BMB5113
COMPUTER VISION

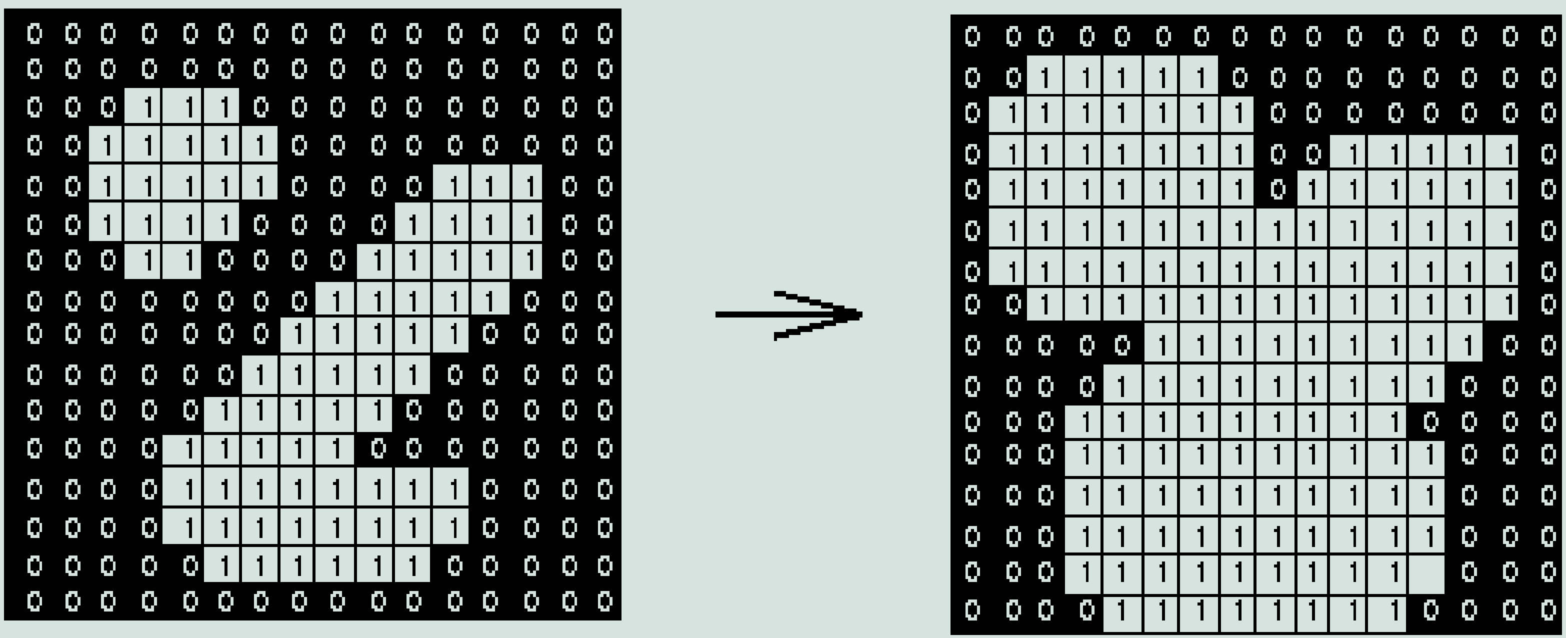
MORPHOLOGICAL OPERATIONS

Binary Morphology

- Treat an object within a binary image as the set of ‘1’s
  - set A
- Instead of a kernel window use a “structuring element”
  - set B
- Define the following operations based on set intersection, union, difference:
- Dilation: $A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \emptyset\}$
- Erosion: $A \ominus B = \{z \mid (B)_z \cap A^c = \emptyset\}$

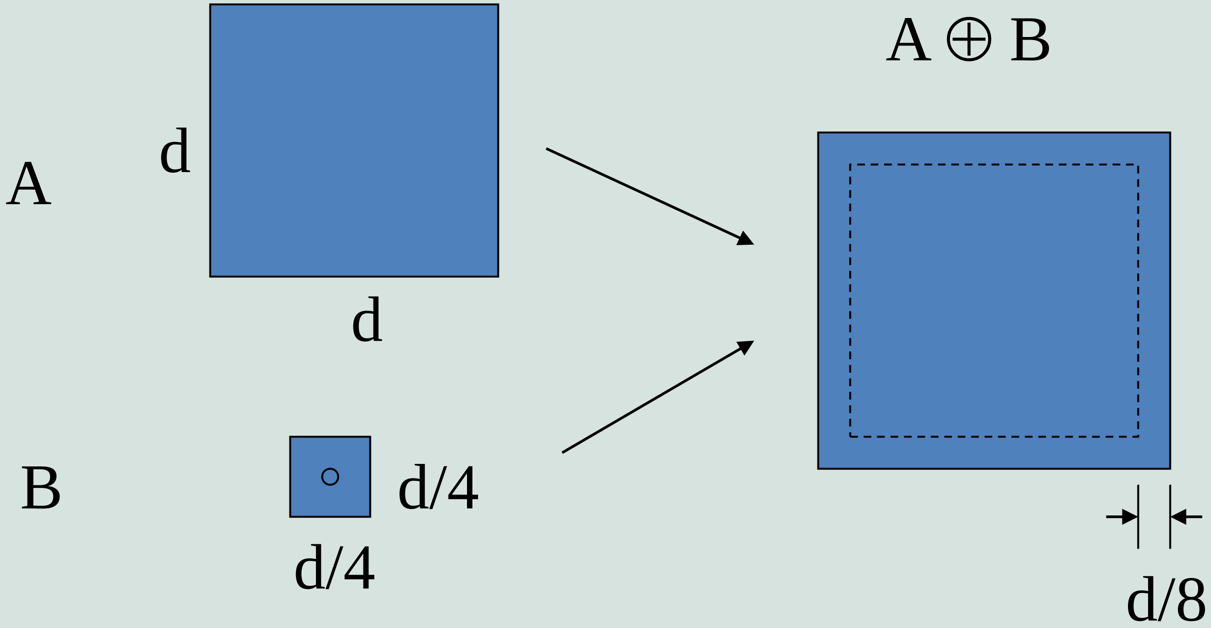
Dilation

- Dilation: $A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \emptyset\}$
- Example: 3 x 3 square structuring element



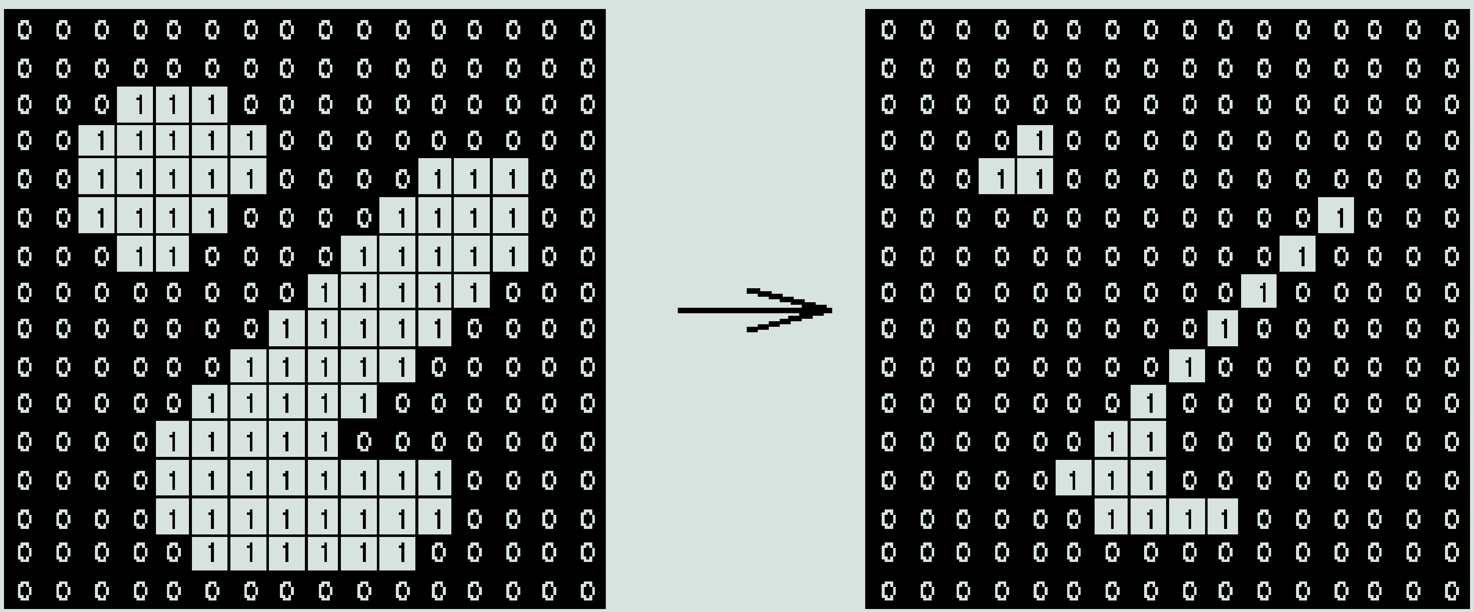
Dilation

- Example of dilation



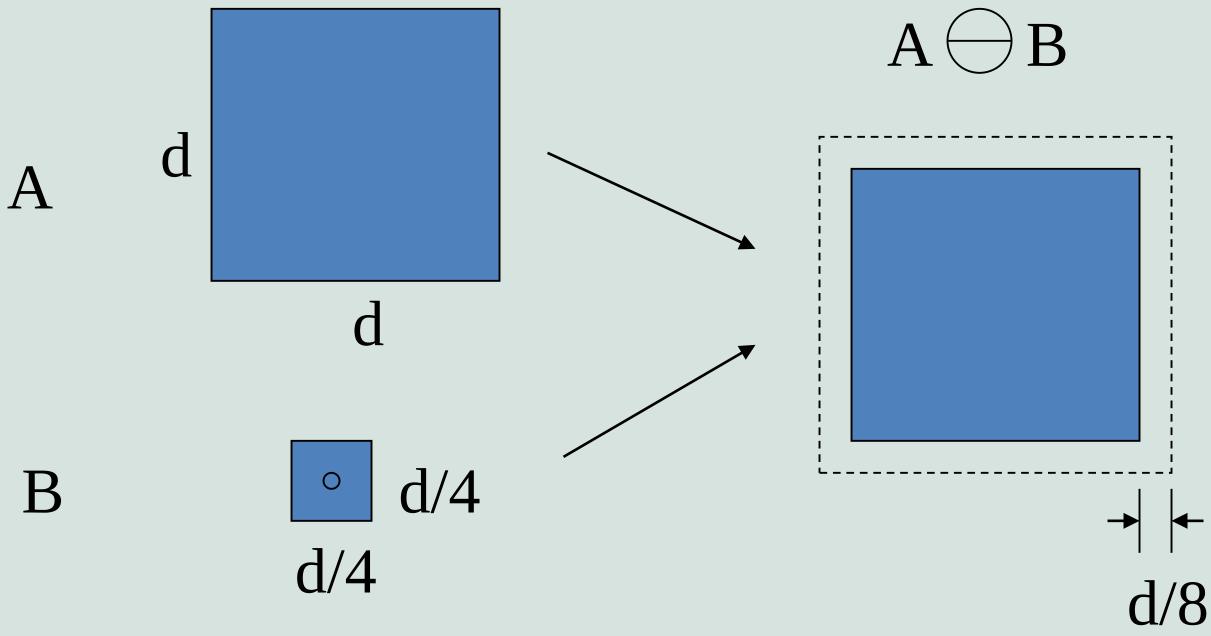
Erosion

- Erosion: $A \ominus B = \{z \mid (B)_z \cap A^c = \emptyset\}$
- Example: 3 x 3 square structuring element



Erosion

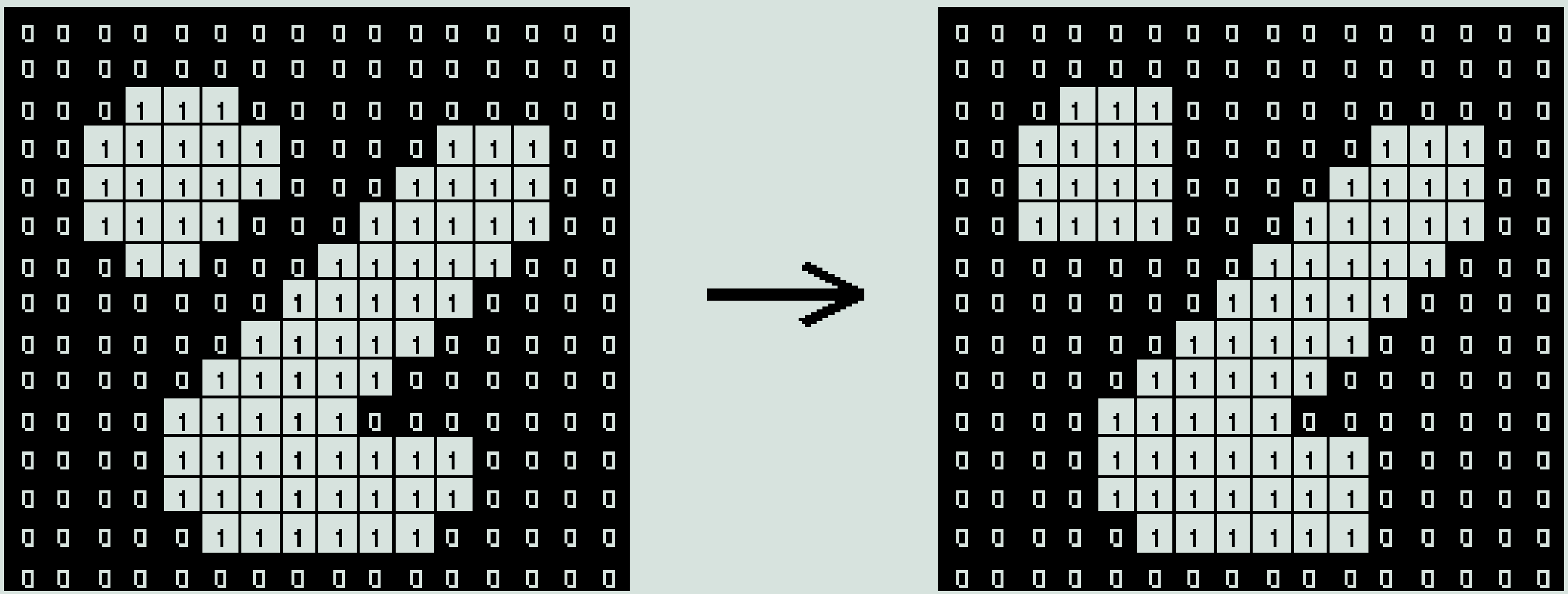
- Example of erosion



Opening

- Erosion followed by dilation

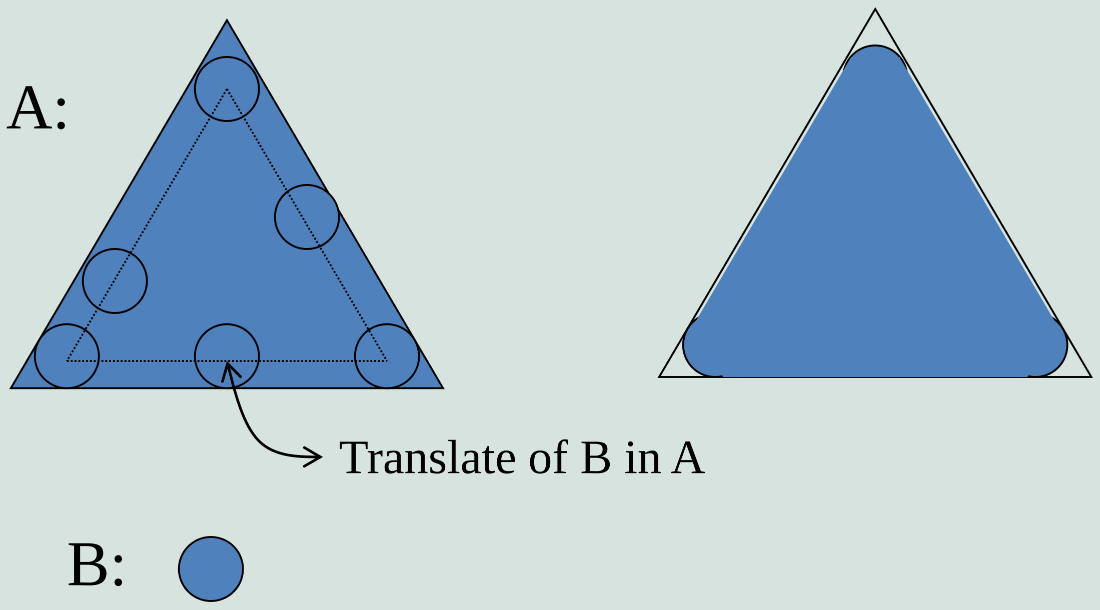
$$A \circ B = (A \ominus B) \oplus B$$



- Fit the structuring element inside the object

Opening

- Example of opening



Closing

- Dilation followed by erosion

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

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

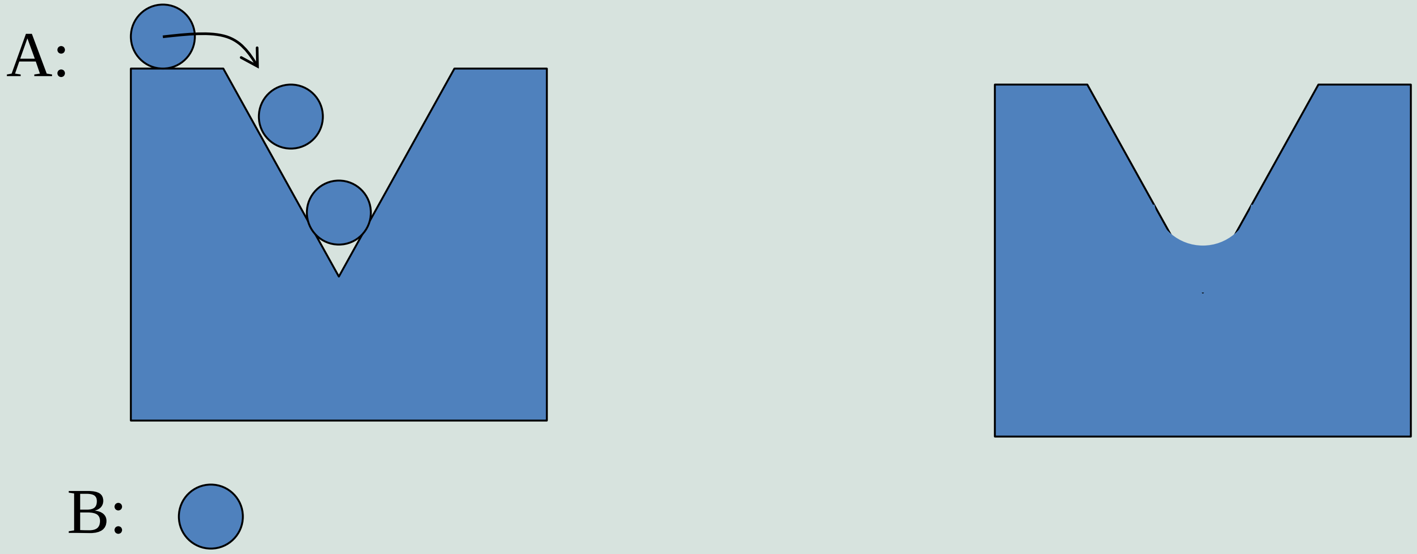
$\longrightarrow$

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

- Fit the structuring element in the background

Closing

- Example of closing



Python Functions

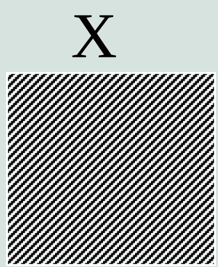
```
from scipy.ndimage import measurements, morphology
# load image and threshold to make sure it is binary
im = array(Image.open('houses.png').convert('L'))
im = 1*(im<128)
labels, nbr_objects = measurements.label(im)
print "Number of objects:", nbr_objects

# morphology - opening to separate objects better
im_open = morphology.binary_opening(im,ones((9,5)),iterations=2)
labels_open, nbr_objects_open = measurements.label(im_open)
print "Number of objects:", nbr_objects_open
```

Other Morphological Operations

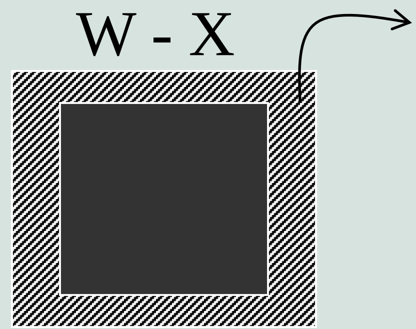
- Hit-or-miss transform
- Thinning
- Thickening
- Skeletonization

Hit-or-Miss Transform

- Shape detection
- Using two structure elements



Structure element I



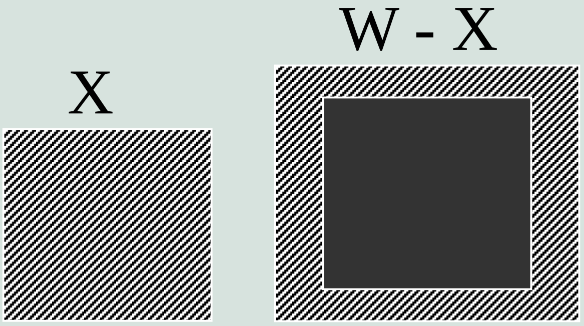
Structure element II:
complement of X with respect to W

Hit-or-Miss Transform

- The match (or fit) of B in A is called hit-or-miss transform,
  - denoted $A \circledast B$
  - B is composed of X (object) and (W-X) (background)

$$A \circledast B = (A \ominus X) \cap [A^c \ominus (W\text{-}X)]$$

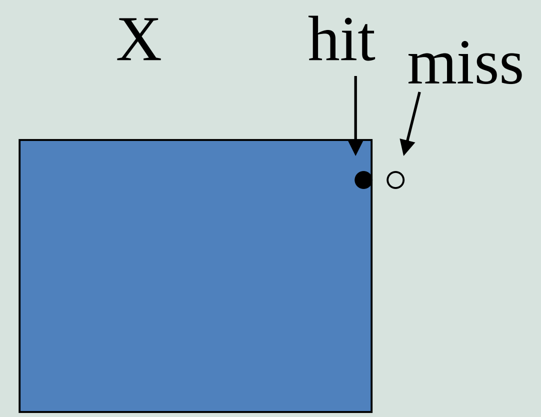
- This set contains all the (origin) points, at which, X found a match (hit) in A and (W-X) found a match in A^c (miss), simultaneously.

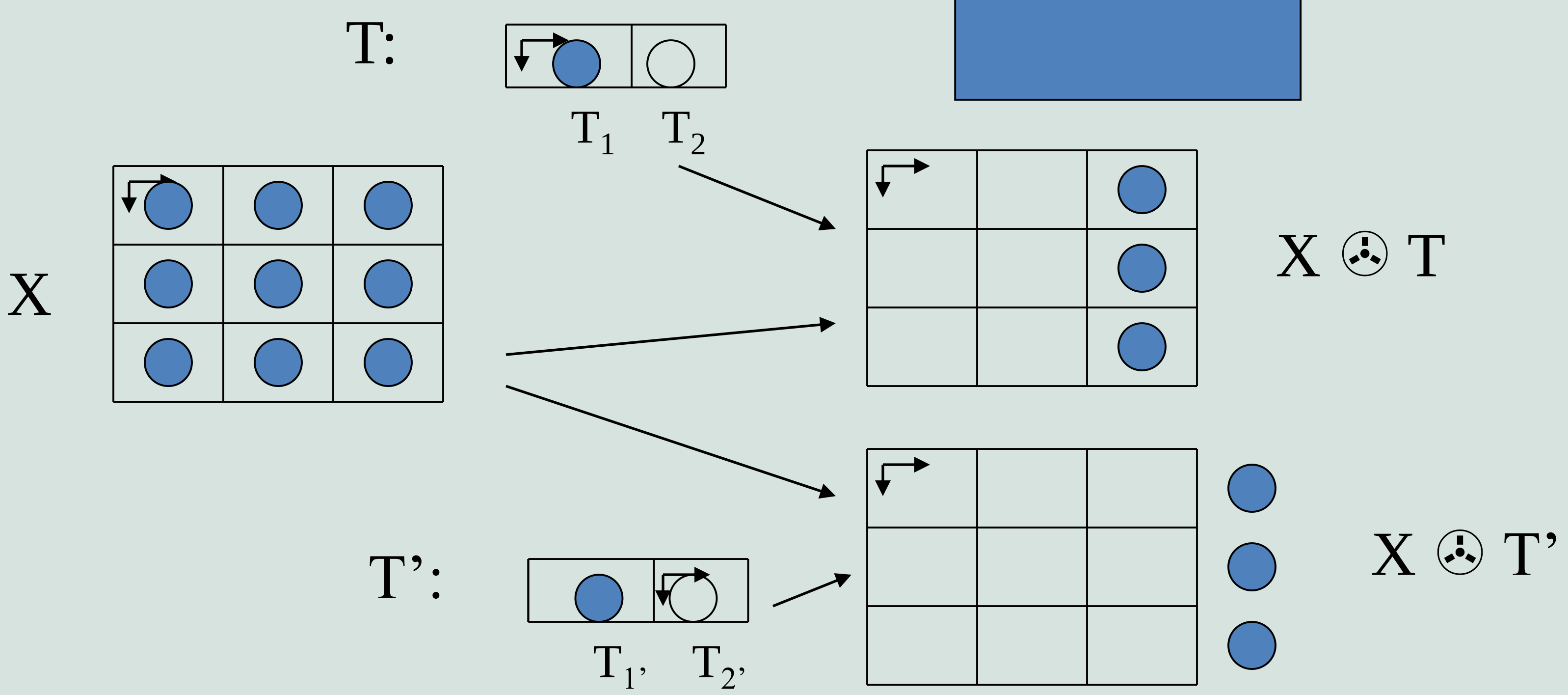


Hit-or-Miss Transform

- Example of hit-or-miss

$X \circledast T = \{ X \mid T_{1x} \subset X, T_{2x} \subset X^c \}$





Other Applications

- Boundary extraction

$$\text{Boundary}(A) = A - (A \ominus B)$$

- Region filling
  - given a set A which defines a region boundary
  - start with a non-boundary point P within the region
  - let $X_0 = P$
  - $X_k = (X_{k-1} \oplus B) \cap A^c, \quad k = 1,2,3,\ldots$
  - iterate increasing the value of k by 1 for each step
  - terminate if $X_k = X_{k-1}$

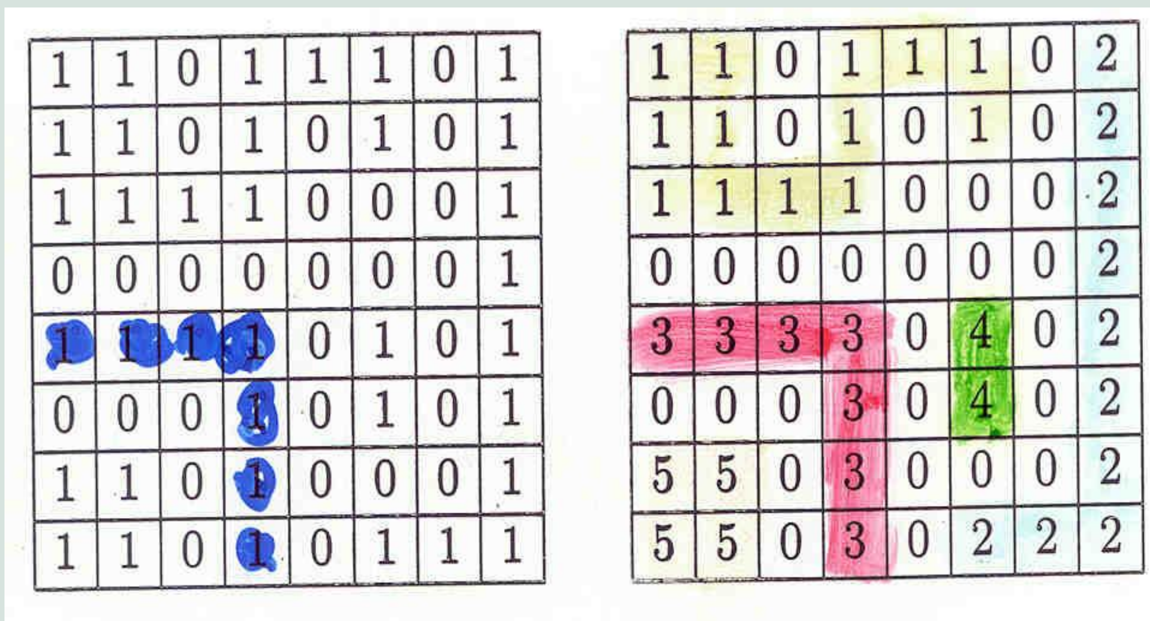
Note: $A \cup X_k$ includes the filled set and the boundary

Connected Components

- Connected component extraction
  - similar to the region filling
  - start with a point P which is contained in A
  - let $X_0 = P$
  - $X_k = (X_{k-1} \oplus B) \cap A, \quad k = 1,2,3,\ldots$
  - iterate increasing the value of k by 1 for each step
  - terminate if $X_k = X_{k-1}$

Connected Components: Object Coloring

- Each object is a connected set of pixels
- Object label is “color”
- How is this done?



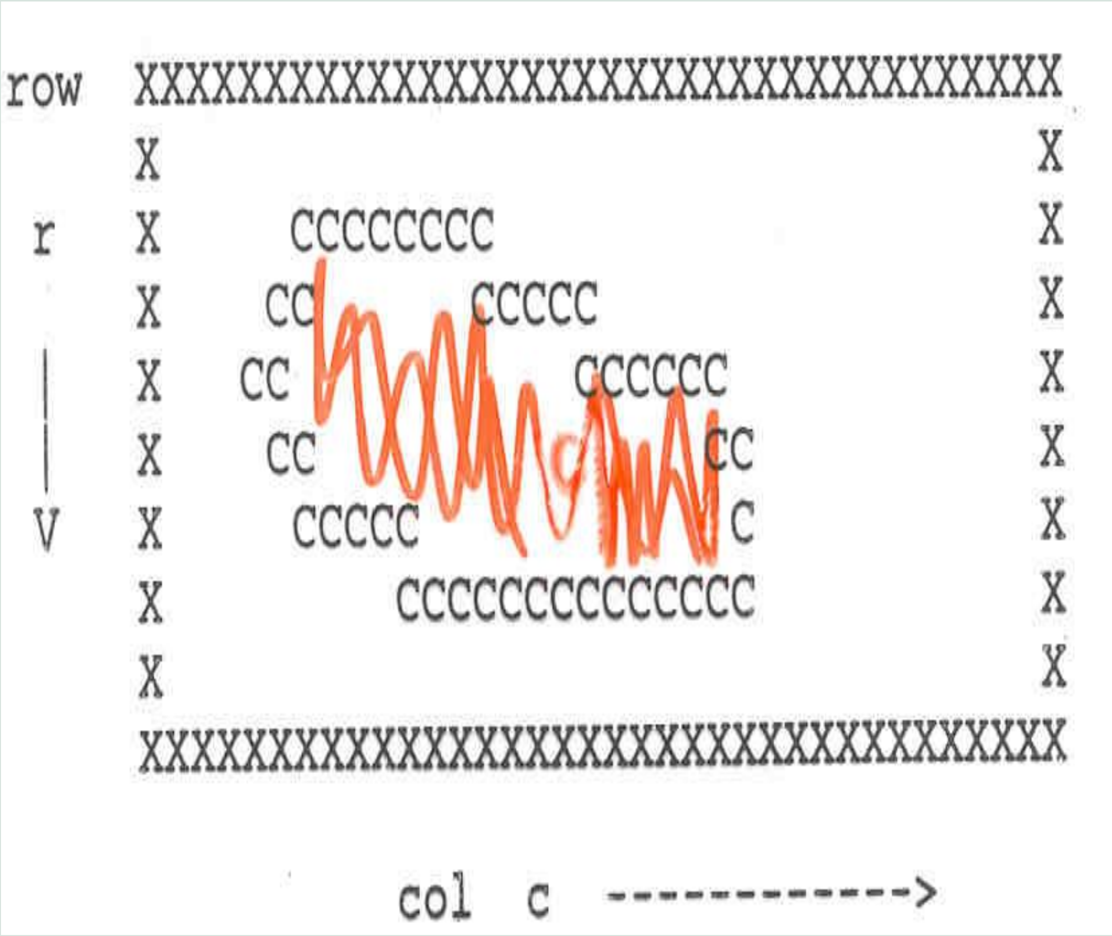
a) binary image   b) 5 components

Extracting Components

- Collect foreground pixels into separate objects – label pixels with same color
- Can then compute many features from each set of pixels
    A. collect by “random access” of pixels using “paint” or “fill” algorithm
    B. collect by “raster” (row-by-row) scanning all pixels

Paint/Fill Algorithm

- Object region must be bounded by background
- Start at any pixel [r,c] inside object
- Recursively color neighbors



Paint/Fill Major Functions

- Raster scan until object pixel found
- Assign new color for new object
- Search through all connected neighbors until the entire object is labeled
- Return to the raster scan to search for another object pixel

Events of Paint/Fill Algorithm

- PP denotes “processing point”
  - If PP outside image, return to prior PP
  - If PP already labeled, return to prior PP
  - If PP is background pixel, return to prior PP
  - If PP is unlabeled object pixel, then

    1) label PP with current color code

    2) recursively label neighbors N1, …, N8 (or N1, …, N4)

Connected Components Labeling

- Recursive algorithm
  - For each object pixel {assign label L

    recursively assign label L to all neighbors

    stop if there are no more unlabeled 1's }
- Sequential algorithm
  - Scan the image left to right, top to bottom
  - For each pixel that is 1 {

    if only one of **upper** or **left** has a label, copy it

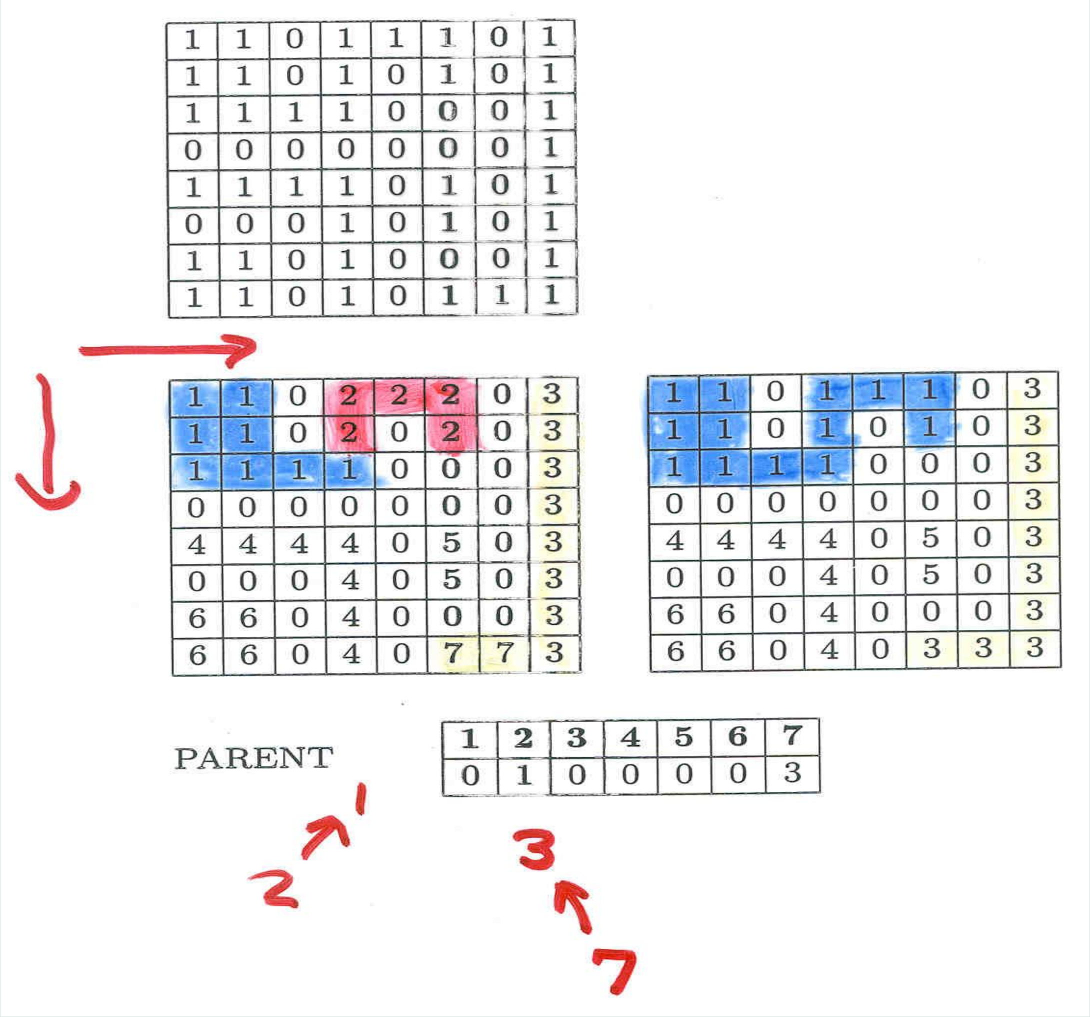
    if both have the same label, copy it

    if different, copy **lower** label and enter labels as equivalent

    otherwise, assign a new label }
  - Find the lowest label for each equivalence set; relabel entries

Merging Connecting Regions

- Detect and record merges while raster scanning.
- Use equivalence table to recode



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

1	1	0	2	2	2	0	3
1	1	0	2	0	2	0	3
1	1	1	1	0	0	0	3
0	0	0	0	0	0	0	3
4	4	4	4	0	5	0	3
0	0	0	4	0	5	0	3
6	6	0	4	0	0	0	3
6	6	0	4	0	7	7	3

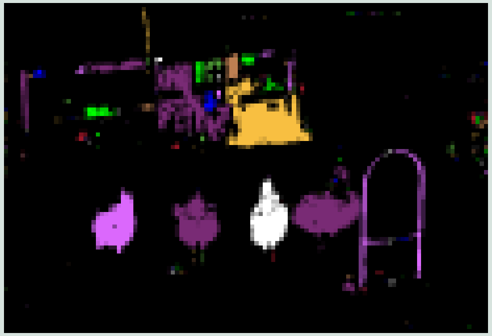
1	1	0	1	1	1	0	3
1	1	0	1	0	1	0	3
1	1	1	1	0	0	0	3
0	0	0	0	0	0	0	3
4	4	4	4	0	5	0	3
0	0	0	4	0	5	0	3
6	6	0	4	0	0	0	3
6	6	0	4	0	3	3	3

PARENT

1	2	3	4	5	6	7
0	1	0	0	0	0	3

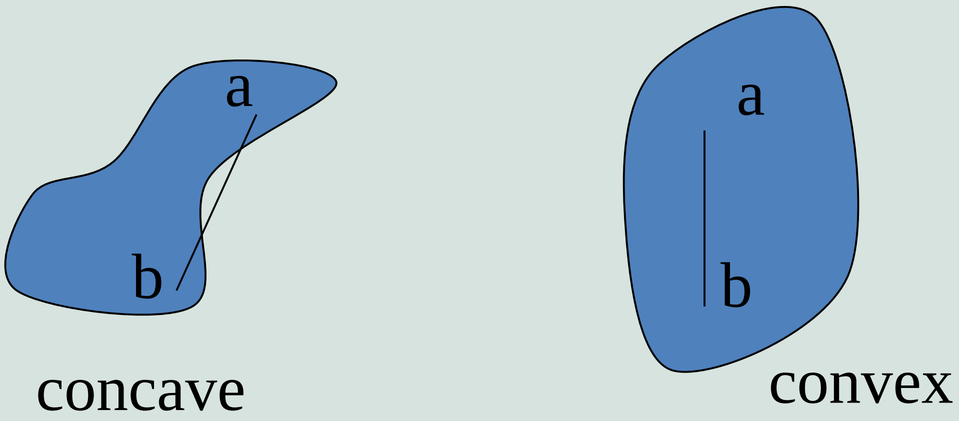
Example Revisited

- Find the turkeys in the picture

Convex Hull

- Convex hull extraction
  - set A is convex if any line $ab \subset A$ $(a \in A,\ b \in A)$



  - H is a convex hull if an arbitrary set S is the smallest convex set which contains A

Convex Hull

- Convex hull extraction algorithm:

  Given a set A and four structure elements B^1, B^2, B^3, B^4

  calculate the convex hull region: $C(A) = D^1 \cup D^2 \cup D^3 \cup D^4$

  where:

  D^i is derived from: $X^i_k = (X^i_{k-1} \circledast B^i) \cup A$

  $(i=1,2,3,4), (k=1,2,\ldots)$

  $D^i = X^i_k$ when $X^i_k = X^i_{k-1}$

  Initial $X^i_0 = A$

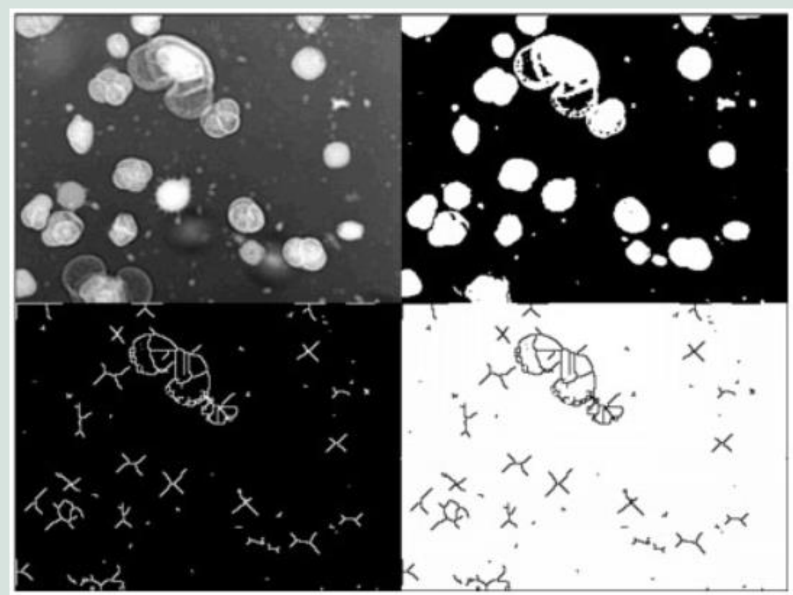
Thinning

- Thinning
  - peel from outside into inside, which is defined in terms of the hit-or-miss transform:

$$A \otimes B = A - (A \circledast B)$$
$$B = \{B^1, B^2, \ldots, B^n\}$$
$$A \otimes \{B\} = (((A \otimes B^1) \otimes B^2) \ldots) \otimes B^n)$$

0	0	0
	1	
1	1	1

	0	0
1	1	0
	1	



Thickening

- Thickening
  - The structure element B is similar to the structure element for thinning, except that regions of 1's and 0's are exchanged.
  - morphological dual of thinning

$$A \odot B = A \cup (A \circledast B)$$
$$B = \{B^1, B^2, \ldots, B^n\}$$
$$A \odot \{B\} = (((A \odot B^1) \odot B^2) \ldots) \odot B^n)$$

  - Alternative algorithm to thicken a set A,
    - apply "thinning" algorithm on A^c,
    - obtain region R
    - then take R^c as the thickening result

Skeletons

- Skeletons
  - can be implemented by the operations of erosions and openings

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

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

$$A \ominus kB = (((A \ominus B) \ominus B)\ldots) \ominus B)$$

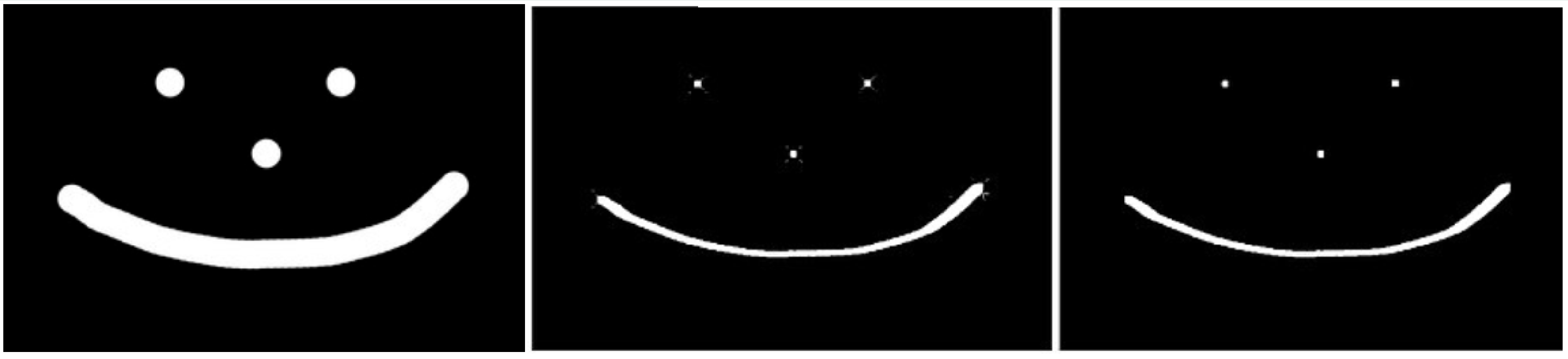
$$K = \max\{k \mid (A \ominus kB) \neq \varnothing\}$$

Pruning

- Pruning
  - it is complement to thinning and skeletonizing algorithm
  - removes small spurs on binary image
  - steps: thinning, finding end points, dilating end points, union of thinning and dilated end points
  - example:
    hand-writing recognition

$$B_1 = \begin{bmatrix} x & 0 & 0 \\ 1 & 1 & 0 \\ x & 0 & 0 \end{bmatrix} B_2 = \begin{bmatrix} x & 1 & x \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} B_3 = \begin{bmatrix} 0 & 0 & x \\ 0 & 1 & 1 \\ 0 & 0 & x \end{bmatrix} B_4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ x & 1 & x \end{bmatrix}$$

$$B_5 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} B_6 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} B_7 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} B_8 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$



Many Other Morphological Operations

- Gradient
- Top-hat
- Black-hat