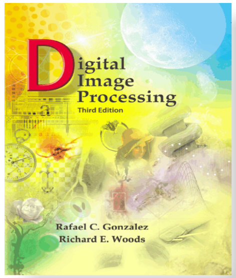
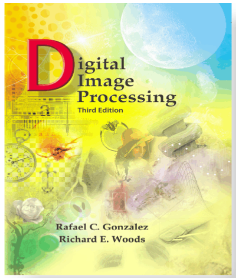


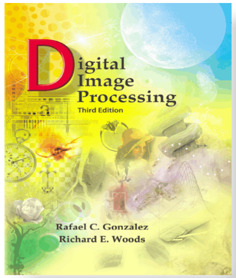
- Representation
  - The segmentation techniques yield data in the form of pixels along a boundary or pixels contained in a region.
  - It is standard practice to use schemes that compact the segmented data into representations that facilitate the computation of descriptors.



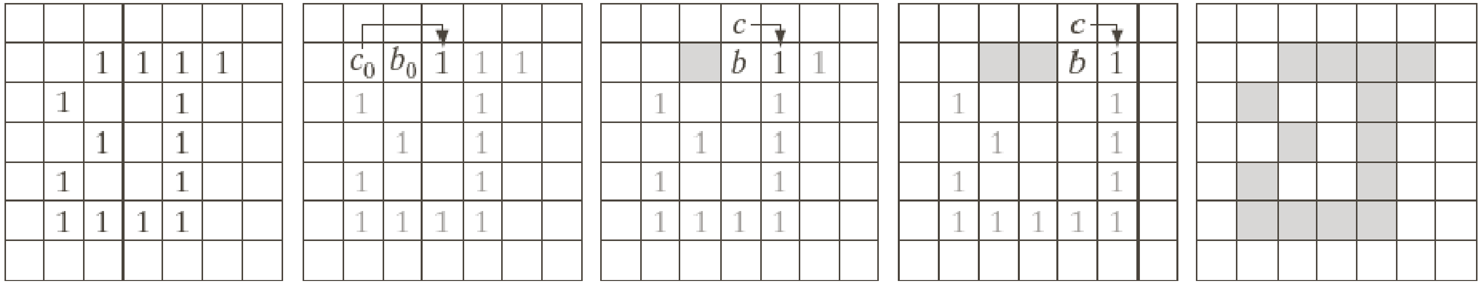
- Boundary Following:
  - We assume
    - (1) that we are working with binary images in which object and background points are labeled 1 and 0, respectively
    - (2) the images are padded with a border of 0s to eliminate the possibility of an object merging with the image border.

Given a binary region R or its boundary, an algorithm for following the border of R, or the given boundary, consists of the following steps:

1. Let the starting point, b_0, be the *uppermost, leftmost* point in the image that is labeled 1. Denote by c_0 the *west* neighbor of b_0. Clearley, c_0 always is a background point. Examine the 8-neighbors of b_0, starting at c_0 and proceeding in clockwise direction. Let b_1 denote the *first* neighbor encountered whose value is 1, and let c_1 be the (background) point immediately preceding b_1 in the sequence. Store the location of b_0 and b_1 for use in Step 5.



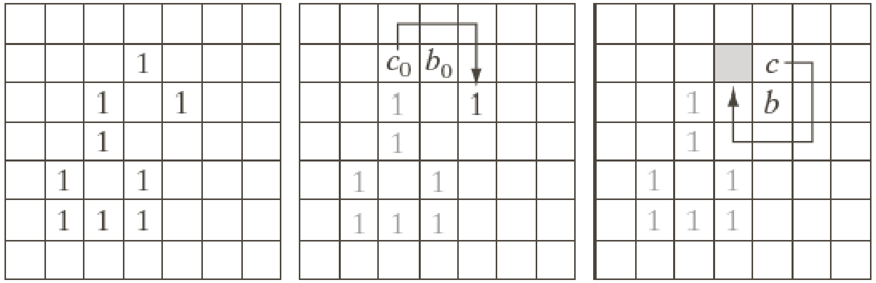
2. Let $b=b_1$ and $c=c_1$
3. Let the 8-neighbors of b, starting at c and proceeding in a clockwise direction, be denoted by $n_1, n_2, \ldots, n_8$. Find the first n_k labeled 1.
4. Let $b=n_k$ and $c=n_{k-1}$
5. Repeat Step 3 and 4 until $b=b_0$ and the boundary point found is b_1. The sequence of b points found when the algorithm stops constitutes the set of ordered boundary points.



a b c d e

FIGURE 11.1 Illustration of the first few steps in the boundary-following algorithm. The point to be processed next is labeled in black, the points yet to be processed are gray, and the points found by the algorithm are labeled as gray squares.

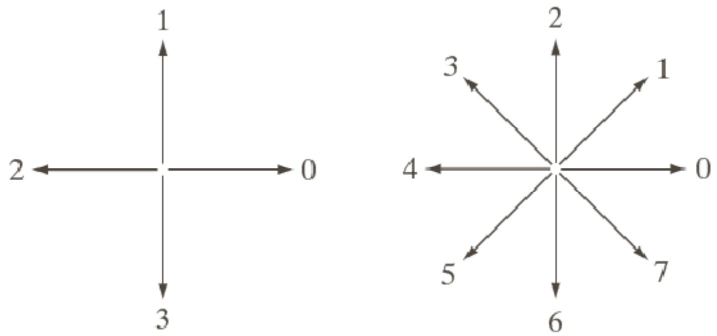
- The boundary-following algorithm works equally well if a region, rather than its boundary, is given in Fig.11.2
- That is, procedure extracts the *outer boundary* of a binary region. If the objective is to find the boundaries of holes in a region(these are called the *inner boundaries* of the region), a simple approach is to extract the holes, a simple approach is to extract and treat them as 1-valued region on a background of 0s.



a b c

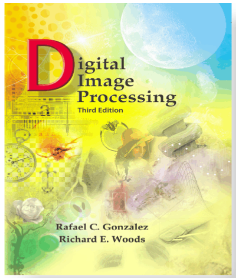
FIGURE 11.2 Illustration of an erroneous result when the stopping rule is such that boundary-following stops when the starting point, b_0, is encountered again.

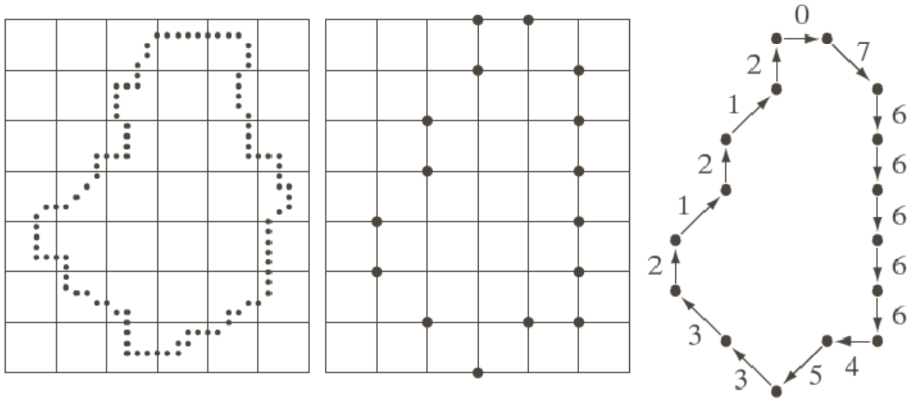
- Chain Codes:
  - Chain codes are used to represent a boundary by a connected sequence of straight-line segments of specified length and direction.
  - Typically, this representation is based on 4- or 8-connectivity of the segments. The direction of each segment is coded by using a numbering scheme, as in Fig.11.3



a b

FIGURE 11.3 Direction numbers for (a) 4-directional chain code, and (b) 8-directional chain code.

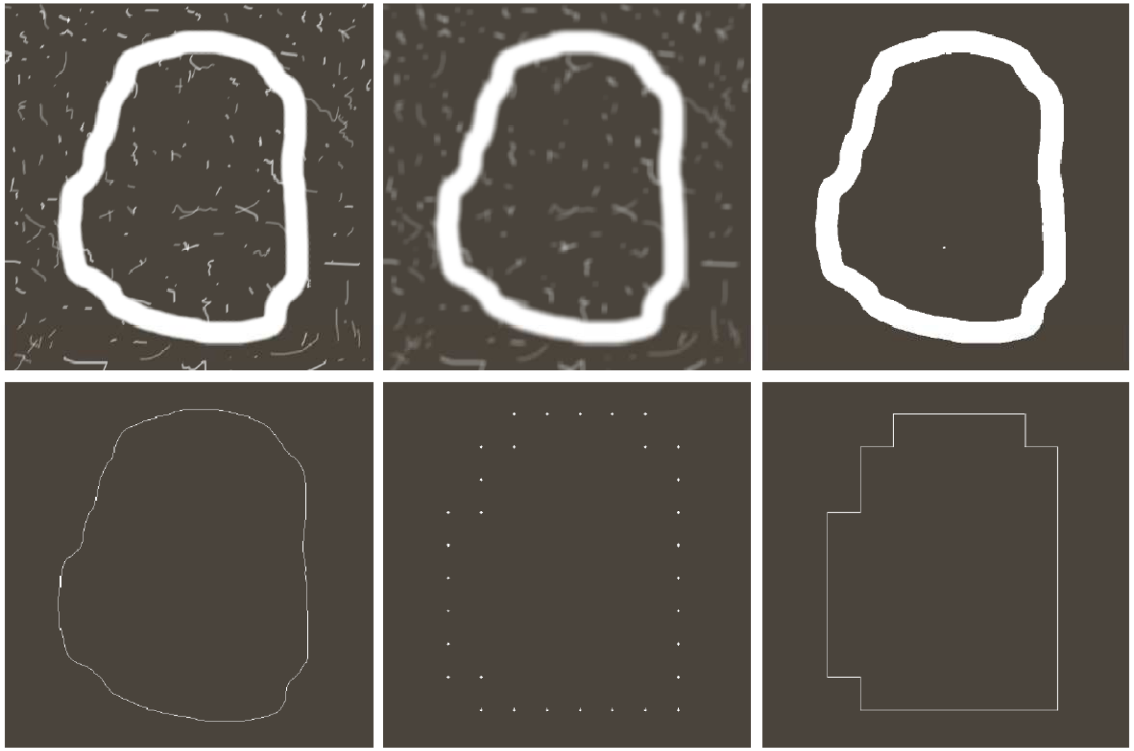




a b c

FIGURE 11.4 (a) Digital boundary with resampling grid superimposed. (b) Result of resampling. (c) 8-directional chain-coded boundary.

- The chain code of boundary depends on the starting point. However, the code can be normalized with respect to starting point by a straightforward procedure.
- We simply treat the chain code as a circular sequence of direction numbers and redefine the starting point so that the resulting sequence of numbers forms an integer of minimum magnitude.
- We can normalize also for rotation (in angles that are integer multiples of the directions in Fig. 11.3) by using the first difference of the chain code instead of the code itself.



a b c
d e f

FIGURE 11.5 (a) Noisy image. (b) Image smoothed with a 9 × 9 averaging mask. (c) Smoothed image, thresholded using Otsu's method. (d) Longest outer boundary of (c). (e) Subsampled boundary (the points are shown enlarged for clarity). (f) Connected points from (e).

Connected Components

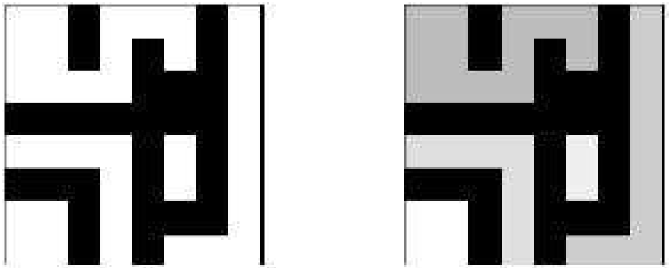
- Components are objects that share at least one common neighbor (in 4- or 8- neighborhood).
- Defn: A **connected component labeling** of binary image **B** is a labeled image **LB** in which the value of each pixel is the label of its connected component.

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

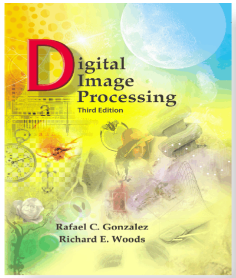
a) binary image

1	1	0	1	1	1	0	2
1	1	0	1	0	1	0	2
1	1	1	1	0	0	0	2
0	0	0	0	0	0	0	2
3	3	3	3	0	4	0	2
0	0	0	3	0	4	0	2
5	5	0	3	0	0	0	2
5	5	0	3	0	2	2	2

b) connected components labeling

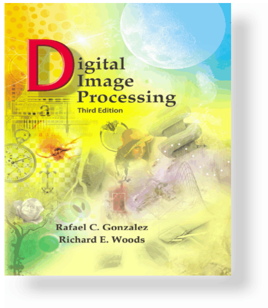


c) binary image and labeling, expanded for viewing



Region Properties

- Once a binary image has been processed we could obtain properties about the regions in the processed image.
- Some of those properties are
  - Area, centroid, perimeter, perimeter length, circularity of the region and second circularity measure



Area and Centroid

- Area A = sum of all the 1-pixels in the region.

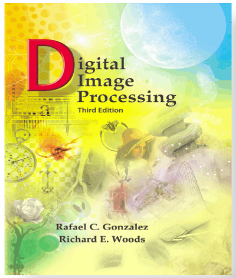
- Centroid [r’,c’] is the average location of the pixels in the region
  - r’ = 1/A*Sum of all the rows in the region
  - c’ = 1/A*Sum of all the columns in the region.

Perimeter

- A simple definiton of the perimeter of a region without holes i set of its interior border pixels.
- A pixel of a region is a border pixel if it has some neighboring pixel that is outside the region.
- When 8-connectivity is used to determine whether a pixel inside the region is connected to a pixel outside the region, the resulting set of perimeter pixel is 4-connected.
- When 4-connectivity is used to determine whether a pixel inside the region is connected to a pixel outside the region, the resulting set of perimeter pixel is 8-connected.

The Perimeter:

$$
\begin{aligned}
P_4 &= \{(r,c) \in R \mid N_8(r,c) - R \neq \emptyset\} \\
P_8 &= \{(r,c) \in R \mid N_4(r,c) - R \neq \emptyset\}
\end{aligned}
$$



The length of Perimeter

- To compute length |P| of perimeter P, the pixels in P must be ordered in a sequence $P=<(r_0,c_0),. . . , (r_{k-1},c_{k-1})>$, each pair of successive pixels in the sequence being neighbor, including the first and last pixels.

$$
\begin{aligned}
|P| &= |\{k|(r_{k+1},c_{k+1}) \in N_4(r_k,c_k)\}| \\
&+ \sqrt{2}|\{k|(r_{k+1},c_{k+1}) \in N_8(r_k,c_k) - N_4(r_k,c_k)\}|
\end{aligned}
$$

Circularity(1):

- With the area A and perimeter P defined, a common measure of the circularity of the region is the length of the perimeter squared divided by the area.

$$C_1 = \frac{|P|^2}{A}$$

circularity(2):

$$C_2 = \frac{\mu_R}{\sigma_R} \tag{3.12}$$

where μ_R and σ_R are the mean and standard deviation of the distance from the centroid of the shape to the shape boundary and can be computed according to the following formulas.

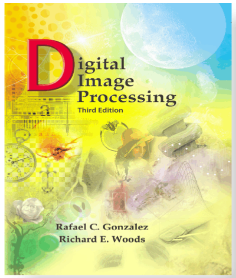
mean radial distance:

$$\mu_R = \frac{1}{K} \sum_{k=0}^{K-1} ||(r_k, c_k) - (\bar{r}, \bar{c})|| \tag{3.13}$$

standard deviation of radial distance:

$$\sigma_R = (\frac{1}{K} \sum_{k=0}^{K-1} [||(r_k, c_k) - (\bar{r}, \bar{c})|| - \mu_R]^2)^{1/2} \tag{3.14}$$

where the set of pixels (r_k, c_k), $k = 0, \ldots, K-1$ lie on the perimeter P of the region. The circularity measure C_2 increases monotonically as the digital shape becomes more circular and is similar for digital and continuous shapes.



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

labeled connected-components image

region num.	**region area**	**row of center**	**col of center**	**perim. length**	**circu-larity$_1$**	**circu-larity$_2$**	**radius mean**	**radius var.**
1	44	6	11.5	21.2	10.2	15.4	3.33	.05
2	48	9	1.5	28	16.3	2.5	3.80	2.28
3	9	13	7	8	7.1	5.8	1.2	0.04

properties of the three regions