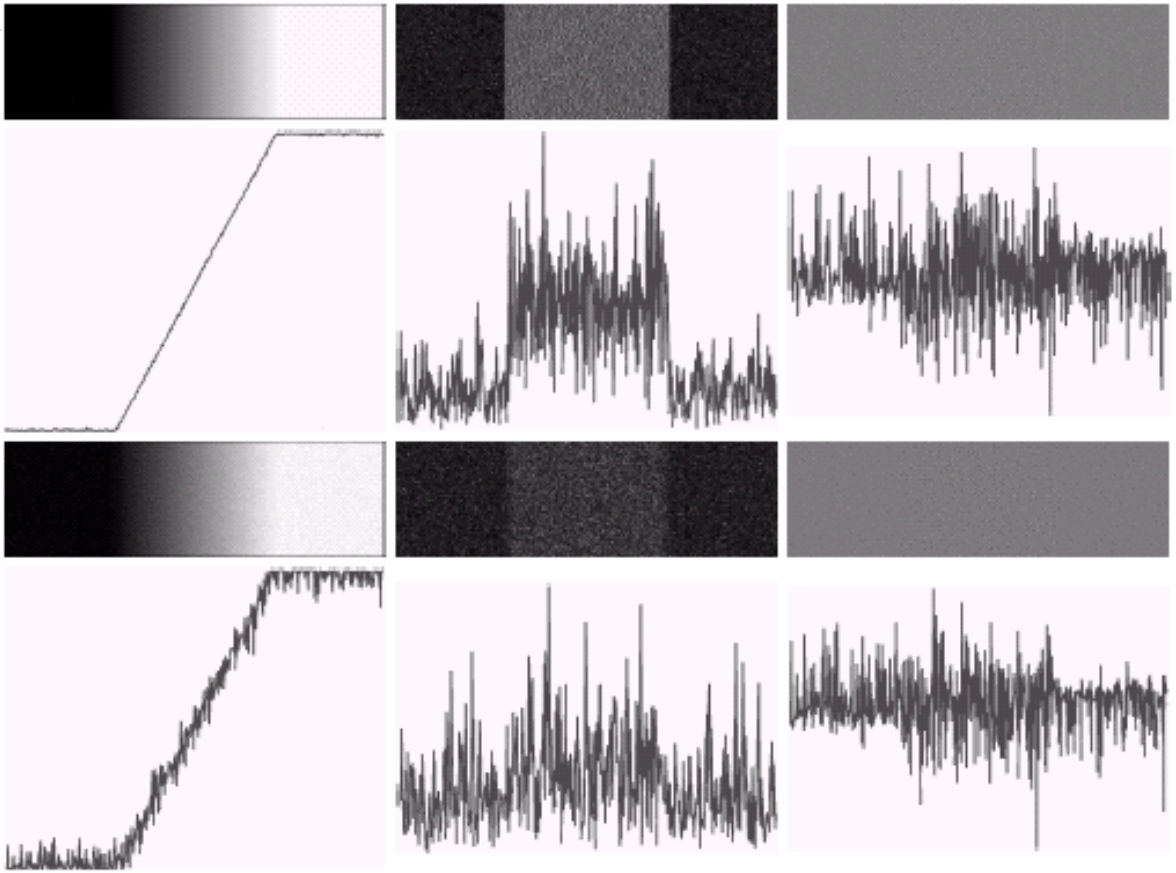
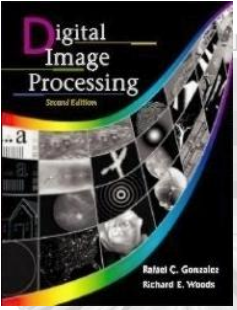


FIGURE 10.7 First column: images and gray-level profiles of a ramp edge corrupted by random Gaussian noise of mean 0 and $\sigma = 0.0, 0.1, 1.0,$ and $10.0,$ respectively. Second column: first-derivative images and gray-level profiles. Third column: second-derivative images and gray-level profiles.

a
b
c
d

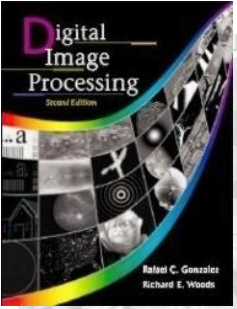


Detection of Discontinuities
Gradient Operators

- First-order derivatives:
  - The gradient of an image $f(x,y)$ at location (x,y) is defined as the vector:

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

  - The magnitude of this vector: $\nabla f = \text{mag}(\nabla \mathbf{f}) = \left[G_x^2 + G_y^2\right]^{1/2}$
  - The direction of this vector: $\alpha(x,y) = \tan^{-1}\left(\frac{G_x}{G_y}\right)$



Detection of Discontinuities
Gradient Operators

Roberts cross-gradient operators →

−1	0	0	−1
0	1	1	0

Roberts

Prewitt operators →

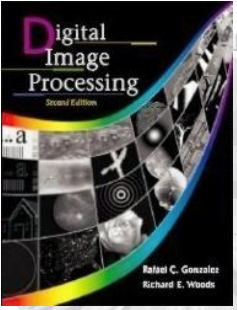
−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Prewitt

Sobel operators →

−1	−2	−1	−1	0	1
0	0	0	−2	0	2
1	2	1	−1	0	1

Sobel



Detection of Discontinuities
Gradient Operators

Prewitt masks for detecting diagonal edges

0	1	1	−1	−1	0
−1	0	1	−1	0	1
−1	−1	0	0	1	1

Prewitt

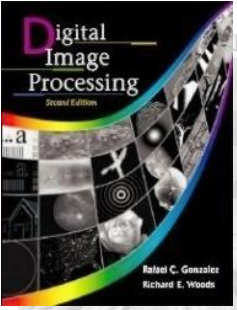
Sobel masks for detecting diagonal edges

0	1	2	−2	−1	0
−1	0	1	−1	0	1
−2	−1	0	0	1	2

Sobel

a b
c d

FIGURE 10.9 Prewitt and Sobel masks for detecting diagonal edges.



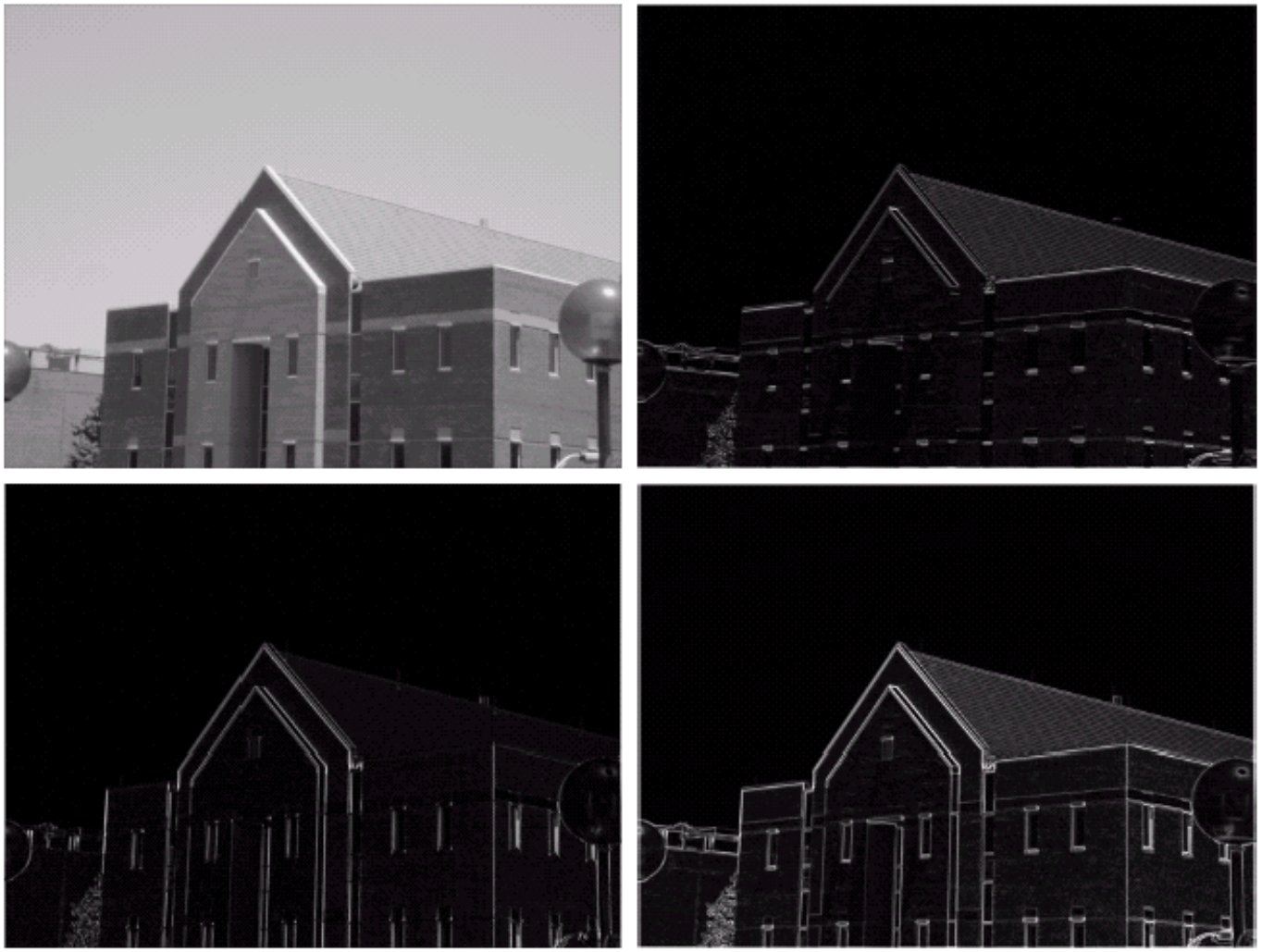
Detection of Discontinuities
Gradient Operators: Example

a b
c d

FIGURE 10.10
(a) Original image. (b) $|G_x|$, component of the gradient in the x-direction. (c) $|G_y|$, component in the y-direction. (d) Gradient image, $|G_x| + |G_y|$.

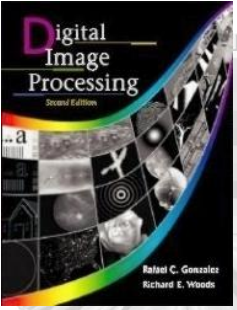
$$\nabla f \approx |G_x| + |G_y|$$

Detection of Discontinuities
Gradient Operators: Example



a b
c d

FIGURE 10.11
Same sequence as in Fig. 10.10, but with the original image smoothed with a 5×5 averaging filter.



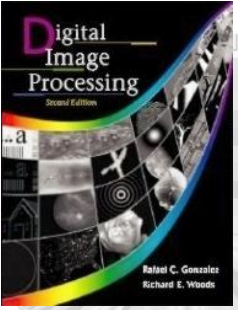
Detection of Discontinuities
Gradient Operators: Example

a b

FIGURE 10.12 Diagonal edge detection. (a) Result of using the mask in Fig. 10.9(c). (b) Result of using the mask in Fig. 10.9(d). The input in both cases was Fig. 10.11(a).

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2



Detection of Discontinuities
Gradient Operators

- Second-order derivatives: (The Laplacian)
  - The Laplacian of an 2D function $f(x,y)$ is defined as

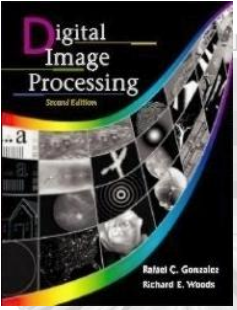
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

  - Two forms in practice:

FIGURE 10.13
Laplacian masks used to implement Eqs. (10.1-14) and (10.1-15), respectively.

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1



Detection of Discontinuities
Gradient Operators

- Consider the function:

A Gaussian function

$$h(r) = -e^{-\frac{r^2}{2\sigma^2}} \quad \text{where} \quad r^2 = x^2 + y^2$$

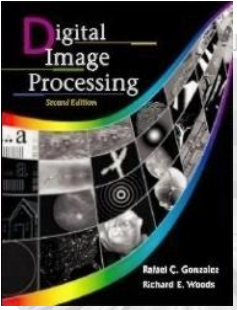
and σ : the standard deviation

- The Laplacian of h is

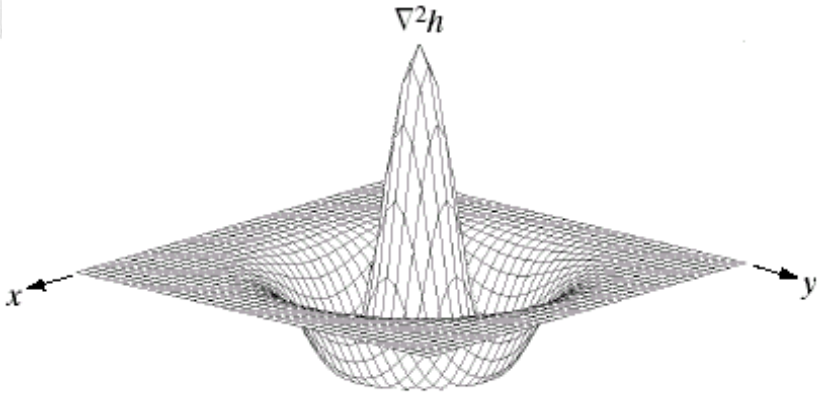
$$\nabla^2 h(r) = -\left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}}$$

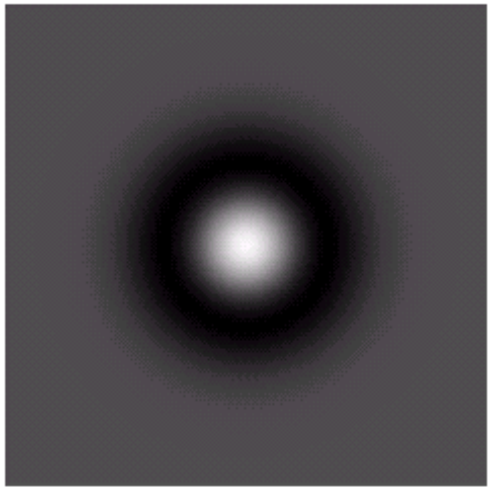
The Laplacian of a Gaussian (LoG)

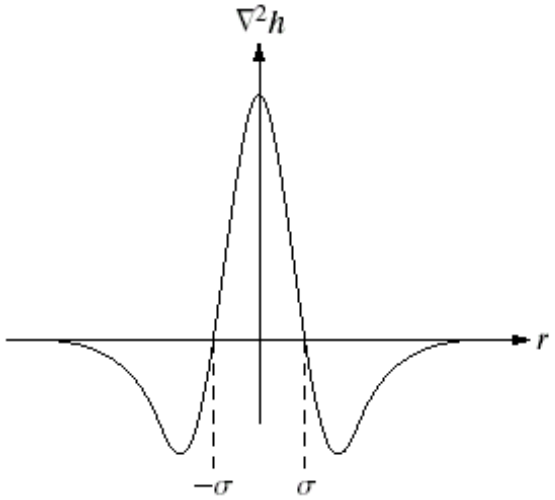
- The Laplacian of a Gaussian sometimes is called the Mexican hat function. It also can be computed by smoothing the image with the Gaussian smoothing mask, followed by application of the Laplacian mask.



Detection of Discontinuities
Gradient Operators



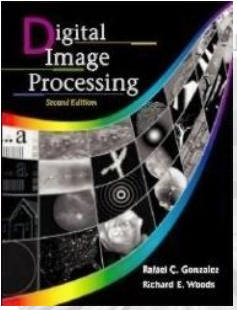




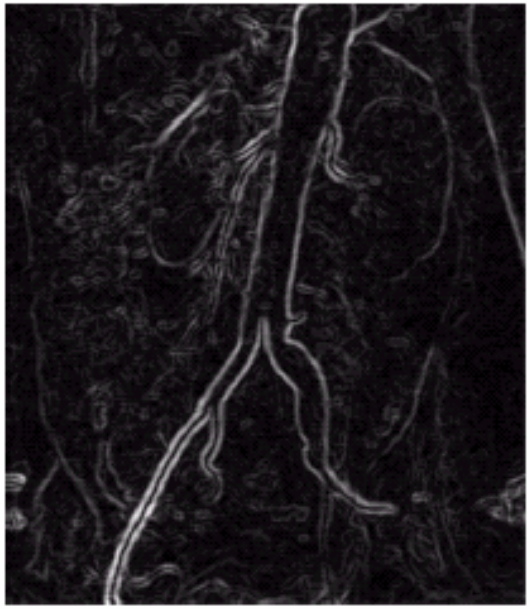
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

a b
c d

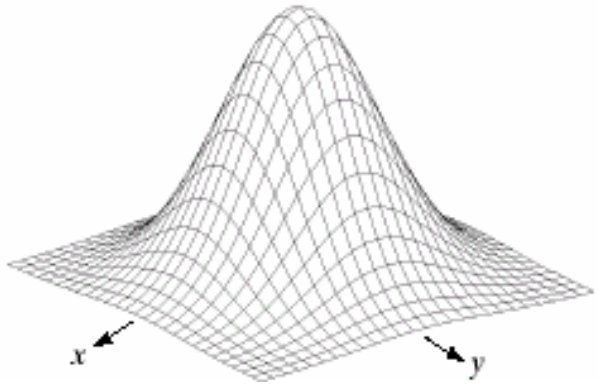
FIGURE 10.14 Laplacian of a Gaussian (LoG). (a) 3-D plot. (b) Image (black is negative, gray is the zero plane, and white is positive). (c) Cross section showing zero crossings. (d) 5 × 5 mask approximation to the shape of (a).



Detection of Discontinuities
Gradient Operators: Example



Sobel gradient

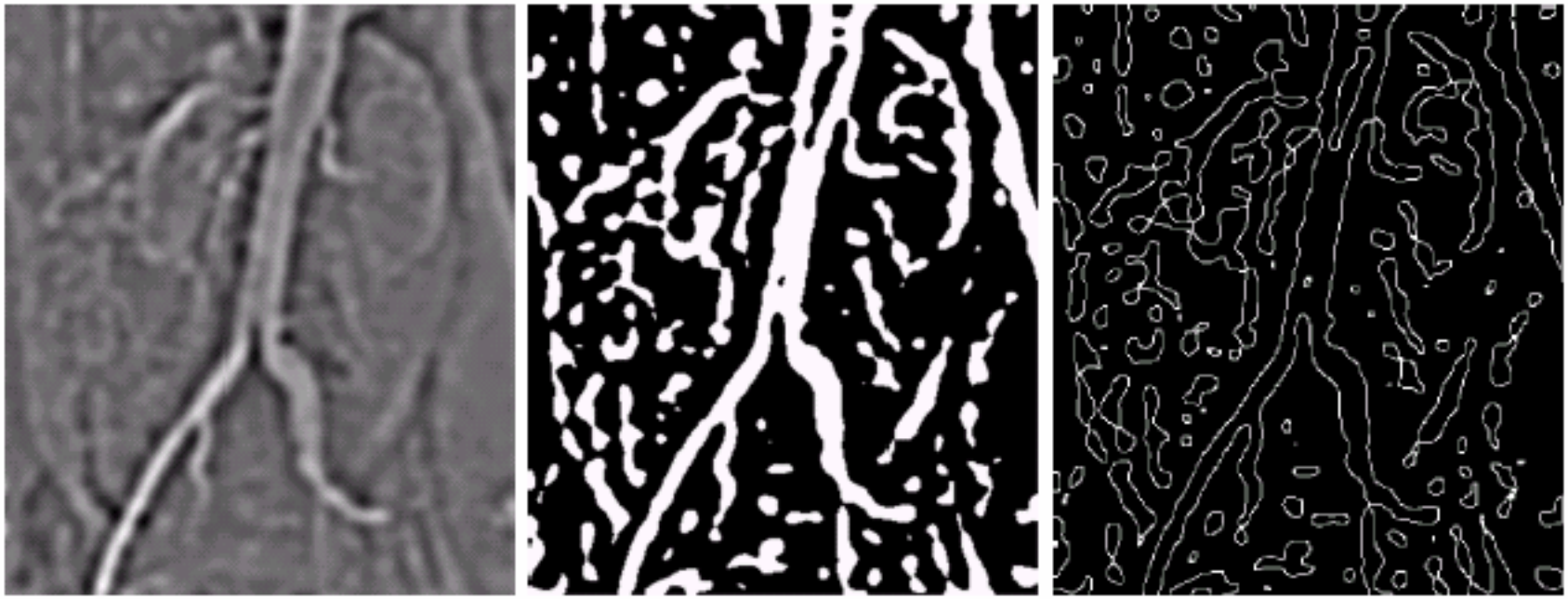


Gaussian smooth function

−1	−1	−1
−1	8	−1
−1	−1	−1

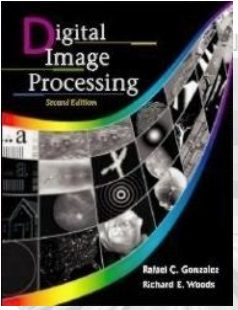
Laplacian mask

Detection of Discontinuities
Gradient Operators: Example



a b
c d
e f g

FIGURE 10.15 (a) Original image. (b) Sobel gradient (shown for comparison). (c) Spatial Gaussian smoothing function. (d) Laplacian mask. (e) LoG. (f) Thresholded LoG. (g) Zero crossings. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

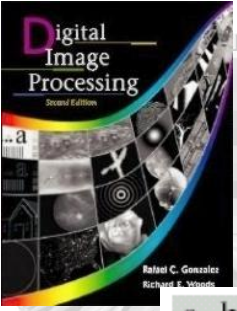


Edge Linking and Boundary Detection
Local Processing

- Two properties of edge points are useful for edge linking:
  - the strength (or magnitude) of the detected edge points
  - their directions (determined from gradient directions)
- This is usually done in local neighborhoods.
- Adjacent edge points with similar magnitude and direction are linked.
- For example, an edge pixel with coordinates (x_0,y_0) in a predefined neighborhood of (x,y) is similar to the pixel at (x,y) if

$$|\nabla f(x,y) - \nabla(x_0,y_0)| \le E, \quad E: \text{a nonnegative threshold}$$

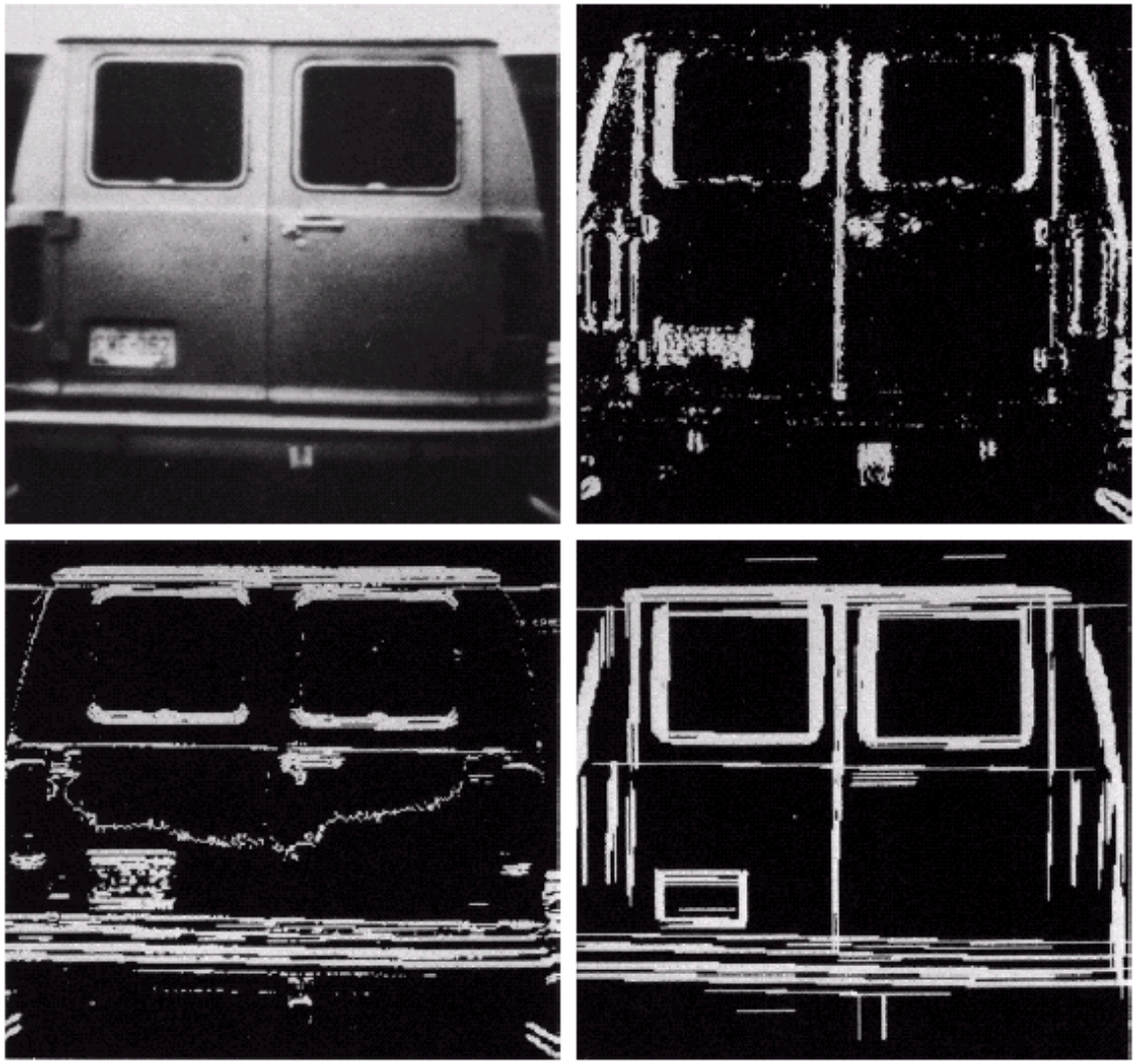
$$|\alpha(x,y) - \alpha(x_0,y_0)| < A, \quad A: \text{a nonegative angle threshold}$$



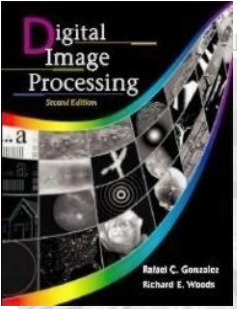
Edge Linking and Boundary Detection
Local Processing: Example

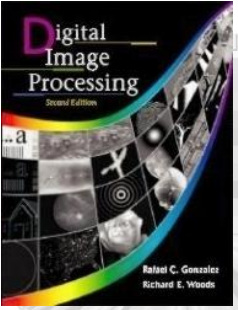
a b
c d

FIGURE 10.16
(a) Input image.
(b) G_y component of the gradient.
(c) G_x component of the gradient.
(d) Result of edge linking. (Courtesy of Perceptics Corporation.)



In this example, we can find the license plate candidate after edge linking process.

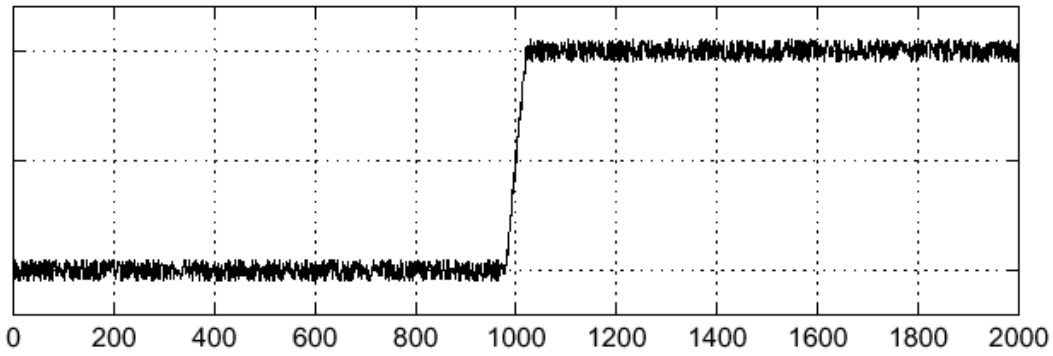




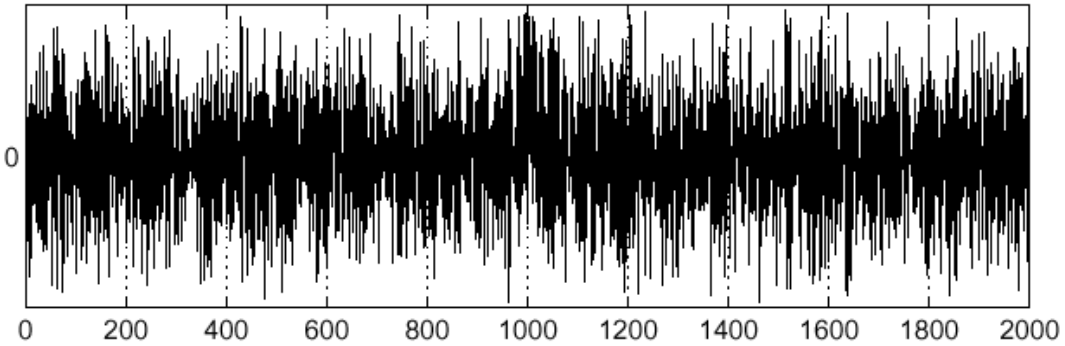
Effects of Noise

- Consider a single row or column of the image
  - Plotting

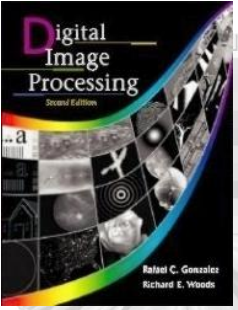
$f(x)$



$\frac{d}{dx}f(x)$



Where is the edge??



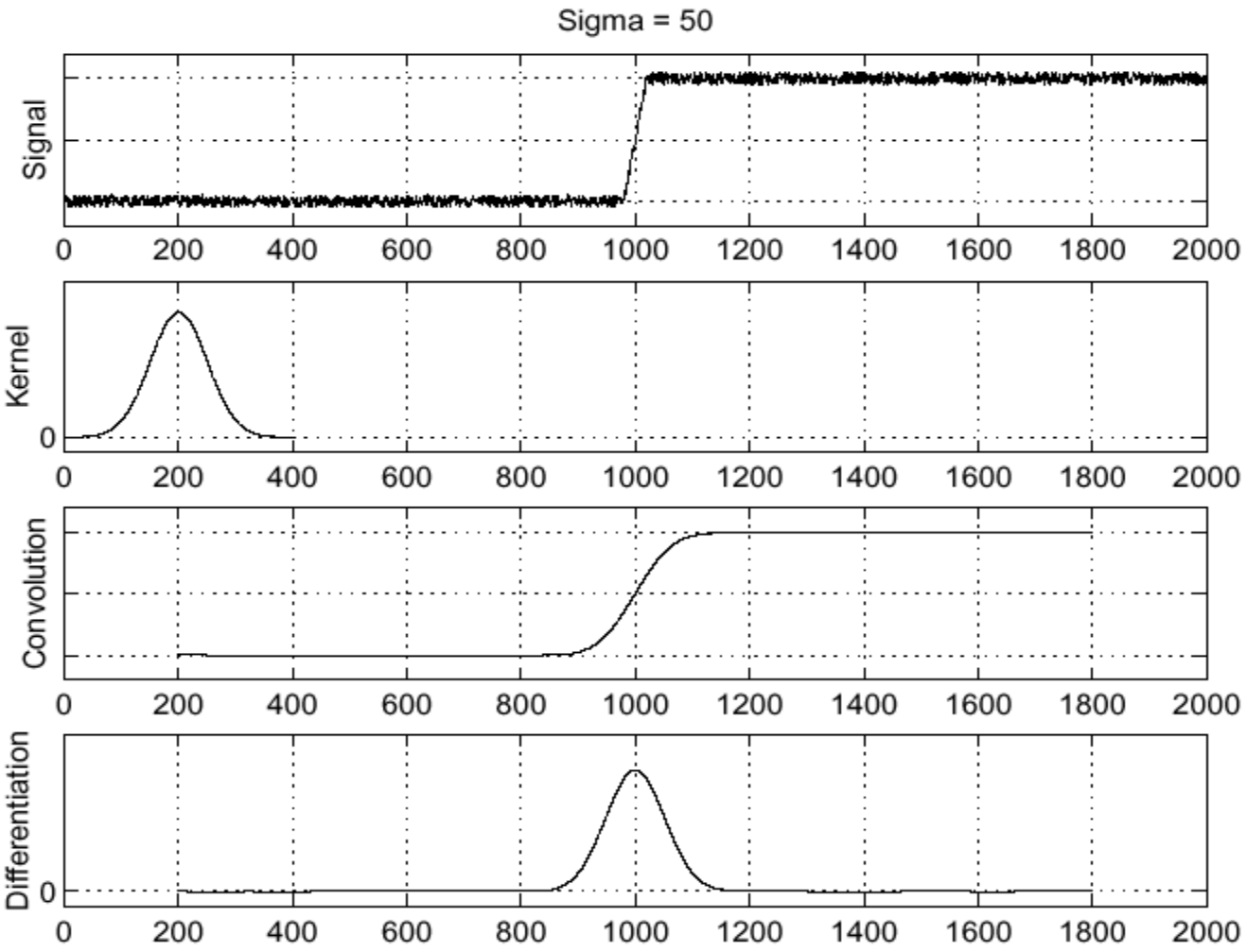
Solution: Smooth First

f

h

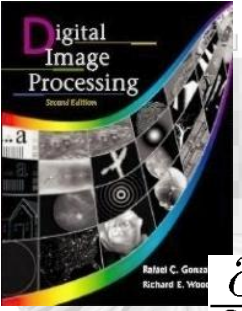
$h \star f$

$\frac{\partial}{\partial x}(h \star f)$



Where is the edge?

Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

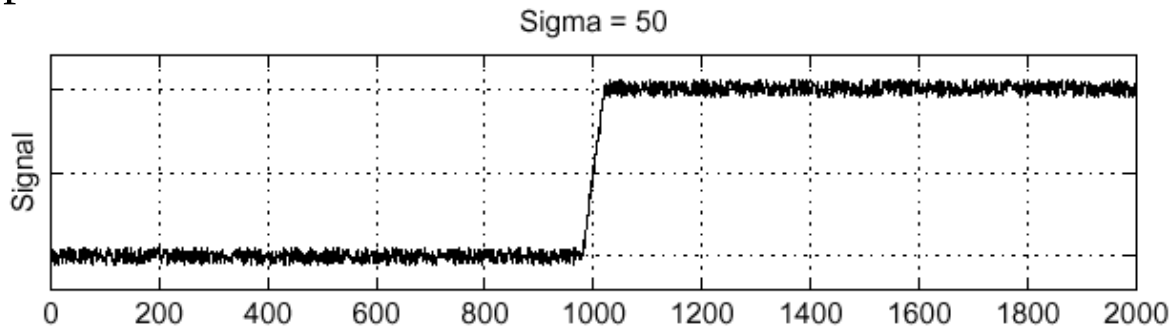


Derivative Theorem of Convolution

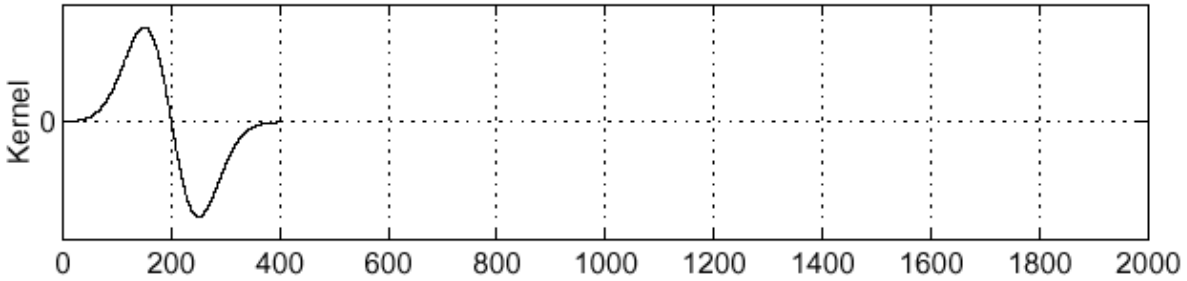
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

…saves us one operation.

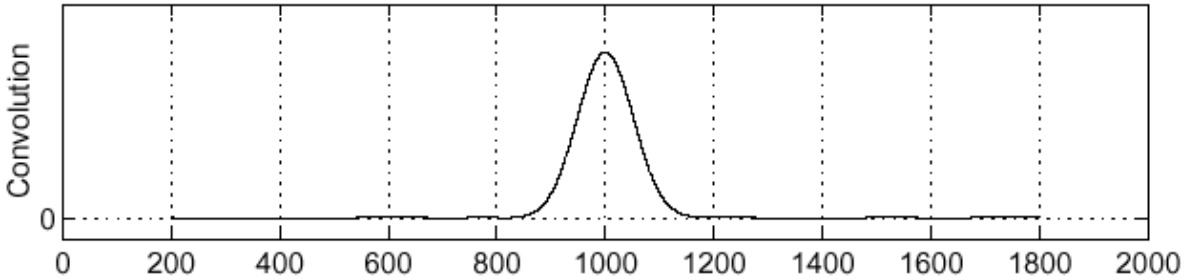
f

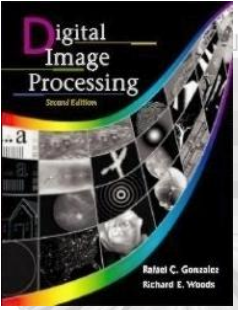


$\frac{\partial}{\partial x}h$



$(\frac{\partial}{\partial x}h) \star f$





Laplacian of Gaussian (LoG)

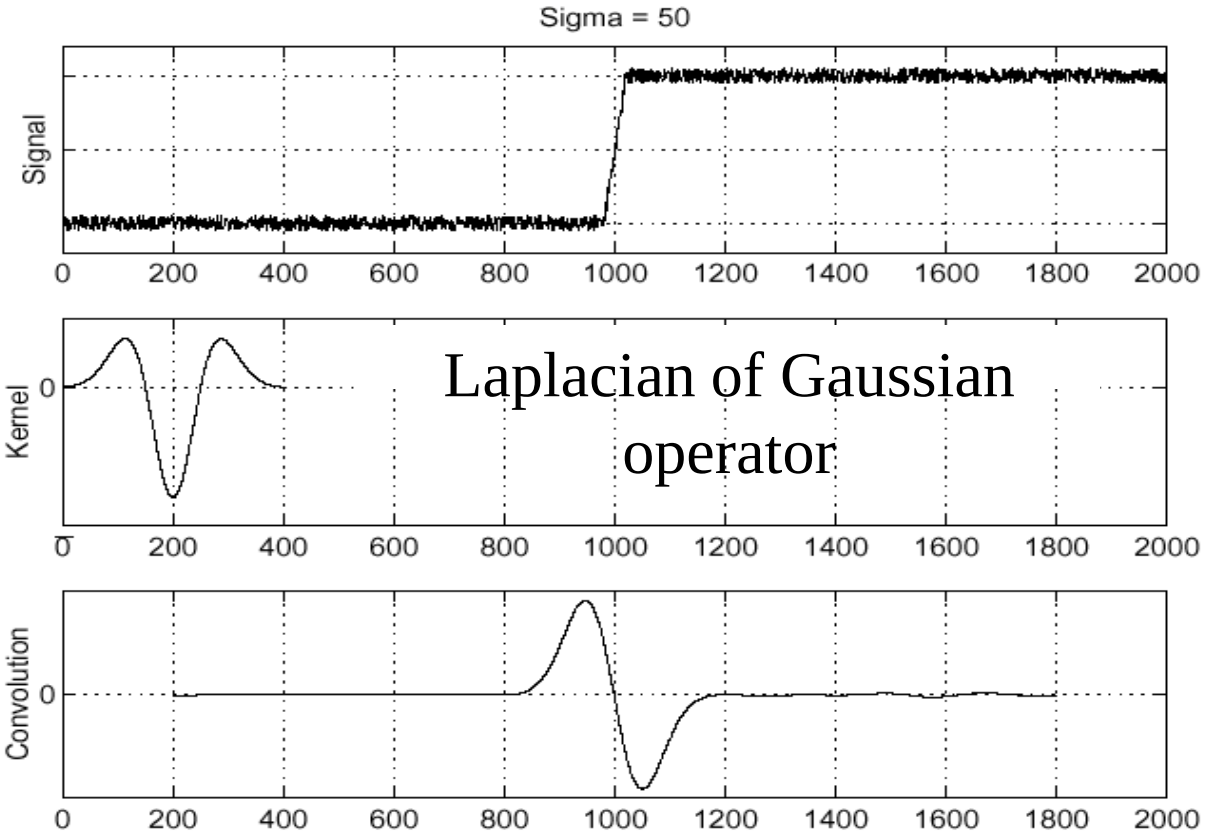
$$\frac{\partial^2}{\partial x^2}(h * f) = \left(\frac{\partial^2}{\partial x^2} h\right) * f$$

Laplacian of Gaussian

f

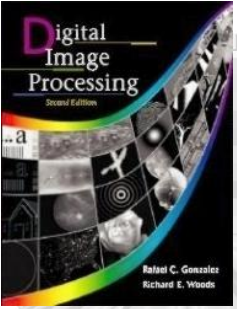
$\frac{\partial^2}{\partial x^2} h$

$(\frac{\partial^2}{\partial x^2} h) \star f$



Where is the edge?

Zero-crossings of bottom graph !



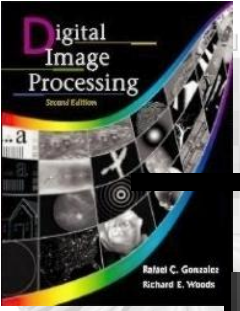
Canny Edge Operator

- Smooth image I with 2D Gaussian: $G * I$
- Find local edge normal directions for each pixel

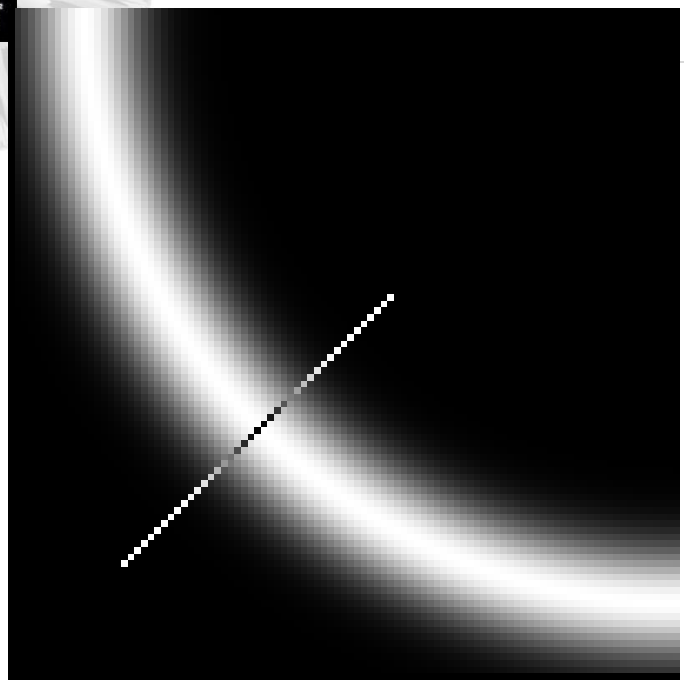
$$\bar{\mathbf{n}} = \frac{\nabla(G * I)}{|\nabla(G * I)|}$$

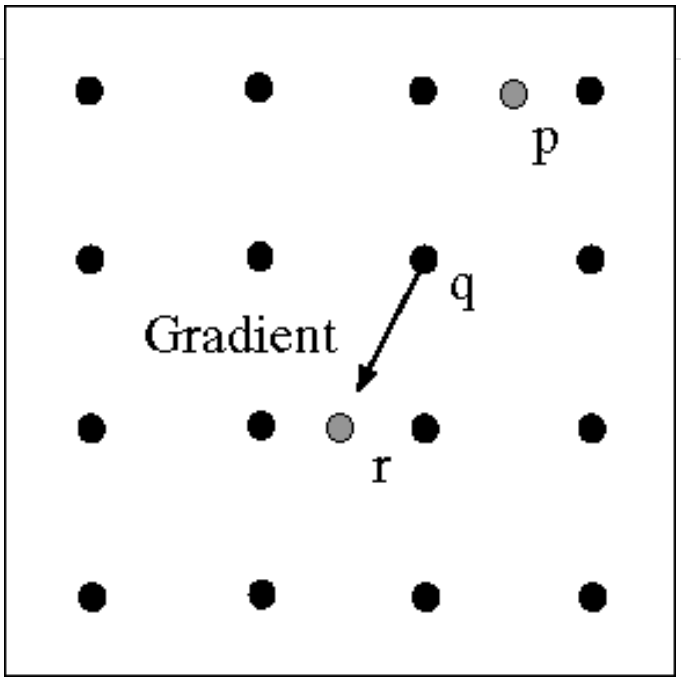
- Compute edge magnitudes $|\nabla(G * I)|$
- Locate edges by finding zero-crossings along the edge normal directions (**non-maximum suppression**)

$$\frac{\partial^2(G * I)}{\partial \bar{\mathbf{n}}^2} = 0$$

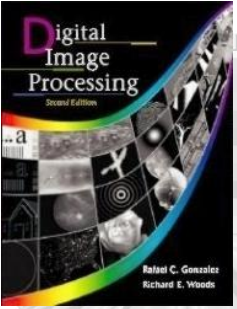


Non-maximum Suppression

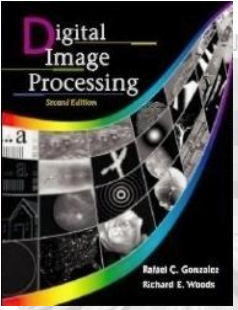




- Check if pixel is local maximum along gradient direction
  - requires checking interpolated pixels p and r

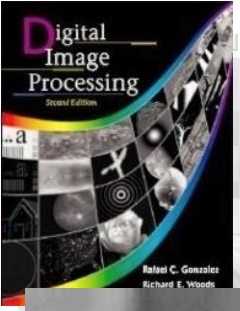


The Canny Edge Detector



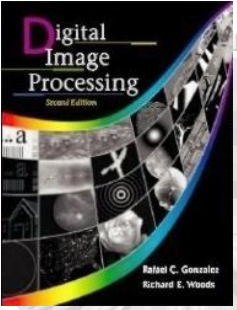
The canny edge detector

magnitude of the gradient



The Canny Edge Detector

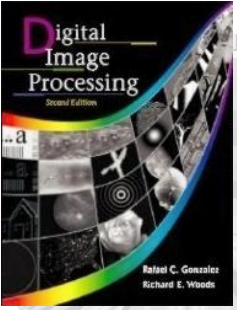
After non-maximum suppression



Edge Linking and Boundary Detection
Global Processing via the Hough Transform

Canny Edge Detection Algorithm

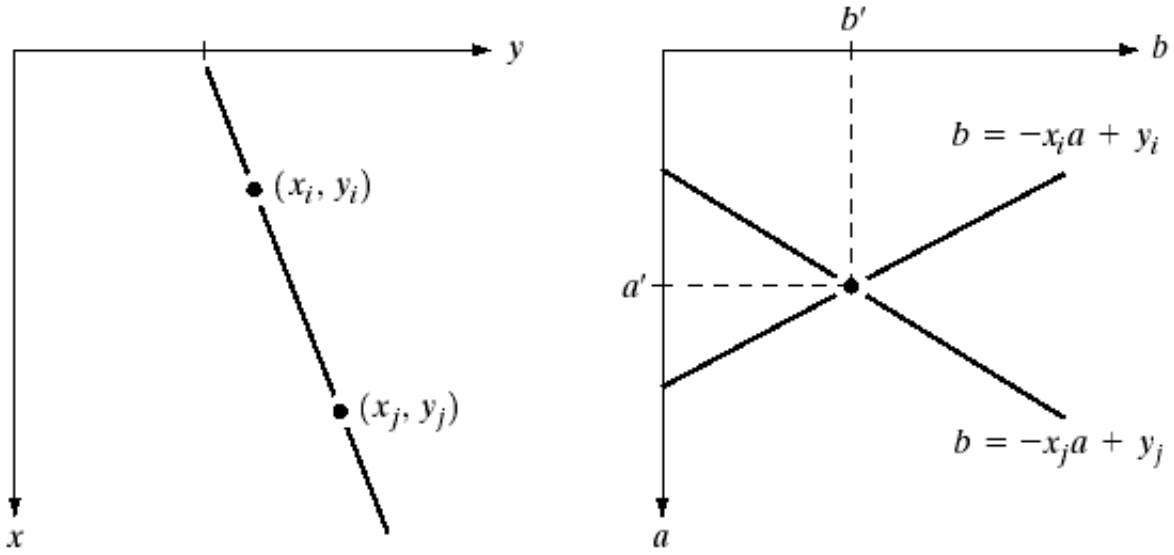
- An edge detection algorithm detects edges in an image by determining where the brightness/intensity of an image changes drastically.
- For the Hough Transform algorithm, it is crucial to perform edge detection first to produce an edge image which will then be used as input into the algorithm.



Edge Linking and Boundary Detection
Global Processing via the Hough Transform

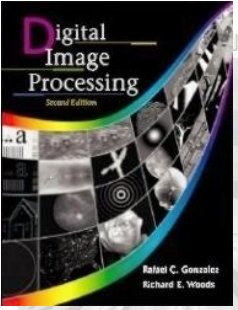
- Hough transform: a way of finding edge points in an image that lie along a straight line.
- Example: *xy*-plane v.s. *ab*-plane (parameter space)

$$y_i = ax_i + b$$



a b

FIGURE 10.17
(a) *xy*-plane. (b) Parameter space.



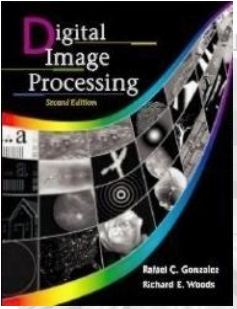
Problem: Given set of points, use Hough transform to join these points. A(1,4), B(2,3) ,C(3,1) ,D(4,1) ,E(5,0)

Solution:

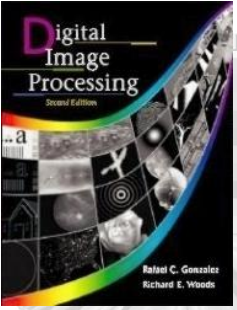
Lets think about equation of line that is y=ax+b.

Now, if we rewrite the same line equation by keeping b in LHS, then we get b=-ax+y. So if we write the same equation for point A(1,4), then consider x=1 and y=4 so that we will get b=-a+4. The following table shows all the equations for a given point

Point	X and y values	Substituting the value in b=-ax+y
A(1,4)	x=1 ; y=4	b= -a+4
B(2,3)	x=2 ; y=3	b= -2a+3
C(3,1)	x=3 ; y=1	b= -3a+1
D(4,1)	x=4 ; y=1	b= -4a+1
E(5,0)	x=5 ; y=0	b= -5a+0

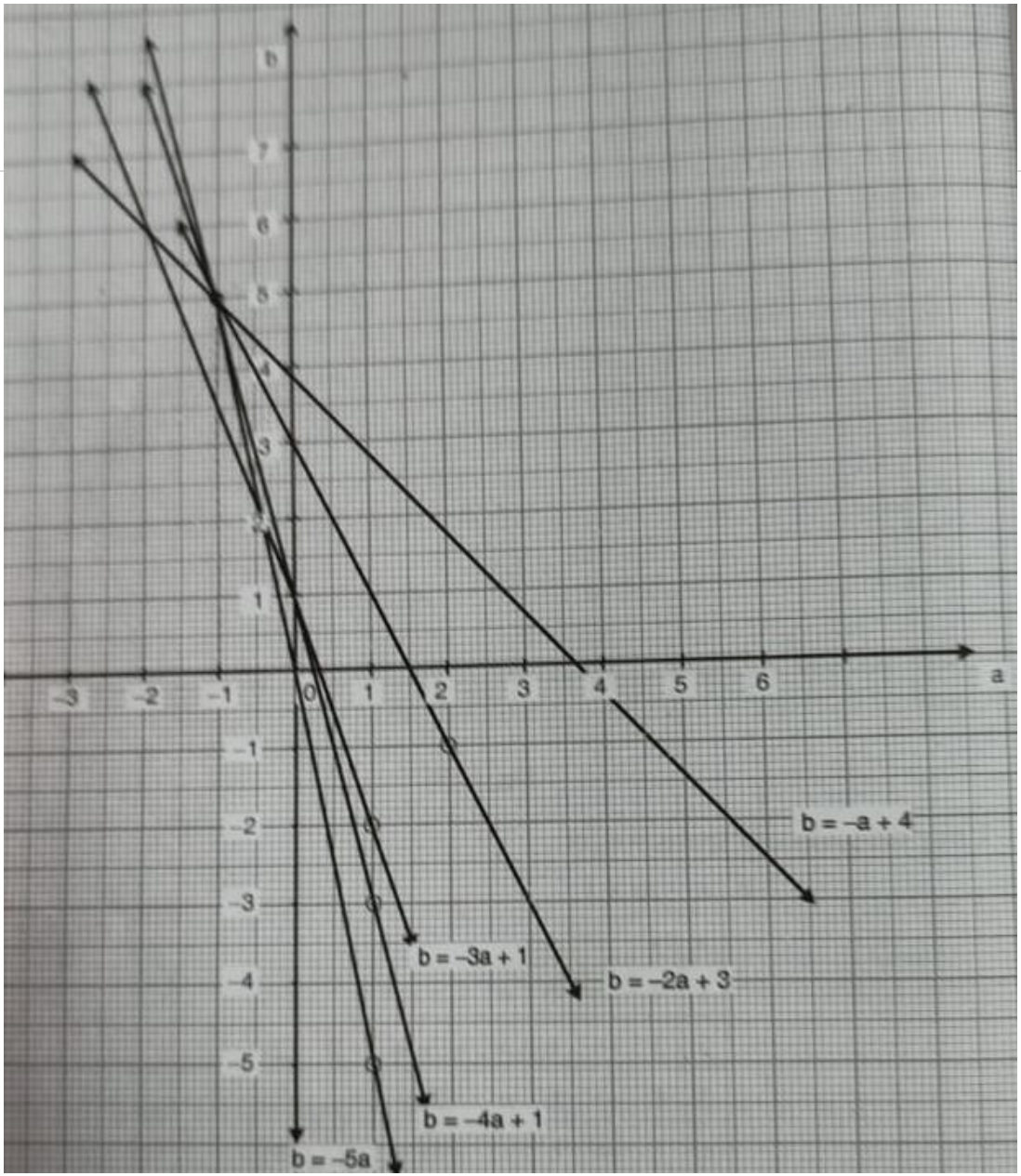


Point	equations	Now a=0	New point (a,b)	Now a=1	New point (a,b)
A(1,4)	b= -a+4	b= -(0)+4 =4	(0,4)	b= -(1)+4 =3	(1,3)
B(2,3)	b= -2a+3	b= -2(0)+3=3	(0,3)	b= -2(1)+3=1	(1,1)
C(3,1)	b= -3a+1	b= -3(0)+1=1	(0,1)	b= -3(1)+1=-2	(1,-2)
D(4,1)	b= -4a+1	b= -4(0)+1=1	(0,1)	b= -4(1)+1=3	(1,-3)
E(5,0)	b= -5a+0	b= -5(0)+0=0	(0,0)	b= -5(1)+0=-5	(1,-5)



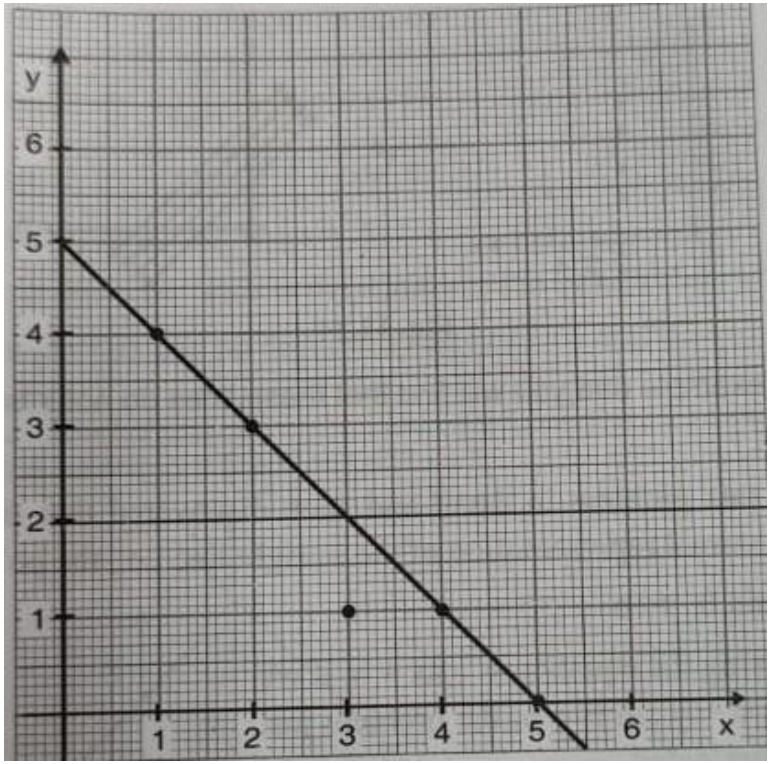
Let us plot the new point on the graph as given below

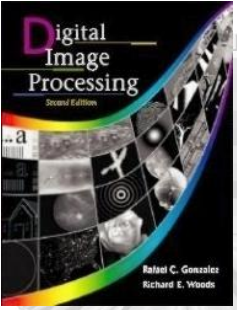
Point	New point (a,b)	New point (a,b)
A(1,4)	(0,4)	(1,3)
B(2,3)	(0,3)	(1,1)
C(3,1)	(0,1)	(1,-2)
D(4,1)	(0,1)	(1,-3)
E(5,0)	(0,0)	(1,-5)



We can see that almost all line cross each other at a point (-1,5). So here now a=-1 and b =5.

Now let's put these values in the y=ax+b equation so we get y=-1x+5 so y=-x+5 is the line equation that will link all the edges.



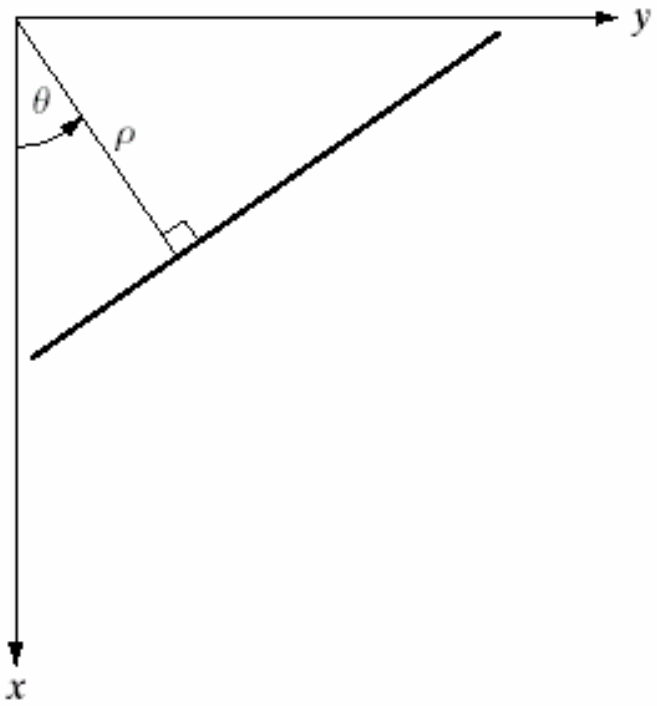


Edge Linking and Boundary Detection
Global Processing via the Hough Transform

- There is one flaw with representing lines in the form of $y = ax + b$ and the Hough Space with the slope and intercept.
- In this form, the algorithm won't be able to detect vertical lines because the slope a is undefined/infinity for vertical lines (Leavers, 1992).
- Programmatically, this means that a computer would need an infinite amount of memory to represent all possible values of a.
- To avoid this issue, a straight line is instead represented by a line called the normal line that passes through the origin and perpendicular to that straight line. The form of the normal line is $\rho = x\cos(\theta) + y\sin(\theta)$ where ρ is the length of the normal line and θ is the angle between the normal line and the x axis.

Edge Linking and Boundary Detection
Global Processing via the Hough Transform

- The Hough transform consists of finding all pairs of values of θ and ρ which satisfy the equations that pass through (x,y).
- These are accumulated in what is basically a 2-dimensional histogram.
- When plotted these pairs of θ and ρ will look like a sine wave. The process is repeated for all appropriate (x,y) locations.

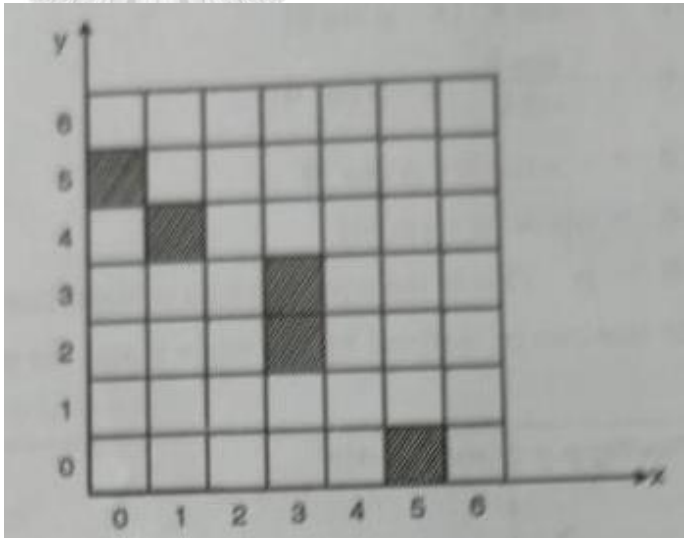


$$x\cos\theta + y\sin\theta = \rho$$

https://sbme-tutorials.github.io/2021/cv/notes/4_week4.html

For the given image use Hough Transform to form the edge.

Solution:



The pixel coordinates of interest are (0, 5), (1, 4), (3, 2), (3, 3), (5, 0)

For each point, we shall use the equation

$$\rho = x \cos\theta + y \sin\theta$$

Using the first value of x and y i.e. (0, 5) we get

$$\rho = 0 \cos\theta + 5 \sin\theta$$

We now vary θ from –90 to +90 and get the values of ρ.

Similarly, equations for all the 5 points are

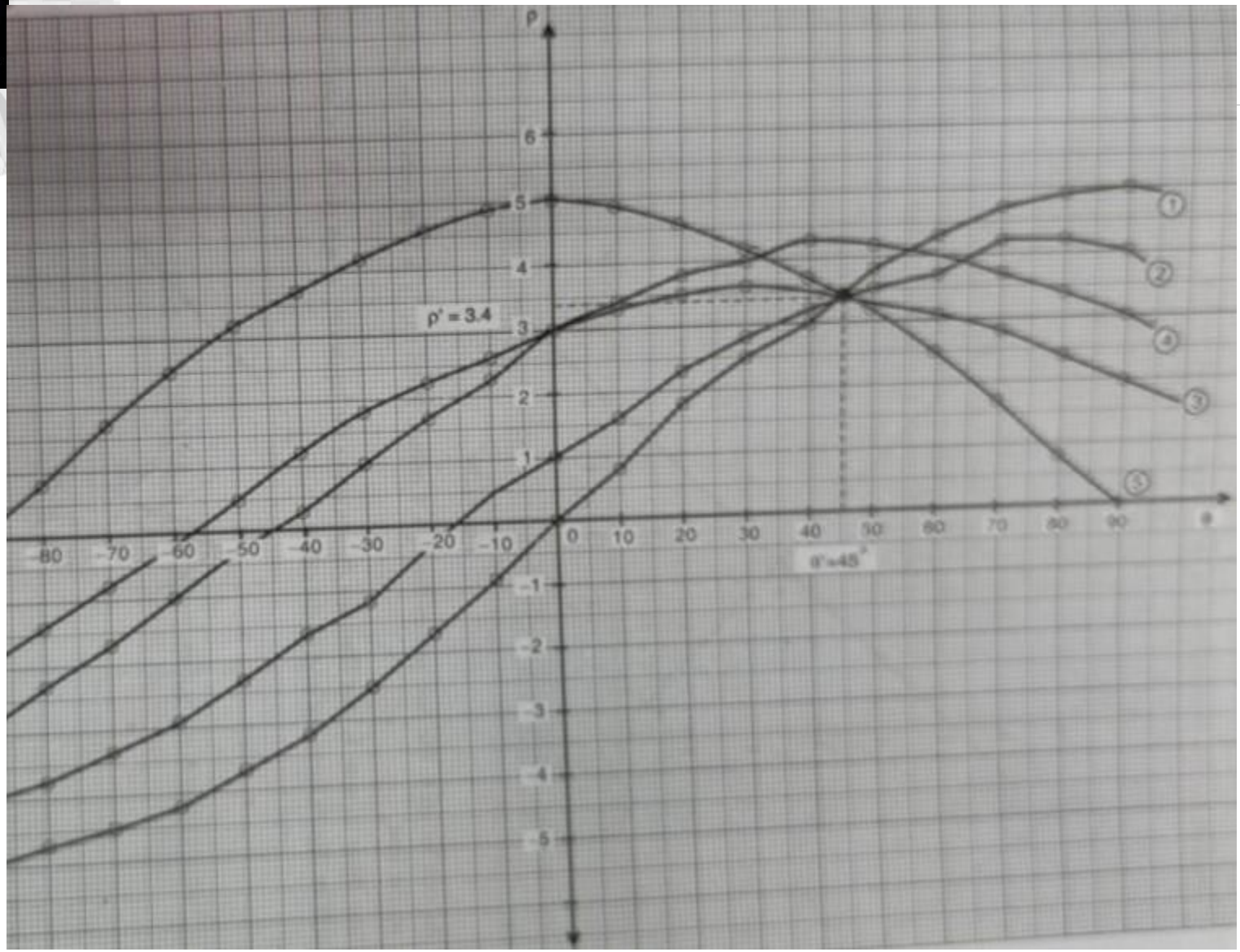
$$(0, 5) \rightarrow \rho = 0 \cos\theta + 5 \sin\theta$$

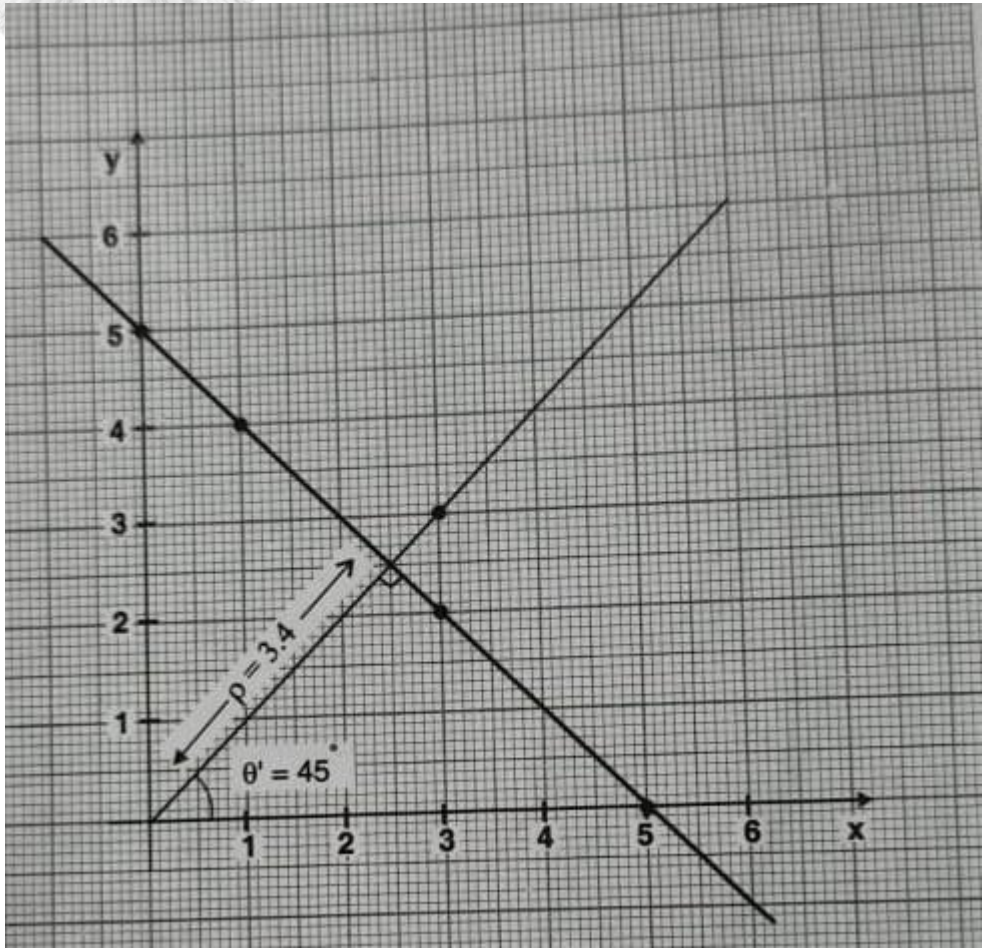
$$(1, 4) \rightarrow \rho = 1 \cos\theta + 4 \sin\theta$$

$$(3, 2) \rightarrow \rho = 3 \cos\theta + 2 \sin\theta$$

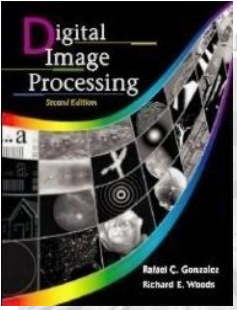
$$(3.3) \rightarrow \rho = 3 \cos\theta + 3 \sin\theta$$

$$(5, 0) \rightarrow \rho = 5 \cos\theta + 0 \sin\theta$$





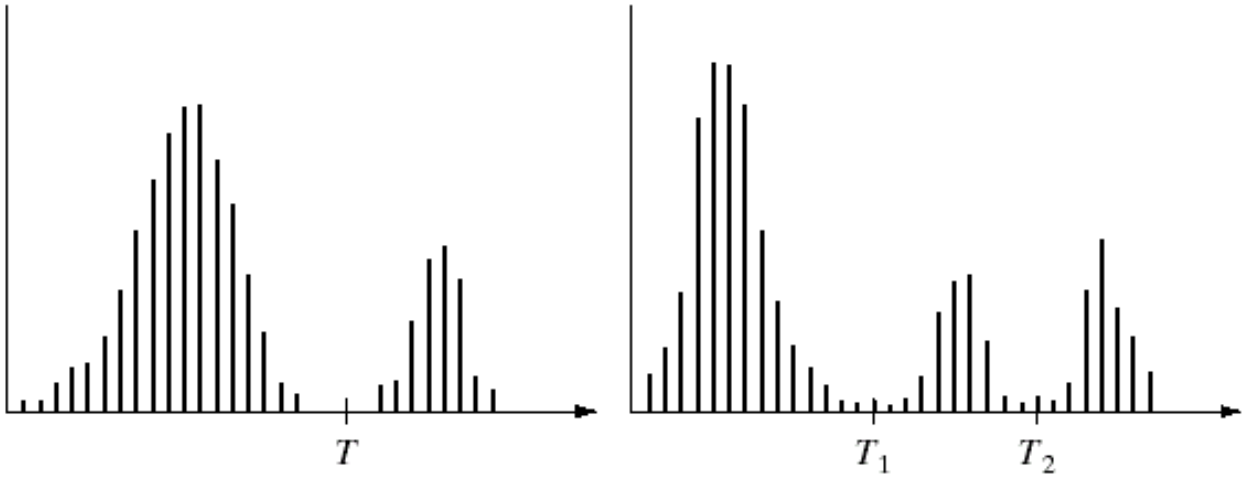
As can be seen, the line passes through 4 points, point(3,3) is kept outside as it does not lie on the line



Thresholding

- Assumption: the range of intensity levels covered by objects of interest is different from the background.

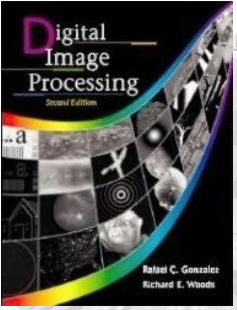
$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{if } f(x,y) \le T \end{cases}$$



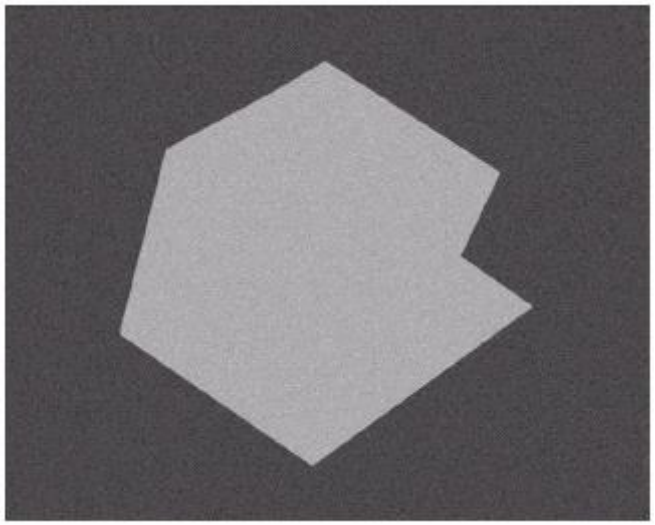
Single threshold Multiple threshold

a b

FIGURE 10.26 (a) Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds.



Thresholding
The Role of Illumination



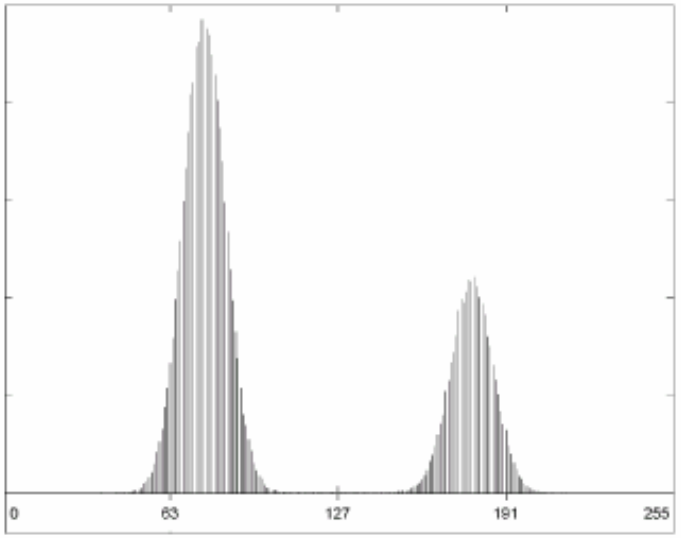
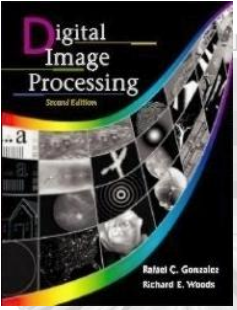
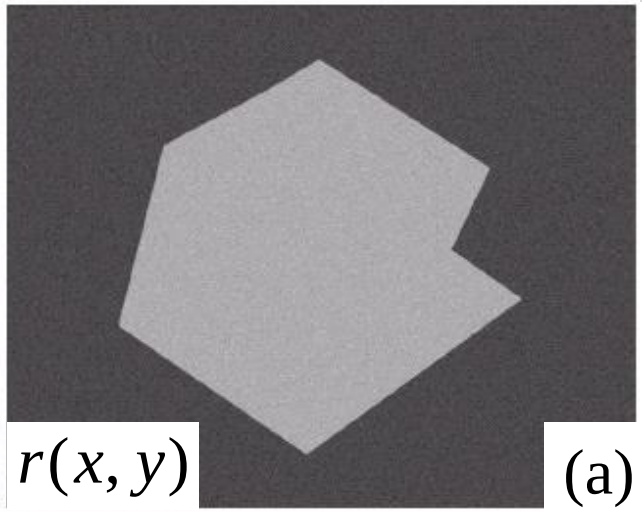


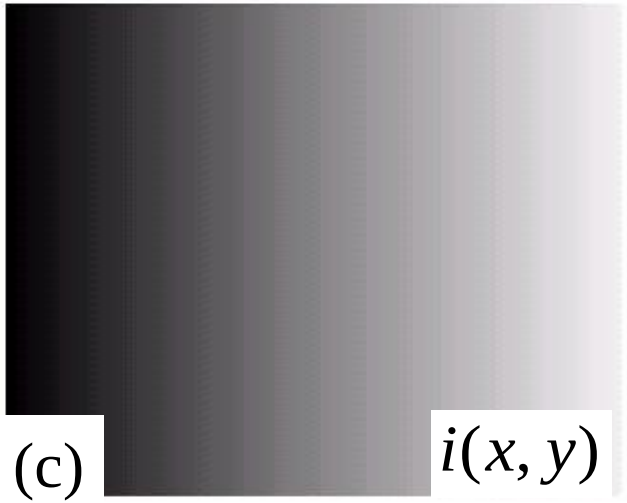
FIGURE 10.27
(a) Computer generated reflectance function.
(b) Histogram of reflectance function.



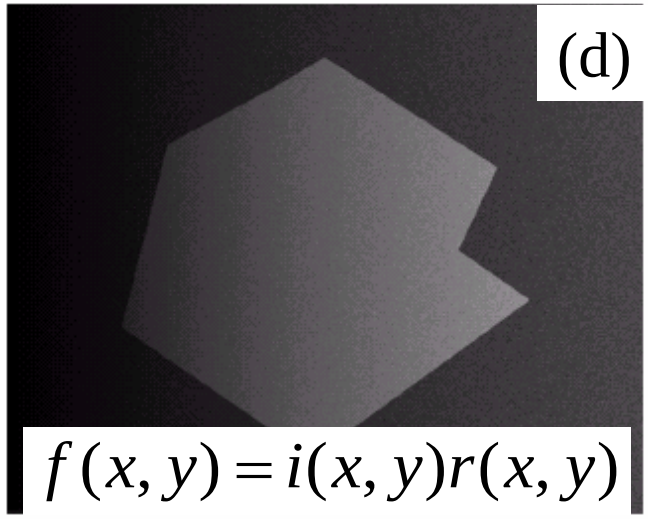
Thresholding
The Role of Illumination



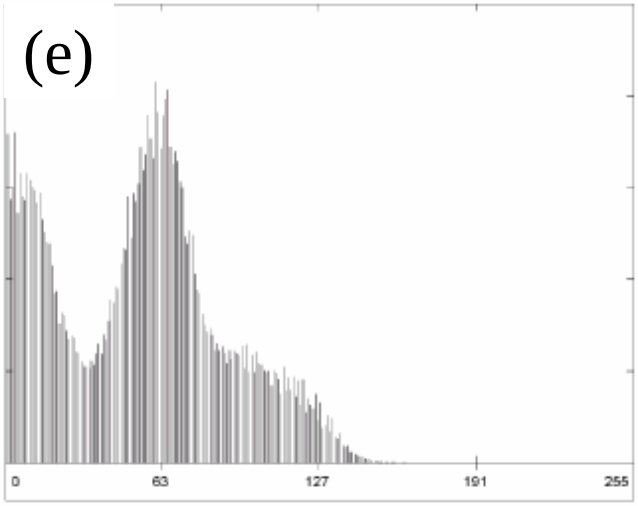
(a)



(c)



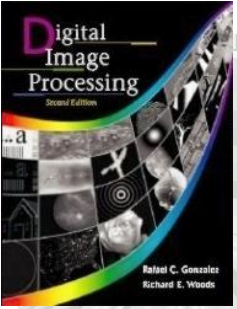
(d)



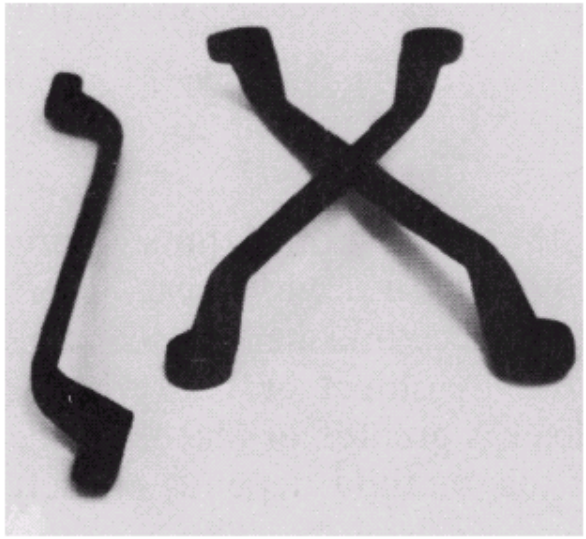
(e)

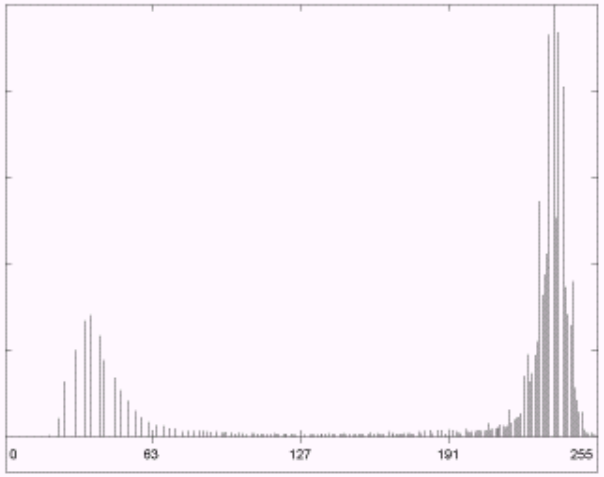
a
b c
d e

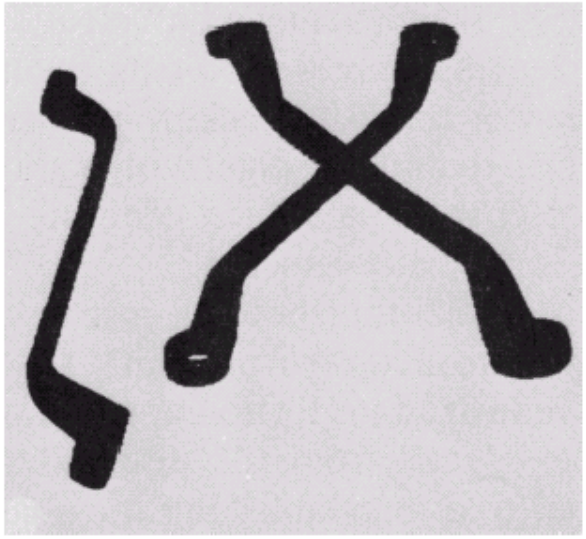
FIGURE 10.27
(a) Computer generated reflectance function.
(b) Histogram of reflectance function.
(c) Computer generated illumination function.
(d) Product of (a) and (c).
(e) Histogram of product image.



Thresholding
Basic Global Thresholding

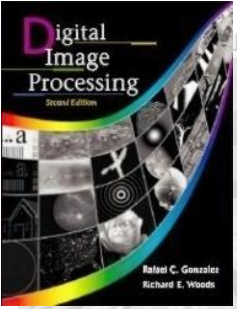




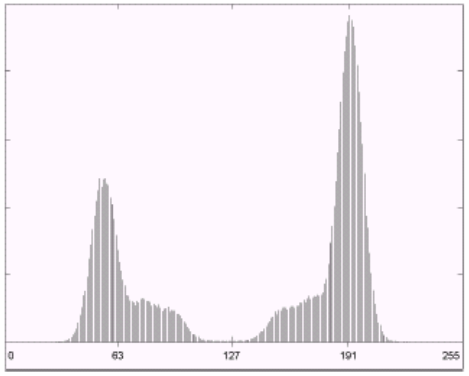


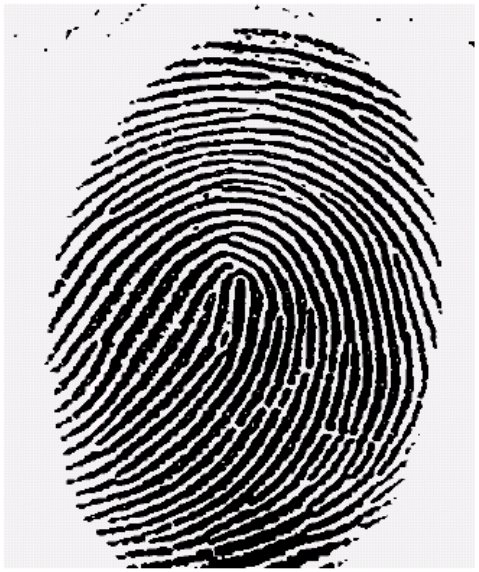
a
b c

FIGURE 10.28
(a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.



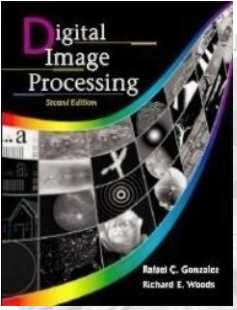
Thresholding
Basic Global Thresholding





a b
c

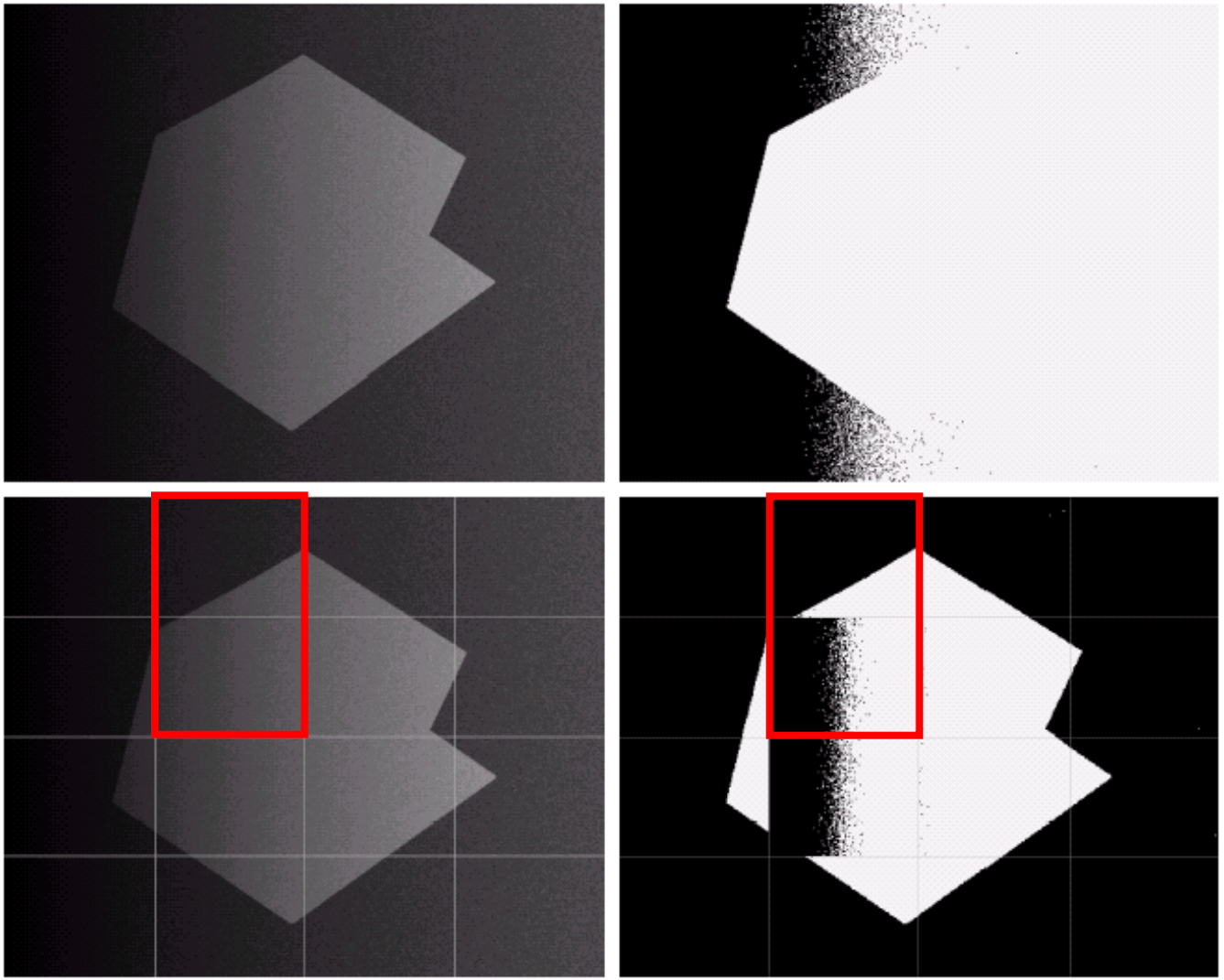
FIGURE 10.29
(a) Original image. (b) Image histogram. (c) Result of segmentation with the threshold estimated by iteration. (Original courtesy of the National Institute of Standards and Technology.)

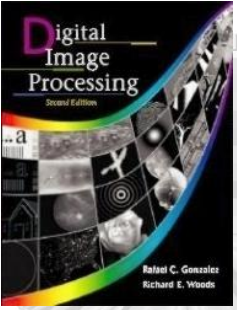


Thresholding
Basic Adaptive Thresholding

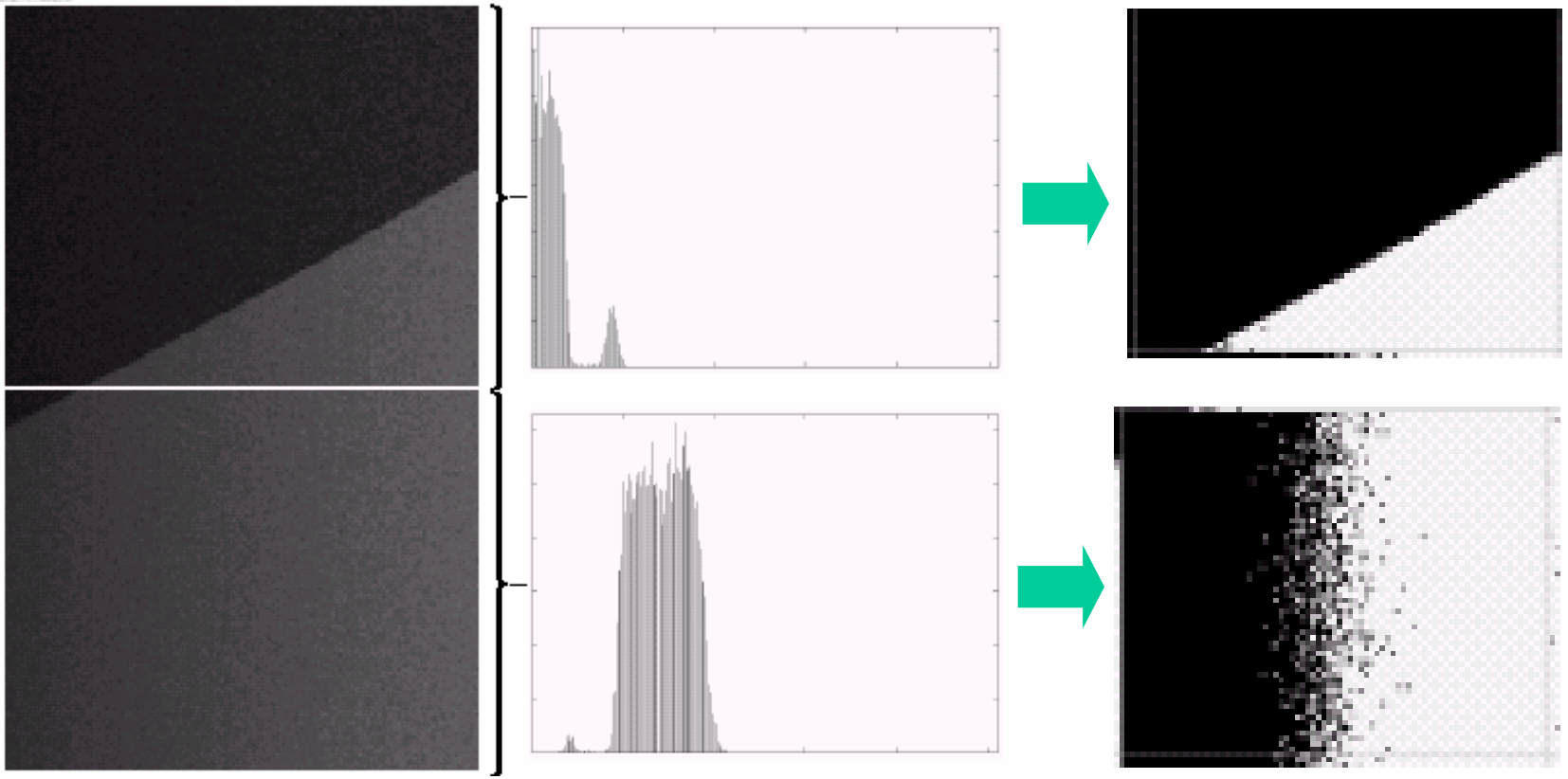
a b
c d

FIGURE 10.30
(a) Original image. (b) Result of global thresholding. (c) Image subdivided into individual subimages. (d) Result of adaptive thresholding.

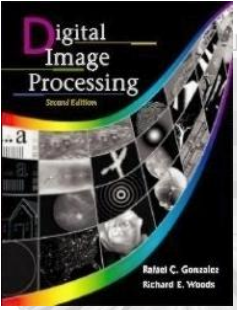




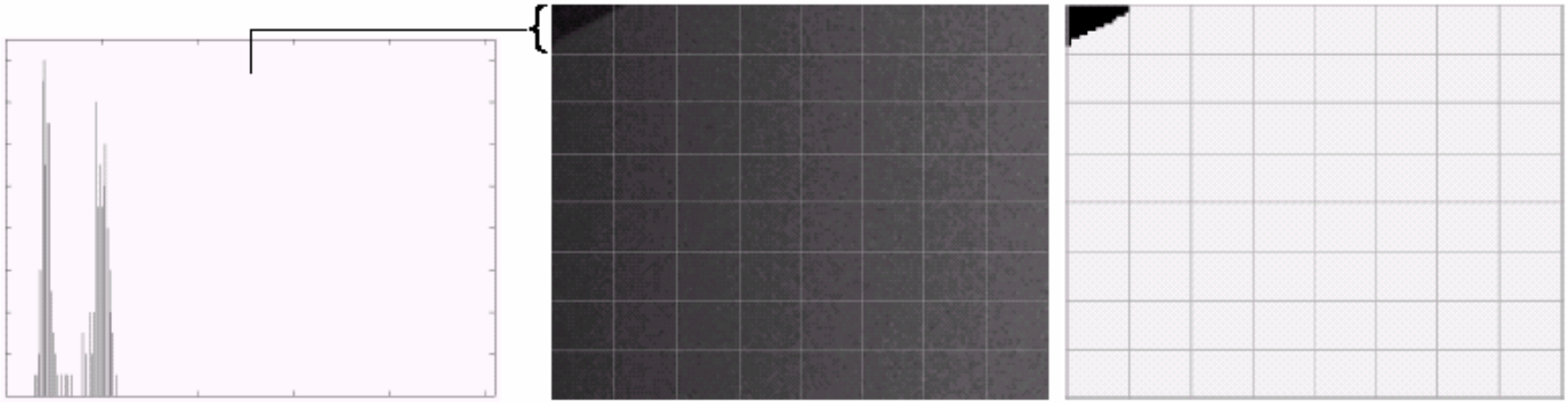
Thresholding
Basic Adaptive Thresholding



How to solve this problem?



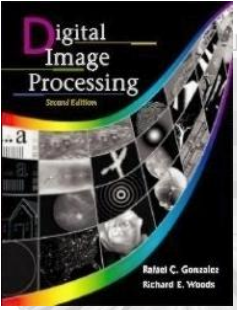
Thresholding
Basic Adaptive Thresholding



Answer: subdivision

a b c
e d f

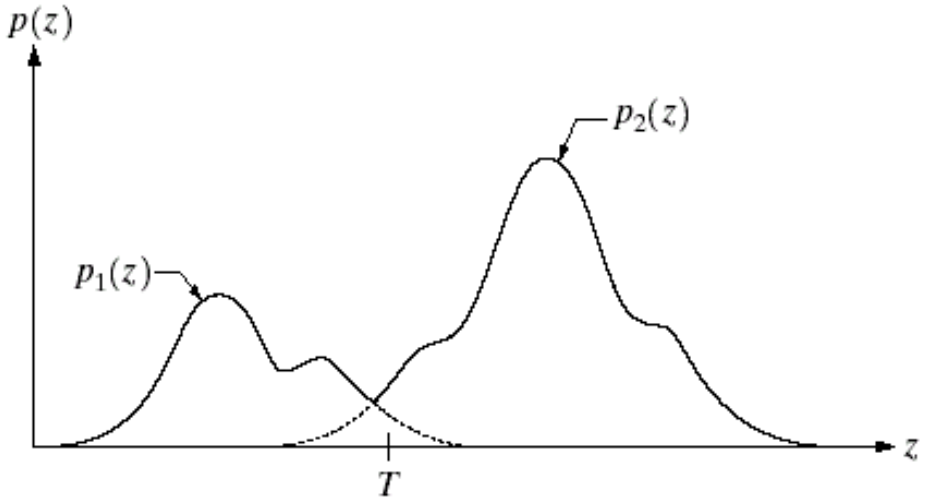
FIGURE 10.31 (a) Properly and improperly segmented subimages from Fig. 10.30. (b)–(c) Corresponding histograms. (d) Further subdivision of the improperly segmented subimage. (e) Histogram of small subimage at top, left. (f) Result of adaptively segmenting (d).

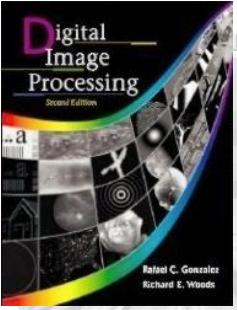


Thresholding
Optimal Global and Adaptive Thresholding

- This method treats pixel values as probability density functions.
- The goal of this method is to minimize the probability of misclassifying pixels as either object or background.
- There are two kinds of error:
  - mislabeling an object pixel as background, and
  - mislabeling a background pixel as object.

FIGURE 10.32 Gray-level probability density functions of two regions in an image.

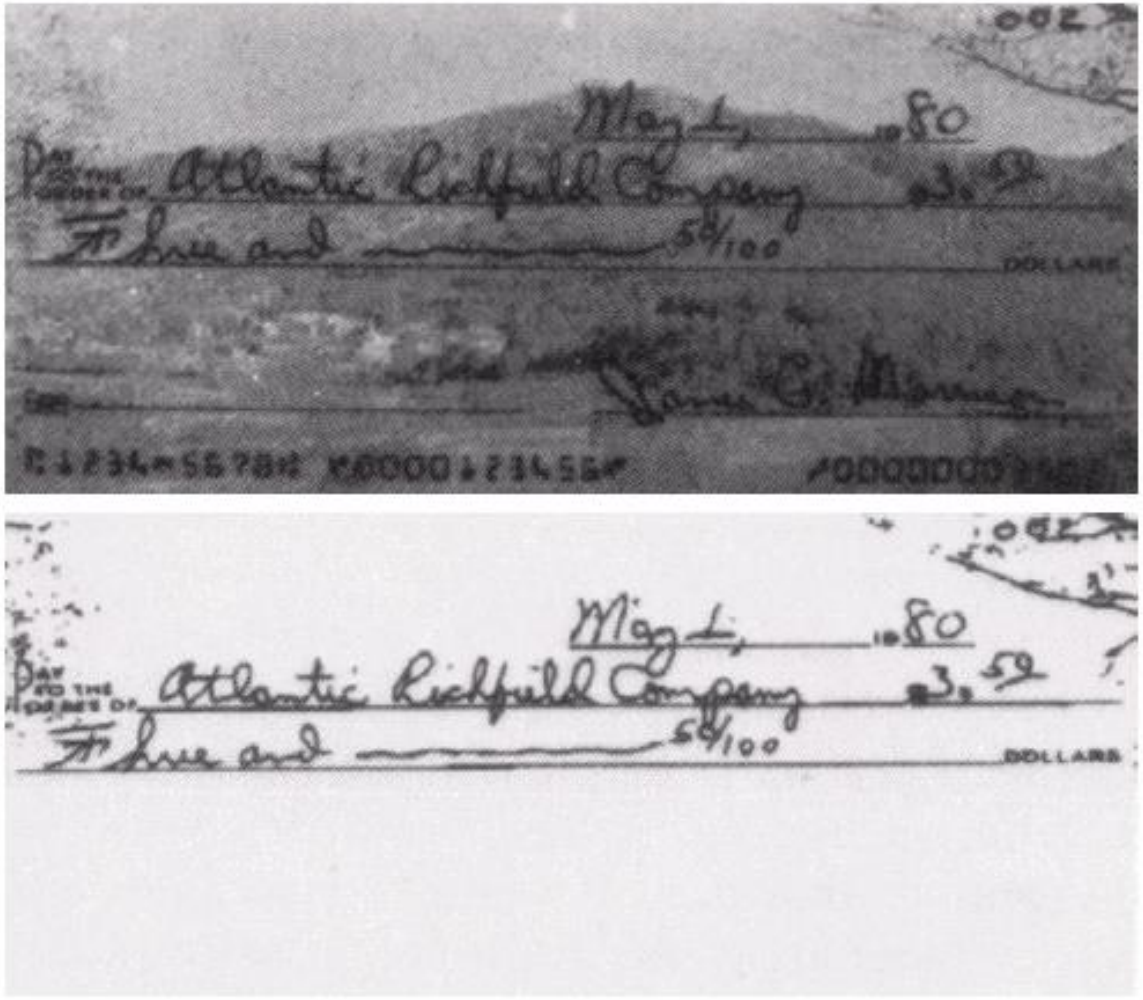


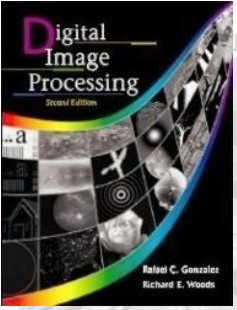


Thresholding
Use of Boundary Characteristics

a
b

FIGURE 10.37
(a) Original image. (b) Image segmented by local thresholding. (Courtesy of IBM Corporation.)



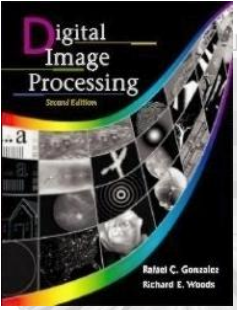


Thresholding
Thresholds Based on Several Variables

Color image

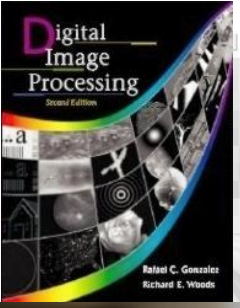
a b c

FIGURE 10.39 (a) Original color image shown as a monochrome picture. (b) Segmentation of pixels with colors close to facial tones. (c) Segmentation of red components.



Region-Based Segmentation

- 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.
  - Each region must be uniform.
  - Connectivity of the pixels within the region is very important.
- There are two main approaches to region-based segmentation: region growing and region splitting.



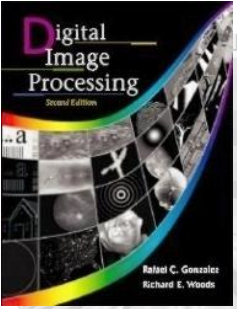
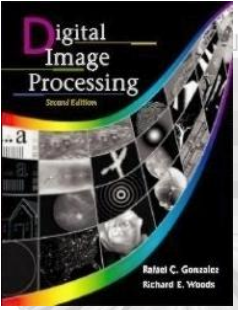


Figure 1: Example of Image Segmentation



Region-Based Segmentation
Basic Formulation

- Let R represent the entire image region.
- Segmentation is a process that partitions R into subregions, $R_1, R_2, \ldots, R_n$, such that

(a) $\bigcup_{i=1}^{n} R_i = R$

(b) R_i is a connected region, $i = 1,2,\ldots,n$

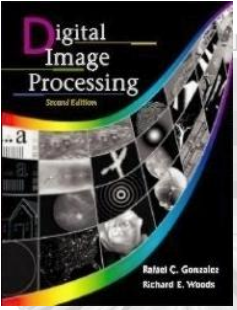
(c) $R_i \cap R_j = \phi$ for all i and $j, i \neq j$

(d) $P(R_i) = \text{TRUE}$ for $i = 1,2,\ldots,n$

(e) $P(R_i \cup R_j) = \text{FALSE}$ for any adjacent regions R_i and R_j

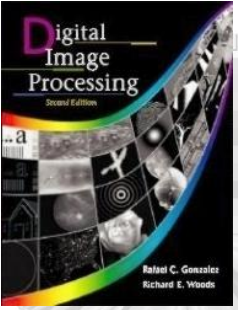
where $P(R_k)$: a logical predicate defined over the points in set R_k

For example: $P(R_k)$=TRUE if all pixels in R_k have the same gray level.



Region Growing

1. Region growing is a procedure that groups pixels or subregions into larger regions.
2. The simplest of these approaches is ***pixel aggregation***, which starts with a set of "seed" points and from these grows regions by appending to each seed points those **neighboring pixels** that have **similar properties** (such as gray level, texture, color, shape).
3. Region growing based techniques are better than the edge-based techniques in noisy images where edges are difficult to detect.



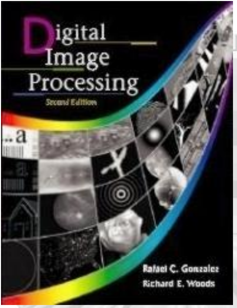
Region Growing

Suppose that we have the image given below.

(a) Use the region growing idea to segment the object. The seed for the object is the center of the image. Region is grown in horizontal and vertical directions

Threshold=10

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	**60**	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10



10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

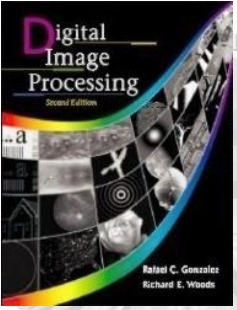
10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	56	60
10	59	10	60	70	10	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10



Region-Based Segmentation
Region Growing

- Fig. 10.41 shows the histogram of Fig. 10.40 (a). It is difficult to segment the defects by thresholding methods. (Applying region growing methods are better in this case.)

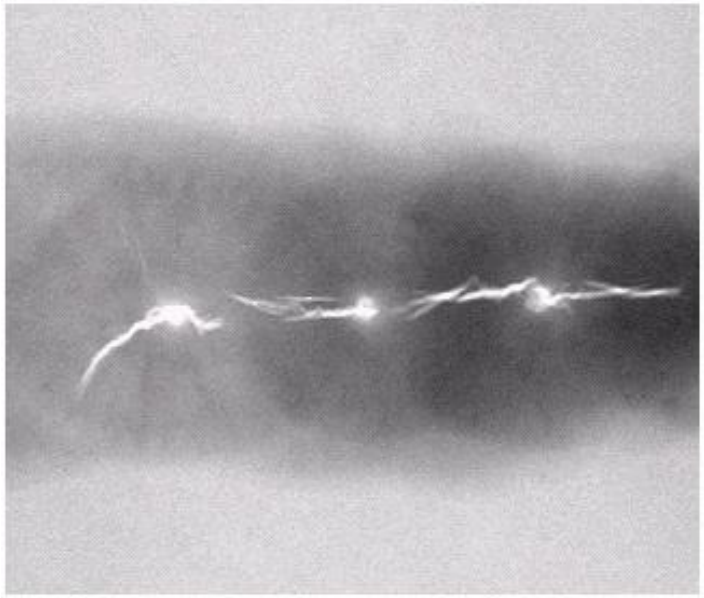


Figure 10.40(a)

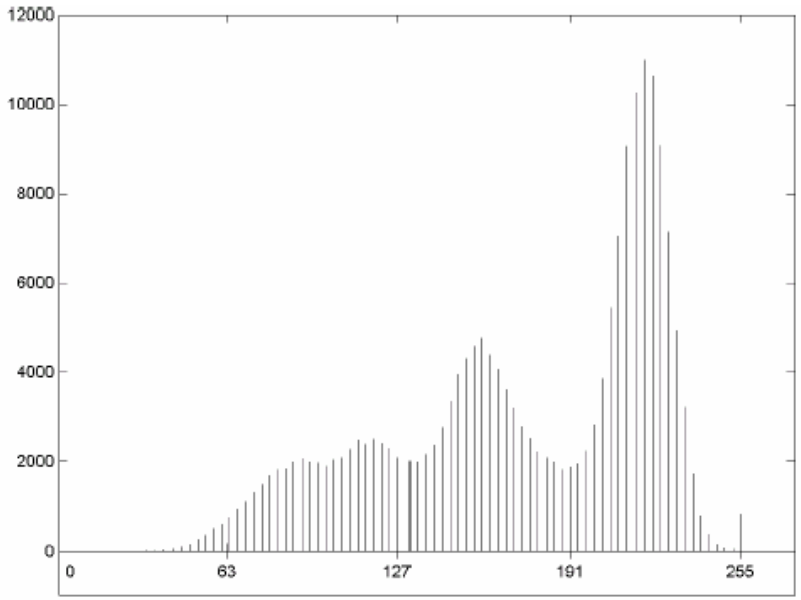
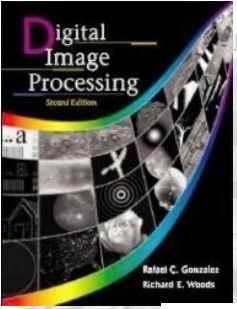
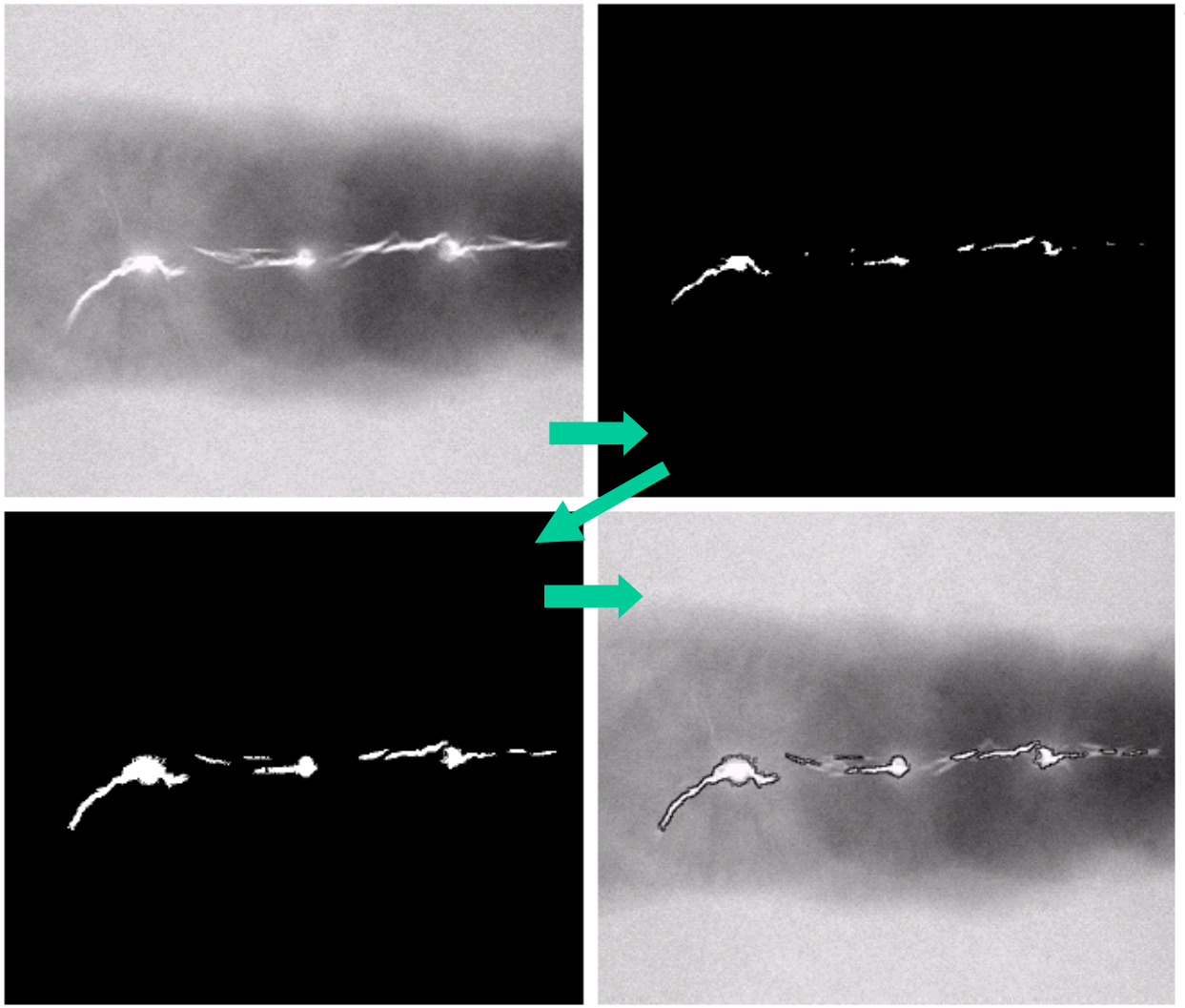


Figure 10.41

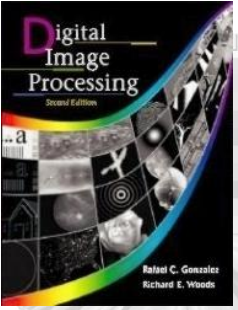


Region-Based Segmentation
Region Growing



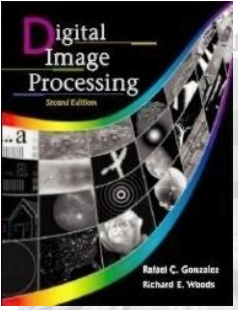
a b
c d

FIGURE 10.40
(a) Image showing defective welds. (b) Seed points. (c) Result of region growing. (d) Boundaries of segmented defective welds (in black). (Original image courtesy of X-TEK Systems, Ltd.).



Region-Based Segmentation
Region Splitting and Merging

- Region splitting is the opposite of region growing.
  - First there is a large region (possible the entire image).
  - Then a predicate (measurement) is used to determine if the region is uniform.
  - If not, then the method requires that the region be split into two regions.
  - Then each of these two regions is independently tested by the predicate (measurement).
  - This procedure continues until all resulting regions are uniform.

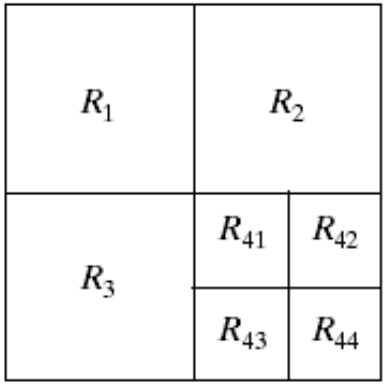


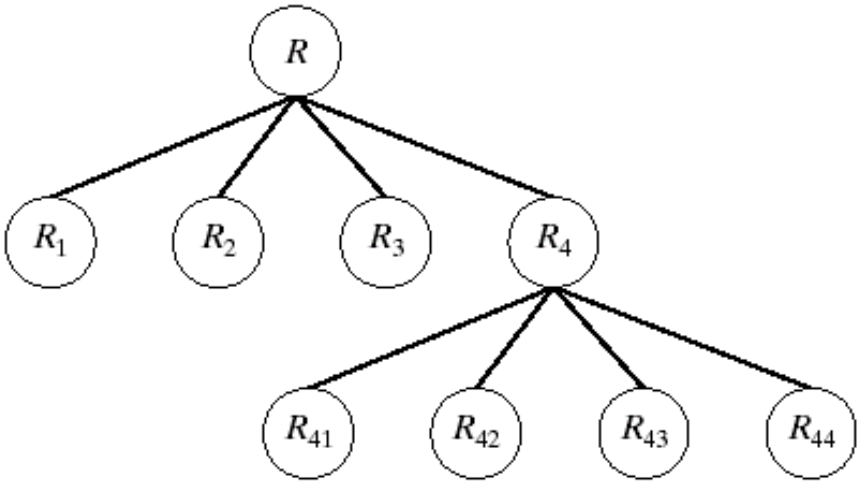
Region-Based Segmentation
Region Splitting

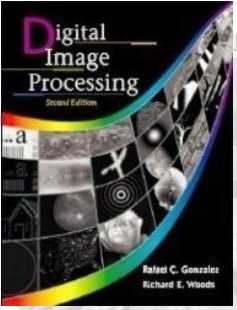
- The main problem with region splitting is determining where to split a region.
- One method to divide a region is to use a quadtree structure.
- Quadtree: a tree in which nodes have exactly four descendants.

a b

FIGURE 10.42
(a) Partitioned image.
(b) Corresponding quadtree.

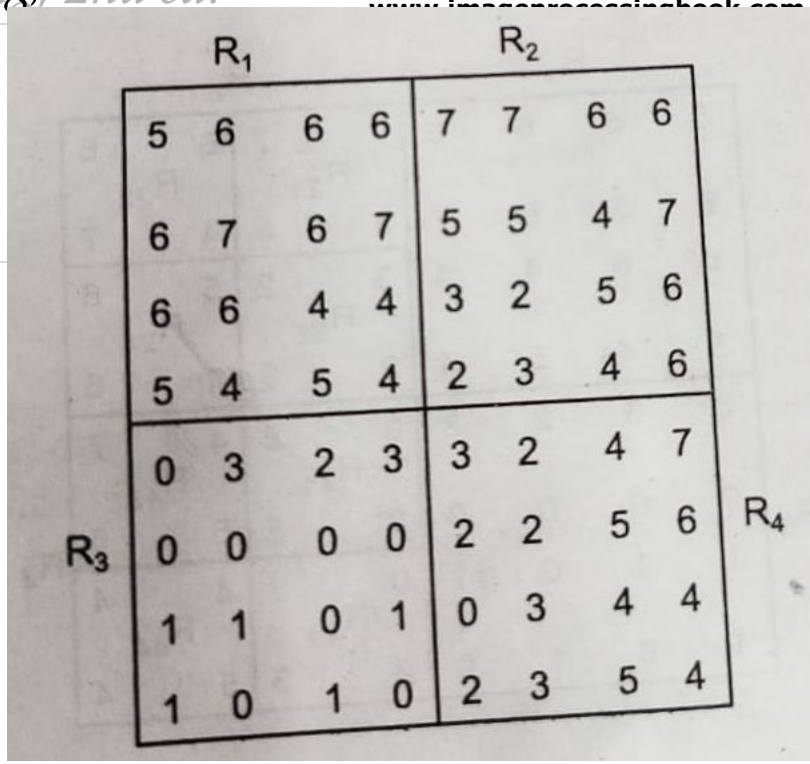


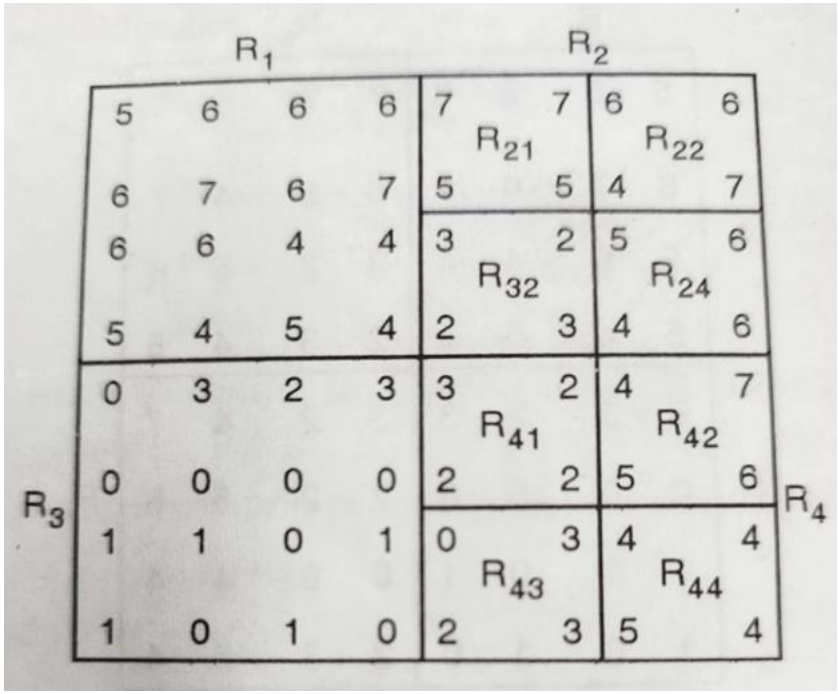


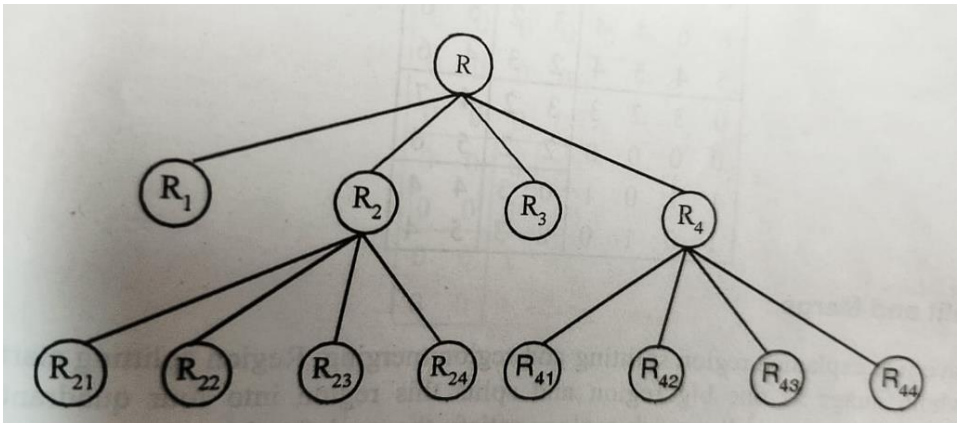


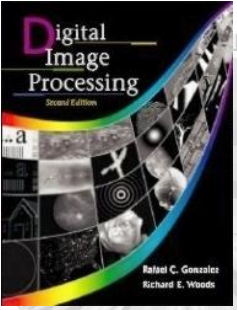
Threshold<=3

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







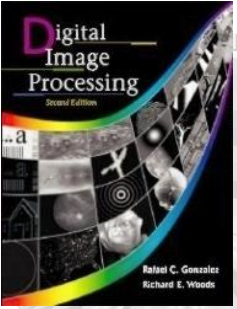


Region merging algorithms

- The region merging method is exactly opposite to the region splitting method.
- In this method, the pixel level and consider each of them as a homogenous region.
- At any level of merging, check if the four adjacent homogenous regions arranged in a 2 x 2 fashion together satisfy the homogeneity property.
- If yes, they are merged to form a bigger region, otherwise the regions are left as they are.
- max - min ≤ 3

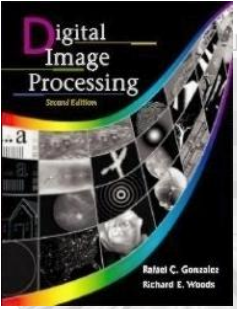
5	6	6	6	7	7	6	6
6	7	6	7	5	5	4	7
6	6	4	4	3	2	5	6
5	4	5	4	2	3	4	6
0	3	2	3	3	2	4	7
0	0	0	0	2	2	5	6
1	1	0	1	0	3	4	4
1	0	1	0	2	3	5	4

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

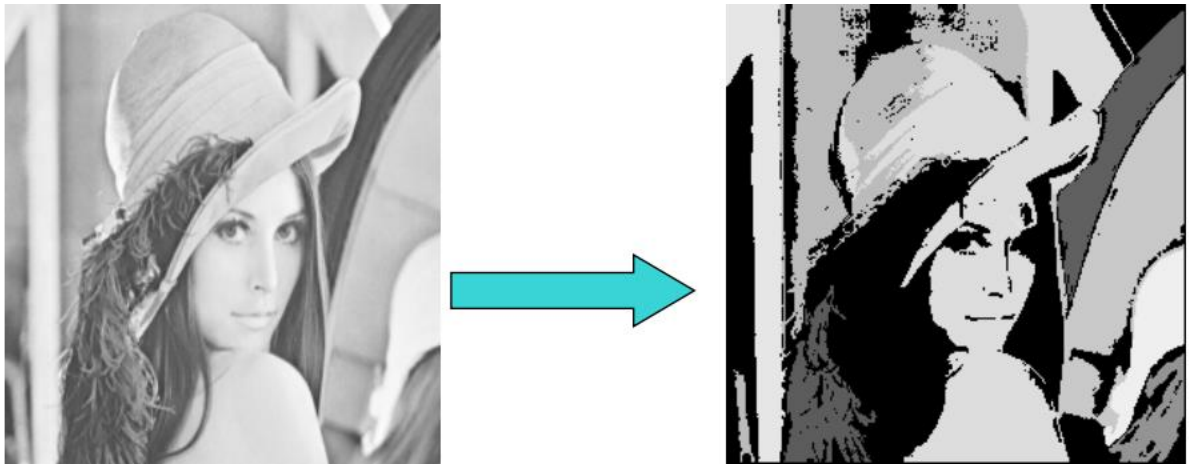


Region-Based Segmentation
Region Splitting and Merging

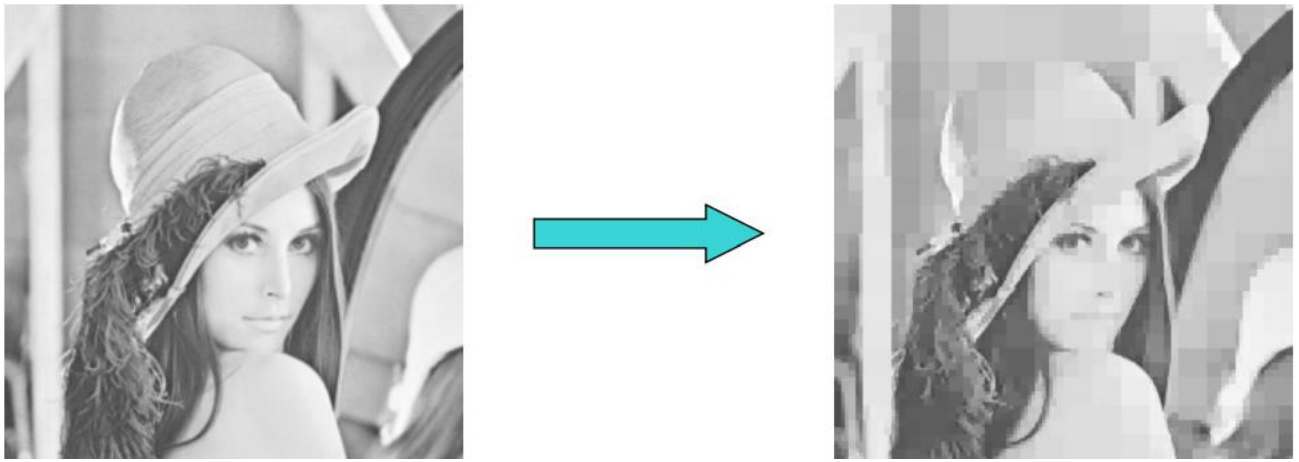
- START: consider entire image as one region
- 1. If region satisfies homogeneity criteria, leave it unmodified.
- 2. If not, split it into four quadrants and recursively apply 1 and 2 to each newly generated region. STOP when all regions in the quadtree satisfy the homogeneity criterion
- 3. If any two adjacent regions Ri , Rj can be merged into a homogeneous region, merge them. STOP when no merging is possible any more.

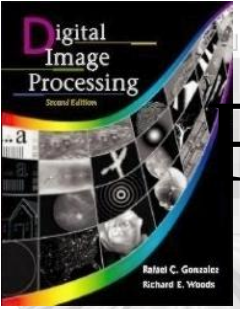


Results – Region grow

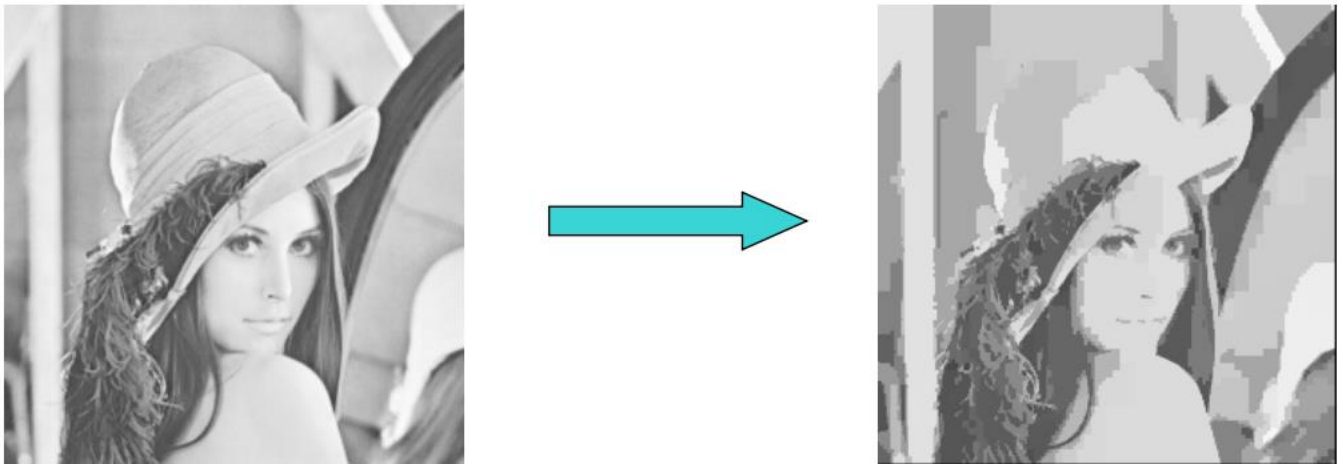


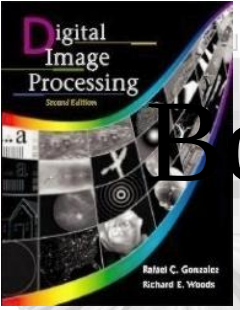
Results – Region Split





Results-Region Split and Merge

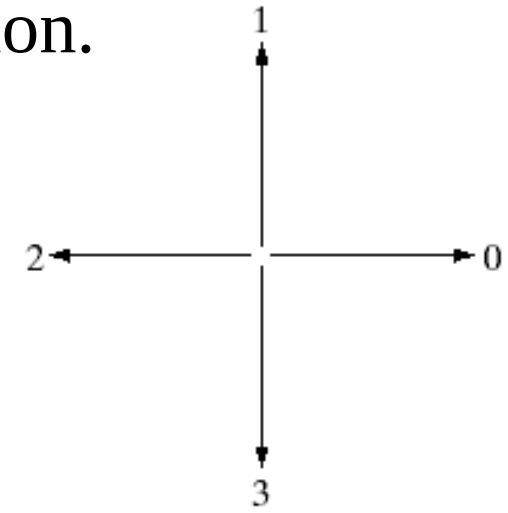




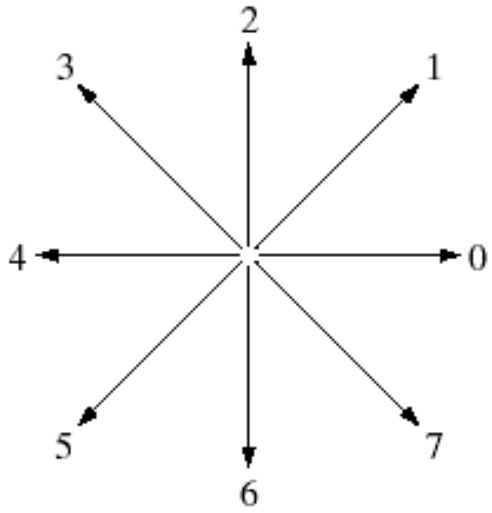
Boundary Descriptors-Chain Code

Boundary is a good representation of an object shape.

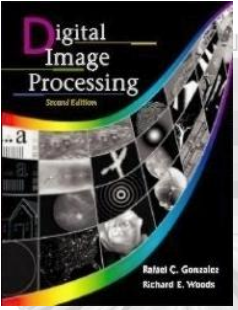
Chain codes: represent an object boundary by a connected sequence of straight line segments of specified length and direction.



4-directional chain code

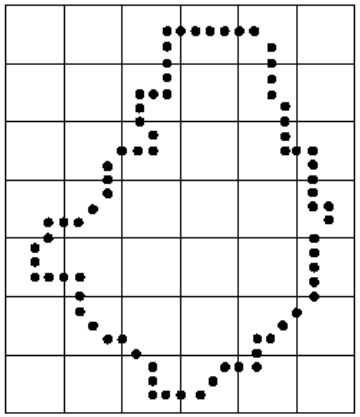


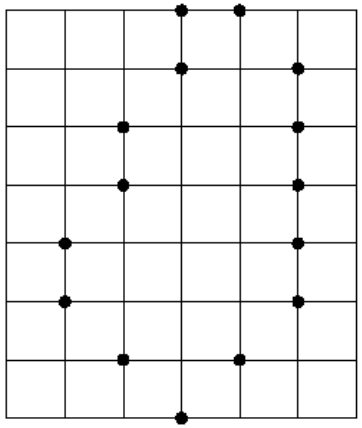
8-directional chain code



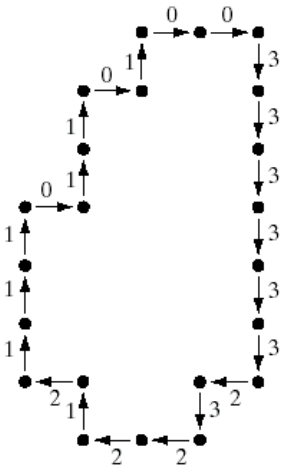
To avoid noise degradation and long chains a resampling of the image grid is commonly used to describe the boundary at a coarser level.

Object boundary (resampling)

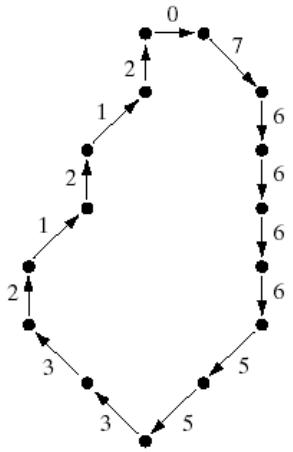




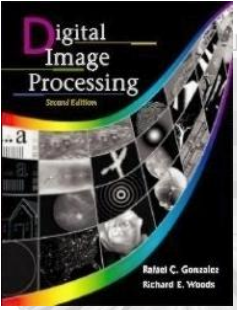
Boundary vertices



4-directional chain code



8-directional chain code



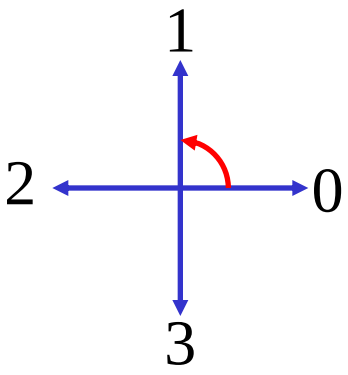
First Difference

Problem of a chain code: a chain code sequence depends on a starting point.

Solution: treat a chain code as a circular sequence and redefine the starting point so that the resulting sequence of numbers forms an integer of minimum magnitude.

The first difference of a chain code: counting the number of direction change (in counterclockwise) between 2 adjacent elements of the code.

Example:



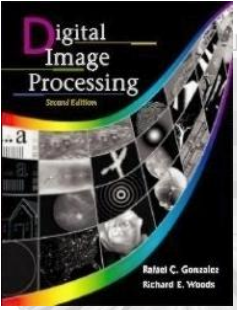
Chain code : The first difference

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

Example:

- a chain code: 10103322
- The first difference = 3133030
- Treating a chain code as a circular sequence, we get the first difference = 33133030

The first difference is rotational invariant.



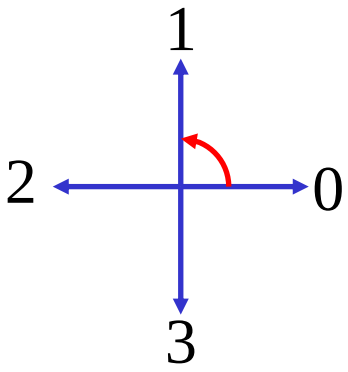
Shape Number

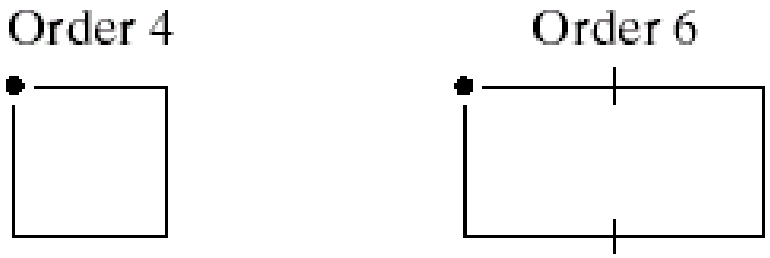
Shape number of the boundary definition:

the first difference of smallest magnitude

The order n of the shape number:

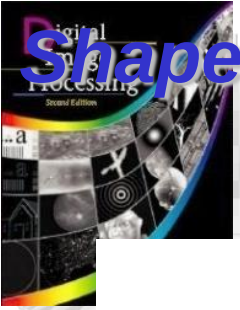
the number of digits in the sequence

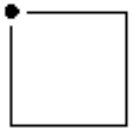


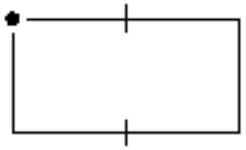


Chain code:	0 3 2 1	0 0 3 2 2 1
Difference:	3 3 3 3	3 0 3 3 0 3
Shape no.:	3 3 3 3	0 3 3 0 3 3

Shape Number (cont.)

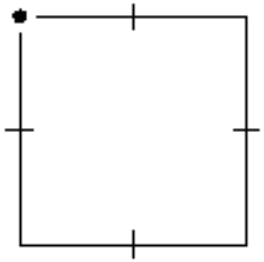


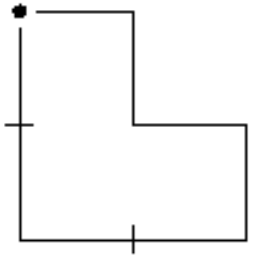




	Order 4	Order 6
Chain code:	0 3 2 1	0 0 3 2 2 1
Difference:	3 3 3 3	3 0 3 3 0 3
Shape no.:	3 3 3 3	0 3 3 0 3 3

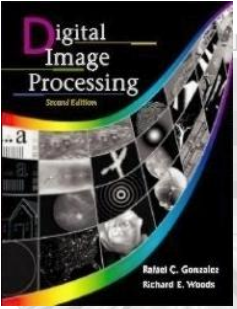
Shape numbers of order 4, 6 and 8





	Order 8		
Chain code:	0 0 3 3 2 2 1 1	0 3 0 3 2 2 1 1	0 0 0 3 2 2 2 1
Difference:	3 0 3 0 3 0 3 0	3 3 1 3 3 0 3 0	3 0 0 3 3 0 0 3
Shape no.:	0 3 0 3 0 3 0 3	0 3 0 3 3 1 3 3	0 0 3 3 0 0 3 3

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.

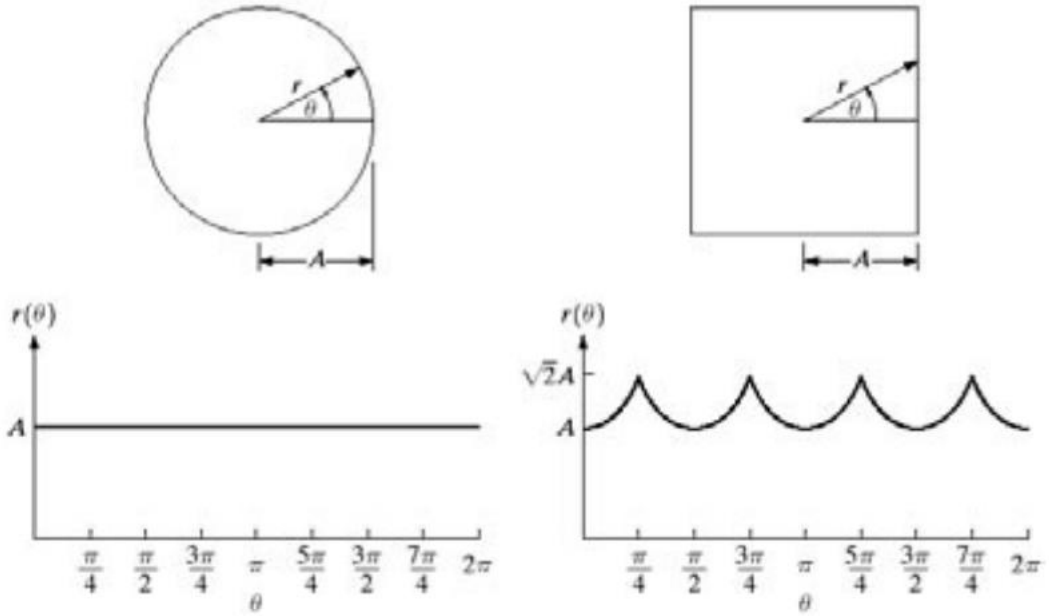


Signature

A simple functional representation that can be used to describe and reconstruct the boundary with appropriate accuracy

1D functional representation of a boundary.

A simple way s to plot the distance from the centroid as a function of the angle.



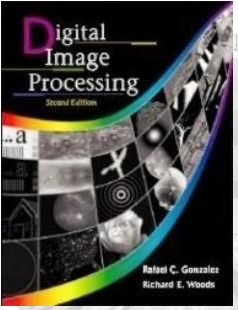
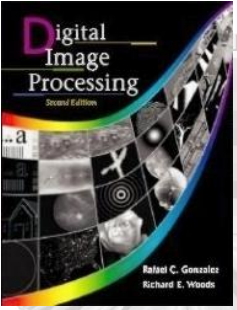


Image Moments

- Image moments are a set of statistical parameters to measure the distribution of where the pixels are and their intensities. Mathematically, the image moment M_{ij} of order (i,j) for a greyscale image with pixel intensities I(x,y) is calculated as

$$M_{ij} = \sum_{x} \sum_{y} x^i y^j I(x,y)$$

Here, x, y refers to the row and column index and I(x,y) refers to the intensity at that location (x,y).

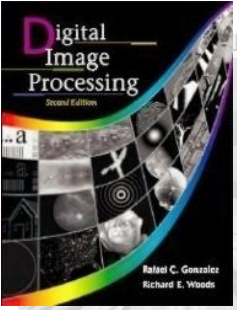


Area:

For a binary image, the zeroth order moment corresponds to the area. Using the above formulae, the zeroth order moment (M00) is given by

$$M_{00} = \sum_x \sum_y I(x,y)$$

For a binary image, this corresponds to counting all the non-zero pixels and that is equivalent to the area. For greyscale image, this corresponds to the sum of pixel intensity values.

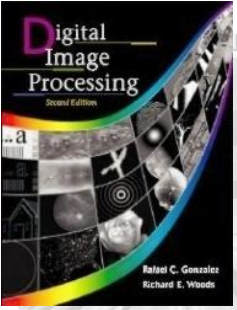


Centroid:

Centroid simply is the arithmetic mean position of all the points. In terms of image moments, centroid is given by the relation

$$\text{Centroid } \{\bar{x}, \bar{y}\} = \left\{ \frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right\}$$

This is simple to understand. For instance, for a binary image M10 corresponds to the sum of all non-zero pixels (x-coordinate) and M00 is the total number of non-zero pixels and that is what the centroid is.



Let's take a simple example to understand how to calculate image moments for a given image.

column →

row ↓	1	2	3	4
1	0	0	0	0
2	1	1	1	1
3	1	1	1	1
4	0	0	0	0

4x4 binary image

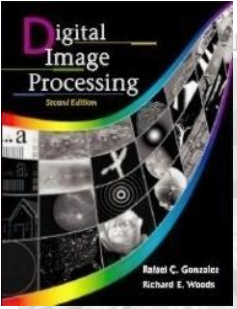
m_{00} = count of all non-zero pixels
= 8

$$\bar{x} = \frac{M_{10}}{M_{00}} = \frac{\sum_x \sum_y x\, I(x,y)}{\sum_x \sum_y I(x,y)}$$

$$= \frac{2\times1 + 2\times1 + 2\times1 + 2\times1 + 3\times1 + 3\times1 + 3\times1 + 3\times1}{8}$$

$$= \frac{20}{8}$$

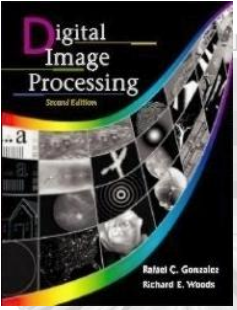
$$\bar{y} = \frac{M_{01}}{M_{00}} = \frac{1+2+3+4+1+2+3+4}{8} = \frac{20}{8}$$



- Central moments are defined as

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x, y)$$

where $\bar{x} = \frac{M_{10}}{M_{00}}$ and $\bar{y} = \frac{M_{01}}{M_{00}}$ are the components of the centroid.



The central moments of order up to 3 are:

$$\mu_{00} = M_{00},$$
$$\mu_{01} = 0,$$
$$\mu_{10} = 0,$$
$$\mu_{11} = M_{11} - \bar{x}M_{01} = M_{11} - \bar{y}M_{10},$$
$$\mu_{20} = M_{20} - \bar{x}M_{10},$$
$$\mu_{02} = M_{02} - \bar{y}M_{01},$$
$$\mu_{21} = M_{21} - 2\bar{x}M_{11} - \bar{y}M_{20} + 2\bar{x}^2 M_{01},$$
$$\mu_{12} = M_{12} - 2\bar{y}M_{11} - \bar{x}M_{02} + 2\bar{y}^2 M_{10},$$
$$\mu_{30} = M_{30} - 3\bar{x}M_{20} + 2\bar{x}^2 M_{10},$$
$$\mu_{03} = M_{03} - 3\bar{y}M_{02} + 2\bar{y}^2 M_{01}.$$