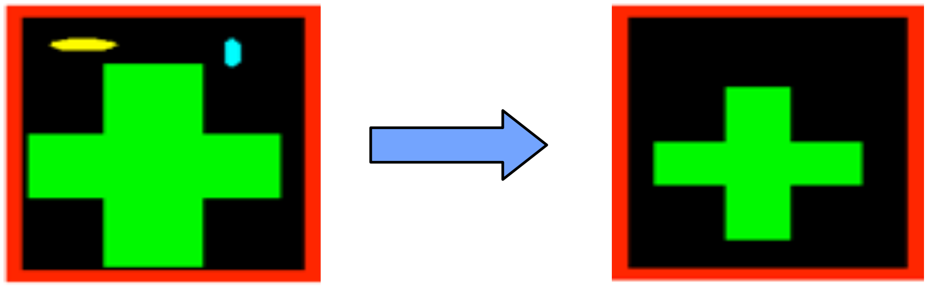
Basic tools of image analysis: Mathematical morphology and relatives

Laurent Jacques, Christophe De Vleeschouwer
LELEC2885

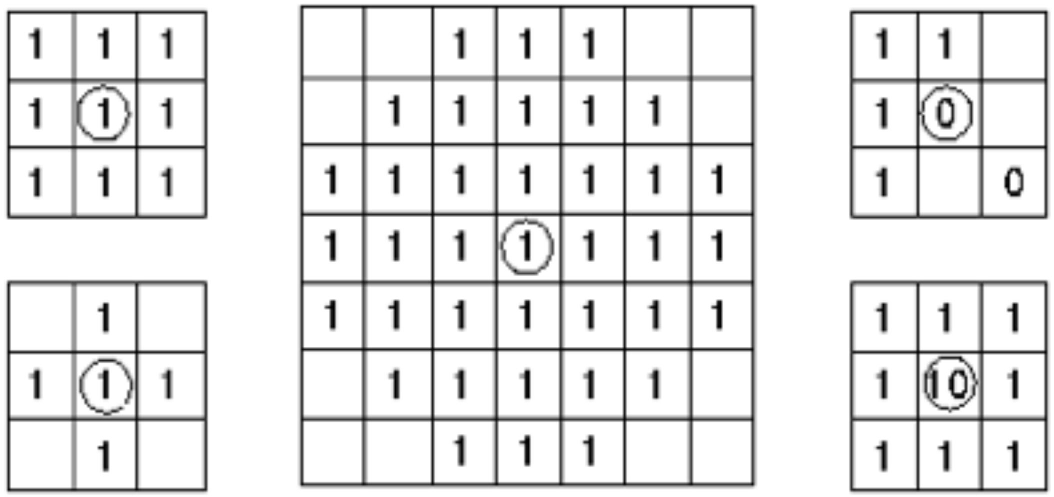
Abstract

Morphological operators

- Take a binary image and a *structuring element* as input and combine them using a set operator (intersection, union, inclusion, complement).
- Process objects in the input image based on characteristics of its shape, which are encoded in the structuring element.
- Can also be applied to gray-level images.
  - e.g. To reduce noise or to brighten the image
- Basic morphological operator
  - Foreground pixels (i.e. ones) and ‘don't care's’ in the structuring element
- Sophisticated morphological operator
  - Zeros as well as ones and ‘don't care's’ in the structuring element.



Structuring Elements



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		

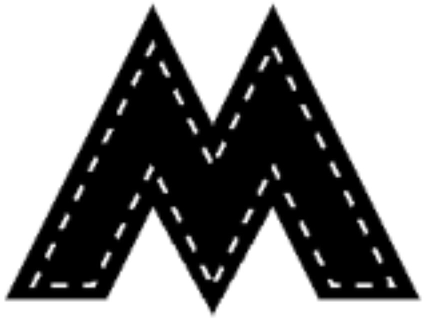
1	1	
1	0	
1		0

1	1	1
1	10	1
1	1	1

Examples of structuring elements

Dilation (I)

- **Brief Description**
  - One of the two basic operators
  - Basic effect
    - Gradually enlarge the boundaries of regions of foreground pixels on a binary image.



Common names: Dilate, Grow, Expand

Dilation (II)

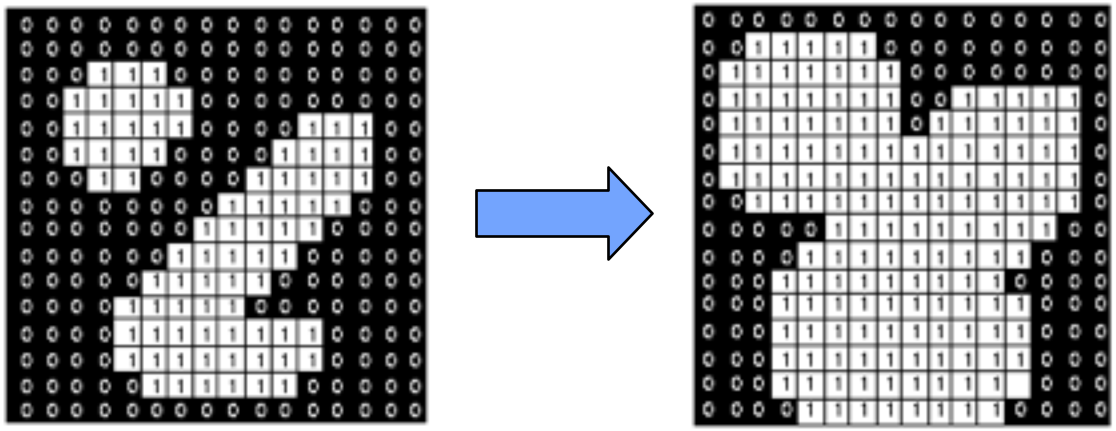
How It Works

- X : set of Euclidean coordinates corresponding to the input binary image
- K : set of coordinates for the structuring element
- K_x : translation of K so that its origin is at x.
- The dilation of X by K is simply the set of all points x such that the intersection of K_x with X is non-empty.

$$X \oplus K = \{x \mid K_x \cap X \neq \emptyset\}$$

Dilation (III)

- **Guideline for Use**



Effect of dilation using a 3×3 square structuring element

1	1	1
1	1	1
1	1	1

Set of coordinate points =
{ (-1, -1), (0, -1), (1, -1),
(-1, 0), (0, 0), (1, 0),
(-1, 1), (0, 1), (1, 1) }

3×3 square structuring element

Properties of Dilation

- **Commutative**

$$\boxed{A \oplus B = B \oplus A}$$

- **Associative**

$$\boxed{A \oplus (B \oplus C) = (A \oplus B) \oplus C}$$

- **Extensivity**

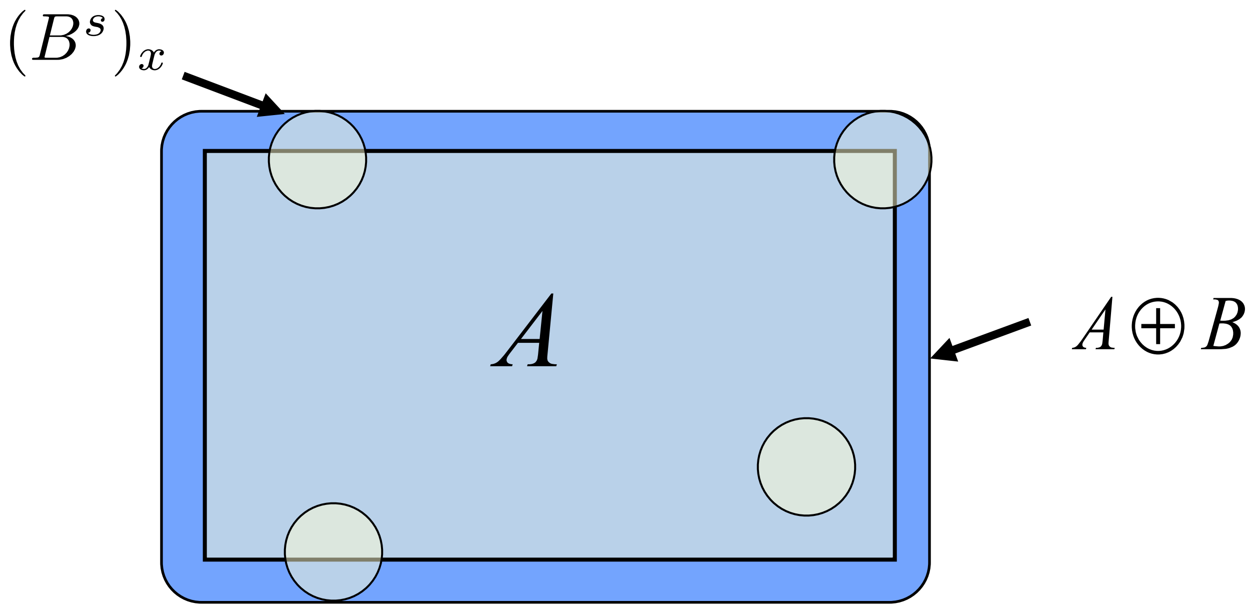
$$\boxed{if\ \ 0 \in B, A \subseteq A \oplus B}$$

- **Dilation is increasing**

$$\boxed{A \subseteq B\ implies\ \ A \oplus D \subseteq B \oplus D}$$

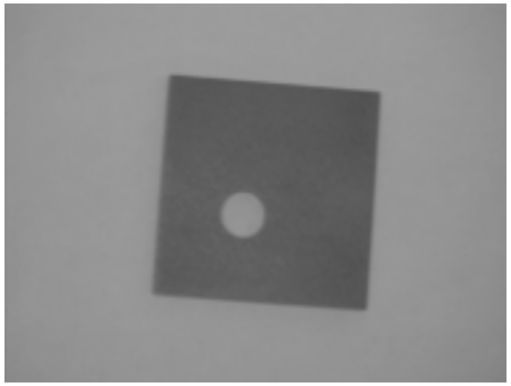
Dilation

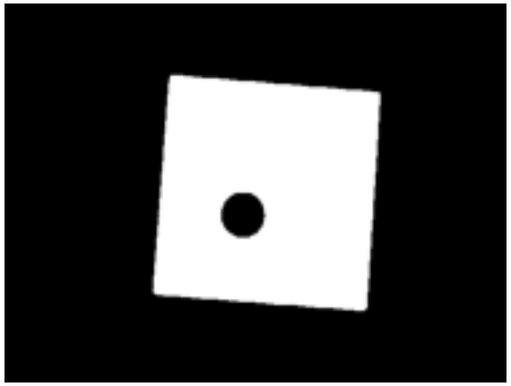
$$\boxed{A \oplus B = \bigcup_{b \in B} (A^s)_b = \bigcup_{a \in A} (B^s)_a}$$ with: $A^s = \{x \mid -x \in A\}$

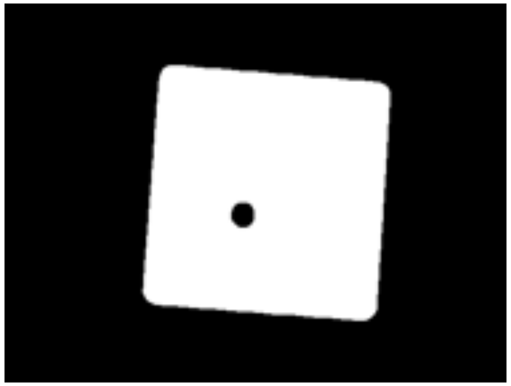


Dilation (IV)

- Example: Binary dilation


Original image

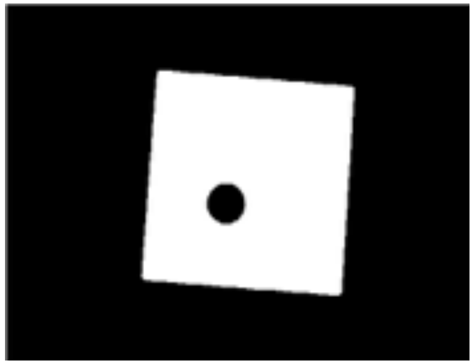

Thresholded image


Result of two Dilation passes using a disk shaped structuring element of 11 pixels radius

- Note that the corners have been rounded off.
- When dilating by a disk shaped structuring element, **convex** boundaries will become **rounded**, and **concave** boundaries will be **preserved** as they are.

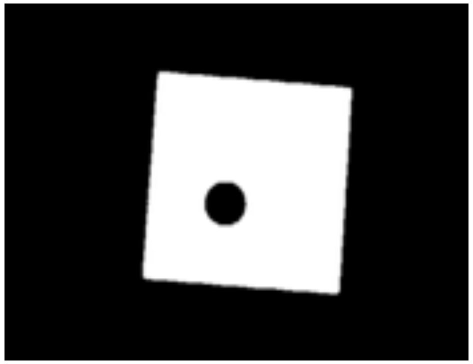
Dilation (V)

- Example: Binary dilation (*edge detection*)



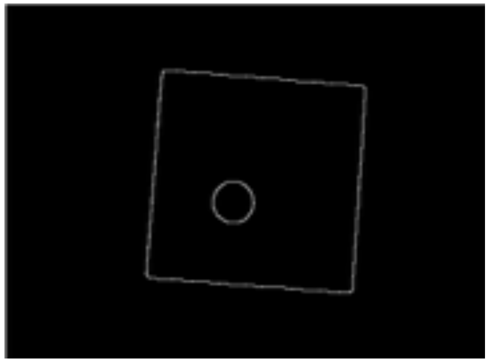
Result of dilation

−



Original image

=

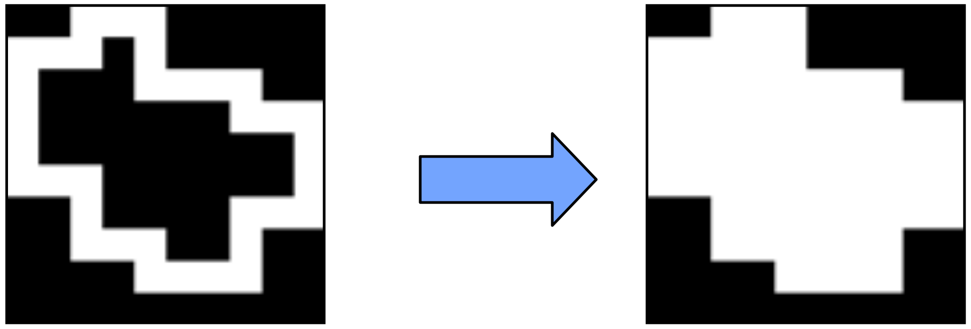


Edge detection

➜ Dilation can also be used for edge detection by taking the dilation of an image and then subtracting away the original image.

Dilation (VI)

- Example: Binary dilation (*Region Filling*)
  - Dilation is also used as the basis for many other mathematical morphology operators, often in combination with some logical operators.



Original image **Region filling**

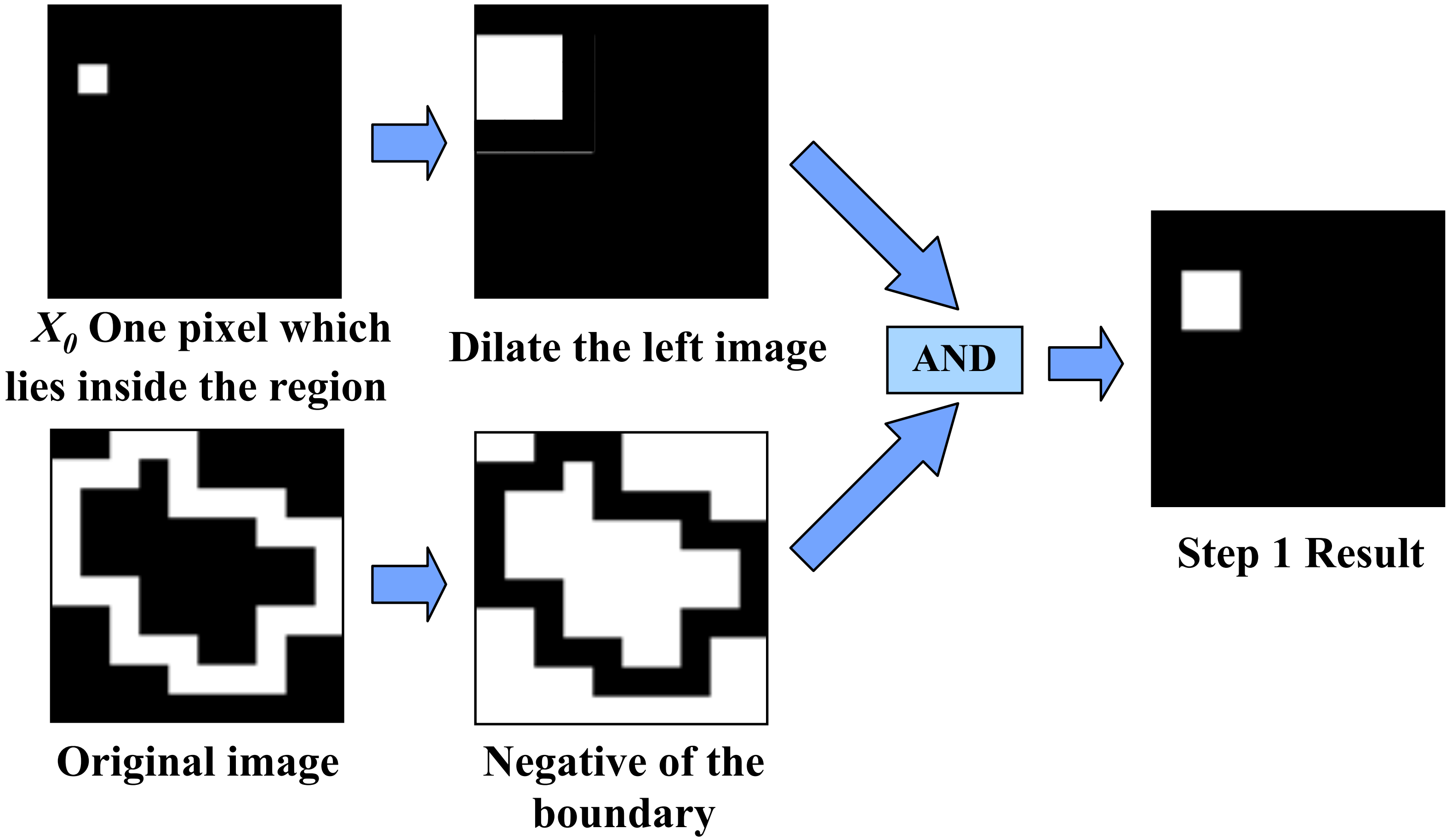
Dilation (VII)

- Conditional dilation
  - **Combination of the dilation operator and a logical operator**
  - **Region filling applies logical NOT, logical AND and dilation iteratively.**

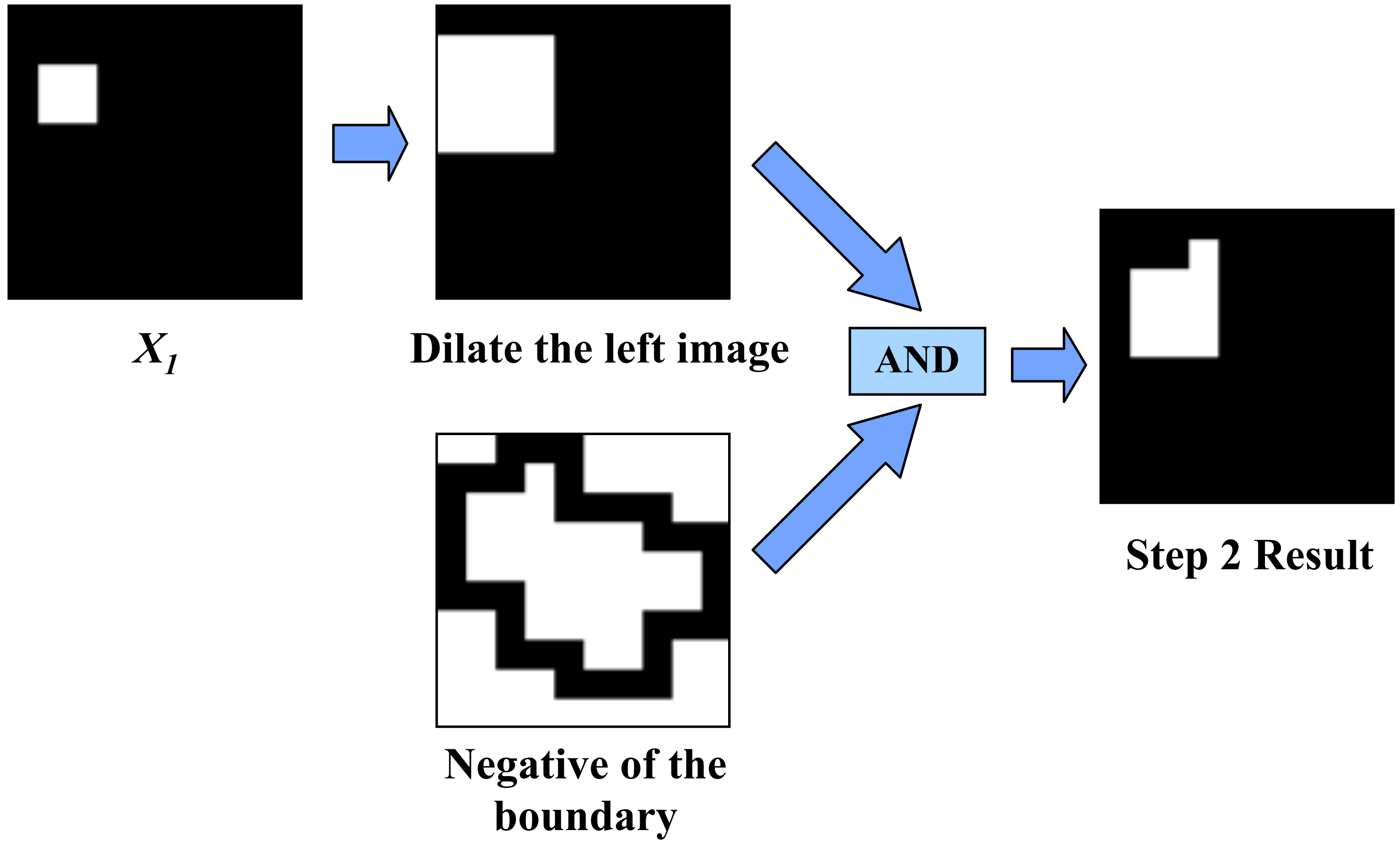
$$X_k = \mathrm{Dilate}(X_{k-1}, J) \cap A_{not}$$

X_k is the region which after convergence fills the boundary.
J is the structuring element.
A_{not} is the negative of the boundary.

Dilation (VIII)



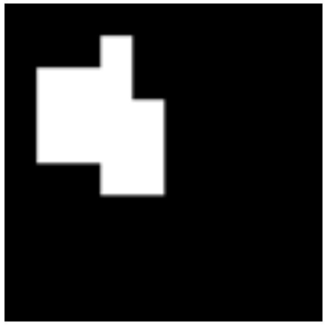
Dilation (IX)

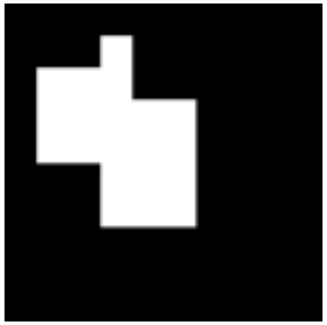


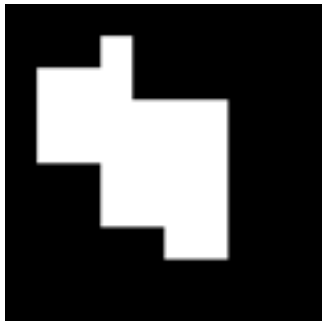
X_1
Dilate the left image
AND
Step 2 Result
Negative of the boundary

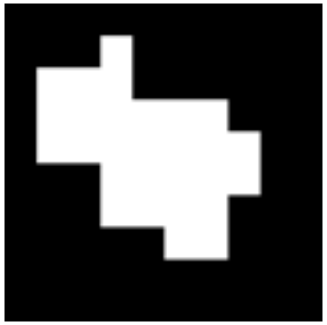
Dilation (X)

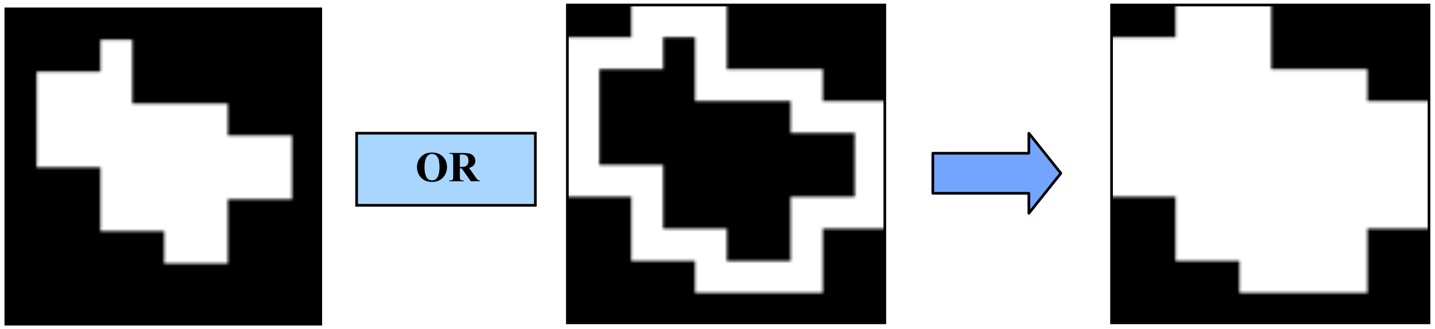
- Repeating dilation and AND with the inverted boundary until convergence, yields


Step 3 Result


Step 4 Result


Step 5 Result

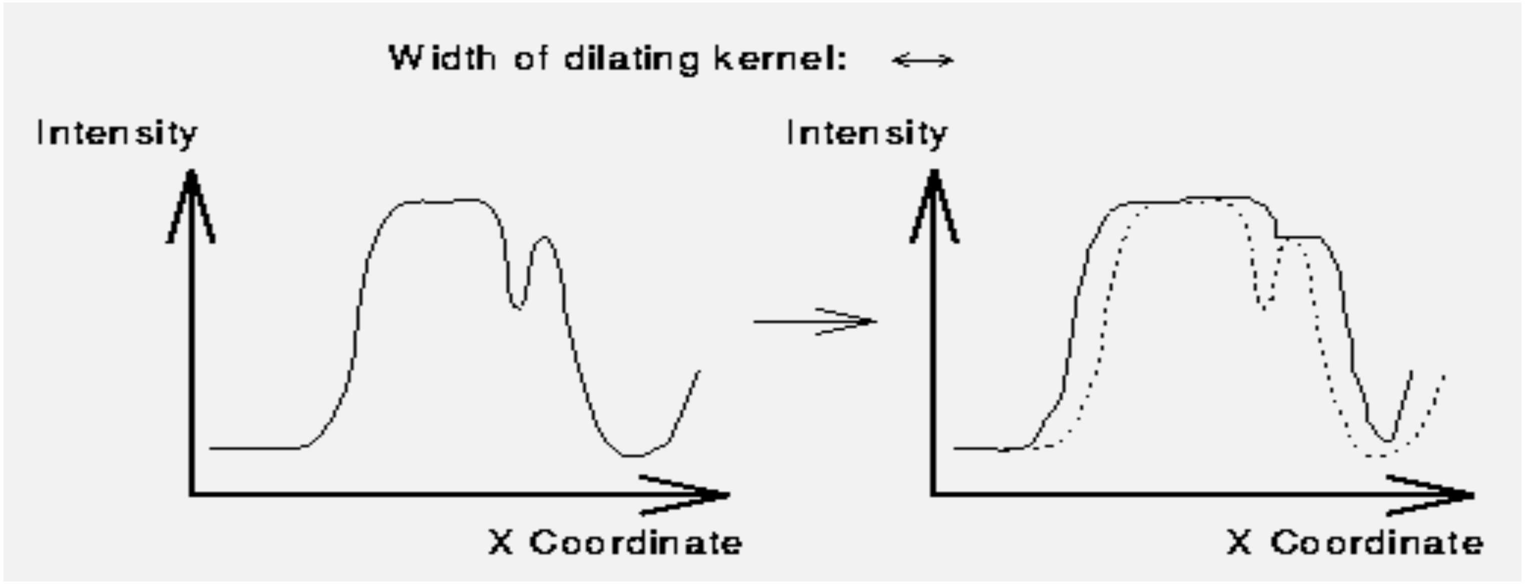

Step 6 Result



Final Result **Original image** **Result of Region Filling**

Dilation (XI)

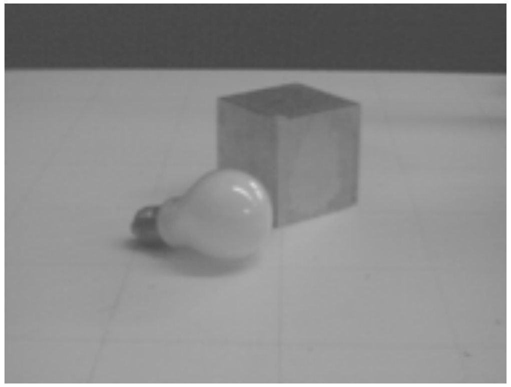
- **Grayscale Dilation** $(f \oplus b)(x) = \sup_{y \in E}[f(y) + b(x-y)]$
  - Generally brighten the image
    - Bright regions surrounded by dark regions grow in size, and dark regions surrounded by bright regions shrink in size.

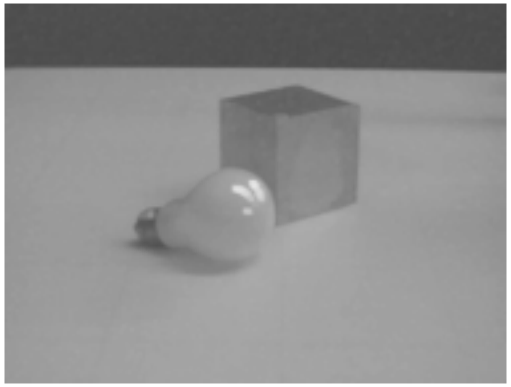


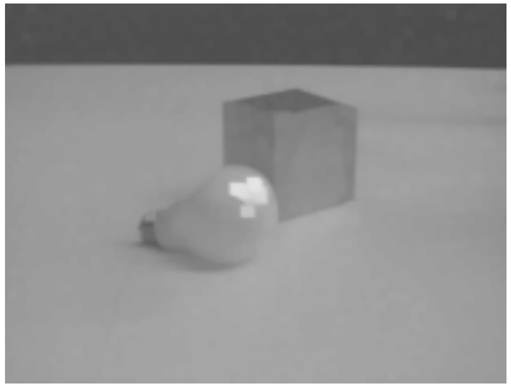
– **The effect of dilation using a disk shaped structuring element**

Dilation (XII)

- Example: Grayscale dilation (*brighten the image*)


Original image


Two passes Dilation by 3×3 flat square structuring element

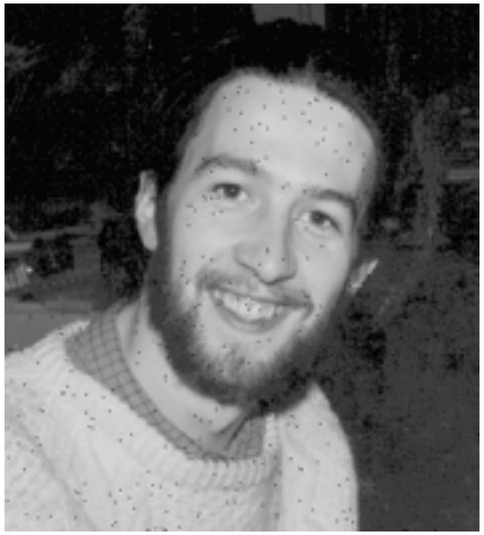

Five passes Dilation by 3×3 flat square structuring element

➜ The highlights on the bulb surface have increased in size.

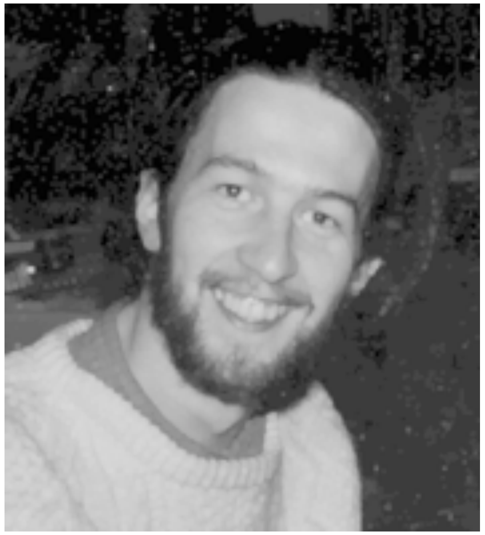
➜ The dark body of the cube has shrunk in size since it is darker than its surroundings.

Dilation (XIII)

- Example: Grayscale dilation (*Remove pepper noise*)



Pepper noise image

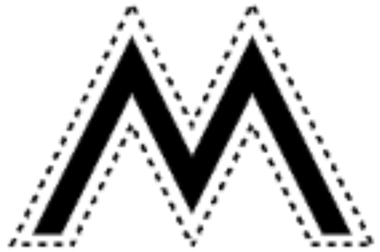


Dilation by 3×3 flat square structuring element

- The noise has been effectively removed.
- The image has been degraded significantly.

Erosion (I)

- **Brief Description**
  - Erosion is one of the basic operators in the area of mathematical morphology. Dual of dilation!
  - Basic effect
    - Erode away the boundaries of regions of foreground pixels (*i.e.* white pixels, typically).



Common names: Erode, Shrink, Reduce

Erosion (II)

- **How It Works**
  - X : set of Euclidean coordinates corresponding to input binary image
  - K : set of coordinates for the structuring element
  - K_x : translation of K so that its origin is at x
  - The erosion of X by K is simply the set of all points x such that K_x is a subset of X

$$X \ominus K = \{x \mid K_x \subseteq X\}$$

Properties of Erosion

- **Erosion is not commutative!**

$$\boxed{A \ominus B \neq B \ominus A}$$

- **Extensivity**

$$\boxed{\text{if } 0 \in B \text{ then } A \ominus B \subset A}$$

- **Dilation is increasing**

$$\boxed{\begin{array}{l} A \subseteq C \text{ implies } A \ominus B \;\subseteq\; C \ominus B \\ B \supseteq C \text{ implies } A \ominus B \;\subseteq\; A \ominus C \end{array}}$$

- **Chain rule**

$$A \ominus (B \oplus C) = (A \ominus B) \ominus C \Rightarrow$$

$$\boxed{A \ominus (B_1 \oplus \cdots \oplus B_k) = (\cdots(A \ominus B_1) \ominus \cdots \ominus B_k)}$$

Properties of Erosion

- **Translation Invariance**

$$\boxed{A_x \ominus B = (A \ominus B)_x \text{ and } A \ominus B_x = (A \ominus B)_{-x}}$$

- **Linearity**

$$\boxed{(A \cap B) \ominus C = (A \ominus C) \cap (B \ominus C)}$$

- **Containment**

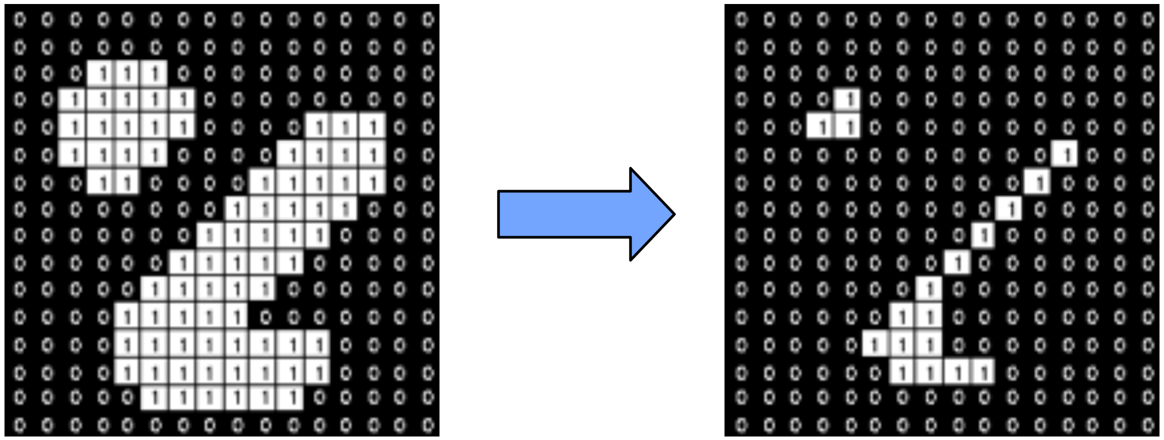
$$\boxed{(A \cup B) \ominus C \supseteq (A \ominus C) \cup (B \ominus C)}$$

- **Decomposition of structuring element**

$$\boxed{A \ominus (B \cup C) = (A \ominus B) \cap (A \ominus C)}$$

Erosion (III)

- **Guideline for Use**



Effect of erosion using a 3×3 square structuring element

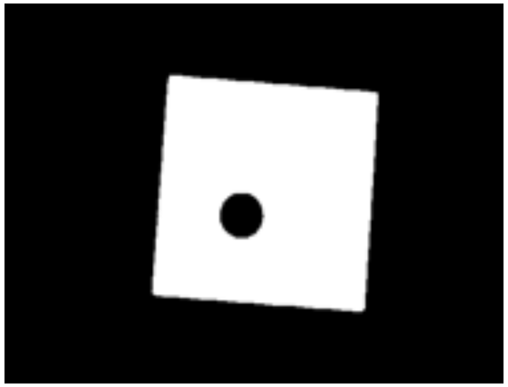
1	1	1
1	1	1
1	1	1

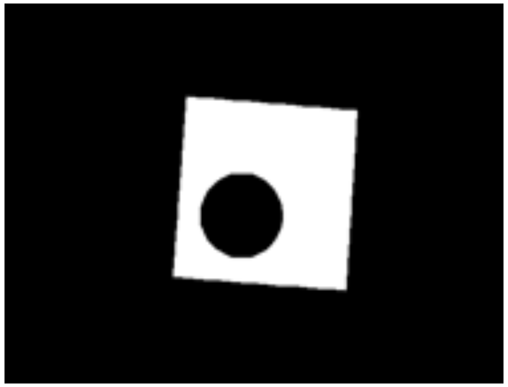
Set of coordinate points =
{ (-1, -1), (0, -1), (1, -1),
(-1, 0), (0, 0), (1, 0),
(-1, 1), (0, 1), (1, 1) }

A 3×3 square structuring element

Erosion (IV)

- Example: Binary erosion


Original thresholded image

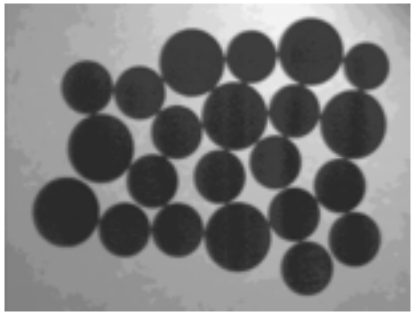

Result of erosion four times with a disk shaped structuring element of 11 pixels in diameter

➔ It shows that the hole in the middle of the image increases in size as the border shrinks.

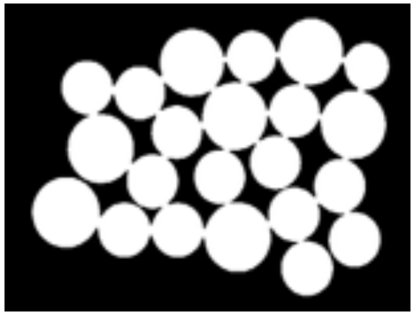
➔ Erosion using a disk shaped structuring element will tend to **round concave** boundaries, but will **preserve** the shape of **convex** boundaries.

Erosion (V)

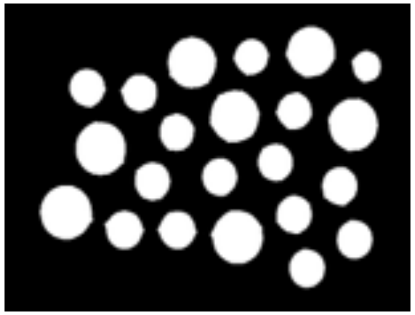
- Example: Binary erosion (*separate touching objects*)



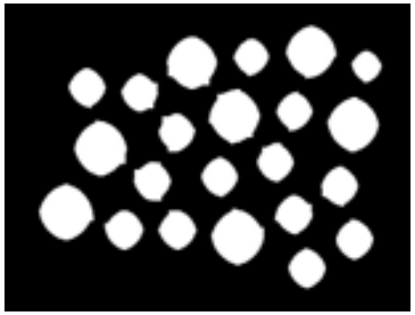
Original image (a number of dark disks)



Inverted image after thresholding



The result of eroding twice using *a disk shaped structuring element 11 pixels in diameter*



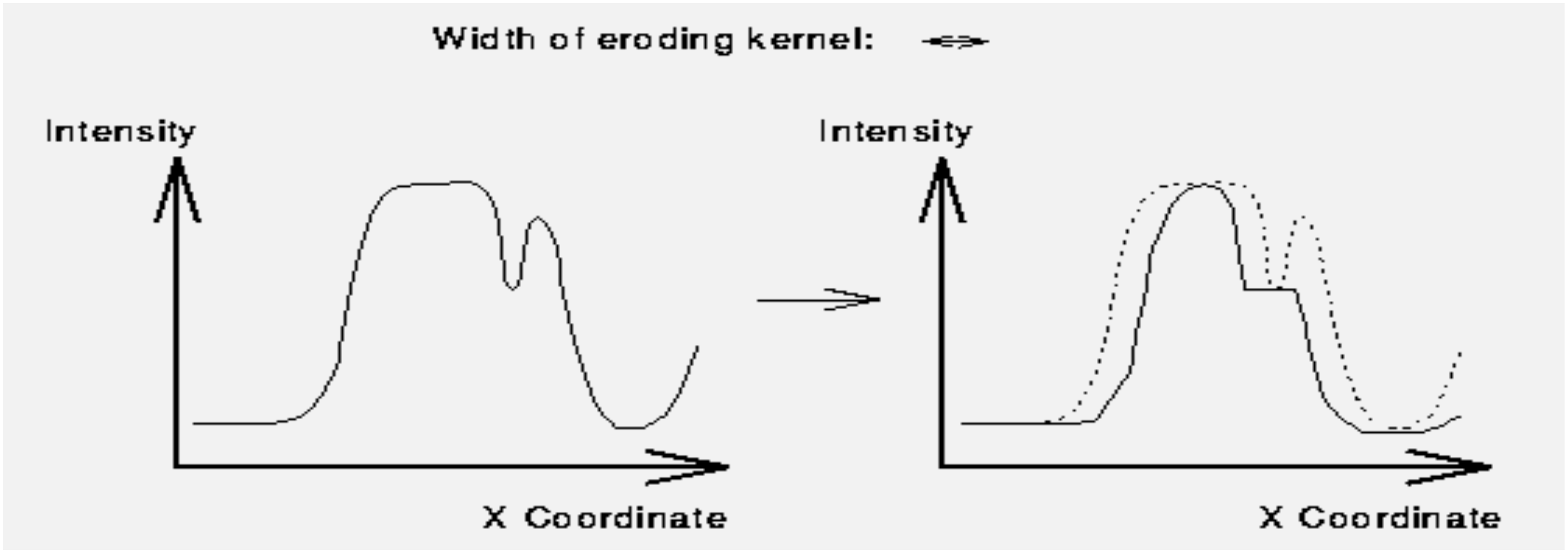
Using *9×9 square structuring element* leads to distortion of the shapes

Erosion (VI)

- Grayscale erosion

$$(f \ominus b)(x) = \inf_{y \in E} [f(y) - b(y - x)]$$

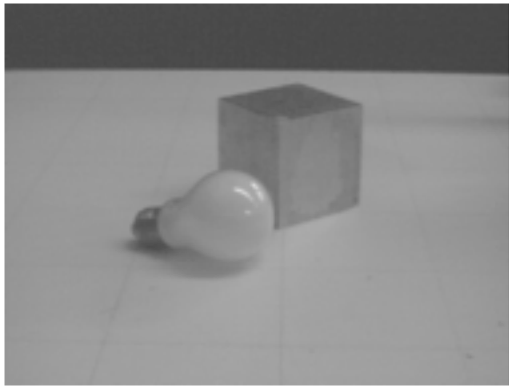
  - Generally <u>darken</u> the image
    - **Bright regions surrounded by dark regions shrink in size, and dark regions surrounded by bright regions grow in size.**

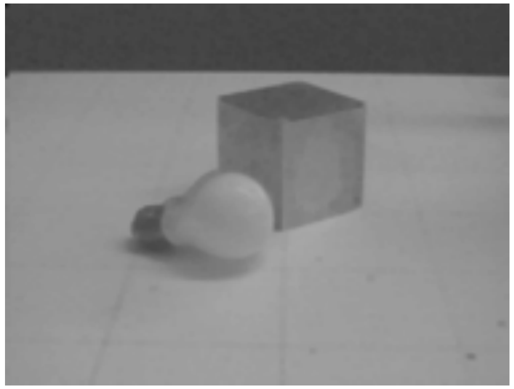


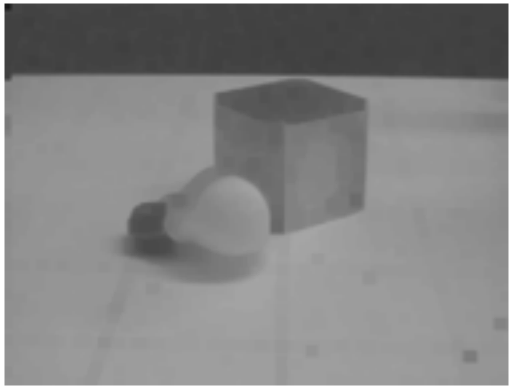
* **The effect of *erosion* using a disk shaped structuring element**

Erosion (VII)

- Example: Grayscale erosion (*darken the image*)


Original image

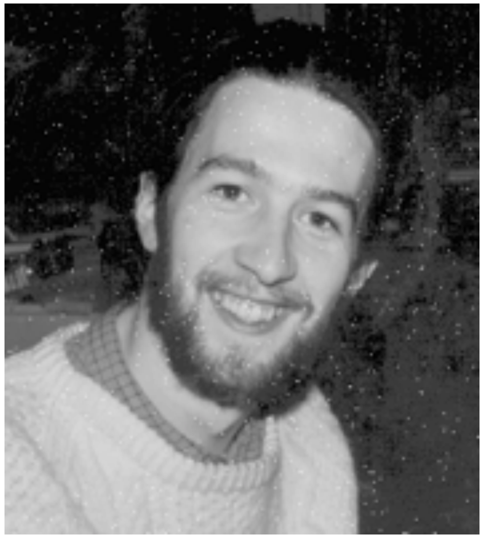

Two passes Erosion by 3×3 flat square structuring element


Five passes Erosion by 3×3 flat square structuring element

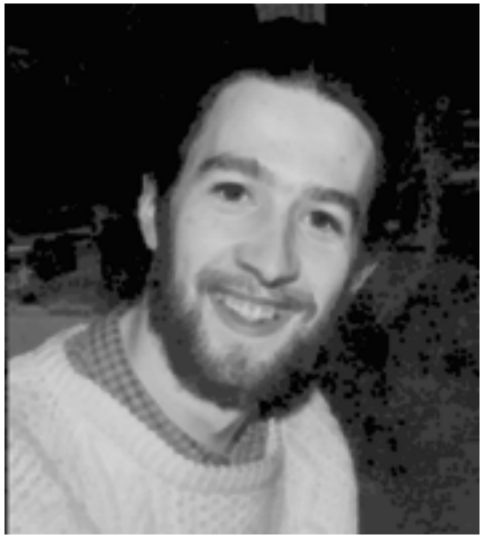
- ➔ The highlights have disappeared.
- ➔ The body of the cube has grown in size since it is darker than its surroundings.

Erosion (VIII)

- Example: Grayscale erosion (*Remove salt noise*)



Salt noise image



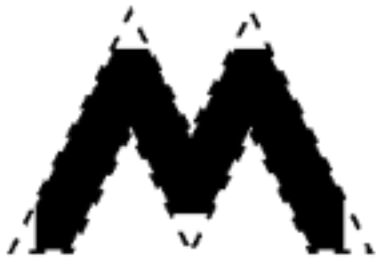
Erosion by 3×3 flat square structuring element

➔ The noise has been removed.

➔ The rest of the image has been degraded significantly.

Opening (I)

- **Brief Description**
  - The Basic Effect
    - Somewhat like erosion in that it tends to remove some of the foreground(bright) pixels from the edges of regions of foreground pixels.
    - To preserve *foreground* regions that have a similar shape to structuring element.



Common names: Opening

Opening (II)

- **How It works**
  - Opening is defined as an erosion followed by a dilation.

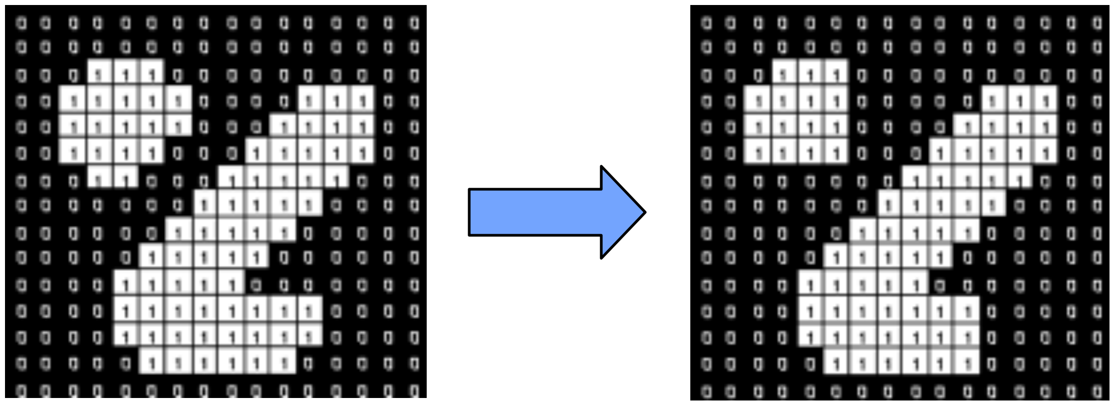
$$X \circ K = (X \ominus K) \oplus K$$

    - Gray-level opening consists simply of a gray-level erosion followed by a gray-level dilation.
  - Opening is the dual of closing.
    - Opening the foreground pixels with a particular structuring element is equivalent to closing the background pixels with the same element.

Opening (III)

Guidelines for Use

- Idempotence
  - After the opening has been carried out, the new boundaries of foreground regions will all be such that the structuring element fits inside them.
  - So further openings with the same element have no effect

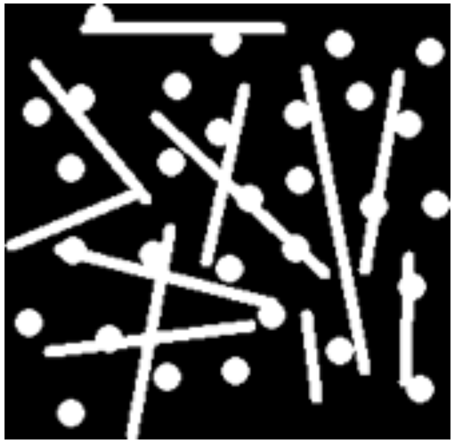


  - Effect of opening using a 3×3 square structuring element

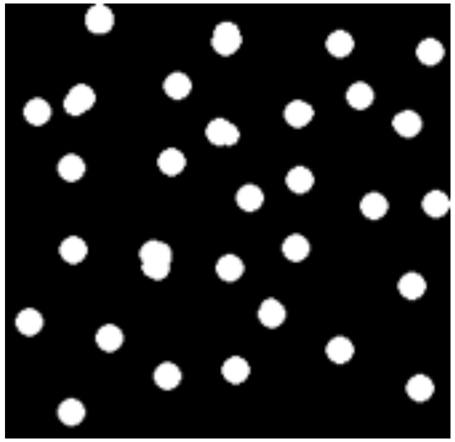
Opening (IV)

Binary Opening Example

- Separate out the circles from the lines



A mixture of circle and lines

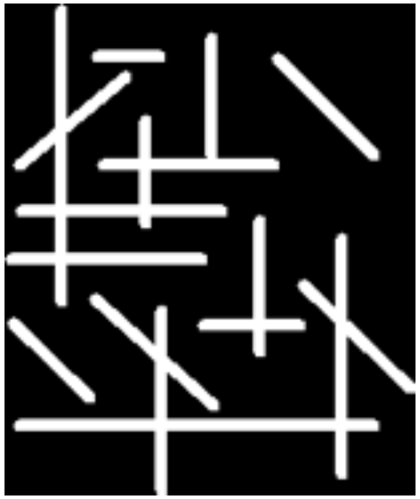


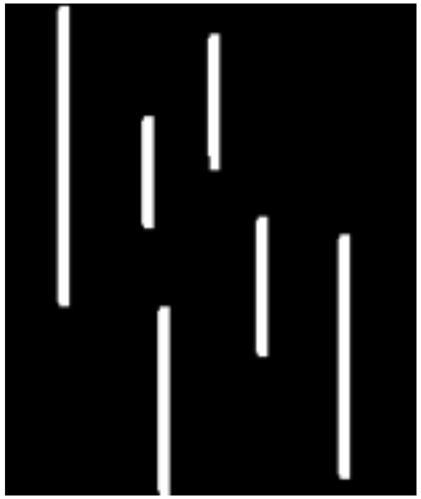
Opening with a disk shaped structuring element with 11 pixels in diameter

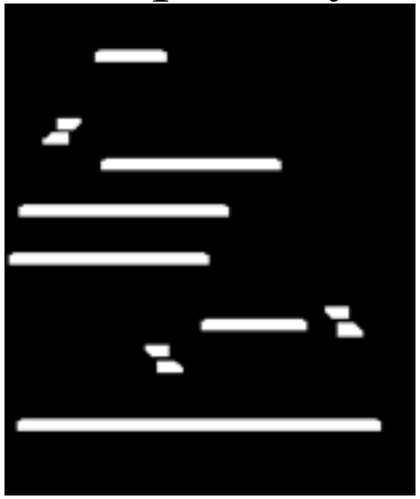
➔ The lines have been almost completely removed while the circles remain almost completely unaffected.

Opening (V)

- **Binary Opening Example**
  - Extract the horizontal and vertical lines separately


Original image

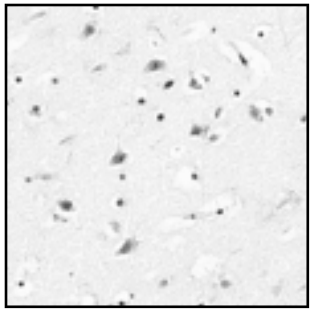

The result of an Opening with 3×9 vertically oriented structuring element

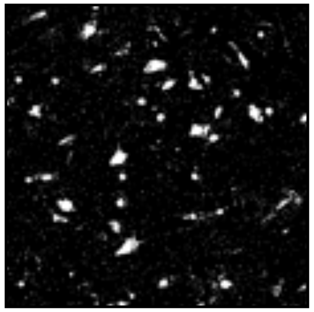

The result of an Opening with 9×3 horizontally oriented structuring element

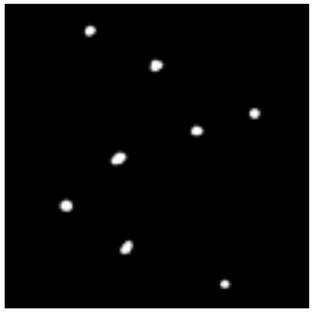
- **There are a few glitches in rightmost image where the diagonal lines cross vertical lines.**
- **These could easily be eliminated, however, using a slightly longer structuring element.**

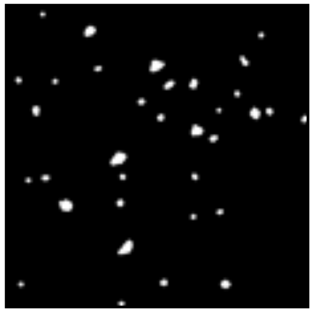
Opening (VI)

- Example: Binary opening


Original image

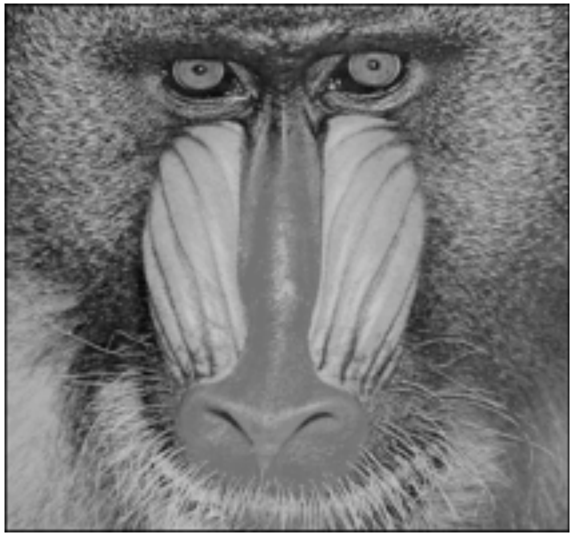

Inverted image after thresholding


The result of an Opening with
<u>11 pixel circular</u> structuring element

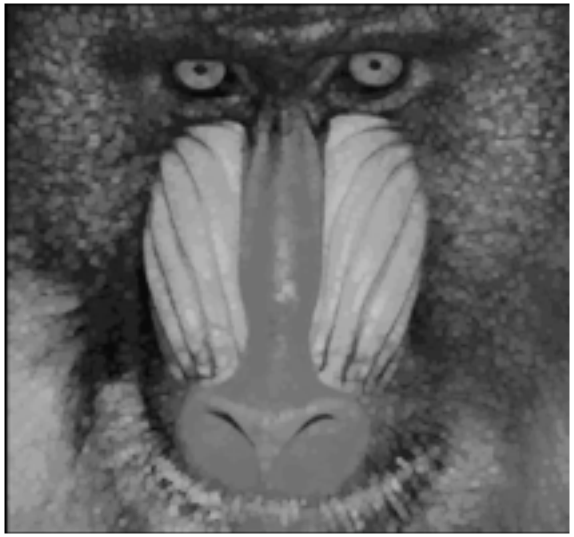

The result of an Opening with
<u>7 pixel circular</u> structuring element

Opening (VII)

- Example: Grayscale opening



• Original image

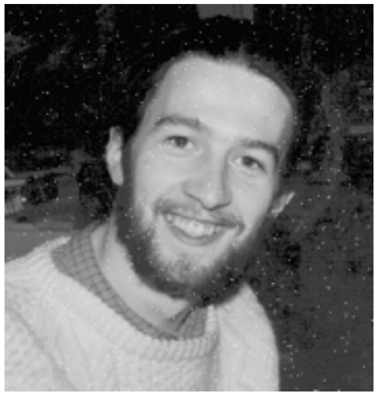


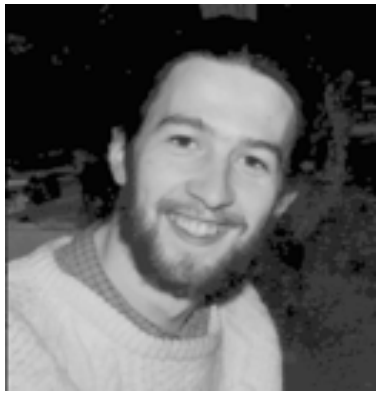
• Opening with a flat 5 ×5 square structuring element

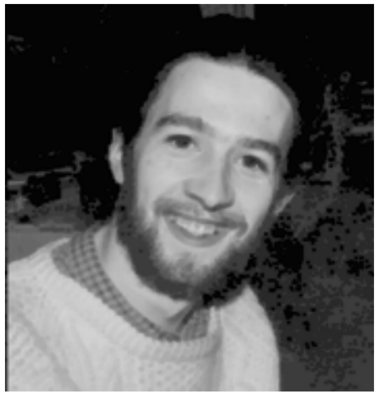
- **The important thing to notice here is the way in which bright features smaller than the structuring element have been greatly reduced in intensity.**
- **The fine grained hair and whiskers have been much reduced in intensity.**

Opening (VIII)

- **Compare Opening with Erosion**


Salt noise Image

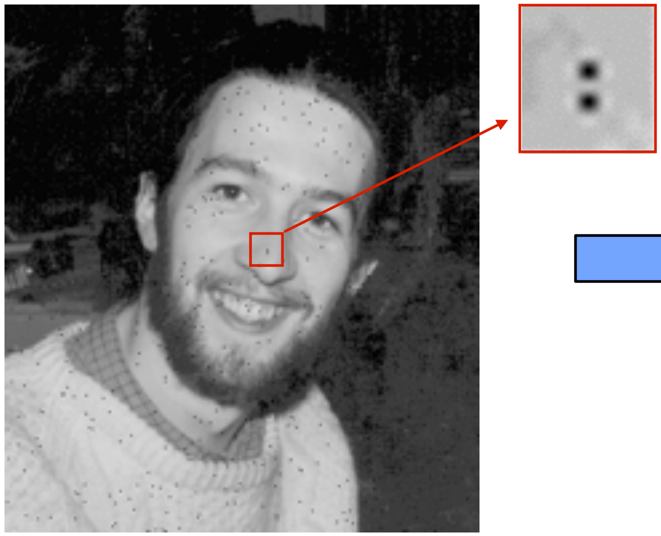

Opening with 3×3 square structuring element

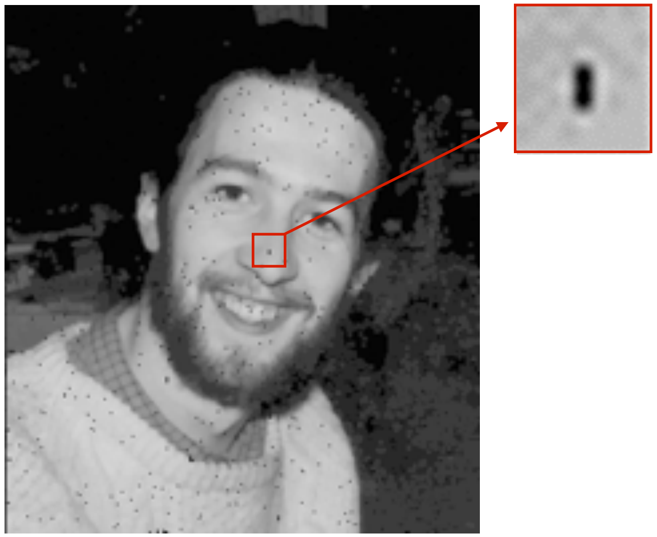

Erosion by 3×3 flat square structuring element

- **Opening**
  - The noise has been entirely removed with relatively little degradation of the underlying image.
- **Erosion**
  - The noise has been removed.
  - The rest of the image has been degraded significantly.

Opening (IX)

- Example: Grayscale opening (pepper noise)


Pepper noise image


Result of Opening

➔ **<u>No noise has been removed</u>.**
➔ **At some places where two nearby noise pixels have merged into one larger point, the noise level has even been increased.**

Closing (I)

Brief Description

- Closing is one of the two important operators from mathematical morphology.
- Closing is similar in some ways to dilation in that it tends to enlarge the boundaries of foreground (bright) regions in an image.

Common names : Closing

Closing (II)

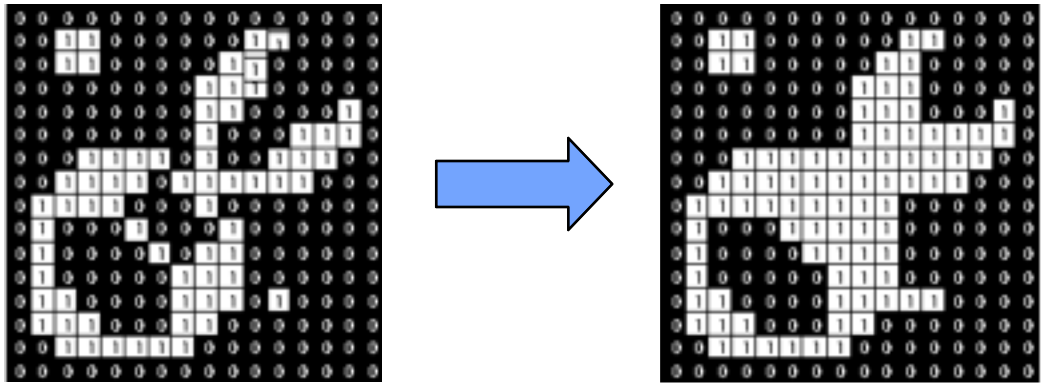
- **How It works**
  - Closing is defined as a dilation followed by an erosion.

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

  - Closing is the dual of opening.
    - Closing the foreground pixels with a particular structuring element is equivalent to opening the background pixels with the same element.

Closing (III)

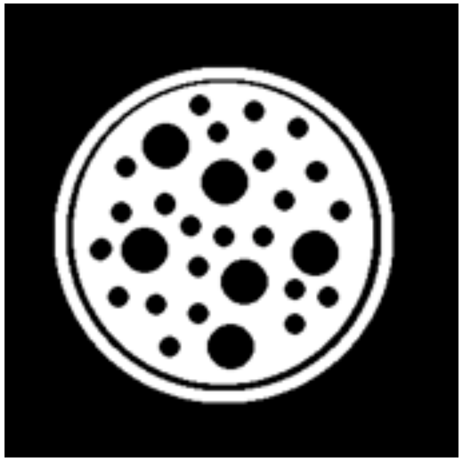
- **Guidelines for Use**
  - Idempotence
    - After the closing has been carried out, the background region will be such that the structuring element can be made to cover any point in the background without any part of it also covering a foreground point.
    - So further closings will have no effect.



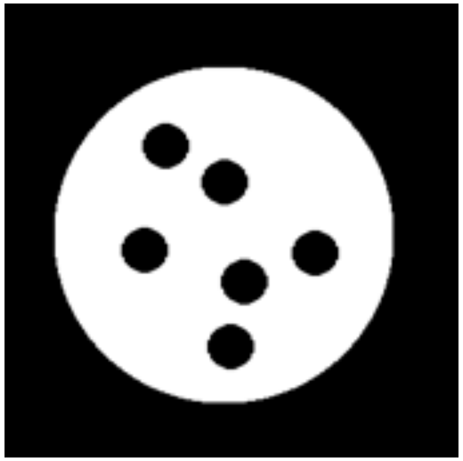
Effect of closing using a 3×3 square structuring element

Closing (IV)

- Example: Binary closing



Original image



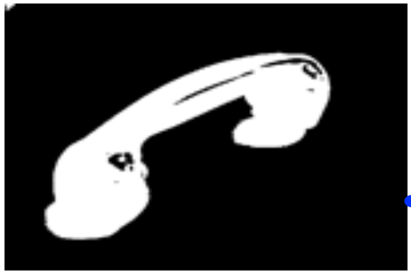
Result of a closing with a 22 pixel diameter disk

➔ If it is desired to remove the small holes while retaining the large holes, then we can simply perform a closing with a disk-shaped structuring element with a diameter larger than the smaller holes, but smaller than the larger holes.

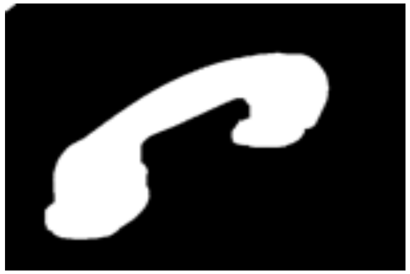
Closing (V)

- Example: Binary closing
  - Enhance binary images of objects obtained from thresholding.
  - We can see that skeleton (B) is less complex, and it better represents the shape of the object.

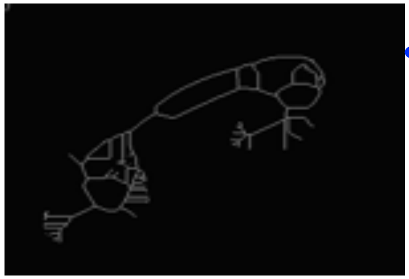
Original image



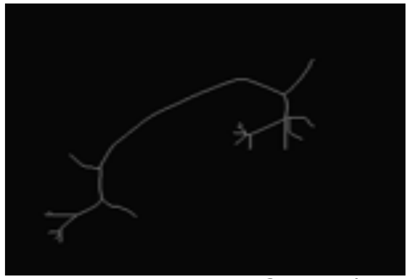
Thresholded image



Result of closing with a circular structuring element of size 20



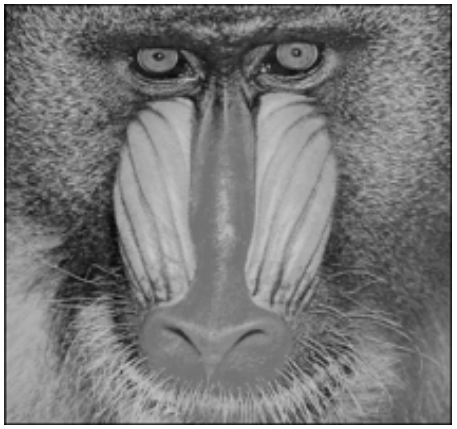
(A) The skeleton of the image which was only thresholded

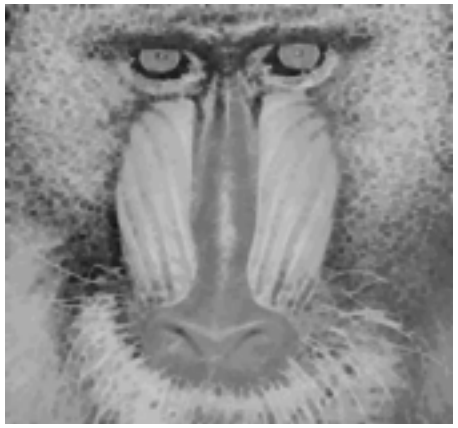


(B) The skeleton of the image produced by the closing operator

Closing (VI)

- Example: Grayscale closing
  - Gray-level closing can similarly be used to select and preserve particular intensity patterns while attenuating others.

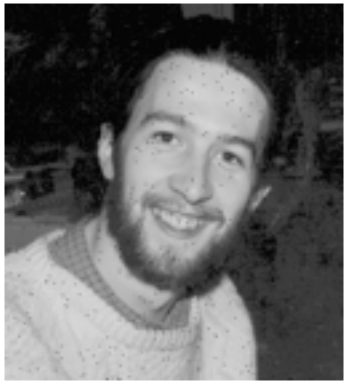

Original image

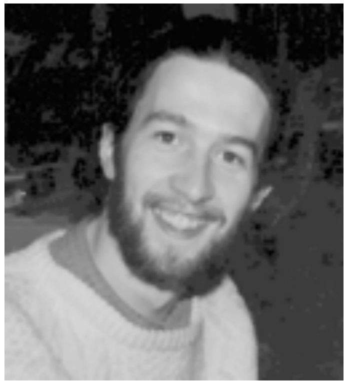

Opening with a flat 5 ×5 square structuring element

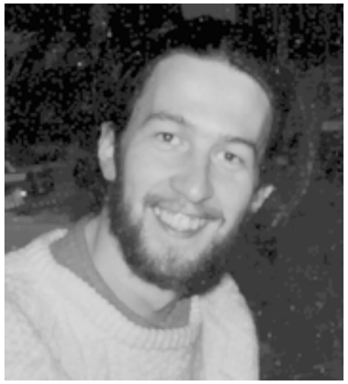
  - Notice how the dark specks in between the bright spots in the hair have been largely filled in to the same color as the bright spots.
  - But, the more uniformly colored nose area is largely the same intensity as before.

Closing (VII)

- Compare Closing with Dilation


Pepper noise image

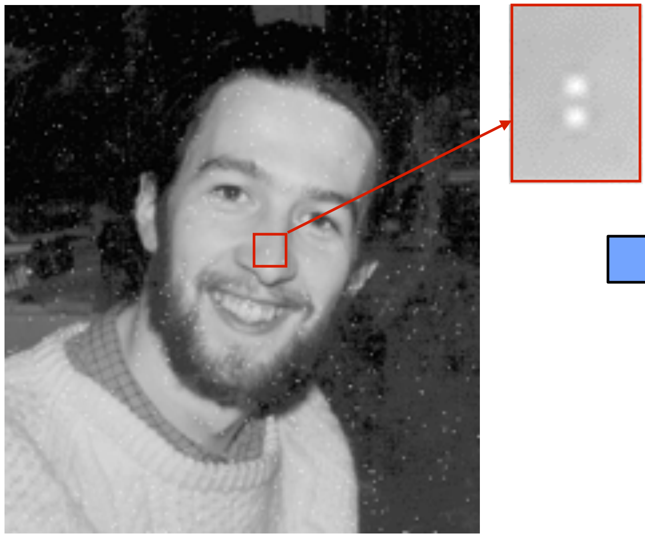

Closing with 3×3 square structuring element

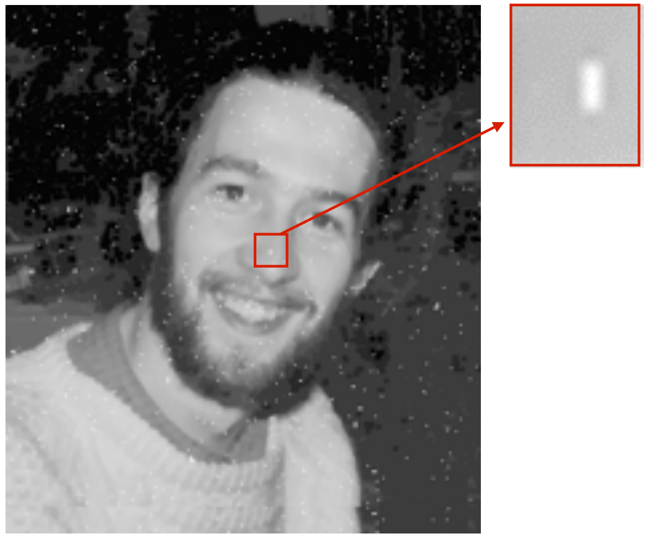

Dilation by 3×3 flat square structuring element

- Closing
  - **The noise has been completely removed with only a little degradation to the image.**
- Dilation
  - **Note that although the noise has been effectively removed, the image has been degraded significantly.**

Closing (VIII)

- Example: Grayscale closing (*salt noise*)


Salt noise image


Result of Closing

➔ No noise has been removed.

➔ The noise has even been increased at locations where two nearby noise pixels have merged together into one larger spot.

Opening and Closing

- **Translation invariance**

$$\boxed{A \circ (B)_x = A \circ B} \qquad \boxed{A \bullet (B)_x = A \bullet B}$$

- **Antiextensivity of opening**

$$\boxed{A \circ B \subseteq A}$$

- **Extensivity of closing**

$$\boxed{A \subseteq A \bullet B}$$

- **Duality**

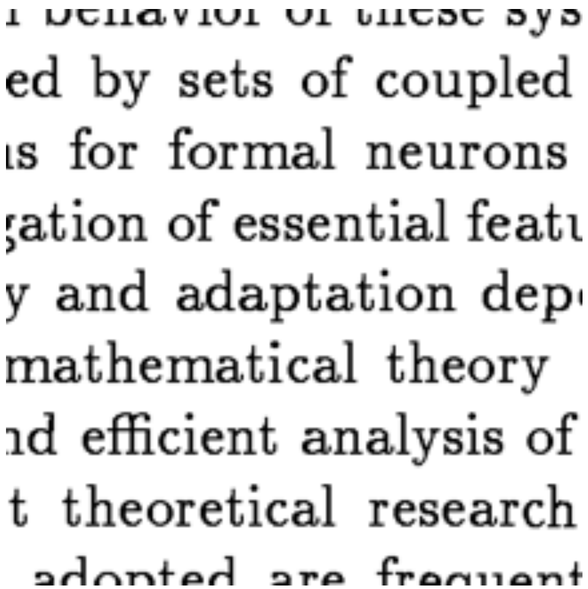
$$\boxed{(A \bullet B)^c = A^c \circ B^s}$$

Morphological Filtering

- **Main idea**
  - Examine the **geometrical** structure of an image by matching it with small patterns called structuring elements at various locations
  - By varying the **size** and **shape** of the matching patterns, we can extract useful information about the shape of the different parts of the image and their interrelations.

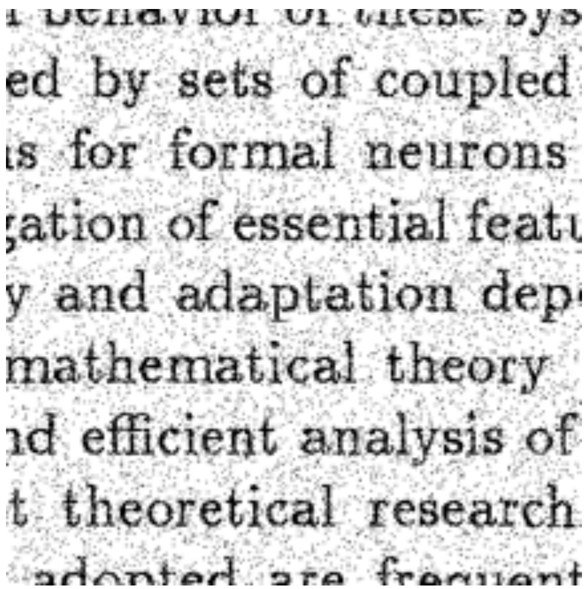
Morphological filtering

- **Noisy image will break down OCR systems**



ı behavior of these sys
ed by sets of coupled
ıs for formal neurons
gation of essential featı
y and adaptation dep
mathematical theory
ıd efficient analysis of
t theoretical research
adopted are frequent

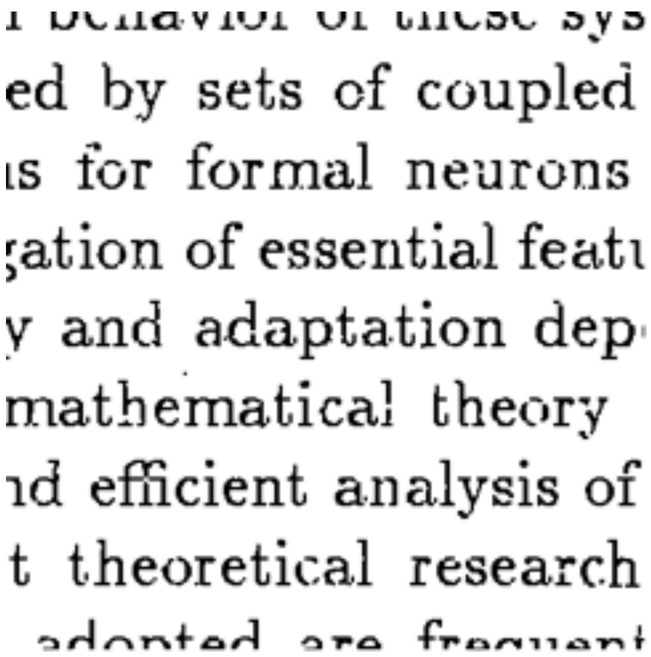
Clean image



ı behavior of these sys
ed by sets of coupled
ıs for formal neurons
gation of essential featı
y and adaptation dep
mathematical theory
ıd efficient analysis of
t theoretical research
adopted are frequent

Noisy image

Morphological filtering


ed by sets of coupled
ıs for formal neurons
;ation of essential featı
y and adaptation dep
mathematical theory
ıd efficient analysis of
t theoretical research

By applying MF, we increase the OCR accuracy!

Restored image

Summary

- **Mathematical morphology is an approach for processing digital image based on its shape**
- **The language of morphology is set theory**
- **The basic morphological operations are erosion and dilation**
- **Morphological filtering can be developed to extract useful shape information**

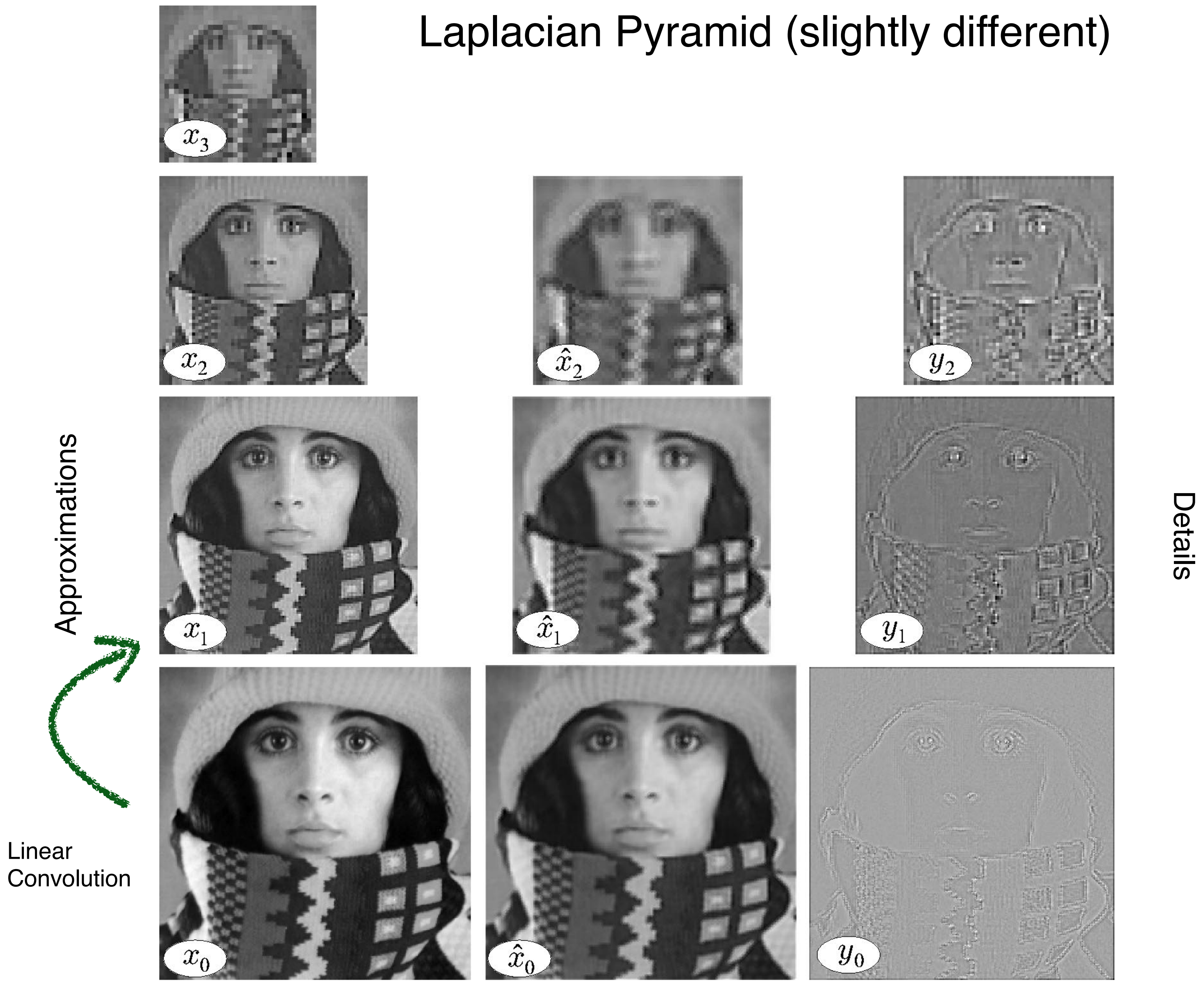
Advanced Morphomat Methods

Morphological Pyramid

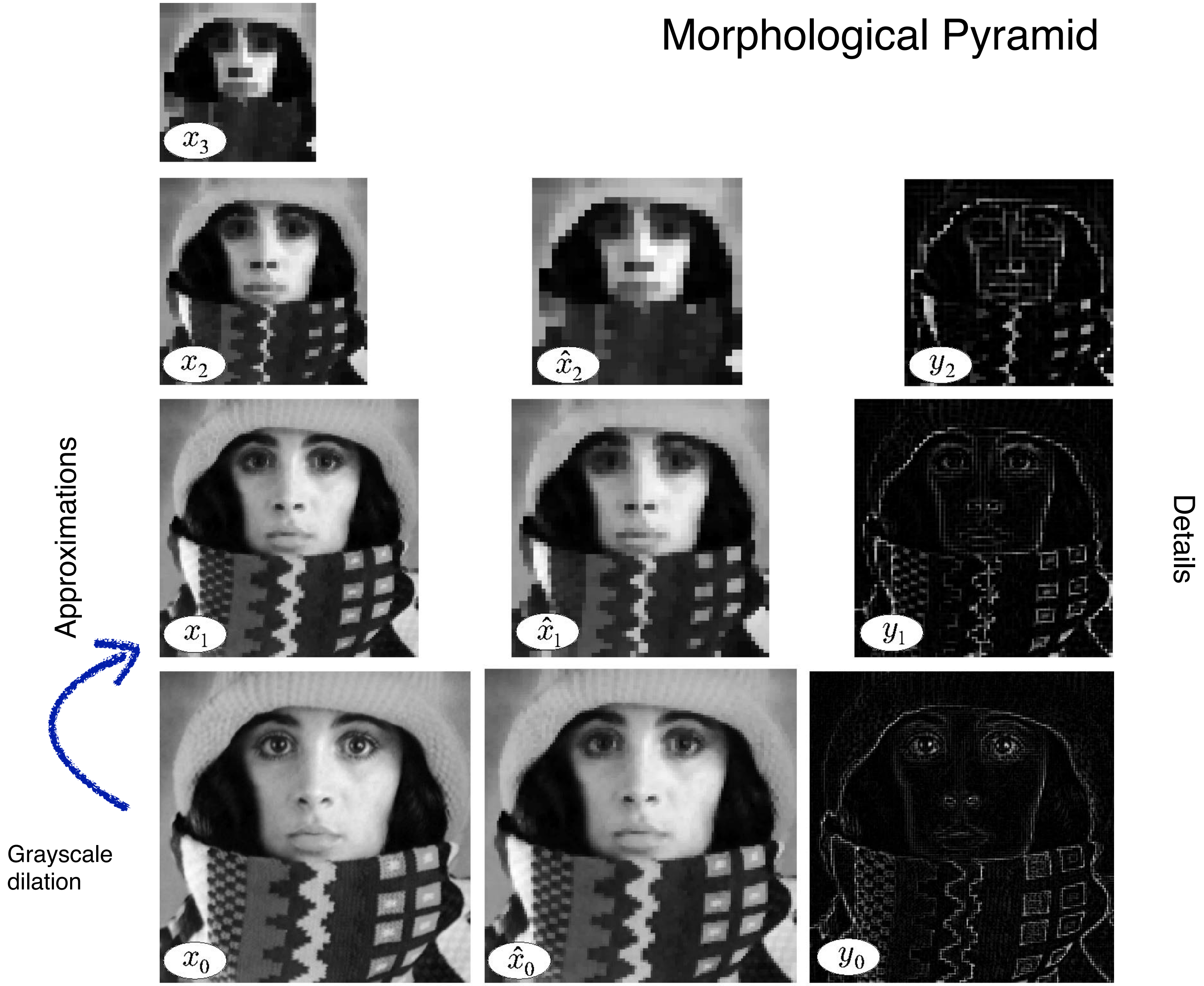
- **Replace convolution by morphological dilation in Gaussian/Laplacian pyramid of Burt and Adelson**
- **Interest: detect different features, different properties (no maxima creation in the scale-space)**

Goutsias, J., & Heijmans, H. J. (2000).
Nonlinear multiresolution signal decomposition schemes. I. Morphological pyramids.
Image Processing, IEEE Transactions on, 9(11), 1862-1876.

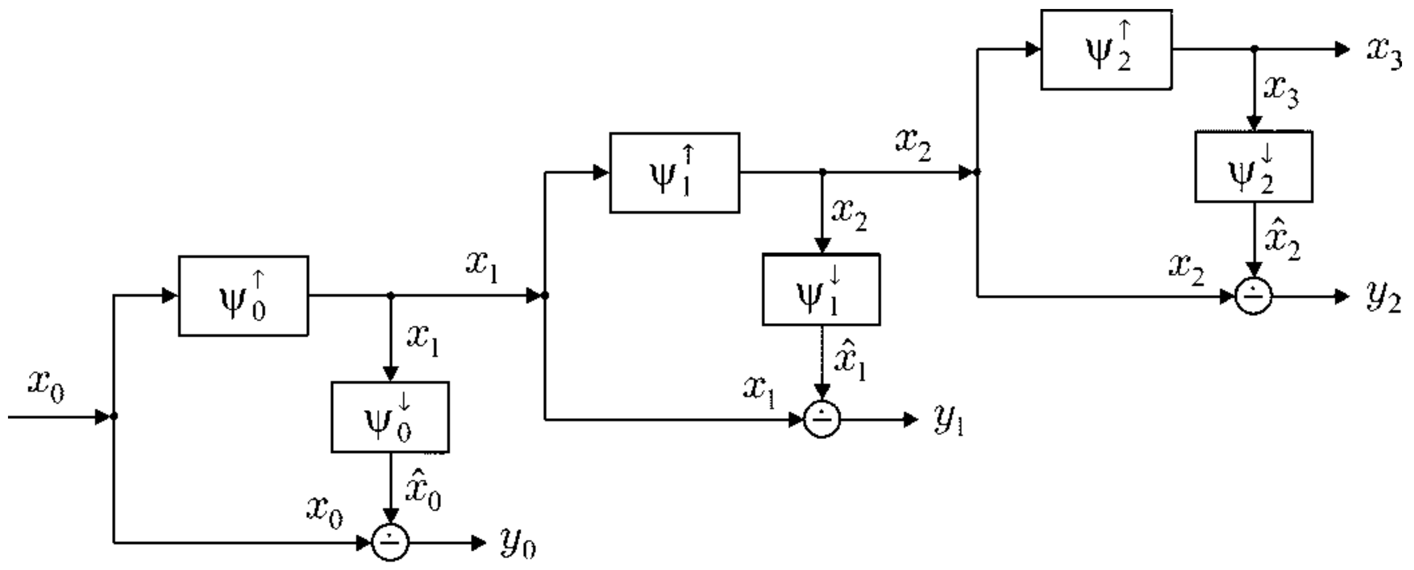
Laplacian Pyramid (slightly different)



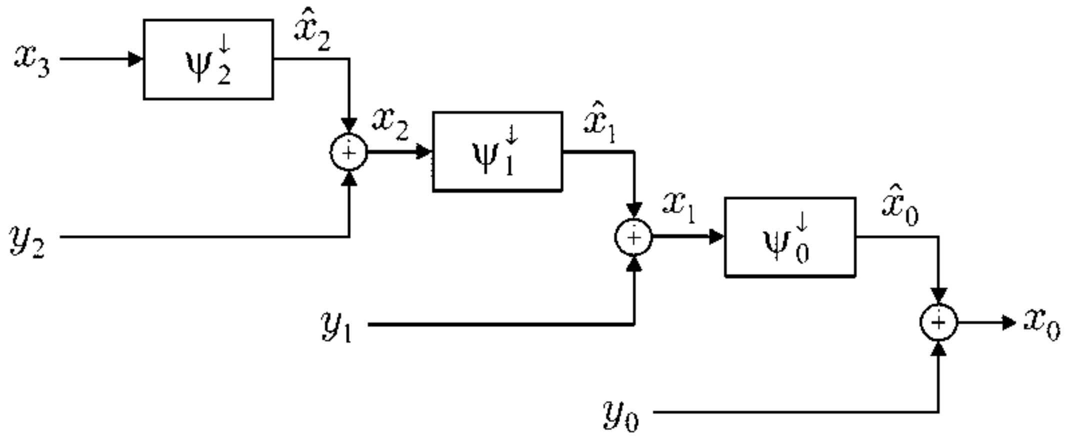
Morphological Pyramid



Forward and Inverse Transform



(a)



Hit-and-Miss Transform (I)

- **Brief Description**
  - General binary morphological operation that can be used to look for particular patterns in an image.
  - Basic operation of binary morphology
    - Almost all the other binary morphological operators can be derived from Hit-and-Miss Transform.

Common names: Hit-and-miss Transform, Hit-or-miss Transform

Hit-and-Miss Transform (II)

- **How It Works**
  - The structuring element used in the hit-and-miss can contain both foreground and background pixels.

	1	
0	1	1
0	0	

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

Hit-and-Miss Transform (III)

- Effect of the hit-and-miss based right angle convex corner detector

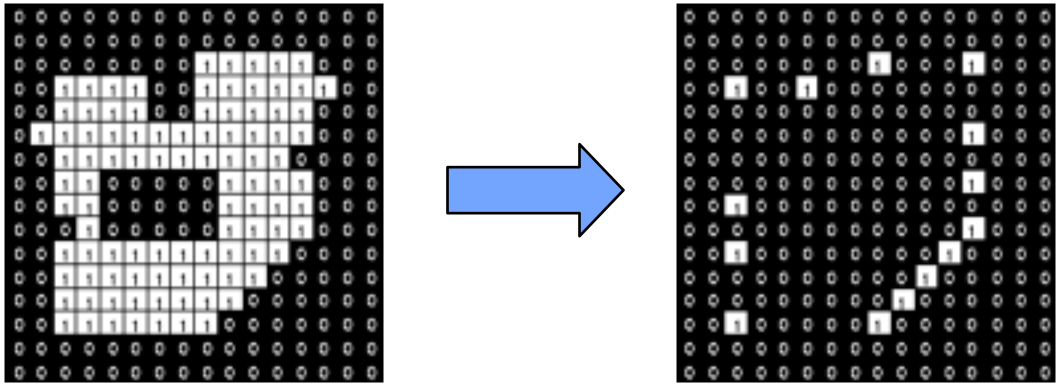
	1	
0	1	1
0	0	

	1	
1	1	0
	0	0

	0	0
1	1	0
	1	

0	0	
0	1	1
	1	

Four structuring elements used for corner finding in binary images



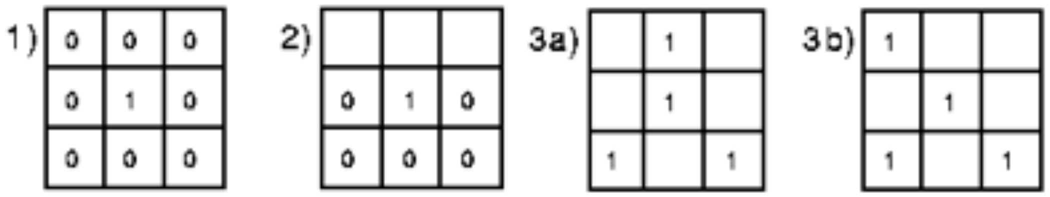
➔ After obtaining the locations of corners in each orientation, we can then simply OR all these images together to get the final result showing the locations of all right angle convex corners in any orientation.

Hit-and-Miss Transform (IV)

- **Guidelines for Use**
  - The hit-and-miss transform is used to look for occurrences of particular binary patterns.
  - It can be used to look for several patterns.
    - Simply by running successive transforms using different structuring elements, and then ORing the results together.
  - The operations of erosion, dilation, opening, closing, thinning and thickening can all be derived from the hit-and-miss transform in conjunction with simple set operations.

Hit-and-Miss Transform (V)

- Some structuring elements that can be used for locating various binary features



1)

0	0	0
0	1	0
0	0	0

2)

0	1	0
0	0	0

3a)

	1	
	1	
1		1

3b)

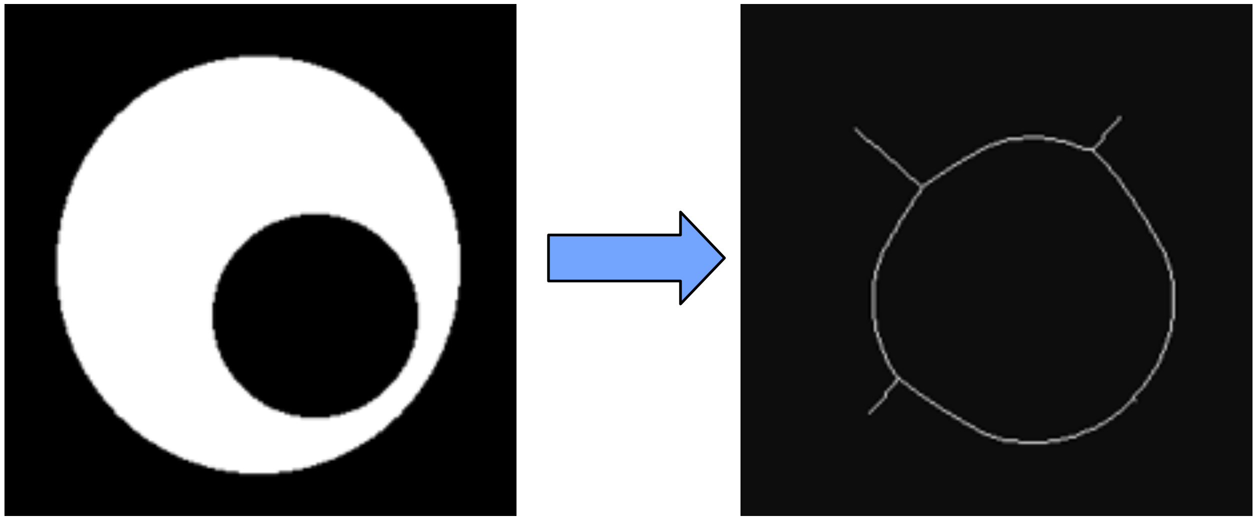
1		
	1	
1		1

Some applications of the hit-and-miss transform

- 1) is used to locate isolated points in a binary image.
- 2) is used to locate the end points on a binary skeleton.
  - **Note that this structuring element must be used in all its orientations, and thus the four hit-and-miss passes are required.**
- 3a) and 3b) are used to locate the triple points on a skeleton.
  - **Both structuring elements must be run in all orientations so eight hit-and-miss passes are required.**

Hit-and-Miss Transform (VI)

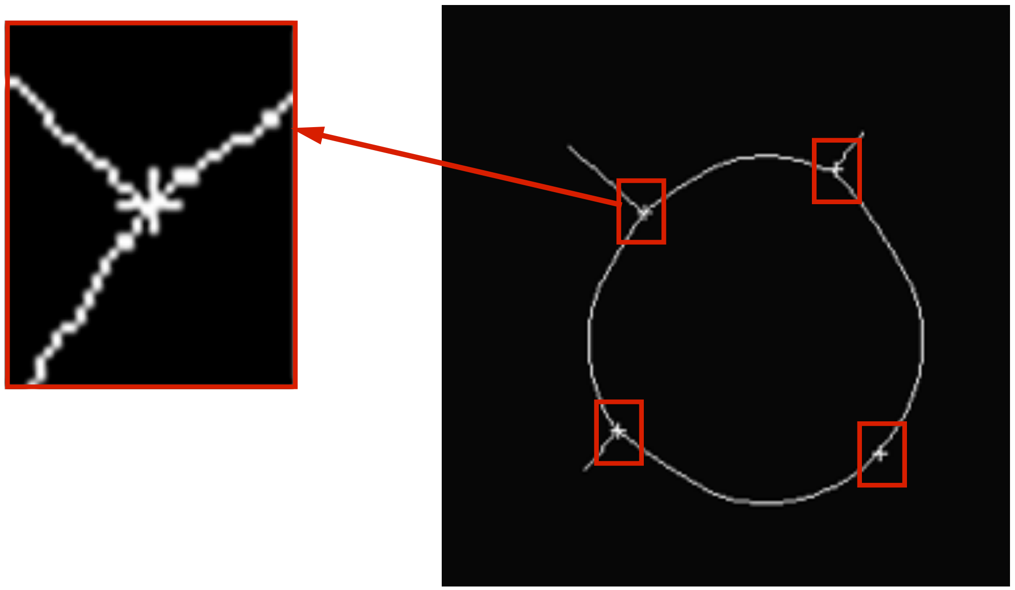
- Example: Hit-and-miss transform
  - Generate skeleton image



Original image **Skeleton image**

Hit-and-Miss Transform (VII)

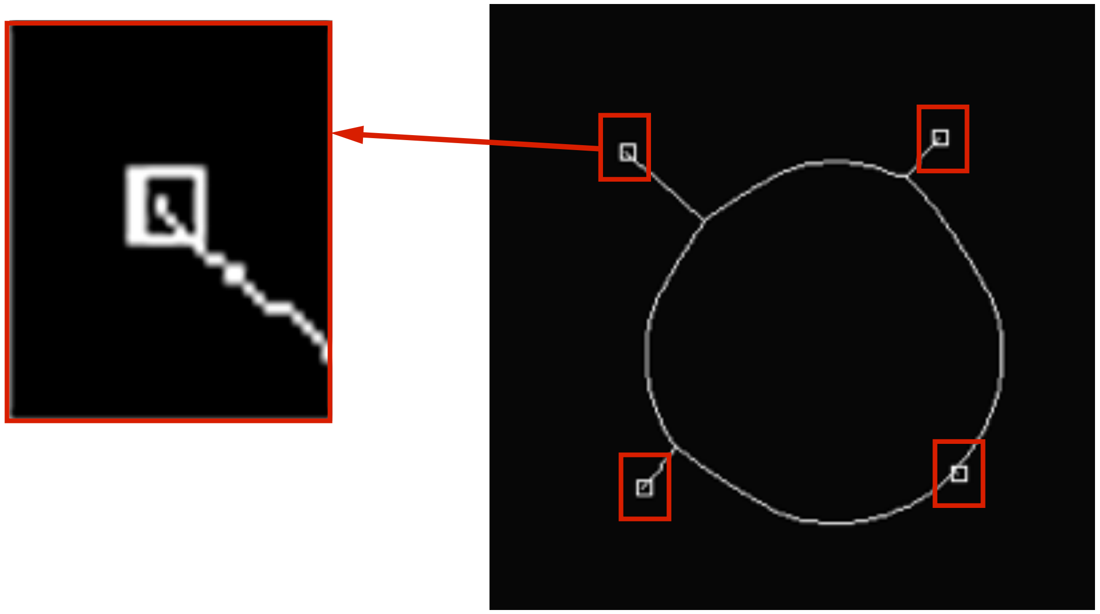
- The triple points(points where three lines meet) of the skeleton



  - **The hit-and-miss transform outputs single foreground pixels at each triple point by structuring elements 3a) and 3b).**
  - **This image was dilated once using a cross-shaped structuring element in order to mark these isolated points clearly, and this was then ORed with the original skeleton.**

Hit-and-Miss Transform (VIII)

- The end points of the skeleton



- The hit-and-miss transform outputs single foreground pixels at each end point by structuring element 2).
- This image was dilated once using a square-shaped structuring element, and this was then ORed with the original skeleton.

Thinning (I)

- **Brief Description**
  - Remove selected foreground pixels from binary images, somewhat like erosion or opening.
  - Tidy up the output of edge detectors by reducing all lines to single pixel thickness.
  - Thinning is normally applied only to binary images.

Common names: Thinning

Thinning (II)

How It Works

$$\boxed{Thin(I, J) = I \; - \; \text{hit-and-miss}(I, J)}$$

I: image, ***J***: structuring element $\qquad X - Y = X \cap Y^c$

- Thinning is the dual of thickening (see later).
  - Thickening the foreground is equivalent to thinning the background.
- The operator is normally applied repeatedly until it causes no further changes to the image.
- Alternatively, in some applications, e.g. pruning, the operations may only be applied for a limited number of iterations.

Thinning (III)

- Operation
  - The thinning operation is calculated by translating the origin of the structuring element to each possible pixel position in the image, and at each such position comparing it with the underlying image pixels.
  - If the foreground and background pixels in the structuring element exactly match foreground and background pixels in the image, then the image pixel underneath the origin of the structuring element is set to background (zero).
  - Otherwise, it is left unchanged.
- Note that the structuring element must always have a one or a blank at its origin if it is to have any effect.

Thinning (IV)

- **Guidelines for Use**
  - One of the most common uses of thinning is to reduce the thresholded output of an edge detector such as the Sobel operator, to lines of a single pixel thickness.
    - While preserving the full length of those lines (i.e. pixels at the extreme ends of the lines should not be affected.)
  - A simple algorithm for doing thinning
    - Consider all pixels on the boundaries of foreground regions.
    - Delete any such point that has more than one foreground neighbor, as long as doing so does not locally disconnect the region containing that pixel.
    - Iterate until convergence.

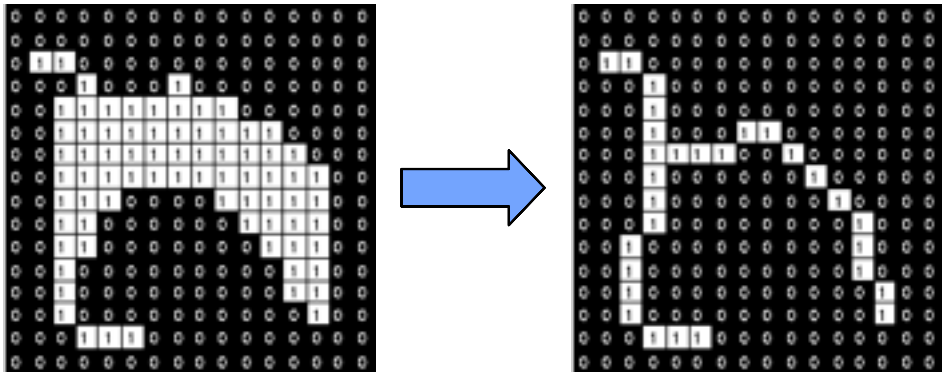
Thinning (V)

- Skeletonization by Morphological Thinning

0	0	0
	1	
1	1	1

	0	0
1	1	0
	1	

Structuring elements for skeletonization by morphological thinning



- At each iteration, the image is first thinned by the left hand structuring element, and then by the right hand one, and then with the remaining six 90°rotations of the two elements.
- The process is repeated in cyclic fashion until none of the thinnings produces any further change.

Thinning (VI)

- Some other common structuring elements

1)

1	1	1
1	1	1
1	1	1

2)

	1	
1	1	1
	1	

3a)

0	0	0
0	1	0
0		

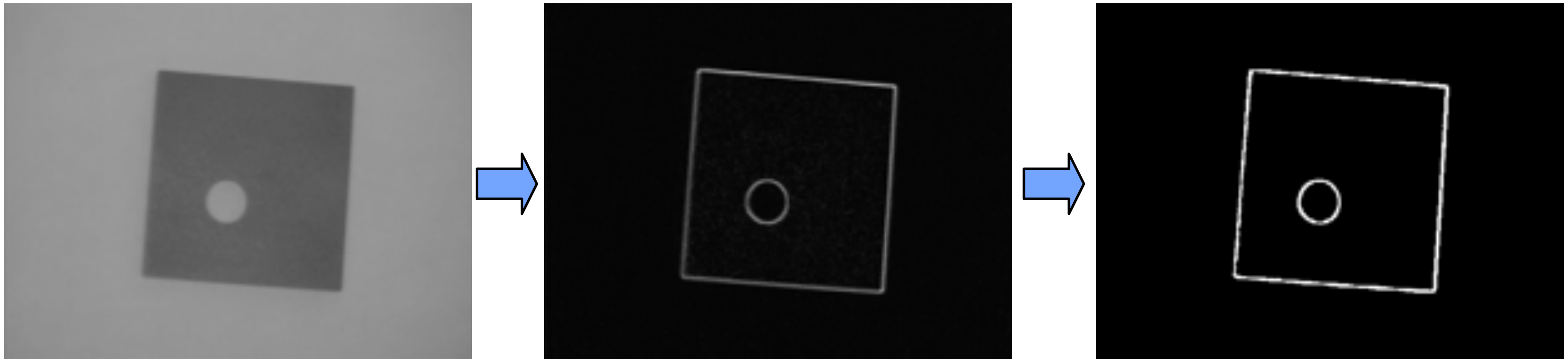
3b)

0	0	0
0	1	0
		0

  - 1) Simply finds the boundary of a binary object, the detected boundary is 4-connected.
  - 2) Does the same thing but produces an 8-connected boundary.
  - 3a) and 3b) are used for pruning
    - **Note that skeletons produced by this method often contain undesirable short spurs produced by small irregularities in the boundary of the original object.**
    - **These spurs can be removed by a process called pruning, which is in fact just another sort of thinning.**

Thinning (VII)

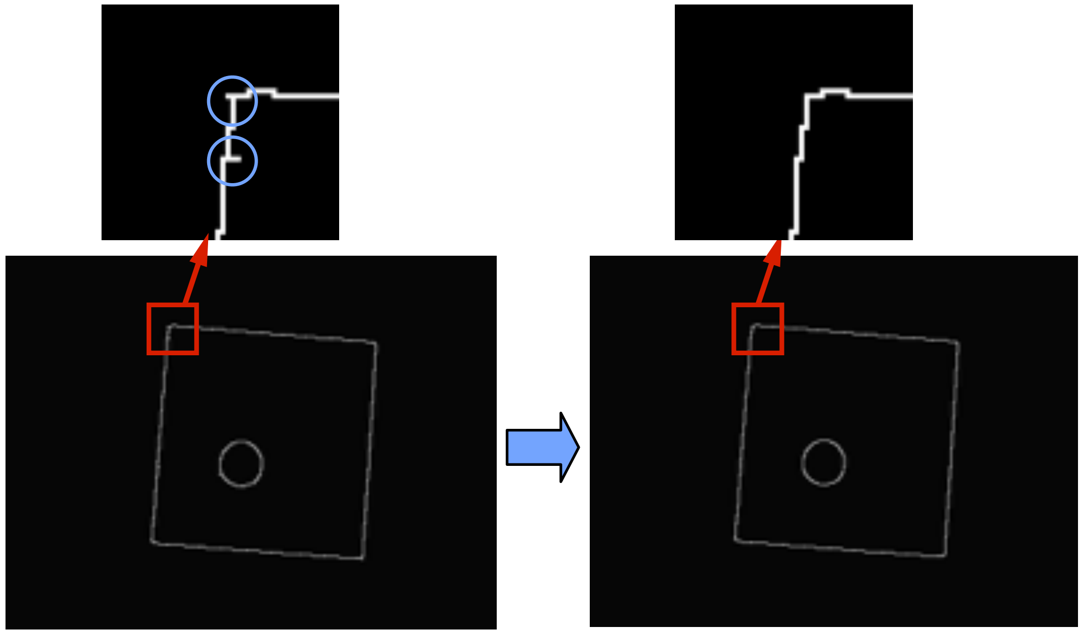
- Example: Thinning & pruning



Original image **Result of applying Sobel operator** **Thresholded image**

- First threshold the image at a gray-level value of 60 producing.
- Note that the detected boundaries of the object are several pixels thick.

Thinning (VIII)

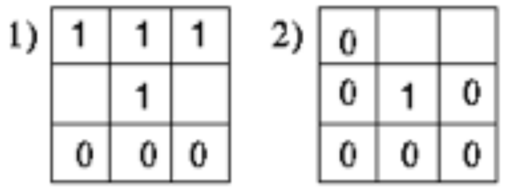


Result of thinning **Result of pruning for the five iterations**

- The detected lines have all been reduced to a single pixel width.
- Note however that there are still one or two ‘spurs’ present, which can be removed using pruning.

Thinning (IX)

- Example: Thinning (*character recognition*)



1)

1	1	1
	1	
0	0	0

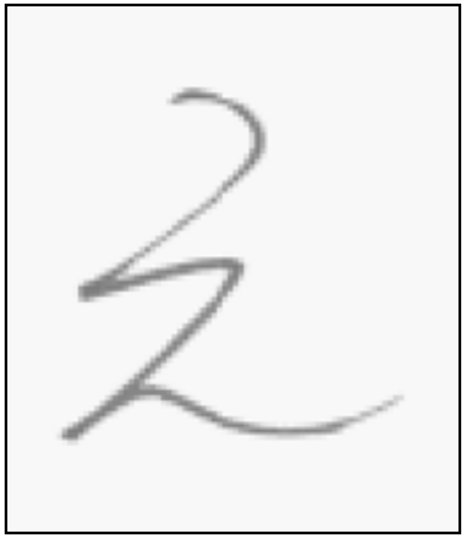
2)

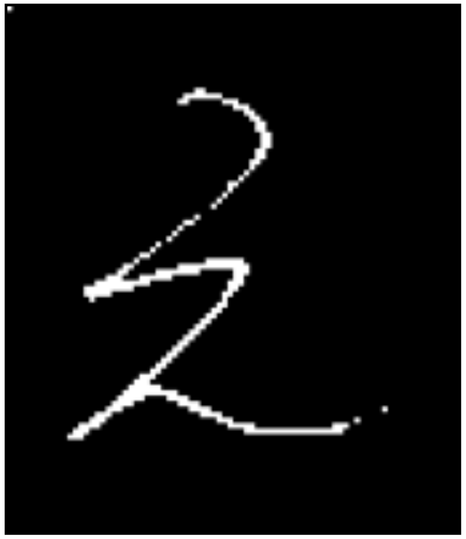
0		
0	1	0
0	0	0

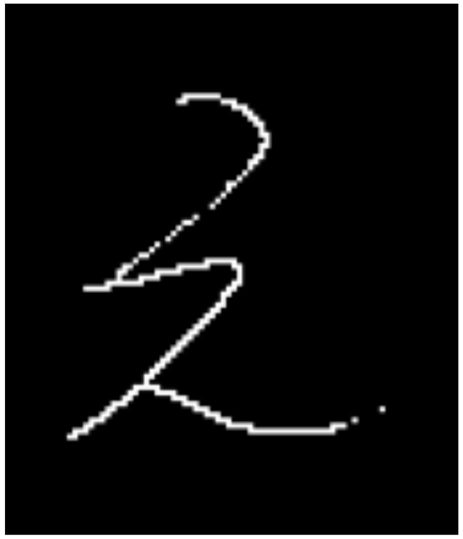
Structuring elements used in the character recognition example

  - 1) shows the structuring element used in combination with thinning to obtain the skeleton.
  - 2) was used in combination with thinning to prune the skeleton and with the hit-and-miss operator to find the end points of the skeleton.
  - Each structuring element was used in each of its 45°rotations.

Thinning (X)


Original image

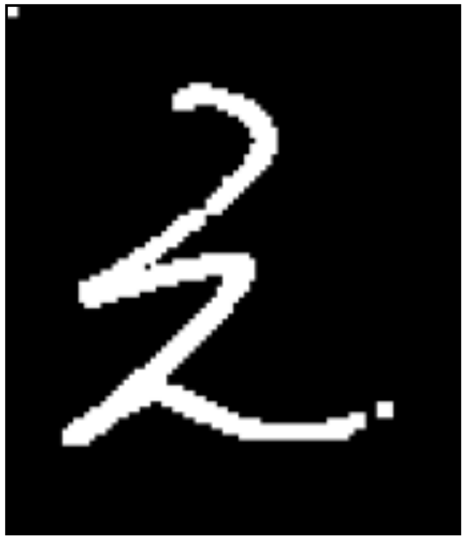

Thresholded image


Thinning result

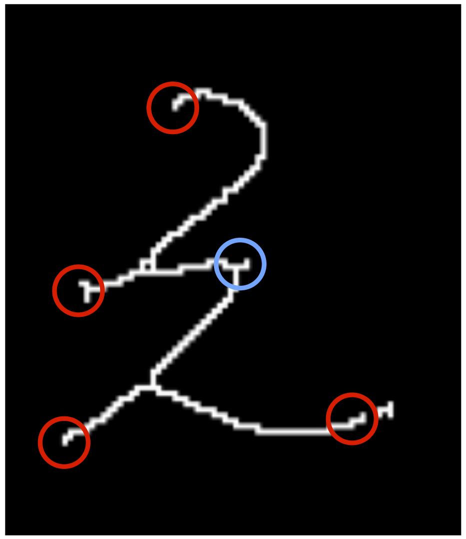
- A simple way to obtain the skeleton of the character is to thin the image until convergence.
- The line is broken at some locations, which might cause problems during the recognition process.

Thinning (XI)

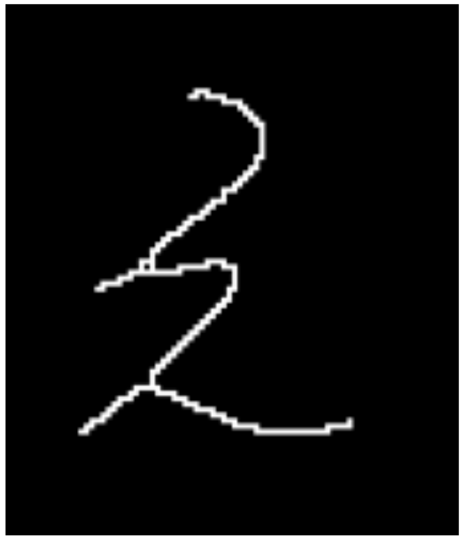
- Example: Improve broken line case (*dilating twice*)



Dilating thresholded image twice with a 3×3 square structuring element



Result of thinning



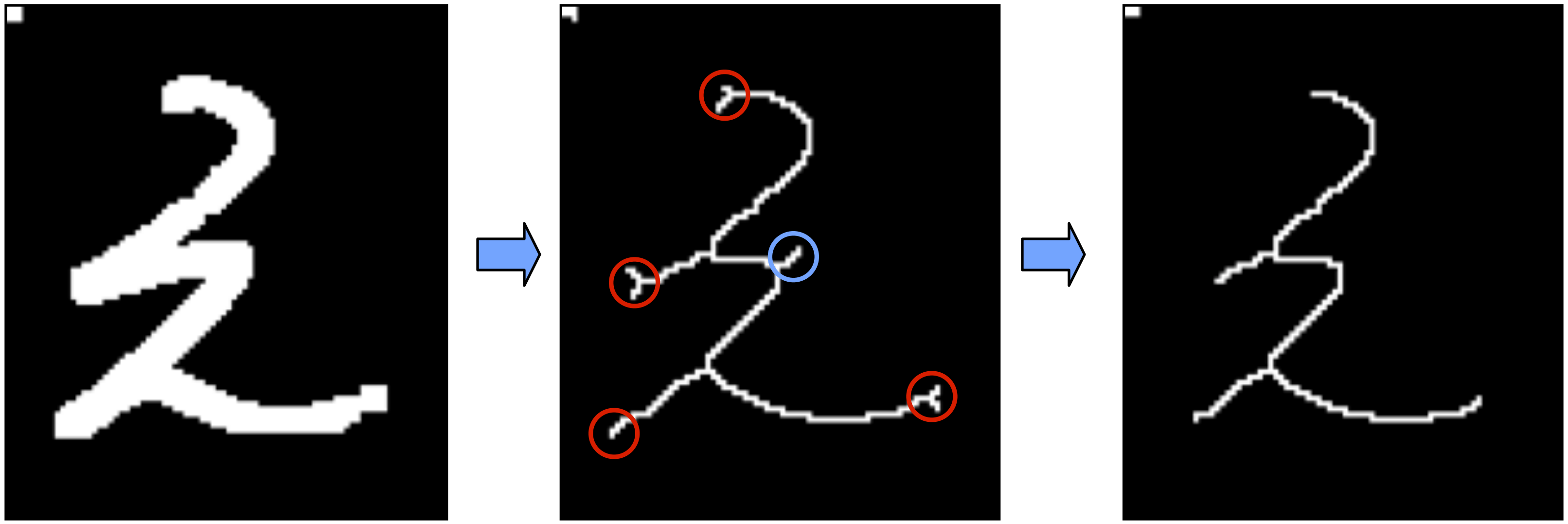
Result of pruning using two iterations for each orientation of the structuring element

○ Blue ring: Spur to be deleted

○ Red ring: Spur not to be deleted

Thinning (XII)

- Example: Improve broken line case (*dilating three times*)



Dilating thresholded image three times with a 3×3 square structuring element

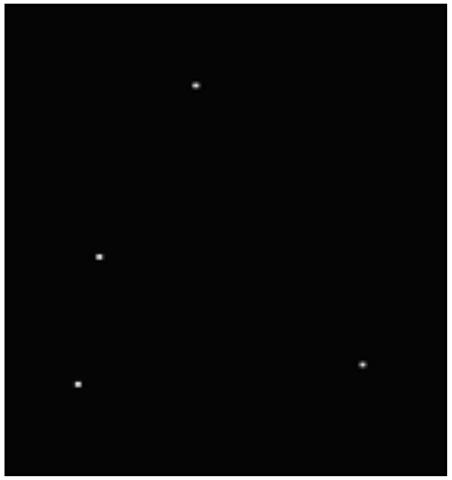
Result of thinning

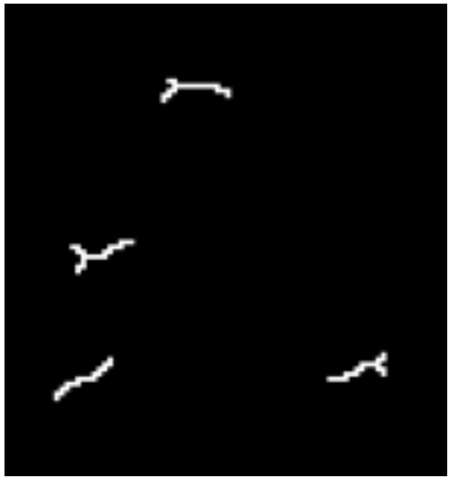
Result of pruning using four iterations for each orientation of the structuring element

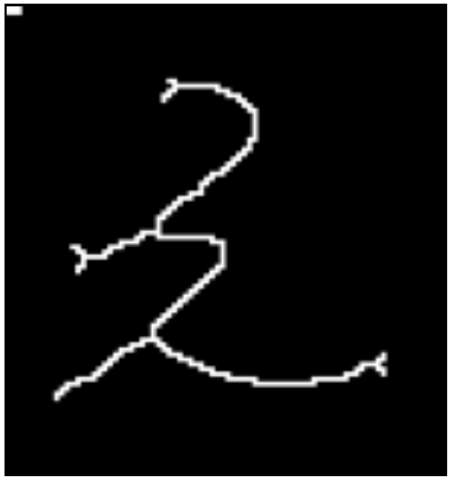
➔ The spurs have now disappeared, however, the pruning has also suppressed pixels at the end of correct lines (red ring ◯).

Thinning (XIII)

- Example: Restore pixels at the end of correct lines


Result of applying a hit-and-miss operator

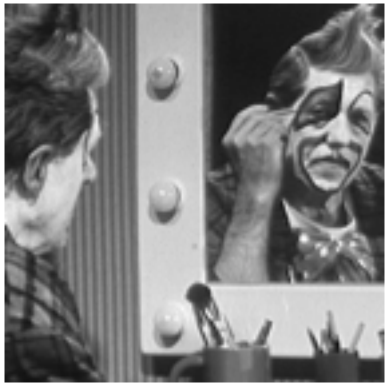

Result of repeating Conditional Dilation 5 times

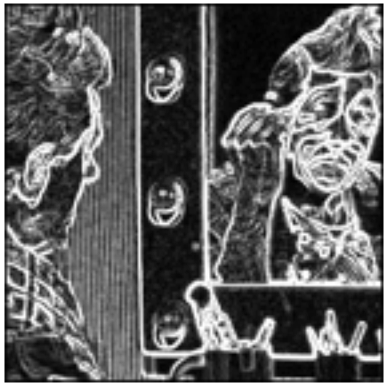

Result of ORing left image with the pruning output

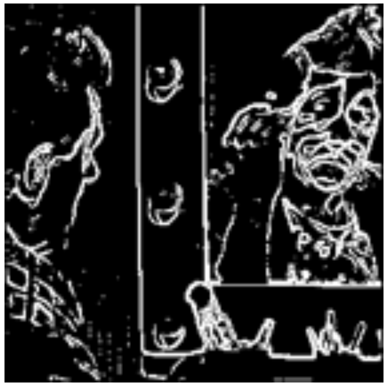
- Applying a hit-and-miss operator to know where to start the dilation.
- Dilating left image using a 3×3 structuring element, ANDing it with the thinned, but not pruned image.
- Final step is to OR this image with the pruning output.

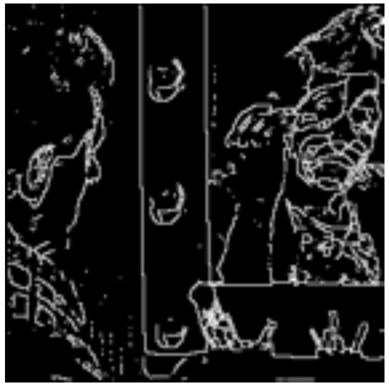
Thinning (XIV)

- Example: Thinning (*more complicated image*)


Original image


Result of applying the Sobel operator


Thresholded image


Result of Skeletonization by thinning

- The result is a lot less clear than before.
- Thinning more complicated images often produces less spectacular results.

Thickening (I)

- **Brief Description**
  - Growing selected regions of foreground pixels in binary images, somewhat like dilation or closing.
  - Several applications
    - Determining the approximate convex hull of a shape
    - Determining the skeleton by zone of influence
  - Normally only applied to the binary images and produces another binary image as output.

Common names: Thickening

Thickening (II)

- **How It Works**

$$\boxed{Thicken(I,J) = I \cup \text{hit-and-miss}(I,J)}$$

I: image, ***J***: structuring element

- The thickened image consists of the original image plus any additional foreground pixels switched on by the hit-and-miss transform.

- Thickening is the dual of thinning.
  - Thinning the foreground is equivalent to thickening the background.
- The operator is normally applied repeatedly until it causes no further changes to the image.
  - c.f. The operations may only be applied for a limited number of iterations.

Thickening (III)

- Operation
  - The thickening operation is calculated by translating the origin of the structuring element to each possible pixel position in the image, and at each position comparing it with the underlying image pixels.
  - If the foreground and background pixels in the structuring element exactly match foreground and background pixels in the image, then the image pixel underneath the origin of the structuring element is set to foreground (one).
  - Otherwise, it is left unchanged.

Thickening (IV)

- **Guidelines for Use**
  - Two applications of thickening
    - Determining the convex hull
    - Finding the skeleton by zone of influence or SKIZ
  - The convex hull of a binary shape
    - Visualized quite easily by imagining stretching an elastic band around the shape.
    - The elastic band will follow the convex contours of the shape, but will ‘bridge’ the concave contours.
    - The resulting shape will have no concavities and contains the original shape.

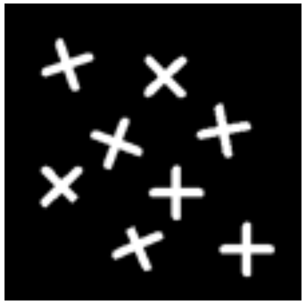
Thickening (V)

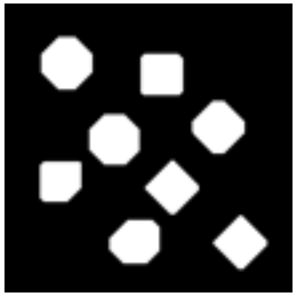
- Example: *convex hull*

1	1	
1	0	
1		0

	1	1
	0	1
0		1

  - Each element should be used in turn, and then in each of their 90°rotations, giving 8 effective structuring elements in total.
  - This process took a considerable amount of time --- over 100 thickening passes with each of the eight structuring elements


Cross-shaped binary objects

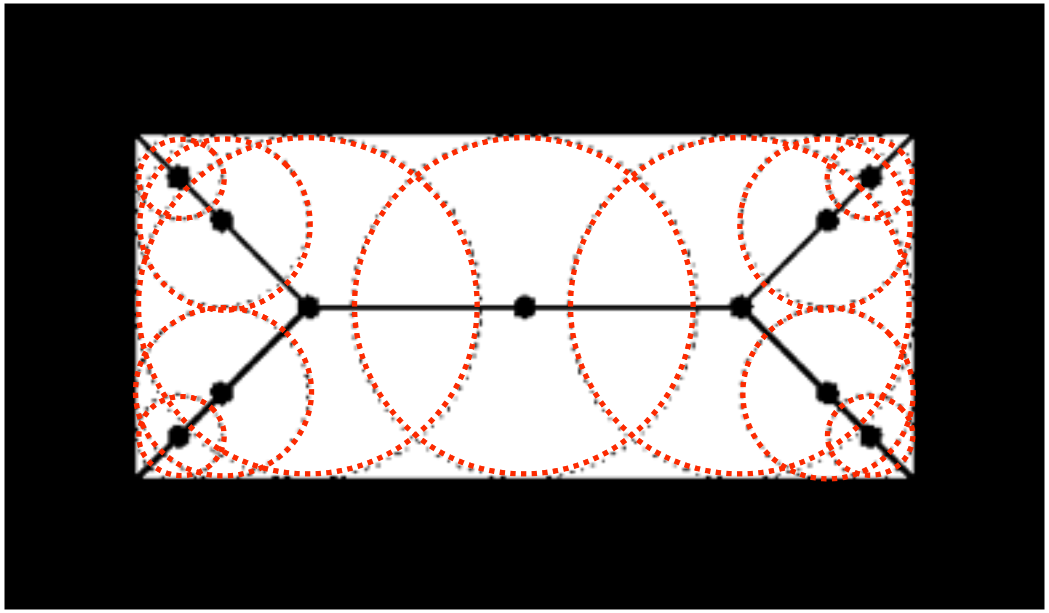

Result of applying the 45° convex hull algorithm

Skeletonization/ Medial Axis Transform (I)

- **Brief Description**
  - Skeletonization is a process for reducing foreground regions in a binary image to a skeletal remnant.
    - The foreground regions in the input binary image are made of some uniform slow-burning material.
    - Light fires simultaneously at all points along the boundary of this region and watch the fire move into the interior.
    - At points where the fire traveling from two different boundaries meets itself, the fire will extinguish itself and the points at which this happens form the so called "quench line".

Skeletonization/ Medial Axis Transform (II)

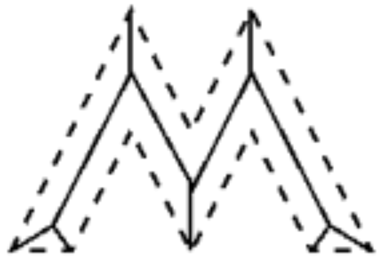
- The skeleton is as the loci of centers of bi-tangent circles that fit entirely within the foreground region being considered.



Skeleton of a rectangle defined in terms of bi-tangent circles

Skeletonization/ Medial Axis Transform (III)

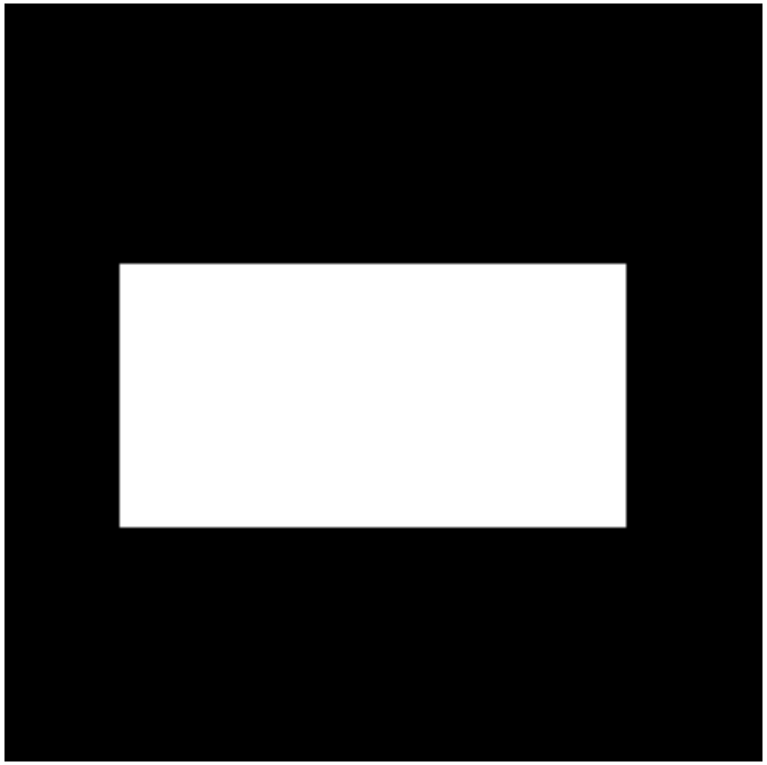
- The terms Medial Axis Transform (MAT) and skeletonization are often used interchangeably, but we will distinguish between them slightly.
  - The skeleton is simply a binary image showing the simple skeleton.
  - The MAT on the other hand is a gray-level image where each point on the skeleton has an intensity which represents its distance to a boundary in the original object.

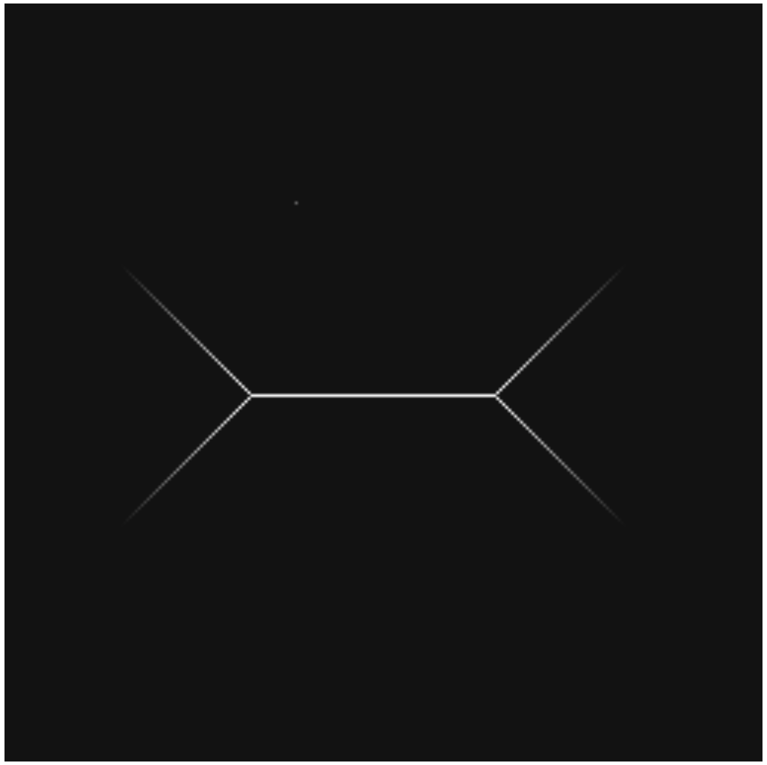


Common names: Skeletonization, Medial axis transform

Skeletonization/ Medial Axis Transform (IV)

- **How It Works**
  - The skeleton/MAT can be produced in two main ways.
    - The first is to use some kind of morphological thinning that successively erodes away pixels from the boundary until no more thinning is possible.
    - The alternative method is to first calculate the distance transform of the image.
      - **The MAT is often described as being the "locus of local maxima" on the distance transform.**
      - **The MAT can be imagined as the *ridges* on the 3-D surface with the third dimension representing the gray-value.**


Original image

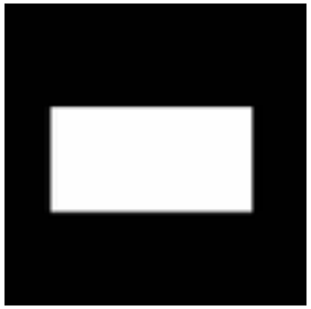

Medial axis transform

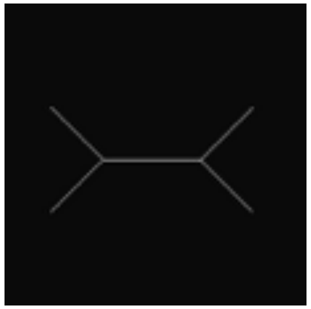
Skeletonization/ Medial Axis Transform (V)

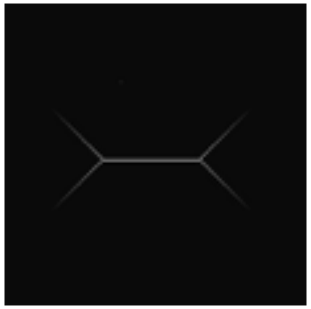
- **Guide Lines for Use**
  - The skeleton is useful because it provides a simple and compact representation of a shape that preserves many of the topological and size characteristics of the original shape.
    - Get a rough idea of the length of a shape.
    - Distinguish many qualitatively different shapes from one another on the basis of how many ‘triple points’.
      - **Triple points: points where at least three branches of the skeleton meet**
  - The MAT (not the pure skeleton) has the property that it can be used to exactly reconstruct the original shape if necessary.
  - Note that some implementations of skeletonization algorithms produce skeletons that are not guaranteed to be continuous.

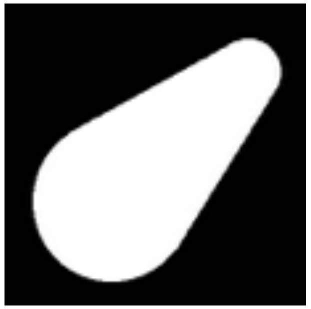
Skeletonization/ Medial Axis Transform (VI)

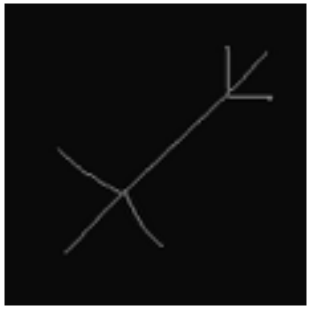
- Examples: Skeleton/MAT
  - Note that the MATs have been contrast-stretched in order to make them more visible.

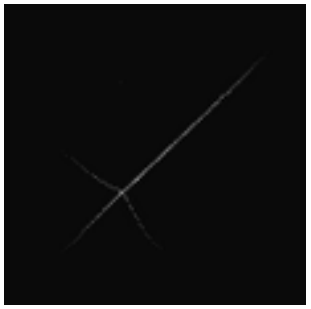

Original Image


Skeleton

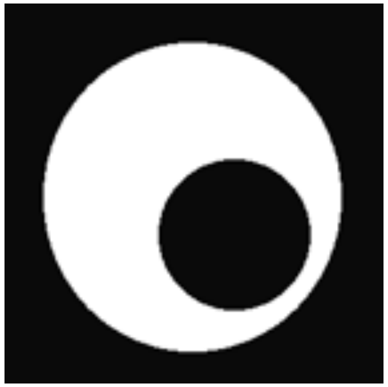

Medial Axis Transform

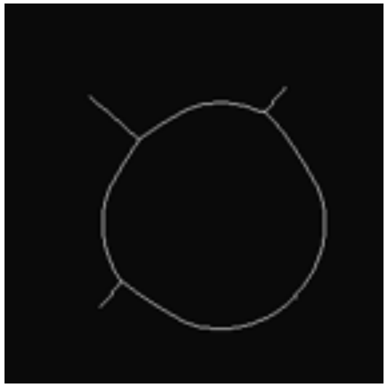

Original Image

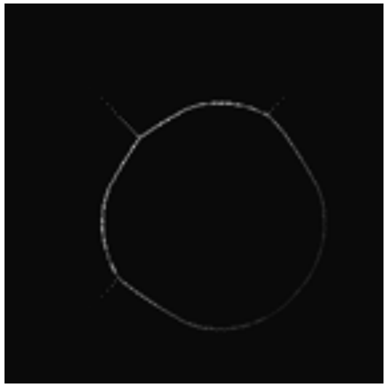

Skeleton

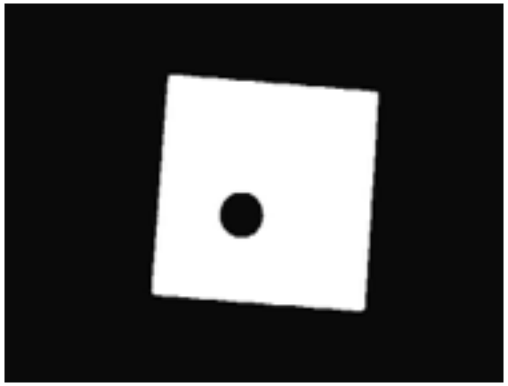

Medial Axis Transform

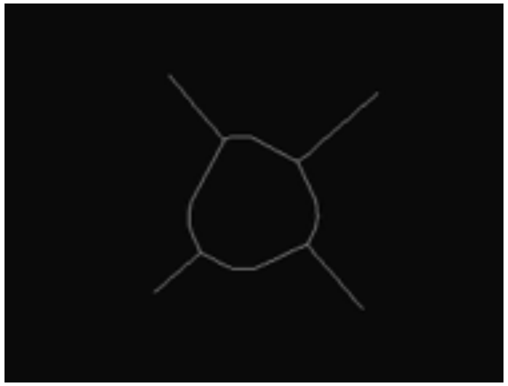
Skeletonization/ Medial Axis Transform (VII)

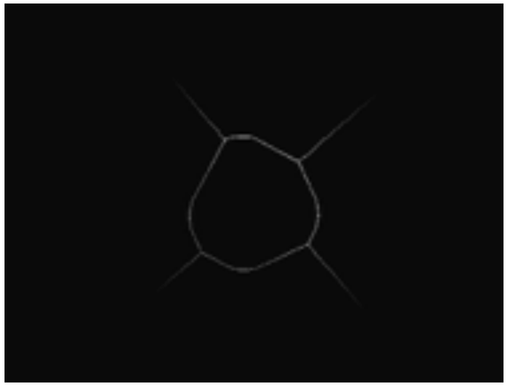

Original Image


Skeleton


Medial Axis Transform

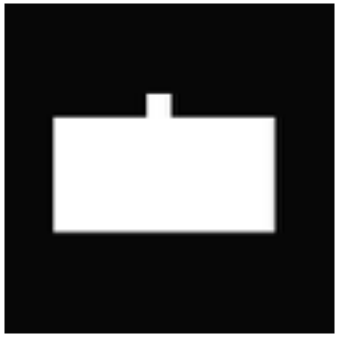

Original Image

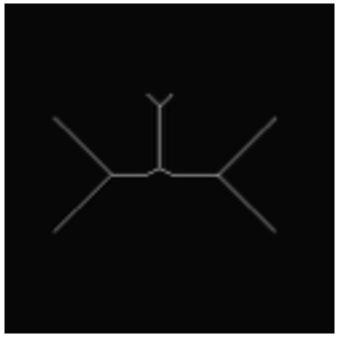

Skeleton

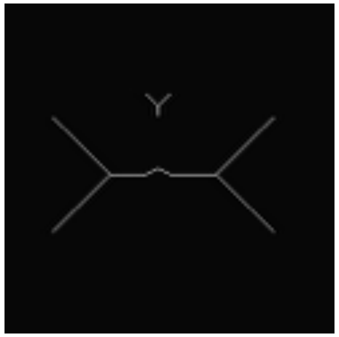

Medial Axis Transform

Skeletonization/ Medial Axis Transform (VIII)

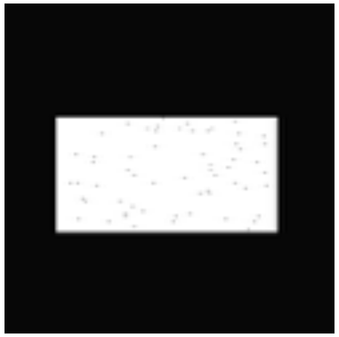
- Using a different algorithm which does not guarantee a connected skeleton.

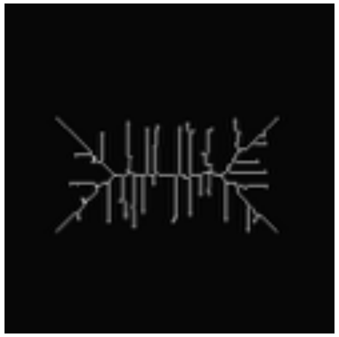

Original Image


Skeleton I


Skeleton II

- The corresponding skeleton connects each noise point to the skeleton obtained from the noise free image.


Pepper Noise in Original Image


Skeleton

Appendix

Thickening (VI)

- Example : *SKIZ*
  - The SKIZ is a skeletal structure that divides an image into regions, each of which contains just one of the distinct objects in the image.
  - The SKIZ is also sometimes called the Voronoi diagram.

1a)

0	0	0
	0	
1	1	1

1b)

	0	
1	0	0
1	1	

2a)

1		
1	0	1
1	1	1

2b)

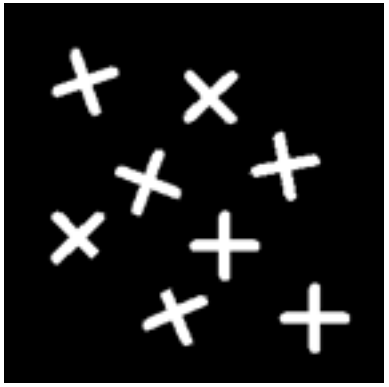
		1
1	0	1
1	1	1

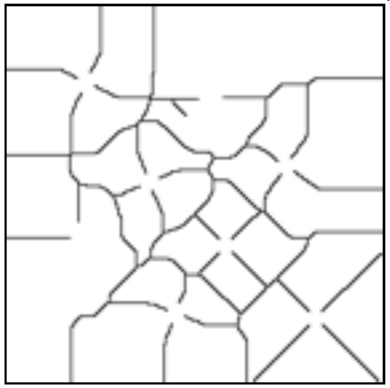
Structuring elements used in determining the SKIZ

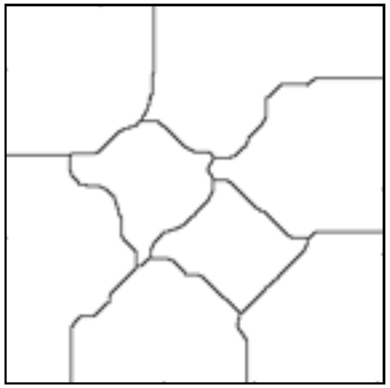
- **1a and 1b are used to perform the skeletonization of the background until convergence.**
- **On each thickening iteration, each element is used in turn, and in each of its 90°rotations.**
- **2a and 2b are used in similar fashion to prune the skeleton until convergence and leave behind the SKIZ.**

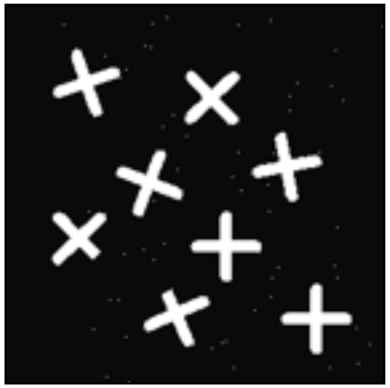
Thickening (VII)

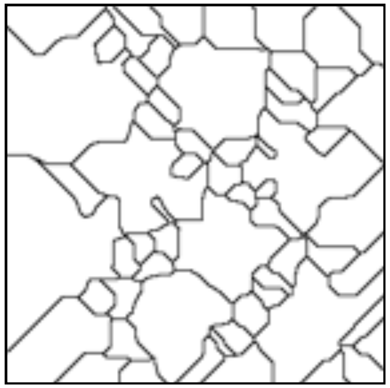
- Since the SKIZ considers each foreground pixel as an object to which it assigns a zone of influence, it is sensitive to noise.


Cross-shaped binary objects


The skeleton of the background


Pruning until convergence, the SKIZ of original image


Adding Salt Noise


The SKIZ of left image