ECE 470: Introduction to Robotics

Lecture 25

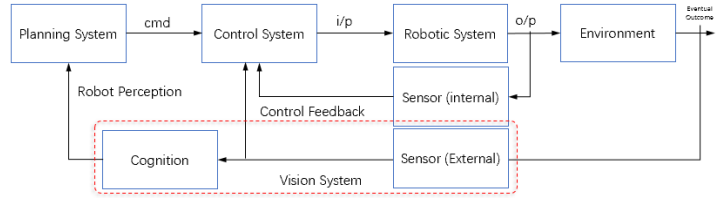
Liangjing Yang

Assistant Professor, ZJU-UIUC Institute

liangjingyang@intl.zju.edu.cn

Wechat ID: Liangjing_Yang

Overview of Robot Vision



O. Introduction to Robot Vision

- **What is Robot Vision?**

I. Image Formation

- The science behind machine vision (+ represent as a form of signal)

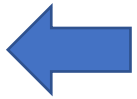
II. Image Processing

- Common techniques to manipulate, enhance & analyse images

III. Robot Vision Applications

- 3D Vision; Photogrammetry; Vision-based techniques in robotics- visual servo, pose estimation, localization, mapping, navigation

Image Processing

- ☑Thresholding & Histogram Processing
- ☑ Filtering
- ☑ Edge & Corner Detection
- Interest Points/ Feature Description 
- Lines & Shapes

Recap: Canny Edge Detection

- Involves 5 steps:
  1. Apply Gaussian filter: remove the noise
  2. Find the intensity gradients of the image
  3. Apply non-maximum suppression: remove spurious response
  4. Apply edge detection using two threshold value
  5. Finalize edge detection by hysteresis

ZJU-UIUC INSTITUTE

Canny Edge Detection

- Step (1): Apply Gaussian filter to smooth the image in order to remove the noise
  - Edge detection are easily affected by image noise
  - A Gaussian filter is applied to convolve with the image
    - $h(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$
  - This will smooth the image to reduce the effects of obvious noise on the edge detector

ZJU-UIUC INSTITUTE

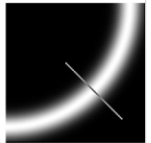
Canny Edge Detection

- Step (2): Find the intensity gradients of the image
  - $\frac{\partial I(x,y)}{\partial x} = \frac{I(x+1,y)-I(x-1,y)}{2}$
  - $\frac{\partial I(x,y)}{\partial y} = \frac{I(x,y+1)-I(x,y-1)}{2}$
  - Gradient Vector: $\nabla I(x,y) = \left[\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y}\right]^T$
    - $|\nabla I(x,y)| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2}$
    - $\theta(x,y) = \tan^{-1}\left(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}\right)$

ZJU-UIUC INSTITUTE

Canny Edge Detection

- Step (3): Apply non-maximum suppression to get rid of spurious response to edge detection
  - compare values along the gradient vector to retain only the maximum



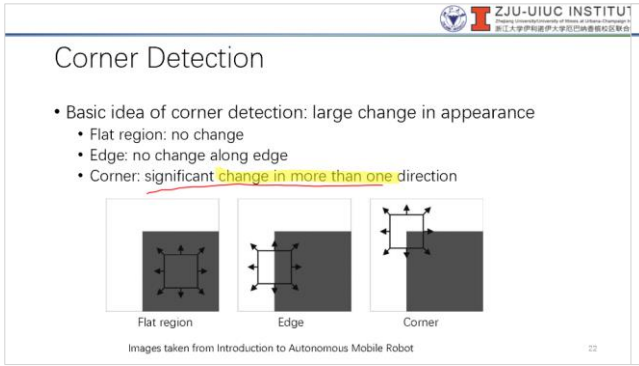
ZJU-UIUC INSTITUTE

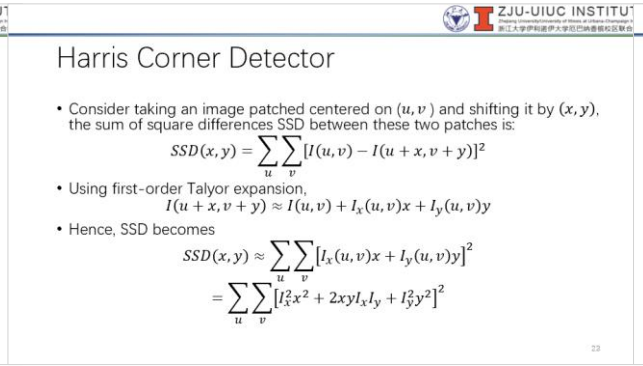
Canny Edge Detection

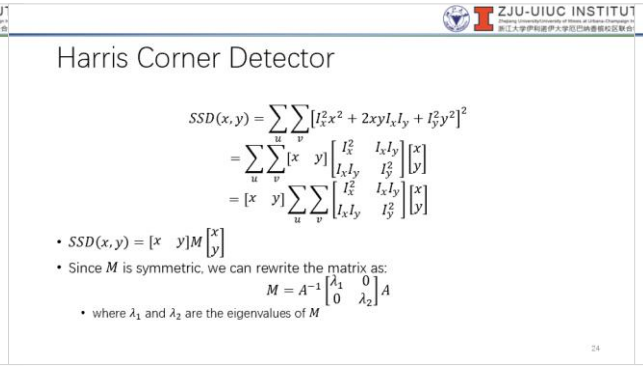
- Step (4): Apply edge detection using two threshold value K_H and K_L
- $Edge(x,y) = \begin{cases} E_{strong} & if\ |\nabla I(x,y)| > K_H \\ E_{average} & if\ K_L \le |\nabla I(x,y)| \le K_H \\ E_{weak} & if\ |\nabla I(x,y)| < K_L \end{cases}$
- Step (5): Finalize edge detection by hysteresis thresholding
  - Small K means more details
  - High K includes more noise
  - Hysteresis Thresholding allows us to apply both
    - Keep both high threshold K_H and low threshold K_L
    - Any edges with magnitude < K_L are discarded
    - Any edges with magnitude > K_H are kept
    - An edge with magnitude between the two threshold values is kept if there is a path of edges with magnitude > K_L connecting the edge to another edge with magnitude > K_H

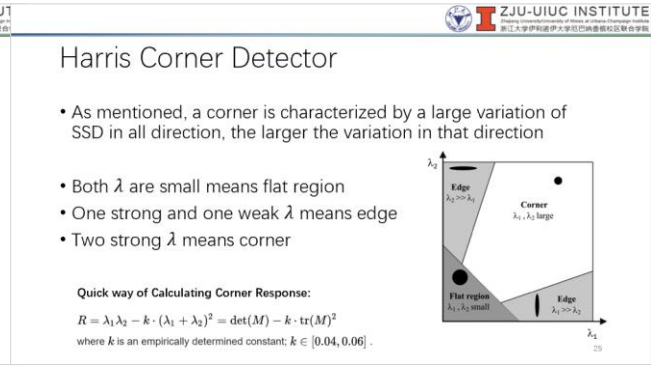
Recap: Harris Corner Detection

- Involves 5 steps:
  1. Apply Gaussian filter: remove the noise
  2. Find the intensity gradients of the image
  3. Apply non-maximum suppression: remove spurious response
  4. Apply edge detection using two threshold value
  5. Finalize edge detection by hysteresis









Feature Extraction

- (Local) Features
  - (Locally) Unique structures e.g. Edges, Corners, Blobs
- Detector
  - Methods for deciding if a pixel is associated with particular a feature e.g. Canny, Harris, LoG
- Descriptor
  - Representations of (local) features

Feature Extraction

- Ideal features should be
  - Repeatable
    - Robustly detectable in different viewpoints and imaging conditions (e.g. noise)
  - Distinctive
    - Uniquely representable for comparison
  - Localizable
    - Provide spatial information

Feature Extraction

- Detector
  - Methods for deciding if a pixel is associated with particular a feature e.g. Canny, Harris, LoG

Detector	Feature Type	Function	Scale Independent
FAST [1]	Corner	`detectFASTFeatures`	No
Minimum eigenvalue algorithm [4]	Corner	`detectMinEigenFeatures`	No
Corner detector [3]	Corner	`detectHarrisFeatures`	No
SURF [11]	Blob	`detectSURFFeatures`	Yes
KAZE [12]	Blob	`detectKAZEFeatures`	Yes
BRISK [6]	Corner	`detectBRISKFeatures`	Yes
MSER [8]	Region with uniform intensity	`detectMSERFeatures`	Yes
ORB [13]	Corner	`detectORBFeatures`	No

https://www.mathworks.com/help/vision/ug/local-feature-detection-and-extraction.html#bulf47t-4

Feature Extraction

- Descriptor
  - Representations of (local) features

Descriptor	Binary	Function and Method	Invariance		Typical Use	
			Scale	Rotation	Finding Point Correspondences	Classification
HOG	No	`extractHOGFeatures(I, ...)`	No	No	No	Yes
LBP	No	`extractLBPFeatures(I, ...)`	No	Yes	No	Yes
SURF	No	`extractFeatures(I,points,'Method','SURF')`	Yes	Yes	Yes	Yes
KAZE	No	`extractFeatures(I,points,'Method','KAZE')`	Yes	Yes	Yes	Yes
FREAK	Yes	`extractFeatures(I,points,'Method','FREAK')`	Yes	Yes	Yes	No
BRISK	Yes	`extractFeatures(I,points,'Method','BRISK')`	Yes	Yes	Yes	No
ORB	Yes	`extractFeatures(I,points,'Method','ORB')`	No	Yes	Yes	No
• Block • Simple pixel neighborhood around a keypoint	No	`extractFeatures(I,points,'Method','Block')`	No	No	Yes	Yes

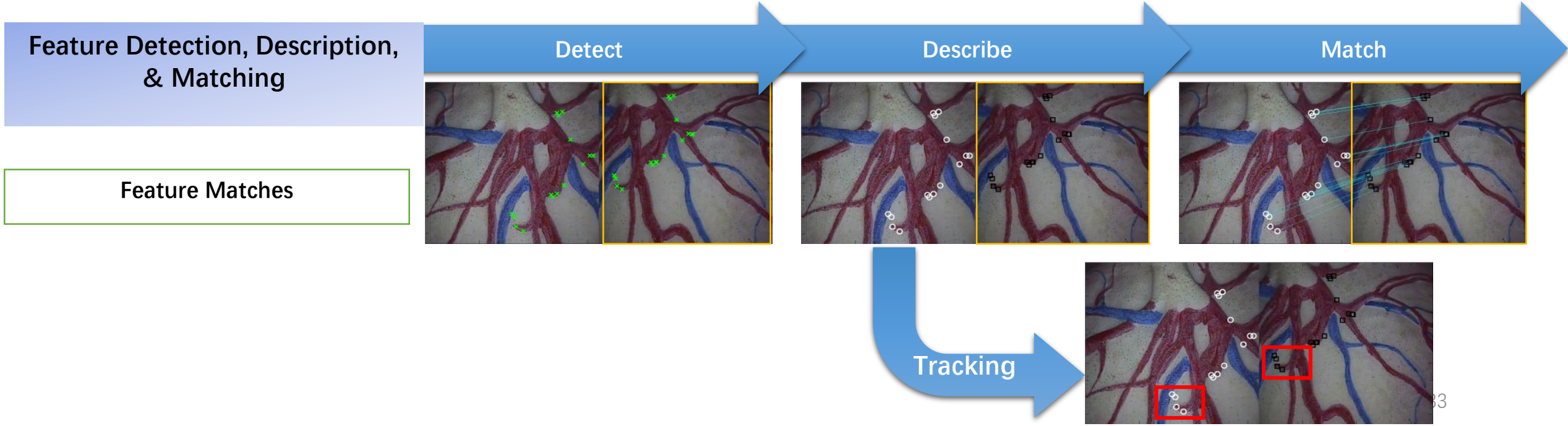
https://www.mathworks.com/help/vision/ug/local-feature-detection-and-extraction.html#bulf47t-4

Feature Extraction Practical Consideration

- Select appropriate types of feature-detector-descriptor
  - Based on
  - Nature of scenes
  - Application context
  - Operation requirement

Feature Matching

- Given views of the same scene, how to match the features?
  - Detect; Describe; Match; Transform (for image mapping) or Tracking (Visual tracking)



Similarity Scores between image patches

- How do we measure the similarity/ difference?

Similarity Score

- Sum of Squared Difference (SSD)

SSD between two image patches I_1 *and* I_2

$$w_{SSD}(x,y) = \sum_{p}^{P}\sum_{q}^{Q}\left[I_1(p,q) - I_2(p,q)\right]^2$$

Recall in corner detection, we look at change of patch I centered on (p,q) and itself when shifted by (x,y), we used SSD to represent the change

$$w_{SSD}(x,y) = \sum_{p}^{P}\sum_{q}^{Q}\left[I(p,q) - I(p+x,q+y)\right]^2$$

Similarity Score

- Normalized Cross-correlation (NCC)

For a U x V image and P x Q patch, the cross-correlation $w_{cc}(u,v)$ at a particular image coordinates (u, v) of a template patch $g(p,q)$ and the image $f(p,q)$ is

$$w_{cc}(u,v) = \sum_{p=0}^{P}\sum_{q=0}^{Q} g(p,q) f(p+u, q+v)$$

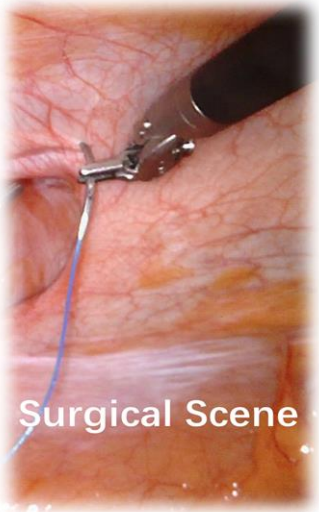
To reduce the over sensitivity toward intensity variant, normalized cross-correlation coefficient (Similarity Score),

$$w_{ncc}(u,v) = \frac{\sum_{p=0}^{P}\sum_{q=0}^{Q}\left(g(p,q)-\bar{g}\right)\left(f(p+u,q+v)-\bar{f}(u,v)\right)}{\left[\left(\sum_{p=0}^{P}\sum_{q=0}^{Q}\left(g(p,q)-\bar{g}\right)^2\right)\left(\sum_{p=0}^{P}\sum_{q=0}^{Q}\left(f(p+u,q+v)-\bar{f}(u,v)\right)^2\right)\right]^{0.5}}$$

is used where $\bar{g}$ *and* $\bar{f}$ denotes the mean intensity in the template and overlapping region, respectively.

Detection of Line and Shape

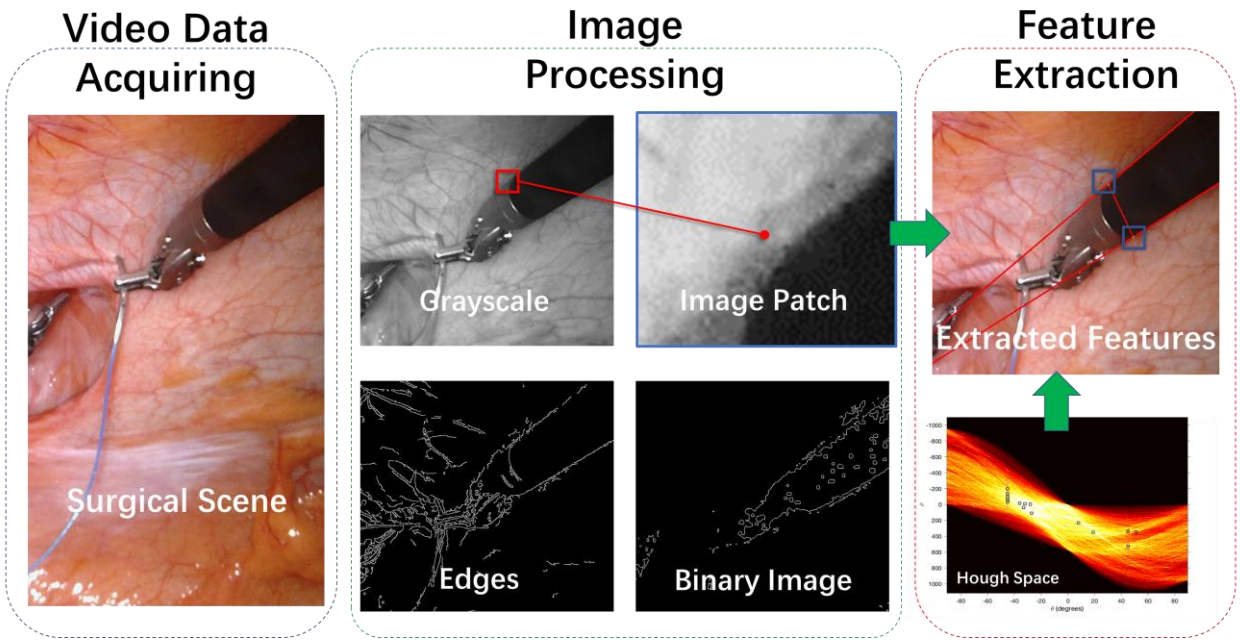
- After detecting the edges and local interest points, how do we detect lines and other geometries?
- A problem of pattern recognition

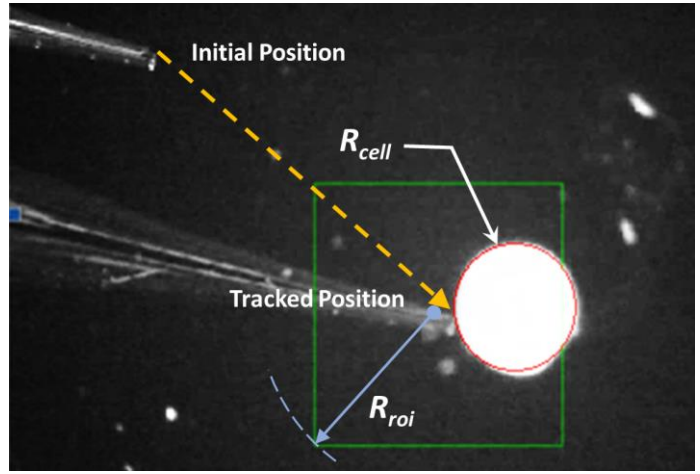


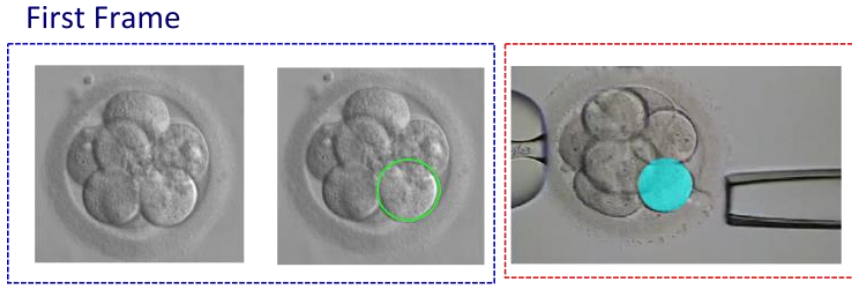
Huang, J., Li, X., Kesavadas, T. and Yang, L., 2019, July. Feature Extraction of Video Data for Automatic Visual Tool Tracking in Robot Assisted Surgery. In *Proceedings of the 2019 4th International Conference on Robotics, Control and Automation* (pp. 121-127).

Detection of Line and Shape

- After detecting the edges and local interest points, how do we detect lines and other geometries?
- A problem of pattern recognition



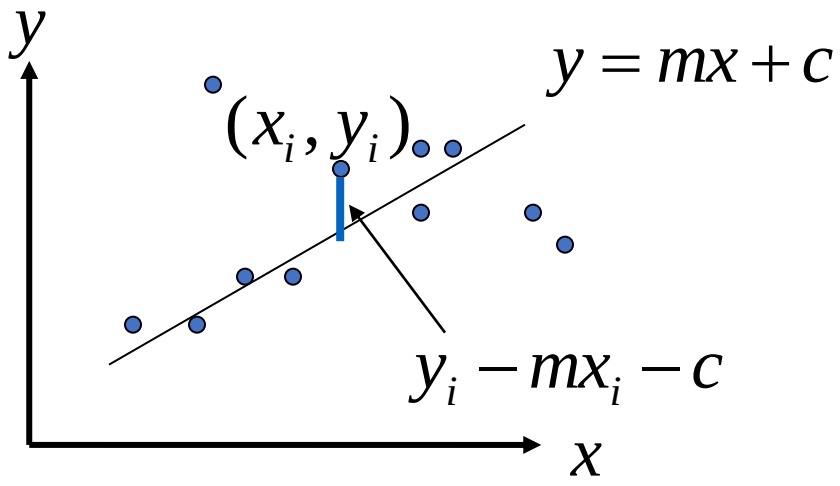




Huang, J., Li, X., Kesavadas, T. and Yang, L., 2019, July. Feature Extraction of Video Data for Automatic Visual Tool Tracking in Robot Assisted Surgery. In *Proceedings of the 2019 4th International Conference on Robotics, Control and Automation* (pp. 121-127).

Fitting lines to edges using least square

- Given:
  - Many edges (x_i, y_i)
  - Equation of line: $y = mx + c$
  - Parameters: m , c

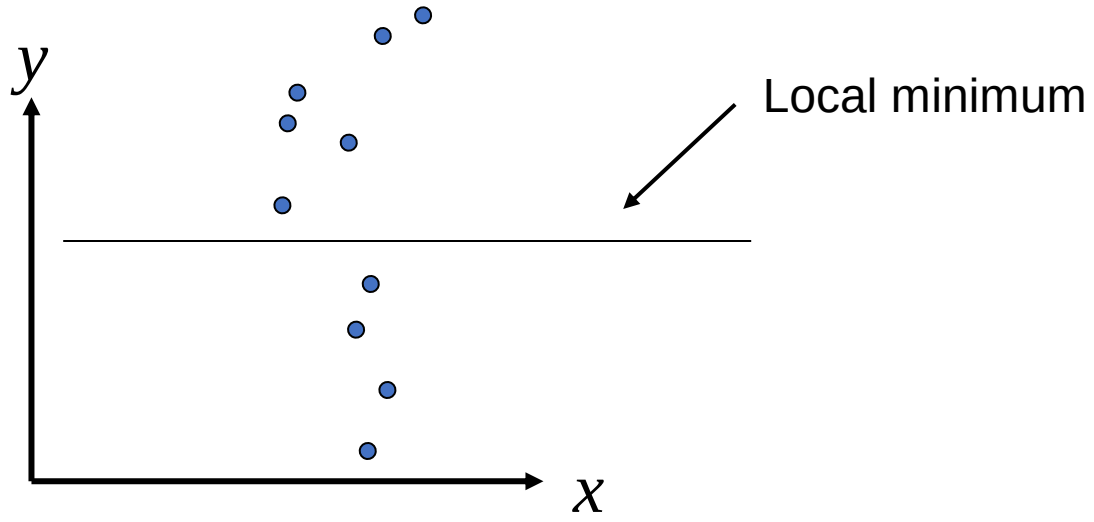


- Objective function is to minimize average square distance:
  - $E = \sum \frac{(y_i - mx_i - c)^2}{N}$
- Using $\frac{\partial E}{\partial m} = 0$ & $\frac{\partial E}{\partial c} = 0$

$$c = \bar{y} - m\,\bar{x}$$

$$m = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2}$$

Fitting lines to edges using least square



- Possible Solution: Hough Transformation

Hough Transform

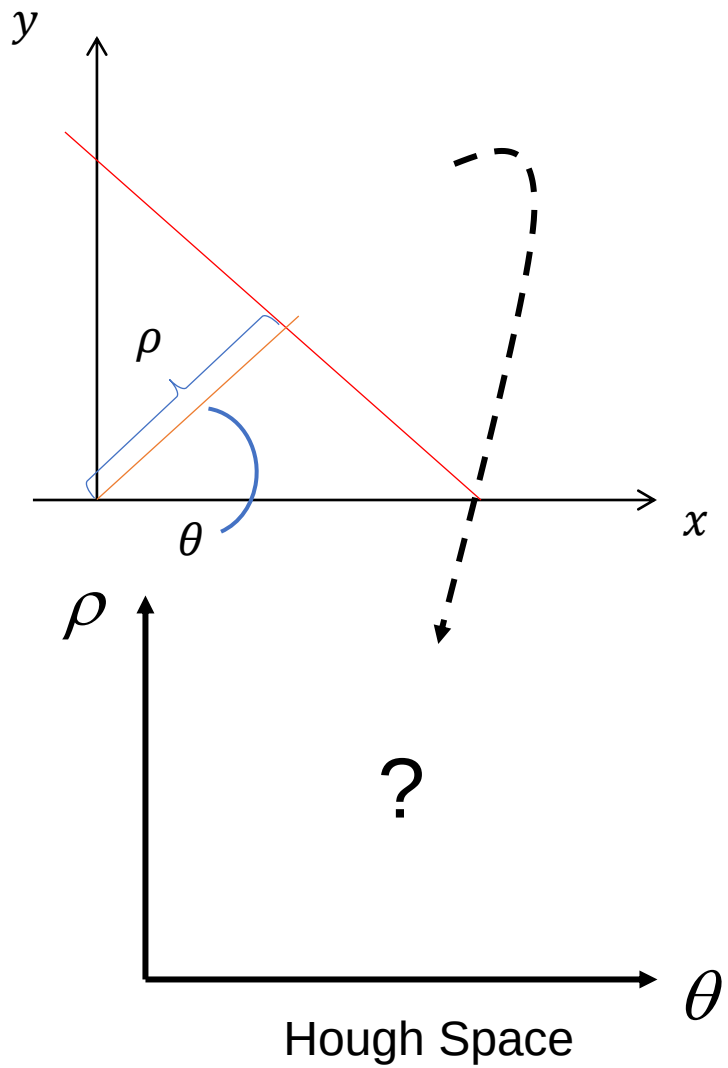
- Ability to detect edges
- What about lines? Shapes (eg: circle)?

- Hough Transform
  - Elegant method for direct object recognition
  - Edges need not be connected
  - Key idea: edges “vote” for the possible model

Hough Space – $(\boldsymbol{\rho}, \boldsymbol{\theta})$ space

- $\rho = x\cos\theta + y\sin\theta$
- $y = \frac{\rho}{\sin\theta} - \frac{x}{\tan\theta}$

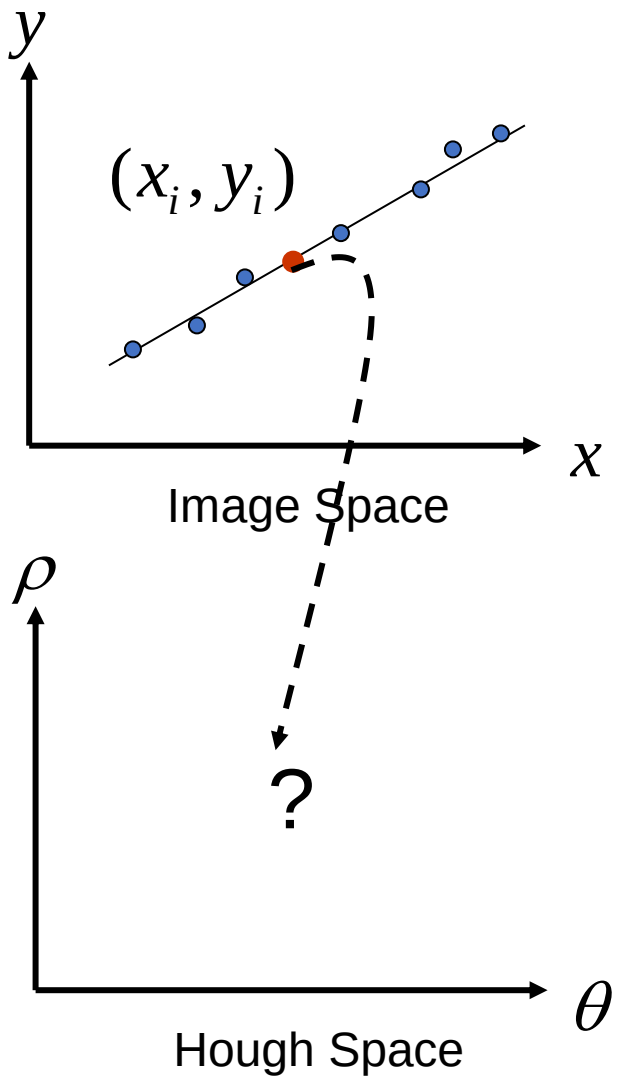
- Line equation: $\rho = x\cos\theta + y\sin\theta$
  - Parameters: ρ and θ
  - Where $0 \leq \theta \leq 2\pi$



Hough Space – $(\boldsymbol{\rho}, \boldsymbol{\theta})$ space

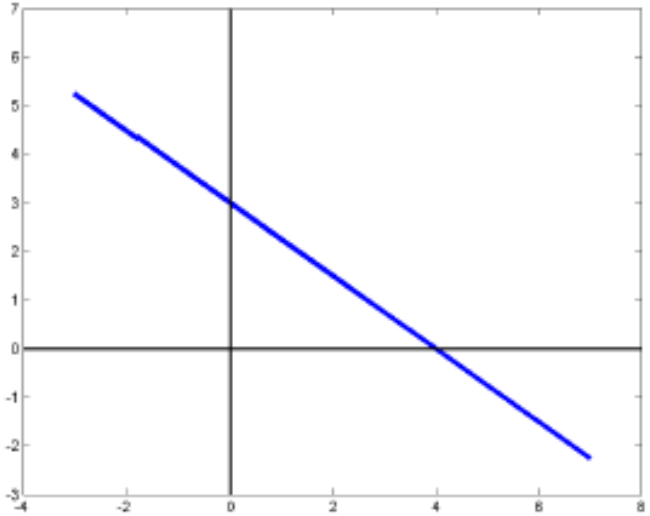
- Line equation: $\rho = x\cos\theta + y\sin\theta$
  - Parameters: ρ and θ
  - Where $0 \leq \theta \leq 2\pi$

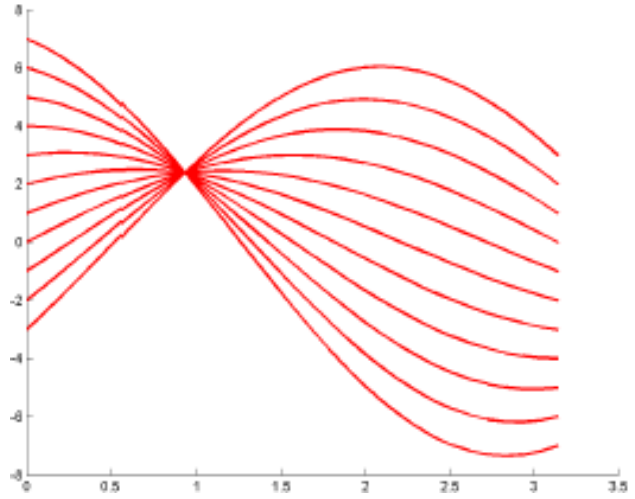
- How to map onto the Hough Space (sometimes also known as parameter space)?
  - $\rho = x_i\cos\theta + y_i\sin\theta$
  - Sinusoid curve



Hough Space – $(\boldsymbol{\rho}, \boldsymbol{\theta})$ space

- $\rho = x\cos\theta + y\sin\theta$
- Points in picture → sinusoids in parameter space
- Points in parameter space → lines in picture
- There will be a unique intersection point if the points in the picture form a straight line





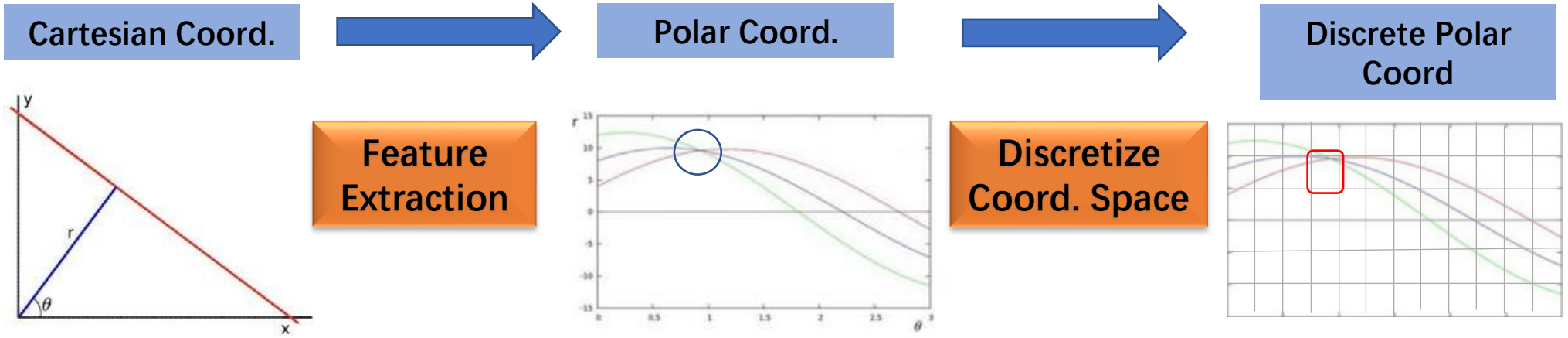
But how to choose the solution if there are noise?

Quantizing parameter space and Voting

- Discretize the (ρ, θ) space
  - For each point (x_i, y_i), compute only for a finite set of angles $\theta = \theta_1, \theta_2, \dots, \theta_N$
  - For each θ_j, obtain $\rho_{ij} = x_i \cos\theta_j + y_i \sin\theta_j$
- Create a matrix, called the accumulator matrix
  - Each column corresponds to angles $\theta = \theta_1, \theta_2, \dots, \theta_N$
  - Each row corresponds to the "bins" (intervals) of the resulting distance ρ
- Voting:
  - For each point in the image and for each θ_j, compute the ρ_{ij}, and increment the corresponding element of the accumulator matrix
  - Highest value means highest "vote"

Quantizing parameter space and Voting

- Voting (continue)
  - If more than one line, you can set a threshold value (number of vote) to obtain more lines
  - For example, if number of votes (ie value in that element) is more than the threshold value, then consider that (ρ, θ) to be a line

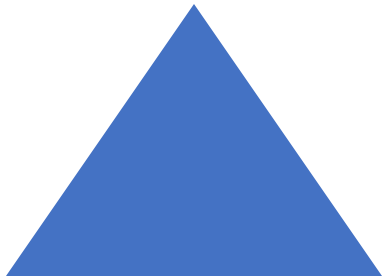


Case Problem (1)

Using what you have learnt, design an algorithm that is capable of identifying the following simple blocks on white background.

1)Rectangle

2)Triangle



Solution

Case Problem (1) Solution

- Stage (1): **Edge detection** using Canny edge detection
  - Apply Gaussian filter to smoothen the image in order to remove the noise
  - Find the intensity gradients vector of the image
    - $|\nabla I(x,y)| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2}$
    - $\theta(x,y) = \tan^{-1}\left(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}\right)$
  - Apply non-maximum suppression to get rid of spurious response to edge detection
  - Apply edge detection using two threshold value K_H and K_L
    - $Edge(x,y) = \begin{cases} E_{strong} & if \ |\nabla I(x,y)| > K_H \\ E_{average} & if \ K_L \le |\nabla I(x,y)| \le K_H \\ E_{weak} & if \ |\nabla I(x,y)| < K_L \end{cases}$ Finalize edge detection by hysteresis
    - Any edges with magnitude < K_L are discarded
    - Any edges with magnitude > K_H are kept
    - An edge with magnitude between the two threshold values is kept if there is a path of edges with magnitude > K_L connecting the edge to another edge with magnitude > K_H

Case Problem (1) Solution

- Stage (2): Hough Transformation – **line detection**
  - Use $\rho = x\cos\theta + y\sin\theta$
  - Map it onto the Hough space – (ρ, θ) space
  - Intersection point exist if the points in the picture form a straight line
  - As more than one line is involved, various (ρ, θ) are considered to be a line number if number of votes is more than the threshold value

Case Problem (1) Solution

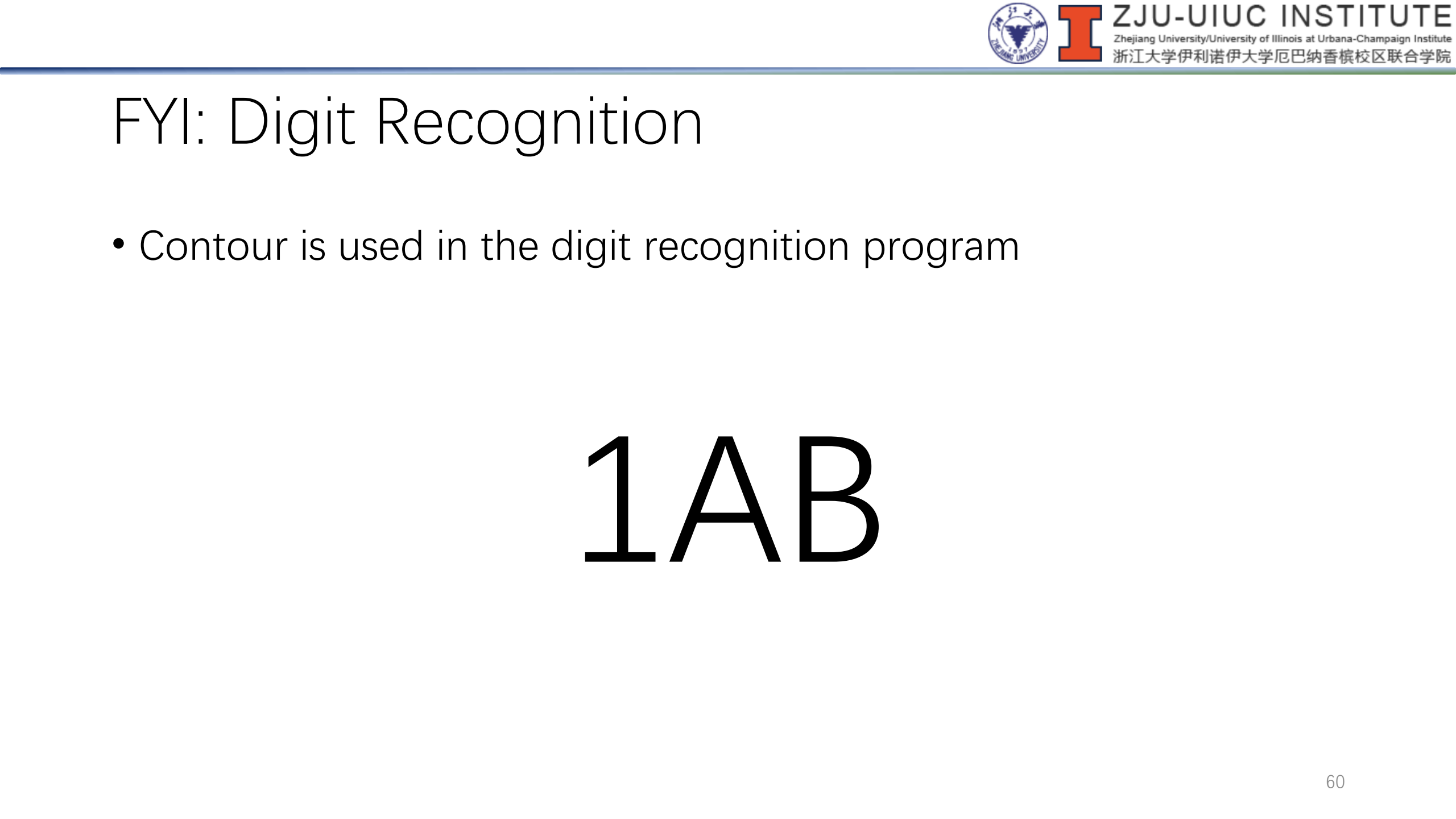
- Stage (2): Hough Transformation - **voting**
  - Discretize the (ρ, θ) space
    - For each point (x_i, y_i), compute only for a finite set of angles $\theta = \theta_1, \theta_2, \dots, \theta_N$
    - For each θ_j, obtain $\rho_{ij} = x_i \cos\theta_j + y_i \sin\theta_j$
  - Create a matrix, called the accumulator matrix
    - Each column corresponds to angles $\theta = \theta_1, \theta_2, \dots, \theta_N$
    - Each row corresponds to the "bins" (intervals) of the resulting distance ρ
  - Voting:
    - For each point in the image and for each θ_j, compute the ρ_{ij}, and increment the corresponding element of the accumulator matrix
    - Highest value means highest "vote"

Case Problem (1) Solution

- Stage (3) – Shape identification
  - Rectangle:
    - 4 vertices
    - All angles are ~ 90 degrees apart
      - Checked using θ
  - Triangle
    - 3 vertices

FYI: Digit Recognition

- Contour is used in the digit recognition program



Detection of other shapes

- Hough Transform can be generalized to find other shapes like circles and ellipses.
- However, the computational complexity increases
  - More computational time is required

Hough circle transform

- Equation of circle:

$$(x-a)^2+(y-b)^2=R^2$$

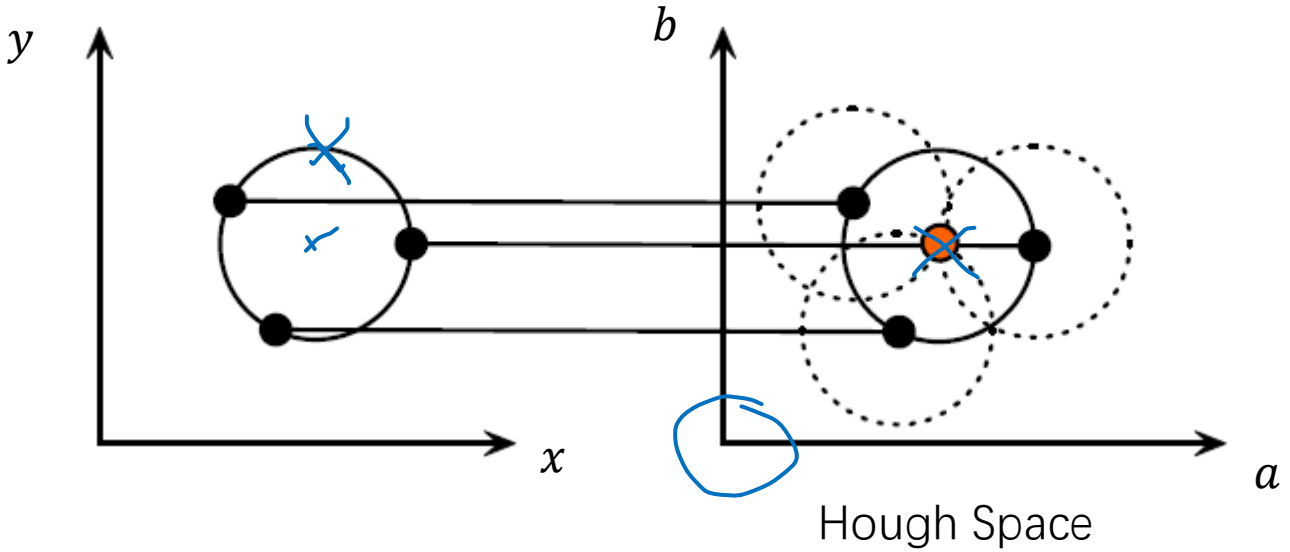
  - (a,b) is the center of the circle
  - R is the radius of the circle
- Parameters: a, b, R
- Rewriting the equation:

$$(a-x)^2+(b-y)^2=R^2$$

Circle with known $\boldsymbol{R}$

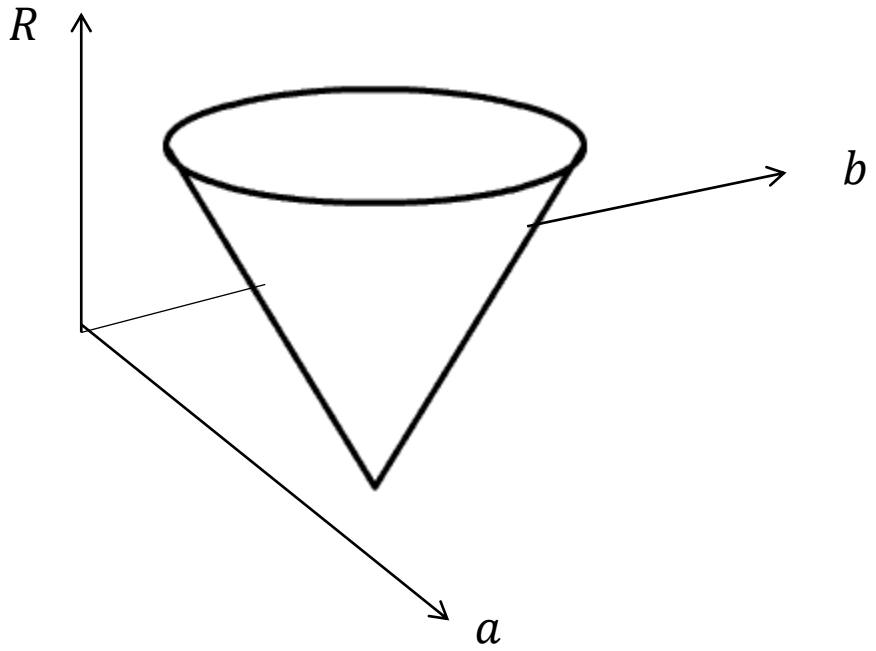
$$(a-x)^2+(b-y)^2=R^2$$

- Parameters: a, b
- Points (edge) in picture → circle in parameter space
- Points in parameter space → circle in picture
- Like line detection, discretization and voting are used
- Example:



Circle with unknown R

- Parameters: a, b, R
- Points (edge) in picture ↔ conical surface in parameter space
- Additional computational time



Application: Extraction of Blastomere

- Embryo Biopsy
- 8-cell stage
- Tracking of the extraction of blastomere for biopsy