Chapter 3

Binary Image Analysis

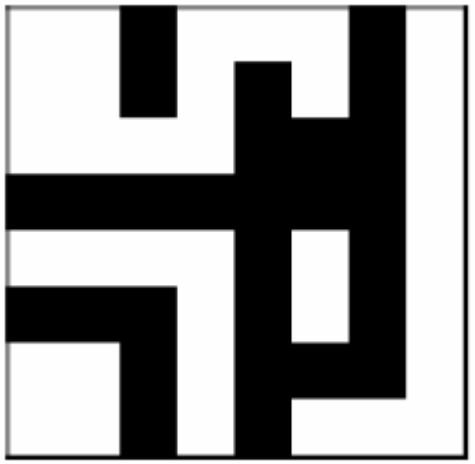
Bastian Leibe, RWTH Aachen University

Department of Mechatronics

Binary Images

- *Just two pixel values*
- *Foreground and background*
- *Regions of interest (ROI)*

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



Uses: Industrial Inspection

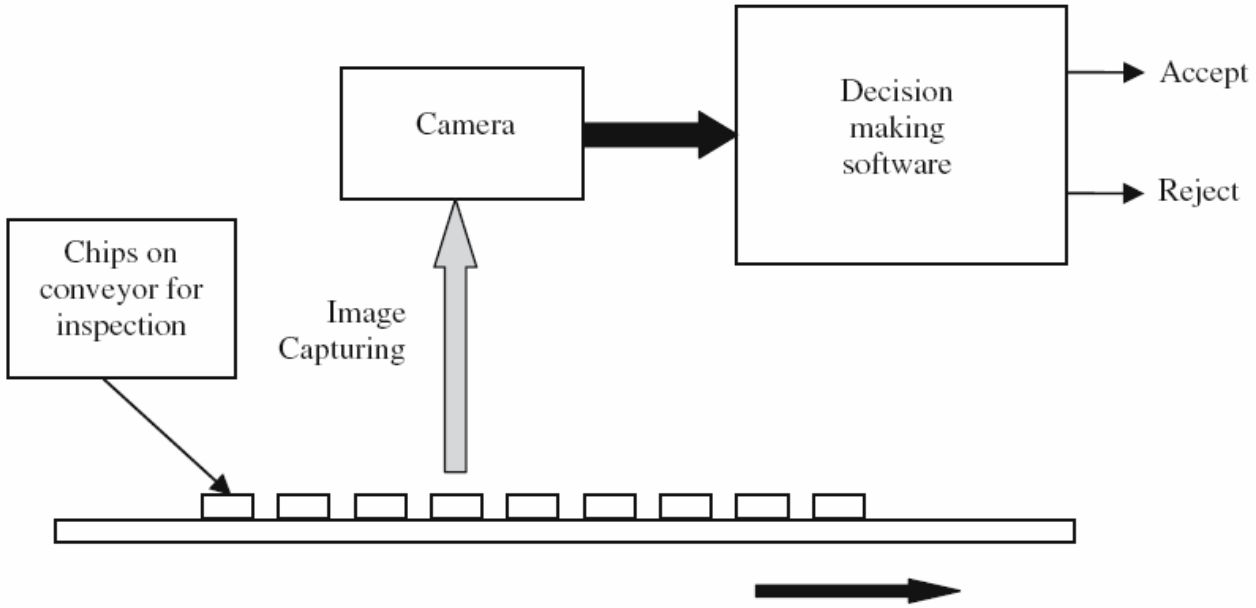


Fig. 3 Schematic diagram of marking inspection setup at Texas Instruments

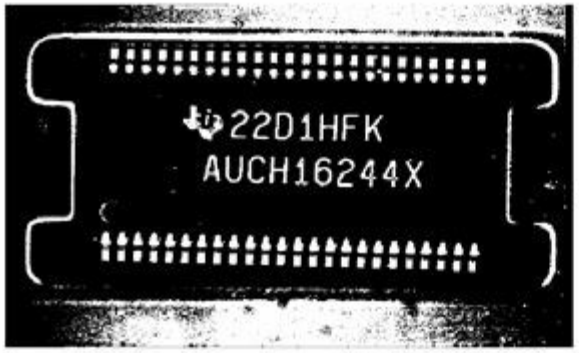
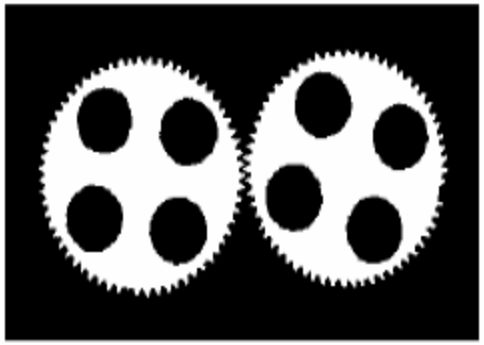


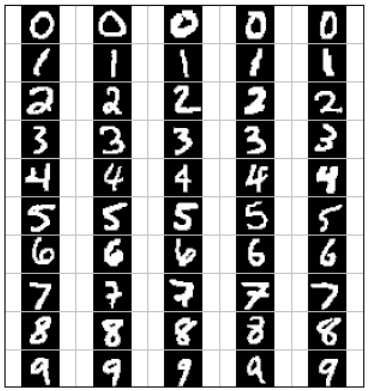
Fig. 7 Binarized image

Fig. 9 Row sum for separating a row



R. Nagarajan et al. "A real time marking inspection scheme for semiconductor industries", 2006

Uses: Document Analysis, Text Recognition



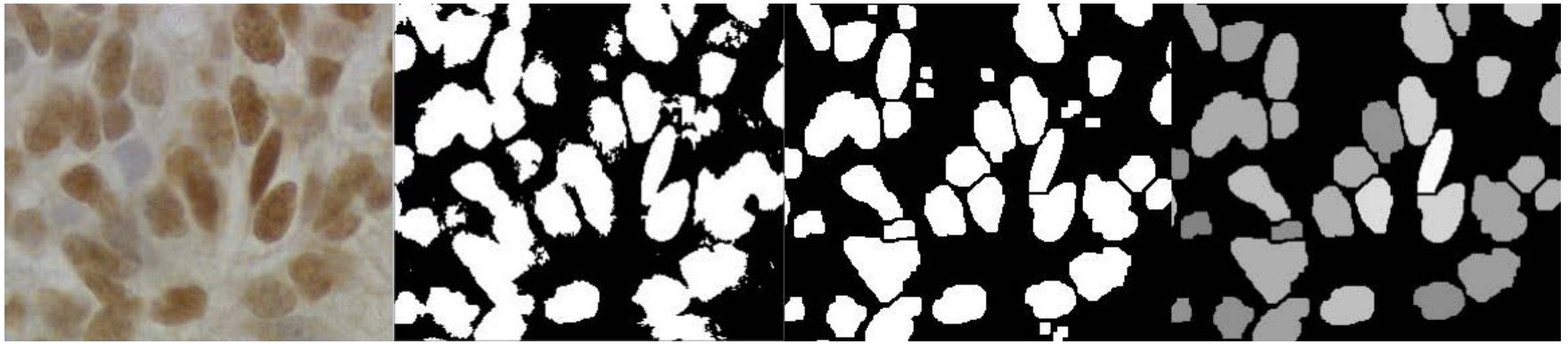
Handwritten digits

Natural text (after detection)

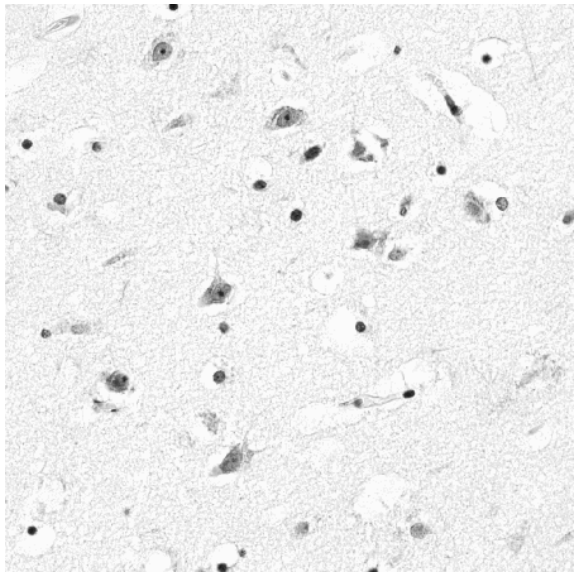
Scanned documents

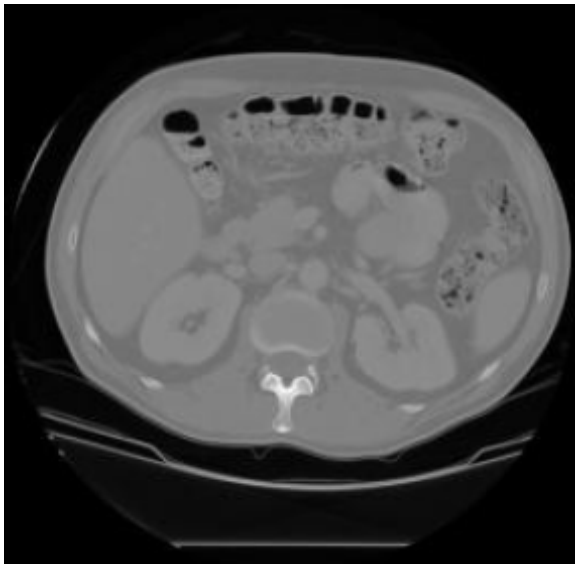
Source: Till Quack, Martin Renold

Uses: Medical/Bio Data



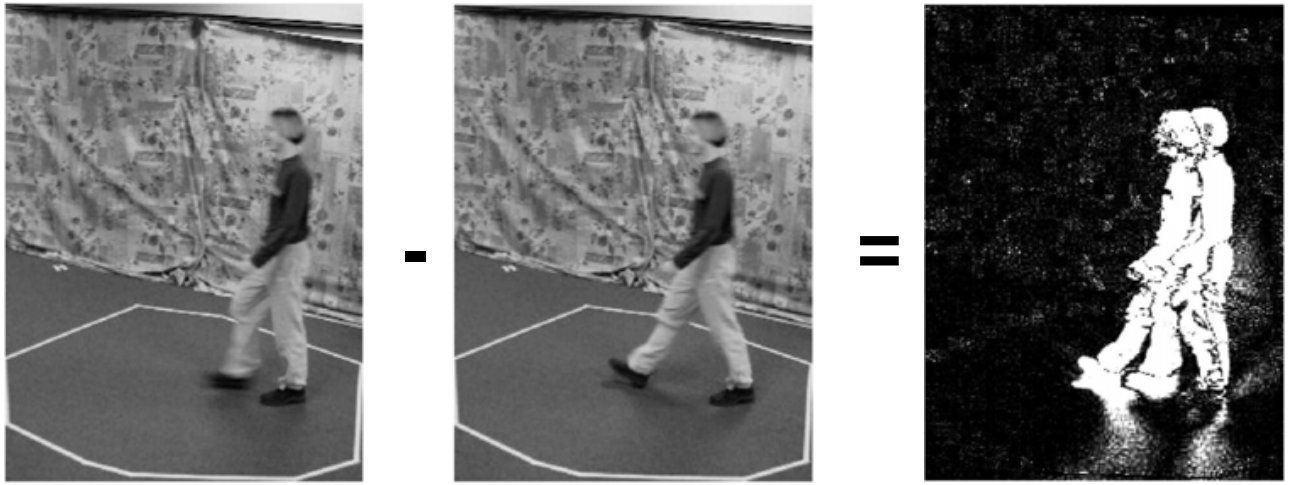
Source: D. Kim et al., Cytometry 35(1), 1999





Uses: Blob Tracking & Motion Analysis

Frame Differencing

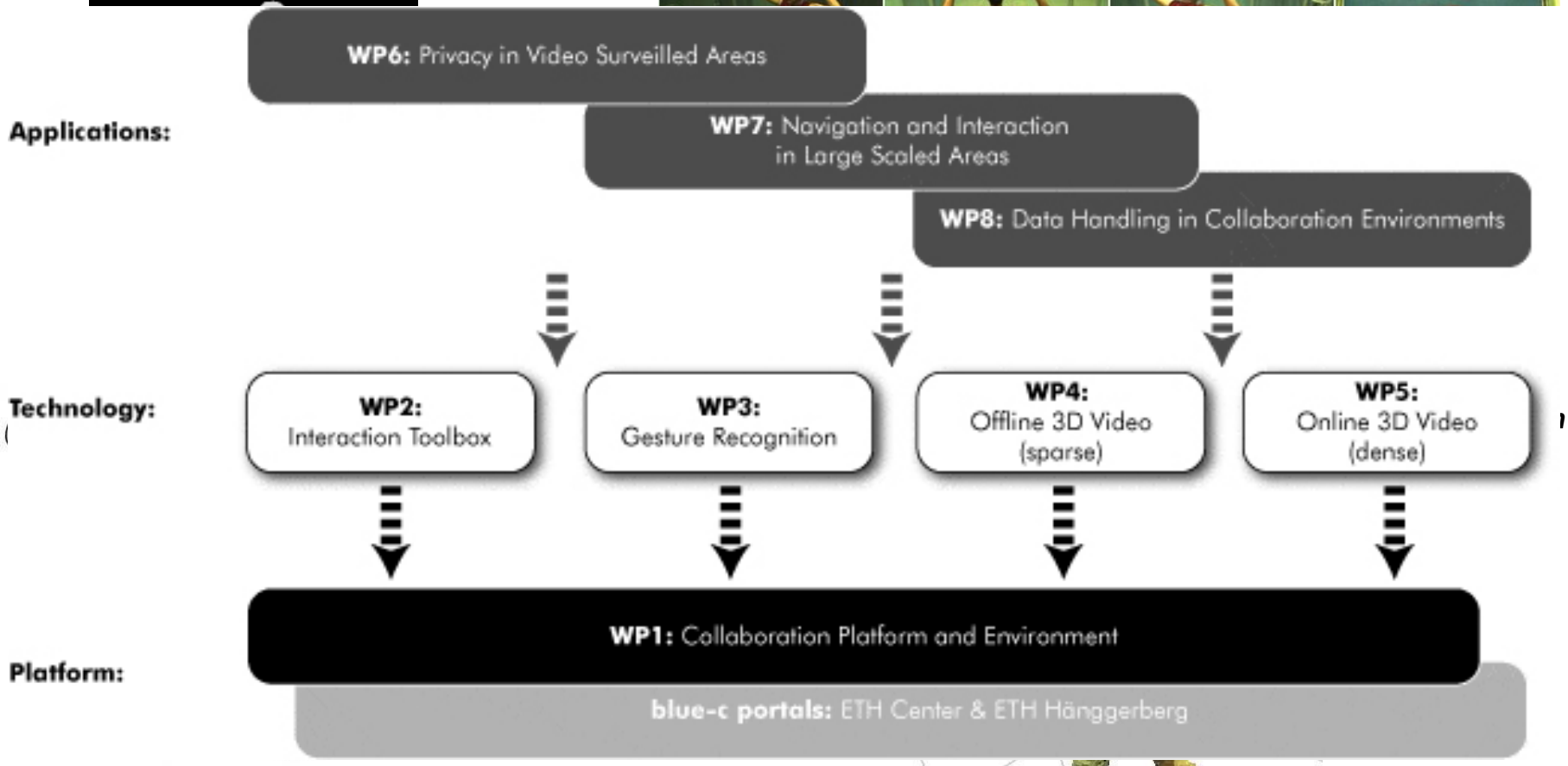


Source: Kristen Grauman

Background Subtraction

Source: Tobias Jäggli

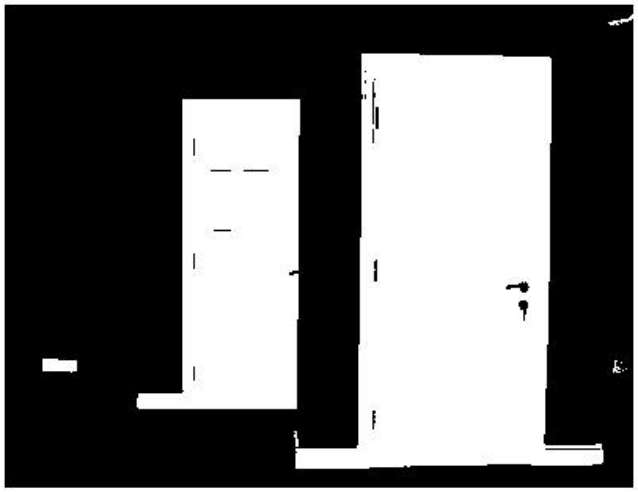
Uses: Shape Analysis, Free-Viewpoint Video



Medial axis

Uses: Intensity Based Detection

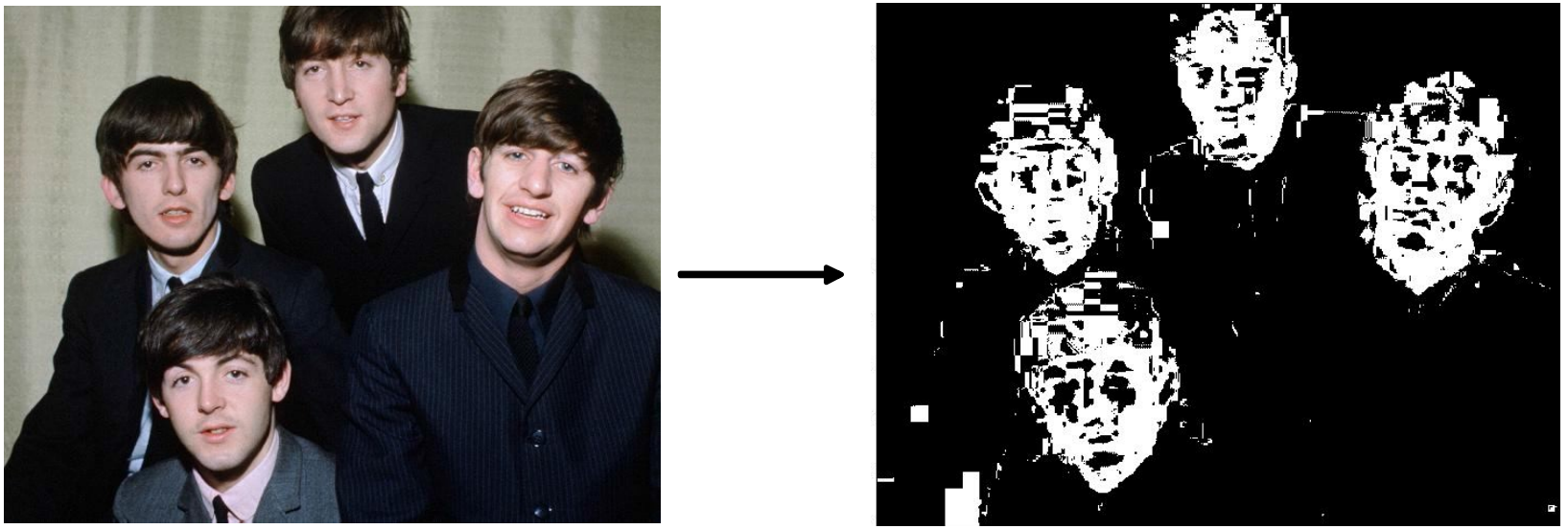
- ***Looking for dark pixels…***



```
fg_pix = find(im < 65);
```

Uses: Color Based Detection

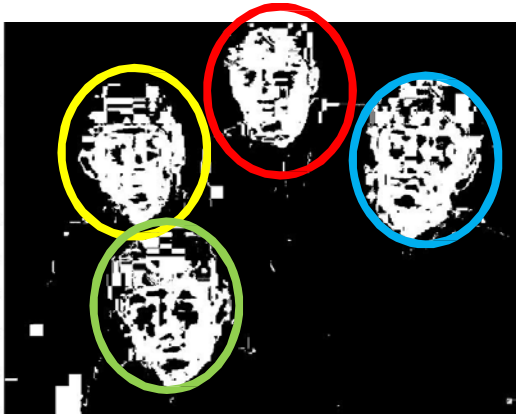
- ***Looking for pixels within a certain color range…***



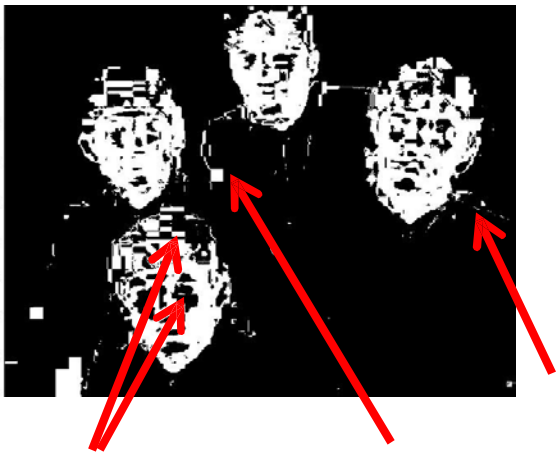
```
fg_pix = find(hue > t1 & hue < t2);
```

Issues

- ***How to demarcate multiple regions of interest?***
  - ***Count objects***
  - ***Compute further features per object***

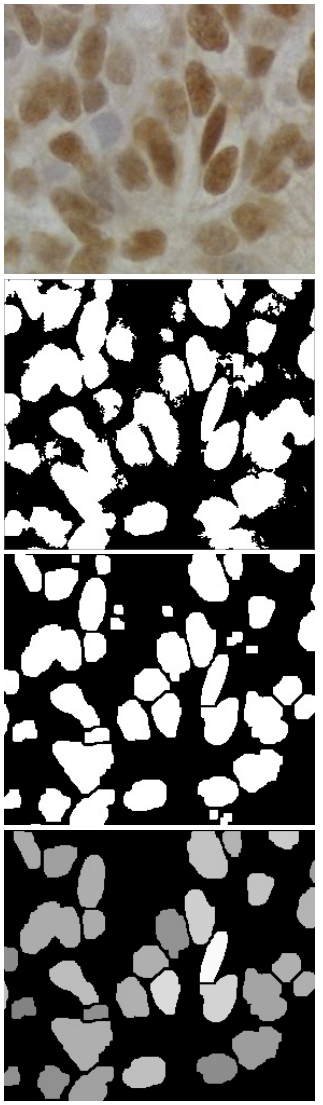


- ***What to do with "noisy" binary outputs?***
  - ***Holes***
  - ***Extra small fragments***



Outline of Today's Lecture

- ***Convert the image into binary form***
  - ***Thresholding***
- ***Clean up the thresholded image***
  - ***Morphological operators***
- ***Extract individual objects***
  - ***Connected Components Labeling***



Thresholding

- ***Grayscale image ⇒ Binary mask***
- ***Different variants***
  - ***One-sided***

$$F_T[i,j] = \begin{cases} 1, if\ F[i,j] \geq T \\ 0, otherwise \end{cases}$$

  - ***Two-sided***

$$F_T[i,j] = \begin{cases} 1, if\ T_1 \leq F[i,j] \leq T_2 \\ 0, otherwise \end{cases}$$

  - ***Set membership***

$$F_T[i,j] = \begin{cases} 1, if\ F[i,j] \in Z \\ 0, otherwise \end{cases}$$

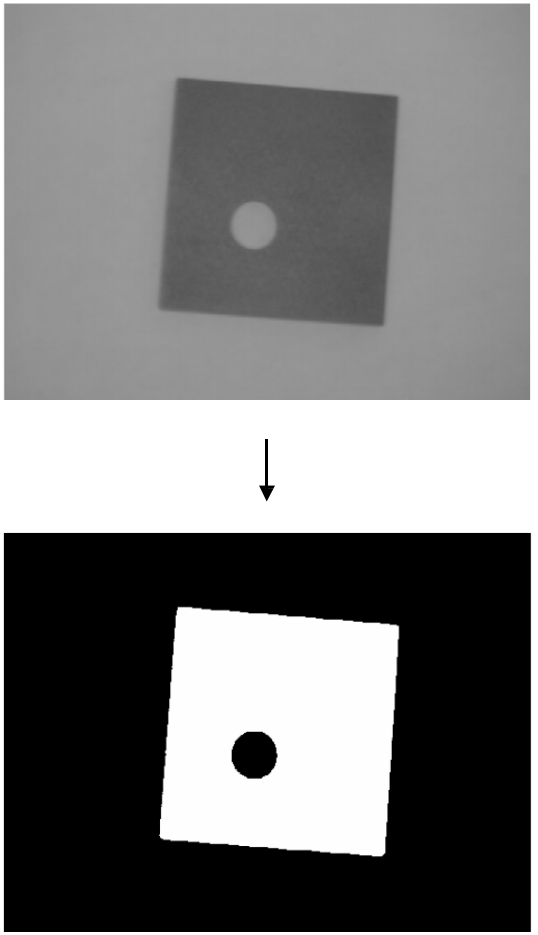


Image Source: ***http://homepages.inf.ed.ac.uk/rbf/HIPR2/***

Thresholding

22 DEFINITION *The* **histogram** *h of gray-tone image I is defined by*

$$h(m) = |\{(r,c) \mid I(r,c) = m\}|,$$

where m spans the gray-level values.

Compute the histogram H of gray-tone image I.

```
procedure histogram(I,H);
{
"Initialize the bins of the histogram to zero."
for i := 0 to MaxVal
   H[i] := 0;
"Compute values by accumulation."
for L := 0 to MaxRow
   for P := 0 to MaxCol
      {
      grayval := I[r,c];
      H[grayval] := H[grayval] + 1;
      } ;
}
```

Thresholding

- ***Load trees***
- ***Binary_Image = Gray_Image.ThresholdBinary(new Gray(120), new Gray(255));***

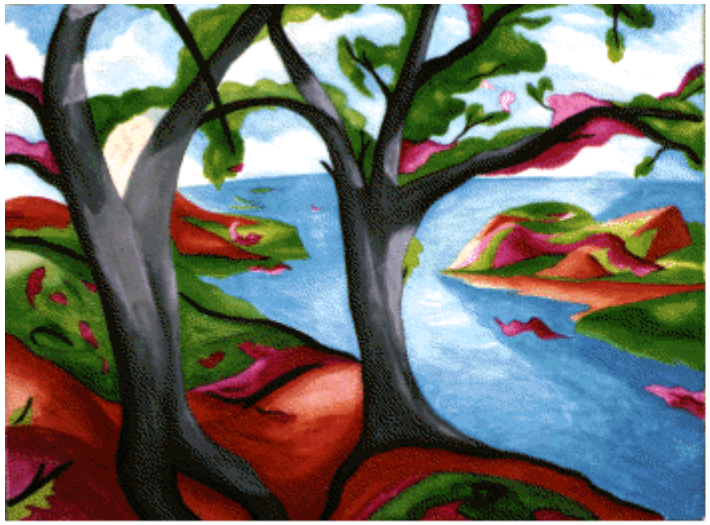
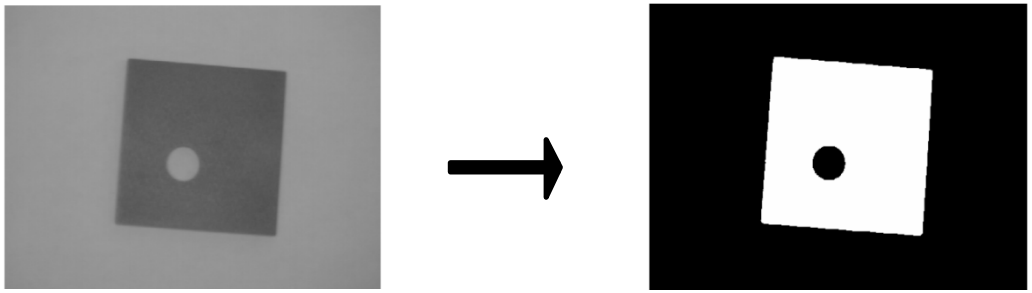


Image Courtesy of Susan Cohen

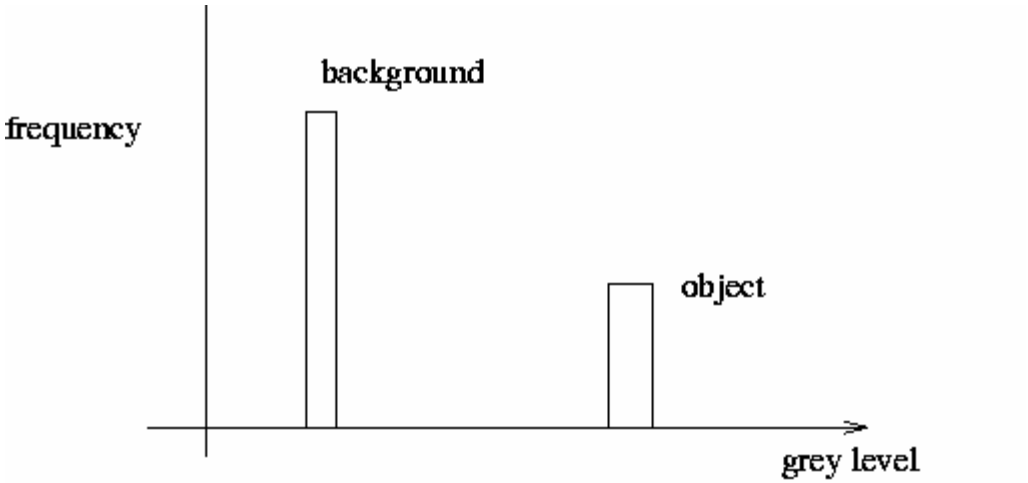
Selecting Thresholds

- ***Typical scenario***
  - ***Separate an object from a distinct background***

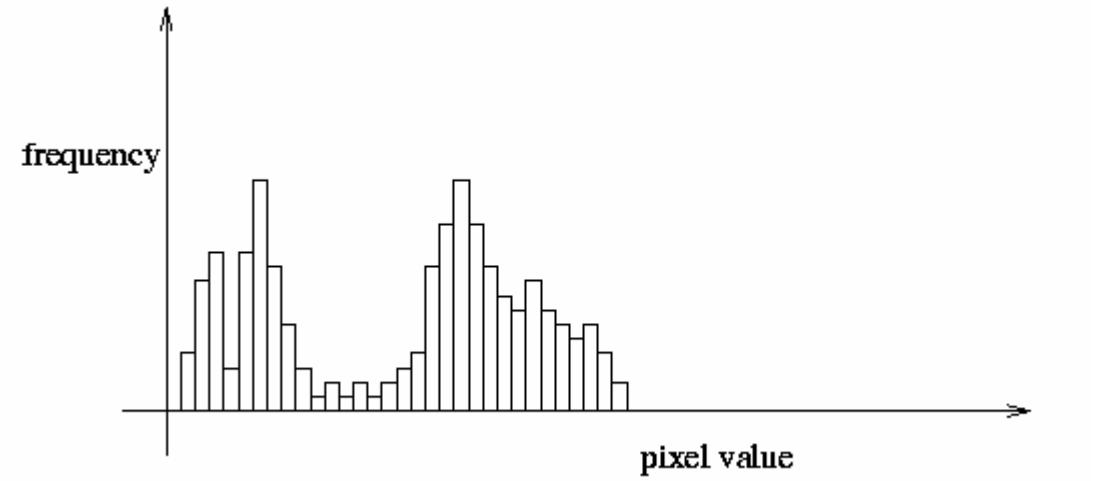


- ***Try to separate the different grayvalue distributions***
  - ***Partition a bimodal histogram***
  - ***Fit a parametric distribution (e.g. Mixture of Gaussians)***
  - ***Dynamic or local thresholds***
- ***In the following, I will present some simple methods.***

A Nice Case: Bimodal Intensity Histograms



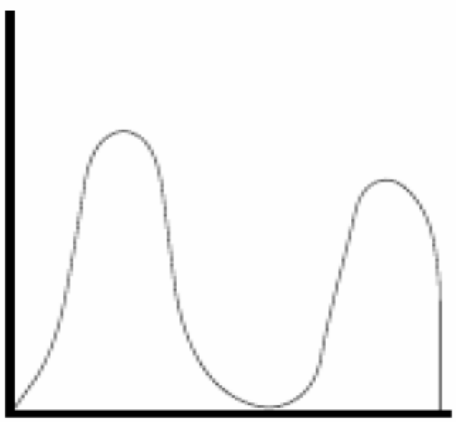
Ideal histogram, light object on dark background



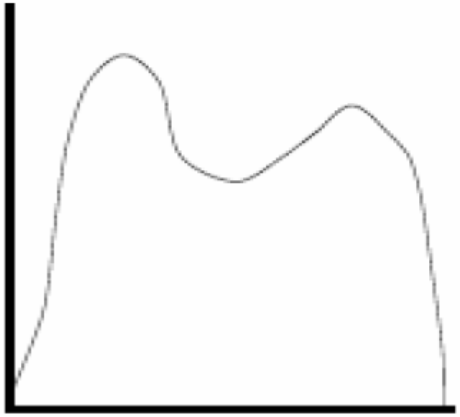
Actual observed histogram with noise

Not so Nice Cases…

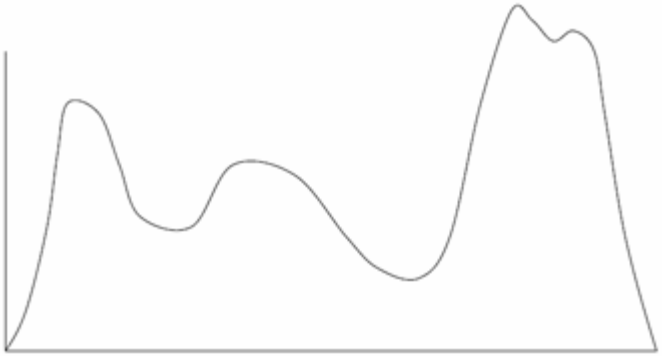
- ***How to separate those?***



Two distinct modes



Overlapping modes



Multiple modes

- ***Threshold selection is difficult in the general case***
  - ***Domain knowledge often helps***
  - ***E.g. Fraction of text on a document page (histogram quantile)***
  - ***E.g. Size of objects/structure elements***

Global Binarization [Otsu'79]

- ***Search for the threshold T that minimizes the within- class variance σ_{within} of the two classes separated by T***

$$\sigma^2_{within}(T) \stackrel{\text{def}}{=} n_1(T)\sigma_1^2(T) + n_2(T)\sigma_2^2(T)$$

where

$$n_1(T) = \left|\left\{I_{(x,y)} < T\right\}\right|, \qquad n_2(T) = \left|\left\{I_{(x,y)} \geq T\right\}\right|$$

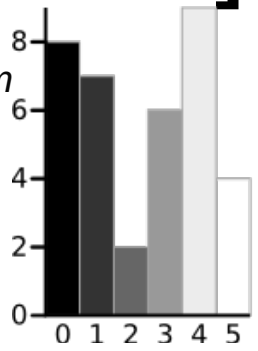
- ***This is the same as maximizing the between-class variance $\sigma_{between}$***

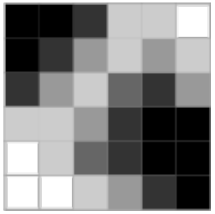
$$\sigma^2_{between}(T) = \sigma^2 - \sigma^2_{within}(T)$$

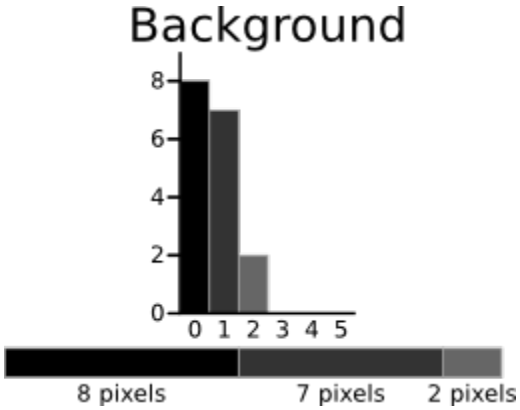
$$= n_1(T)n_2(T)\left[\mu_1(T) - \mu_2(T)\right]^2$$

Global Binarization [Otsu'79]

A 6-level greyscale image and its histogram







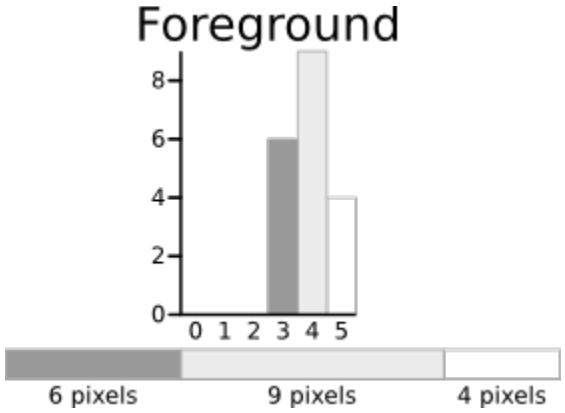
$$\text{Weight} \quad W_b = \frac{8+7+2}{36} = 0.4722$$

$$\text{Mean} \quad \mu_b = \frac{(0\times 8)+(1\times 7)+(2\times 2)}{17} = 0.6471$$

$$\text{Variance} \quad \sigma_b^2 = \frac{((0-0.6471)^2\times 8)+((1-0.6471)^2\times 7)+((2-0.6471)^2\times 2)}{17}$$

$$= \frac{(0.4187\times 8)+(0.1246\times 7)+(1.8304\times 2)}{17}$$

$$= 0.4637$$



$$\text{Weight} \quad W_f = \frac{6+9+4}{36} = 0.5278$$

$$\text{Mean} \quad \mu_f = \frac{(3\times 6)+(4\times 9)+(5\times 4)}{19} = 3.8947$$

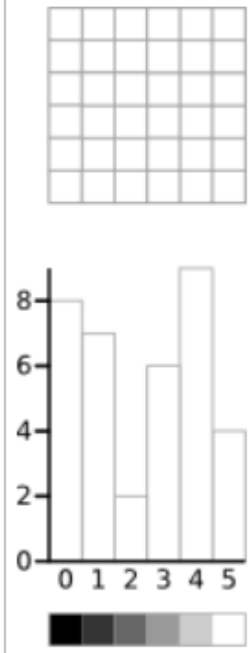
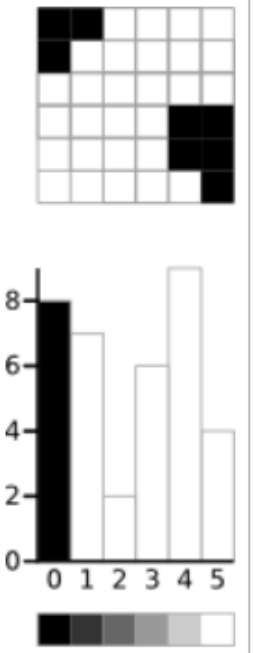
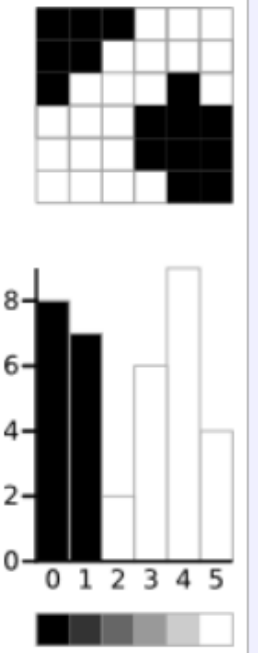
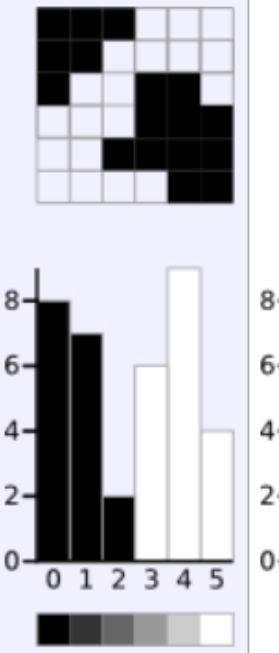
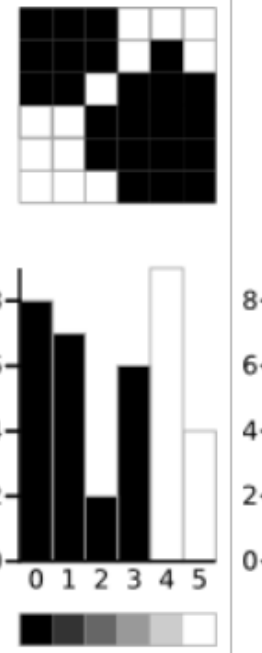
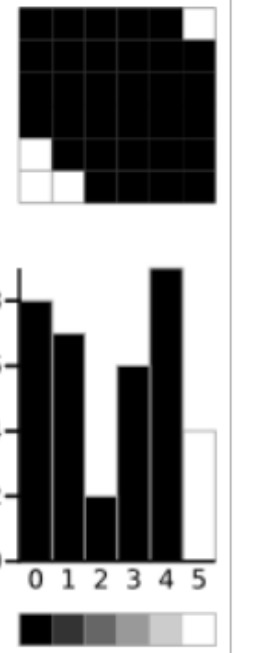
$$\text{Variance} \quad \sigma_f^2 = \frac{((3-3.8947)^2\times 6)+((4-3.8947)^2\times 9)+((5-3.8947)^2\times 4)}{19}$$

$$= \frac{(4.8033\times 6)+(0.0997\times 9)+(4.8864\times 4)}{19}$$

$$= 0.5152$$

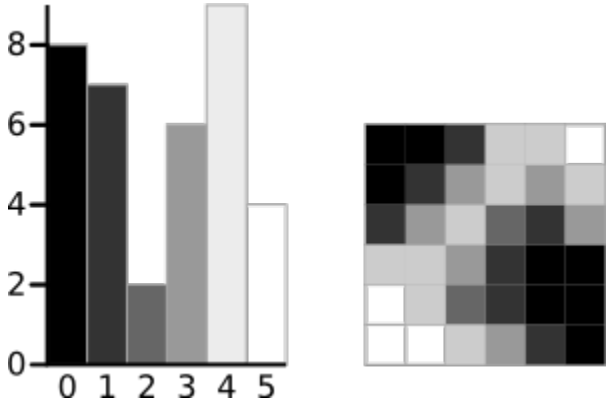
http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html

Global Binarization [Otsu'79]

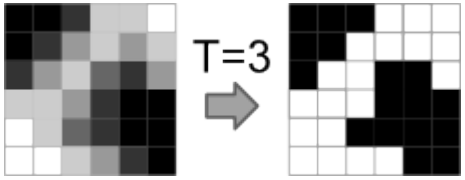
Threshold	T=0	T=1	T=2	T=3	T=4	T=5
						
Weight, Background	$W_b = 0$	$W_b = 0.222$	$W_b = 0.4167$	$W_b = 0.4722$	$W_b = 0.6389$	$W_b = 0.8889$
Mean, Background	$M_b = 0$	$M_b = 0$	$M_b = 0.4667$	$M_b = 0.6471$	$M_b = 1.2609$	$M_b = 2.0313$
Variance, Background	$\sigma^2_b = 0$	$\sigma^2_b = 0$	$\sigma^2_b = 0.2489$	$\sigma^2_b = 0.4637$	$\sigma^2_b = 1.4102$	$\sigma^2_b = 2.5303$
Weight, Foreground	$W_f = 1$	$W_f = 0.7778$	$W_f = 0.5833$	$W_f = 0.5278$	$W_f = 0.3611$	$W_f = 0.1111$
Mean, Foreground	$M_f = 2.3611$	$M_f = 3.0357$	$M_f = 3.7143$	$M_f = 3.8947$	$M_f = 4.3077$	$M_f = 5.000$
Variance, Foreground	$\sigma^2_f = 3.1196$	$\sigma^2_f = 1.9639$	$\sigma^2_f = 0.7755$	$\sigma^2_f = 0.5152$	$\sigma^2_f = 0.2130$	$\sigma^2_f = 0$
Within Class Variance	$\sigma^2_W = 3.1196$	$\sigma^2_W = 1.5268$	$\sigma^2_W = 0.5561$	$\mathbf{\sigma^2_W = 0.4909}$	$\sigma^2_W = 0.9779$	$\sigma^2_W = 2.2491$

Within Class Variance $\sigma_W^2 = W_b\,\sigma_b^2 + W_f\,\sigma_f^2 = 0.4722 * 0.4637 + 0.5278 * 0.5152$
$= 0.4909$

Global Binarization [Otsu'79]



A 6-level greyscale image and its histogram



Result of Otsu's method

- ***A faster approach***

$$\text{Within Class Variance} \quad \sigma_W^2 = W_b\,\sigma_b^2 + W_f\,\sigma_f^2 \quad \text{(as seen above)}$$

$$\begin{aligned}\text{Between Class Variance} \quad \sigma_B^2 &= \sigma^2 - \sigma_W^2 \\ &= W_b(\mu_b - \mu)^2 + W_f(\mu_f - \mu)^2 \quad (\text{where } \mu = W_b\,\mu_b + W_f\,\mu_f) \\ &= W_b\,W_f\,(\mu_b - \mu_f)^2\end{aligned}$$

Threshold	T=0	T=1	T=2	T=3	T=4	T=5
Within Class Variance	$\sigma^2_W = 3.1196$	$\sigma^2_W = 1.5268$	$\sigma^2_W = 0.5561$	**$\sigma^2_W = 0.4909$**	$\sigma^2_W = 0.9779$	$\sigma^2_W = 2.2491$
Between Class Variance	$\sigma^2_B = 0$	$\sigma^2_B = 1.5928$	$\sigma^2_B = 2.5635$	**$\sigma^2_B = 2.6287$**	$\sigma^2_B = 2.1417$	$\sigma^2_B = 0.8705$

Algorithm

- **Precompute a cumulative grayvalue histogram** h**.**

For each potential threshold *T*

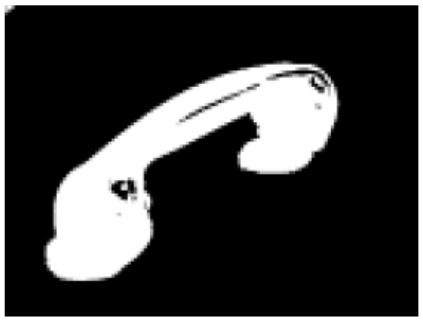
1.) Separate the pixels into two clusters according to *T*

2.) Look up n_1**,** n_2 **in** h **and compute both cluster means**

3.) Compute $\sigma^2_{\text{between}}$

Choose $T^* = \arg\max_T [\sigma^2_{between}(T)]$

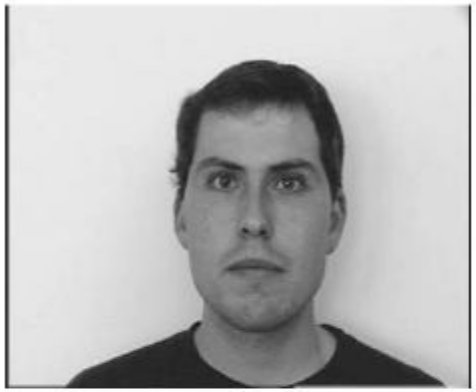
$\arg\max_x f(x) := \{x \mid \forall y : f(y) \leq f(x)\}$ $\arg\max_{x \in [0,4\pi]} \cos(x) = \{0, 2\pi, 4\pi\}$

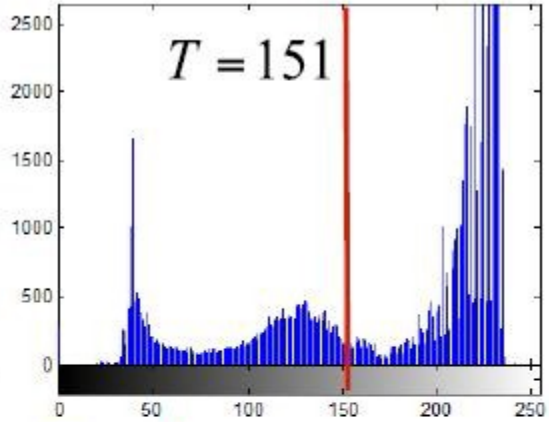


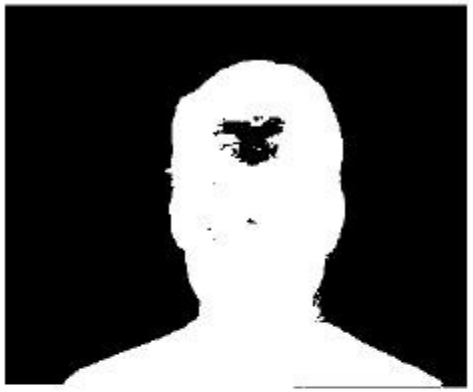
Source code

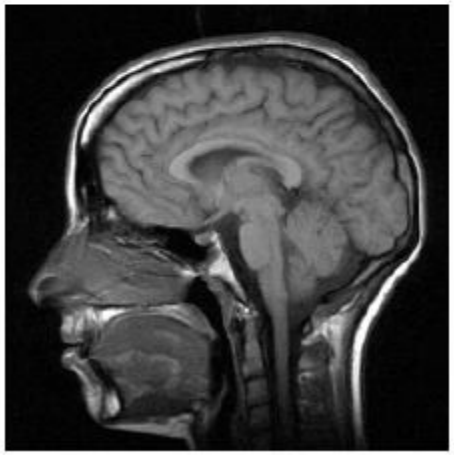
Effects

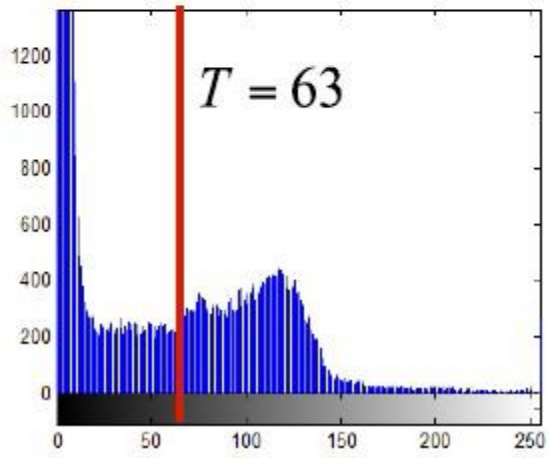
graythresh - Global image threshold using Otsu's method











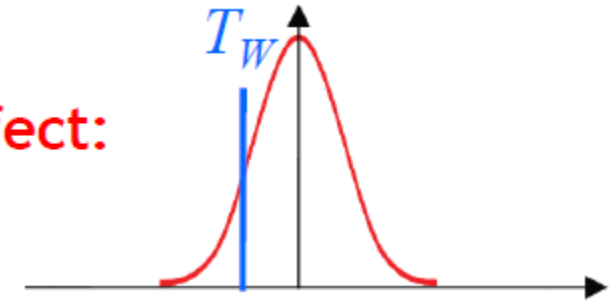
Local Binarization [Niblack'86]

- ***Estimate a local threshold within a small neighborhood***

$$T_W = \mu_W + k \cdot \sigma_W$$

where $k \in [-1,1]$ is a user-defined parameter.

Effect:



- ***Improved version to suppress background noise for document binarization [Sauvola'00]***

$$T_W = \mu_W \left[1 + k \cdot \left(\frac{\sigma_W}{R} - 1 \right) \right]$$

Useful for text, but not for larger objects

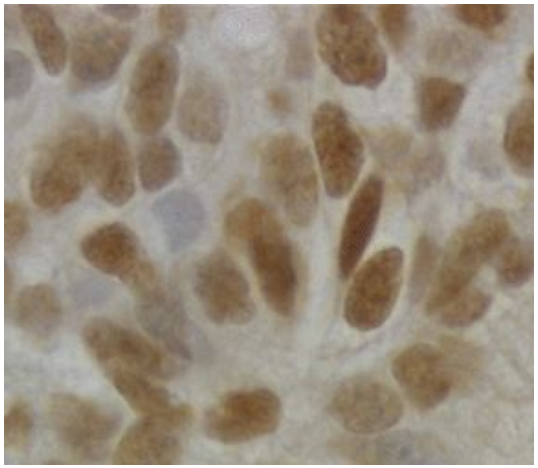
where R is the dynamic range of σ and $k>0$

- ***Typical values: $R = 128$ for 8-bit images and $k \approx 0.5$***

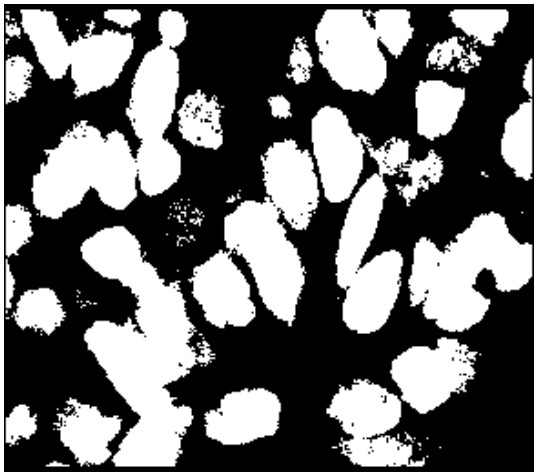
Source code

Effects

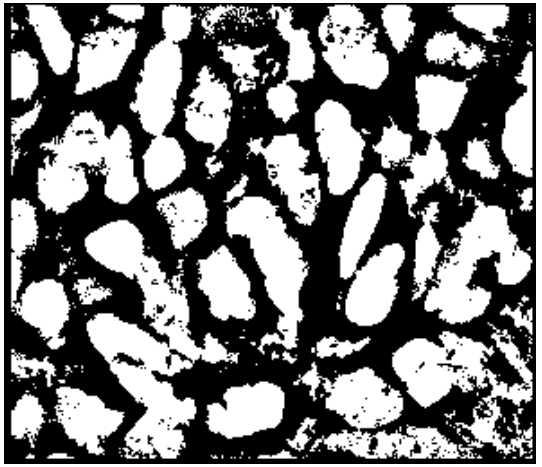
```
CvInvoke.cvAdaptiveThreshold(Gray_Image,
Binary_Image, 255,
Emgu.CV.CvEnum.ADAPTIVE_THRESHOLD_TYPE.CV_
ADAPTIVE_THRESH_GAUSSIAN_C,
Emgu.CV.CvEnum.THRESH.CV_THRESH_BINARY,
win_size, 0.5);
```



Original image



Global threshold selection (Otsu)



Local threshold selection (Niblack)

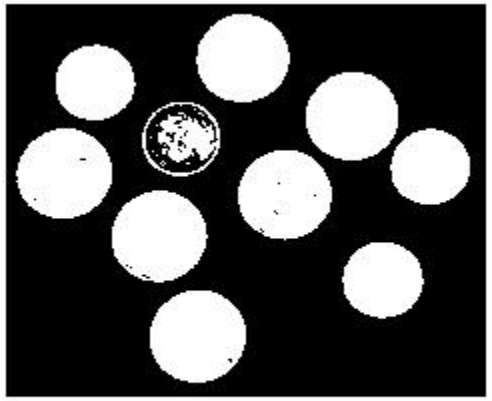
```
CvInvoke.cvAdaptiveThreshold(Gray_Image,
Binary_Image, 255,
Emgu.CV.CvEnum.ADAPTIVE_THRESHOLD_TYPE.CV_
ADAPTIVE_THRESH_MEAN_C,
Emgu.CV.CvEnum.THRESH.CV_THRESH_BINARY,
win_size, 0.5);
```

Effects

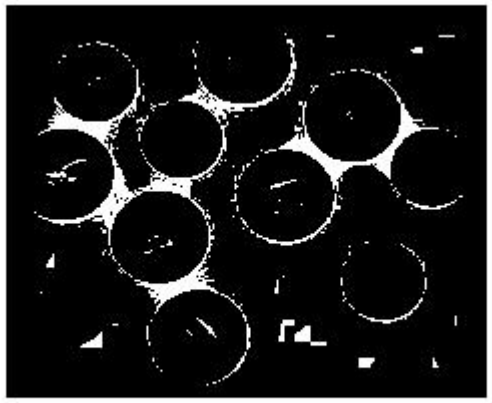
graythresh - Global image threshold using Otsu's method

```
I = imread('coins.jpg');
level = graythresh(I);
BW = im2bw(I,level);
Imshow(BW)
```

Original image



Global threshold(Otsu)



Local threshold(Niblack)

```
CvInvoke.cvThreshold(Gray_Image,
Binary_Image, 1, 255,
Emgu.CV.CvEnum.THRESH.CV_THRESH_BINARY |
Emgu.CV.CvEnum.THRESH.CV_THRESH_OTSU);
```

Additional Improvements

- ***Document images often contain a smooth gradient***

⇒ ***Try to fit that gradient with a polynomial function***

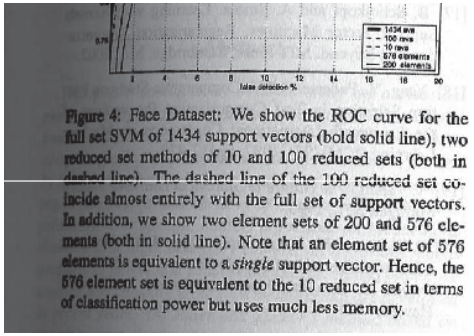
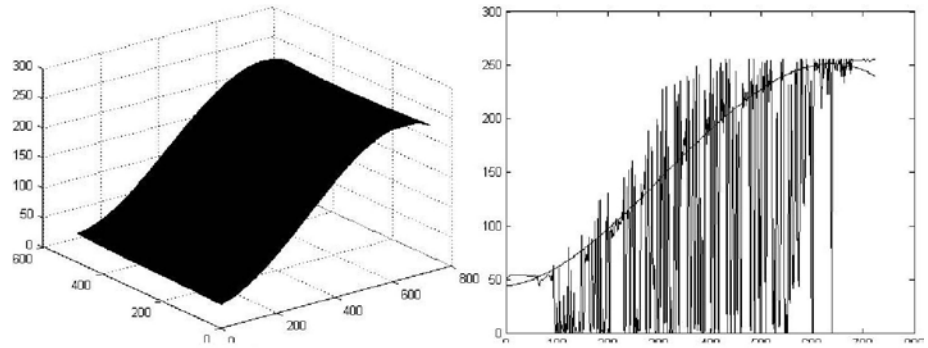


Figure 4: Face Dataset: We show the ROC curve for the full set SVM of 1434 support vectors (bold solid line), two reduced set methods of 10 and 100 reduced sets (both in dashed line). The dashed line of the 100 reduced set coincide almost entirely with the full set of support vectors. In addition, we show two element sets of 200 and 576 elements (both in solid line). Note that an element set of 576 elements is equivalent to a *single* support vector. Hence, the 576 element set is equivalent to the 10 reduced set in terms of classification power but uses much less memory.

Original image



Fitted surface

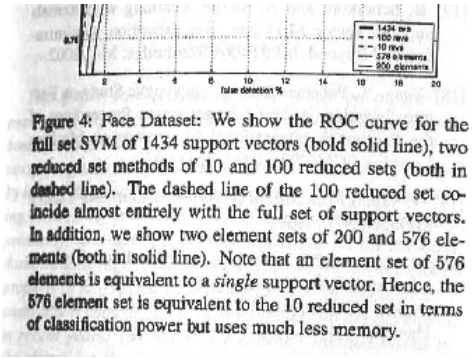


Figure 4: Face Dataset: We show the ROC curve for the full set SVM of 1434 support vectors (bold solid line), two reduced set methods of 10 and 100 reduced sets (both in dashed line). The dashed line of the 100 reduced set coincide almost entirely with the full set of support vectors. In addition, we show two element sets of 200 and 576 elements (both in solid line). Note that an element set of 576 elements is equivalent to a *single* support vector. Hence, the 576 element set is equivalent to the 10 reduced set in terms of classification power but uses much less memory.

Shading compensation

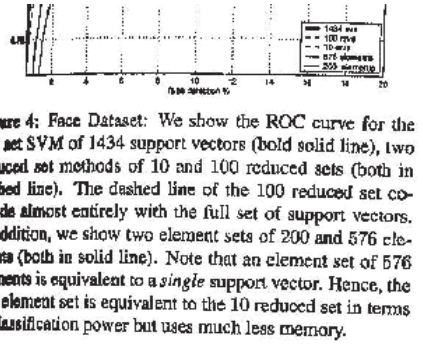


Figure 4: Face Dataset: We show the ROC curve for the full set SVM of 1434 support vectors (bold solid line), two reduced set methods of 10 and 100 reduced sets (both in dashed line). The dashed line of the 100 reduced set coincide almost entirely with the full set of support vectors. In addition, we show two element sets of 200 and 576 elements (both in solid line). Note that an element set of 576 elements is equivalent to a *single* support vector. Hence, the 576 element set is equivalent to the 10 reduced set in terms of classification power but uses much less memory.

Binarized result

Source: S. Lu & C. Tan, ICDAR'07

Least squares and Projections

- ***We want to fit a straight line $y = c+dx$ to the data (0, 1), (1, 4), (2, 2), (3, 5). This means we must find the c and d that satisfy the equations***

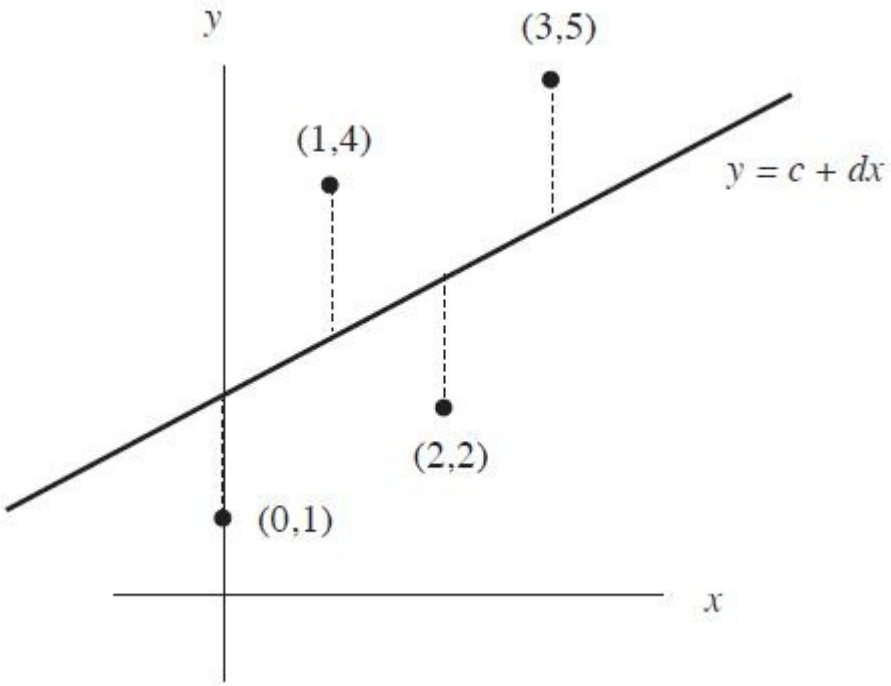
c + d ·0 = 1

c + d ·1 = 4

c + d ·2 = 2

c + d ·3 = 5

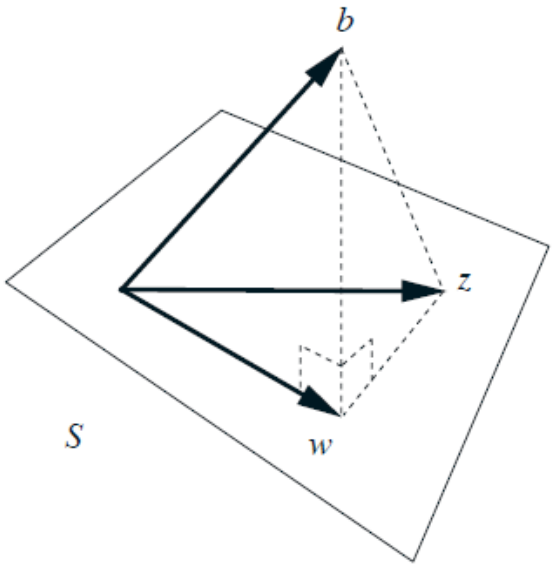
$$\begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 2 \\ 5 \end{bmatrix}$$



Linear least-squares using normal equations

Ax = b

A is an m × n matrix with m > n.



minimizes the norm $\|Ax - b\|$

Ax – b must be a vector orthogonal to the column space of A. This means, explicitly, that Ax – b is perpendicular to each of the columns of A.

$$A^T(Ax-b) = 0.$$

$$(A^TA)x = A^Tb.$$

Objective

Find x that minimizes $\|Ax - b\|$.

Algorithm

(i) Solve the normal equations $A^TAx = A^Tb$.
(ii) If A^TA is invertible, then the solution is $x = (A^TA)^{-1}A^Tb$.

Surface Fitting

- ***Polynomial surface of degree d***

$$f(x,y) = \sum_{i+j=0}^{d} b_{i,j} x^i y^j$$

$$f(x,y) = Ax^3+Bx^2y+Cxy^2+Dy^3+Ex^2+Fxy+Gy^2+Hx+Iy+J$$

- ***Least-squares estimation, e.g. for d=3 (m=10)***

$$Ab = I$$

$$\begin{bmatrix} 1 & x_0 & y_0 & x_0^2 & x_0 y_0 & \cdots & y_0^3 \\ 1 & x_1 & y_1 & x_1^2 & x_1 y_1 & \cdots & y_1^3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & y_n & x_n^2 & x_n y_n & \cdots & y_n^3 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_m \end{bmatrix} = \begin{bmatrix} I_0 \\ I_1 \\ \vdots \\ I_n \end{bmatrix},$$

Solution with pseudo-inverse:

$$b = (A^T A)^{-1} A^T I$$

Surface Fitting

- ***Iterative Algorithm***

1.) Fit parametric surface to all points in region.
2.) Subtract estimated surface.
3.) Apply global threshold (e.g. with Otsu method)

Initial guess

4.) Fit surface to all background pixels in original region.
5.) Subtract estimated surface.
5.) Apply global threshold (Otsu)

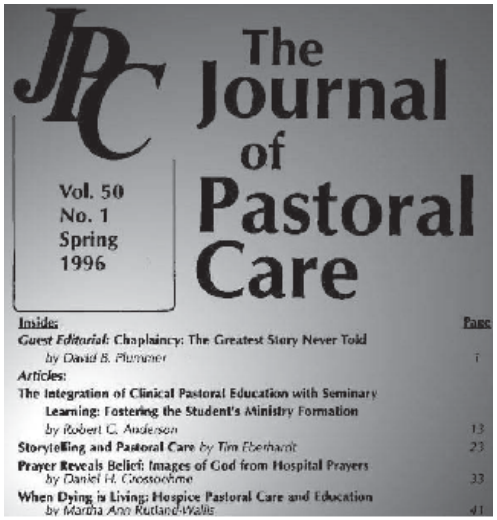
Refined guess

6.) Iterate further if needed…

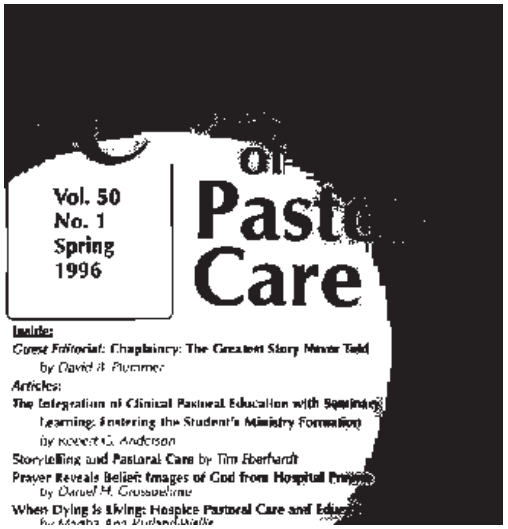
- ***The first pass also takes foreground pixels into account.***
  - ***This is corrected in the following passes.***
  - ***Basic assumption here: most pixels belong to the background.***

Result Comparison

Original image

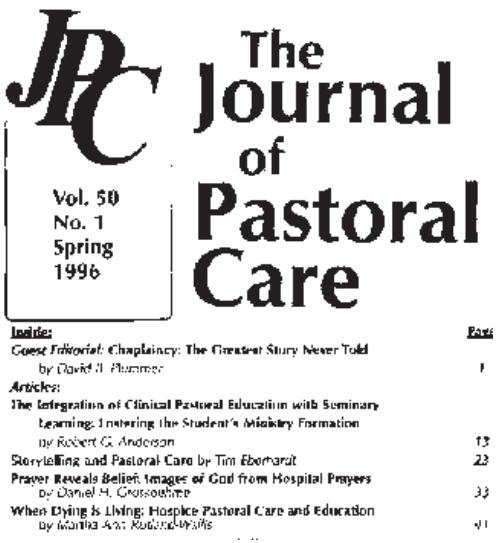


Global (Otsu)



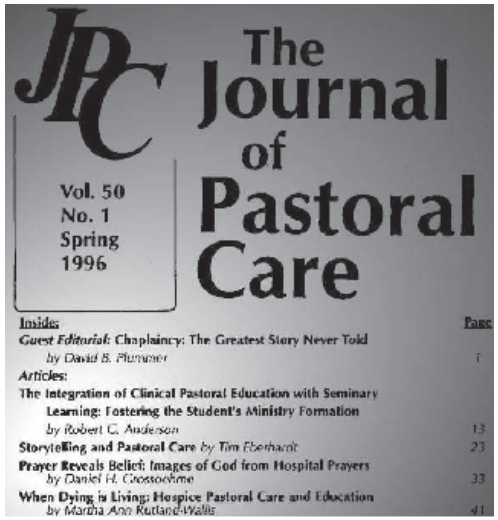
Local (Niblack)

Polynomial + Global

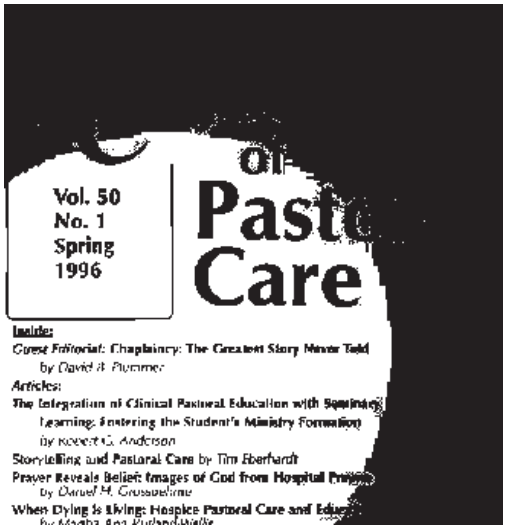


Result Comparison

Original image

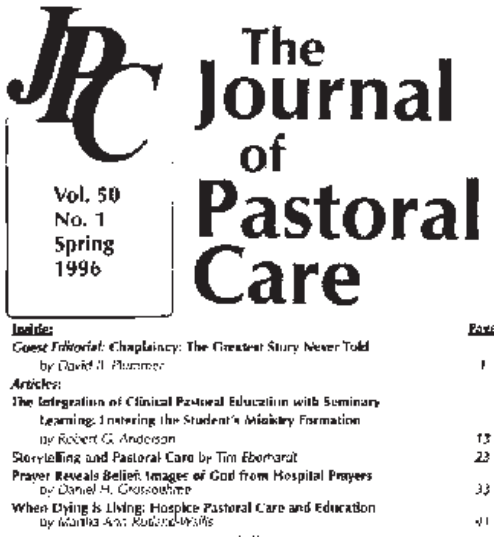


Global (Otsu)



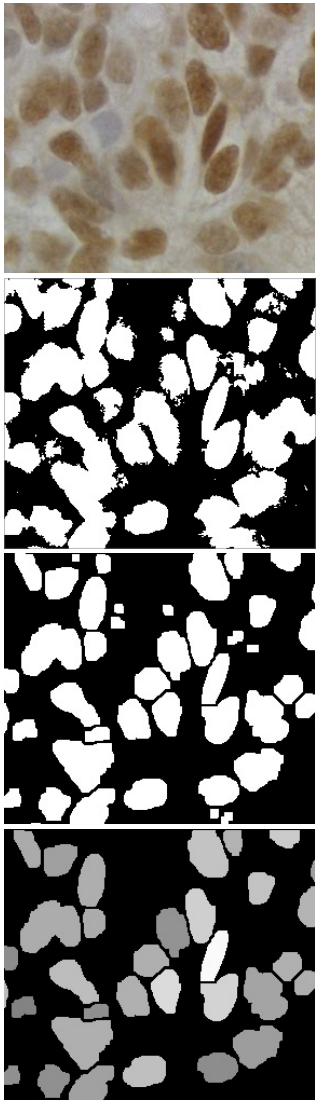
Local (Sauvola)

Polynomial + Global



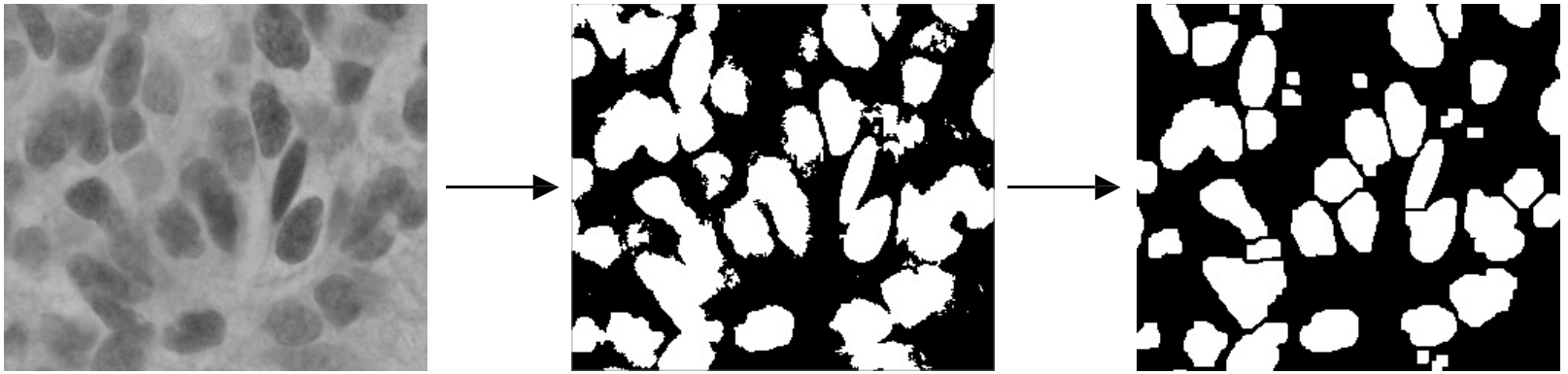
Outline of Today's Lecture

- ***Clean up the thresholded image***
  - ***Morphological operators***



Cleaning the Binarized Results

- ***Results of thresholding often still contain noise***



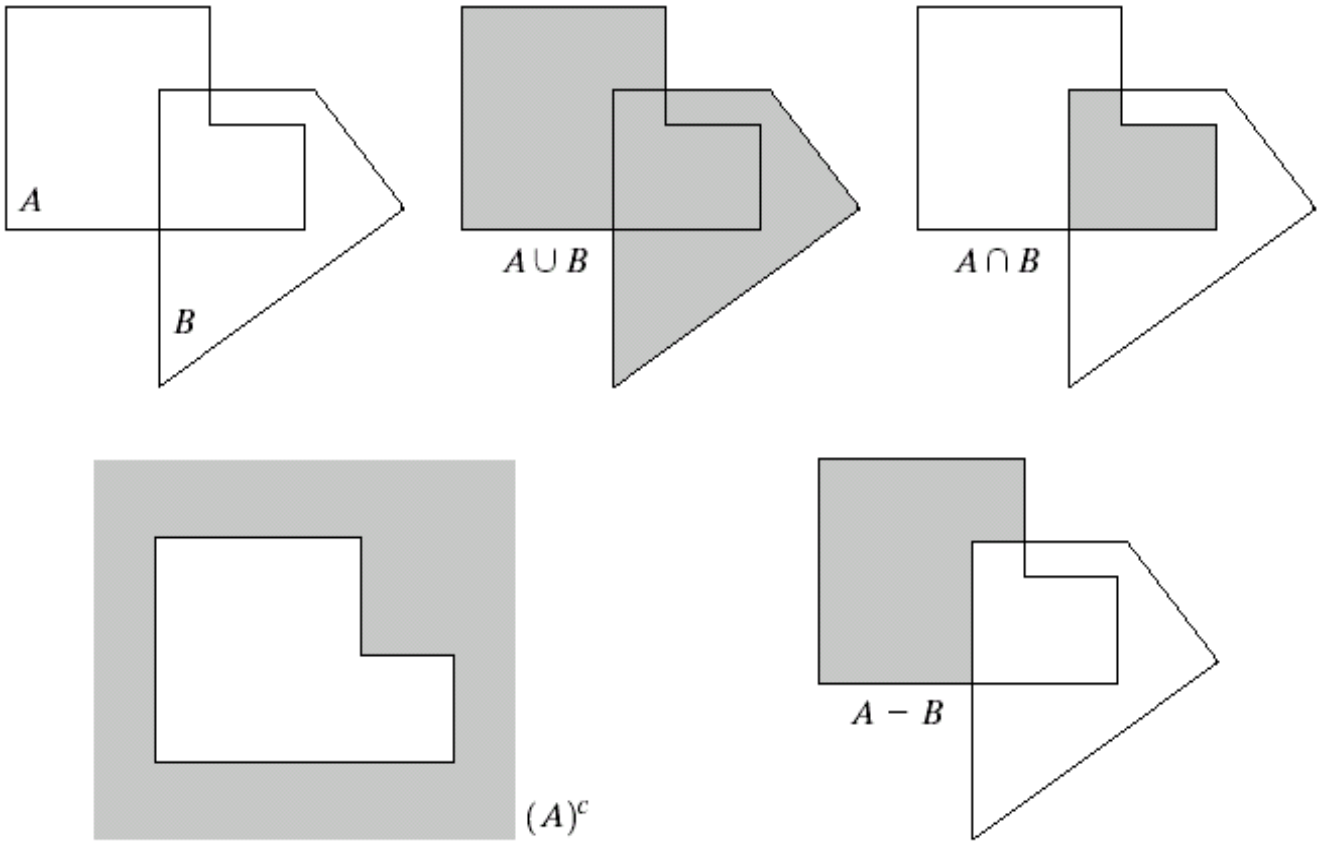
- ***Necessary cleaning operations***
  - ***Remove isolated points and small structures***
  - ***Fill holes***

⇒ ***Morphological Operators***

Z^2 and Z^3

- ***Set in mathematic morphology represent objects in an image***
  - ***binary image (0 = white, 1 = black) : the element of the set is the coordinates (x,y) of pixel belong to the object ⇨ Z^2***
- ***Gray-scaled image : the element of the set is the coordinates (x,y) of pixel belong to the object and the gray levels ⇨ Z^3***

Basic Set Theory



a b c
d e

FIGURE 9.1
(a) Two sets A and B. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B.

Logic Operations

p	q	p AND q (also $p \cdot q$)	p OR q (also $p + q$)	NOT (p) (also $\bar{p}$)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Example

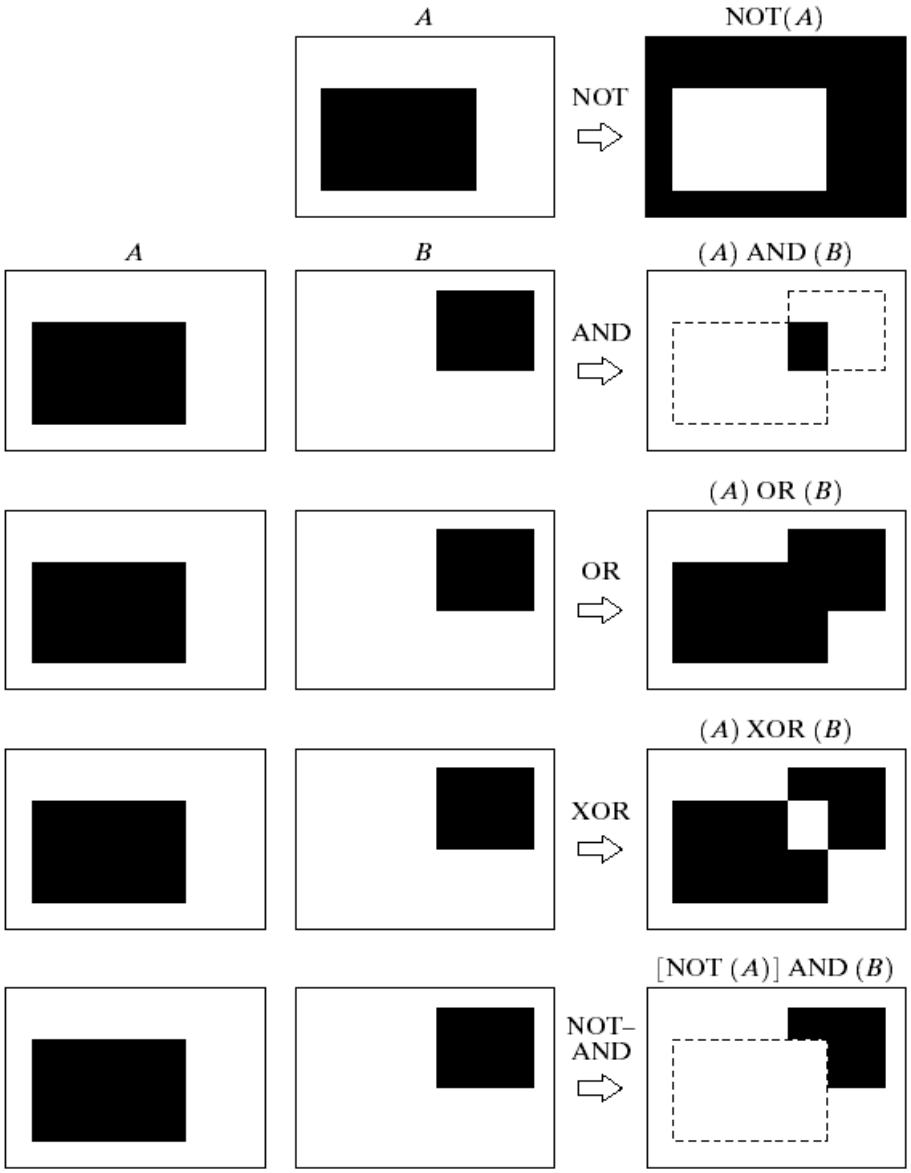
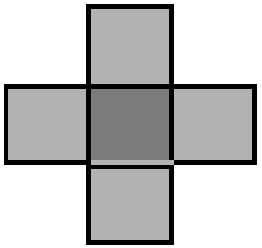
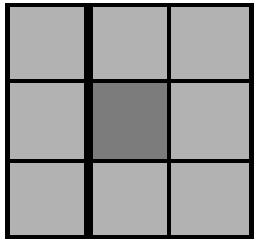


FIGURE 9.3 Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

Structuring Element (SE)

- ***Small set to probe the image under study***
- ***Shape and size must be adapted to geometric properties for the objects***





Basic morphological operations

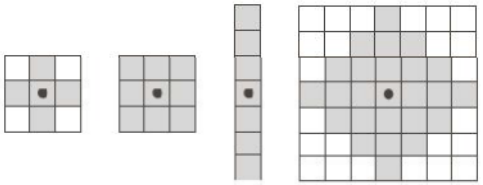
- ***Erosion***

- ***Dilation***

- ***Combine to***
  - ***Opening***
  - ***Closing***

Morphological Operators

- ***Basic idea***
  - ***Scan the image with a structuring element***
  - ***Perform set operations (intersection, union) of image content with structuring element***

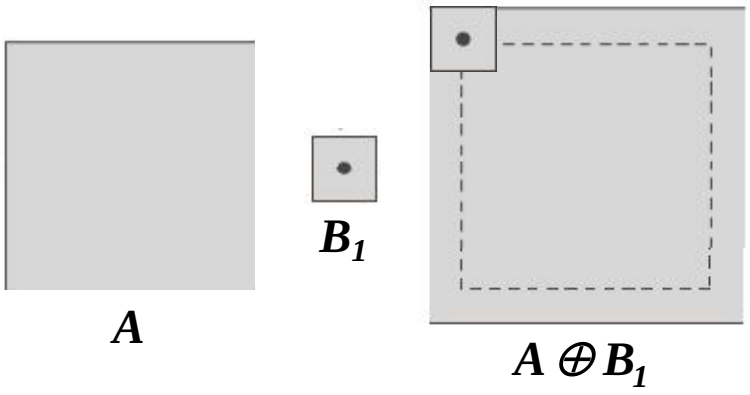


- ***Two basic operations***
  - ***Dilation*** ***(Matlab: `imdilate`)***
  - ***Erosion*** ***(Matlab: `imerode`)***
- ***Several important combinations***
  - ***Opening*** ***(Matlab: `imopen`)***
  - ***Closing*** ***(Matlab: `imclose`)***
  - ***Boundary extraction***

```
Image<Gray, Byte> src = new Image<Gray, Byte>( "Your Image.png" );
Image<Gray, Byte> dst = new Image<Gray, Byte>( src.Width, src.Height );
StructuringElementEx element = new StructuringElementEx( 3, 3, 1, 1, Emgu.CV.CvEnum.CV_ELEMENT_SHAPE.CV_SHAPE_CROSS );
CvInvoke.cvMorphologyEx( src, dst, IntPtr.Zero, element, CV_MORPH_OP.CV_MOP_OPEN, 1 );
```

Dilation

- ***Definition***
  - *"The dilation of A by B is the set of all displacements z, such that $(\hat{B})_z$ and A overlap by at least one element".*
  - *($(\hat{B})_z$ is the mirrored version of B, shifted by z)*

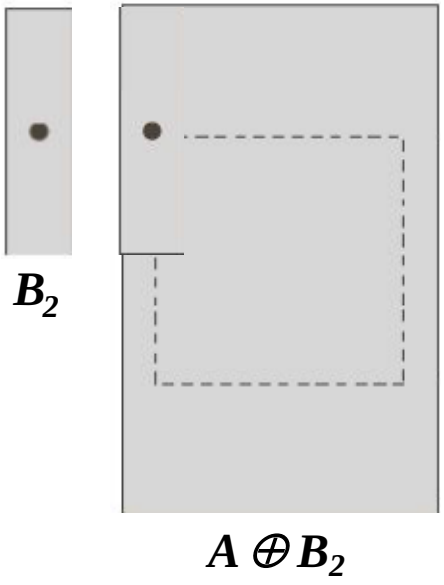


- ***Effects*** *(foreground=1, background=0)*
  - *If current pixel z is foreground, set all pixels under $(B)_z$ to foreground.*

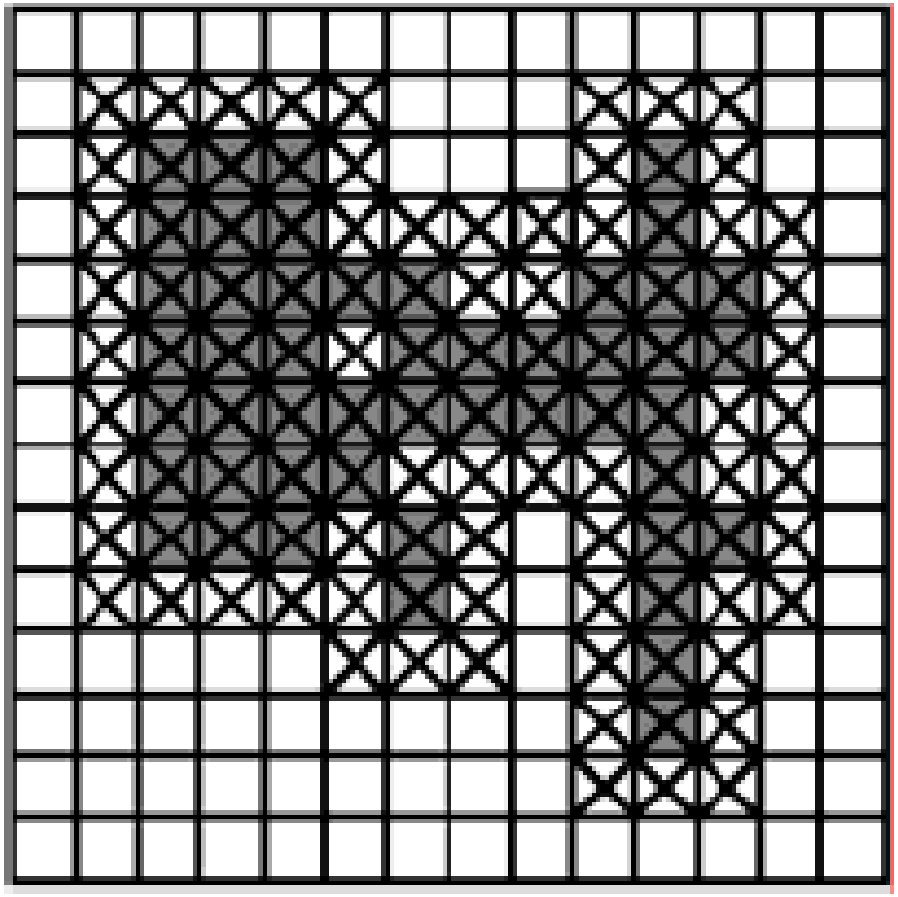
$\Rightarrow$ *Expand connected components*

$\Rightarrow$ *Grow features*

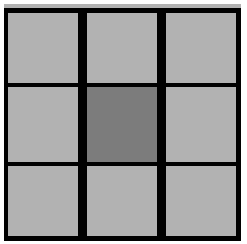
$\Rightarrow$ *Fill holes*



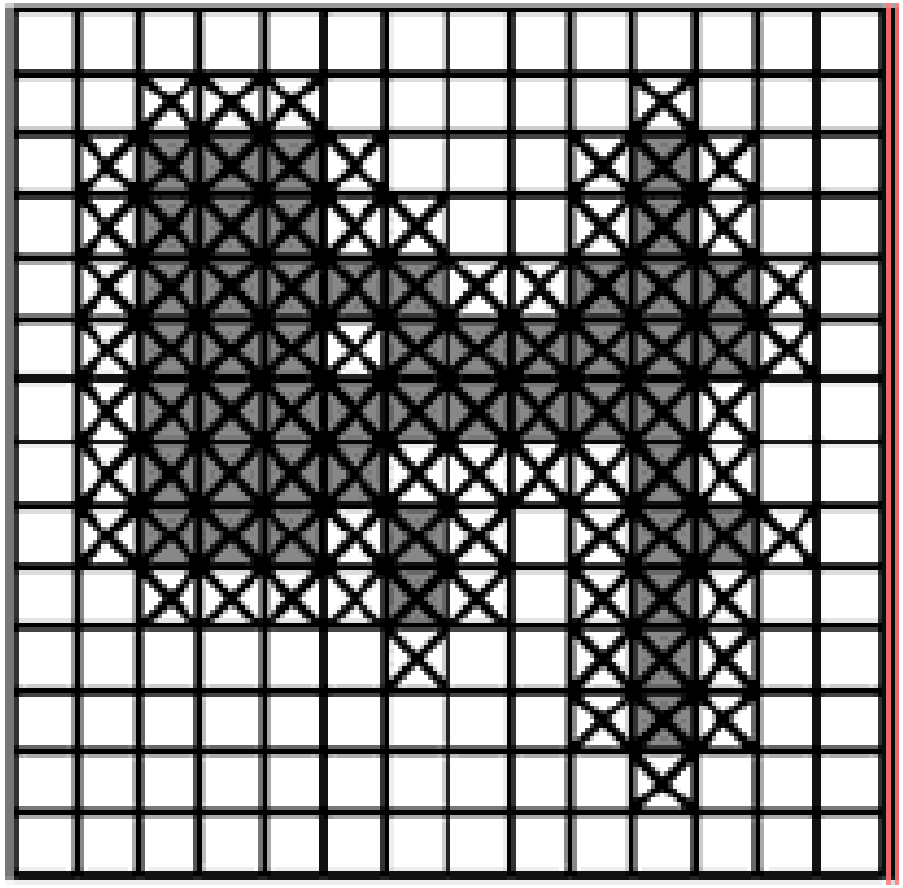
Dilation



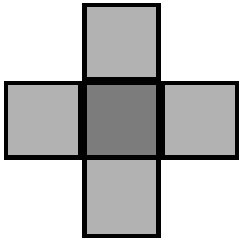
SE=



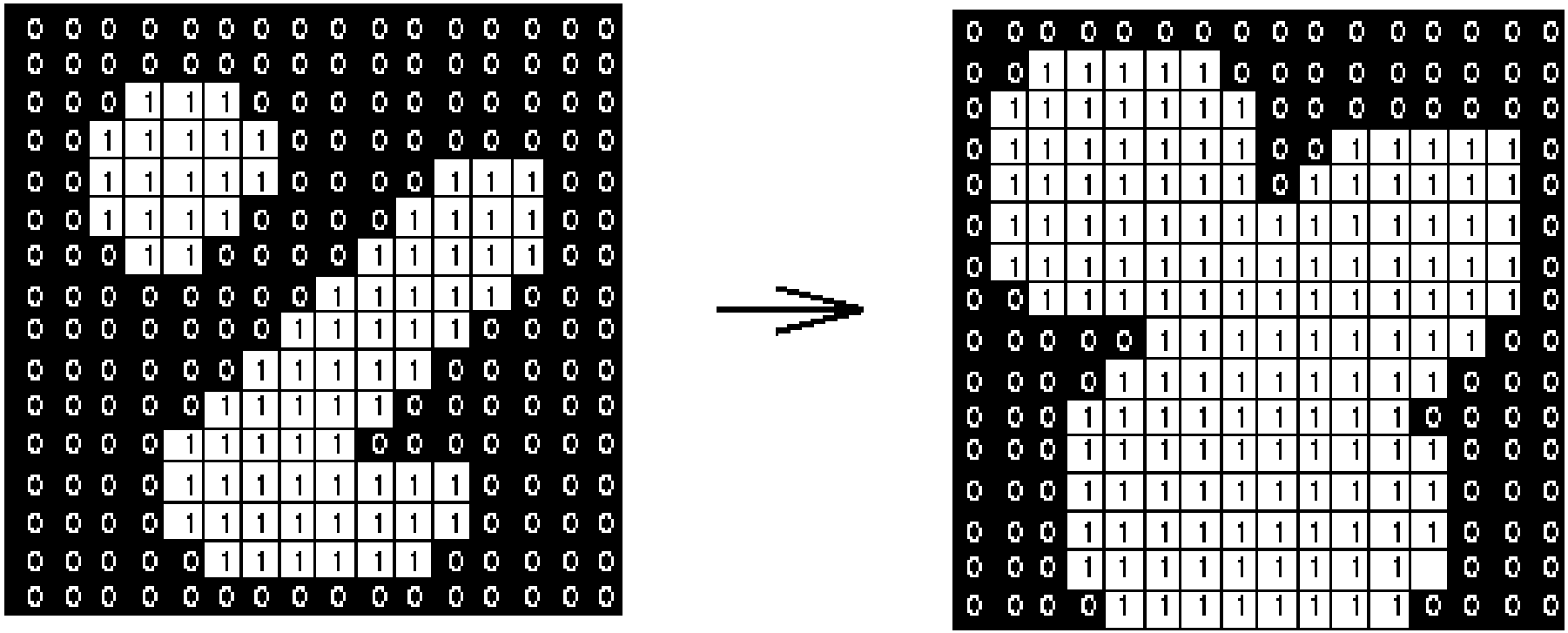
Dilation



SE=

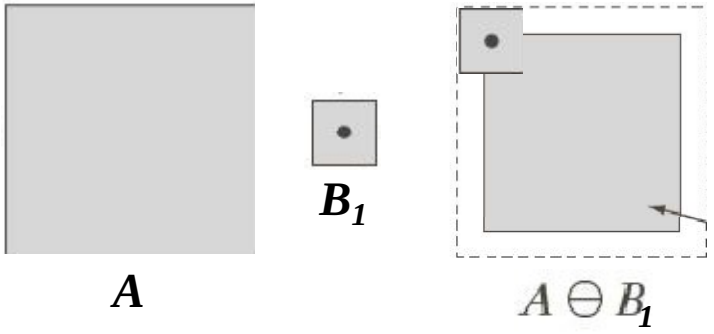


Dilation



Erosion

- ***Definition***
  - ***"The erosion of A by B is the set of all displacements z, such that $(B)_z$ is entirely contained in A"***

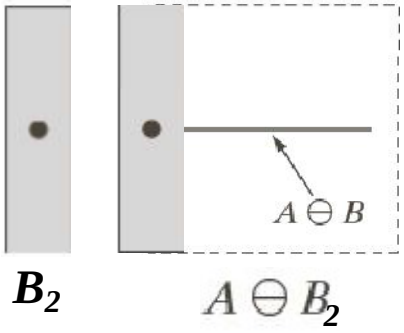


- ***Effects***
  - ***If not every pixel under $(B)_z$ is foreground, set the current pixel z to background.***

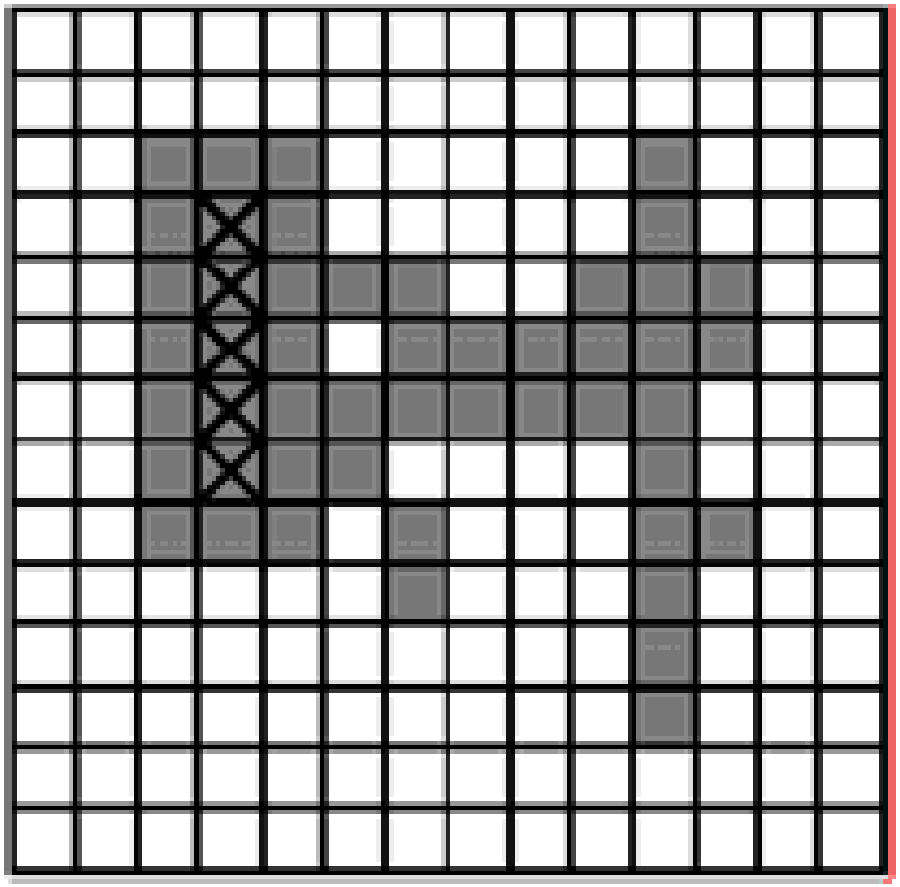
⇒ Erode connected components

⇒ Shrink features

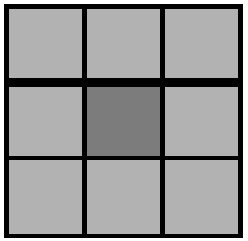
⇒ Remove bridges, branches, noise



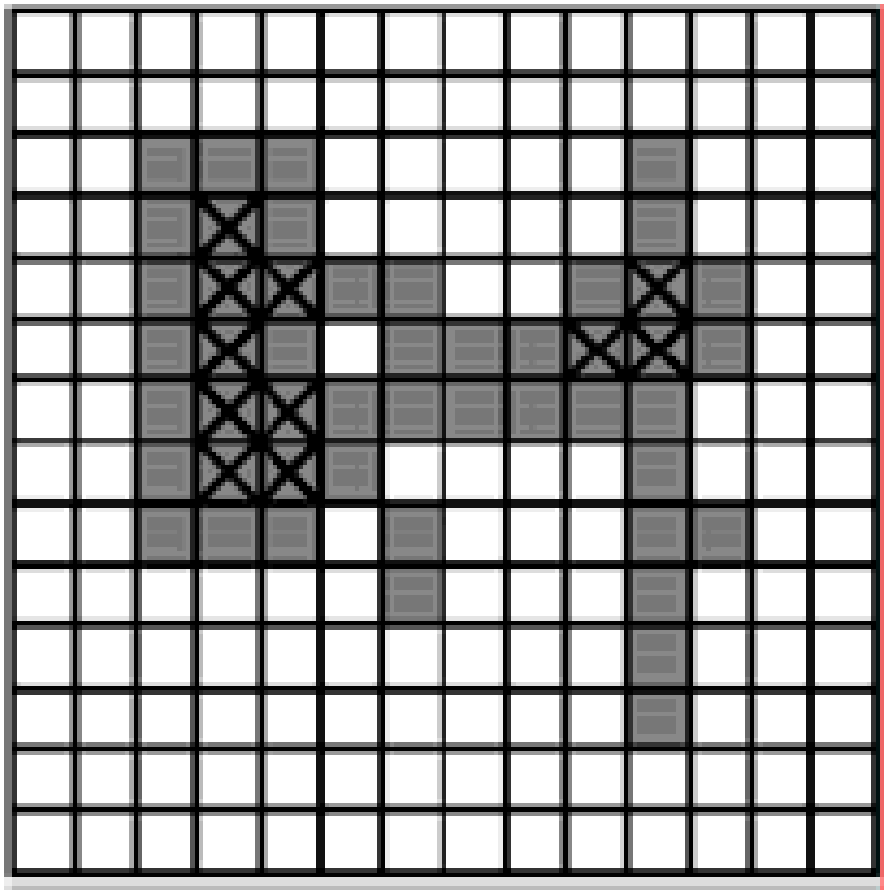
Erosion



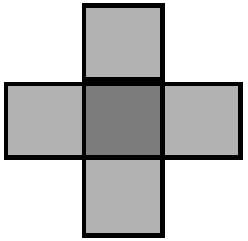
SE=



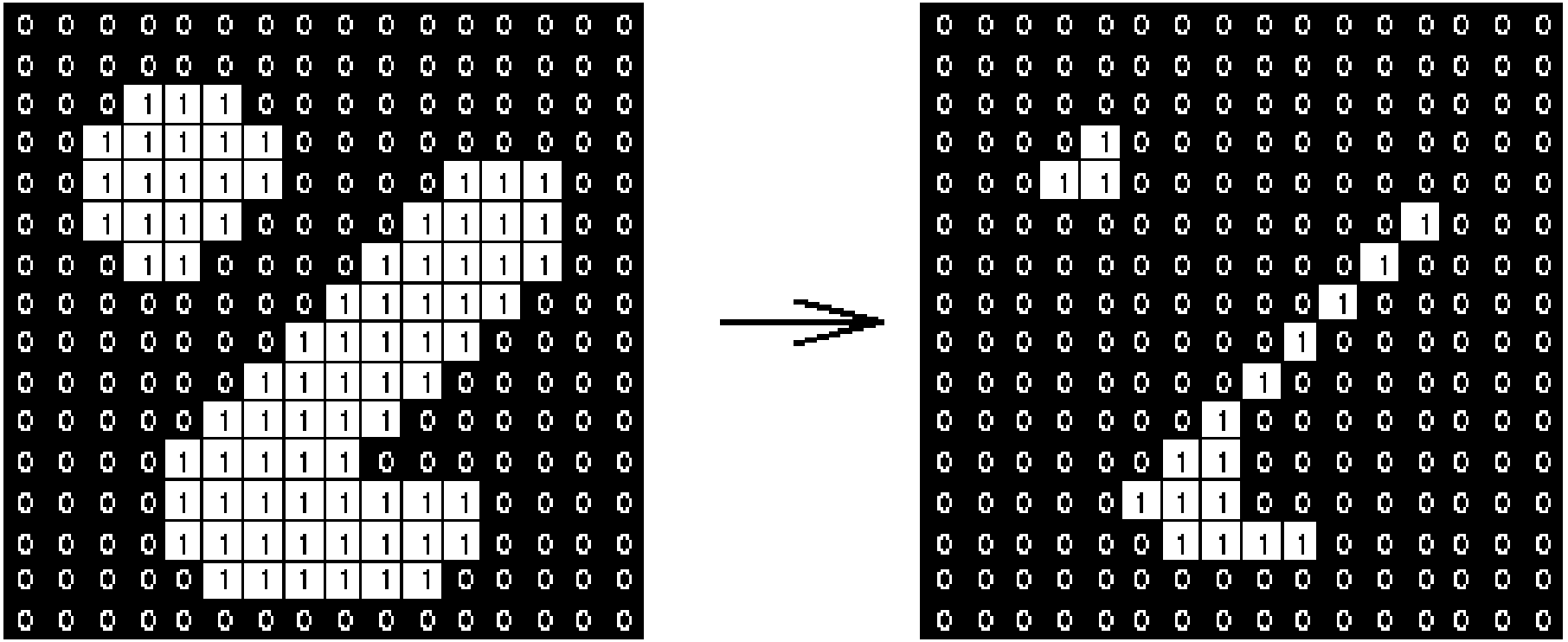
Erosion



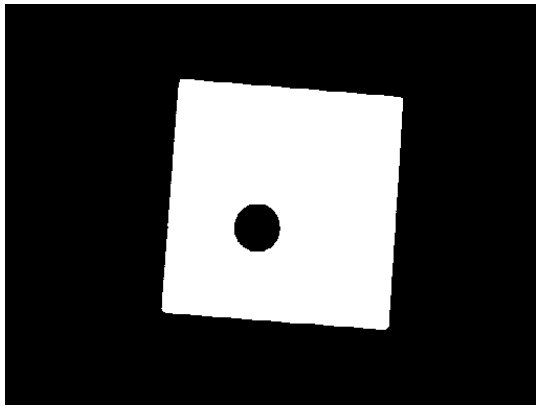
SE=



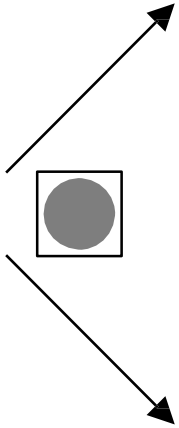
Erosion



Effects



Original image

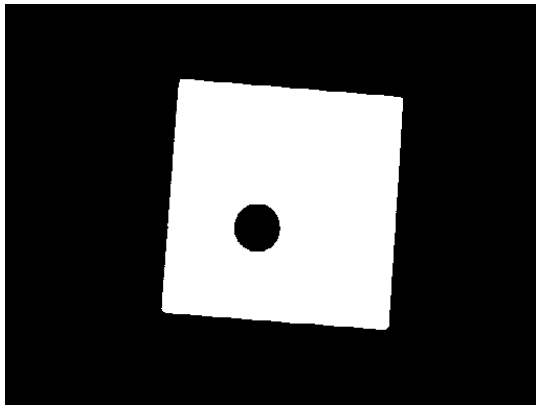


Dilation with circular structuring element

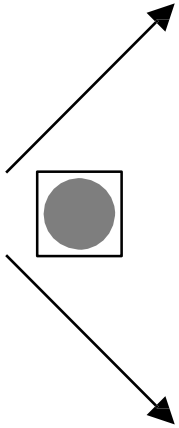
Erosion with circular structuring element

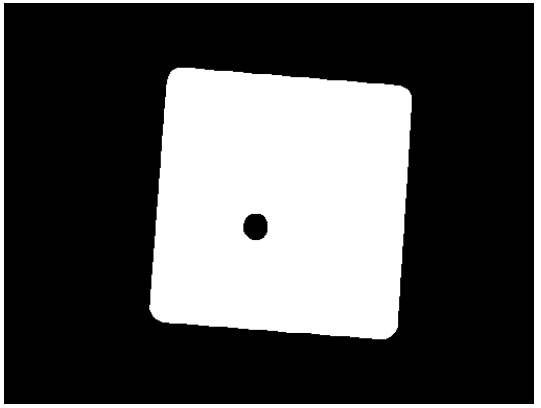
Image Source:
http://homepages.inf.ed.ac.uk/rbf/HIPR2/

Effects

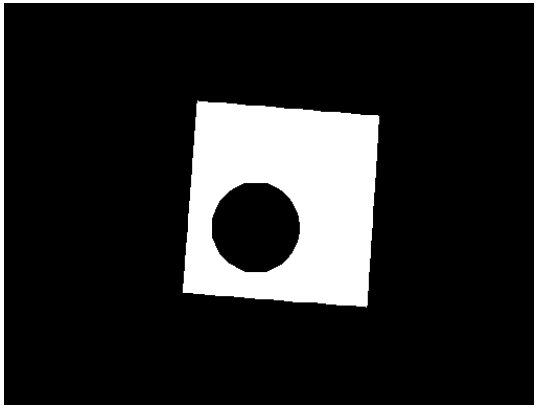


Original image





Dilation with circular structuring element



Erosion with circular structuring element

Image Source:
http://homepages.inf.ed.ac.uk/rbf/HIPR2/

Useful

- ***Erosion***
  - ***removal of structures of certain shape and size, given by SE***
- ***Dilation***
  - ***filling of holes of certain shape and size, given by SE***

Opening

Erosion followed by dilation, denoted ∘

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

- ***Eliminate protrusions***
- ***Break necks***
- ***Smooth contour***

Opening

- **Effect**
  - $A \circ B$ **is defined by the points that are reached if** B **is rolled around inside** A**.**

➔ **Remove small objects, keep original shape.**

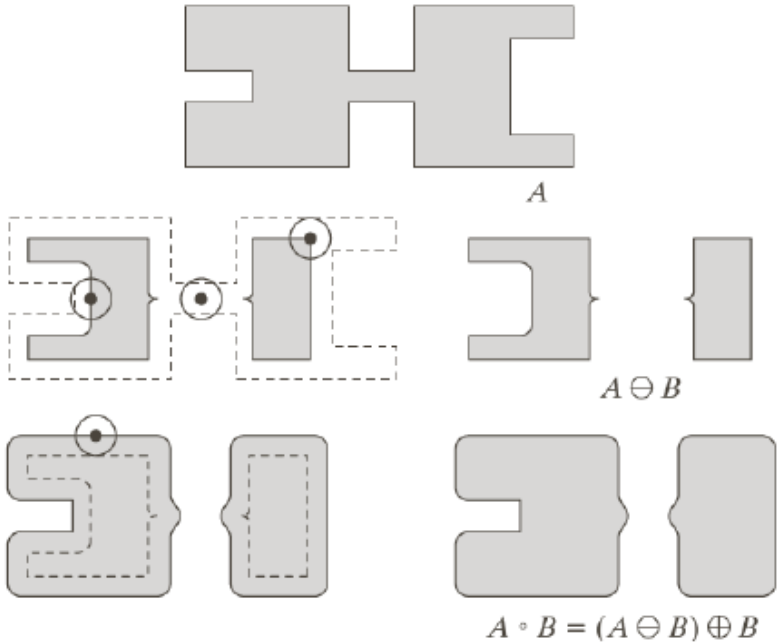
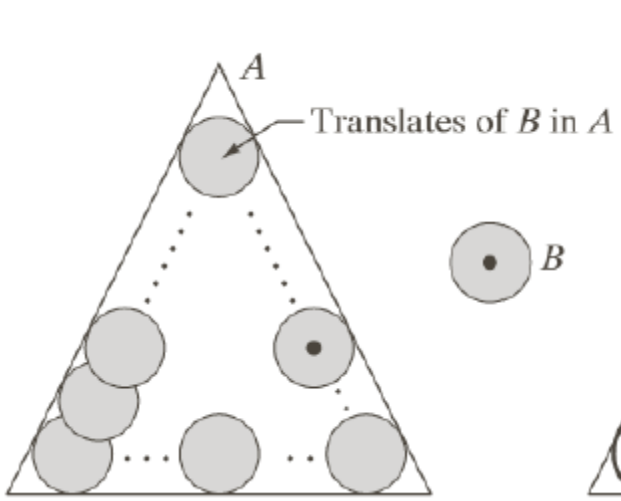
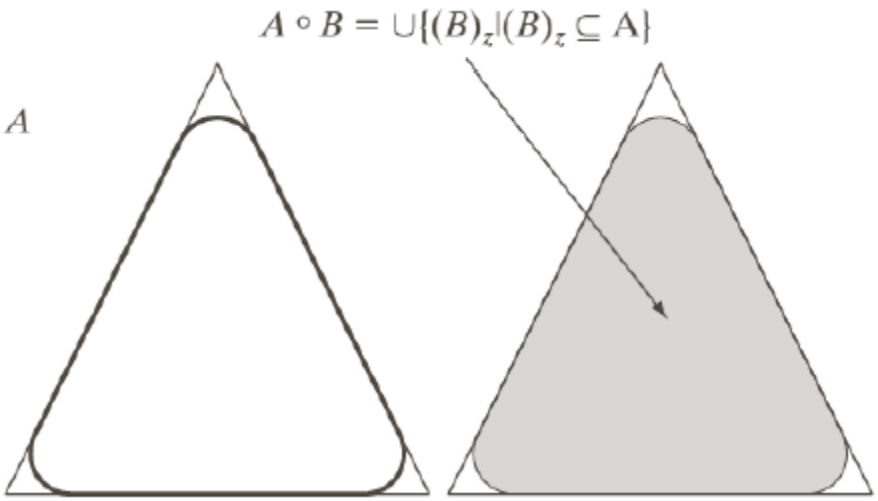


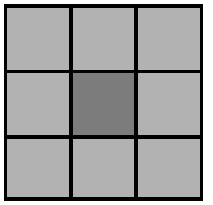
Image Source: R.C. Gonzales & R.E. Woods

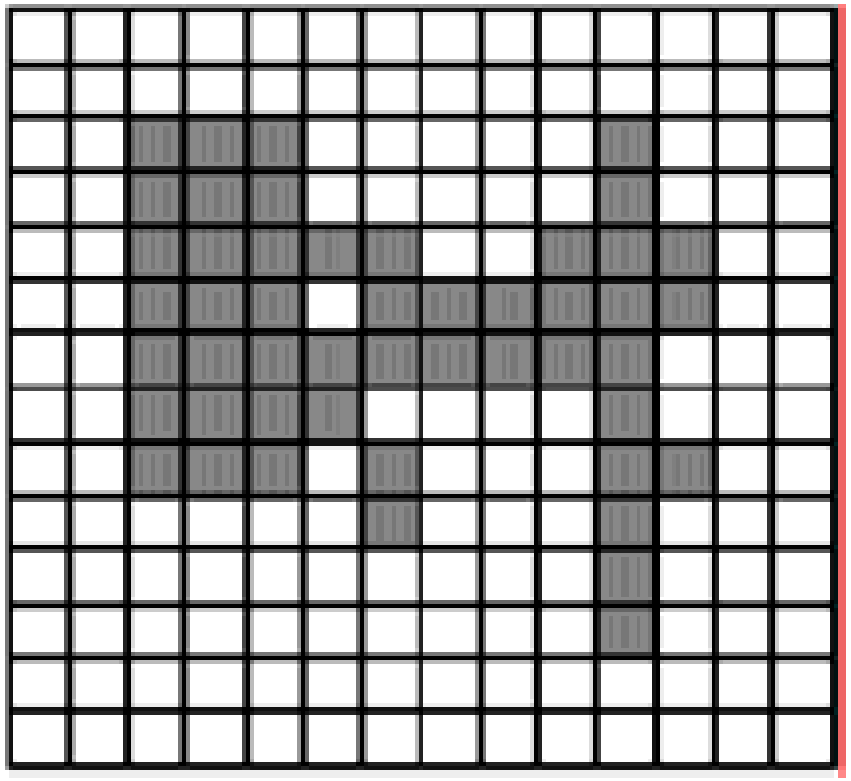




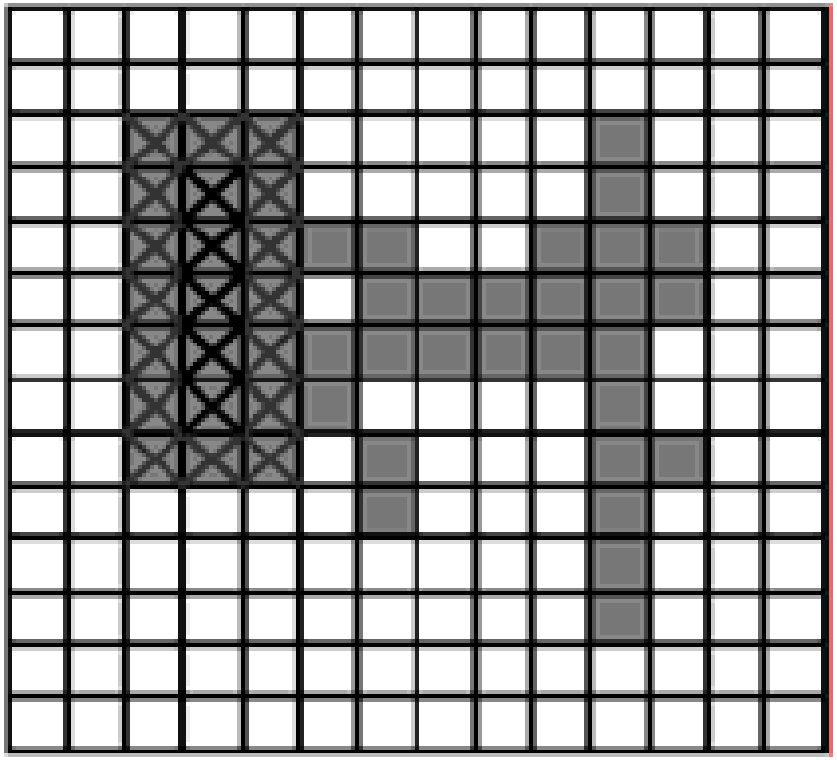
Opening

B=





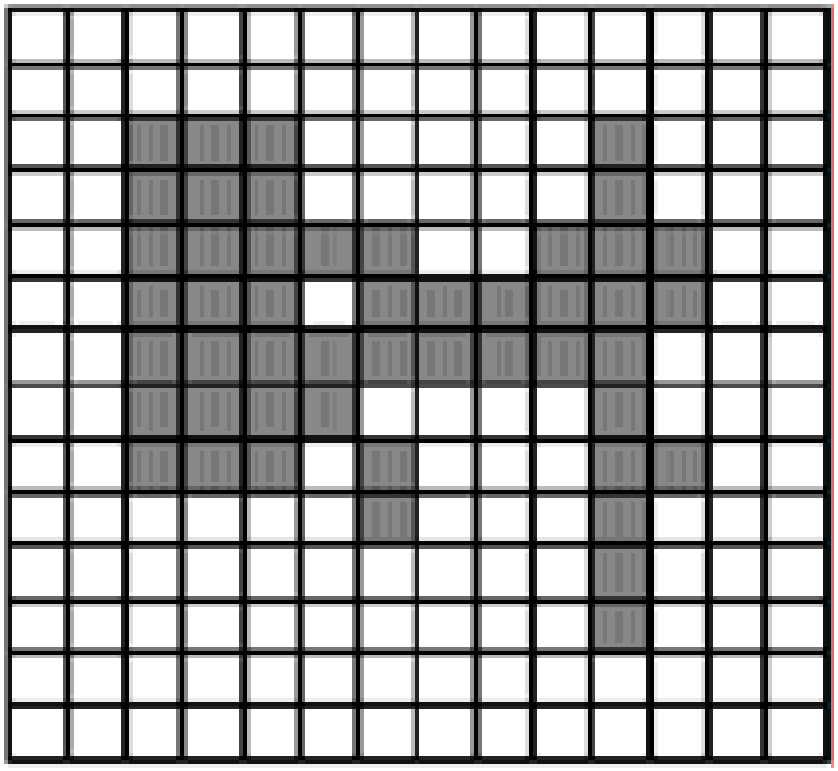
A



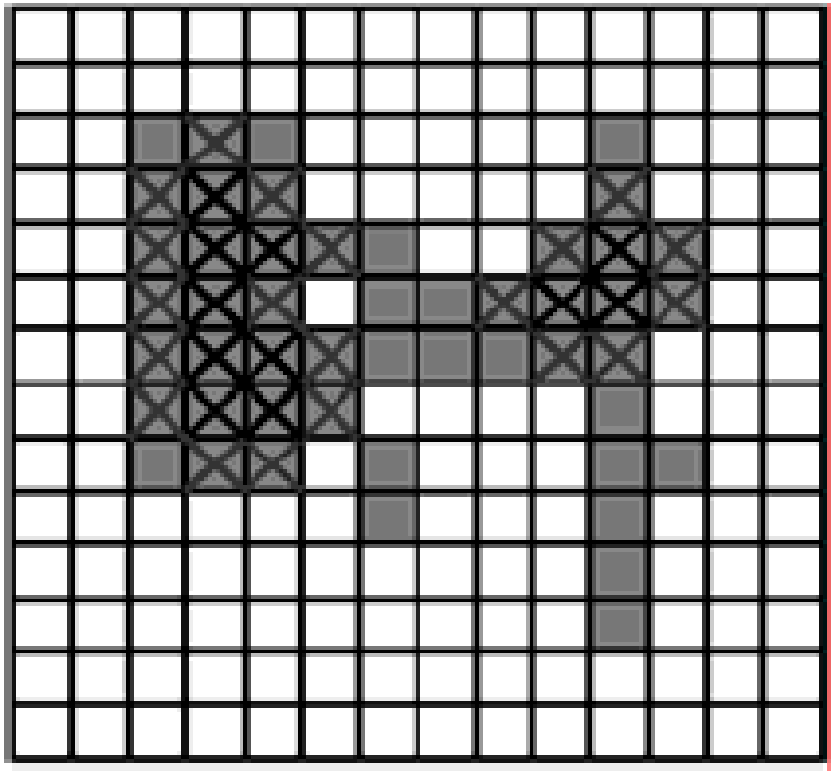
$A \ominus B$ $A \circ B$

Opening

B=

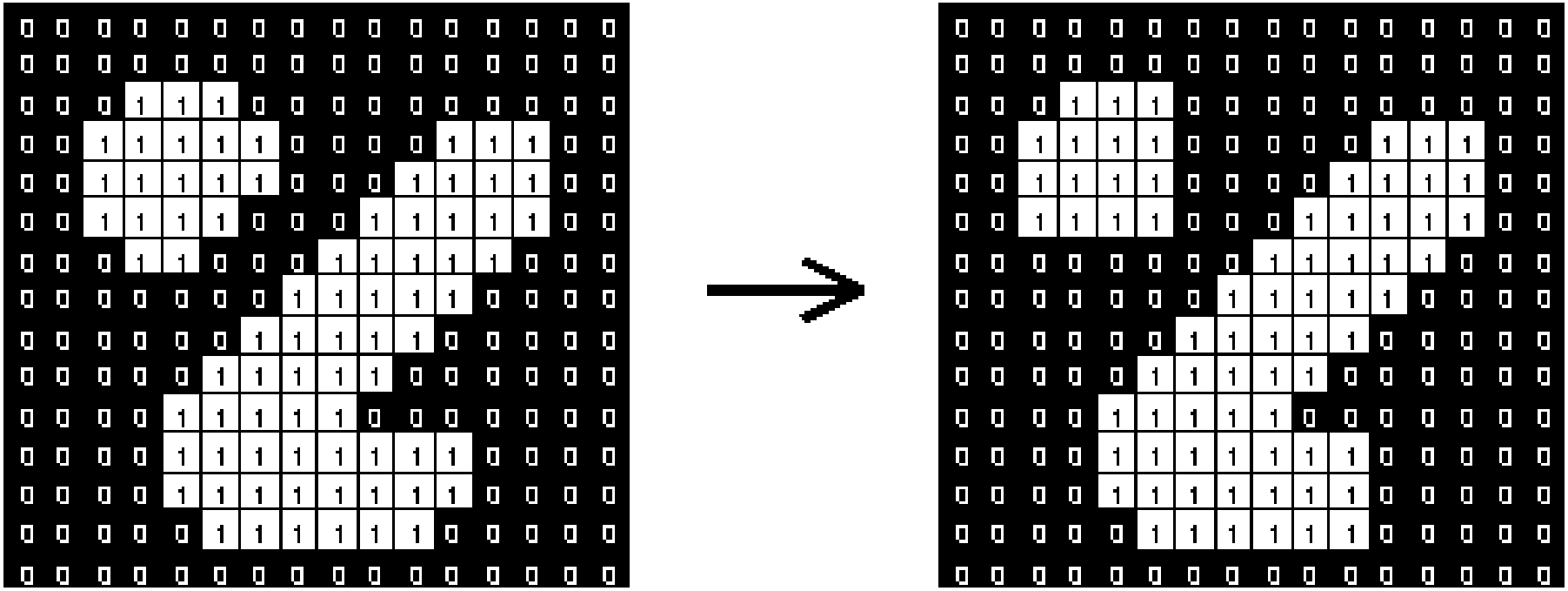


A



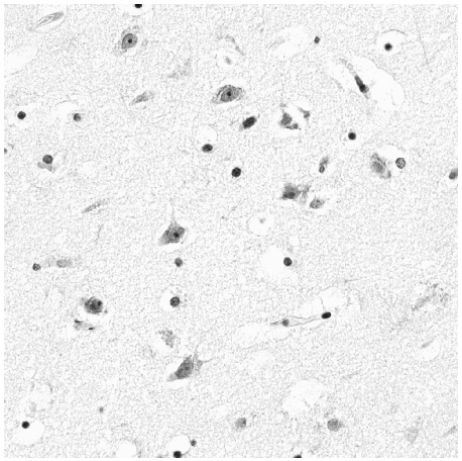
A⊖B A∘B

Opening

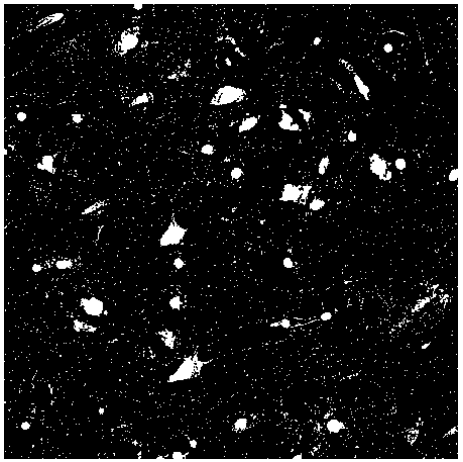


Effect of Opening

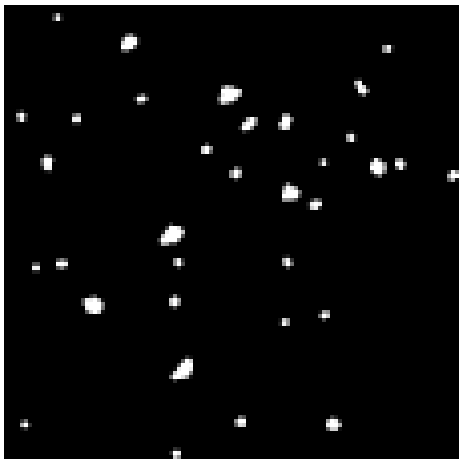
- ***Feature selection through size of structuring element***



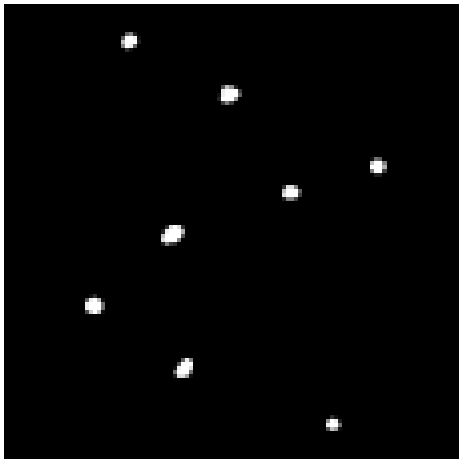
Original image



Thresholded



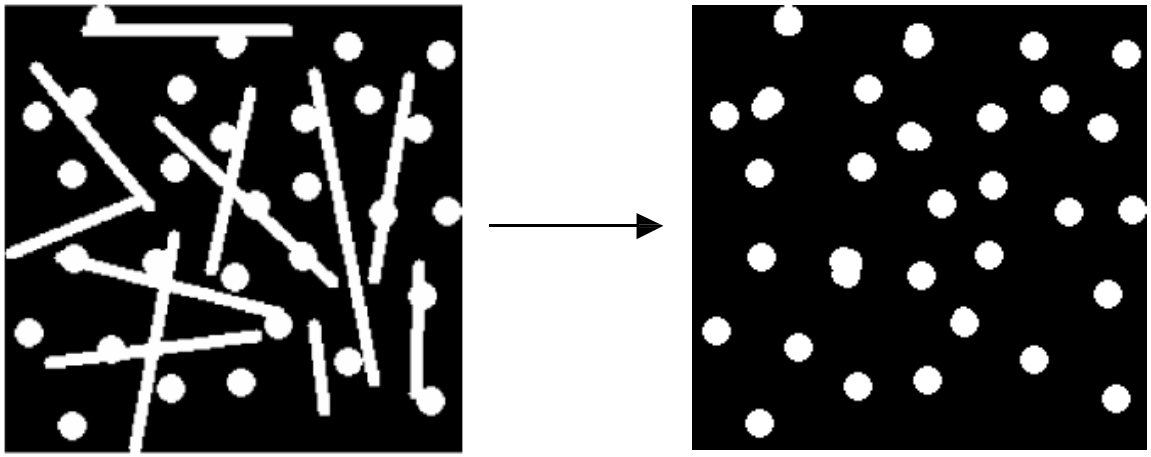
Opening with small structuring element



Opening with larger structuring element

Effect of Opening

- ***Feature selection through shape of structuring element***



Input Image

Opening with circular structuring element

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***

Closing

- **Effect**
  - $A \bullet B$ **is defined by the points that are reached if** B **is rolled around on the outside of** A**.**

➔ **Fill holes, keep original shape.**

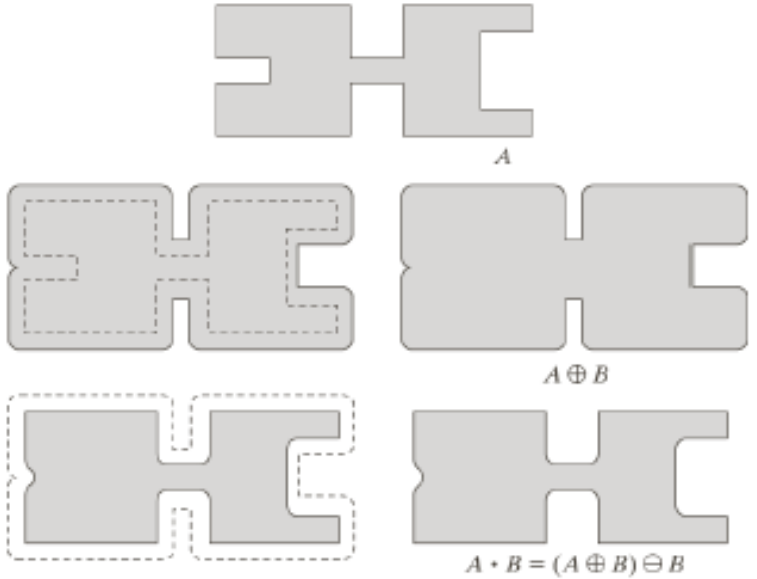
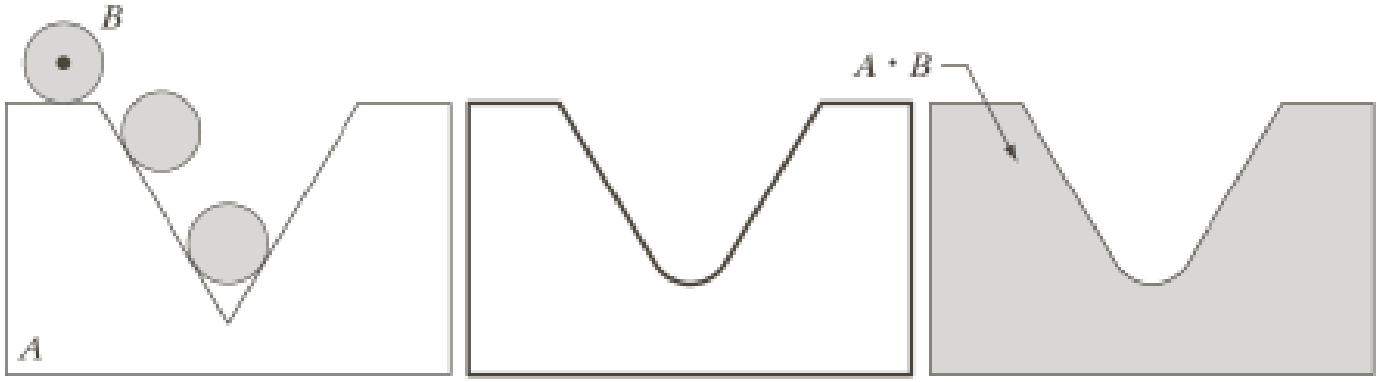
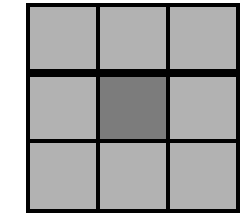


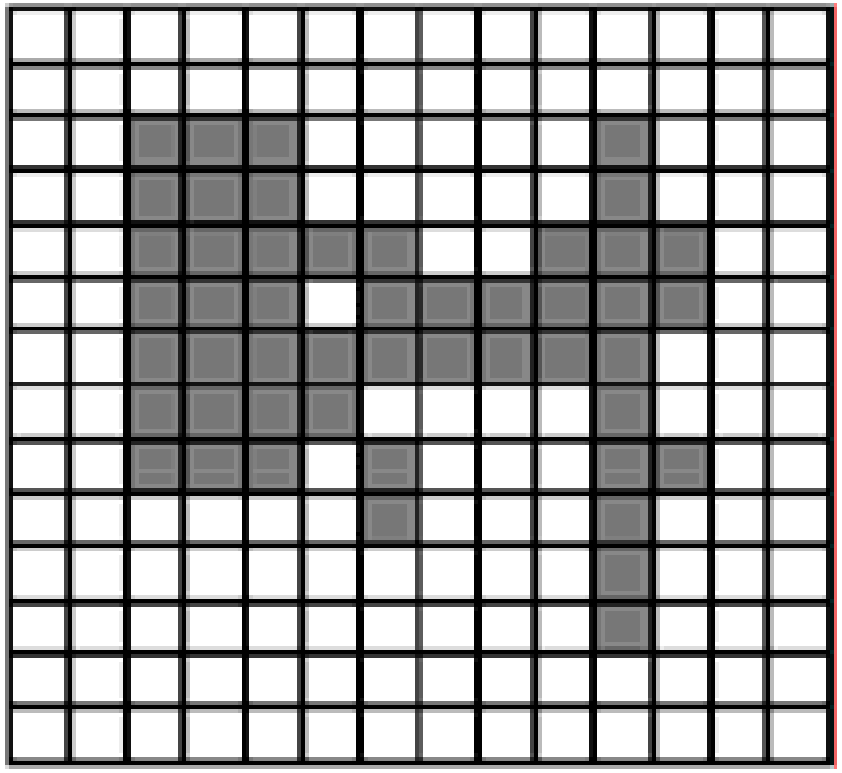
Image Source: R.C. Gonzales & R.E. Woods



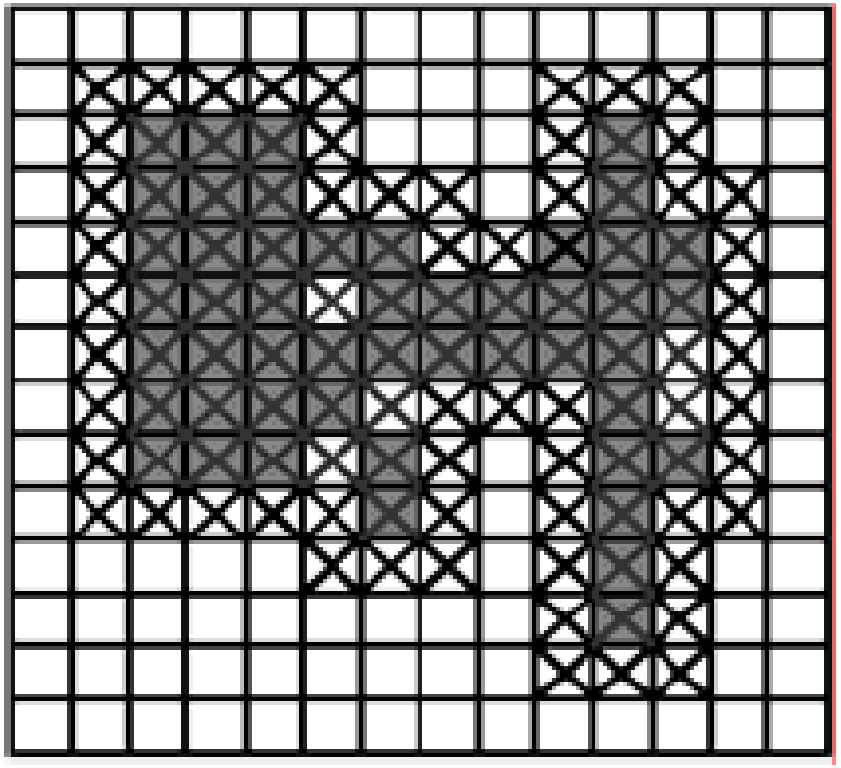
Closing

B=





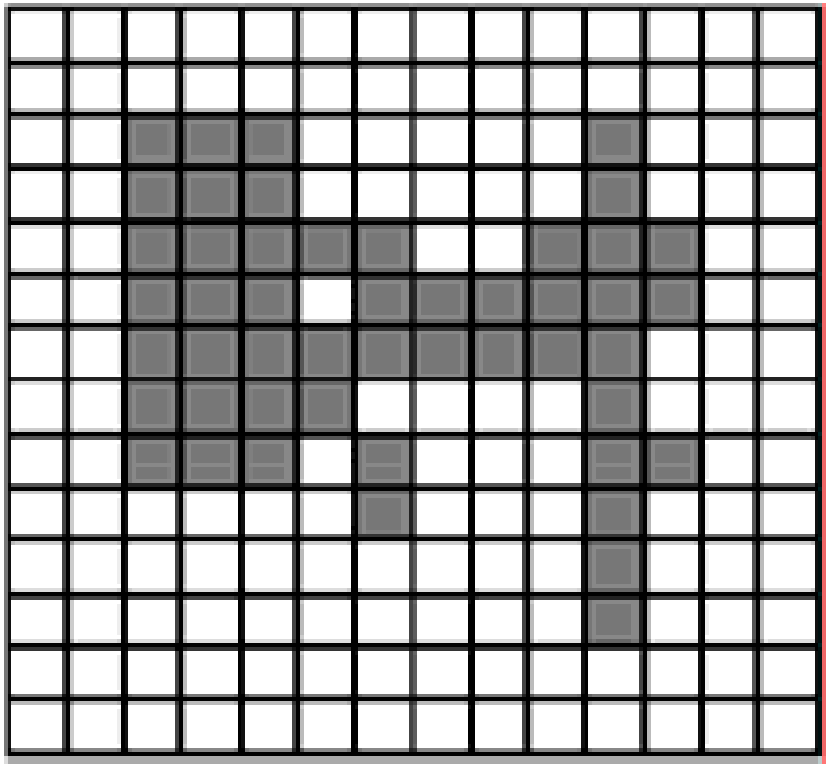
A



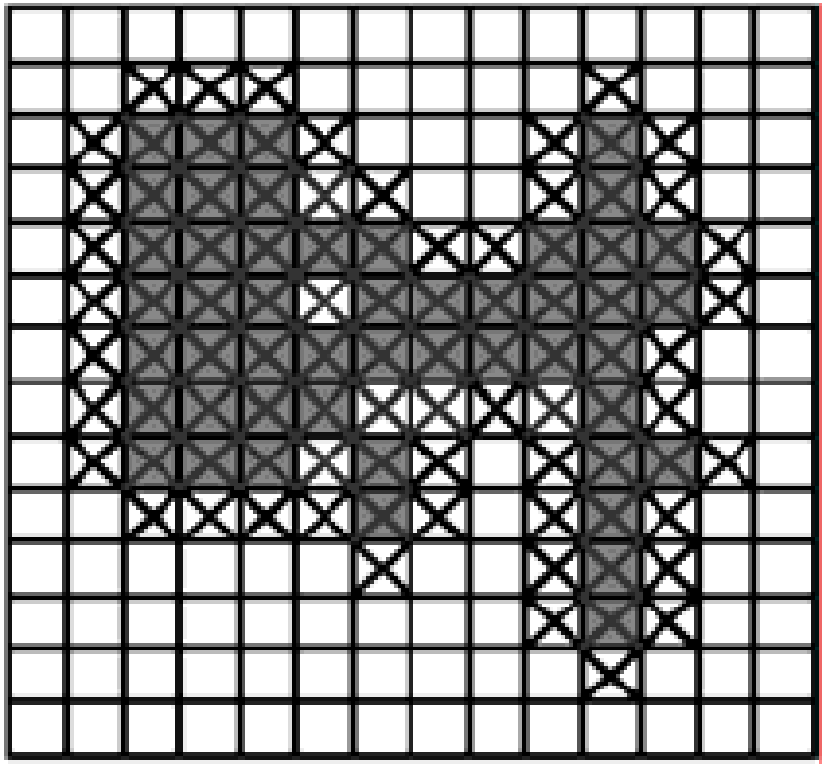
A⊕B A • B

Closing

B=

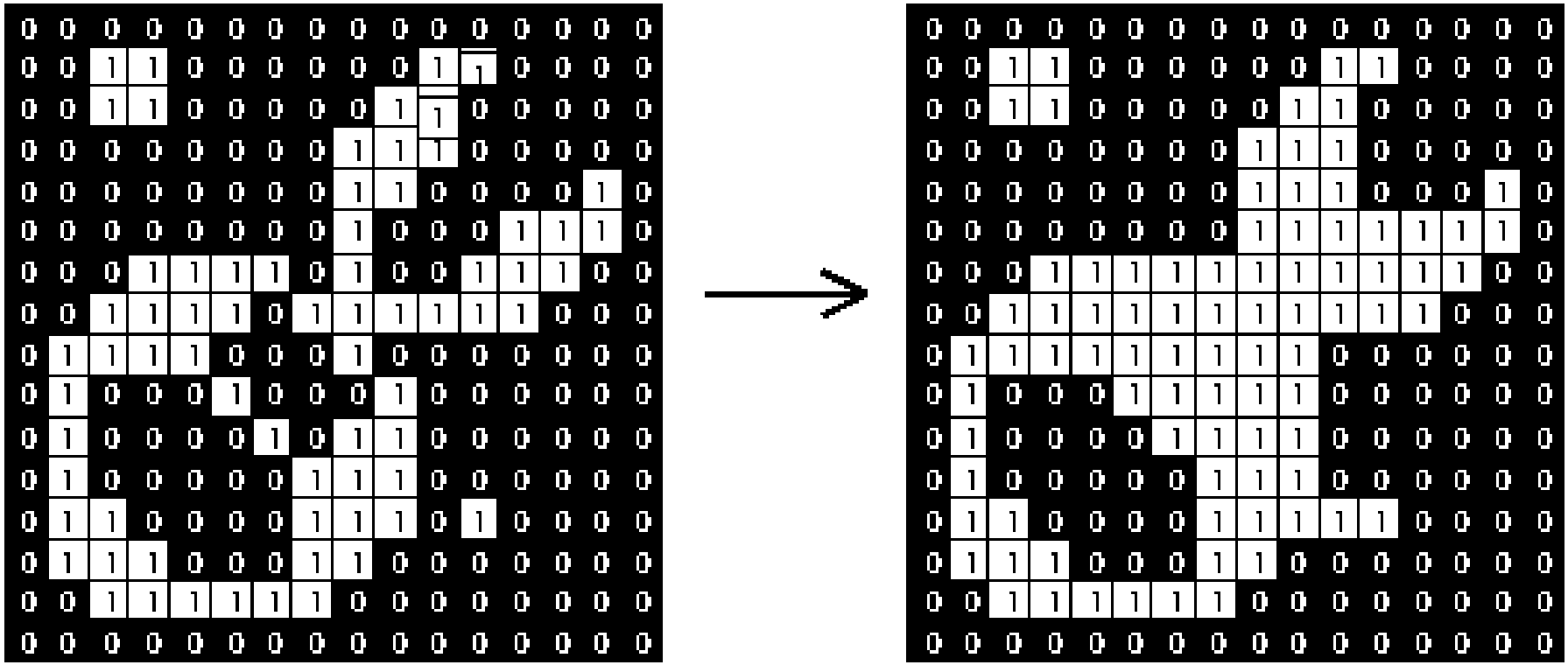


A



A⊕B A • B

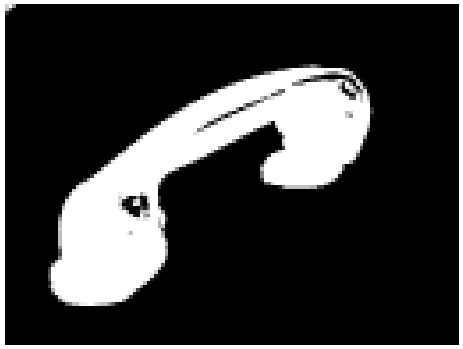
Closing



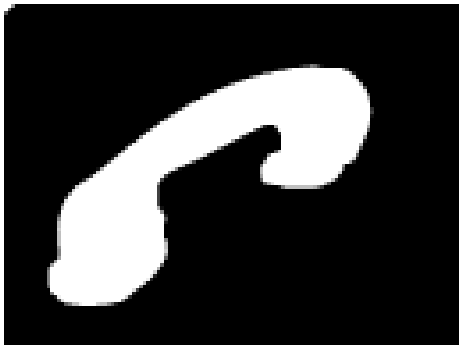
Effect of Closing

- ***Fill holes in thresholded image (e.g. due to specularities)***

Original image

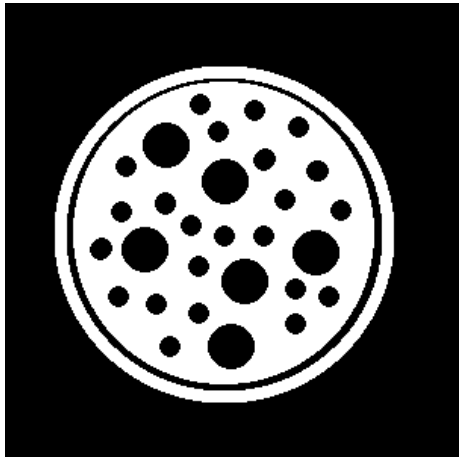


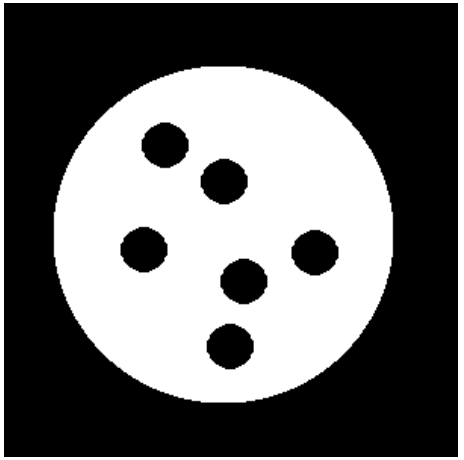
Thresholded



Closing with circular structuring element

Size of structuring element determines which structures are selectively filled.





Example Application: Opening + Closing

Original image

Opening

Closing

Structuring element

Erosion

Dilation

Dilation

Erosion

Eroded image

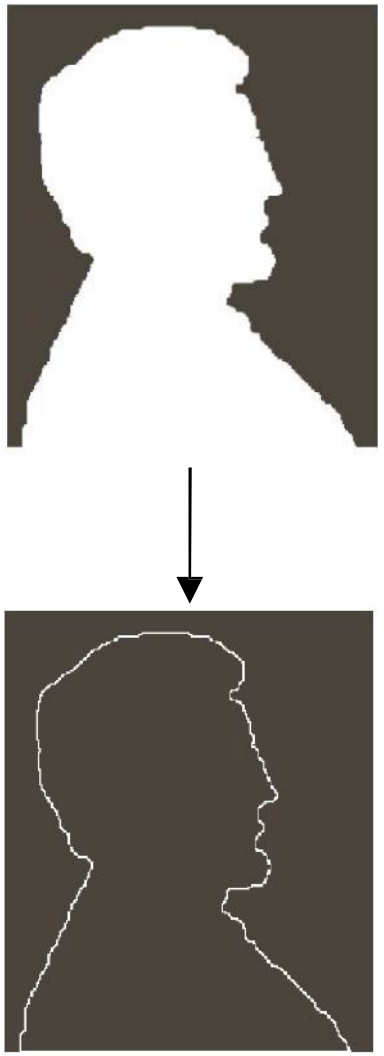
Dilated image

Morphological Boundary Extraction

- ***Definition***
  - ***First erode A by B, then subtract the result from the original A.***

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

- ***Effects***
  - ***If a 3x3 structuring element is used, this results in a boundary that is exactly 1 pixel thick.***

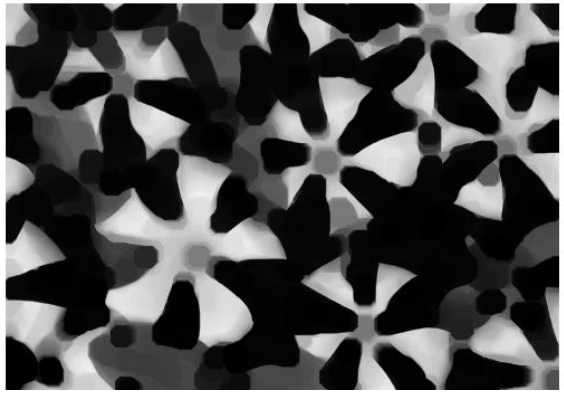


Morphology Operators on Grayscale Images

- ***Dilation and erosion typically performed on binary images.***
- ***If image is grayscale: for dilation take the neighborhood max, for erosion take the min.***

Original

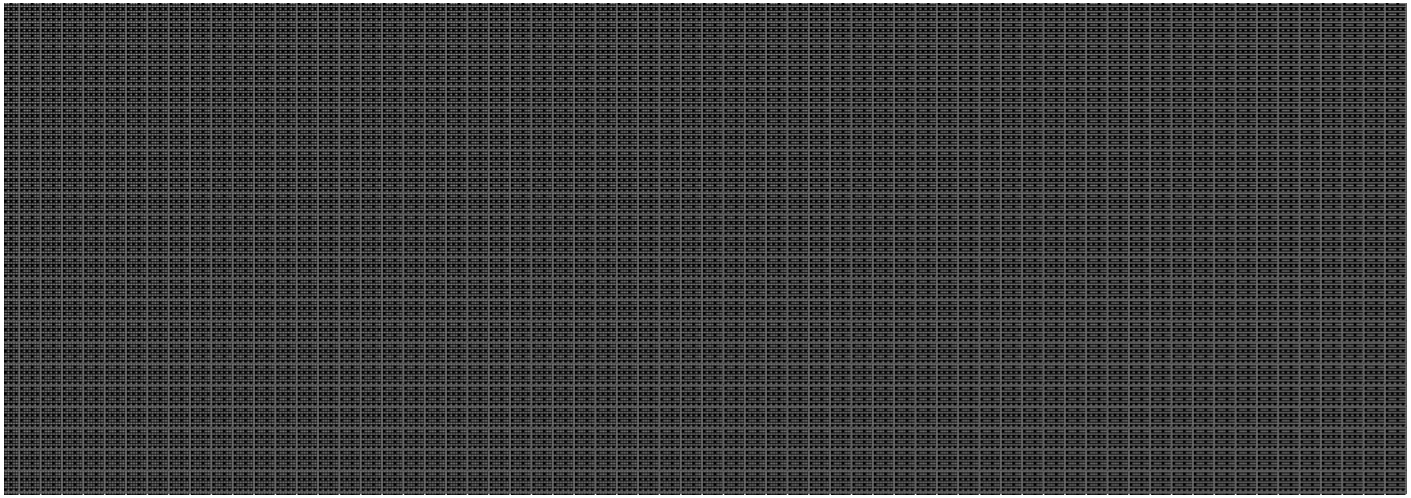
Dilated

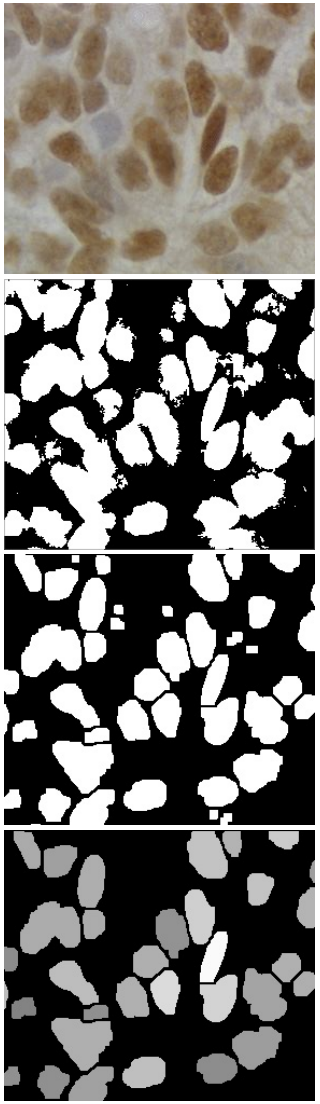


Eroded

- ***In dilation, image becomes lighter, and dark details are reduced.***
- ***In erosion, image becomes darker, and light details are reduced.***

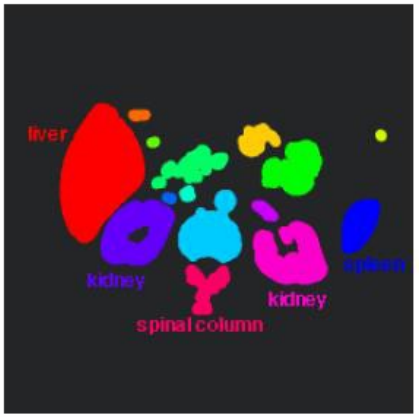
Outline of Today's Lecture



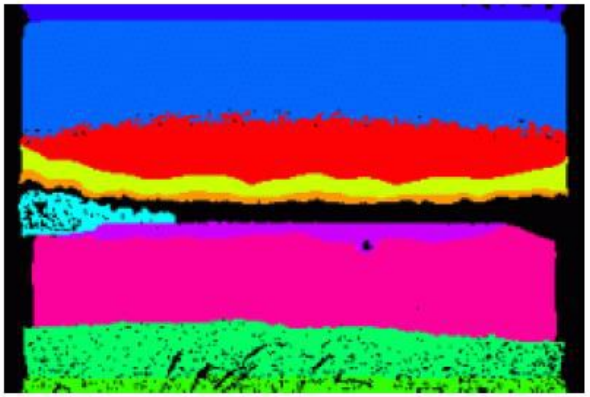


- ***Extract individual objects***
  - ***Connected Components Labeling***

Connected Components Examples



connected components of 1's from thresholded image



connected components of cluster labels

Connected Components Labeling

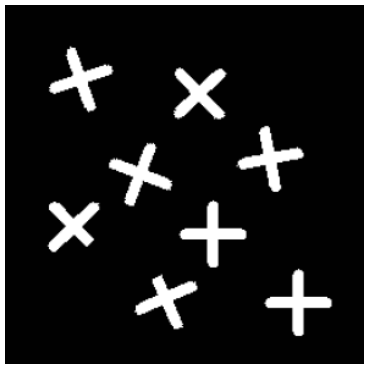
- ***Goal: Identify distinct regions***

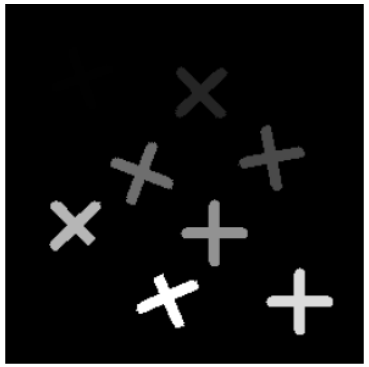
1	1	0	1	1	1	0	1
1	1	0	1	0	1	0	1
1	1	1	1	0	0	0	1
0	0	0	0	0	0	0	1
1	1	1	1	0	1	0	1
0	0	0	1	0	1	0	1
1	1	0	1	0	0	0	1
1	1	0	1	0	1	1	1

Binary image

1	1	0	1	1	1	0	2
1	1	0	1	0	1	0	2
1	1	1	1	0	0	0	2
0	0	0	0	0	0	0	2
3	3	3	3	0	4	0	2
0	0	0	3	0	4	0	2
5	5	0	3	0	0	0	2
5	5	0	3	0	2	2	2

Connected components labeling





Recursive Labeling
Connected Component Exploration

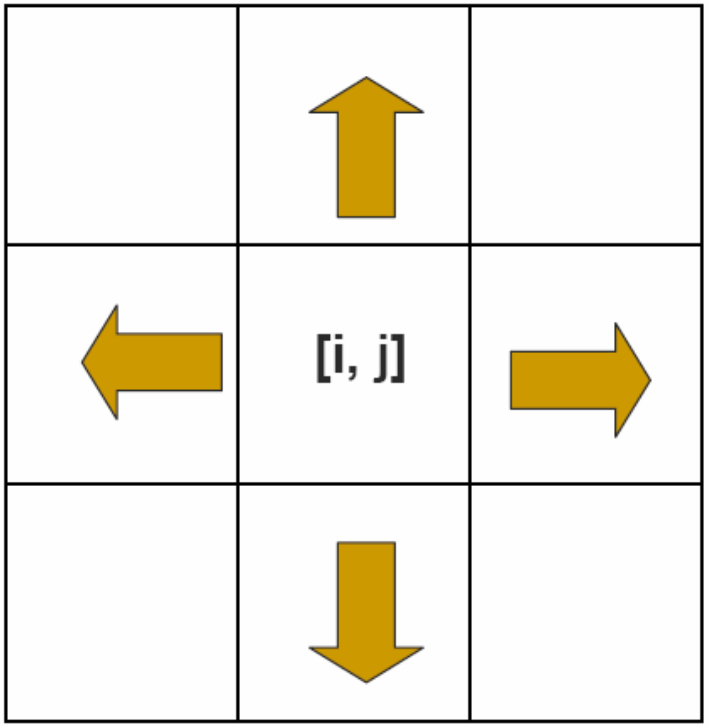
```
Procedure Label (Pixel)
BEGIN
  Mark(Pixel) <- Marker;
  FOR neighbor in Neighbors(Pixel) DO
     IF Image (neighbor) = 1 AND Mark(neighbor)=nil THEN
           Label(neighbor)
END

BEGIN Main
  Marker <- 0;
  FOR Pixel in Image DO
     IF Image(Pixel) = 1 AND Mark(Pixel)=nil THEN
           BEGIN
              Marker <- Marker + 1;
              Label(Pixel);
           END;
END
```

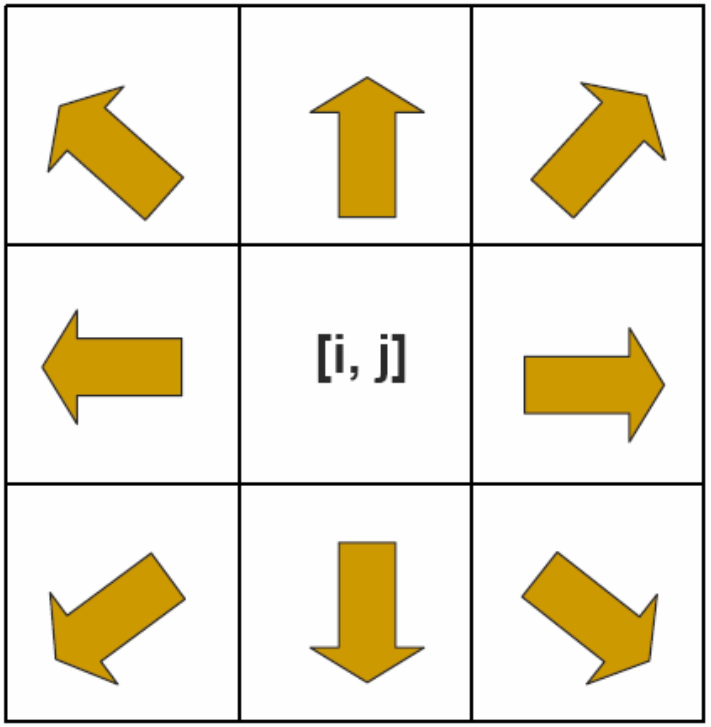
Globals:
Marker: integer
Mark: Matrix same size as Image,
initialized to NIL

Connectedness

- ***Which pixels are considered neighbors?***



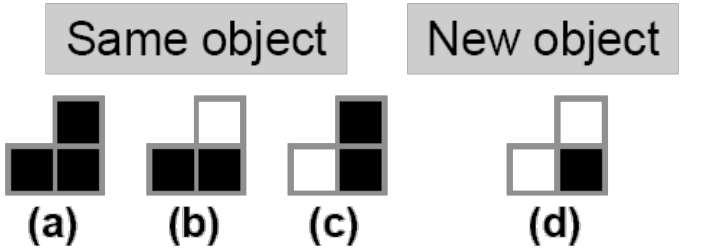
4-connected



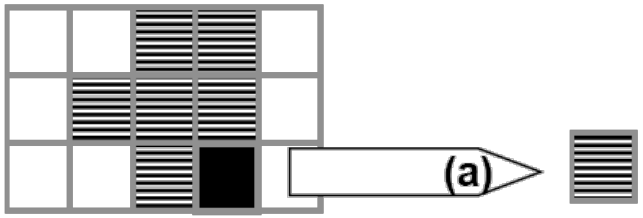
8-connected

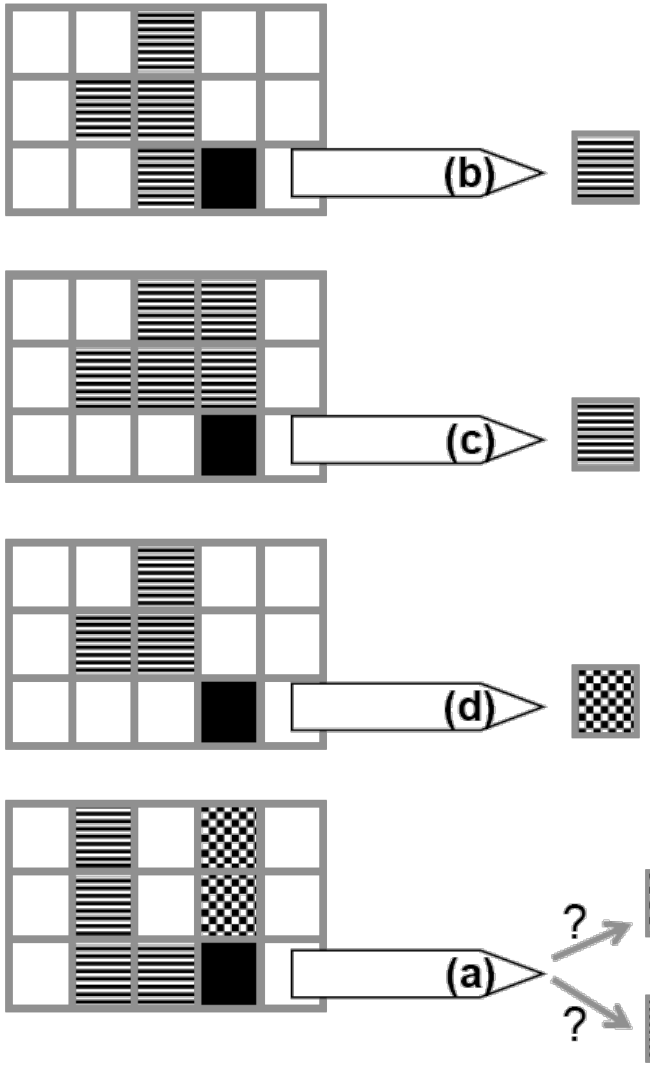
Sequential Connected Components

- Labeling a pixel only requires to consider its prior and superior neighbors.
- It depends on the type of connectivity used for foreground (4-connectivity here).



What happens in these cases?



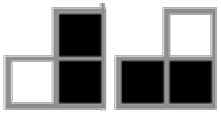


Equivalence table

Sequential Connected Components

- ***Process the image from left to right, top to bottom:***

  1.) If the next pixel to process is 1



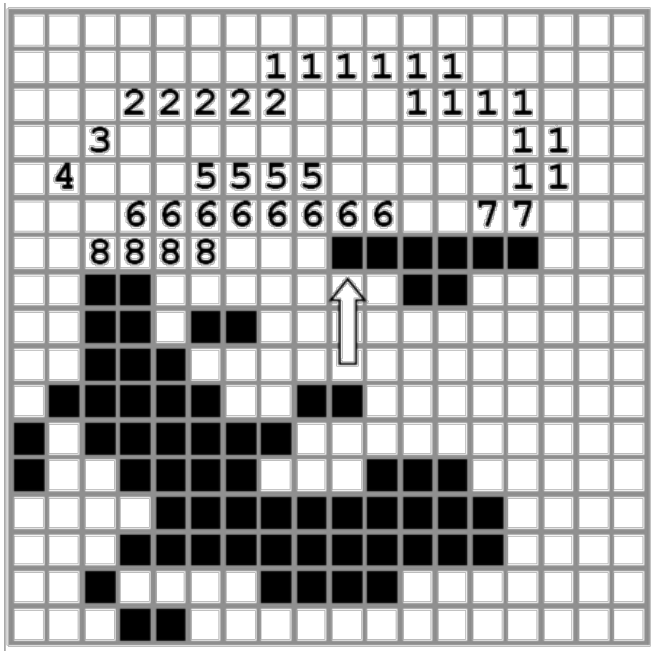
i.) If only one of its neighbors (top or left) is 1, copy its label.

ii.) If both are 1 and have the same label, copy it.

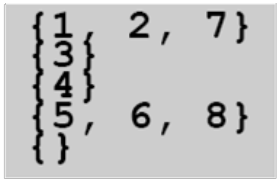
iii.) If they have different labels

- ***Copy the label from the left.***
- ***Update the equivalence table.***

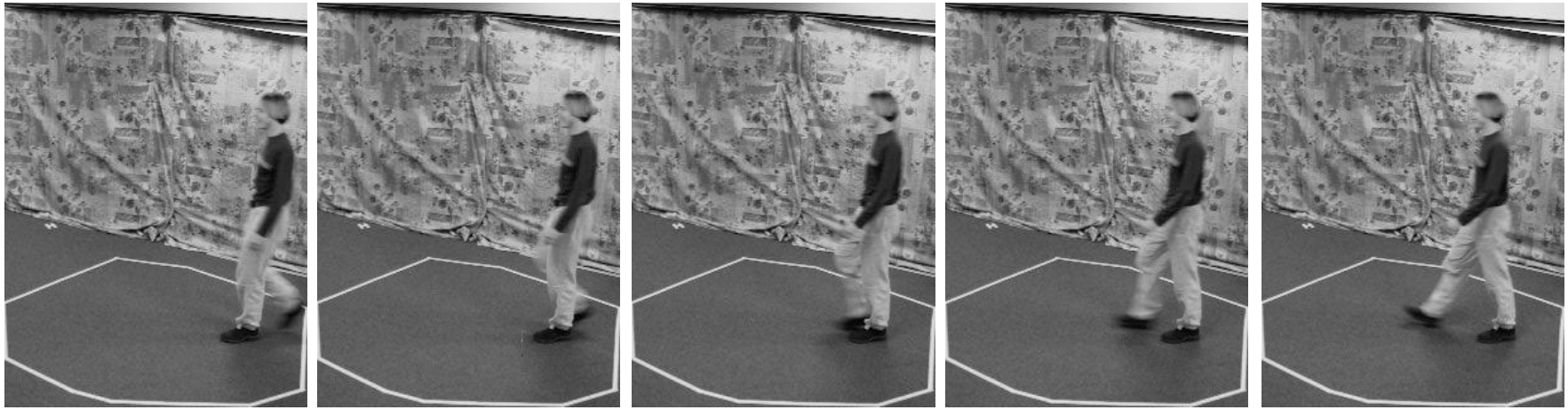
iv.) Otherwise, assign a new label.



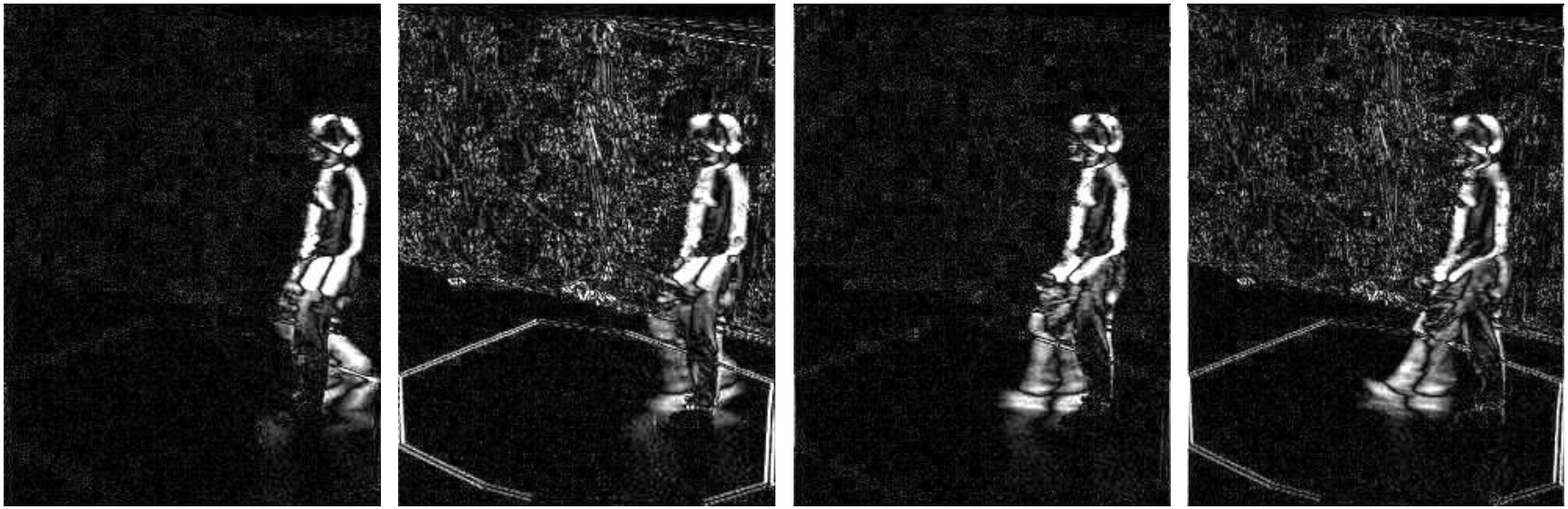
- ***Re-label with the smallest of equivalent labels***

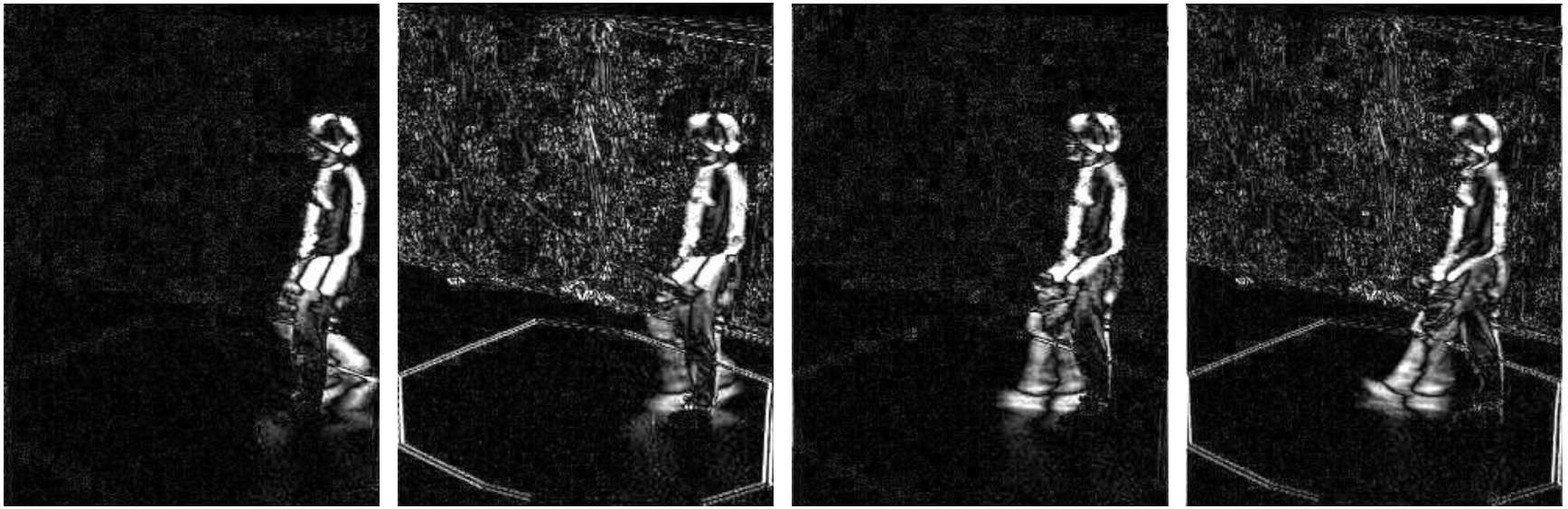


Application: Blob Tracking

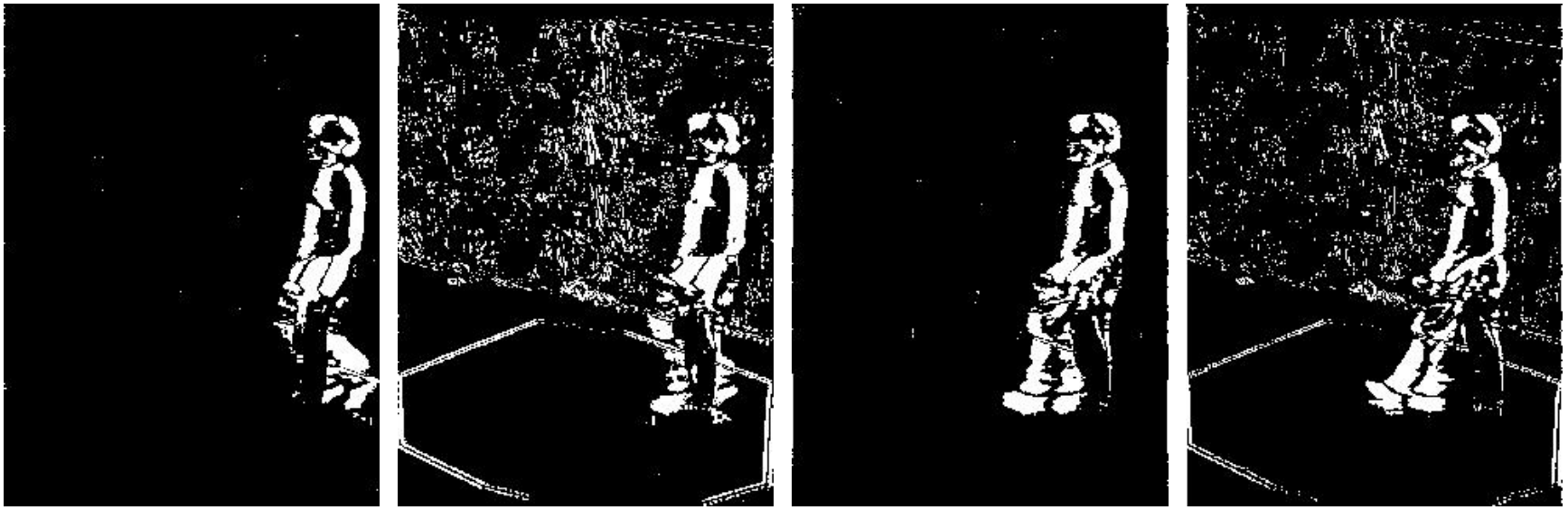


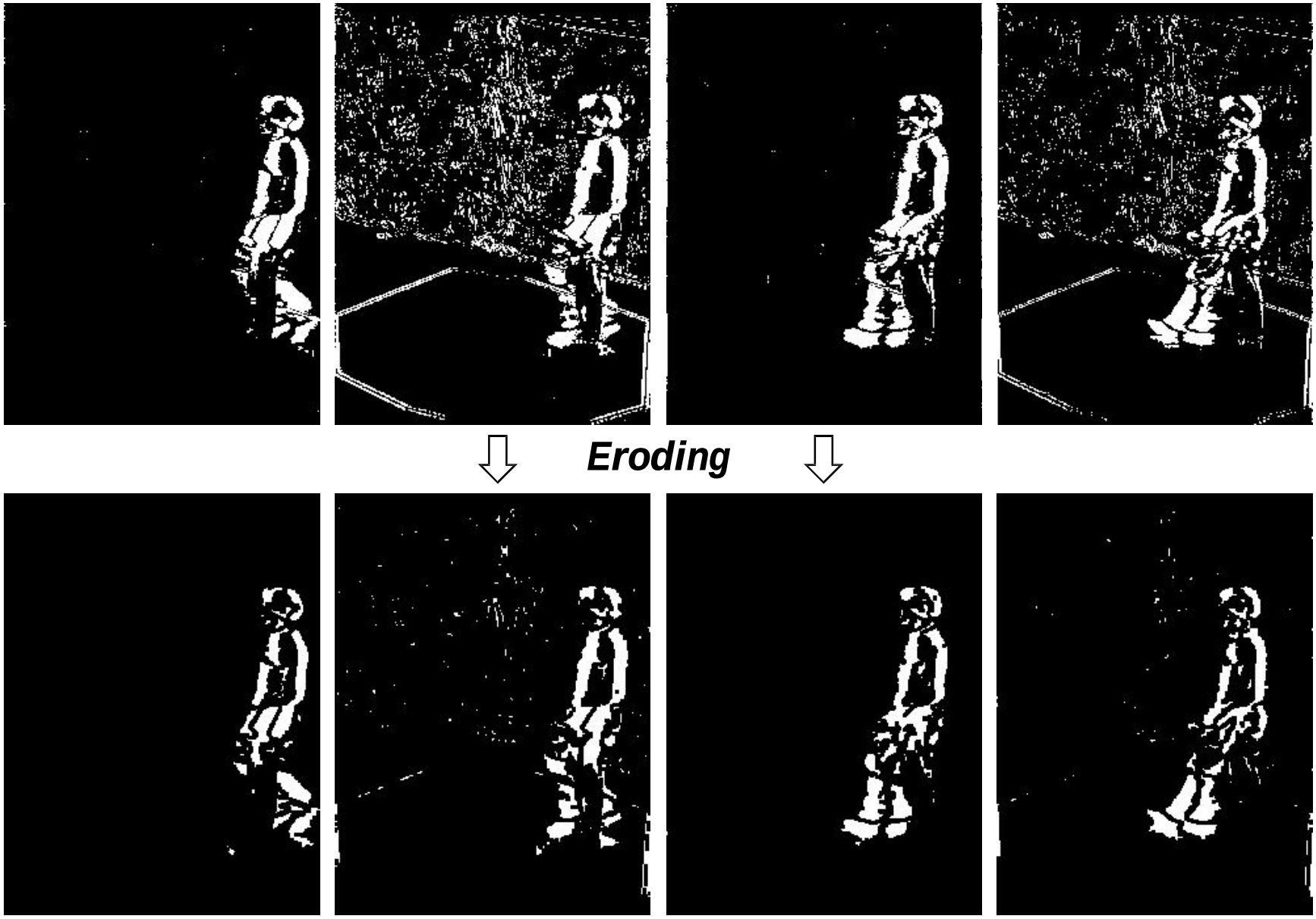
⇩ ***Absolute differences from frame to frame*** ⇩



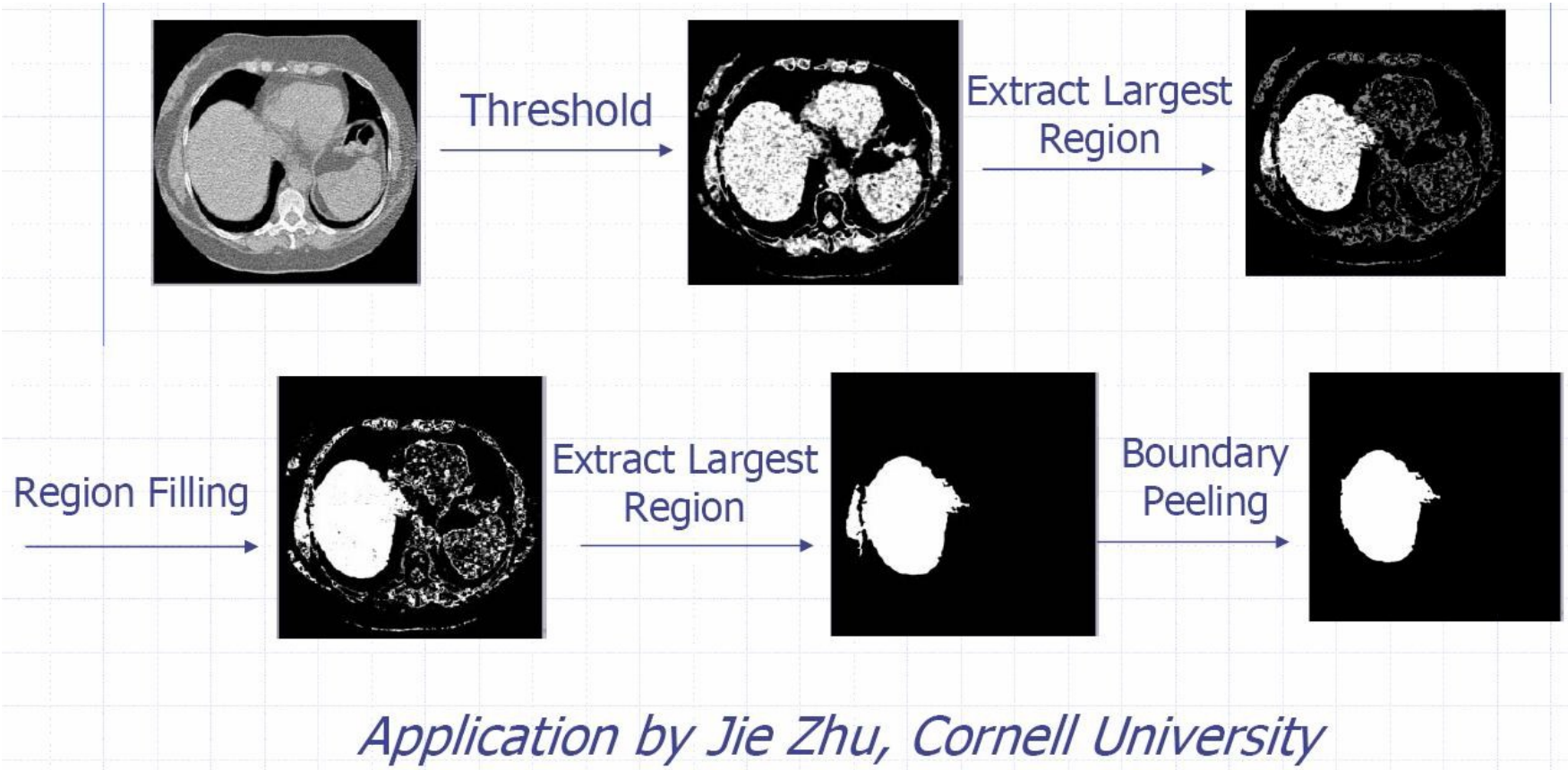


⇩ ***Thresholding*** ⇩




Eroding

Application: Segmentation of a Liver



Application by Jie Zhu, Cornell University

Summary: Binary Image Processing

- ***Pros***
  - ***Fast to compute, easy to store***
  - ***Simple processing techniques***
  - ***Can be very useful for constrained scenarios***
- ***Cons***
  - ***Hard to get "clean" silhouettes***
  - ***Noise is common in realistic scenarios***
  - ***Can be too coarse a representation***
  - ***Cannot deal with 3D changes***

References and Further Reading

- ***More on morphological operators can be found in***
  - ***R. C. Gonzales, R. E. Woods, Digital Image Processing. Prentice Hall, 2001.***
- ***Online tutorial and Java demos available on***
  - ***http://homepages.inf.ed.ac.uk/rbf/HIPR2/***