Advanced (Mid-Level) DIP
-- Morphological Image Processing

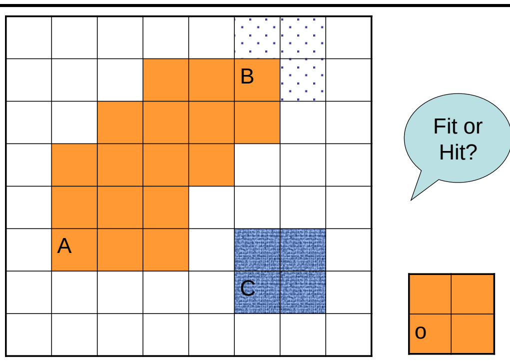
Xiaojun Qi

Morphology

- Morphology: It is a tool for extracting image components that are useful in the representation and description of region shape, such as boundaries, skeletons, and the convex hull.
- Morphological operations preserve the basic properties of object while removing irrelevant features.
- Applications: segmentation, enhancement, restoration, edge detection, texture analysis, shape analysis, thinning, and curve filling.

- The binary images are normally produced by simple segmentation techniques such as thresholding. They may contain numerous imperfections caused by noise, texture or the inaccurate specification of a threshold. Morphological image processing techniques can remove the imperfections and provide us with information on the **form and structure** of the image.
- Morphological techniques are also applicable to grayscale images, where they can be used for non-linear smoothing and feature enhancement.

- The term “morphological image processing” describes a range of **non-linear** image processing techniques that deal with the shape of features in an image.
- Morphological techniques typically probe an image with a small shape or template known as a **structuring element (SE)**. The SE is positioned at all possible locations in the image and it is compared with the corresponding neighborhood of pixels.
- Morphological operations differ in how they carry out the comparison. Some test whether the structuring element ‘fits’ within the neighborhood; others test whether it ‘hits’ or ‘intersects’ the neighborhood.



Probe an image with a structuring element at A, B, and C positions.

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

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

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

The dimensions of the matrix determine the overall size of the structuring element, and its shape is determined by the pattern of ones and zeros.

1) What are the sizes of the above three SEs?
2) What are the shapes of the above three SEs?

Fit or Hit?

- A structuring element (SE) is said to fit an image if for **each** of its pixels that is set to 1, the corresponding image pixel is also 1. The SE pixels that are 0 define points where the corresponding image value is irrelevant.
- The SE is said to hit an image if for **any** of its pixels that is set to 1, the corresponding pixels that is set to 1, the corresponding image pixel is also 1. We also ignore image pixels for which the corresponding SE pixel is 0.

7

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

Suppose s1 is a 3*3 square SE and s2 is a 3*3 cross-shaped SE, determine whether they are hit or fit in three positions above.

8

Basic Idea of Morphology

- Probe an image with a SE and determine the manner in which the SE fits, hits, (or does not fit or hit) within the image.
- The SE will be small compared to the image.
- By marking the locations where the SE fits or hits within the image, one obtains information about the structure of the image which depends upon the size and shape of the SE.

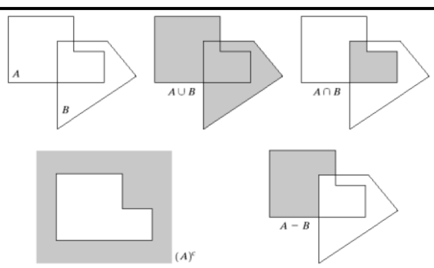
9

Set Theory and Notation

- **Translation:** $(A)_x = \{c \mid c = a + x, a \in A\}$
- **Reflection:** $\hat{A} = \{c \mid c = -a, a \in A\}$
- Complement: $A^c = \{c \mid c \notin A\}$
- Intersection: $A \cap B = \{c \mid (c \in A) \wedge (c \in B)\}$
- Union: $A \cup B = \{c \mid (c \in A) \vee (c \in B)\}$
- Difference: $A - B = \{c \mid (c \in A) \wedge (c \notin B)\}$
- SubSet: $A \subseteq B = \{c \mid \text{if } c \in A, \text{then } c \in B\}$
- Disjoint (Mutually Exclusive) $A \cap B = \phi$

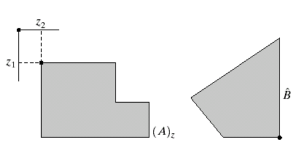
The elements of the sets are the coordinates of pixels (i.e., members of the 2D integer space Z^2) representing objects or other features of interest in an image.

10



a b c
d e

FIGURE 9.1 (a) Two sets A and B. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B.



a b

FIGURE 9.2 (a) Translation of A by z. (b) Reflection of B. The sets A and B are from Fig. 9.1.

Logical Operations

TABLE 9.1 The three basic logical operations.

p	q	p AND q (also $p \cdot q$)	p OR q (also $p + q$)	NOT (p) (also $\bar{p}$)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

These three operations are functionally complete in the sense that they can be combined to form any other logic operation.

12

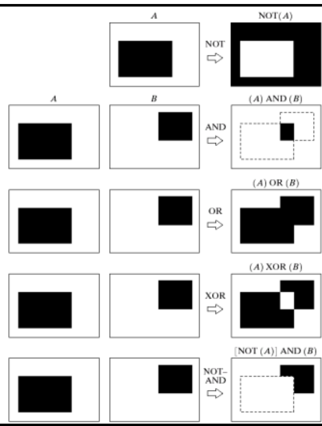


FIGURE 9.3 Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

13

Binary Dilation

- The dilation of A by B is defined as:

$$A \oplus B = \{ p \mid (\hat{B})_p \cap A \neq \phi \} = \{ c \mid c = a + b, a \in A, b \in B \}$$

$$= \{ \cup \{ A + b \} \mid b \in B \} = \{ p \mid (\hat{B} + p) \cap A \neq \phi \}.$$

- Set B is commonly referred to as the **structuring element** in dilation, as well as in other morphological operations.
- Dilation **expands the object** if the origin is in B.

14

Binary Dilation Implementation View 1

- Position a SE **s** such that its origin is at image pixel coordinates (x, y) and apply the rule:

$$g(x,y) = \begin{cases} 1 & \text{if } s \text{ hits } f. \\ 0 & \text{otherwise.} \end{cases}$$

repeating for all x and y.

- Dilation creates a new image that shows all the locations of a SE's origin at which it hits the input image.

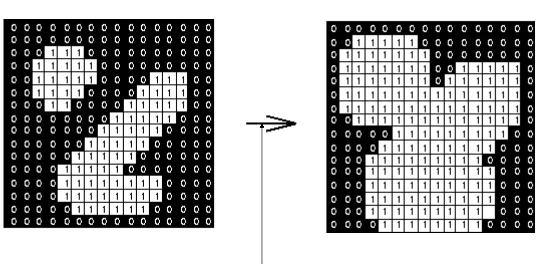
15

The mathematical definition of dilation for *binary* images $A \oplus B = \{ p \mid (\hat{B})_p \cap A \neq \phi \}$ is as follows:

- Suppose that *X* is the set of Euclidean coordinates corresponding to the input binary image, and that *P* is the set of coordinates for the SE. Let $(P^\wedge)_X$ denote the translation of reflected *P* so that its origin is at *x*.
- Then the dilation of *X* by *P* is simply the set of all points *x* such that the intersection of $(P^\wedge)_X$ with *X* is non-empty.

16

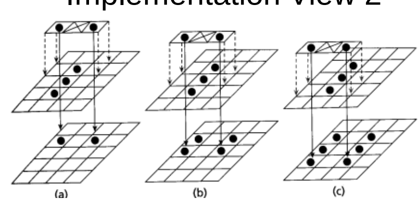
Binary Dilation Example 1



We used a 3*3 square SE to do the dilation

17

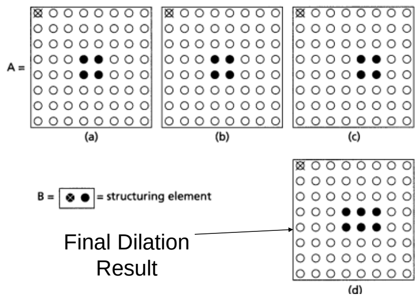
Binary Dilation Implementation View 2



Think of the SE as a template, and move it over the image. When the origin of the SE aligns with a black pixel in the image, all of the image pixels that correspond to black pixels in the SE are marked, and will later be changed to black. After the entire image has been swept by the SE, the dilation calculation is complete.

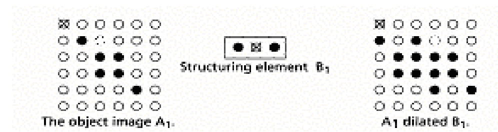
18

Binary Dilation Example 2
-- Origin is part of the SE



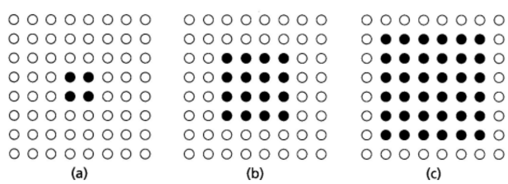
19

Binary Dilation Example 3
-- Origin isn't part of the SE



20

Binary Dilation Example 4
-- Origin is part of the SE



(a) → (b) → (c) dilation by one pixel each time

1) What kind of SE can achieve (a)→(b) and (b)→(c)?

2) What kind of SE can achieve (a) → (c)?

21

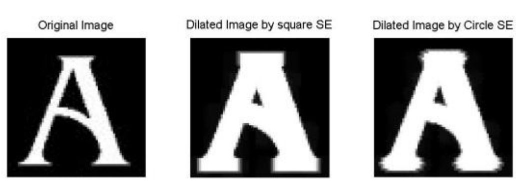
Binary Dilation Example 5
-- Effects of the SE size

Here we used square SE

What is the effect after applying the dilation operation?
-- Inner and outer boundaries, holes, and gaps

22

Binary Dilation Example 6
-- Effects of the SE shape

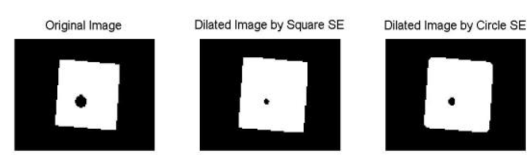


Here we used a square SE of size 6*6 and a circle SE of radius 3

What is the effect after applying the dilation operation?
-- Inner and outer boundaries, holes, gaps and corner

23

Binary Dilate Example 7
-- Effects of the SE shape



5*5 Square SE

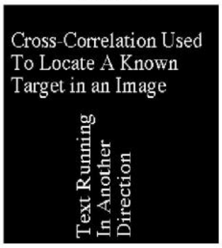
Circle SE with radius of 2

What is the effect after applying the dilation operation?
-- Boundaries, holes, and corner

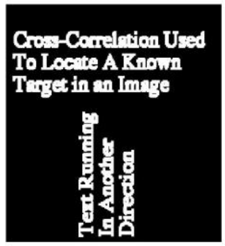
24

Binary Dilation Example 8

Original Image

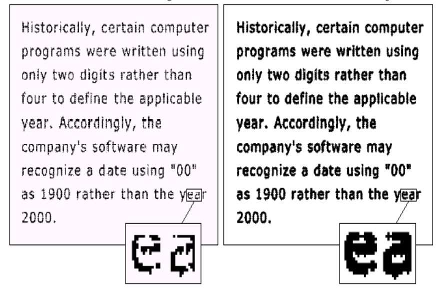


Dilated Image by sqaure SE



We used 3*3 square SE to do the dilation

Binary Dilation Example 9



a c
b

FIGURE 9.5
(a) Sample text of poor resolution with broken characters (magnified view).
(b) Structuring element.
(c) Dilation of (a) by (b). Broken segments were joined.

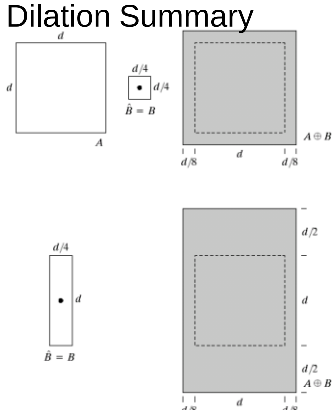
0	1	0
1	1	1
0	1	0

Assume the maximum length of the breaks is known to be two pixels.

Binary Dilation Summary

a b c
d e

FIGURE 9.4
(a) Set A.
(b) Square structuring element (dot is the center).
(c) Dilation of A by B, shown shaded.
(d) Elongated structuring element.
(e) Dilation of A using this element.



Binary Erosion

- The erosion of A by B is defined as:

$$A \ominus B = \{p \mid (B)_p \subseteq A\} = \{p \mid p + B \subseteq A\}$$
$$= \{\cap (A - b) \mid b \in B\}$$

- If the translated set is contained in set A, the p is in the erosion of A with B.
- Erosion results in a **shrink** of the image.
- Dilation and erosion are duals of each other with respect to set complementation and reflection.

$$(A \ominus B)^c = A^c \oplus \hat{B}$$

Binary Erosion Implementation View 1

- Position a SE **s** such that its origin is at image pixel coordinates (x, y) and apply the rule:

$$g(x,y) = \begin{cases} 1 & \text{if } s \text{ fits } f. \\ 0 & \text{otherwise.} \end{cases}$$

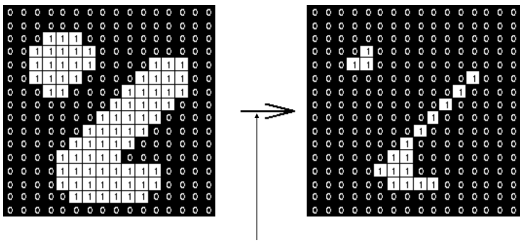
repeating for all x and y.

- Erosion creates a new image that marks all the locations of a SE's origin at which it fits the input image.

The mathematical definition of erosion for *binary* images is $A \ominus B = \{p \mid (B)_p \subseteq A\}$ as follows:

- Suppose that *X* is the set of Euclidean coordinates corresponding to the input binary image, and that *P* is the set of coordinates for the SE. Let $(P)_x$ denote the translation of *P* so that its origin is at *x*.
- Then the erosion of *X* by *P* is simply the set of all points *x* such that *(P)x* is a subset of *X*.

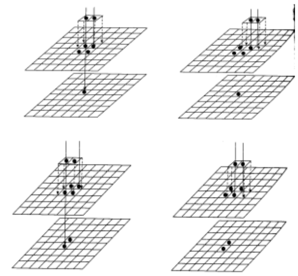
Binary Erosion Example 1



We used 3*3 square SE to do the erosion

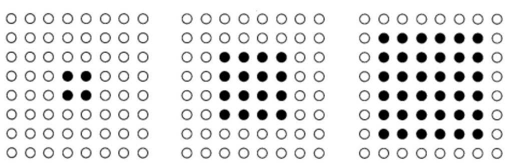
31

Binary Erosion Implementation View 2



32

Binary Erosion Example 2



(a) (b) (c)

(c) → (b) → (a) erosion by one pixel each time

1) What kind of SE can achieve (c)→(b) and (b)→(a)?

2) What kind of SE can achieve (c) → (a)?

33

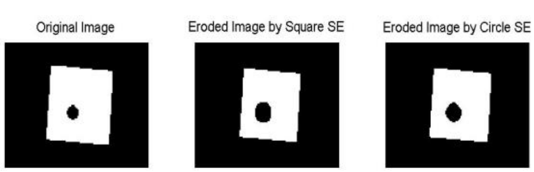
Binary Erosion Example 3
-- Effects of the SE size

Here we used square SE

What is the effect after applying the erosion operation?
-- Inner and outer boundaries, holes, and gaps

34

Binary Erosion Example 4
-- Effects of the structuring element size



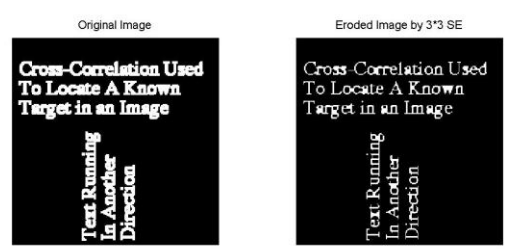
5*5 Square SE

Circle SE with radius of 2

What is the effect after applying the erosion operation?
-- Boundaries, holes, and corner

35

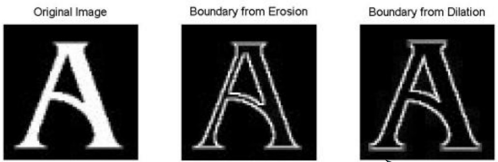
Binary Erosion Example 5



We used 3*3 square SE to do the Erosion

36

Finding a Region Boundary



Source Code to Generate the Boundary

```
A = imread('char_a.jpg') ;
se1 = strel('square', 3) ;
SquareErodeA1 = imerode(A, se1) ;
Boundary1 = imsubtract(A, SquareErodeA1) ;
SquareDilateA1 = imdilate(A, se1) ;
Boundary2 = imsubtract(SquareDilateA1, A) ;
```

What is the difference between these two boundaries?

Binary Erosion Summary

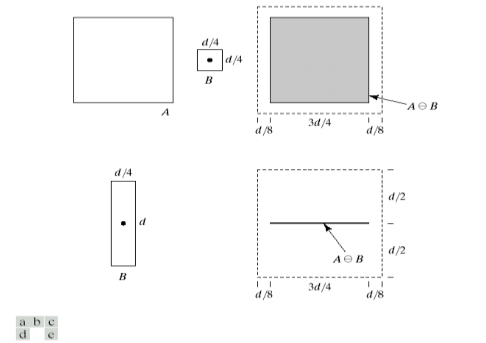


FIGURE 9.6 (a) Set A. (b) Square structuring element. (c) Erosion of *A* by *B*, shown shaded. (d) Elongated structuring element. (e) Erosion of *A* using this element.

Summary
-- Binary Dilation and Binary Erosion

- The result of dilation/erosion depends on SE size and shape, with larger SEs having a more pronounced effect.
- The result of dilation/erosion with a large SE is similar to the result obtained by iterated dilation/erosion using a smaller SE of the same shape.
- Dilation by a disc enlarges the object and smoothes its convex corner; Erosion by a disc shrinks the object and smoothes its concave corner.

Relationship Between Binary Dilation and Binary Erosion

- Dilation and erosion are duals of each other with respect to set complementation and reflection. That is:

$$(A \ominus B)^C = A^C \oplus \hat{B}$$

How to prove this dual relationship between dilation and erosion?

- Dilation expands an image and erosion shrinks it.

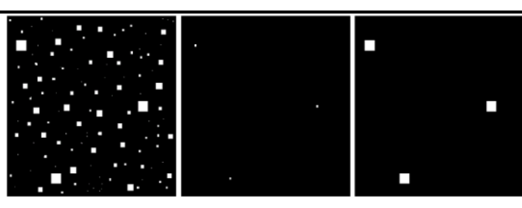


FIGURE 9.7 (a) Image of squares of size 1, 3, 5, 7, 9, and 15 pixels on the side. (b) Erosion of (a) with a square structuring element of 1's, 13 pixels on the side. (c) Dilation of (b) with the same structuring element.

1. What is the structuring element?
2. Does erosion followed by dilation fully restore the eroded objects in this case?
3. Can we expand the conclusion from 2 to a general case?

The “active” elements of structuring elements assume the same binary values as the objects of interest.

Binary Opening

- The opening of A by B is defined as:

$$A \circ B = (A \ominus B) \oplus B$$

- The opening of A by B is the erosion of A by B, followed by a dilation of the result by B.
- Opening generally smoothes the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions. → “open” small gaps or spaces between touching objects in an image.
- Remove much of the black pixel noise in the background of the image. (Pepper Noise)

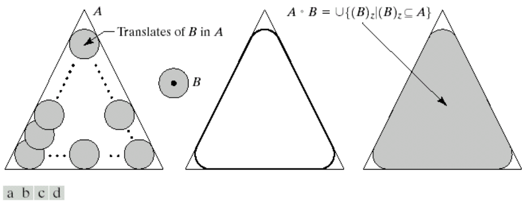


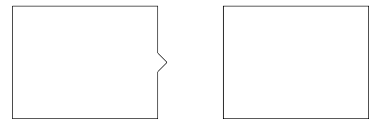
FIGURE 9.8 (a) Structuring element B "rolling" along the inner boundary of A (the dot indicates the origin of B). (c) The heavy line is the outer boundary of the opening. (d) Complete opening (shaded).

The boundary of the $A \circ B$ is established by the points in B that reach the farthest into the boundary of A as B is rolled around the **inside** of this boundary

$$A \circ B = \cup \{(B)_z \mid (B)_z \subseteq A\}$$

43

Binary Opening Example 1 -- Eliminate Thin Protrusions



Original Image A   Opening A by B

B
SE

44

Binary Opening Example 2 -- Break Narrow Connections

Original Image A   Opening A by B

B
SE

45

Binary Opening Example 3 -- Remove Pepper Noise

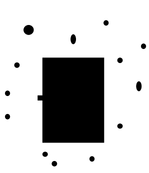
- Pepper noise consists of collections of small dark objects.
- Open operator can remove pepper noise.
- The trick is to find a structuring element which passes the image but eliminates the pepper noise.
- One often uses a symmetric structuring element so that results do not depend on the orientation of the picture.
- One often starts with small structuring elements B and increases the size systematically.

$$S \circ nB = (S \ominus nB) \oplus nB$$

$$nB = \underbrace{B \oplus B \oplus \ldots\ldots \oplus B}_{n\ times}$$

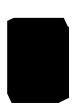
46

Binary Opening Example 3 -- Remove Pepper Noise (Cont.)



A

B

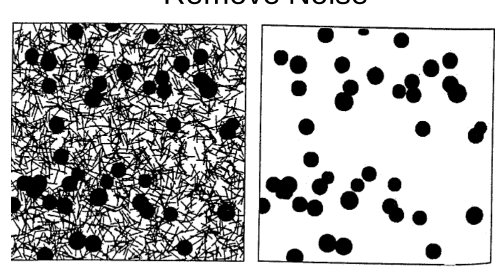


Opening A by B

Can we use a square SE to remove pepper noise by using the opening operation?

47

Binary Opening Example 4 -- Remove Noise



(a) A binary image (b) Opening of (a) with a disc SE

48

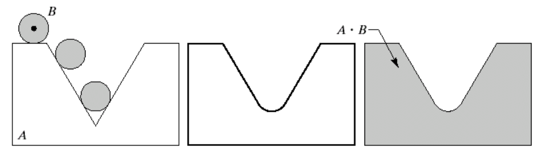
Binary Closing

- The closing of A by B is defined as:

$$A \bullet B = (A \oplus B) \Theta B$$

- The closing of A by B is simply the dilation of A by B, followed by the erosion of the result by B.
- Closing also tends to smooth sections of contours, but it generally fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour.
- Remove much of the white pixel noise in the foreground (objects) of the image, giving a fairly clean image. (Salt Noise)

49



a b c

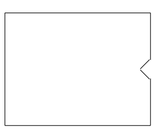
FIGURE 9.9 (a) Structuring element B "rolling" on the outer boundary of set A. (b) Heavy line is the outer boundary of the closing. (c) Complete closing (shaded).

Roll B on the **outside** of the boundary

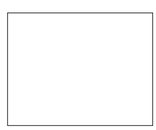
$$A \bullet B = \cup \{(B)_z \mid (B)_z \cap A \neq \phi\}$$

50

Binary Closing Example 1 -- Eliminate Small Holes



Original Image A



Closing A by B

B
SE

51

Binary Closing Example 2 -- Fuse Narrow Breaks and Long Thin Gulfs

Original Image A

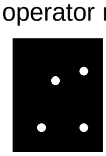
Closing A by B

B
SE

52

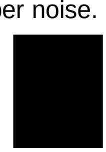
Binary Closing Example 3 -- Remove Salt Noise

- Salt noise consists of small white objects.
- The close operator will remove salt noise in a manner similar to the way the open operator removes pepper noise.



A

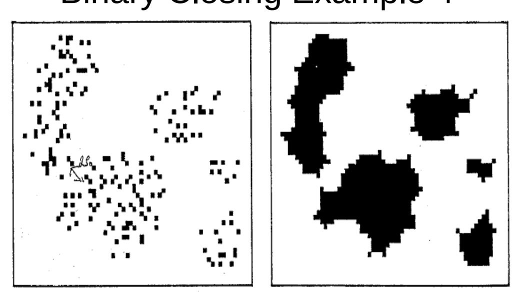
B



Closing A by B

53

Binary Closing Example 4



(a) A binary image with five clusters of points. Points within each cluster satisfy the partition property with the distance ρ0, and the clusters are farther from each other than 2ρ0 pixels.
(b) The image of a closed by a disk with a radius just greater than 2ρ0

54

Binary Opening vs. Binary Closing -- Original Image

	1	1	1	1	1	1	1			1	1	1	1	1	1	1	
	1	1	1	1	1	1	1			1	1	1	1	1	1	1	
	1	1	1	1	1	1	1			1	1	1					
				1	1	1	1			1	1	1					
	1	1	1	1	1	1	1	1	1	1	1	1					
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
	1	1	1	1	1	1	1			1	1	1	1	1	1	1	

55

Binary Opening vs. Binary Closing -- Opening (Break Narrow Connections)

	1	1	1	1	1	1	1			1	1	1					
	1	1	1	1	1	1	1			1	1	1					
	1	1	1	1	1	1	1			1	1	1					
				1	1	1	1			1	1	1					
	1	1	1	1	1	1	1			1	1	1					
	1	1	1	1	1	1	1			1	1	1					
	1	1	1	1	1	1	1			1	1	1					

56

Binary Opening vs. Binary Closing -- Closing (Fuse Narrow Breaks and Long Thin Gulfs)

	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
	1	1	1	1	1	1	1	1	1	1	1	1					
	1	1	1	1	1	1	1	1	1	1	1	1					
	1	1	1	1	1	1	1	1	1	1	1	1					
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

57

a
b c
d e
f g
h i

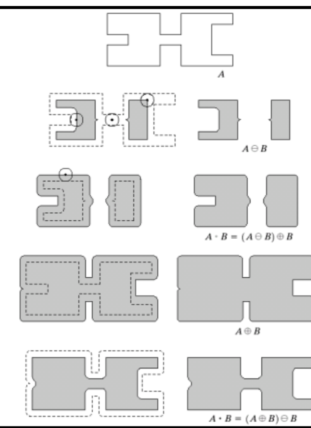
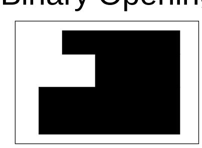


FIGURE 9.10 Morphological opening and closing. The structuring element is the small circle shown in various positions in (b). The dark dot is the center of the structuring element.

58

Relationship Between Binary Opening and Closing

$$(A \bullet B)^c = (A^c \circ \hat{B})$$

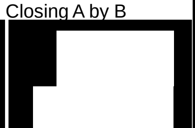

Original Image A


B(SE)


Closing A by B


Complement of image A


Open of image complement


Complement of Close

59

Properties of Binary Opening and Closing

- The opening operation satisfies the following properties:

1) $A \circ B$ is a subset (subimage) of A.

2) If C is a subset of D, then $C \circ B$ is a subset of $D \circ B$

3) $(A \circ B) \circ B = A \circ B$

- The closing operation satisfies the following properties:

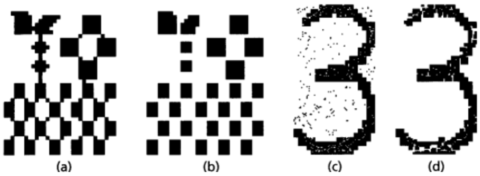
1) A is a subset (subimage) of $A \bullet B$.

2) If C is a subset of D, then $C \bullet B$ is a subset of $D \bullet B$

3) $(A \bullet B) \bullet B = A \bullet B$

60

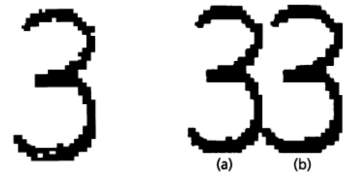
Application of Opening and Closing



(a) An image having many connected objects, (b) Objects can be isolated by opening using the simple structuring element, (c) An image that has been subjected to noise, (d) The noisy image after opening showing that the black noise pixels have been removed.

61

Application of Opening and Closing (Cont.)



The result of closing Figure (d) using the simple structuring element

Multiple closings for outline smoothing. (a) after a depth 2 closing, (b) after a depth 3 closing.

62

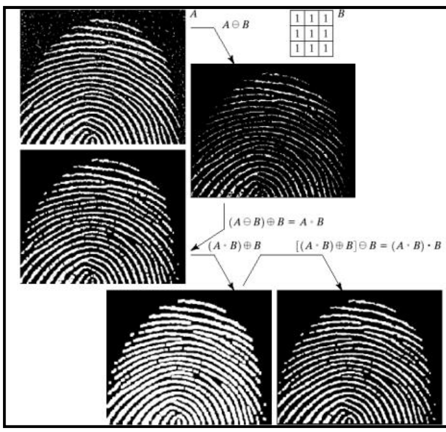
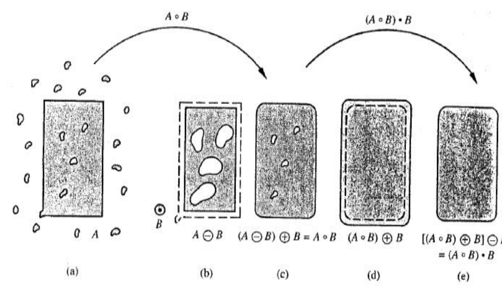


FIGURE 9.11 (a) Noisy image. (c) Eroded image. (d) Opening of A. (d) Dilation of the opening. (e) Closing of the opening. (Original image for this example courtesy of the National Institute of Standards and Technology.)



Morphological filtering: (a) original, noisy image; (b) result of erosion; (c) opening of A; (d) result or performing dilation on the opening; (e) final result showing the closing of the opening.

64

Hit-or-Miss Transform

- The hit-or-miss transform is a basic tool for shape detection or object recognition.
- The hit-and-miss transform is a general binary morphological operation that can be used to look for particular patterns of foreground and background pixels in an image.
- The hit-or-miss transform is defined as:

  Let B ={B1, B2}, where B1 is the set formed from elements of B associated with an object and B2 is the set of elements of B associated with the corresponding background, where B1 and B2 are disjoint.

$$A \circledast B = (A \,\Theta\, B1) \cap (A^c \,\Theta\, B2)$$
$$= (A \,\Theta\, B1) - (A \oplus \hat{B}2)$$

65

- A pixel belonging to an object is preserved by the hit-or-miss transform if and only if B1 translated to that pixel fits inside the object and B2 translated to that pixel fits outside the object.
- B1 and B2 cannot intersect, otherwise it would be impossible for both fits to occur simultaneously.

66

Four SEs used for corner finding in binary images using the hit-and-miss transform.

	1	
0	1	1
0	0	

	1	
1	1	0
	0	0

	0	0
1	1	0
	1	

0	0	
0	1	1
	1	

If the foreground and background pixels in the structuring element *exactly match* foreground and background pixels in the image, then the pixel underneath the origin of the structuring element is set to the foreground color. If it doesn't match, then that pixel is set to the background color.

67

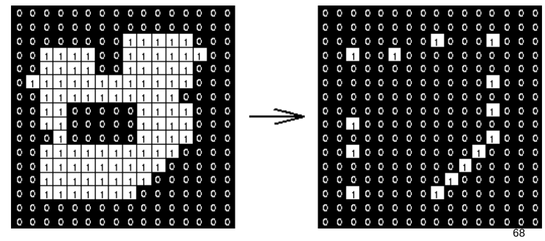
Four SEs used for **corner finding** in binary images using the hit-and-miss transform.

	1	
0	1	1
0	0	

	1	
1	1	0
	0	0

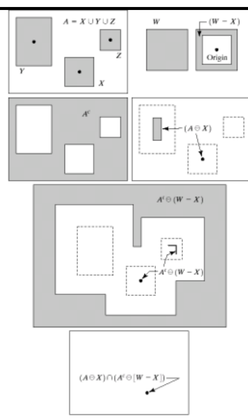
	0	0
1	1	0
	1	

0	0	
0	1	1
	1	



68

The Graphical Description of the Hit-or-Miss Transform



69

Shape Detection
-- Find the character ‘n’

B1 =

1 1 0 1 1 1 1 0 0
1 1 1 1 1 1 1 1 0
1 1 1 0 0 0 1 1 1
1 1 0 0 0 0 0 1 1
1 1 0 0 0 0 0 1 1
1 1 0 0 **0** 0 0 1 1
1 1 0 0 0 0 0 1 1
1 1 0 0 0 0 0 1 1
1 1 0 0 0 0 0 1 1
1 1 0 0 0 0 0 1 1
1 1 0 0 0 0 0 1 1

B2 =

1 1 1 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 1 1
1 0 0 0 0 0 0 0 0 1 1
1 0 0 0 1 1 1 0 0 0 1
1 0 0 1 1 1 1 1 0 0 1
1 0 0 1 1 1 1 1 0 0 1
1 0 0 1 1 **1** 1 1 0 0 1
1 0 0 1 1 1 1 1 0 0 1
1 0 0 1 1 1 1 1 0 0 1
1 0 0 1 1 1 1 1 0 0 1
1 0 0 1 1 1 1 1 0 0 1
1 0 0 1 1 1 1 1 0 0 1
1 1 1 1 1 1 1 1 1 1 1

70