MASURAREA PROPRIETĂŢI GEOMETRICE SIMPLE ALE OBIECTELOR DIN IMAGINI BINARE

Simple geometrical properties of binary objects

Technical University of Cluj Napoca
Computer Science Department

Simple geometrical properties

Simplification: imagines with a single object

Notations:

$$b(x,y)=\begin{cases}1 & pixel\_obj\\ 0 & pixel\_bck\end{cases}$$ $\Rightarrow$ value/label of the pixel at location (x,y)

Area

A= $\iint_I b(x,y)dxdy$

or in the discrete case:

$$A=\sum_{i=1}^{n}\sum_{j=1}^{m}b(i,j)$$ where: i =[1 .. n] and j = [1 .. m]

Technical University of Cluj Napoca
Computer Science Department

Center of mass (CM)

Center of mass:= the point in which the entire mas of the object can be concentrated without changing the first order moment on any axis:

1-st order moment on axis x:

$$\bar{x} \cdot \iint_I b(x,y)dxdy = \iint_I xb(x,y)dxdy \quad \text{or} \quad \bar{i} \cdot \sum_{i=1}^{n}\sum_{j=1}^{m} b(i,j) = \sum_{i=1}^{n}\sum_{j=1}^{m} ib(i,j)$$

1-st order moment on axis y:

$$\bar{y} \cdot \iint_I b(x,y)dxdy = \iint_I yb(x,y)dxdy \quad \text{or} \quad \bar{j} \cdot \sum_{i=1}^{n}\sum_{j=1}^{m} b(i,j) = \sum_{i=1}^{n}\sum_{j=1}^{m} jb(i,j)$$

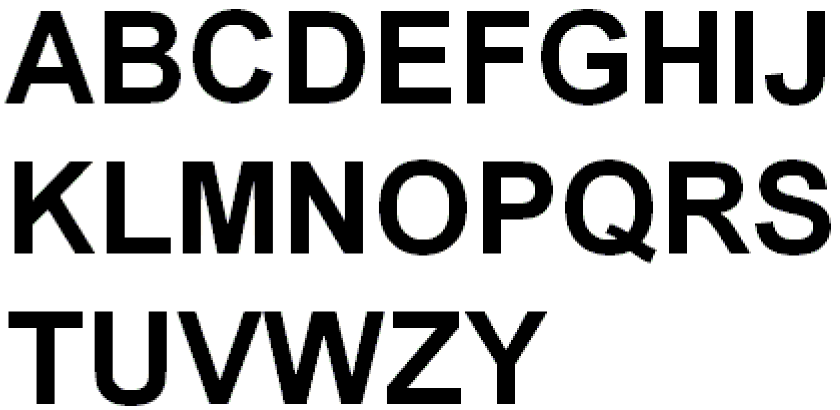
where ($\bar{x}$, $\bar{y}$) respectively ($\bar{i}$, $\bar{j}$) are the coordinates of the center of mass

$$\bar{i} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} ib(i,j)}{A} \quad , \quad \bar{j} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} jb(i,j)}{A}$$

Technical University of Cluj Napoca
Computer Science Department

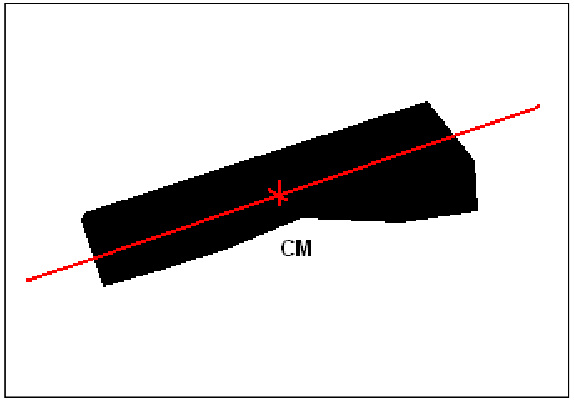
Example: labelling + cetner of mass computation

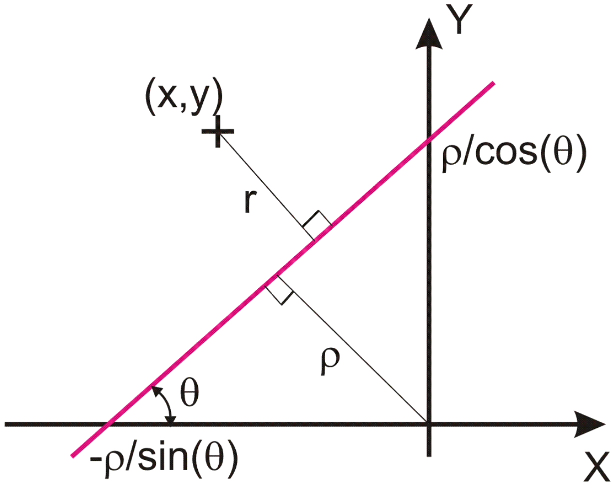
```
%      Label the connected pixel components in the text.png image, compute
%      their centroids, and superimpose the centroid locations on the
%      image.

bw = imread('labeling.bmp');
bw = ~bw;
L = bwlabel(bw);
s  = regionprops(L, ' ');
centroids = cat(1, s.Centroid);
imshow(bw)
hold on
plot(centroids(:,1), centroids(:,2), 'm*')
hold off
```



Orientation (*elongation axis*)

(minimum inertia axis / axis with the smallest 2-nd order moment)





2-nd order moment:

$$E = \iint_I r^2 b(x,y)\,dxdy \qquad (1)$$

Where:

r - is the distance between point *(x,y)* and the searched axis (minimum inertia axis).

The equation of the axis (polar coordinates):

$$x \sin\theta - y \cos\theta + \rho = 0 \qquad (2)$$

Solution: compute the line with the with the smallest 2-nd order moment: $E'(\rho,\theta)=0$

For a point *(x,y)* belonging to the object, its distance *r* to the line will be:

$$r^2 = (x \sin\theta - y \cos\theta + \rho)^2 \qquad (3)$$

Replacing (3) in (1):

$$E = \iint_I (x \sin\theta - y \cos\theta + \rho)^2\, b(x,y)\, dxdy \qquad (4)$$

The derivative of E by ρ equaled with 0:

$$E'_\rho = \iint_I 2(x \sin\theta - y \cos\theta + \rho) b(x,y) dxdy = 2\sin\theta \iint_I x b(x,y) dxdy - 2\cos\theta \iint_I y b(x,y) dxdy +$$
$$2\rho \iint_I b(x,y) dxdy = 2A(\bar{x} \sin\theta - \bar{y} \cos\theta + \rho) = 0 \qquad (5)$$

where: $(\bar{x}, \bar{y})$ is the center of mass of the object.

⇒ The minimum inertia axis passes through the center of mass (CM) of the object !

Translate the origin of the coordinate system in the CM:

$$x' = x - \bar{x} \quad \text{si} \quad y' = y - \bar{y}$$

⇒

$$x \sin\theta - y \cos\theta + \rho = x' \sin\theta - y' \cos\theta \qquad (6)$$

Replacing (6) in (4):

$E = a \cdot \sin^2\theta - b \cdot \sin\theta\cos\theta + c \cdot \cos^2\theta$

where a, b, c are the 2-nd order centered moments:

$$a = \iint_{I'} (x')^2\, b(x,y)\,dx'dy'$$

$$b = \iint_{I'} (x'y')\, b(x,y)\,dx'dy'$$

$$c = \iint_{I'} (y')^2\, b(x,y)\,dx'dy'$$

Rewriting *E* as:

$$E = \frac{1}{2}(a+c) - \frac{1}{2}(a-c)\cos 2\theta - \frac{1}{2} b \sin 2\theta \qquad (7)$$

The derivative of E by θ equaled with 0:

$$E'_\theta = (a-c)\sin 2\theta - b\cos 2\theta = 0$$

$$\tan 2\theta = \frac{b}{a-c}$$

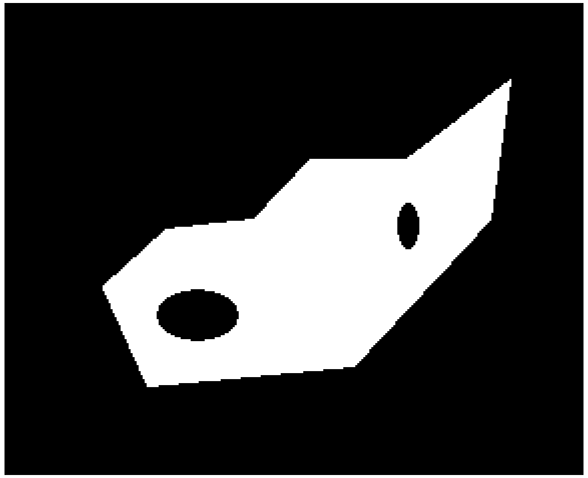
The case b=0 and a=c correspond to the horizontal and vertical lines and should be treated separately

$$\sin 2\theta = \pm\frac{b}{\sqrt{b^2+(a-c)^2}} \quad \text{and} \quad \cos 2\theta = \pm\frac{a-c}{\sqrt{b^2+(a-c)^2}}$$

the positive solution $\Rightarrow$ E-minim | the negative solution $\Rightarrow$ E-maxim

Technical University of Cluj Napoca
Computer Science Department

$$Shapefactor = \frac{L_{E\max}}{L_{E\min}}$$ (0 – linie, 1 cerc)

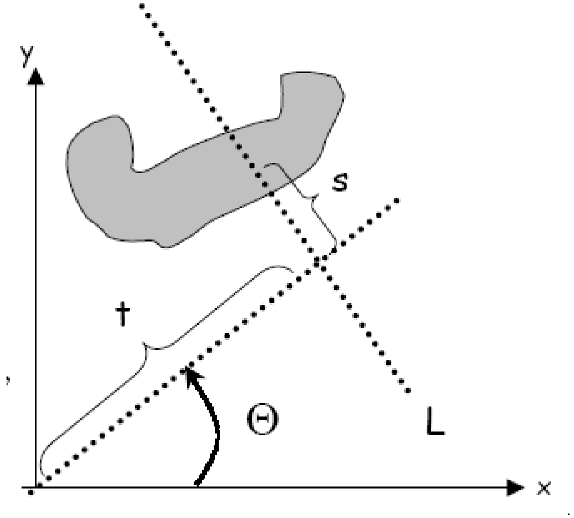


```
BW = imread('ob2.bmp');
I = double(BW);

A = regionprops(I, 'area')
P = regionprops(I, 'perimeter')
Euler = regionprops(I, 'EulerNumber')
teta = regionprops(I, 'orientation')
LEmin = regionprops(I, 'MajorAxisLength')
LEmax = regionprops(I, 'MinorAxisLength')
Shape_factor = LEmax.MinorAxisLength / LEmin.MajorAxisLength
Eccentricitaty = regionprops(I, 'Eccentricity')

Results:

A = Area: 14451
P = Perimeter: 588.3991
Eeler = EulerNumber: -1
teta = Orientation: 27.2073
LEmin = MajorAxisLength: 223.2852
LEmax = MinorAxisLength: 96.1508
Shape_factor = 0.4306
Eccentricitaty = Eccentricity: 0.9025
```

PROJECTIONS

Projection of an object on a line with direction θ:

$$p_\theta(t) = \int_L b(t\cdot\cos\theta - s\cdot\sin\theta, t\cdot\sin\theta + s\cdot\cos\theta)ds$$

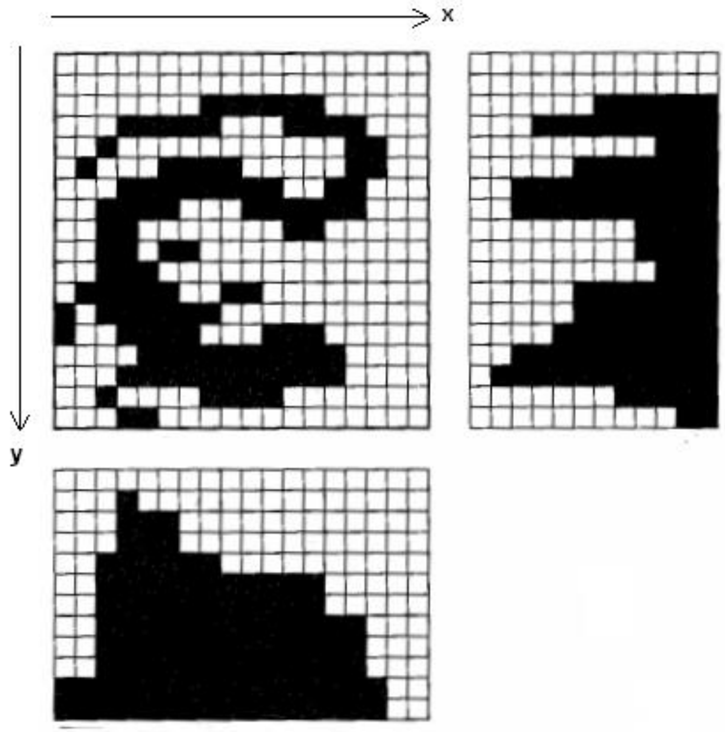


Vertical projection (θ= 0)

$$v(x) = \int_L b(x,y)dy$$

Horizontal projection (θ= π/2)

$$h(y) = \int_L b(x,y)dx$$



Area

$$A = \iint_I b(x,y)dxdy \quad ; \quad A = \int v(x)dx = \int h(y)dy$$

Center of mass (CM)

$$\bar{x}\,A = \iint_I xb(x,y)dxdy = \int xv(x)dx$$

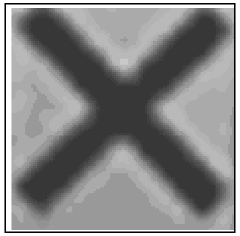
$$\bar{y}\,A = \iint_I yb(x,y)dxdy = \int yh(y)dy$$

Applications of the projections

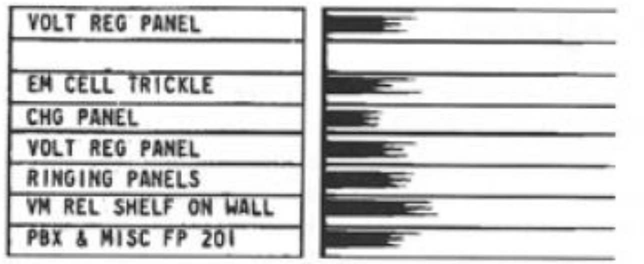
Problem 1: Segment / label each letter from the text

ABTH HGH
AJJA
ABSN ANS
ALOPL

Problem 2:
Detect the center of the ‚X' shape:



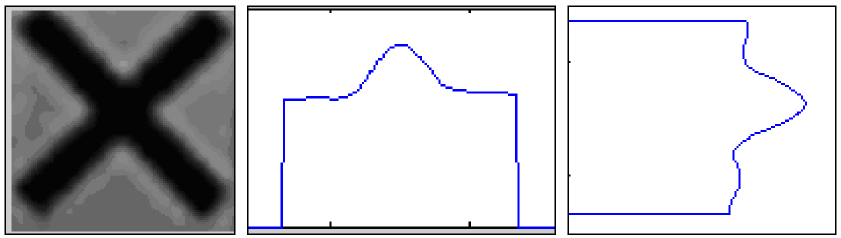
First step in character recognition ⇒ detection of each character (lines and columns)



Horizontal projection h(y)

Vertical projection v(x)

Pattern recognition / geometrical properties of shapes



a b c

a. Rectangular region around an “X’’-shape
b. Sum of the intensities around columns;
c. Sum of the intensities around rows

SVR

Run-Length encoding (compression)

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

Ver. 1

Start position and length of "1" sequences:

(1,3) (7,2) (12,4) (17,2) (20,3)
(5,13) (19,4)
(1,3) (17,6)

Ver. 2

Length of 0 and 1 sequences : 0,3,3,2,4,1,2,1,3
4,13,1,4 i
3,13,6

k

Notations:

- r_{ik} sequence *k* of line *i*
- first sequence in each line is a seq. of 0 $\Rightarrow$ even seq. correspond to "1" ($k=2j$).
- m_i sequences on line *i*

Technical University of Cluj Napoca
Computer Science Department

OPERATII MORFOLOGICE

Morphological operations

Technical University of Cluj Napoca
Computer Science Department

OPERATII MORFOLOGICE / *Morphological operations*

Morphology := [moprphos = shape] shape and structure of living organisms

Mathematical morphology $\Rightarrow$ tools for modifying the shape / detection of components / representation and description of regions / of an object

Set theory $\Rightarrow$ Language used in mathematical morphology

Let A a **set** in Z^2. If a = (a_1,a_2) is an element in A:

$a \in A$.

Similar, if a **is not** an element in A:

$a \notin A$.

Set vid zero elements: Ø.

Notation: { … }

Elements of the considered sets: pixels *b(x,y)* of binary images

Technical University of Cluj Napoca
Computer Science Department

Operatuions on sets

1. Inclusion

$A \subseteq B$

2. Union

$C = A \cup B$

3. Intersection

$D = A \cap B$

4. Disjunctive sets (mutual exclusive)

$A \cap B = \varnothing$.

5. Complement

$A^C = \{w \mid w \notin A\}$

6. Difference

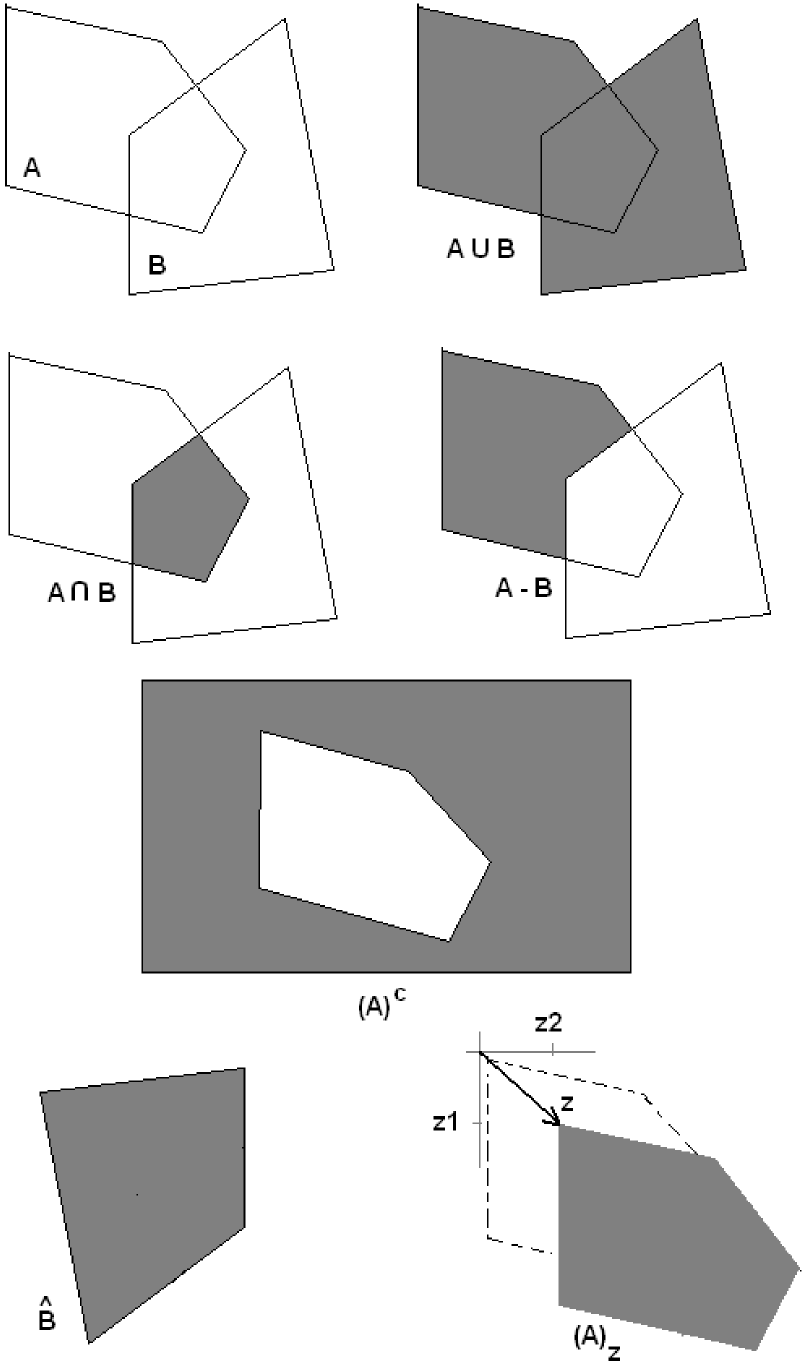
$A\text{-}B = \{w \mid w \in A, w \notin B\} = A \cap B^C$

7. Reflection (horizontal + vertical flip)

$\hat{B} = \{w \mid w = -b, \text{ for } b \in B\}$

8. Translation (of set *A* by *z=(z₁,z₂)*)

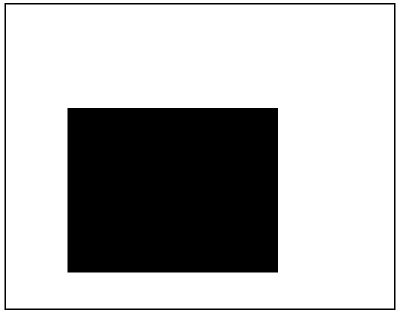
$(A)_Z = \{c \mid c = a + z, \text{ for } a \in A\}$



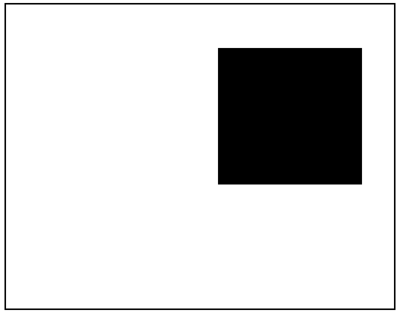
Logical and arithmetical operations applied on sets (binary images)

- Unary: image op scalar_operand
- Binary: image1 op image2
- Applied on pixel level !!!

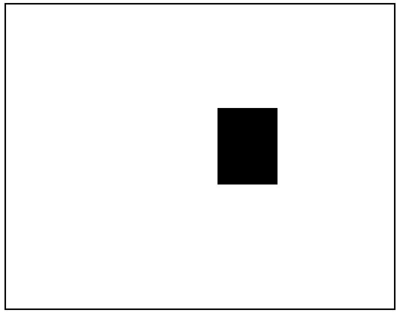
Logical operations: AND, OR, and NOT (COMPLEMENT) + combinations



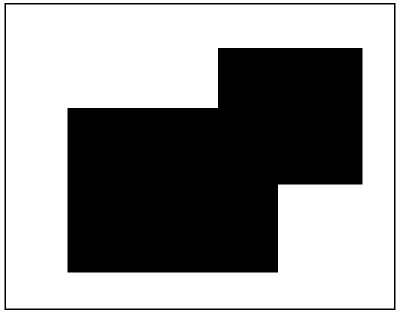
A



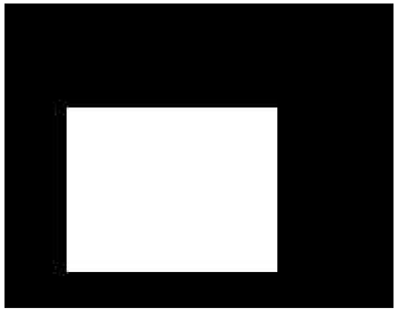
B



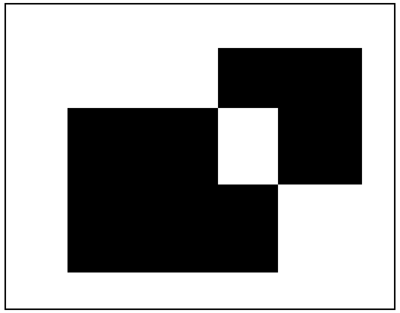
A and B



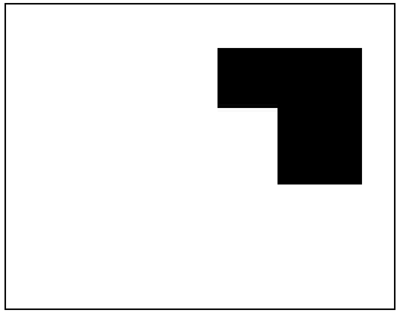
A or B



not (A) = A^C



A xor B



not(A) and B = B-A

DILATION AND EROSION

The primitives of the morphological operations !

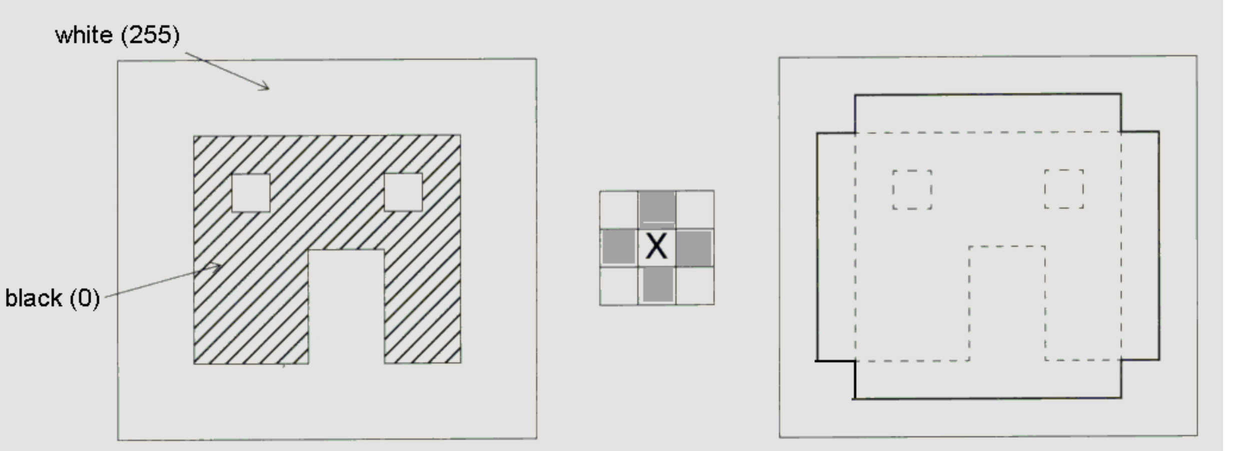
A, B $\subset$ Z^2

DILATION

Dilation of ***A*** by ***B***

$$A\oplus B=\{z|(\hat{B})_z\cap A\neq \varnothing\} \quad \text{or} \quad A\oplus B=\{z|[(\hat{B})_z\cap A]\subseteq A\}$$

B – structuring element

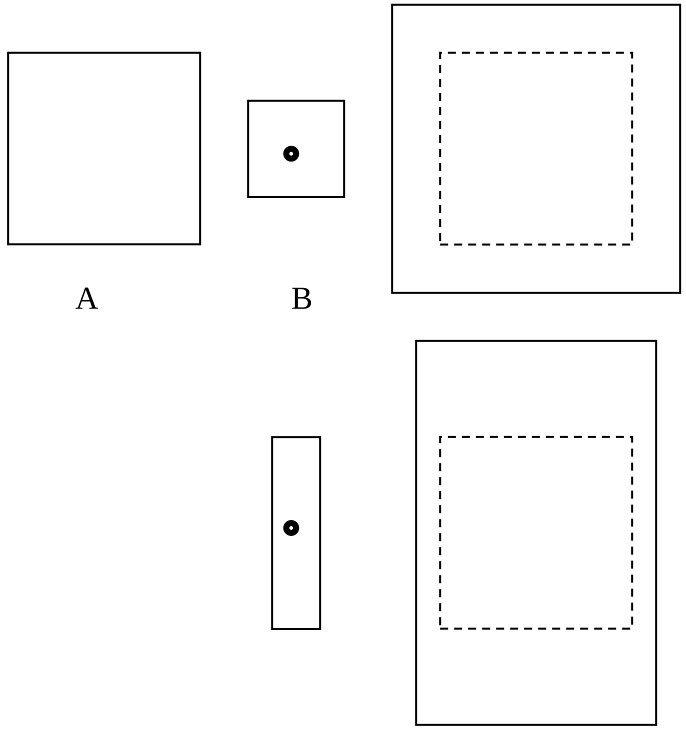


a. Original image.

b. Structural element; x = origin.

c. Image after dilation; original in dashes.

Technical University of Cluj Napoca
Computer Science Department



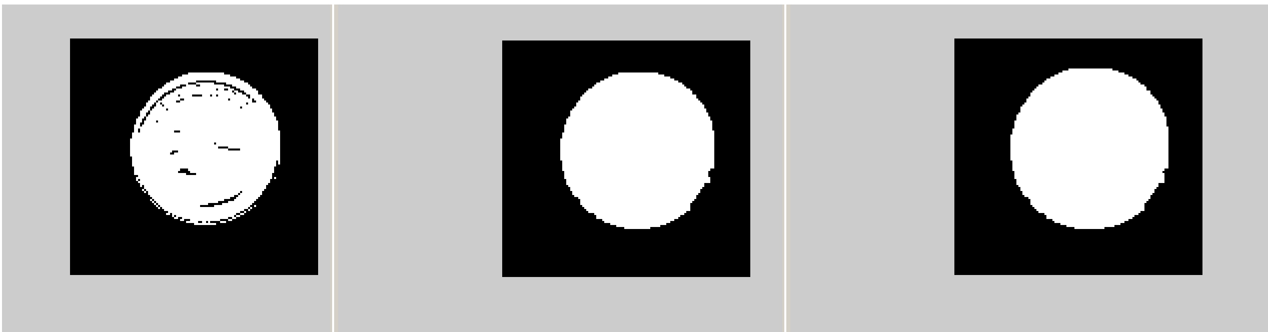
Practical algorithm (dilation)

Apply the structuring element by systematically scanning the source image:

1. If the origin of the structuring element B is applied over a background pixel (‘0’) ⇒ do nothing

2. If the origin of the structuring element is applied over a foreground/object pixel (‘1’) ⇒ perform a logic ‘OR’ between the pixels of the structuring element and the overlapped image pixels.

:



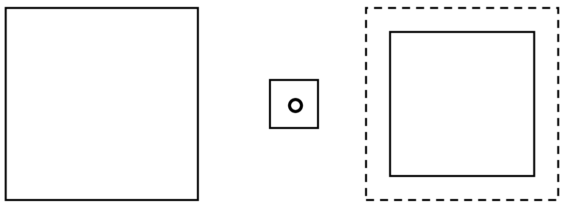
Applications: *holes filling ....*

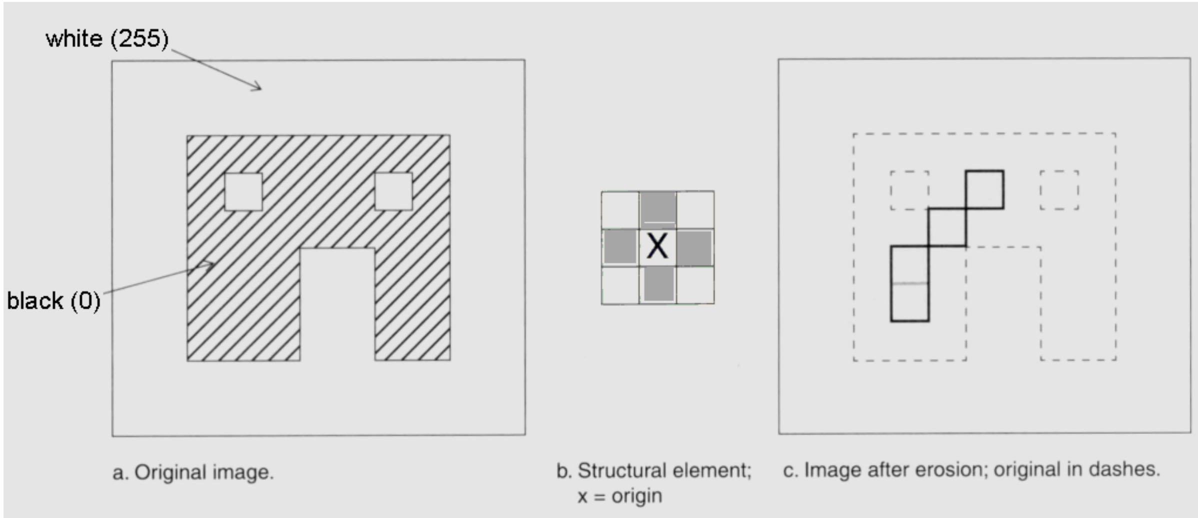
```
SE = strel('square',3)

BW = imread('ob1.bmp');
figure; imshow(BW);

BW1 = imdilate(BW,SE);
figure; imshow(BW1);

BW2 = imdilate(BW1,SE);
figure; imshow(BW2);
```

Erosion

Erosion of *A* by *B*

$$A \ominus B = \{ z \mid (\hat{B})_z \subseteq A \}$$





a. Original image.

b. Structural element; x = origin

c. Image after erosion; original in dashes.

Practical algorithm (erosion)

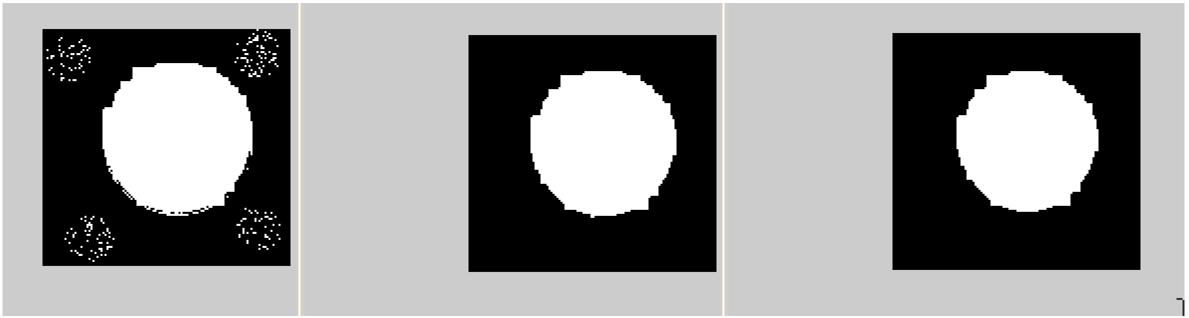
Apply the structuring by systematically scanning the source image:

1. If the origin of the structuring element B is applied over a background pixel (‘0’) ⇒ do nothing

2. 2. If the origin of the structuring element is applied over a foreground/object pixel (‘1’) AND any of the of the ‘1’ pixels of the structuring element B overlaps a background pixel (‘0’) in the source image A (extends outside A) ⇒ change the pixel in the destination image into background pixel (‘0’).

Application: Eliminate small objects (noise) ...

```
SE = strel('square',3)

BW = imread('ob11.bmp');
figure; imshow(BW);

BW1 = imerode(BW,SE);
figure; imshow(BW1);

BW2 = imerode(BW1,SE);
figure; imshow(BW2);
```



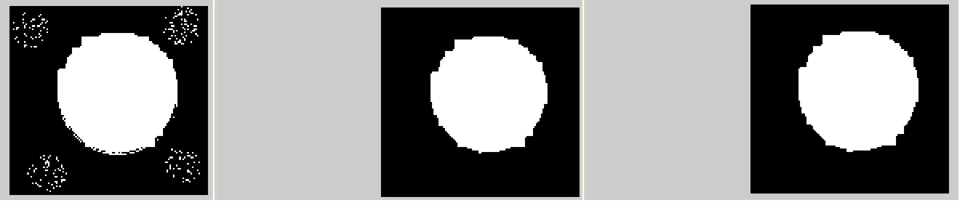
OPENING and CLOSING

Opening

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

Applications: *contour smoothing*, eliminate small objects (noise), ...

```
SE = strel('square',3)

BW = imread('ob11.bmp');
figure; imshow(BW);

BW1 = imerode(BW,SE);
figure; imshow(BW1);

BW2 = imdilate(BW1,SE);
figure; imshow(BW2);
```

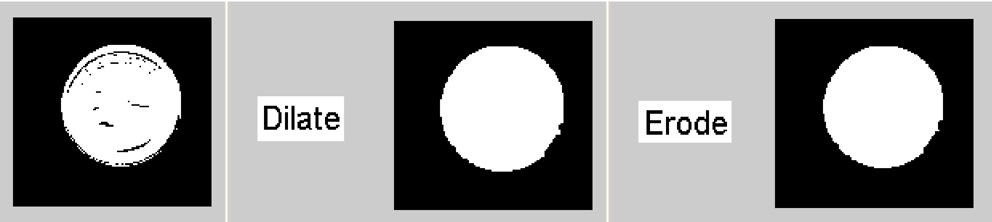


CLOSING

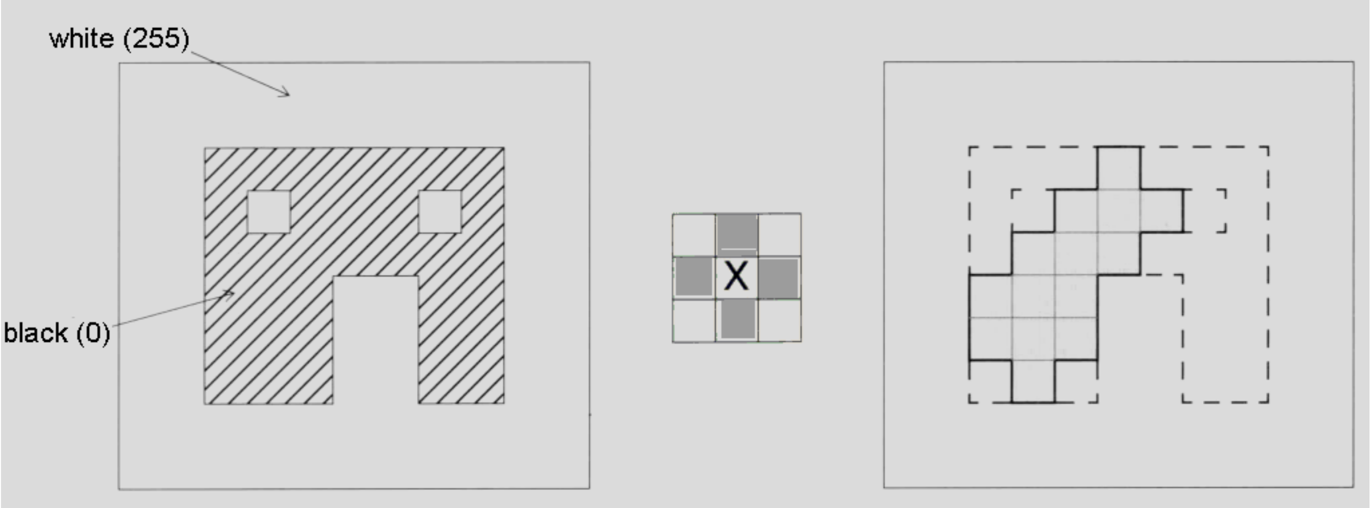
$$A \bullet B = (A \oplus B) \ominus B$$

Applications: contour smoothing, holes filling, ...

```
SE = strel('square',3)

BW = imread('ob1.bmp');
figure; imshow(BW);

BW1 = imdilate(BW,SE);
figure; imshow(BW1);

BW2 = imerode(BW1,SE);
figure; imshow(BW2);
```



Opening:

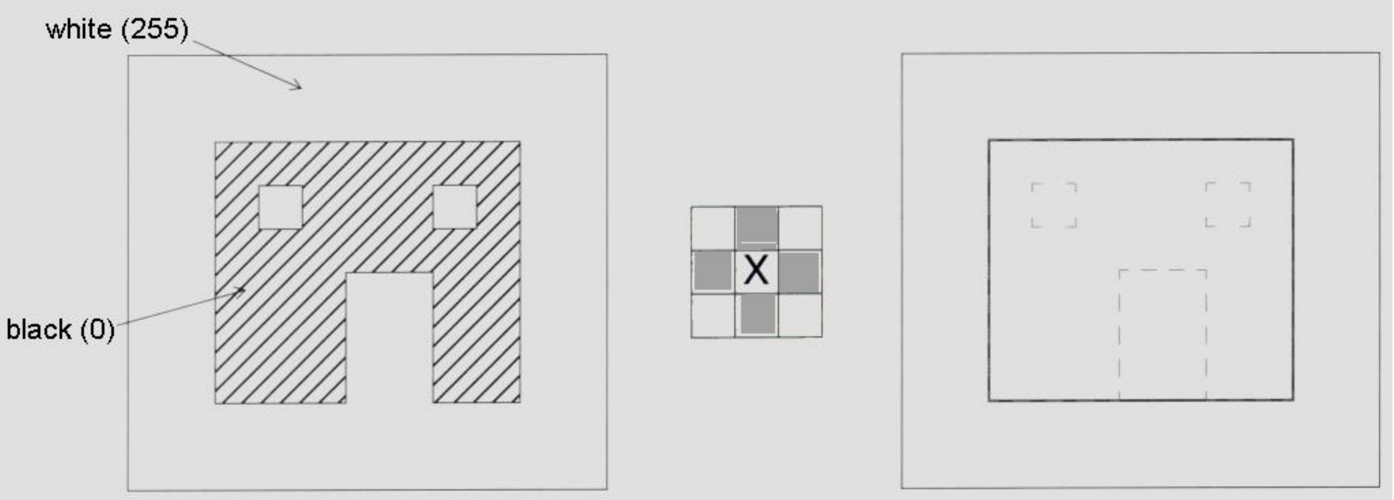


a. Original image.

b. Structural element; x = origin.

c. Image after opening; erosion followed by dilation.

Closing



a. Original image.

b. Structural element; x = origin.

c. Image after closing; dilation followed by erosion; original in dashes.

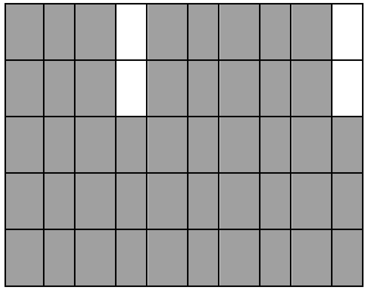
Properties of the morphological operators

1. $A \oplus B = B \oplus A$
2. $(A \ominus B)^C = A^C \oplus B$
3. $A \circ B \subseteq A$
4. $C \subseteq D \Rightarrow C \circ B \subseteq D \circ B$
5. $(A \circ B) \circ B = A \circ B$ (IDEMPOTENCY)
6. $A \subseteq A \bullet B$
7. $C \subseteq D \Rightarrow C \bullet B \subseteq D \bullet B$
8. $(A \bullet B) \bullet B = A \bullet B$ (IDEMPOTENCY)

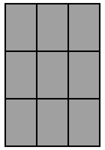
Applications of the basic morphological operators

Contour extraction

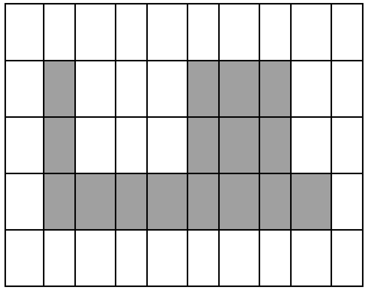
$\beta^i(A)=A-(A\Theta B)$ (interior contour)



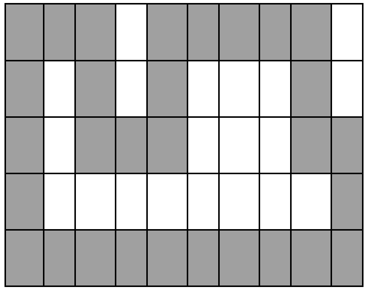
A



B



$A\,\Theta B$



$\beta^i(A)$

$\beta^e(A)=(A\oplus B)-A$ (exterior contour)

Region filling

Let p a point in the interior of the contour of A

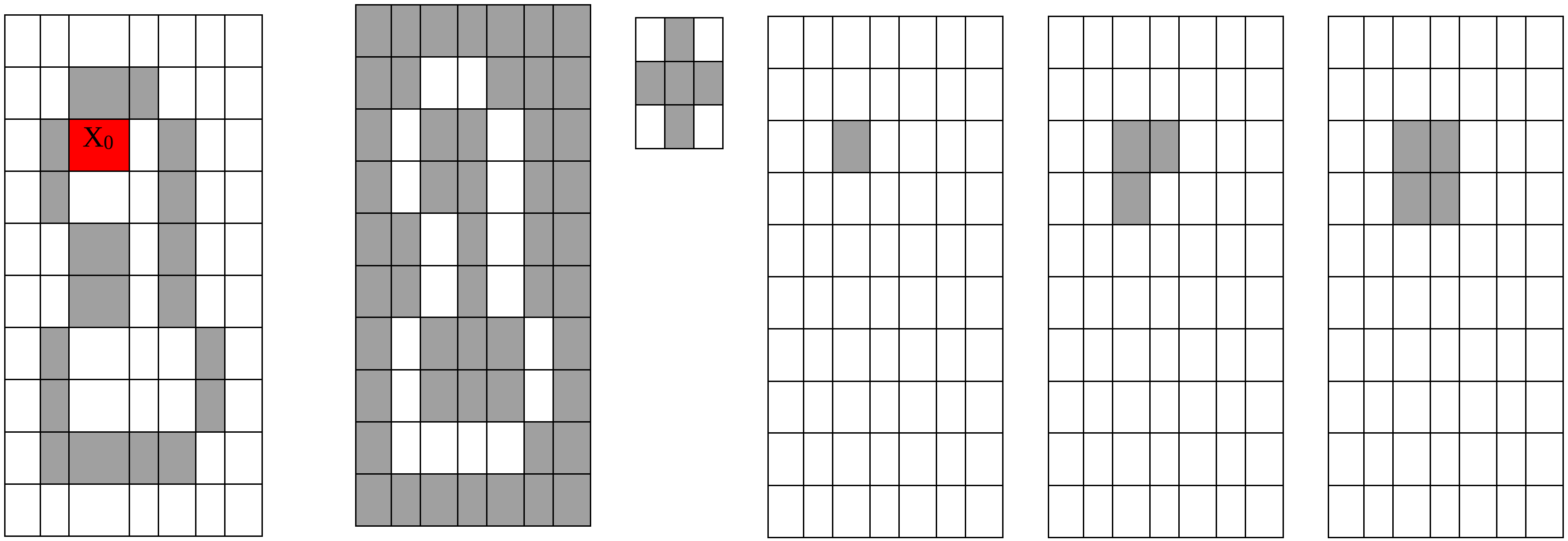
1. X_0 = p, (p='1')

2. $X_k = (X_{k-1} \oplus B) \cap A^C \quad k=1,2,3,$

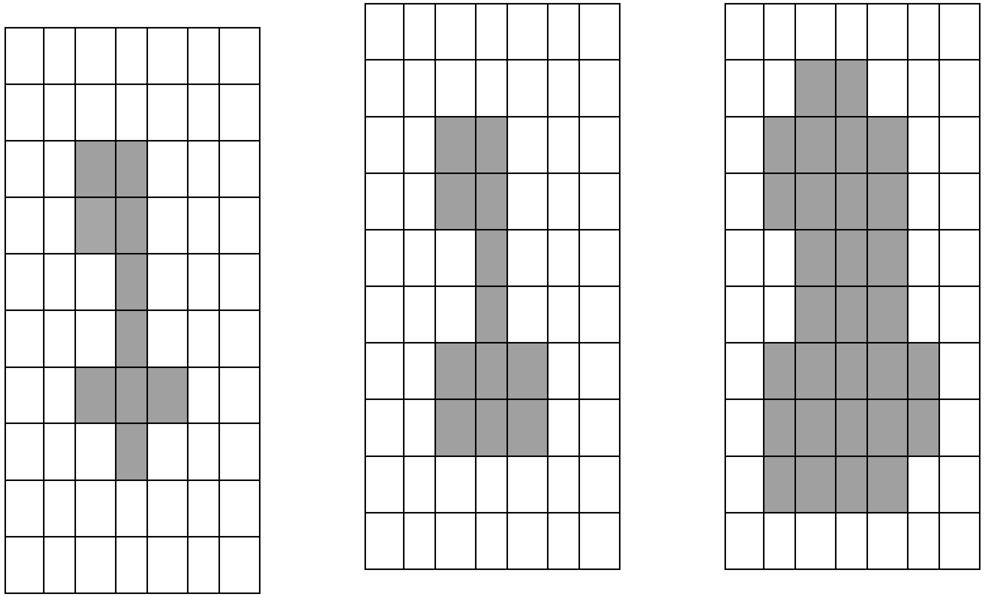
3. *If* $X_k = X_{k-1} \Rightarrow$ stop. Otherwise repeat 2.

Final filled object: $A \cup X_k$

Algorithm tracing:

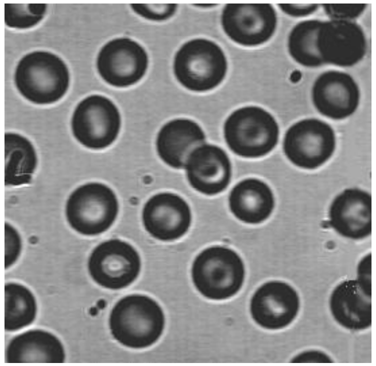


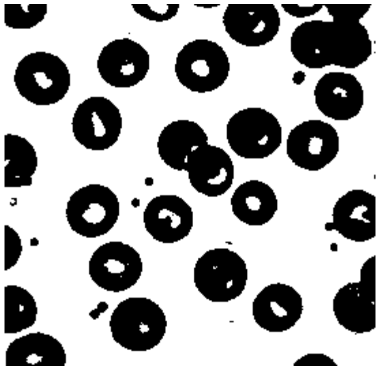
A **A^C** **B** **X0** **X1** **X2**



$\mathbf{X_6}$ $\mathbf{X_7}$ $\mathbf{X_7} \cup A$

Typical situation for usage:





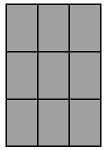
Labeling (not efficient for large images / objects)

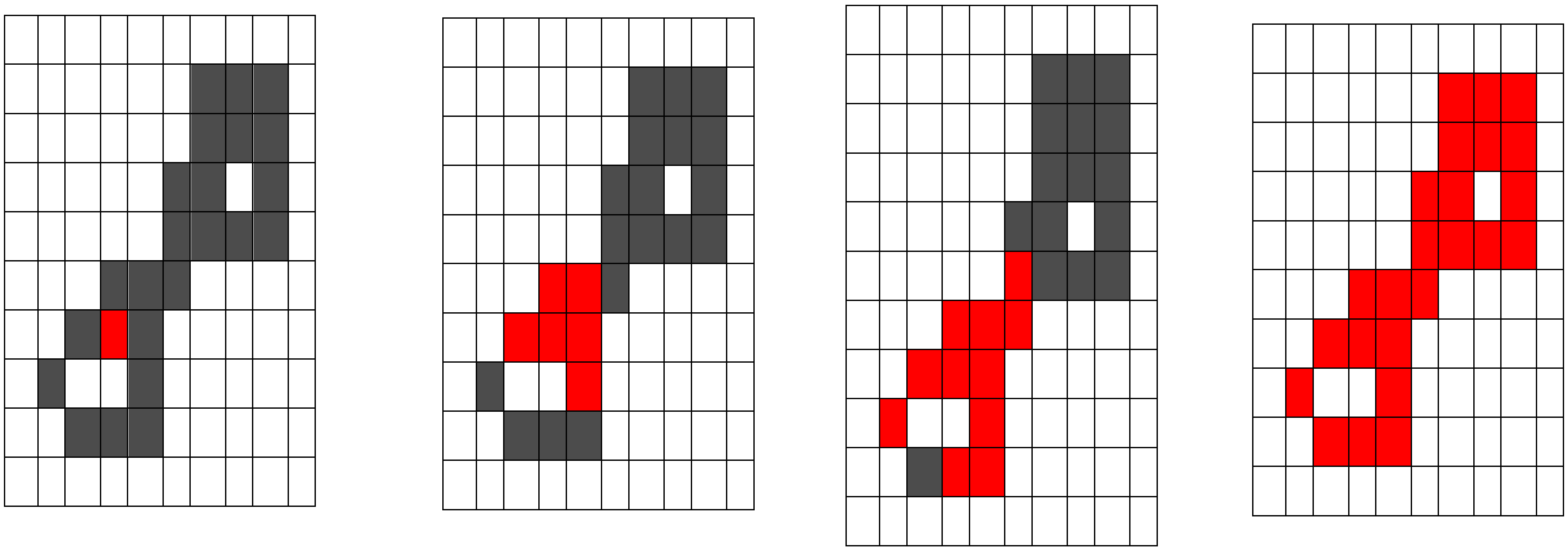
A = { Y_1, Y_2, ... Y_n}, Y_i – connected components, $Y_i \subseteq A$

1. $p \in Y$. $X_0 = p$

2. $X_k = (X_{k-1} \oplus B) \cap A \quad k=1,2,3,...$

3. Dc. $X_k = X_{k-1} \Rightarrow$ stop ($Y_i = X_k$). Otherwise repeat 2.





Y, p **X_1** **X_2** **X_3 = Y**

HIT-AND-MISS transform

Used to select pixels with specific geometrical properties: corners, isolated points, contour points, template matching, thinning, thickening etc.

The hit & miss transform of a set A by structuring elements *(J,K)*:

$$A \otimes (J,K) = (A \ominus J) \cap (A^C \ominus K).$$

0	0	0
	1	
1	1	1

J

	0	0
1	1	0
	1	

K

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

A

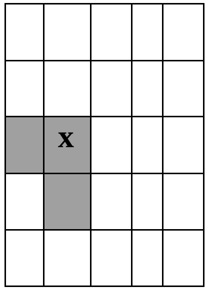
$\longrightarrow$

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

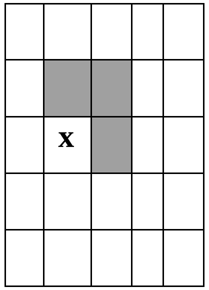
A⊗(J,K)

Technical University of Cluj Napoca
Computer Science Department

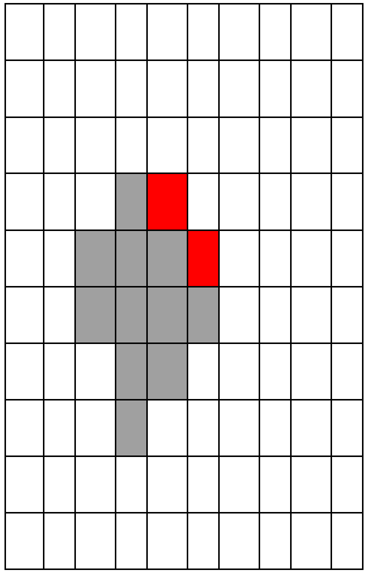
Ex.2: Corners detection (NE)



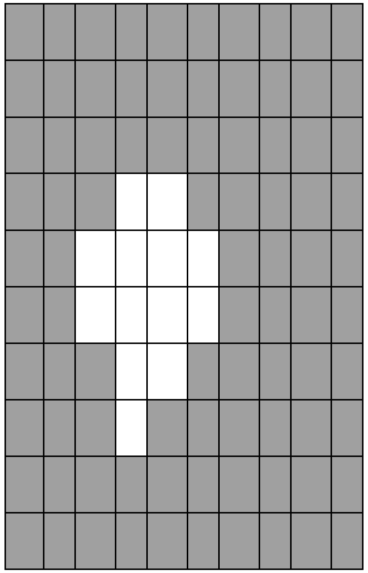
J



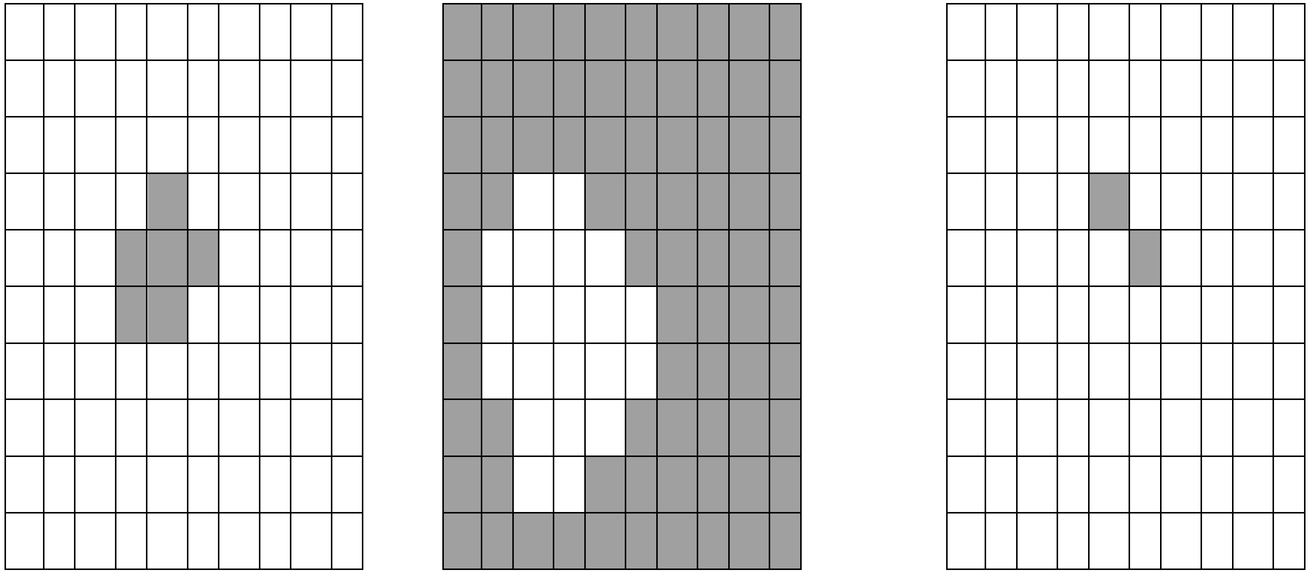
K



A



A^C

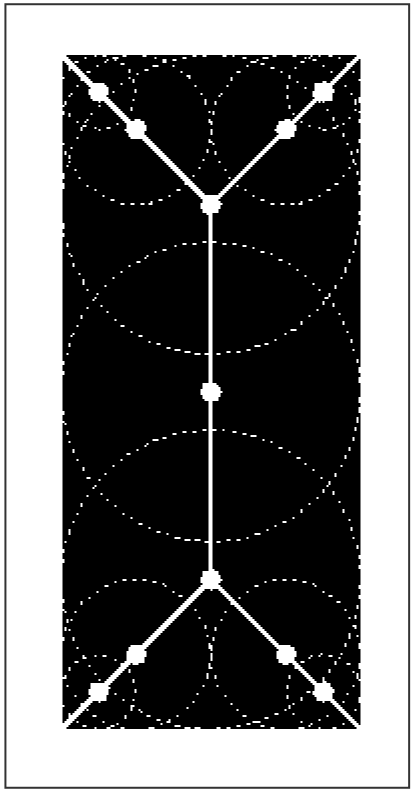


$(A\Theta J)(A^C\Theta K)$ $A\otimes(J,K)=(A\Theta J)\cap(A^C\Theta K)$

Application (Matlab)

```
BW2 = bwhitmiss(BW1,SE1,SE2)
```

SKELETON EXTRACTION



The Skeleton of set A:

$$S(A) = \bigcup_{k=0}^{K} S_k(A)$$

$$S_k(A) = (A\Theta kB) - (A\Theta kB) \circ B$$

$$A\Theta kB = (...(A\Theta B)\Theta B).......)\Theta B$$

$$K = \max\{k \mid (A\Theta kB) \neq \Phi$$

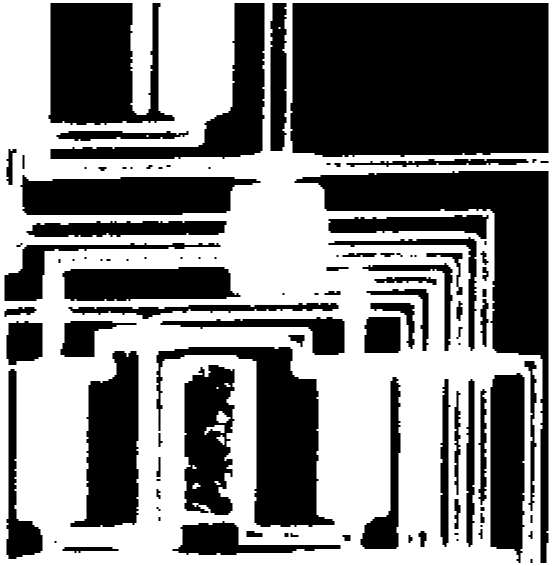
Reconstruction of set A (K – should be known):

$$A = \bigcup_{k=0}^{K} (S_k(A) \oplus kB)$$

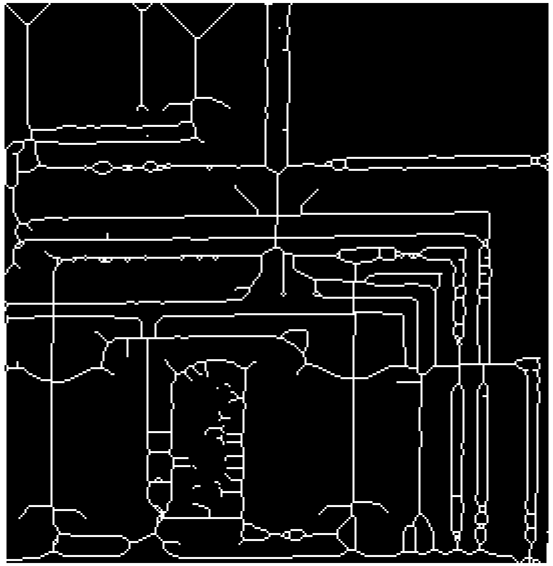
Technical University of Cluj Napoca
Computer Science Department

Example:

```
BW1 = imread('circbw.tif');
BW2 = bwmorph(BW1,'skel',Inf);
imshow(BW1)
figure, imshow(BW2)
```



Original Image



Skeletonization of Image

References:

[1] Robert M. Haralick, Linda G. Shapiro, *Computer and Robot Vision*, Addison-Wesley Publishing Company, 1993

[2] Rafael C. Gonzalez, *Digital Image Processing*, Prentice-Hall, 2002