

## Morphological Image Processing

- Morphology is concerned with image analysis methods whose outputs describe image content (i.e. extract “meaning” from an image).
- Mathematical morphology is a tool for extracting image components that can be used to represent and describe region shapes such as boundaries and skeletons.
- Morphological methods include filtering, thinning and pruning. These techniques are based on set theory. All morphology functions are defined for binary images, but most have natural extension to grayscale images.

## Basic Concepts of Set Theory

A set is specified by the elements between two braces: { }. The elements of the sets are the coordinates  $(x,y)$  of pixels representing objects or other features in an image.

Let  $A$  be a set in 2D image space  $Z^2$ :

- If  $a = (a_1, a_2)$  is an element of  $A$ , then  $a \in A$
- If  $a$  is not an element of  $A$ , then  $a \notin A$
- Empty set is a set with no elements and is denoted by  $\emptyset$
- If every element of a set  $A$  is also an element of another set  $B$ , then  $A$  is said to be a subset of  $B$ , denoted as  $A \subseteq B$
- The union of two sets  $A$  and  $B$ , denoted by  $C = A \cup B$
- The intersection of two sets  $A$  and  $B$ , denoted by  $C = A \cap B$
- Two sets  $A$  and  $B$  are said to be disjoint, if they have no common elements. This is denoted by  $A \cap B = \emptyset$

- The *complement* of a set  $A$  is the set of elements not contained in  $A$ . This is denoted by  $A^c = \{w \mid w \notin A\}$
- The *difference* of two sets  $A$  and  $B$ , denoted  $A - B$ , is defined as  $A - B = \{w \mid w \in A, w \notin B\} = A \cap B^c$
- The *reflection* of set  $B$ , denoted  $\hat{B}$ , is defined as  $\hat{B} = \{w \mid w = -b, \text{ for } b \in B\}$
- The *translation* of set  $A$  by point  $z = (z_1, z_2)$ , denoted  $(A)_z$  is defined as  $(A)_z = \{c \mid c = a + z, \text{ for } a \in A\}$

The figure below illustrates the preceding concepts.

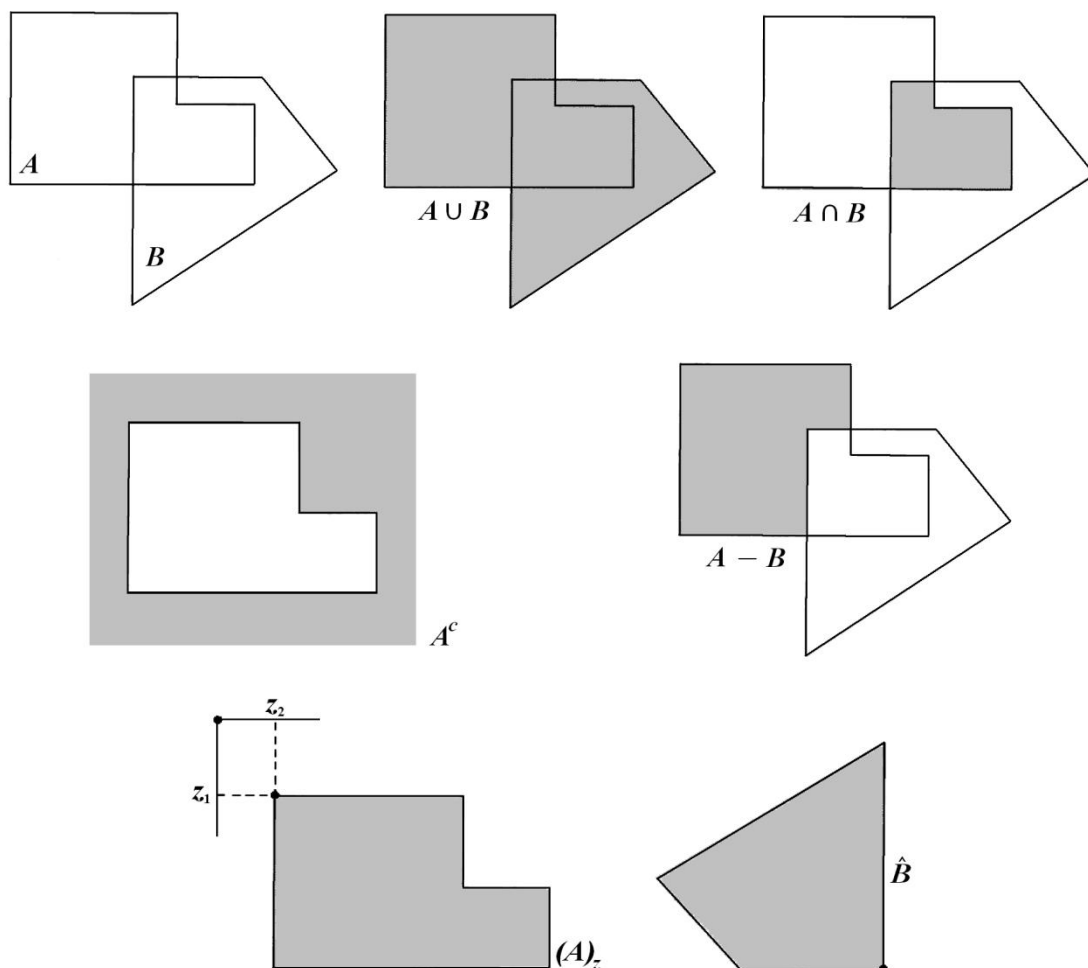


Figure 11.1 Basic concepts of Set Theory

## Logic Operations Involving Binary Images

A binary image is an image whose pixel values are 0 (representing black) or 1 (representing white, i.e. 255). The usual set operations of complement, union, intersection, and difference can be defined easily in terms of the corresponding logic operations NOT, OR and AND. For example:

- Intersection operation  $\cap$  is implemented by AND operation
- Union operation  $\cup$  is implemented by OR operation

The figure below shows an example of using logic operations to perform set operations on two binary images.

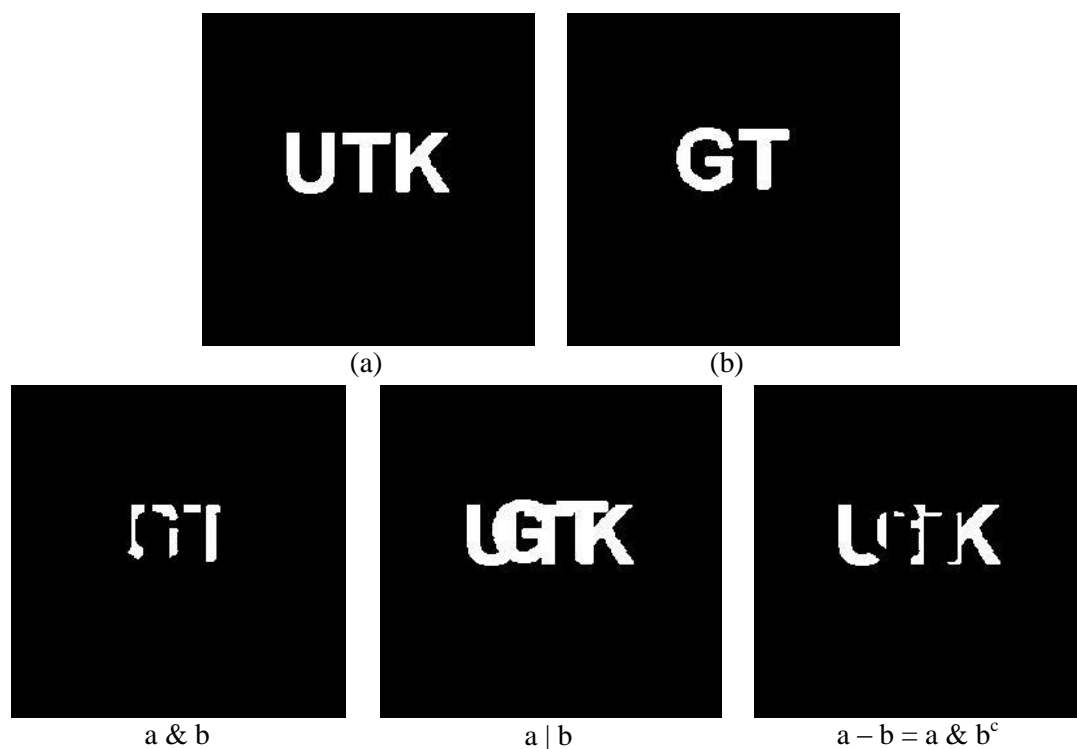


Figure 11.2 Using logic operations for applying set operations on two binary images

## Structuring Element

A morphological operation is based on the use of a filter-like binary pattern called the structuring element of the operation. Structuring element is represented by a matrix of 0s and 1s; for simplicity, the zero entries are often omitted.

Symmetric with respect to its origin:

Lines:

$$\begin{array}{|c|c|c|c|c|} \hline 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & \boxed{1} & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 \\ \hline \end{array} = \begin{array}{c} 1 \\ 1 \\ \boxed{1} \\ 1 \\ 1 \end{array} \quad \begin{array}{c} 1 \\ 1 \\ \boxed{1} \\ 1 \\ 1 \end{array}$$

Diamond:

$$\begin{array}{ccc} 0 & 1 & 0 \\ 1 & \boxed{1} & 1 \\ 0 & 1 & 0 \end{array}$$

Non-symmetric:

$$\begin{array}{ccccc} & & & & 1 \\ 1 & 1 & \boxed{1} & 1 & 1 \\ 1 & 1 & & & \\ 1 & & & & \end{array} \xrightarrow{\text{Reflection on origin}} \begin{array}{ccccc} & & & & 1 \\ & & & & 1 \\ & 1 & 1 & 1 & \boxed{1} \\ 1 & & & & 1 \end{array}$$

## Dilation

Dilation is an operation used to grow or thicken objects in binary images.

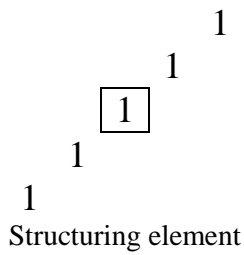
The dilation of a binary image  $A$  by a structuring element  $B$  is defined as:

$$A \oplus B = \{ z : (\hat{B})_z \cap A \neq \emptyset \}$$

This equation is based on obtaining the reflection of  $B$  about its origin and translating (shifting) this reflection by  $z$ . Then, the dilation of  $A$  by  $B$

is the set of all structuring element origin locations where the reflected and translated  $B$  overlaps with  $A$  by at least one element.

**Example:** Use the following structuring element to dilate the binary image below.

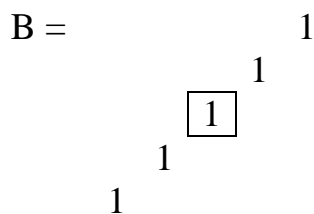


0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

Binary image

**Solution:**

We find the reflection of B:

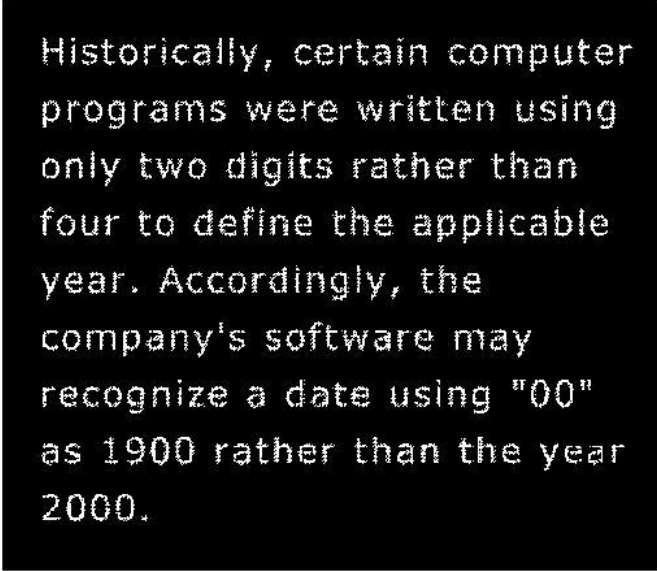


In this case  $\hat{B} = B$

$A \oplus B =$

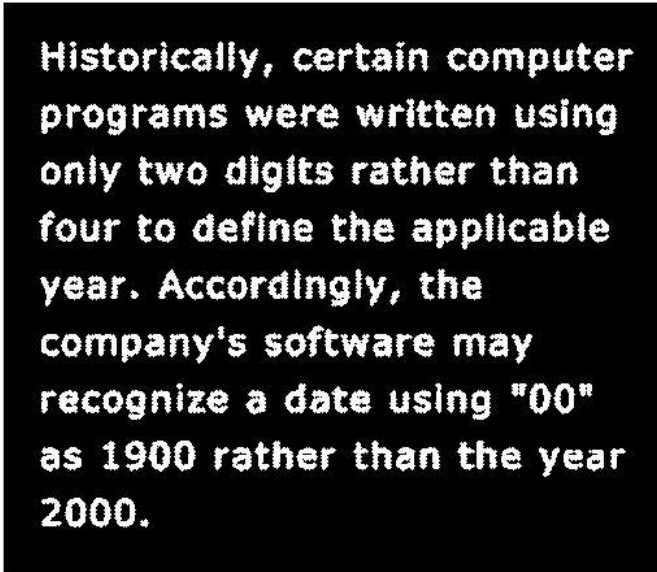
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	0	1	1	1	1	1	1	1	0
0	0	0	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

Dilation can be used for bridging gaps, for example, in broken/unclear characters as shown in the figure below.

A binary image showing a block of text on a black background. The text is rendered in white pixels, but there are significant gaps and missing pixels, particularly in the middle of the words, making it difficult to read. The text reads: "Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000."

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

(a)

A binary image showing the same block of text as in (a), but after a dilation operation. The white pixels are now thicker and more solid, filling in the gaps and making the text much clearer and easier to read. The text reads: "Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000."

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

(b)

Figure 11.3 (a) Broken-text binary image. (b) Dilated image.

## Erosion

Erosion is used to shrink or thin objects in binary images. The erosion of a binary image  $A$  by a structuring element  $B$  is defined as:

$$A \ominus B = \{ z : (B)_z \cap A^c \neq \emptyset \}$$

The erosion of  $A$  by  $B$  is the set of all structuring element origin locations where the translated  $B$  does not overlap with the background of  $A$ .

**Example:** Use the following structuring element to erode the binary image below.

$$\begin{array}{c} 1 \\ \boxed{1} \\ 1 \end{array}$$
 Structuring element

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	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	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

Binary image

## Solution

$A \ominus B =$

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

Erosion can be used to remove isolated features (i.e. irrelevant detail) which may include noise or thin edges as shown in the figure below.

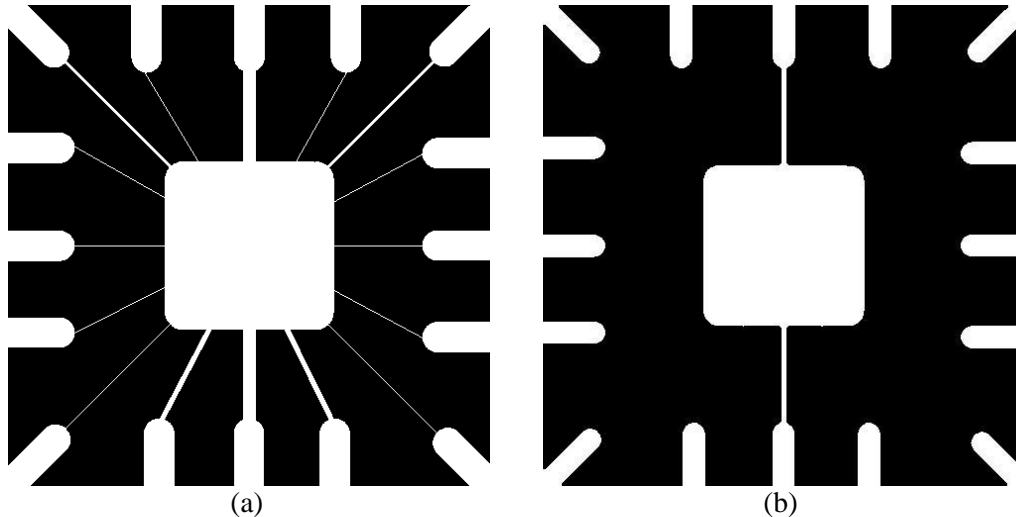


Figure 11.4 (a) Binary image. (b) Eroded image.

### Combining Dilation & Erosion - Opening Morphology

The opening operation erodes an image and then dilates the eroded image using the same structuring element for both operations, i.e.

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

where  $A$  is the original image and  $B$  is the structuring element.

The opening operation is used to remove regions of an object that cannot contain the structuring element, smooth objects contours, and breaks thin connections as shown in the figure below.



(a)



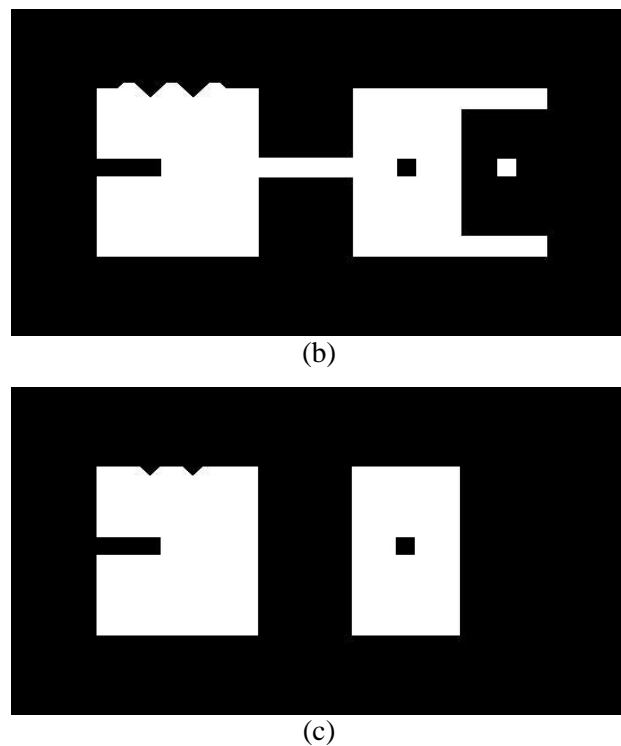


Figure 11.5 (a) Original binary image. (b) Result of opening with square structuring element of size 10 pixels. (c) Result of opening with square structuring element of size 20 pixels.

The opening operation can also be used to remove small objects in an image while preserving the shape and size of larger objects as illustrated in the figure below.

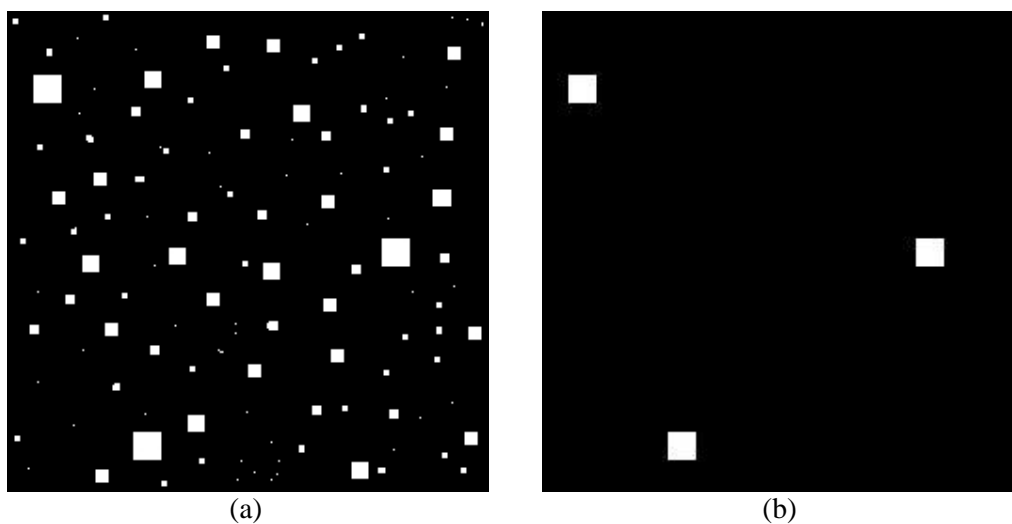


Figure 11.6 (a) Original binary image. (b) Result of opening with square structuring element of size 13 pixels.

## Combining Dilation & Erosion - Closing Morphology

The closing operation dilates an image and then erodes the dilated image using the same structuring element for both operations, i.e.

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

where  $A$  is the original image and  $B$  is the structuring element.

The closing operation fills holes that are smaller than the structuring element, joins narrow breaks, fills gaps in contours, and smoothes objects contours as shown in the figure below.



(a)



(b)

Figure 11.7 (a) Result of closing with square structuring element of size 10 pixels. (c) Result of closing with square structuring element of size 20 pixels.

## Combining Opening & Closing Morphology

Combining opening and closing can be quite effective in removing noise as illustrated in the next figure.



Figure 11.8 (a) Noisy fingerprint. (b) Result of opening (a) with square structuring element of size 3 pixels. (c) Result of closing (b) with the same structuring element.

Note that the noise was removed by opening the image, but this process introduced numerous gaps in the ridges of the fingerprint. These gaps can be filled by following the opening with a closing operation.