Morphological Image Processing

Lecture 5

Robin Strand
robin@cb.uu.se

Centre for Image Analysis
Uppsala University

Computer Assisted Image Analysis
2014-02-12

Morphology

Form and structure

Mathematical framework used for:

- Pre-processing
  - Noise filtering, shape simplification, ...
- Enhancing object structure
- Segmentation
- Quantitative description
  - Area, perimeter, ...

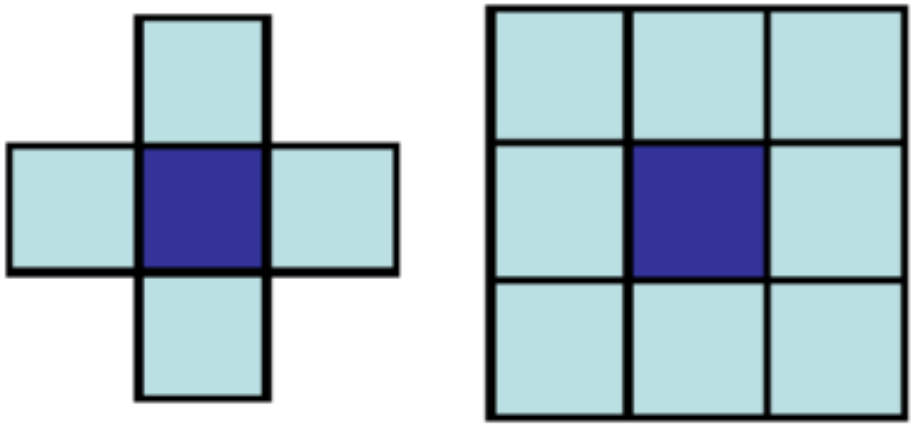
Neighborhoods and adjacencies

- N_4, 4-neighbors
- N_8, 8-neighbors

In a binary image, two pixels p and q are

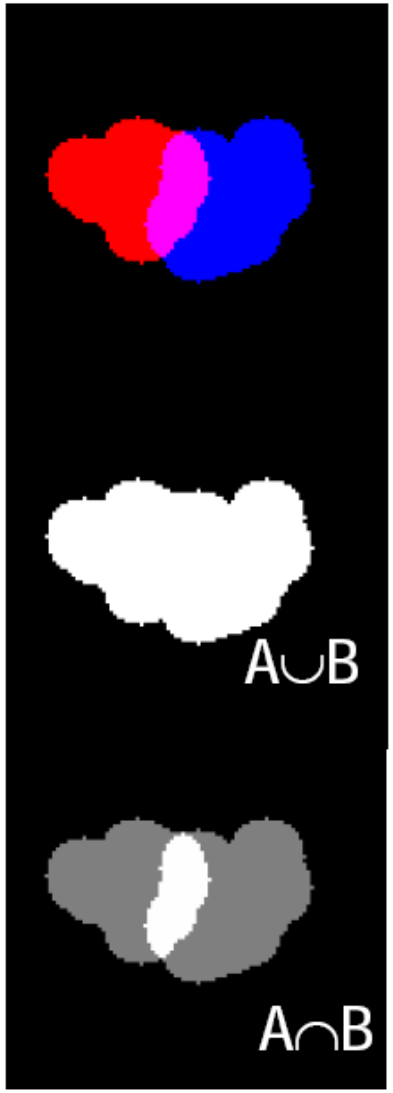
- 4-adjacent if they have the same value and q is in the set $N_4(p)$.
- 8-adjacent if they have the same value and q is in the set $N_8(p)$.

Two pixels are 4- or 8-connected if a 4- or 8-path can be drawn between them.



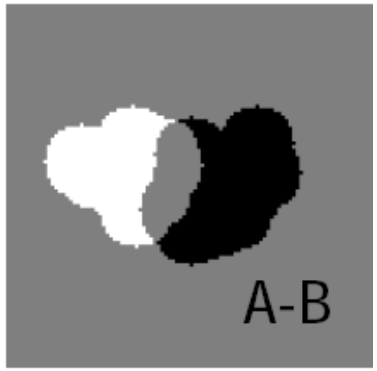
Some set theory

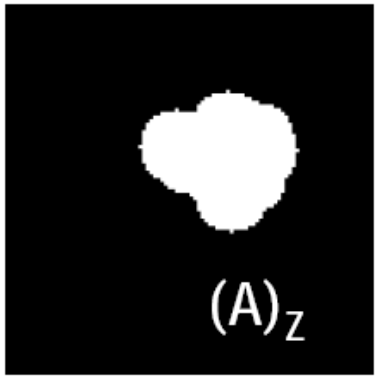
- A is a subset of $\mathbb{Z}^2$.
- If $a = (a_1, a_2)$ is an element in A: $a \in A$.
- If $a = (a_1, a_2)$ is *not* an element in A: $a \notin A$.
- *Empty set*: $\emptyset$.
- Set specified using $\{\,\}$, e.g., $C = \{w|\ \|w\| \leq 4\}$.
- Every element in A is also in B (subset): $A \subseteq B$.
- *Union* of A and B:
  $C = A \cup B = \{c|c \in A$ or $c \in B\}$.
- *Intersection* of A and B:
  $C = A \cap B = \{c|c \in A$ and $c \in B\}$.
- *Disjoint/mutually exclusive*: $A \cap B = \emptyset$.



Some more set theory

- *Complement* of A: $A^C = \{w | w \notin A\}$.
- *Difference* of A and B:
  $A - B = \{w | w \in A,\ w \notin B\} = A \cap B^C$.
- *Reflection* of A: $\hat{A} = \{w | -w \in A\}$.
- *Translation* of A by a vector $z = (z_1,\ z_2)$:
  $(A)_Z = \{c | c = a + z,\ \forall\, a \in A\}$.





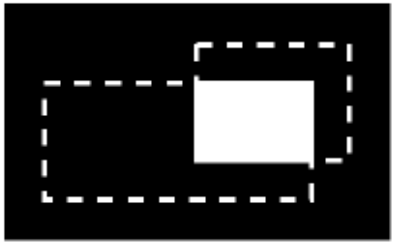
Logical operations

Pixel-wise combination of images (AND, OR, NOT, XOR)

A

B

NOT A



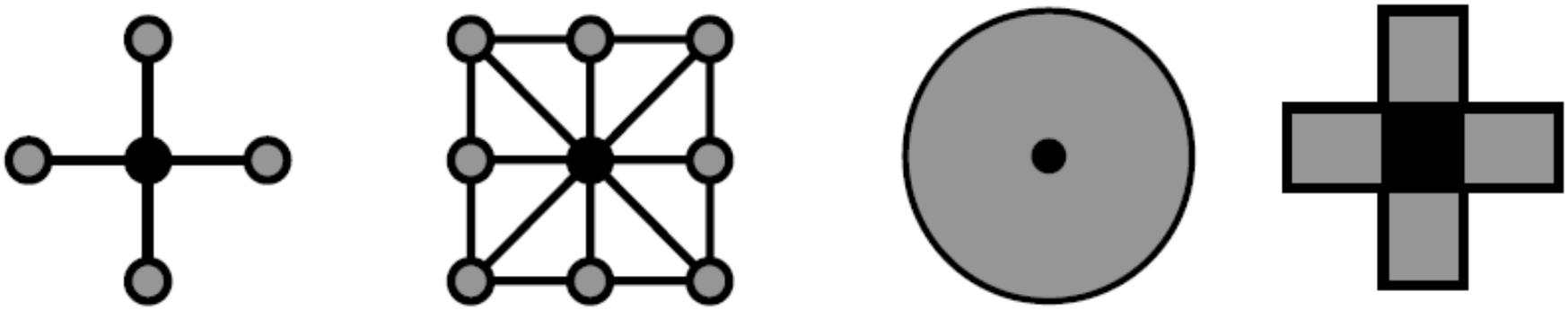
A AND B

A OR B

A XOR B

Structuring element (SE)

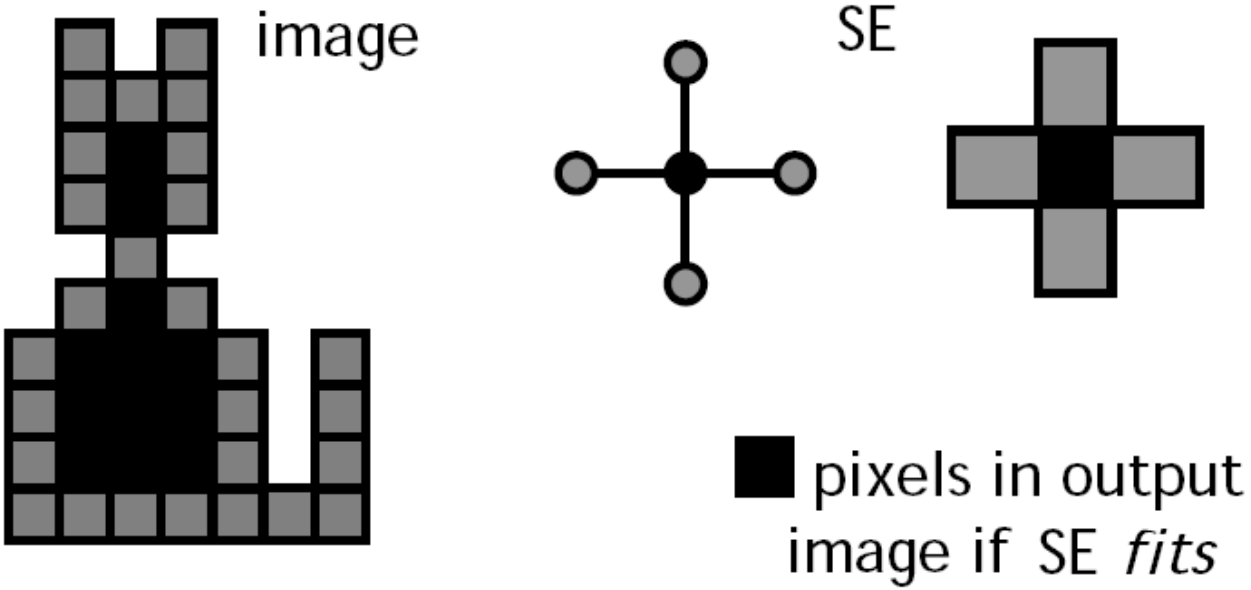
- Small set to probe the image under study.
- For each SE, define an origin:
  - SE in point p; origin coincides with p.
- Shape and size must be adapted to geometric properties for the objects.



Basic idea

In parallel for each pixel in binary image:

- Check if SE is *satisfied*.
- Output pixel is set to 0 or 1 depending on used operation.

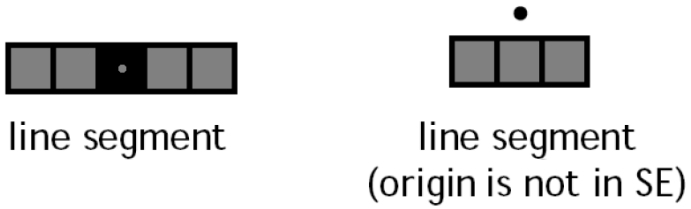


How to describe the SE

Possible in many different ways!

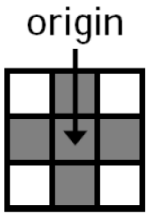
Information needed:

- Position of origin for SE.
- Position of elements belonging to SE.



N.b.

Matlab assumes it's center element to be the origin!



Five binary morphological transforms

- $\ominus$ Erosion.
- $\oplus$ Dilation.
- $\circ$ Opening.
- $\bullet$ Closing.
- $\otimes$ Hit-or-Miss transform.

$\ominus$ Erosion (shrinking)

Does the structuring element fit the set?

Erosion of a set X by structuring element B, $\varepsilon_B(X)$: all x in X such that B is in X when origin of $B = x$.

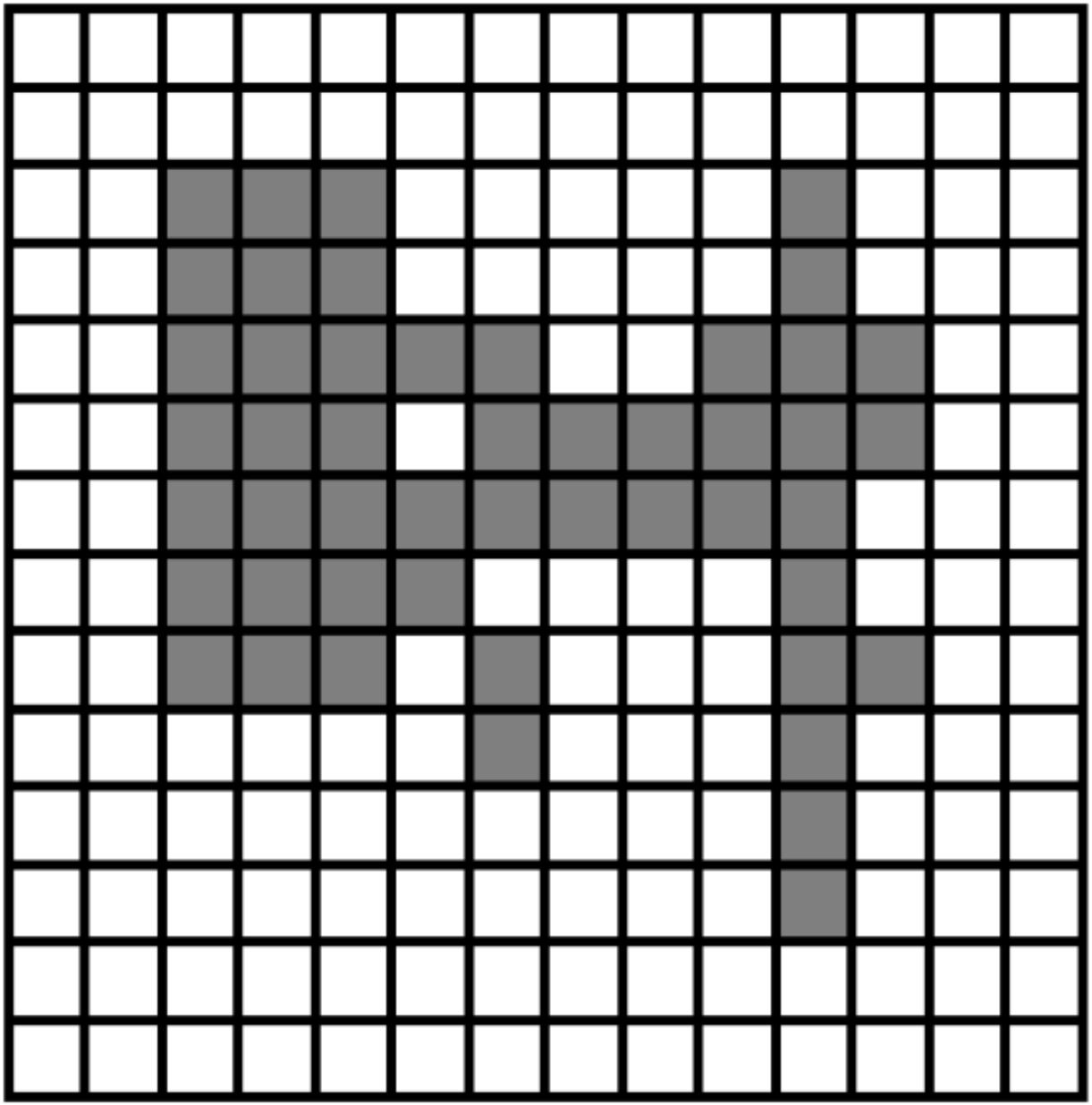
$$\varepsilon_B(X) = \{x | B_x \subseteq X\}.$$

Gonzalez-Woods:

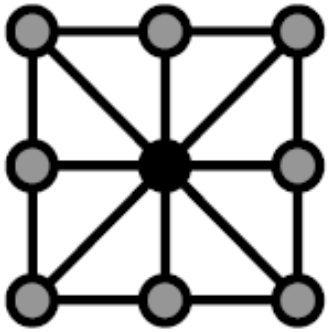
$$X \ominus B = \{x | (B)_x \subseteq X\}.$$

Shrink the object.

Example: erosion (fill in!)



SE=



$\oplus$ Dilation (growing)

Does the structuring element hit the set?

Dilation of a set X by structuring element B, $\delta_B(X)$: all x in X such that the reflection of B hits X when origin of $B = x$.

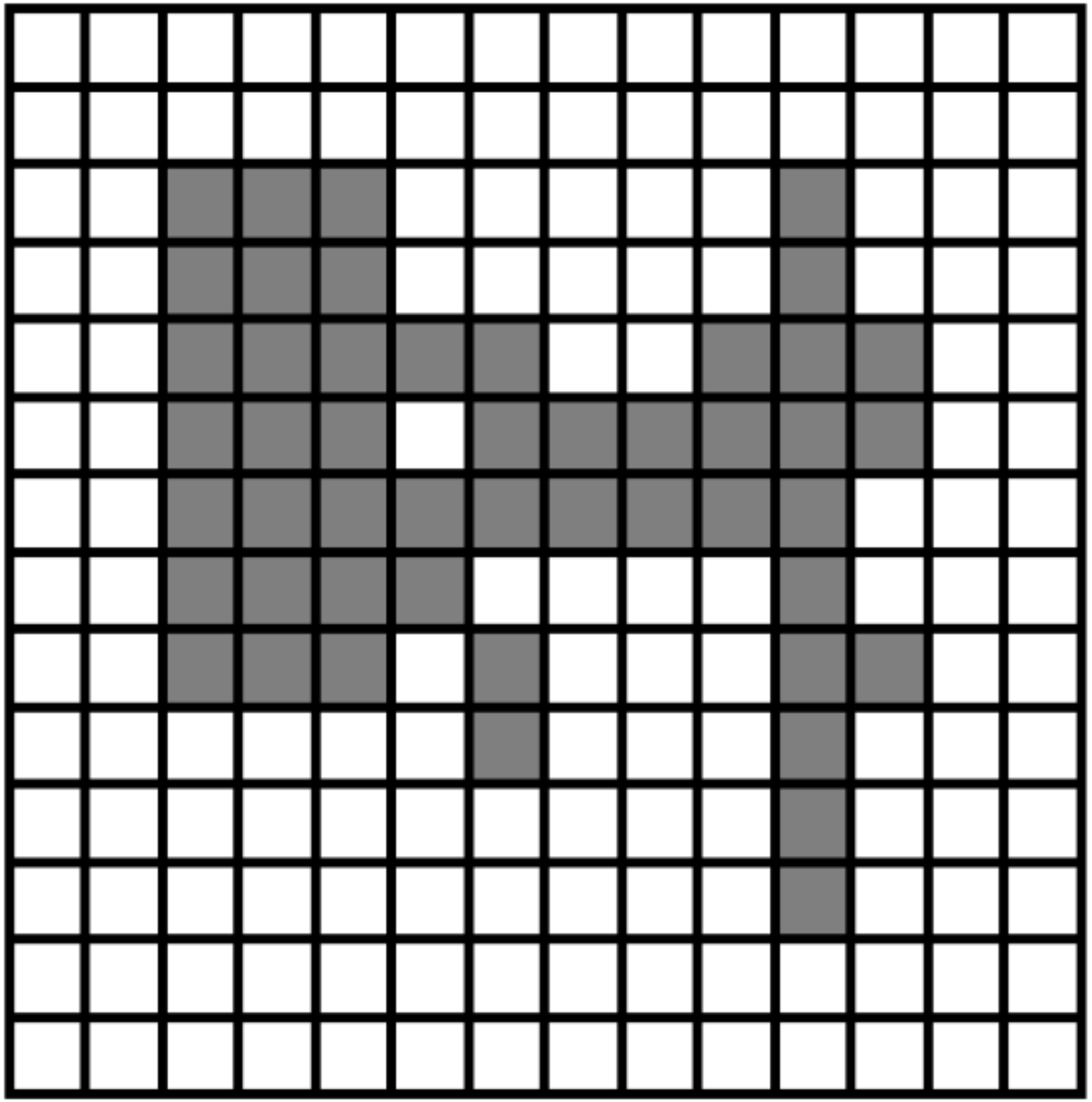
$$\delta_B(X) = \{x | (\hat{B})_x \cap X \neq \emptyset\}.$$

Gonzalez-Woods:

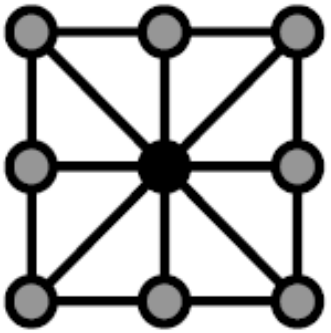
$$X \oplus B = \{x | (\hat{B})_x \cap X \neq \emptyset\}.$$

Grow the object.

Example: dilation (fill in!)

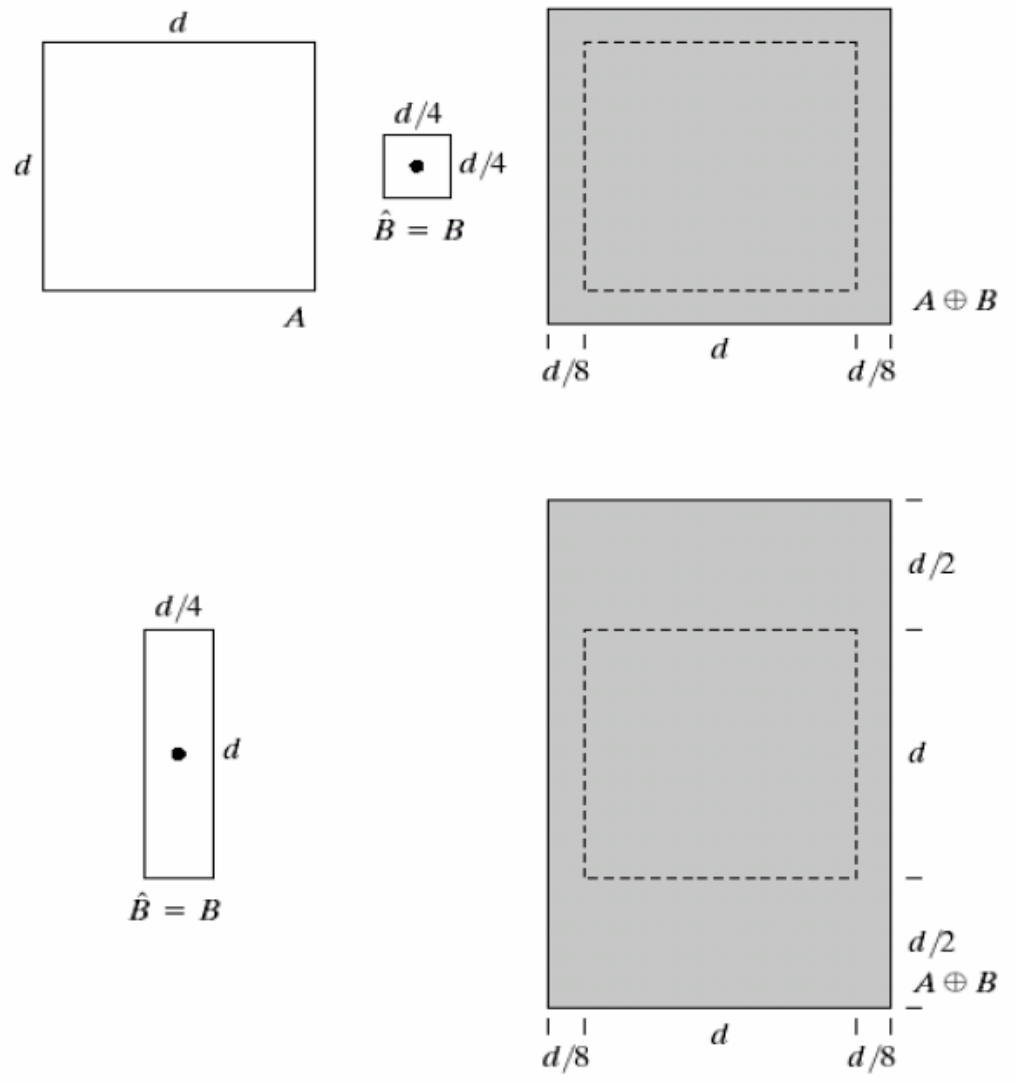


SE=



Different SE give different results

- Set A.
- Square structuring element (dot is the center).
- Dilation of A by B, shown shaded.
- Elongated structuring element (dot is the center).
- Dilation of A using this element.



Duality

Erosion and dilation are dual with respect to complementation and reflection,

$$(A \ominus B)^C = A^C \oplus \hat{B}.$$

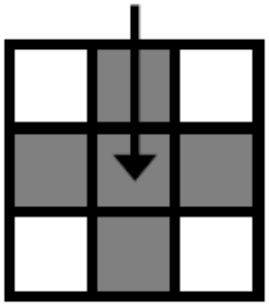
Examples

A

$A \ominus B$

$(A \ominus B)^c$

$B = \hat{B}$
origin



A^c

$A^c \oplus B$

Typical application

Erosion

Removal of structures of certain shape and size, given by SE (structure element).

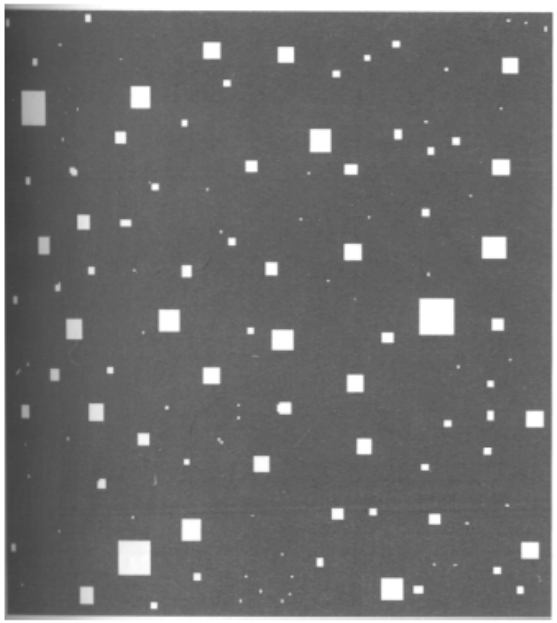
Example 3×3 SE

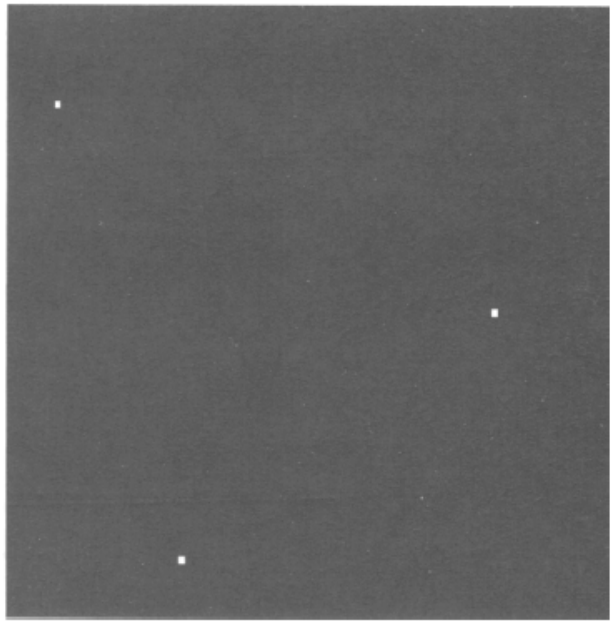
Dilation

Filling of holes of certain shape and size, given by SE.

Example 3×3 SE

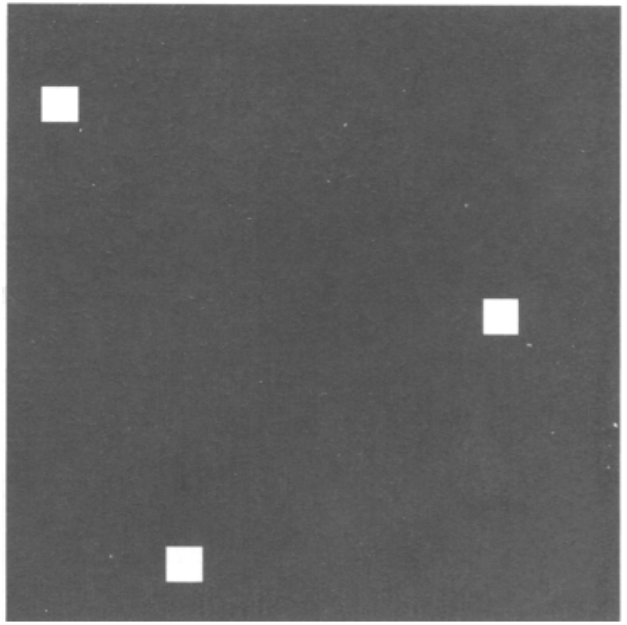
Examples





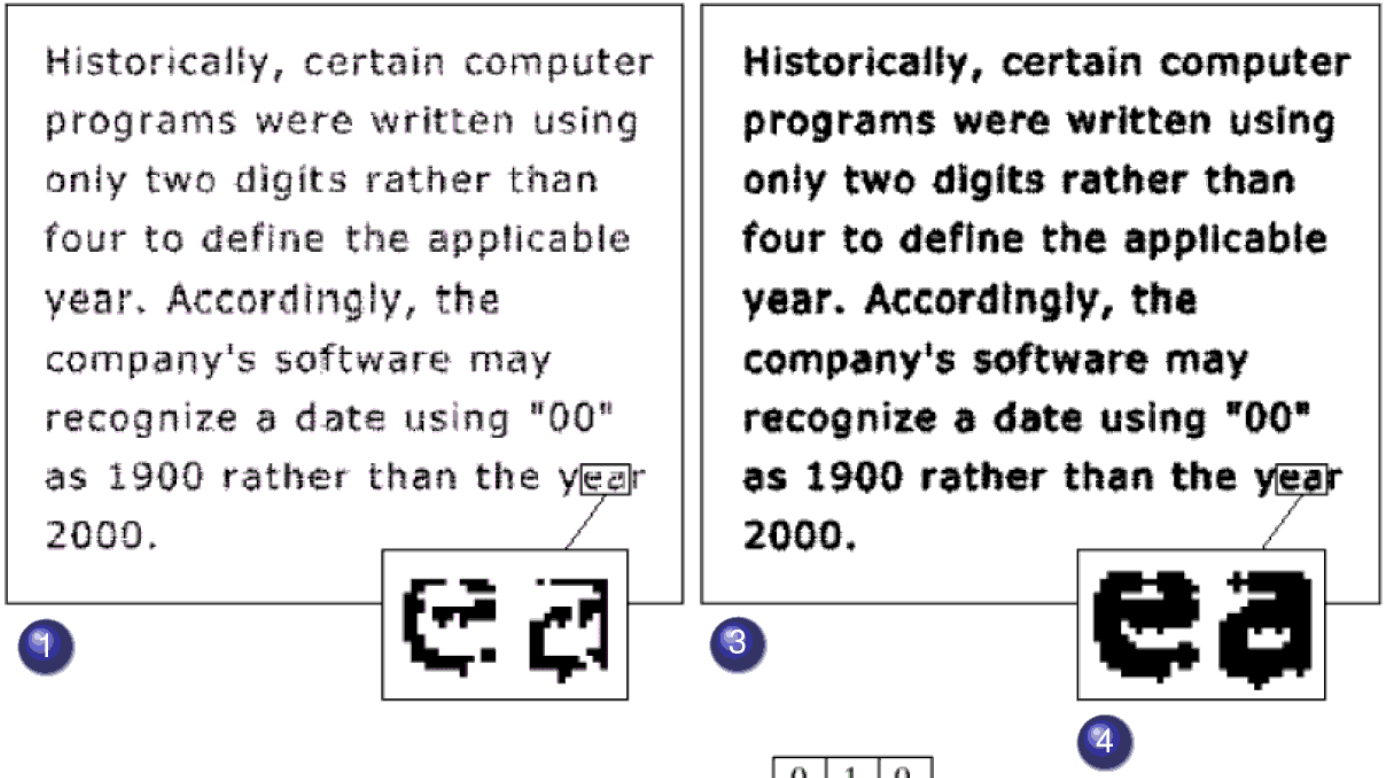
Erosion: SE = square of size 13×13.

Input: squares of size 1×1, 3×3, 5×5, 7×7, 9×9, and 15×15 pixels.



Dilation of erosion result: SE = square of size 13×13.

Use dilation to bridge gaps of broken segments



2

0	1	0
1	1	1
0	1	0

1. Sample text of poor resolution with broken characters (magnified view).
2. Structuring element.
3. Dilation of (1) by (2).
4. Broken segments were joined.

Use dilation to bridge gaps of broken segments

Wanted:

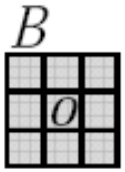
Remove structures/fill holes without affecting remaining parts.

Solution:

Combine erosion and dilation (using same SE).

- ○ Opening.
- • Closing.

$\circ$ Opening

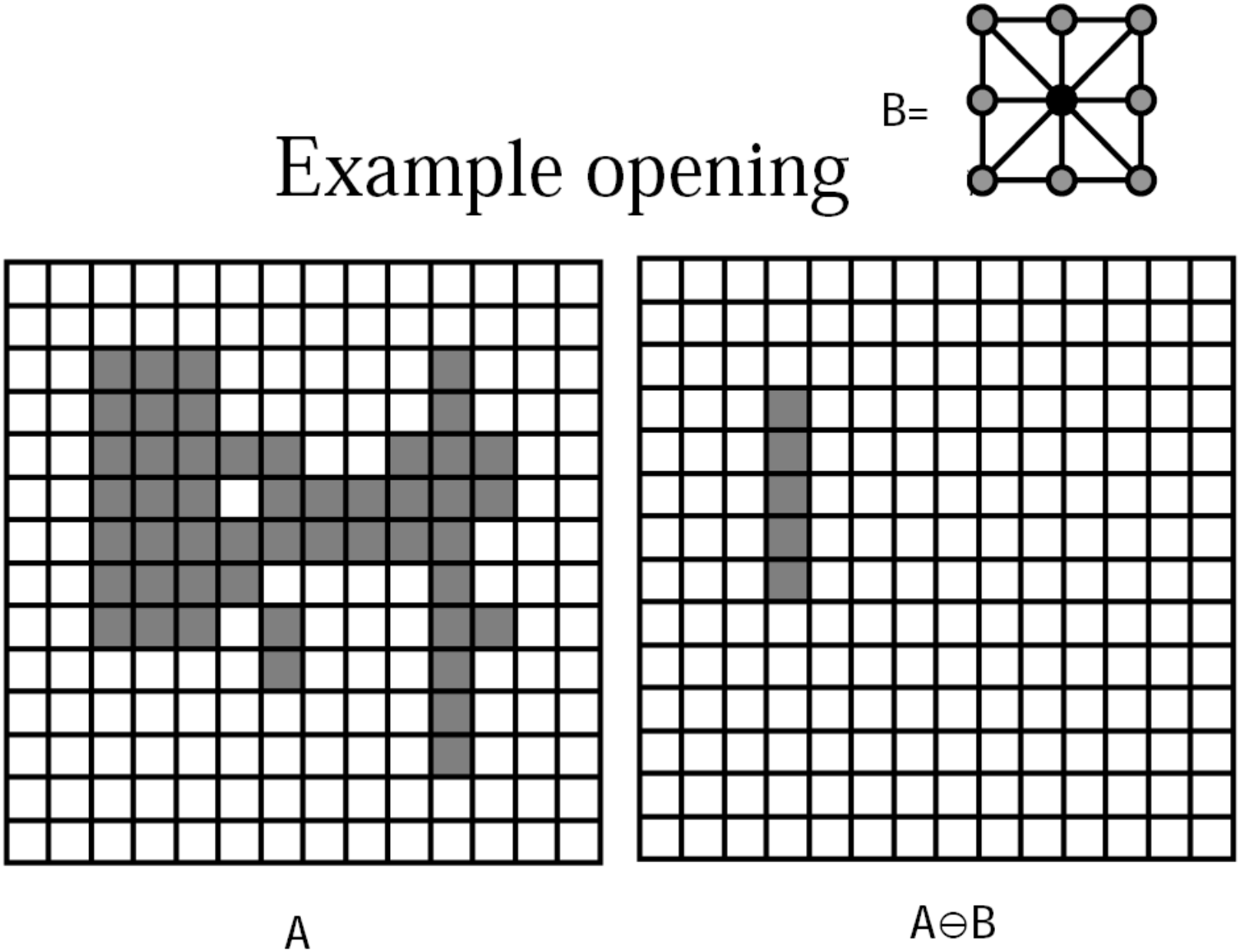


Erosion followed by dilation, denoted $\circ$.

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

- Eliminates protrusions.
- Break necks.
- Smooths contour.

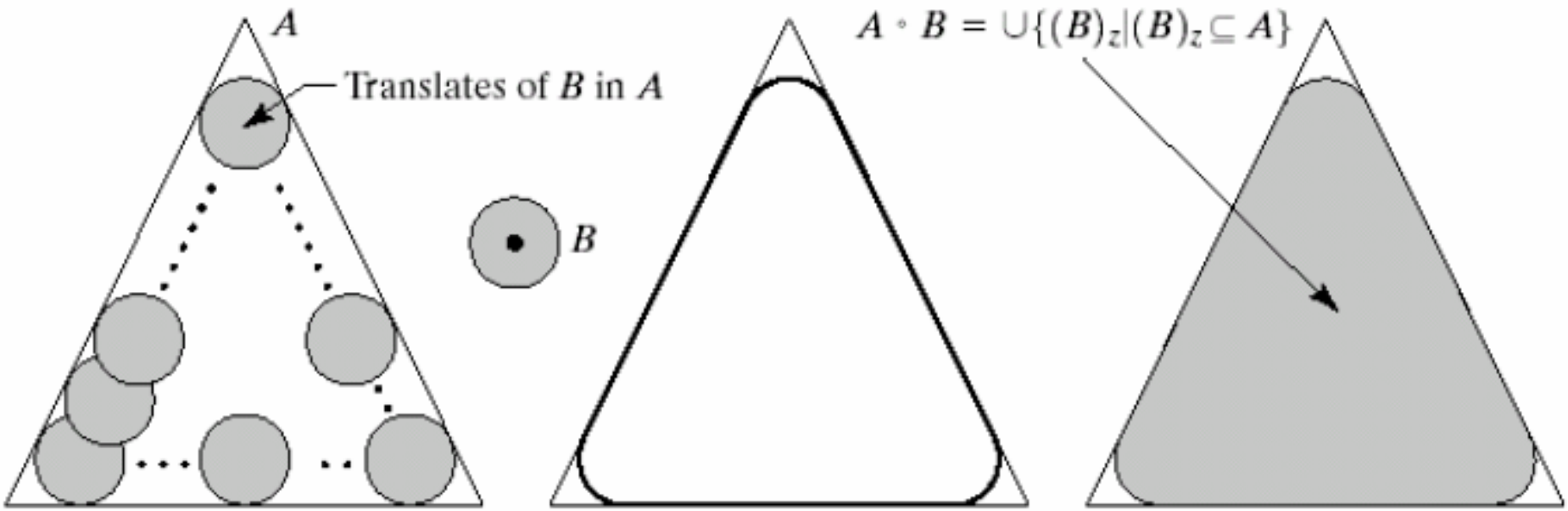
Example opening (fill in!)



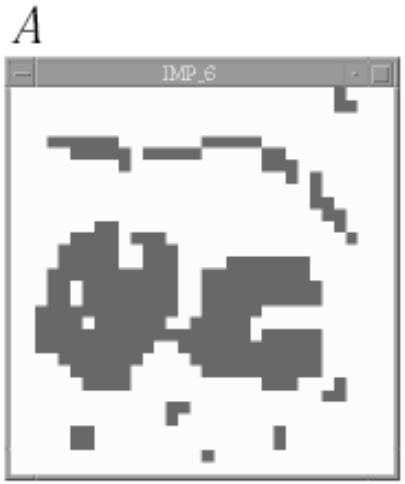
Opening: roll ball (=SE) inside object

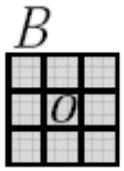
See B as a "rolling ball"

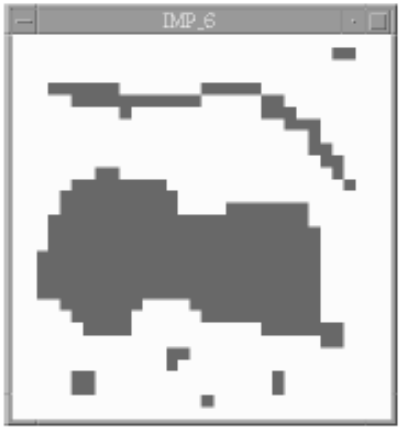
Boundary of $A \circ B$ are equal to points in B that reaches closest to the boundary A when B is rolled *inside* A.



• Closing





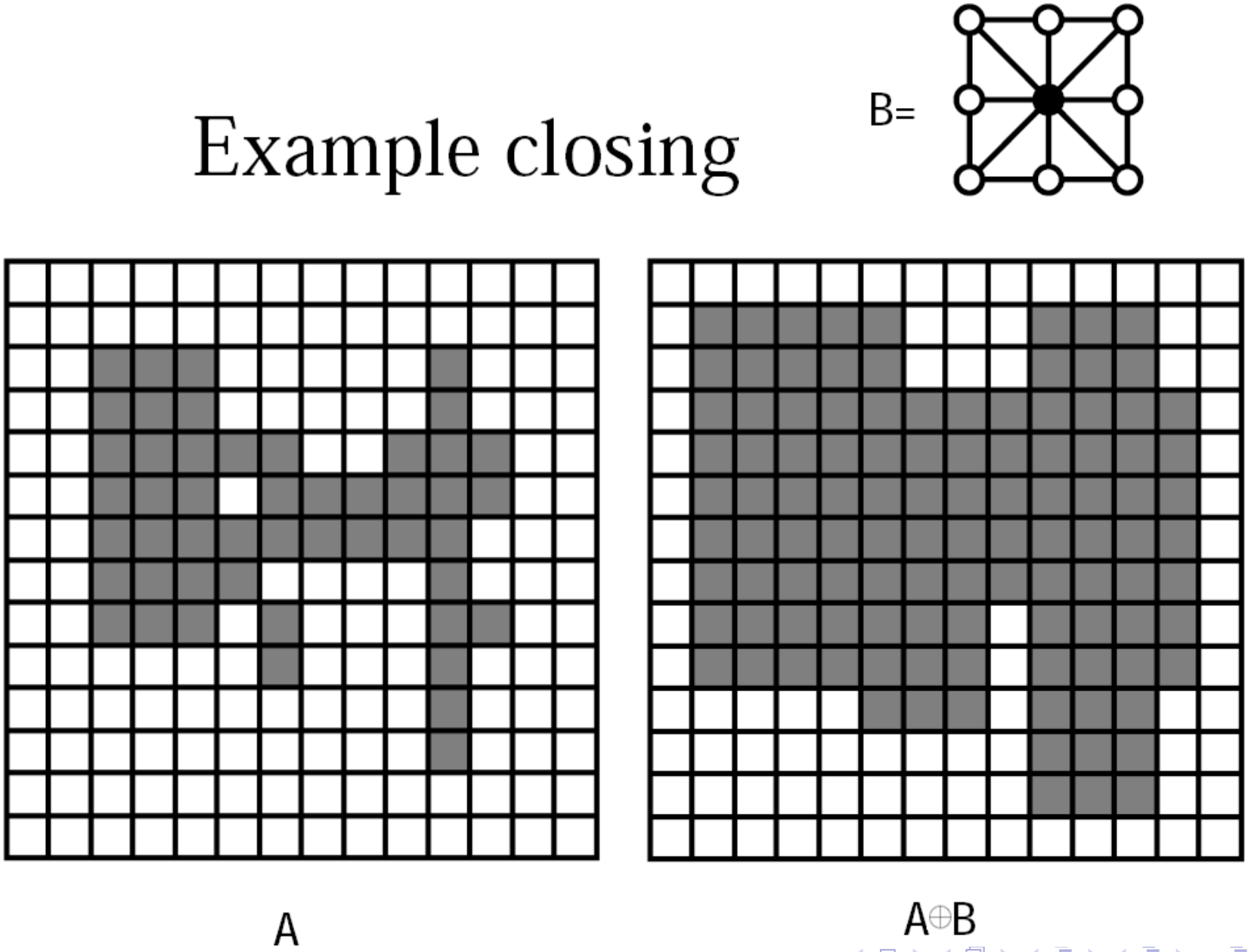


Dilation followed by erosion, denoted •.

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

- Smooth contour.
- Fuse narrow breaks and long thin gulfs.
- Eliminate small holes.
- Fill gaps in the contour.

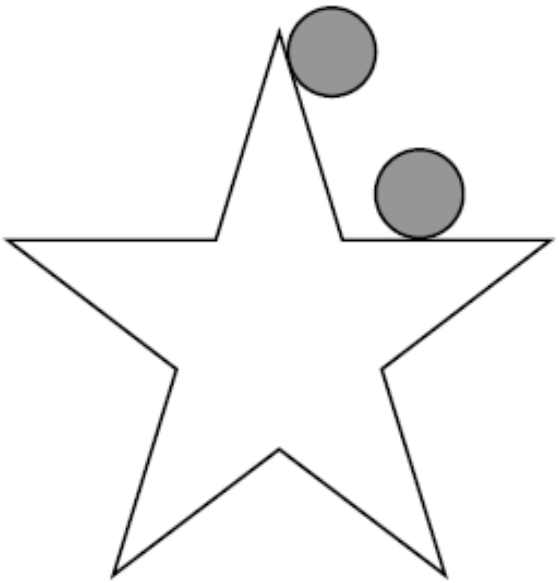
Example closing (fill in!)

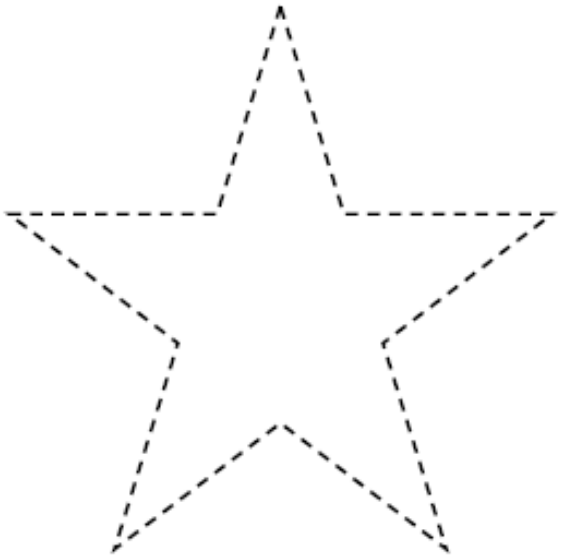


Closing: roll ball (=SE) outside object

(Fill in border after closing with ball as SE!)

Boundary of $A \bullet B$ are equal to points in B that reaches closest to the boundary of A when B is rolled *outside* A.





$\otimes$ hit-or-miss transformation ($\otimes$ or HMT)

Find location of one shape among a set of shapes ("template matching").

$$A \otimes B = (A \ominus B_1) \cap (A^C \ominus B_2)$$

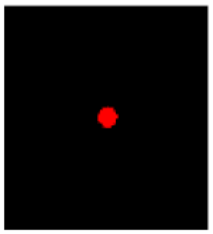
Composite SE: Object part (B_1) and background (B_2).

Does B_1 *fit the object* while, simultaneously, B_2 misses the object, i.e., *fit the background*.

Hit-or-miss transformation ($\otimes$ or HMT)

Find location of one shape among a set of shapes.

$$A \otimes B = (A \ominus X) \cap \left(A^C \ominus (W - X)\right)$$

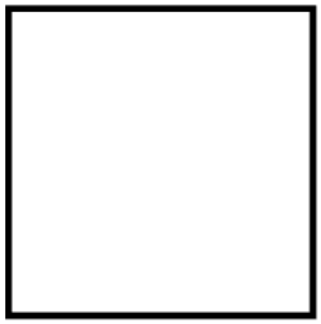


X

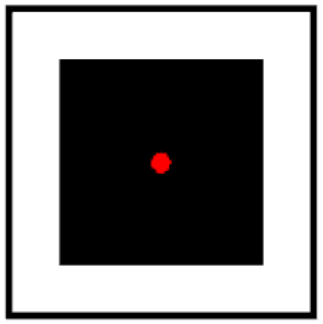
$B=(B_1,B_2)$

$B_1=X$

$B_2=W-X$



W



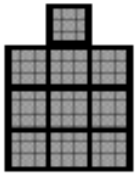
W-X

Alternative:

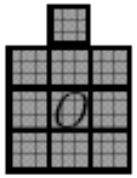
$$\begin{aligned} A \otimes B &= (A \ominus B_1) \cap (A^C \ominus B_2) \\ &= (A \ominus B_1) \cap (A \oplus \hat{B}_2)^C \\ &= (A \ominus B_1) - (A \oplus \hat{B}_2) \end{aligned}$$

Example hit-or-miss transform (fill in!)

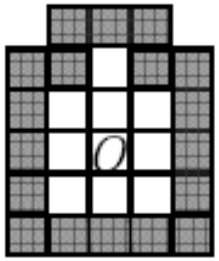
Search for:



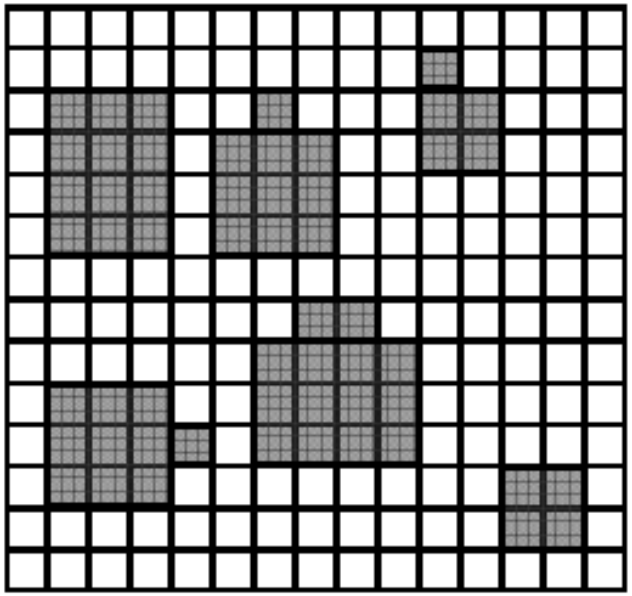
B_1



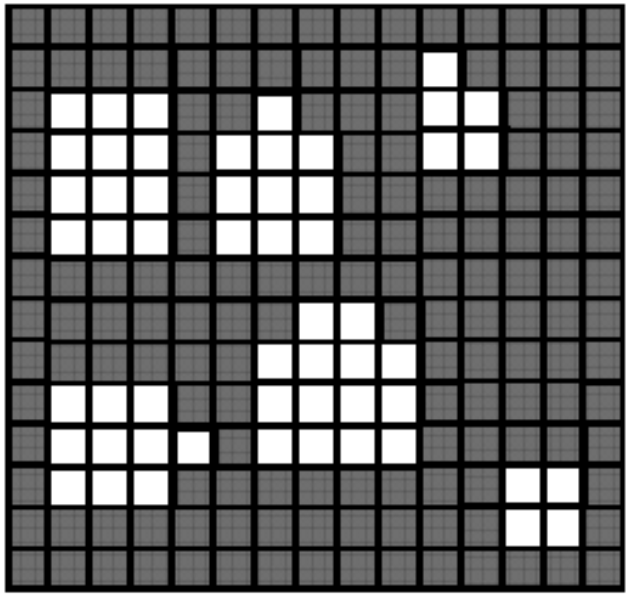
B_2



A



A^c



Basic morphological algorithms

Use erosion, dilation, opening, closing, hit-or-miss transform for

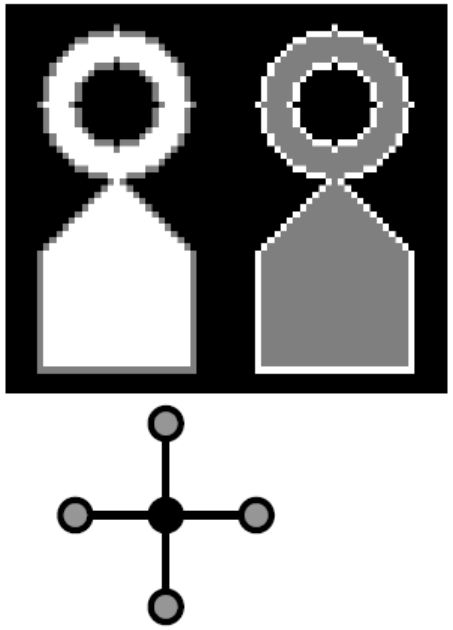
- Boundary extraction.
- Region filling.
- Extraction of connected components (labeling).
- Defining the convex hull.
- Defining the skeleton.

Boundary extraction

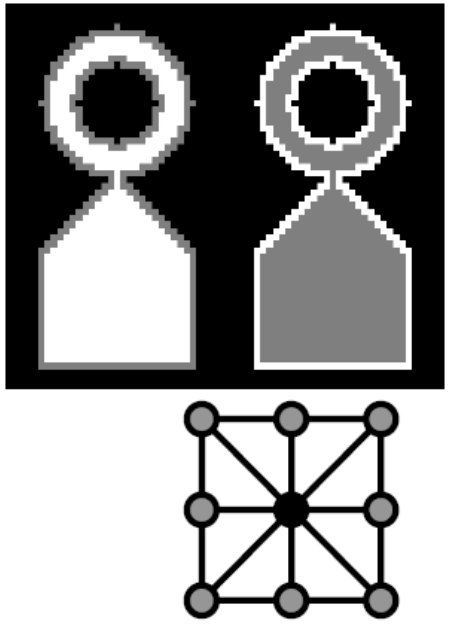
by erosion and set difference (boundary of $A = \beta(A)$)

Extract the boundary of:

$\beta(A) =$
$A - (A \ominus B)$.



8-connected boundary
$\beta(A) =$ pixels with edge neighbour in A^C.



4-connected boundary
$\beta(A) =$ pixels with edge or point neighbour in A^C.

"Morphological gradient"

Region filling

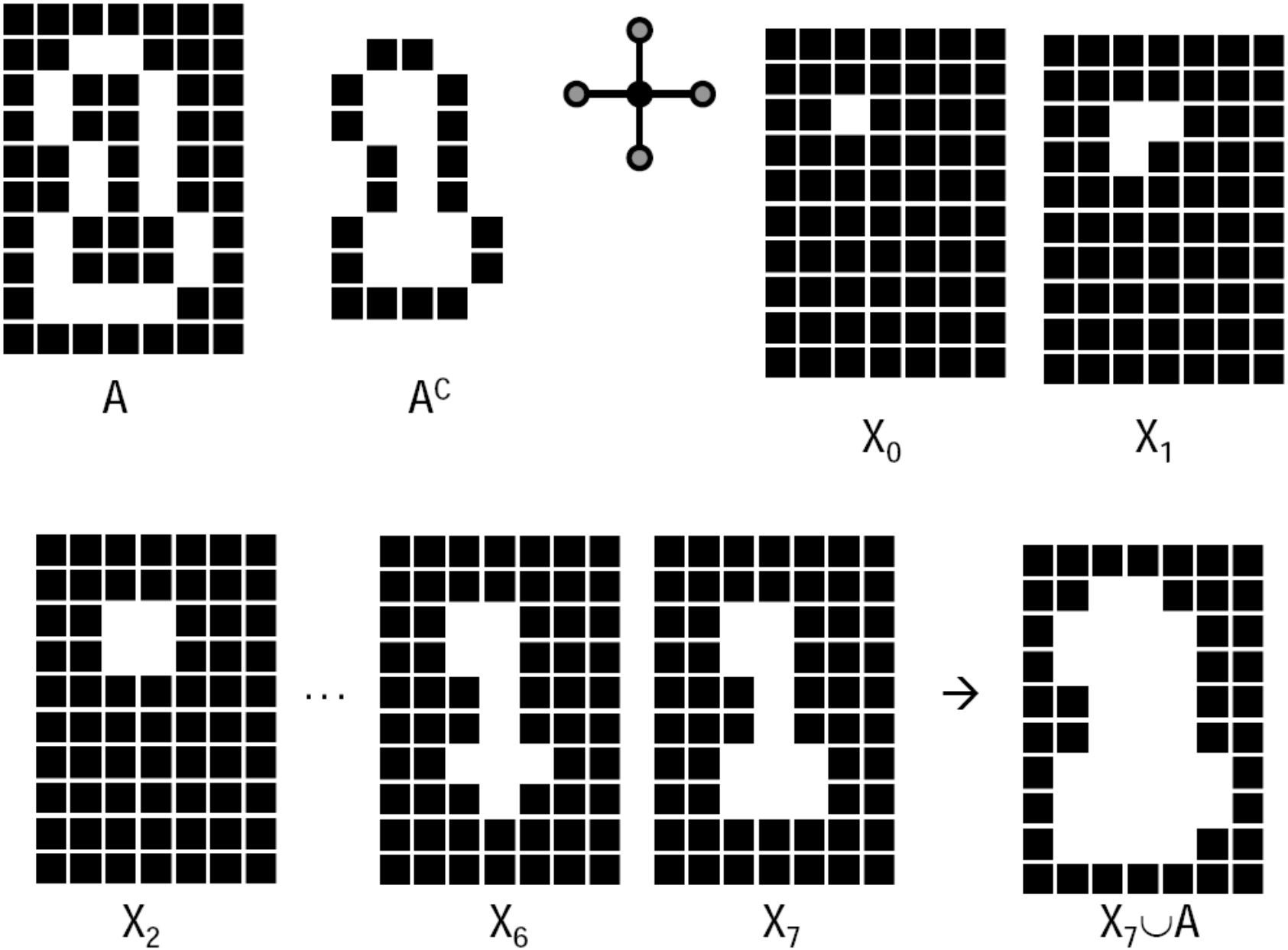
Fill a region A given its boundary $\beta(A)$.
$x = X_0$ is known and inside $\beta(A)$.

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

Continue until $X_k = X_{k-1}$.
Filled region $A \cup X_k$.

Use to fill holes! *Conditional dilation!*

Example of region filling



Compare with removing holes using two-pass labeling algorithm

See segmentation lecture

Connected component labeling

- Label the inverse image.
- Remove connected components touching the image border.
- Output = holes + original image.

$\longrightarrow$ 2 scans + 1 scan (straight forward...)

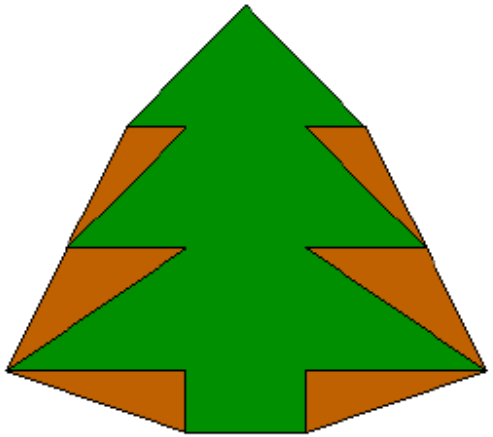
Mathematical morphology

- Iterate: dilation, set intersection

$\longrightarrow$ Dependent on size and shape of the hole needed: *initialization!*

Convex hull

- Region R is convex if
  - For any points $x_1, x_2 \in R$, straight line between x_1 and x_2 is in R.
- Convex hull H of a region R
  - Smallest convex set containing R.
- Convex deficiency $D = H - R$.



Convex hull (morphological algorithm)

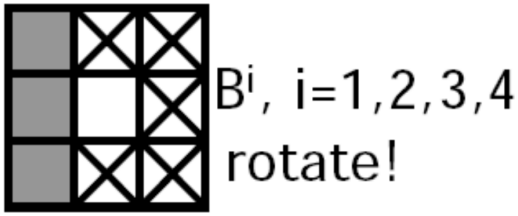
Algorithm for computing the convex hull $CH(A)$:

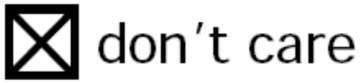
$$X_k^i = (X_{k-1} \otimes B^i) \cup A, \quad i = 1, 2, 3, 4, \quad k = 1, 2, 3, \ldots$$

$$X_0^i = A$$

Converges to $D^i(X_k = X_{k-1})$.

$$CH(A) = \bigcup_{i=1}^{4} D^i$$





Convex hull (morphological algorithm) - example

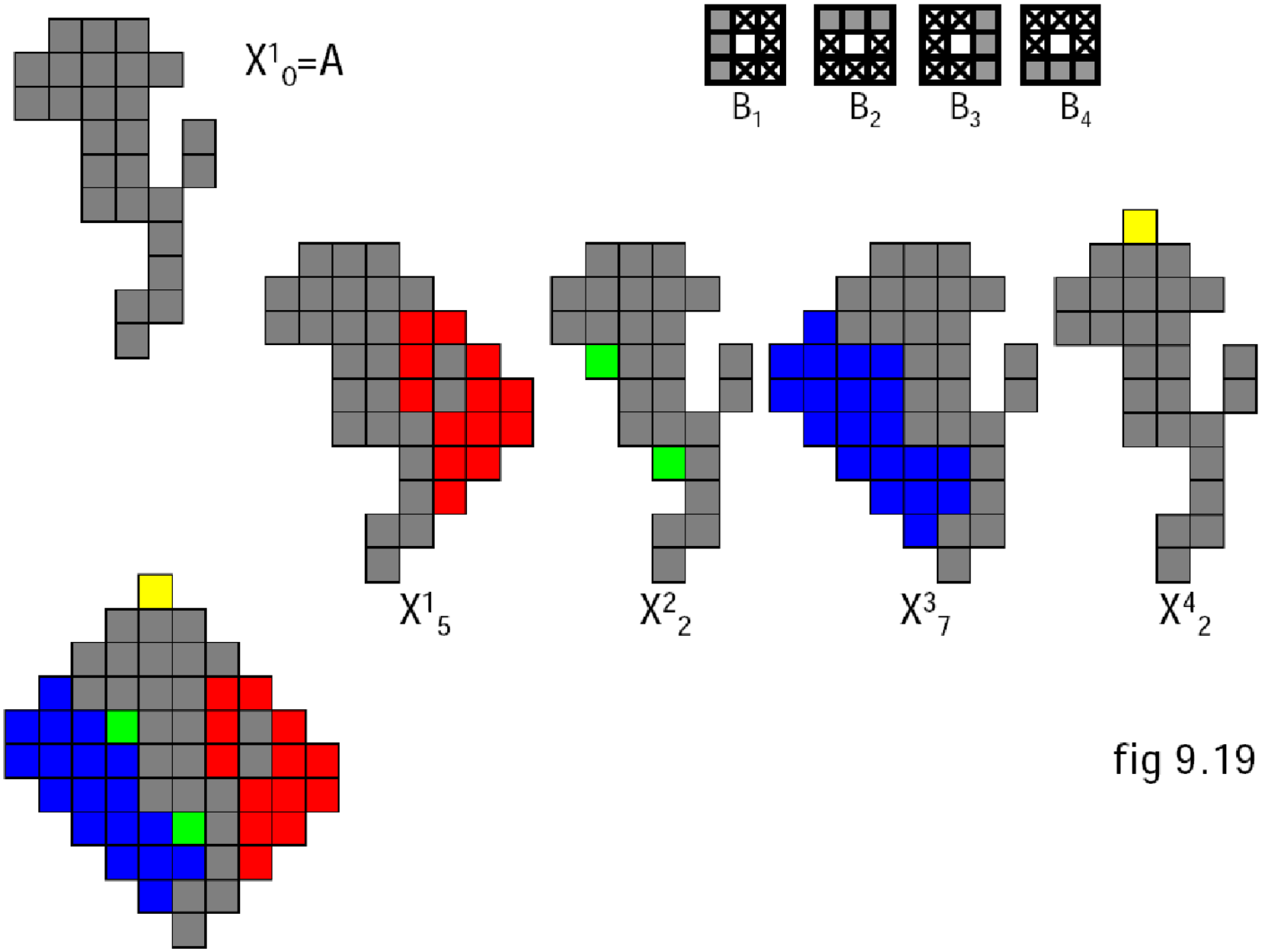
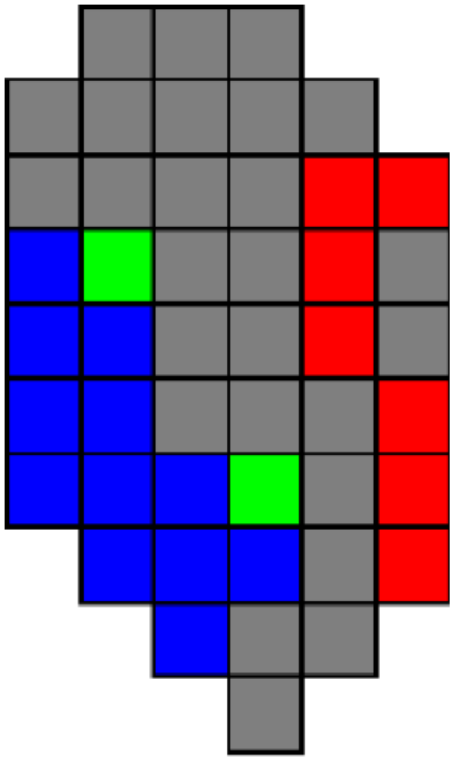


fig 9.19

Convex hull (morphological algorithm) - example



- The growth of the convex hull is limited to the maximum dimensions of the original set of points along the vertical and horizontal directions.

Distance transforms

Input: Binary image.
Output: In each object (background) pixel, write the distance to the closest background (object) pixel.

Definition

A function D is a metric (distance measure) for the pixels p, q, and z if

a $D(p, q) \geq 0$

b $D(p, q) = 0$ iff $p = q$

c $D(p, q) = D(q, p)$

d $D(p, z) \leq D(p, q) + D(q, z)$

Different metrics

Minkowski distances

Euclidean $D_E(p, q) = \sqrt{(\Delta x)^2 + (\Delta y)^2}$.
City block $D_4(p, q) = |\Delta x| + |\Delta y|$.
Chess-board $D_8(p, q) = \max(|\Delta x|, |\Delta y|)$.

Chess-board mask:

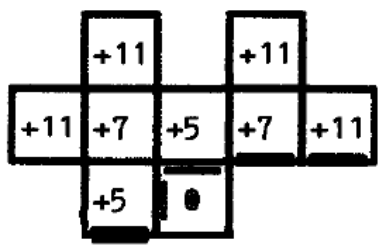
1	1	1
1	p	

Weighted measures

4	3	4
3	p	

Chamfer(3-4) since $4/3 \approx 1.33$ is close to $\sqrt{2}$ and fulfills other criteria.
If distance between two 4-adjacent is said to be 3, then the distance between m-adjacent pixels should be 4.

Chamfer(5-7-11) is even better measure.



Algorithm for distance transformation

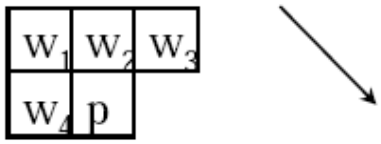
Distance from each object pixel to the closest background pixel

p current pixel

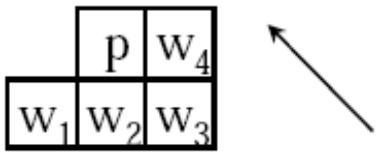
$g_1 - g_4$ neighboring pixels

$w_1 - w_4$ weights (according to choice of metric)

1. Set background pixels to zero and object pixels to infinity (or maximum intensity, e.g., 255).
2. Forward pass, from $(0,\ 0)$ to $(\max(x), \max(y))$:
   if $p > 0$, $p = \min(g_i + w_i)$, $i = 1,\ 2,\ 3,\ 4$.



3. Backward pass, from $(\max(x), \max(y))$ to $(0,\ 0)$:
   If $p > 0$, $p = \min(p,\ \min(g_i + w_i))$, $i = 1,\ 2,\ 3,\ 4$.



Chamfer $(3-4)$ distance

Binary original image

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

Chamfer $(3-4)$ distance

1. Starting image

0	0	0	0	0	0	0	0
0	∞	0	0	∞	0	0	0
0	∞	∞	0	∞	0	0	0
0	∞	∞	∞	∞	∞	∞	0
0	∞	∞	∞	∞	∞	∞	0
0	∞	∞	∞	∞	∞	∞	0
0	0	0	∞	∞	∞	0	0
0	0	0	0	0	0	0	0

Chamfer $(3-4)$ distance

2. First pass from top left down to bottom right

0	0	0	0	0	0	0	0
0	3	0	0	3	0	0	0
0	3	3	0	3	0	0	0
0	3	4	3	4	3	3	0
0	3	6	6	7	6	4	0
0	3	6	9	10	8	4	0
0	0	0	3	6	8	0	0
0	0	0	0	0	0	0	0

Chamfer $(3-4)$ distance

3. Second pass from bottom right down to top left

0	0	0	0	0	0	0	0
0	3	0	0	3	0	0	0
0	3	3	0	3	0	0	0
0	3	4	3	4	3	3	0
0	3	6	6	7	6	3	0
0	3	3	4	6	4	3	0
0	0	0	3	3	3	0	0
0	0	0	0	0	0	0	0

Applications using the distance transform (DT)

I. Find the shortest path between two points a and b.
   1. Generate the DT with a as the object.
   2. Go from b in the steepest gradient direction.

II. Find the radius of a round object
   1. Generate the DT of the object.
   2. The maximum value equals the radius.

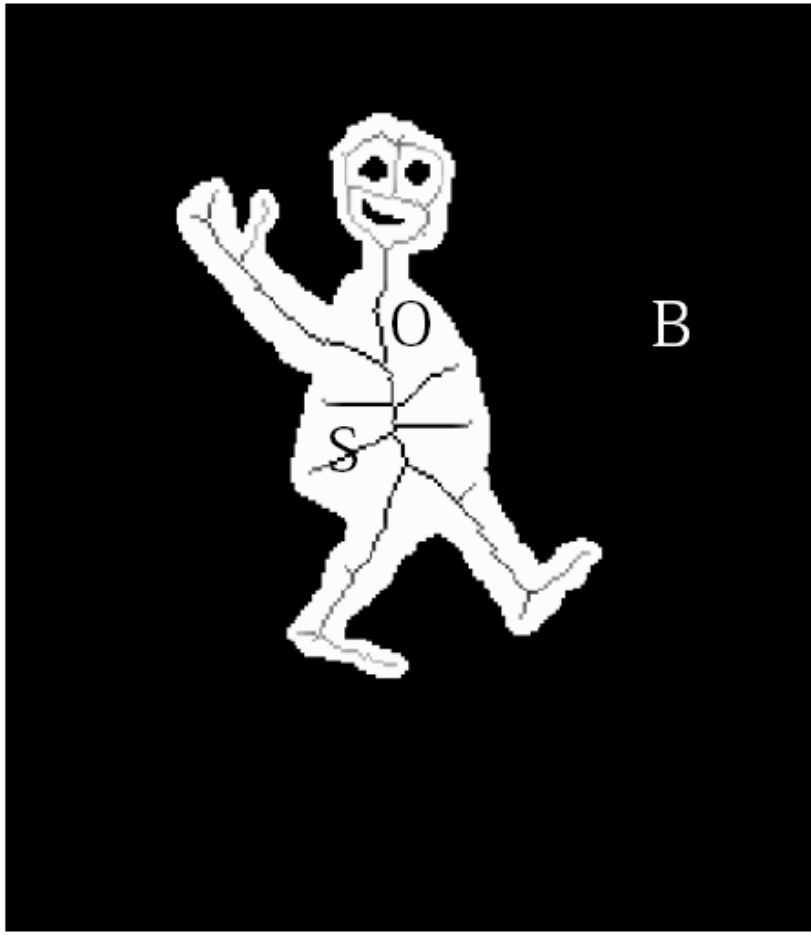
- See segmentation using watershed algorithm in $L05$!

Applications using the distance transform (DT)

III . Skeletons

Definitions: If O is the object, B is the background, and S is the skeleton, then

- S is topological equivalent to O
- S is centered in O
- S is one pixel wide (difficult!)
- O can be reconstructed from S



Skeletons (Centers of Maximal Discs)

A disc is made of all pixels that are within a given radius r. A disc in an object is *maximal* if it is not covered by any other disc in the object. A reversible representation of an object is the set of centers of maximal discs.

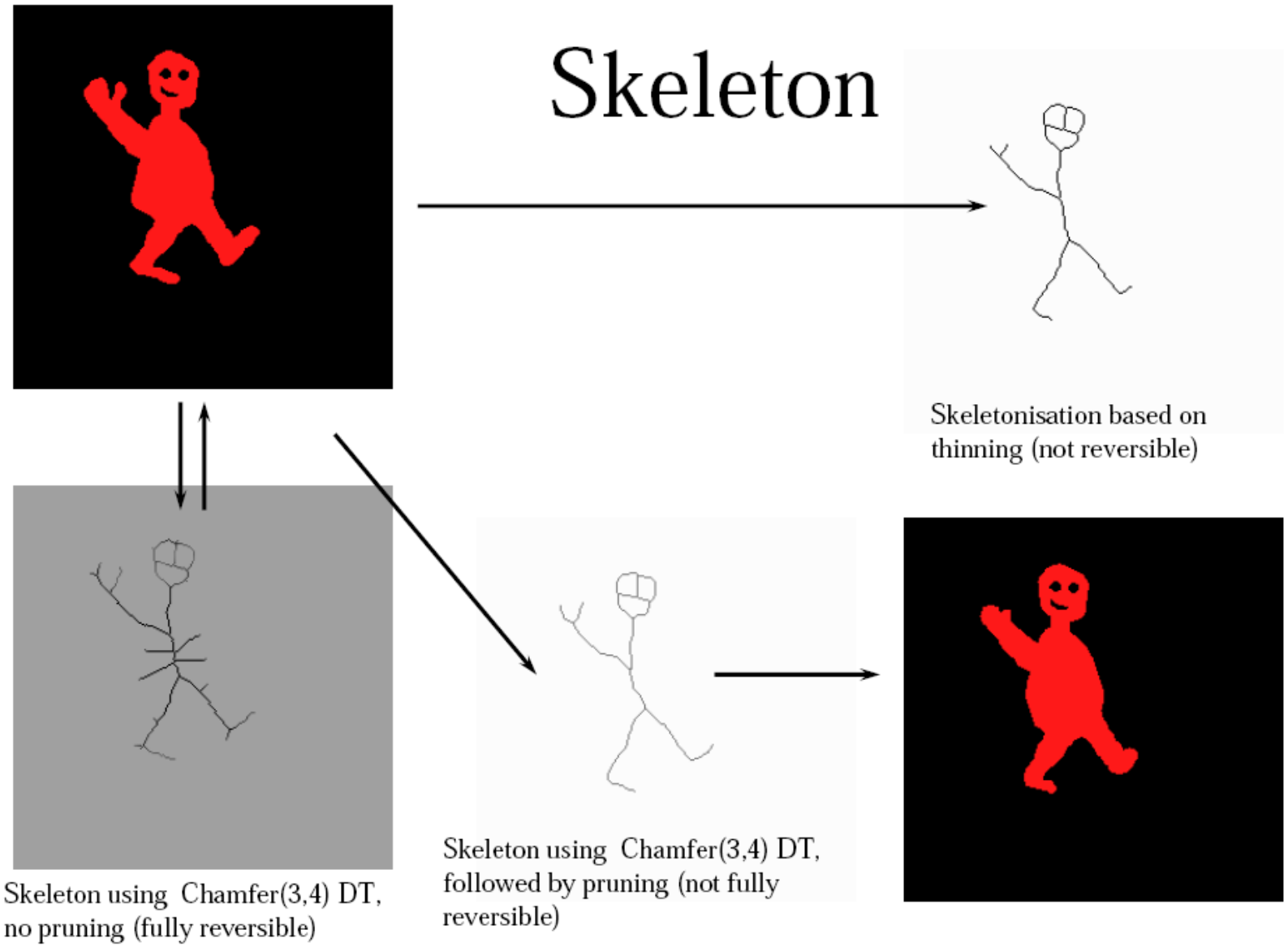
Algorithm

Find the skeleton with Centers of Maximal Discs (CMD)
Completely reversible situation

1. Generate distance transform of object
2. Identify CMDs (smallest set of maxima)
3. Link CMDs

"Pruning" is to remove small branches (no longer fully reversible.)

Skeleton


Skeleton
Skeletonisation based on thinning (not reversible)
Skeleton using Chamfer(3,4) DT, no pruning (fully reversible)
Skeleton using Chamfer(3,4) DT, followed by pruning (not fully reversible)