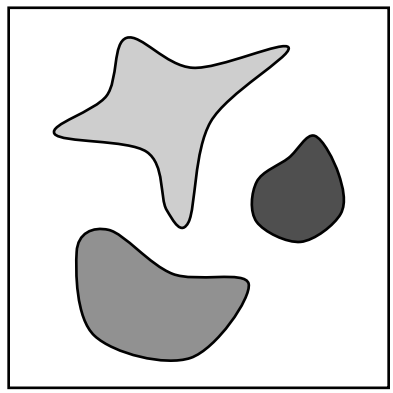
Binary Images

- Threshold
- Binary Image - Definition
- Connected Components
- Euler Number
- Chain Code
- Edge Following

Output of the segmentation process

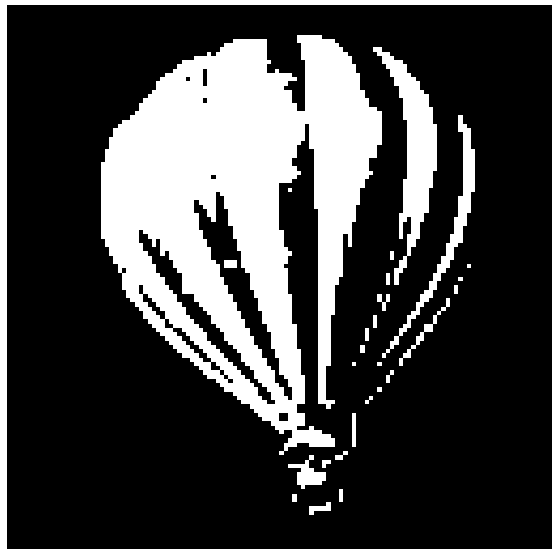


- How many objects
- Size of objects
- Shape of objects

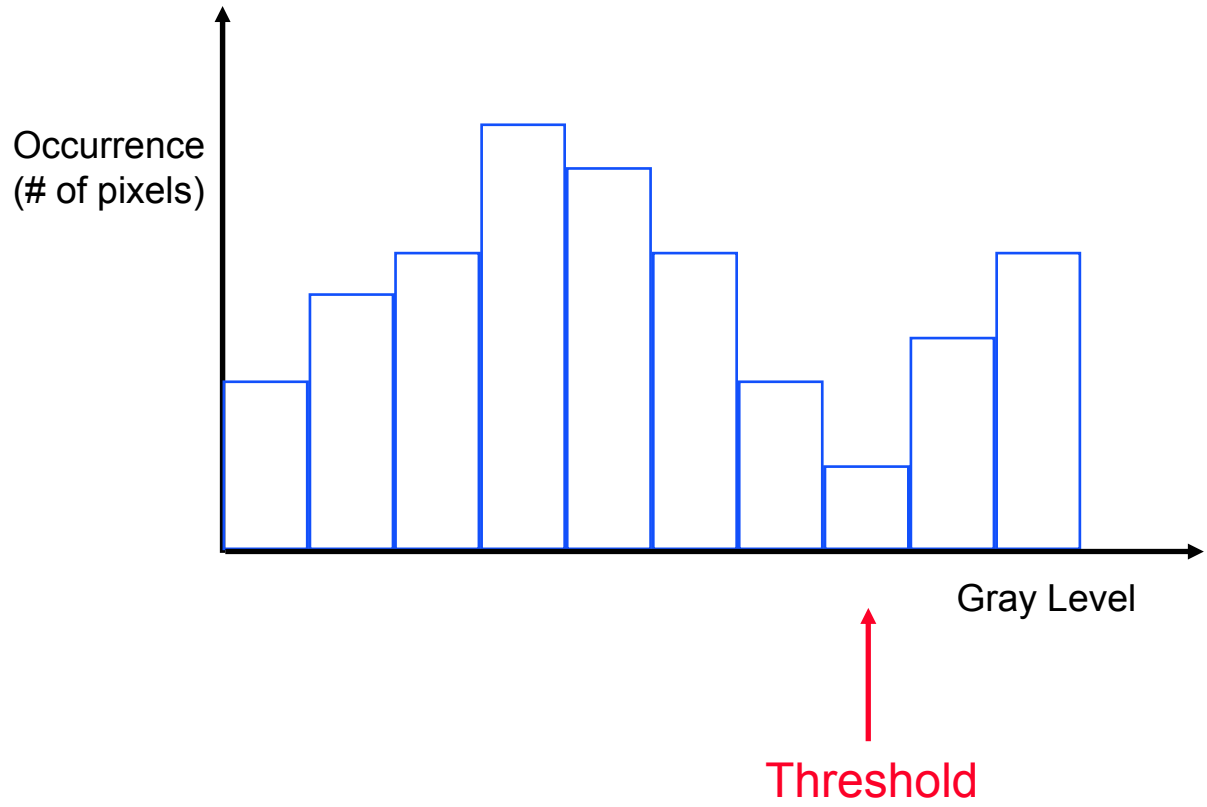
Binary Images

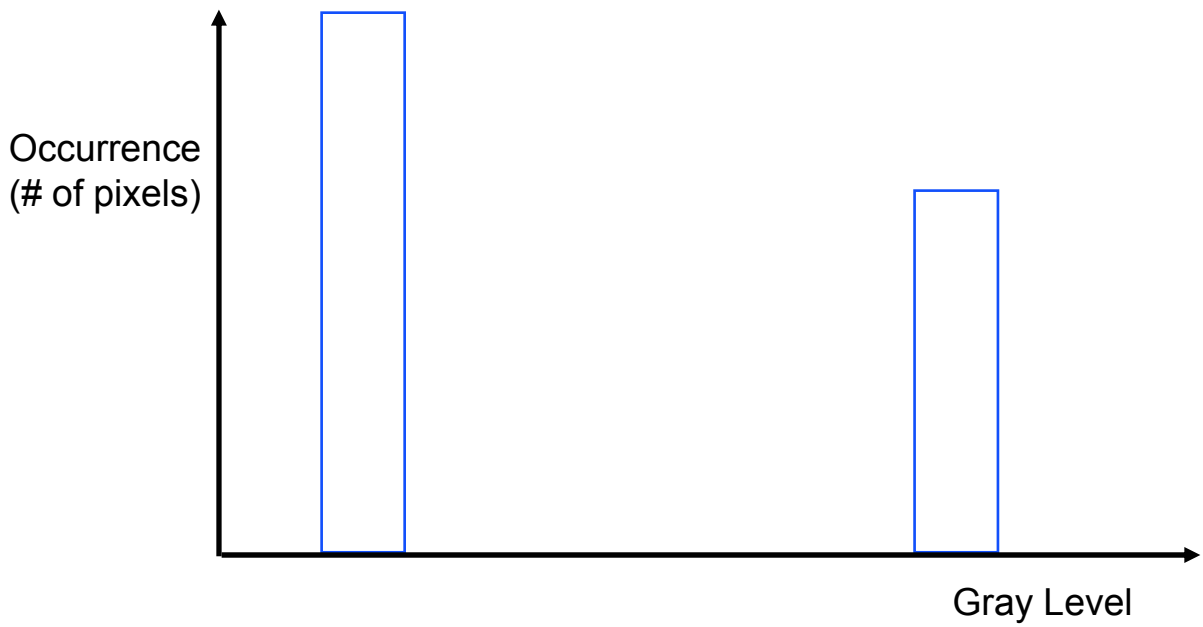
Grayscale Image

Binary Image



Thresholding

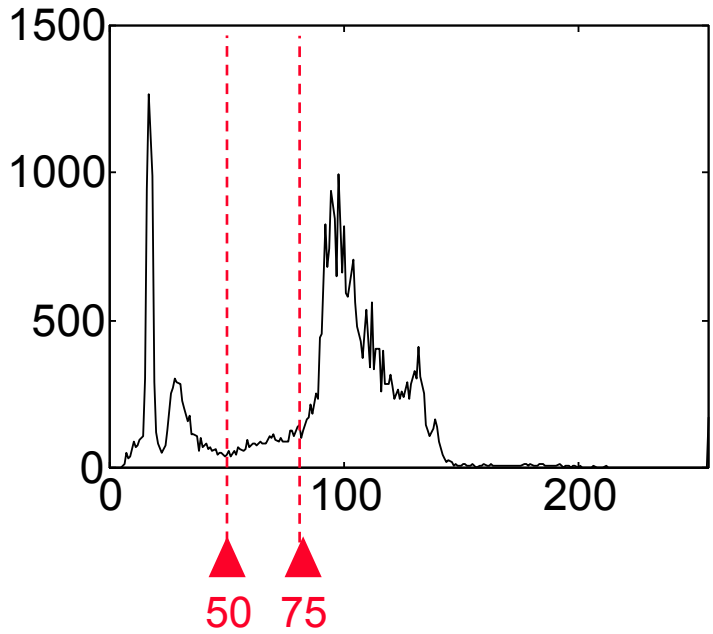


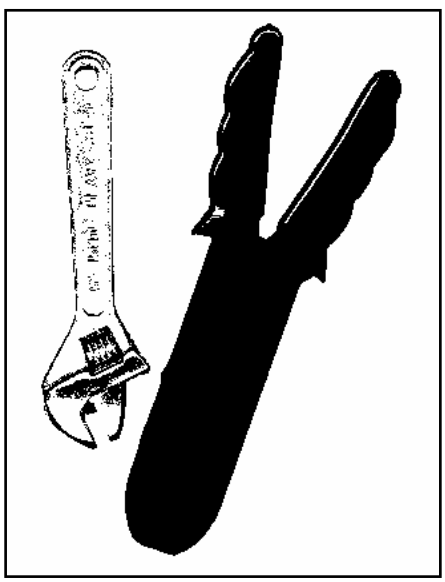


Segmentation using Thresholding

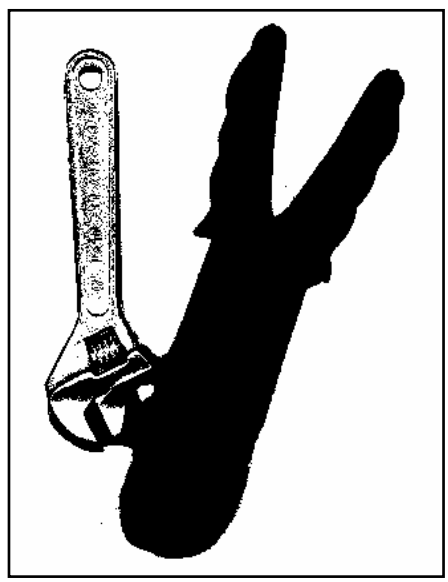
Original

Histogram





Threshold = 50

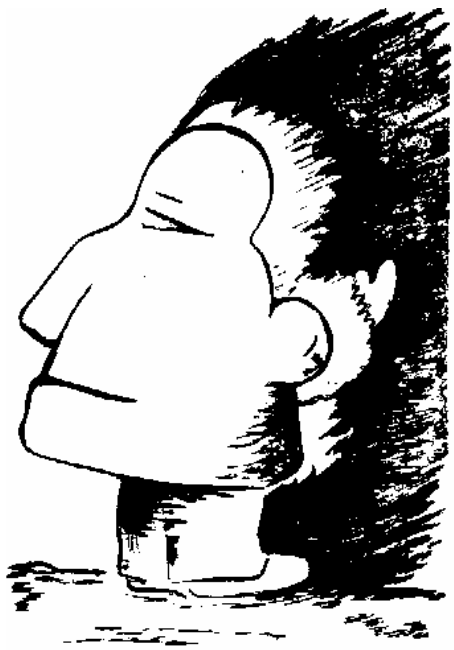


Threshold = 75

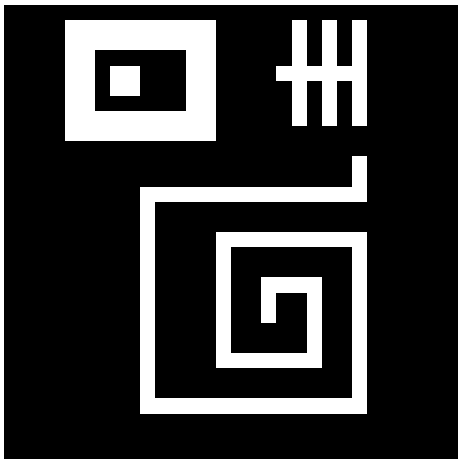
Figure - Ground

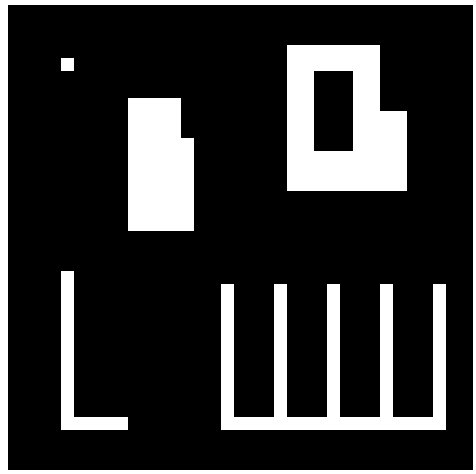
Binary Image = Figure + Ground

(Edgar Rubin 1915)

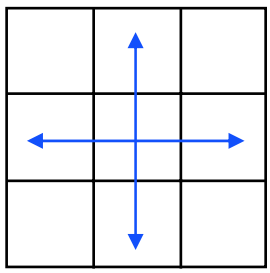


Connected Components

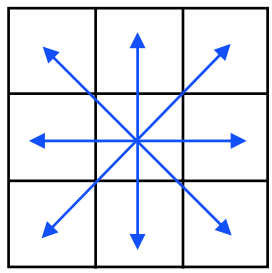




Neighborhoods:



4-neighbor metric



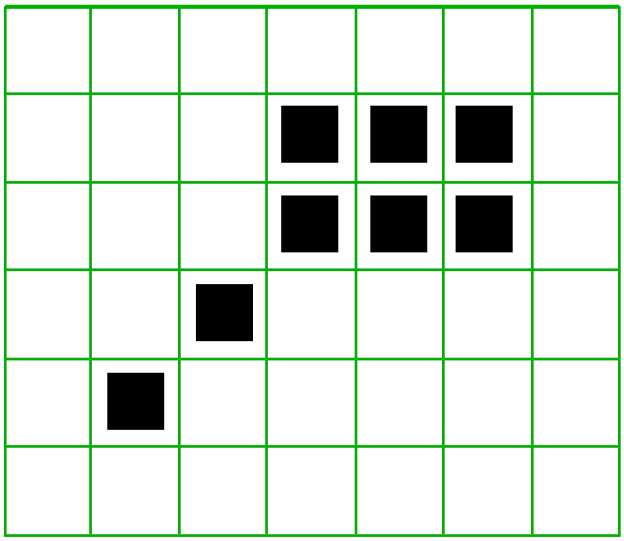
8-neighbor metric

Connected Components:

S = the set of object pixels

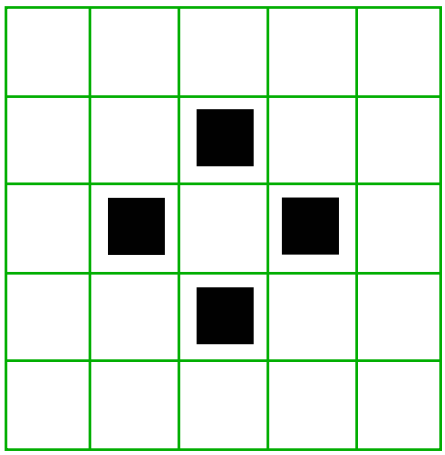
S is a Connected Component if for each pixel pair $(x_1,y_1) \in S$ and $(x_2,y_2) \in S$ there is a path passing through X-neighbors in S. (X = 4,8).

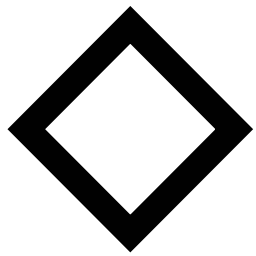
S may contain several connected components.



1 connected component-8

3 connected components-4





8-neighborhood:

1 object connected component

1 background connected component

4-neighborhood:

2 background connected components

4 object connected components

Always choose different neighborhood metrics for objects and backgrounds.

Marking the Connected Components

Connected Component Algorithm: Two passes over the image.

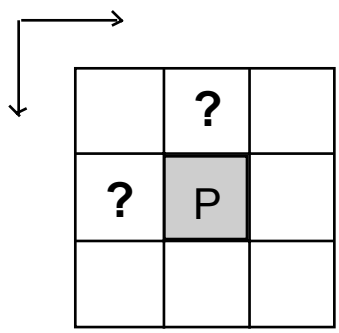
Pass 1:

Scan the image pixels from left to right and from top to bottom.
For every pixel P of value 1 (an object pixel), test top and left neighbors (4-neighbor metric).

- If 2 of the neighbors equal 0: assign a new mark to P.
- If 1 of the neighbors equals 1: assign the neighbor's mark to P.
- If 2 of the neighbors equal 1: assign the left neighbor's mark to P and note equivalence between 2 neighbor's marks.

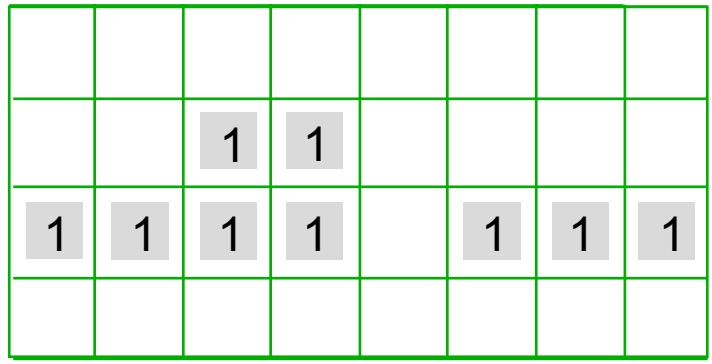
Pass 2:

Divide all marks in to equivalence classes (marks of neighboring pixels are considered equivalent).
Replace each mark with the number of it's equivalence class.

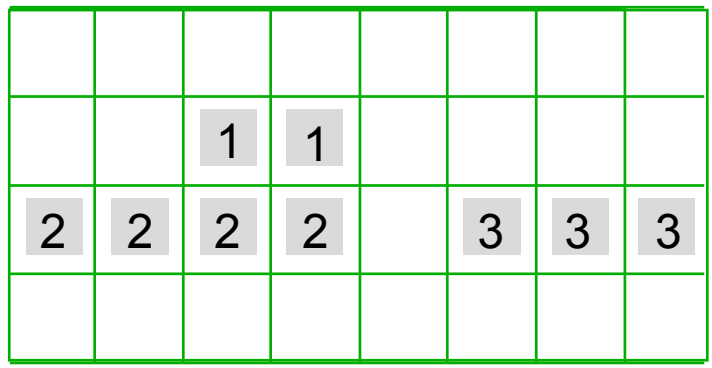


Connected Components - Example

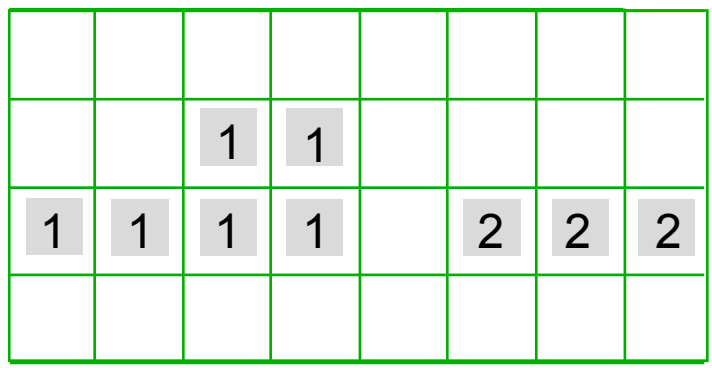
Original Binary image



Pass 1:



Pass 2:



Equivalence Class number	Original mark
1	1,2
2	3

Euler Number

S = object pixels

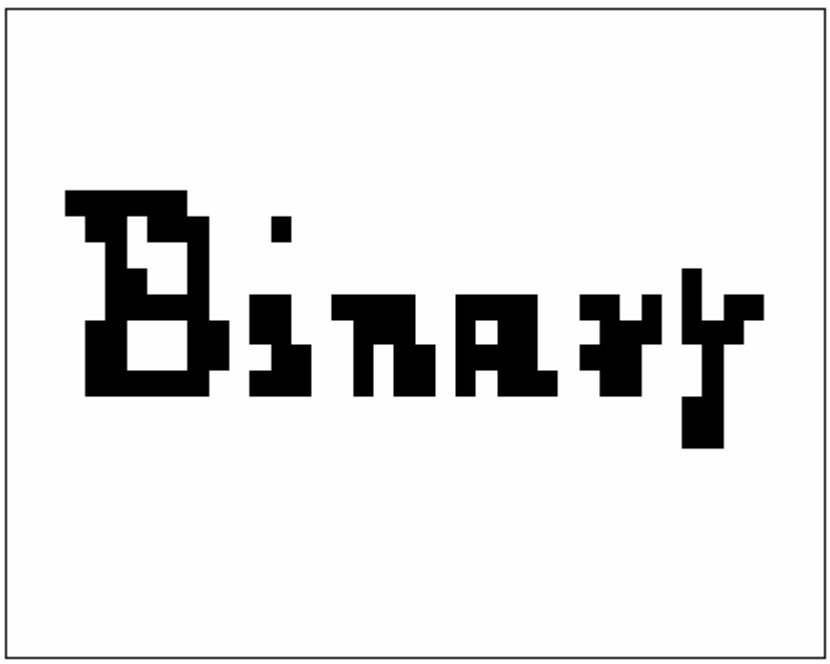
$\bar{S}$ = all other pixels

Background = connected components of $\bar{S}$ that touch the edge of the image.

Hole = connected components of $\bar{S}$ that is not in the background.

Simply Connected Component = a component without holes.

Euler Number = the number of objects minus the number of holes.



Example:

7 objects
3 holes

Euler = 4

Calculating the Euler Number

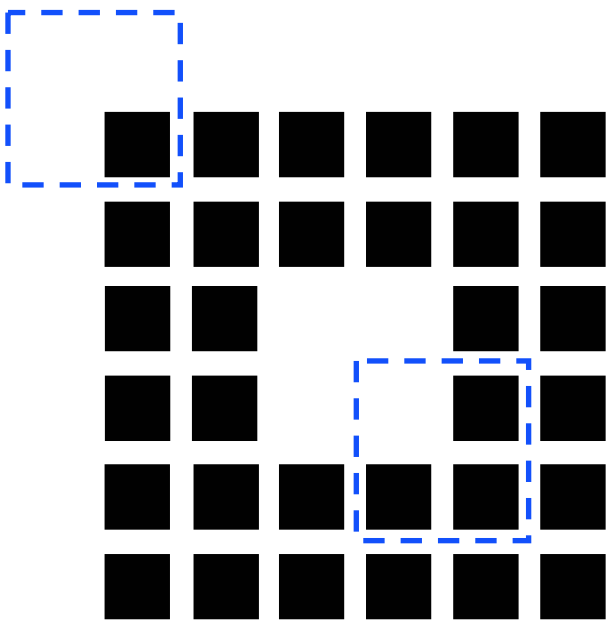
X = the number of occurrences of the 2x2 pattern:

0	0
0	1

V = the number of occurrences of the 2x2 pattern:

0	1
1	1

Euler = X - V

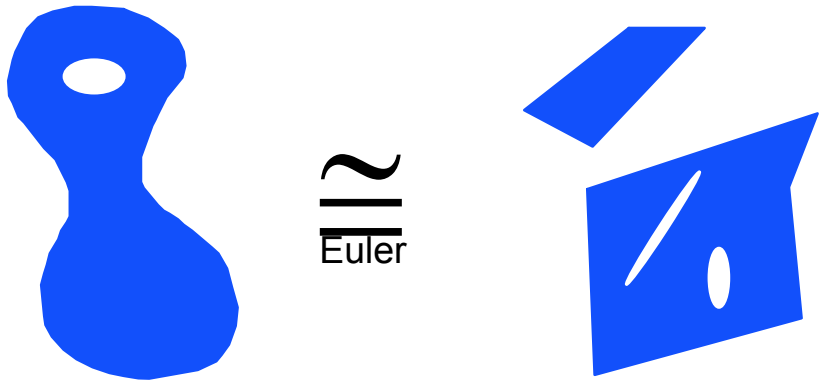


Euler = 1-1 = 0

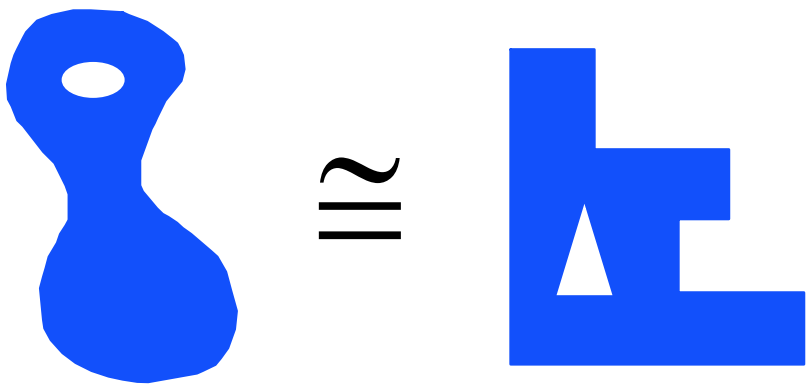
Note: finding the number of holes or the number of objects can **NOT** be achieved using local computations.

Euler Number is a Shape Descriptor

Euler Number = 0



Euler Number = 0
Objects = 1
Holes = 1



The Euler Number is **not** a good shape descriptor.

Characterize connected components by describing the edges..

Edges

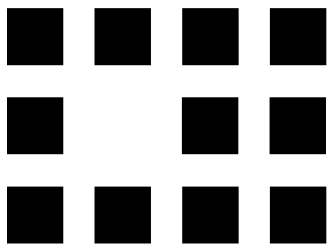
C = connected component of object S.

D = connected component of $\bar{S}$.

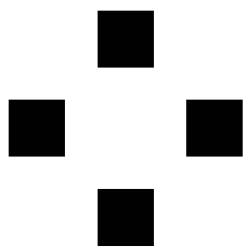
The D-Edge of C = the set of all pixels in C that have a neighboring pixel in D.
(neighboring-8 if C is 4-connected
neighboring-4 if C is 8-connected).

Example:

Object C

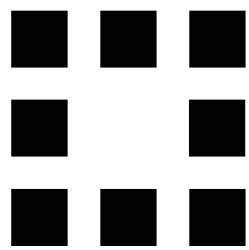


C is 8-connected

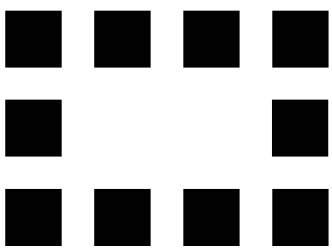


(4-neighbor)

C is 4-connected



(8-neighbor)



The Edge of C for background D.

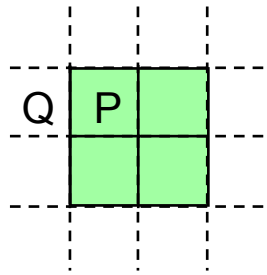
The Edge of C for hole D.

Edge Following

Edge Following Algorithm (for 8-neighbor object):

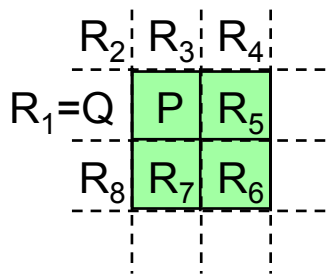
At all stages of the algorithm maintain:
- 1 object pixel - P
- 1 background pixel - Q

Where P and Q are 4-neighbors.



1. Start with P_0 and Q_0.
2. Given P and Q find the next pair P' and Q' as follows:
   The 8 neighbors of P in clockwise order starting at Q are:

$Q=R_1, R_2, ..., R_8$

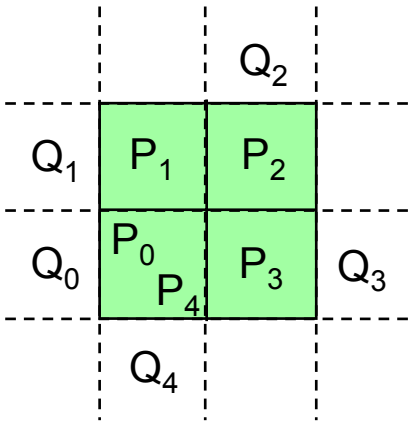


Let R_i be the first R in this sequence which is equal 1 (object).
Set:

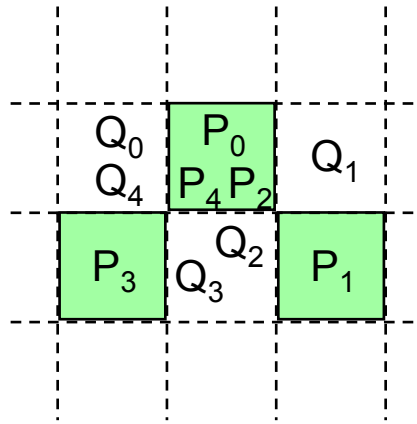
$P'=R_i$ and $Q'=R_{i-1}$

3. Repeat step 2 until P_0 is reached again under the condition that Q_0 will be reached in searching for the following P'.
4. The sequence $P_0, P_1, P_2, ...$ are the edge pixels of the connected component.

Examples:



Edges = P_0,P_1,P_2,P_3



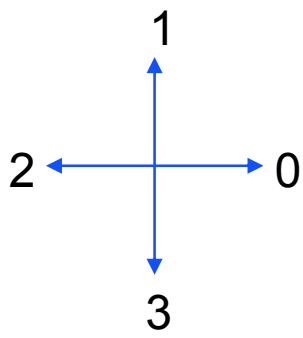
Edges = P_0,P_1,P_2,P_3

Finding **ALL** edges in an image:

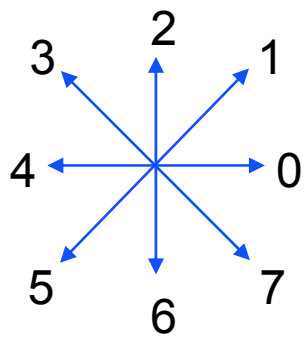
Scan the image (for example left-to-right top-to-bottom).
When an unmarked object pixel is found bordering the background - follow the edge of the connected component starting at that pixel while marking the edge pixels.

Chain Code

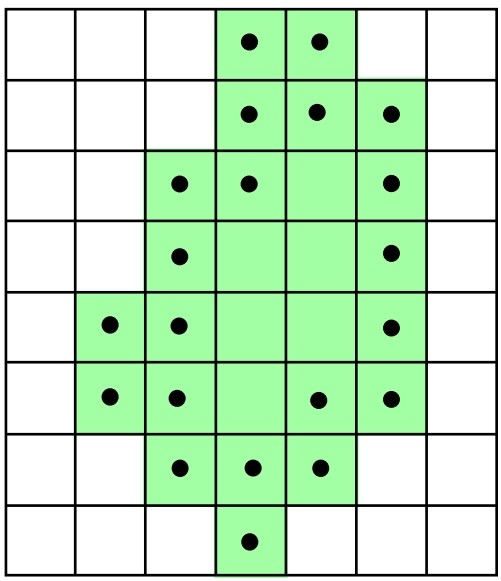
Each direction is assigned a code:

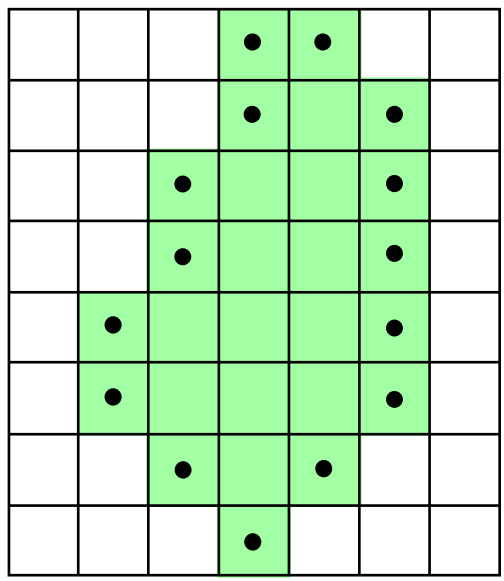


4-neighbor

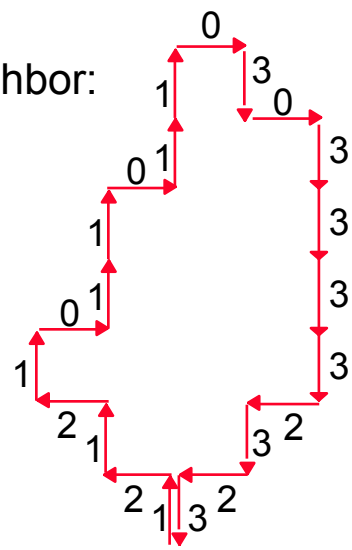


8-neighbor



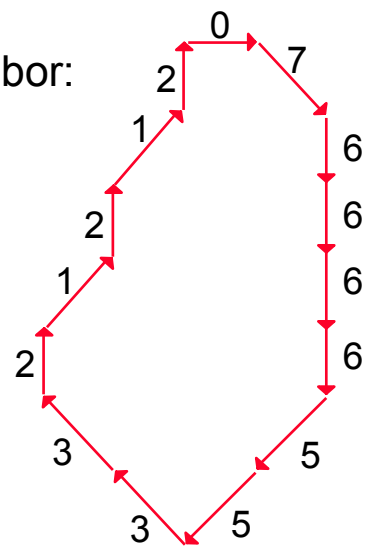


4-neighbor:



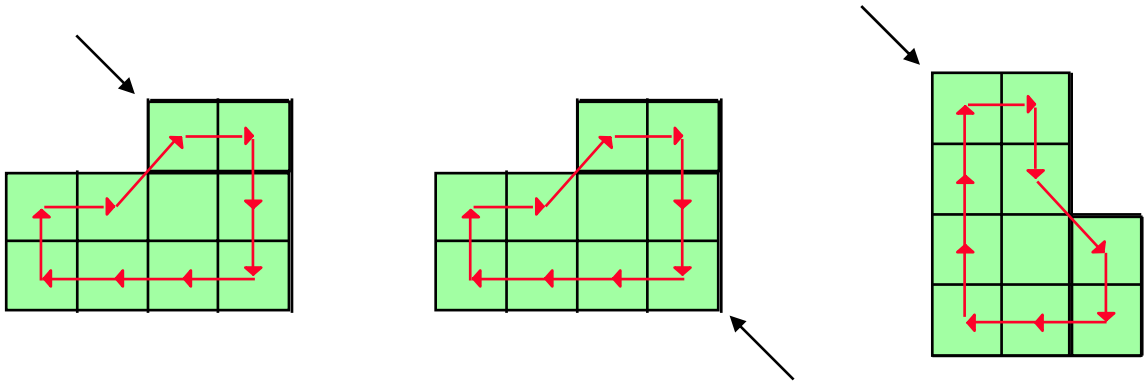
0303333232312121011011

8-neighbor:



076666553321212

Chain Code Independent of Location, Starting Point and Orientation



Chain Code:

444201066 067644222 066444201

Curvature = differences in chain code values :

0022-11-602 -6-11202002 -6020022-11

Normalize curvature = mod 8 :

002271202 271202002 202002271

Code is the cyclic permutation which produces the smallest number:

002271202

Distances

Two grid point: P = (x,y) and Q = (u,v)

Euclidean Distance

$$d_e(P,Q) = \sqrt{(x-u)^2 + (y-v)^2}$$

City Block Distance

$$d_4(P,Q) = |(x-u)| + |(y-v)|$$

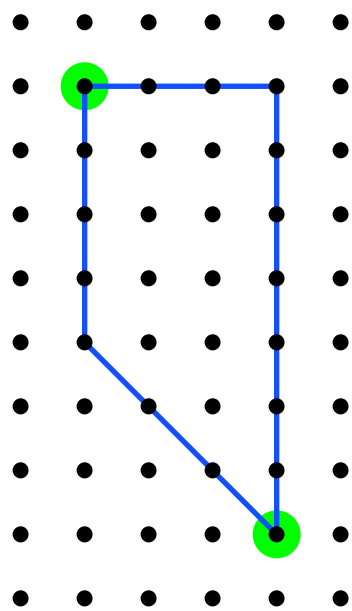
Chessboard Distance

$$d_8(P,Q) = \max(|(x-u)|, |(y-v)|)$$

$d_e = \sqrt{3^2 + 7^2} = 7.6$

$d_8 = 7$

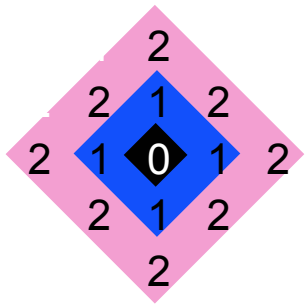
$d_4 = 10$



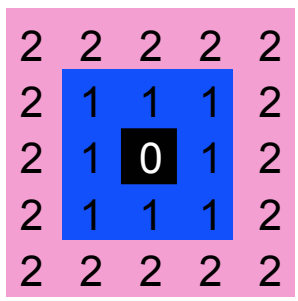
d_e d_8 d_4 are all **metrics**:

1. Distance metric: $d(P,Q) \geq 0$
2. Positive: $d(P,Q) = 0$ iff $P=Q$
3. Symmetric: $d(P,Q) = d(Q,P)$
4. Triangular inequality: $d(P,Q) \leq d(P,R) + d(R,Q)$

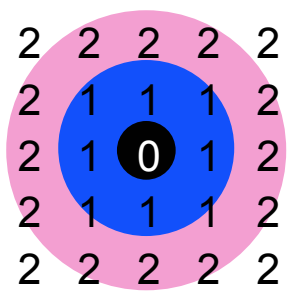
All pixels at equal d_4 distance form a “diamond” :



All pixels at equal d_8 distance form a “square” :

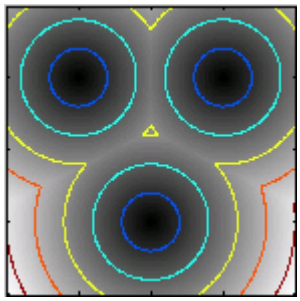


All pixels at equal d_e distance form a “circle” :

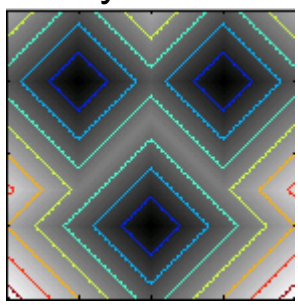


Distance Map

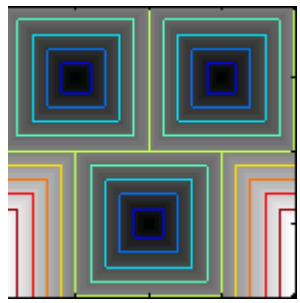
Euclidean

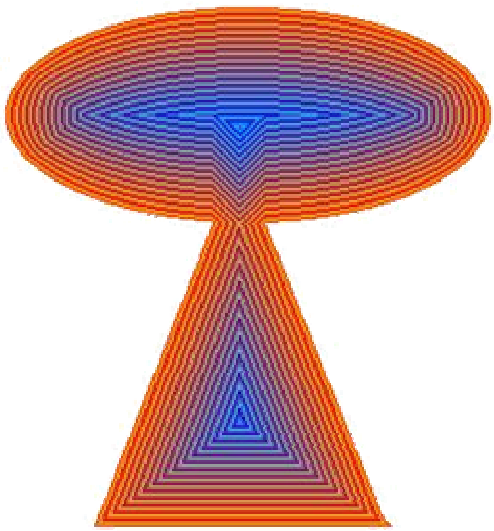


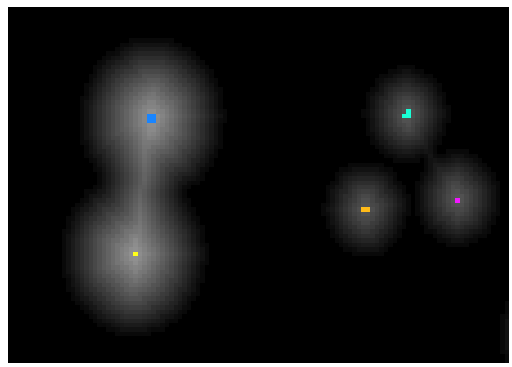
City Block

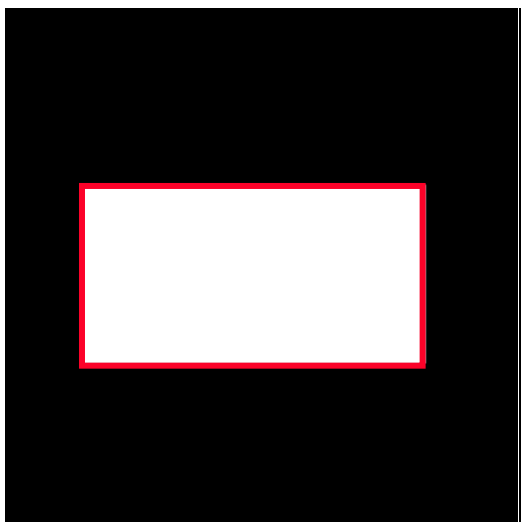


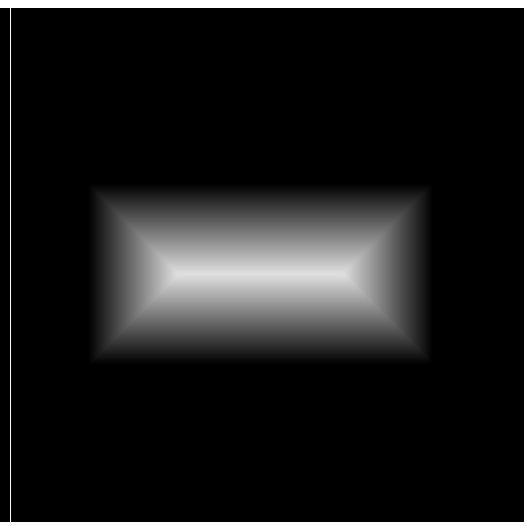
Chessboard











2-Pass Distance Algorithm

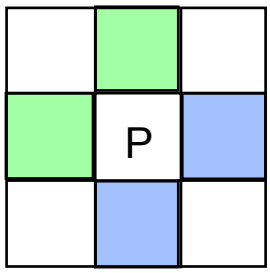
For each pixel calculate the d_4 or d_8 distance from a pixel in set S.

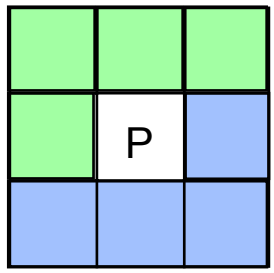
2 passes:

Pass 1: scan image left-to-right and top-to-bottom

Pass 2: scan image right-to-left and bottom-to-top.

For each pixel P mark as follows:





Pass 1: consider all neighbors of P that have been scanned N_1 =

$$d'(P,S) = \begin{cases} 0 & \text{if } P \in S \\ \min\limits_{Q \in N_1} \{d'(Q,S)\}+1 & \text{if } P^- \ S \end{cases}$$

Pass 2: consider all neighbors of P that have been scanned N_2 =

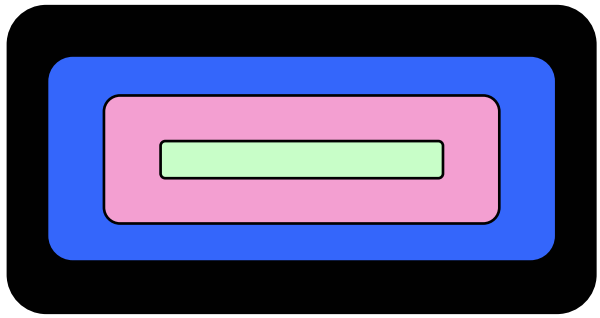
$$d''(P,S) = \min_{Q \in N_2} \{d'(P,S) , d''(Q,S)+1\}$$

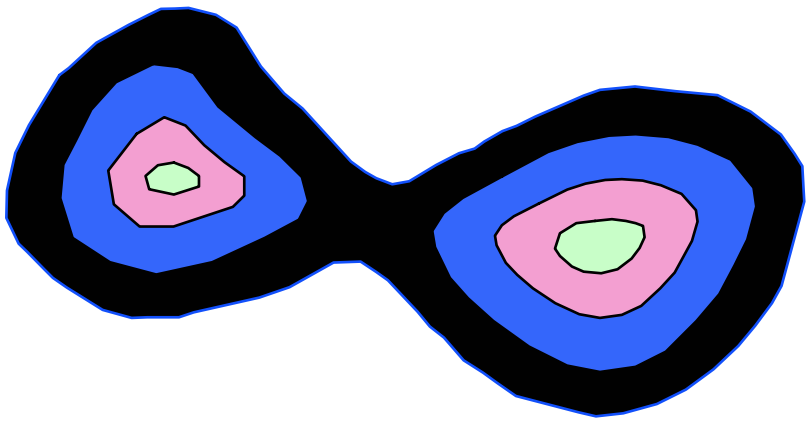
Example measuring d_4:

1 0 0 0	0 1 2 3	0 1 2 1
0 0 0 1	1 2 3 0	1 2 1 0
0 0 0 0	2 3 4 1	2 3 2 1
S is marked as 1	Pass 1: d'(P,S)	Pass 2: d''(P,S)

Skeletons

Consider all edge pixels of an object as the seed group S.



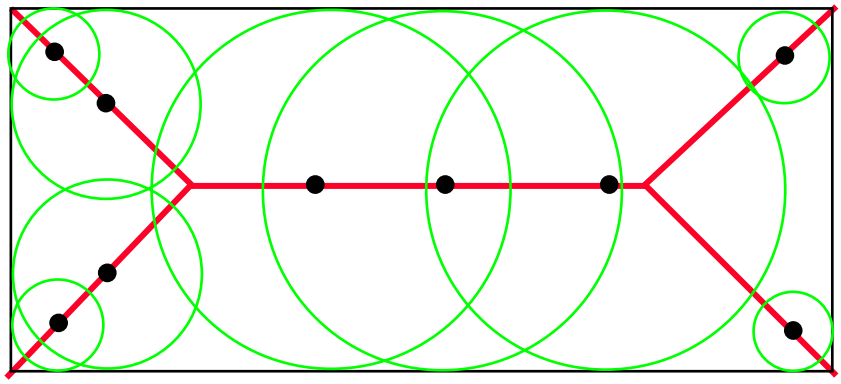


The pixels whose distance is a local maxima are the Skeleton of the object.

The Skeleton can be used as a shape descriptor.

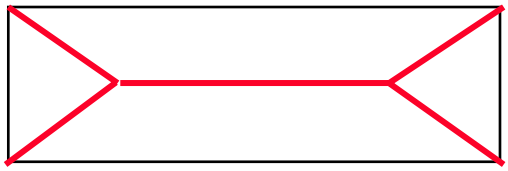
MAT = Medial Axis Transform

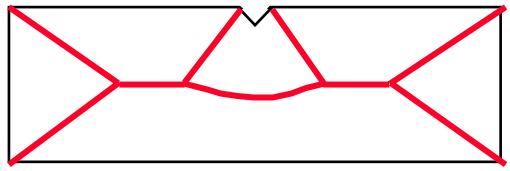
Grass fire technique (Blum, 1993)

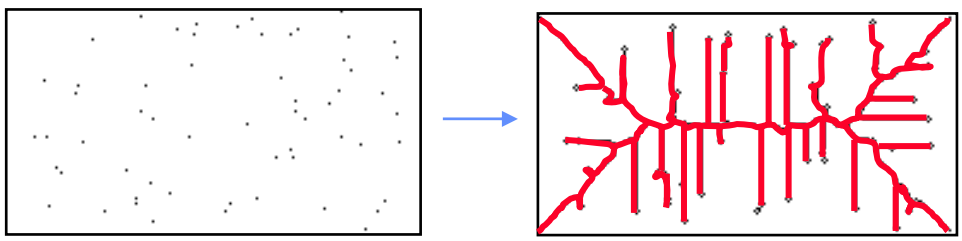


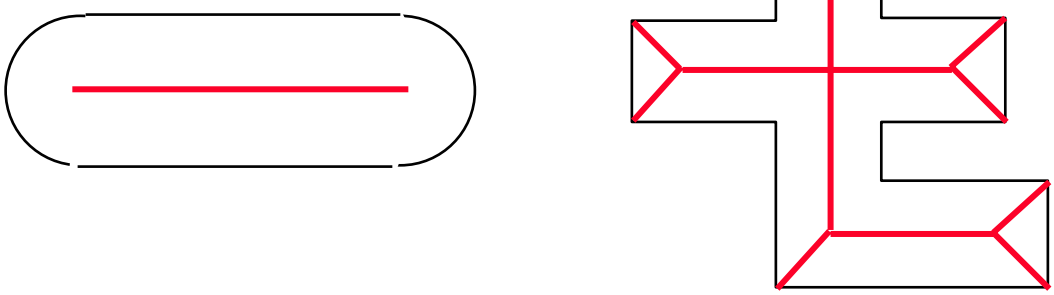
Centers of Maximal discs

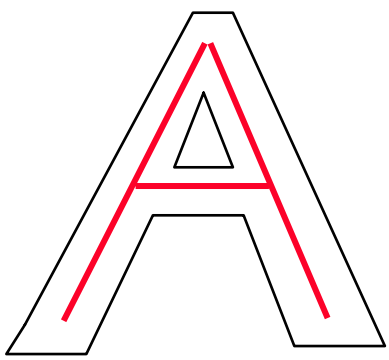
MAT = Medial Axis Transform

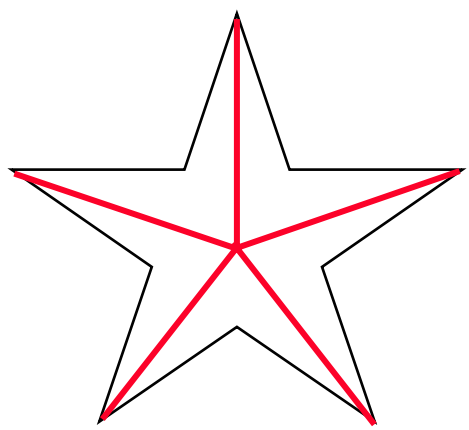








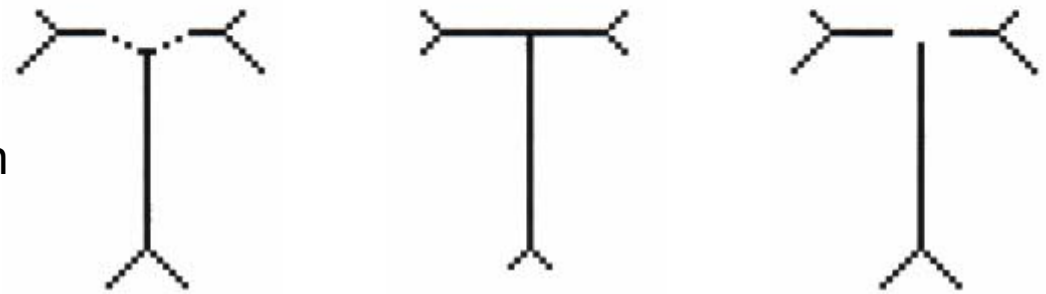




Skeletons - Example

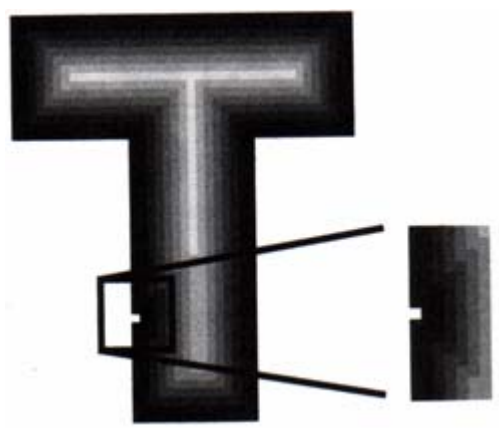
Distance Map

Skeleton

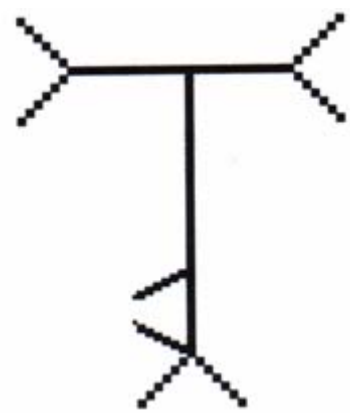


d_e d_4 d_8

Sensitivity to contour changes:



Distance Map



Skeleton

Skeleton-Example