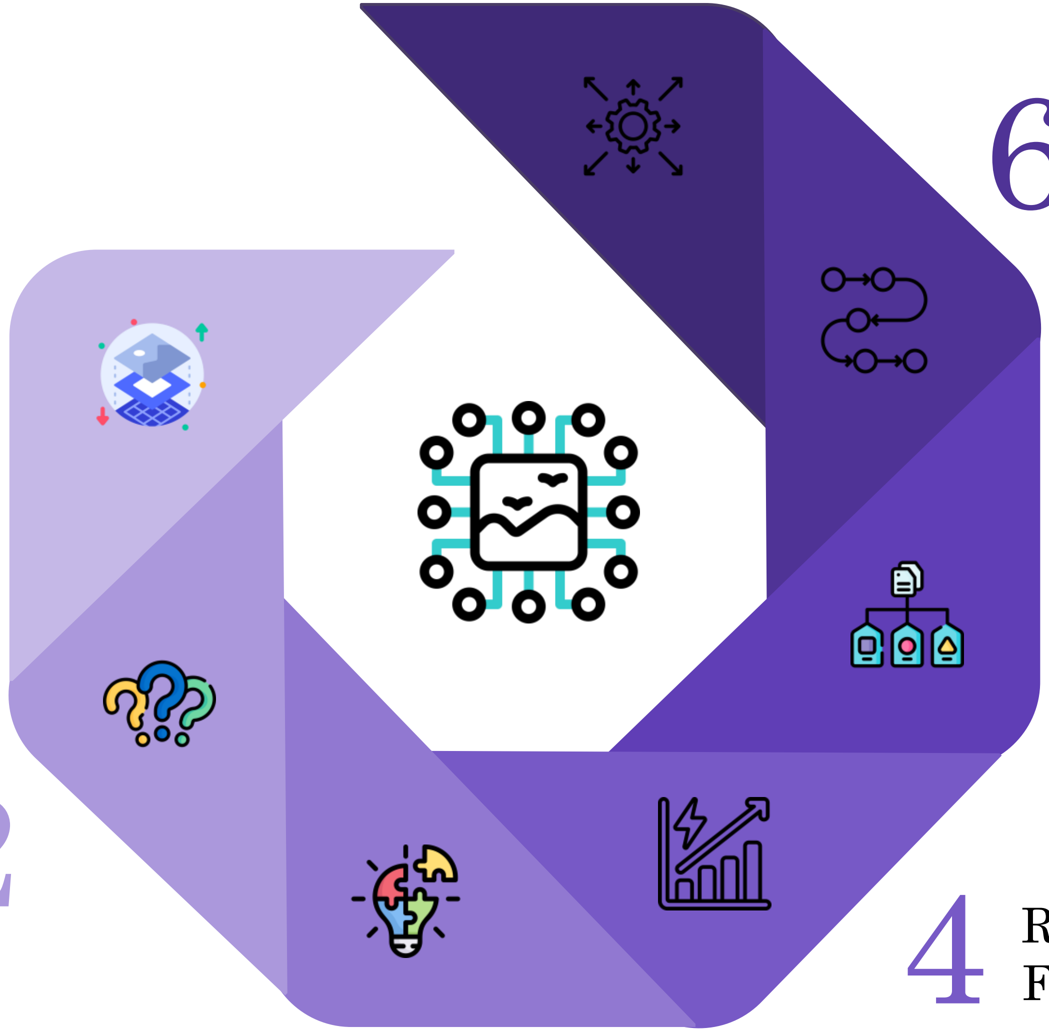
Digital Image Processing

Lecture 06

Basic Morphological Algorithms

Dr. Muhammad Sajjad
R.A: Asad Ullah

Overview



1. Hit or Miss Transform (HMT)
2. Basic Morphological Algorithms
3. Boundary Extraction
4. Region (hole) Filling
5. Thinning
6. Thickening
7. Gray Level Image Morphological Operations

Hit or Miss Transform (HMT)

- Erosion asks, "Does the SE *fit*?" Dilation asks, "Does the SE *hit*?" We need a new operation that asks, "Does the foreground *match* this pattern, AND does the background *simultaneously match* this other pattern?"
- This is the **Hit-or-Miss Transform**.
- The HMT utilizes two structuring elements: B1, for detecting shapes in the foreground, and B2, for detecting shapes in the background.
- The HMT of image **A** is defined as

$$\mathrm{A} \circledast \mathrm{B} = (A \ominus B_1) \cap (A^c \ominus B_2)$$

- Therefore, the hit-or-miss operation comprises three steps:
  1. Erode image A with structuring element B1.
  2. Erode the complement of image A (Ac) with structuring element B2.
  3. AND results from step 1 and step 2.

The structuring elements B1 and B2 can be combined into a single element B. Let's see an example:

0	1	0
1	0	1
0	1	0

0	0	0
0	1	0
0	0	0

0	1	0
1	-1	1
0	1	0

Structuring elements (kernels). Left: kernel to 'hit'. Middle: kernel to 'miss'. Right: final combined kernel

Hit or Miss Transform (HMT)

In this case, we are looking for a pattern in which the **central** pixel belongs to the **background** while the **north, south, east, and west** pixels belong to **the foreground**. The rest of pixels in the neighborhood can be of any kind; we don't care about them. Now, let's apply this kernel to an input image:

Foreground Pixels

Background Pixels

0	1	0
1	0	1
0	1	0

0	0	0
0	1	0
0	0	0

0	1	0
1	-1	1
0	1	0

Don’t Care

Structuring elements (kernels). Left: kernel to 'hit'. Middle: kernel to 'miss'. Right: final combined kernel

0	0	0	0	0	0	0	0
0	255	255	255	0	0	0	255
0	255	255	255	0	0	0	0
0	255	255	255	0	255	0	0
0	0	255	0	0	0	0	0
0	0	255	0	0	255	255	0
0	255	0	255	0	0	255	0
0	255	255	0	0	0	0	0

Input binary image

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	255	0	0	0	0	0
0	0	0	0	0	0	0	0

Output binary image

Hit or Miss Transform (HMT)

Here you can find the output results of applying different kernels to the same input image used before:

0	0	0	0	0	0	0	0
0	255	255	255	0	0	0	255
0	255	255	255	0	0	0	0
0	255	255	255	0	255	0	0
0	0	255	0	0	0	0	0
0	0	255	0	0	255	255	0
0	255	0	255	0	0	255	0
0	255	255	0	0	0	0	0

Input binary image

$\circledast$

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

$\Rightarrow$

0	0	0	0	0	0	0	0
0	0	0	255	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	255	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Output binary image

Hit or Miss Transform (HMT)

0	0	0	0	0	0	0	0
0	255	255	255	0	0	0	255
0	255	255	255	0	0	0	0
0	255	255	255	0	255	0	0
0	0	255	0	0	0	0	0
0	0	255	0	0	255	255	0
0	255	0	255	0	0	255	0
0	255	255	0	0	0	0	0

Input binary image

$\circledast$

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

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	255
0	0	0	0	0	0	0	0
0	0	0	0	0	255	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	255	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Output binary image

Basic Morphological Algorithms

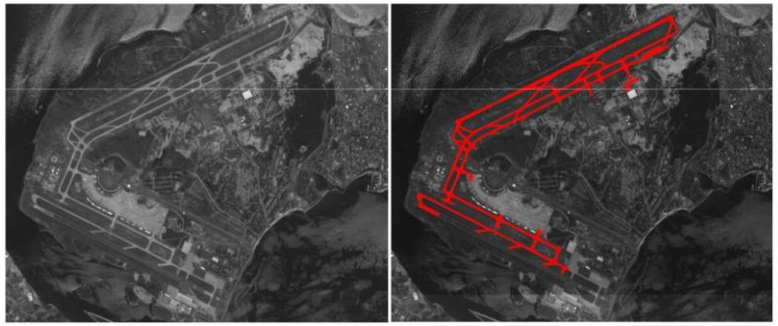
Using the simple technique we have looked at so far, we can begin to consider some more interesting morphological algorithms.

We will Look at:

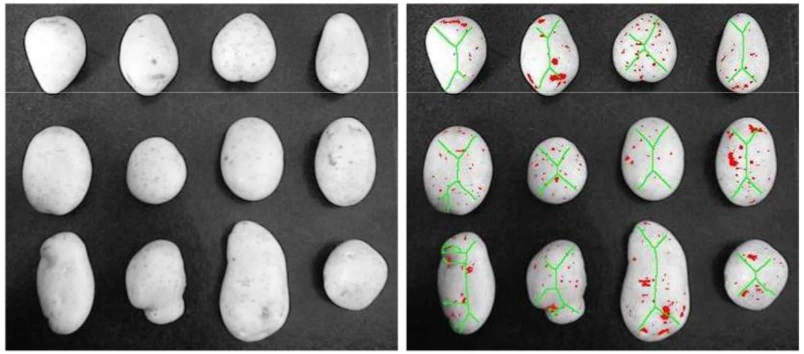
- Boundary extraction
- Region filling
- Extraction of connected components

There are lots of others as well though:

- Thinning/thickening
- Skeletonization



Detecting runways in satellite airport imagery

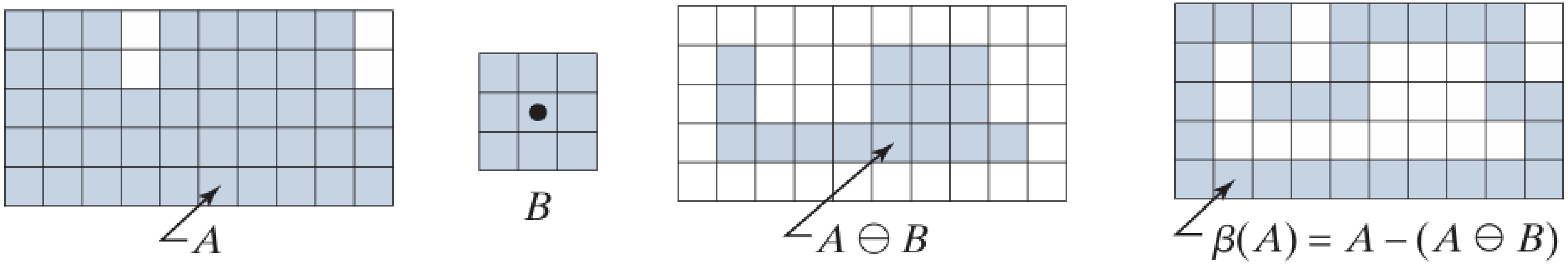


Grading potato quality by shape and skin spots

Boundary Extraction

The boundary of set A denoted by ß(A) is obtained by first eroding A by a suitable structuring element B and then taking the difference between A and its erosion.

$$\beta(A) = A - (A \ominus B)$$



Boundary Extraction

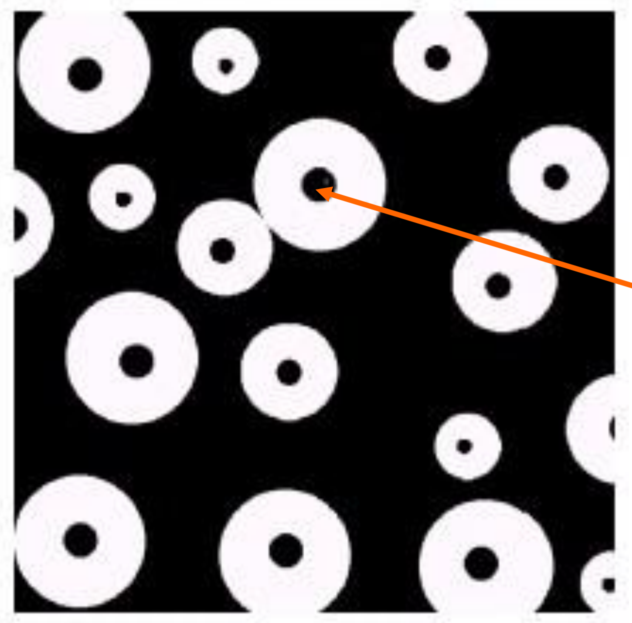
A simple image and the result of performing boundary extraction using a square 3*3 structuring element

Original Image

Extracted Boundary

Region (hole) Filling

Given a pixel inside a boundary, region filling attempts to fill that boundary with object pixels (1s)



Given a point inside here,
Can we fill the whole
circle?

Region (hole) Filling

Let A is a set containing a subset whose elements are 8-connected boundary points of a region, enclosing a background region i.e. hole

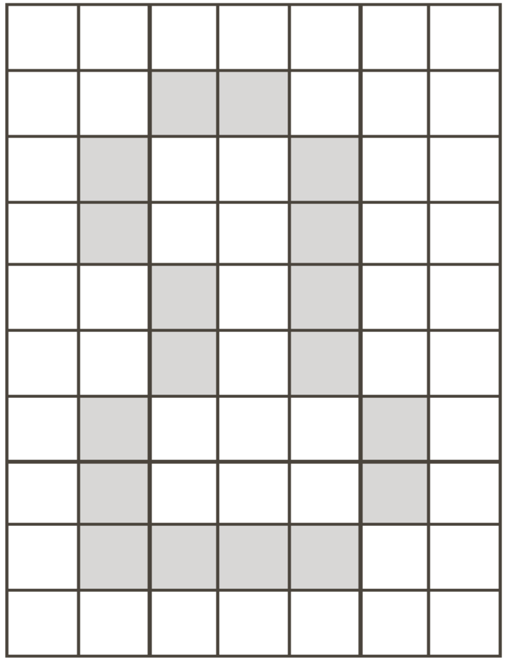
If all boundary points are labeled 1 and non boundary points are labeled 0, the following procedure fills the region:

- Start from a known point p and taking X_0= p,
- Then taking the next values of X_k as:

$$X_{k+1} = (X_k \oplus B) \cap A^c \qquad k = 1,2,3, \ldots$$

B is suitable structuring element

- Terminate iterations if X_k and A contains the filled set and its boundaries.
- The set union of X_k and A contains the filled set and its boundaries



A

Region (hole) Filling

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

A

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

A^c

0	1	0
1	1	1
0	1	0

B

Region (hole) Filling

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

X_0

0	1	0
1	1	1
0	1	0

B

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

$(X_0 \oplus B)$

$$X_{k+1} = (X_k \oplus B) \bigcap A^c$$

$$k = 1, 2, 3, \cdots$$

Region (hole) Filling

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

$X_1 = (X_0 \oplus B)$

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

A^c

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

$X_1 = (X_0 \oplus B) \cap A^c$

$$X_{k+1} = (X_k \oplus B) \cap A^c$$

$$k = 1, 2, 3, \cdots$$

Region (hole) Filling

$$X_{k+1} = (X_k \oplus B) \cap A^c$$

$k = 1,2,3, \ldots$

NOTE:
The intersection of dilation and the complement of A limits the result to inside the region of interest

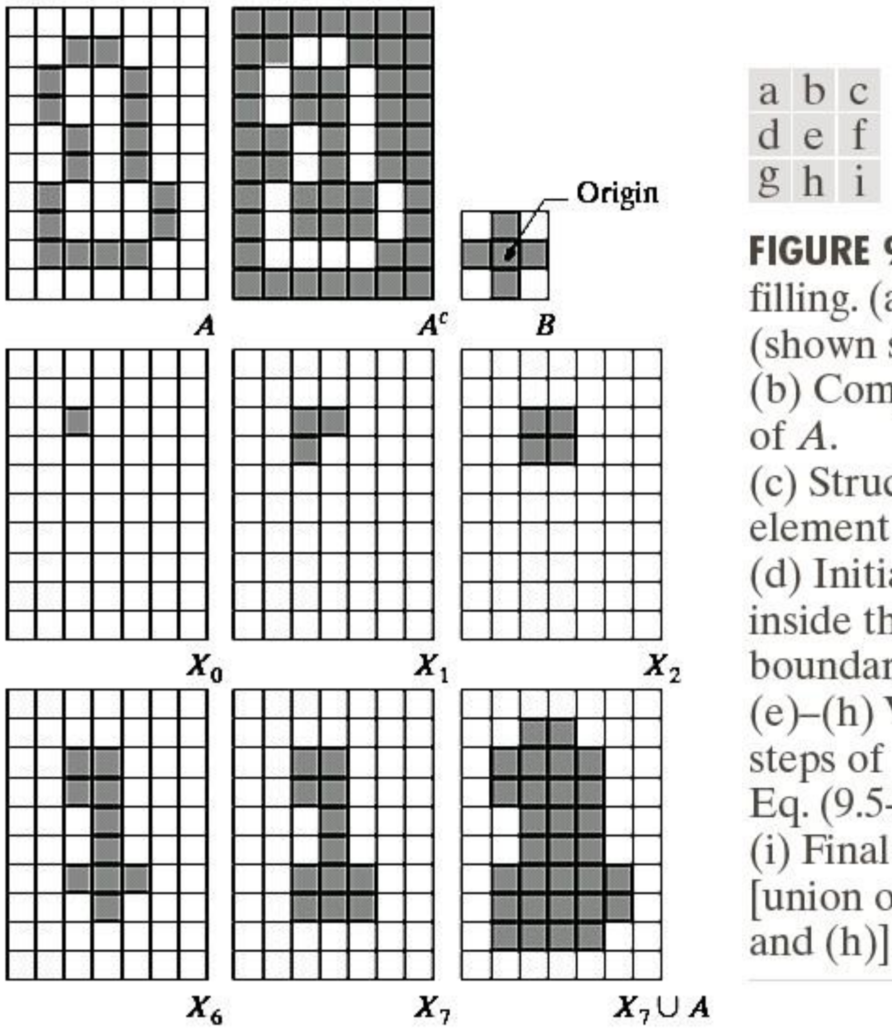
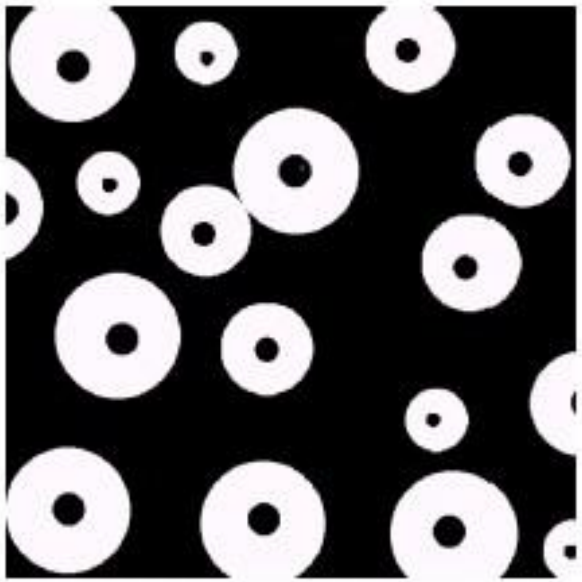
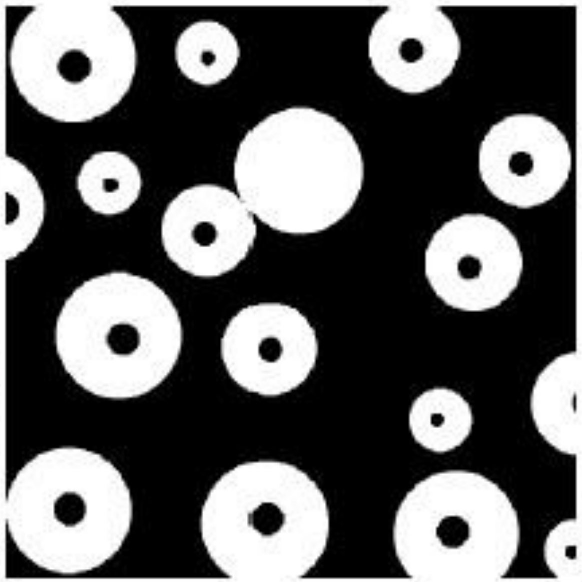


FIGURE 9.15 Hole filling. (a) Set A (shown shaded). (b) Complement of A. (c) Structuring element B. (d) Initial point inside the boundary. (e)–(h) Various steps of Eq. (9.5-2). (i) Final result [union of (a) and (h)].

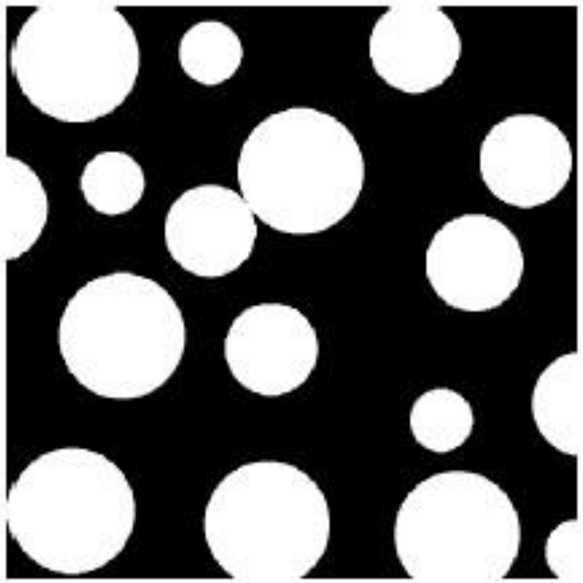
Region Filling: Example



Original Image



One Region Filled



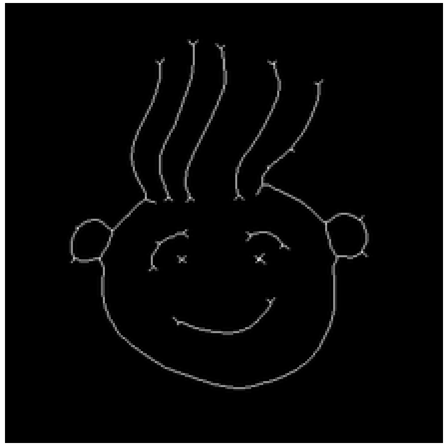
All Regions Filled

Thinning

- The thinning of a set A by a structuring element B, denoted by $A \otimes B$, can be defined in terms of the hit-or-miss transform:

$$A \otimes B = A - (A \circledast B)$$
$$= A \cap (A \circledast B)^C$$

- Thinning is done using a sequence of structuring elements where each one is the rotated version of the previous one in the sequence.



Thinning

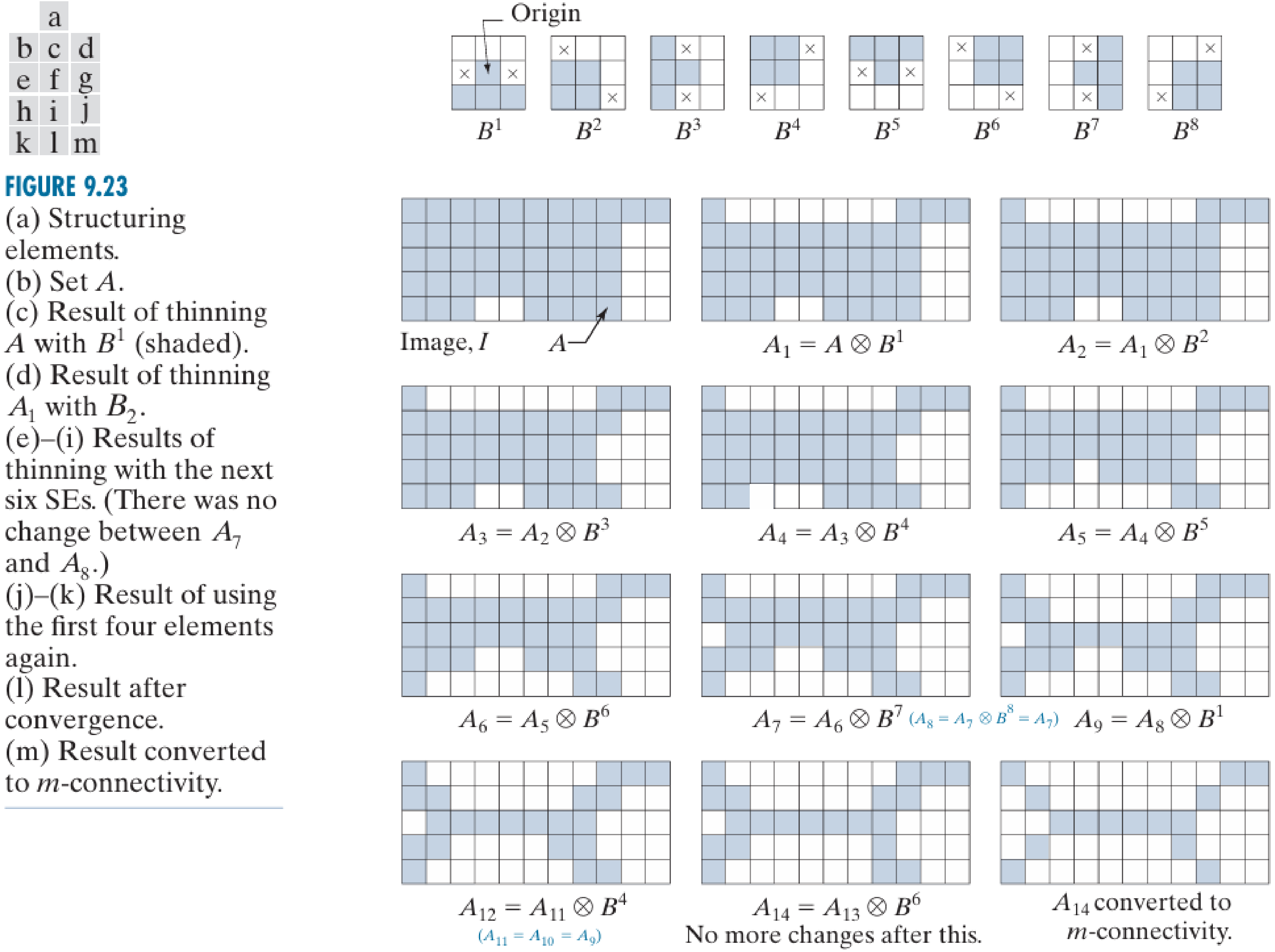


FIGURE 9.23
(a) Structuring elements.
(b) Set A.
(c) Result of thinning A with B^1 (shaded).
(d) Result of thinning A_1 with B_2.
(e)–(i) Results of thinning with the next six SEs. (There was no change between A_7 and A_8.)
(j)–(k) Result of using the first four elements again.
(l) Result after convergence.
(m) Result converted to m-connectivity.

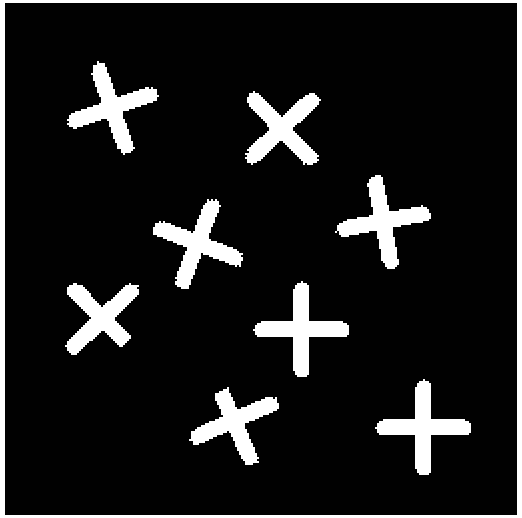
Thickening

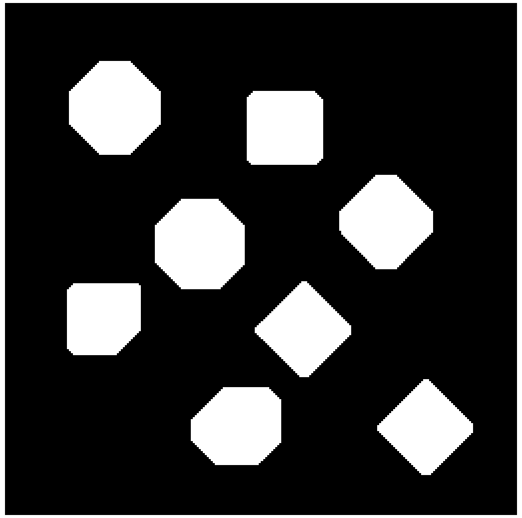
- Thickening is the morphological dual of thinning and is defined by,

$$A \odot B = A \cup (A \circledast \mathrm{B})$$

Where B is the structuring element suitable for thickening.

- A sequence of Structuring elements of the same form as in thinning but with all 1's and 0's interchanged can be used.





Thickening

a b
c d
e

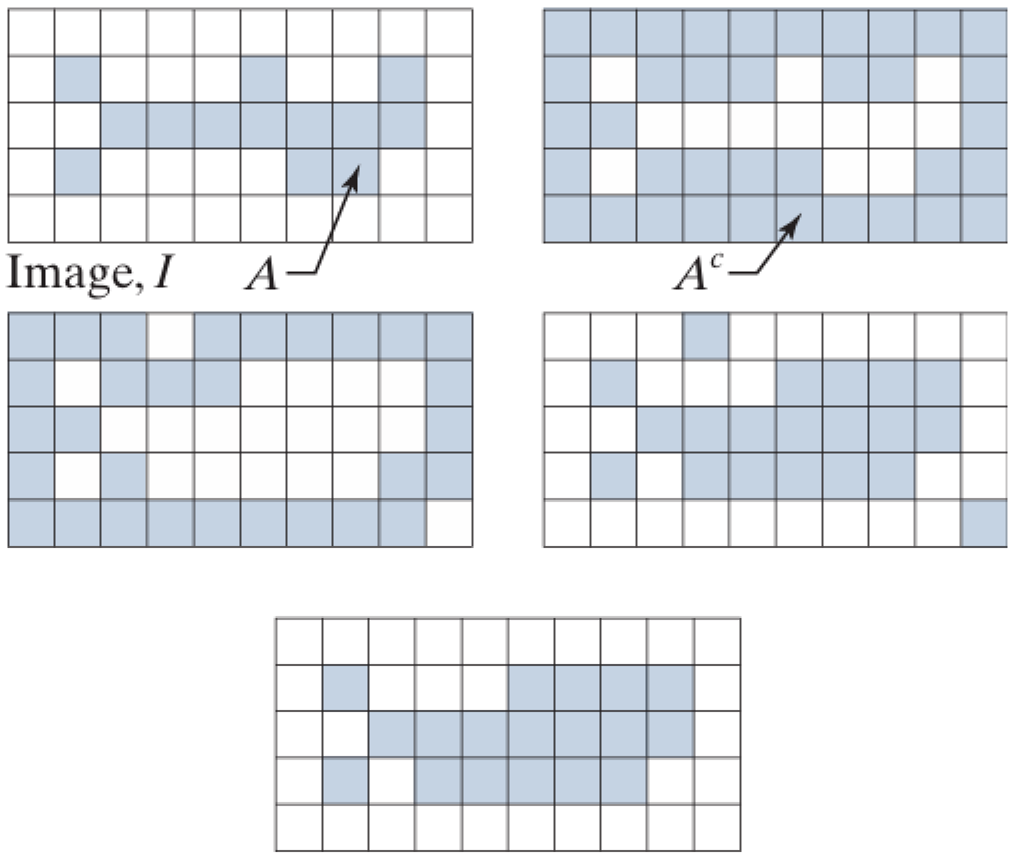
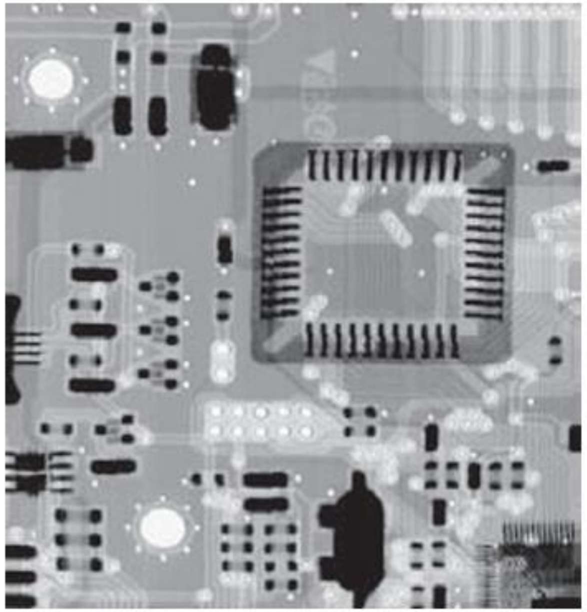


FIGURE 9.24
(a) Set A.
(b) Complement of A.
(c) Result of thinning the complement.
(d) Thickened set obtained by complementing (c).
(e) Final result, with no disconnected points.

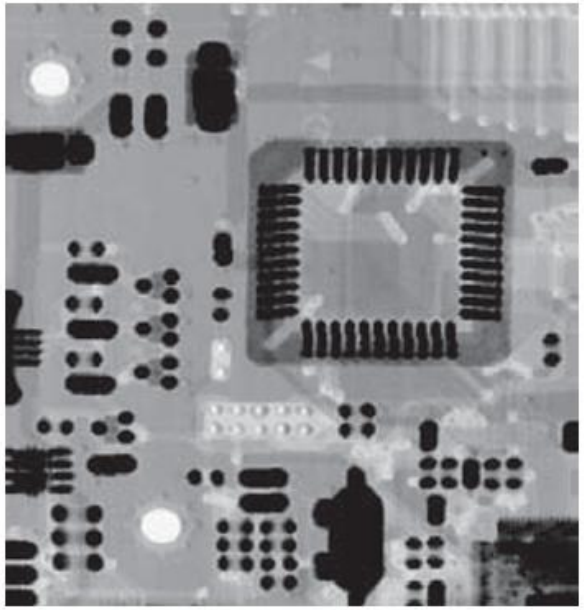
Gray Level Image Morphological Operations

Dilation & Erosion

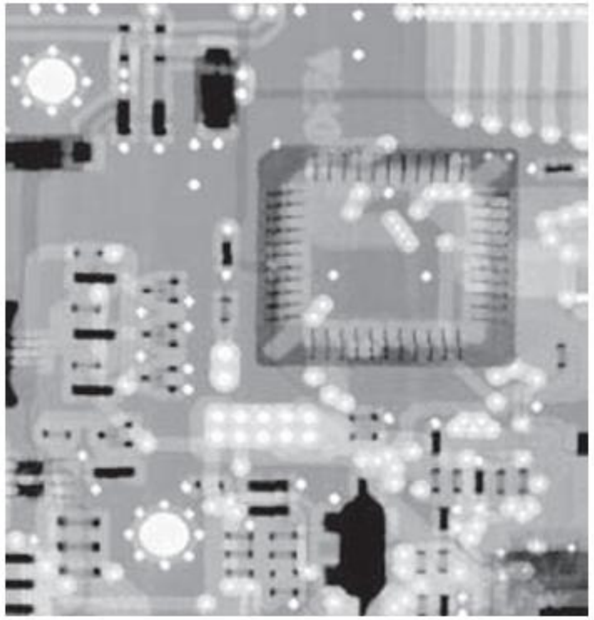
$$f \oplus b\,(x, y) = max\{f(s - x, t - y)\}$$
$$f \ominus b\,(x, y) = min\{f(s - x, t - y)\}$$



Gray-scale X-ray image of size 448 425 × pixels.



Erosion using a flat disk SE with a radius of 2 pixels.

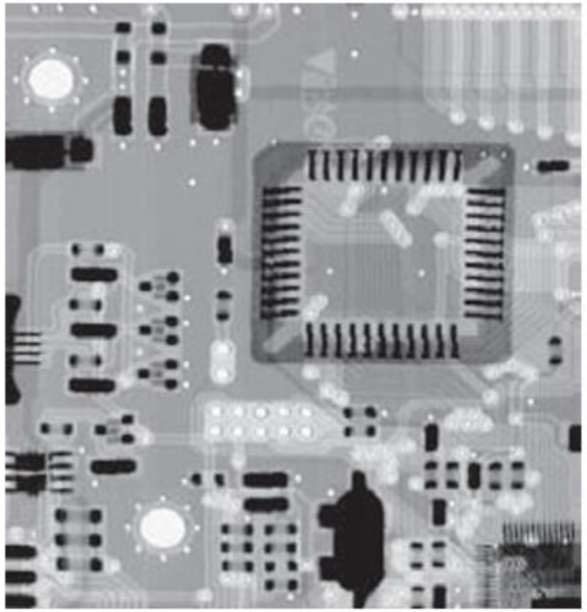


Dilation using the same SE.

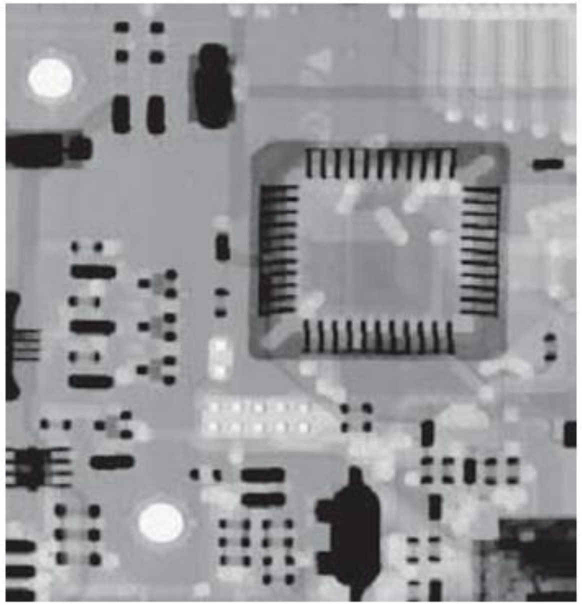
Gray Level Image Morphological Operations

Opening & Closing

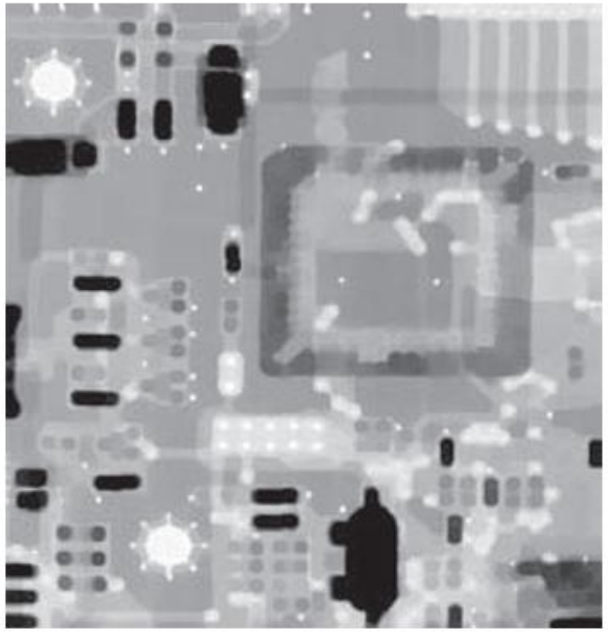
$$f \circ b = (f \ominus b) \oplus b$$
$$f \bullet b = (f \oplus b) \ominus b$$



Gray-scale X-ray image of size 448 425 × pixels.



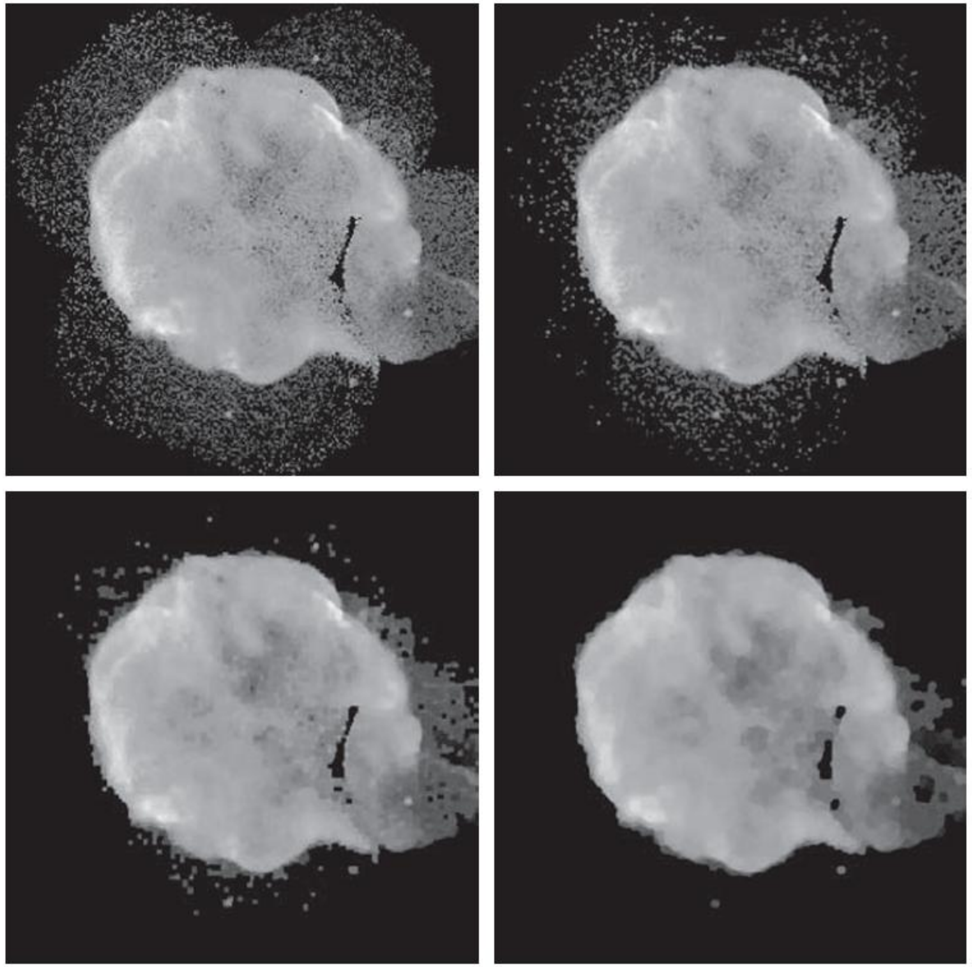
Opening using a disk SE with a radius of 3 pixels.



Closing using an SE of radius 5.

Gray Level Image Morphological Operations

Opening & Closing

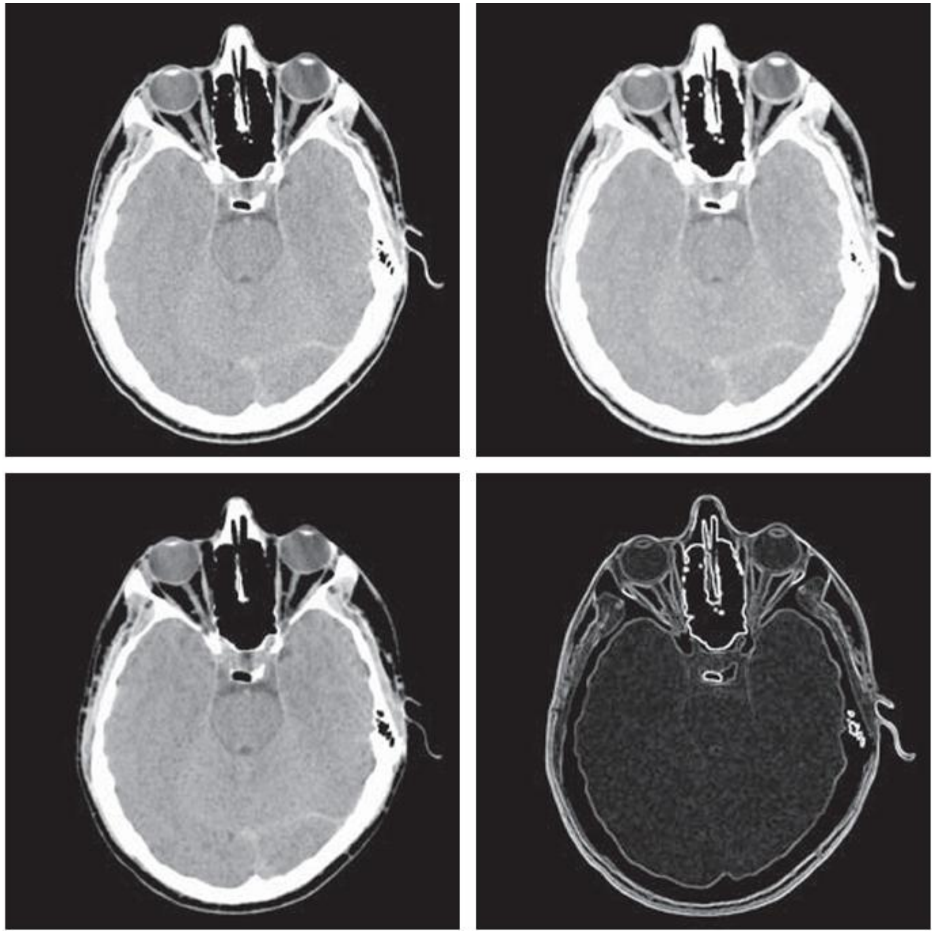


(a) 566×566 image of the Cygnus Loop supernova, taken in the X-ray band by NASA's Hubble Telescope.
(b)–(d) Results of performing opening and closing sequences on the original image with disk structuring elements of radii, 1, 3, and 5, respectively.

Gray Level Image Morphological Operations

Morphological Gradient

$$g = (f \oplus b) - (f \ominus b)$$

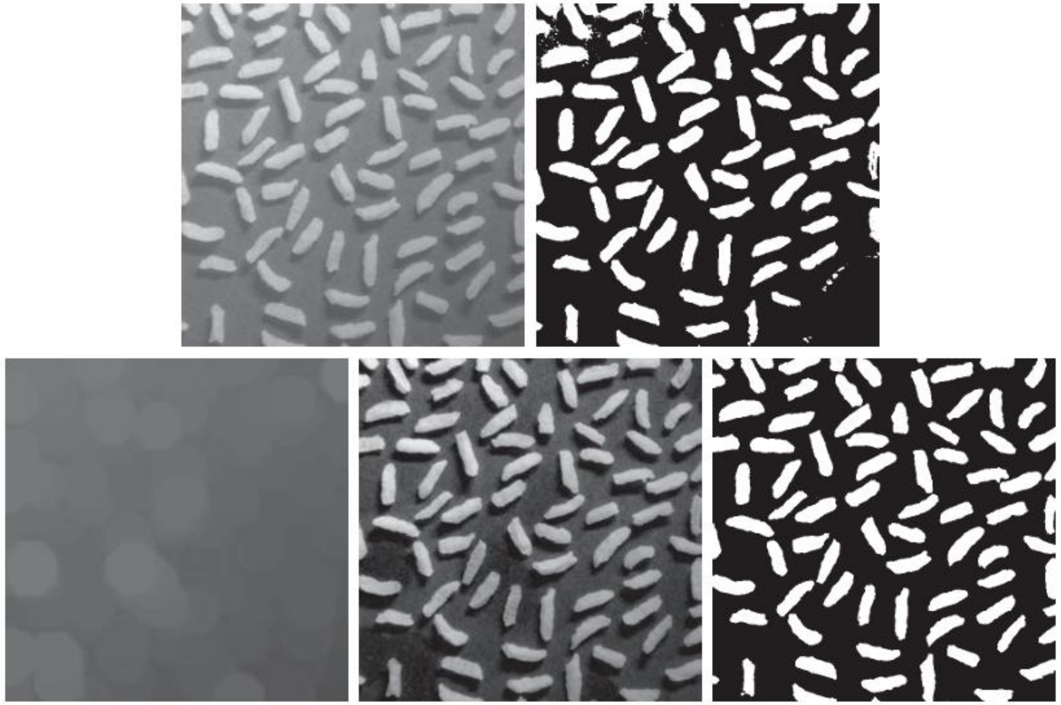


(a) 512 × 512 image of a head CT scan. (b) Dilation. (c) Erosion. (d) Morphological gradient, computed as the difference between (b) and (c). (Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)

Gray Level Image Morphological Operations

Top Hat & Bottom Hat Transformations

$$g_{top} = f - (f \circ b) \qquad g_{bot} = f - (f \bullet b)$$



Using the top-hat transformation for shading correction. (a) Original image of size 600 600 pixels. (b) Thresholded image. (c) Image opened using a disk SE of radius 40. (d) Top-hat transformation (the image minus its opening). (e) Thresholded top-hat image.

Connected Components Labeling

An operation in image processing used to detect connected regions in binary images.

It groups pixels that are connected and share the same value (usually “1” for foreground).

Key Concept

- If you can walk from Pixel A to Pixel B without stepping off the foreground pixels, they belong to the same object.

Mathematical Definition

Let **B** be a binary image where a pixel at coordinates **(r, c)** is denoted as **p**.V = {1} is the set of foreground values (the object pixels). Then Connectivity Relation (~),two pixels **p** and **q** are "connected" (**p ~ q**) if:

Value Check: Both **p** and **q** have a value of **1**.

Path Existence: There exists a sequence of pixels $p_0, p_1, \dots, p_n$ such that:

- $p_0 = p \text{ and } p_n = q$
- Every pixel in the sequence is a neighbor of the previous one.

Connected Components Labeling

Algorithm & Output Example

Scan the image pixel by pixel.

Check Neighbors of current pixel (based on 4 or 8 connectivity).

Label:

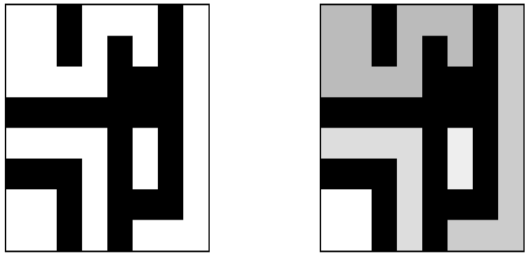
- If no neighbor has a label, assign a **new ID**.
- If a neighbor has a label, assign the **same ID**.
- If neighbors have conflicting labels, record them as "equivalent" to merge later.

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

Geometric and Shape Properties

In this section we defined the operations of binary and show how they can be useful in processing the regions, derived from the connected components labeling operation.

- area
- centroid
- perimeter
- perimeter length
- circularity
- elongation
- mean and standard deviation of radial distance
- bounding box

Which are statistical? Which are structural?

Region Properties

Properties of the regions can be used to recognize objects.

- geometric properties
- gray-tone properties
- color properties
- texture properties
- shape properties
- motion properties
- relationship properties

Area and Centroid

We denote the set of pixels in a region by **R**.

assuming square pixels:

area:

$$A = \sum_{(r,c)\,\epsilon\,R} 1$$

centroid:

$$\bar{r} = \frac{1}{A} \sum_{(r,c)\,\epsilon\,R} r$$

$$\bar{c} = \frac{1}{A} \sum_{(r,c)\,\epsilon\,R} c$$

$(\bar{r}\,, \bar{c})$ is generally not a pair of integers.

A precision of tenths of a pixel is often justifiable for the centroid.

Perimeter pixels and length

Let perimeter **P** be the actual set of boundary pixels.

P must be ordered in a sequence $P \; = <(r_0,\; c_0), \ldots, (r_{k-1}, c_{k-1})>$.

Each pair of successive pixels in **P** are neighbors, including the first and last pixels.

perimeter length:

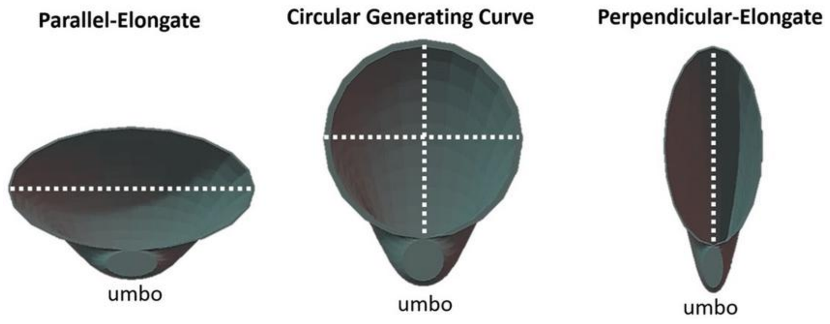
$$|p| = \#\,\{k \mid \left(r_{(k+1)}, c_{k+1}\right) \epsilon\, N_4(r_k, c_k)\}$$
$$+\sqrt{2}\,\#\{k \mid \left(r_{(k+1)}, c_{k+1}\right) \epsilon\, N_8(r_k, c_k) - N_4(r_k, c_k)$$

where k + 1 is computed modulo K.

Perimeter can vary significantly with object orientation.

Circularity or elongation

Common measure of circularity of a region is length of the parameter squared divided by area.



Circularity (1): $\boldsymbol{C_1 = \frac{|P|^2}{A}}$

Circularity as variance of "radius"

a second measure uses variation off of a circle

Circularity (2): $\boldsymbol{C_2 = \frac{\mu R}{\sigma R}}$

where $\boldsymbol{\mu R}$ and $\boldsymbol{\sigma^2 R}$ are the mean and variance of the distance from the centroid of the shape to the boundary pixels (r_k, c_k).

mean radial distance: $\boldsymbol{\mu R = \frac{1}{K} \sum_{k=0}^{K-1} ||(r_k, c_k) - (\bar{r}, \bar{c})||}$

variance of radial distance: $\boldsymbol{\sigma^2 R = \frac{1}{K} \sum_{k=0}^{K-1} \left[||(r_k, c_K) - (\bar{r}, \bar{c})|| - \mu R \right]^2}$

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

Calculating Centroid For Object 1

	8	9	10	11	12	13	14	15
3	0	0	1	1	1	1	0	0
4	0	1	1	1	1	1	1	0
5	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1
8	0	1	1	1	1	1	1	0
9	0	0	1	1	1	1	0	0

This the object in the previous image. Look into Row and Column numbers.

A = 44

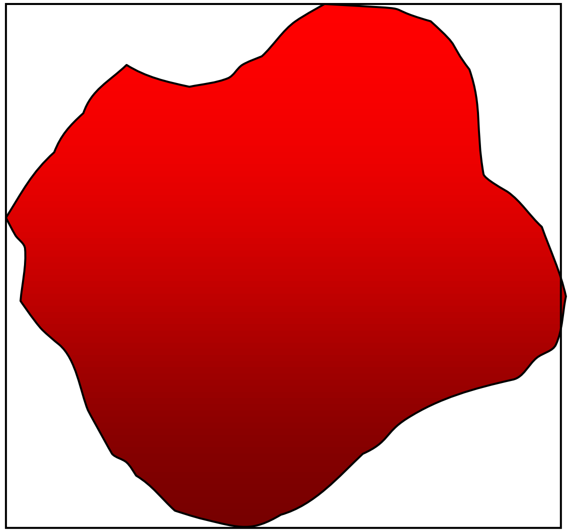
- 3*4 + 4*6 + 5*8 + 6*8 + 7*8 + 8*6 + 9*4 = 264/44 = 6
- 8*3 + 9*5 + 10*7 + 11*7 + 12*7 + 13*7 + 14*5 + 15*3 = 506 / 44 = 11.5

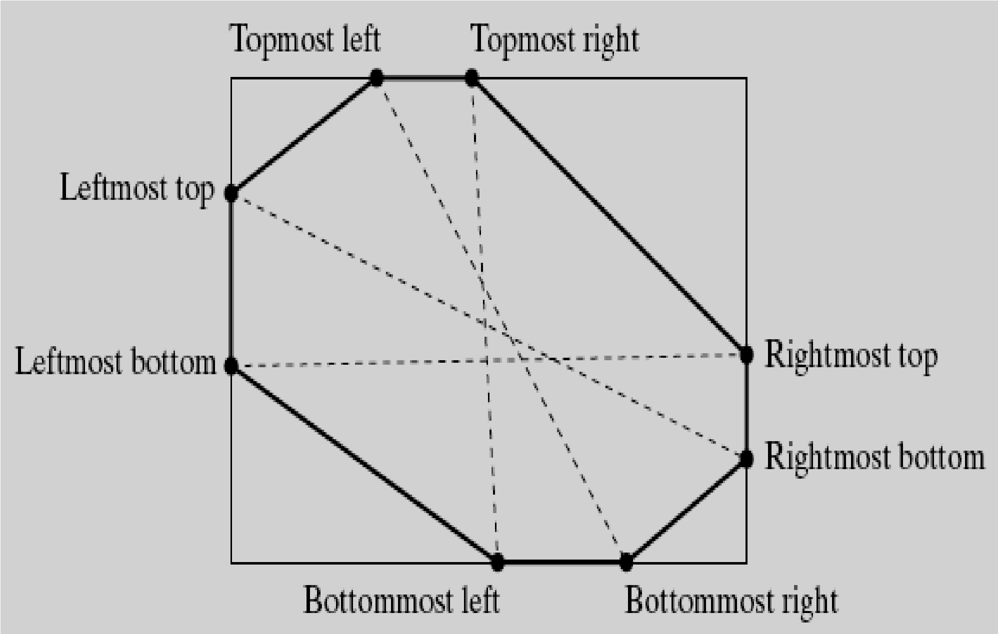
$$\bar{r} = \frac{\sum Row\ index\ \times\ Counts\ of\ 1's\ in\ that\ row}{Total\ Area}$$

$$\bar{c} = \frac{\sum Col\ index\ \times\ Counts\ of\ 1's\ in\ that\ col}{Total\ Area}$$

Bounding Box

- It is often useful to have rough idea to know the object placement in a image.
- Geometric property of bounding box is to bound any shape of object in two vertical and two horizontal lines.
- For an object there will always be four points (topmost, bottommost, leftmost and rightmost)





Thank You

For Your Attention