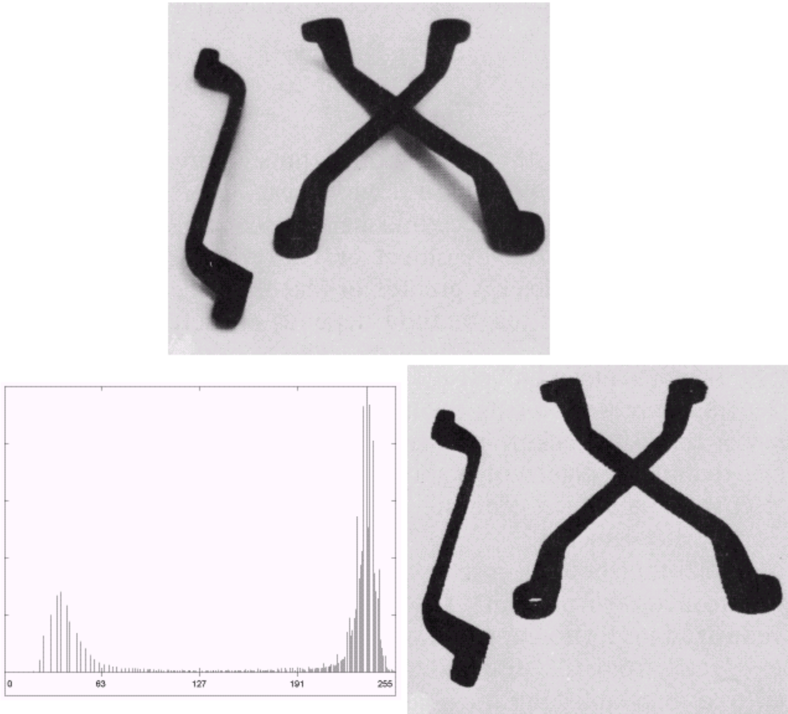
Introduction to Image Processing

Ch 10.3. Thresholding with Otsu's Method
Ch 9. Morphological Image Processing
Ch 10.2. Line Detection with Hough Transform

Kuan-Wen Chen

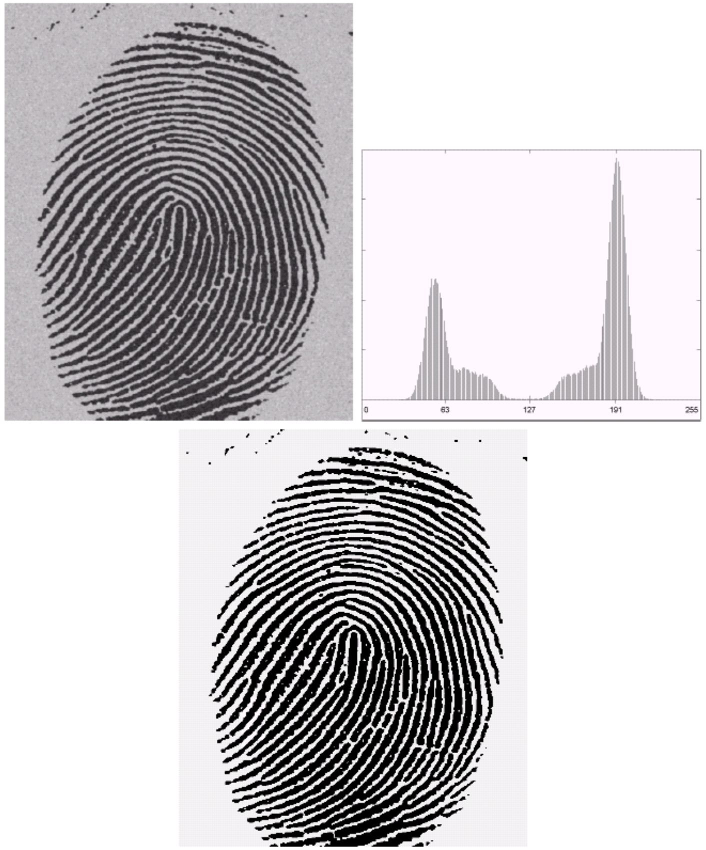
10.3 Thresholding with Otsu’s Method

- Histogram Analysis for Image Thresholding



a
b c

FIGURE 10.28 (a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.



a b
c

FIGURE 10.29 (a) Original image. (b) Image histogram. (c) Result of segmentation with the threshold estimated by iteration. (Original courtesy of the National Institute of Standards and Technology.)

10.3 Thresholding with Otsu's Method

- Histogram Analysis for Image Thresholding

Example:

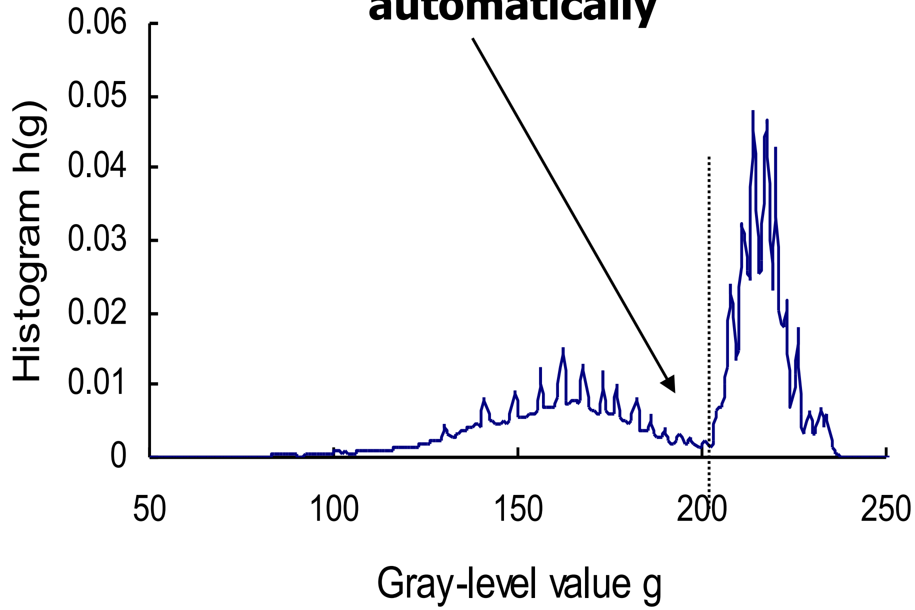
background

Object

Image *X*

Histogram

Detect optimal threshold T automatically



Thresholding at T=200

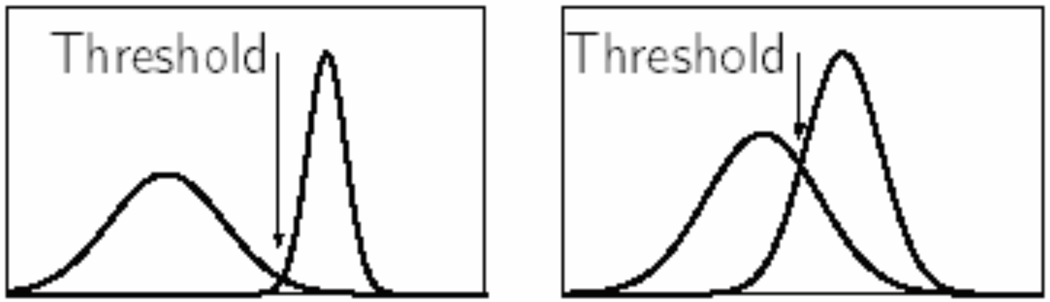
Binary image *Y*

10.3 Thresholding with Otsu's Method

- Histogram Analysis for Image Thresholding

  - Histogram shape can be useful in locating the threshold.
    - However it is not reliable for threshold selection when peaks are not clearly resolved.

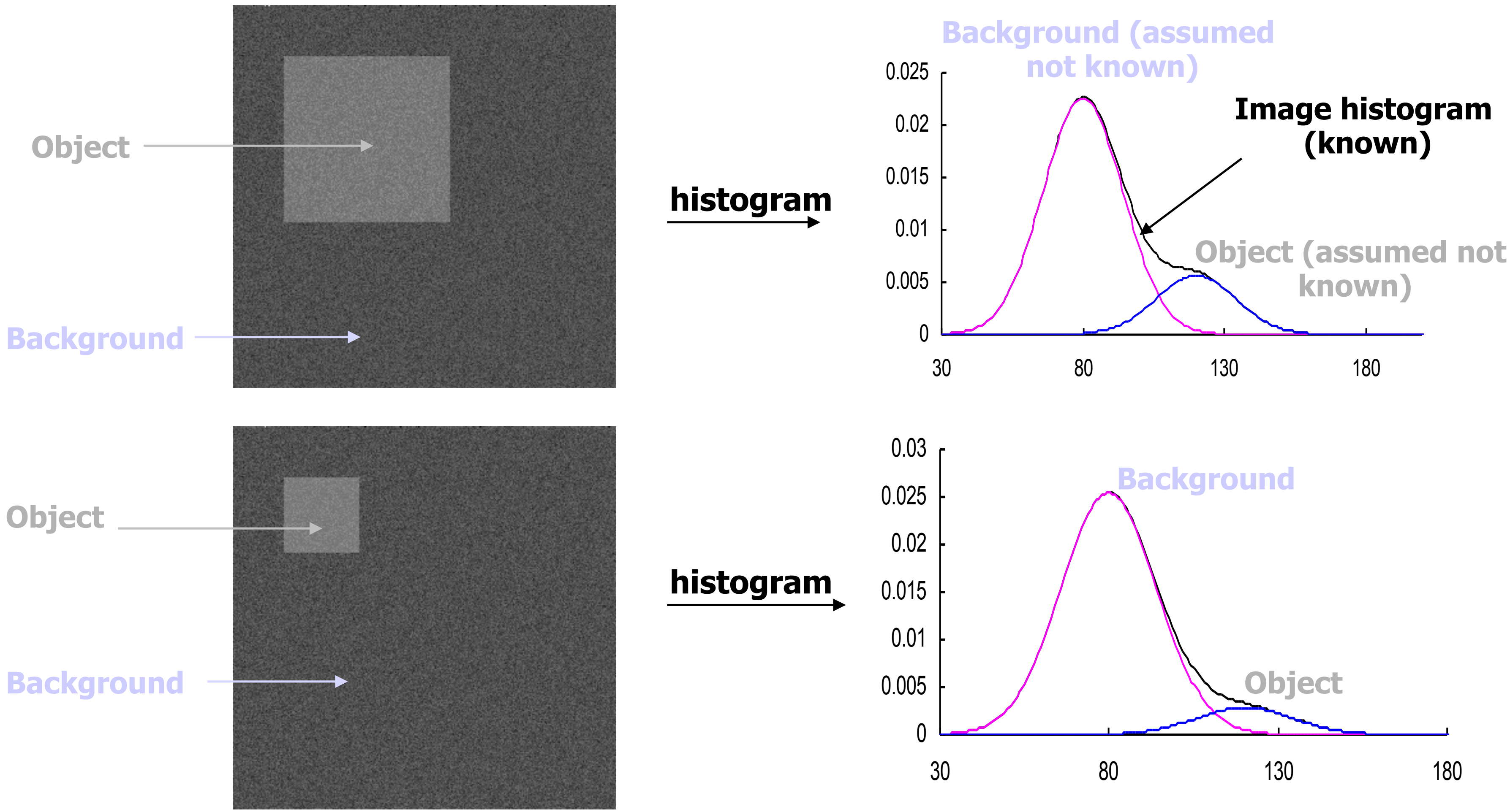
  - Choosing a threshold in the valley between two overlapping peaks, and inevitably some pixels will be incorrectly classified by the thresholding.



The determination of peaks and valleys is a non-trivial problem

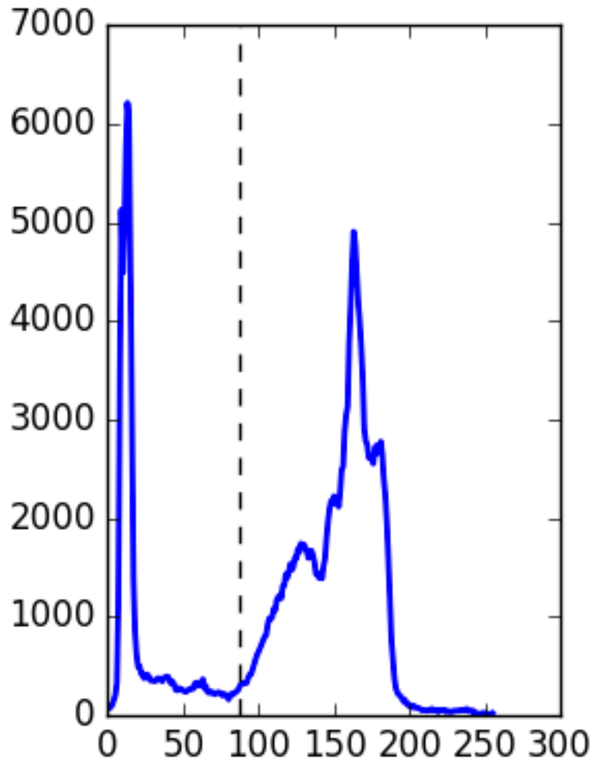
10.3 Thresholding with Otsu's Method

- Sensitivity to object size



10.3 Thresholding with Otsu's Method

- **Otsu Thresholding**:
  - Select a threshold that minimize the within-class variance



10.3 Thresholding with Otsu’s Method

- **Otsu Thresholding**:

- define the *within-class* variance as the weighted sum of the variances of each cluster:

$$\boxed{\sigma^2_{\text{Within}}(T) = n_B(T)\sigma^2_B(T) + n_O(T)\sigma^2_O(T)}$$

Minimize

$$
\begin{aligned}
n_B(T) &= \sum_{i=0}^{T-1} p(i) \\
n_O(T) &= \sum_{i=T}^{N-1} p(i) \\
\sigma^2_B(T) &= \text{the variance of the pixels in the background (below threshold)} \\
\sigma^2_O(T) &= \text{the variance of the pixels in the foreground (above threshold)}
\end{aligned}
$$

10.3 Thresholding with Otsu's Method

- **Otsu Thresholding**:

Minimize

$$\begin{aligned}\sigma^2_{\text{Between}}(T) &= \sigma^2 - \sigma^2_{\text{Within}}(T) \\ &= n_B(T)\left[\mu_B(T) - \mu\right]^2 + n_O(T)\left[\mu_O(T) - \mu\right]^2\end{aligned}$$

$$\boxed{\mu = n_B(T)\mu_B(T) + n_O(T)\mu_O(T)}$$

$$\Rightarrow \sigma^2_{\text{Between}}(T) = n_B(T)n_O(T)[\mu_B(T) - \mu_O(T)]^2$$

Maximize

10.3 Thresholding with Otsu's Method

- Algorithm of **Otsu Thresholding**:

Goal: select a *T* that maximizes $n_B(T)n_O(T)[\mu_B(T) - \mu_O(T)]^2$

- For each potential threshold *T* ,
  1. Separate the pixels into two clusters according to the threshold.
  2. Find the mean of each cluster.
  3. Square the difference between the means.
  4. Multiply by the number of pixels in one cluster times the number in the other.

10.3 Thresholding with Otsu's Method

- Algorithm of **Otsu Thresholding**:
  - The computations aren't independent as we change from one threshold to another.
  - We can incrementally update with the following equations:

$$n_B(T+1) = n_B(T) + n_T$$

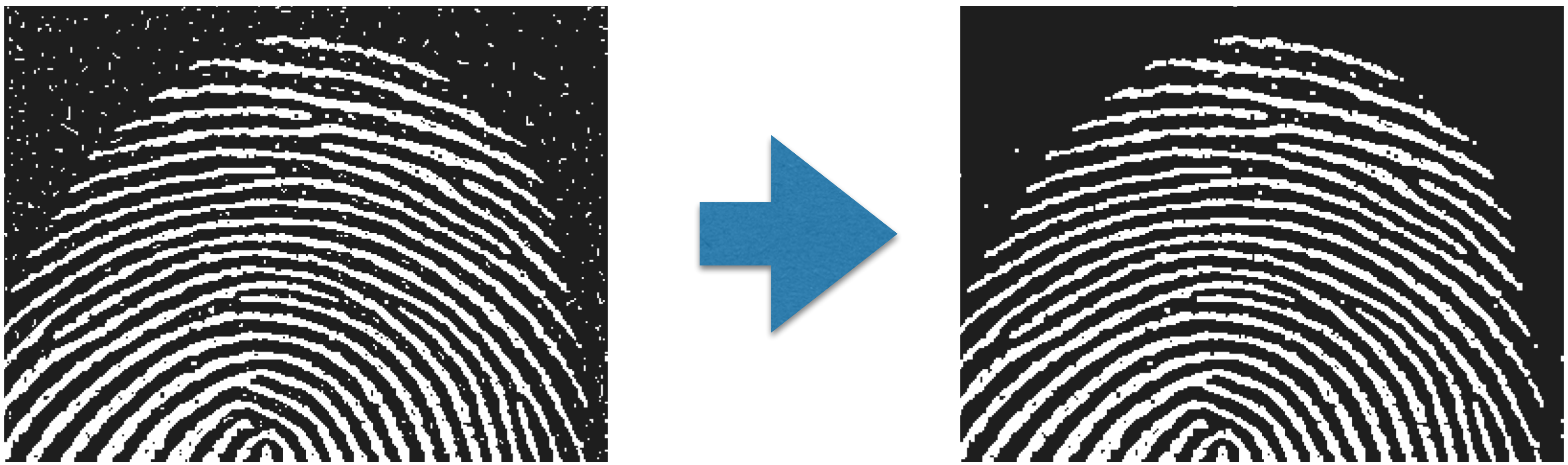
$$n_O(T+1) = n_O(T) - n_T$$

$$\mu_B(T+1) = \frac{\mu_B(T)n_B(T) + n_T T}{n_B(T+1)}$$

$$\mu_O(T+1) = \frac{\mu_O(T)n_O(T) - n_T T}{n_O(T+1)}$$

9 Morphological Image Processing

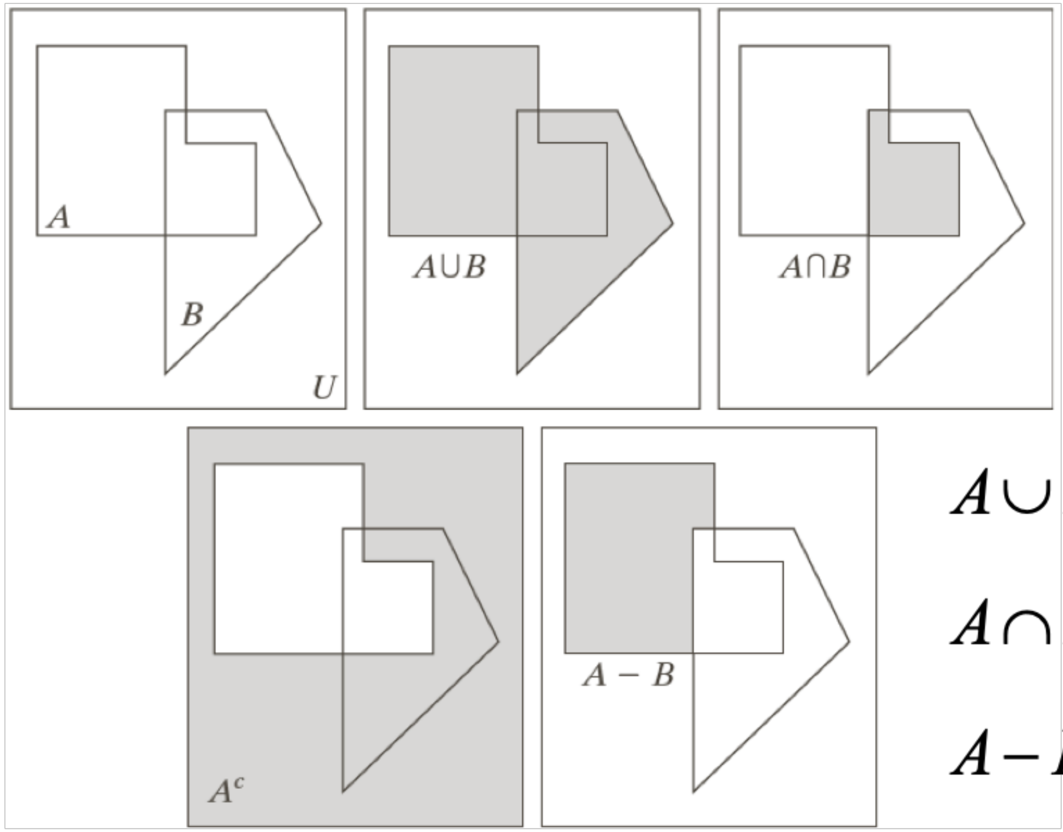
- Mathematical morphology provides tools for the representation and description of image regions (e.g. boundary extraction, skeleton, convex hull).
- It provides techniques for pre- and post-processing of an image (morphological thinning, pruning, filtering).
- Its principles are based on set theory.
- Applications to both binary and gray-scale images.



credit of this slide: C. Nikou

9 Morphological Image Processing

- Preliminaries - Basic set operations



$$A \cup B = \{w \mid w \in A \text{ OR } w \in B\}$$

$$A \cap B = \{w \mid w \in A \text{ AND } w \in B\}$$

$$A - B = \{w \mid w \in A,\ w \notin B\} = A \cap B^c$$

$$A^c = \{w \mid w \notin A\}$$

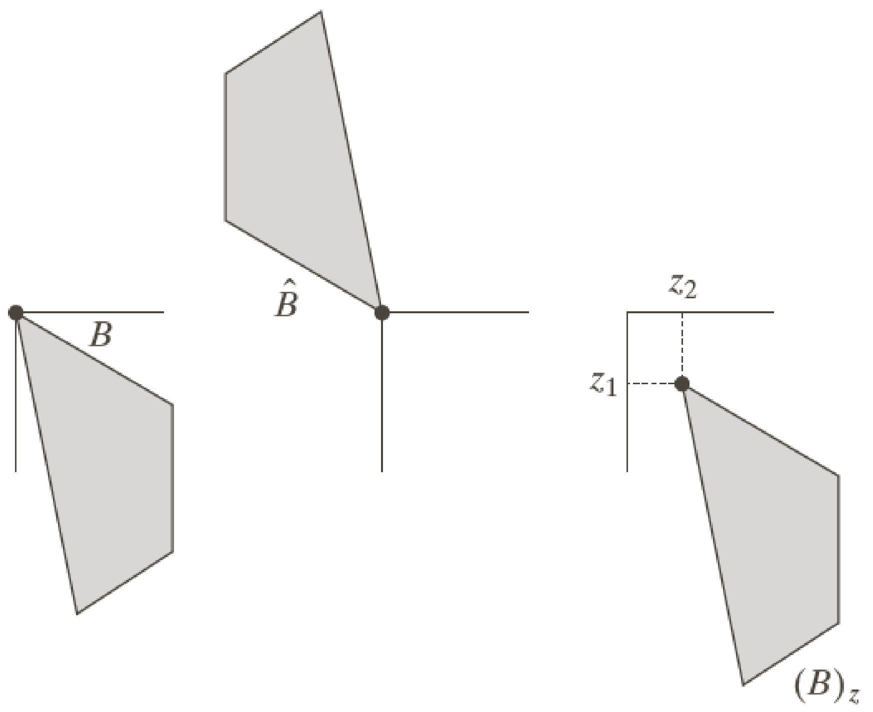
credit of this slide: C. Nikou

9 Morphological Image Processing

- Preliminaries - Basic set operations

Set reflection: $\hat{B} = \{w \mid w = -b, \text{ for } b \in B\}$

Set translation by z: $(B)_z = \{c \mid c = b + z, \text{ for } b \in B\}$



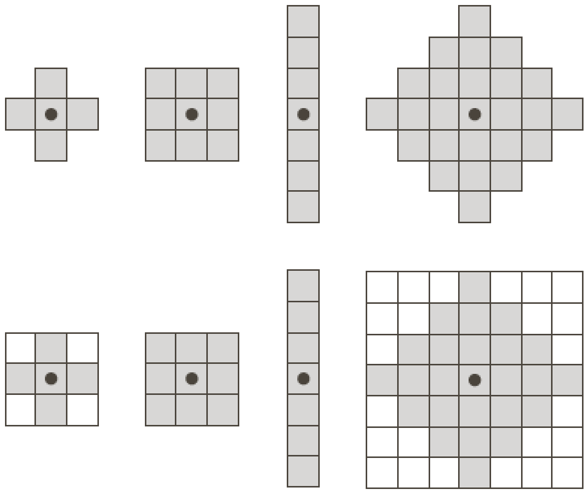
credit of this slide: C. Nikou

9 Morphological Image Processing

- In image processing, we use morphology with two types of sets of pixels: *objects* and *structuring elements* (SE's).

- SE are small sets or subimages used to examine the image under study for properties of interest. The origin must be specified. Zeros are appended to SE to give them a rectangular form.

Example of *objects*



Example of *SE*

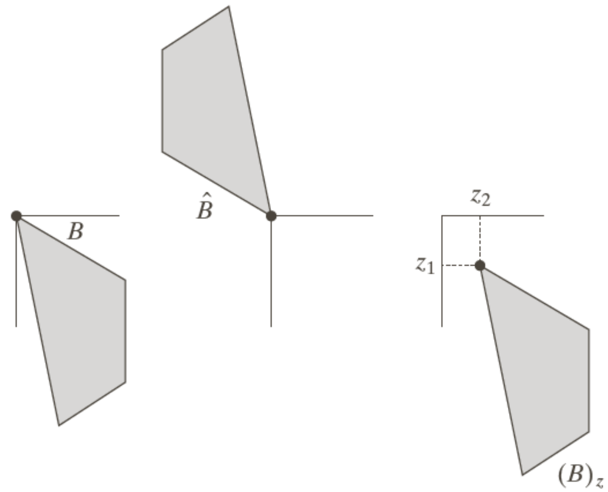
credit of this slide: C. Nikou

9 Morphological Image Processing

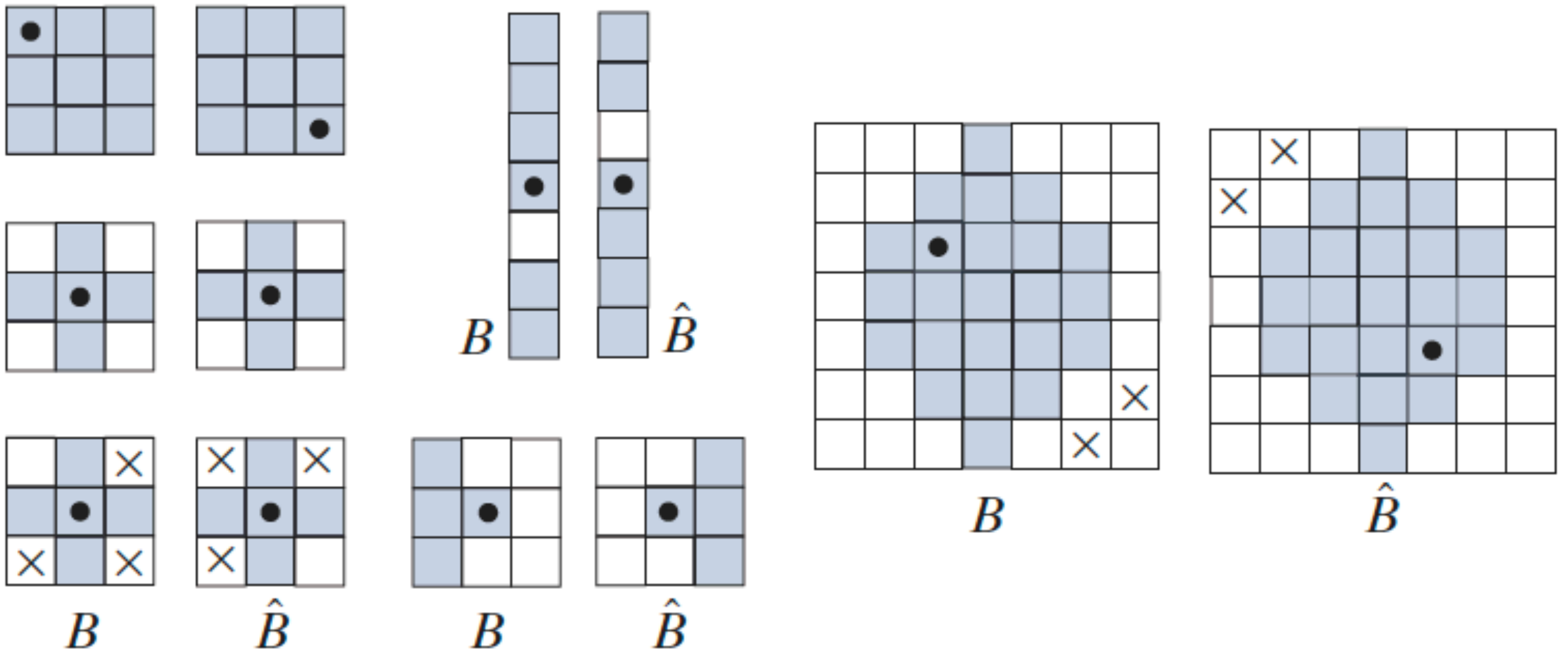
- Set reflection and translation are employed to SE.

Set reflection: $\hat{B} = \{w \mid w = -b, \text{ for } b \in B\}$

Set translation by z**:** $(B)_z = \{c \mid c = b + z, \text{ for } b \in B\}$



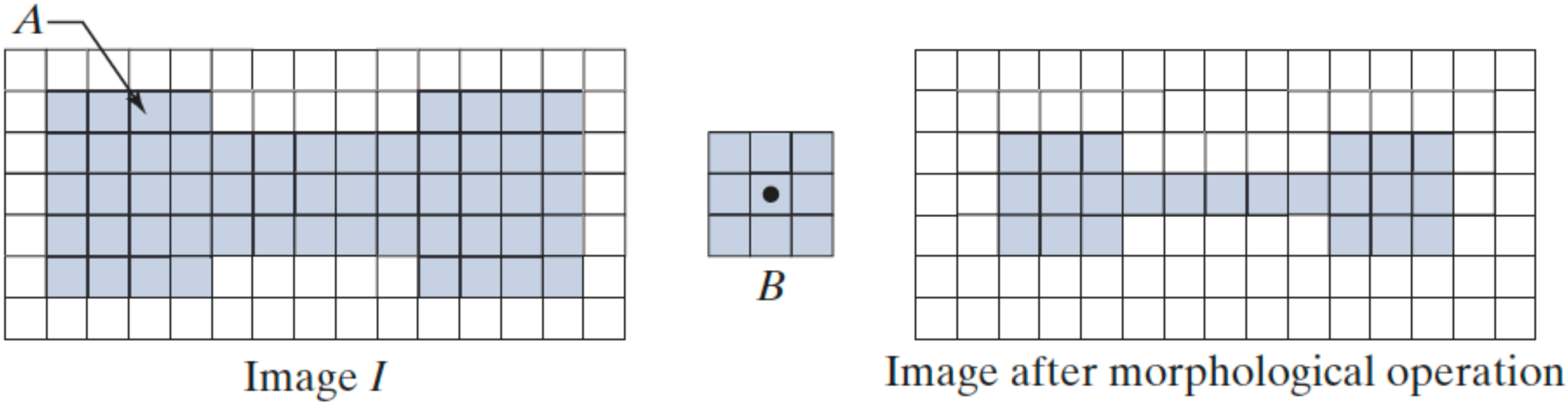
Structuring elements and their reflections about the origin

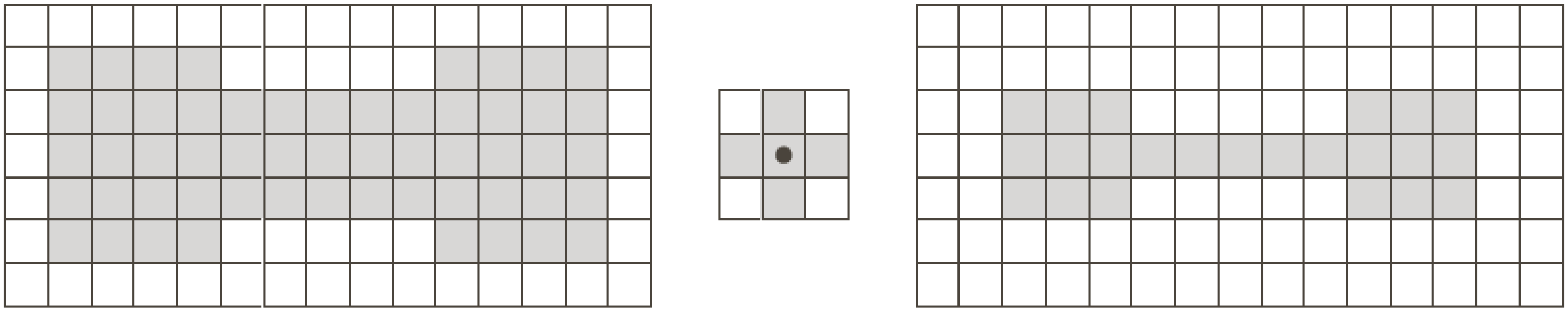


credit of this slide: C. Nikou

9 Morphological Image Processing

- Example of image after morphological operation - erosion



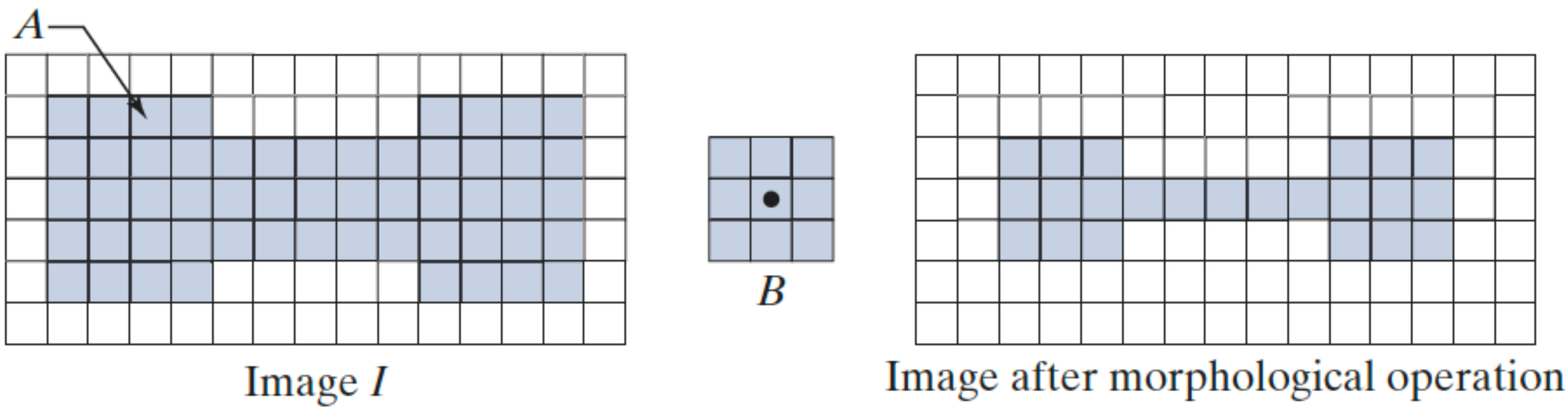


9 Morphological Image Processing

- Erosion

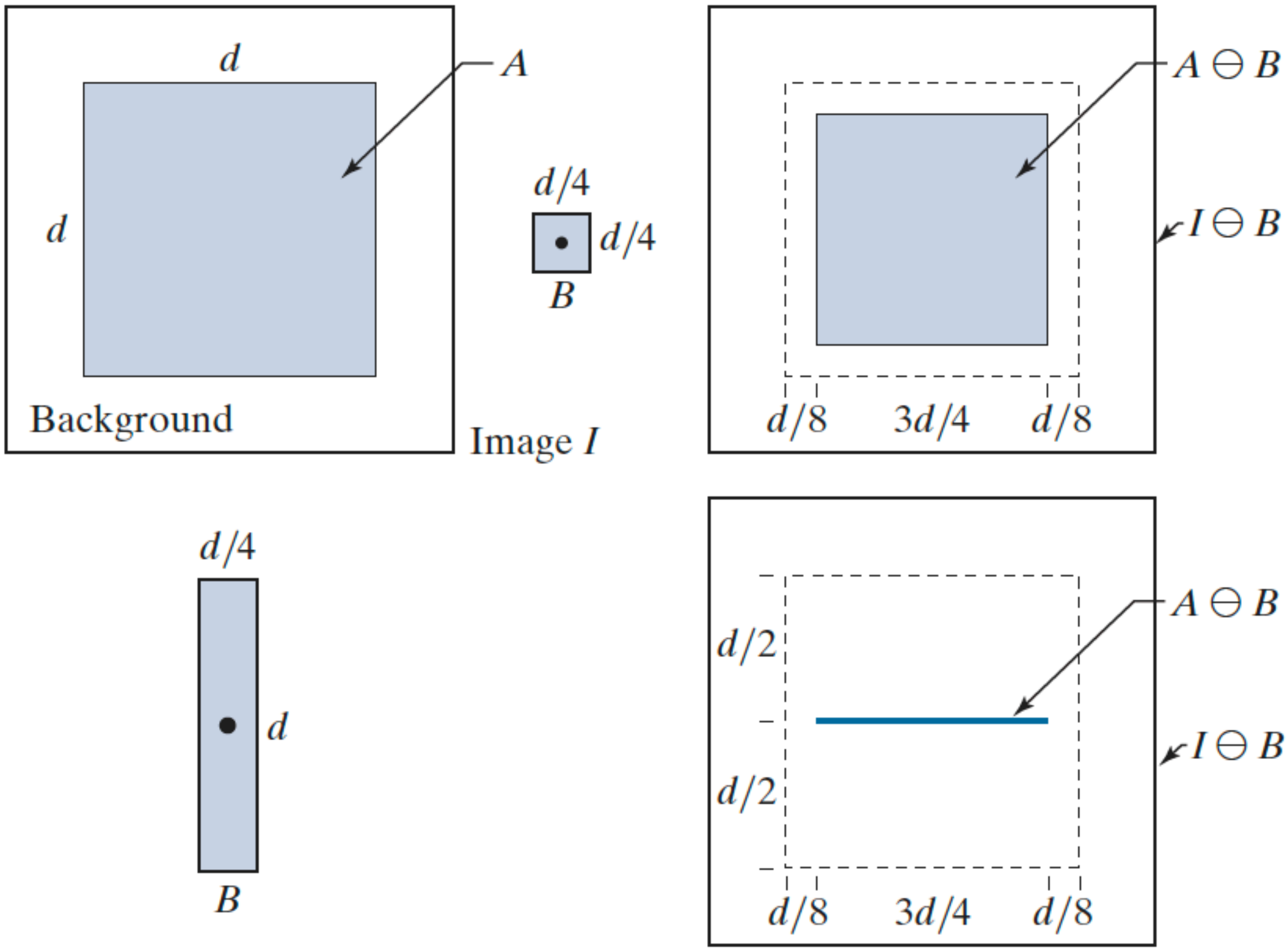
$$A \ominus B = \left\{ z \middle| (B)_z \subseteq A \right\}$$

where A is a set of foreground pixels, B is a structuring element, and the z’s are foreground values (1’s).



9 Morphological Image Processing

- Erosion $A \ominus B = \{ z \mid (B)_z \subseteq A \}$

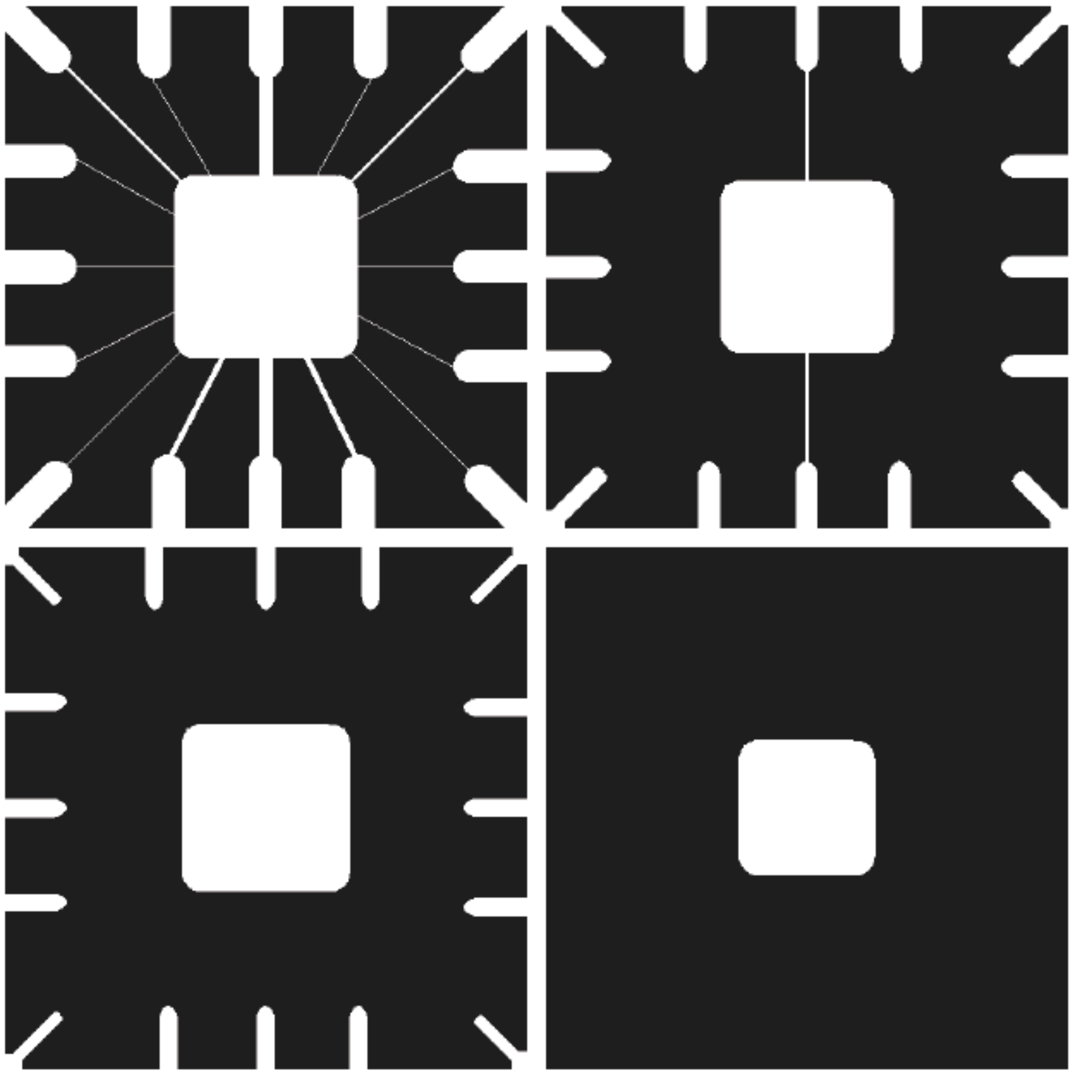


9 Morphological Image Processing

- Erosion by a square SE of varying size

Original image

11x11

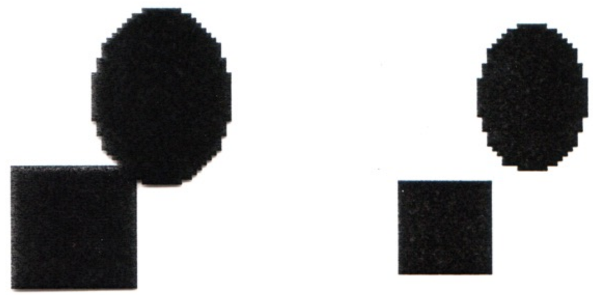


15x15

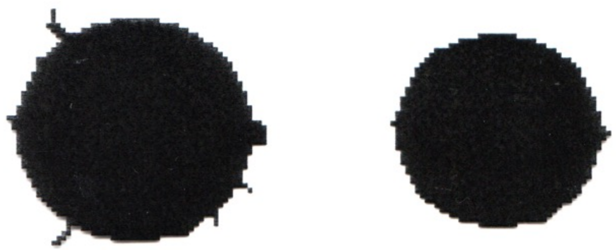
45x45

9 Morphological Image Processing

- Erosion
  - Erosion can split apart joined objects



  - Erosion can strip away extrusions



  - **Watch out**: Erosion shrinks objects

credit of this slide: C. Nikou

9 Morphological Image Processing

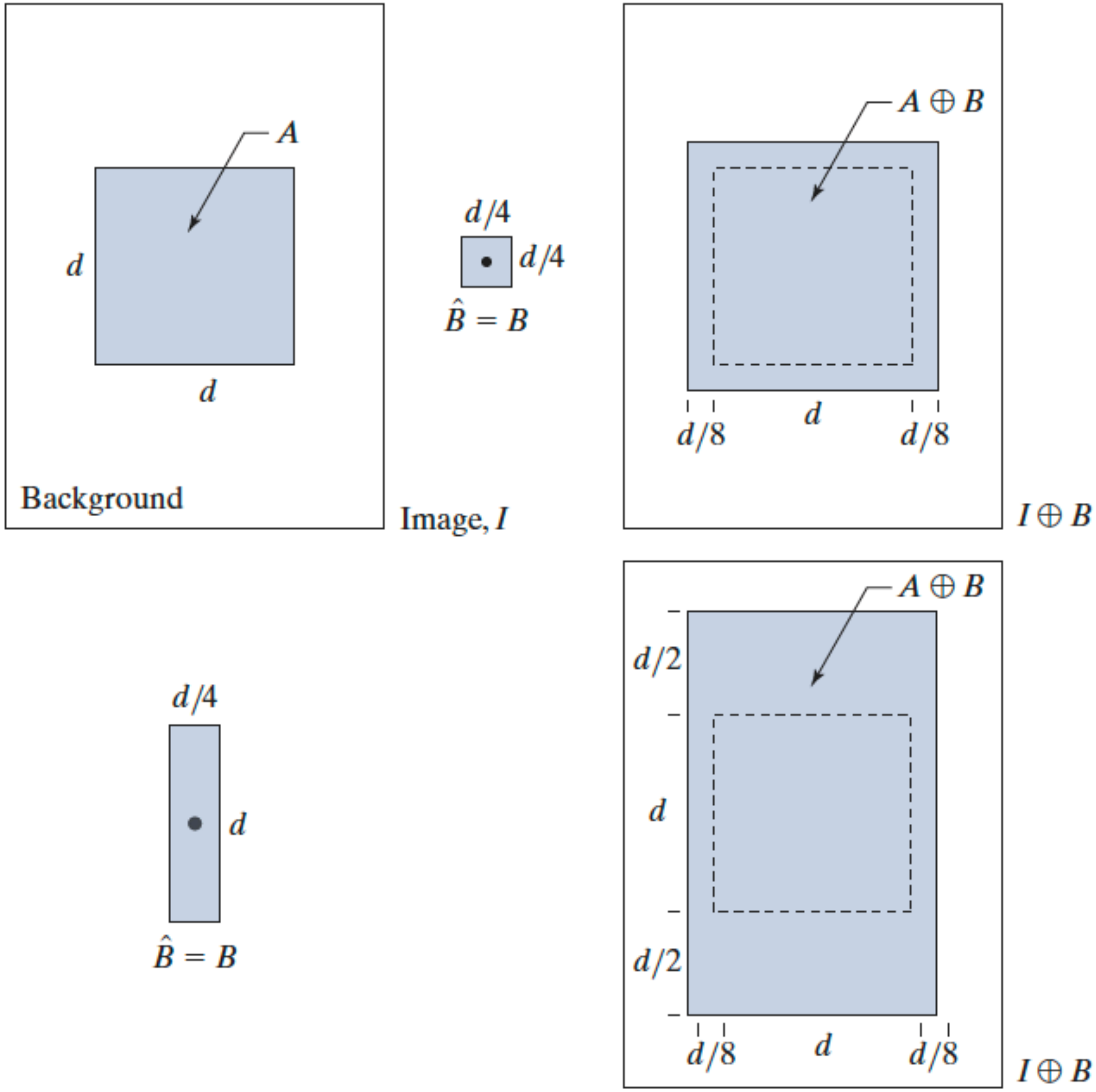
- Dilation

$$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \varnothing\}$$

The result is the set of all points z such that the reflected B translated overlap with A at at least one element.

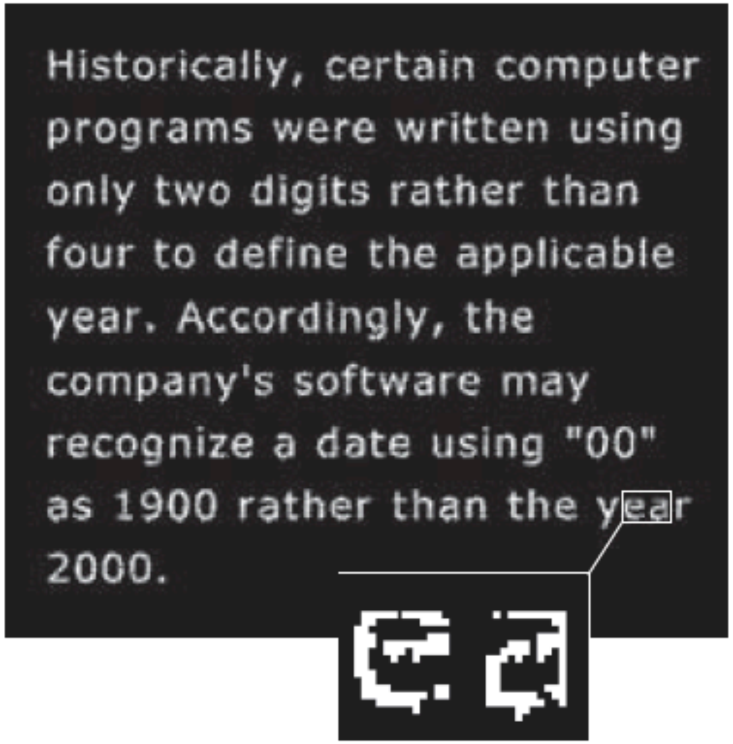
9 Morphological Image Processing

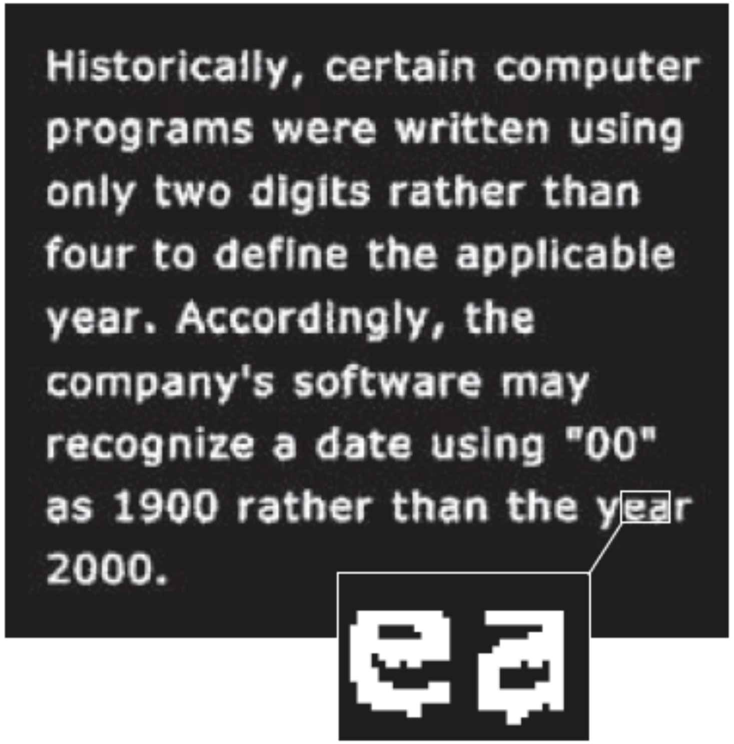
- Dilation $A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \varnothing\}$



9 Morphological Image Processing

- Dilation





1	1	1
1	1	1
1	1	1

- Dilation bridges gaps.
- Contrary to low pass filtering it produces a binary image.

credit of this slide: C. Nikou

9 Morphological Image Processing

- Dilation
  - Dilation can repair breaks

  - Dilation can repair intrusions

  - **Watch out**: Dilation enlarges objects

credit of this slide: C. Nikou

9 Morphological Image Processing

- Duality
  - Erosion and dilation are dual operations with respect to set complementation and reflection:

$$(A \ominus B)^c = A^c \oplus \hat{B}$$

$$(A \oplus B)^c = A^c \ominus \hat{B}$$

  - The duality is useful when the SE is symmetric: The erosion of an image is the dilation of its background.

credit of this slide: C. Nikou

9 Morphological Image Processing

- Compound Operations - Opening and Closing
  - Opening

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

  - Closing

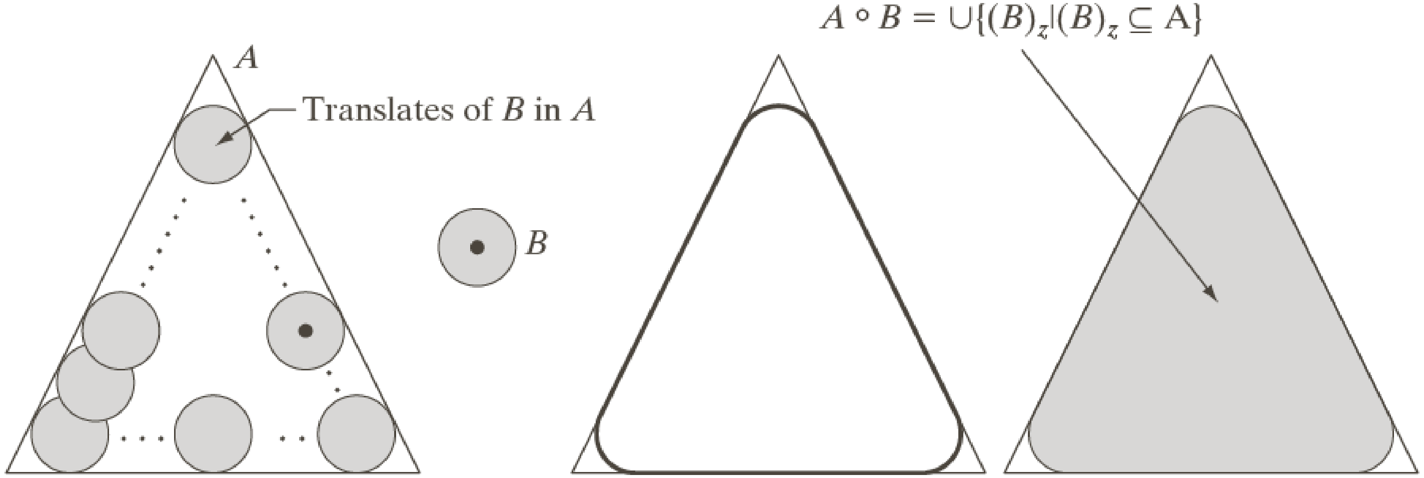
$$A \bullet B = (A \oplus B) \ominus B$$

credit of this slide: C. Nikou

9 Morphological Image Processing

- Opening $$A \circ B = (A \ominus B) \oplus B$$
  - Geometric interpretation: The boundary of the opening is defined by points of the SE that reach the farthest into the boundary of A as B is “rolled” inside of this boundary

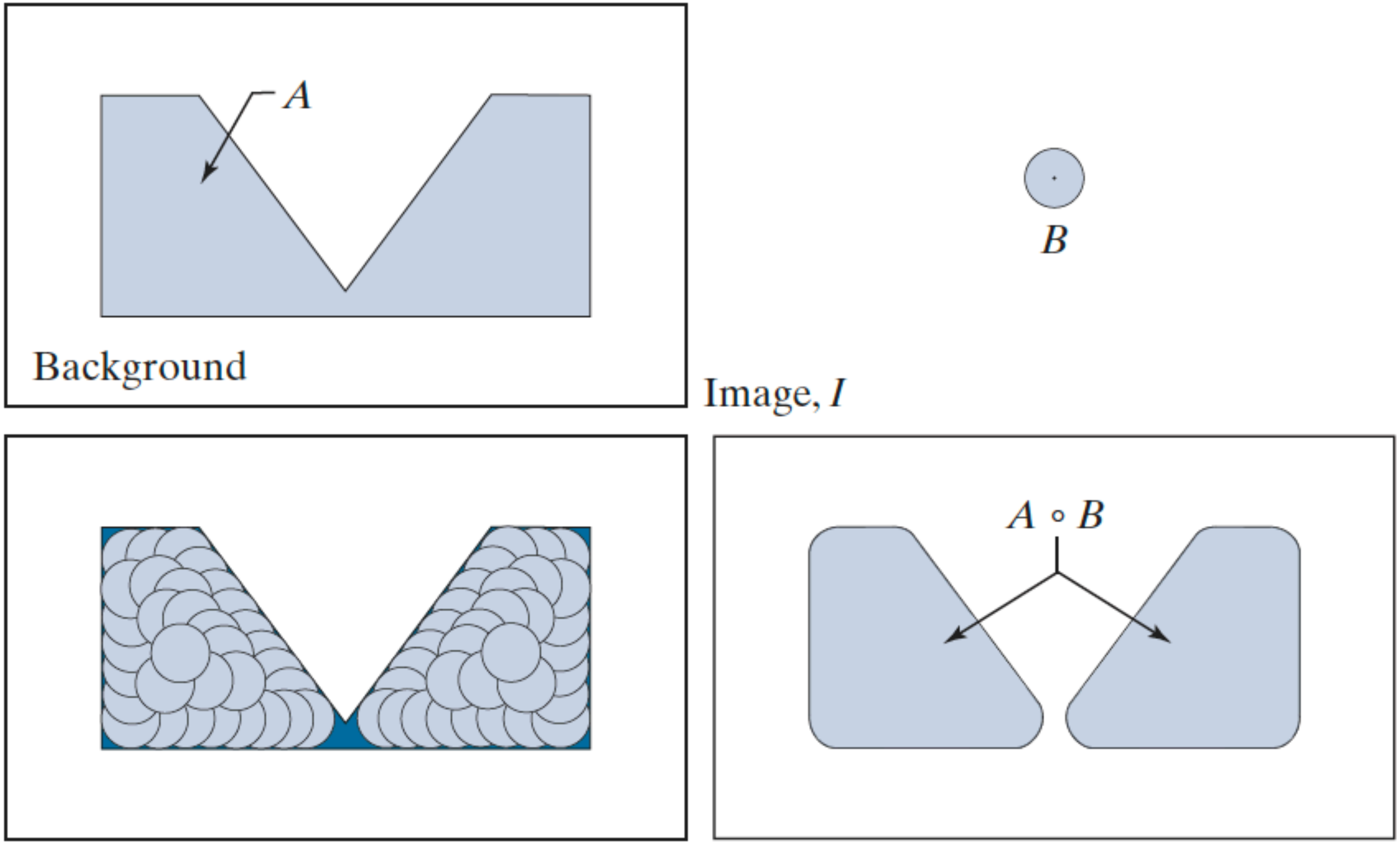
Equivalently: $$A \circ B = \bigcup \{(B)_z \mid (B)_z \subseteq A\}$$



credit of this slide: C. Nikou

9 Morphological Image Processing

- Opening $A \circ B = \bigcup \{(B)_z \mid (B)_z \subseteq A\}$

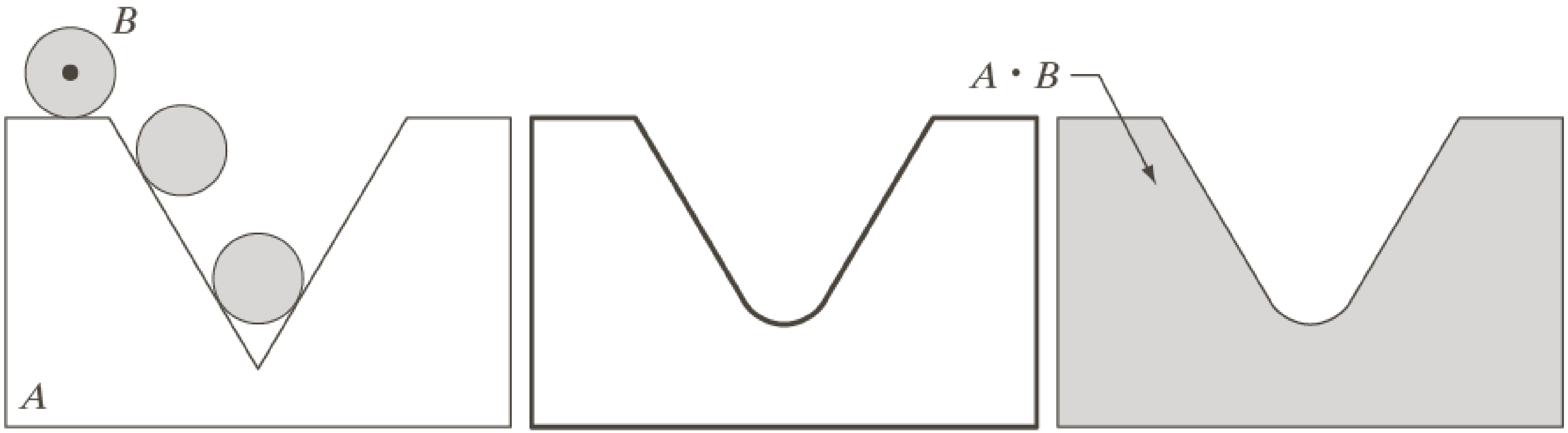


credit of this slide: C. Nikou

9 Morphological Image Processing

- Closing $A \bullet B = (A \oplus B) \ominus B$
  - Geometric interpretation: It has a similar geometric interpretation except that B is rolled on the outside of the boundary.

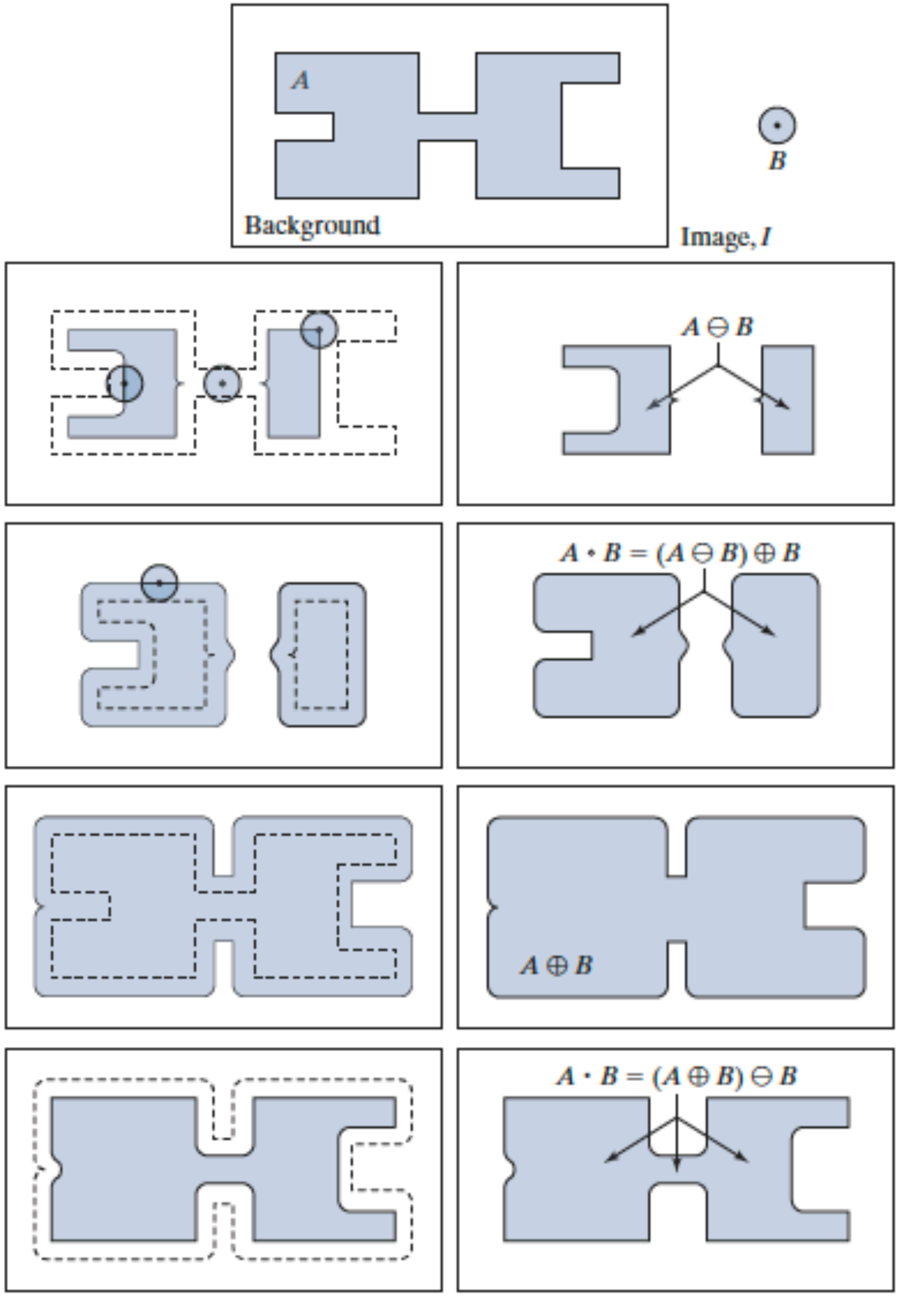
Equivalently: $A \bullet B = \left[\bigcup \left\{ (B)_z \middle| (B)_z \cap A = \varnothing \right\} \right]^c$



credit of this slide: C. Nikou

9 Morphological Image Processing

- **Opening and Closing**



Erosion: elements where the disk can not fit are eliminated

Opening: outward corners are rounded

Dilation: inward intrusions are reduced in depth

Closing: inward corners are rounded

credit of this slide: C. Nikou

9 Morphological Image Processing

- Duality: Opening and closing are dual operations

Erosion-Dilation duality

$$(A \ominus B)^c = A^c \oplus \hat{B}$$

$$(A \oplus B)^c = A^c \ominus \hat{B}$$

Opening-Closing duality

$$(A \bullet B)^c = A^c \circ \hat{B}$$

$$(A \circ B)^c = A^c \bullet \hat{B}$$

credit of this slide: C. Nikou

9 Morphological Image Processing

- Property of Opening and Closing

Opening:

$$A \circ B \subseteq A$$

$$C \subseteq D \Rightarrow C \circ B \subseteq D \circ B$$

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

Closing:

$$A \subseteq A \bullet B$$

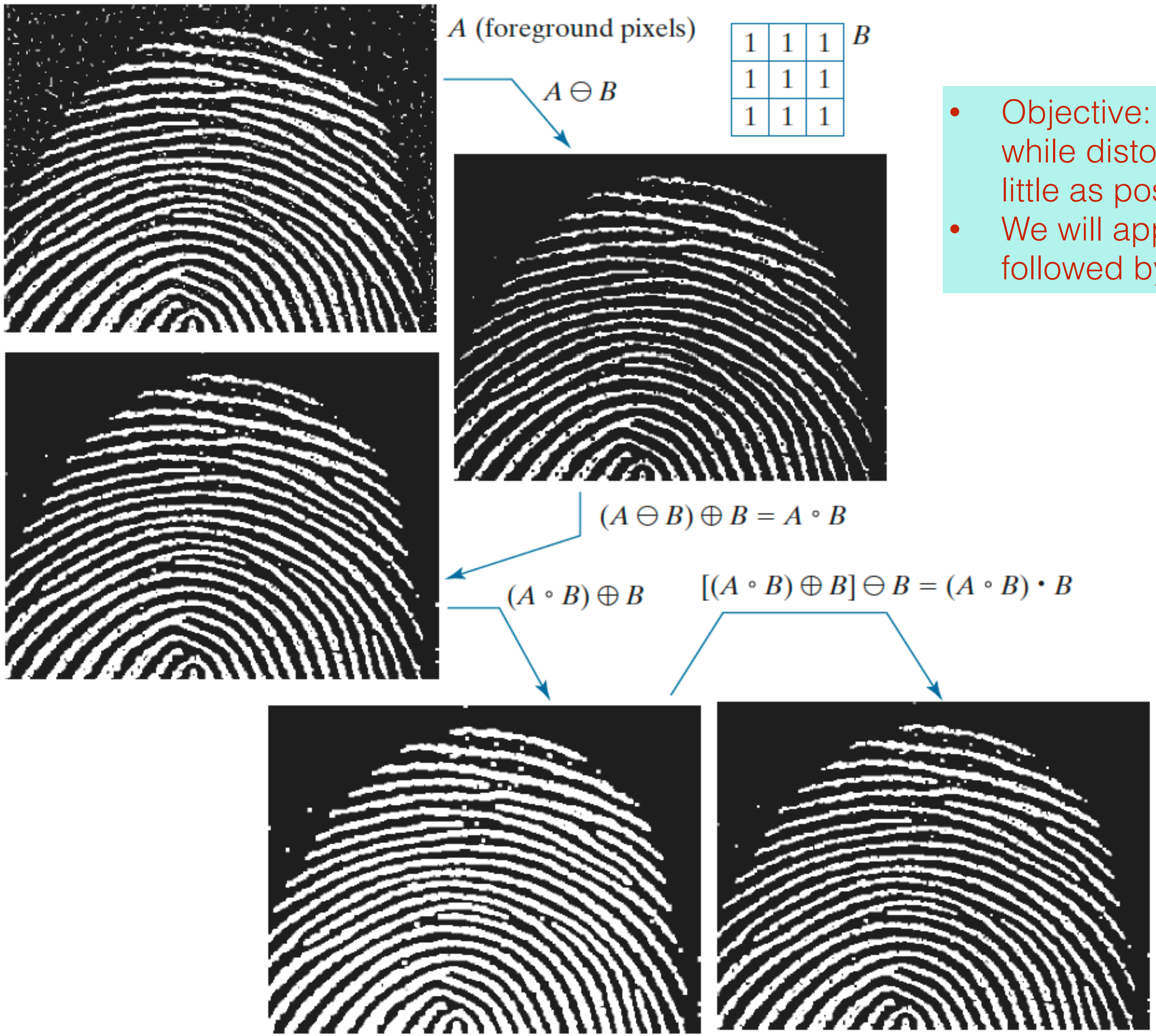
$$C \subseteq D \Rightarrow C \bullet B \subseteq D \bullet B$$

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

The last properties, in each case, indicate that multiple openings or closings have no effect after the first application of the operator

credit of this slide: C. Nikou

- Example of morphological filtering



- Objective: Eliminate noise while distorting the image as little as possible.
- We will apply an opening followed by closing.

10.2 Line Detection with Hough Transform

- Review - Gradient magnitude for edge detection (Sec. 3.6)

$$M(x, y) = \|\nabla f\| = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

−1	0
0	1

0	−1
1	0

Sobel Operators

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

10.2 Line Detection with Hough Transform

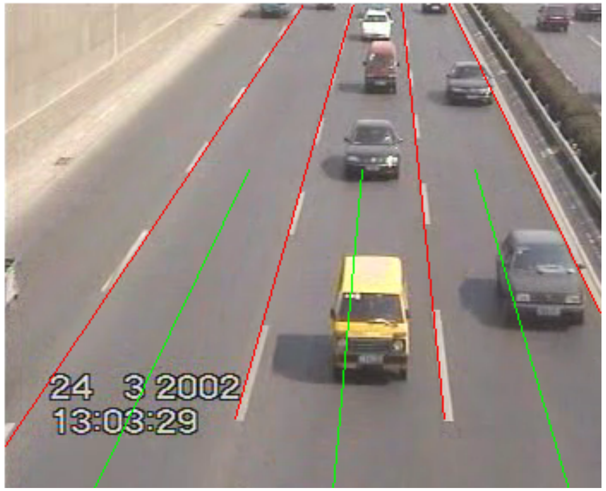
- Review - Gradient magnitude for edge detection (Sec. 3.6)

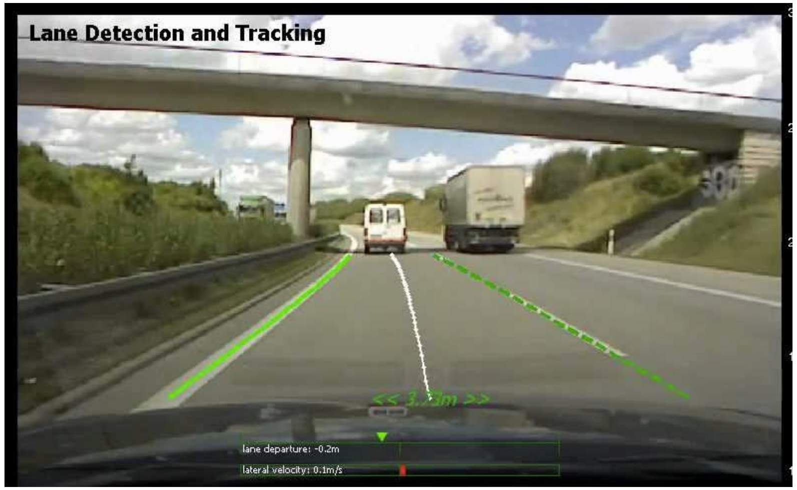
Thresholded gradient of the smoothed image

How to estimate lines from these edge points?

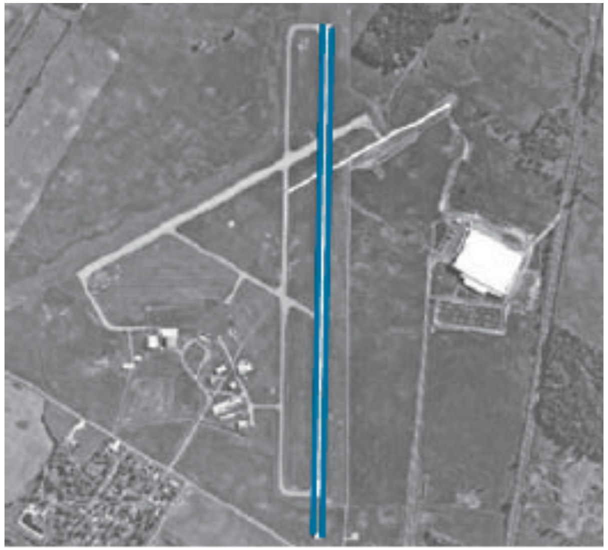
10.2 Line Detection with Hough Transform

- Other applications - lane detection





- Other applications - extracting the principal runway

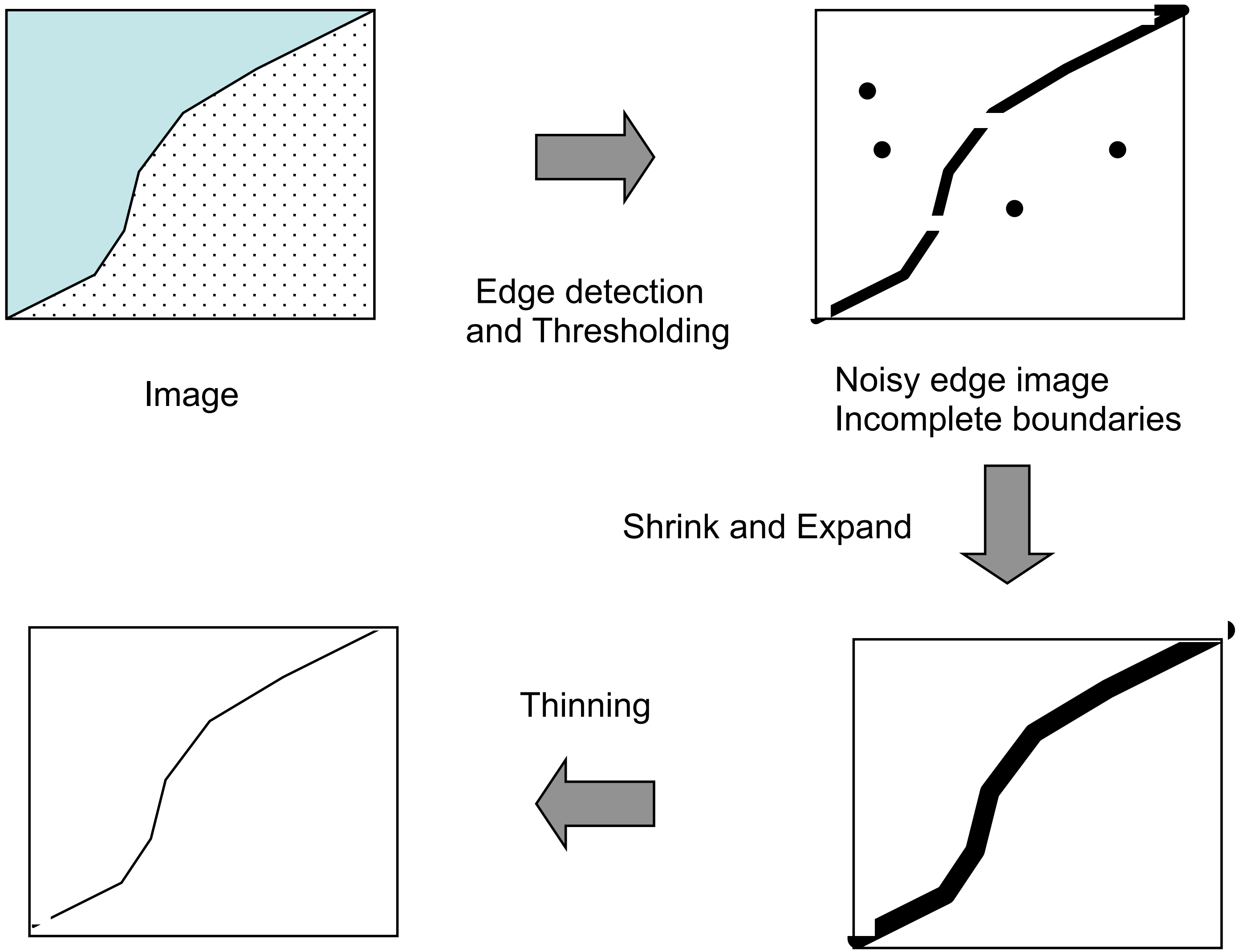


10.2 Line Detection with Hough Transform

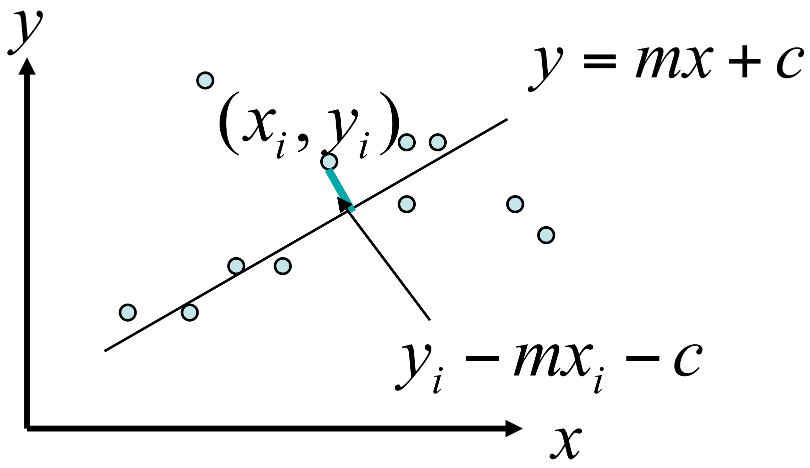
- **Hough Transform**

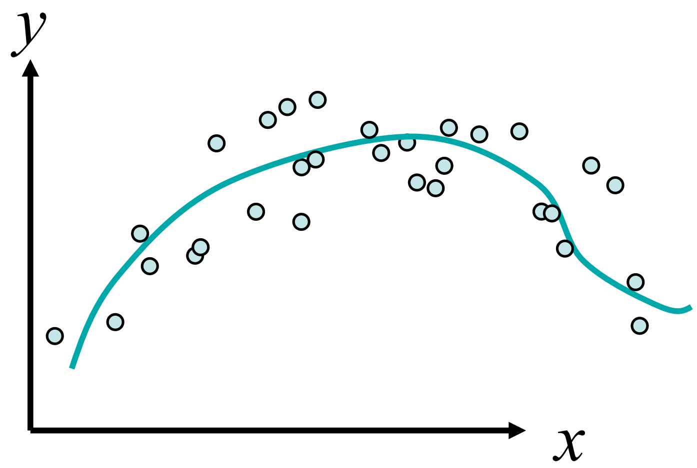
- Performed after Edge Detection
- It is a technique to isolate the curves of a given shape / shapes in a given image
- Classical Hough Transform can locate regular curves like straight lines, circles, parabolas, ellipses, etc.
    - Requires that the curve be specified in some parametric form

Preprocessing Edge Images



Line Fitting or Curve Fitting





Minimize:

$$E = \sum_i \frac{(y_i - mx_i - c)^2}{N}$$

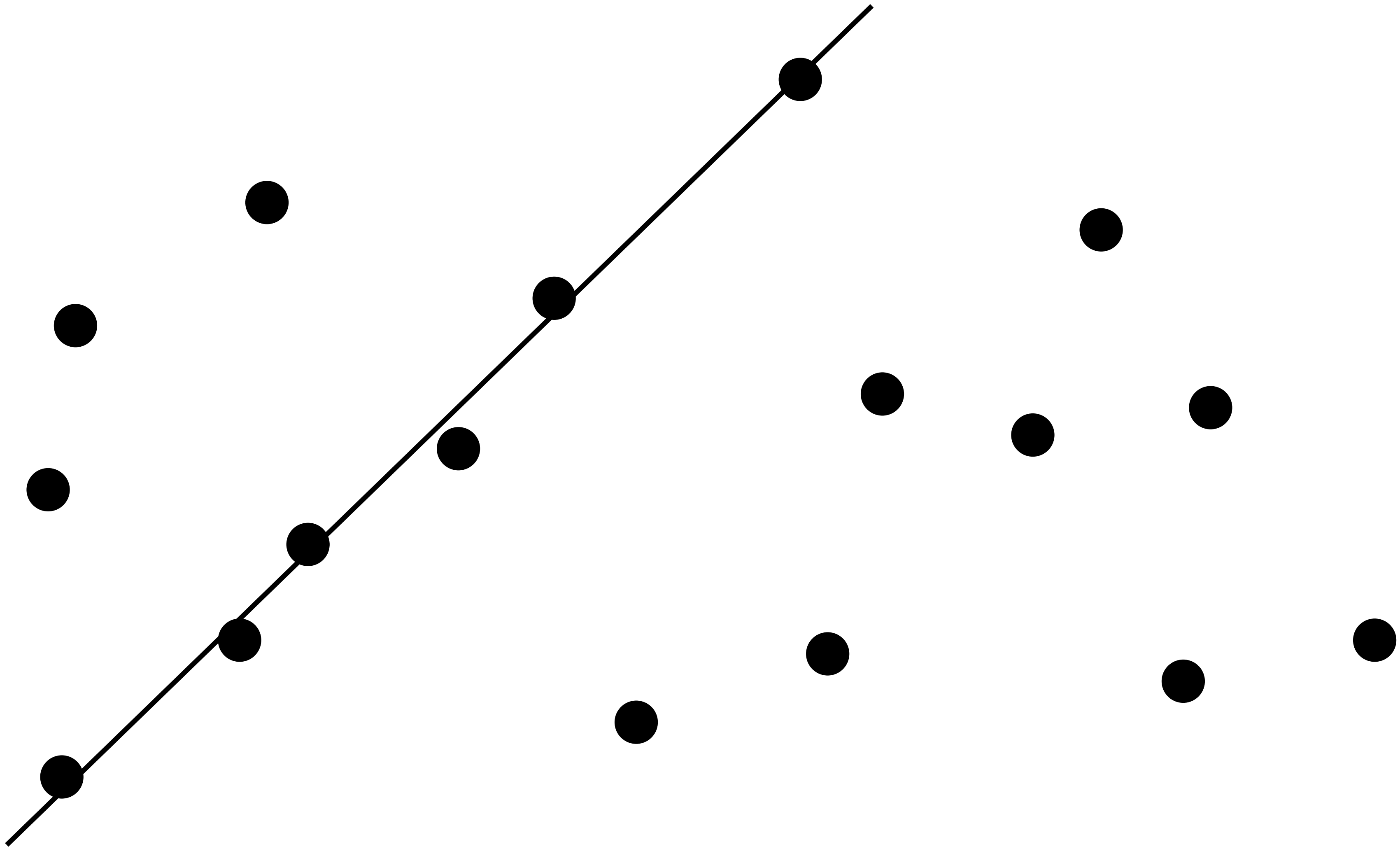
Find

$$y = f(x) = ax^3 + bx^2 + cx + d$$

Minimize:

$$\frac{1}{N} \sum_i [y_i - (ax_i^{\ 3} + bx_i^{\ 2} + cx_i + d)]^2$$

Line Grouping Problem



Slide credit: David Jacobs

This is difficult because of

- Extraneous data: clutter or multiple models
  - We do not know what is part of the model?
  - Can we pull out models with a few parts from much larger amounts of background clutter?
- Missing data: only some parts of model are present
- Noise
- Cost:
  - It is not feasible to check all combinations of features by fitting a model to each possible subset

Hough Transform

- Elegant method for direct object recognition
  - Edges need not be connected
  - Complete object need not be visible
  - **Key Idea**: Edges VOTE for the possible model

Image and Parameter Spaces

Equation of Line: $y = mx + c$

Find: (m, c)

Consider point: (x_i, y_i)

$$y_i = mx_i + c \quad or \quad c = -x_i m + y_i$$

Parameter space also called Hough Space

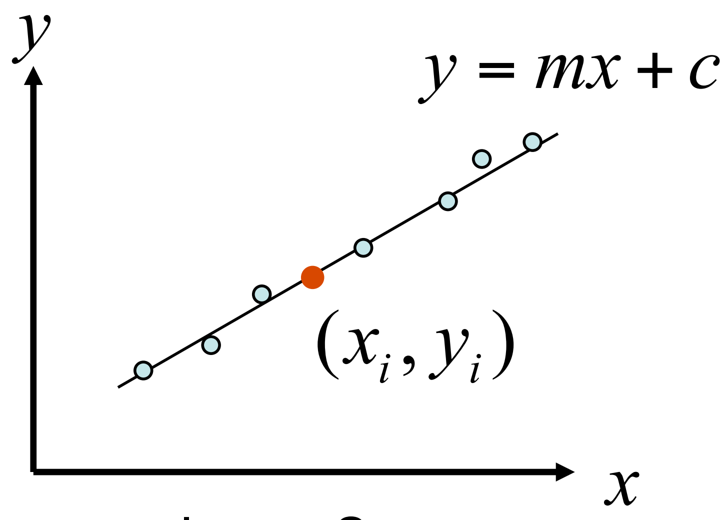
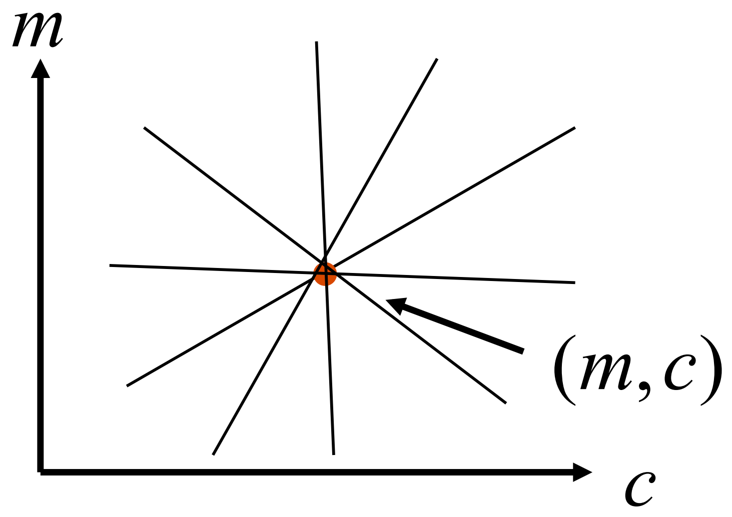


Image Space



Parameter Space

Line Detection by Hough Transform

Algorithm:

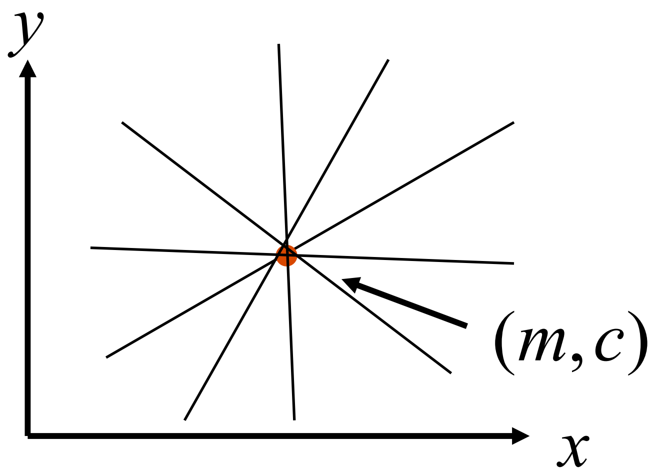
- Quantize Parameter Space (m, c)
- Create Accumulator Array $A(m, c)$
- Set $A(m, c) = 0 \quad \forall m, c$
- For each image edge (x_i, y_i) increment:

$$A(m, c) = A(m, c) + 1$$

- If (m, c) lies on the line:

$$c = -x_i m + y_i$$

- Find local maxima in $A(m, c)$



Parameter Space

$A(m, c)$

	1						1		
		1				1			
			1		1				
				2					
			1		1				
		1				1			
	1						1		

Better Parameterization

NOTE: $-\infty \leq m \leq \infty$

Large Accumulator

More memory and computations

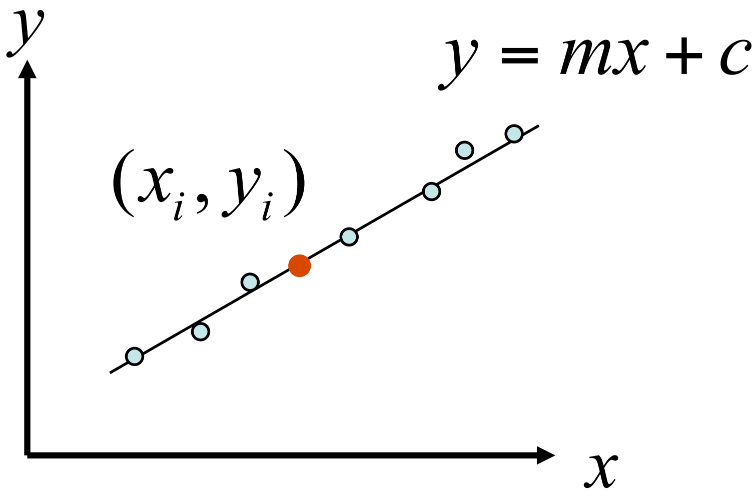


Image Space

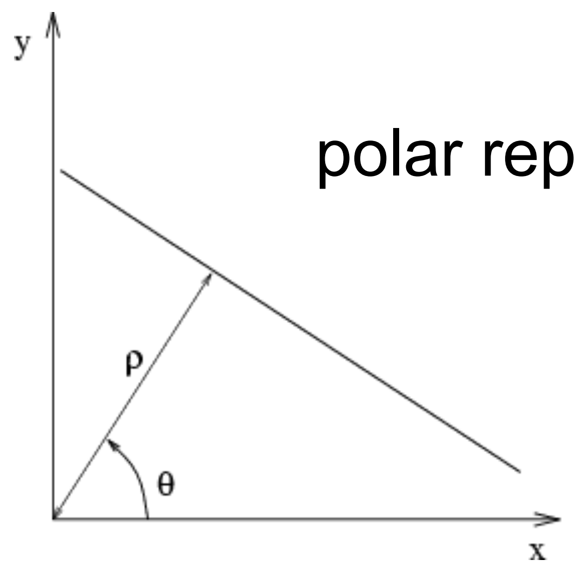
Improvement: (Finite Accumulator Array Size)

Line equation: $\rho = -x\cos\theta + y\sin\theta$

Here $0 \leq \theta \leq 2\pi$

$0 \leq \rho \leq \rho_{\max}$

Given points (x_i, y_i) find (ρ, θ)



polar representation

Better Parameterization

NOTE: $-\infty \leq m \leq \infty$

Large Accumulator

More memory and computations

Improvement: (Finite Accumulator Array Size)

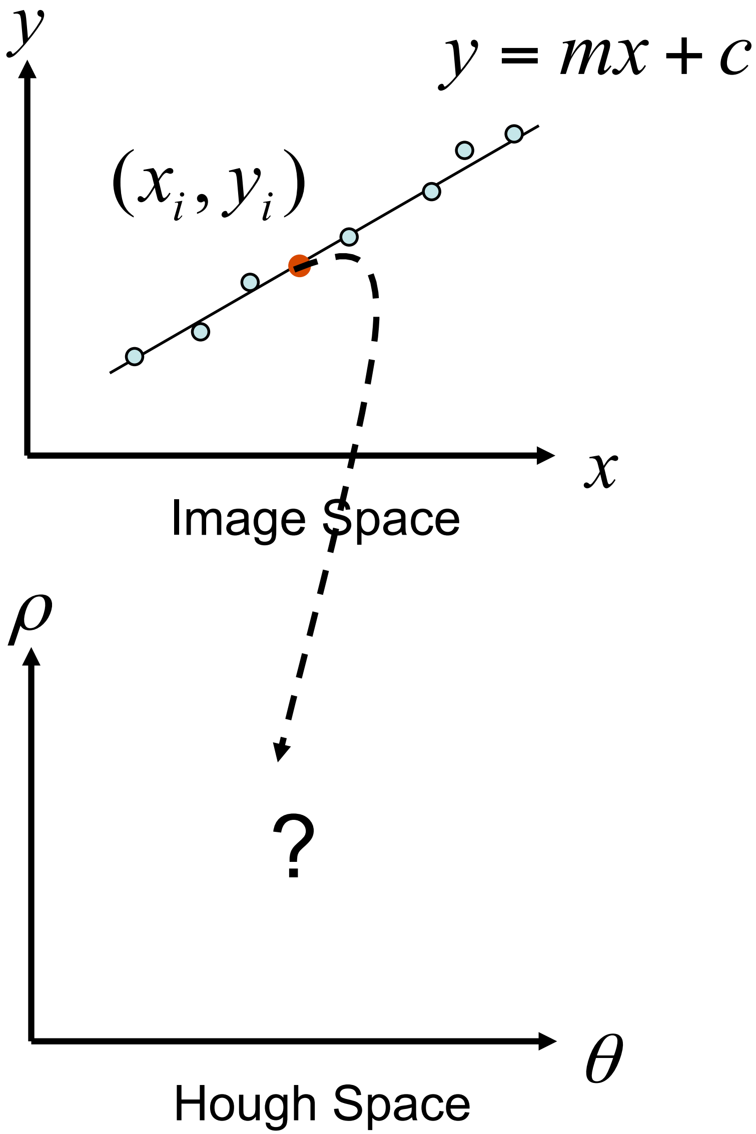
Line equation: $\rho = -x\cos\theta + y\sin\theta$

Here $0 \leq \theta \leq 2\pi$

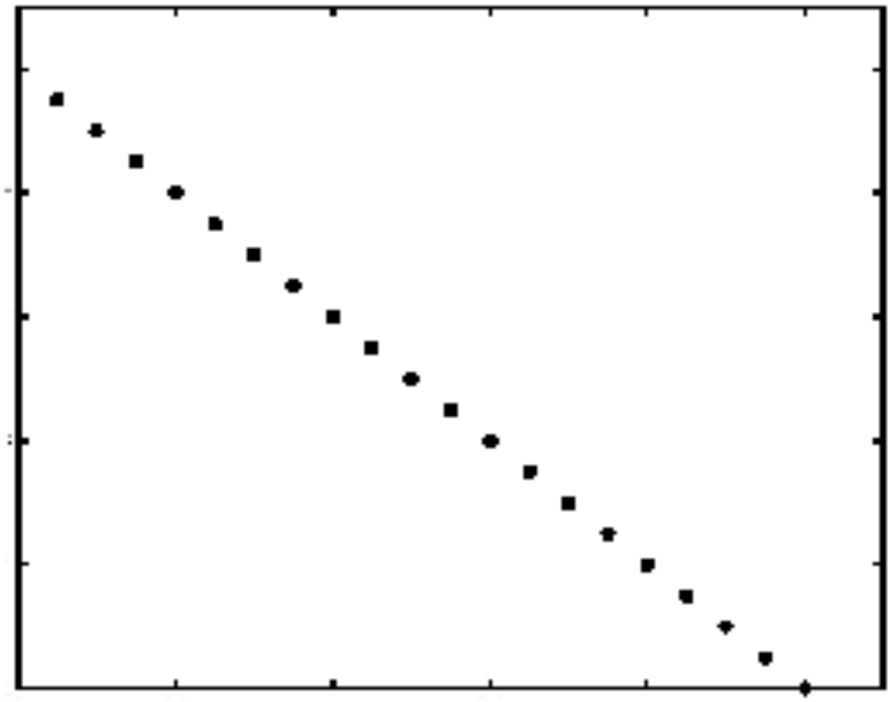
$0 \leq \rho \leq \rho_{\max}$

Given points (x_i, y_i) find (ρ, θ)

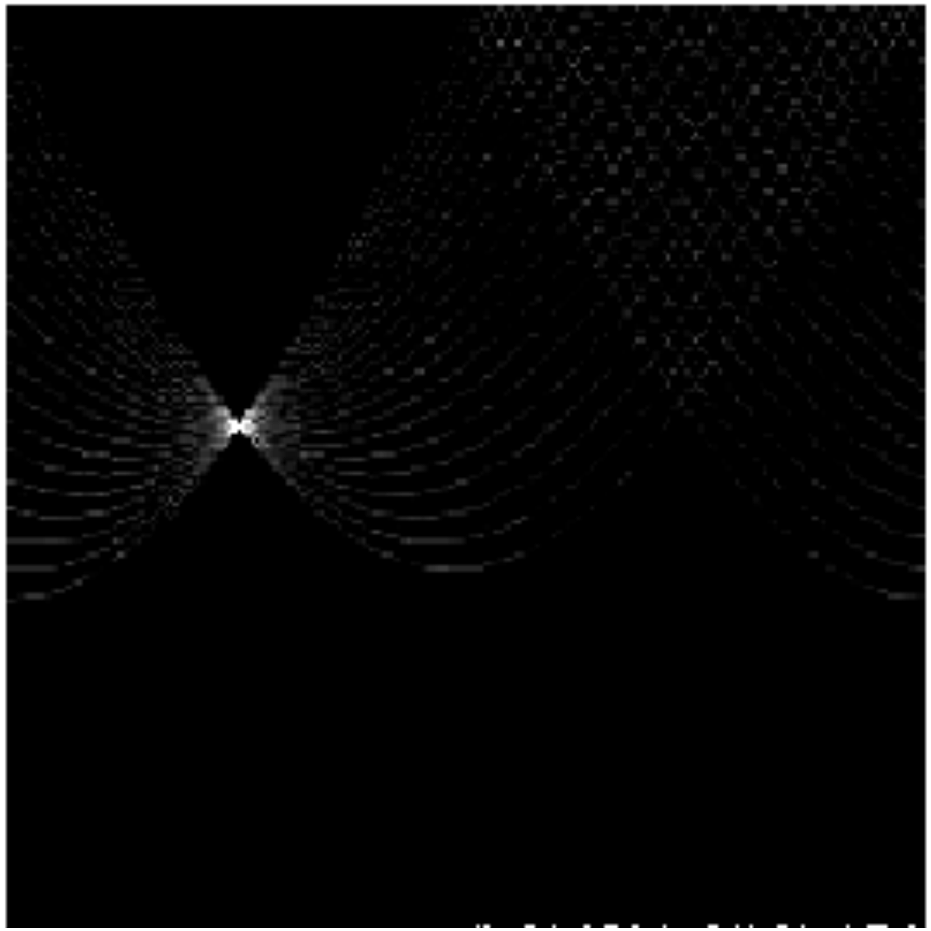
Hough Space Sinusoid



Basic illustration

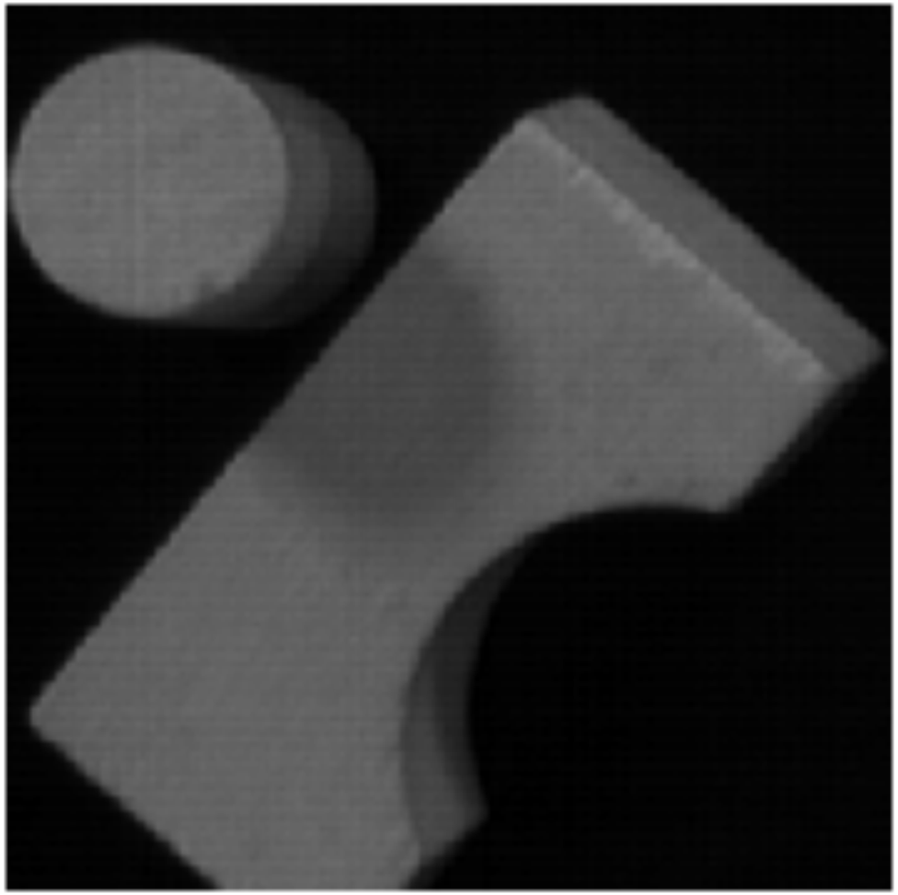


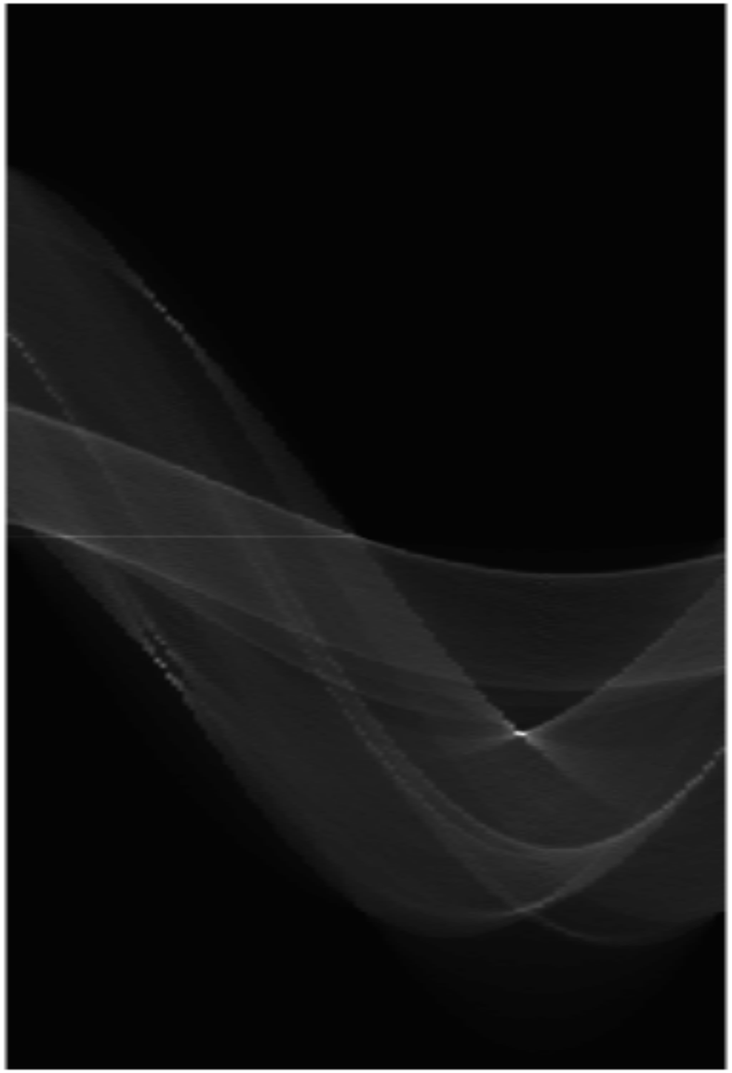
features



votes

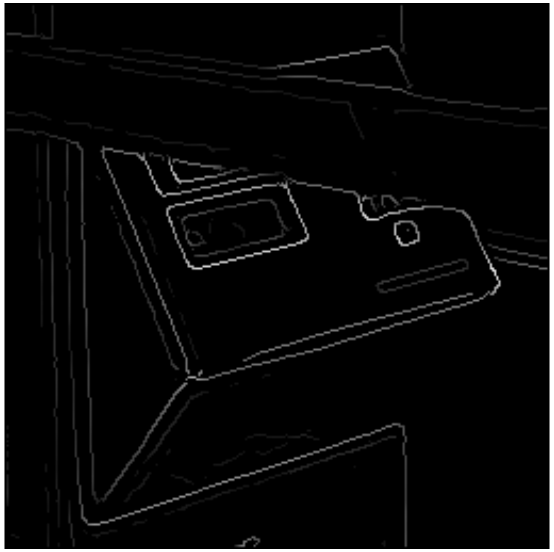
Several lines



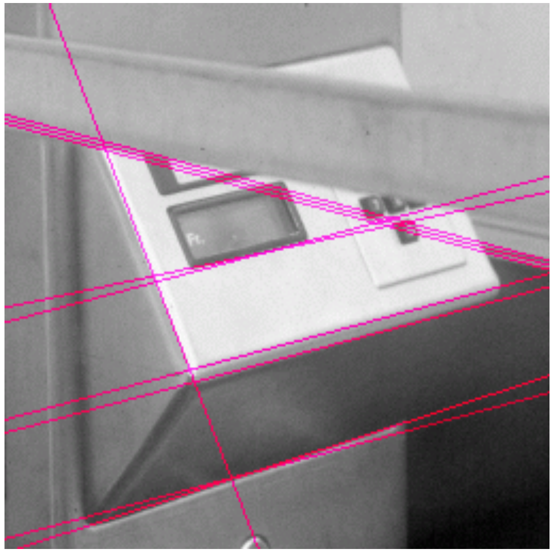


Real World Example

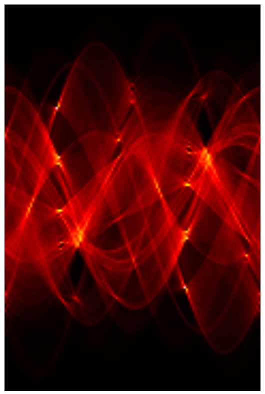
Original



Edge Detection

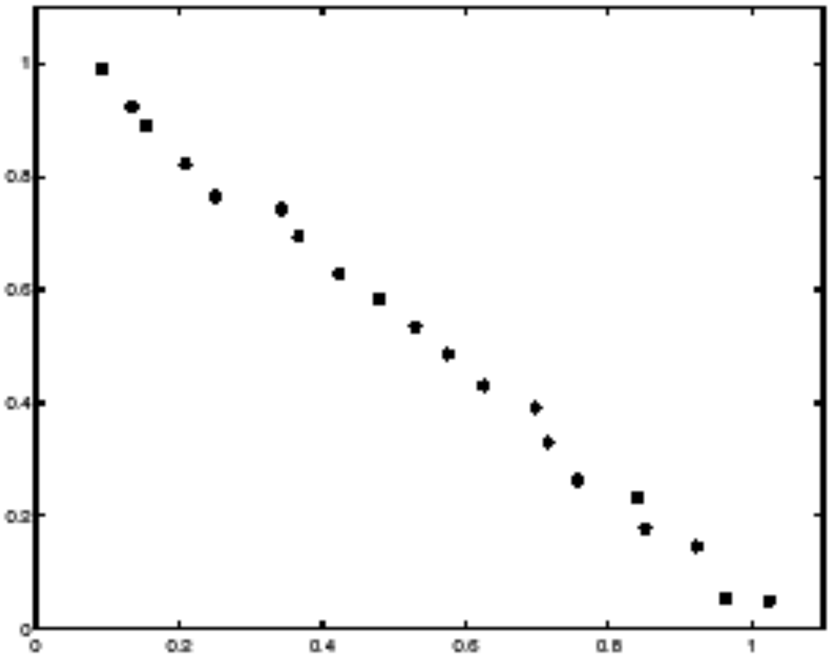


Found Lines

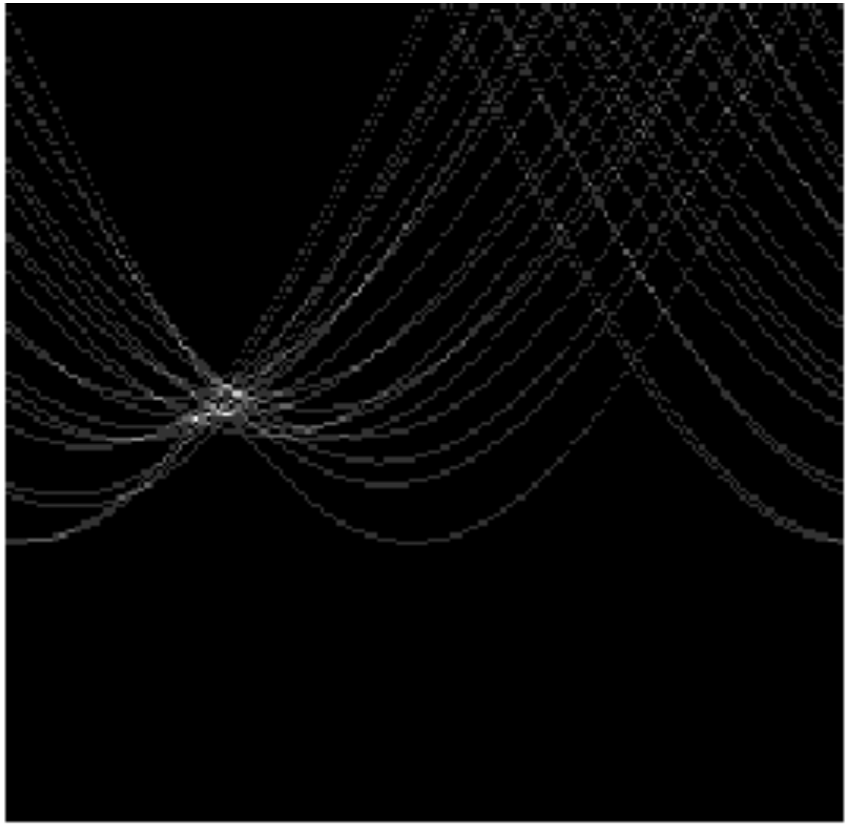


Parameter Space

Effect of noise



features

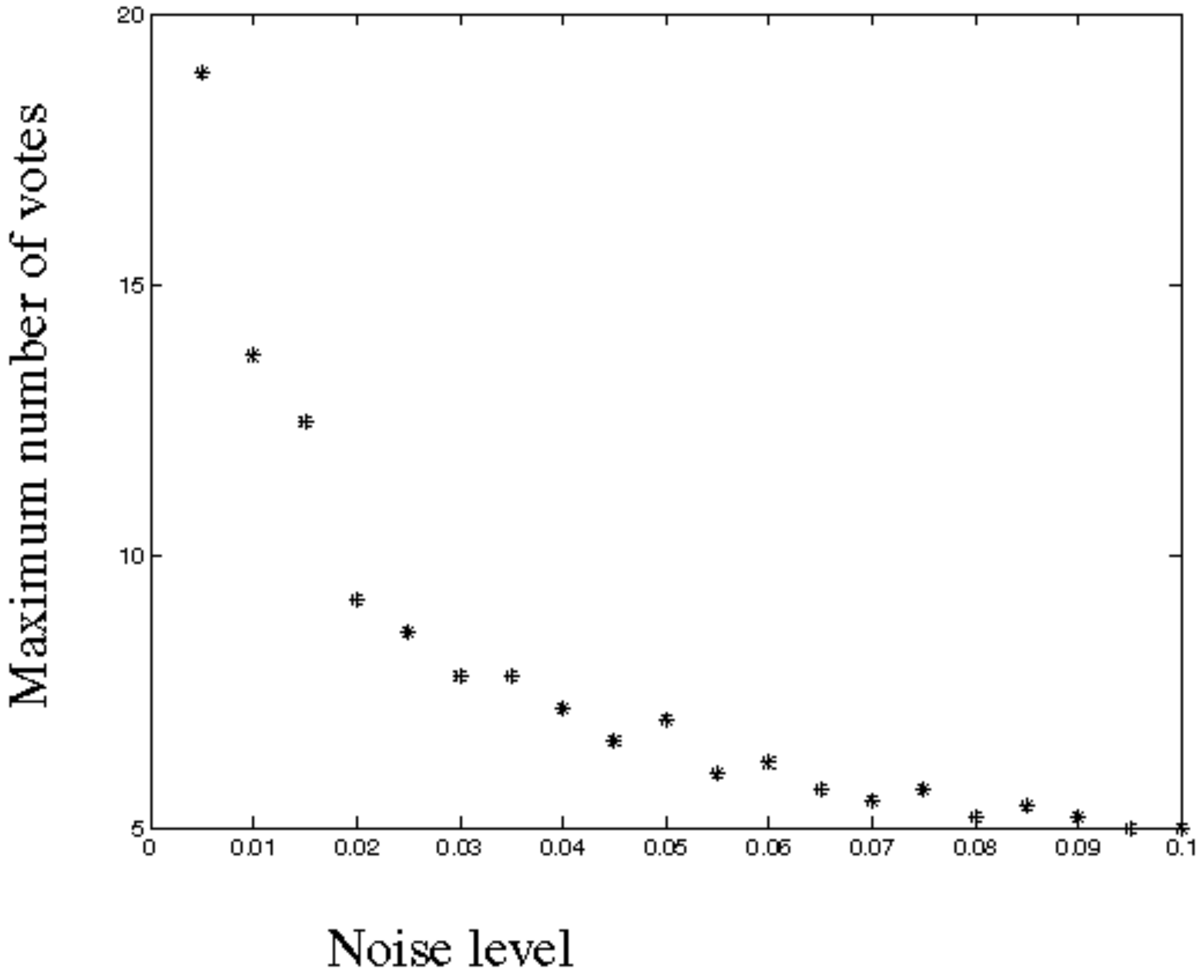


votes

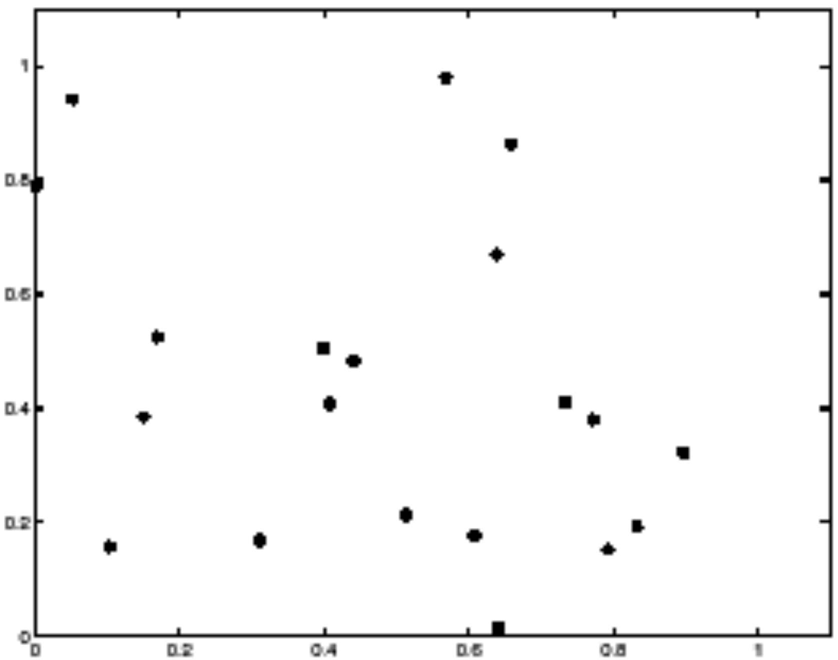
Peak gets fuzzy and hard to locate

Effect of noise

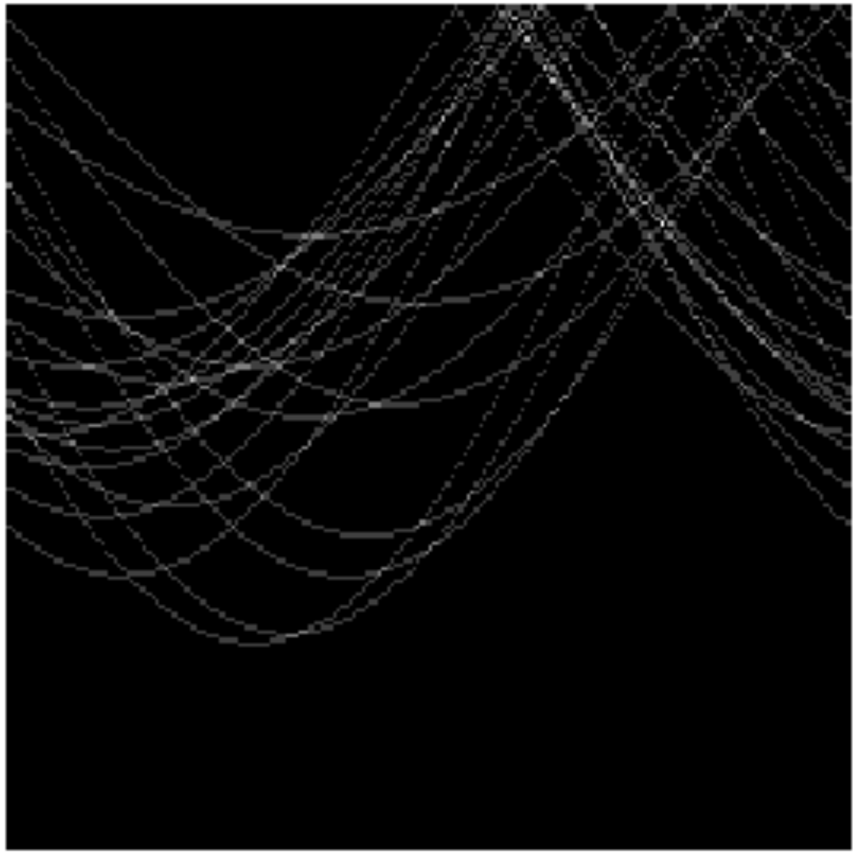
- Number of votes for a line of 20 points with increasing noise:



Random points



features

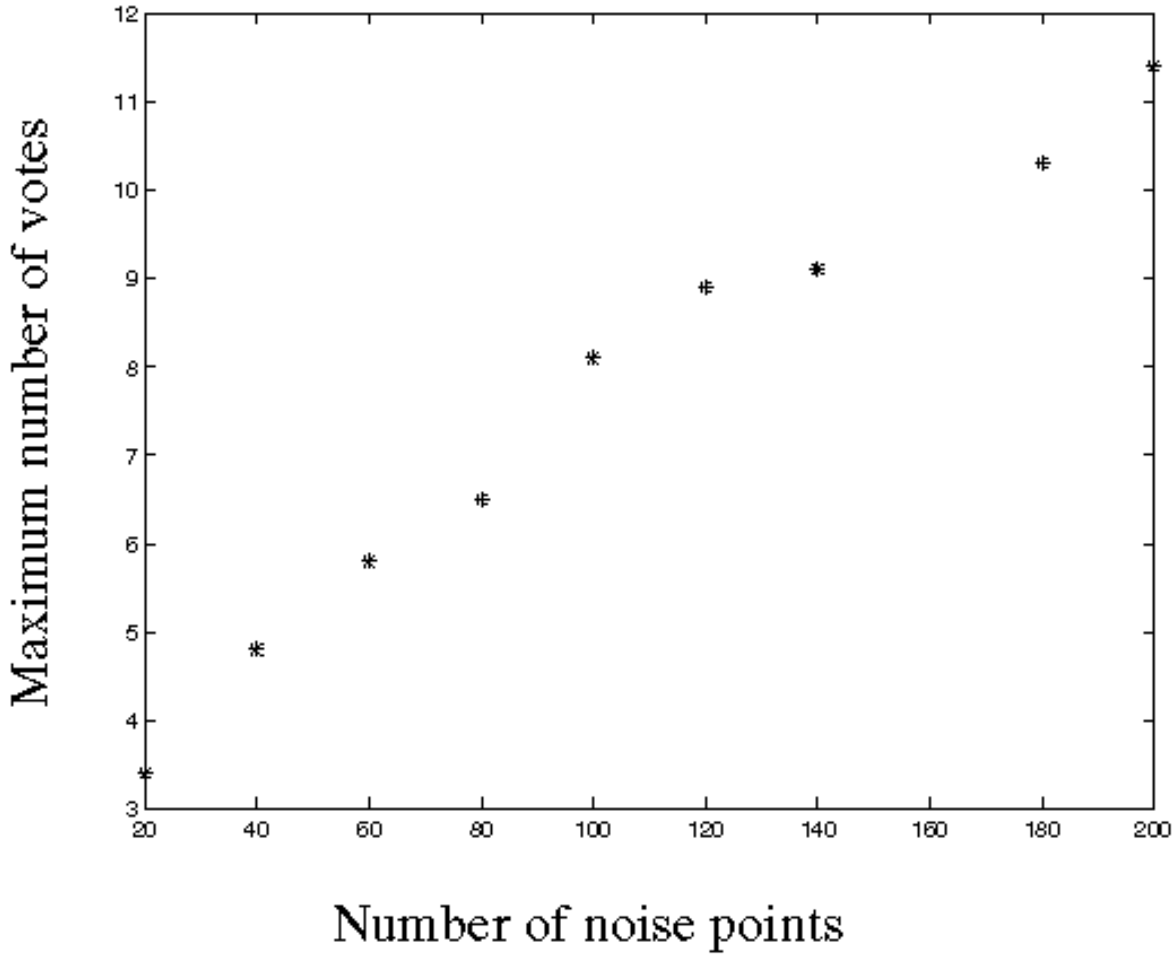


votes

Uniform noise can lead to spurious peaks in the array

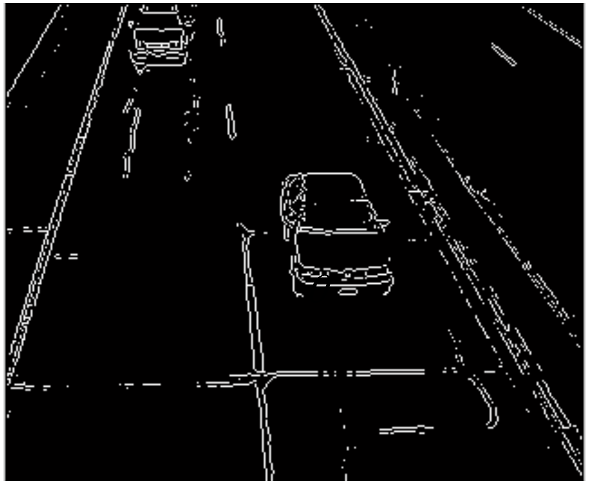
Random points

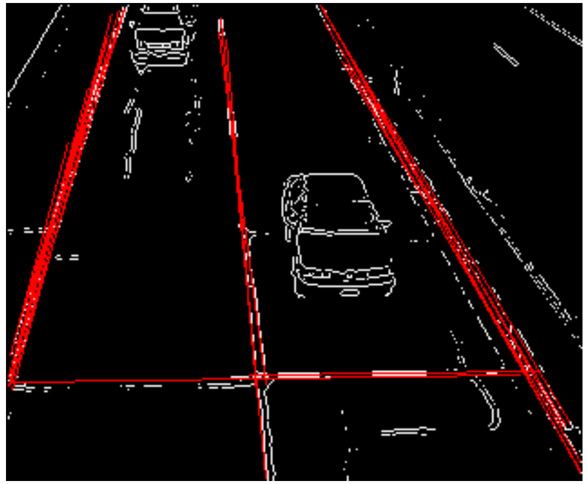
- As the level of uniform noise increases, the maximum number of votes increases too:

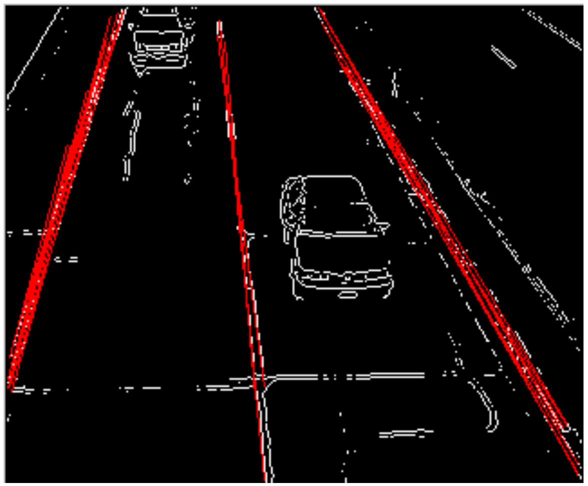


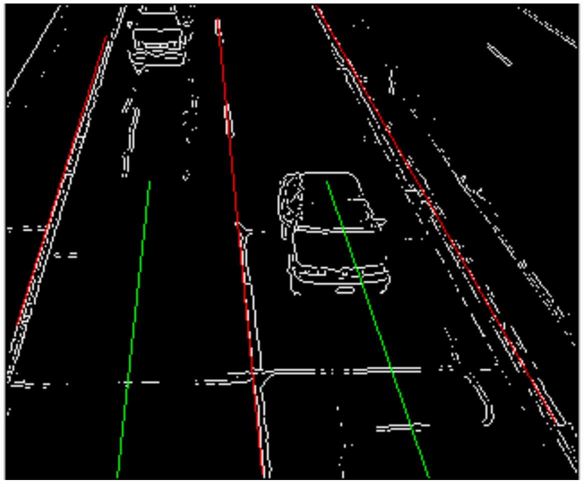
Lane Detection

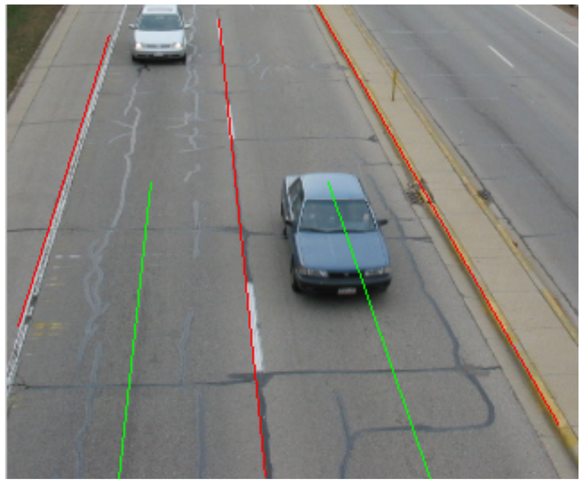

University Ave.


Edge image


Without restriction to θ


Restrict θ


Decide the lane markings


Detected lane markings

Practical details

- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude

- Choose a good grid / discretisation
  - Too coarse: large votes obtained when too many different lines correspond to a single bucket
  - Too fine: miss lines because some points that are not exactly collinear cast votes for different buckets

- Increment neighboring bins (smoothing in accumulator array)

- Who belongs to which line?
  - Tag the votes

Hough transform: Pros

- Can deal with non-locality and occlusion
- Can detect multiple instances of a model in a single pass
- Some robustness to noise: noise points unlikely to contribute consistently to any single bin

Hough transform: Cons

- Complexity of search time increases exponentially with the number of model parameters
- Non-target shapes can produce spurious peaks in parameter space
- It’s hard to pick a good grid size

Finding Circles by Hough Transform

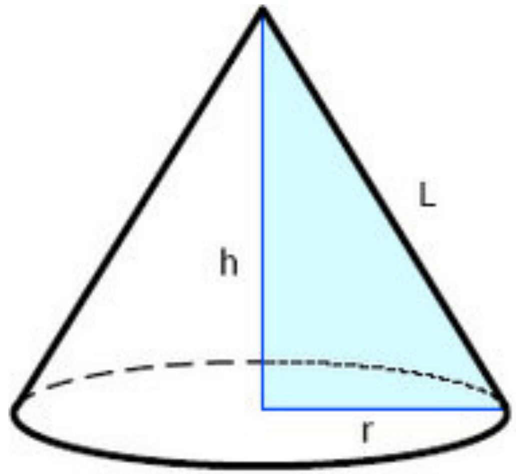
Equation of Circle:

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

The equation has three parameters – a, b, r

If radius is not known: 3D Hough Space!

Use Accumulator array $A(a, b, r)$

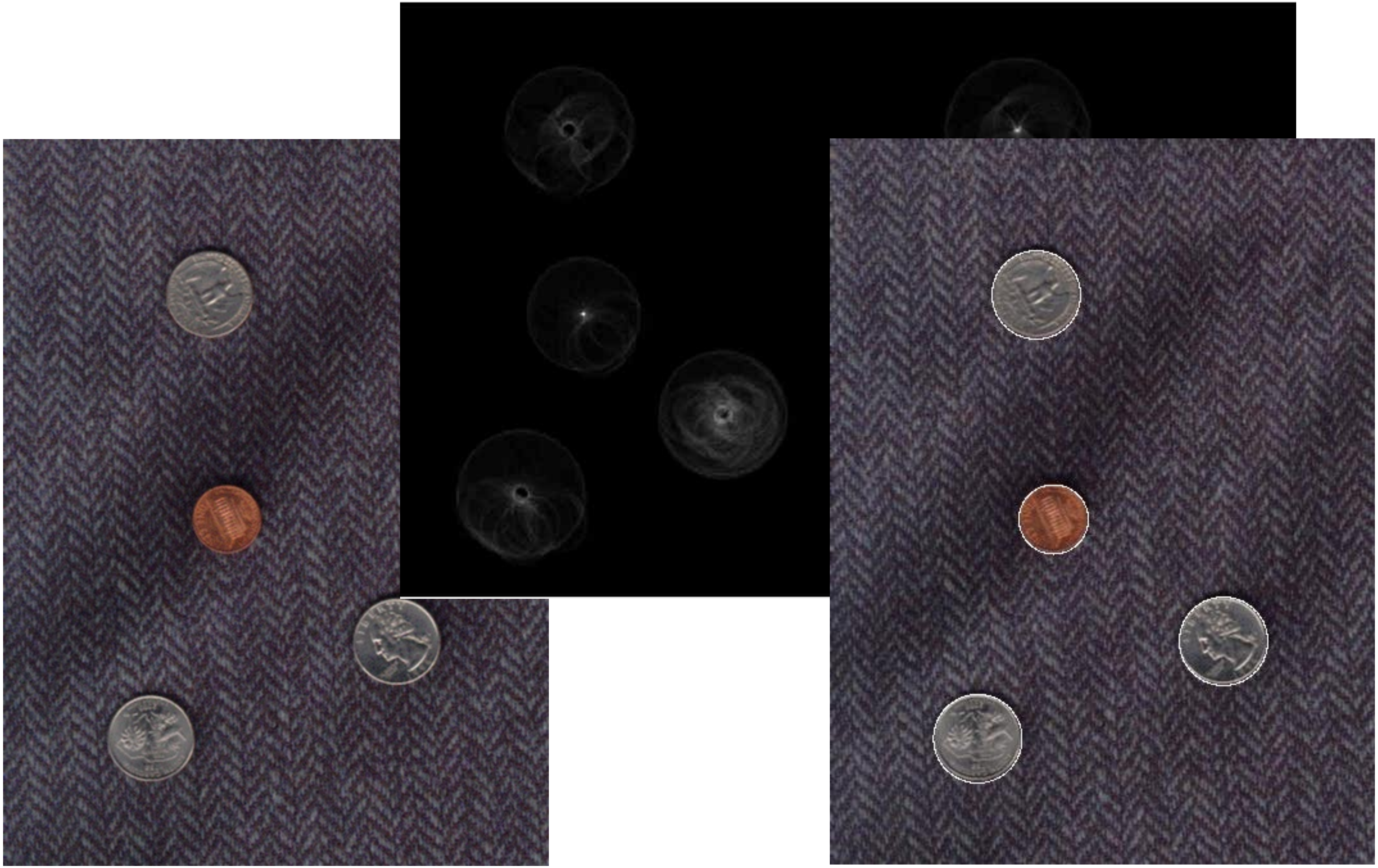


Right Circular Cone

- The curve obtained in the Hough Transform space for each edge point will be a right circular cone

- Point of intersection of the cones gives the parameters a, b, r

Real World Circle Examples



Introduction to Image Processing

Ch 10.3. Thresholding with Otsu’s Method
Ch 9. Morphological Image Processing
Ch 10.2. Line Detection with Hough Transform

Kuan-Wen Chen