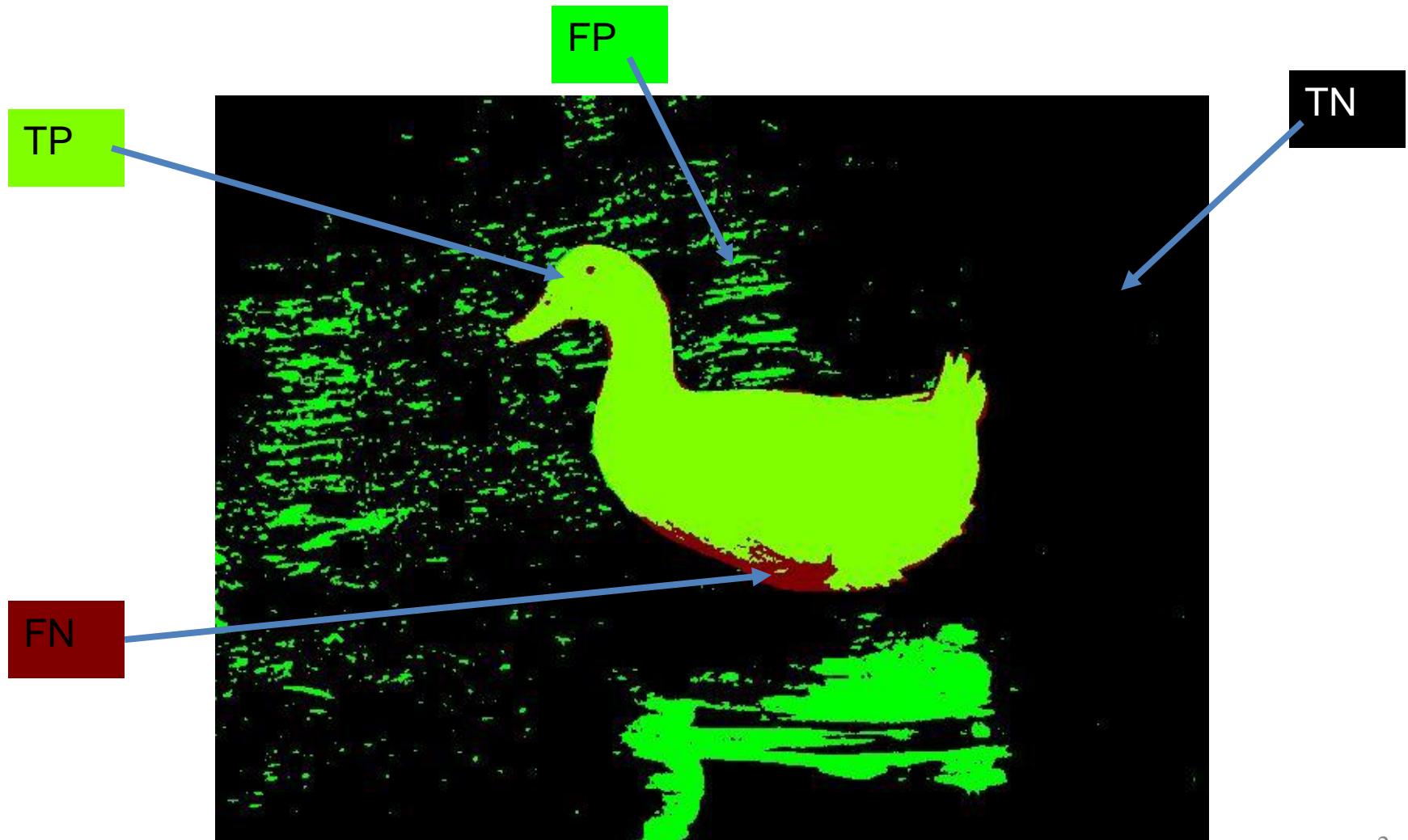


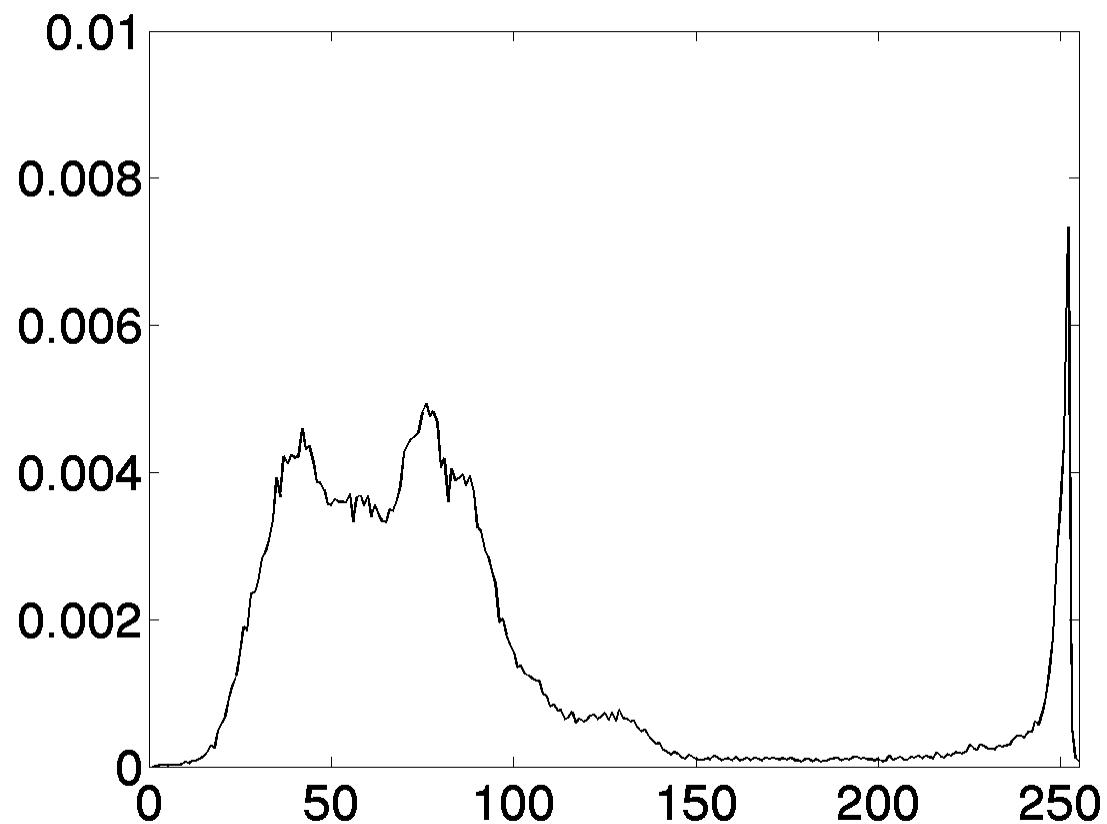
Visual Computing: Image Segmentation

Prof. Marc Pollefeys

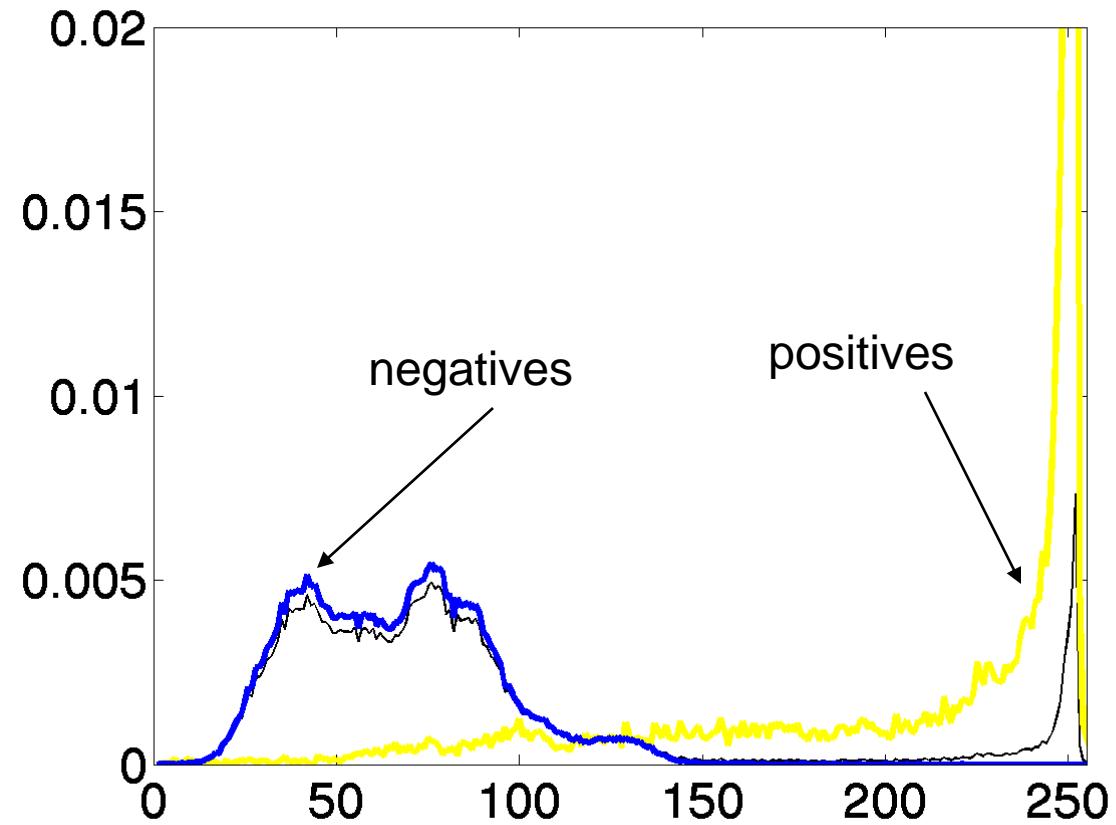
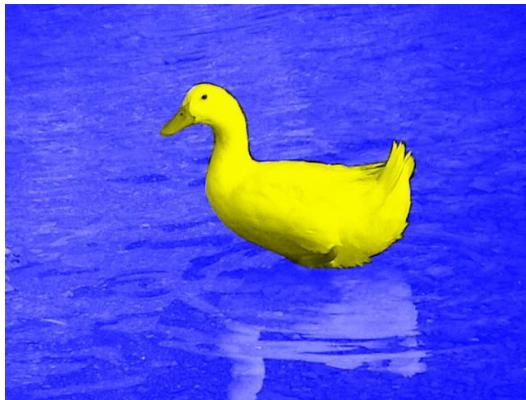
Classification outcomes



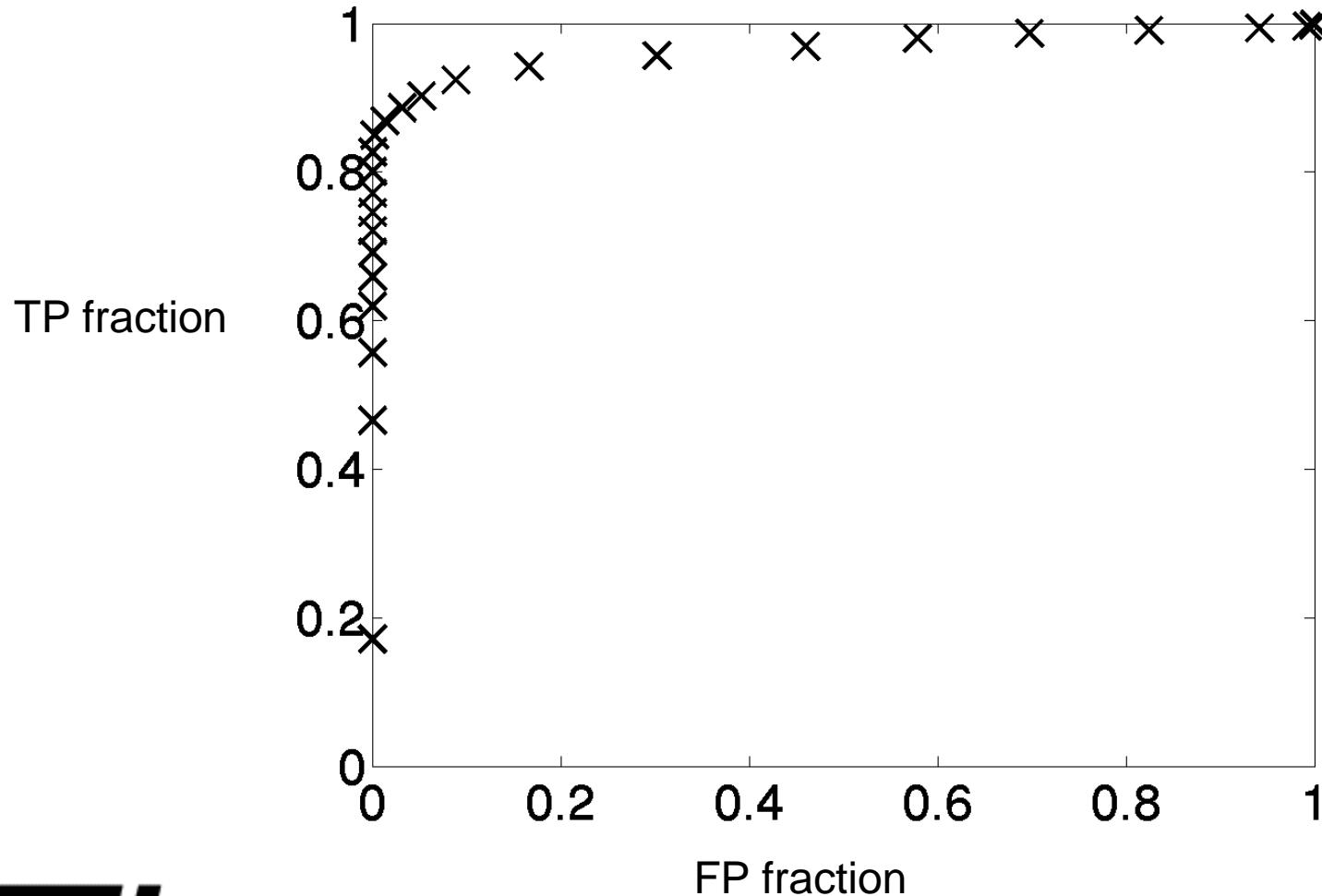
Greylevel Histograms



Positives and Negatives



ROC curve



Limitations of Thresholding

- Why can we segment images much better by eye than through thresholding processes?
- We might improve results by considering image **context**: Surface Coherence

Note on Performance Assessment

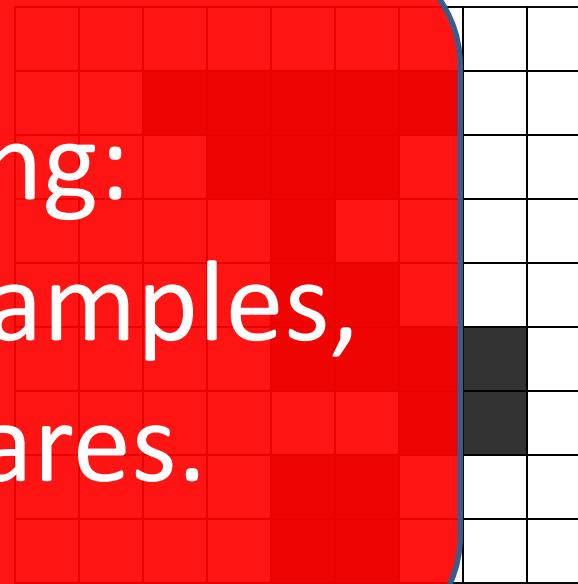
- In real-life, we use two or even three separate sets of test data:
 1. A ***training set***, for tuning the algorithm
 2. A ***validation*** set for tuning the performance score
 3. An unseen ***test set*** to get a final performance score on the tuned algorithm

Pixel connectivity

- We need to define which pixels are neighbors.
- Are the dark pixels in this image connected?

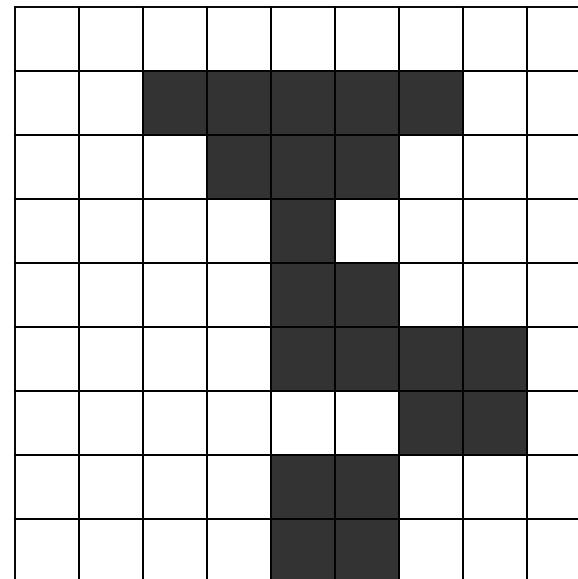
Warning:

**Pixels are samples,
not squares.**

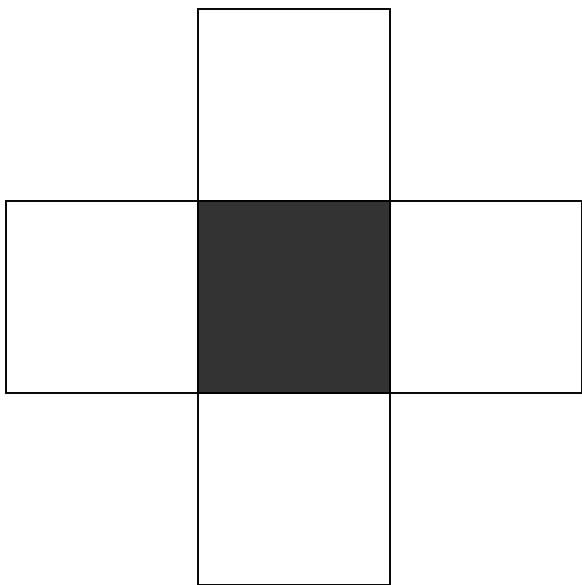


Pixel connectivity

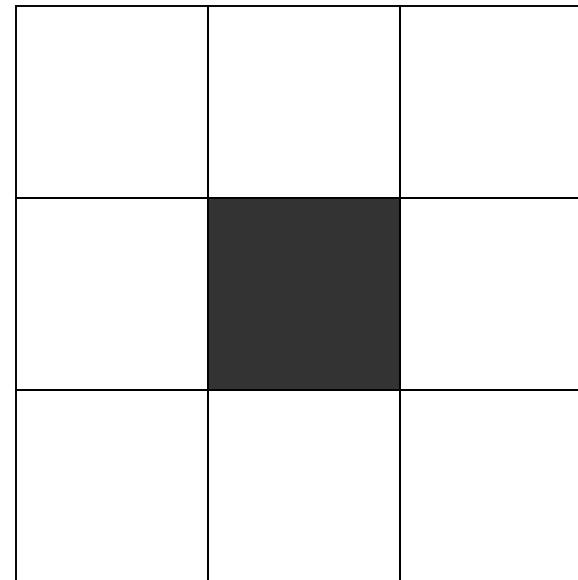
- We need to define which pixels are neighbors.
- Are the dark pixels in this array connected?



Pixel Neighborhoods



4-neighborhood



8-neighborhood

Pixel paths

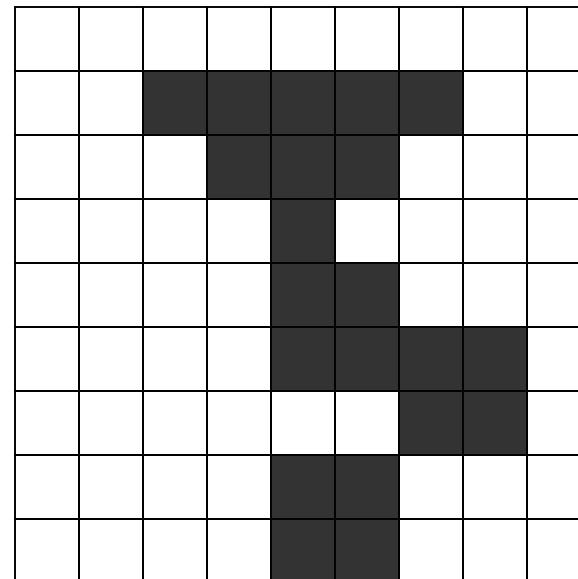
- A 4-connected path between pixels p_1 and p_n is a set of pixels $\{p_1, p_2, \dots, p_n\}$ such that p_i is a 4-neighbor of p_{i+1} , $i=1,\dots,n-1$.
- In an 8-connected path, p_i is an 8-neighbor of p_{i+1} .

Connected regions

- A region is 4-connected if it contains a 4-connected path between any two of its pixels.
- A region is 8-connected if it contains an 8-connected path between any two of its pixels.

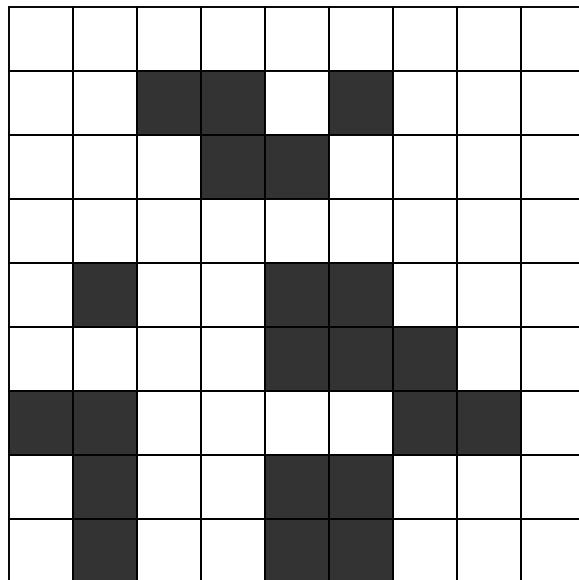
Connected regions

- Now what can we say about the dark pixels in this array?
- What about the light pixels?



Connected components labelling

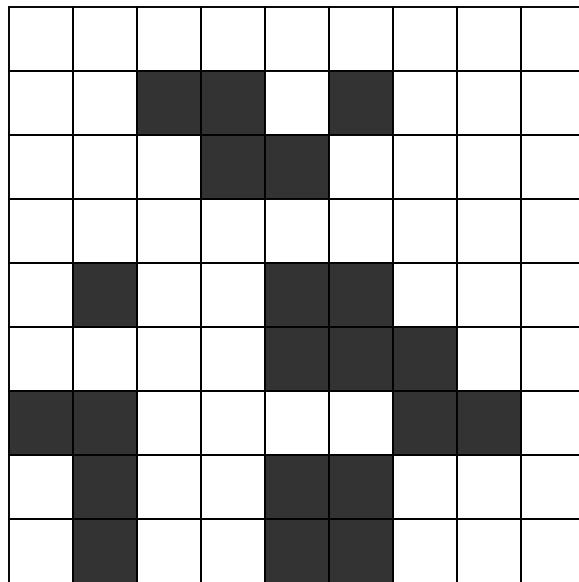
- Labels each connected component of a binary image with a separate number.



1	1	1	1	1	1	1	1	1	1
1	1	2	2	1	3	1	1	1	1
1	1	1	2	2	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	4	1	1	5	5	1	1	1	1
1	1	1	1	5	5	5	1	1	1
6	6	1	1	1	1	5	5	5	1
7	6	1	1	8	8	1	1	1	1
7	6	1	1	8	8	1	1	1	1

Foreground labelling

- Only extract the connected components of the foreground

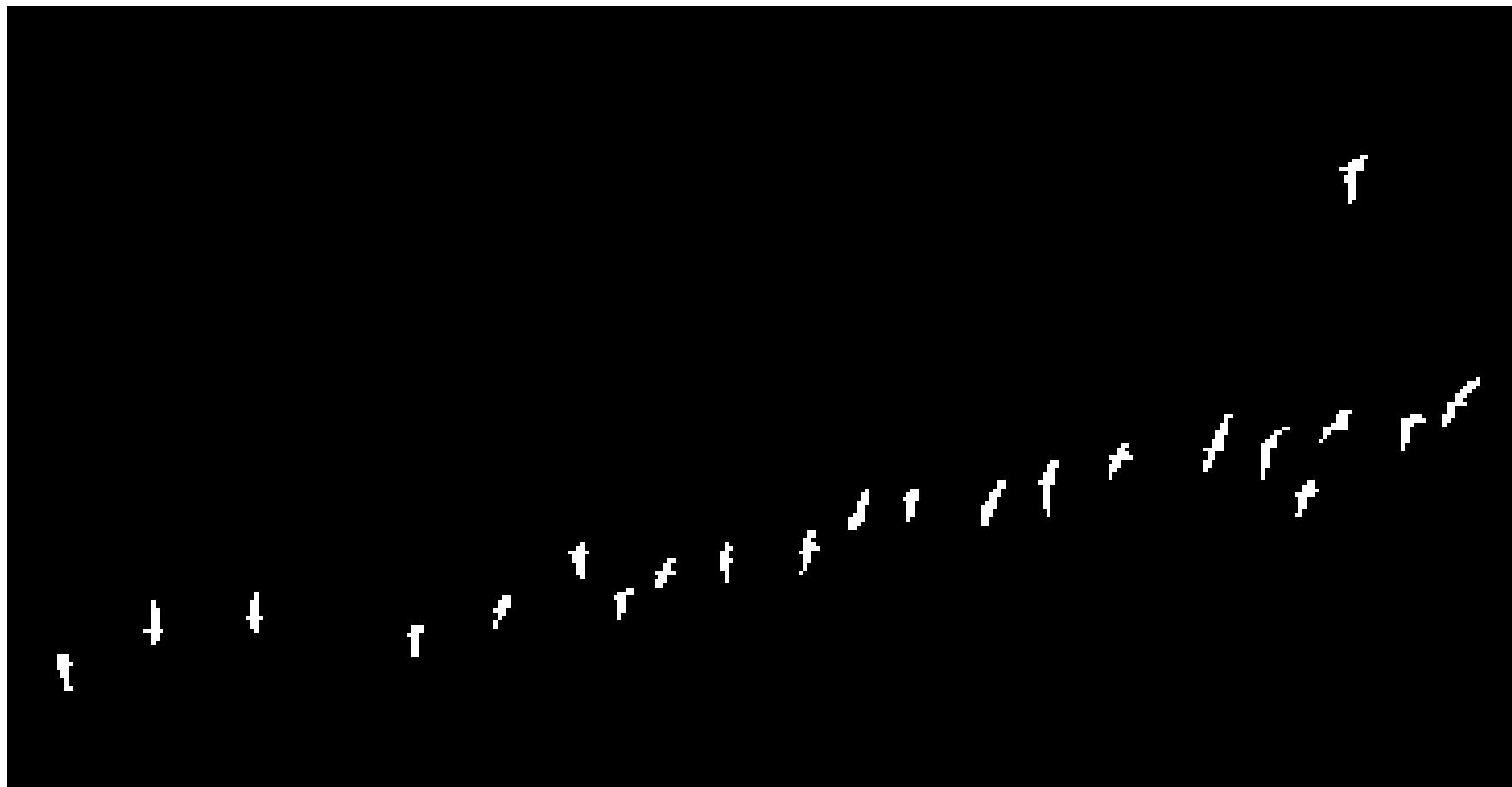


1	1	1	1	1	1	1	1	1	1
1	1	0	0	1	0	1	1	1	1
1	1	1	0	0	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	0	1	1	0	0	1	1	1	1
1	1	1	1	0	0	0	1	1	1
0	0	1	1	1	1	0	0	0	1
2	0	1	1	0	0	1	1	1	1
2	0	1	1	0	0	1	1	1	1

Goose detector



Goose detector



Region Growing

- Start from a seed point or region.
- Add neighboring pixels that satisfy the criteria defining a region.
- Repeat until we can include no more pixels.

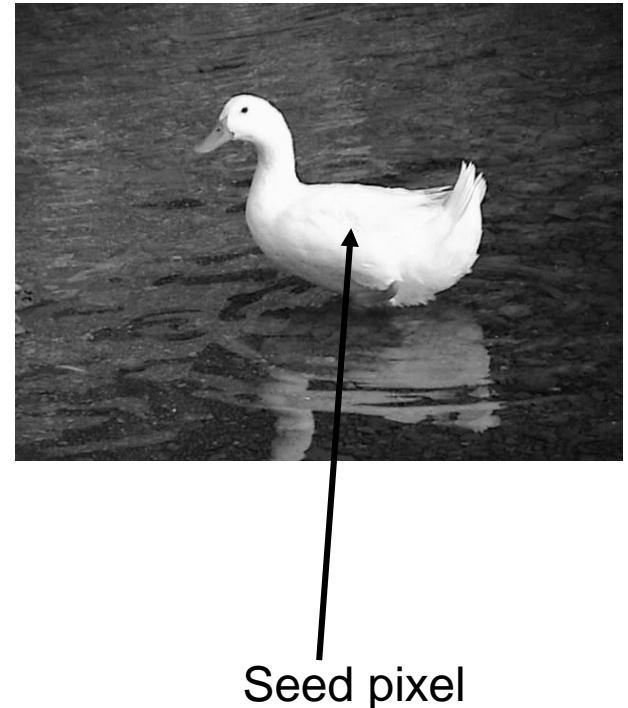
Region Growing

```
def regionGrow(I, seed):  
    X, Y = I.shape  
    visited = np.zeros((X,Y))  
    visited[seed] = 1  
    boundary = []  
    boundary.append(seed)  
    while len(boundary) > 0:  
        nextPoint = boundary.pop()  
        if include(nextPoint, seed):  
            visited[nextPoint] = 2  
            for (x, y) in neighbors(nextPoint):  
                if visited[x,y] == 0:  
                    boundary.append((x, y))  
                    visited[x,y] = 1
```

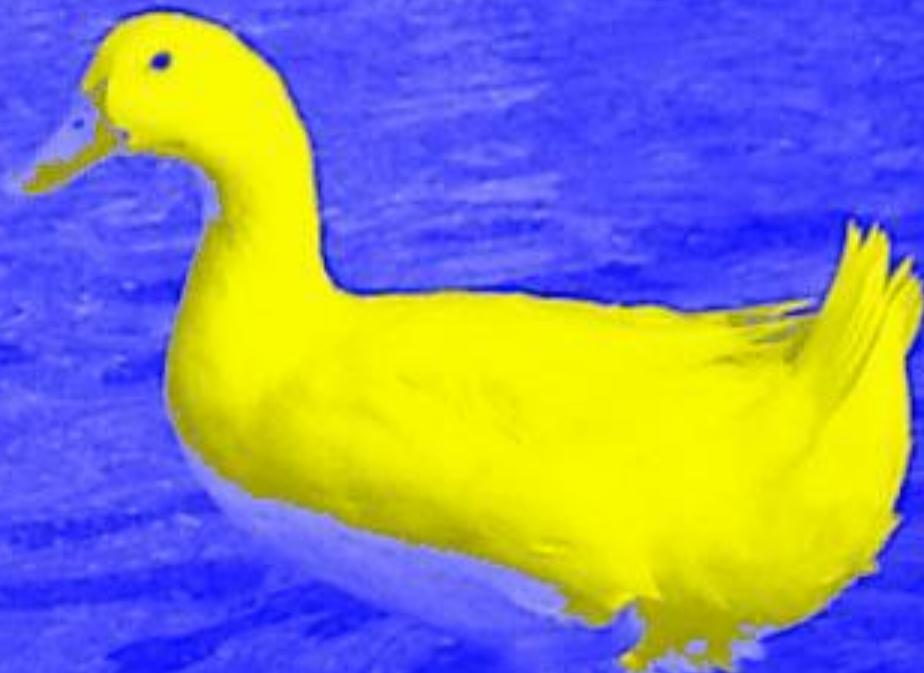
Region Growing example

- Pick a single seed pixel
- Inclusion test is up to you:

```
def include(p, seed) :  
    test = ??  
    return test
```



$T=150$



Variations

- Seed selection
- Inclusion criteria
- Boundary constraints and snakes

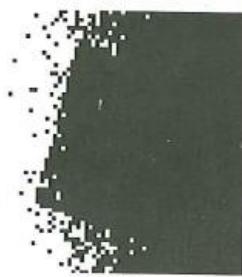
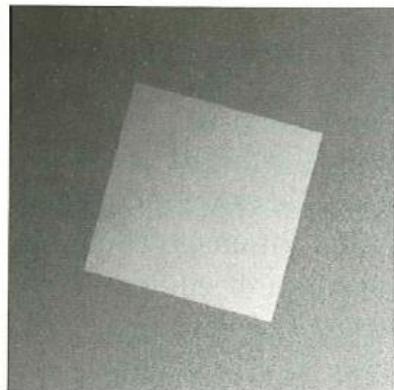
Seed selection

- Point and click seed point
- Seed region
 - By hand
 - Automatically, e.g., from a conservative thresholding.
- Multiple seeds
 - Automatically labels the regions

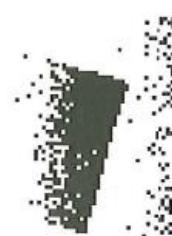
Inclusion criteria

- Greylevel thresholding
- Greylevel distribution model
 - Use mean μ and standard deviation σ in seed region:
 - Include if $(I(x, y) - \mu)^2 < (n\sigma)^2$. Eg: $n = 3$.
 - Can update the mean and standard deviation after every iteration.
- Color or texture information

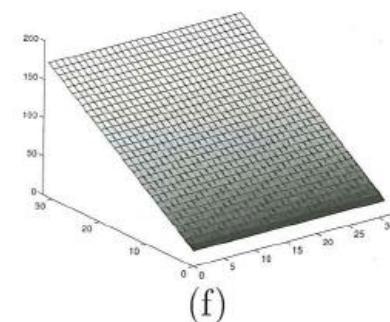
Inclusion criteria ?



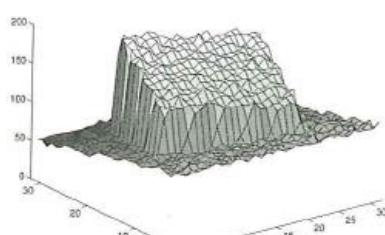
(d)



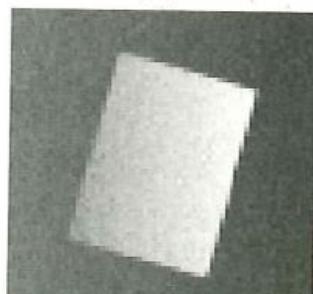
(e)



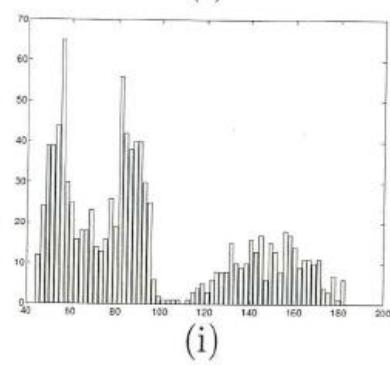
(f)



(g)



(h)



(i)

e.g. $(a.l+b.x+c.y+d)^2 < \text{threshold}$
(keep refitting a,b,c to included pts)



(j)

Snakes

- A snake is an *active contour*
- It is a polygon, i.e., an ordered set of points joined up by lines
- Each point on the contour moves away from the seed while its image neighborhood satisfies an inclusion criterion
- Often the contour has smoothness constraints

Snakes

- The algorithm iteratively minimizes an energy function:
- $E = E_{\text{tension}} + E_{\text{stiffness}} + E_{\text{image}}$
- See Kass, Witkin, Terzopoulos, IJCV 1988

Example



Interim Summary

- Segmentation is hard
- But it is easier if you define the task carefully
 - Is the segmentation task binary or continuous?
 - What are the regions of interest?
 - How accurately must the algorithm locate the region boundaries?
- Research problems remain!

Foreground-Background segmentation

Roundabout example

- Input



- Output



Distance Measures

Plain Background-subtraction metric:

$$I_{\alpha} = |I - I_{bg}| > T$$

$T = [20 \ 20 \ 10]$ (for example)

I_{bg} = Background Image

Where Does I_{bg} Come From?



When possible, fit a Gaussian model per pixel,
just as we did for an entire green-screen:

- mean $\mu \rightarrow I_\mu$
- standard deviation $\sigma \rightarrow I_\Sigma$



Distance Measures

Plain Background-subtraction metric:

$$I_\alpha = |I - I_{bg}| > T$$

$$T = [20 \ 20 \ 10] \quad (\text{for example})$$

I_{bg} = Background Image

or better

$$I_\alpha = \sqrt{\left(I - I_{bg} \right)^T \Sigma^{-1} \left(I - I_{bg} \right)} > T = 4 \quad (\text{for example})$$

Σ background pixel appearance covariance matrix
(computed separately for each pixel, from many examples)
(sometimes need more than one Gaussian, use Gaussian Mixture Models)



A Word About Shadows



A Word About Shadows



What happened
to the color here?

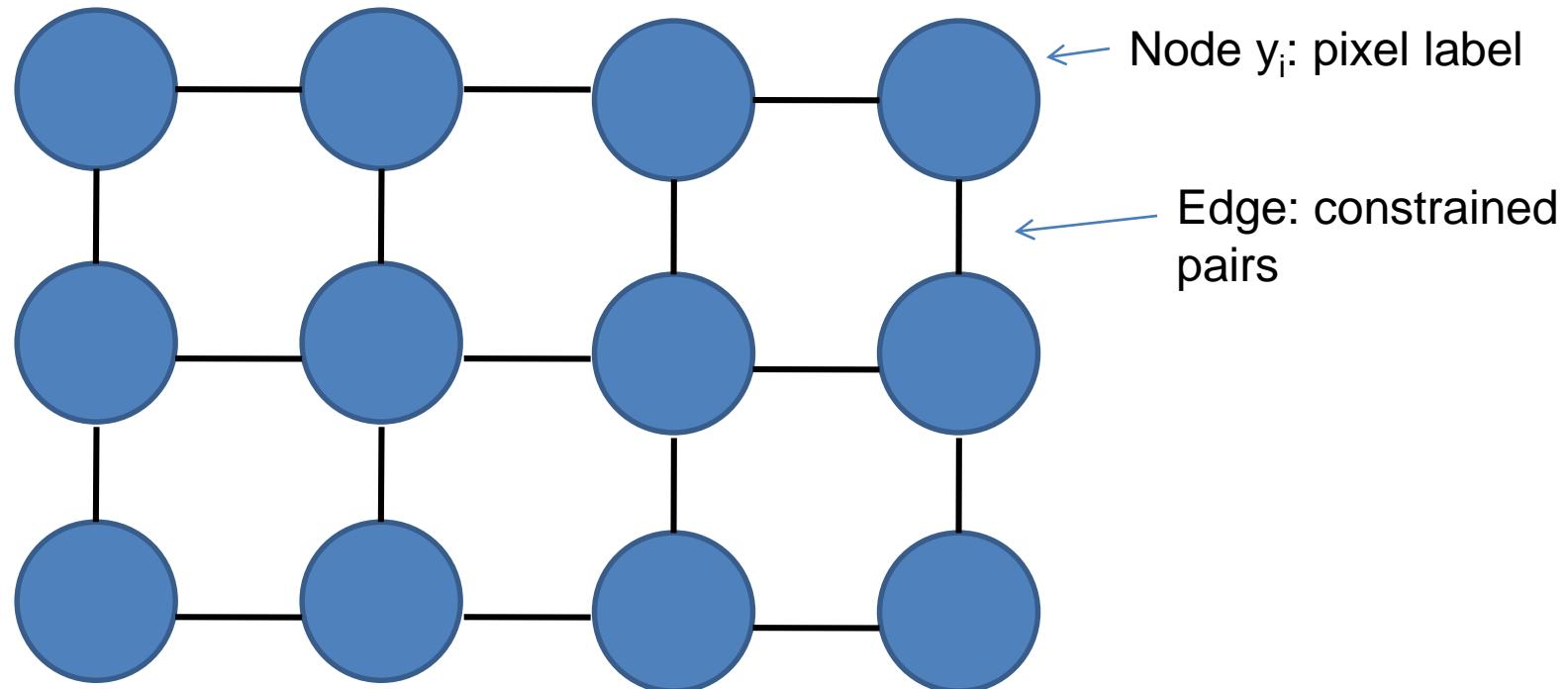
Gaussian's symmetry
could mislead a little...
(Brighter-only example)

Adding spatial relations

Markov Random Fields

- Markov chains have 1D structure
 - At every time, there is one state.
 - This enabled use of dynamic programming.
- Markov Random Fields break this 1D structure.
 - Field of sites, each of which has a label, simultaneously.
 - Label at one site dependent on others, no 1D structure to dependencies.
 - This means no optimal, efficient algorithms, except for 2-label problems.

Markov Random Fields



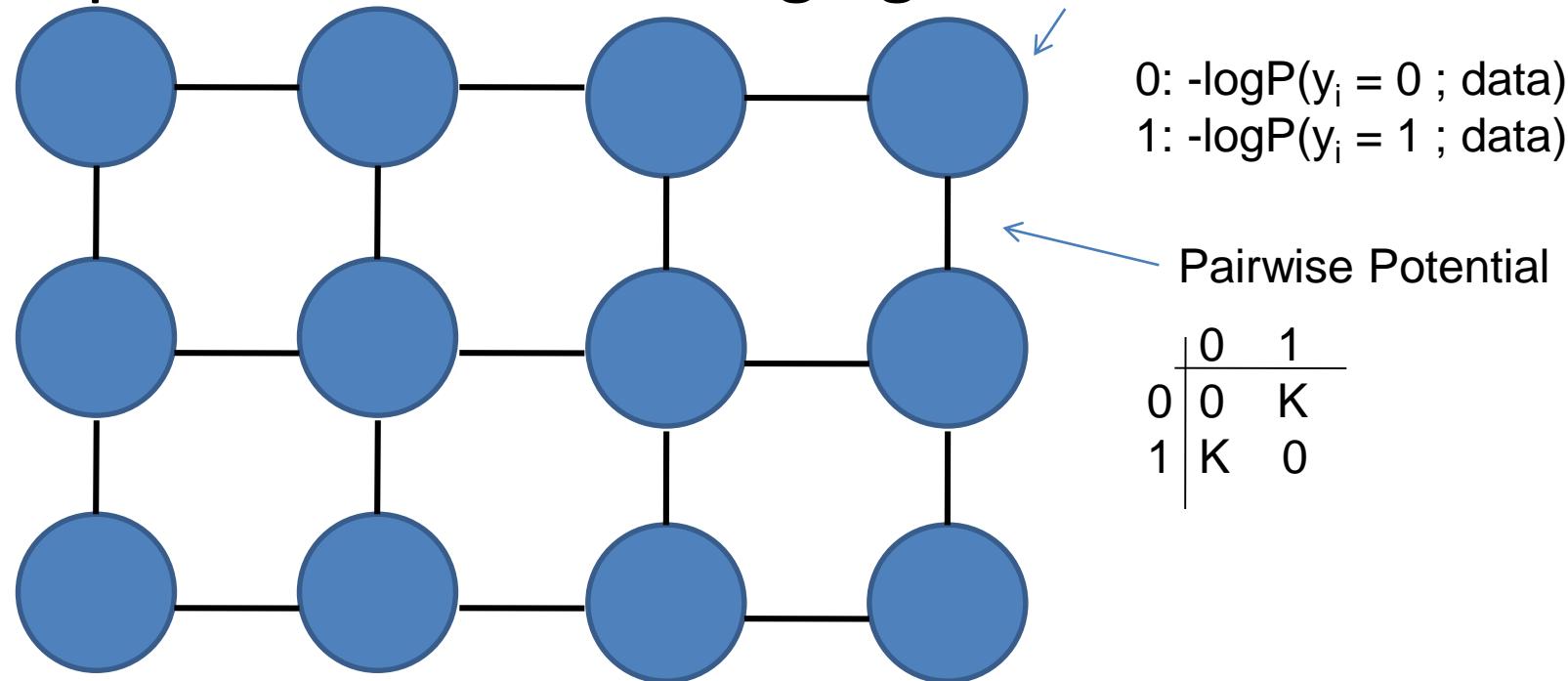
Cost to assign a label to
each pixel

Cost to assign a pair of labels to
connected pixels

$$Energy(\mathbf{y}; \theta, data) = \sum_i \psi_1(y_i; \theta, data) + \sum_{i, j \in edges} \psi_2(y_i, y_j; \theta, data)$$

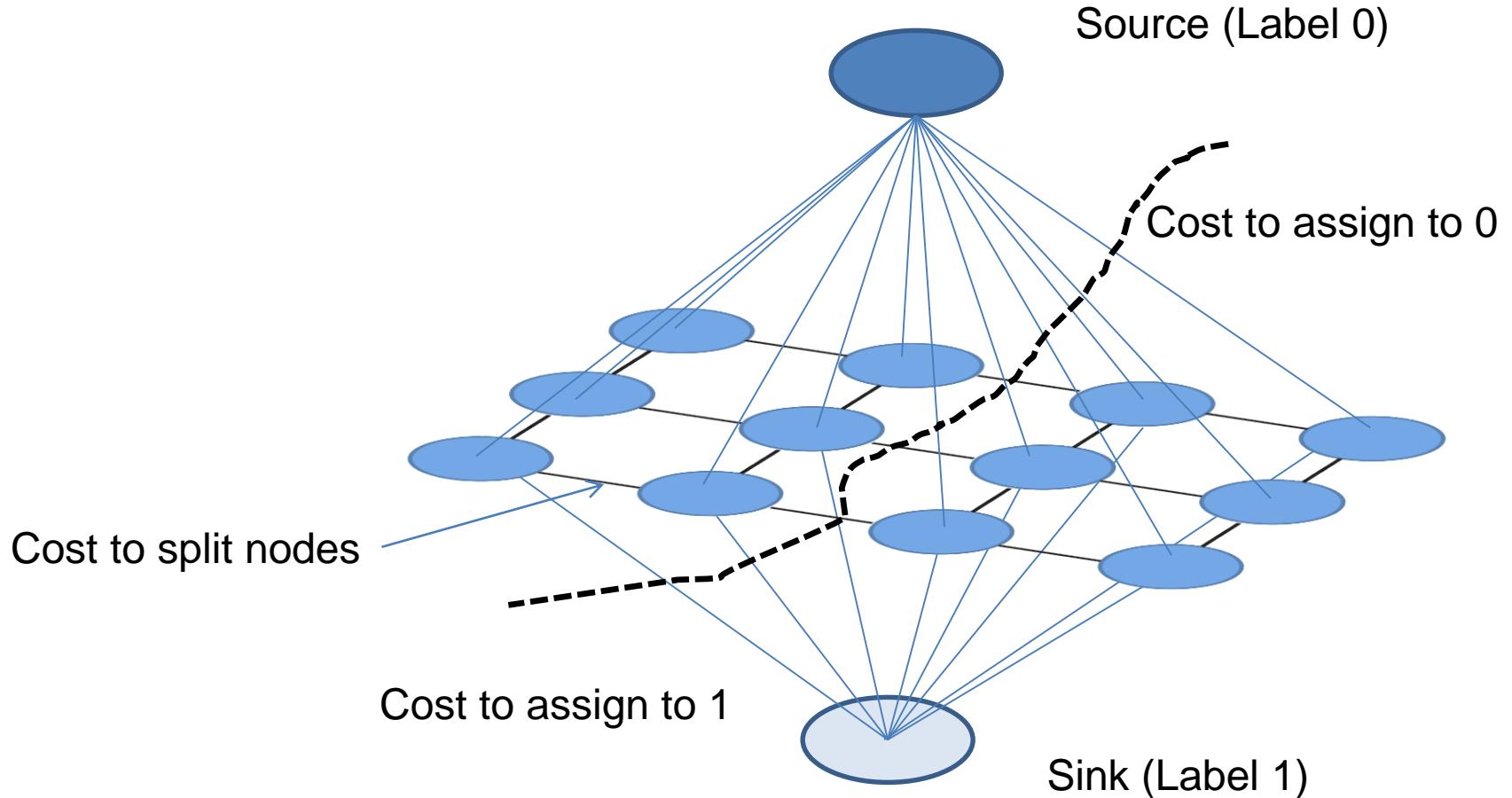
Markov Random Fields

- Example: “label smoothing” grid



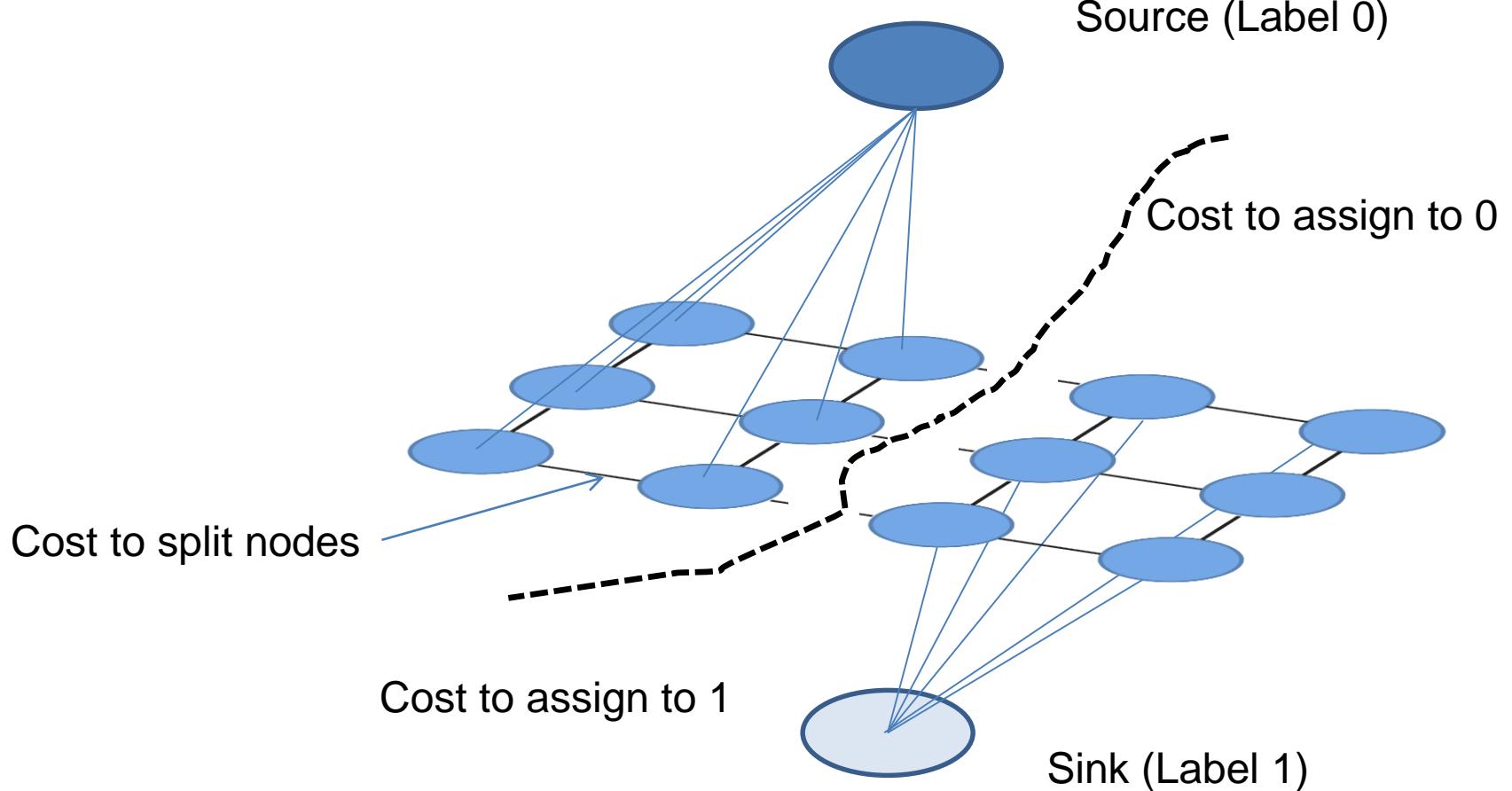
$$\text{Energy}(\mathbf{y}; \theta, \text{data}) = \sum_i \psi_1(y_i; \theta, \text{data}) + \sum_{i, j \in \text{edges}} \psi_2(y_i, y_j; \theta, \text{data})$$

Solving MRFs with graph cuts



$$Energy(\mathbf{y}; \theta, data) = \sum_i \psi_1(y_i; \theta, data) + \sum_{i, j \in edges} \psi_2(y_i, y_j; \theta, data)$$

Solving MRFs with graph cuts

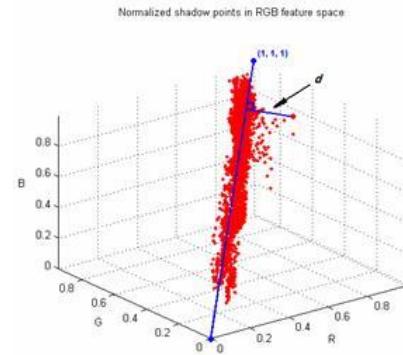
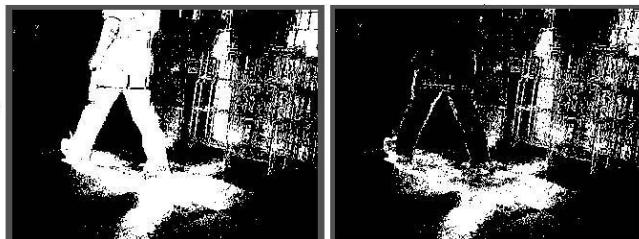


$$Energy(\mathbf{y}; \theta, data) = \sum_i \psi_1(y_i; \theta, data) + \sum_{i, j \in edges} \psi_2(y_i, y_j; \theta, data)$$

Foreground-Background segmentation

The code does the following:

- background RGB Gaussian model training (from many images)
- shadow modeling (hard shadow & soft shadow).



- Graphcut foreground-background segmentation

Foreground-Background segmentation



Background Image

Foreground Image

Background Weight

Shadow Weight

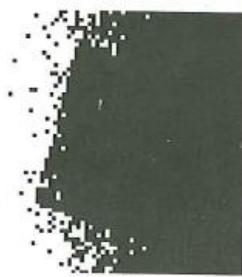
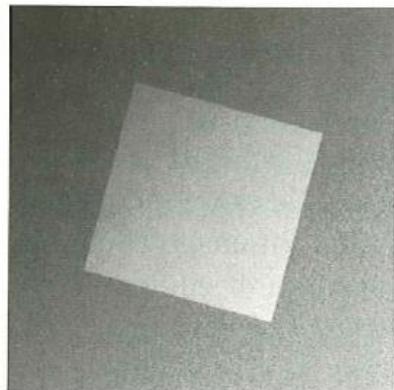
Foreground Result

Graphcut (non-black)
Blob finding (white)

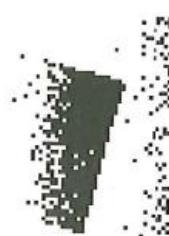
Foreground-Background segmentation



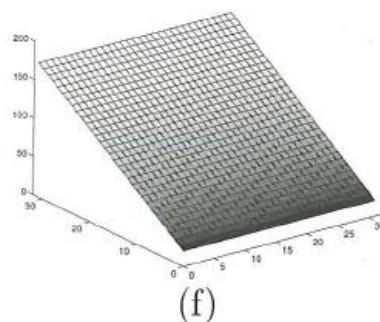
Inclusion criteria ?



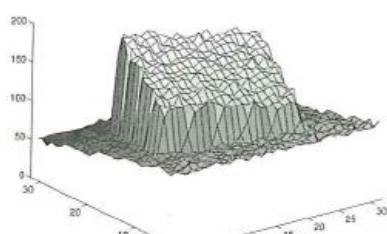
(d)



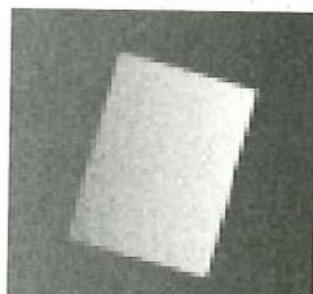
(e)



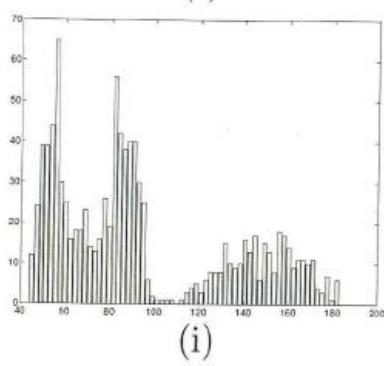
(f)



(g)



(h)



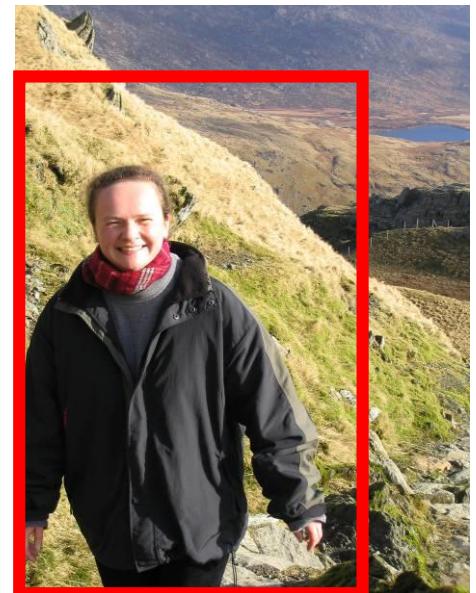
(i)

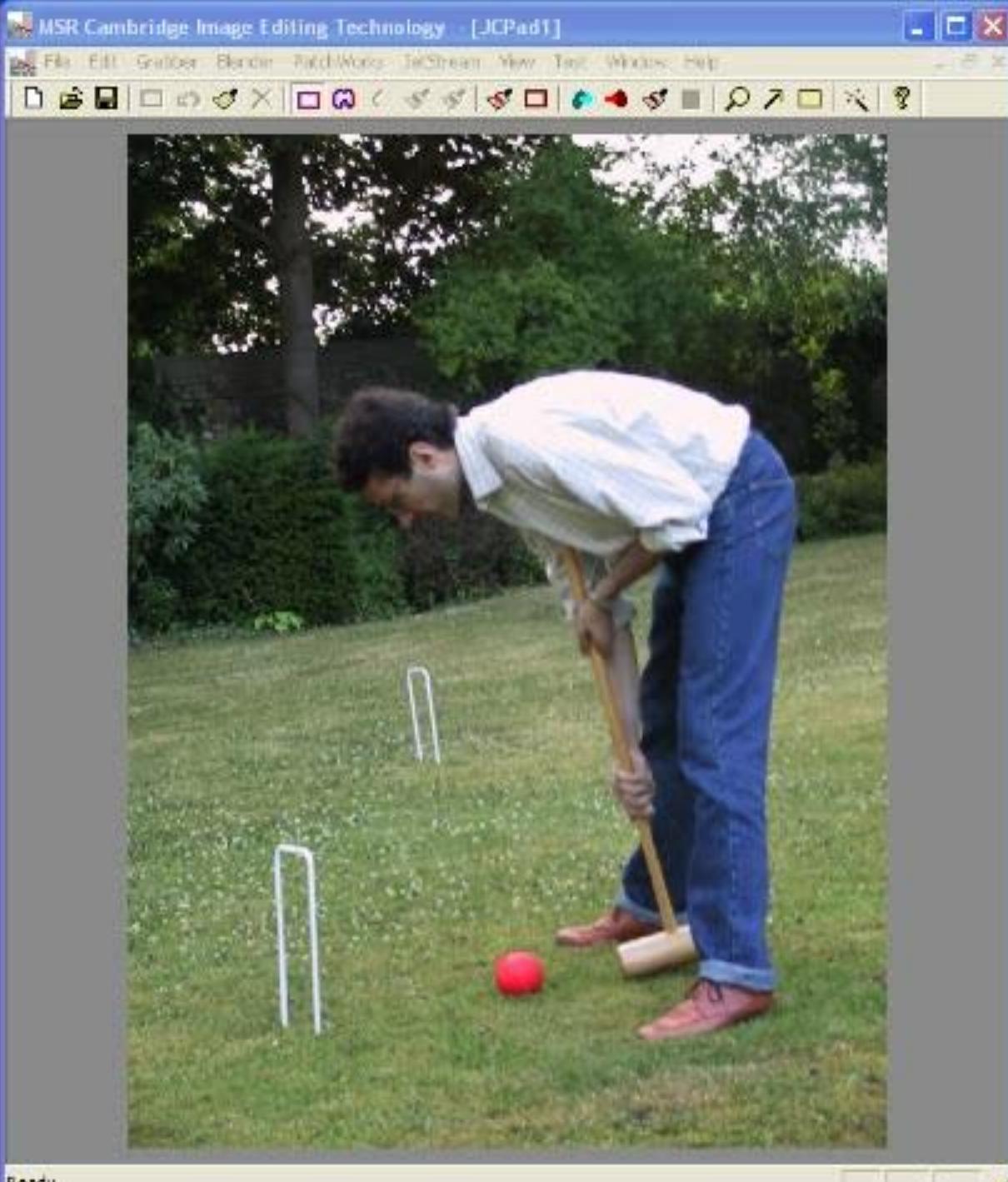


(j)

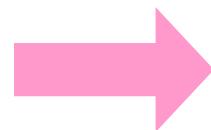
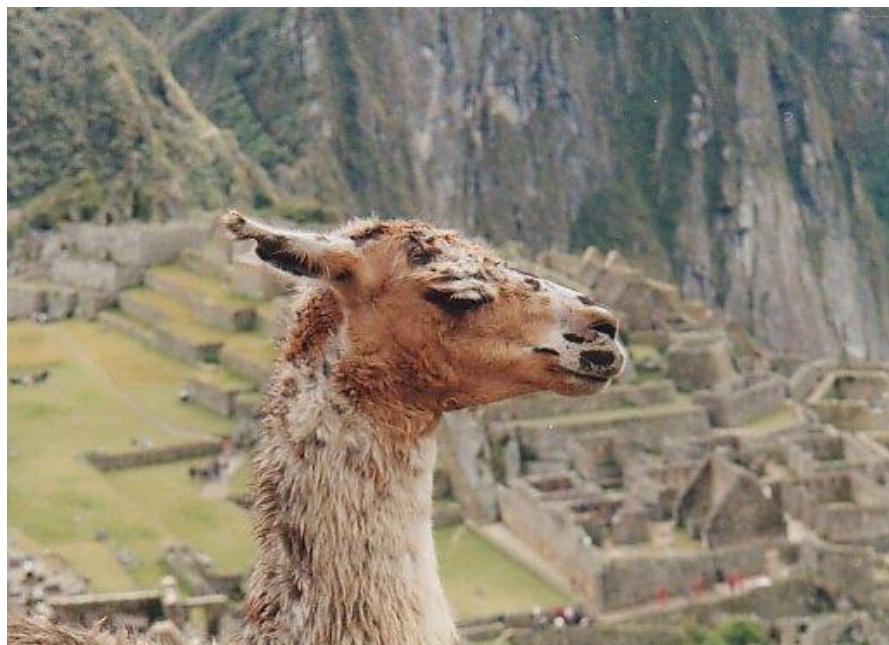
e.g. $(a.l+b.x+c.y+d)^2 < \text{threshold}$
(keep refitting a,b,c to included pts)

GrabCut – interactive foreground segmentation





Problem



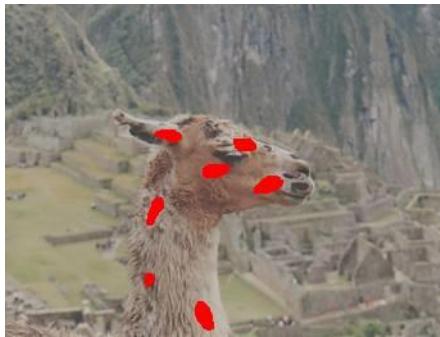
Fast &
Accurate ?



What GrabCut does

User
Input

Magic Wand
(198?)



Regions

Intelligent Scissors
Mortensen and Barrett (1995)



Boundary

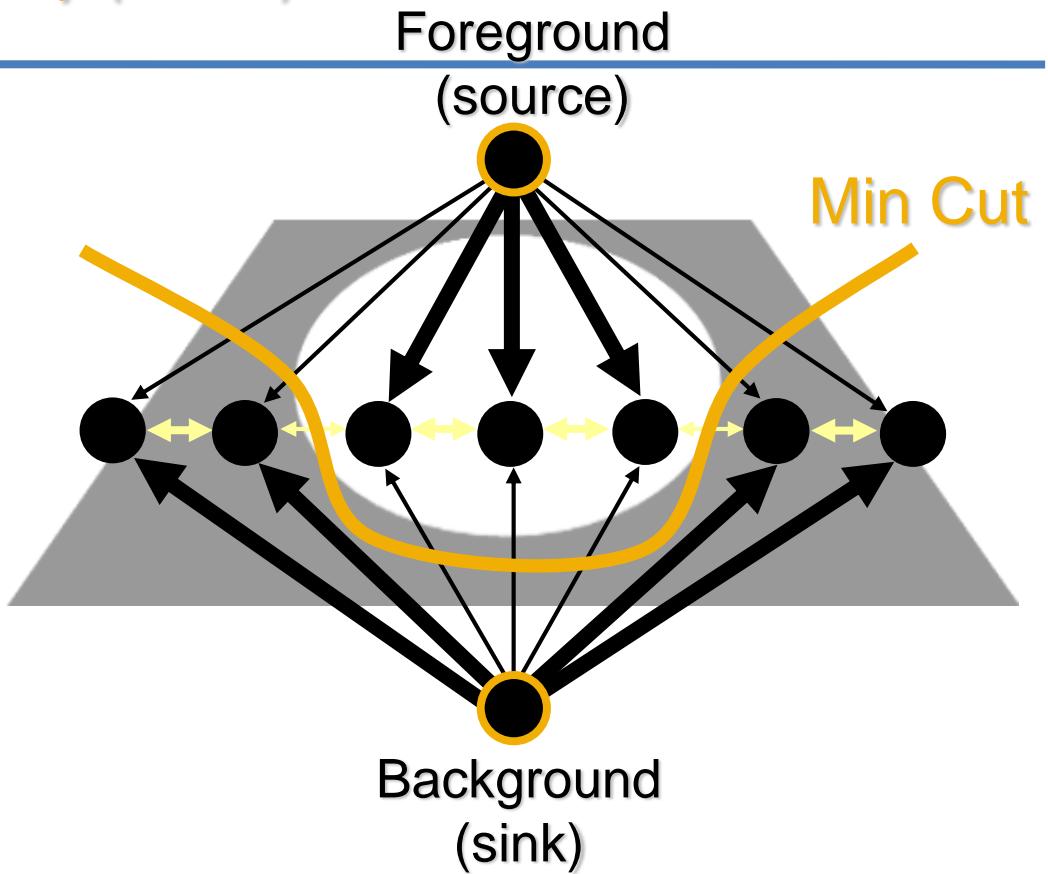
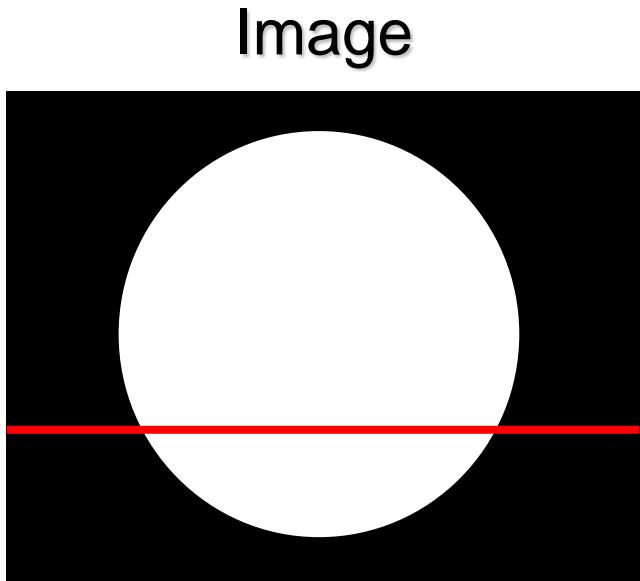
GrabCut



Regions & Boundary

Graph Cuts

Boykov and Jolly (2001)



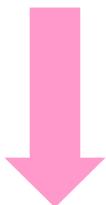
Cut: separating source and sink; Energy: collection of edges

Min Cut: Global minimal energy in polynomial time

Iterated Graph Cut



User Initialisation



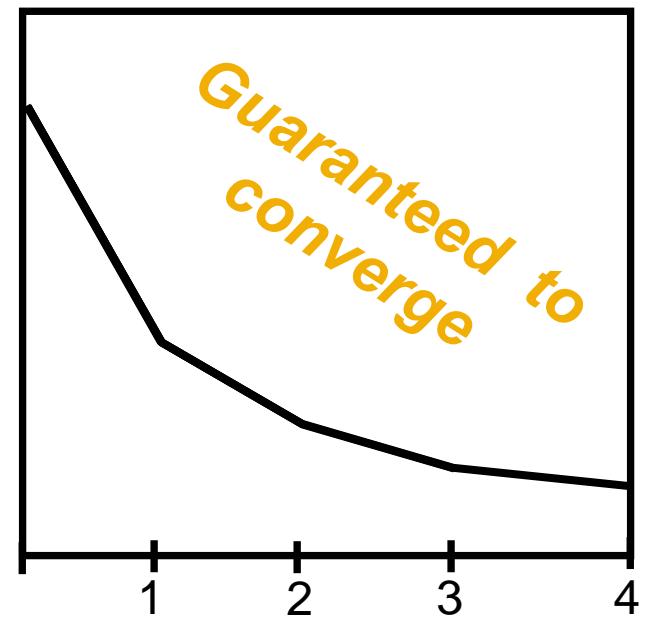
K-means for learning
colour distributions

Graph cuts to
infer the
segmentation

Iterated Graph Cuts



Result

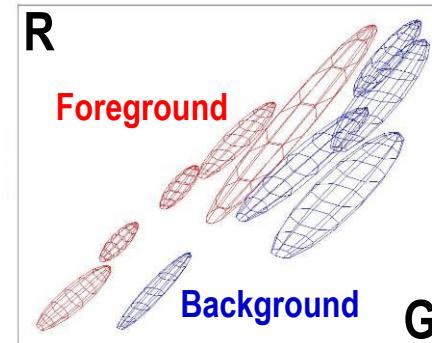
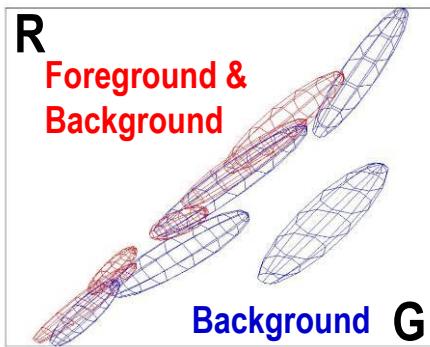


Energy after each Iteration

Colour Model



Iterated
graph cut



Gaussian Mixture Model (typically 5-8 components)

Moderately straightforward examples



... GrabCut completes automatically

Difficult Examples

Camouflage &
Low Contrast



Initial
Rectangle

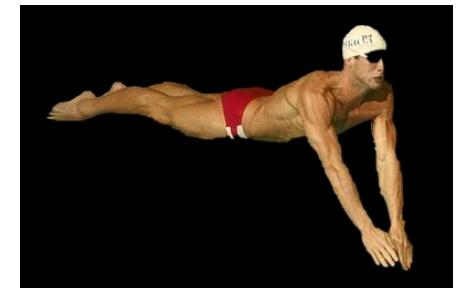
Fine structure



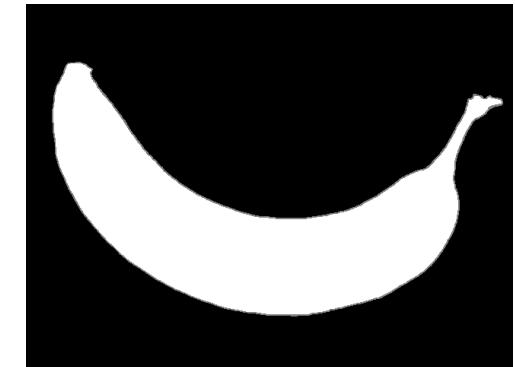
No telepathy



Initial
Result



Evaluation – Labelled Database



Available online: <http://research.microsoft.com/vision/cambridge/segmentation/>

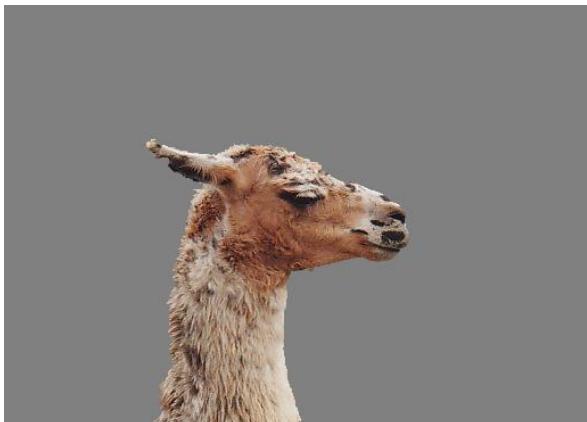
Comparison

Boykov and Jolly (2001)

User
Input

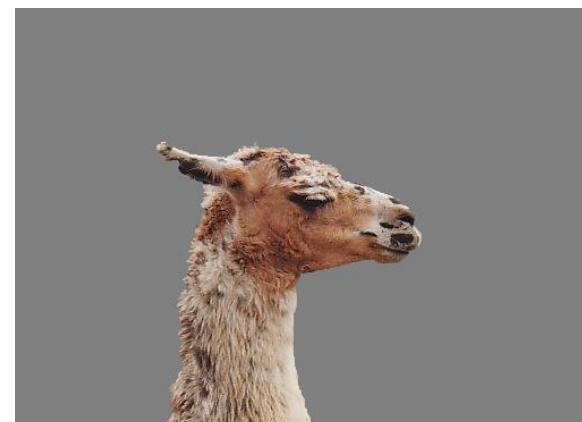
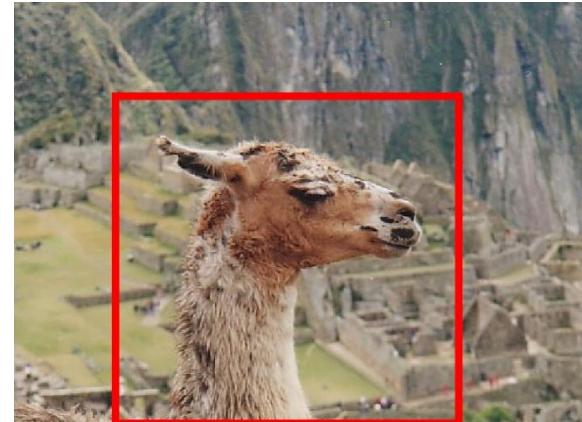


Result



Error Rate: 0.72%

GrabCut



Error Rate: 0.72%

Slides from Carsten Rother (MSR)

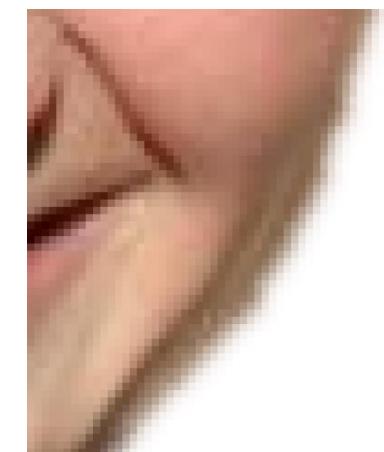
Border Matting



Hard Segmentation

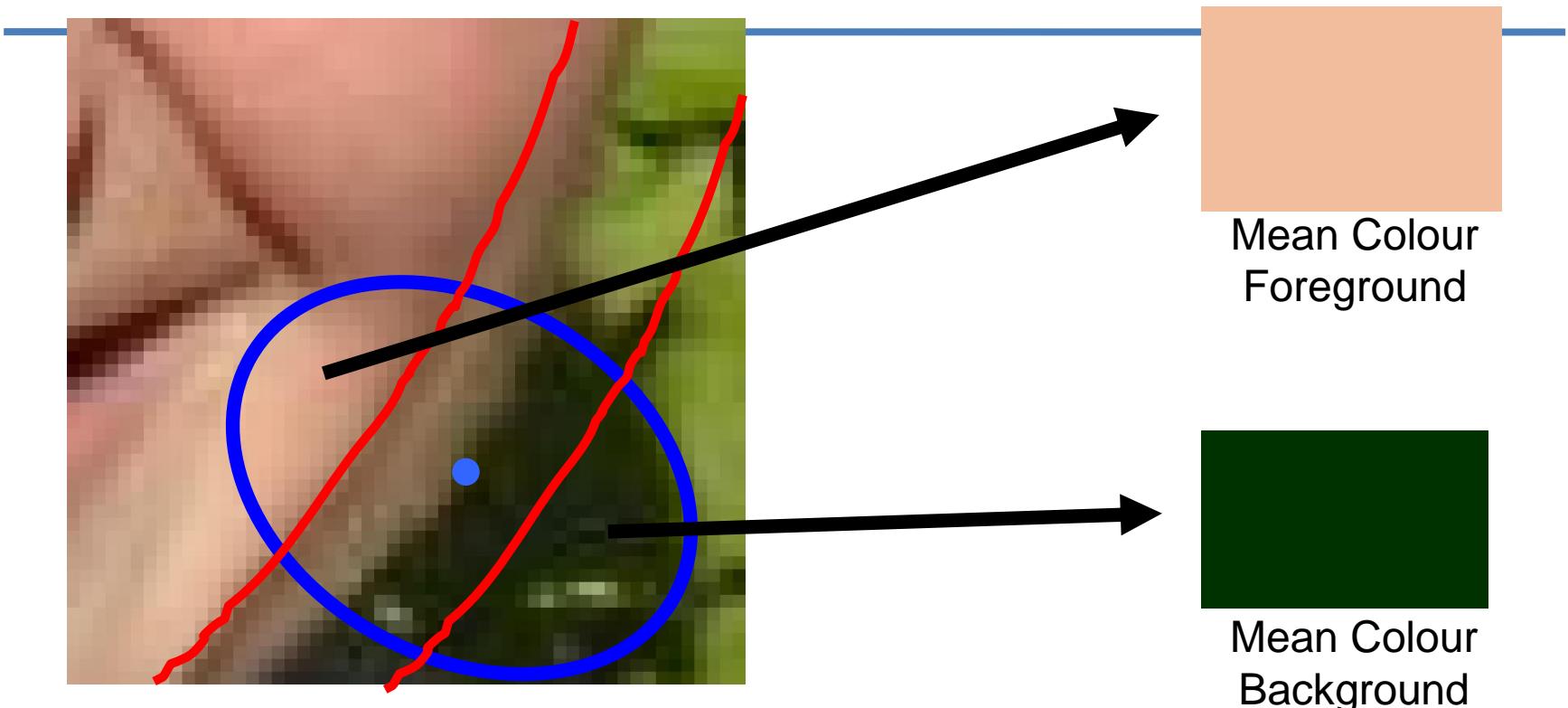


Automatic Trimap



Soft Segmentation

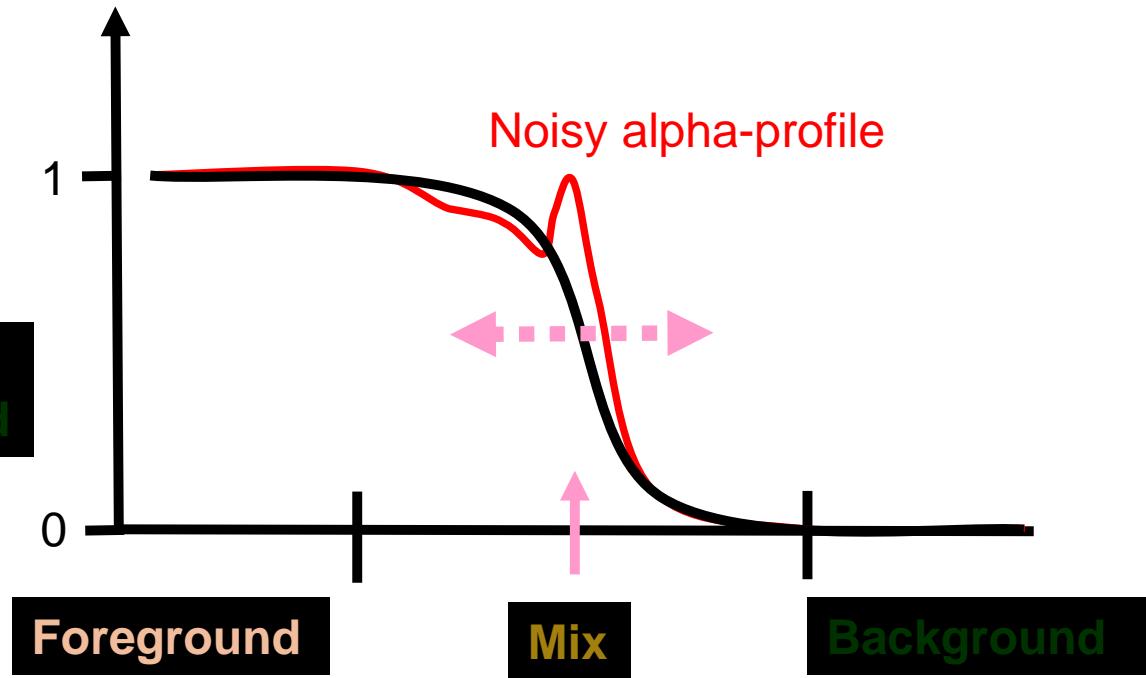
Natural Image Matting



Solve

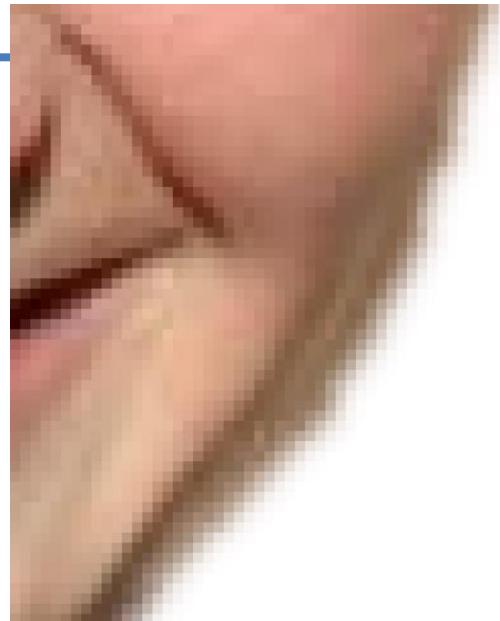
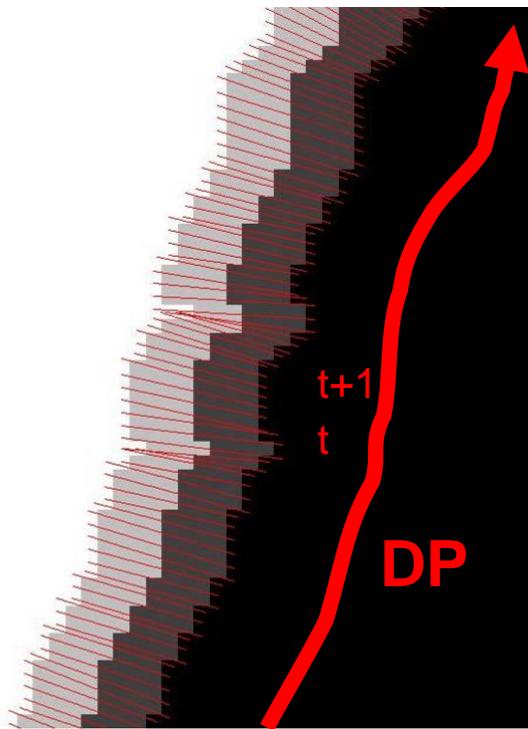
Ruzon and Tomasi (2000): Alpha estimation in natural images

Border Matting



Fit a smooth alpha-profile with parameters

Dynamic Programming

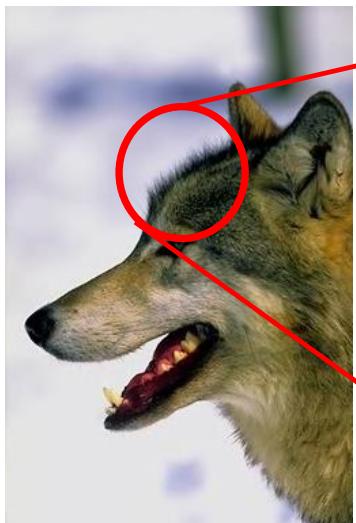
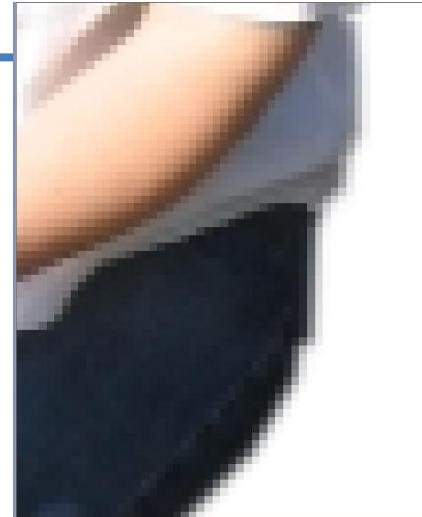


Result using DP Border Matting

Noisy alpha-profile

Regularisation

Results





Switching to Spatial-domain only:

Morphological Operations

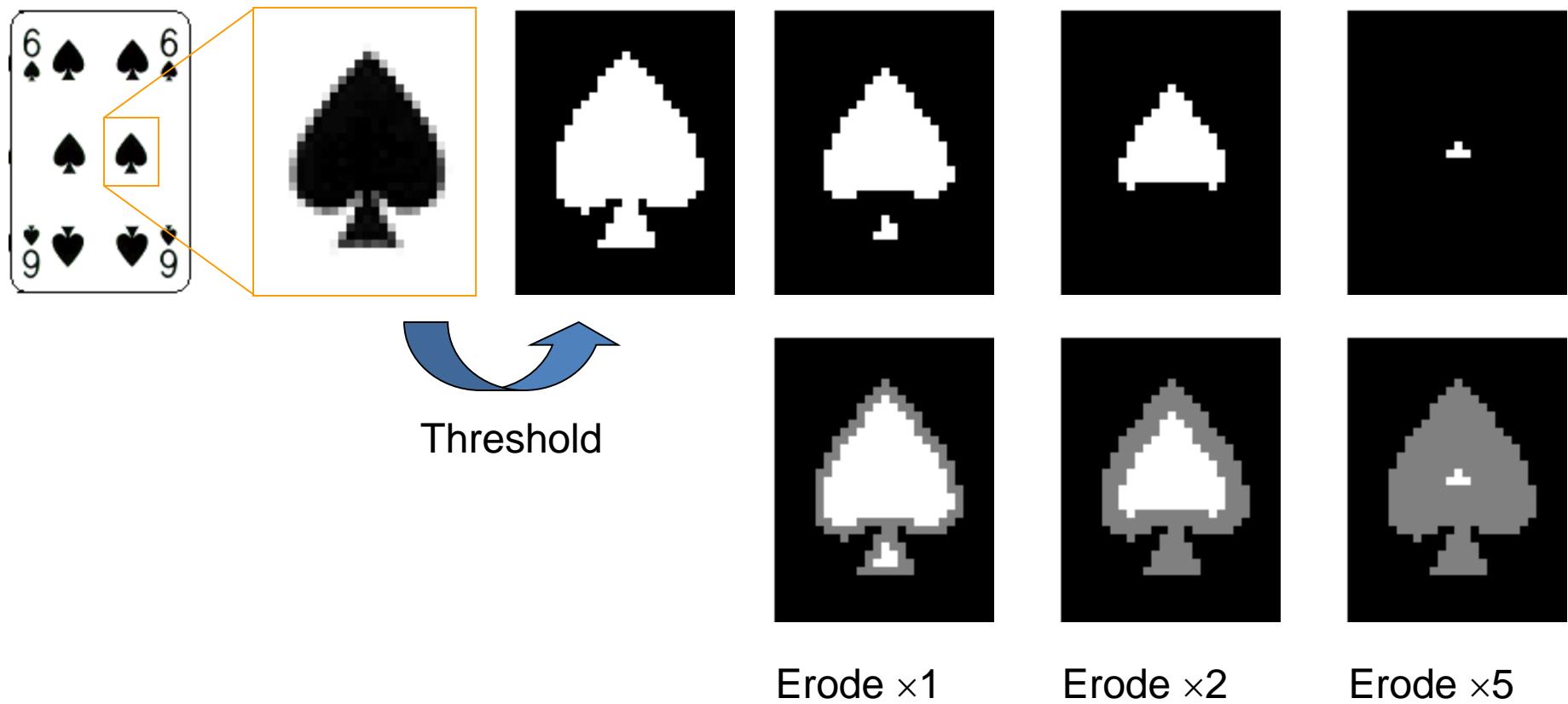
What Are Morphological Operators?

- Local pixel transformations for processing region shapes
- Most often used on binary images
- Logical transformations based on comparison of pixel neighborhoods with a pattern.

Simple Operations - Examples

- Eight-neighbor erode
 - a.k.a. Minkowsky subtraction
- Erase any foreground pixel that has one eight-connected neighbor that is background.

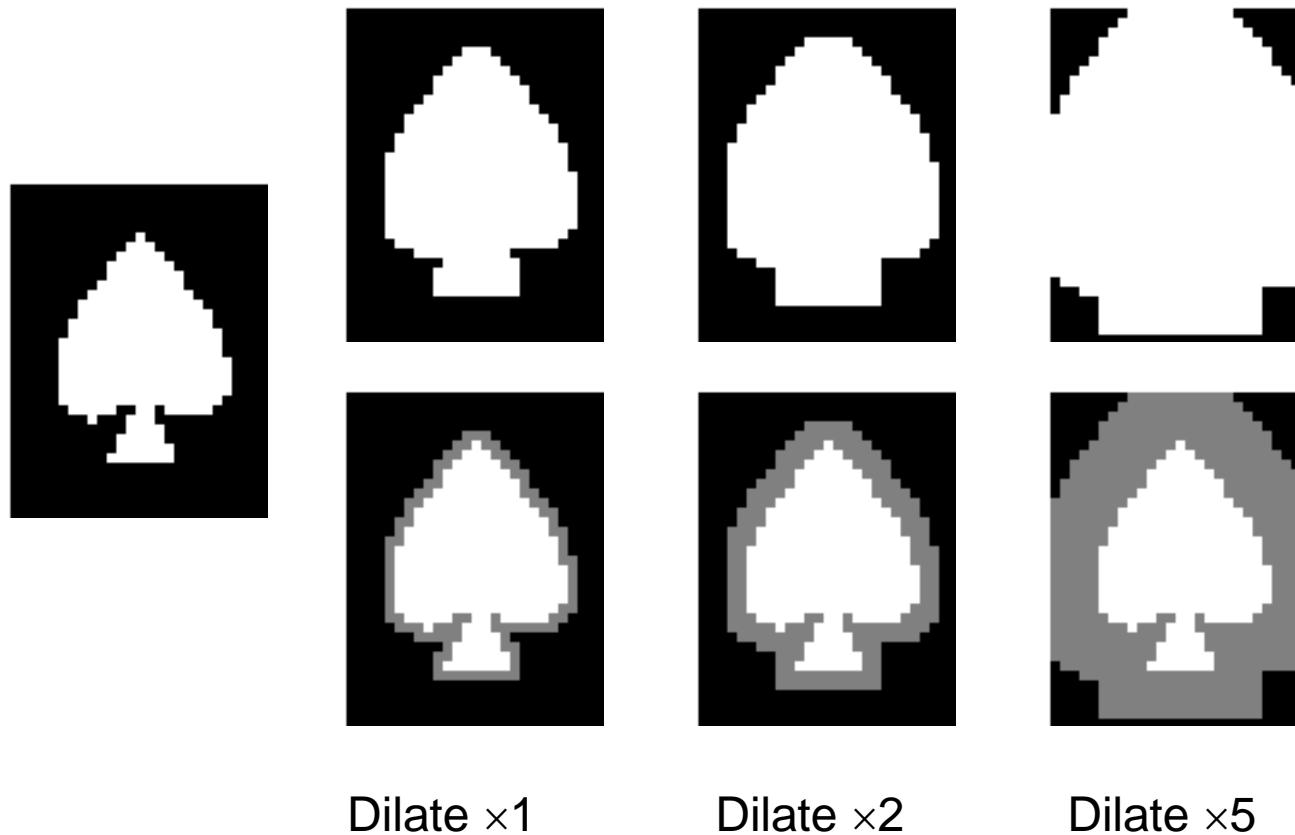
8-neighbor erode



8-neighbor dilate

- Eight-neighbor dilate
 - a.k.a. Minkowsky addition
- Paint any background pixel that has one eight-connected neighbor that is foreground.

8-neighbor dilate



Why?

- Smooth region boundaries for shape analysis.
- Remove noise and artefacts from an imperfect segmentation.
- Match particular pixel configurations in an image for simple object recognition.

Structuring Elements

- Morphological operations take two arguments:
 - A binary image
 - A *structuring element*.
- Compare the structuring element to the neighborhood of each pixel.
- This determines the output of the morphological operation.

Structuring elements

- The structuring element is also a binary array
- A structuring element has an origin

1	1	1
1	1	1
1	1	1

0	1	0
1	1	1
0	1	0

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

Binary images as sets

- We can think of the binary image and the structuring element as sets containing the pixels with value 1.

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

$$I = \{(1,1), (2,1), (3,1), (2,2), (3,2), (4,4)\}$$

Some set notation

- Union and intersection:

$$I_1 \cup I_2 = \{\underline{x} : \underline{x} \in I_1 \text{ or } \underline{x} \in I_2\}$$

$$I_1 \cap I_2 = \{\underline{x} : \underline{x} \in I_1 \text{ and } \underline{x} \in I_2\}$$

- Complement

$$I^C = \{\underline{x} : \underline{x} \notin I\}$$

- Difference

$$I_1 \setminus I_2 = \{\underline{x} : \underline{x} \in I_1 \text{ and } \underline{x} \notin I_2\}$$

- We use ϕ for the empty set .

Fitting, Hitting and Missing

- S fits I at \underline{x} if

$$\{\underline{y} : \underline{y} = \underline{x} + \underline{s}, \underline{s} \in S\} \subset I$$

- S hits I at \underline{x} if

$$\{\underline{y} : \underline{y} = \underline{x} - \underline{s}, \underline{s} \in S\} \cap I \neq \emptyset$$

- S misses I at \underline{x} if

$$\{\underline{y} : \underline{y} = \underline{x} - \underline{s}, \underline{s} \in S\} \cap I = \emptyset$$



Fitting, Hitting and Missing

Image

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

Structuring element

0	1	0
1	1	1
0	1	0

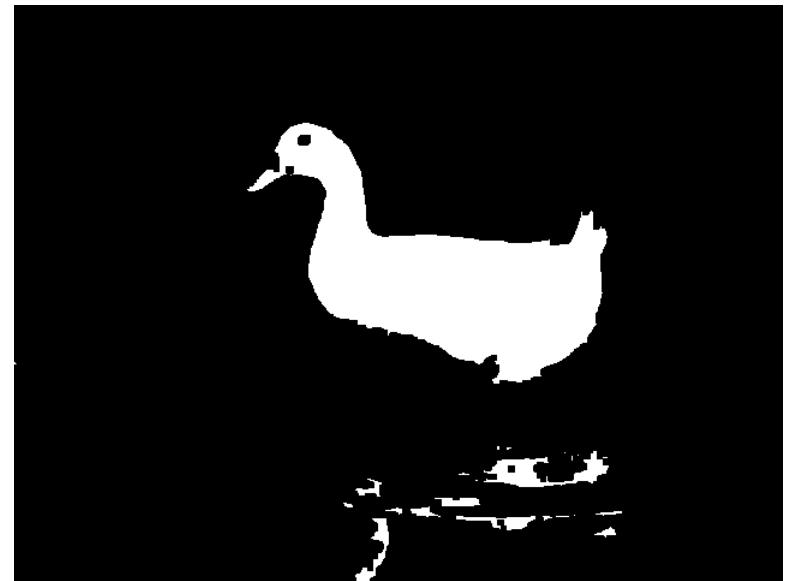
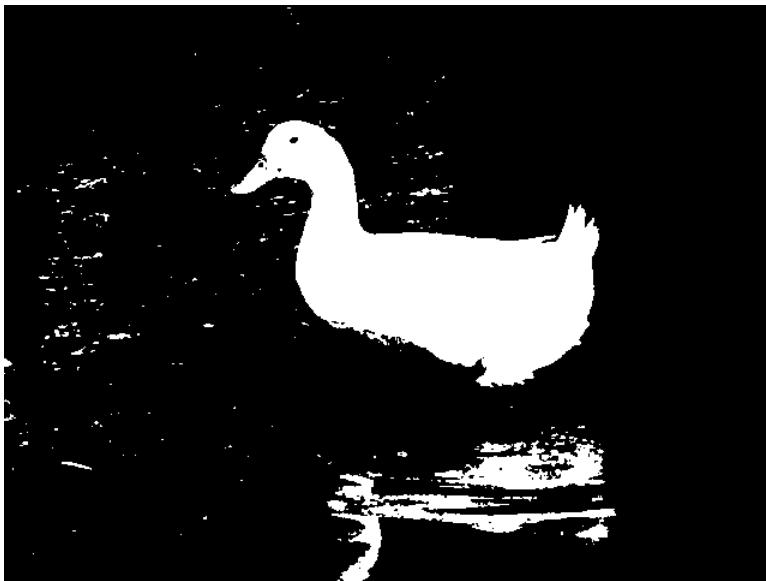
Erosion

- The image $E = I \ominus S$ is the *erosion* of image I by structuring element S .

$$E(\underline{x}) = \begin{cases} 1 & \text{if } S \text{ fits } I \text{ at } \underline{x} \\ 0 & \text{otherwise} \end{cases}$$

$$E = \{\underline{x} : \underline{x} + \underline{s} \in I \text{ for every } s \in S\}$$

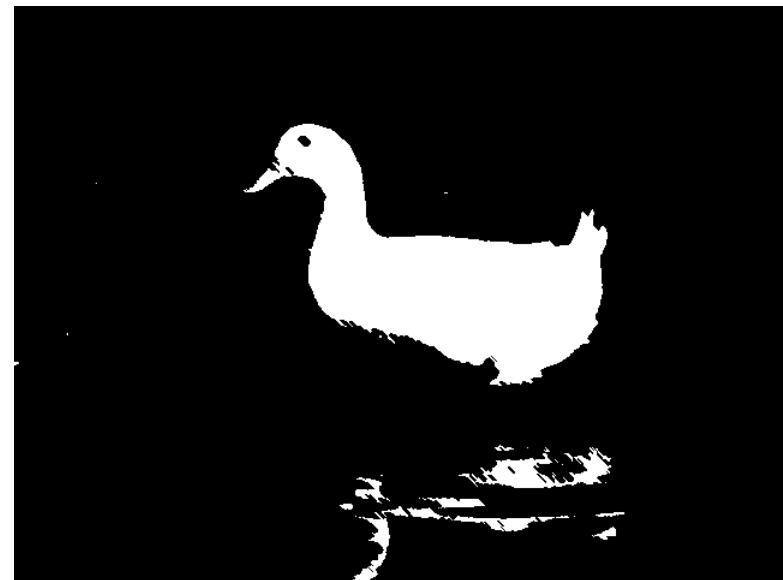
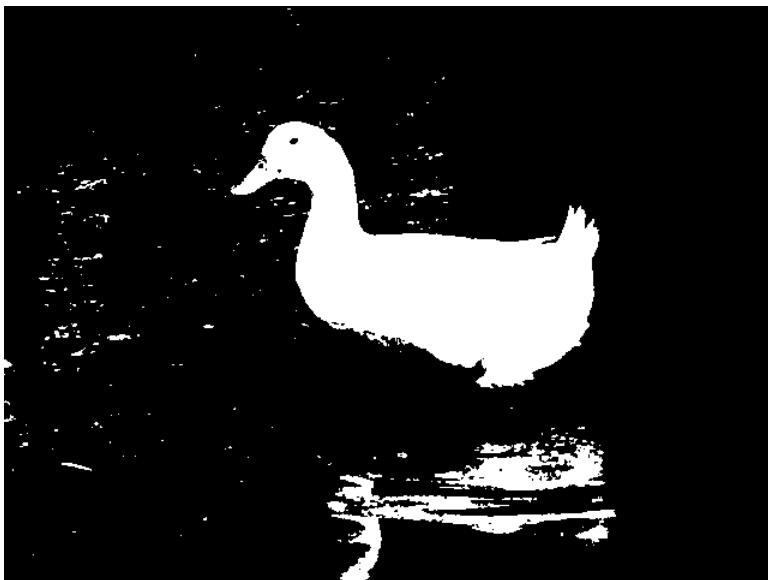
Example



Structuring
element

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

Example



Structuring
element

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

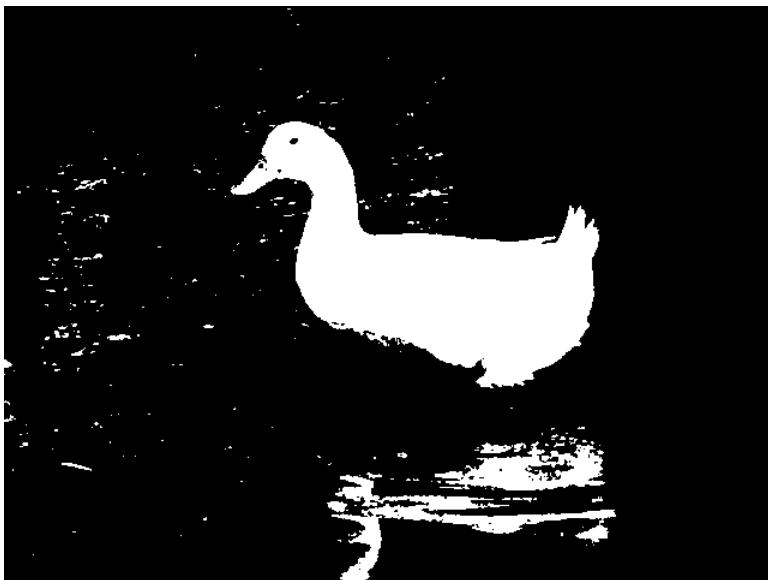
Dilation

- The image $D = I \oplus S$ is the *dilation* of image I by structuring element S .

$$D(\underline{x}) = \begin{cases} 1 & \text{if } S \text{ hits } I \text{ at } \underline{x} \\ 0 & \text{otherwise} \end{cases}$$

$$D = \left\{ \underline{x} : \underline{x} - \underline{s}, \underline{y} \in I \text{ and } \underline{s} \in S \right\}$$

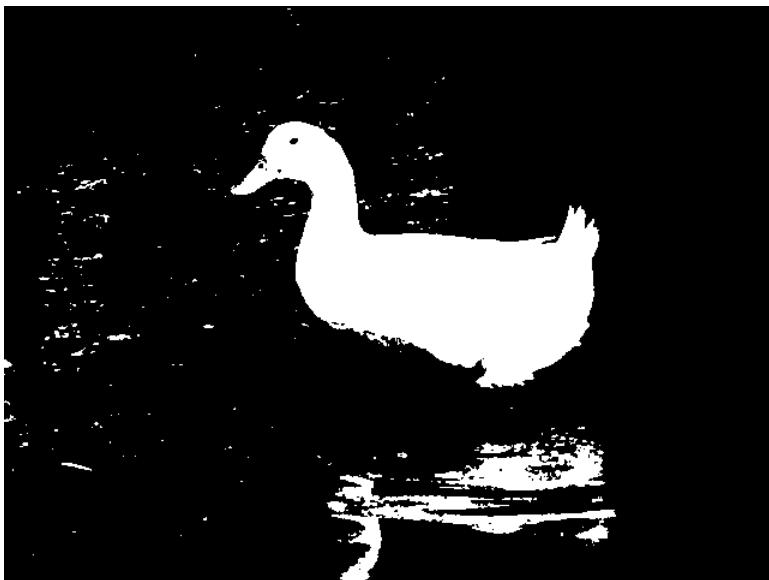
Example



Structuring
element

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

Example



Structuring
element

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

Opening and Closing

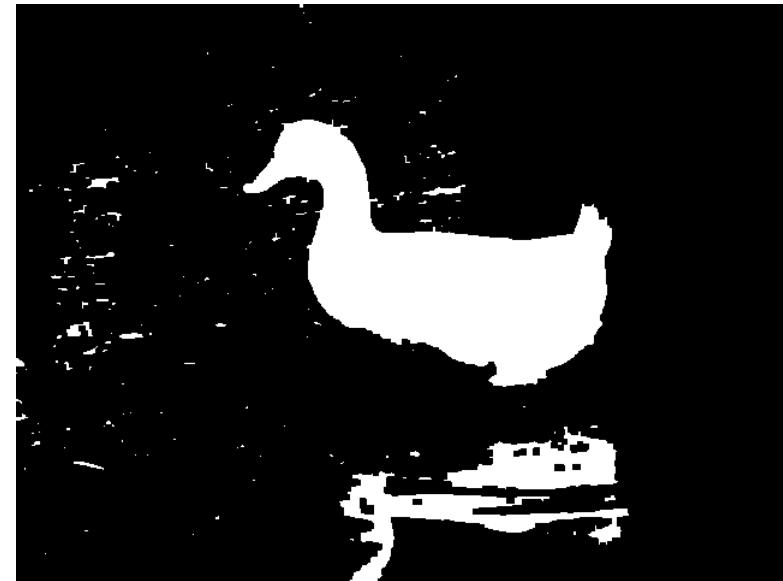
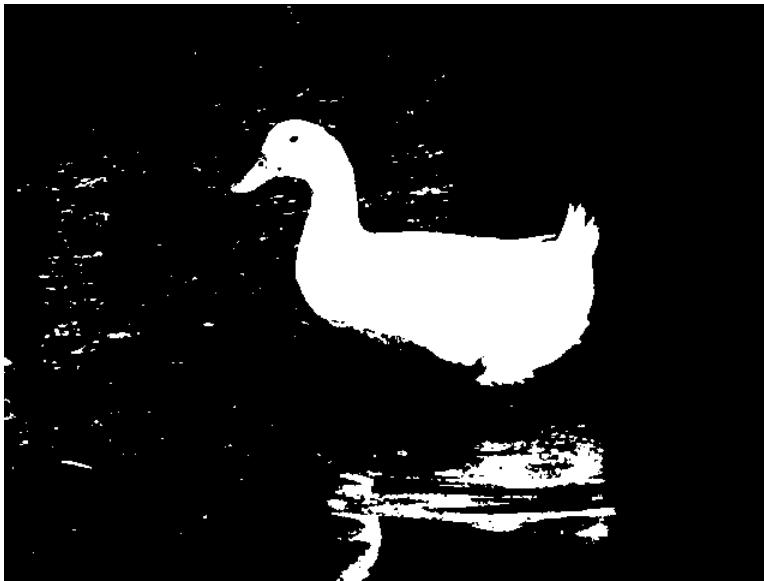
- The *opening* of I by S is

$$I \circ S = (I \ominus S) \oplus S$$

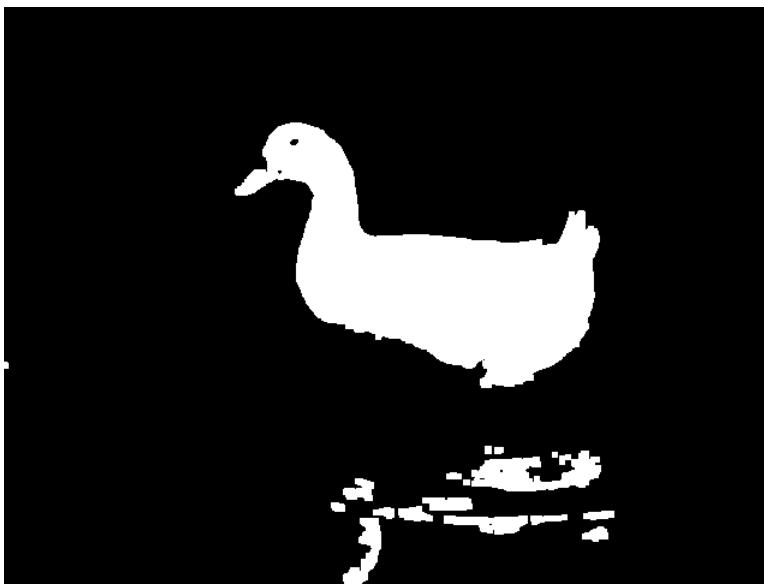
- The *closing* of I by S is

$$I \bullet S = (I \oplus S) \ominus S$$

Example



close



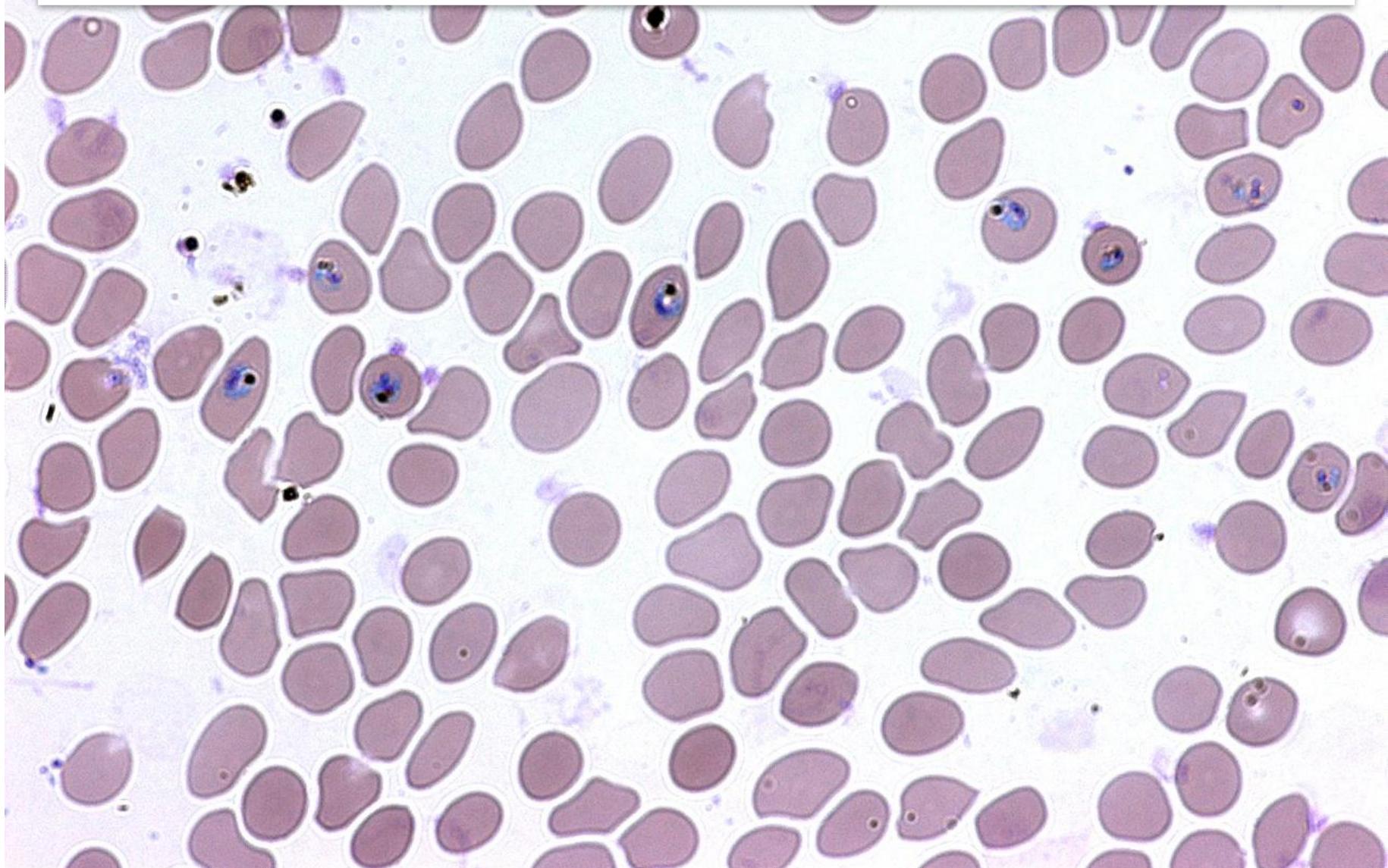
Structuring
element
open

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

Morphological filtering

- To remove holes in the foreground and islands in the background, do both opening and closing.
- The size and shape of the structuring element determine which features survive.
- In the absence of knowledge about the shape of features to remove, use a circular structuring element.

Count the Red Blood Cells

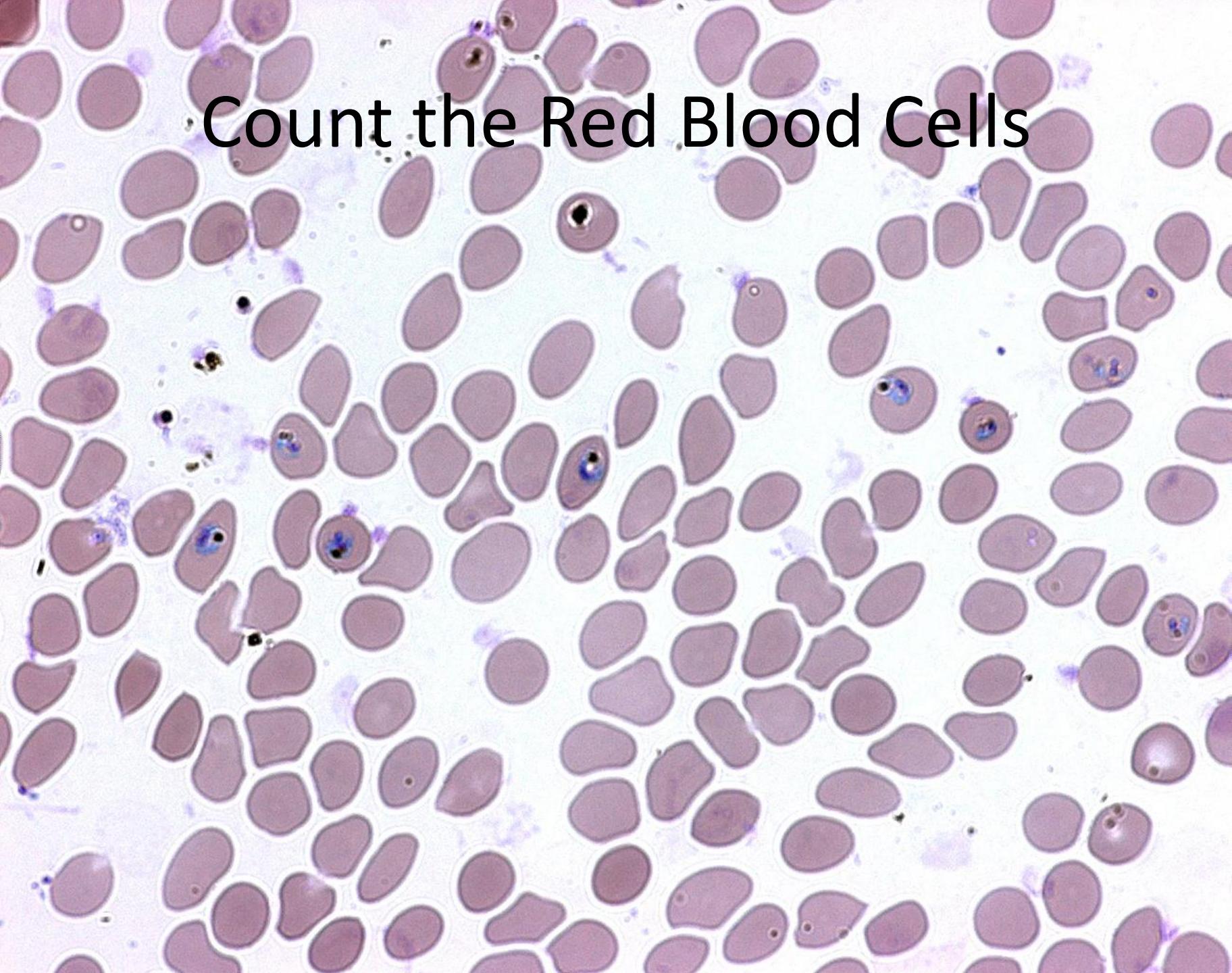


Granulometry

- Provides a size distribution of distinct regions or “granules” in the image.
- We open the image with increasing structuring element size and count the number of regions after each operation.
- Creates a “morphological sieve”

Granulometry

```
def granulo(I, T, maxRad):  
    B = (I > T) # Segment the image I  
    # Open the image at each structuring element size up to a  
    # maximum and count the remaining regions.  
    numRegions = []  
    for x in range(1, maxRad + 1):  
        kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (x, x))  
        O = cv2.morphologyEx(B, cv2.MORPH_OPEN, kernel)  
        numComponents, _ = cv2.connectedComponents(O)  
        numRegions.append(numComponents)  
    return numRegions
```

A microscopic image showing a dense population of red blood cells. The cells are flattened, biconcave discs with a dark purple center and a lighter purple periphery. Some cells contain small, dark blue nuclei. The background is white, and the overall image has a slightly grainy texture.

Count the Red Blood Cells



Threshold and Label



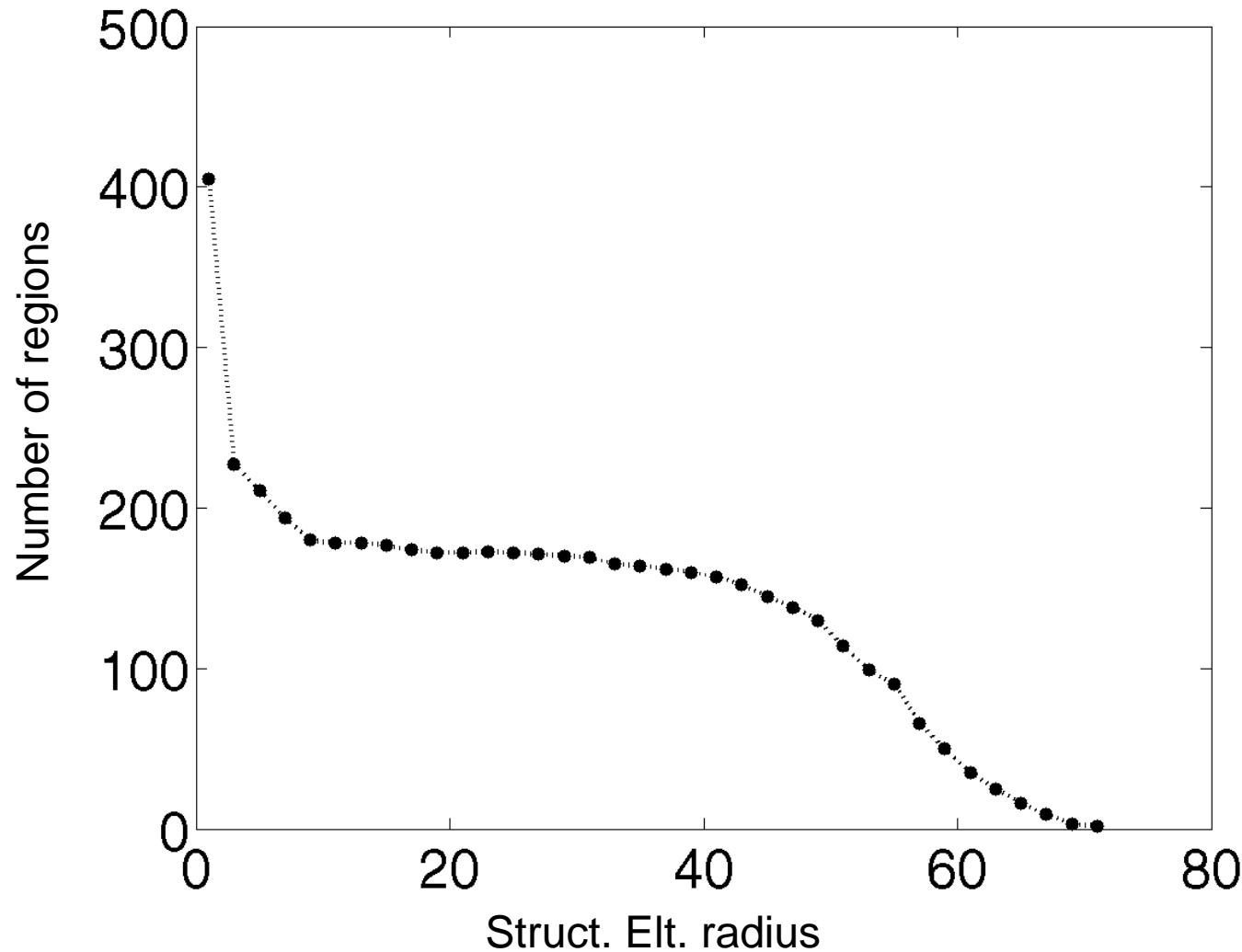
Disc(11)

Disc(19)

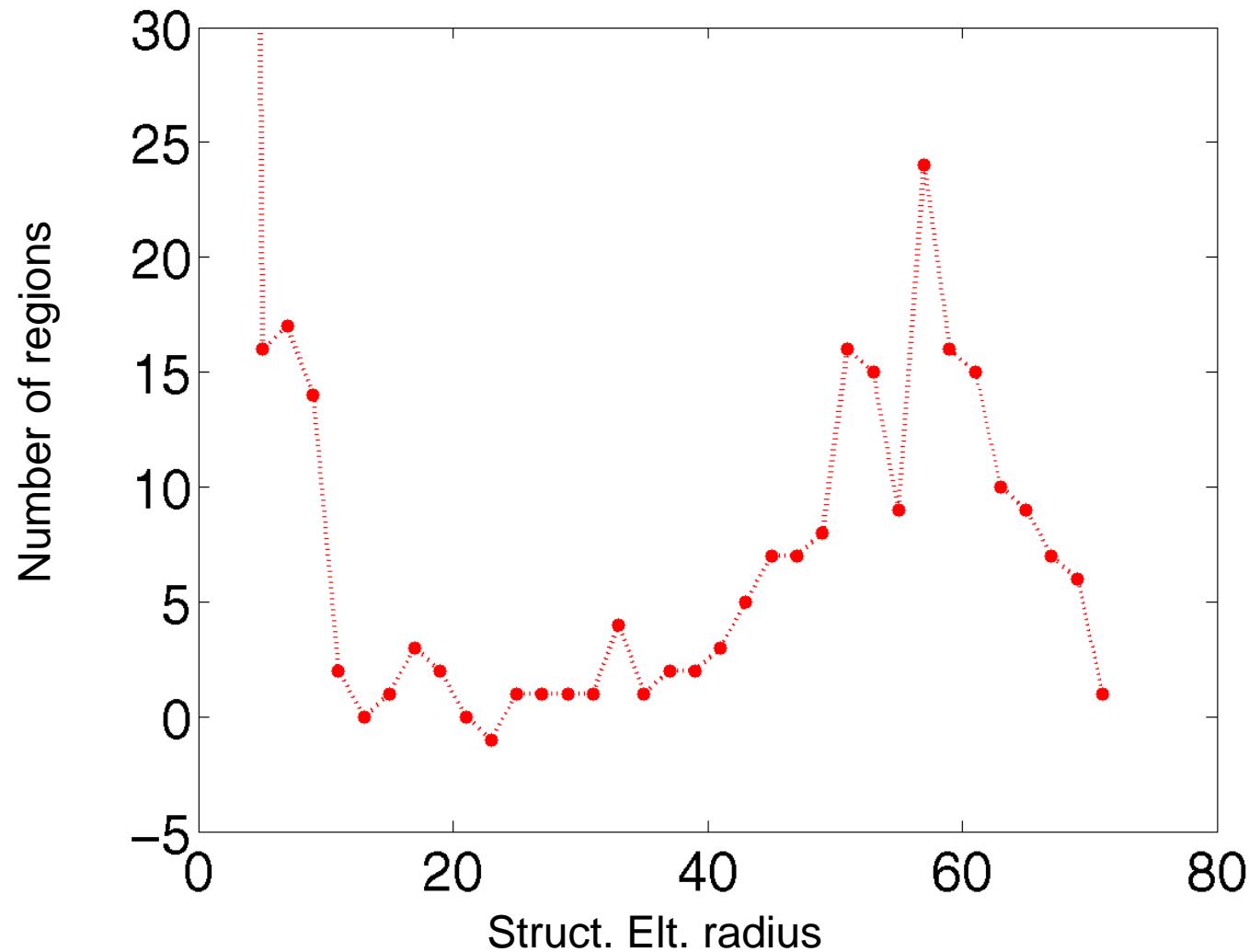


Disc(59)

Number of Regions



Granulometric Pattern Spectrum



Hit-and-miss transform

- Searches for an exact match of the structuring element.
- $H = I \otimes S$ is the hit-and-miss transform of image I by structuring element S .
- Simple form of template matching.

Hit-and-miss transform

$$\begin{array}{|c|c|c|c|c|} \hline 1 & 0 & 1 & 1 & 1 \\ \hline 1 & 1 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 1 & 1 \\ \hline 1 & 1 & 0 & 1 & 1 \\ \hline 1 & 0 & 1 & 0 & 1 \\ \hline \end{array} \otimes \begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|} \hline 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 \\ \hline \end{array}$$

$$\begin{array}{|c|c|c|c|c|} \hline 1 & 0 & 1 & 1 & 1 \\ \hline 1 & 1 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 1 & 1 \\ \hline 1 & 1 & 0 & 1 & 1 \\ \hline 1 & 0 & 1 & 0 & 1 \\ \hline \end{array} \otimes \begin{array}{|c|c|} \hline 1 & 0 \\ \hline * & 1 \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|} \hline 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline \end{array}$$

Upper-Right Corner Detector

x	0	0
1	1	0
x	1	x

0	0	0
1	1	0
0	1	0

J

0	1	1
0	0	1
0	0	0

K

Find the Letter ‘e’

Let freedom ring from every hill and molehill of Mississippi and every mountainside.

Let freedom ring from every hill and molehill of Mississippi and every mountainside.



Let freedom ring from the curvaceous slopes of California.



Thinning and Thickening

- Defined in terms of the hit-and-miss transform:

- The *thinning* of I by S is

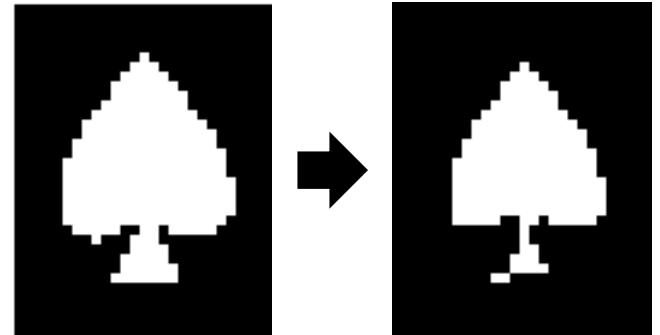
$$I \oslash S = I \setminus (I \otimes S)$$

- The *thickening* of I by S is

$$I \odot S = I \cup (I \otimes S)$$

- Dual operations:

$$(I \odot S)^C = I^C \oslash S$$



0	0	0
	1	
1	1	1

	0	0
1	1	0
	1	

Sequential Thinning/Thickening

- These operations are often performed in sequence with a selection of structuring elements S_1, S_2, \dots, S_n .
- Sequential thinning:

$$I \oslash \{S_i : i = 1, \dots, n\} = (((I \oslash S_1) \oslash S_2) \dots \oslash S_n)$$

- Sequential thickening:

$$I \odot \{S_i : i = 1, \dots, n\} = (((I \odot S_1) \odot S_2) \dots \odot S_n)$$

Sequential Thinning/Thickening

- Several sequences of structuring elements are useful in practice
- These are usually the set of rotations of a single structuring element.
- Sometimes called the *Golay alphabet*.

0	0	0
1		
1	1	1

	0	0
1	1	0
	1	

-		0
-	-	0
-		0

	-	
-	-	0
	0	0

	1	
0	1	1
0	0	

1	1	1
1		
0	0	0

0	0	
0	-	-
-		

0		-1
0	-1	-1
0		-1

Sequential Thinning

- See *morphologyEx* in python.



0 iterations



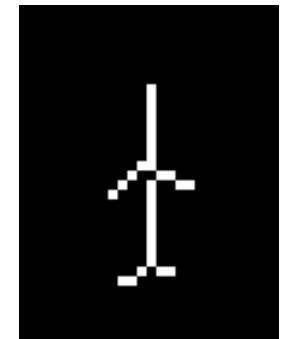
1 iteration



2 iterations

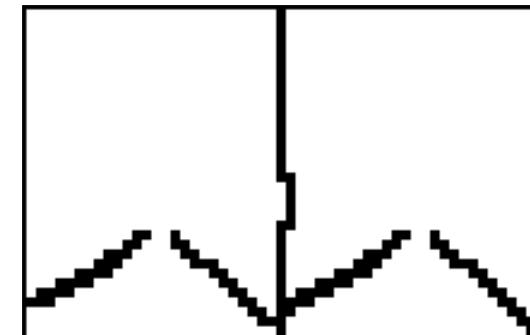


5 iterations



Inf iterations

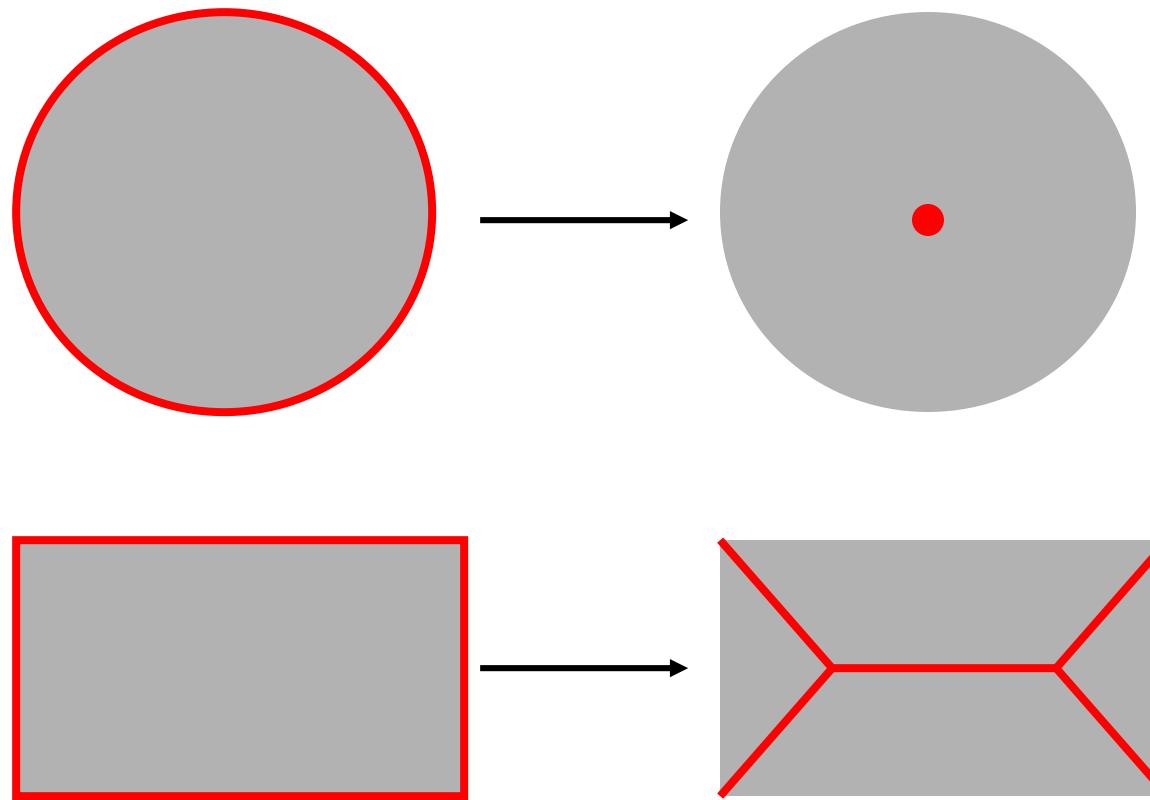
Sequential Thickening



Skeletonization and the Medial Axis Transform

- The skeleton and *medial axis transform* (MAT) are stick-figure representations of a region $X \subset \mathbb{R}^2$.
- Start a grassfire at the boundary of the region.
- The skeleton is the set of points at which two fire fronts meet.

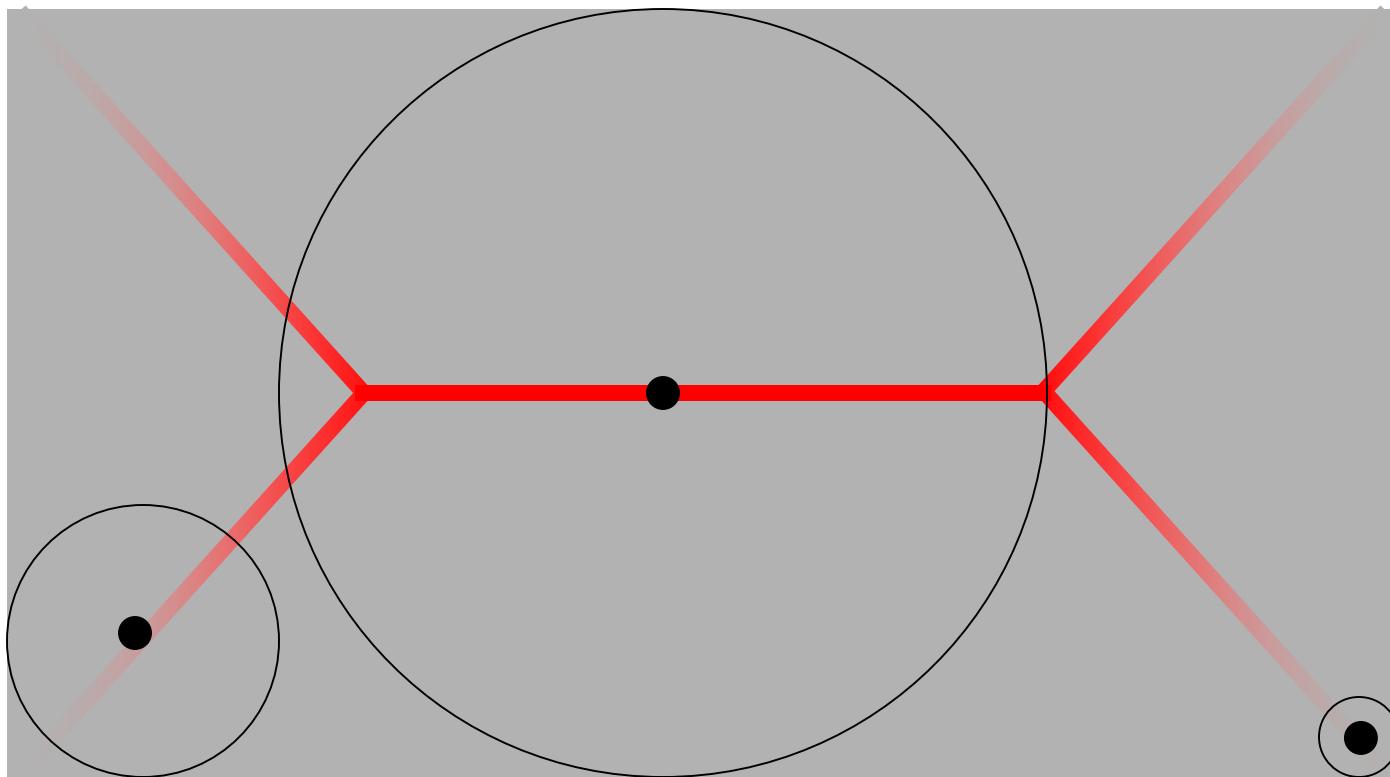
Skeletons



Medial axis transform

- Alternative skeleton definition:
 - The skeleton is the union of centres of maximal discs within X .
 - A *maximal* disc is a circular subset of X that touches the boundary in at least two places.
- The MAT is the skeleton with the maximal disc radius retained at each point.

Medial axis transform



Skeletonization using morphology

- Use structuring element $B =$

0	1	0
1	1	1
0	1	0

- The n -th skeleton subset is

$$S_n(X) = (X \ominus_n B) \setminus [(X \ominus_n B) \circ B]$$

\ominus_n denotes n successive erosions.

- The skeleton is the union of all the skeleton subsets:

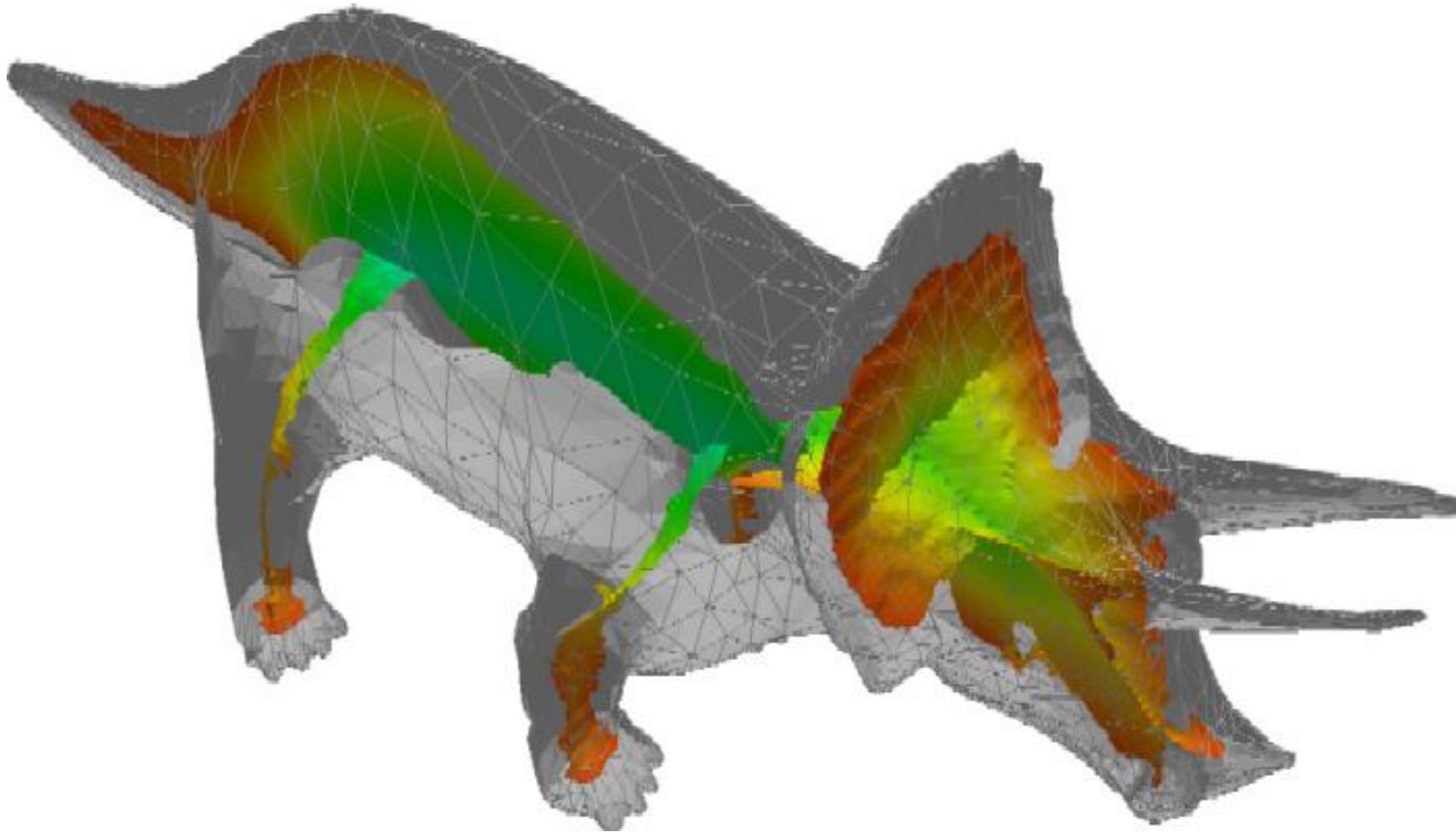
$$S(X) = \bigcup_{n=1}^{\infty} S_n(X)$$

Reconstruction

- We can reconstruct region X from its skeleton subsets:

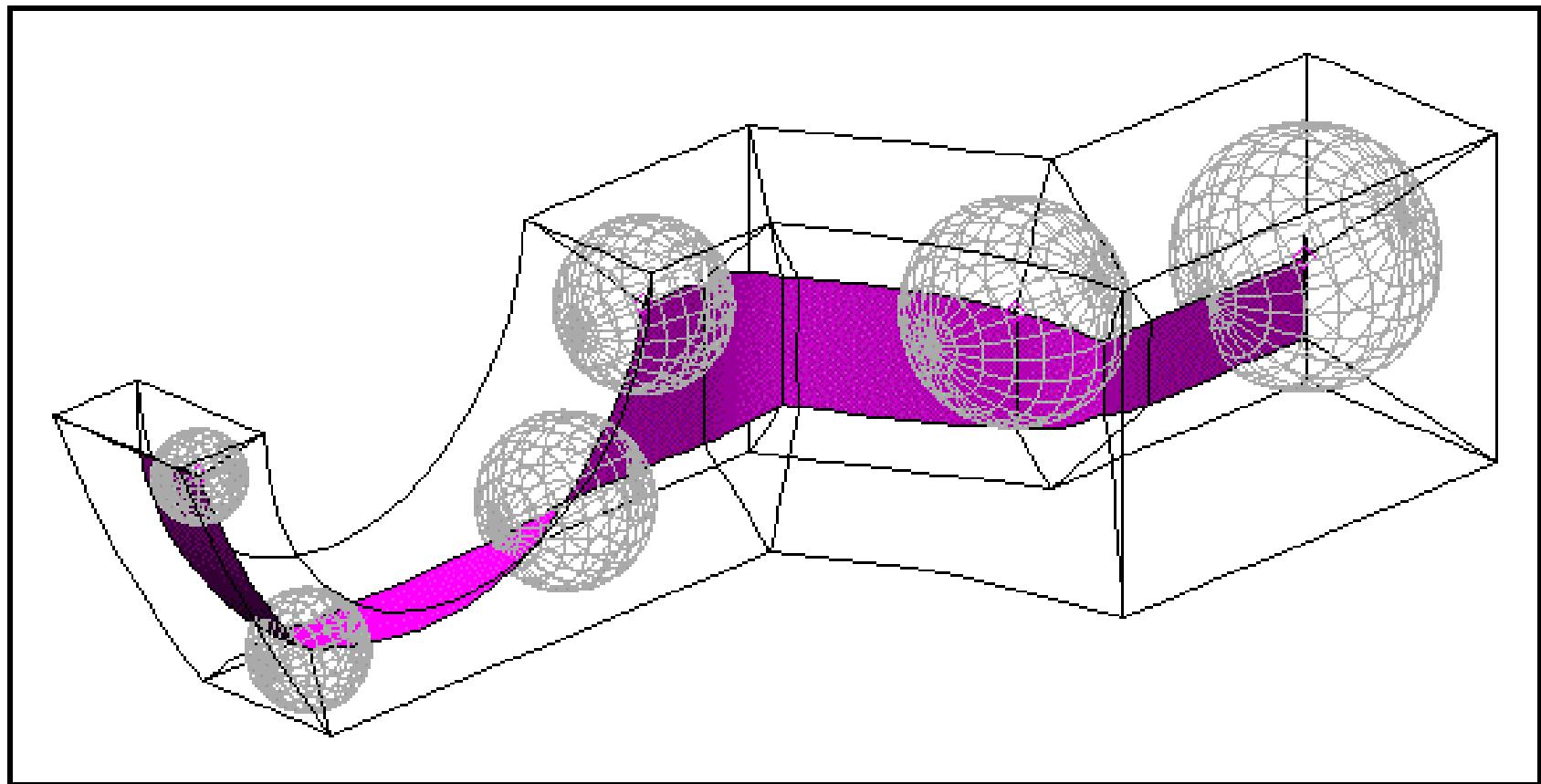
$$X = \bigcup_{n=0}^{\infty} S_n(X) \oplus_n B$$

- We can reconstruct X from the MAT.
- We cannot reconstruct X from $S(X)$.



DiFi: Fast 3D Distance Field Computation Using Graphics Hardware
Sud, Otaduy, Manocha, Eurographics 2004

MAT in 3D



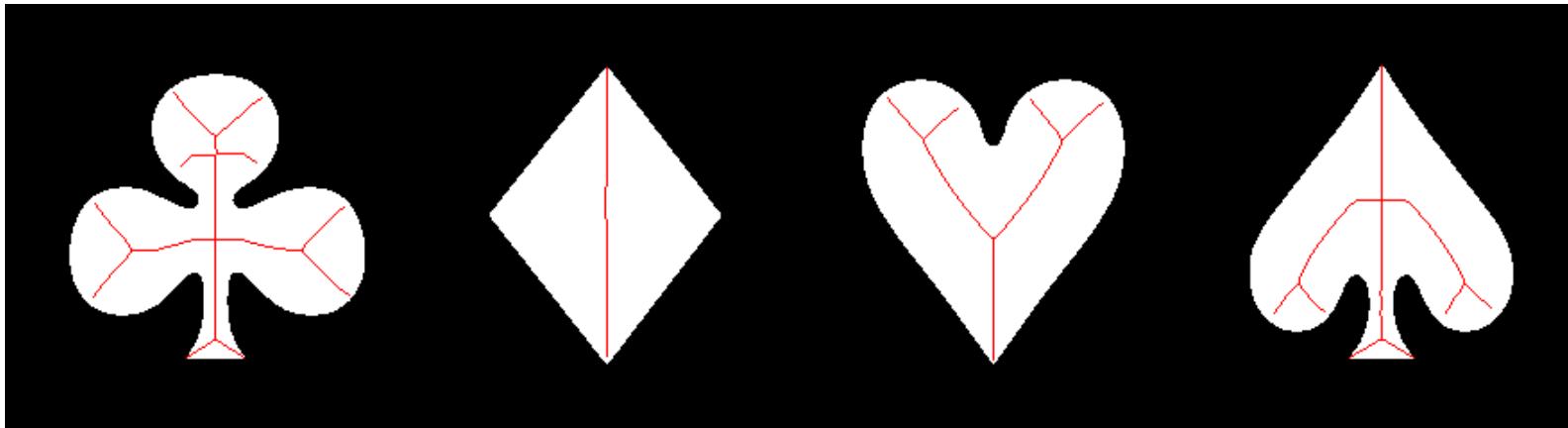
from Transcendata Europe Medial Object
Price, Stops, Butlin Transcendata Europe Ltd

Applications and Problems

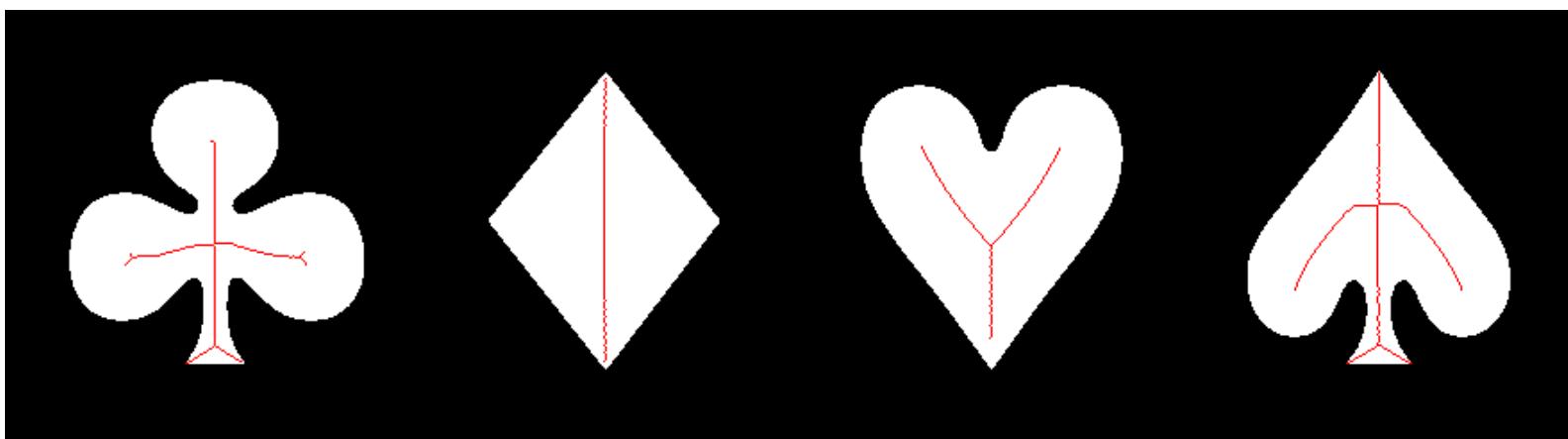
- The skeleton/MAT provides a stick figure representing the region shape
- Used in object recognition, in particular, character recognition.
- Problems:
 - Definition of a maximal disc is poorly defined on a digital grid.
 - Sensitive to noise on the boundary.
- Sequential thinning output sometimes preferred to skeleton/MAT.

Example

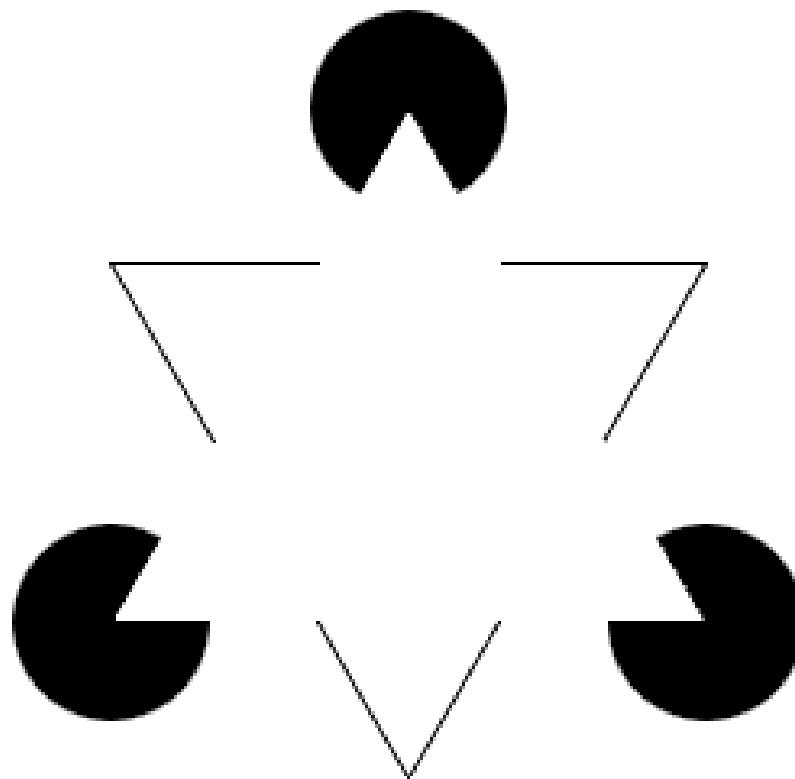
Skeletons:



Thinned:



Kanizsa Triangle



Next Week:
Image Features