Binary Image Processing

By:

Amin Khajeh Djahromi

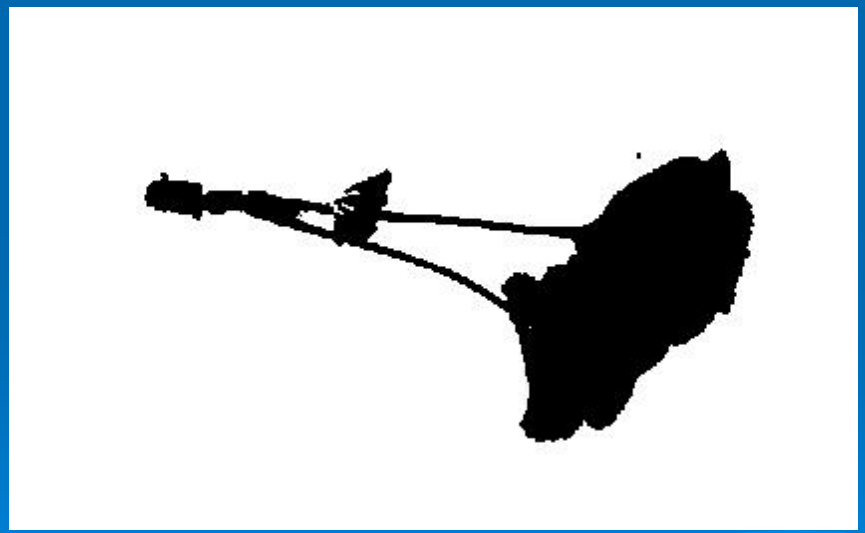
Department of Electrical Engineering
University of Texas at Arlington

Roadmap

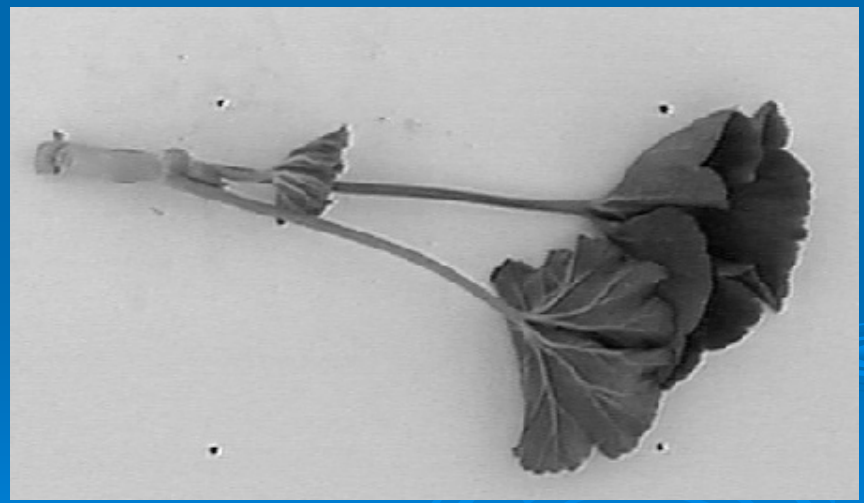
- Binary Image Processing
- Thresholding
- Geometric Properties
- Run-Length Encoding
- Binary Algorithms
- Morphological Operators
- Applications

What is Binary Image?

- Binary images are images whose pixels have only two possible intensity values.



Binary Image



Gray Level Image

Why do we use binary images?

- Smaller memory requirements
- Faster execution time
- Many techniques developed for these systems are also applicable to vision systems which use gray scale images
- Less expensive

Disadvantages of using binary images

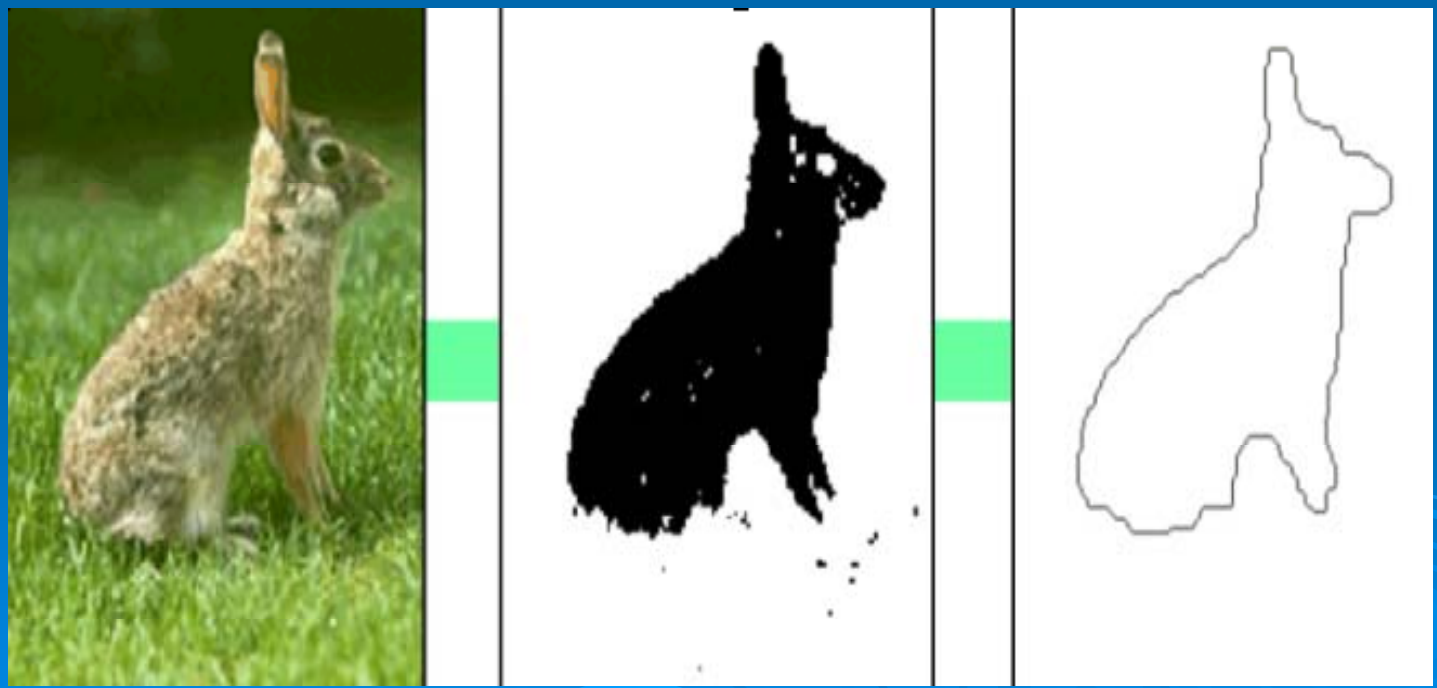
- Limited application
- Can not be extended to 3D
- Losing internal details of objects (i.e. in inspection tasks)
- difficult to control the contrast between the background and the objects (i.e. in material handling and assembly tasks)

Some aspects of Binary Vision Systems

- Formation of binary images : Binary is obtained by thresholding the gray scale image. Many cameras are designed to perform thresholding in hardware.
- Geometric Properties: Size, Position, Orientation and Projection.
- Topological Properties
- Object recognition in binary images

Segmentation

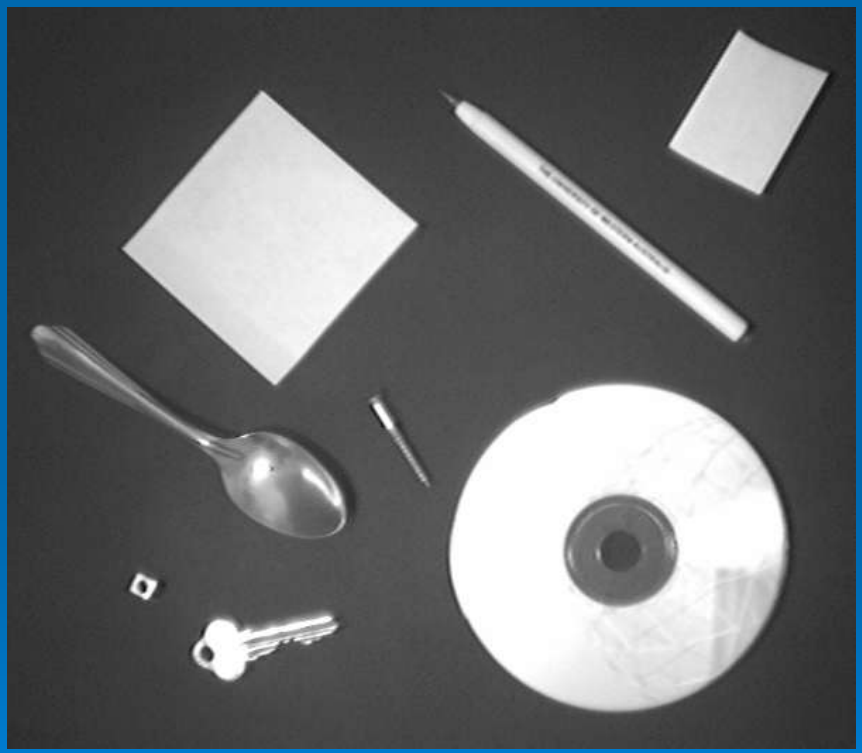
- Segmentation: The partitioning of an image into regions is called segmentation



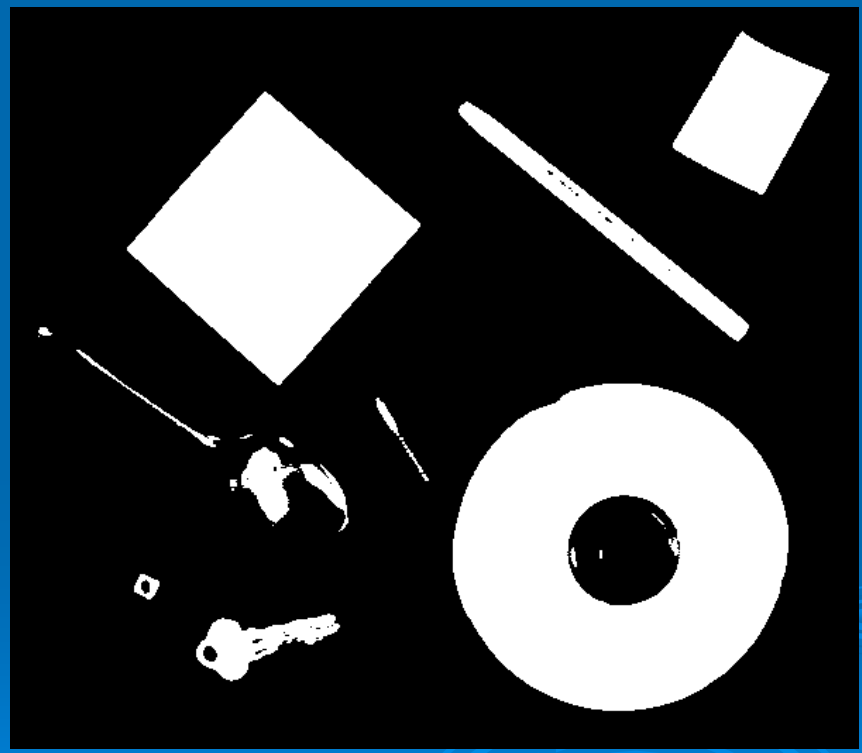
Thresholding

- Thresholding: Is a method to convert a Gray Scale Image into a Binary Image, so that objects of interest are separated from the background.

Example of Thresholding



Original



Binary (Threshold =140)

Importance of object-background separation

For thresholding to be effective in object-background separation, it is necessary that the objects and background have sufficient contrast and that we know the intensity levels of either the objects or the background. In a fixed thresholding scheme these intensity characteristics determine the value of threshold.

Some Types of Thresholding

- Fixed Thresholding
- Histogram-derived Thresholds
- Isodata algorithm
- Background Symmetry algorithm
- Triangular algorithm

Geometric properties

- Size
- Position
- Orientation
- Projection

Size

- Size: summation of all pixel values or the number of pixels of the object.

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

Size = 4

- **Position:** The center of the area of the object. In order to calculate the position of the object we need to calculate the first-order moments.

- **Orientation:** The way an object is situated in the image. For some shapes (such as circle), orientation is not unique and to define unique orientation an object must be elongated.

- **Projection:**

1. *Horizontal projection*
2. *Vertical projection*
3. *Diagonal projection*

Run-length Encoding (RLE)

- Compact representation of a binary image
- Used for image transmission
- Some properties, such as the area, projection of the objects, may be directly computed from their run-length codes.
- Greater compression achieved for longer and more frequent runs
- Widely used for binary images.

General example for RLE

- “AAAAABBBBB” require 10 bytes to store
- Run Counts : “5 5”
- Run Value : “A B “
- RLE code, “5A5B”
- RLE require only 4 bytes
- Compression ratio : 10/4=2.5

Another example of RLE

- Input : “DDDDBBBBGVVVVVV”
- RLE codes: “4D4B1G6V”
- RLE Packet : 8 bytes
- Compression ratio : 15/8

Two approaches for RLE

1. Using start position and length of 1s for each row.

2. Using length of runs, starting with the length of the 1 run for each row.

Example for approach 1

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

Start position and length of 1 runs: (1,4) (8,7)
(4,5) (16,2)

Example for approach 1

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

Length of 1 and 0 runs (starting with the length of 1 runs) :

4,5,5,3
0,4,6,5,2

Binary Algorithms

Before discussing Binary Algorithms, we first need to define some related terms.

- Neighbors
- Path
- Foreground
- Background
- Connectivity
- Connected Components
- Boundary
- Interior
- Surrounds

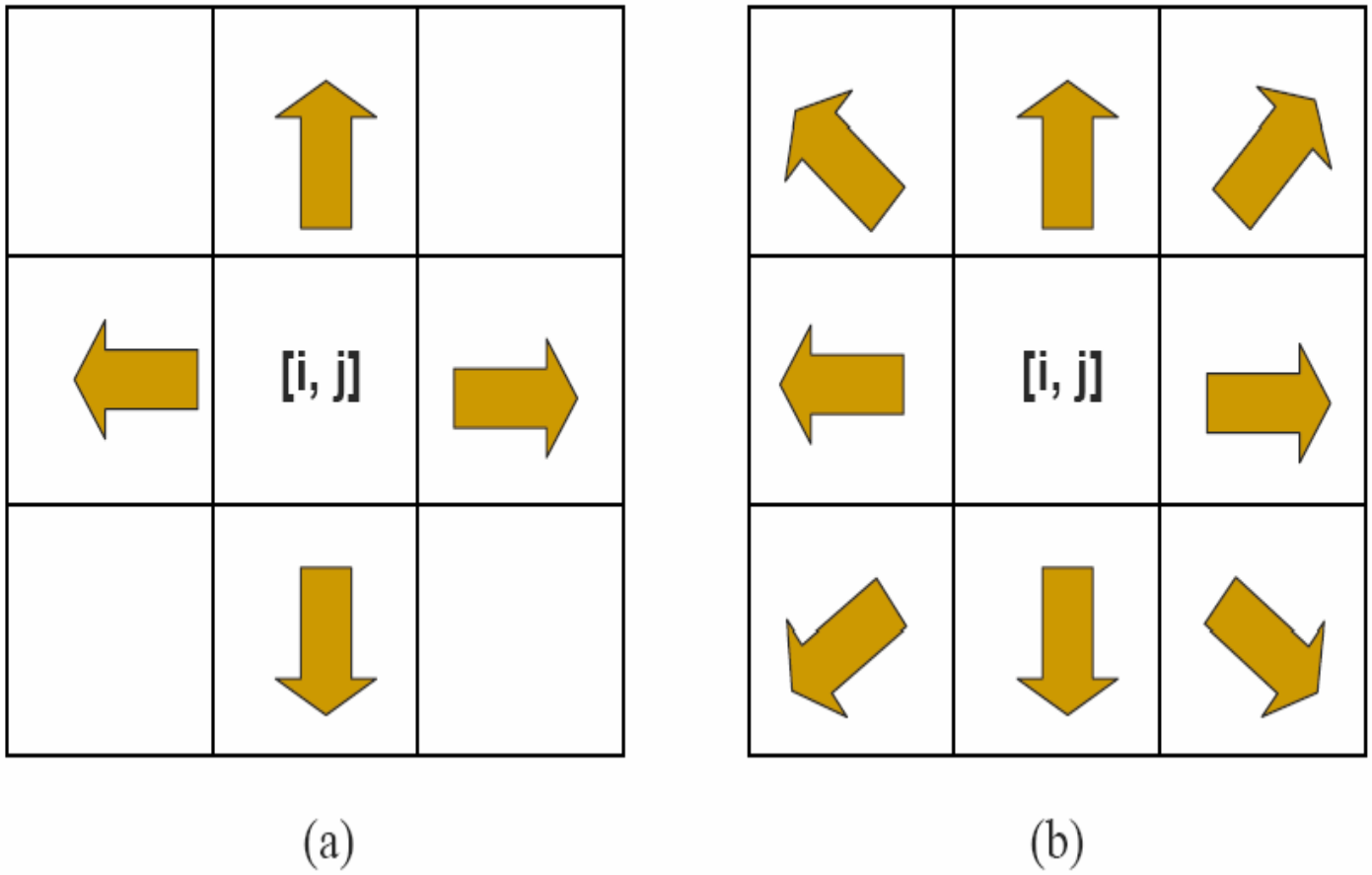
Neighborhood

A pixel has a common boundary with four pixels and shares a corner with four additional pixels.

- 4-neighbors: Two pixels are *4-neighbors* if they share a common boundary.
- 8-neighbors: Two pixels are *8-neighbors* if they share at least one corner.

Examples of *4-neighbors* and *8-neighbors*

- Neighborhood



(a) 4-neighbors, (b) 8-neighbors

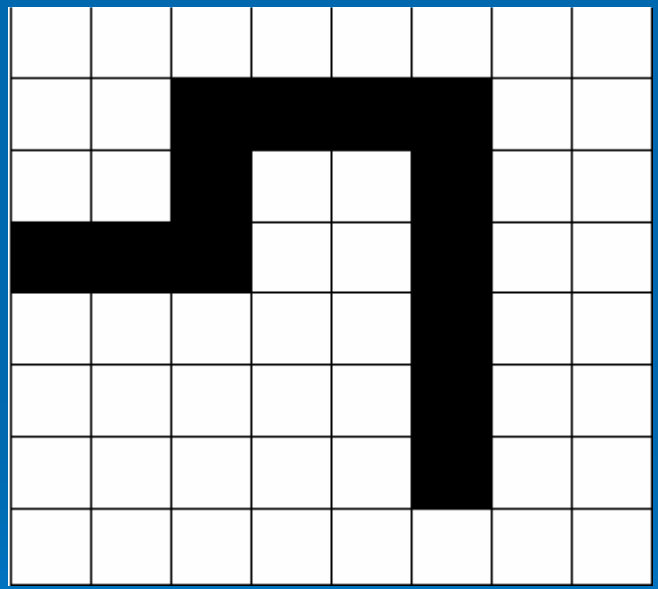
Path

- A path from the pixel at [$i_{0,j0}$] is a sequence of pixel indices $[i_0,j_0],[i_1,j_1],\ldots[i_n,j_n]$ such that the pixel at $[i_k,j_k]$ is neighbor of the pixel at $[i_{k+1},j_{k+1}]$ for all *k* with $0 \leq k \leq n-1$.

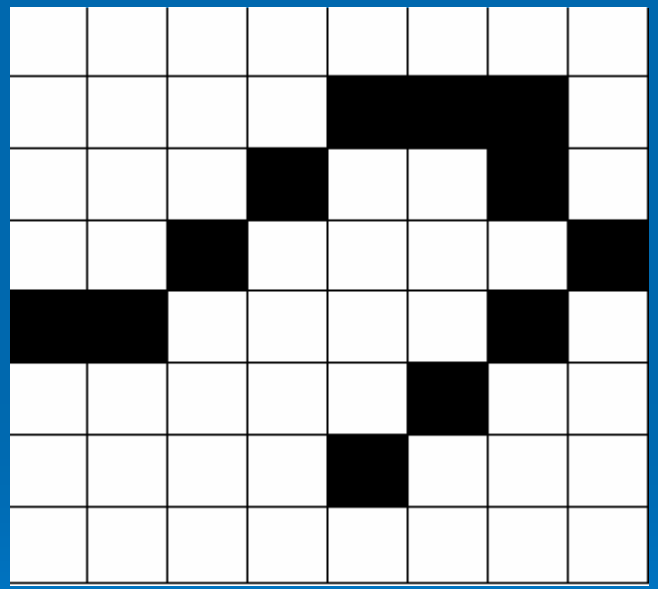
According to the above definition we will have to different paths:

1. 4-path : using *4-connection*
2. 8-path : using *8-connection*

Examples of 4-path and 8-path



(a) 4-path



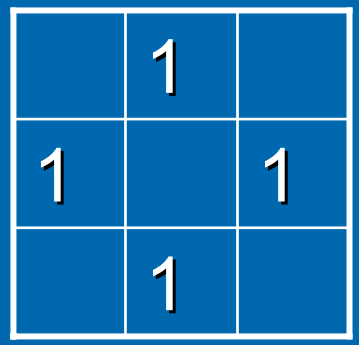
(b) 8-path

Foreground, Background and Holes

- Foreground: The set of all 1 pixels in an image is called the *foreground* and is denoted by "*S*"
- Background: The set of all connected components of complement of "S" that have points on the border of an image is called the background.
- All the other components of complement of "S" are called *holes*.

Example of foreground and background

Consider the below picture:



If we consider 4-connectedness for both foreground and background, there are four Objects that are 1 pixel in size and there is no hole.

But,

if we use 8-connectedness, then there is one object and no hole.

So

to avoid this awkward situations, if we use 8-connectedness for foreground, then 4-connecteness should be used for background.

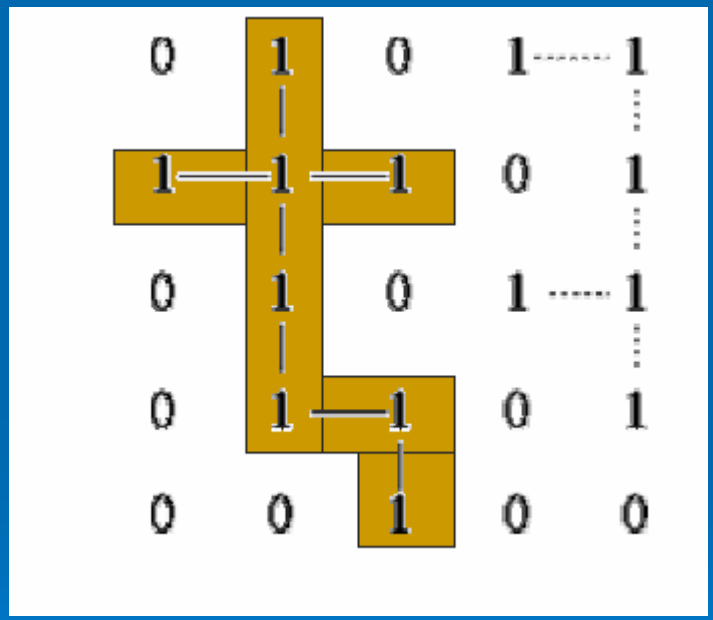
Connectivity

- A pixel *p* Є S is said to be *connected* to *q* Є S if there is a path from *p* to *q* consisting entirely of pixels of S.

For any three pixels *p*,*q* and r in S, we have:

1. Pixel *p* is connected to *p (*reflexivity).
2. If *p* is connected to *q*, then q is connected to p (commutativity).
3. If *p* is connected to *q* and *q* is connected to *r*, then *p* is connected to *r* (transitivity).

Example of connectivity

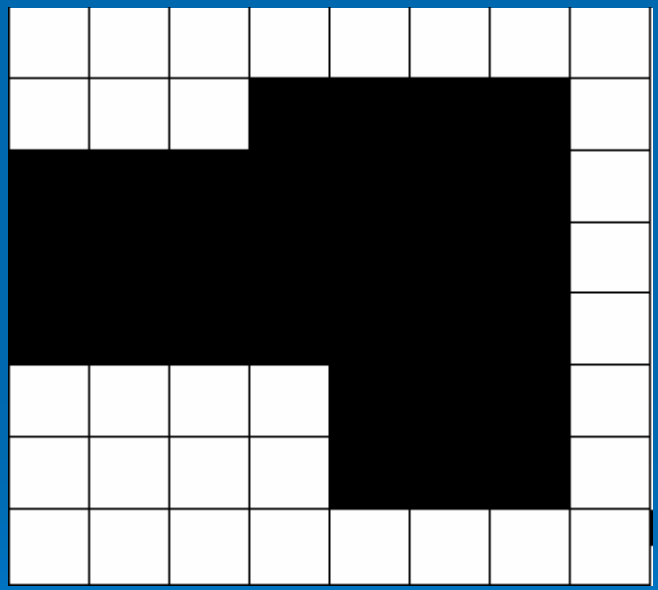


Two connected components based on 4-connectivity

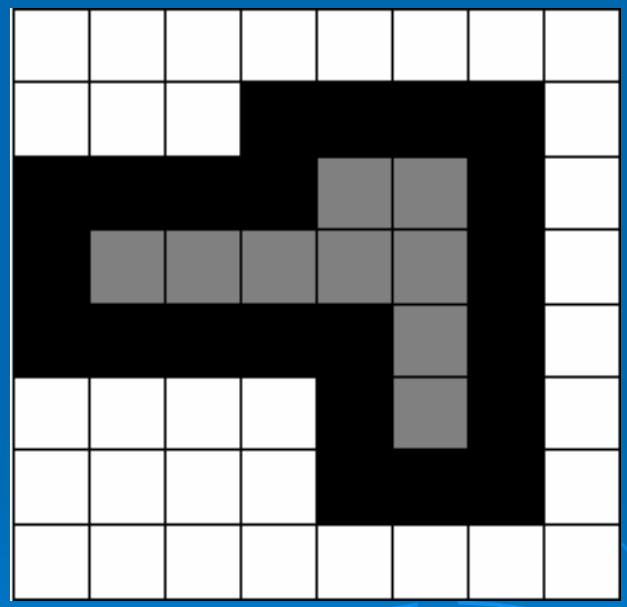
Boundary, Interior and surrounds

- Boundary: The boundary of S is the set of pixels of S that have 4-neighbors in Sbar. The boundary is usually denoted by S` .
- Interior: The interior is the set of pixels of S that are not in it's boundary. The interior of S is (S-S`).
- Surrounds: T surrounds region S, if any 4-path from any points of S to the border of picture must intersect T.

Example of Boundary, Interior and surrounds



Original image

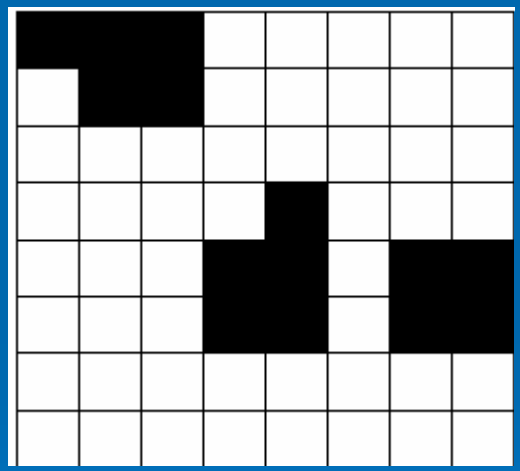


Black: boundary pixels
Gray: interior pixels
White: surrounds pixels

Components Labeling

- To find connected components in an image. The points in a connected components form a region that represent an object.
- Require the characteristics of the components (size, position, orientation and …)

Example of Component Labeling



Original image

1	1	1					
	1	1					
				2			
			2	2		3	3
			2	2		3	3

It’s connected components

Two algorithm for Component Labeling

1- Recursive algorithm

- Rarely used on general-purpose computers
- Commonly used on parallel machines

2- Sequential algorithm

- Usually require two passes over the image
- Works with only two rows of an image at a time so there is no need to bring the whole image into memory

Recursive algorithm

1. *Scan the image to find an unlabeled 1 pixel and assign it a new label L.*
2. *Recursively assign a label L to all it's 1 neighbors.*
3. *Stop if there are no more unlabeled 1 pixel.*
4. *Go to step 1.*

Sequential algorithm

1- Scan the image left to right, top to bottom

2- If this pixel is one

(a) if only one of its upper and left neighbors has a label, then copy the label

(b) If both have the same label, then copy the label.

(c) If both have different labels, then copy the upper's label and enter the labels in the equivalence table as equivalent labels

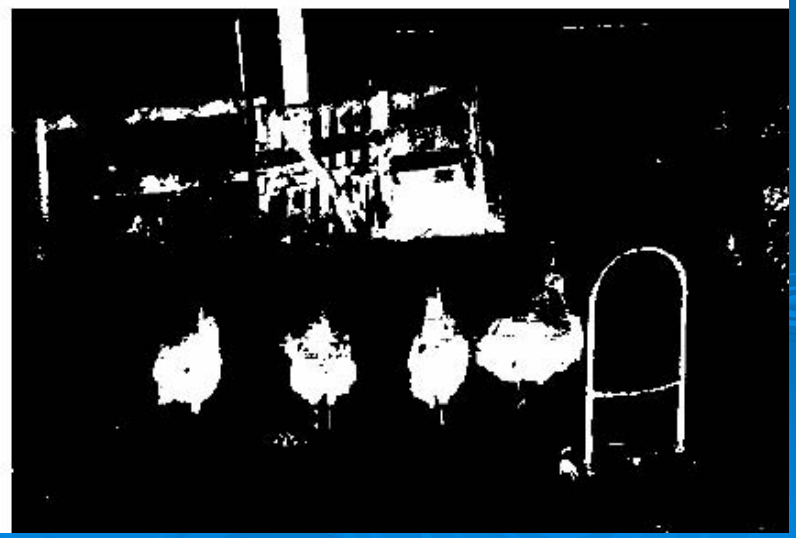
(d) Otherwise assign a new label to this pixel and enter this label in the equivalence table.

3- If there are more pixels to consider, then go to step 2.

4- Find the lowest label for each equivalent set in the equivalence table.

5- Scan the picture. Replace each label by the lowest label in its equivalent table.

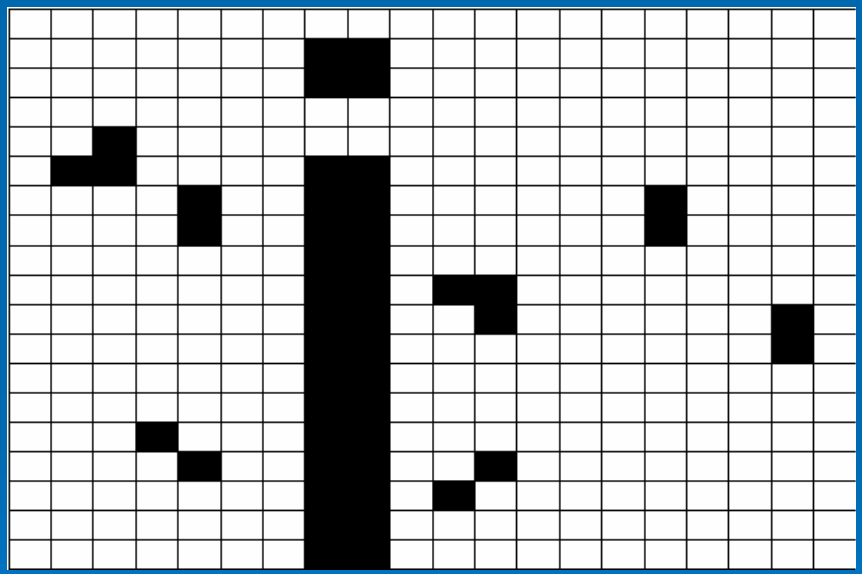
Example of Component Labeling



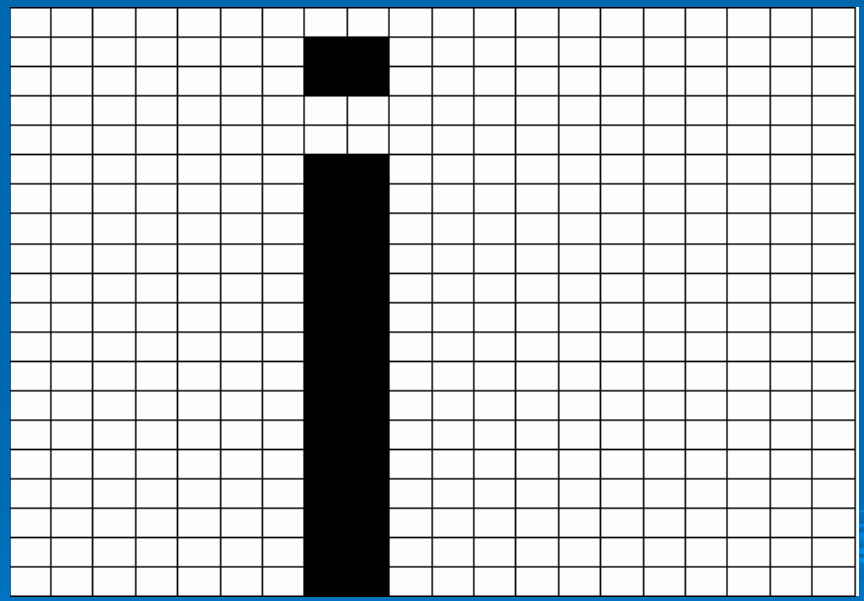
Size filtering

- If we know that objects of interest are of size greater than T_0, then we can use size filter to remove noise after component labeling. In this case all components below T_0 in size are removed by changing the corresponding pixels to 0.

Example of size filtering



Original image (noisy letter “ i “)



size filtered image T=3

Euler number

- E (Euler No.) used as a feature of an object.
- E= C – H

Where C = number of connected components (objects) and H = number of Holes.

- E is invariant to translation, rotation and scaling. (fixed feature of an object)

Example of Euler number

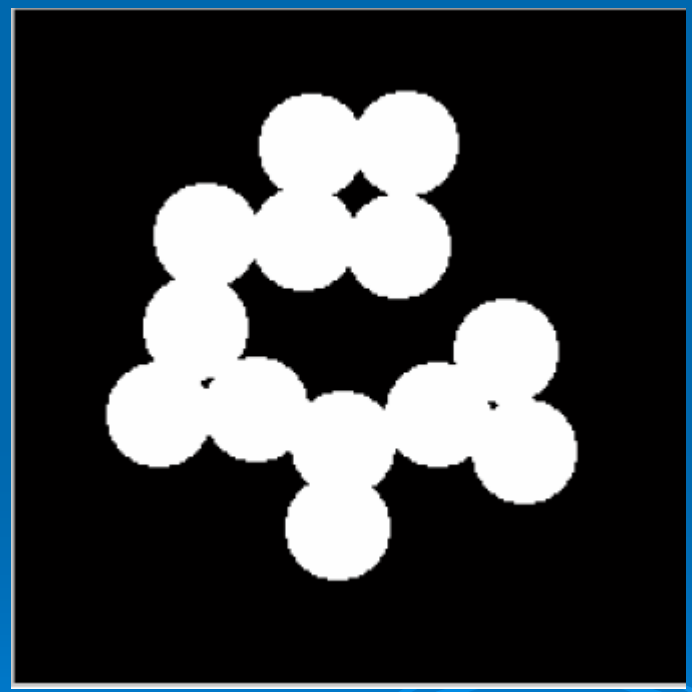
- For this image:

C (number of objects)=?

H (number of holes) =?

E=C-H = ?

- C=1 !!!
- H=3 !!!
- E= -2



Distance measure

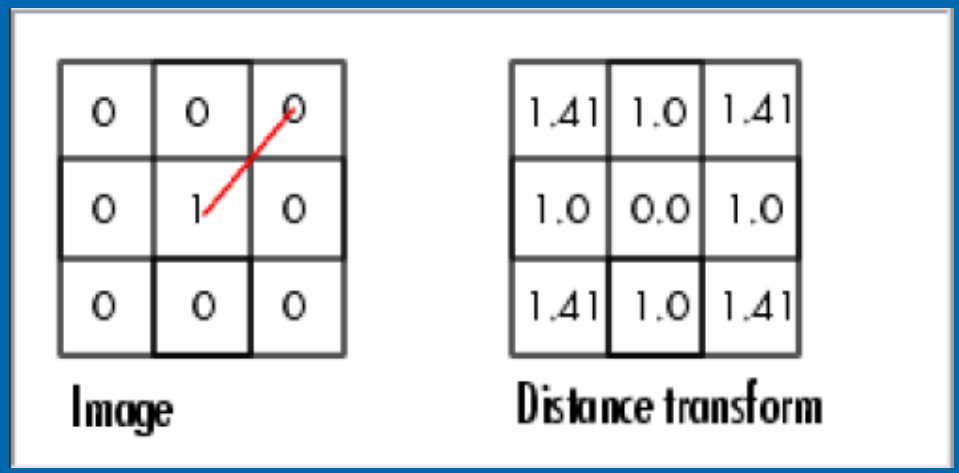
1. Euclidean
2. City-block
3. Chessboard

- The city-block and chessboard measure are preferred over Euclidean due to their simplicity of calculation.
- Euclidean distance is closest to the continue case, but it is computationally the most expensive.

Distance transform

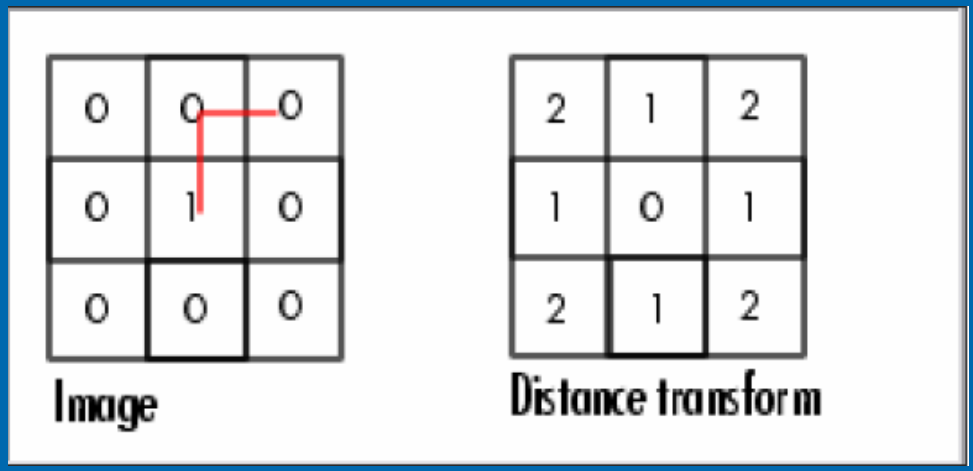
- Representing an image in terms of distance.
- It is useful in character recognition.
- It can calculated by using a parallel iterative algorithm

Distance Measure -Euclidean



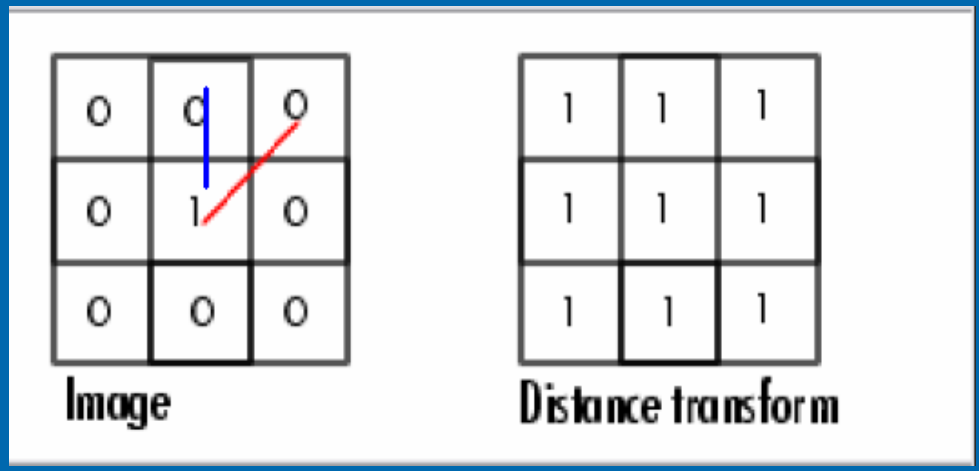
- The Euclidean distance is the straight line between two pixels.

Distance Measure -Cityblock



- Measure the path between the pixels based on a 4-connected neighborhood.
- Pixels whose edges touch are 1 unit apart.
- Pixels diagnally touching are 2 units apart.

Distance Measure -Chessboard

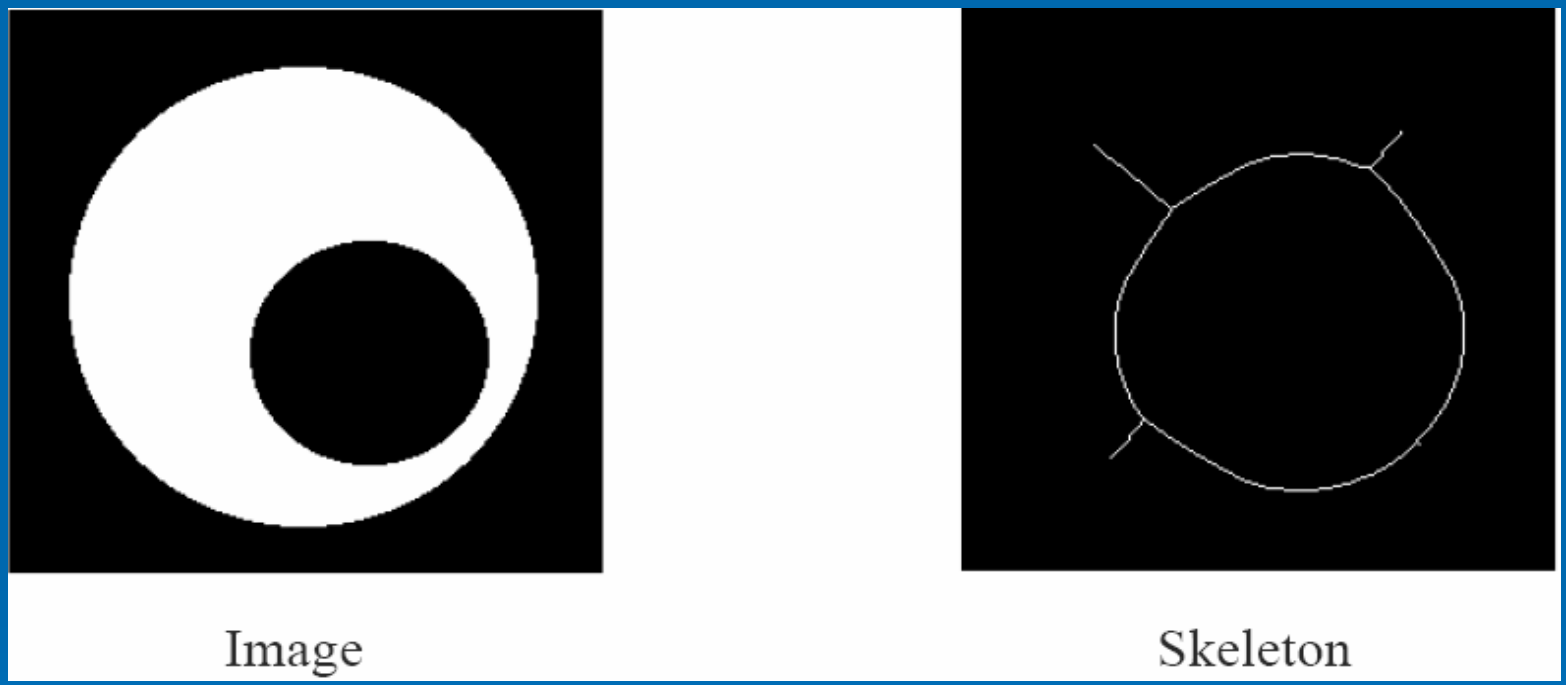


- Measure the path between the pixels based on an 8-connected neighborhood.
- Pixels whose edges or corners touch are 1 unit apart.

Medial Axis/ Skeletonization

- Used to reduce foreground regions in a binary image to a skeletal remnant that largely preserves the extent and connectivity of the original region while throwing away most of the original foreground pixels.
- Methods of skeletonization:
  - *Thinning*
  - *Calculate distance transform* of the image -the skeleton then lies along the *singularities* (*i.e.* creases or curvature discontinuities) in the distance transform.

Example



Skeletonization

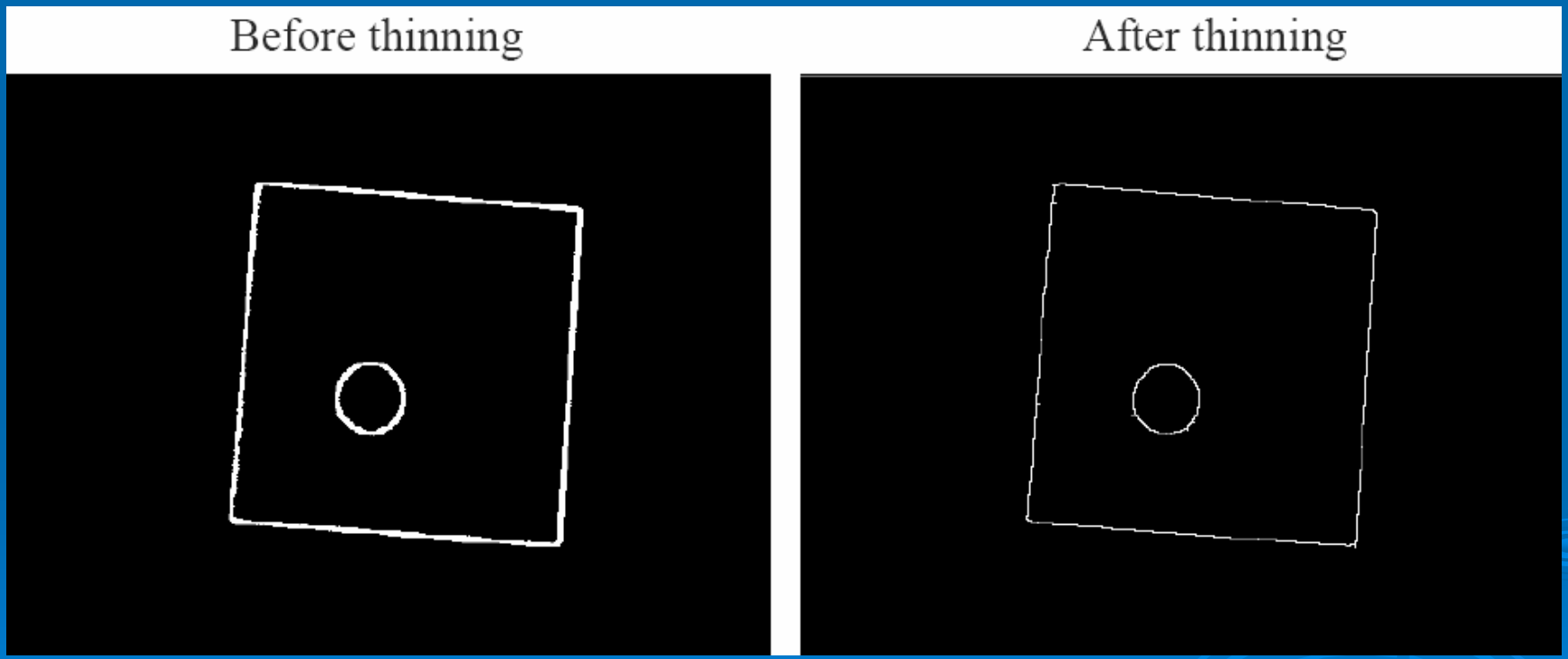
Thinning

- Thinning is an image processing operation in which binary-valued image regions are reduced to lines that approximate their center lines.
- Also called as “skeleton” or “core”
- Reduce image components to their essential information

Thinning requirements

1. Connected image regions must thin to connected line structures.
2. The thinned result should be minimally 8-connected.
3. Approximate endline location should be maintained.
4. The thinning results should approximate the medial lines.
5. Short branches caused by thinning should be minimized.

Example of Thinning



Threshold = 60

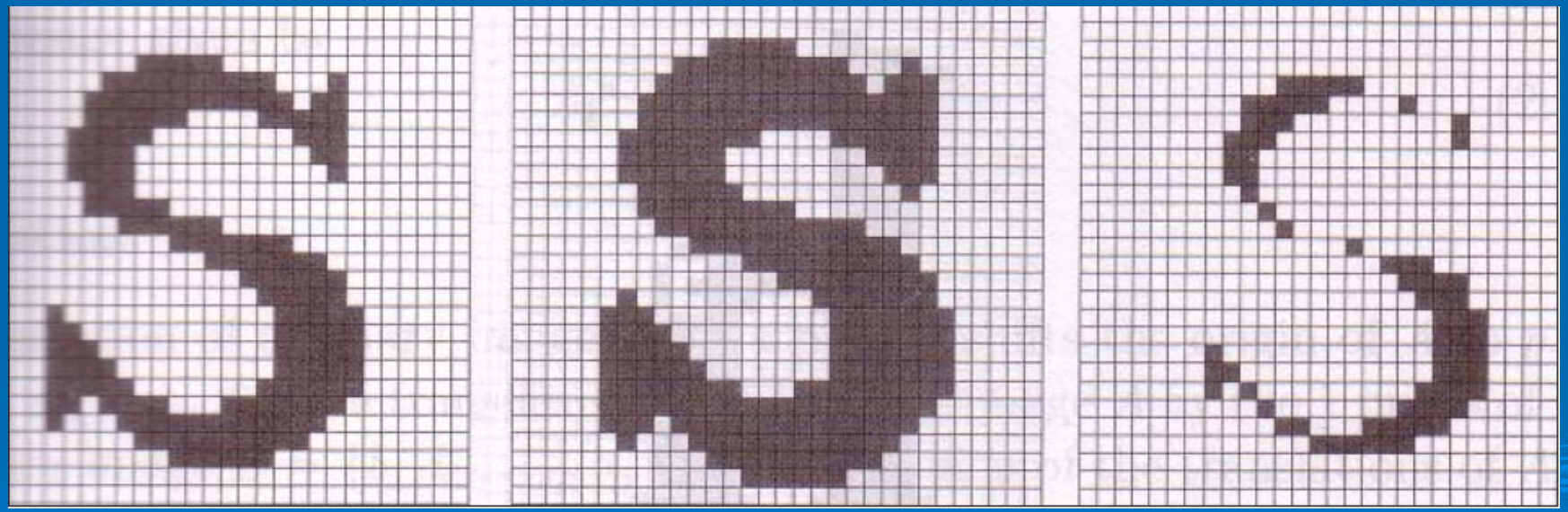
Expanding and shrinking

- Expanding : Change a pixel from 0 to 1 if any neighbors of the pixel are 1.
- Shrinking : Change a pixel from 1 to 0 if any neighbors of the pixel are 0.
- According to the above definition, Shrinking may be consider as expanding the background.

Properties of Shrinking and Expanding

- Expanding followed by Shrinking can be used for filling undesirable holes.
- Shrinking followed by Expanding can be used for removing isolated noise pixels.
- Expanding and Shrinking can be used to determine isolated components and clusters.

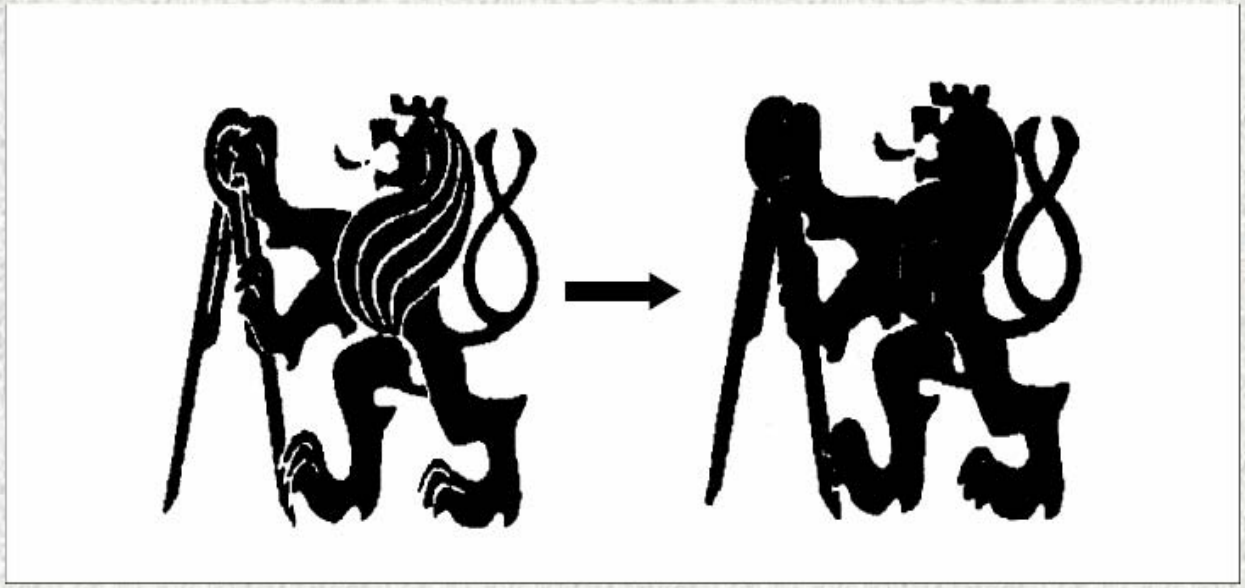
Example of Shrinking and Expanding



An example of expanding and shrinking operations on the letter “s”
Left: the original image *Middle*: Expanded image *Right*: Shrunken

Mathematical Morphology

- Stands as a relatively separate part of image analysis
- Based on the algebra of non-linear operators operating on object shape
- Supersedes the linear convolution
- Yields fast algorithms



A binary morphological operation

Applications

- Binary morphology (2D or 3D)
  - Shape simplification
  - Shape enhancing
  - Skeletonizing
  - Thinning and thickening
- Gray-scale morphology
  - Image segmentation
  - Noise filtering

Basic operations

Binary mathematical morphology consists of two basic operations:

dilation and **erosion**

and several composite relations:

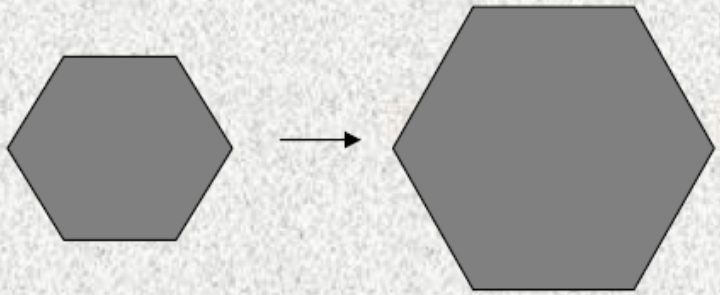
closing and **opening**
hit or miss transformation
. . .

Dilation

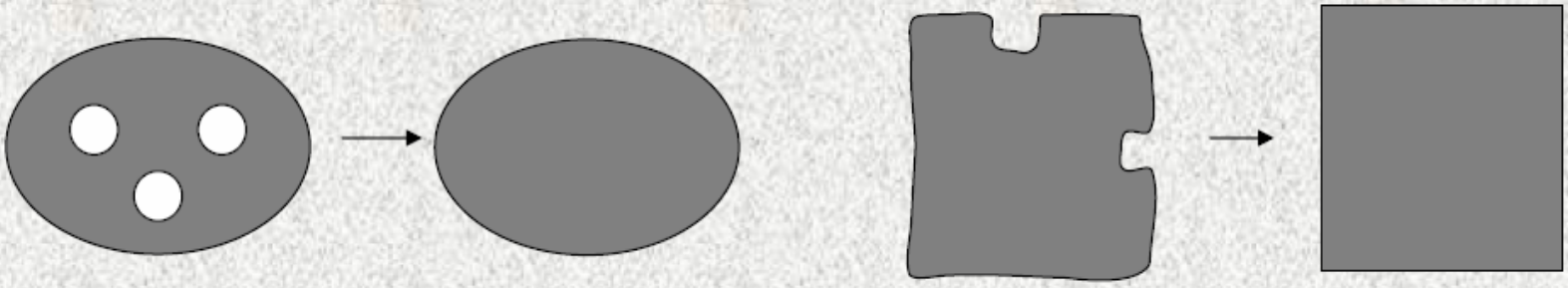
Dilation expands the connected sets of 1s of a binary image.

It can be used for

1. expanding shapes:



2. filling holes, gaps and gulfs:

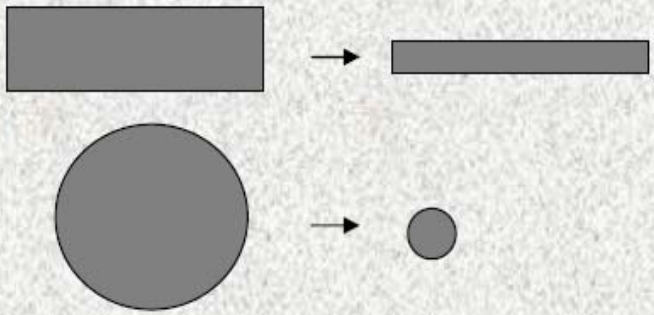


Erosion

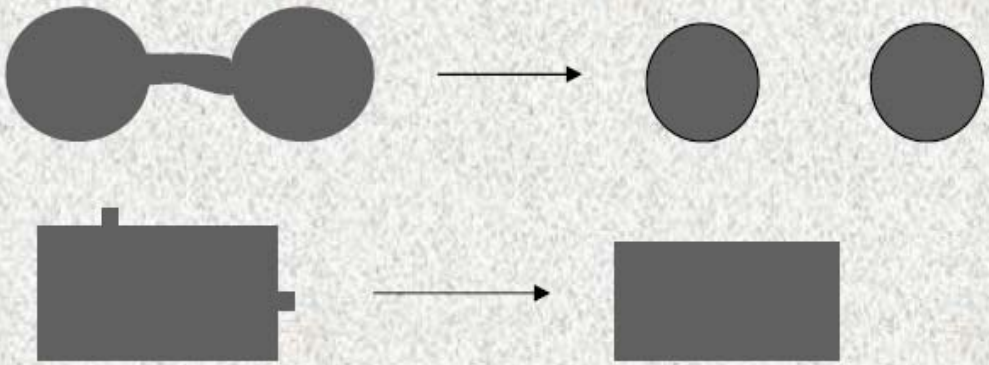
Erosion shrinks the connected sets of 1s of a binary image.

It can be used for

1. shrinking shapes:



2. removing bridges, branches and small protrusions:



Structuring Elements

A structuring element is a shape mask used in the basic morphological operations.

They can be any shape and size that is digitally representable, and each has an origin.

box

hexagon

disk

any shape

box(length,width)

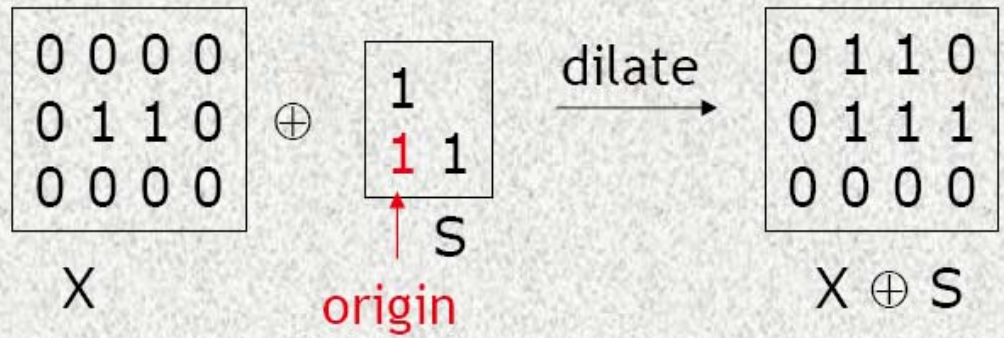
disk(diameter)

Dilation with Structuring Elements

The arguments to dilation and erosion are

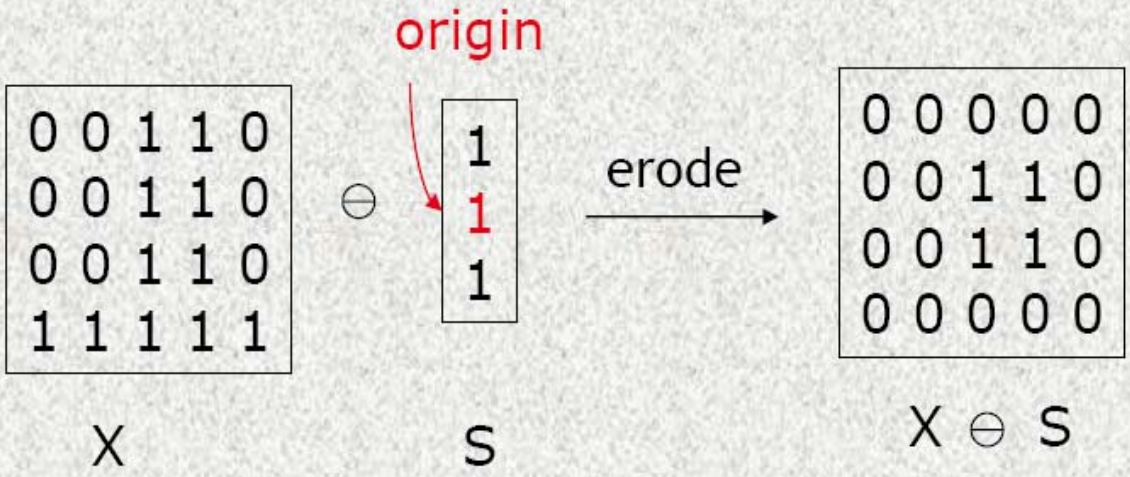
1. a binary image X
2. a structuring element S

dilate(X,S) takes binary image X, places the origin of structuring element S over each 1-pixel, and ORs the structuring element S with X to produce the output image at the corresponding position.

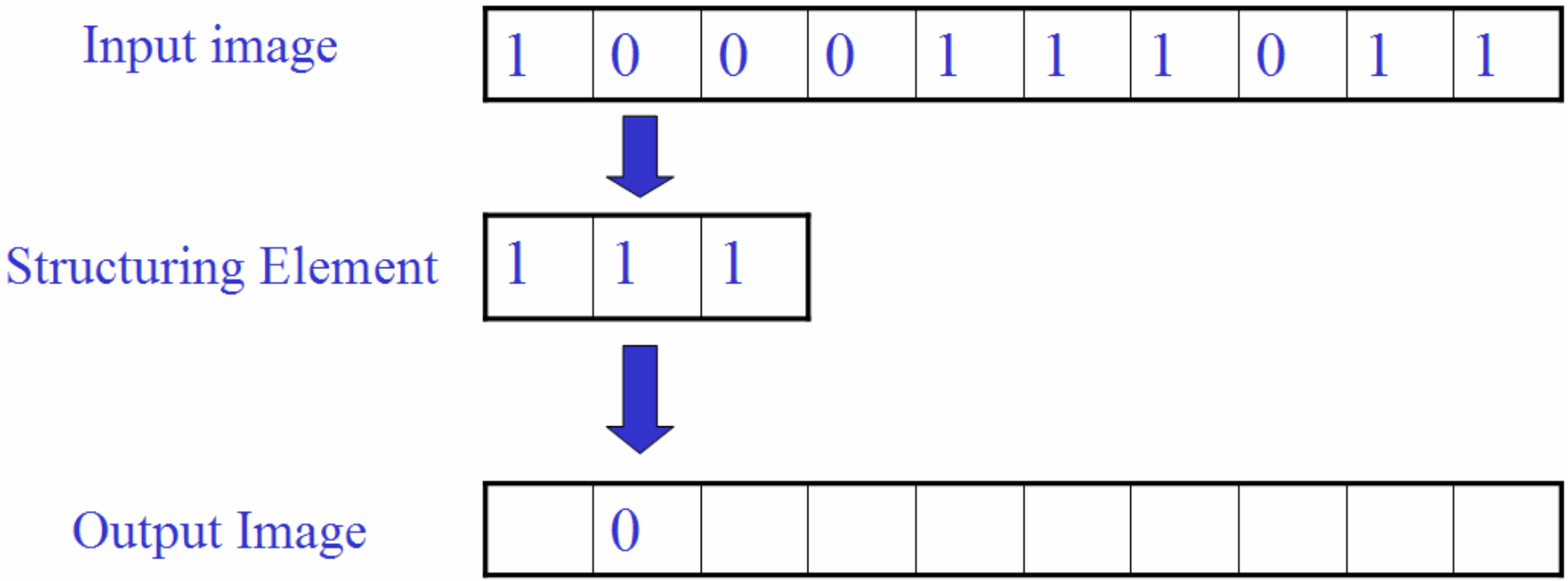


Erosion with Structuring Elements

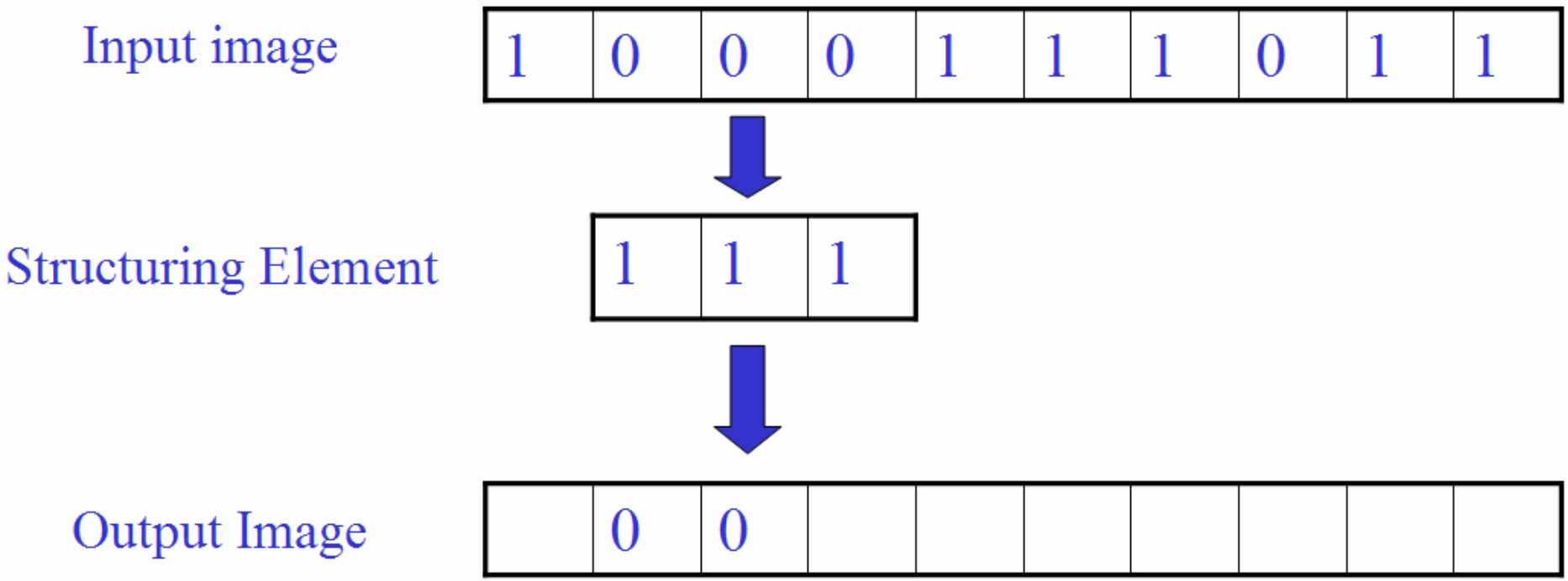
erode(X,S) takes a binary image X, places the origin of structuring element S over every pixel position, and puts a binary 1 into that position of the output image only if every position of S (with a 1) covers a 1 in X.



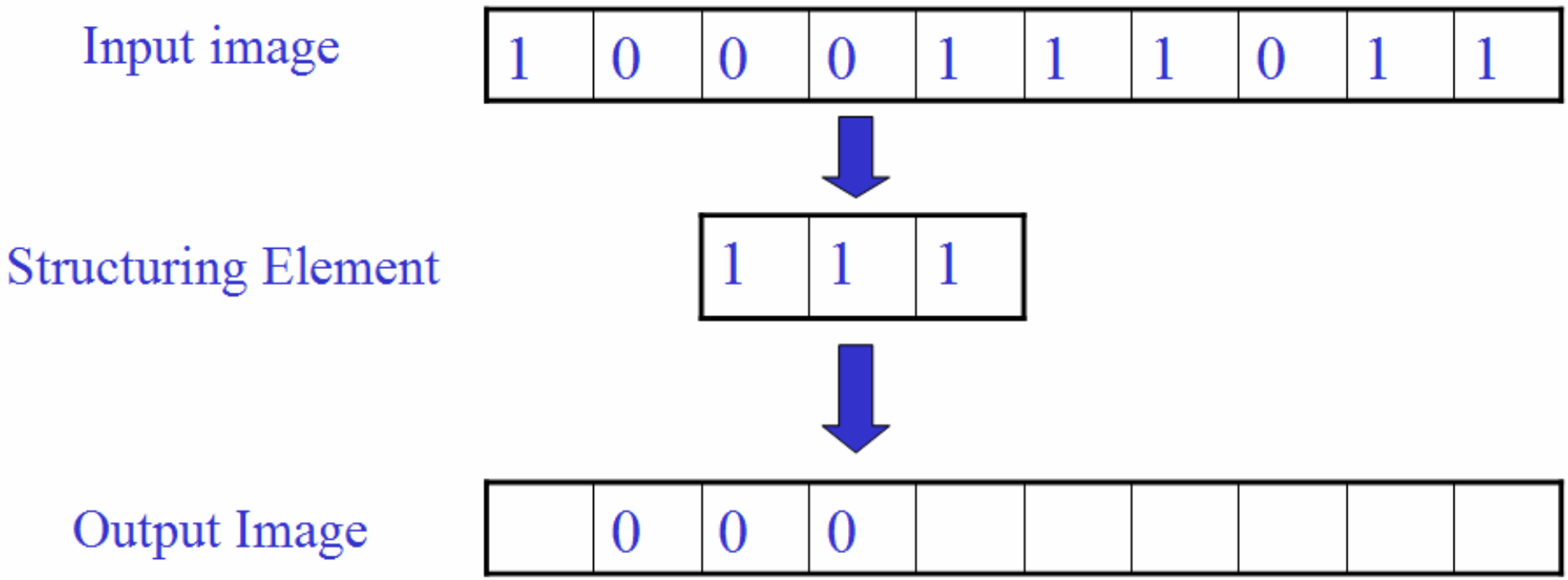
Example for Erosion



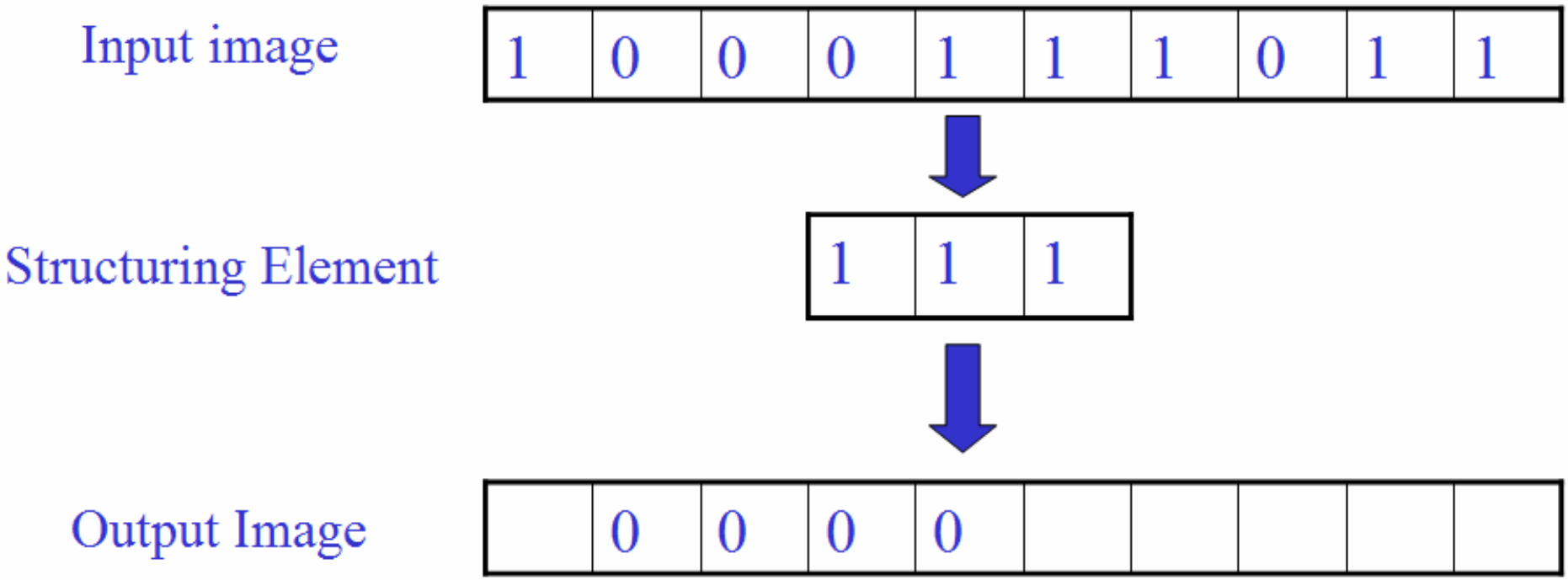
Example for Erosion



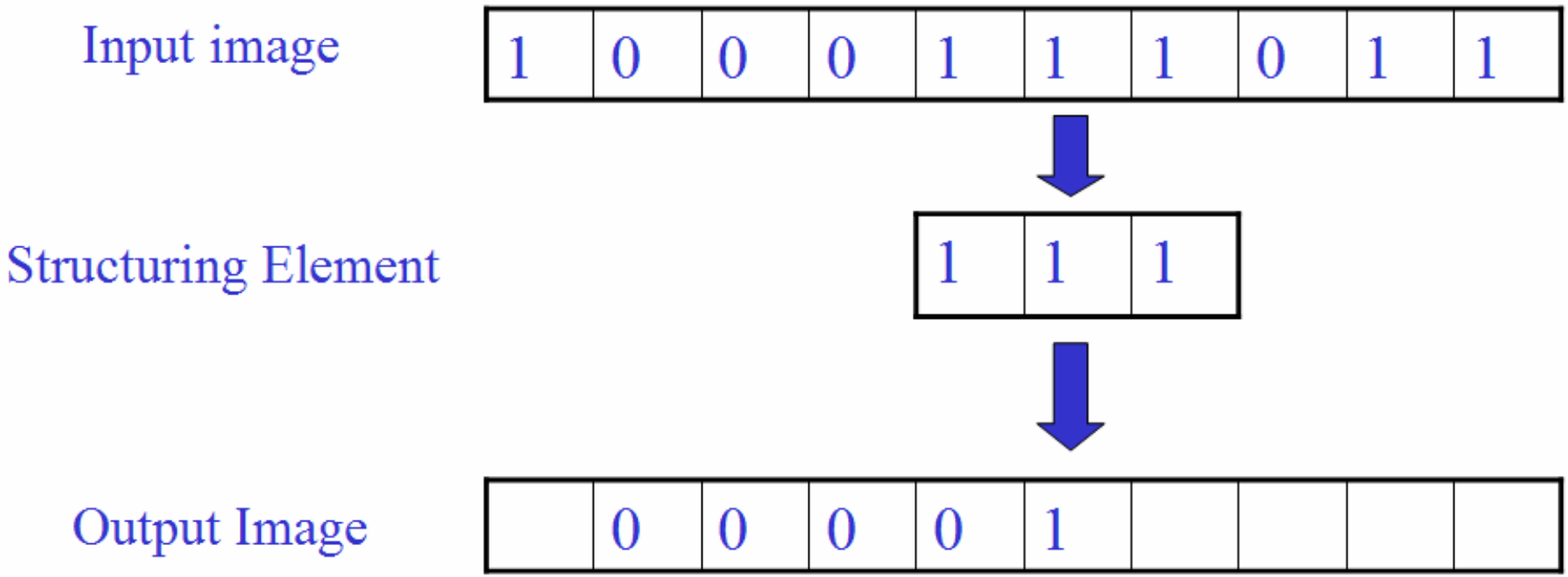
Example for Erosion



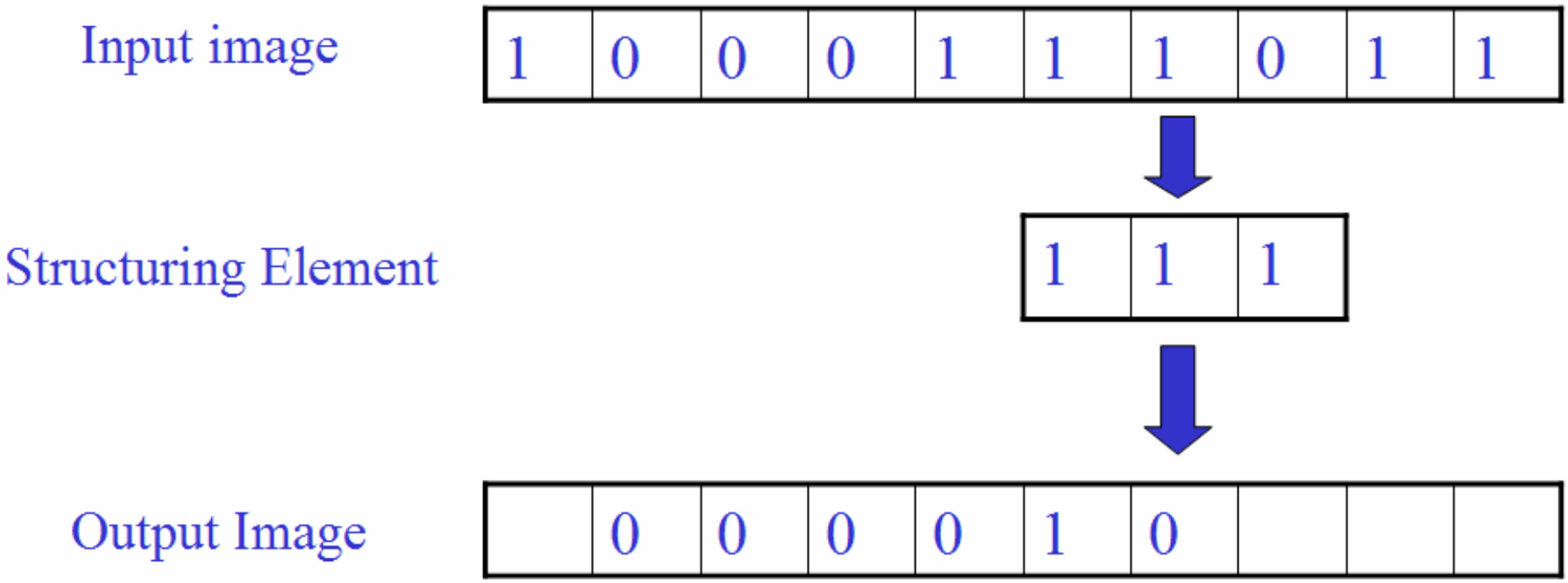
Example for Erosion



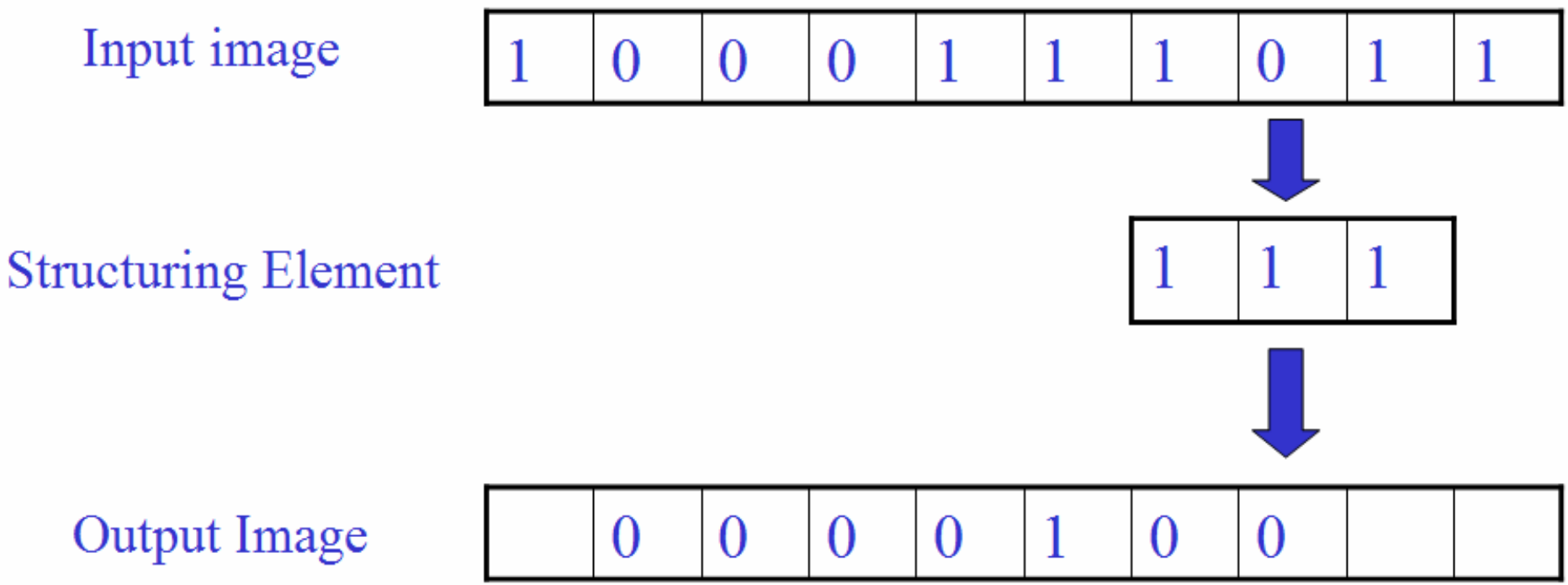
Example for Erosion



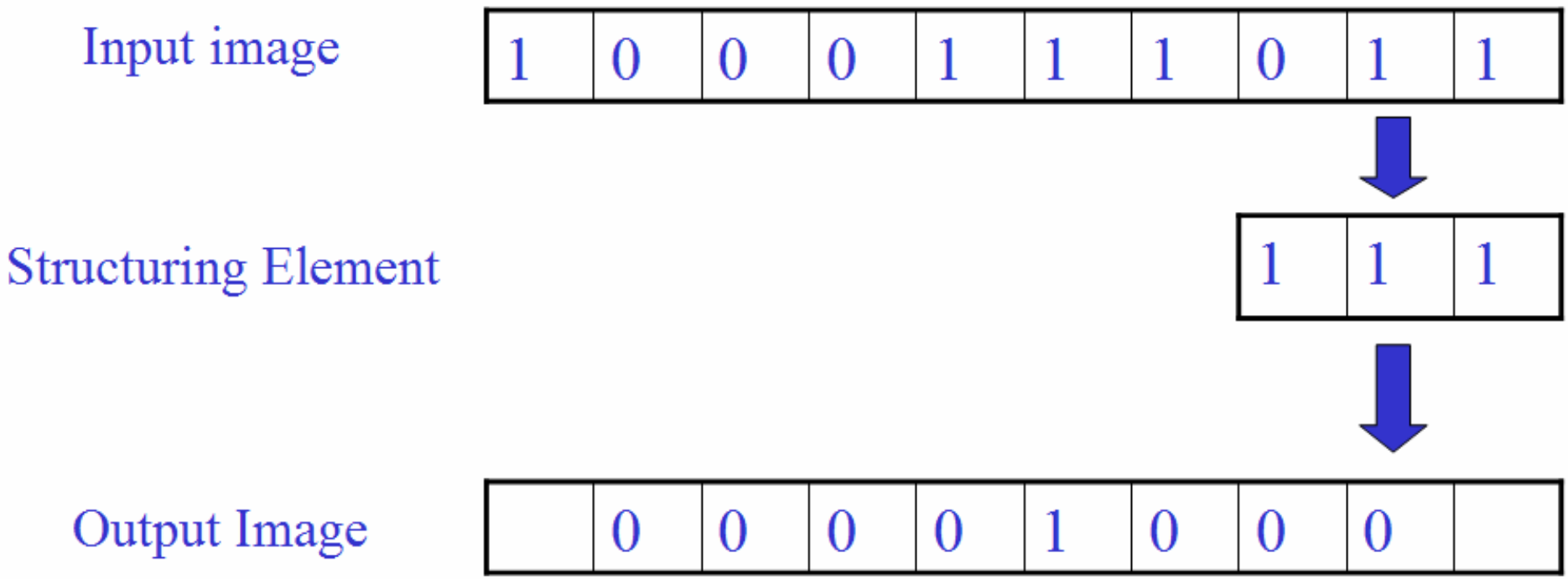
Example for Erosion



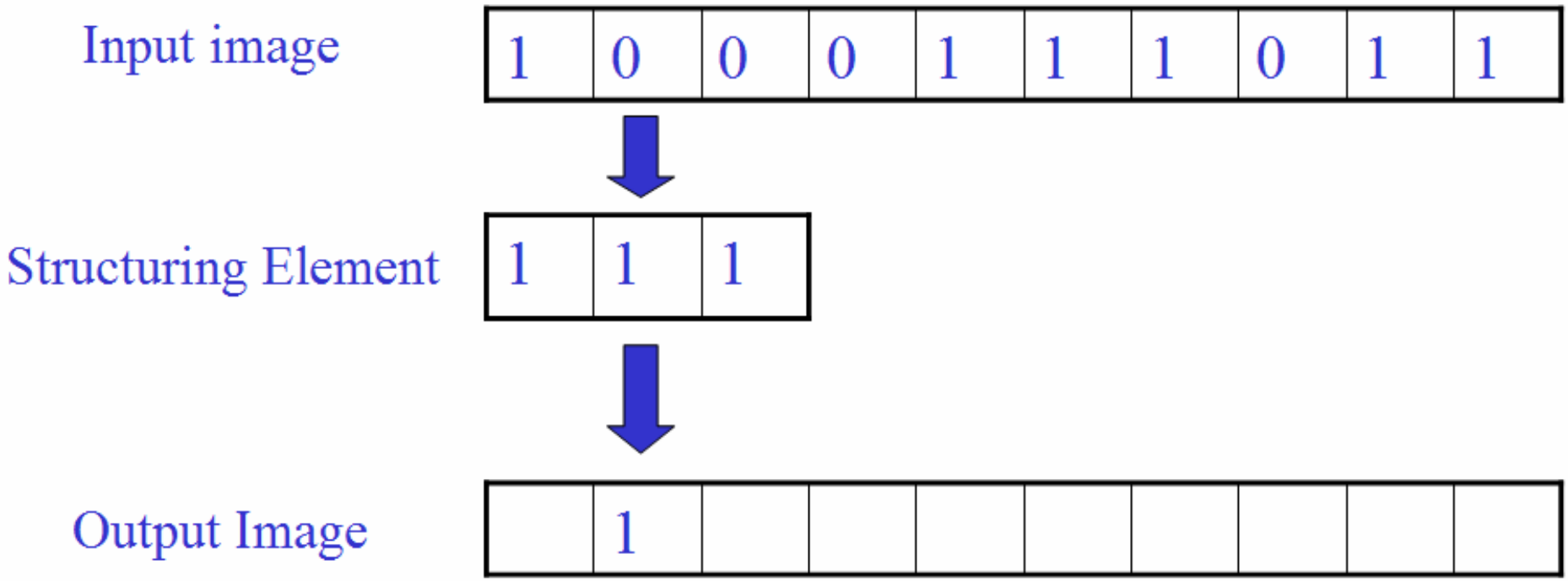
Example for Erosion



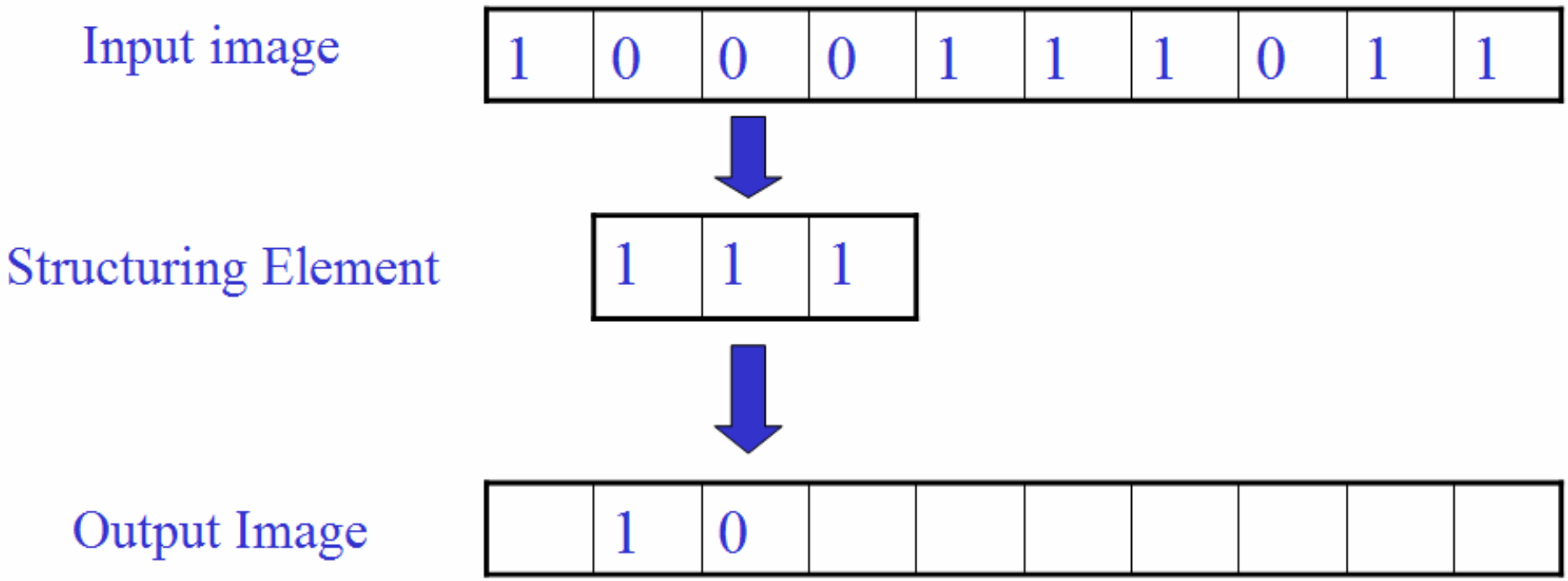
Example for Erosion



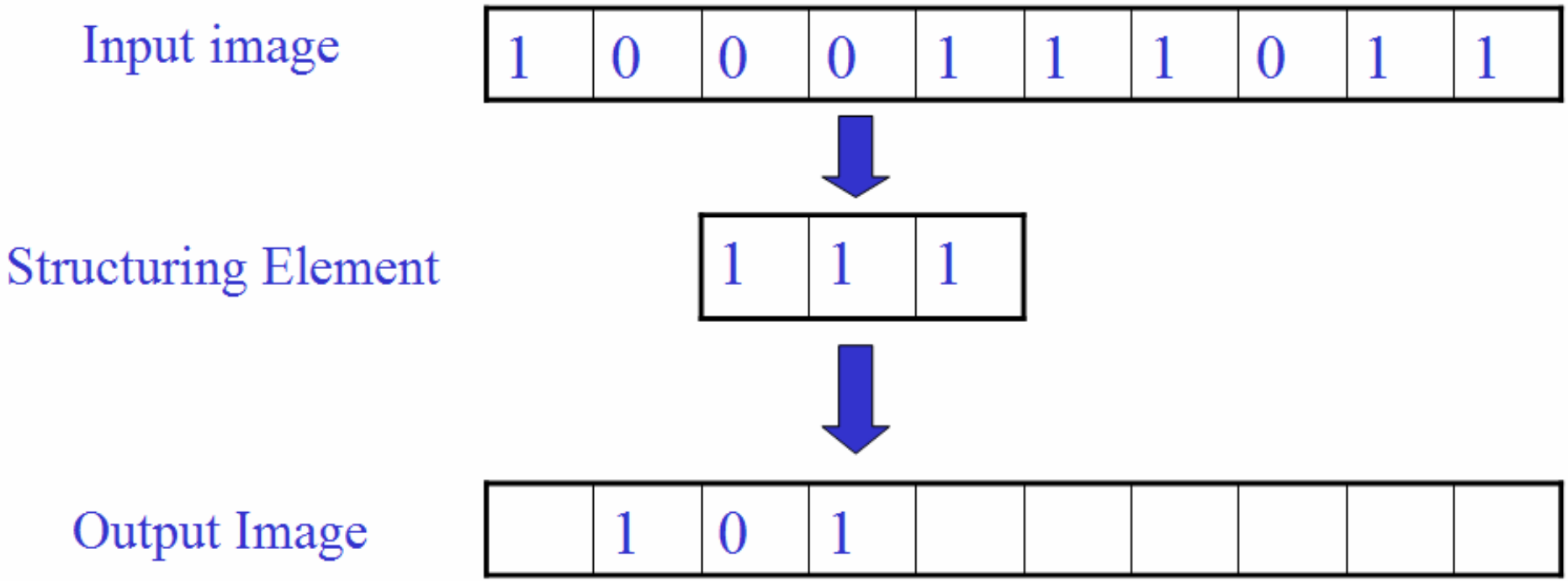
Example for Dilation



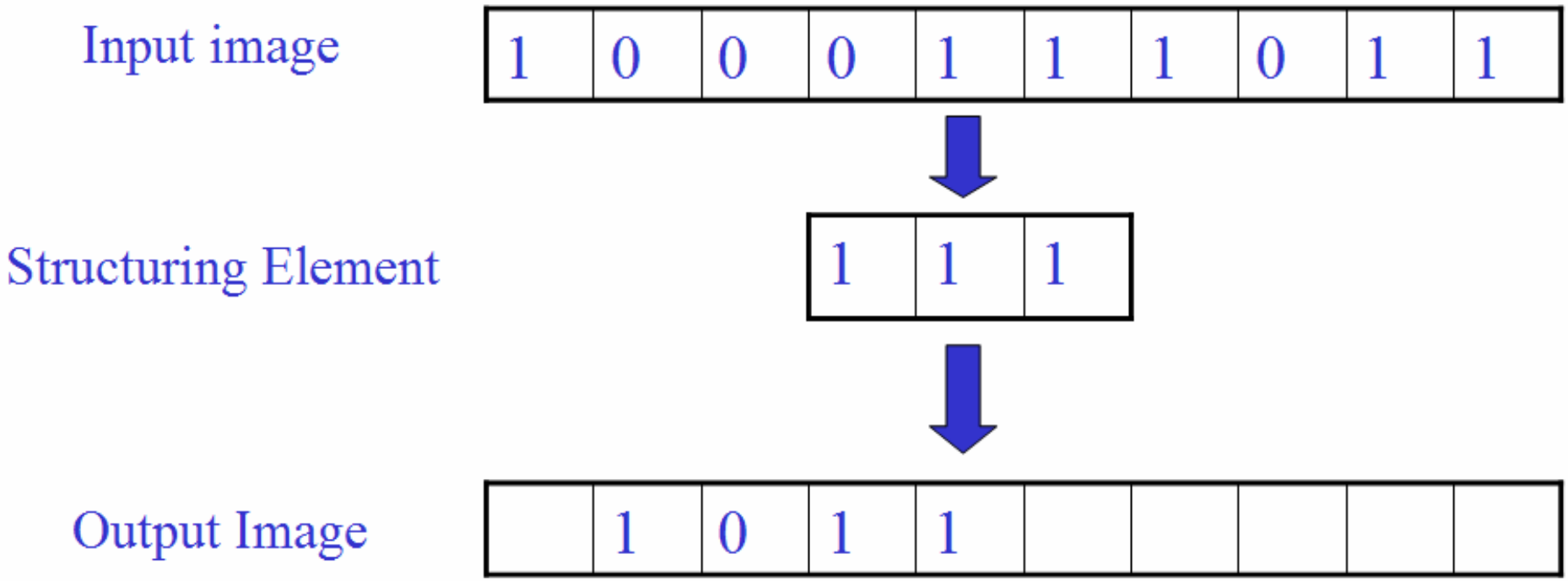
Example for Dilation



Example for Dilation



Example for Dilation



Dilation and Erosion as a Set Operation

A binary image can be thought of as a set of points on the Euclidean space ε^2.

Binary image $\longrightarrow$ Set X

Structuring element $\longrightarrow$ Set S

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

X={(2,1),(2,2),(3,2),(3,3)}

$$\text{Dilation}: X \oplus S = \left\{ p \in \varepsilon^2 : p = x + s, x \in X \text{ and } s \in S \right\}$$

$$\text{Erosion}: X \ominus S = \left\{ p \in \varepsilon^2 : p + s \in X, \forall s \in S \right\}$$

$$p = (p_1, p_2) = (x_1, s_1) + (x_2, s_2)$$

Opening and Closing

• Closing is the compound operation of dilation followed by erosion (with the same structuring element)

$$X \bullet S = (X \oplus S) \ominus S$$

Has similar effects with dilation; but preserves global shape.

(filling holes, gaps and gulfs)

• Opening is the compound operation of erosion followed by dilation (with the same structuring element)

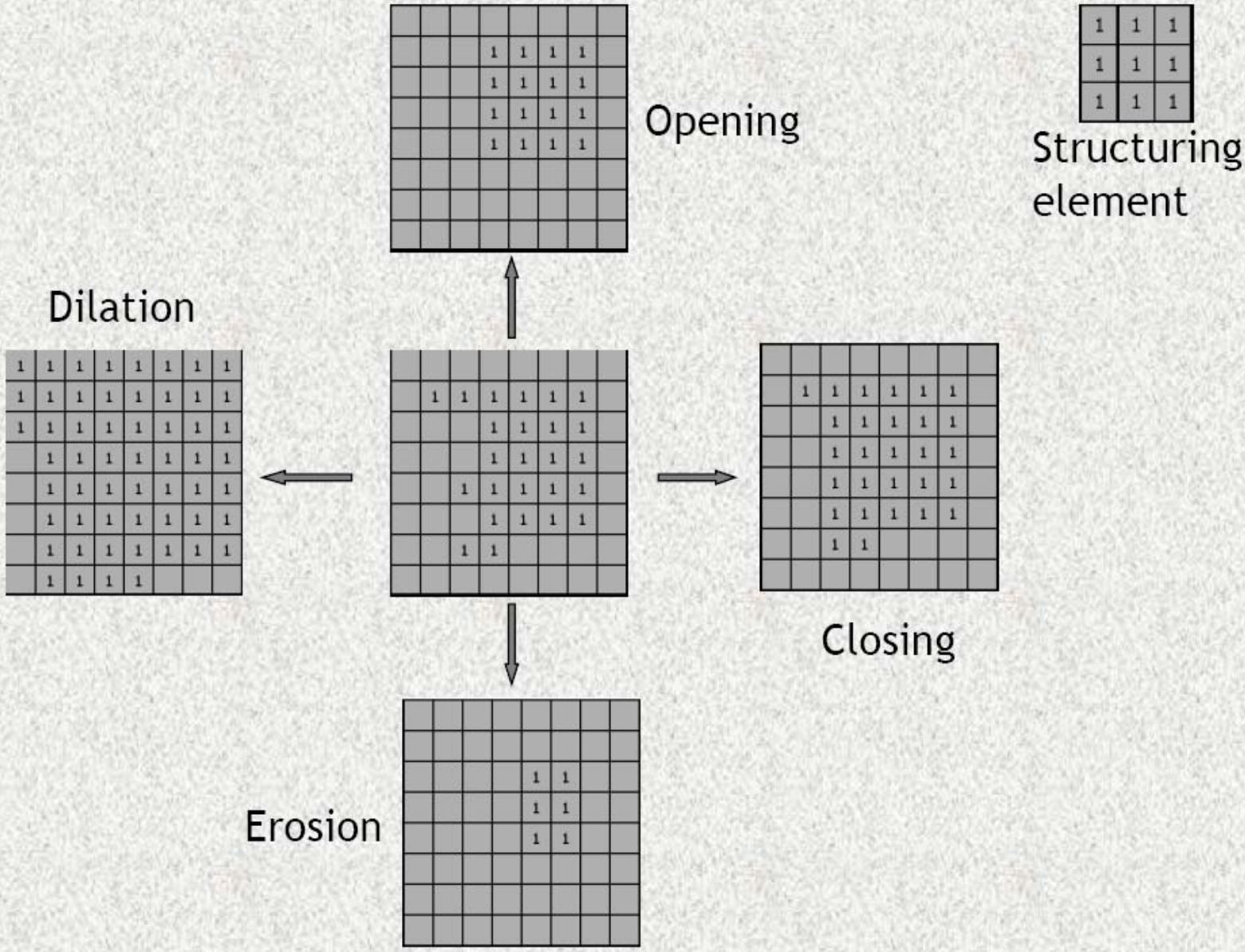
$$X \circ S = (X \ominus S) \oplus S$$

Has similar effects with erosion; but preserves global shape.

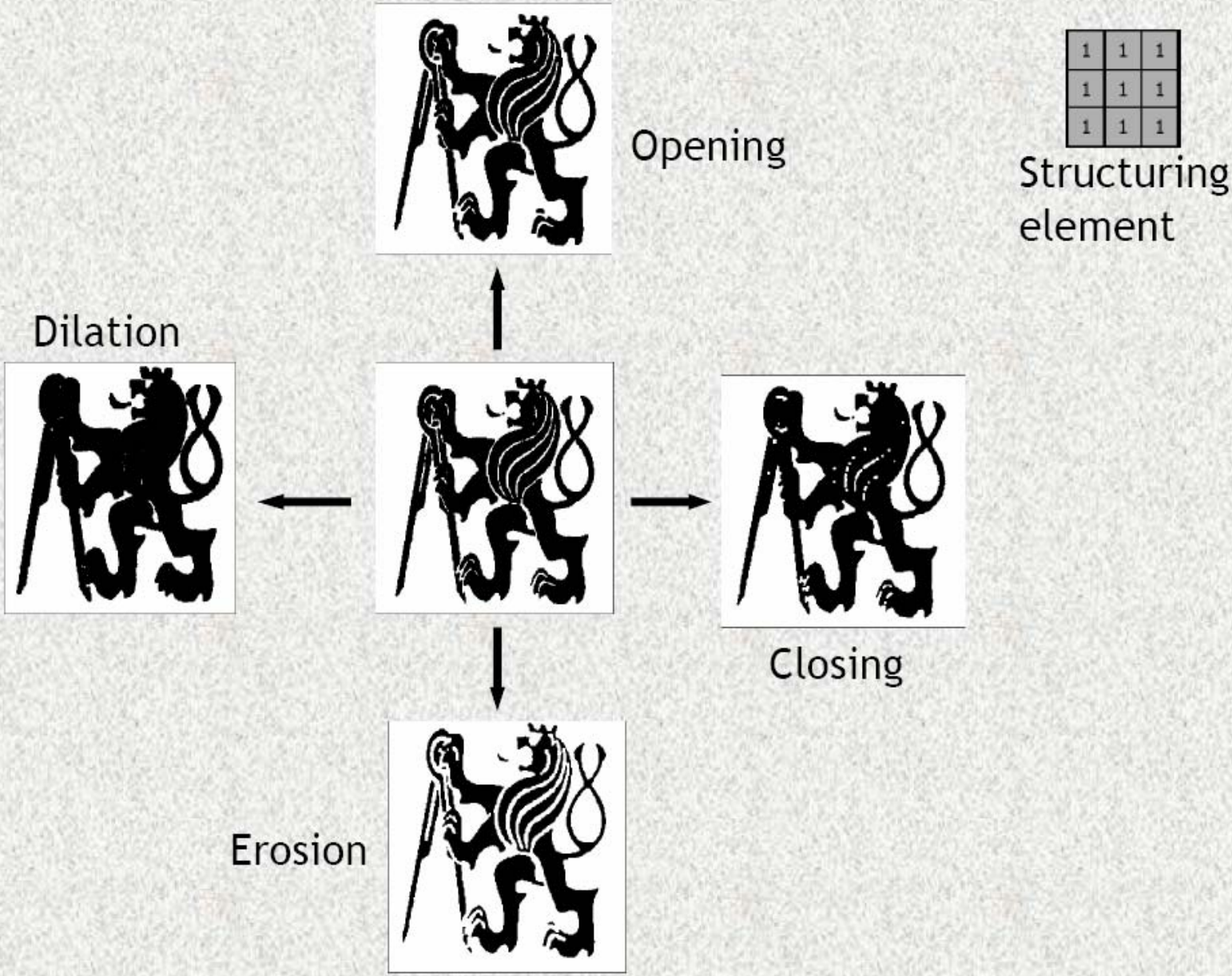
(removing bridges, branches and small protrusions)

Opening and closing are **idempotent**, i.e. successive application does not change the result.

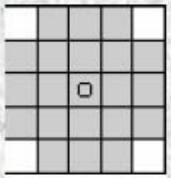
Example



Example


Opening
1 1 1
1 1 1
1 1 1
Structuring element
Dilation
Closing
Erosion

Example



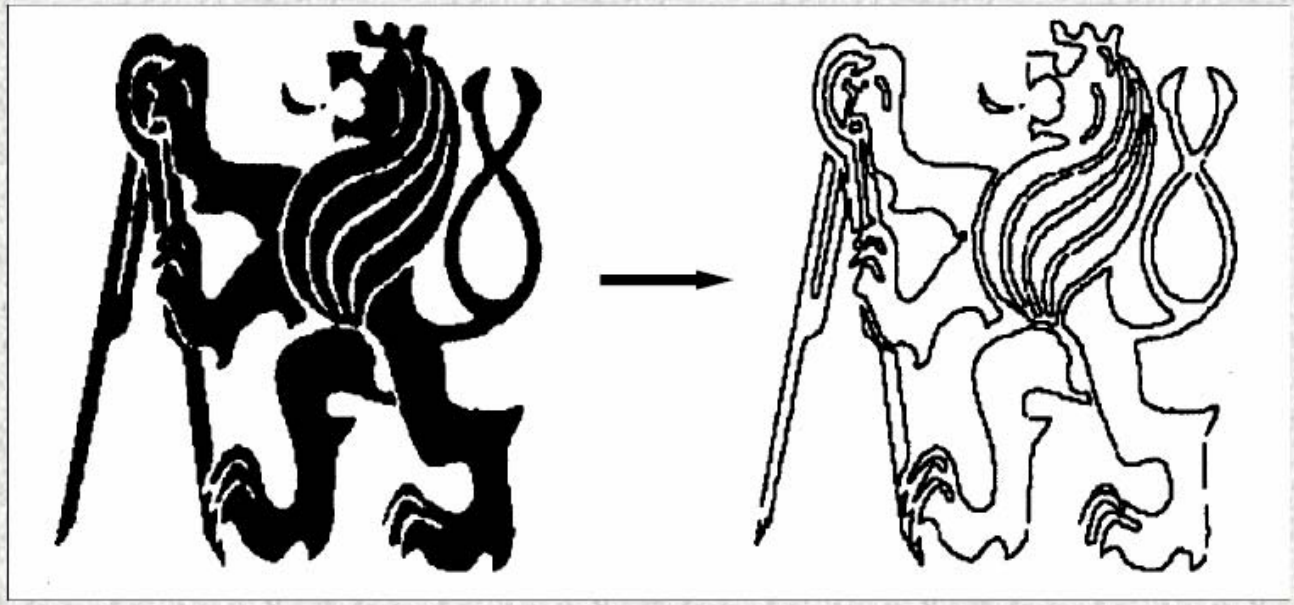
Original

Closing

Opening

Boundary extraction

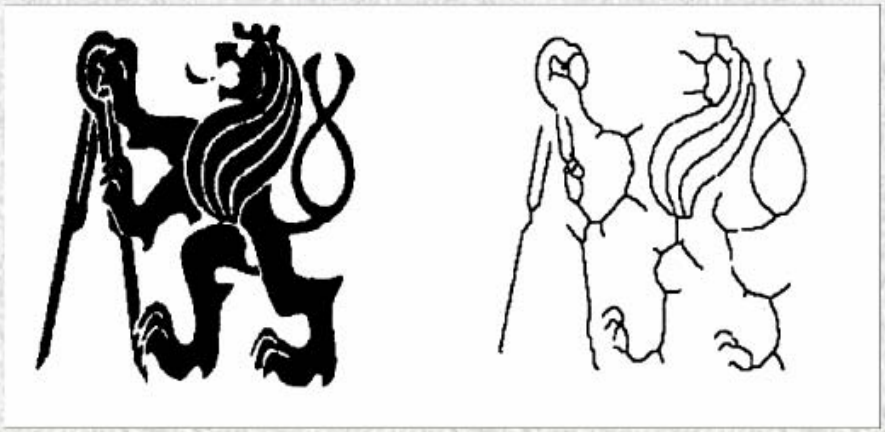
Contours can be extracted by subtraction of the eroded image from the original.



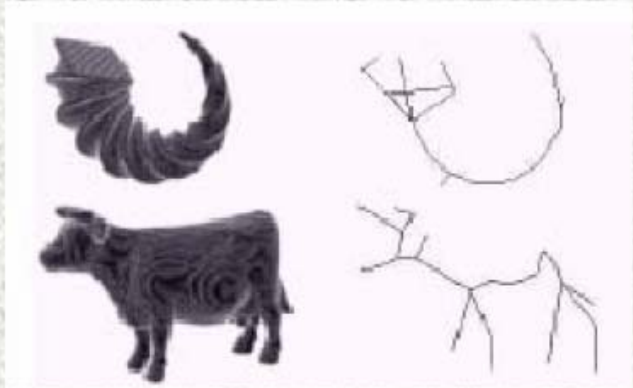
Skeletonization

Skeletonization: Converting a 2D or 3D object to a stick figure that represents its internal structure.

2D skeletons



3D skeletons

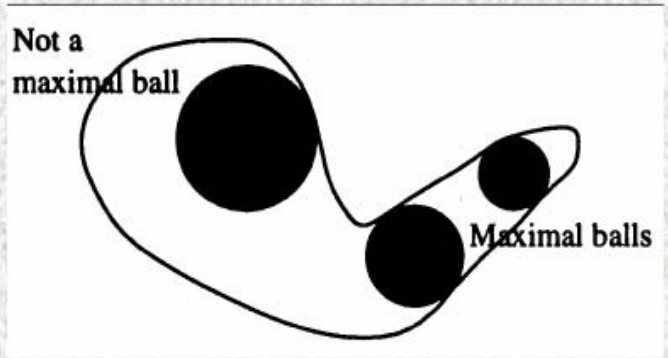


✓ Homotopic skeletons preserve connectivity and topology,

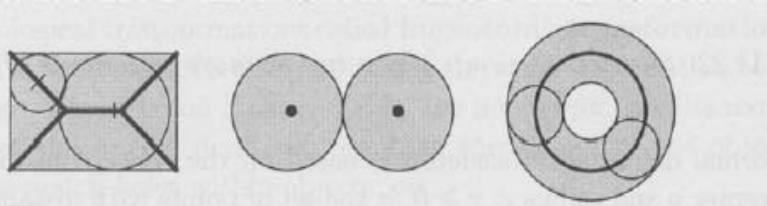
✓ Can be used for example in shape matching via graph matching algorithms.

Skeleton by Maximal Balls

The ball B included in a set X is maximal iff there is no larger ball included in X that contains B:



The skeleton by maximal balls S(X) of a set X is the set of centers of maximal balls:



Remark: A skeleton defined in this way is not necessarily homotopic and not necessarily of one pixel wide in the discrete plane.

Homotopic Skeletons

- A homotopic skeleton can be obtained by morphological operations.
- We need to define two composite operations:

  hit-or-miss and thinning

in terms of a composite structuring element $\boldsymbol{S}$:

$$\boldsymbol{S} = (S_1, S_2) \quad \text{such that} \quad S_1 \cap S_2 = \emptyset$$

Hit-or-miss

A variant of template matching that finds collections of pixels with certain shape properties (such as corners, border pixels).

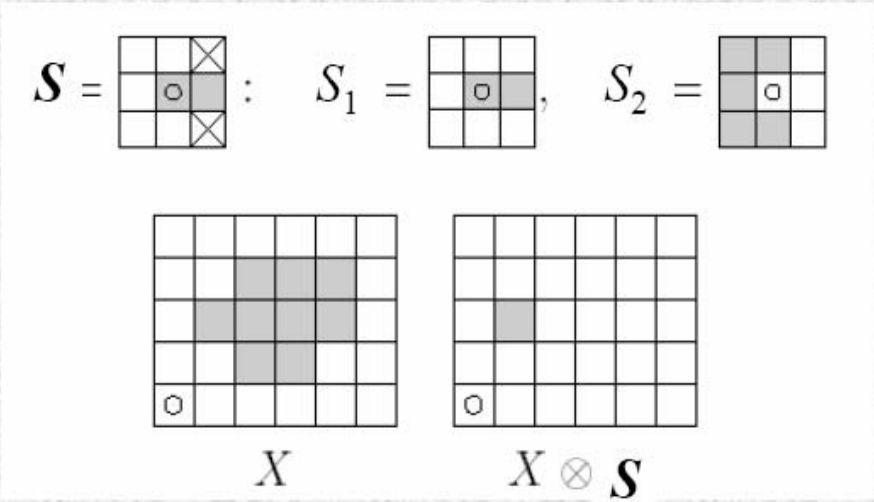
$$X \otimes \boldsymbol{S} = \left\{ x \mid S_1 \in X \text{ and } S_2 \in X^c \right\}$$
$$= (X \ominus S_1) \cap (X^c \ominus S_2)$$

Gives a part of the object boundary

- ✓ $X \ominus S_1$: locus of object pixels similar to S_1
- ✓ $X^c \ominus S_2$: locus of background pixels similar to S_2

Example:

Detection of "endpoints" from the left.



Thinning

When thinning, a part of the boundary of the object is subtracted from it by the set difference operation:

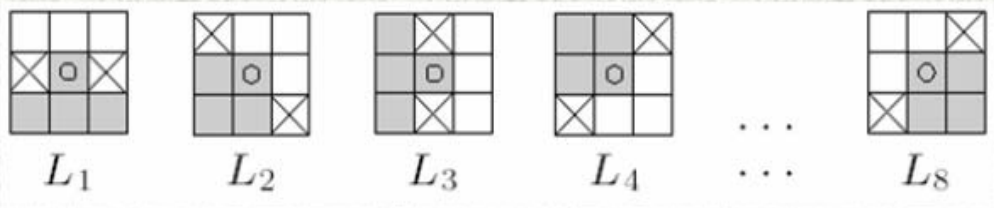
$$X \oslash \boldsymbol{S} = X / (X \otimes \boldsymbol{S})$$

A homotopic skeleton can be found by sequential thinning:

$$X \oslash \{\boldsymbol{S}_i\} = (((X \oslash \boldsymbol{S}_1) \oslash \boldsymbol{S}_2) \cdots \oslash \boldsymbol{S}_n)$$

Number n of iterations depends on the object type.

Golay alphabet of rotated elements



$\{\boldsymbol{L}_i\}_{i=1}^{8}$ repeat in order till convergence

Skeletonization by Sequential Thinning

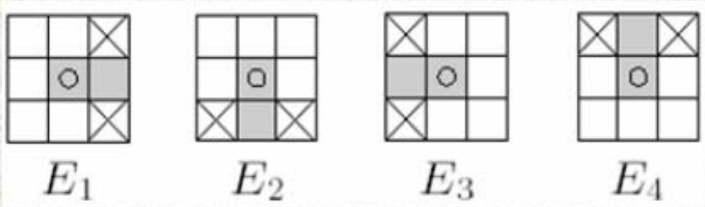
Sequential thinning using element L after five iterations

Homotopic skeleton after convergence

Skeleton Smoothing

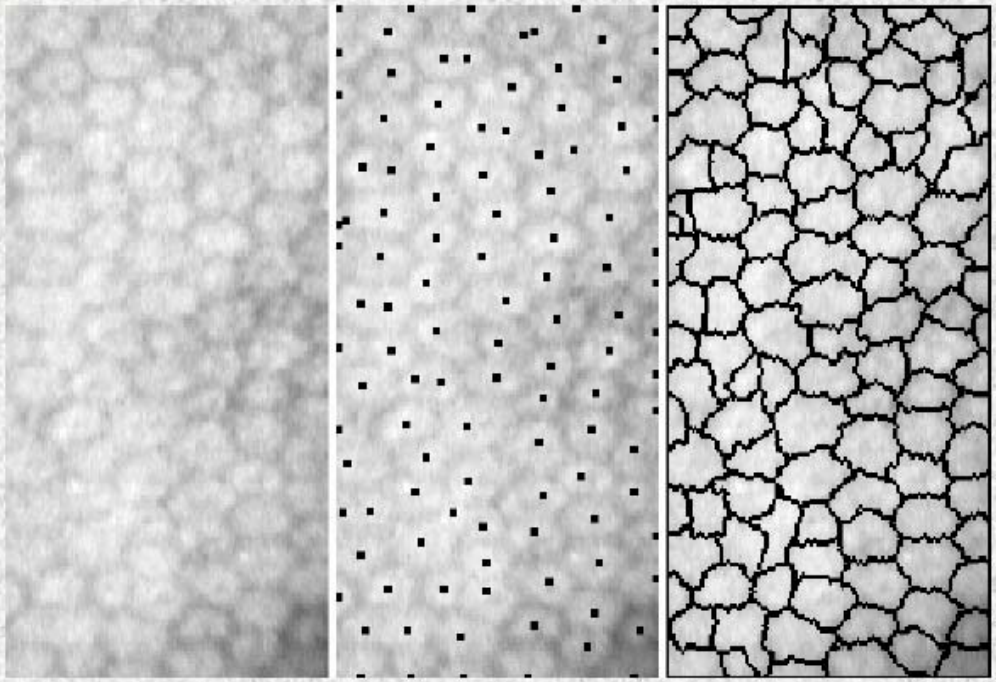
The skeleton is usually jagged; it can be smoothed by further sequential thinning with another series of structuring elements

After five iterations of sequential thinning by element E



Gray-scale Morphology

- Binary morphological operations are easily extendible to gray-scale images using min and max operations.
- Erosion (respectively, dilation) of an image is the operation of assigning to each pixel the minimum (maximum) value found over a neighborhood of the corresponding pixel.
- Application areas: Segmentation, noise removal, etc.



Example:
Segmentation of cell boundaries in the images of human cornea.

References (books)

- Machine Vision (Ramesh Jain, Rangachar Kasturi, Brian G. Schunck)
- Automated Visual Inspection (B. G. Batchelor, D. A. Hill, and D. C. Hodgson)
- Intelligent Image Processing in Prolog (B. G. Batchelor)

References (Web)

- http://bruce.cs.cf.ac.uk/bruce/Machine_vision_tutorial/Binary_image_processing.html
- University of North Dakota, Department of Computer Science, computer vision website
- http://www.machinevisiononline.org/public/articles/archivedetails.cfm?id=1396
- Computer Engineering Department, Koç University (course website)