Image Segmentation

Cristina P Santos
cristina@dei.uminho.pt

28/11/2019

Visão por Computador

Contents

- **Thresholding**
- **Morphology**
- **Image Recognition**
- **Image Segmentation**
  - Overview
  - Identifying Regions
    - KNN clustering
    - Otsu's Algorithm

Based on slides from Mac Namee and Bastian Leide

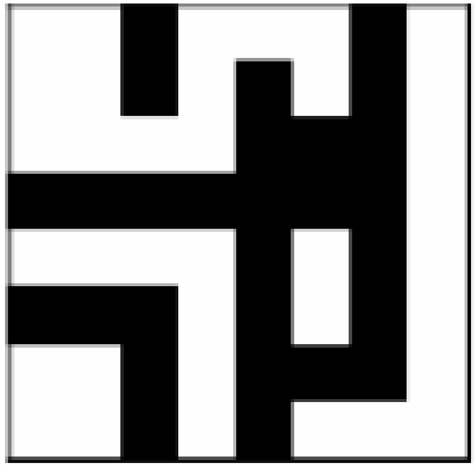
Binary Image Processing

Slide Credit: Bastian Leibe

Binary Images

- Just two pixel values
- Foreground and background
- Regions of interest

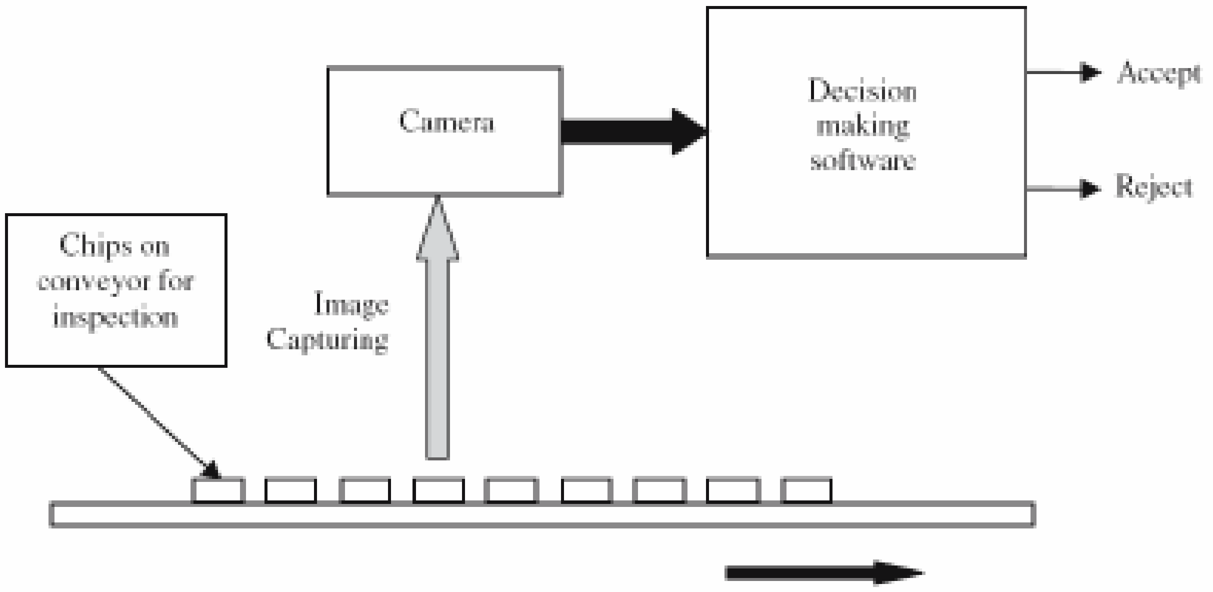
1	1	0	1	1	1	0	1
1	1	0	1	0	1	0	1
1	1	1	1	0	0	0	1
0	0	0	0	0	0	0	1
1	1	1	1	0	1	0	1
0	0	0	1	0	1	0	1
1	1	0	1	0	0	0	1
1	1	0	1	0	1	1	1



Uses: Industrial Inspection

Slide Credit: Bastian Leibe

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



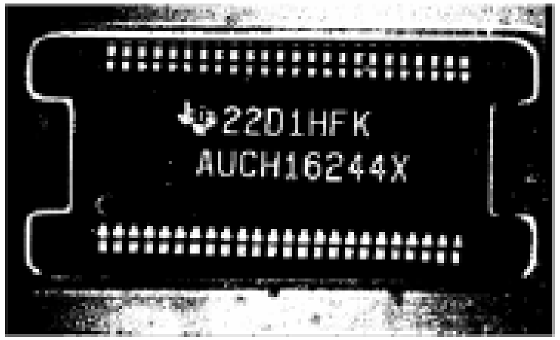
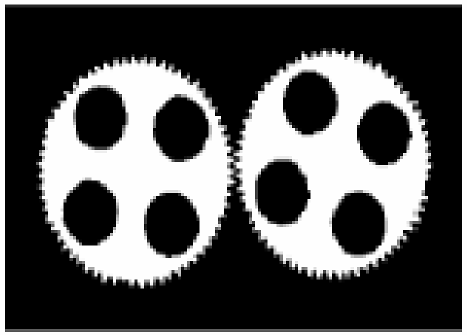


Fig. 7 Binarized image

Fig. 9 Row sum for separating a row

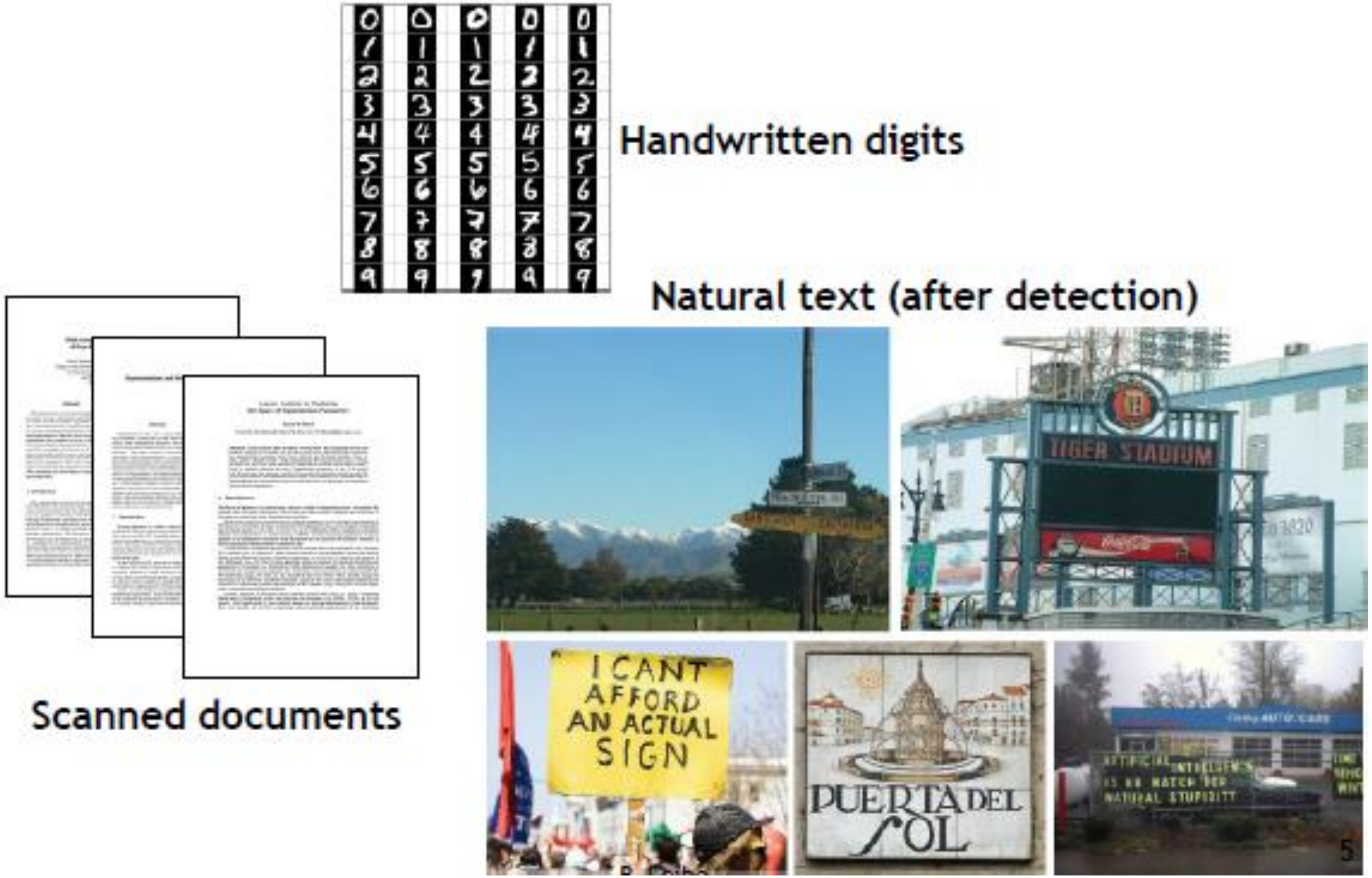


R. Nagarajan et al. “A real time marking inspection scheme for semiconductor industries“, 2006

B. Leibe

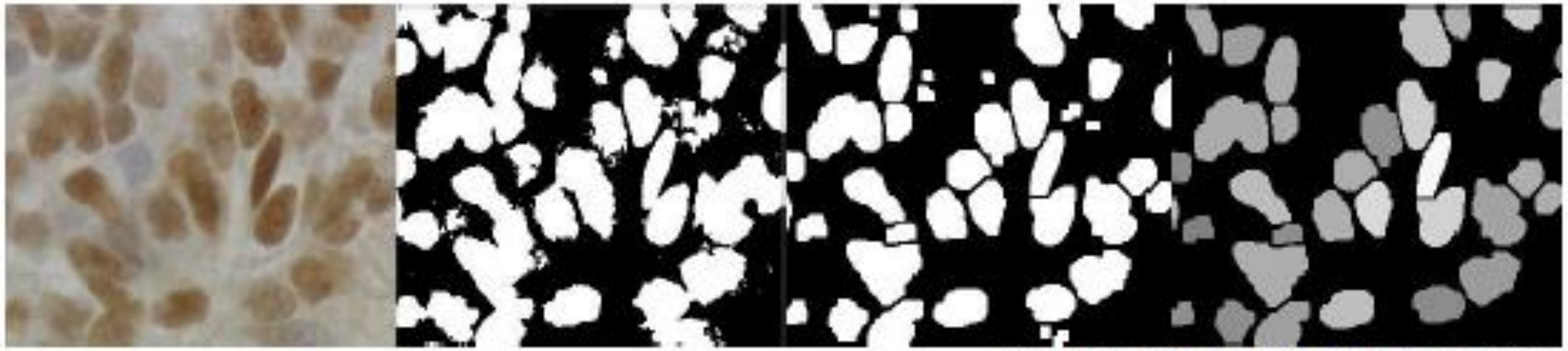
Uses: Document Analysis, text Recognition

Slide Credit: Bastian Leibe

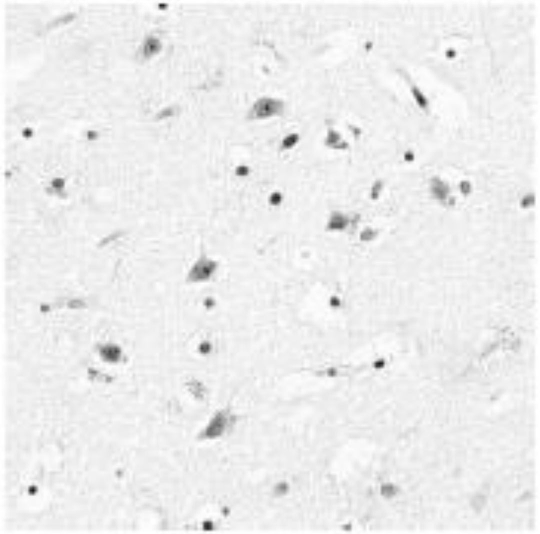


Uses: Medical/Bio Data

Slide Credit: Bastian Leibe

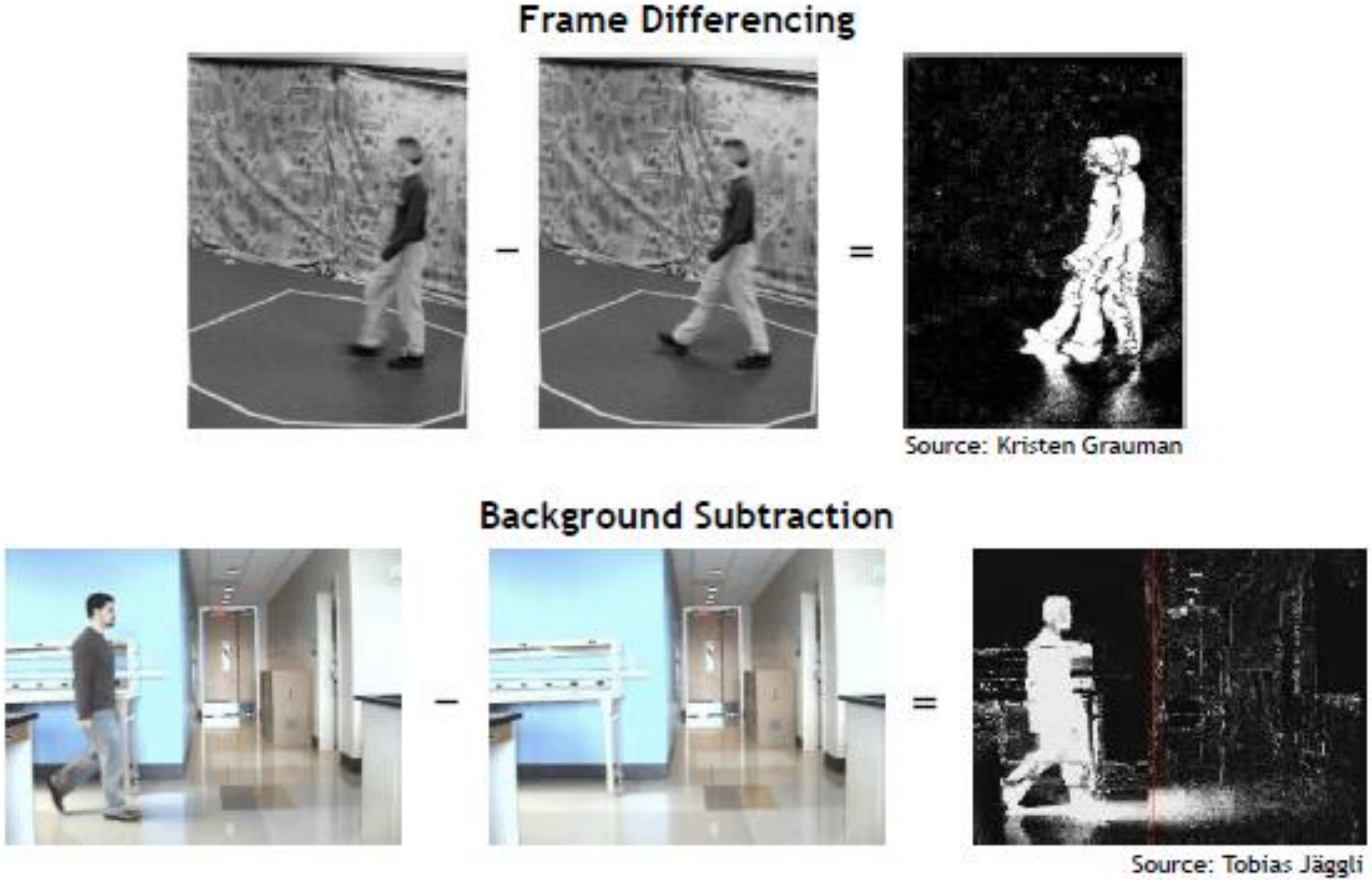


Source: D. Kim et al., Cytometry 35(1), 1999



Uses: Blob Tracking & Motion Analysis

Slide Credit: Bastian Leibe



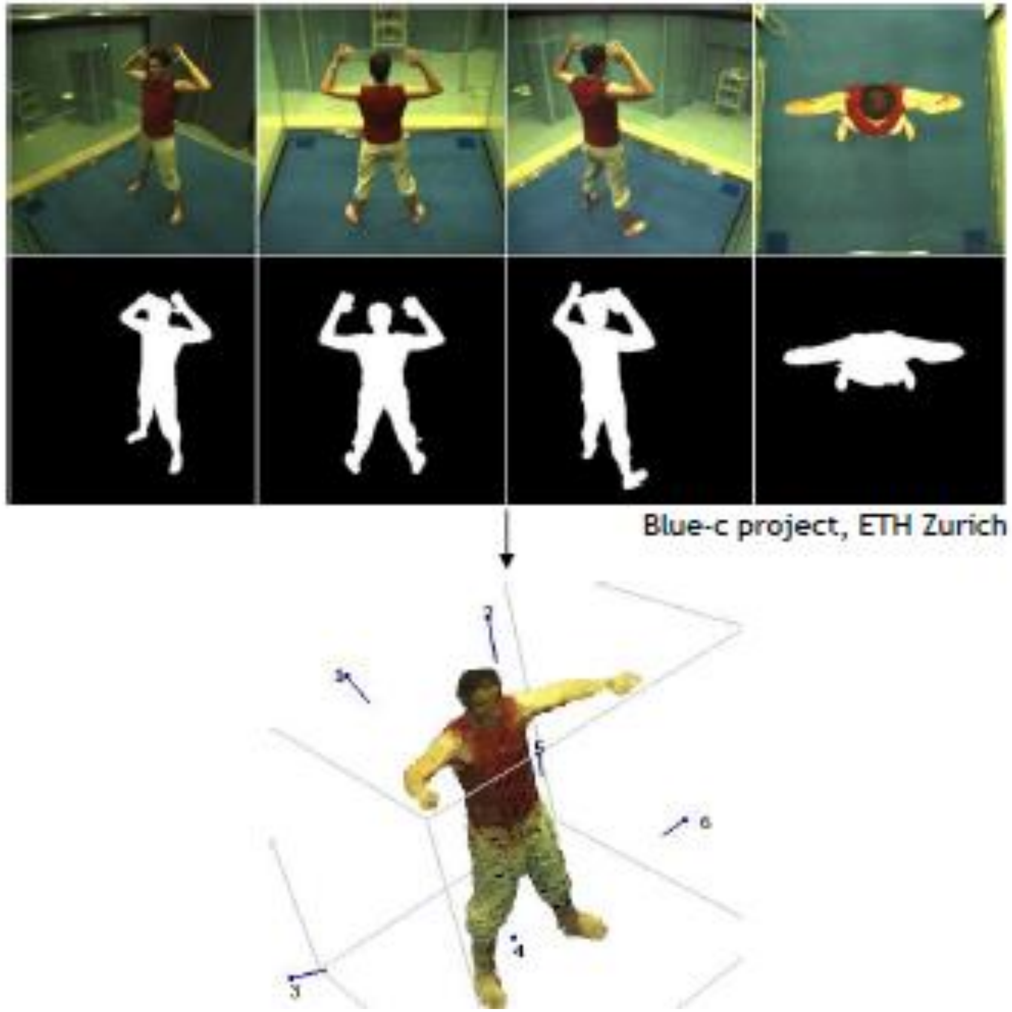
Uses: Shape Analysis, Free-Viewpoint Video

Slide Credit: Bastian Leibe

Silhouette

Medial axis

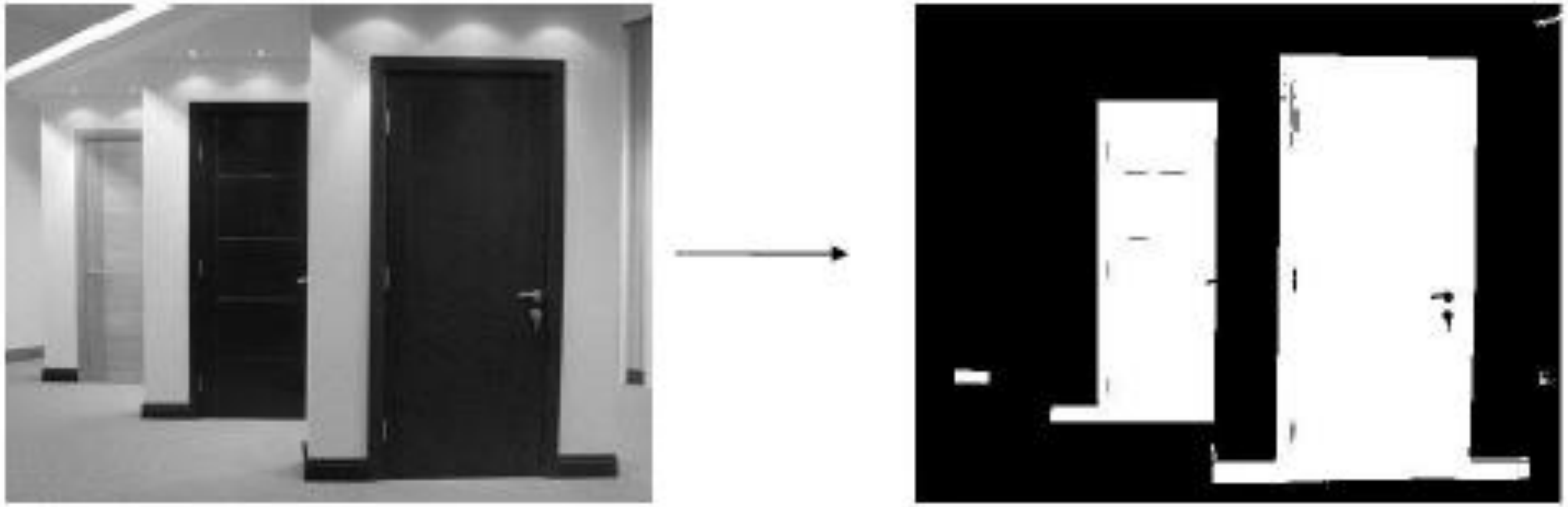
Visual Hull Reconstruction



Uses: Intensity Based Detection

Slide Credit: Bastian Leibe

- Looking for dark pixels...

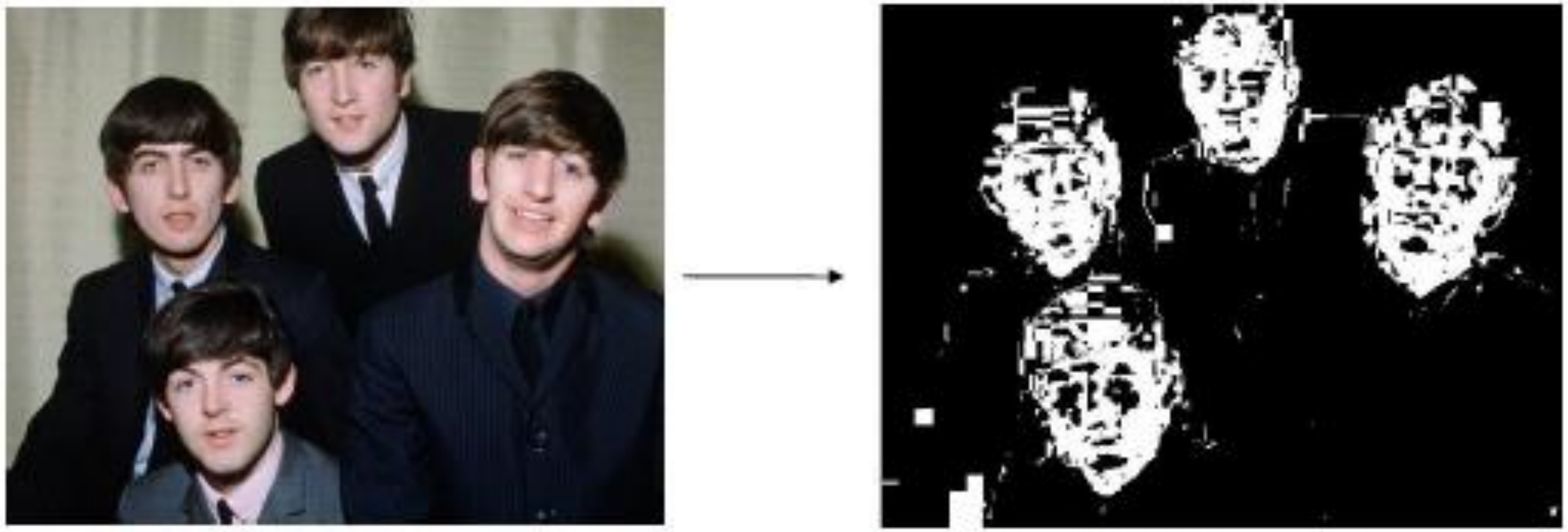


```
fg_pix = find(im < 65);
```

Uses: Color Based Detection

Slide Credit: Bastian Leibe

- Looking for pixels within a certain color range...



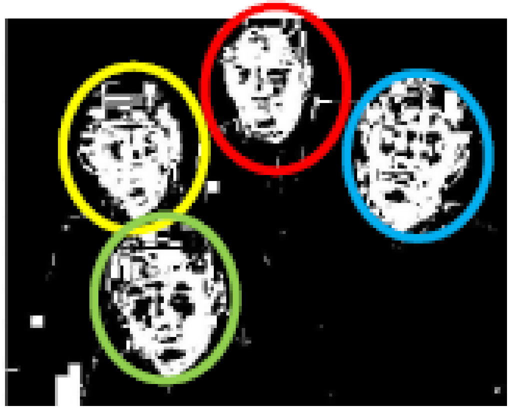
```
fg_pix = find(hue > t1 & hue < t2);
```

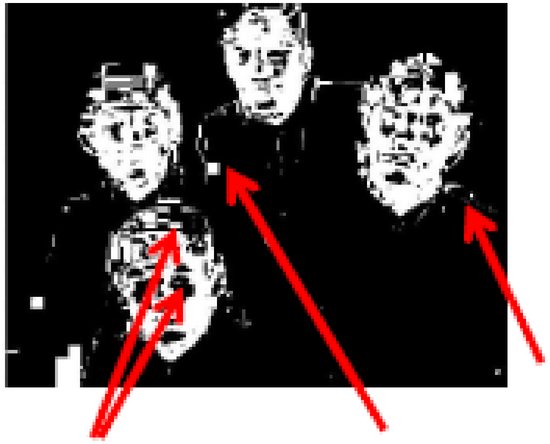
Applications: Vision-based Interfaces

Slide Credit: Bastian Leibe

Issues

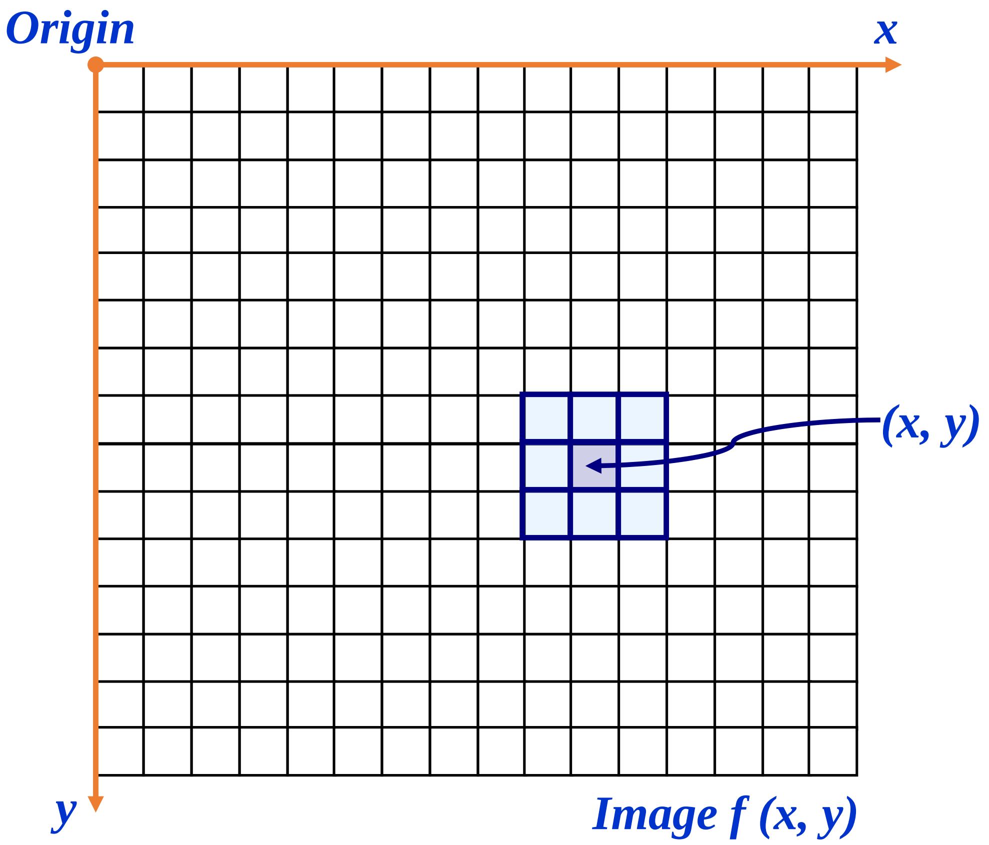
- How to demarcate multiple regions of interest?
  - Count objects
  - Compute further features per object
- What to do with "noisy" binary outputs?
  - Holes
  - Extra small fragments





Basic Spatial Domain Image Enhancement

- Most spatial domain enhancement operations can be reduced to the form
- $g(x, y) = T[f(x, y)]$
- where $f(x, y)$ is the input image, $g(x, y)$ is the processed image and T is some operator defined over some neighbourhood of (x, y)



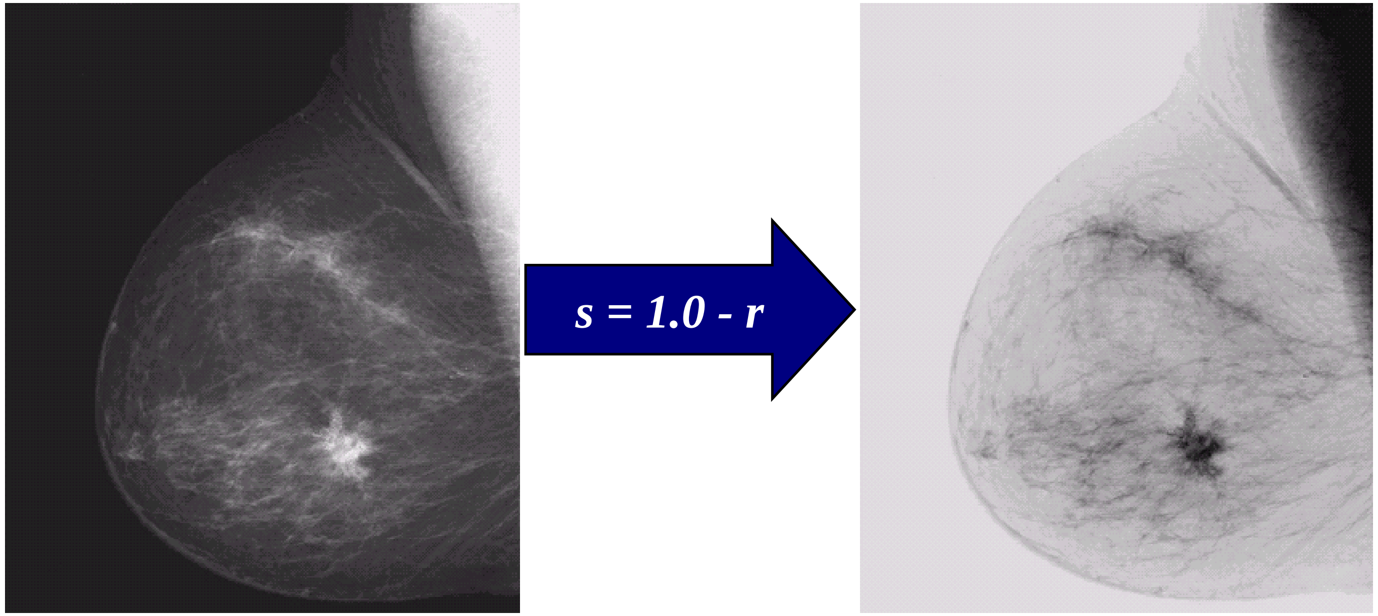
Point Processing

- The simplest spatial domain operations occur when the neighbourhood is simply the pixel itself
- In this case T is referred to as a *grey level transformation function* or a *point processing operation*
- Point processing operations take the form
  - $s = T(r)$
- where s refers to the processed image pixel value and r refers to the original image pixel value

Point Processing Example: Negative Images

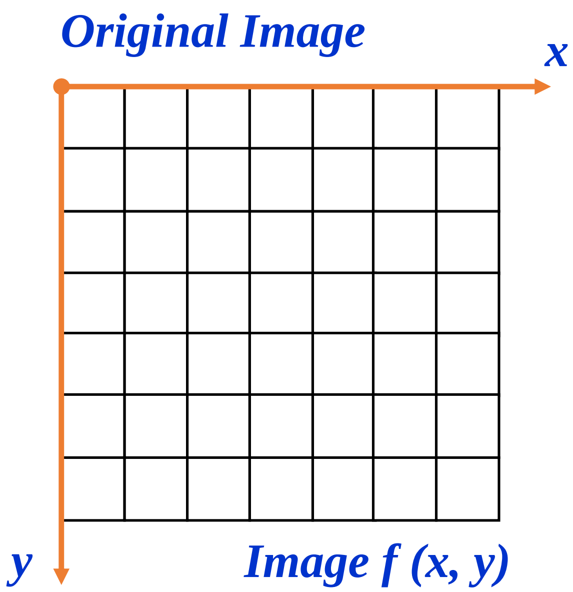
- Negative images are useful for enhancing white or grey detail embedded in dark regions of an image
  - Note how much clearer the tissue is in the negative image of the mammogram below

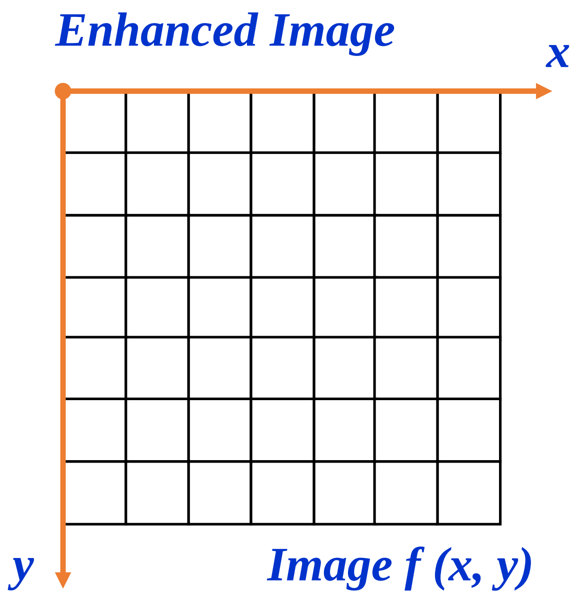
Original Image



Negative Image

Point Processing Example: Negative Images (cont…)



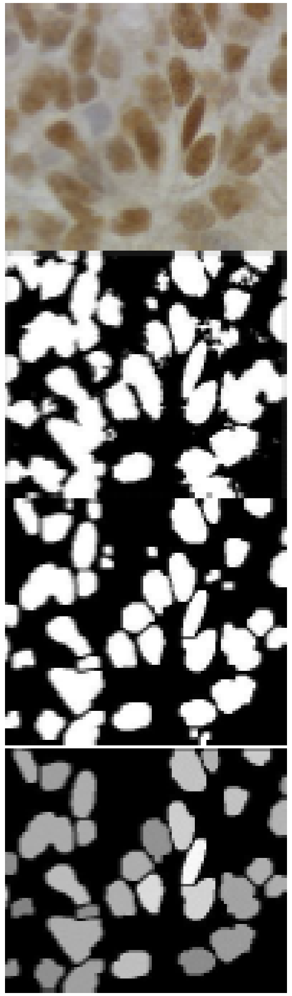


$$s = intensity_{max} - r$$

Outline of this topic Lecture

Slide Credit: Bastian Leibe

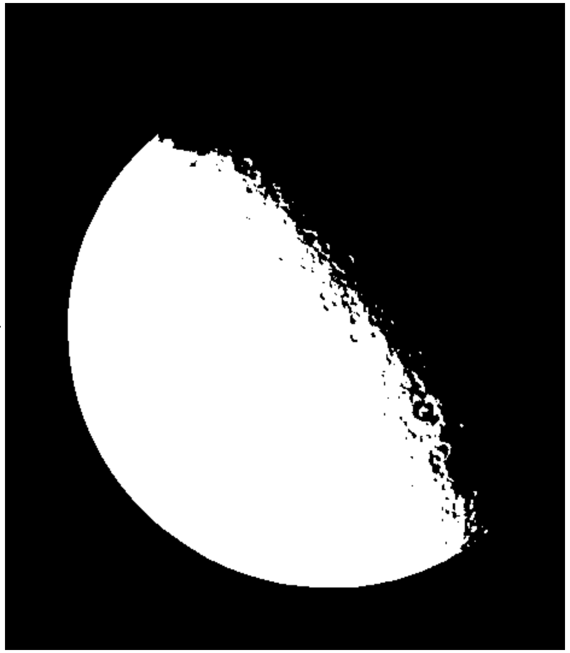
- Convert the image into binary form
  - Thresholding
- Clean up the thresholded image
  - Morphological operators
- Extract individual objects
  - Connected Components Labeling
- Describe the objects
  - Region properties



Point Processing Example: Thresholding

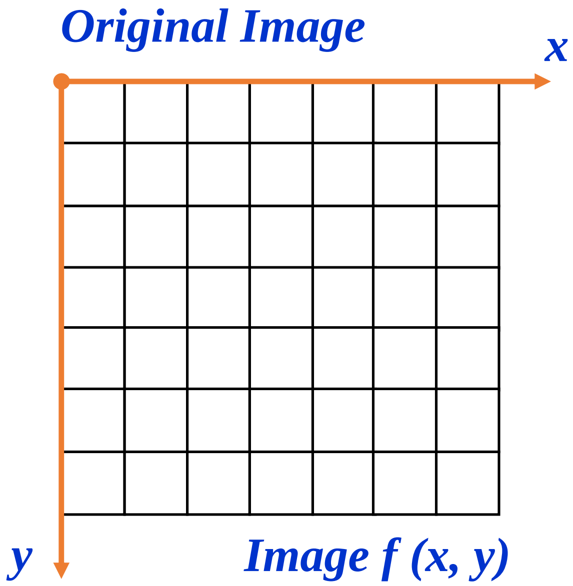
•Thresholding transformations are particularly useful for segmentation in which we want to isolate an object of interest from a background

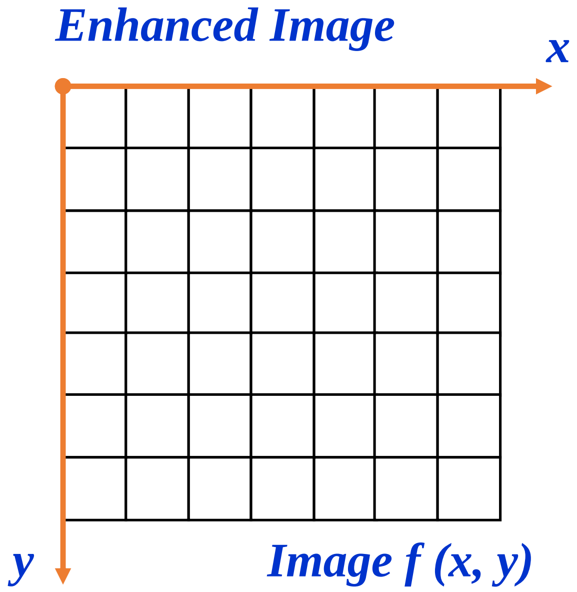
$$s = \begin{cases} 1.0 & r > \text{threshold} \\ 0.0 & r <= \text{threshold} \end{cases}$$



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Point Processing Example: Thresholding (cont…)





$$s = \begin{cases} 1.0 & r > threshold \\ 0.0 & r <= threshold \end{cases}$$

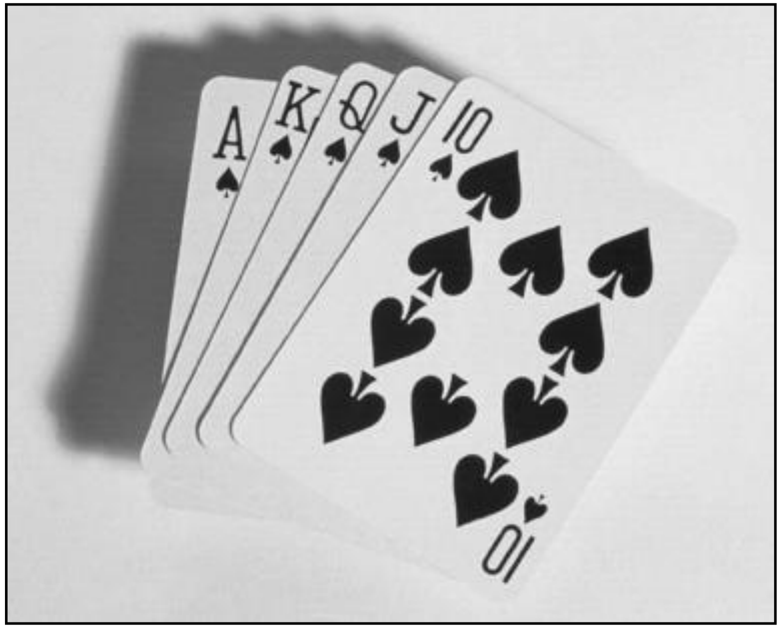
Thresholding

- Thresholding is usually the first step in any segmentation approach
- We have talked about simple single value thresholding already
- Single value thresholding can be given mathematically as follows:

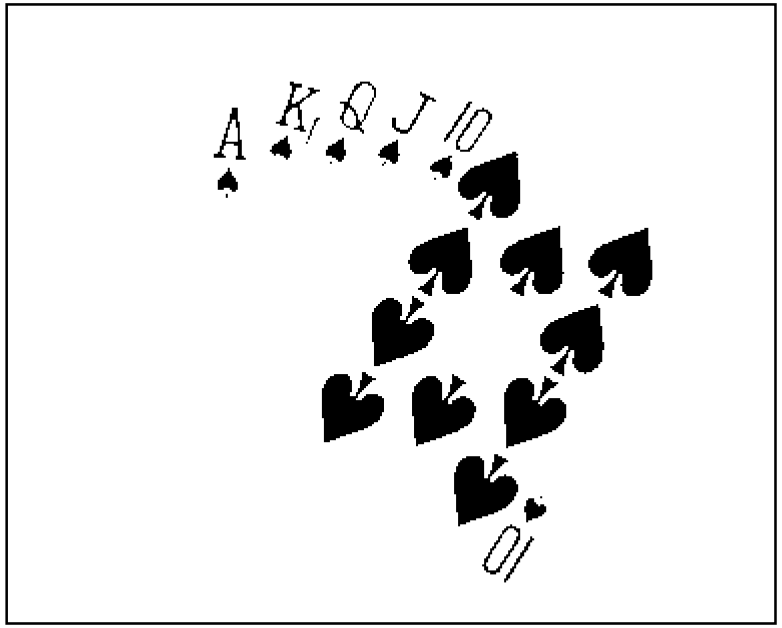
$$g(x,y) = \begin{cases} 1 \, if \;\; f(x,y) > T \\ 0 \, if \;\; f(x,y) \leq T \end{cases}$$

Thresholding Example

- Imagine a poker playing robot that needs to visually interpret the cards in its hand



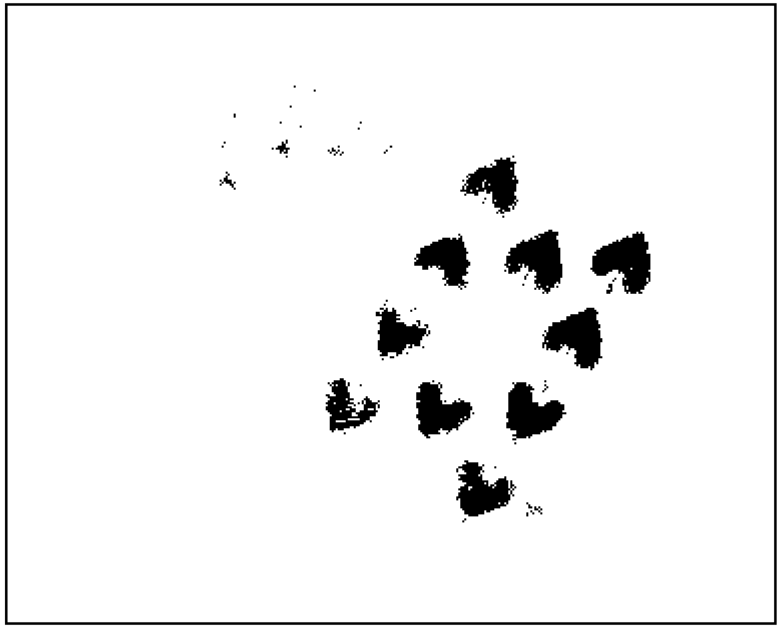
Original Image



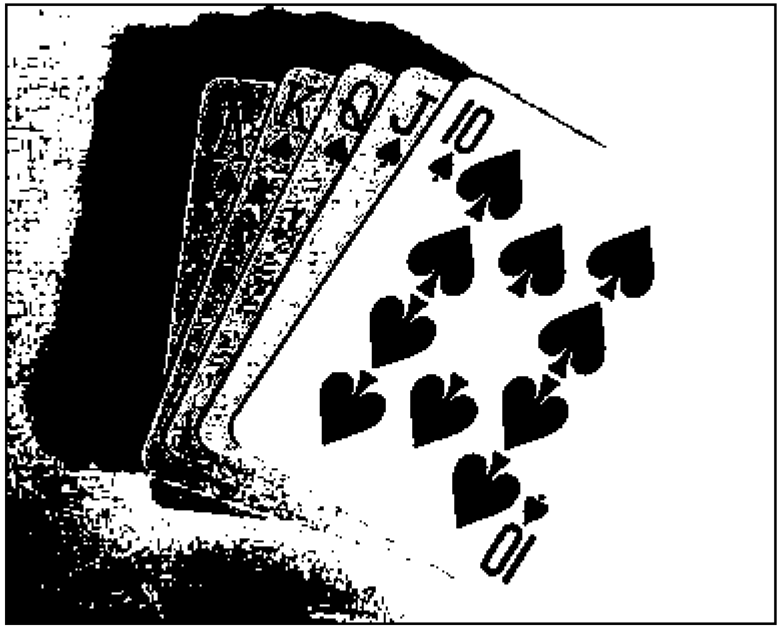
Thresholded Image

But Be Careful

- If you get the threshold wrong the results can be disastrous



Threshold Too Low



Threshold Too High

Thresholding

Slide Credit: Bastian Leibe

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

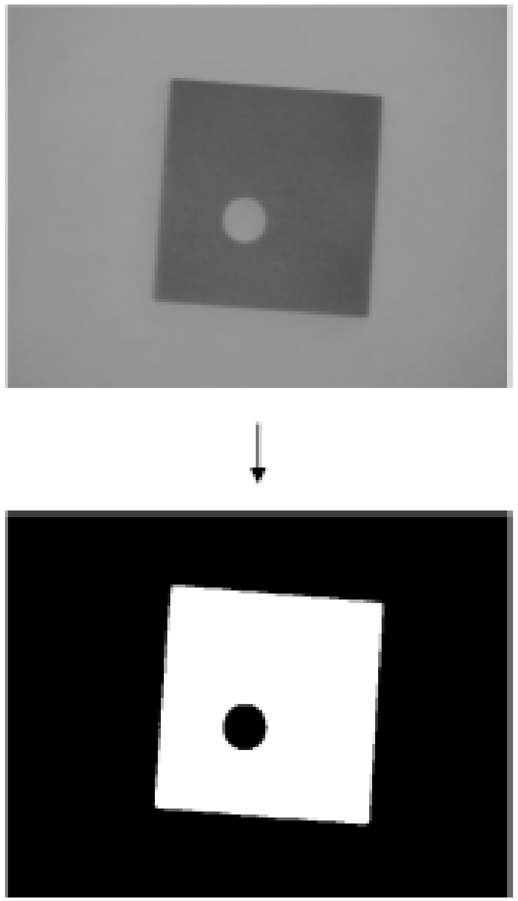
$$F_T[i,j] = \begin{cases} 1, & \text{if } F[i,j] \geq T \\ 0, & \text{otherwise} \end{cases}$$

  - Two-sided

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

  - Set membership

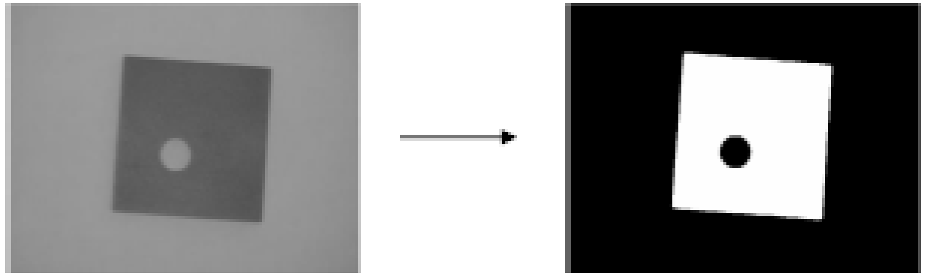
$$F_T[i,j] = \begin{cases} 1, & \text{if } F[i,j] \in Z \\ 0, & \text{otherwise} \end{cases}$$



Selecting Thresholds

Slide Credit: Bastian Leibe

- 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.

See slides 2_1

Basic Global Thresholding

- Based on the histogram of an image
- Partition the image histogram using a single global threshold
- The success of this technique very strongly depends on how well the histogram can be partitioned

Basic Global Thresholding Algorithm

- The basic global threshold, T, is calculated
- as follows:
  1. Select an initial estimate for T (typically the average grey level in the image)
  2. Segment the image using T to produce two groups of pixels: G_1 consisting of pixels with grey levels >T and G_2 consisting pixels with grey levels ≤ T
  3. Compute the average grey levels of pixels in G_1 to give μ_1 and G_2 to give μ_2

Basic Global Thresholding Algorithm

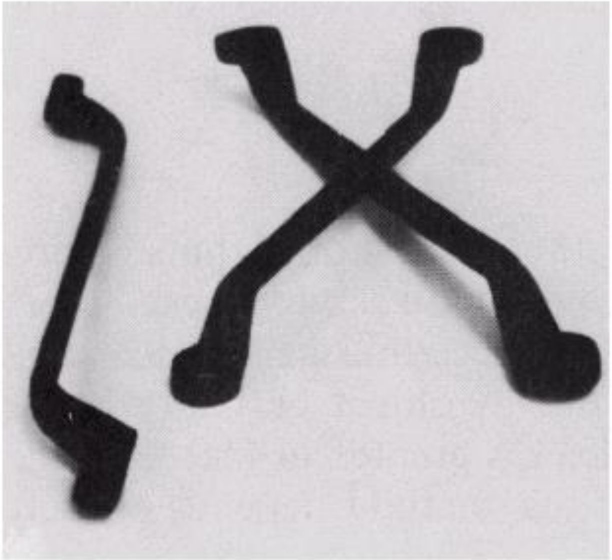
4. Compute a new threshold value:

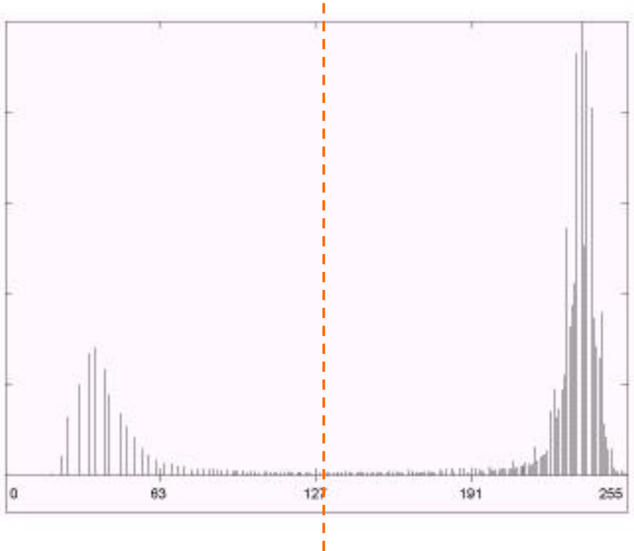
$$T = \frac{\mu_1 + \mu_2}{2}$$

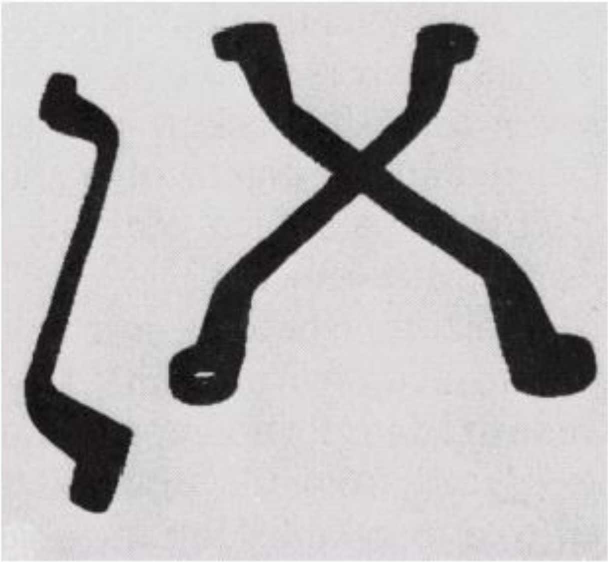
5. Repeat steps 2 – 4 until the difference in T in successive iterations is less than a predefined limit T_∞

- This algorithm works very well for finding thresholds when the histogram is suitable

Thresholding Example 1

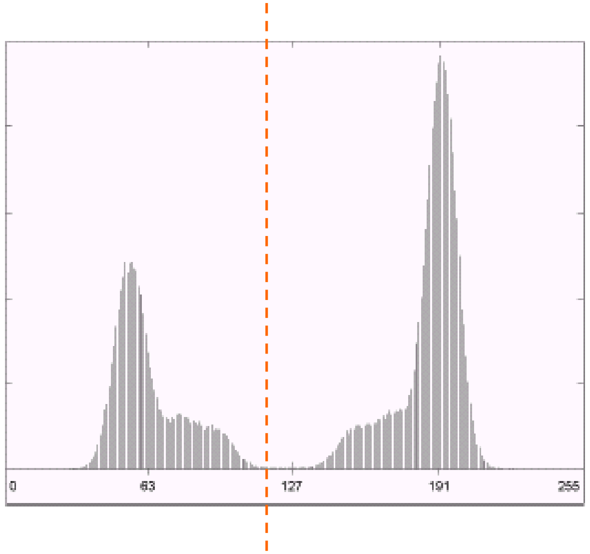






Images taken from Gonzalez & Woods, Digital Image Processing (2002)

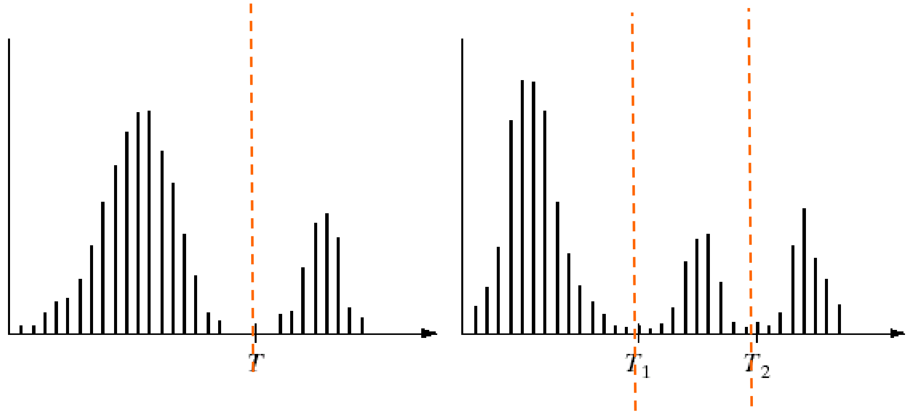
Thresholding Example 2



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Problems With Single Value Thresholding

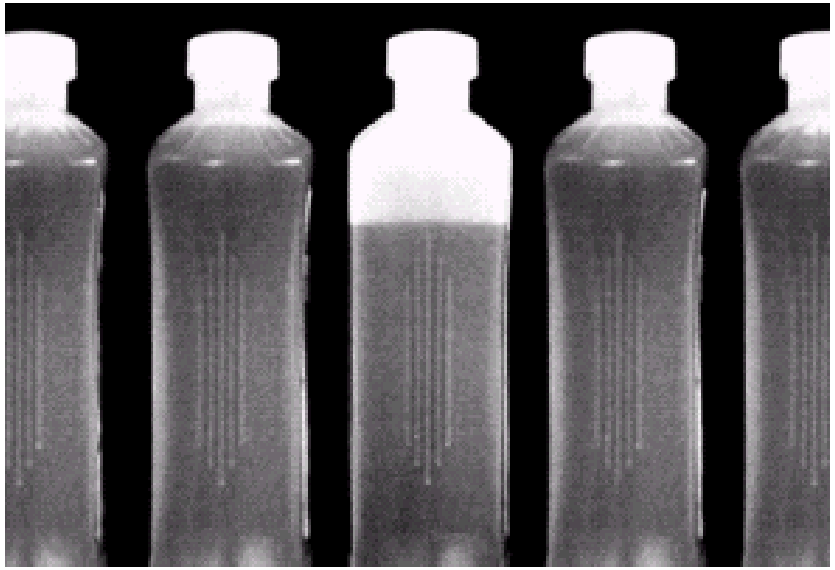
- Single value thresholding only works for bimodal histograms
- Images with other kinds of histograms need more than a single threshold



Problems With Single Value Thresholding (cont…)

Let’s say we want to isolate the contents of the bottles

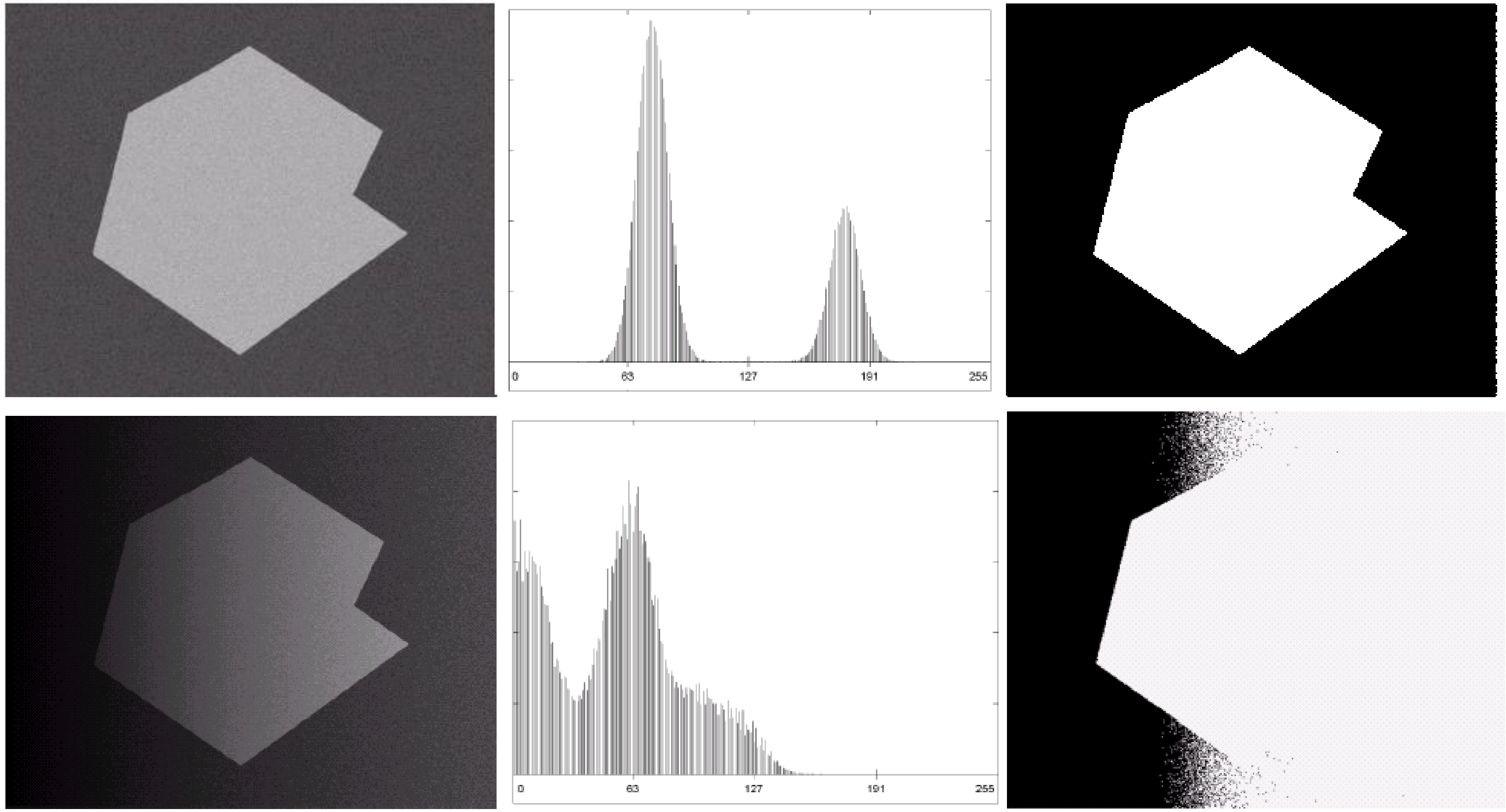
Think about what the histogram for this image would look like



What would happen if we used a single threshold value?

Single Value Thresholding and Illumination

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

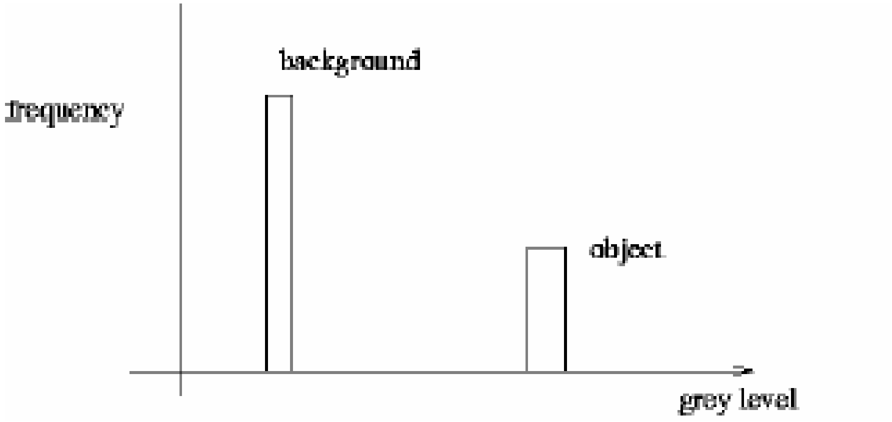


- Uneven illumination can really upset a single valued thresholding scheme

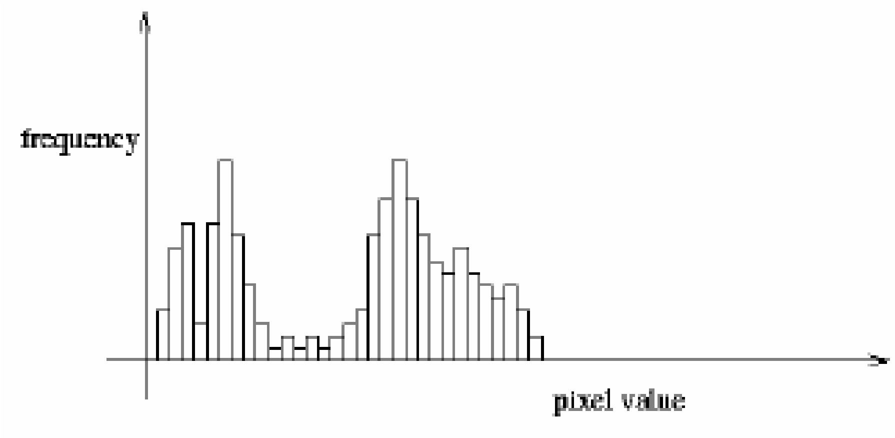
Applications: Vision-based Interfaces

Slide Credit: Bastian Leibe

A Nice Case: Bimodal Intensity Histograms



Ideal histogram, light object on dark background



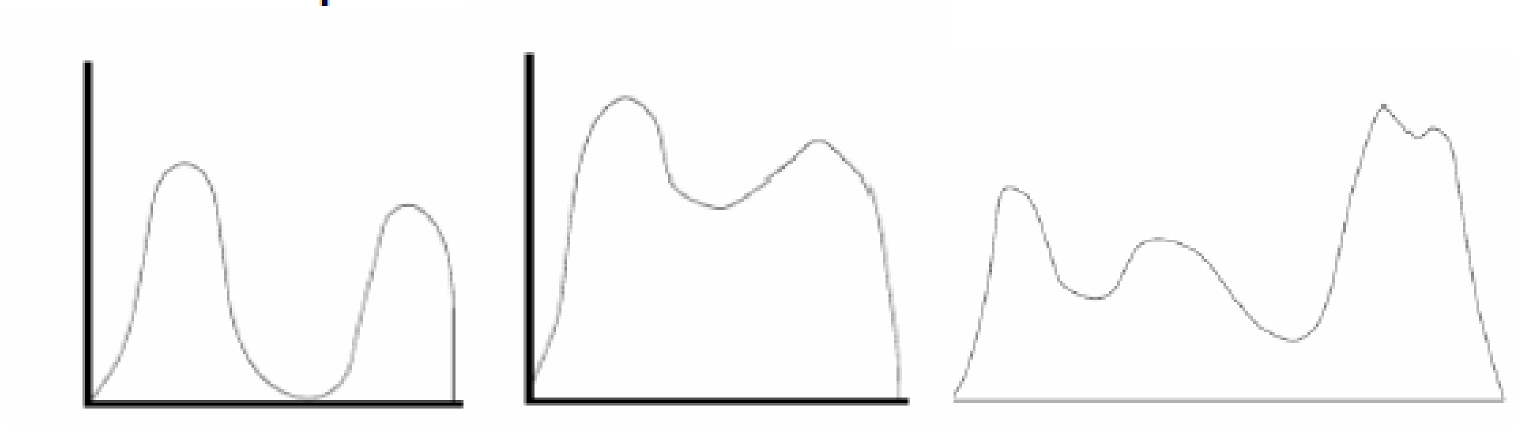
Actual observed histogram with noise

Applications: Vision-based Interfaces

Slide Credit: Bastian Leibe

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

Point operations

- *Binarization – aplicação*

- *seguinte procedimento:*
  - *Determinação do histograma da imagem*
  - *Calculo das médias e variância*
  - *Calculo do Limiar (T)*
  - *Aplicação à imagem da seguinte condição:*

Point operations

- *Binarização*
  - *Existem diversos métodos de determinação automática do limiar*
    - *Russ*
    - *Otsu*

  - *Representando os "pixels" de uma imagem por L níveis de cinzento, g0,…, gi, ... gL-1, e*
  - *ni - o número de ocorrência de cada nível de cinzento, gi*
  - *N - o número total de "pixels"*
  - *pi - a probabilidade de ocorrência de cada nível de cinzento, gi*

  - *pi=ni/N*

Point operations

- *Binarization*
  - *método Russ*
  - *seja* $\mu = \sum_{i=0}^{L-1} gipi$ *a média dos níveis de cinzento da imagem*
    - *e*

$$\sigma^2 = \sum_{i=0}^{L-1} (gi - \mu)^2 \frac{pi}{N}$$

    - *a variância dos níveis de cinzento da imagem*
  - *O limiar apresentado neste método é definido por* $T = \mu + \alpha \sigma^2$

com α representando um valor dependente da probabilidade de ocorrência de objectos relativamente ao fundo da imagem. Este método recorre a uma estimativa do tipo de imagens a binarizar, por forma a quantificar o valor de α.

Point operations

- *Binarization - Método de Otsu*
  - *Supondo que os "pixels" se dividem por duas classes C0 e C1, com o limiar de valor gk-1, isto é*
    - *(g0, ..., gk-1) pertencem a C0*
    - *(gk, ..., gL-1) pertencem a C1*
  - *a probabilidade de ocorrência das classes é dada por*

$$p(C_0) = \sum_{i=0}^{K-1} p_i \qquad p(C_1) = \sum_{i=K}^{L-1} p_i = 1 - p(C_0)$$

  - *As médias de C0 e C1 e Total são dadas por*

$$\mu_0 = \frac{\mu(K)}{w(K)} \qquad \mu_1 = \frac{\mu_T - \mu(K)}{1 - w(K)} \qquad \mu_T = \sum_{i=0}^{L-1} g_i p_i$$

  - w(K) e μ(K) são os momentos cumulativos de ordens zero e um do histograma até ao nível gk

$$w(K) = \sum_{i=0}^{K-1} p_i \qquad \mu(K) = \sum_{i=0}^{K-1} g_i p_i$$

Point operations

- *Binarization - Método de Otsu*
  - *As variâncias por classe são expressas por:*

$$\sigma_0^2 = \sum_{i=0}^{K-1} \frac{(g_i - \mu_0)^2 p_i}{w(K)} \qquad \sigma_1^2 = \sum_{i=K}^{L-1} \frac{(g_i - \mu_1)^2 p_i}{1 - w(K)}$$

  - *O limiar a utilizar. ak; é tal que maximize uma das medidas λ, k e η, definidas como*

$$\lambda = \frac{\sigma_0^2}{\sigma_w^2}$$

definindo a variância intra-classe

$$\sigma_w^2 = w(K)\sigma_0^2 + (1 - w(K))\sigma_1^2$$

$$\eta = \frac{\sigma_\beta^2}{\sigma_T^2}$$

definindo a variância inter-classe

$$\sigma_\beta^2 = w(K)(\mu_0 - \mu_T)^2 + (1 - w(K))(\mu_1 - \mu_T)^2$$

$$k = \frac{\sigma_T^2}{\sigma_w^2}$$

variância global

$$\sigma_T^2 = \sum_{i=0}^{L-1} ((g_i - \mu_T)^2 p_i)$$

Threshold

Slide Credit: Bastian Leibe

Global Binarization [Otsu'79]

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

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

where

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

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

$$\begin{aligned} \sigma^2_{between}(T) &= \sigma^2 - \sigma^2_{within}(T) \\ &= n_1(T)n_2(T)\left[\mu_1(T) - \mu_2(T)\right]^2 \end{aligned}$$

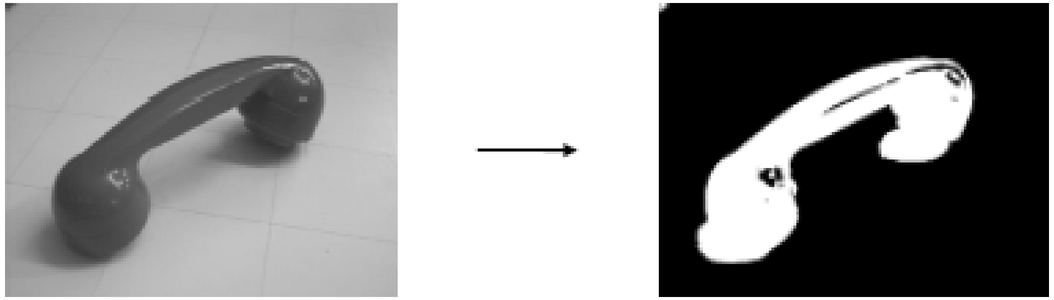
Threshold

Slide Credit: Bastian Leibe

Algorithm

1. Precompute a cumulative grayvalue histogram h.
2. For each potential threshold T
   a) Separate the pixels into two clusters according to T
   b) Look up n_1, n_2 in h and compute both cluster means
   c) Compute $\sigma^2_{between}(T) \;=\; n_1(T)n_2(T)\left[\mu_1(T) - \mu_2(T)\right]^2$
3. Choose

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

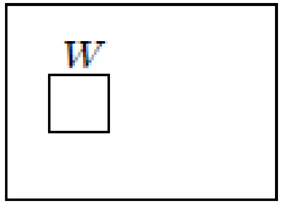


Applications: Vision-based Interfaces

Slide Credit: Bastian Leibe

Local Binarization [Niblack'86]

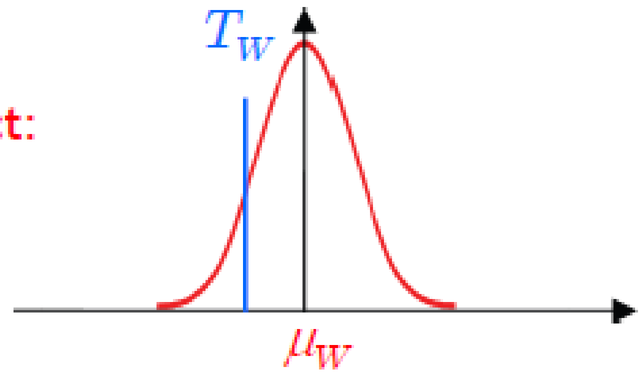
- Estimate a local threshold within a small neighborhood window W



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

where $k \in [-1,0]$ is a user-defined parameter.

Effect:

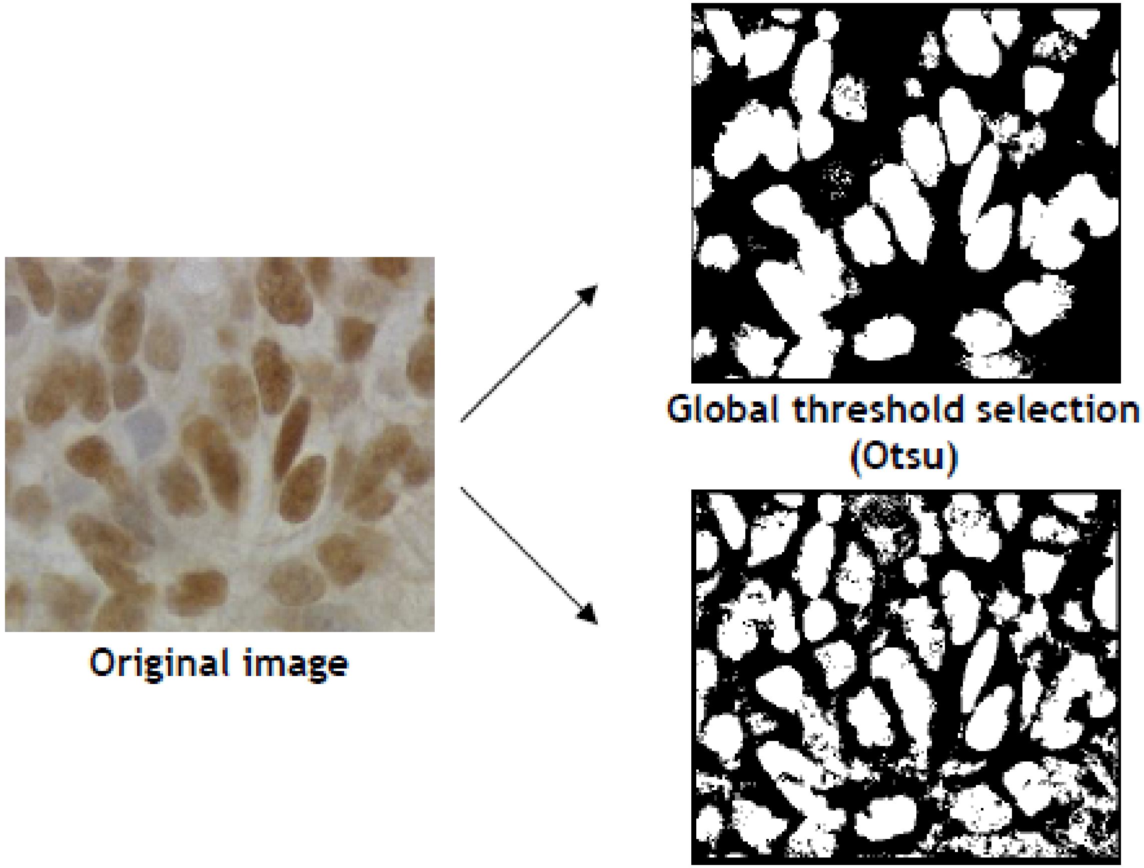


What is the hidden assumption here?

Applications: Vision-based Interfaces

Slide Credit: Bastian Leibe

Effects



Original image

Global threshold selection (Otsu)

Local threshold selection (Niblack)

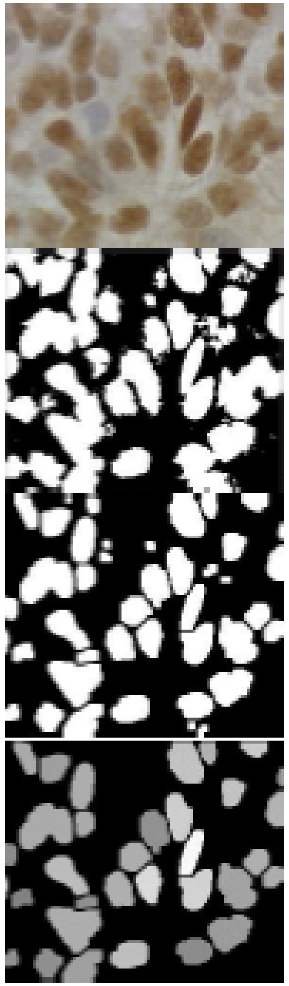
Summary

- In this lecture we have begun looking at segmentation, and in particular thresholding
- We saw the basic global thresholding algorithm and its shortcomings
- We also saw a simple way to overcome some of these limitations using adaptive thresholding

Outline of this lecture

Slide Credit: Bastian Leibe

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



Contents

Once segmentation is complete, morphological operations can be used to remove imperfections in the segmented image and provide information on the form and structure of the image

In this lecture we will consider

- What is morphology?
- Simple morphological operations
- Compound operations
- Morphological algorithms

1, 0, Black, White?

Throughout all of the following slides whether 0 and 1 refer to white or black is a little interchangeable

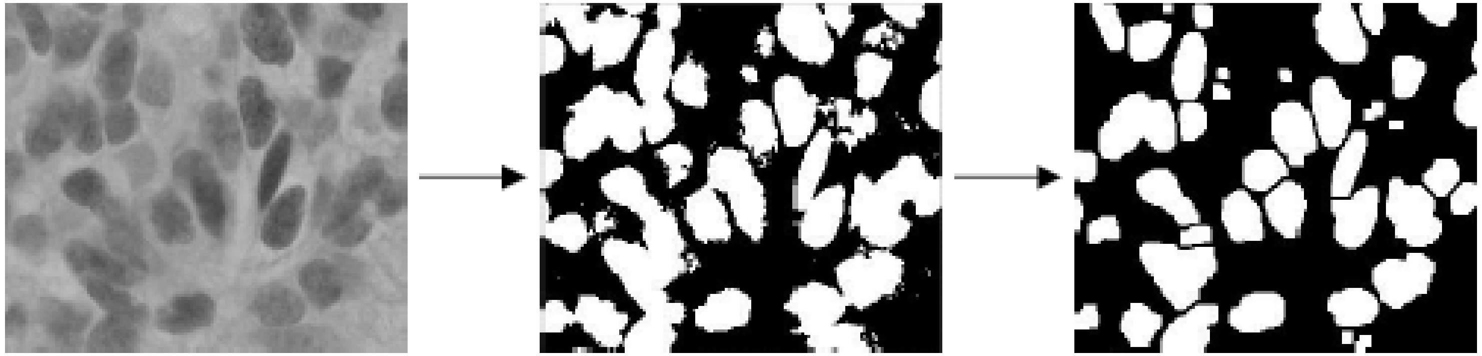
All of the discussion that follows assumes segmentation has already taken place and that images are made up of 0s for background pixels and 1s for object pixels

After this it doesn’t matter if 0 is black, white, yellow, green…….

Cleaning the Binarized Results

Slide Credit: Bastian Leibe

- Results of thresholding often still contain noise



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

⇒ Morphological Operators

What Is Morphology?

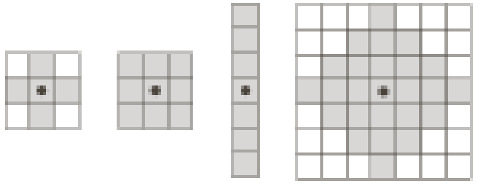
Morphological image processing (or *morphology*) describes a range of image processing techniques that deal with the shape (or morphology) of features in an image

Morphological operations are typically applied to remove imperfections introduced during segmentation, and so typically operate on bi-level images

Morphological Operators

Slide Credit: Bastian Leibe

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



Matlab:
`>> help strel`

- Two basic operations
  - Dilation (Matlab: `imdilate`)
  - Erosion (Matlab: `imerode`)

- Several important combinations
  - Opening (Matlab: `imopen`)
  - Closing (Matlab: `imclose`)
  - Boundary extraction

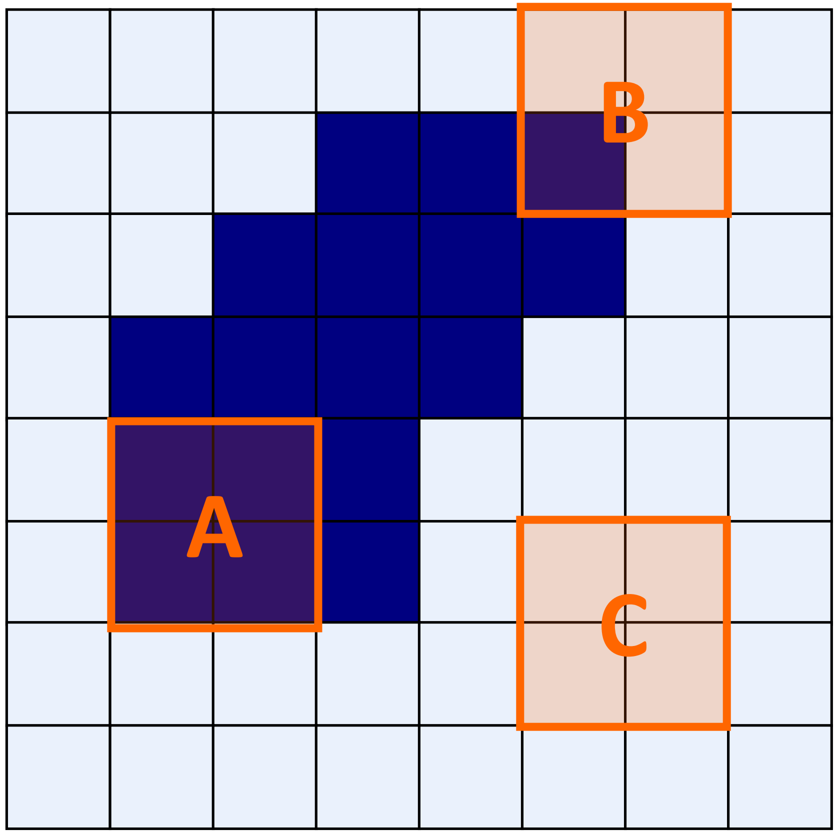
Quick Example

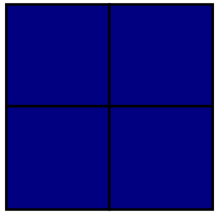
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image after segmentation

Image after segmentation and morphological processing

Structuring Elements, Hits & Fits





Structuring Element

Fit: All *on pixels* in the structuring element cover *on pixels* in the image

Hit: Any *on pixel* in the structuring element covers an *on pixel* in the image

All morphological processing operations are based on these simple ideas

Structuring Elements

Structuring elements can be any size and make any shape

However, for simplicity we will use rectangular structuring elements with their origin at the middle pixel

1	1	1
1	**1**	1
1	1	1

0	1	0
1	**1**	1
0	1	0

0	0	1	0	0
0	1	1	1	0
1	1	**1**	1	1
0	1	1	1	0
0	0	1	0	0

Fitting & Hitting

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0
0	0	1	1 B	1	1	1	0	0 C	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1 A	1	1	1	0
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0

1	1	1
1	1	1
1	1	1

Structuring Element 1

0	1	0
1	1	1
0	1	0

Structuring Element 2

Fundamental Operations

Fundamentally morphological image processing is very like spatial filtering

The structuring element is moved across every pixel in the original image to give a pixel in a new processed image

The value of this new pixel depends on the operation performed

There are two basic morphological operations: **erosion** and **dilation**

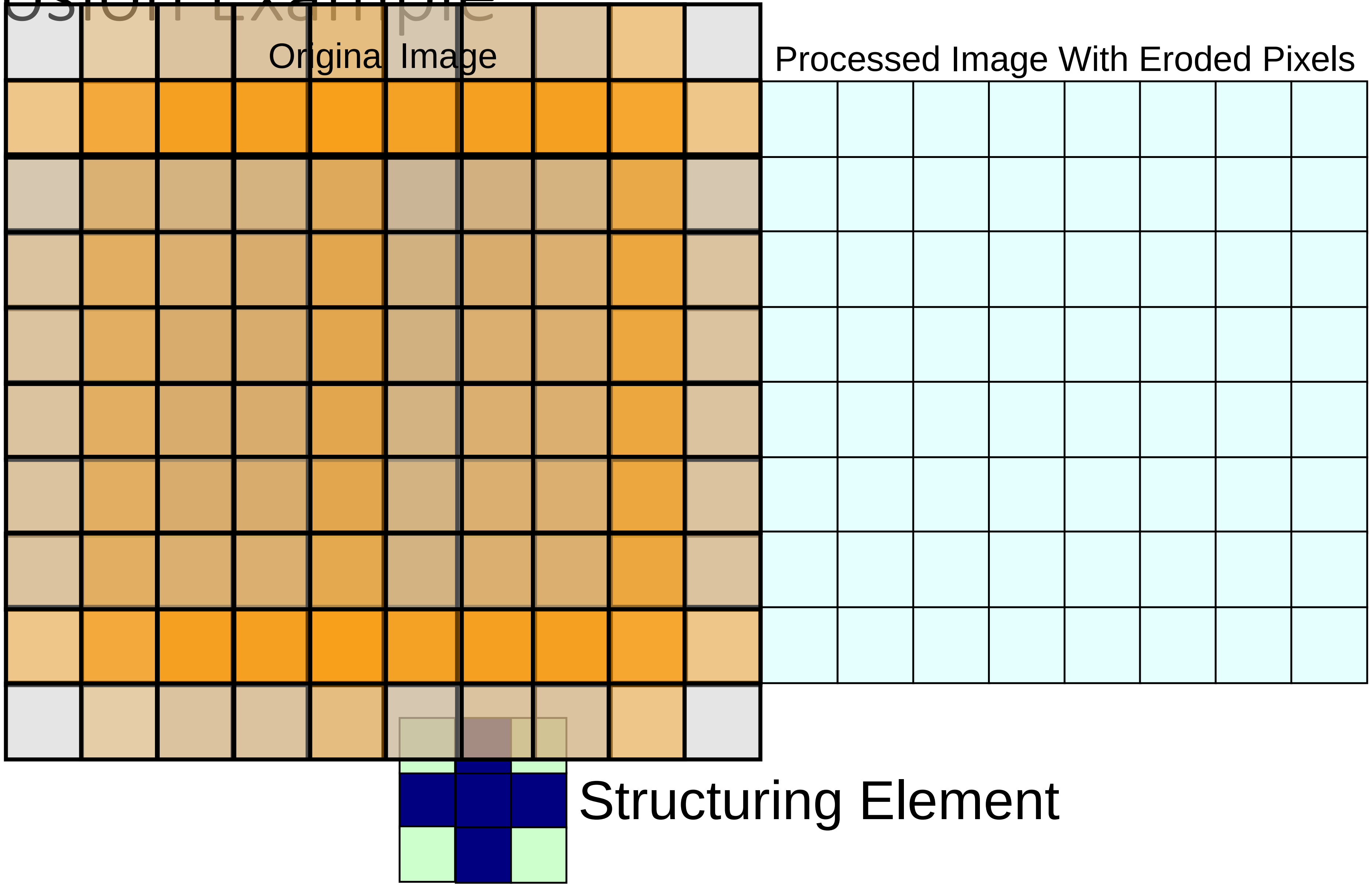
Erosion

Erosion of image f by structuring element s is given by $f \ominus s$

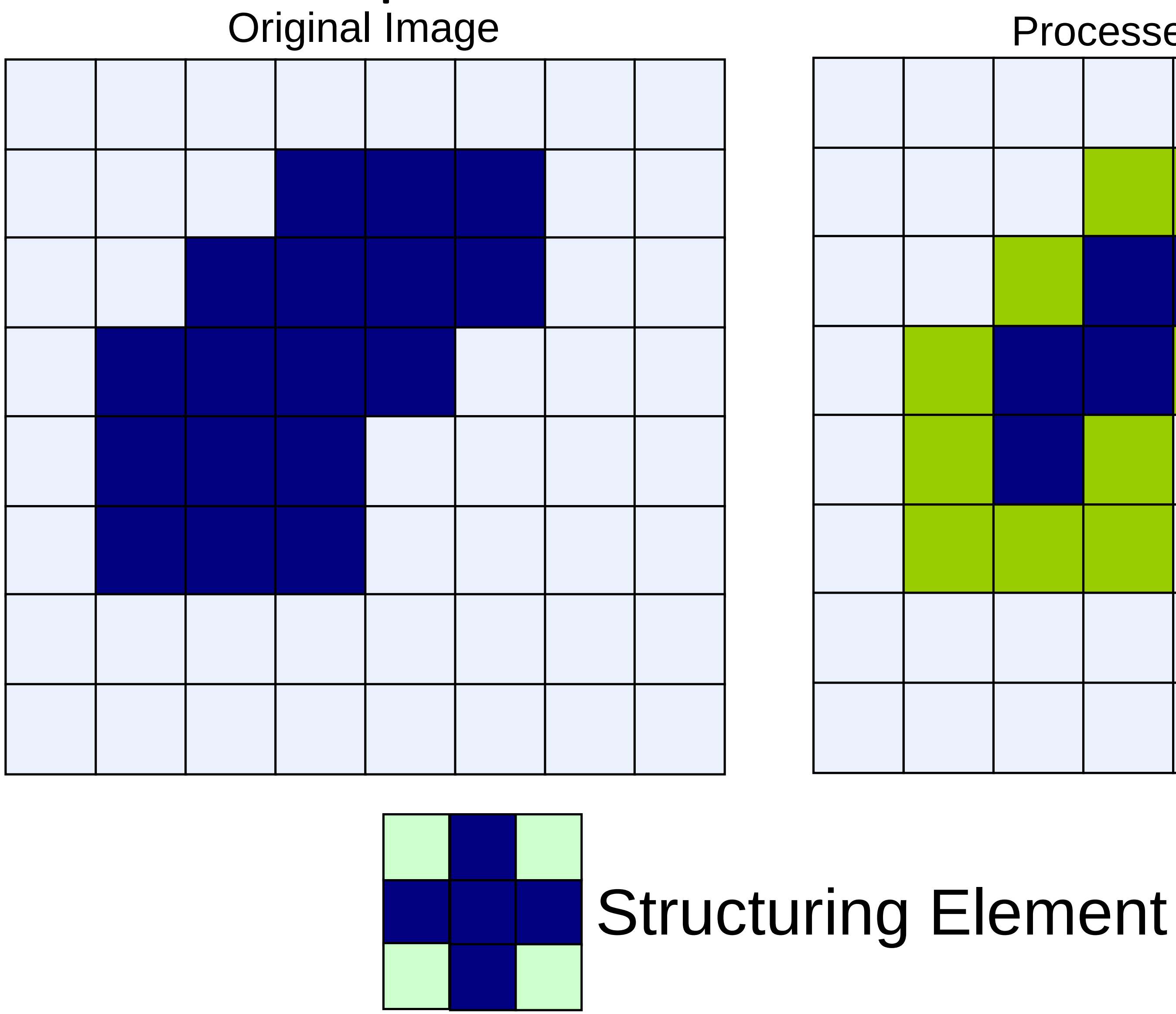
The structuring element s is positioned with its origin at (x, y) and the new pixel value is determined using the rule:

$$g(x,y) = \begin{cases} 1 \text{ if } s \text{ fits } f \\ 0 \text{ otherwise} \end{cases}$$

Erosion Example

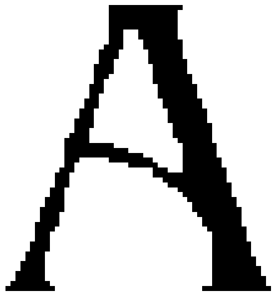


Erosion Example

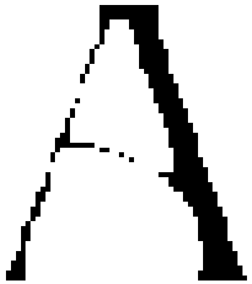


Erosion Example 1

Original image



Erosion by 3*3 square structuring element

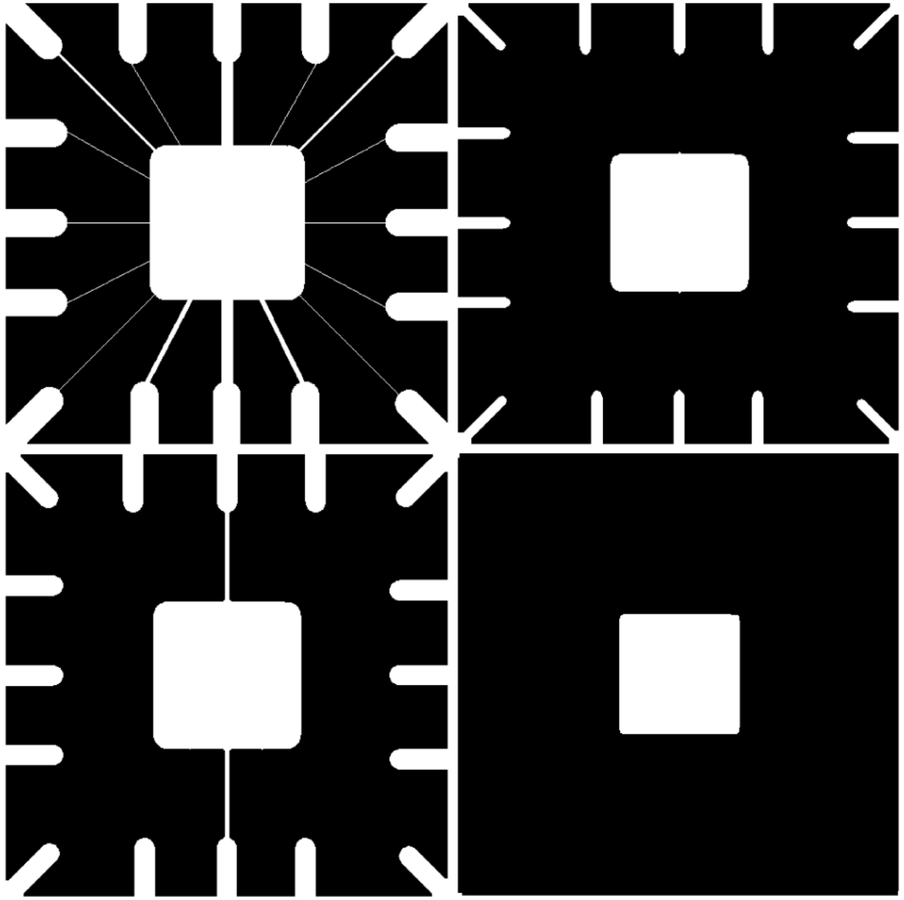


Erosion by 5*5 square structuring element

Watch out: In these examples a 1 refers to a black pixel!

Erosion Example 2

Images taken from Gonzalez & Woods, Digital Image Processing (2002)



Original image

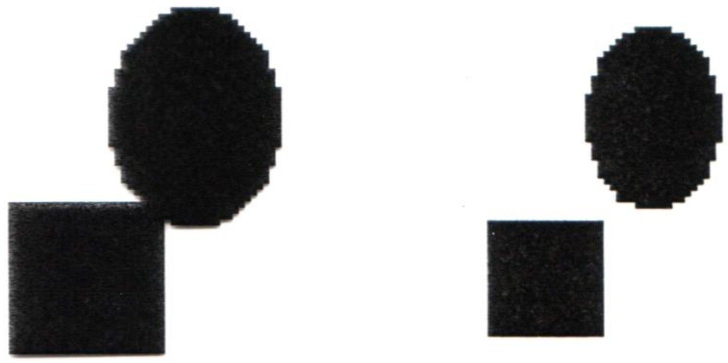
After erosion with a disc of radius 10

After erosion with a disc of radius 5

After erosion with a disc of radius 20

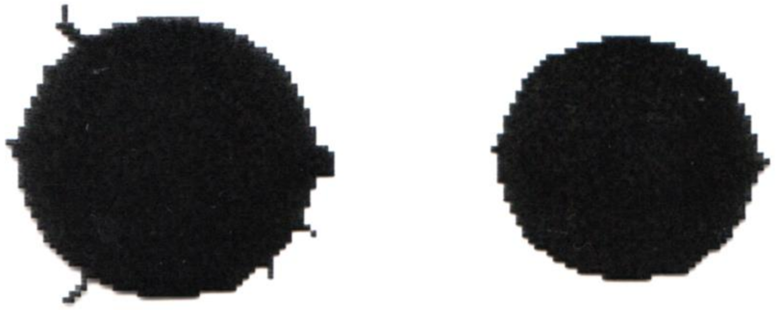
What Is Erosion For?

Erosion can split apart joined objects



Erosion can split apart

Erosion can strip away extrusions

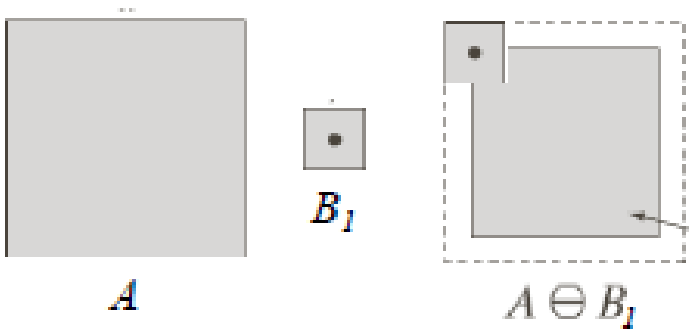


Watch out: E

Erosion

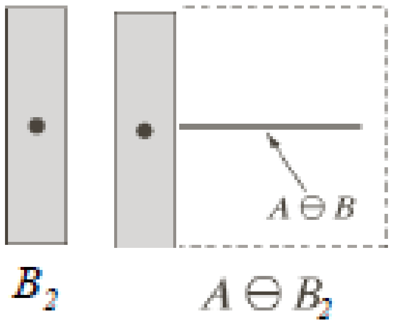
Slide Credit: Bastian Leibe

- **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



Matlab

IPT function strel constructs structuring elements with a variety of shapes and sizes. Its basic syntax is

```
se = strel(shape, parameters)
```

Dilation

$\oplus$

Dilation

$\oplus$

Dilation

$\oplus$

Dilation

Dilation of image f by structuring element s is given by $f \oplus s$

The structuring element s is positioned with its origin at (x, y) and the new pixel value is determined using the rule:

$$g(x,y) = \begin{cases} 1 \text{ if } s \text{ hits } f \\ 0 \text{ otherwise} \end{cases}$$

Dilation

Dilation of image f by structuring element s is given by $f \oplus s$

The structuring element s is positioned with its origin at (x, y) and the new pixel value is determined using the rule:

$$g(x, y) = \begin{cases} 1 \text{ if } s \text{ hits } f \\ 0 \text{ otherwise} \end{cases}$$

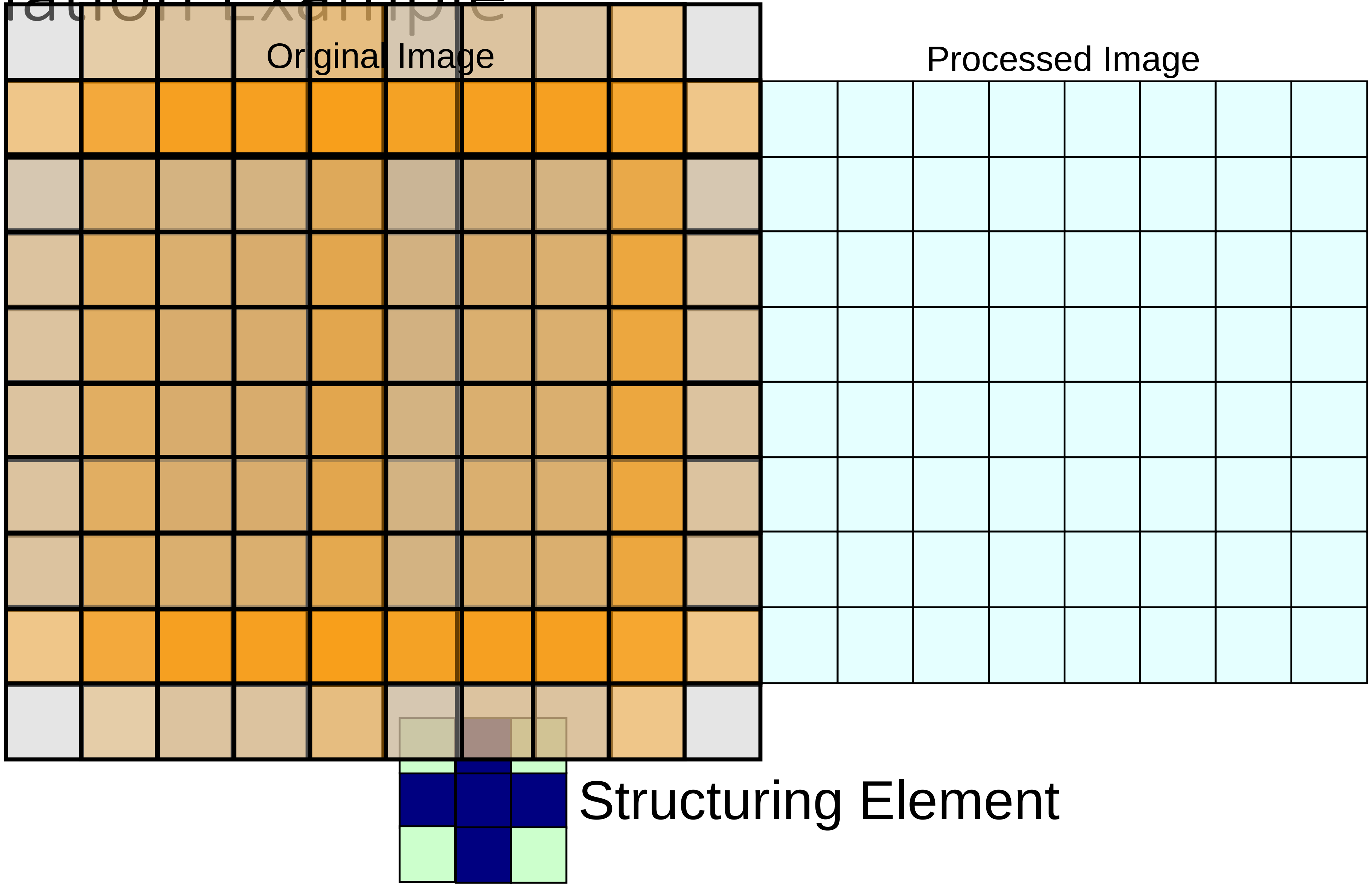
Dilation

Dilation of image f by structuring element s is given by $f \oplus s$

The structuring element s is positioned with its origin at (x, y) and the new pixel value is determined using the rule:

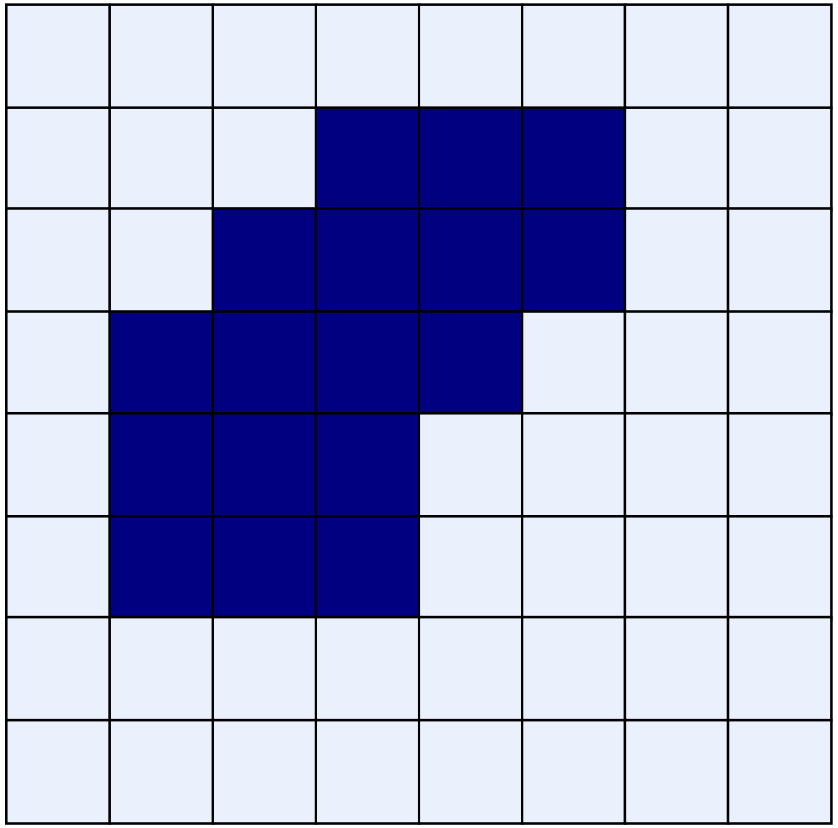
$$g(x, y) = \begin{cases} 1 \text{ if } s \text{ hits } f \\ 0 \text{ otherwise} \end{cases}$$

Dilation Example

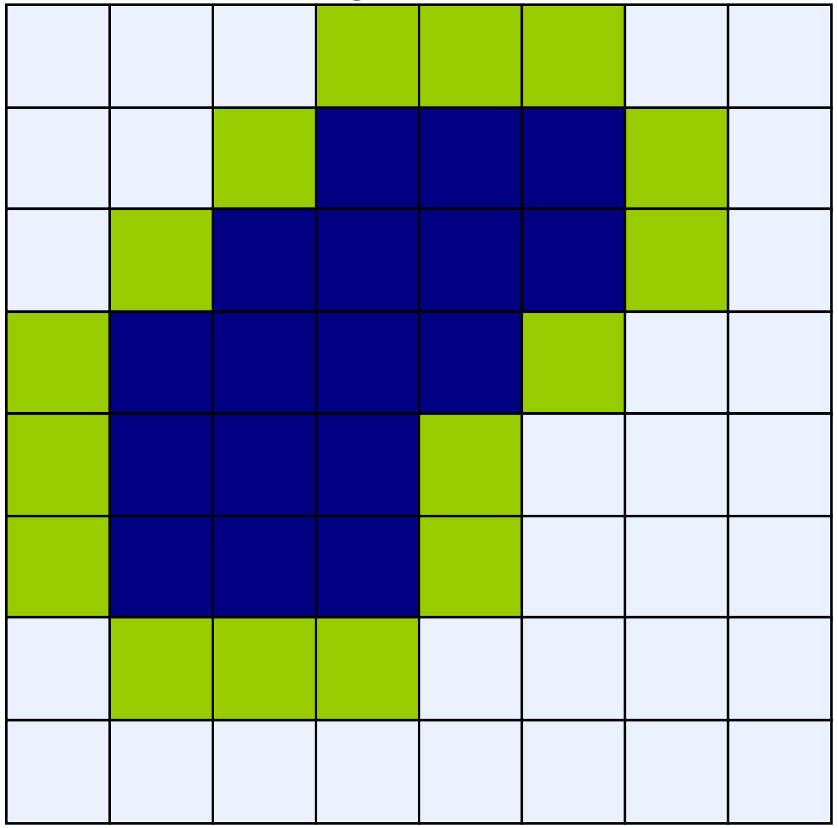


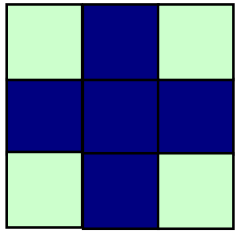
Dilation Example

Original Image



Processed Image With Dilated Pixels





Structuring Element

Dilation Example 1

Original image

Dilation by 3*3 square structuring element

Dilation by 5*5 square structuring element

Watch out: In these examples a 1 refers to a black pixel!

Dilation

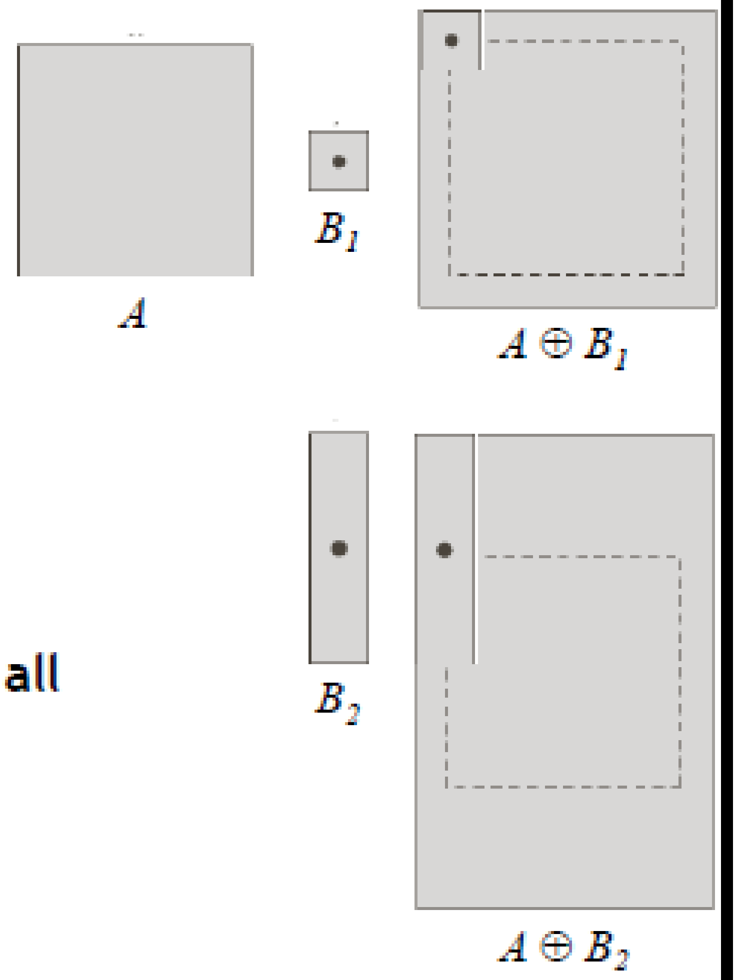
Slide Credit: Bastian Leibe

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

  ⇒ Expand connected components

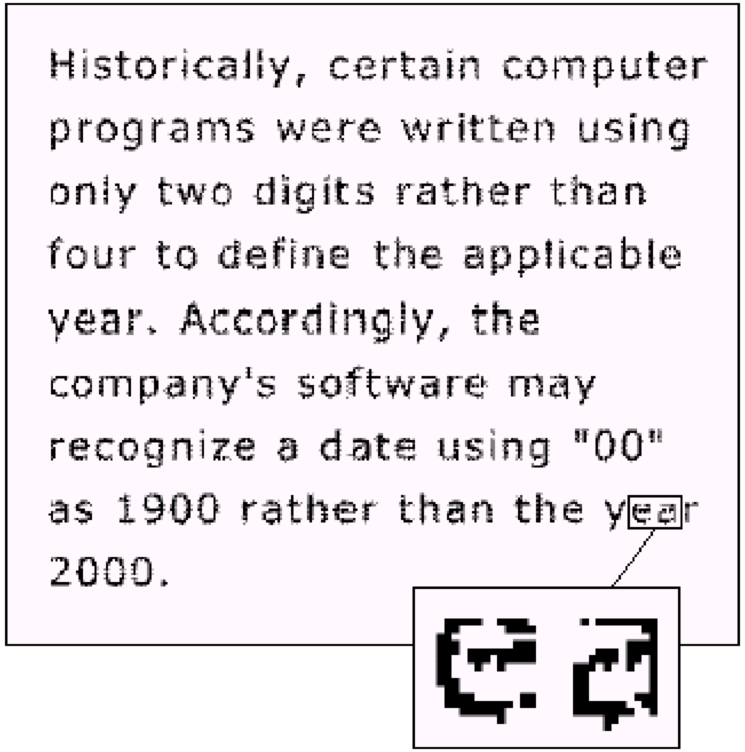
  ⇒ Grow features

  ⇒ Fill holes

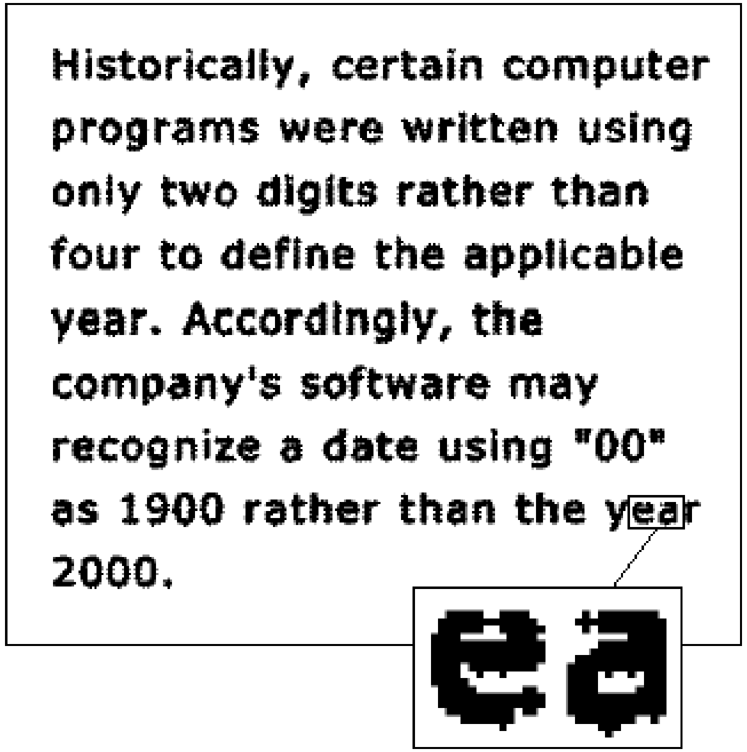


Dilation Example 2

Original image


Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

After dilation


Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

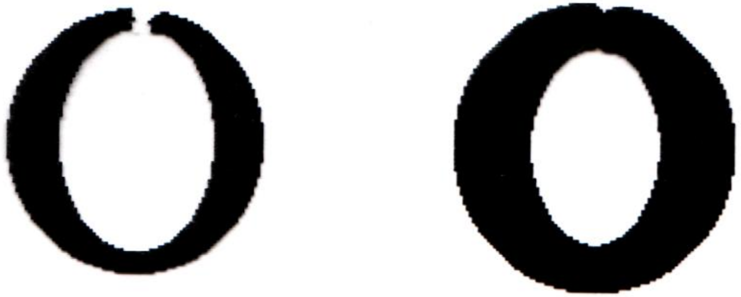
0	1	0
1	1	1
0	1	0

Structuring element

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

What Is Dilation For?

Dilation can repair breaks

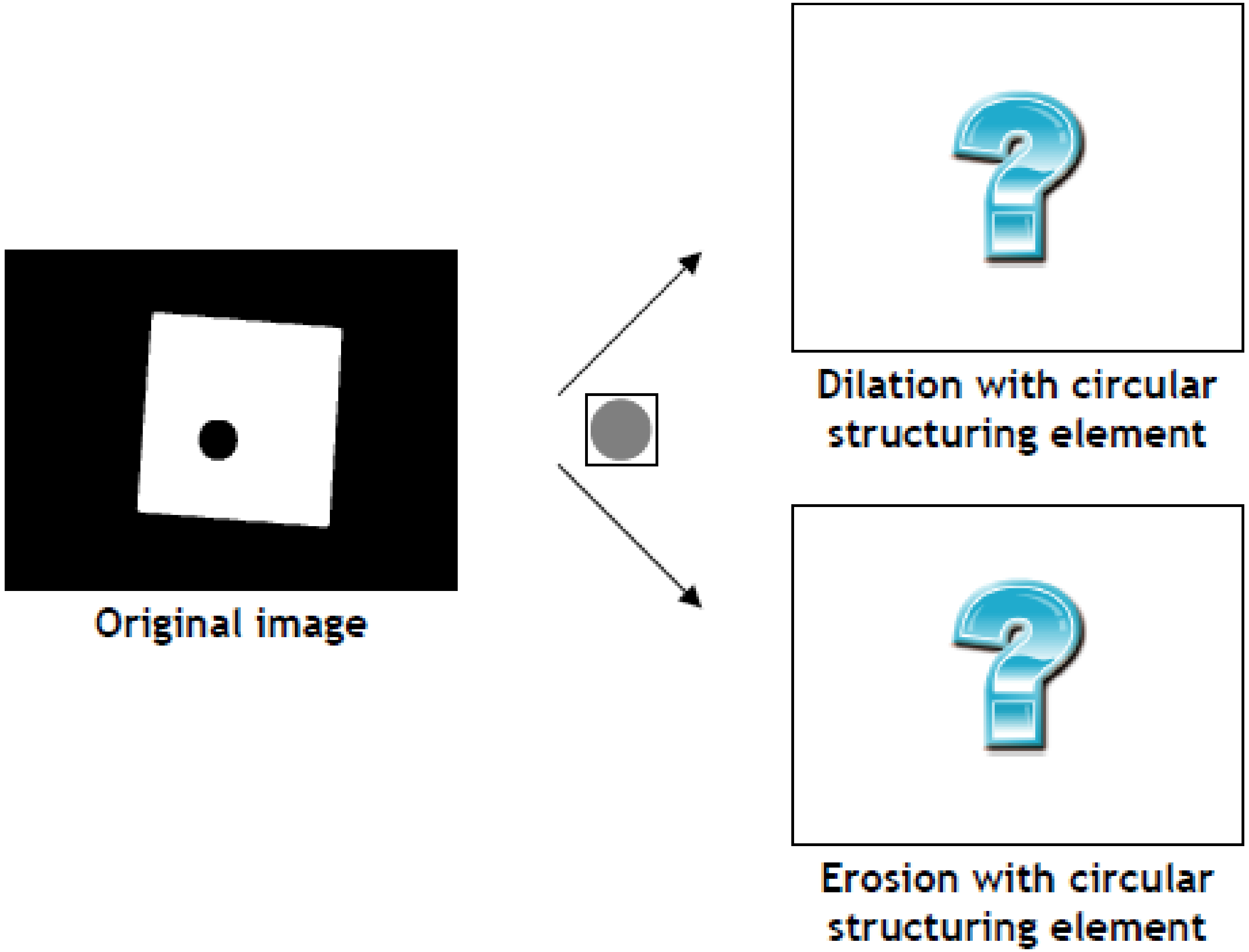


Dilation can repair intrusions

Watch out: Dilation enlarges objects

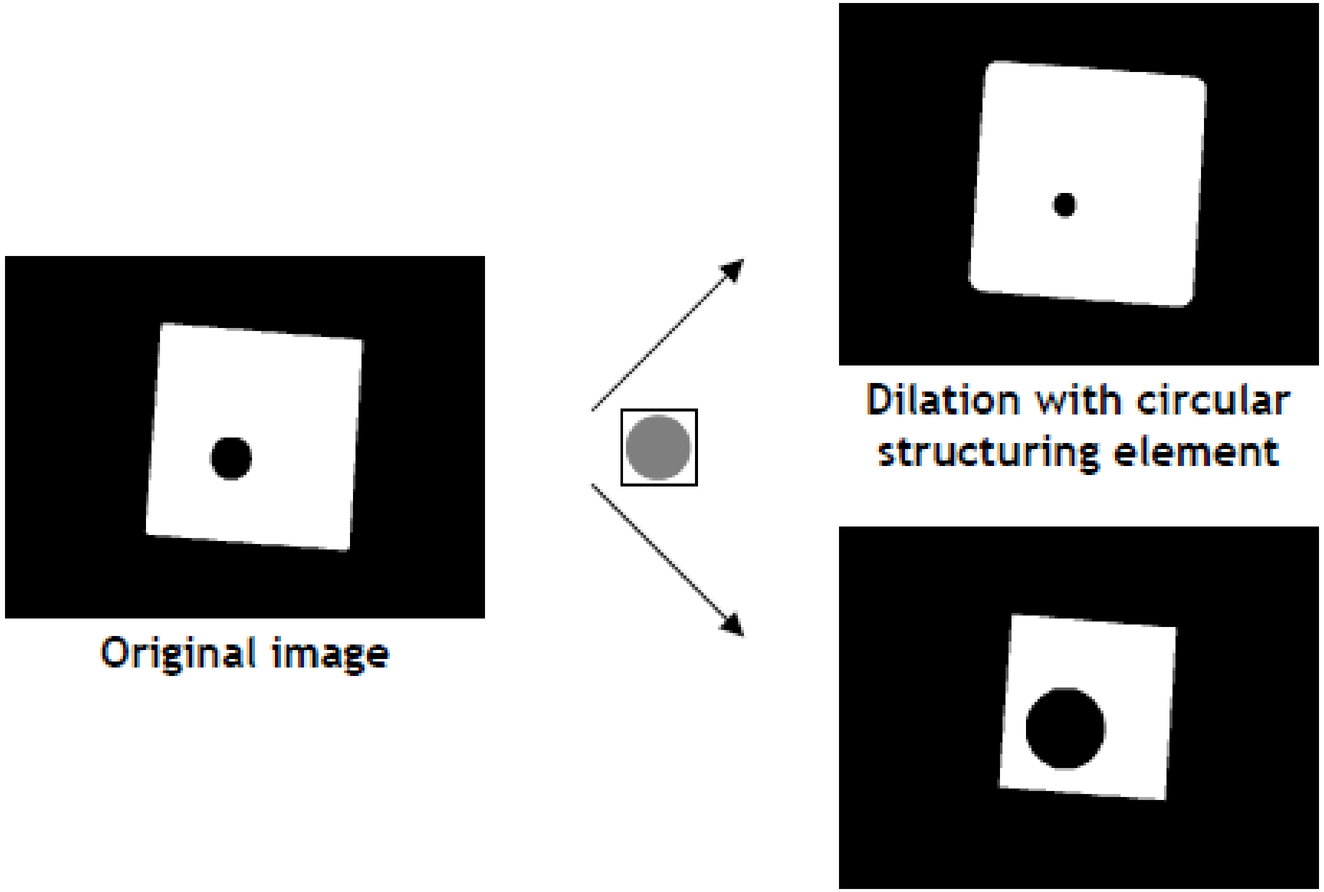
Effects

Slide Credit: Bastian Leibe



Effects

Slide Credit: Bastian Leibe



Original image

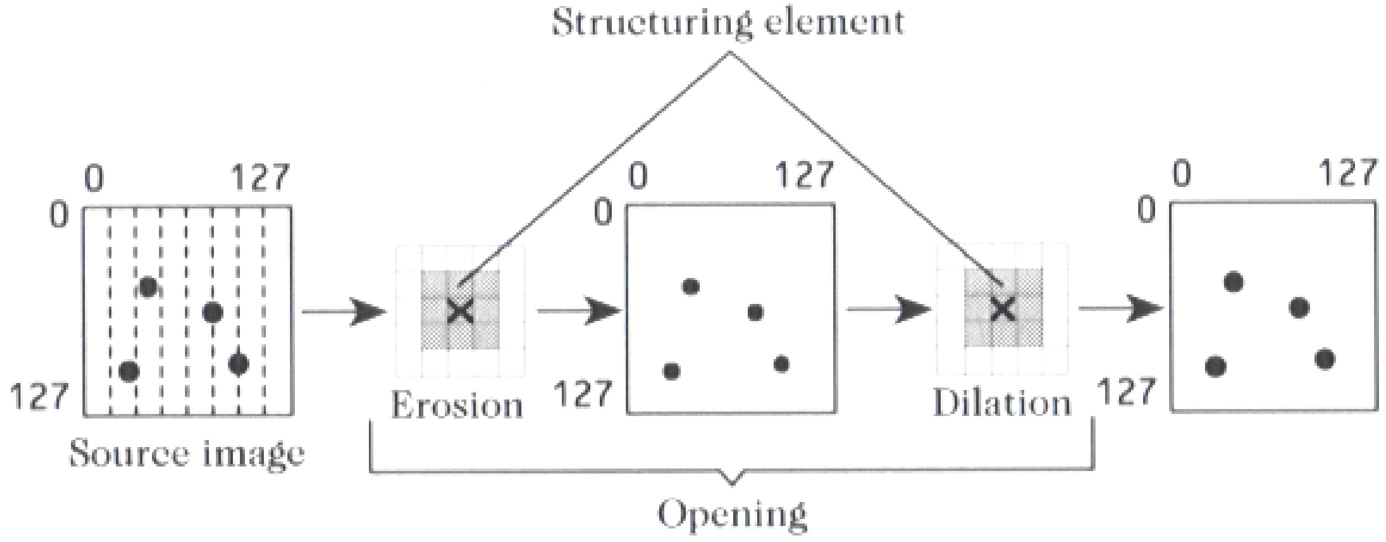
Dilation with circular structuring element

Erosion with circular structuring element

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

Morphological operations: survey

- Binary morphological procedures
  - Basic idea is to exploit prior knowledge of the shape of image distortions (or objects) in order to support the removal of these distortions



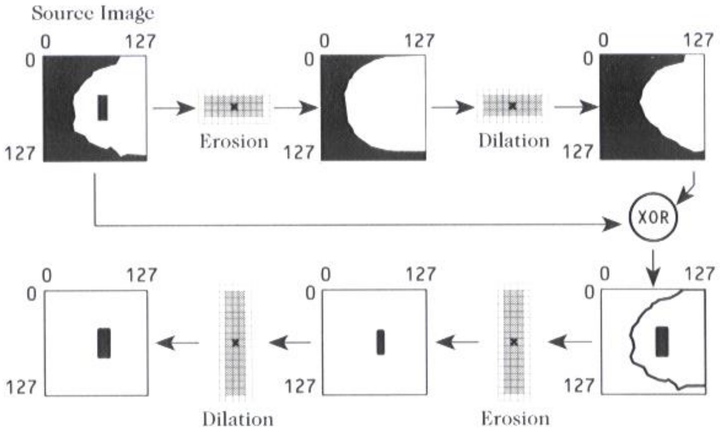
  - Erosion removes pixels from regions borders
  - Dilation adds pixels to borders
  - The removal or addition is determined by a structuring element which is an operator mask of a given shape

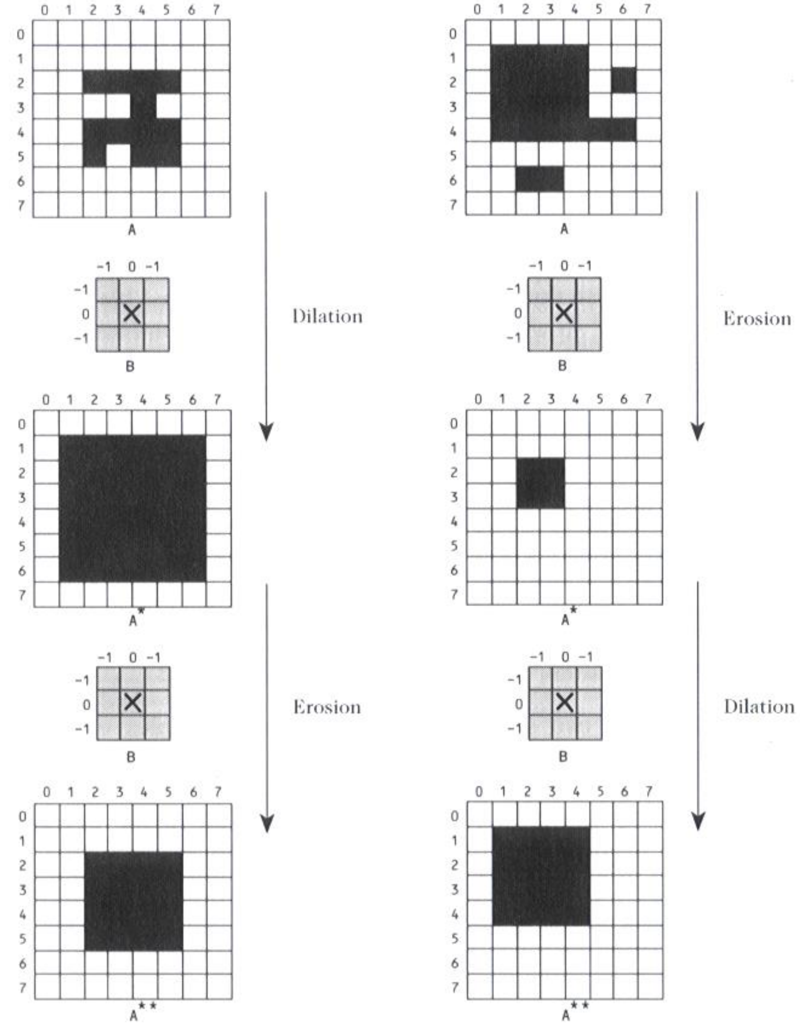
Morphological operations: survey

- Binary morphological procedures
  - The operations using the structuring element are based on the following rules:
    - Erosion (AND)
      - IF THE WHOLE STRUCTURING ELEMENT LIES INSIDE A REGION IN THE SOURCE IMAGE, THEN SET THE CURRENT PIXEL IN THE OUTPUT IMAGE TO 1
    - Dilation (OR)
      - IF AT LEAST ONE PIXEL OF THE STRUCTURING ELEMENT LIES INSIDE A REGION IN THE SOURCE IMAGE, THEN SET THE CURRENT PIXEL IN THE OUTPUT IMAGE TO 1
    - Opening
      - An erosion followed by a dilation. Used for removing tiny regions
    - Closing
      - A dilation followed by an erosion. Fills the gaps between regions

Morphological operations

- Binary morphological procedures





(a) (b)

Compound Operations

More interesting morphological operations can be performed by performing combinations of erosions and dilations

The most widely used of these *compound operations* are:

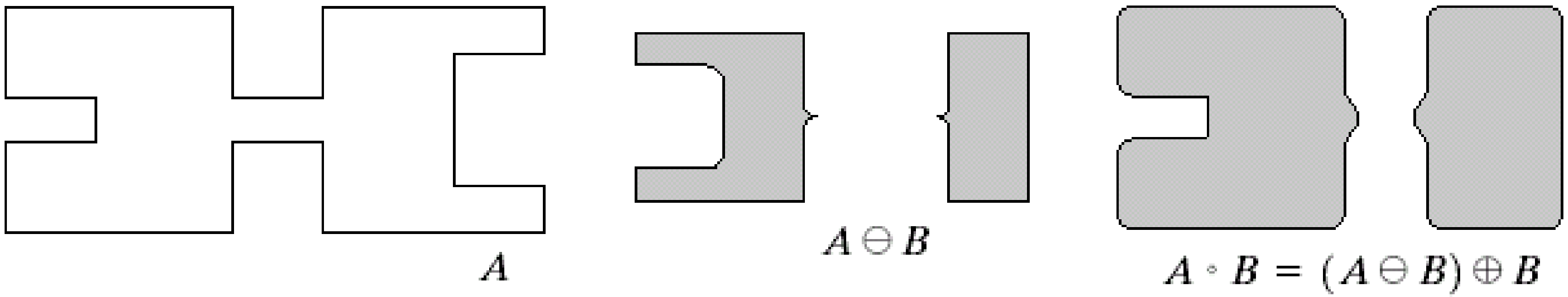
- Opening
- Closing

Opening

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

The opening of image f by structuring element s, denoted $f \circ s$ is simply an erosion followed by a dilation

$$f \circ s = (f \ominus s) \oplus s$$



Original shape

After erosion

After dilation (opening)

Note a disc shaped structuring element is used

Opening Example

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Original Image

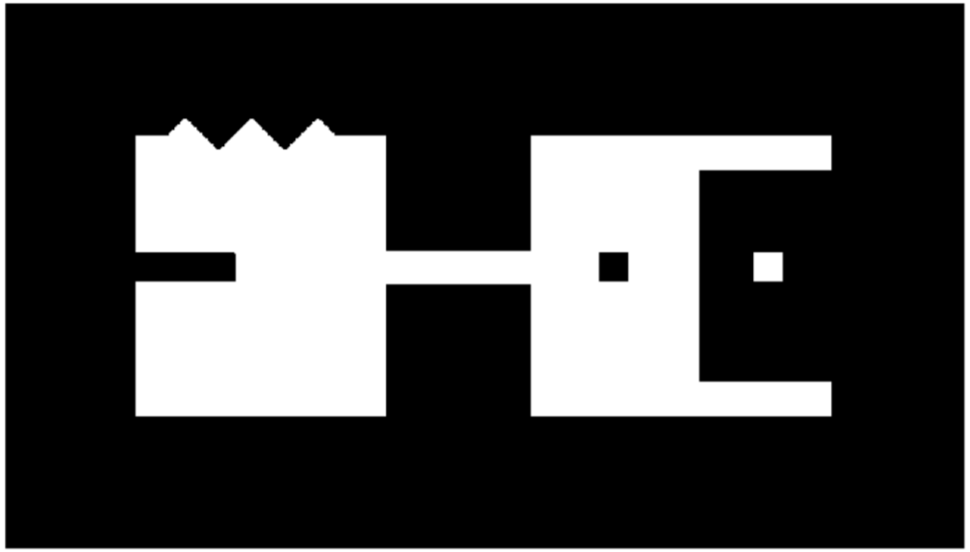
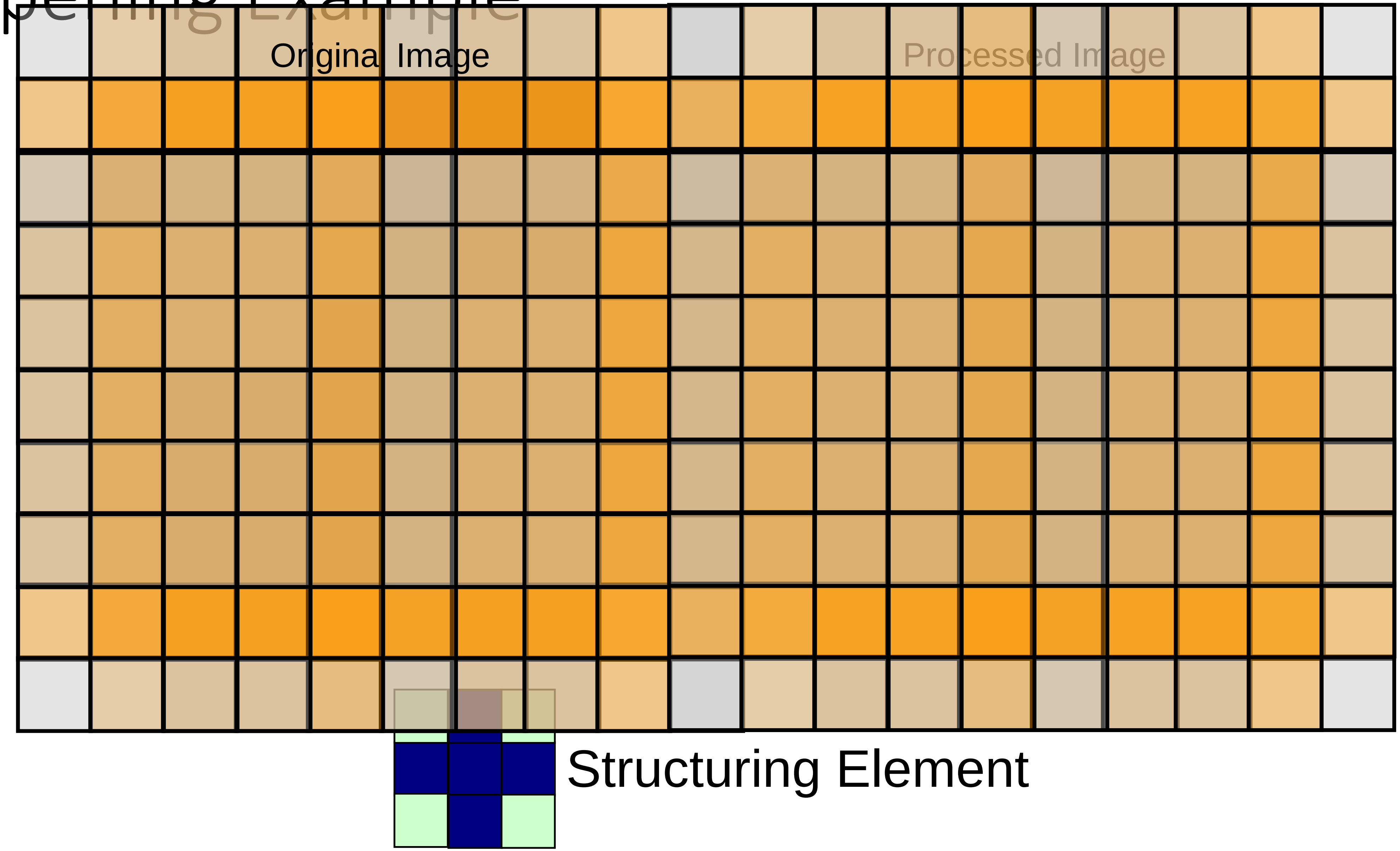


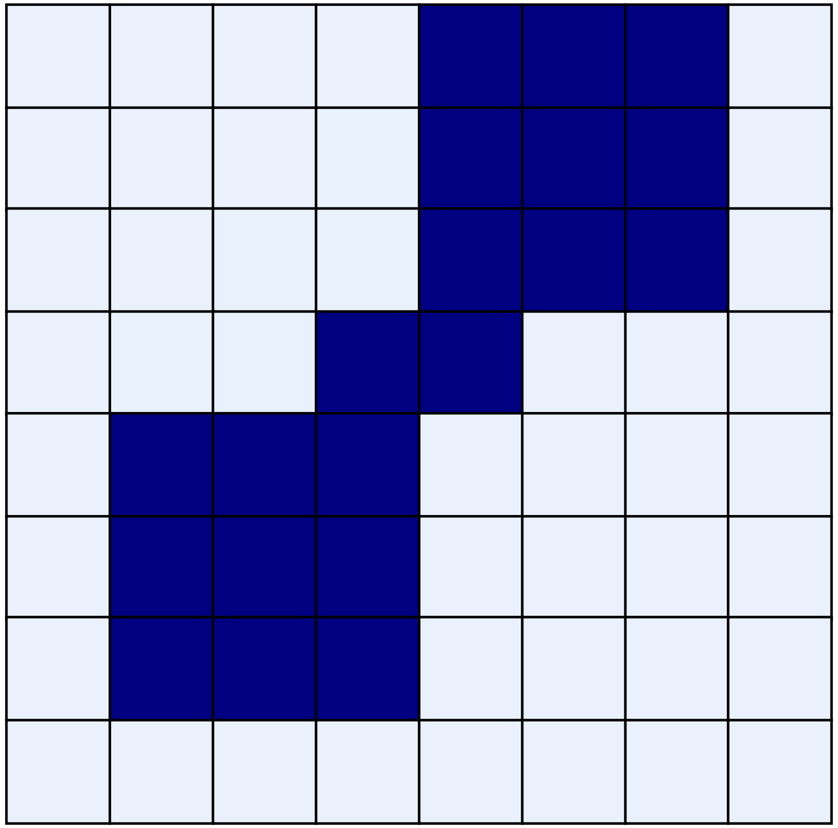
Image After Opening

Opening Example

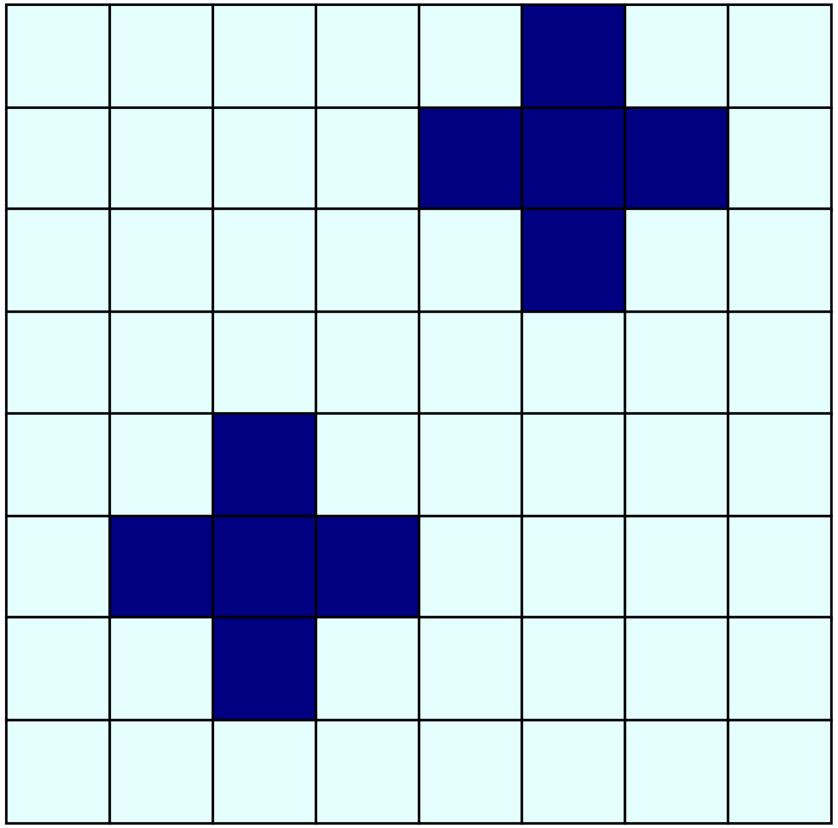


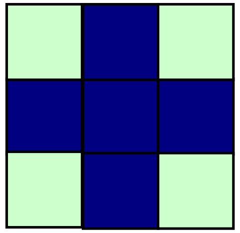
Opening Example

Original Image



Processed Image





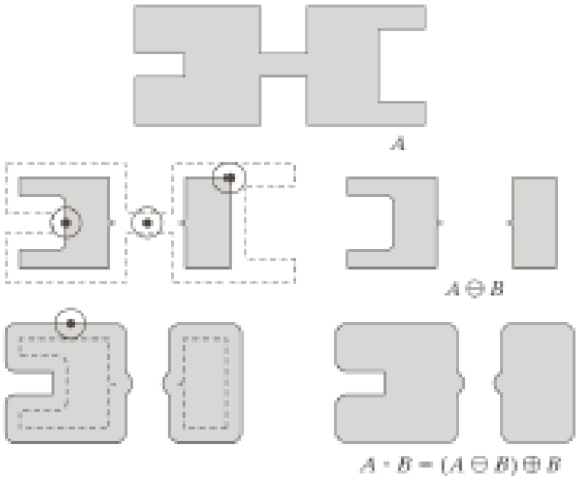
Structuring Element

Opening

Slide Credit: Bastian Leibe

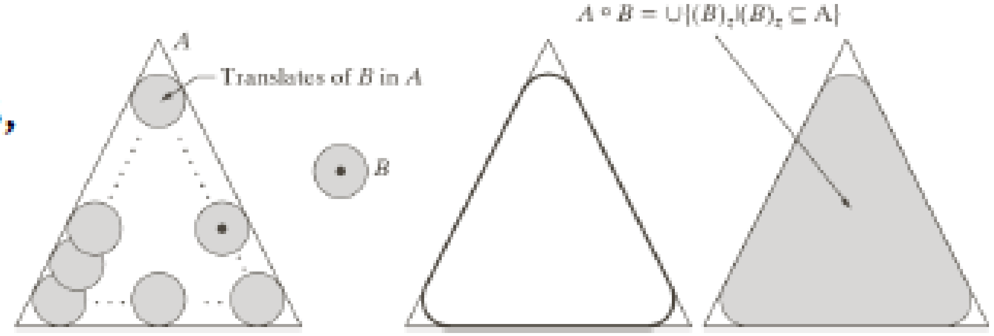
- Definition
  - Sequence of Erosion and Dilation

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



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

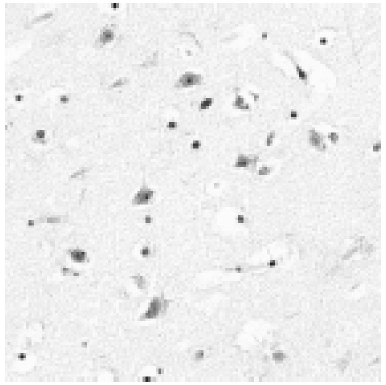
  $\Rightarrow$ Remove small objects, keep original shape.



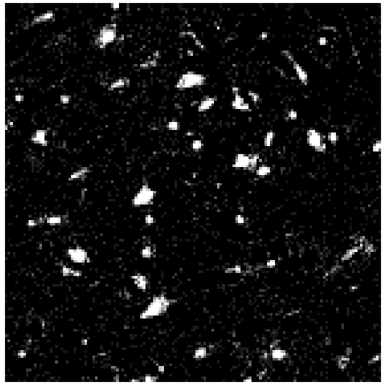
Effect of Opening

Slide Credit: Bastian Leibe

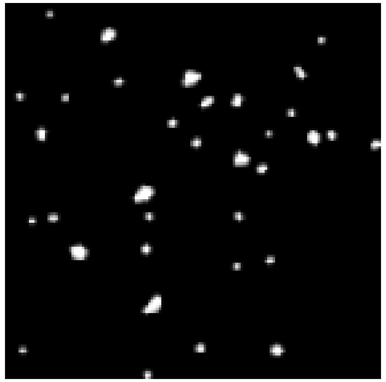
- Feature selection through *size* of structuring element



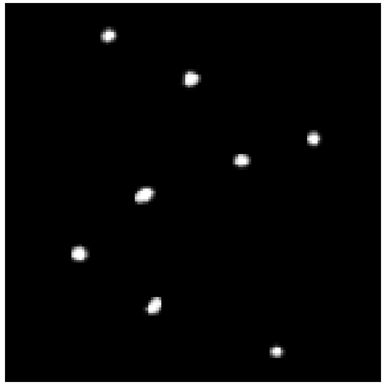
Original image



Thresholded



Opening with small structuring element



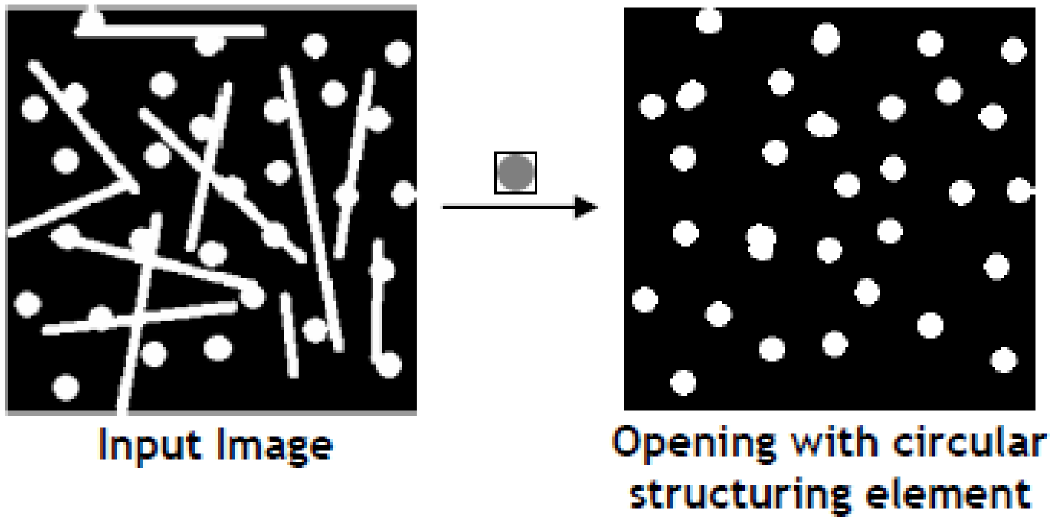
Opening with larger structuring element

mage Source: http://homepages.inf.ed.ac.uk/rbf/HIPR2/ B. Leibe

Effect of Opening

Slide Credit: Bastian Leibe

- Feature selection through *shape* of structuring element



- *How could we have extracted the lines?*

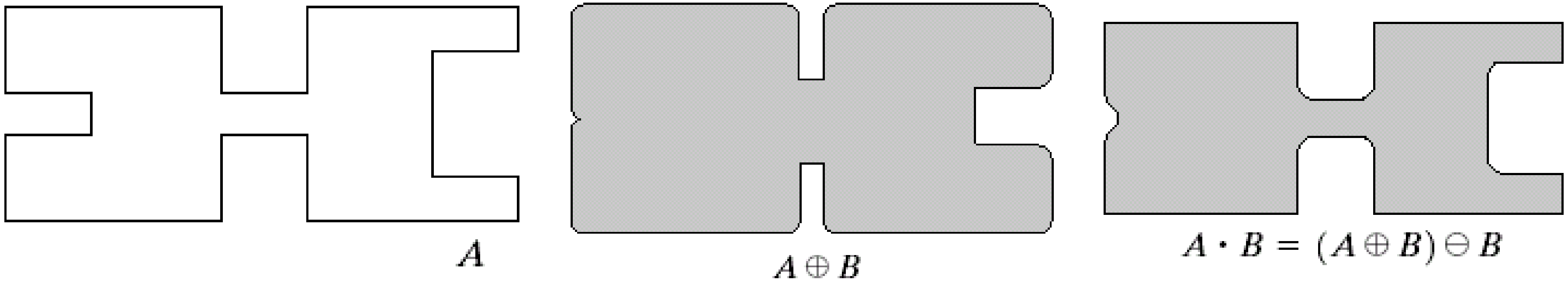
Image Source: http://homepages.inf.ed.ac.uk/rbf/HIPR2/

Closing

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

The closing of image f by structuring element s, denoted $f \bullet s$ is simply a dilation followed by an erosion

$$f \bullet s = (f \oplus s) \ominus s$$



Original shape

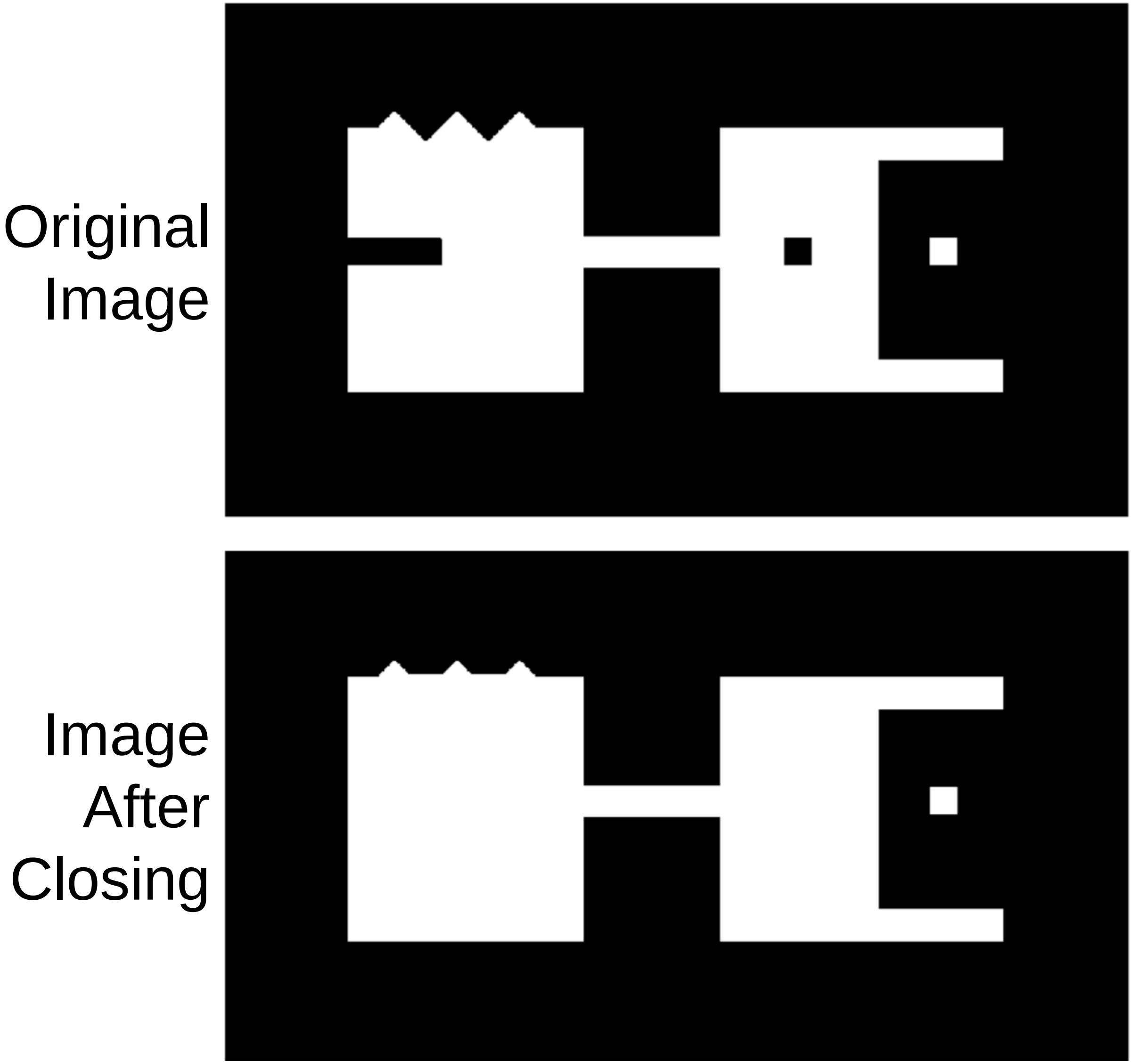
After dilation

After erosion (closing)

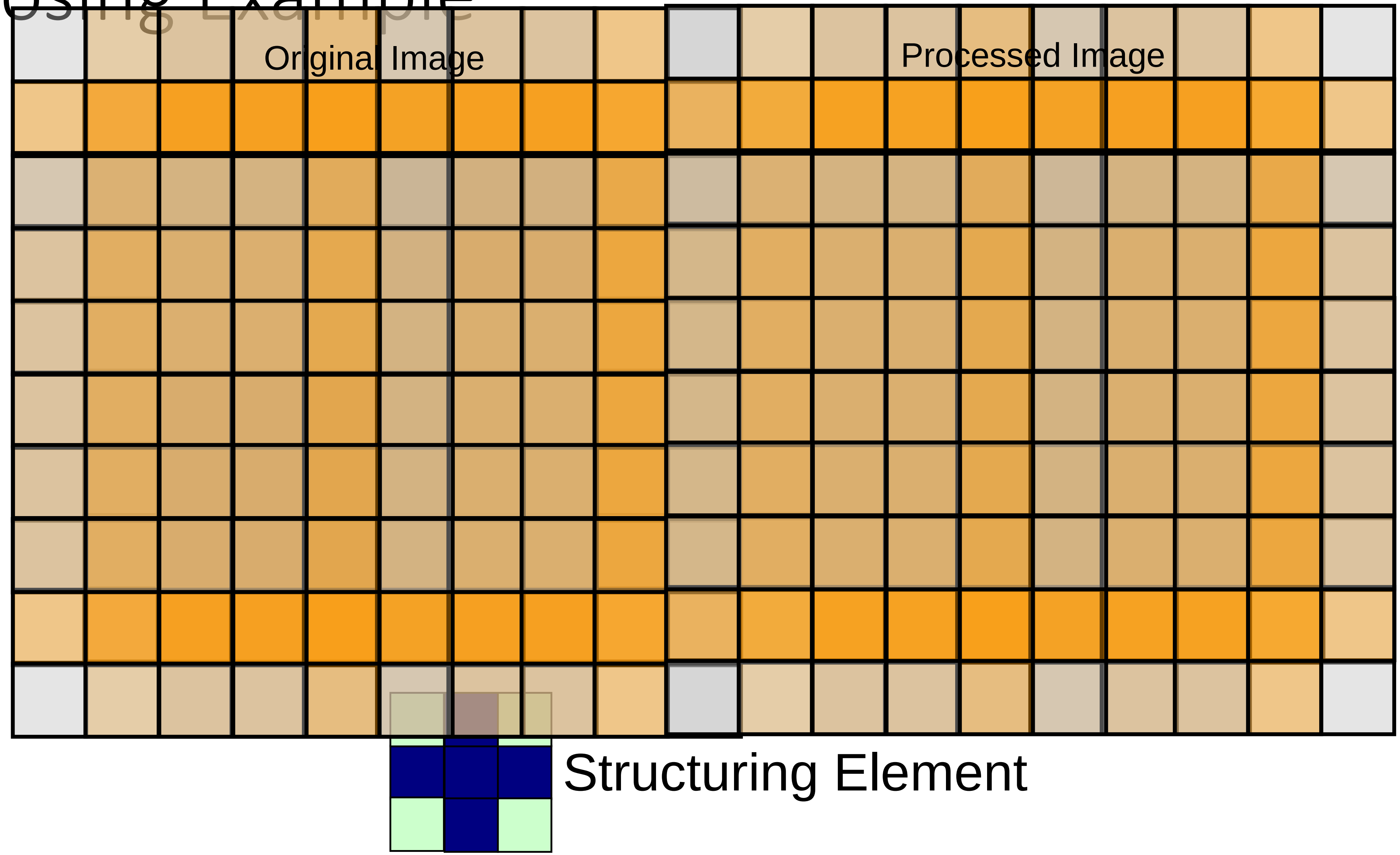
Note a disc shaped structuring element is used

Closing Example

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

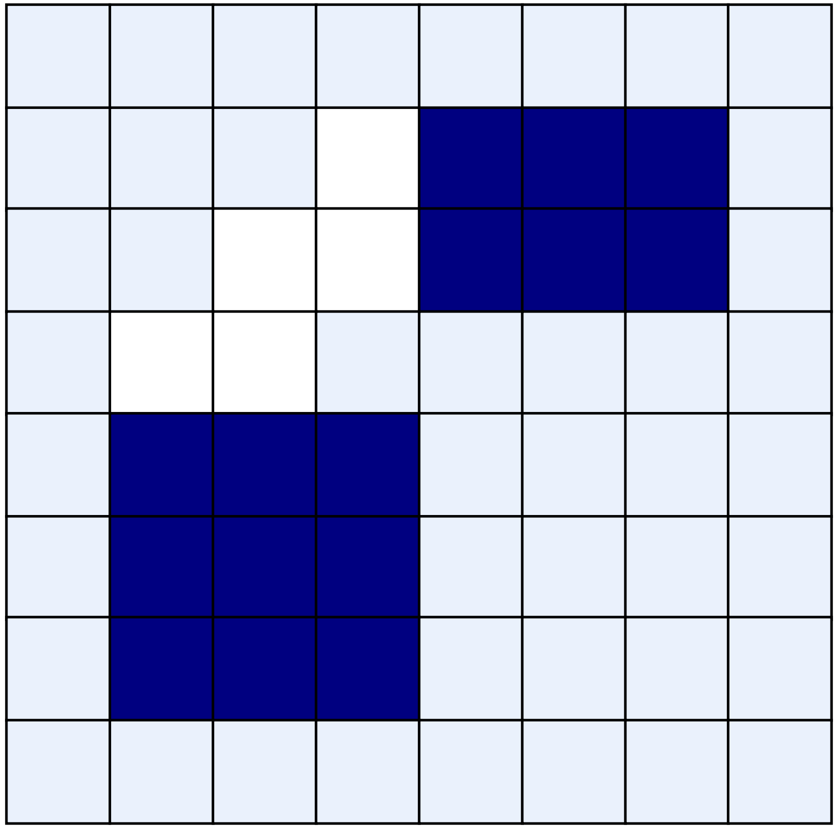


Closing Example

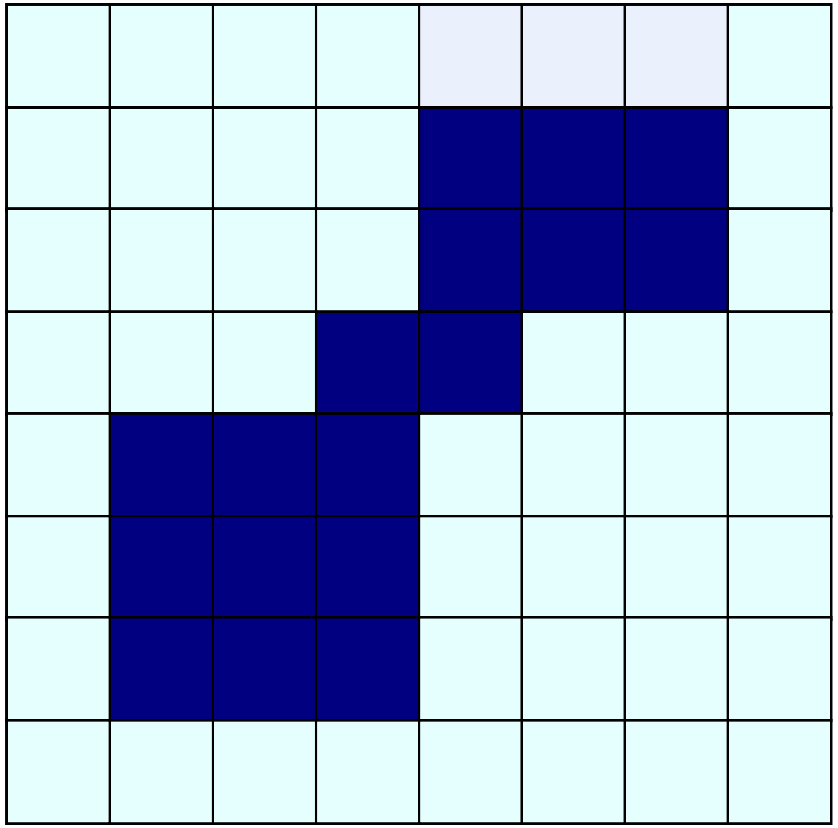


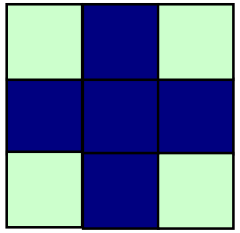
Closing Example

Original Image



Processed Image





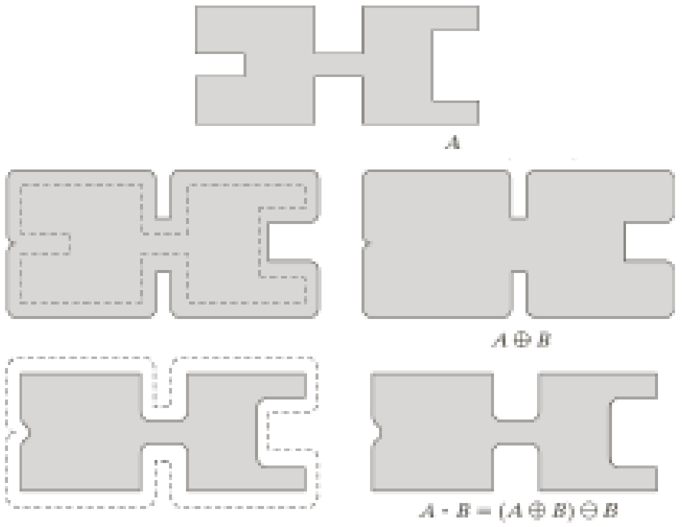
Structuring Element

Closing

Slide Credit: Bastian Leibe

- Definition
  - Sequence of Dilation and Erosion

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



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

  $\Rightarrow$ Fill holes,
  keep original shape.

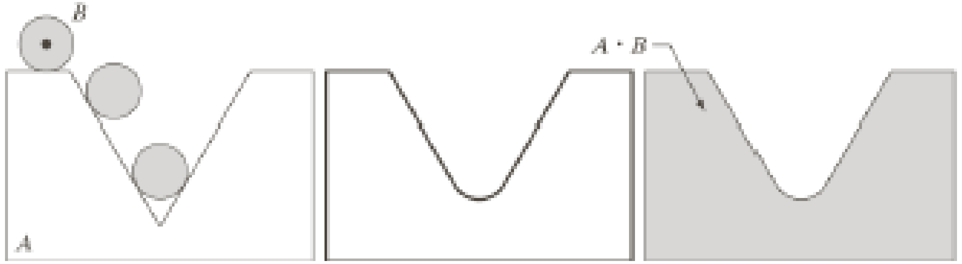


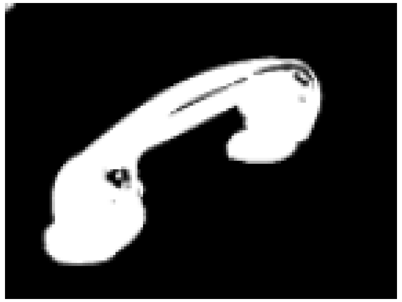
Image Source: R.C. Gonzales & R.E. Woods

Effect of Closing

Slide Credit: Bastian Leibe

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

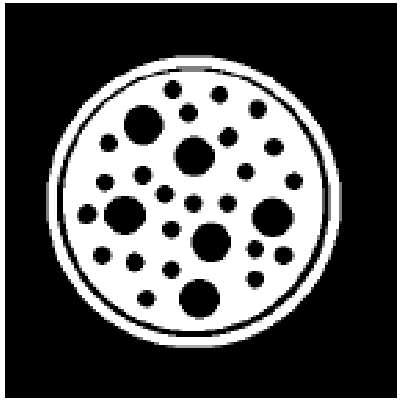
Original image



Thresholded

Closing with circular structuring element

Size of structuring element determines which structures are selectively filled.



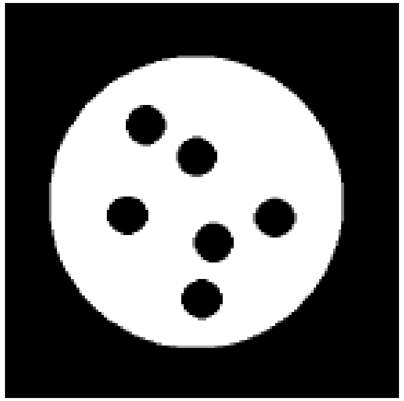
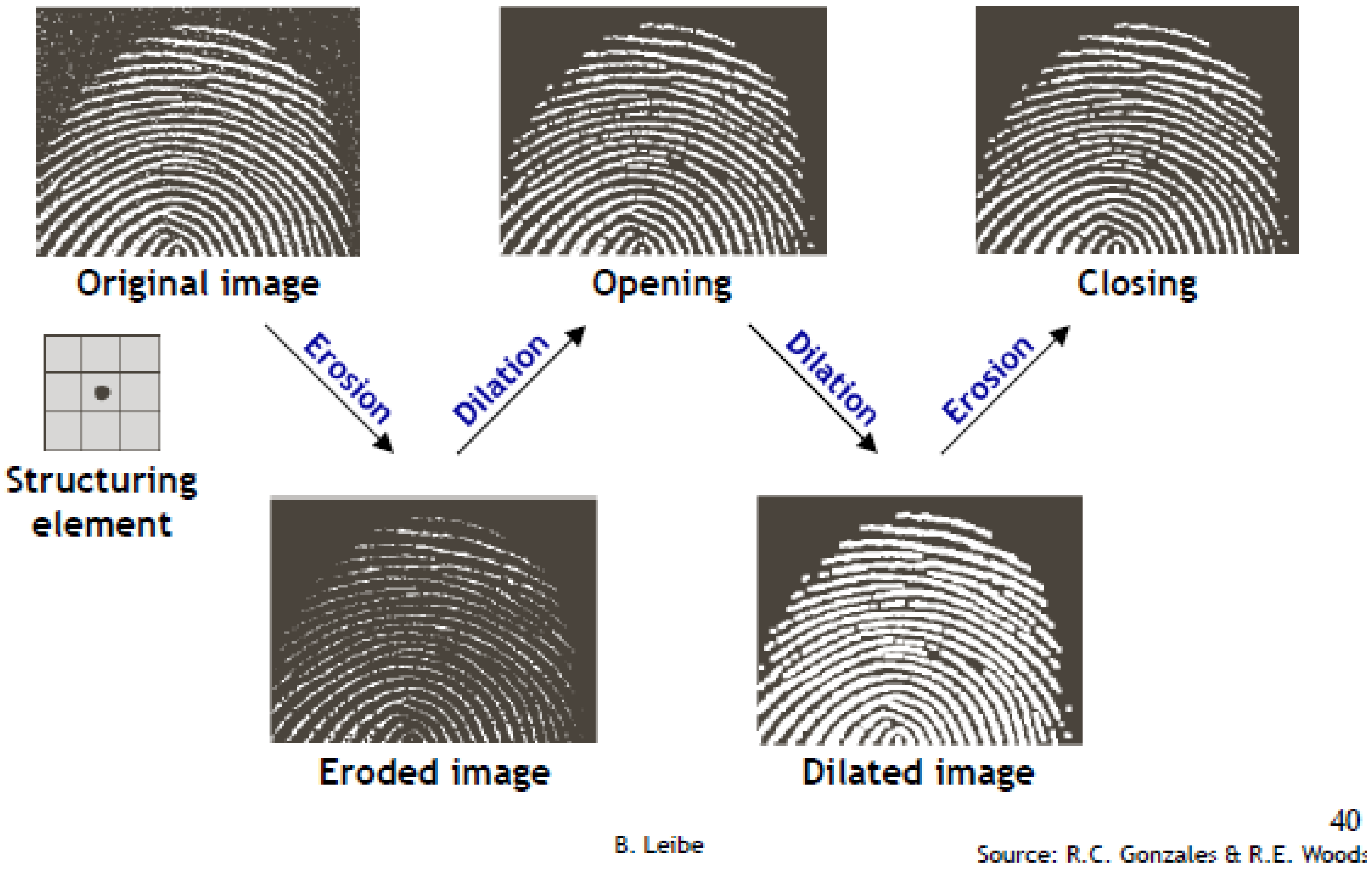


Image Source: http://homepages.inf.ed.ac.uk/rbf/HIPR2/

Example Application: Opening + Closing

Slide Credit: Bastian Leibe

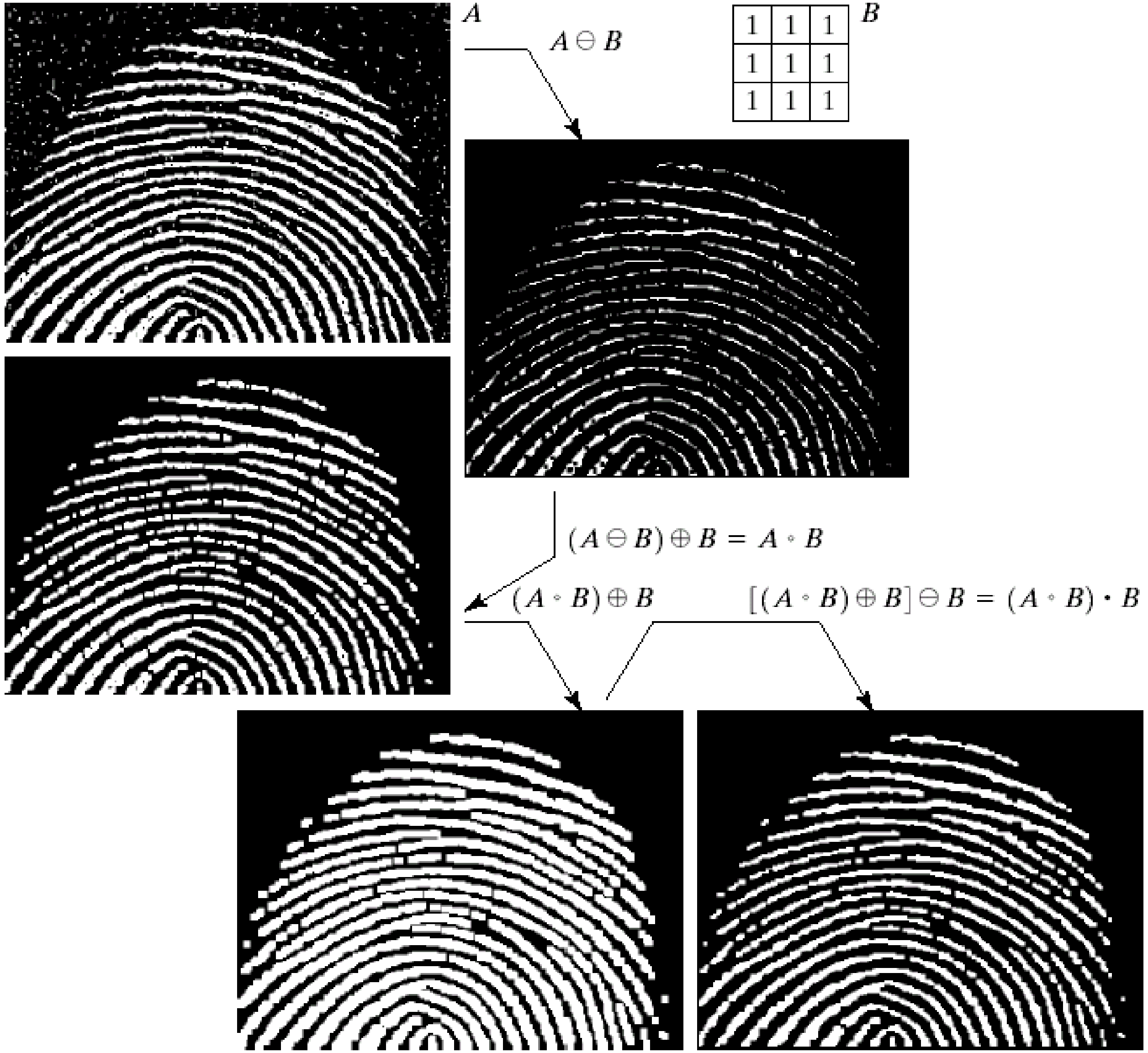


C = imopen(A, B)

C = imclose(A, B)

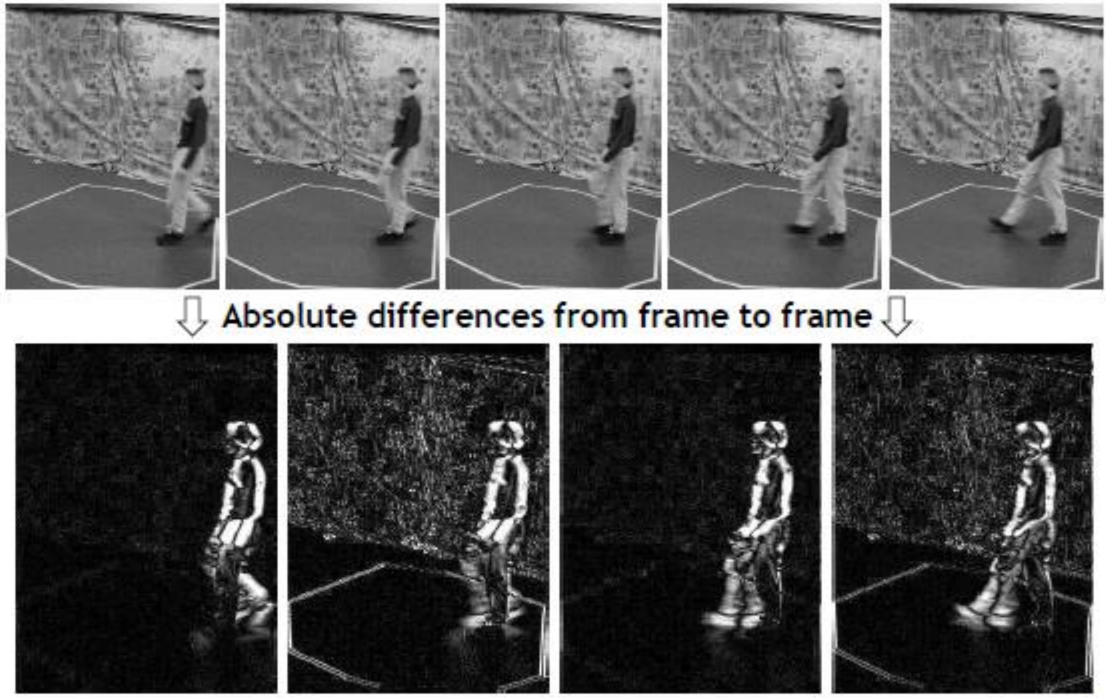
Morphological Processing Example

Images taken from Gonzalez & Woods, Digital Image Processing (2002)



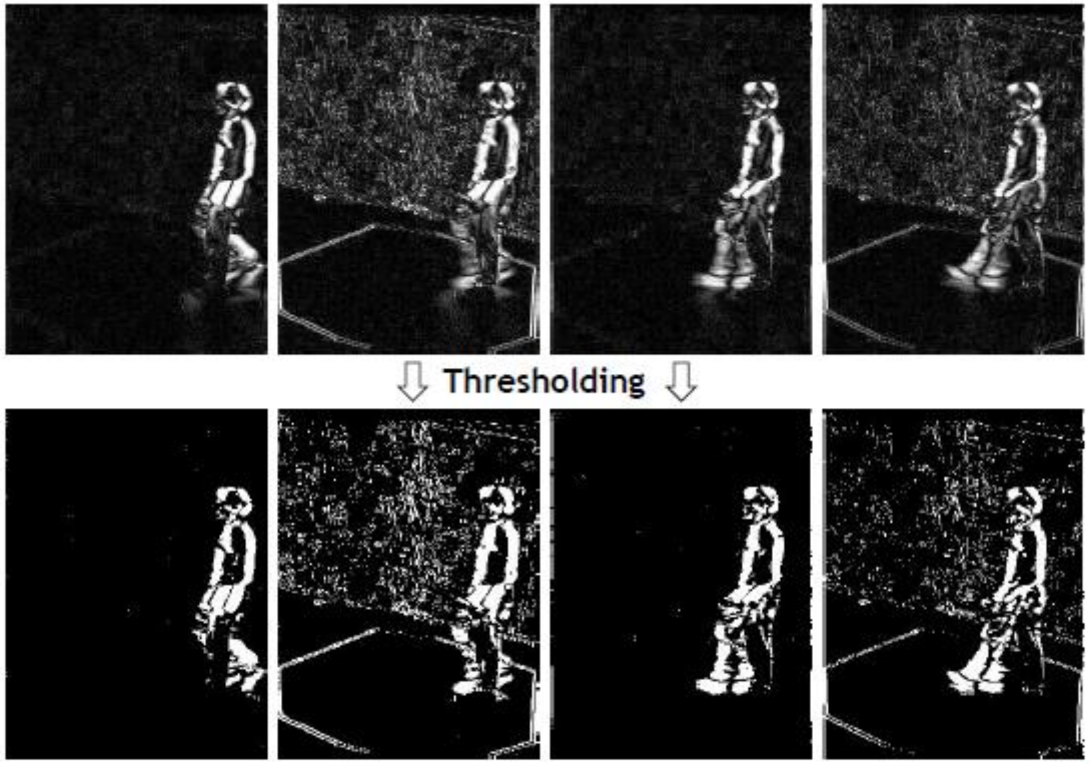
Application: Blob Tracking

Slide Credit: Bastian Leibe



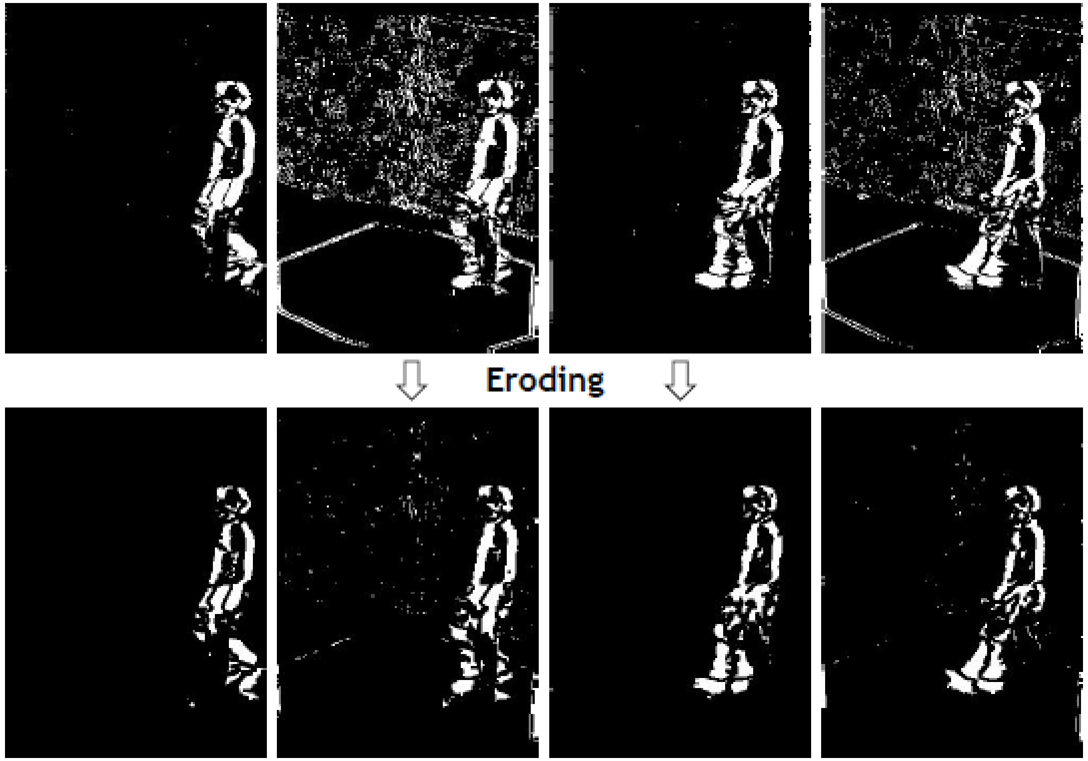
Applications: Vision-based Interfaces

Slide Credit: Bastian Leibe



Applications: Vision-based Interfaces

Slide Credit: Bastian Leibe



Morphological Algorithms

Using the simple technique we have looked at so far we can begin to consider some more interesting morphological algorithms

We will look at:

- Boundary extraction
- Region filling

There are lots of others as well though:

- Extraction of connected components
- Thinning/thickening
- Skeletonisation

IPT function bwmorph implements a variety of useful operations based on com-binations of dilations, erosions, and lookup table operations. Its calling syntax is

```
g = bwmorph(f, operation, n)
```

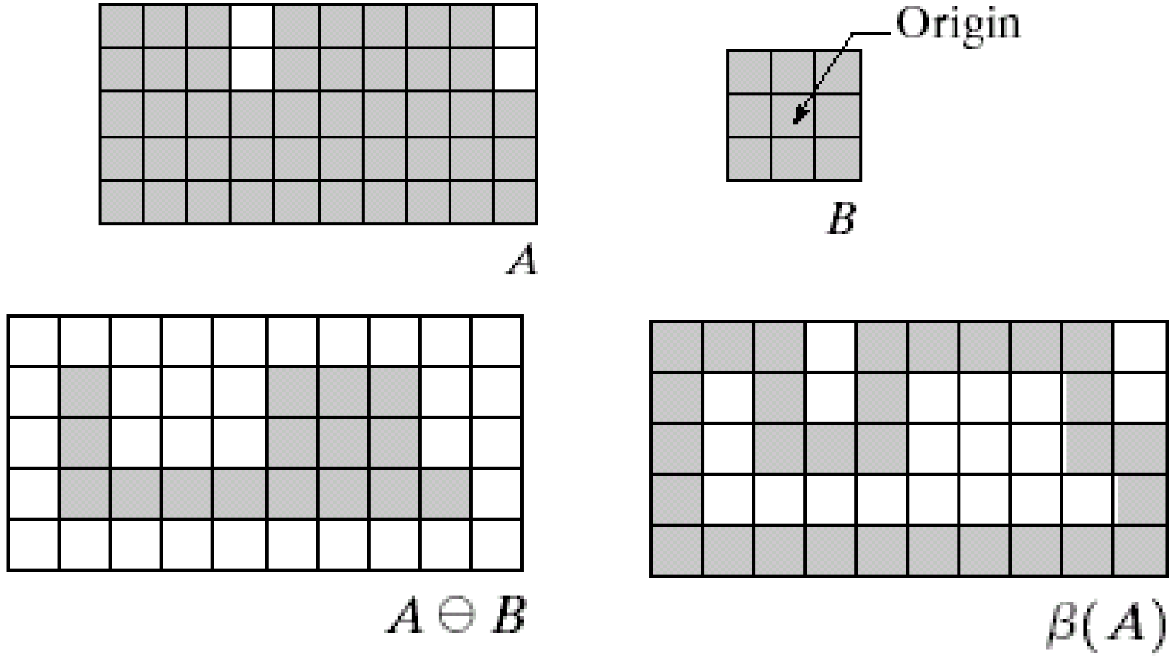
Boundary Extraction

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Extracting the boundary (or outline) of an object is often extremely useful

The boundary can be given simply as

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



Boundary Extraction Example

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

A simple image and the result of performing boundary extraction using a square 3*3 structuring element

Original Image

Extracted Boundary

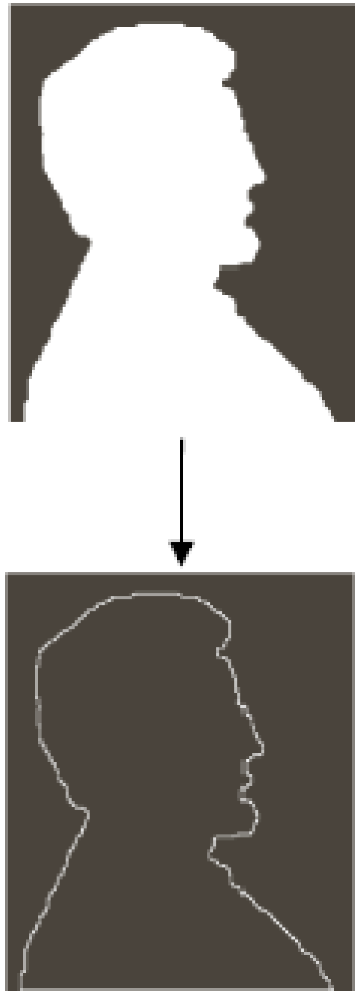
Morphological Boundary Extraction

Slide Credit: Bastian Leibe

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

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

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



Morphology Operators on Grayscale Images

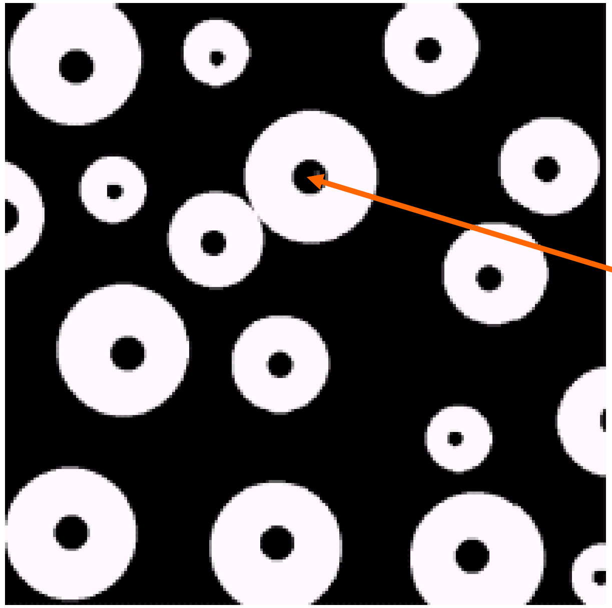
Slide Credit: Bastian Leibe

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

Original Dilated Eroded

Region Filling

Given a pixel inside a boundary, *region filling* attempts to fill that boundary with object pixels (1s)



Given a point inside here, can we fill the whole circle?

Region Filling (cont…)

The key equation for region filling is

$$X_k = (X_{k-1} \oplus B) \cap A^c \qquad k = 1,2,3.....$$

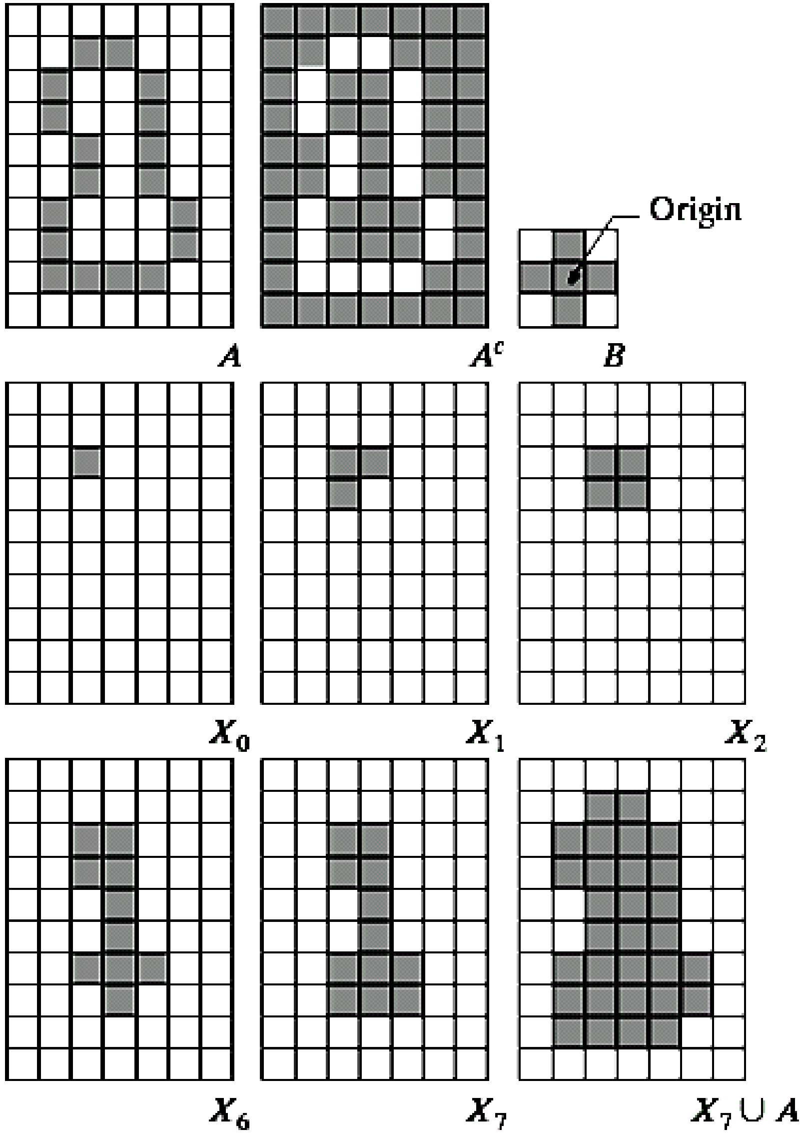
Where X_0 is simply the starting point inside the boundary, B is a simple structuring element and A^c is the complement of A

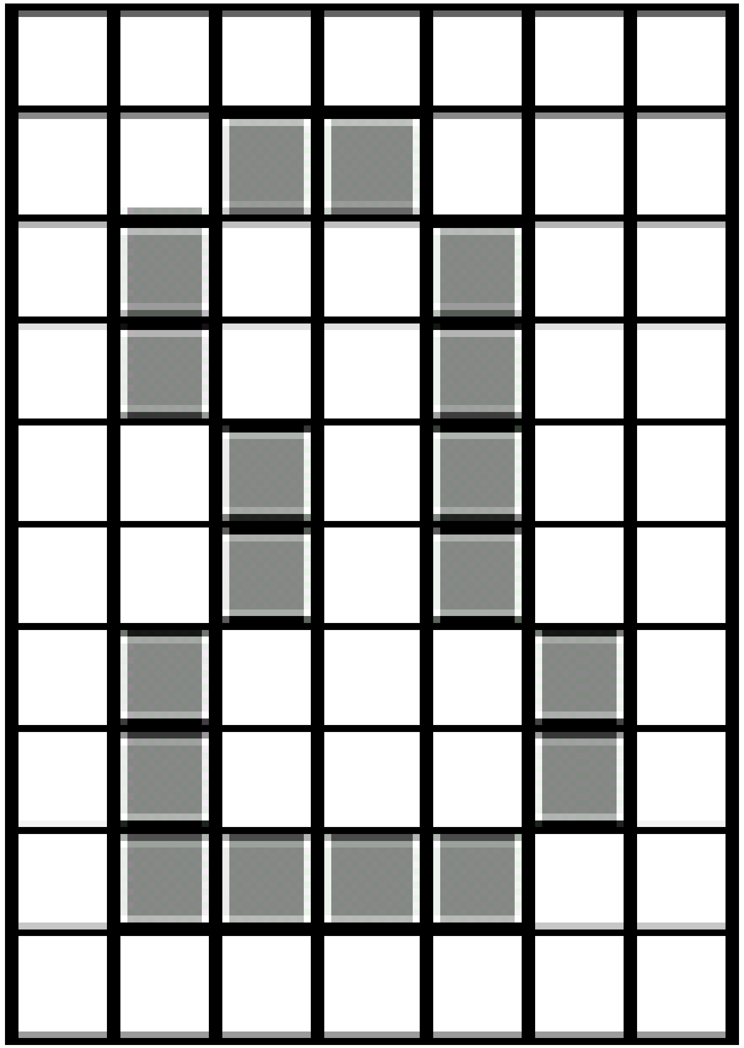
This equation is applied repeatedly until X_k is equal to X_{k-1}

Finally the result is unioned with the original boundary

Region Filling Step By Step

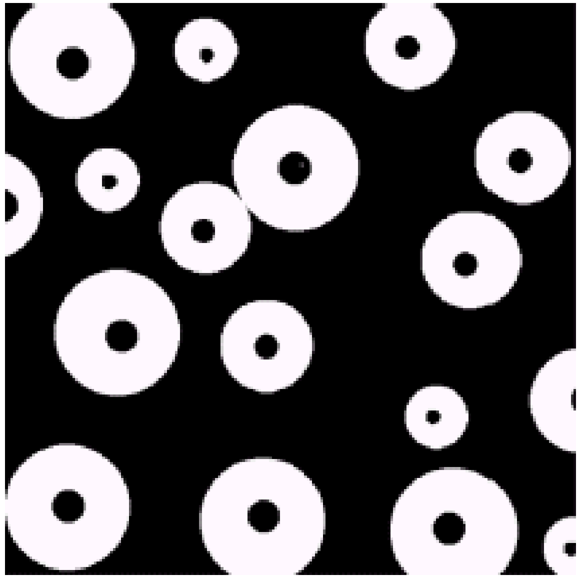
Images taken from Gonzalez & Woods, Digital Image Processing (2002)



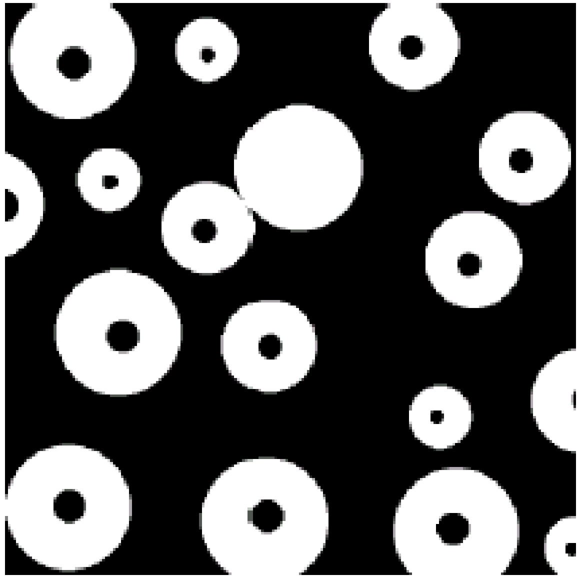


Region Filling Example

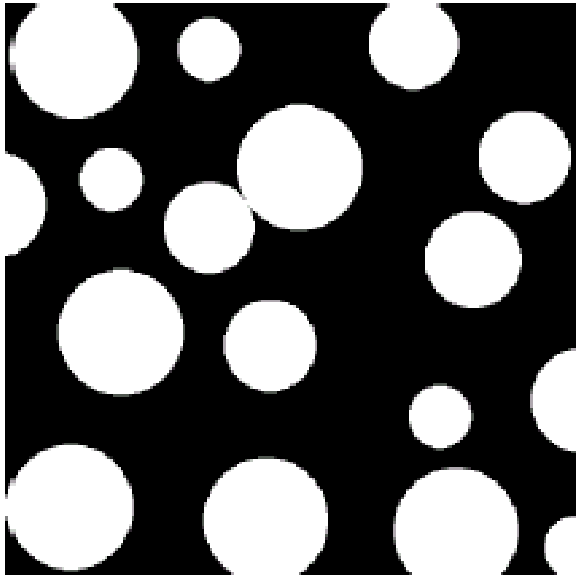
Images taken from Gonzalez & Woods, Digital Image Processing (2002)



Original Image



One Region Filled



All Regions Filled

Summary

The purpose of morphological processing is primarily to remove imperfections added during segmentation

The basic operations are *erosion* and *dilation*

Using the basic operations we can perform *opening* and *closing*

More advanced morphological operation can then be implemented using combinations of all of these