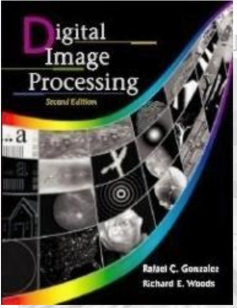


Segmentation
lecture-6
Spring 2019

1. Detection of point discontinuities and Image Edges Detection
   - Detection of discontinuities
   - Detection of lines
   - Detection of edges
2. Edge-point Linking
   - Edge thinning
   - Edge linking
   - Hough Transforms
   - Graph Theoretic techniques
3. Region-based segmentation

- Thresh-holding and threshold selection
- Homogeneity-based segmentation
- Motion-based segmentation
- Image object representation: introduction

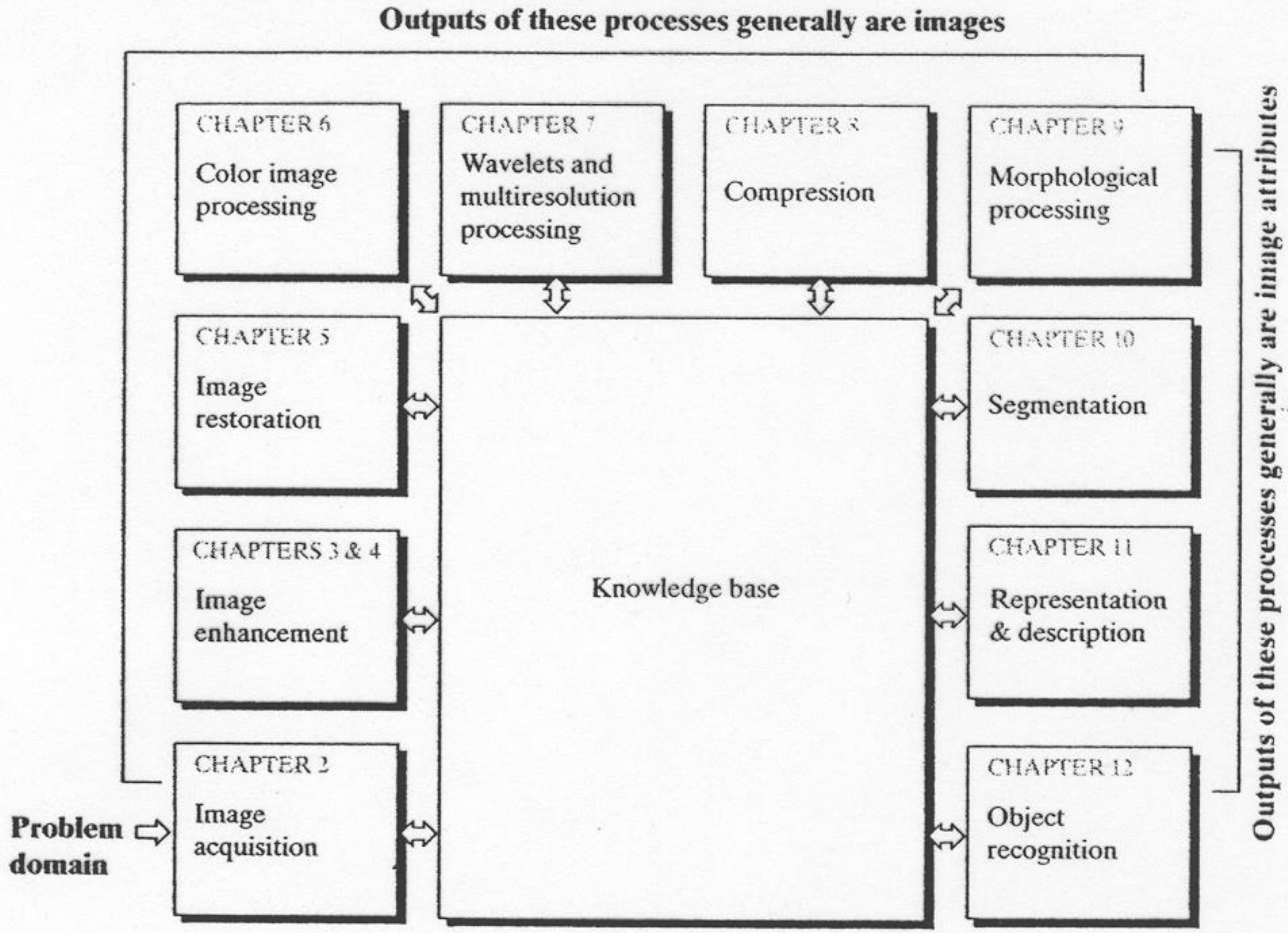


FIGURE 1.23 Fundamental steps in digital image processing.

DETECTION OF POINT DISCONTINUITIES

- Take notes

Point Detection

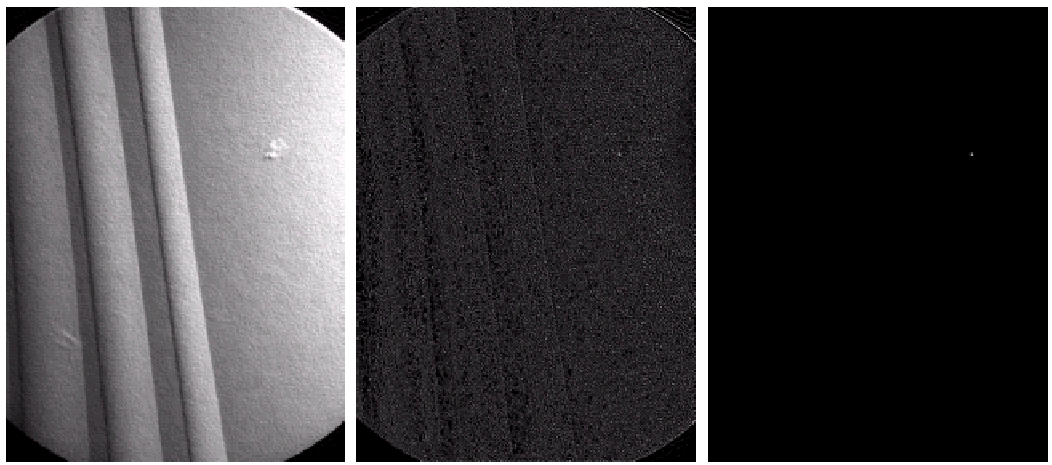
FIGURE 10.1 A general 3 × 3 mask.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Detection of point discontinuities

Image Segmentation

–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.)

Line detection

- Take notes

Line detection masks

FIGURE 10.3 Line masks.

Horizontal

−1	−1	−1
2	2	2
−1	−1	−1

+45°

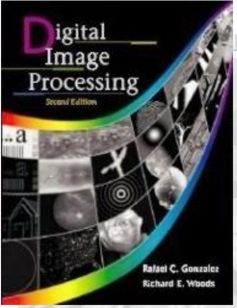
−1	−1	2
−1	2	−1
2	−1	−1

Vertical

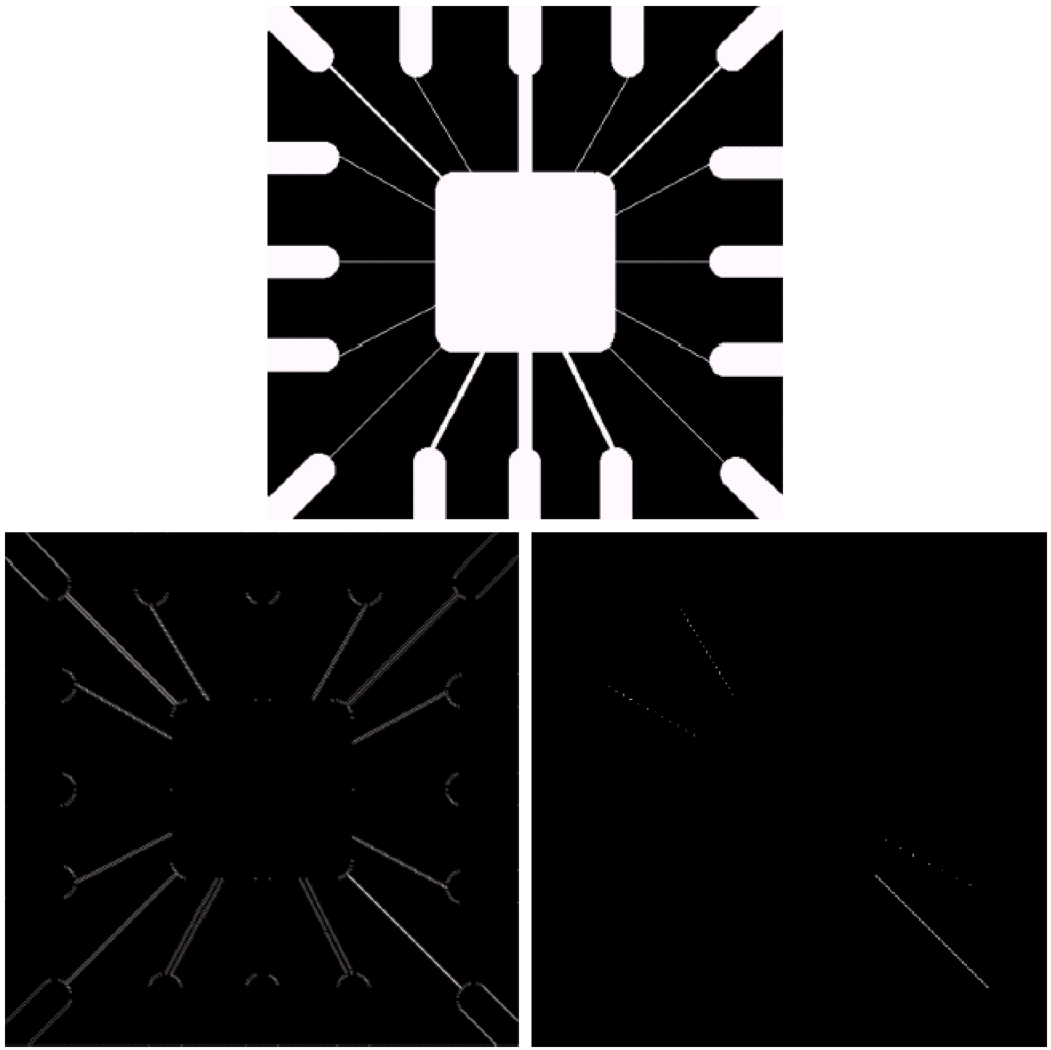
−1	2	−1
−1	2	−1
−1	2	−1

−45°

2	−1	−1
−1	2	−1
−1	−1	2



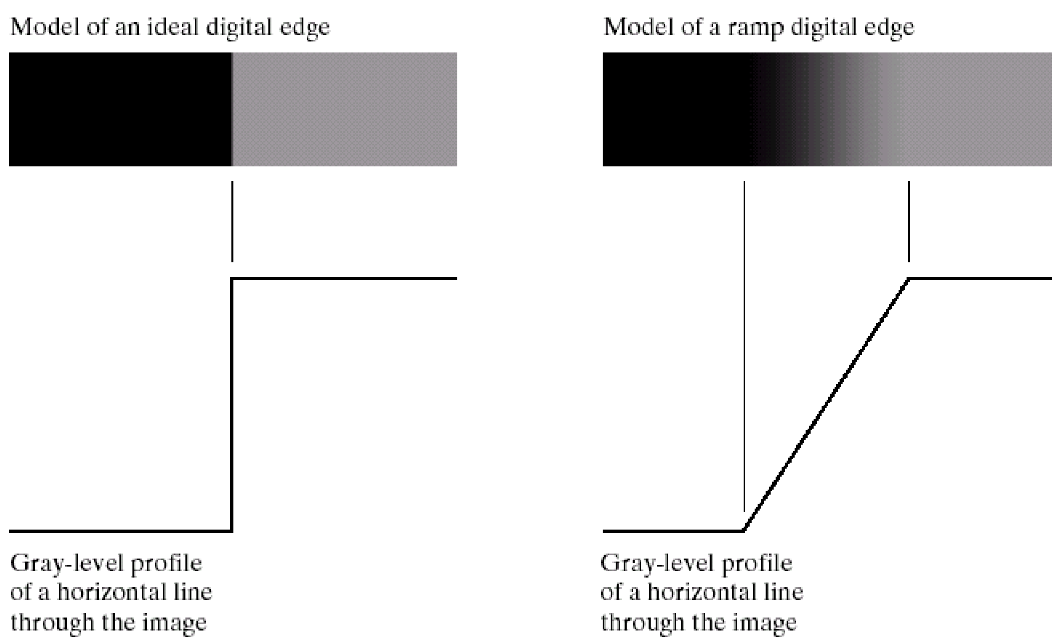
Line Detection examples



a
b c

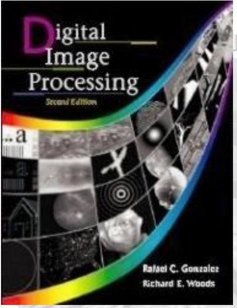
FIGURE 10.4 Illustration of line detection. (a) Binary wire-bond mask. (b) Absolute value of result after processing with $-45°$ line detector. (c) Result of thresholding image (b).

Edge detection
Not a clear cut

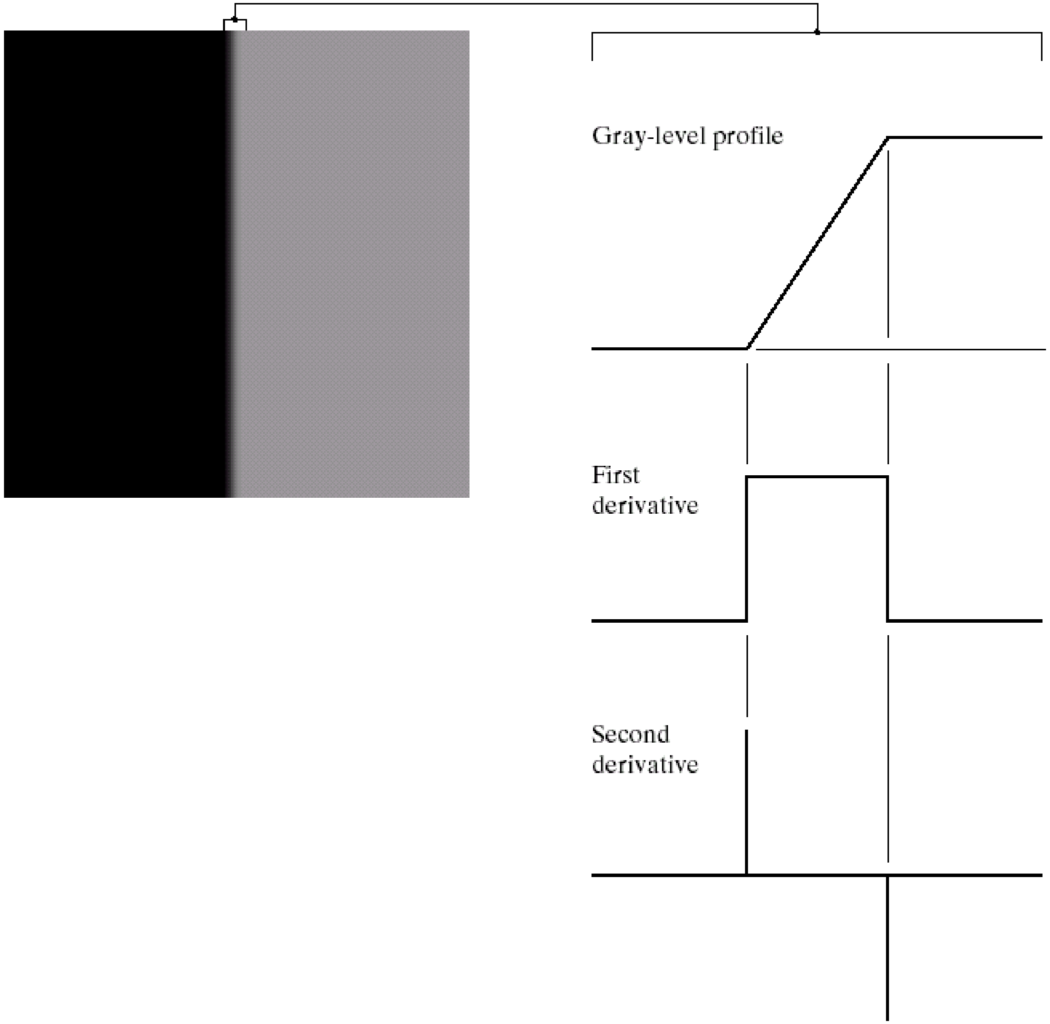


a b

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.



Edge detection:what goes on



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.

Edge detection

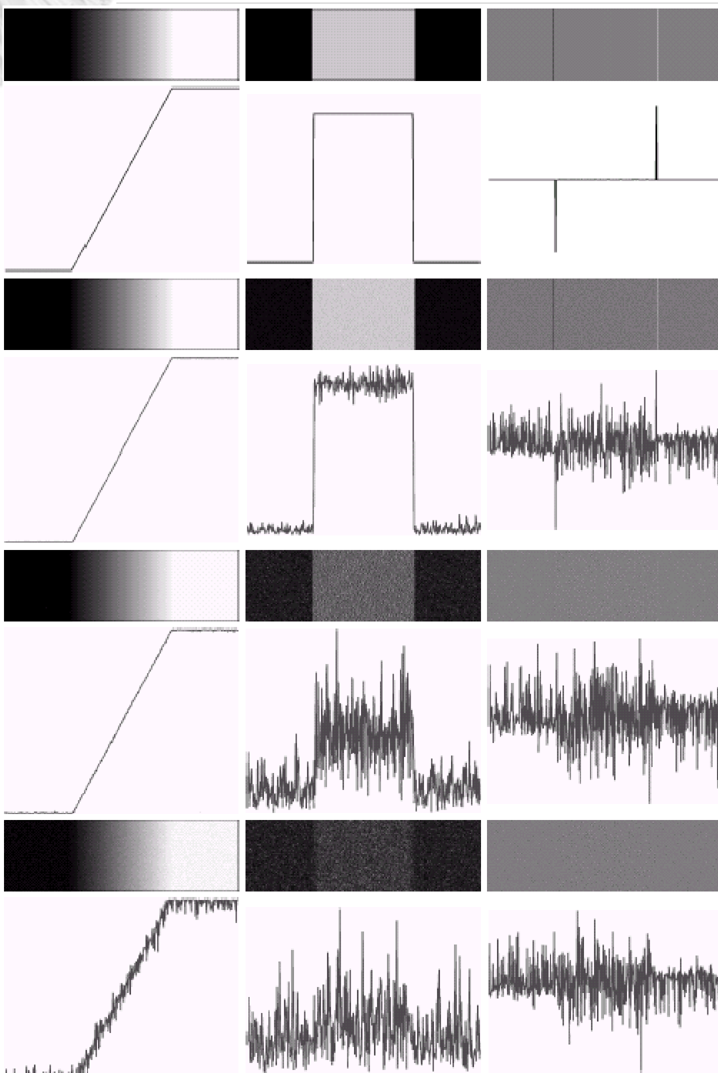
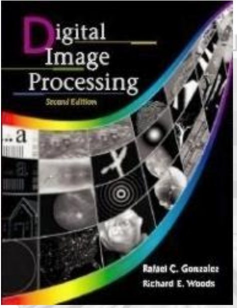


FIGURE 10.7 First column: images and gray-level profiles of a ramp edge corrupted by random Gaussian noise of mean 0 and $\sigma = 0.0, 0.1, 1.0$, and 10.0, respectively. Second column: first-derivative images and gray-level profiles. Third column: second-derivative images and gray-level profiles.

a
b
c
d



Edge detection masks: a brief review

a	
b	c
d	e
f	g

FIGURE 10.8 A 3 × 3 region of an image (the z's are gray-level values) and various masks used to compute the gradient at point labeled z_5.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0
0	1

0	−1
1	0

Roberts

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

Edge detection masks

0	1	1
−1	0	1
−1	−1	0

−1	−1	0
−1	0	1
0	1	1

Prewitt

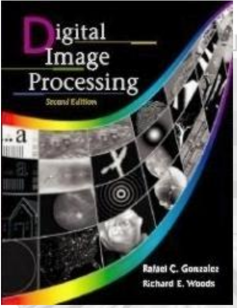
0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

Sobel

a b
c d

FIGURE 10.9 Prewitt and Sobel masks for detecting diagonal edges.

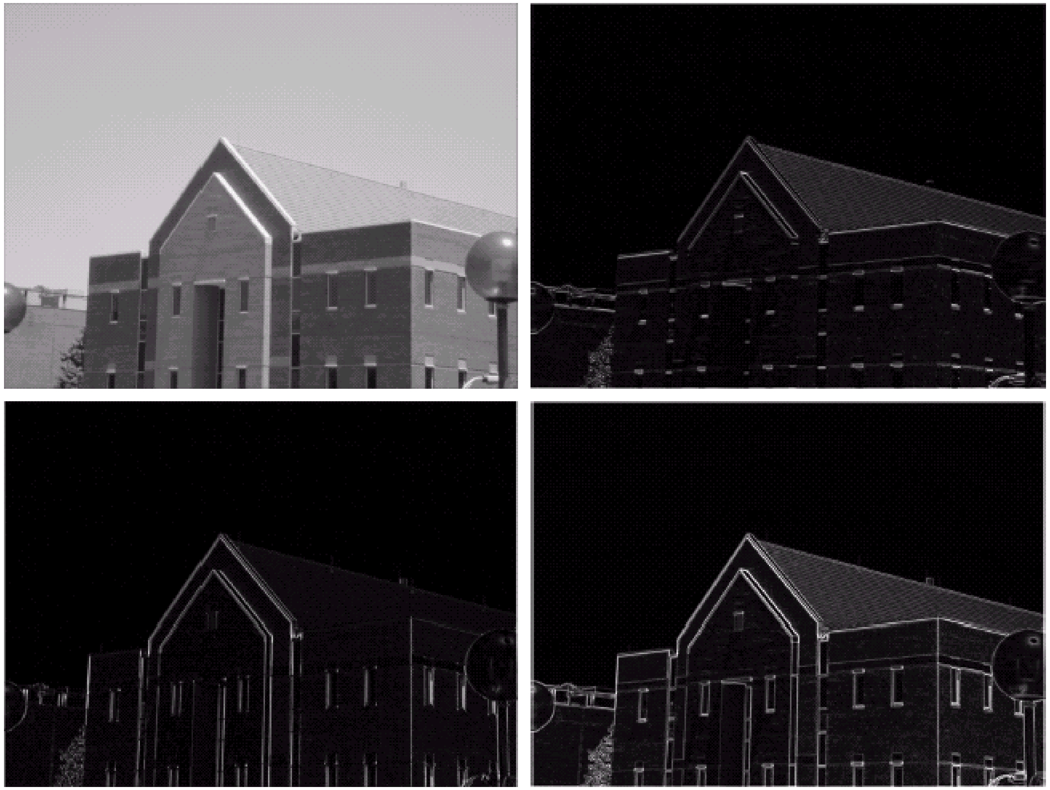


Edge detection

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, $|G_x| + |G_y|$.

Edge detection



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.

Image Segmentation

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).

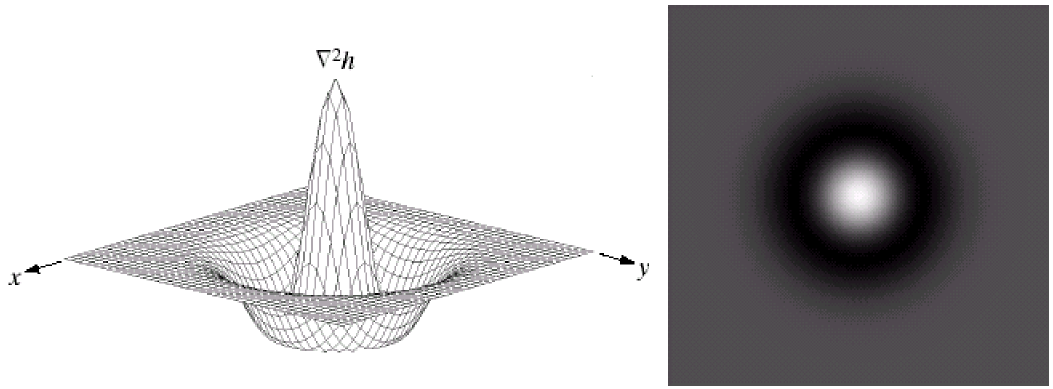
Edge detection: Laplacian

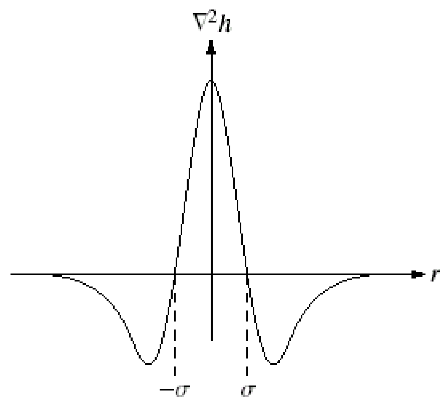
FIGURE 10.13 Laplacian masks used to implement Eqs. (10.1-14) and (10.1-15), respectively.

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

Edge detection: Laplacian



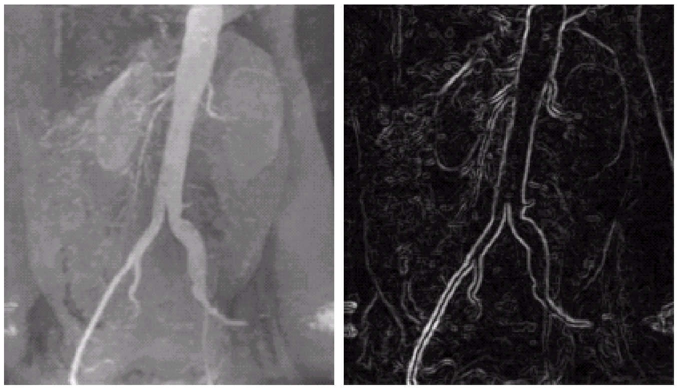


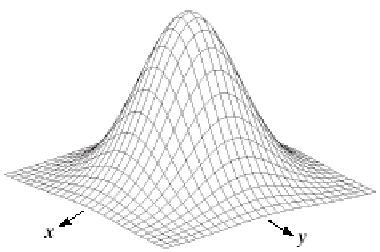
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

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).

Edge detection: laplacian





–1	–1	–1
–1	8	–1
–1	–1	–1

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 POINTS LINKING OF POINT EDGES FOR SEGMENTATION

•

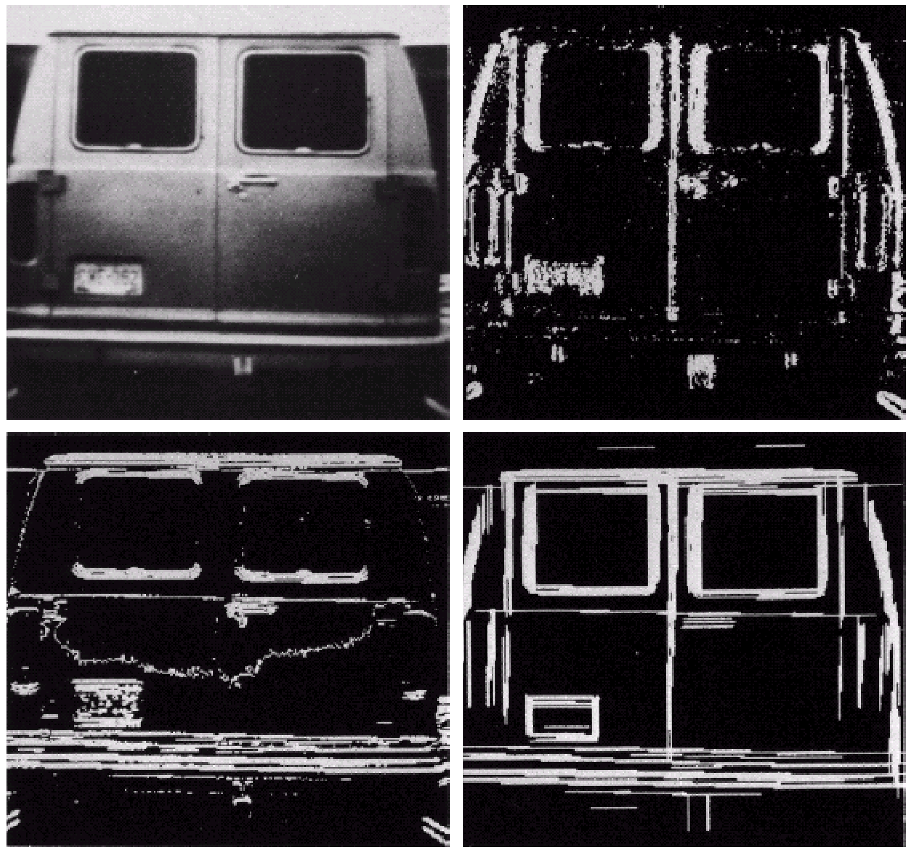
Combined detection for edge detection and linking

- Take notes

Edge Linking: Hough Transform for edge linking

- Take notes

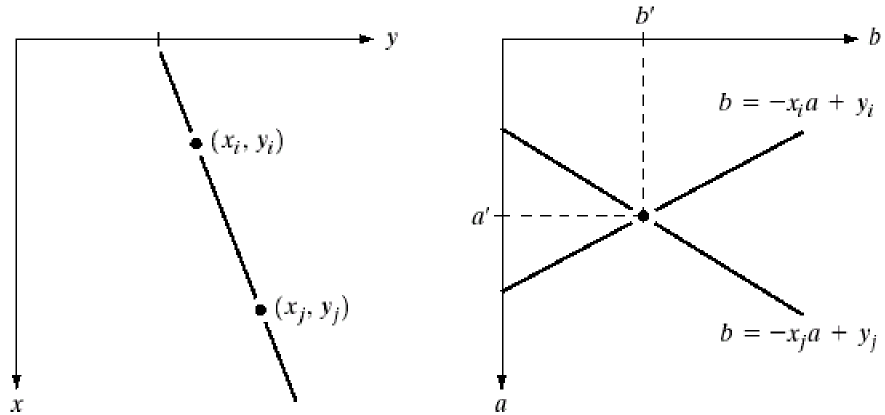
Results of edge points linking (for segmentation) with Hough Transform



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.)

Hough Transform for Image Segmentation



a b

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

Hough Transform for Image Segmentation

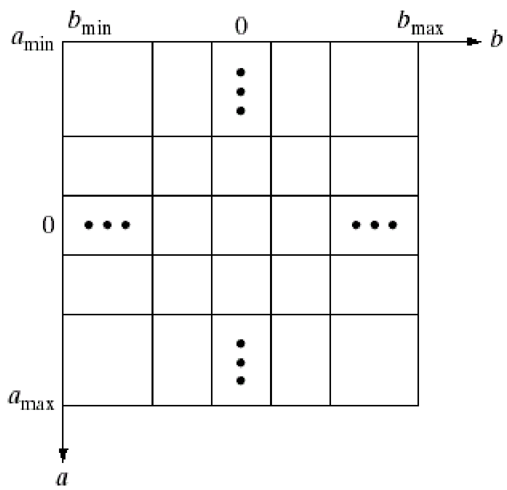
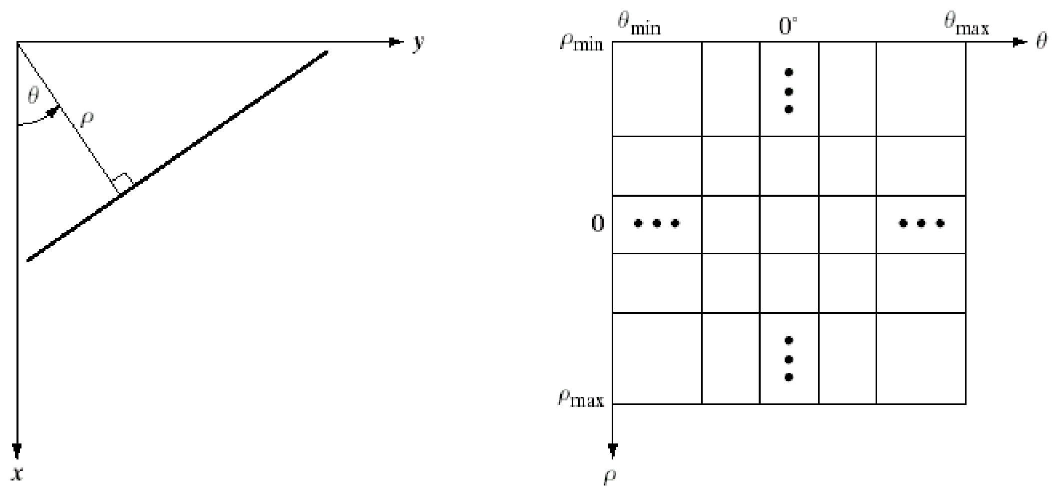


FIGURE 10.18 Subdivision of the parameter plane for use in the Hough transform.

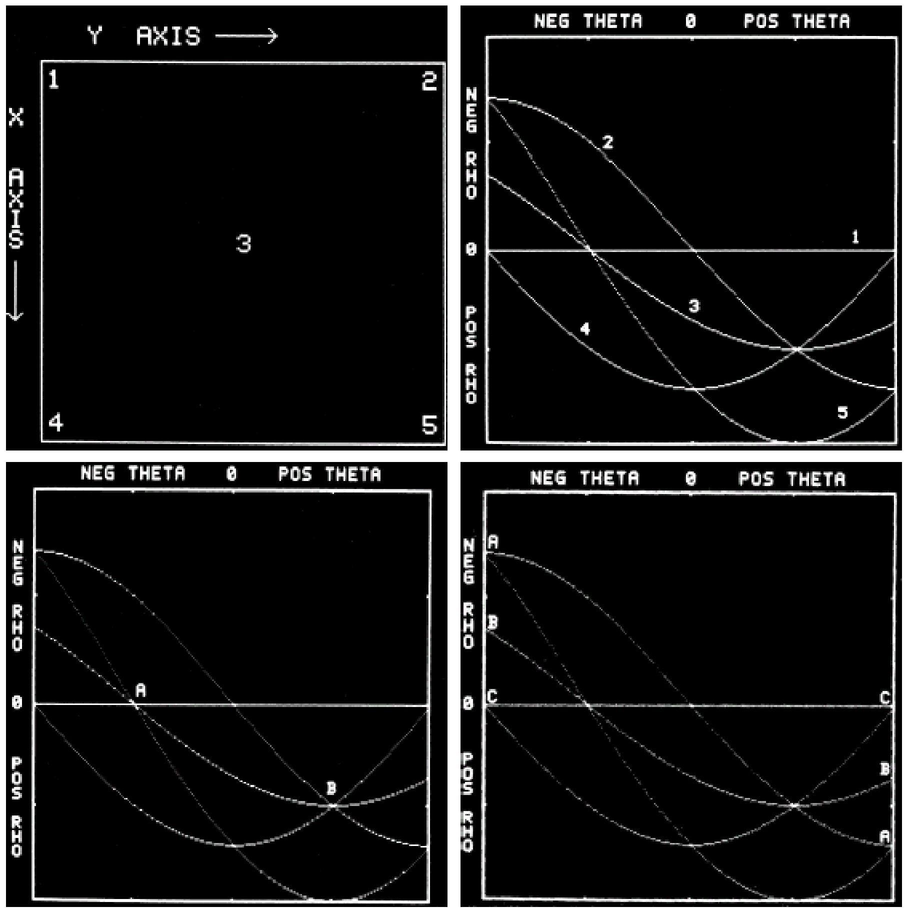
Hough Transform for Image Segmentation



a b

FIGURE 10.19 (a) Normal representation of a line. (b) Subdivision of the $\rho\theta$-plane into cells.

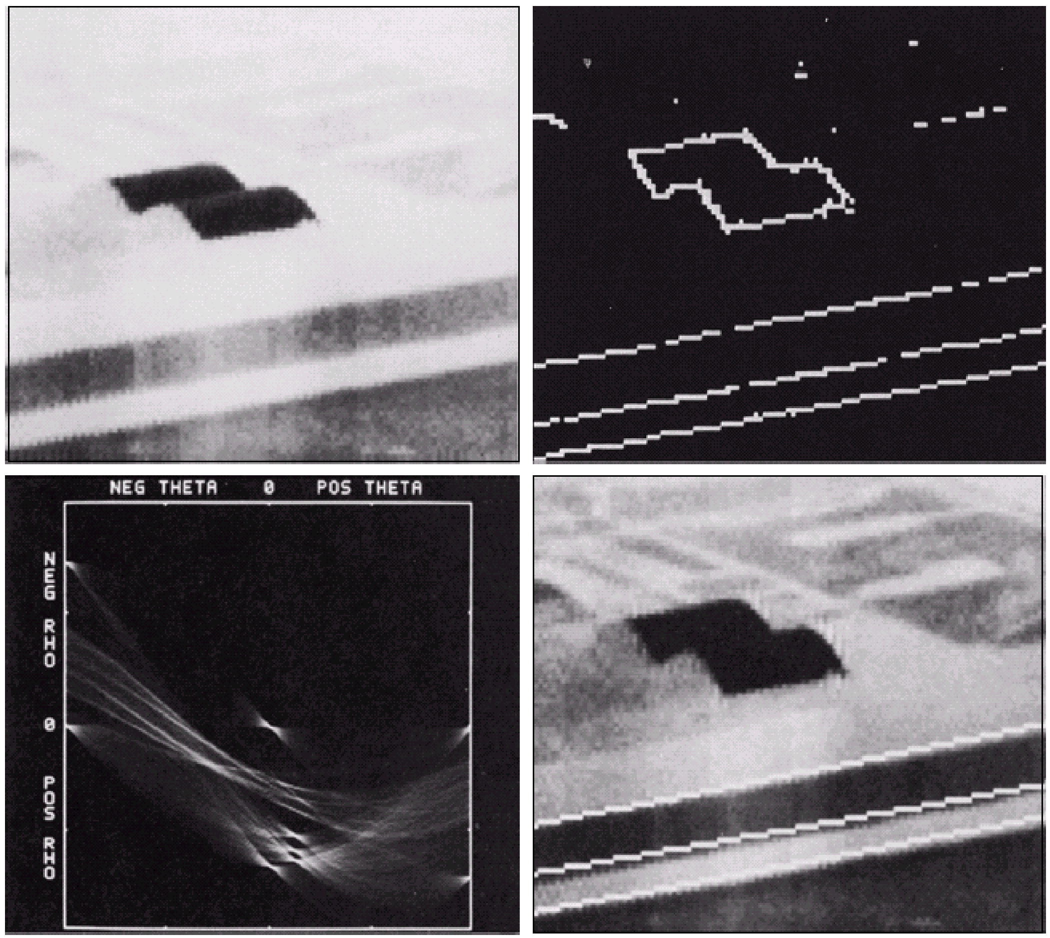
Hough Transform for Image Segmentation



a b
c d

FIGURE 10.20 Illustration of the Hough transform. (Courtesy of Mr. D. R. Cate, Texas Instruments, Inc.)

Image Segmentation With Hough Transform



a b
c d

FIGURE 10.21 (a) Infrared image. (b) Thresholded gradient image. (c) Hough transform. (d) Linked pixels. (Courtesy of Mr. D. R. Cate, Texas Instruments, Inc.)

Edge Linking: Graph Theoretic

•

Graph Theoretic segmentation for edge detection and linking

FIGURE 10.22 Edge element between pixels p and q.

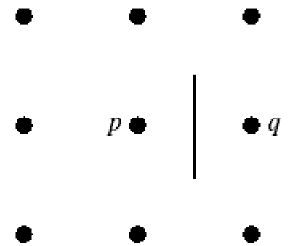
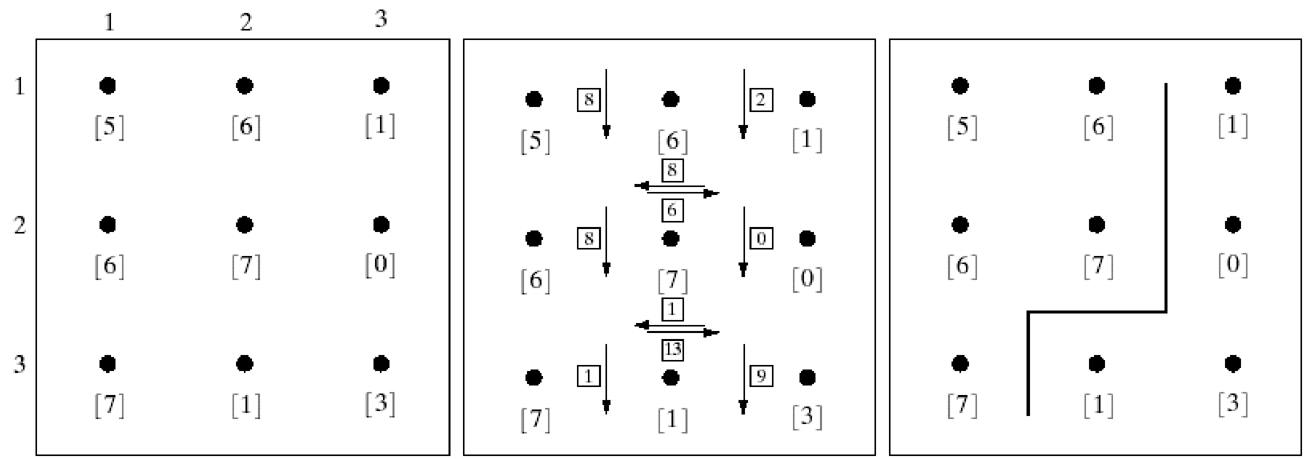


Image Segmentation With Graph Theoretic



a b c

FIGURE 10.23 (a) A 3 × 3 image region. (b) Edge segments and their costs. (c) Edge corresponding to the lowest-cost path in the graph shown in Fig. 10.24.

Graph Theoretic Segmentation

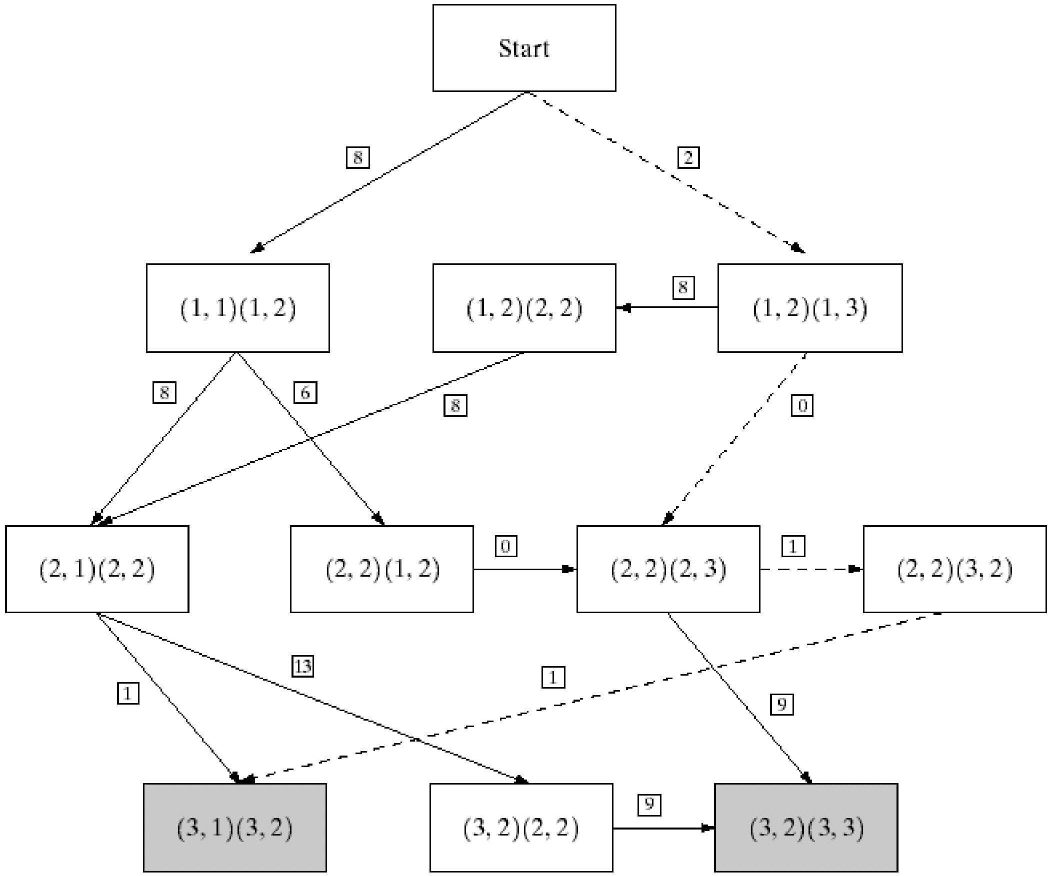


FIGURE 10.24 Graph for the image in Fig. 10.23(a). The lowest-cost path is shown dashed.

Chapter 10
Image Segmentation

FIGURE 10.25 Image of noisy chromosome silhouette and edge boundary (in white) determined by graph search.

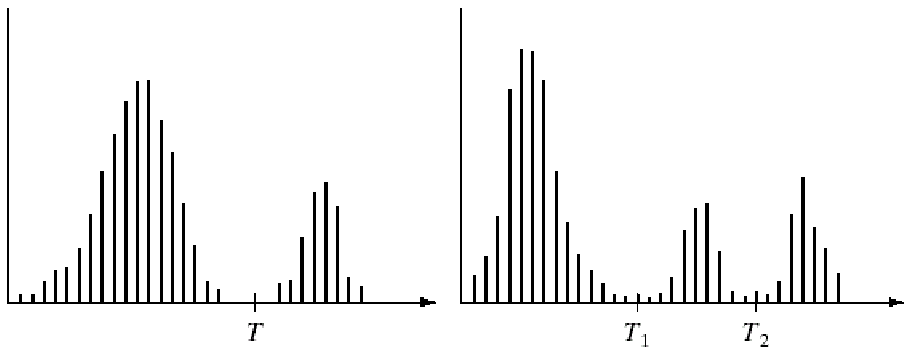
REGION-BASED SEGMENTATION

•

Segmentation by thresh-holding

-

Region-segmentation by thresh-holding



a b

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

Image Segmentation

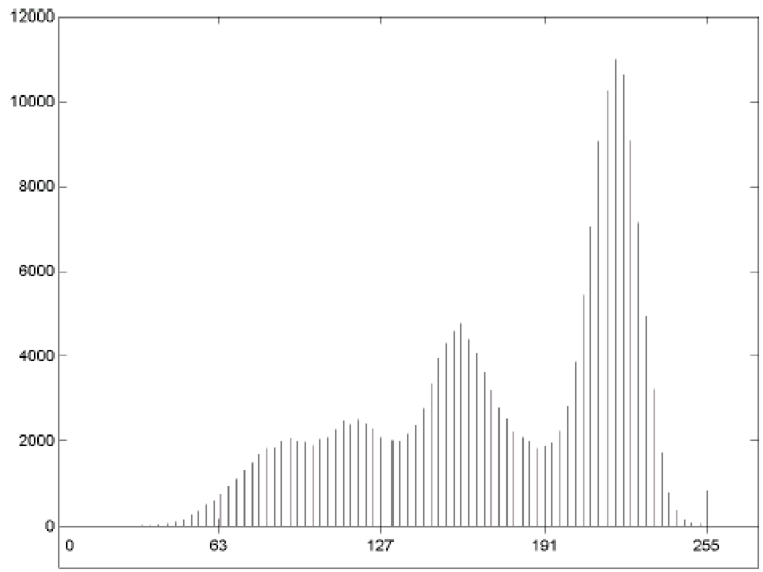
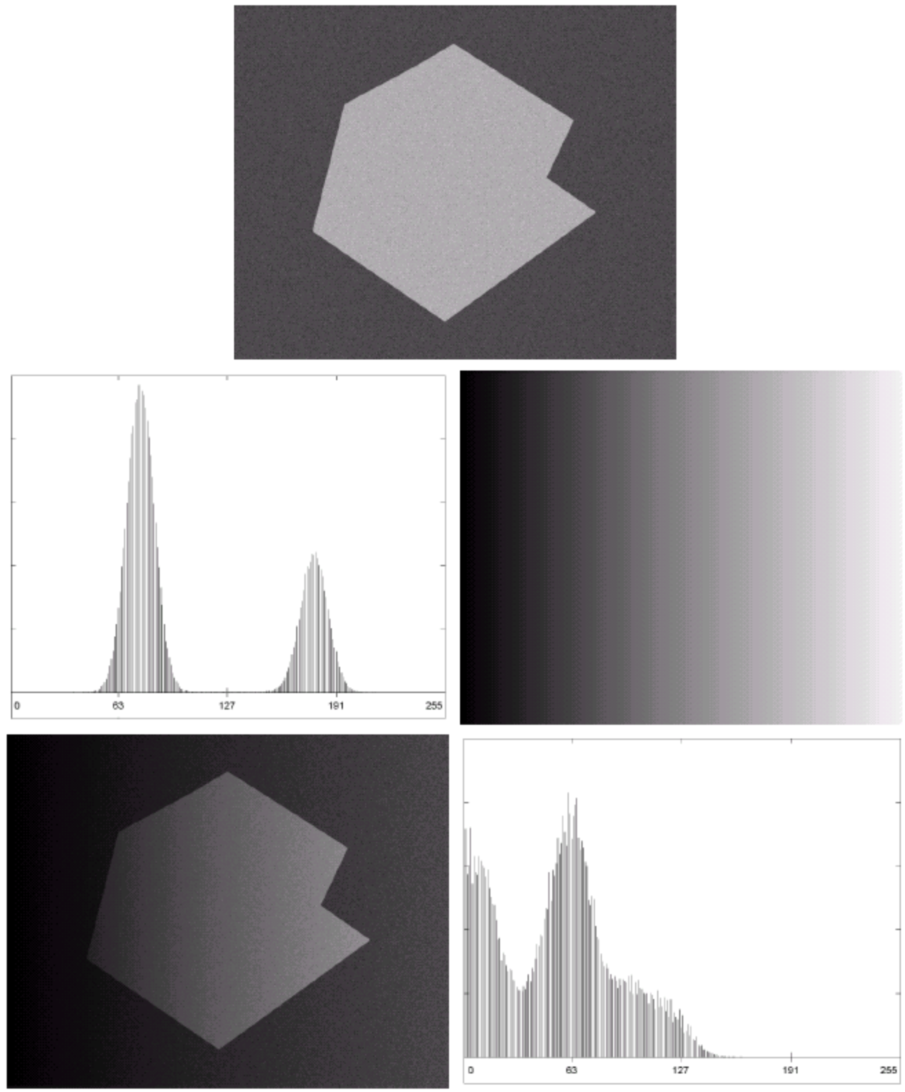


FIGURE 10.41 Histogram of Fig. 10.40(a).

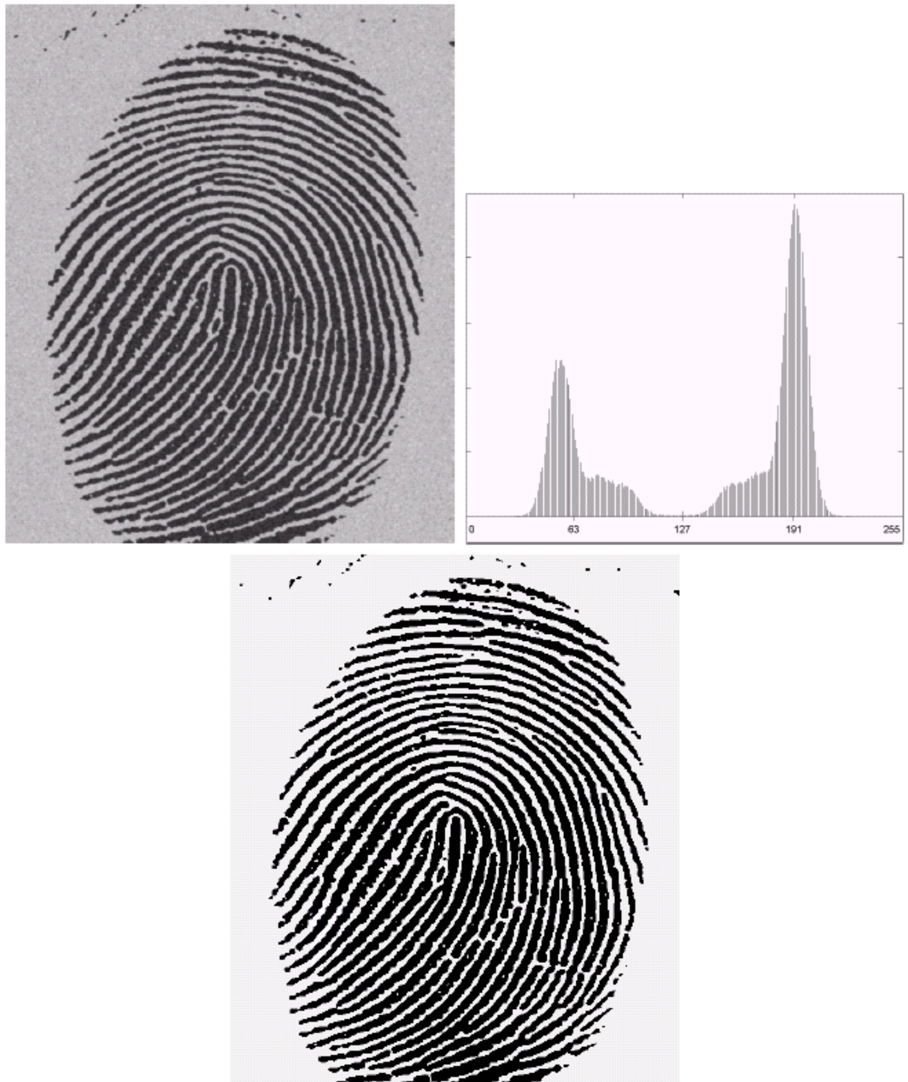
Image Segmentation



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.

Image Segmentation



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.)

IMAGE SEGMENTATION

INTRODUCTION

Segmentation - a process of grouping parts of the image into units (classes, regions, subsets) that are homogeneous with respect to one or more characteristics (or features)

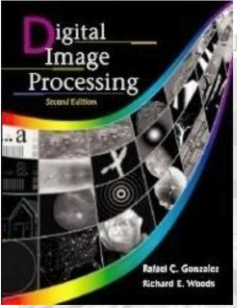
Examples:
(1) when we segment a picture by thresholding its gray level, we are classifying the pixels into "dark" and "light " regions
(2) in edge detection, we are classifying pixels into "edge" and "no edge" classes
(3) in text recognition, we classify pixels into different character classes: "A","B", "a", "b", etc.
(4) when analyzing aerial photo, we classify pixels into terrain regions: "forests", "urban areas", "bodies of water", "roads", etc.
(5) when analyzing medical images, we classify pixels into anatomical regions, such as "bone", "muscle", "blood vessel", etc.

There is no single standard approach to segmentation. The definition of the goal of segmentation varies according to the type of the data and the type of the applications. Different assumptions about the nature of the images being analyzed lead to use of different algorithms.

There are many methods for segmenting an image into regions, which, subsequently, can be analyzed based on their shapes, sizes, relative positions, and other characteristics. The most commonly used segmentation techniques can be classified into two broad categories (1) **region extraction** techniques that look for maximal regions satisfying some homogeneity criterion, (2) **edge extraction** techniques that look for edges occurring between regions with different characteristics.

Thresholding is a common region extraction method. It is based on the assumption that the image has a bimodal histogram and, therefore, contains an object or objects of interest that can be extracted from the background by a simple operation that compares image values with a threshold level. There are several thresholding methods: global methods based on gray level histograms, global methods based on local properties, local threshold selection, and dynamic thresholding.

Clustering is a name of another class of algorithms for image segmentation. Clustering segments the image in terms of sets or clusters of pixels that have strong similarity in the feature space. The basic operation is to examine each pixel and assign it to the cluster that best represents the value of its characteristic vector of features of interest.



Thresholding and clustering operate at the pixel level. Other methods, such as **region growing** operate on groups of pixels. Region growing is a process by which two adjacent regions are assigned to the same segment if their image values are close enough according to some preselected criterion of closeness.

The strategy of **edge-based segmentation** algorithms is to find object boundaries and segment regions enclosed by the boundaries. These algorithms usually operate on edge magnitude and/or phase images produced by an edge operator suited to the expected characteristics of the image. For example, most **gradient operators** such as the Prewitt operator, the Kirsch operator, or the Roberts operator are based on the existence of an ideal step edge. Another example is the Hueckel operator which fits the image function values in a region to an ideal edge element using a least-squares minimization criterion. Other edge-based segmentation techniques are **graph searching**, and **contour following**.

Often there are problems associated with edge detection. First of all, it is difficult to define an adequate model of an edge that will hold over the whole image (since most images are not homogeneous) or over all images of a given application. Also, edge detection does not always provide a complete segmentation and further procedures for edge linking and edge cleaning are required to organize the resulting edges.

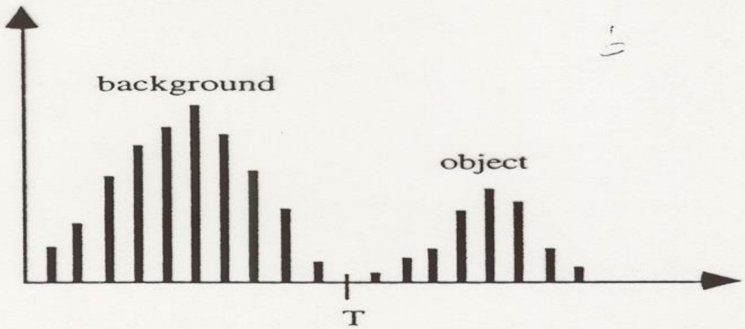
DEFINITION OF SEGMENTATION

Definition 2.2 *Segmentation is grouping pixels into regions, such that*

- $\cup_{i=1}^{k} P_i =$ Entire image ($\{P_i\}$ is an *exhaustive* partitioning.)
- $P_i \cap P_j = \emptyset$, $i \neq j$ ($\{P_i\}$ is an *exclusive* partitioning.)
- Each region P_i satisfies a predicate; that is, all points of the partition have *some* common property.
- Pixels belonging to adjacent regions, when taken jointly, do not satisfy the predicate.

THRESHOLDING

Thresholding is based on the assumption that a histogram has a bimodal histogram and, therefore, the object can be extracted from the background by a simple operation that compares image values with a **threshold level T**. Suppose that we have an image F[i,j] with the histogram shown below:



The object and background pixels have gray levels grouped into two dominant modes. One obvious way to extract the object from the background is to select a threshold T that separates these modes.

Thresholded image $F_T[i,j]$ is defined as:

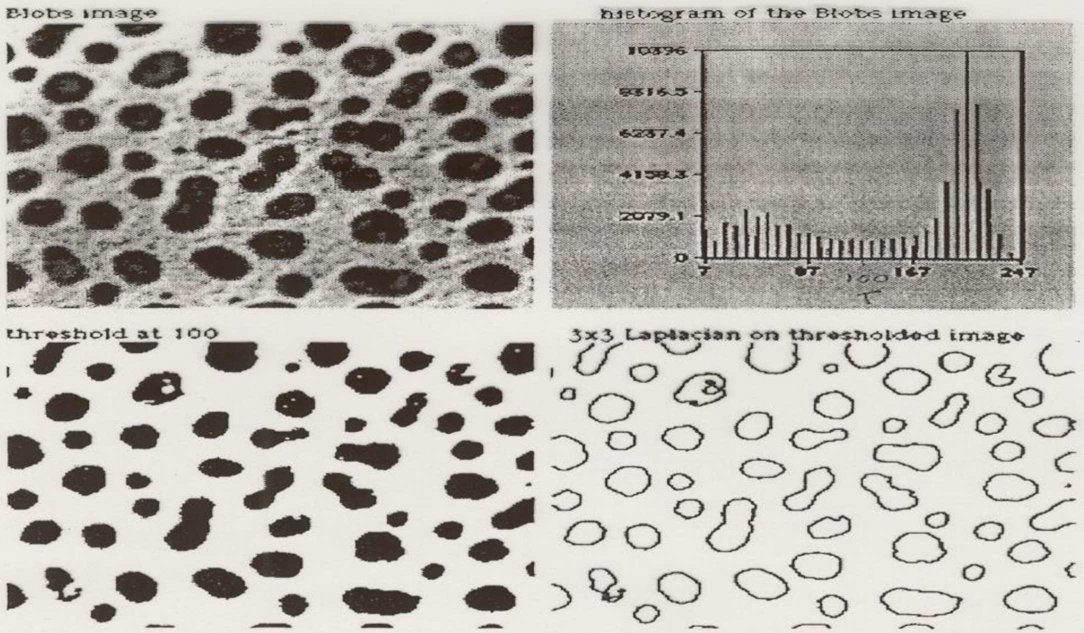
$$F_T[i,j] = \begin{cases} 1 & \text{if } F[i,j] \geq T \\ 0 & \text{if } F[i,j] < T \end{cases} \qquad (1)$$

Thus pixels labeled 1 (or any other convenient intensity level) correspond to objects, whereas pixels labeled 0 correspond to the background.

gray-level values of the pixel [i,j]

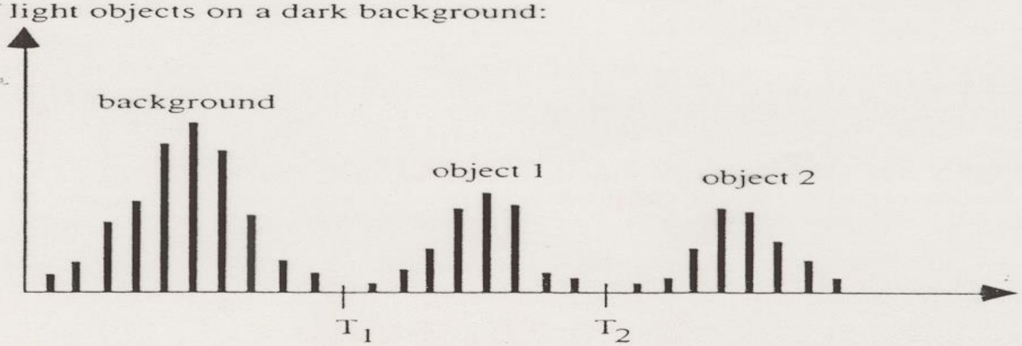
When T depends only on F[i,j], the threshold is called **global**. If T depends on F[i,j] and some local properties of the neighborhood of pixel [i,j] - for example, average gray level of the neighborhood - the threshold is called **local**. If, in addition, T depends on the spatial coordinates i and j, the threshold is called **dynamic** or **adaptive**.

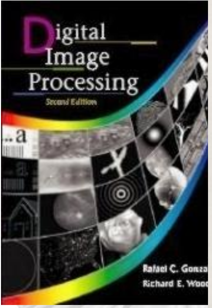
Figure below shows an example of simple thresholding applied to the "Blobs" image. Please note that the edges of the blobs were obtained by a 3x3 Laplacian applied to the thresholded image.



Multilevel thresholding

If an image contains more than two types of regions, it may still be possible to segment it by applying several thresholds. Figure below shows a histogram of an image containing two types of light objects on a dark background:





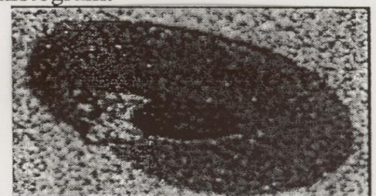
A pixel [i,j] belongs to "object 1" class if $T_1 < F[i,j] \leq T_2$; to "object 2" class if $F[i,j] > T_2$, and to the background if $F[i,j] \leq T_1$.

As the number of region types increases, the peaks become harder to distinguish, and segmentation by thresholding becomes more difficult.

Smoothing and thresholding

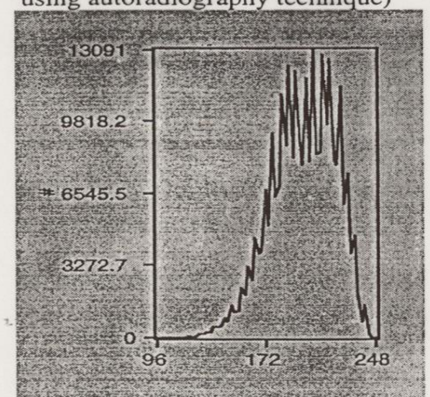
The gray level subpopulations corresponding to the different types of regions in an image will often overlap and segmentation by thresholding becomes difficult. **Smoothing** an image before thresholding usually helps to alleviate that problem. For example, we can simply run the **average** or **median** filter on the original image. Smoothing will, of course, blur the borders of the regions, but thresholding will still extract the regions correctly. Figures below illustrate how a 7x7 median filter sharpened the peaks on an image histogram.

original image (kidney; image obtained using autoradiography technique)

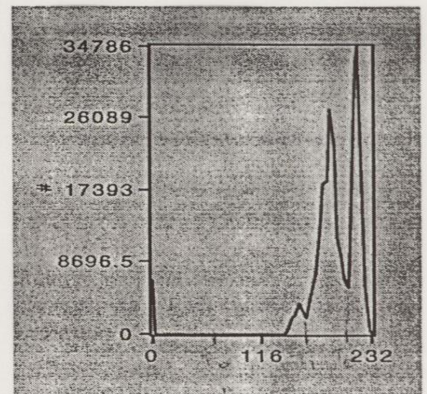


7x7 median filtered image

Smoothing



histogram of the original image



histogram of the median-filtered image

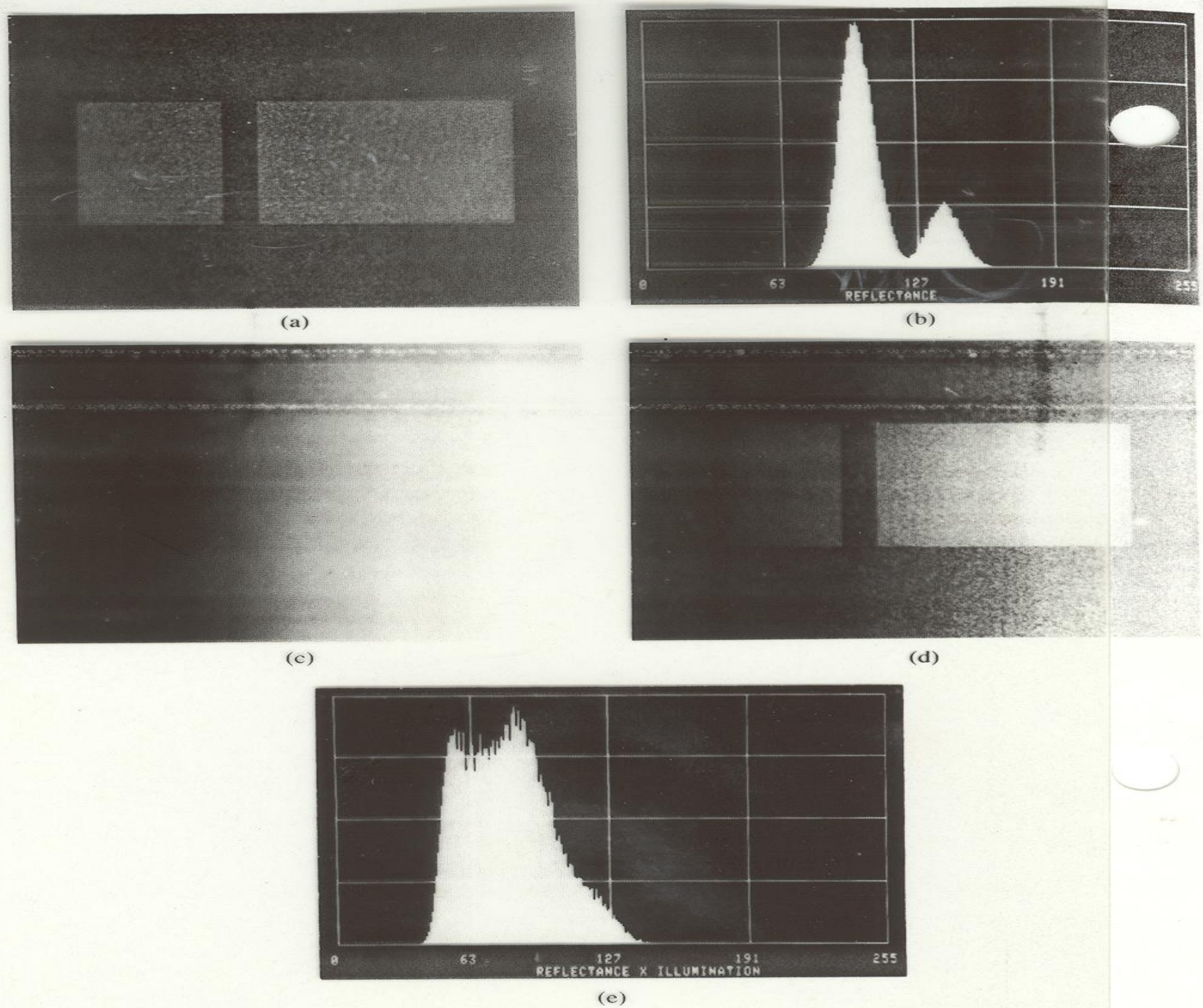


Figure 7.26 *(a) Computer generated reflectance function; (b) histogram of reflectance function; (c) computer generated illumination function; (d) image produced by the product of the illumination and reflectance functions; (e) histogram of image.*

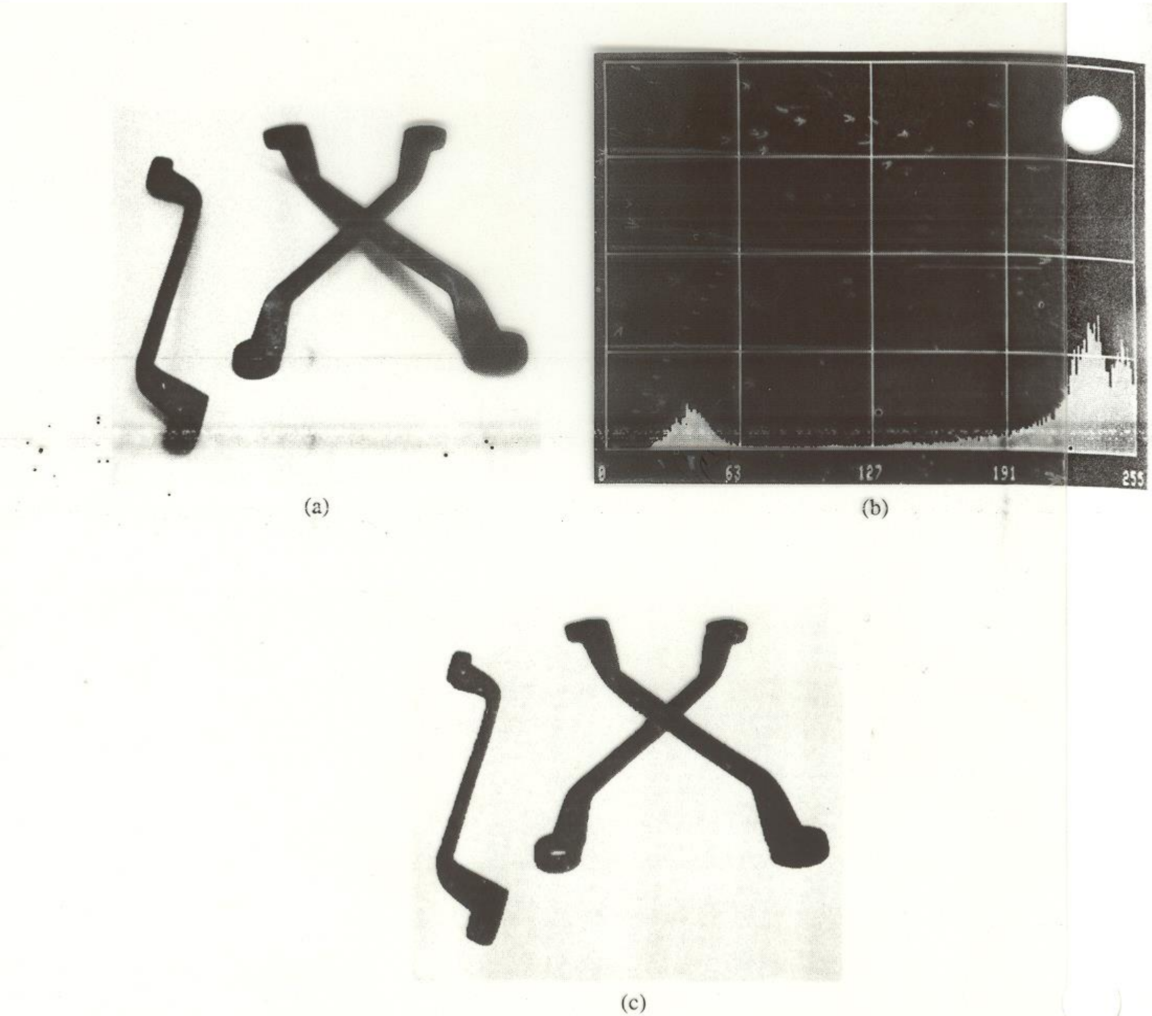
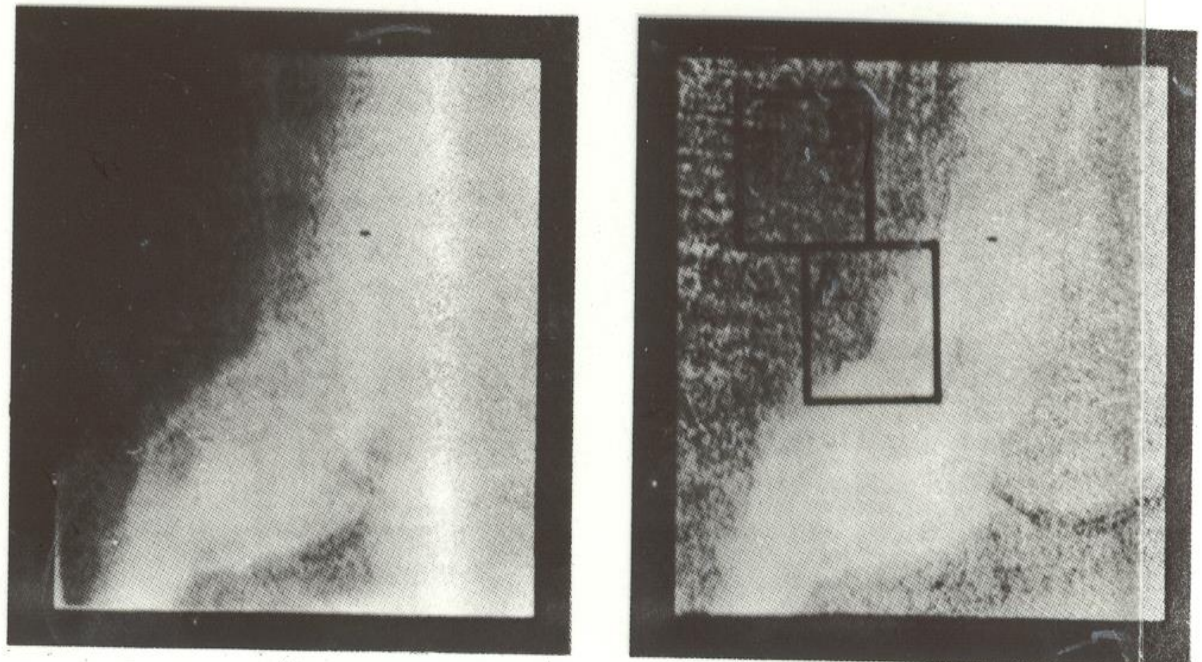


Figure 7.27 *Example of global thresholding: (a) original image and (b) its histogram; (c) result of segmentation with $T = 90$. (From Fu, Gonzalez, and Lee [1987].)*

Dynamic segmentation for thresh-holding

-

Example: The following discussion of an approach developed by Chow and Kaneko [1972] for outlining boundaries of the left ventricle in cardioangiograms



(a) (b)

Figure 7.28 A cardioangiogram before and after processing. (From Chow and Kaneko [1972].)

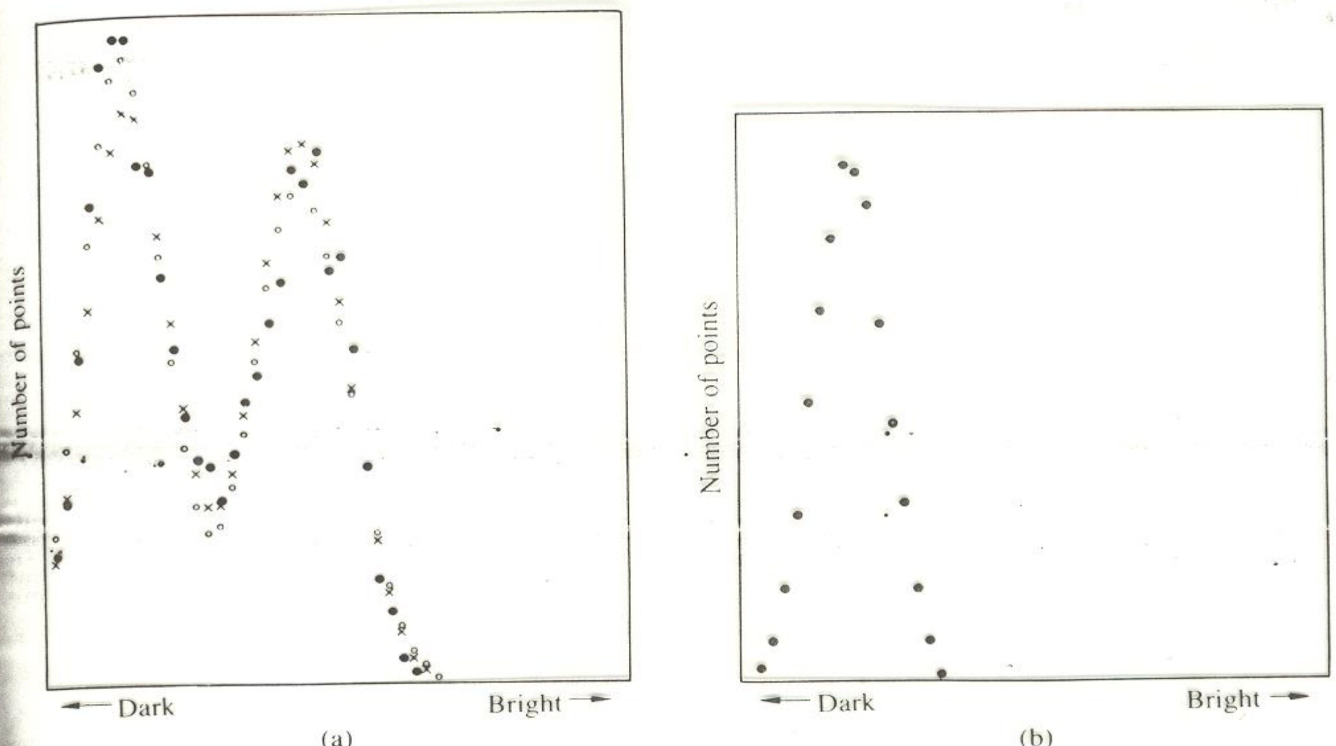
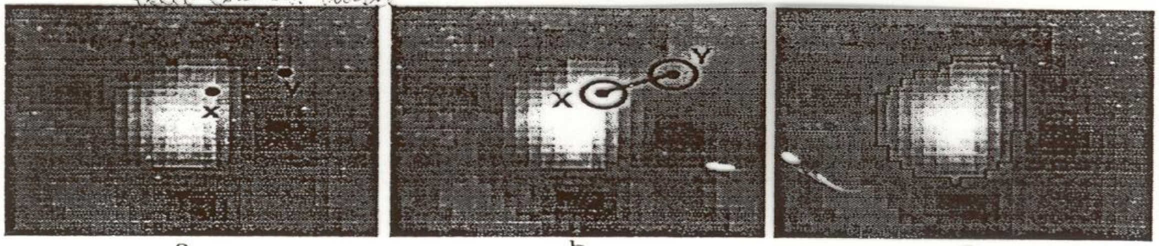


Figure 7.29 Histograms (black dots) of regions A and B in Fig. 7.28(b). (From Chow and Kaneko [1972].)

Figure 7.30 *Cardioangiogram showing superimposed boundaries. (From Chow and Kaneko [1972].)*



Semi-automated thresholding based on local properties technique applied to the lymph node images; (a) original lymph node image with two selected pixels: x - inside and y - outside the node, (b) circular regions (circle X and Y) around pixels x and y, (c) segmented lymph node.

Some other local properties that can be used in selecting the threshold are: mean, variance, skewness, kurtosis, median, mode, entropy, etc.

Dynamic (adaptive) thresholding

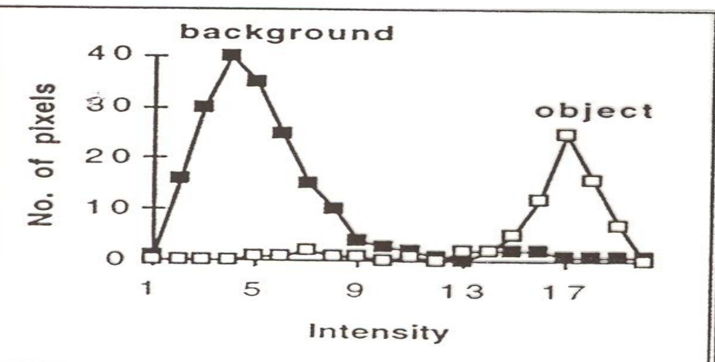
In many cases, no single threshold gives good segmentation results over an entire picture. Suppose, for example, that the image shows dark objects on a light background, but it was obtained under conditions of uneven illumination. The objects will still contrast with the background, but both the background and objects may be lighter on one side of the image than on the other. Thus, a single threshold will not separate objects from background. Similar situation occurs when an image contains shadows, or when the picture was obtained by a sensor whose sensitivity varies form point to point.

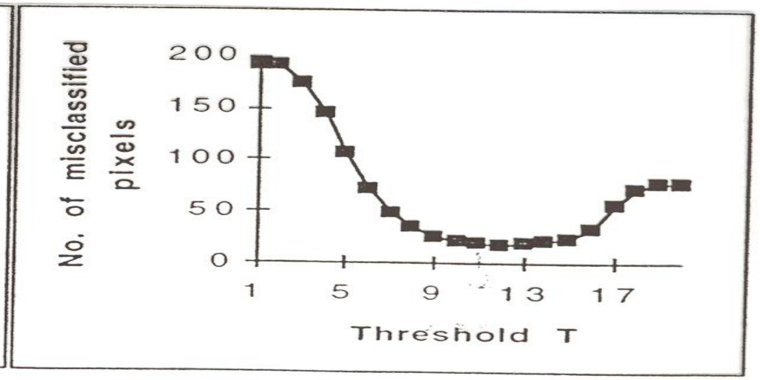
If the uneven illumination is described by some known function of position in the picture, one could attempt to correct for it using gray level correction techniques, after which a single threshold should work for the entire image. If this information is not available, one can **divide the image into blocks and apply threshold selection techniques to each block.** If a block contains both objects and background, its histogram should be bimodal, and the valley bottom should yield a **local threshold**. If a block contains objects only, or background only, it will not have a bimodal histogram, and no threshold will be found for it; but a threshold can still be assigned to it by **interpolation** from the local thresholds found for nearby bimodal blocks (some smoothing of the resulting thresholds may be necessary, since if a threshold changes abruptly from one block to another, artifacts may result).

Another situation in which local thresholding may be useful is where the objects to be segmented are very **small and sparse** (bubble tracks, stars, etc.), so that the image consists almost entirely of background, and the objects produce no detectable peak on its histogram. An example of local histogramming and threshold interpolation is shown in the next figure:

For example, table below shows the pixel distributions for sampled regions of background and object. The number of misclassified pixels for different threshold levels T is calculated and plotted for T. For example, a number of misclassified pixels for threshold level T=11 is 20 (sum of the shadowed areas of object and background distributions). The least number of misclassified pixels (18) is obtained for T=12.

	background	object	Number of misclassified pixels
1	1	0	192
2	16	0	191
3	30	0	175
4	40	0	145
5	35	1	106
6	25	1	72
7	15	2	49
8	10	1	35
9	4	1	26
10	3	0	22
11	2	1	20
12	1	0	18
13	0	2	19
14	2	2	21
15	2	5	24
16	2	12	34
17	1	25	57
18	1	16	72
19	1	7	78
20	1	0	77





This segmentation technique is illustrated on figure below:

Optimal thresholding

Suppose that an image contains only two principal brightness regions. The histogram of such an image may be considered an estimate of the brightness probability density function, p(z). This overall density function is the sum or mixture of two unimodal densities, one for the light and one for the dark regions in the image. If the form of the densities is known or assumed, determining an optimal threshold (in terms of minimum error) for segmenting the image into the two brightness regions is possible.

Suppose that the probability densities of values z for the two classes of pixels be $p(z|1)$ and $p(z|2)$, respectively. Let the a priori probabilities of the classes be P(1) and P(2), so that $P(1)+P(2)=1$; then the overall probability density of the values z for the entire image is:

$$p(z) = P(1)p(z|1) + P(2)p(z|2) \qquad (2)$$

Suppose that we classify the pixels by thresholding z at T; in other words, pixels for which $z<T$ are assigned to class 1, and those for which $z \geq T$, to class 2. Then the probability of misclassifying a class 2 pixel as class 1 is

$$E_1(T) = \int_{-\infty}^{T} p(z|2)dz \qquad (3)$$

Similarly, the probability of misclassifying a class 1 pixel as class 2 is

$$E_2(T) = \int_{T}^{\infty} p(z|1)dz \qquad (4)$$

Therefore the overall probability of error is

$$E(T) = P(2)E_1(T) + P(1)E_2(T) \qquad (5)$$

To find the value of T for which this probability is a minimum requires differentiating E(T) with respect to T and setting the result to zero, thus obtaining

$$P(2)p(z|2) = P(1)p(z|1) \qquad (6)$$

In general, Eq. 6 can be solved for t numerically. If we know the mathematical forms of the probability densities for $p(z|1)$ and $p(z|2)$, it may also be possible to solve it analytically.

Thresholding based on the local properties

Local properties can be used to aid in the selection of global thresholds. One example is the semi-automated thresholding technique, where the two points are selected - one inside an object (x) and one in the background (y). By comparing the distribution of pixel intensities in region X around pixel x and region Y around pixel y, we can use Eqs. 3-5 to determine the border value (threshold T), which results in the least number of misclassified pixels between the two distributions.

Optimal thresh-holding

- Take notes

P-tile method

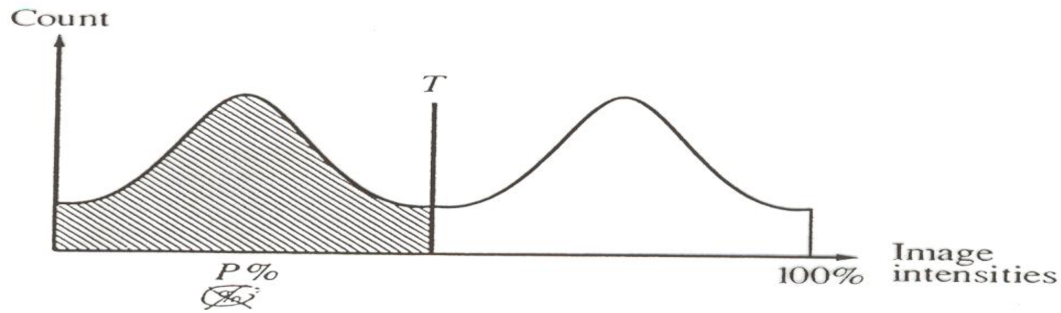


Figure 3.2: The shaded areas in the histogram represent p percent of the image area. The threshold is selected so that p percent of the histogram is assigned to the object.

P-tile method uses knowledge about the area or size of the desired object to threshold an image

Iterative Threshold Method

The method starts with an approximate threshold and then successively refines this estimate.

Algorithm 3.2 Iterative Threshold Selection

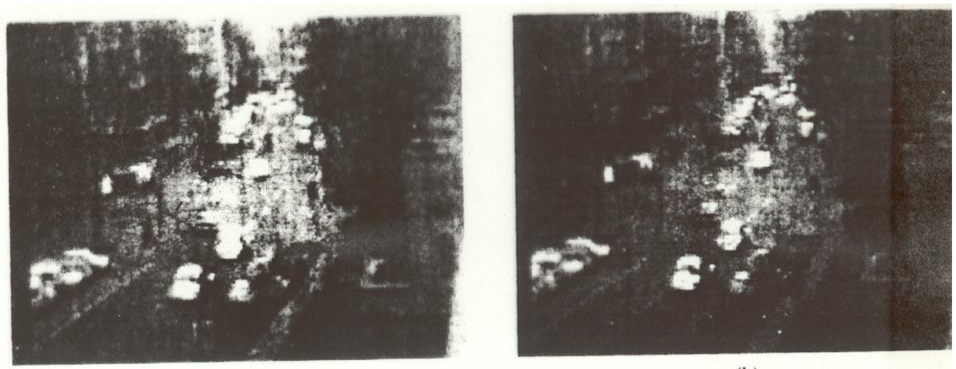
1. *Select an initial estimate of the threshold, T. A good initial value is the average intensity of the image.*
2. *Partition the image into two groups, R_1 and R_2, using the threshold T.*
3. *Calculate the mean gray values μ_1 and μ_2 of the partitions R_1 and R_2.*
4. *Select a new threshold:*

$$T = \frac{1}{2}(\mu_1 + \mu_2).$$

5. *Repeat steps 2–4 until the mean values μ_1 and μ_2 in successive iterations do not change.*

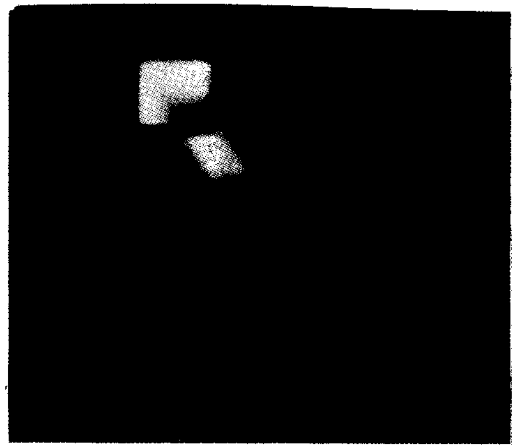
(a) (b)

Figure 7.44 *Two image frames of a traffic scene. There are two principal moving objects: a white car in the middle of the picture and a pedestrian on the lower left. (From Jain [1981].)*

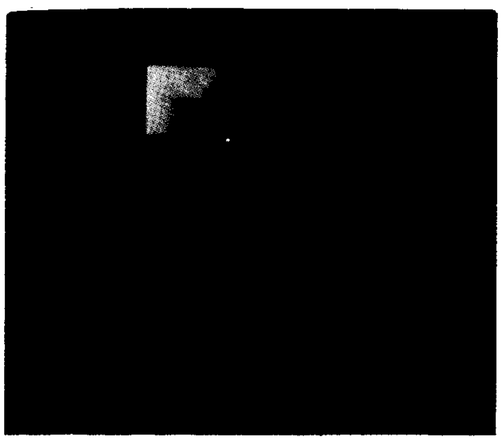


(a) (b)

Figure 7.45 *(a) Image with automobile removed and background restored; (b) image w pedestrian removed and background restored. The latter image can be used as a referen (From Jain [1981].)*



(a)



(b)

(c)

Figure 7.43 *Intensity-coded accumulative difference images for Fig. 7.42: (a) AADI, (b) PADI, and (c) NADI. (From Jain [1983].)*

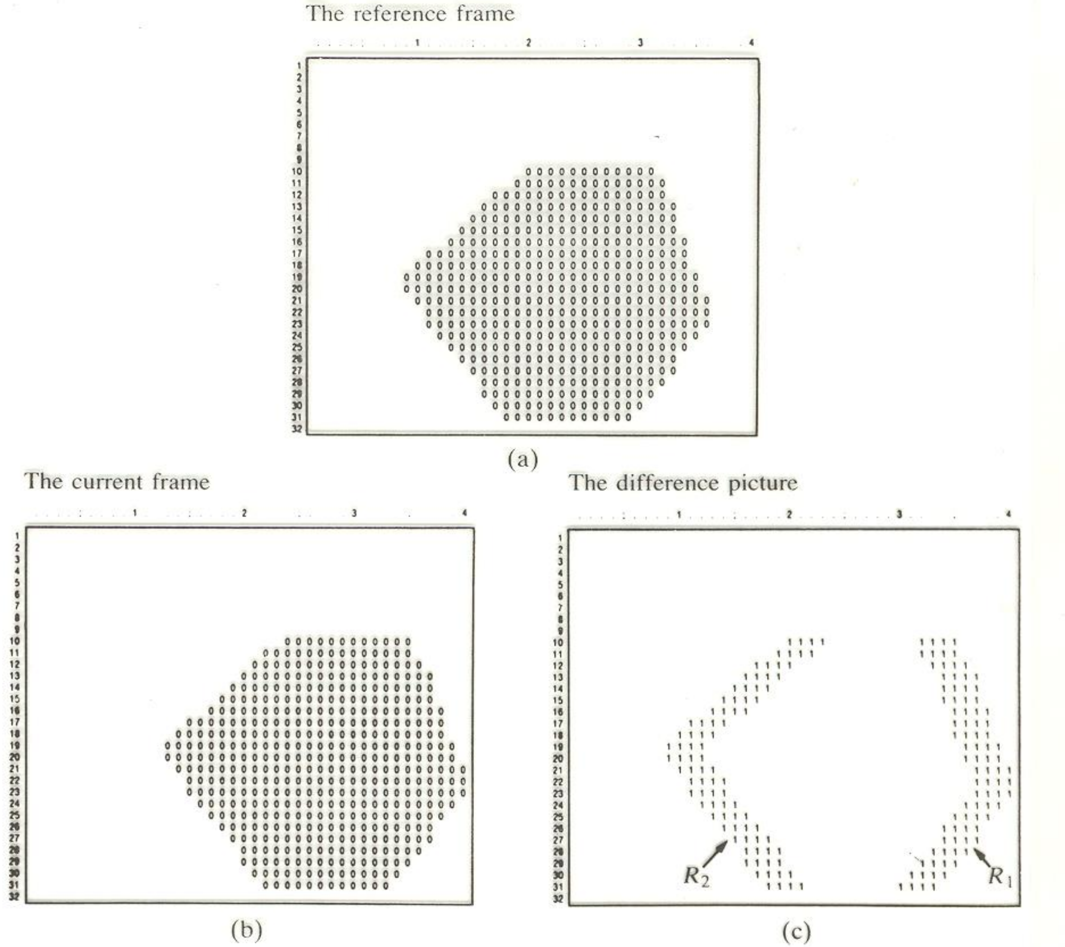
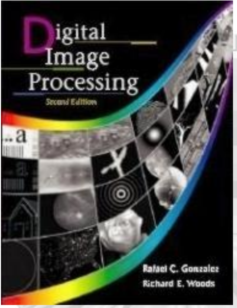
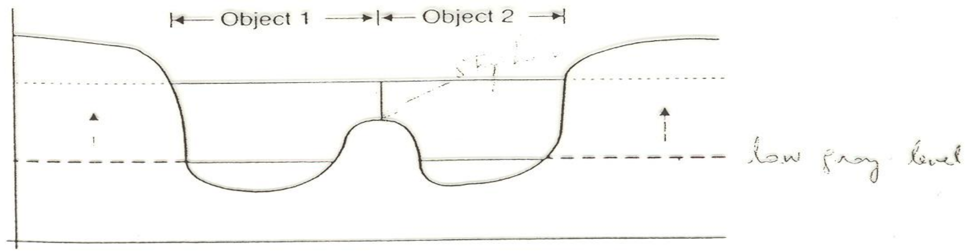


Figure 7.40 *(a) Image taken at time t_i; (b) image taken at time t_j; (c) difference image. (From Jain [1981].)*



The watershed algorithm

A relative of adaptive thresholding is the watershed algorithm. We assume that the objects in the image are of low gray level, on a high gray level background. The figure below shows the gray levels along one scan line that cuts through two objects that are close together.



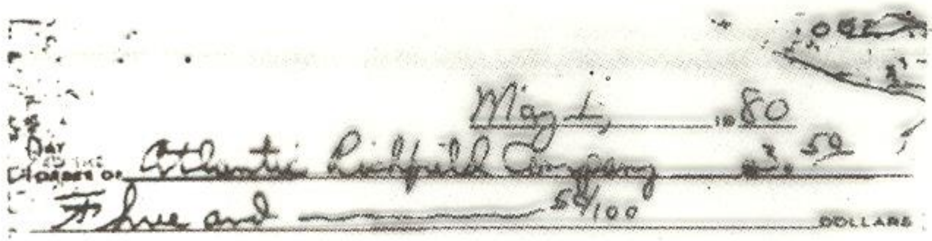
The image is initially thresholded at a low gray level, one that segments the image into the proper number of objects, but with boundaries that are too small. Then the threshold is raised gradually, one gray level at a time. The objects' boundaries will expand as the threshold increases. When they touch, however, the objects are not allowed to merge. Thus, these **points of first contact become the final boundaries** between adjacent objects. The process is terminated before the threshold reaches that gray level of the background - that is, at the point when the boundaries of well-isolated objects are set.

Both the initial and final threshold gray levels must be well chosen. If the initial threshold is too low, then low-contrast objects will be missed at first and then merged with nearby objects as the threshold increases. If the initial threshold is too high, objects will be merged from the start. The final threshold value determines how well the final boundaries fit the objects.

In some implementations of the watershed algorithm both the initial and final threshold gray levels are chosen **locally** (based, for example, on the local properties of an image) - and can vary from object to object.

igure 7.34 Segmentation by the multivariable histogram approach.

(a)

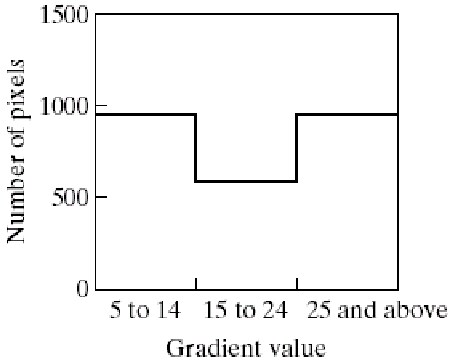


(b)

Figure 7.32 *(a) Original image; (b) segmented image. (From White and Rohrer [1983].)*

Example of Image Segmentation With thresholding

FIGURE 10.38 Histogram of pixels with gradients greater than 5. (Courtesy of IBM Corporation.)



(p.469 Castelman, "Digital Image Processing")

Region growing [24–27] is an approach to image segmentation that has received considerable attention in the computer vision segment of the artificial intelligence community. With this approach, one begins by dividing an image into many tiny regions. These initial regions may be small neighborhoods or even single pixels. In each region, suitably defined properties that reflect membership in an object are computed. The properties that distinguish the pixels inside the different objects might include average gray level, texture, or color information. Thus, the first step assigns to each region a set of parameters whose values reflect the object to which they belong.

Next, all boundaries between adjacent regions are examined. A measure of boundary strength is computed utilizing the differences of the averaged properties of the adjacent regions. A given boundary is *strong* if the properties differ significantly on either side of that boundary, and it is *weak* if they do not. Strong boundaries are allowed to stand, while weak boundaries are dissolved and the adjacent regions merged.

The process is iterated by alternately recomputing the object membership properties for the enlarged regions and then dissolving weak boundaries. The region-merging process is continued until a point is reached where no boundaries are weak enough to be dissolved. Then, image segmentation is complete. Monitoring this procedure gives one the impression of regions in the interior of objects growing until their boundaries correspond with the edges of the object.

Region-growing algorithms are computationally more expensive than the simpler techniques, but region growing is able to utilize several image properties directly and simultaneously in determining the final boundary location. Perhaps it shows greatest promise in the segmentation of natural scenes, where strong a priori knowledge is not available.

Figure 18–19 shows four stages in the region growing of one muscle fiber viewed on a microscope slide. In this example, low gradient was the sole region membership property. The lower right quadrant shows the final boundary.

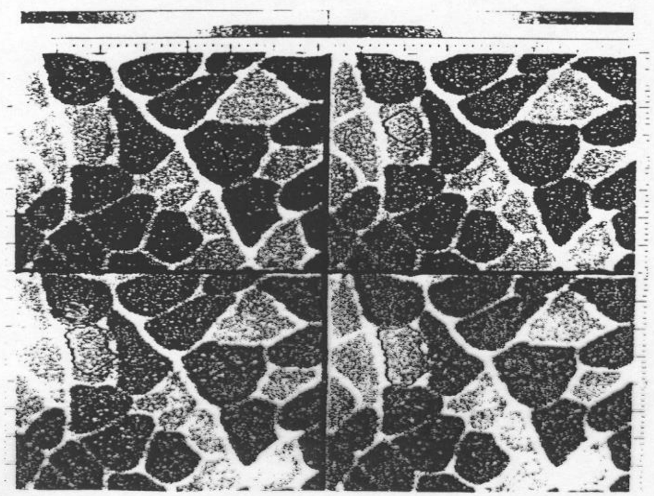
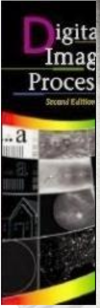


Figure 18–19 Region growing example

17



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

(a)

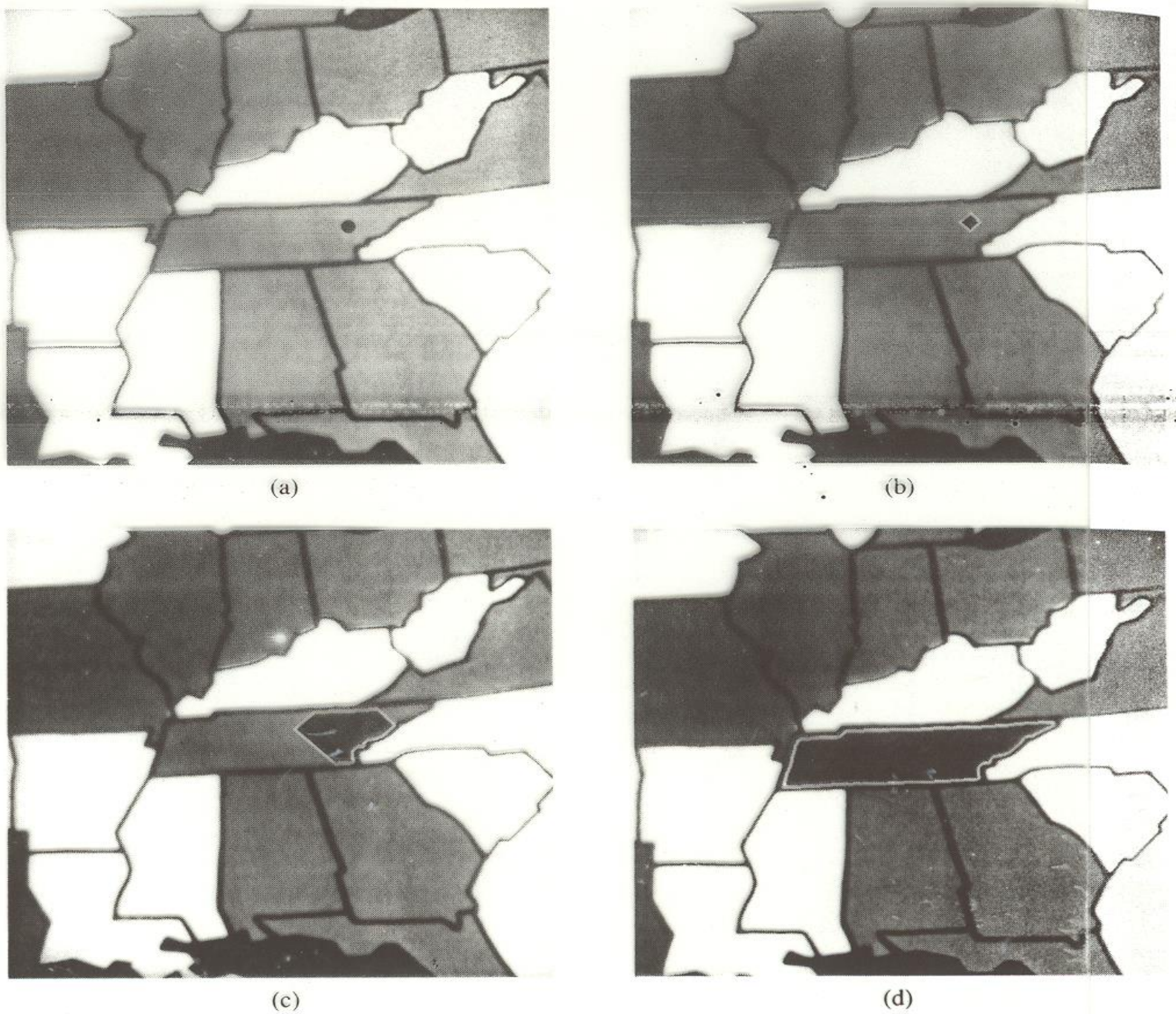
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b

(b)

a	a	a	a	a
a	a	a	a	a
a	a	a	a	a
a	a	a	a	a
a	a	a	a	a

(c)

Figure 7.35 *Example of region growing using known starting points: (a) original image array; (b) segmentation result using an absolute difference of less than 3 between intensity levels; (c) result using an absolute difference of less than 8.*



(a) (b) (c) (d)

Figure 7.36 *(a) Original image showing seed point; (b) early stage of region growth; (c) intermediate stage of growth; (d) final region.*

REGION REPRESENTATION

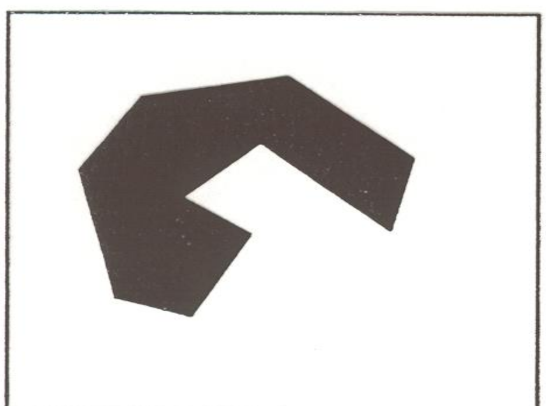
- array representations
- hierarchical representations
- region characteristic-based representations

Array representations

- array with numbered regions:

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

- binary mask



Pyramids

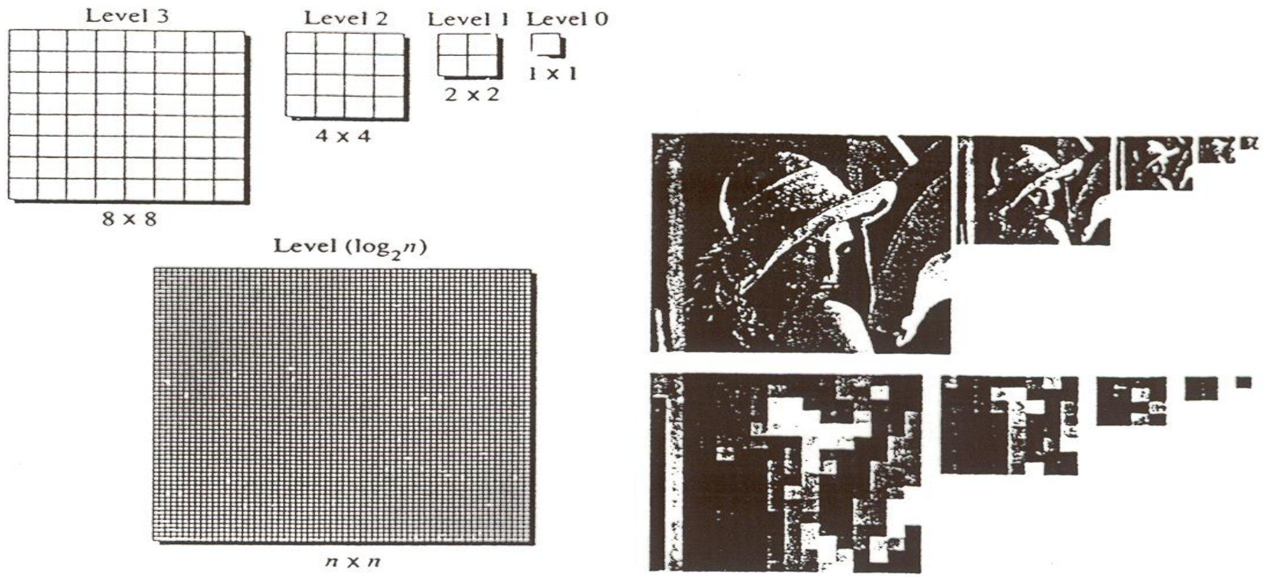
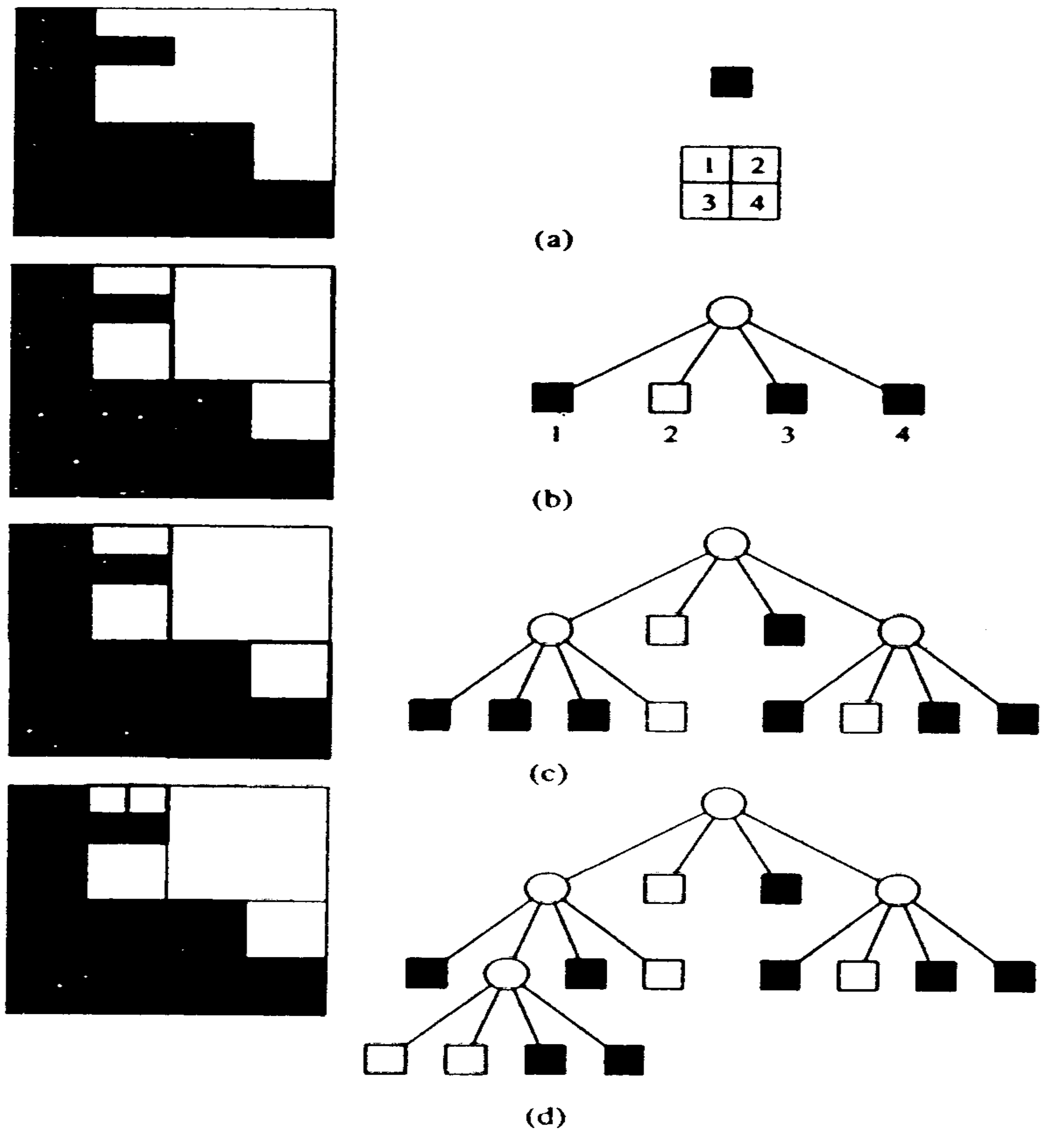


Figure 3.9: The original image is a 512 × 512 image; its reduced-resolution versions are successively obtained by averaging four points. All successive versions are shown here. Note that the low resolution images have been enlarged for display.

Quad Trees

- contains three types of nodes: white, black, and gray
- is obtained by recursive splitting of an image
- each node in this structure is either a leaf node or has four children - thus the name *quad tree*
- it is a form of data compression
- use: in spatial databases, in applications that require reducing the storage space for images
- several algorithms for computing pictorial properties from quad trees have been developed



Used in compression

Figure 3.10: The building of a quad tree. (a) Original image, "gray region." (b) Original split into four subregions (the left node in the tree corresponds to the top left region in the image). Note that two of these regions are also gray regions. (c) Splitting the gray regions from (b) into four subregions. One of

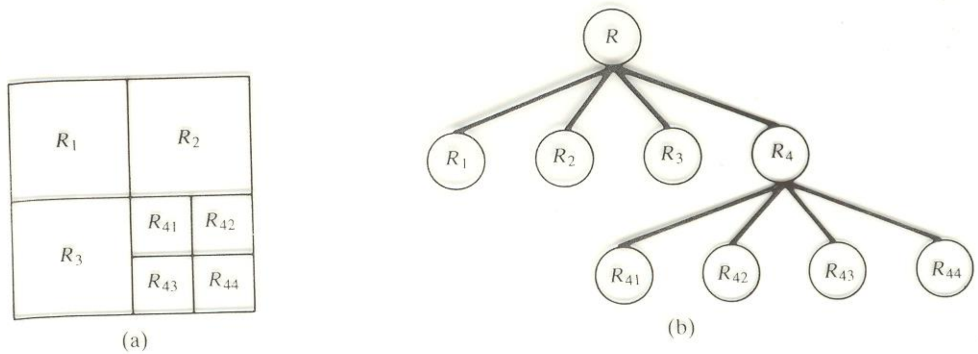


Figure 7.37 (a) Partitioned image; (b) corresponding quadtree.

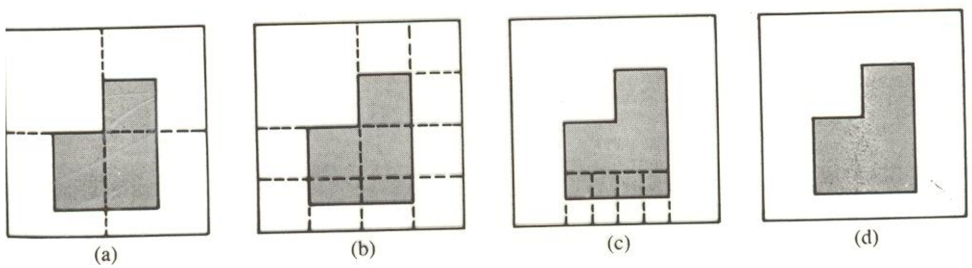


Figure 7.38 Example of split-and-merge algorithm. (From Fu, Gonzalez, and Lee [1987].)

SPLIT AND MERGE

- a simple intensity-based segmentation usually results in too many regions (noise, gradual transitions between gray values in different regions)
- after the initial intensity-based region segmentation, the regions may need to be refined or reformed
- refinement is done using a combination of split and merge operations
- split and merge operations eliminate false boundaries and spurious regions
- **merging** - combines regions that are considered similar

Algorithm 3.5 Region Merging

1. *Form initial regions in the image using thresholding (or a similar approach) followed by component labeling.*
2. *Prepare a region adjacency graph (RAG) for the image.*
3. *For each region in an image, perform the following steps:*
   (a) *Consider its adjacent region and test to see if they are similar.*
   (b) *For regions that are similar, merge them and modify the RAG.*
4. *Repeat step 3 until no regions are merged.*

- **splitting** - dividing a region into two parts

Algorithm 3.6 Region Splitting

1. *Form initial regions in the image.*
2. *For each region in an image, recursively perform the following steps:*
   (a) *Compute the variance in the gray value for the region.*
   (b) *If the variance is above a threshold, split the region along the appropriate boundary.*

- **merge and split** - a succession of splits and merges; useful for segmenting complex scenes

Suppose that an image is partitioned into a set of regions, $\{R_k\}$, for $k = 1, 2, \ldots, m$. All of the pixels in a region will be homogeneous according to some property defined by a predicate P $\cancel{H}$ applied to the region. The predicate represents the similarity between the pixels in a region. For example, the predicate could be defined using the variance in gray values within a region:

$$\text{P}\ \cancel{H}(R) = \begin{cases} 1 & \text{if the variance is small} \\ 0 & \text{otherwise.} \end{cases} \tag{3.11}$$

Algorithm 3.7 Split and Merge Region Segmentation

1. *Start with the entire image as a single region.*
2. *Pick a region R. If* P $\cancel{H}(R)$ *is false, then split the region into four subregions.*
3. *Consider any two or more neighboring subregions, $R_1, R_2, \ldots, R_n$, in the image. If* P $\cancel{H}(R_1 \cup R_2 \cup \cdots \cup R_n)$ *is true, merge the n regions into a single region.*
4. *Repeat these steps until no further splits or merges take place.*

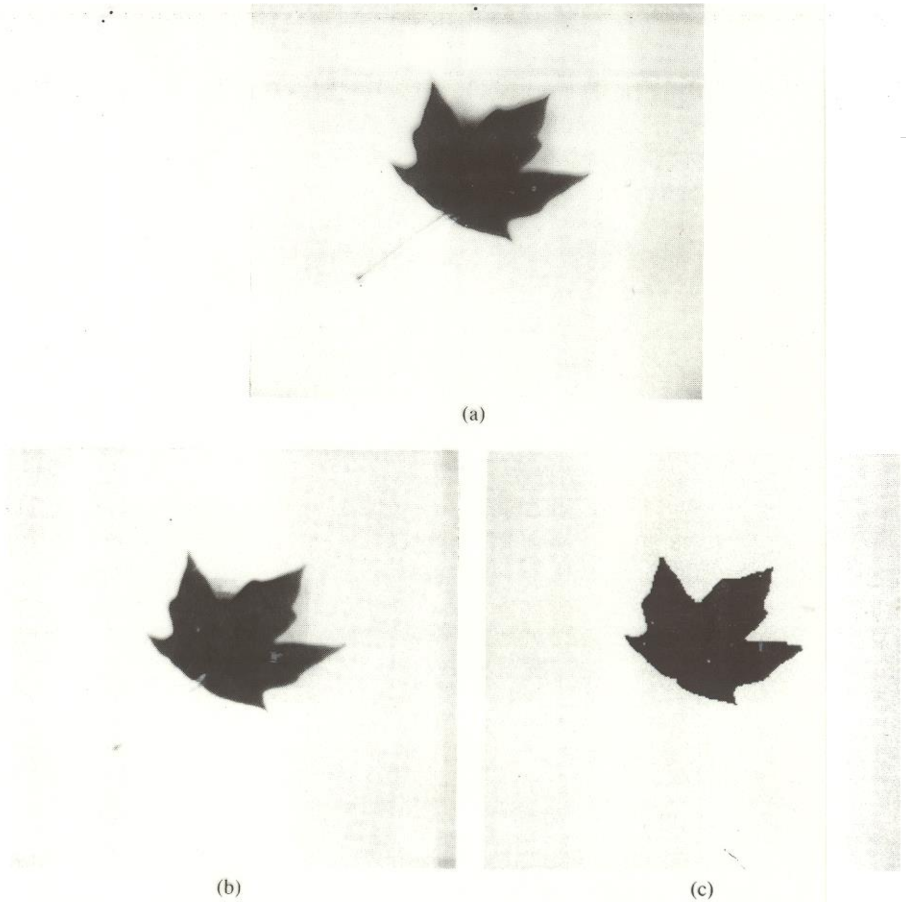


Figure 7.39 *(a) Original image; (b) result of split and merge procedure; (c) result of th olding Fig. 7.39(b).*

5.1.2 *Towards good segmentation*

Segmentation is a critical component of a computer vision system because errors in this process will be propagated to the higher level analysis processes and increase the complexity of the subsequent tasks. Ideally the segmented regions within an image should have the following characteristics [3]:

(a) regions should be uniform and homogeneous with respect to some particular characteristic;
(b) region interiors should be simple and without many small holes;
(c) adjacent regions should have significantly different values with respect to the characteristic on which they are uniform;
(d) boundaries of each segment should be simple, not ragged, and must be spatially accurate.

Most image segmentation techniques are *ad hoc* and domain-dependent. It is difficult to obtain quantitative data on the quality of segmentation as the results are open to subjective interpretation, the only simple criterion available being a measure of the percentage of pixels mis-classified. Good segmentation is somewhat analogous to the definition of 'real-time', the segmentation is good if it provides appropriate output for the solution of the problem under investigation.

Achieving all the above desired properties in practice is extremely difficult. Insisting that adjacent regions have large differences in value can cause adjacent regions to merge and boundaries to be lost. Over-dividing or under-dividing regions may cause them to correspond to more than one surface, or conversely surface variability may split a single real surface into several regions.

The problem of segmentation of natural images is basically one of emulating psychological perception and therefore does not lend itself to a purely analytical solution. Any mathematical algorithms must be supplemented by heuristics, usually involving semantics or descriptions about the class of images under consideration.

Sometimes it is appropriate to go beyond simple heuristics and introduce *a priori* knowledge about the image. In such cases image segmentation proceeds simultaneously with image understanding. In segmentation, *a priori* knowledge refers to implicit or explicit constraints on the likelihood of a given pixel grouping. Such assumptions often arise from restrictions placed on the image as a consequence of domain-dependent considerations.

Even in a back-lit binary image, for example, where segmentation into regions of background and foreground is a trivial task, it is still necessary to label holes that occur within objects. They have the same pixel intensity values as the general background, but have a quite different significance. Holes can be uniquely identified by observing the hypothesis that they consist of regions of background intensity which is entirely enclosed by foreground intensity.

Digital Image Processing

Second Edition

...ok.com

Region characteristic-based representations

- some characteristics for representing regions:

Perimeter - length of region's boundary

Area - number of pixels contained within its boundary

Shape factor - $\text{Perimeter}^2/\text{Area}$

Range - the difference between the largest and the smallest pixel values in a region

Median - median of pixels' intensity values within the region

Mean - average of pixels' intensity values within the region

$$\mu = \frac{\sum_{i=1}^{N} x_i}{N} = \bar{x}$$

Variance - the second moment about the mean

$$\sigma^2 = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N-1}$$

Standard deviation - square root of a variance

$$SD = \sqrt{\sigma^2}$$

Coef. of variation - a ratio of standard deviation to the mean

$$C = \frac{\sigma}{\mu}$$

Skewness - the third moment about the mean; an indicator of asymmetry in a pixel distribution

$$sk = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^3}{(N-1)\sigma^3}$$

Kurtosis - the fourth moment about the mean; relates to the degree of peakedness or flatness of a distribution (for the normal distribution kurtosis =3)

$$kurt = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^4}{(N-1)\sigma^4}$$

Labeling concepts

- Topics
- Thresholding
- Simple geometric features
  - area
  - position
  - orientation
- Projections
- Multiple objects
- Object boundary
- Component labeling
- Topological descriptors
- Boundary descriptors

BINARY IMAGE PROCESSING

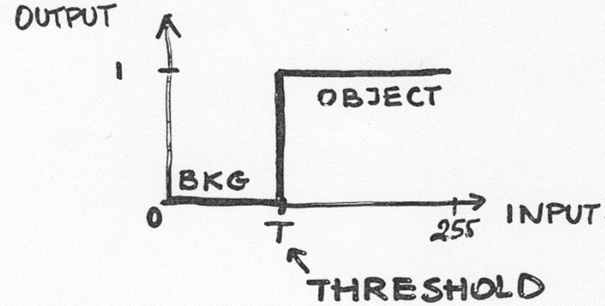
- binary image contains only two gray levels
- in the early days of machine vision binary images were used in many applications (memory and computing power was limited and expensive)
- vision systems based on binary algorithms are still used today (<u>less expensive</u> and <u>faster</u> than gray level systems)
- many techniques developed for binary vision systems are applicable to vision systems which use gray scale images (object in a gray level image can be represented as a <u>mask</u>)
- binary vision systems are useful in cases where the objects are easily separated from the background
  - character recognition
  - chromosome analysis
  - recognition of industrial parts

$\boxed{\text{GRAY SCALE IMAGE}} \rightarrow \text{THRESHOLDING} \rightarrow \boxed{\text{BINARY IMAGE}}$

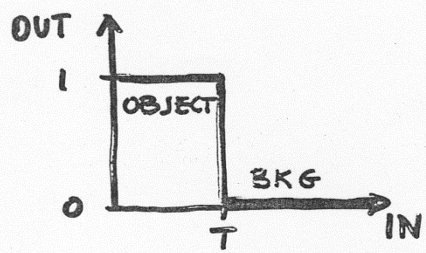
$F(i,j)$ (THRESHOLD T) $B[i,j] = F_T(i,j)$

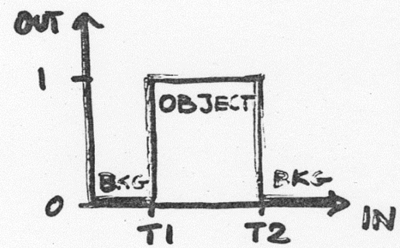
THRESHOLDING

- a method of converting a gray scale image into a binary image so that objects of interest are separated from the background



$$F_T[i,j] = \begin{cases} 1 & \text{if } F[i,j] \geq T \\ 0 & \text{otherwise} \end{cases}$$





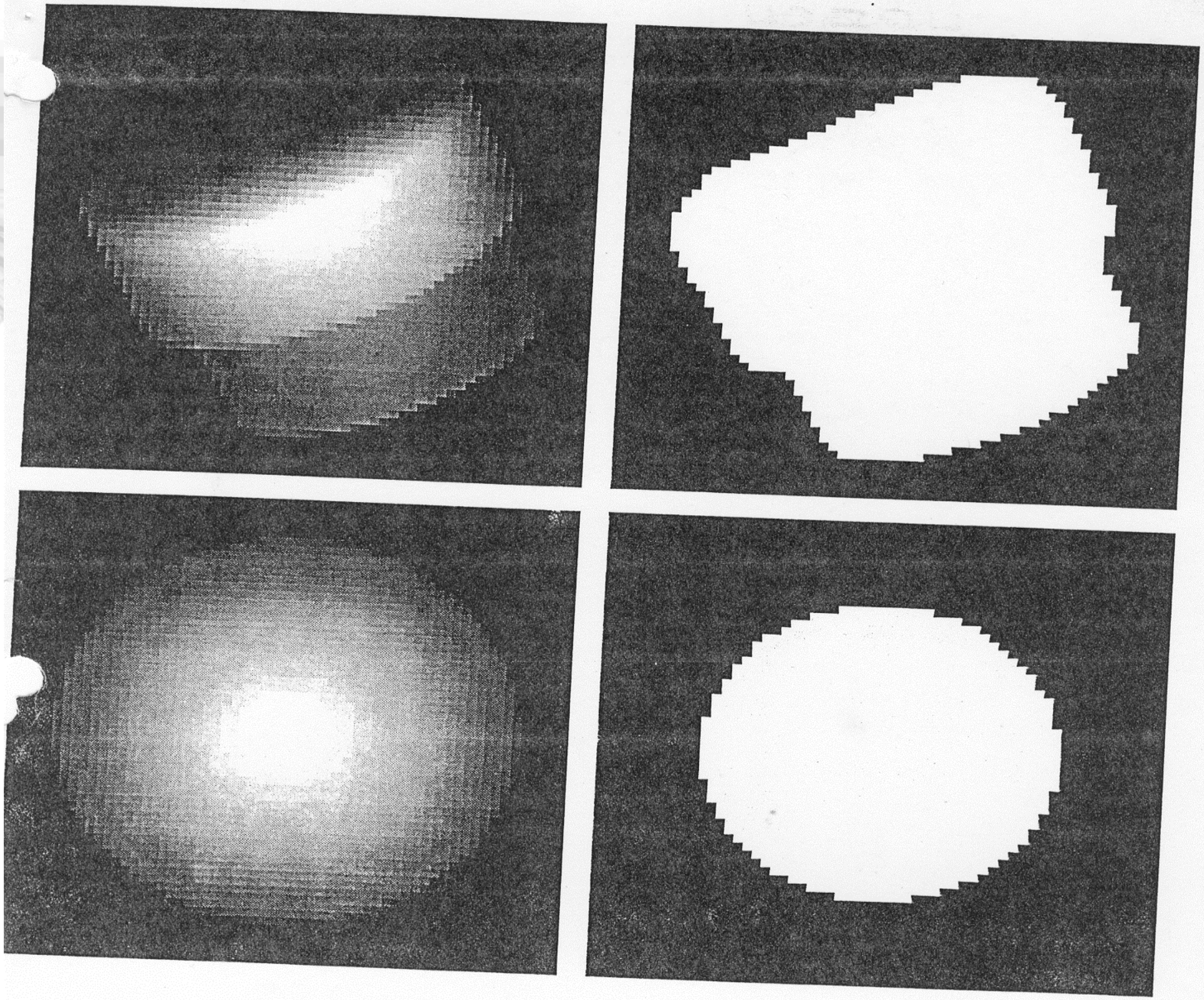
$$F_T[i,j] = \begin{cases} 1 & \text{if } F[i,j] \leq T \\ 0 & \text{otherwise} \end{cases}$$

$$F_T[i,j] = \begin{cases} 1 & \text{if } T_1 \leq F[i,j] \leq T_2 \\ 0 & \text{otherwise} \end{cases}$$

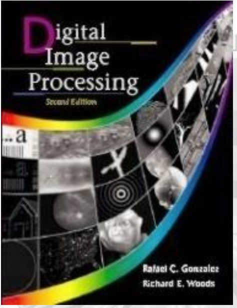
GENERAL THRESHOLDING SCHEME:

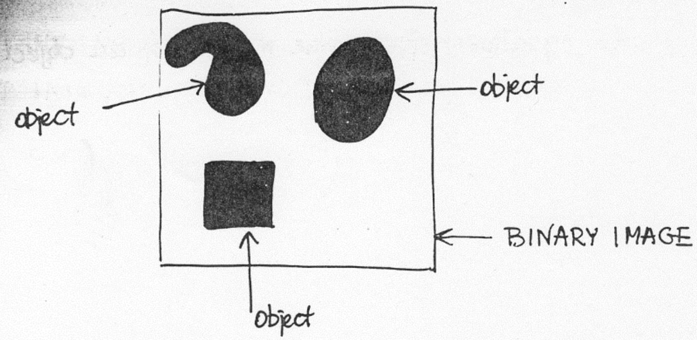
$$F_T[i,j] = \begin{cases} 1 & \text{if } F[i,j] \in Z \\ 0 & \text{otherwise} \end{cases}$$

where Z is a set of intensity values for object components



- the threshold is usually selected on the basis of experience with the application domain
- histogram is helpful in selecting a threshold
- automatic thresholding → will be discussed later...



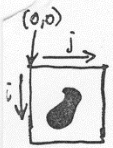


How to recognize and locate objects ?

↓

using simple geometric features:

size
position
orientation



- LET US ASSUME THAT AN IMAGE HAS ONLY ONE OBJECT
- THE AREA A OF THE OBJECT ON THE IMAGE B[i,j] :

binary

$$A = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} B[i,j]$$

object's size

where n is the number of rows
and m is the number of columns

For simplicity we will write:

$$A = \sum_{i} \sum_{j} B[i,j]$$

and we will understand that the summation was done for all i's and all j's

- A is the zeroth moment of B[i,j]

- The basic equation defining the moment of an object is given as:

$$M_{kl} = \sum_i \sum_j i^k j^l \, B[i,j]$$

where the order of the moment is $k+l$,
i, j are the pixel coordinates relative to some arbitrary standard origin

- Zero, and first-order moments can be defined as:

$$M_{00} = \sum_i \sum_j B[i,j]$$

$$M_{10} = \sum_i \sum_j i \, B[i,j]$$

$$M_{01} = \sum_i \sum_j j \, B[i,j]$$

- The centroid (center of area, center of mass) is a good parameter for specifying the location of an object

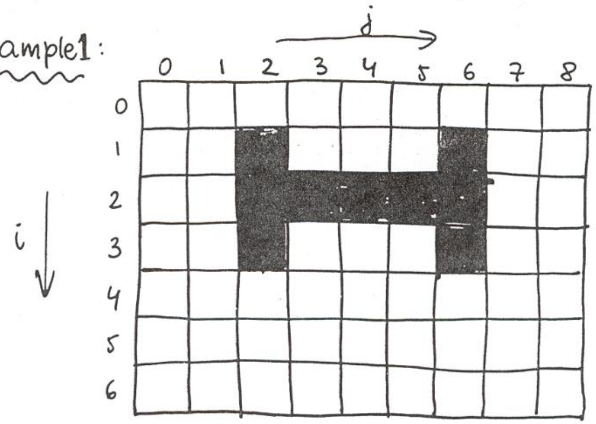
CENTROID IS THE POINT $(\bar{x}, \bar{y})$ SUCH THAT THE SUM OF THE SQUARE OF THE DISTANCE FROM IT TO ALL OTHER POINTS WITHIN THE OBJECT IS A MINIMUM. IT CAN ALSO BE EXPRESSED IN TERMS OF MOMENTS:

$$\boxed{\bar{x} = \frac{M_{10}}{M_{00}}} \qquad \boxed{\bar{y} = \frac{M_{01}}{M_{00}}} \quad \Big\} \text{CENTROID}$$

$$\boxed{\bar{x} = \frac{\sum_i \sum_j i \, B[i,j]}{A}} \qquad \boxed{\bar{y} = \frac{\sum_i \sum_j j \, B[i,j]}{A}}$$

- THE VALUES OF $\bar{x}$ AND $\bar{y}$ ARE NOT NECESSARY INTEGER NUMBERS!

- Example 1:



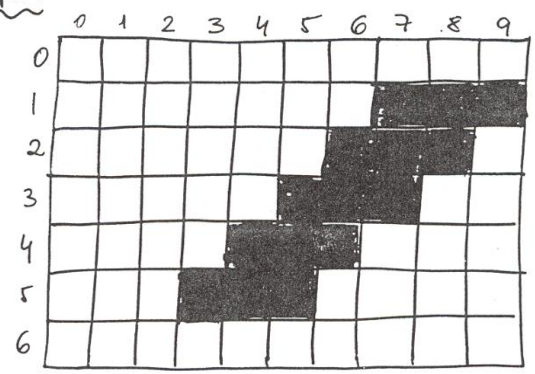
Calculate $\bar{x}$ and $\bar{y}$ for the object.

$$\bar{x} = \frac{M_{10}}{M_{00}} = \frac{\sum_i \sum_j i \cdot B[i,j]}{\sum_i \sum_j B[i,j]} = \frac{1 \cdot 2 + 2 \cdot 5 + 3 \cdot 2}{9} = \frac{18}{9} = 2$$

$$\bar{y} = \frac{M_{01}}{M_{00}} = \frac{\sum_i \sum_j j \cdot B[i,j]}{\sum_i \sum_j B[i,j]} = \frac{2 \cdot 3 + 3 \cdot 1 + 4 \cdot 1 + 5 \cdot 1 + 6 \cdot 3}{9} = \frac{6+3+4+5+18}{9} = \frac{36}{9} = 4$$

$\bar{x} = 2$, $\bar{y} = 4$

- Example 2:



Calculate $\bar{x}$ and $\bar{y}$ for the object.

$$\bar{x} = \frac{M_{10}}{M_{00}} = \frac{1 \cdot 3 + 2 \cdot 3 + 3 \cdot 3 + 4 \cdot 3 + 5 \cdot 3}{15} = \frac{3+6+9+12+15}{15} = \frac{45}{3} = 3$$

$$\bar{y} = \frac{M_{01}}{M_{00}} = \frac{3\times1 + 4\times2 + 5\times3 + 6\times3 + 7\times3 + 8\times2 + 9 \cdot 1}{15} = \frac{90}{15} = 6$$

- SECOND-ORDER MOMENTS CORRESPOND TO THE MOMENTS OF INERTIA → they are limited in their usefulness since they vary according to their position with respect to the origin and the scale and orientation of the object
- We can calculate CENTRAL MOMENTS – moments with respect to the centroid $(\bar{x}, \bar{y})$:

$$\mu_{kl} = \sum_i \sum_j (i-\bar{x})^k (j-\bar{y})^l B[i,j]$$

For example:

$$\mu_{20} = \sum_i \sum_j (i-\bar{x})^2 B[i,j]$$

$$\mu_{02} = \sum_i \sum_j (j-\bar{y})^2 B[i,j]$$

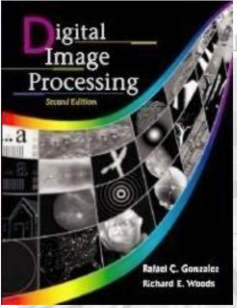
$$\mu_{11} = \sum_i \sum_j (i-\bar{x})(j-\bar{y}) B[i,j] \quad , \text{etc.}$$

- ORIENTATION OF THE OBJECT MAY BE DEFINED AS THE ANGLE OF THE AXIS OF THE MINIMISED MOMENT OF INERTIA.

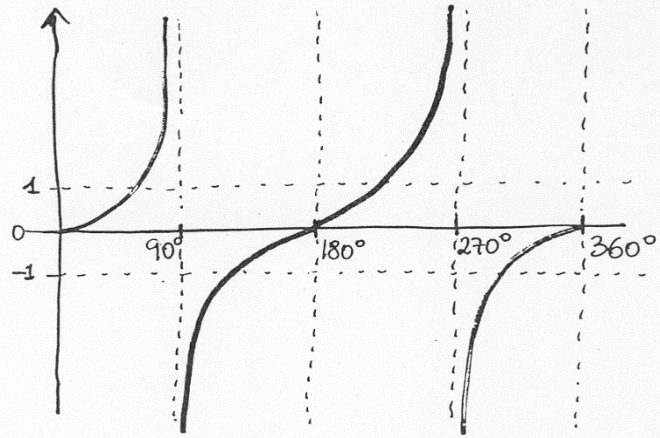
Orientation can be expressed in terms of the second-order central moments as:

$$\boxed{\tan(2\theta) = \frac{2\mu_{11}}{\mu_{20} - \mu_{02}}}$$

where θ is the orientation with respect to the x-axis



- Function $\tan\theta$ for θ from $0°$ to $360°$:



$\tan 0° = 0$; $\tan 30° = \frac{\sqrt{3}}{3}$; $\tan 45° = 1$; $\tan 60° = \sqrt{3}$;
$\tan 90° = \pm\infty$; $\tan 120° = -\sqrt{3}$; $\tan 135° = -1$; $\tan 150° = \frac{\sqrt{3}}{3}$;
$\tan 180° = 0$

- Calculate θ for Examples 1 and 2

For Example 2:

$\bar{x} = 3$, $\bar{y} = 6$

$$\mu_{11} = \sum_i \sum_j (i-3)(j-6)\, B[i,j] = -30$$

$$\mu_{20} = \sum_i \sum_j (i-3)^2 B[i,j] = 30$$

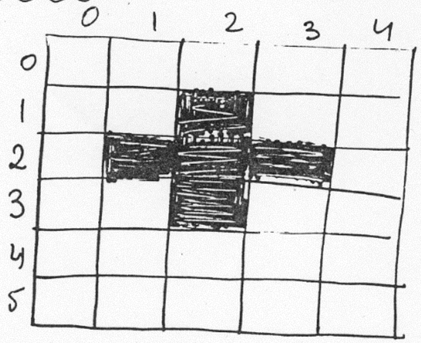
$$\mu_{02} = \sum_i \sum_j (j-6)^2 B[i,j] = 40$$

$$\tan(2\theta) = \frac{2 \cdot \mu_{11}}{\mu_{20} - \mu_{02}} = \frac{-60}{30-40} = \frac{-60}{-10} = 6$$

since: $\tan(80°30') = 5.976 \longrightarrow \theta \approx 40°15'$
(from tables)

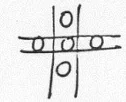
16

- Example 3:

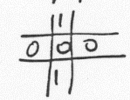


$\bar{x} = 2$
$\bar{y} = 2$

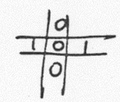
$$\mu_{11} = \sum_i \sum_j (i-2)(j-2)B[i,j] = 0$$



$$\mu_{20} = \sum_i \sum_j (i-2)^2 B[i,j] = 2$$



$$\mu_{02} = \sum_i \sum_j (j-2)^2 B[i,j] = 2$$



$$\tan(2\theta) = \frac{2 \cdot 0}{2-2} = ?$$

If $\mu_{11} = 0$ and $\mu_{20} = \mu_{02}$ the object does not have a unique axis of orientation.

In this case the object is too symmetric to allow us to define an axis in this way.

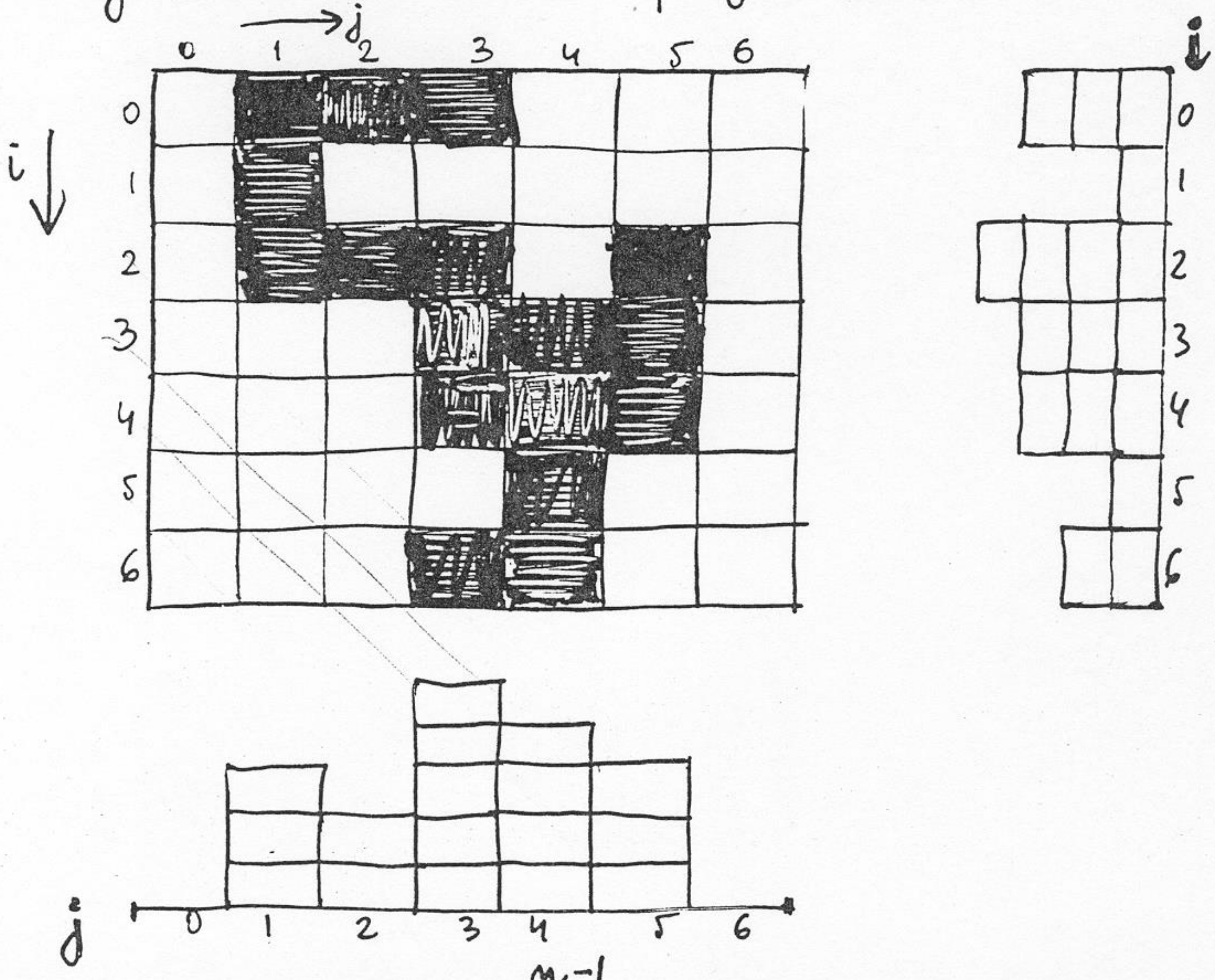
- Another moment-based shape feature:

ECCENTRICITY

$$\epsilon = \frac{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}{\mu_{00}}$$

PROJECTIONS

- Projections are compact representations of Images.
- Projections are not unique → more than one Image may have the same projections.



HORIZONTAL PROJECTION

$$H[i] = \sum_{j=0}^{m-1} B[i,j]$$

In Example 1:

j	0	1	2	3	4	5	6	7	8
$V[j]$	0	0	3	1	1	1	3	0	0

$A = 3+1+1+1+3 = 9$

i	0	1	2	3	4	5	6
$H[i]$	0	2	5	2	0	0	0

$A = 2+5+2 = 9$

$$\bar{y} = \frac{2\cdot 3 + 3\cdot 1 + 4\cdot 1 + 5\cdot 1 + 6\cdot 3}{9} = \frac{36}{9} = 4$$

$$\bar{x} = \frac{1\cdot 2 + 2\cdot 5 + 3\cdot 2}{9} = \frac{18}{9} = 2$$

- The second moments can be computed from the V[j], H[i] and diagonal projections

μ_{20} can be calculated from the horizontal projection $H[i]$:

Example 1:

i	0	1	2	3	4	5	6
$H[i]$	0	2	5	2	0	0	0
$(i-\bar{x})^2 \cdot H[i]$	0	$(1-\bar{x})^2 \cdot 2 = 2$	$(2-\bar{x})^2 \cdot 5 = 0$	$(3-\bar{x})^2 \cdot 2 = 2$	0	0	0

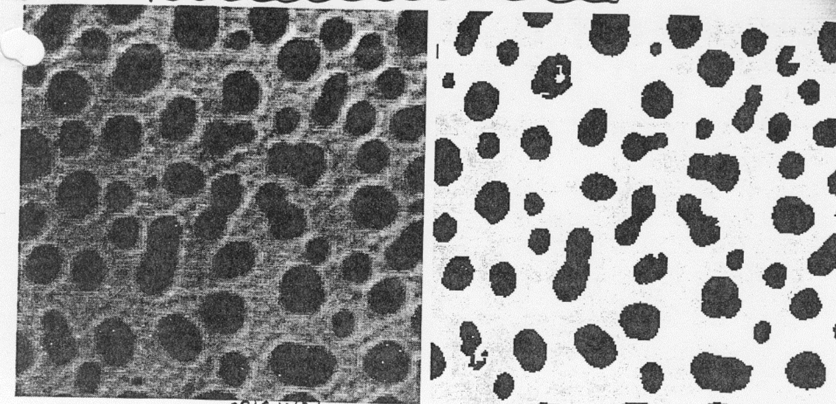
and $\bar{x} = 2$

$$\mu_{20} = \sum_{i=0}^{6} (i-\bar{x})^2 \cdot H[i] = 4$$

μ_{02} can be calculated from the vertical projection in a similar w[ay]

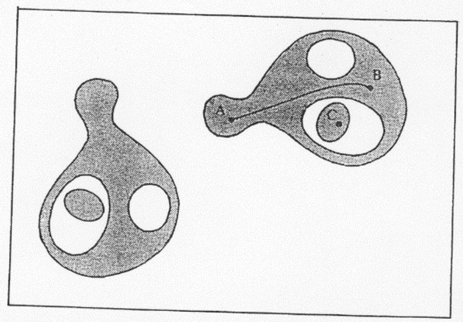
μ_{11} can be calculated from the diagonal projection

MULTIPLE OBJECTS

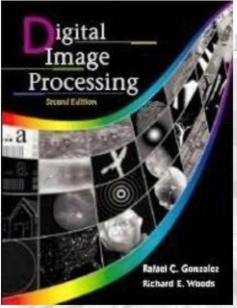


ORIGINAL THRESHOLDED

How to label the separate components on the thresholded image?



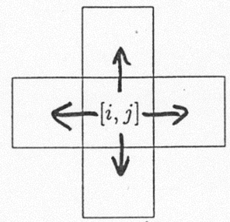
When several regions occur in an image, the computations of position and orientation have to be carried out separately for each. Picture cells must be labeled so that those belonging to a particular region can be distinguished from the rest. To do this we need to decide which points belong to the same region. Here A is considered to be connected to B because we can find a continuous curve connecting the two that lies entirely in the black region. Clearly A is not connected to C, because no such curve can be found.



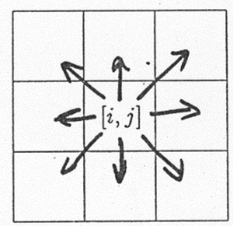
DEFINITIONS:

NEIGHBORS:

4-neighbors $[i+1, j]$, $[i-1, j]$, $[i, j+1]$, $[i, j-1]$



8-neighbors $[i+1, j+1]$, $[i+1, j-1]$, $[i-1, j+1]$, $[i-1, j-1]$ plus all of the 4-neighbors

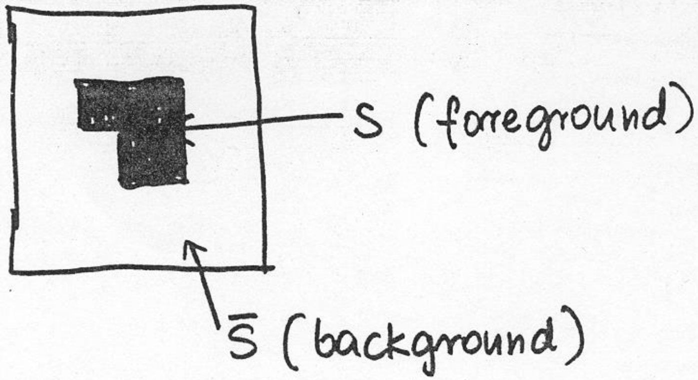


It turns out that neither of these choices is entirely satisfactory.
Consider a simple image:

0	1	0
1	0	1
0	1	0

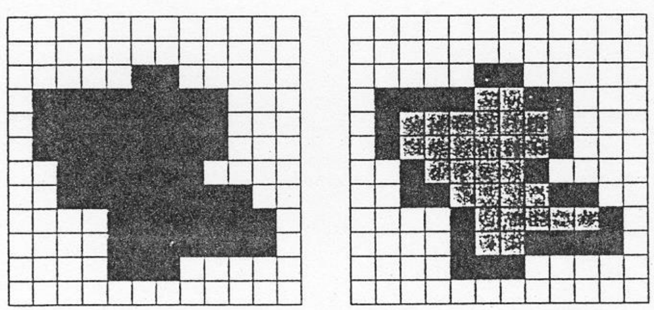
If we consider 4-connectedness for both background and foreground there are 4 objects that are 1 pixel in size and there is one hole.
If we consider 8-connectedness, then there is one object and no hole.

<u>To avoid this situation, different connectedness should be used for objects and backgrounds.</u>



Boundary

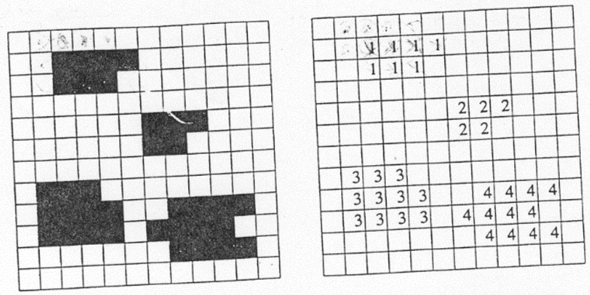
The boundary of S is the set of pixels of S that have 4-neighbors in $\overline{S}$. The boundary is usually denoted by S'



(a) Original image

(b) Boundary pixels
Interior pixels
Surrounds pixels

COMPONENT LABELING



- two algorithms : recursive and sequential

Algorithm 2.1 Recursive Connected Components Algorithm

1. *Scan the image to find an unlabeled 1 pixel and assign it a new label L.*
2. *Recursively assign a label L to all its 1 neighbors.*
3. *Stop if there are no more unlabeled 1 pixels.*
4. *Go to step 1.*

← VERY INEFFICIENT ON GENERAL-PURPOSE COMPUTERS

Algorithm 2.2 Sequential Connected Components Algorithm using 4-connectivity

1. *Scan the image left to right, top to bottom.*
2. *If the pixel is 1, then*
   (a) *If only one of its upper and left neighbors has a label, then copy the label.*
   (b) *If both have the same label, then copy the label.*
   (c) *If both have different labels, then copy the upper's label and enter the labels in the equivalence table as equivalent labels.*
   (d) *Otherwise assign a new label to this pixel and enter this label in the equivalence table.*
3. *If there are more pixels to consider, then go to step 2.*
4. *Find the lowest label for each equivalent set in the equivalence table.*
5. *Scan the picture. Replace each label by the lowest label in its equivalent set.*

← REQUIRES TWO PASSES OVER THE IMAGE

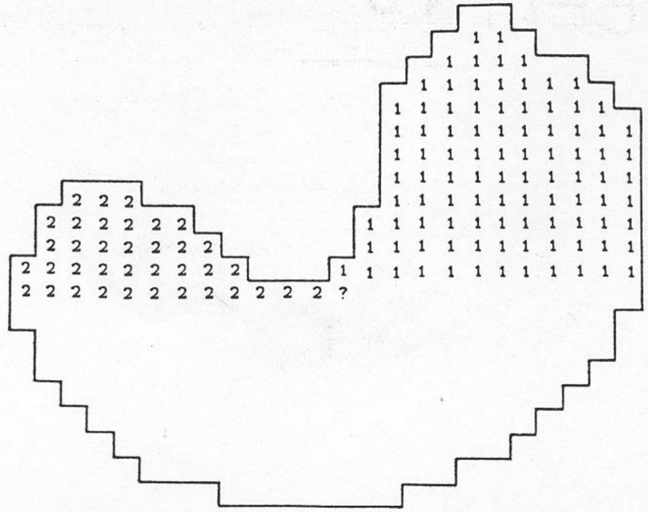


Figure 4-4. In the sequential labeling process we might discover that two regions previously thought to be separate are in fact connected. We must make a note of the equivalence of the two labels.

<u>NOTE:</u> Area, first and second-moments can be calculated for each component as part of the sequential connected components algorithm. Separate variables must be used to accumulate the moment information for each region.

TOPOLOGICAL DESCRIPTORS

TOPOLOGY – the study of the properties of an object (a figure) that are unaffected by any deformation except tearing or folding (invariant to translation, rotation and scaling)

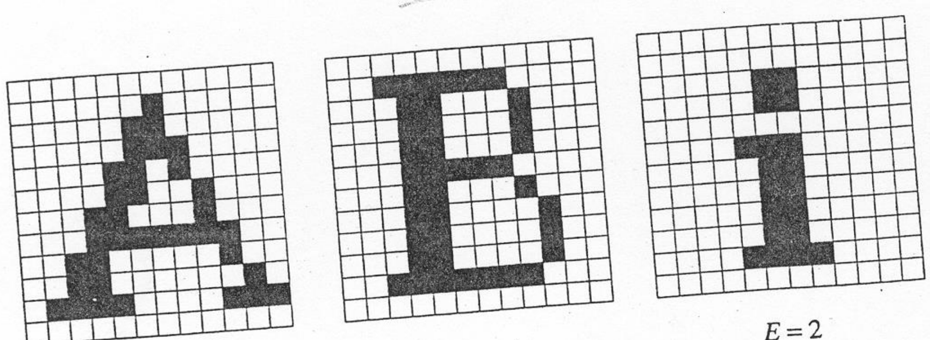
topological features give global description of an object

examples:

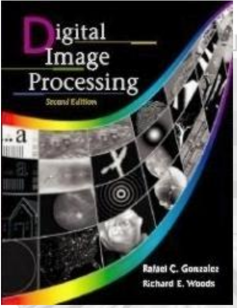
- number of holes (H)
- number of non connected components (C)
- number of edges
- number of vertices

Euler number (the difference between C and H):

$$E = C - H$$



$E=0$ $E=-1$ $E=2$



BOUNDARY DESCRIPTORS

- A simple local operation may be used to find pixels on the boundary, for example: locate the pixels that have value 1 and are next to pixels with value 0.
- To track pixels of a ~~region~~ boundary in a particular order you have to use boundary-following algorithms.
- Common approach is to track all pixels of an object in a clockwise sequence:

Algorithm 2.3 Boundary-Following Algorithm

1. *Find the starting pixel* $s \in S$ *for the region using a systematic scan, say from left to right and from top to bottom of the image.*
2. *Let the current pixel in boundary tracking be denoted by* c. *Set* $c = s$ *and let the 4-neighbor to the west of* s *be* $b \in \bar{S}$.
3. *Let the eight 8-neighbors of* c *starting with* b *in clockwise order be* $n_1, n_2, \ldots, n_8$. *Find* n_i, *for the first* i *that is in* S.
4. *Set* $c = n_i$ *and* $b = n_{i-1}$.
5. *Repeat steps 3 and 4 until* $c = s$.

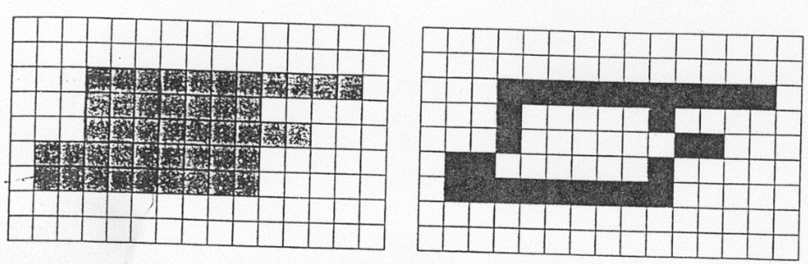


Figure 2.16: Results of a boundary-following algorithm. *Left:* Original binary object. *Right:* Calculated boundary.

boundary descriptors:

- AREA
- PERIMETER – number of boundary pixels
- COMPACTNESS

$$\frac{P^2}{A}$$

wher P – perimeter, A – area

→ The circle ~~tis~~ is the most compact figure. What is the ratio $\frac{P^2}{A}$ for a circle? $\frac{(2\pi r)^2}{\pi r^2}$ $\frac{4\pi^2 r^2}{\pi r^2}$ 4π

- BOUNDARY DIAMETER – maximum distance between two points on the boundary

How would you measure distance?

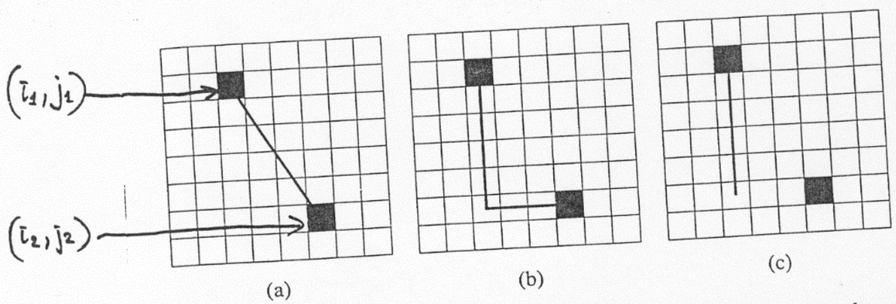
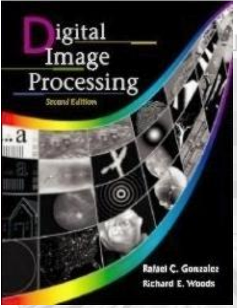


Figure 2.17: Examples of (a) Euclidean, (b) city-block, and (c) chessboard distance measures.

$$\boxed{d_{Euc} = \sqrt{(i_1-i_2)^2+(j_1-j_2)^2}}$$

$$\boxed{d_{city} = |i_1-i_2|+|j_1-j_2|}$$

$$\boxed{d_{chess} = \max(|i_1-i_2|, |j_1, j_2|}$$



The skeleton of a region

- **medial axis transformation** (MAT) generates a "skeleton" of an object
- it is used for compact representation of objects
- MAT algorithm:
  - (1) for each point in the object we find its closest point in boundary,
  - (2) if a point has more than one such a neighbor --> a point belongs to the medial axis (skeleton) of the object

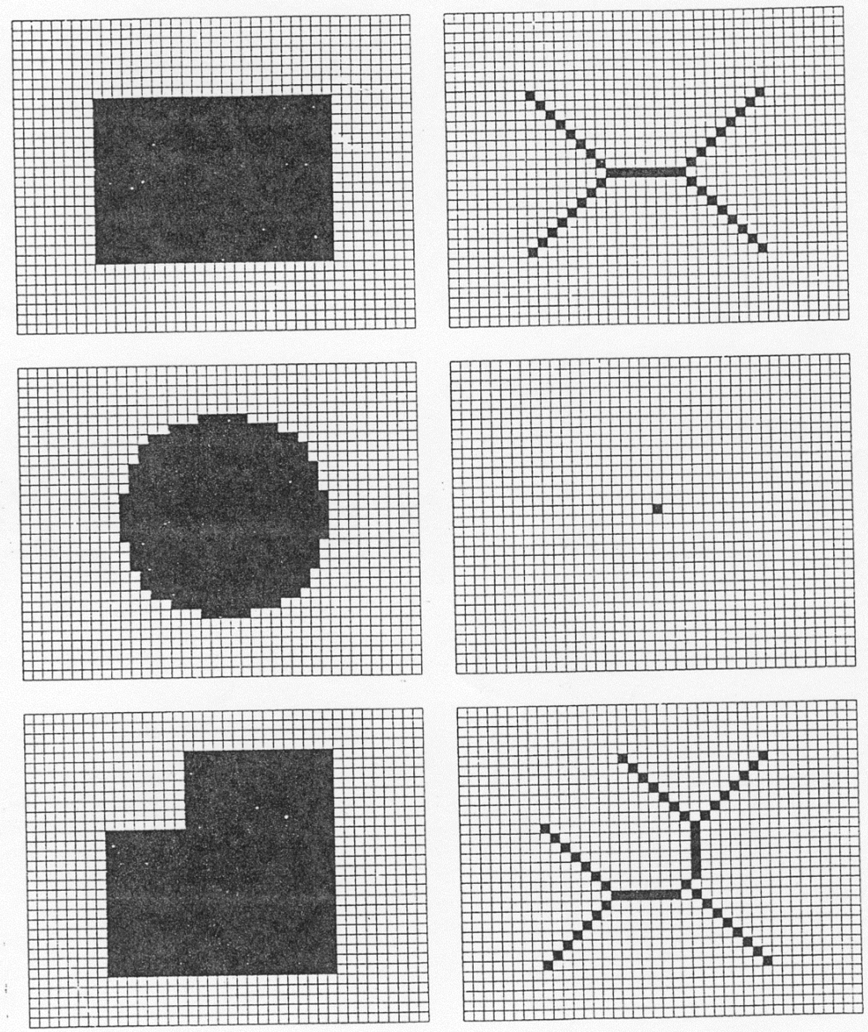


Figure 2.20: Examples of the medial axis transform.

MAT skeletonization

- Although Medial Axis Transform can lead generally to some good skeleton results, however have some disadvantages

-computationally complex

-other thinning algorithms are often preferred - very sensitive to noise

Alternative Thinning algorithm of inary images based on morphology

- Take notes