



# **Digital Image Processing**

**Lecture #4**

**Ming-Sui (Amy) Lee**



# **Morphological Image Processing**

# [Morphological Processing]

- **Morphology**
  - Morpho-: shape/form/structure
  - -ology: study
- **Morphological image processing**
  - Post-processing
  - Binary images → gray-level image



# [Morphological Processing]

- For some applications
  - Structures of objects composed by lines or arcs
  - Care about the pattern connectivity
  - Independent of width



Hand-written characters



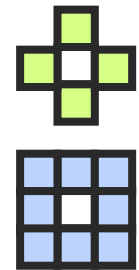
Fingerprint patterns

# Morphological Processing

## ■ Binary image connectivity

### ○ Pixel bond

- Specify the connectivity of a pixel with its neighbors
- Four-connected neighbor  $\rightarrow$  bond = 2
- Eight-connected neighbor  $\rightarrow$  bond = 1




### ○ Minimally connected

- Elimination of any black pixel (except boundary pixels) results in disconnection of the remaining black pixels

# Morphological Processing

## ■ Binary hit or miss transformations

- Select a  $n \times n$  hit pattern (odd-sized mask)
- Compare with a  $n \times n$  image window
  -  Match → hit → change the central pixel value
  - Otherwise → miss → do nothing

## ○ Example

- To clean the isolated binary noise

0	0	0
0	1	0
0	0	0

Hit or miss?



# Morphological Processing

## Binary hit or miss transformations

■ 0 → background

■ 1 → object

0	0	0
0	1	0
0	0	0

Hit or miss?

## Logical expression

$$\begin{bmatrix} X_3 & X_2 & X_1 \\ X_4 & X & X_0 \\ X_5 & X_6 & X_7 \end{bmatrix}$$



$$G(j,k) = X \cap (X_0 \cup X_1 \cup \dots \cup X_7)$$

## Example



○ If  $G(j,k) = X \cap 1 \rightarrow$  do nothing



○ If  $G(j,k) = X \cap 0$

■ If  $X=0 \rightarrow G(j,k)=0 \rightarrow$  do nothing

■ If  $X=1 \rightarrow$  hit  $\rightarrow G(j,k)=0$

# Morphological Processing

## Binary hit or miss transformations

$$G(j,k) = X \cap (X_0 \cup X_1 \cup \dots \cup X_7)$$

$\Rightarrow 2^9$  possible mask patterns

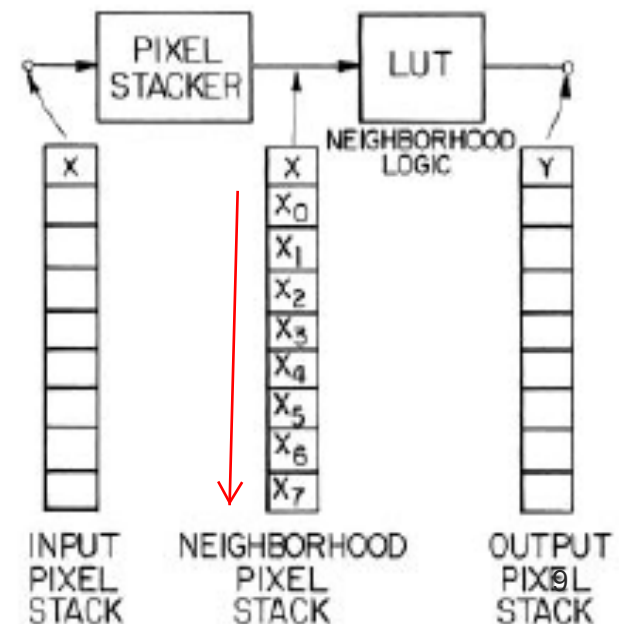
## Implementation

### Pixel stack

- Treat the 8 neighboring pixels as a “byte”

$$\begin{bmatrix} X_3 & X_2 & X_1 \\ X_4 & X & X_0 \\ X_5 & X_6 & X_7 \end{bmatrix} \otimes \begin{bmatrix} 2^{-4} & 2^{-3} & 2^{-2} \\ 2^{-5} & 2^0 & 2^{-1} \\ 2^{-6} & 2^{-7} & 2^{-8} \end{bmatrix}$$

### Look-Up-Table (LUT)



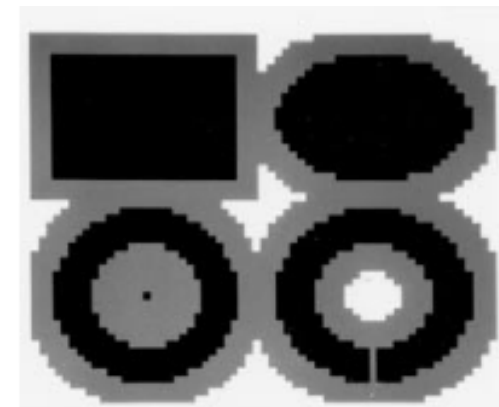
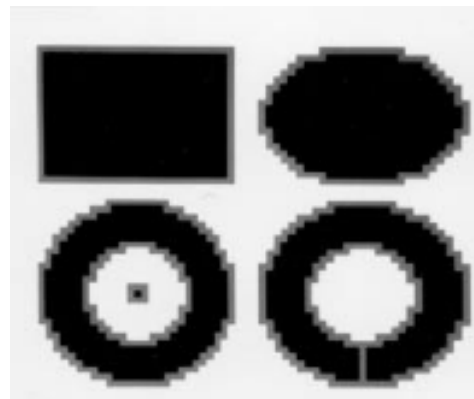
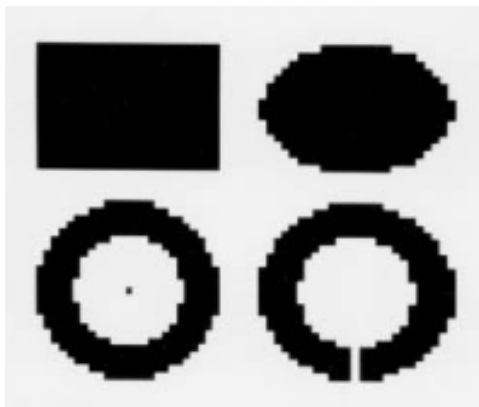


# Morphological Processing

- Simple morphological processing based on binary hit or miss rules

- Additive operators ( $0 \rightarrow 1$ )

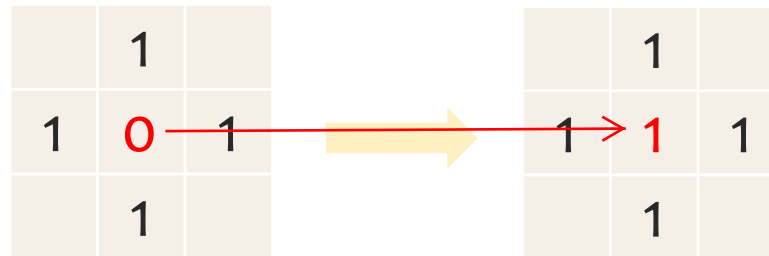
- Interior fill
- Diagonal fill
- Bridge
- 8-neighbor dilate



# Morphological Processing

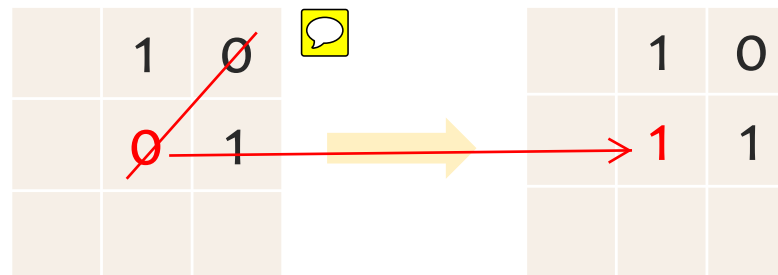
## ○ Interior fill

- Create a pixel if all **four-connected** neighbor pixels are white



## ○ Diagonal fill

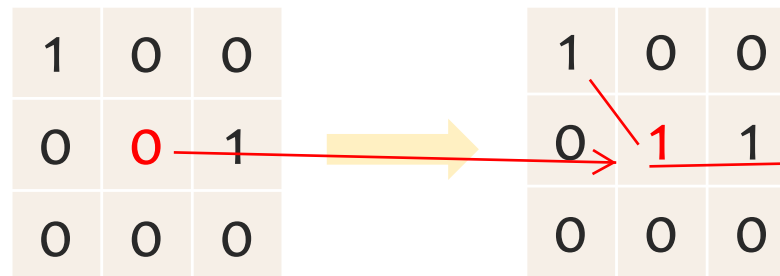
- Create a pixel if creation eliminates the **eight-connectivity** of the background



# Morphological Processing

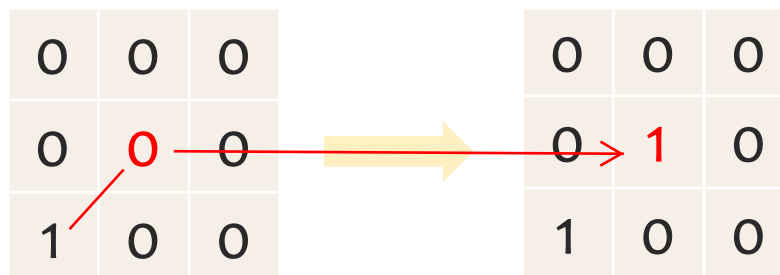
## ○ Bridge

- Create a white pixel if creation results in connectivity of previously unconnected neighboring white pixels



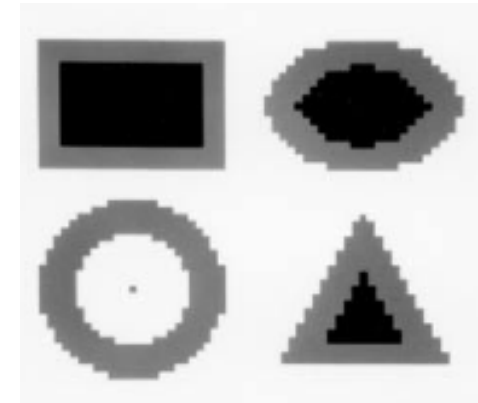
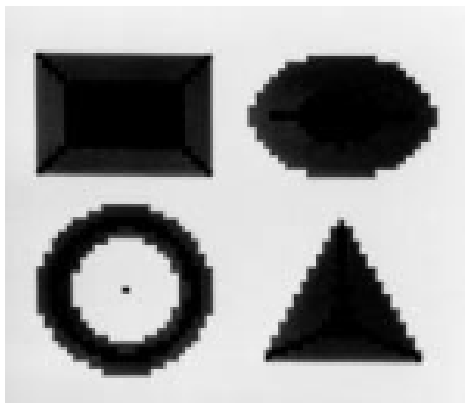
## ○ 8-neighbor dilate

- Create a black pixel if at least one eight-connected neighbor pixel is white



# Morphological Processing

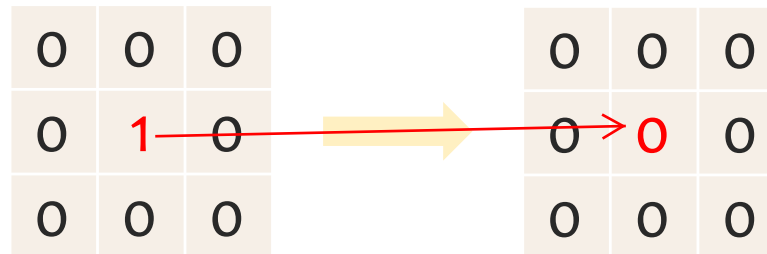
- Simple morphological processing based on binary hit or miss rules
  - **Subtractive** operators ( $1 \rightarrow 0$ )
    - **Isolated** pixel removal
    - **Spur** removal
    - **Interior** pixel removal
    - H-break / Eight-neighbor **erode**



# Morphological Processing

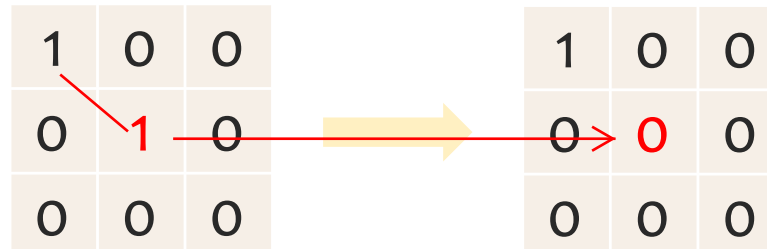
- **Isolated pixel removal**

- Erase a white pixel with eight black neighbors



- **Spur removal**

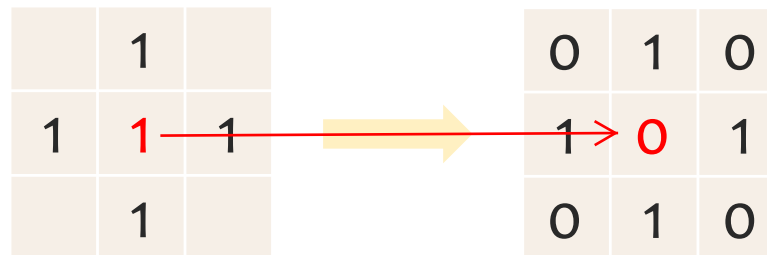
- Erase a white pixel with a single eight-connected neighbor



# Morphological Processing

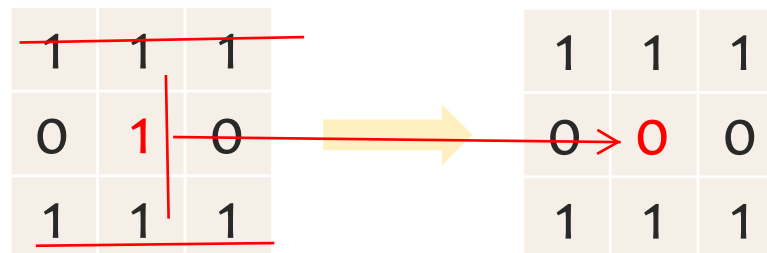
## ○ Interior pixel removal

- Erase a white pixel if all four-connected neighbors are white



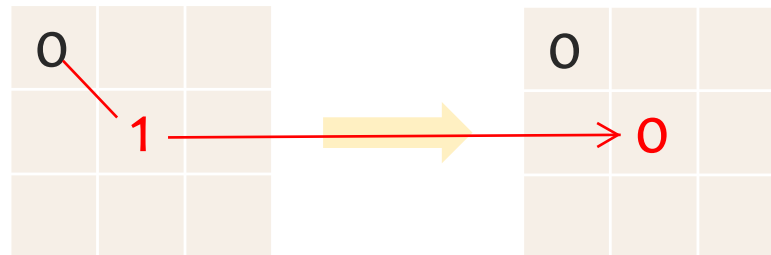
## ○ H-break

- Erase a white pixel that is H-connected



# [Morphological Processing]

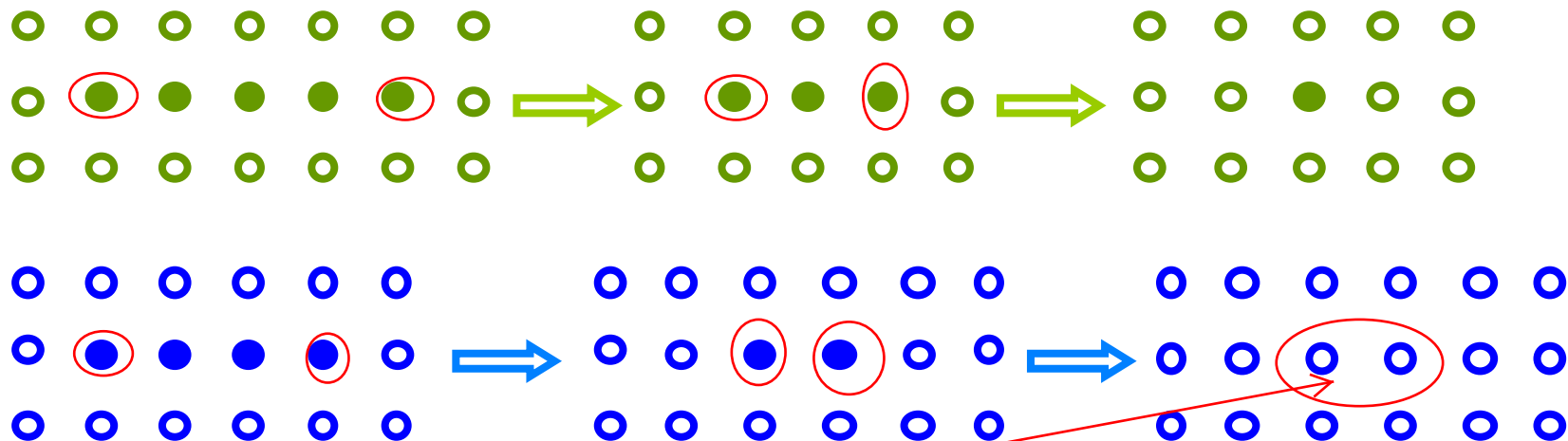
- **Eight-neighbor erode**
  - Erase a white pixel if at least one eight-connected neighbor pixel is black



# Morphological Processing

## ■ Example

### ○ Subtractive operator



- doesn't prevent total erasure and ensure connectivity
- In this case, only a 3x3 window does not sufficient to tell whether the final stage of iteration is reached or not



# [Morphological Processing]

## ■ Solutions

### ○ Approach I

#### ■ Apply a filter with larger size

- “fairly complicated patterns”, “many combinations”

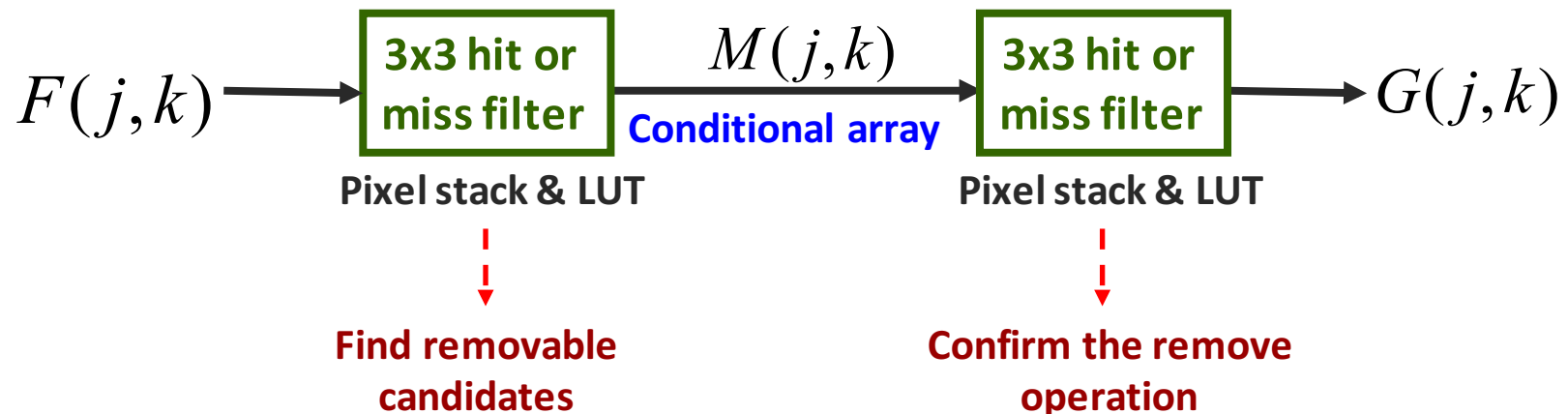
### ○ Approach II

#### ■ Consider a structural (composite) design with 3x3 filters: two-stage approach

- Application dependent
- Thinning, shrinking, skeletonizing
- Share the same structure but vary in some modular details

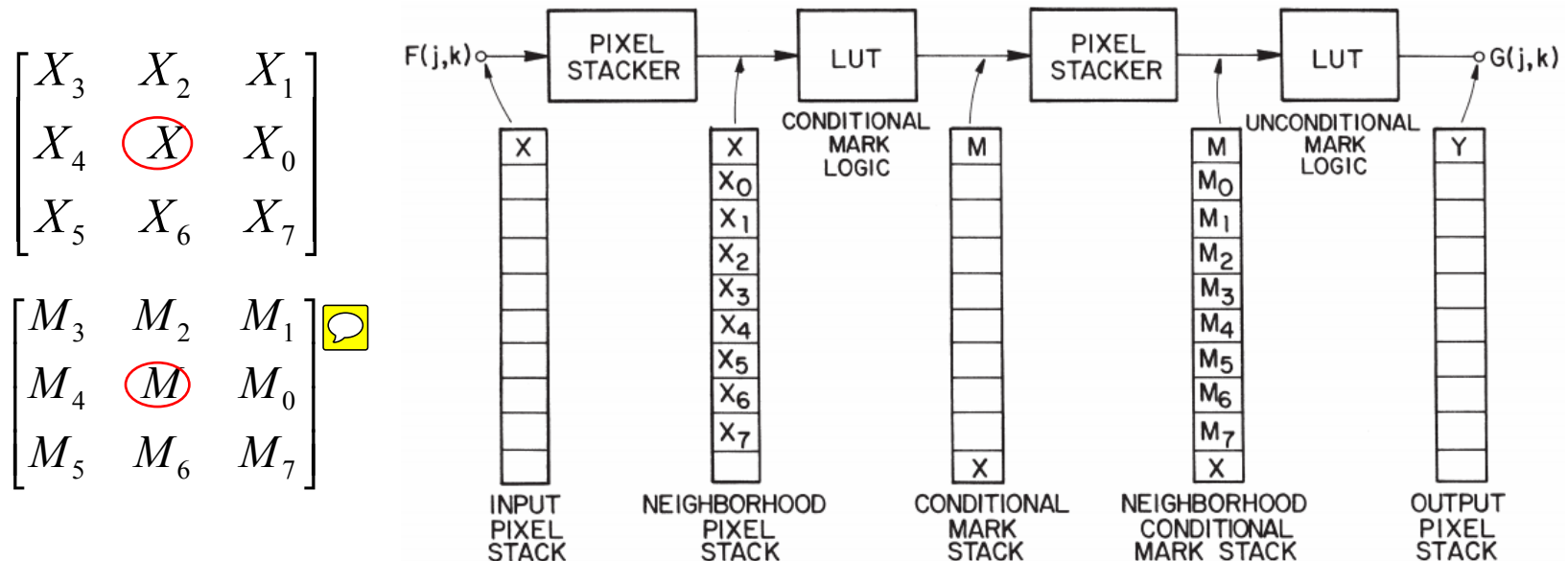
# Morphological Processing

- Advanced morphological processing
  - Shrinking/Thinning/Skeletonizing
    - Conditional erosion
    - Prevent total erasure & Ensure connectivity



# Morphological Processing

- Advanced morphological processing
  - Shrinking/Thinning/Skeletonizing
    - Conditional erosion
    - Prevent total erasure & Ensure connectivity



# Morphological Processing

## ■ Shrinking/Thinning/Skeletonizing

### ○ Stage I

- Generate a binary image  $M(j,k)$  called the conditional array (or mask)

- If  $M(j,k)=1$ , it means  $(j,k)$  is a candidate for erasure
- If  $M(j,k)=0$ , it means no further operation is needed on  $(j,k)$

### ○ Stage II

- Based on the center pixel,  $X$ , and  $M(j,k)$  pattern, we decide whether to erase  $X$  or not in the output  $G(j,k)$ 
  - If there's a hit → do nothing
  - If there's a miss → erase the center pixel

# Morphological Processing

## ■ Stage I → Part of Table 14.3-1



TABLE 14.3-1. Shrink, Thin and Skeletonize Conditional Mark Patterns [ $M = 1$  if hit]

Table	Bond	Pattern							
<i>S</i>	1	0 0 1	1 0 0	0 0 0	0 0 0				
		0 1 0	0 1 0	0 1 0	0 1 0				
		0 0 0	0 0 0	1 0 0	0 0 1				
<i>S</i>	2	0 0 0	0 1 0	0 0 0	0 0 0				
		0 1 1	0 1 0	1 1 0	0 1 0				
		0 0 0	0 0 0	0 0 0	0 1 0				
<i>S</i>	3	0 0 1	0 1 1	1 1 0	1 0 0	0 0 0	0 0 0	0 0 0	0 0 0
		0 1 1	0 1 0	0 1 0	1 1 0	1 1 0	0 1 0	0 1 0	0 1 1
		0 0 0	0 0 0	0 0 0	0 0 0	1 0 0	1 1 0	0 1 1	0 0 1
<i>TK</i>	4	0 1 0	0 1 0	0 0 0	0 0 0				
		0 1 1	1 1 0	1 1 0	0 1 1				
		0 0 0	0 0 0	0 1 0	0 1 0				
<i>STK</i>	4	0 0 1	1 1 1	1 0 0	0 0 0				
		0 1 1	0 1 0	1 1 0	0 1 0				
		0 0 1	0 0 0	1 0 0	1 1 1				

**Bond: classification, narrow down the search space**

**Pattern: coded as an 8-bit symbol for a filter**

$$\begin{bmatrix} X_3 & X_2 & X_1 \\ X_4 & X & X_0 \\ X_5 & X_6 & X_7 \end{bmatrix} \otimes \begin{bmatrix} 2^{-4} & 2^{-3} & 2^{-2} \\ 2^{-5} & 2^0 & 2^{-1} \\ 2^{-6} & 2^{-7} & 2^{-8} \end{bmatrix}$$

# Morphological Processing

## ■ Stage II → Part of Table 14.3-2

TABLE 14.3-2. Shrink and Thin Unconditional Mark Patterns

$[P(M, M_0, M_1, M_2, M_3, M_4, M_5, M_6, M_7) = 1 \text{ if hit}]^a$

				Pattern			
Spur				Single 4-connection			
0 0 M	M 0 0	0 0 0	0 0 0				
0 M 0	0 M 0	0 M 0	0 M M				
0 0 0	0 0 0	0 M 0	0 0 0				
L Cluster							
0 0 M	0 M M	M M 0	M 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 M M	0 M 0	0 M 0	M M 0	M M 0	0 M 0	0 M 0	0 M M
0 0 0	0 0 0	0 0 0	0 0 0	M 0 0	M M 0	0 M M	0 0 M
4-Connected offset							
0 M M	M M 0	0 M 0	0 0 M				
M M 0	0 M M	0 M M	0 M M				
0 0 0	0 0 0	0 0 M	0 M 0				

$$G(j, k) = X \cap [\overline{M} \cup P(M, M_1, \dots, M_7)]$$

where  $P(M, M_1, \dots, M_7)$  is an erasure inhibiting logical variable

$$\begin{bmatrix} M_3 & M_2 & M_1 \\ M_4 & M & M_0 \\ M_5 & M_6 & M_7 \end{bmatrix} \otimes \begin{bmatrix} 2^{-4} & 2^{-3} & 2^{-2} \\ 2^{-5} & 2^0 & 2^{-1} \\ 2^{-6} & 2^{-7} & 2^{-8} \end{bmatrix}$$

# Morphological Processing

## ■ Stage II → Part of Table 14.3-2 (cont'd)

Spur corner cluster

0	A	M	MB	0	0	0	M	M	0	0
0	MB	A	M	0	A	M	0	0	MB	
M	0	0	0	M	MB	0	0	A	M	

Corner cluster

*MMD*

*MMD*

*DDD*

Tee branch

<i>DM</i>	<i>0</i>	<i>MD</i>	<i>0</i>	<i>0</i>	<i>D</i>	<i>D</i>	<i>0</i>	<i>0</i>	<i>DMD</i>	<i>0</i>	<i>M</i>	<i>0</i>	<i>0</i>	<i>M</i>	<i>0</i>	<i>DMD</i>
<i>MMM</i>	<i>MMM</i>	<i>MMM</i>	<i>MMM</i>	<i>MMM</i>	<i>MM</i>	<i>0</i>	<i>MM</i>	<i>0</i>	<i>MM</i>	<i>0</i>	<i>MM</i>	<i>0</i>	<i>MM</i>	<i>0</i>	<i>MM</i>	<i>0</i>
<i>D</i>	<i>0</i>	<i>0</i>	<i>0</i>	<i>0</i>	<i>D</i>	<i>0</i>	<i>MD</i>	<i>DM</i>	<i>0</i>	<i>M</i>	<i>0</i>	<i>DMD</i>	<i>DMD</i>	<i>0</i>	<i>M</i>	<i>0</i>

$$A \cup B \cup C = 1, \quad D = 0 \cup 1, \quad A \cup B = 1$$

# Morphological Processing

## ■ Stage II → Part of Table 14.3-3

TABLE 14.3-3. Skeletonize Unconditional Mark Patterns

$[P(M, M_0, M_1, \underline{M_2}, \underline{M_3}, \underline{M_4}, M_5, M_6, M_7) = 1 \text{ if hit}]^a$       $A \cup B \cup C = 1, \quad D = 0 \cup 1$

Pattern											
Spur											
0	0	0	0	0	0	0	0	$M$	$M$	0	0
0	$M$	0	0	$M$	0	0	$M$	0	0	$M$	0
0	0	$M$	$M$	0	0	0	0	0	0	0	0
Single 4-connection											
0	0	0	0	0	0	0	0	0	0	$M$	0
0	$M$	0	0	$M$	$M$	$M$	$M$	0	0	$M$	0
0	$M$	0	0	0	0	0	0	0	0	0	0
L corner											
0	$M$	0	0	$M$	0	0	0	0	0	0	0
0	$M$	$M$	$M$	$M$	0	0	$M$	$M$	$M$	$M$	0
0	0	0	0	0	0	0	$M$	0	0	$M$	0



# [ Morphological Processing ]

## ■ Example - shrinking

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

$F(j,k)$

0	0	0	0	0	0
0	0	M	M	0	0
0	0	0	0	0	0

$M(j,k)$

0	0	0	0	0	0
0	0	P	0	0	0
0	0	0	0	0	0

$P(j,k)$

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

$G(j,k)$

# [ Morphological Processing ]

## ■ Example - shrinking

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

$F(j,k)$

$M(j,k)$

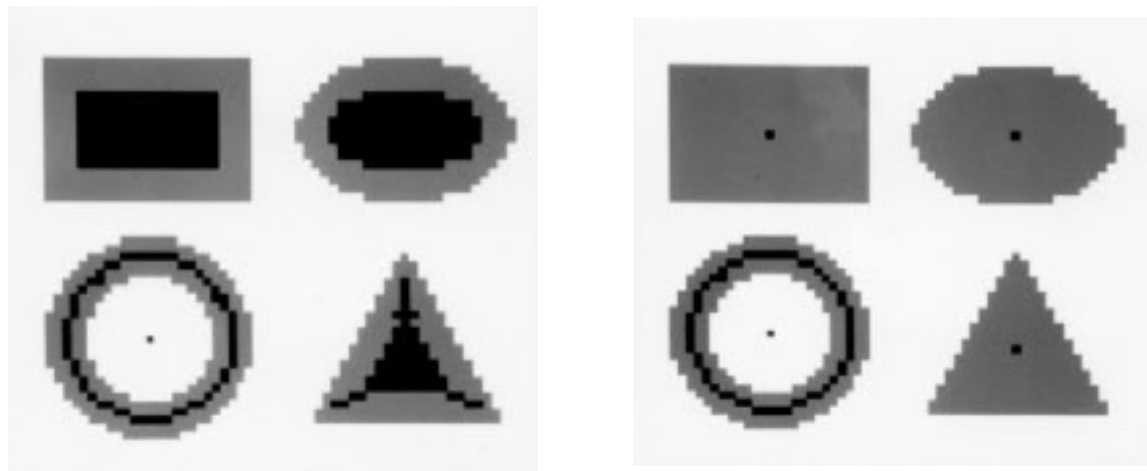
$P(j,k)$

$G(j,k)$

# Morphological Processing

## ■ Shrinking

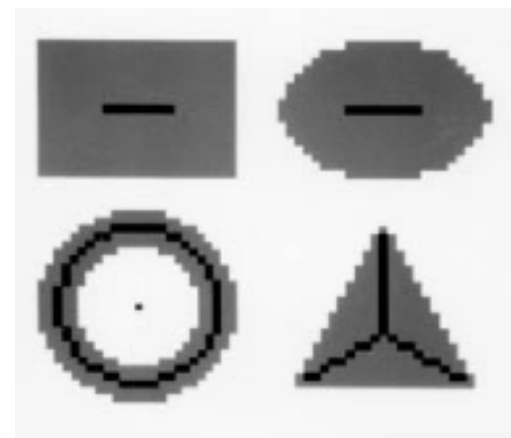
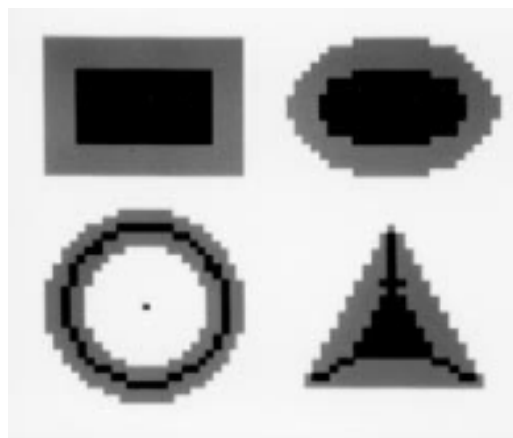
- Erase black pixels such that an object without holes erodes to a single pixel at or near its center of mass, and an object with holes erodes to a connected ring lying midway between each hole and its nearest outer boundary



# Morphological Processing

## ■ Thinning

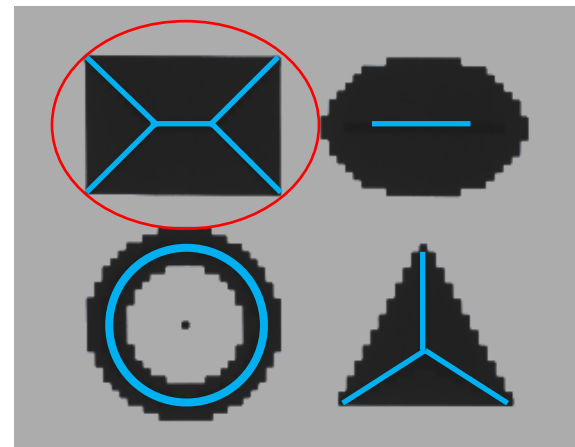
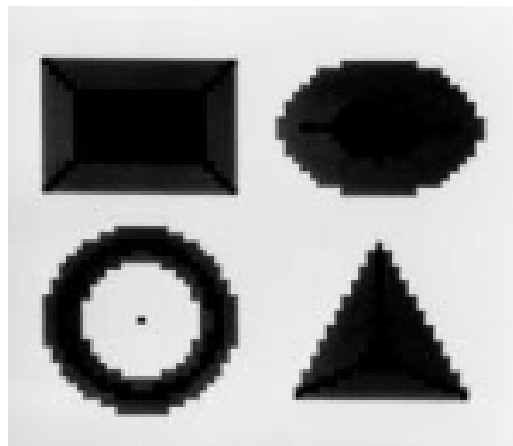
- Erase black pixels such that an object **without holes** erodes to a **minimally connected stroke** located **equidistant from its nearest outer boundaries**, and an object with holes erodes to a minimally connected ring midway between each hole and its nearest outer boundary



# [ Morphological Processing ]

## ■ Skeletonizing

- The **medial axis skeleton** consists of the set of points that are **equally distant** from **two closest** points of an object boundary



# [ Morphological Processing ]

- Algebraic operations on binary arrays

```

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

```

$A$

```

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

```

$B$

```

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

```

$\bar{A}$

complement

```

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

```

$A \cup B$

union

OR

```

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

```

$A \cap B$

intersection

AND

```

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

```

$A \oplus B$

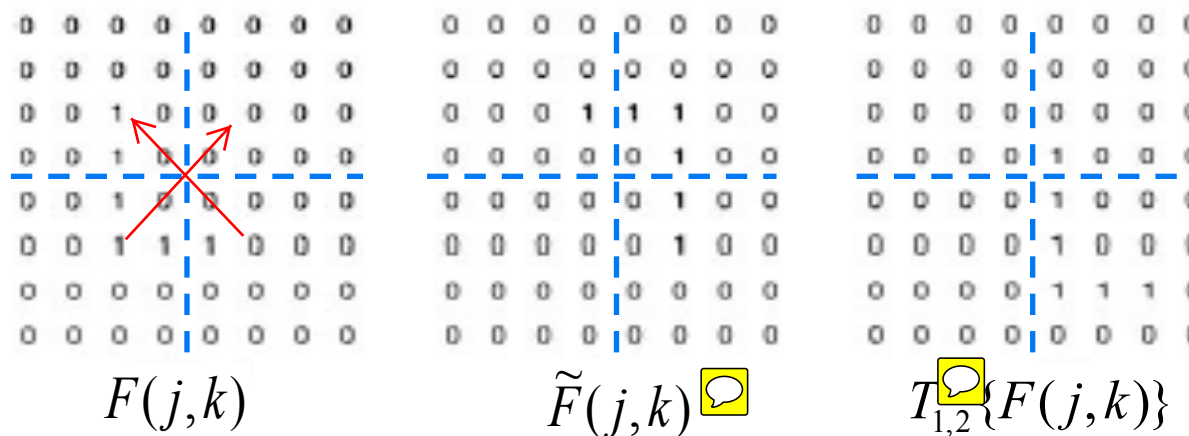
exclusive-OR

XOR

# Morphological Processing

## Generalized dilation and erosion

### Reflection and translation of a binary image



### Dilation

$$G(j,k) = F(j,k) \oplus H(j,k)$$

Structuring element

### Erosion

$$G(j,k) = F(j,k) \ominus H(j,k)$$

# [ Morphological Processing ]

## ■ Dilation $G(j,k) = F(j,k) \oplus H(j,k)$

- Can be implemented in several ways
- Minkowski addition definition

$$G(j,k) = \bigcup_{(r,c) \in H} T_{r,c} \{F(j,k)\}$$

$$G(j,k) = T_{0,0} \{F(j,k)\} \cup T_{0,1} \{F(j,k)\} \cup T_{1,0} \{F(j,k)\} \cup T_{1,1} \{F(j,k)\} \cup T_{2,0} \{F(j,k)\}$$

0 0 0 0 0	1 1 0				
0 0 1 0 0	1 1 0				
0 1 1 0 0	1 0 0				
0 0 1 1 0					
0 0 0 0 0					
$F(j,k)$	$H(j,k)$				
0 0 0 0 0	• 0 0 0 0 0	• • • • •	• • • • •	• • • • •	
0 0 1 0 0	• 0 0 1 0 0	0 0 0 0 0	• 0 0 0 0 0	• • • • •	
0 1 1 0 0	• 0 1 1 0 0	0 0 1 0 0	• 0 0 1 0 0	0 0 0 0 0	
0 0 1 1 0	• 0 0 1 1 0	0 1 1 0 0	• 0 1 1 0 0	0 0 1 0 0	
0 0 0 0 0	• 0 0 0 0 0	0 0 1 1 0	• 0 0 1 1 0	0 1 1 0 0	
		0 0 0 0 0	• 0 0 0 0 0	0 0 1 1 0	
				0 0 0 0 0	
$T_{0,0} \{F(j,k)\}$	$T_{0,1} \{F(j,k)\}$	$T_{1,0} \{F(j,k)\}$	$T_{1,1} \{F(j,k)\}$	$T_{2,0} \{F(j,k)\}$	
					0 0 0 0 0 0 0
					0 0 1 1 0 0 0
					0 1 1 1 0 0 0
					0 1 1 1 1 0 0
					0 1 1 1 1 0 0
					0 0 1 1 0 0 0
					0 0 0 0 0 0 0
					$G(j,k)$



# [ Morphological Processing ]

## ■ Erosion $G(j,k) = F(j,k) \ominus H(j,k)$

- Can be implemented in several ways
- Dual relationship of Minkowski addition

$$G(j,k) = \bigcap_{(r,c) \in H} T_{r,c} \{F(j,k)\}$$

$$\begin{array}{ccccc}
 1 & 1 & 1 & 1 & 1 \\
 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
 1 & 1 & 0 & 0 & 0 & \ominus & 1 & 0 & 0 & = & 1 & 1 & 0 \\
 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 & & & & & & & & \\
 F(j,k) & & H(j,k) & & G(j,k)
 \end{array}$$

$$\begin{array}{ccccc}
 1 & 1 & 1 & 1 & 1 \\
 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
 1 & 1 & 0 & 0 & 0 & \ominus & 1 & 0 & 0 & = & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 & & & & & & & \\
 F(j,k) & & H(j,k) & & G(j,k)
 \end{array}$$

//Sternberg definition//

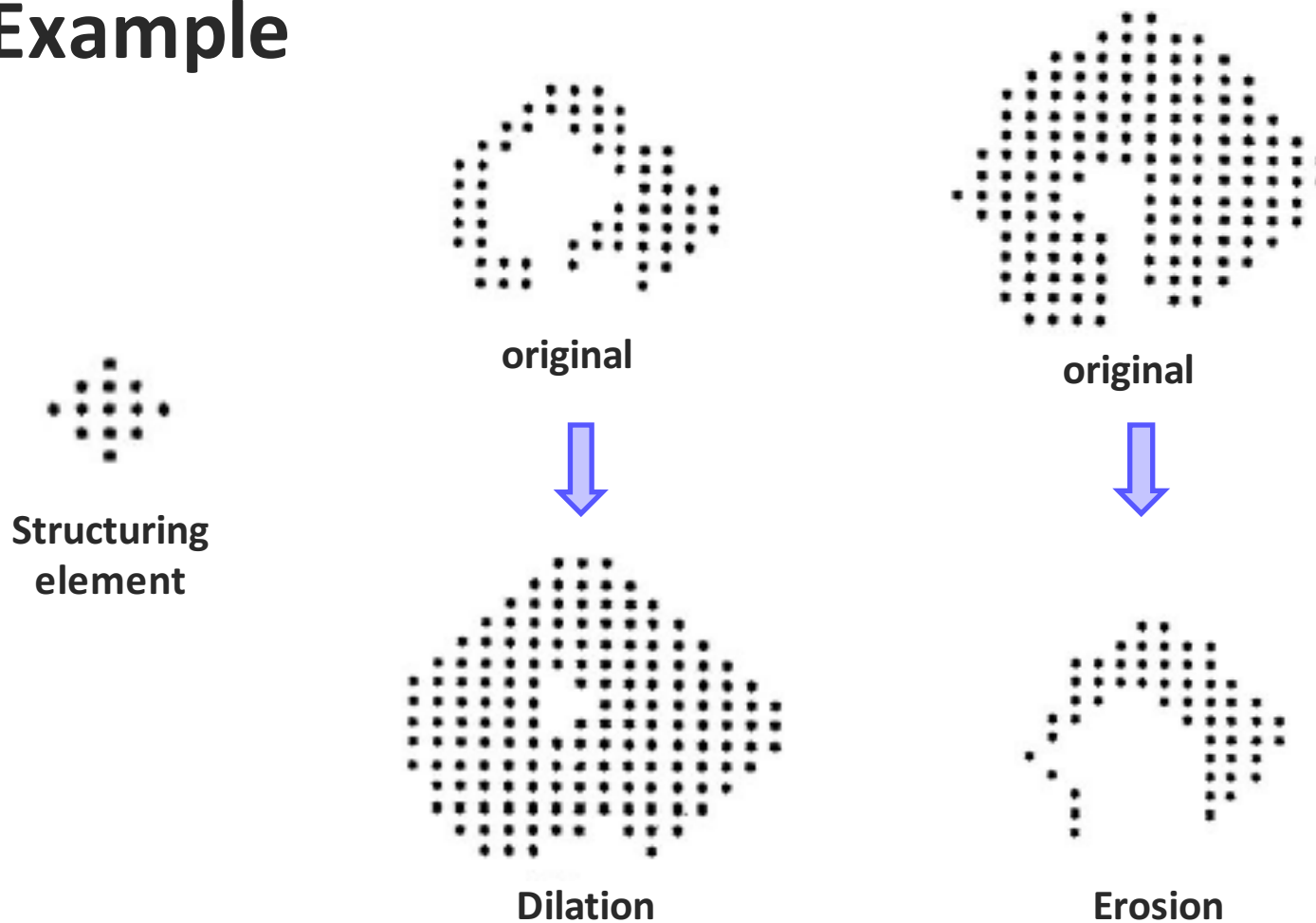
$$G(j,k) = \bigcap_{(r,c) \in H} T_{r,c} \{F(j,k)\}$$

//Serra definition//

$$G(j,k) = \bigcap_{(r,c) \in \tilde{H}} T_{r,c} \{F(j,k)\}$$

# Morphological Processing

## ■ Example



# Morphological Processing

## ■ Example

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.



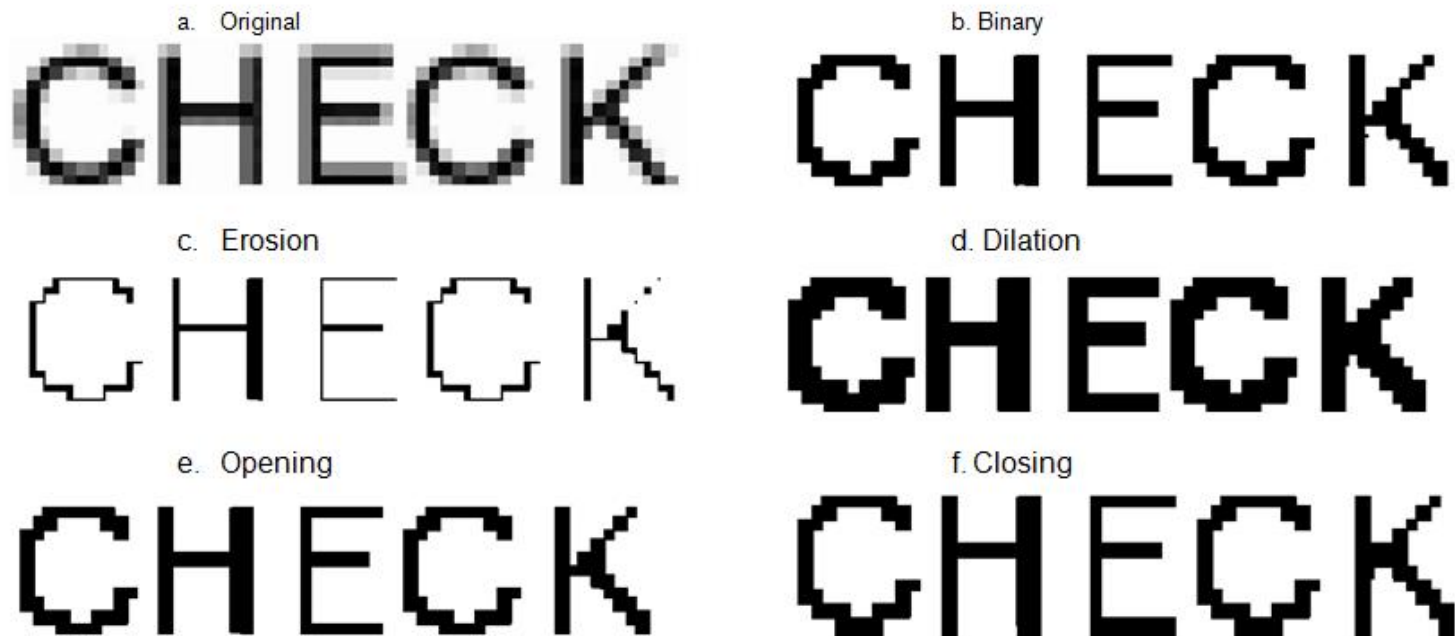
0	1	0
1	1	1
0	1	0

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.



# Morphological Processing

## ■ Example



# Morphological Processing

## ■ Example

Original fingerprint



Skeletonized fingerprint



The original fingerprint contains ridges with width of several pixels.  
The skeletonized fingerprint contains ridges only a single pixel wide.

# [Morphological Processing]

## ■ Applications

### ○ Boundary Extraction

- Extract the boundary (or outline) of an object

### ○ Hole Filling

- Given a pixel inside a boundary, hole filling attempts to fill that boundary with object pixels

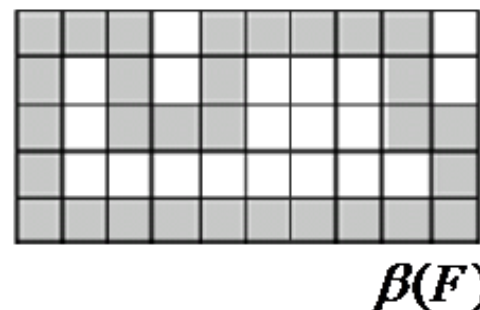
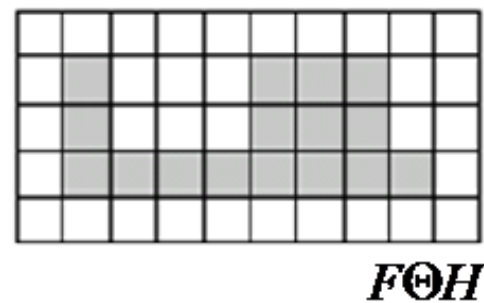
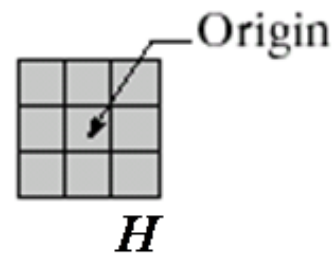
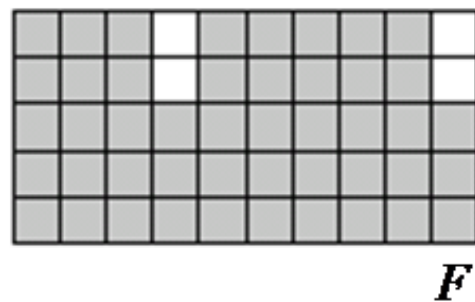
### ○ Connected Component Labeling

- Scan an image and groups its pixels into components based on pixel connectivity

# [ Morphological Processing ]

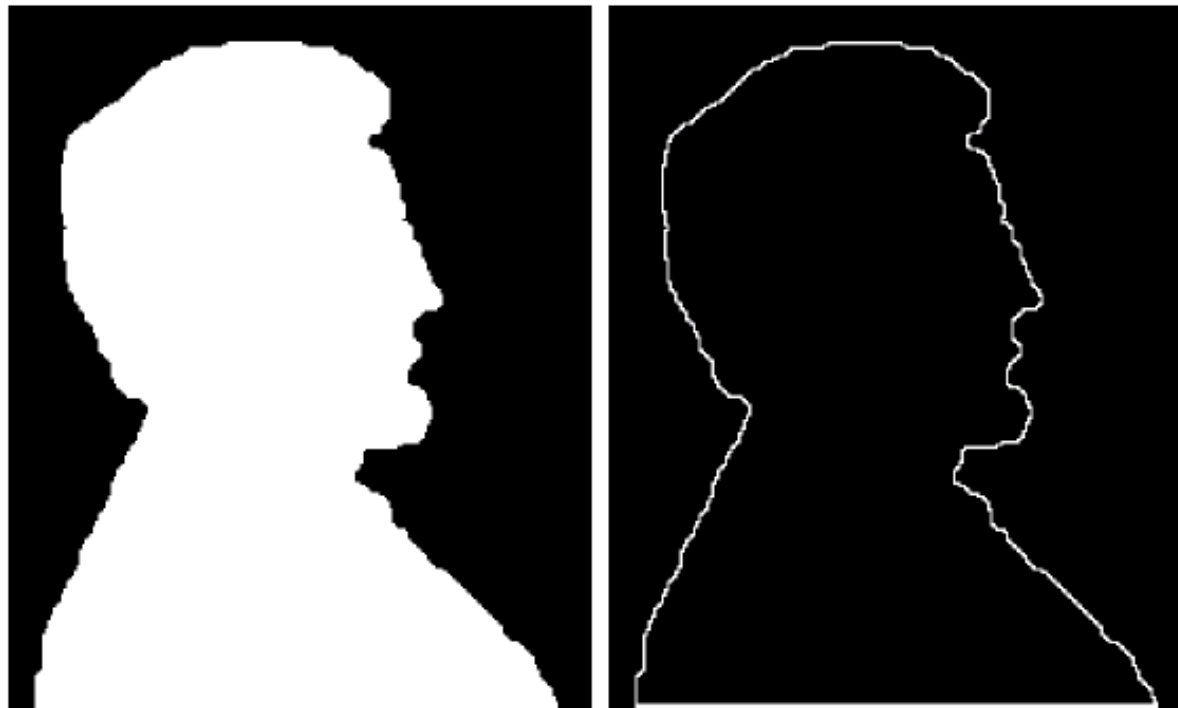
## ■ Boundary Extraction

$$\beta(F(j,k)) = F(j,k) - (F(j,k) \ominus H(j,k))$$



# [ Morphological Processing ]

## ■ Example



Original Image

Extracted Boundary

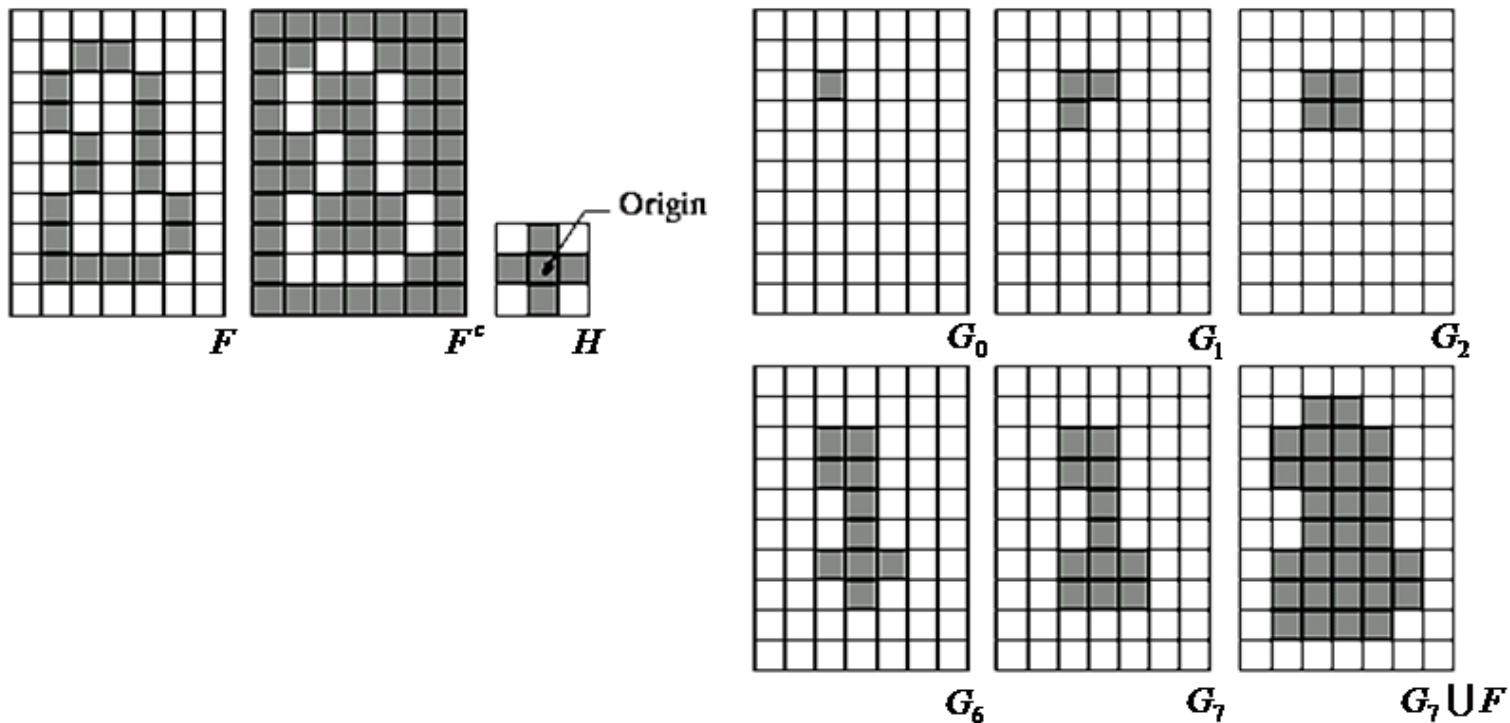


# Morphological Processing

## ■ Hole Filling

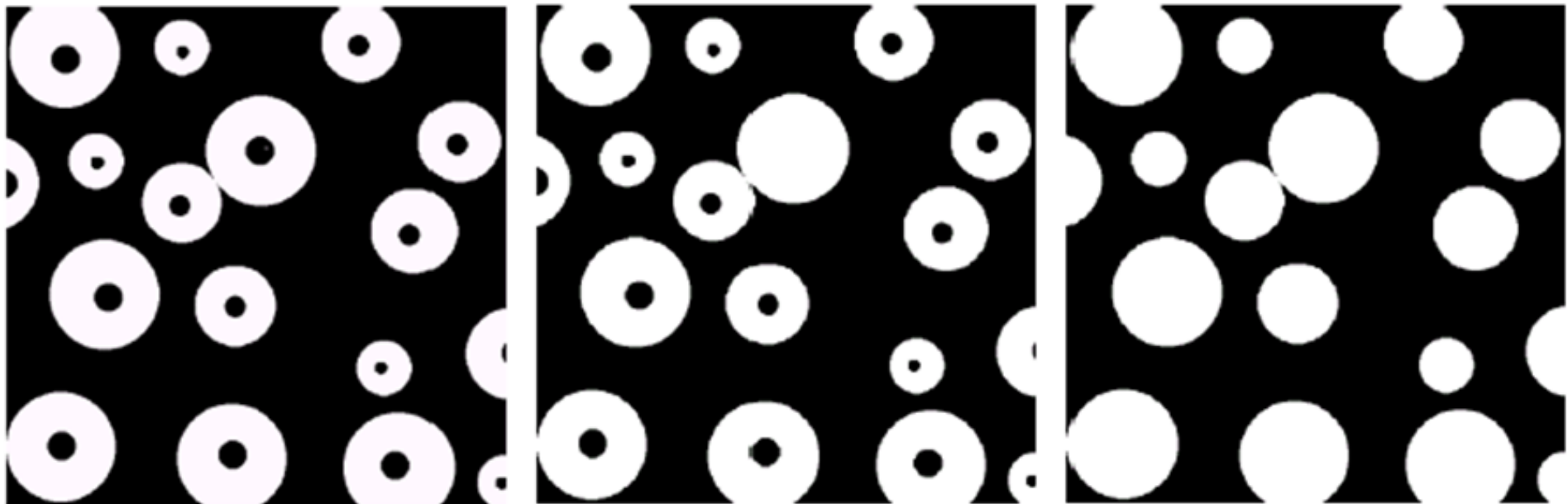
$$G_i(j, k) = (G_{i-1}(j, k) \oplus H(j, k)) \cap F^c(j, k) \quad i = 1, 2, 3 \dots$$

$$G(j, k) = G_i(j, k) \cup F(j, k)$$



# [ Morphological Processing ]

## ■ Example



Original Image

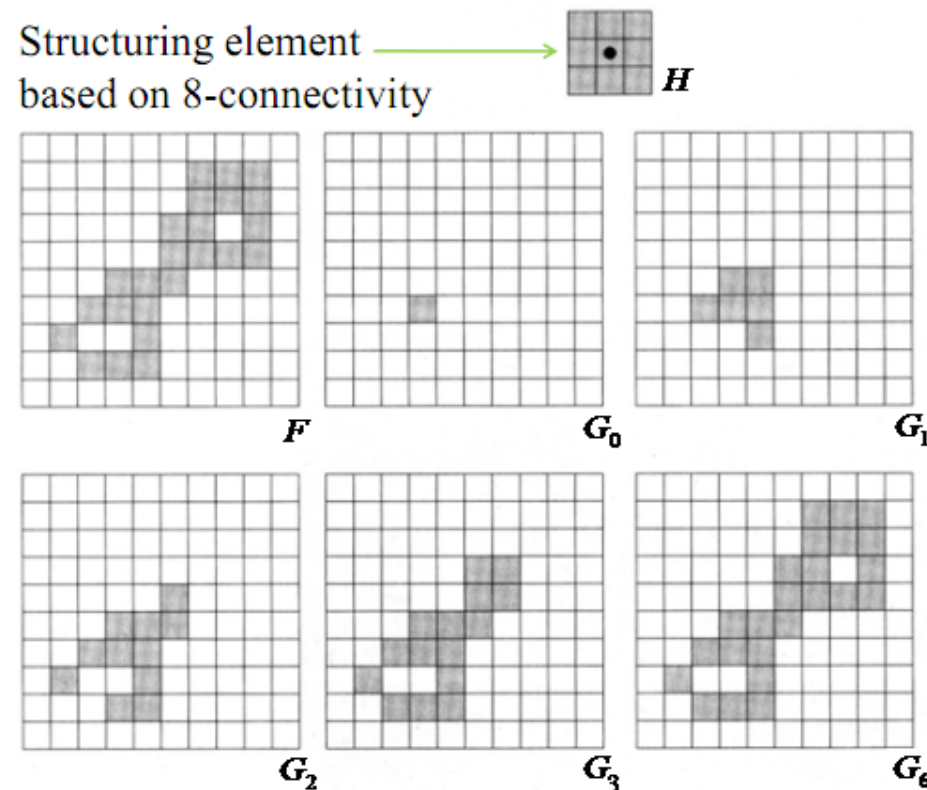
One Hole Filled

All Holes Filled

# [ Morphological Processing ]

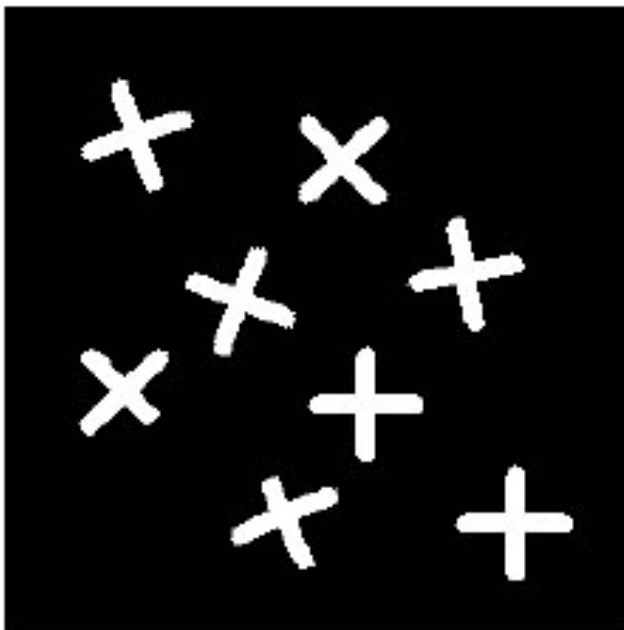
## ■ Connected Component Labeling

$$G_i(j,k) = (G_{i-1}(j,k) \oplus H(j,k)) \cap F(j,k) \quad i = 1, 2, 3, \dots$$

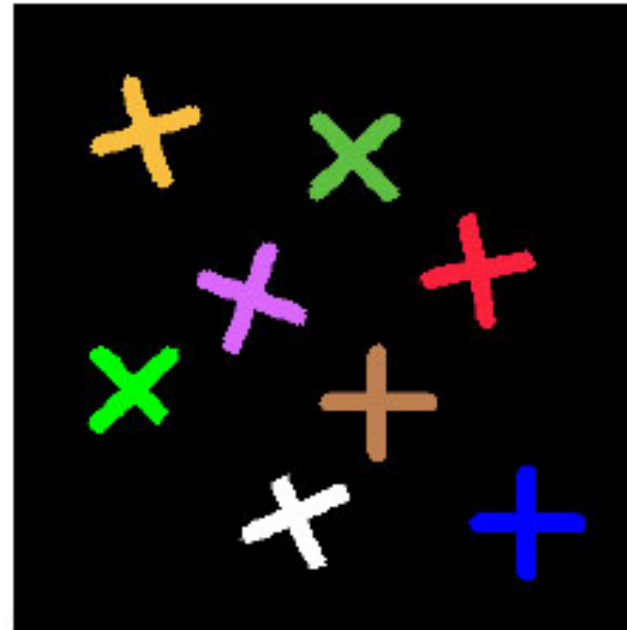


# Morphological Processing

## ■ Example



Original Image



Labelled Components

# [ Morphological Processing ]

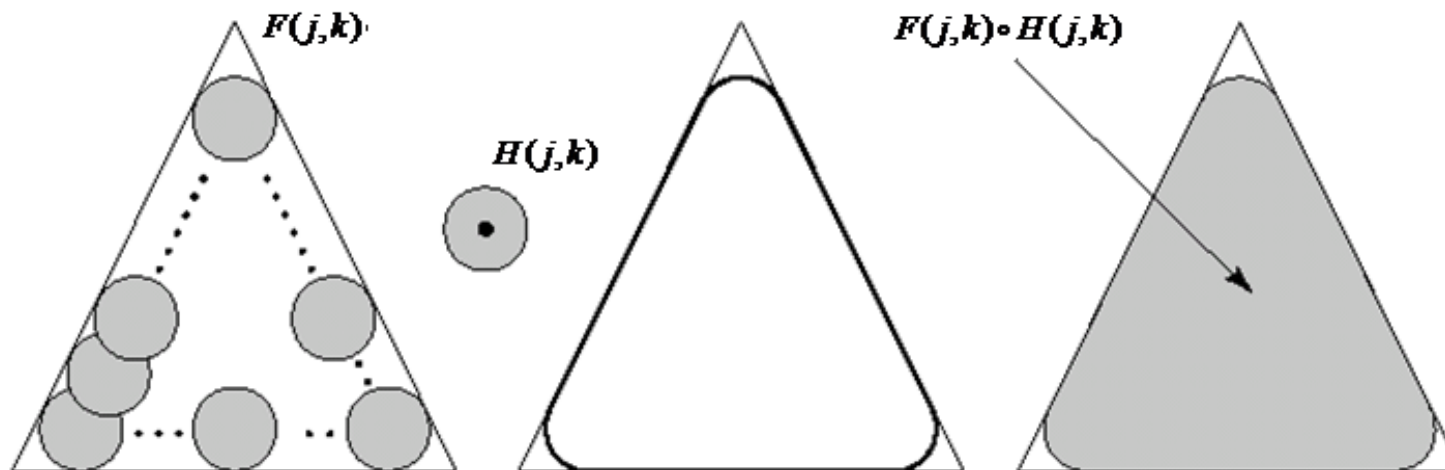
## ■ Applications

### ○ Open operator

$$G(j,k) = F(j,k) \circ H(j,k) = [F(j,k) \ominus \tilde{H}(j,k)] \oplus H(j,k)$$

#### ■ With a compact structuring element

- Smooths contours of objects
- Eliminates small objects
- Breaks narrow strokes



# [Morphological Processing]

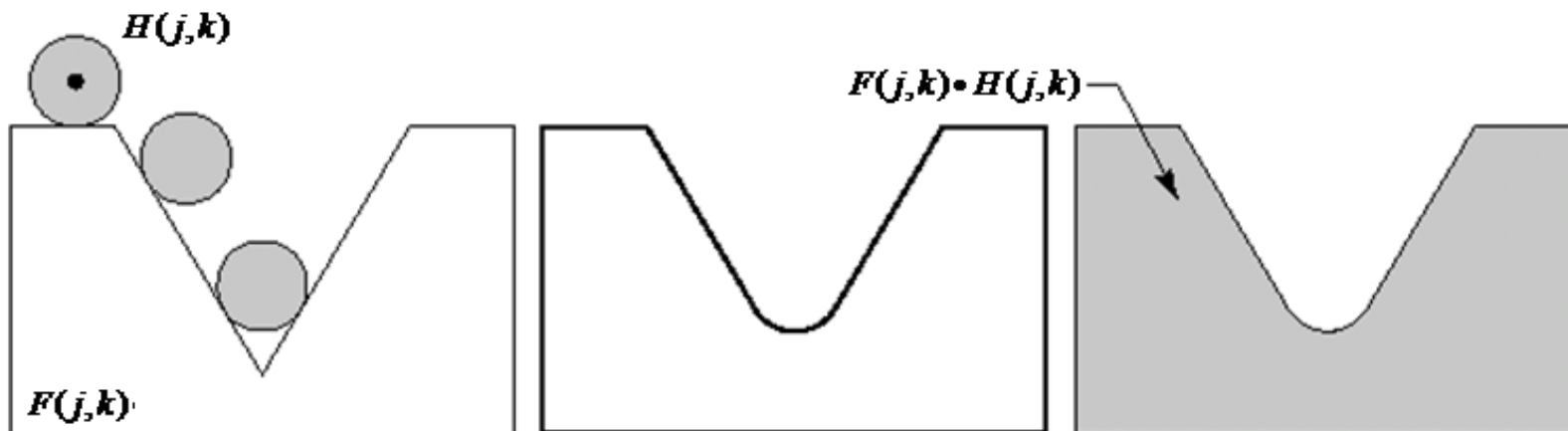
## ■ Applications

### ○ Close operator

$$G(j,k) = F(j,k) \bullet H(j,k) = [F(j,k) \oplus H(j,k)] \ominus \tilde{H}(j,k)$$

#### ■ With a compact structuring element

- Smooths contours of objects
- Eliminate small holes
- Fuses short gaps between objects

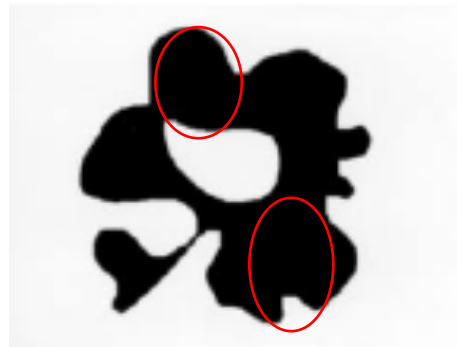


# Morphological Processing

## ■ Example



original

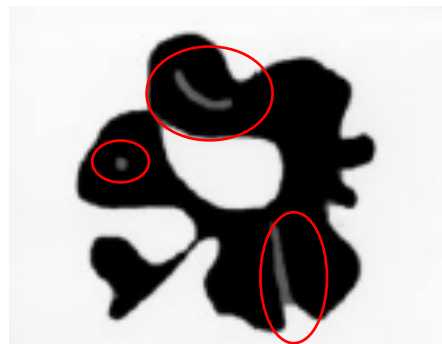


(a) close

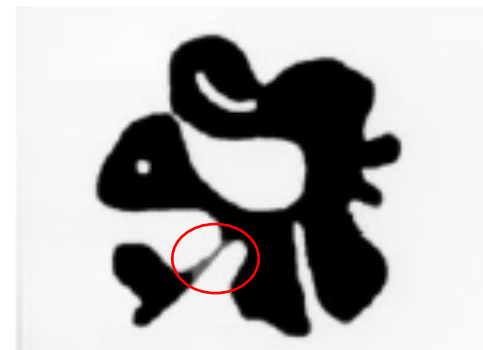


(b) open

Q: repeated openings/closings?



Compare (a) with the original image



Compare (b) with the original image

# MCBall

