CSI 4133 Computer Methods in Picture Processing and Analysis

Fall 2024

Pengcheng Xi, Ph.D.

Outline

- From edge to boundary detection
- Basic principle
- Introduction to Hough Transform
- Line detection with Hough Transform
- Circle detection using Hough Transform
- Advantages and limitations

Transition from edge detection to boundary detection

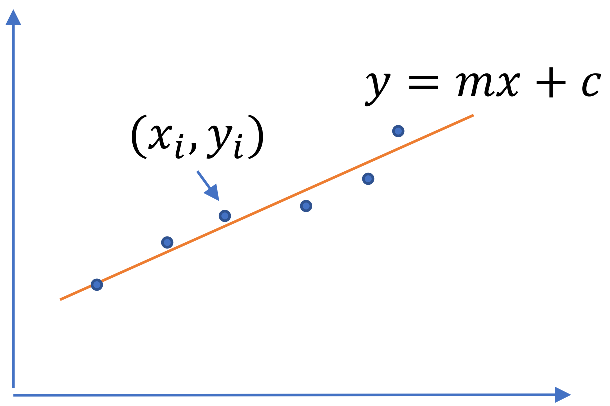
- **Edge detection** helps identify points in an image where intensity changes sharply; however, after detecting edges, we face the challenge of interpreting these edges to identify meaningful shapes or boundaries within an image, such as lines, circles, or more complex curves
- **Boundary detection** aims to extract geometric shapes (like lines, circles) by interpreting these detected edges. This is where Hough Transform comes into play

Example: line and circle detection

- Line detection example:
  - In autonomous driving, edge detection may reveal individual points along the lane markings, but we need to group those points and interpret them as straight lines
- Circle detection example:
  - In a medical image, edge detection may outline parts of a circular boundary of a cell. How can we group these points into a circle, considering how we might not have a complete circular edge due to noise or occlusion?

Fitting lines to edges

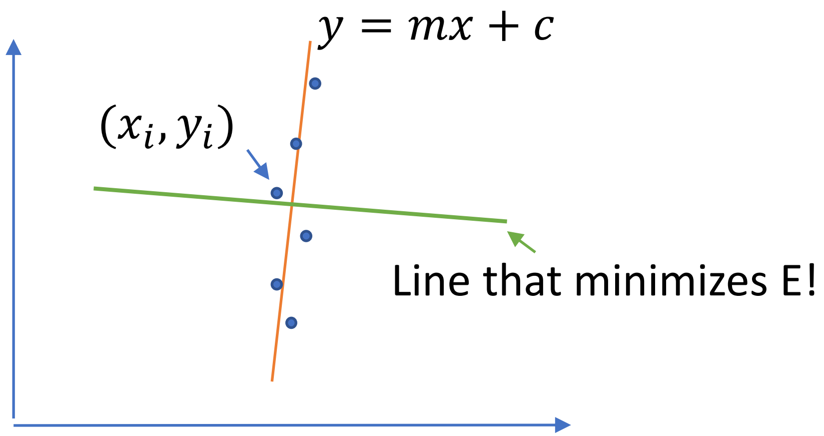
- Given edge points (x_i, y_i)
- Task: find (m, c)



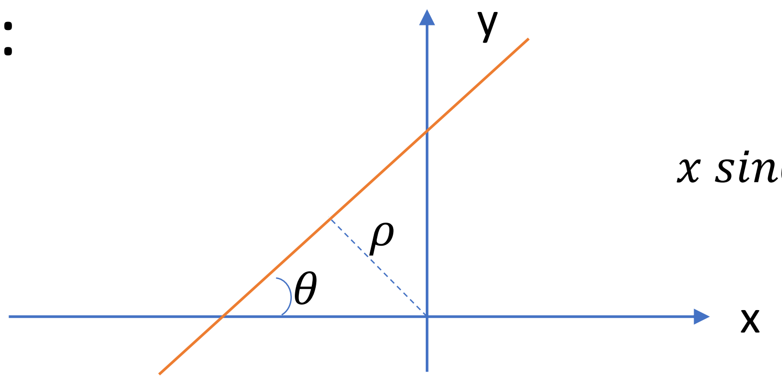
- Solution: to form a cost function through computing squared vertical distance
  - $E = \frac{1}{N} \sum_i (y_i - mx_i - c)^2$
  - $\frac{\partial E}{m} = 0, \frac{\partial E}{c} = 0$ => compute value of *m* and *c*

Fitting lines to edges

- Problem when the points represent a vertical line
  - The solution will lead to a horizontal line!



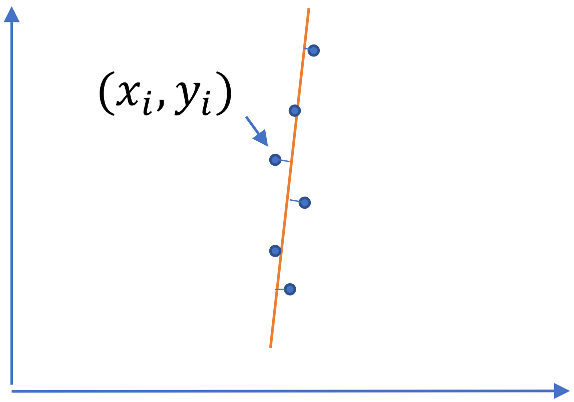
- We need a different line equation:



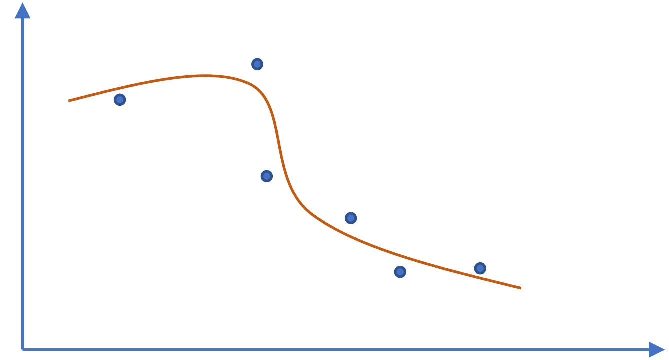
$$x\ sin\theta\ - y\ cos\theta + \ \rho = 0$$

Fitting lines to edges

- Now define an average squared **perpendicular** distance
  - $E = \frac{1}{N}\sum_i (x_i \; sin\theta \; - y_i \; cos\theta \; + \; \rho)^2$



Fitting curves to edges



- Extending the previous approach to **polynomials**
- Task: to find polynomial $y = f(x) = ax^3 + bx^2 + cx + d$ that best fits the points
- Minimize:
  - $E = \frac{1}{N}\sum_i (y_i - a{x_i}^3 - b{x_i}^2 - cx_i - d)^2$
  - Solve the linear system using partial derivatives

What is the Hough Transform

- A feature extraction technique used in image analysis and computer vision
- Commonly used to detect shapes like lines and circles in an image
- Works by transforming points in the **image space** to **parameter space**

Hough Transform Basics

- Converts edge points (from edge detection algorithms like Canny) into curves in a parameter space
- For line detection:
  - Equation of a line: $y = mx + c$
  - Parametric form: $\rho = xcos(\theta) + ysin(\theta)$
- Each point votes for all possible lines passing through it

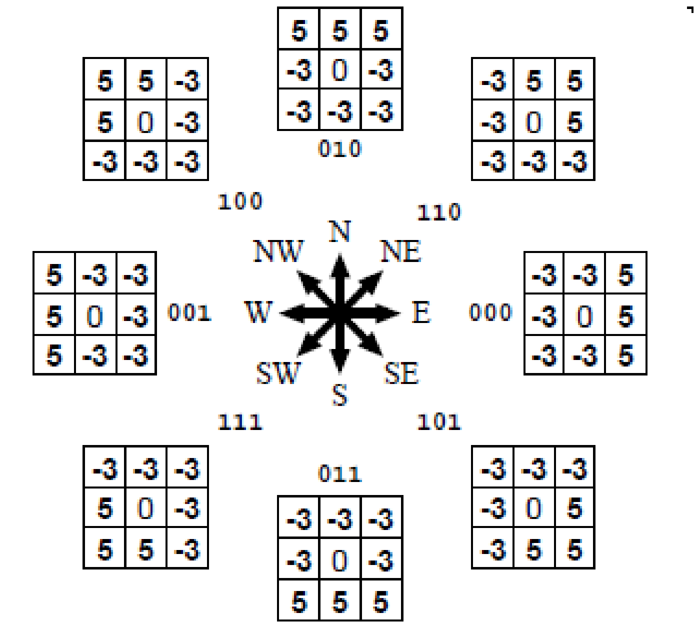
Line detection

- Goal: to detect a sequence of points aligned along a straight line
- Each edge point in the image casts a vote in the Hough space
- Intersection of votes in Hough space corresponds to detected lines
- Visual example

Line detection

- Local edge linking
  - Start from some arbitrary edge point
  - Search for points with similar edge direction in the neighborhood of that point
  - If such similar point is found, add the point to the current set of edges and repeat search from this point

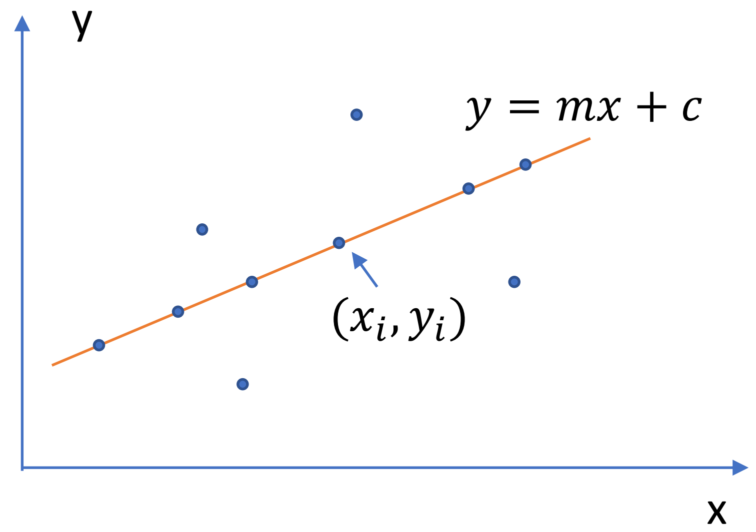
Line detection



- Gradient partitioning
  - Use a gradient operator to partition the image into edges of different orientations.
    - E.g., the Kirsch operator can be used to group the pixels into 8 directions
  - Group pixels of similar orientation in clusters (connected components)
  - For each group, find the best line that fits the data set

Hough transform: line detection

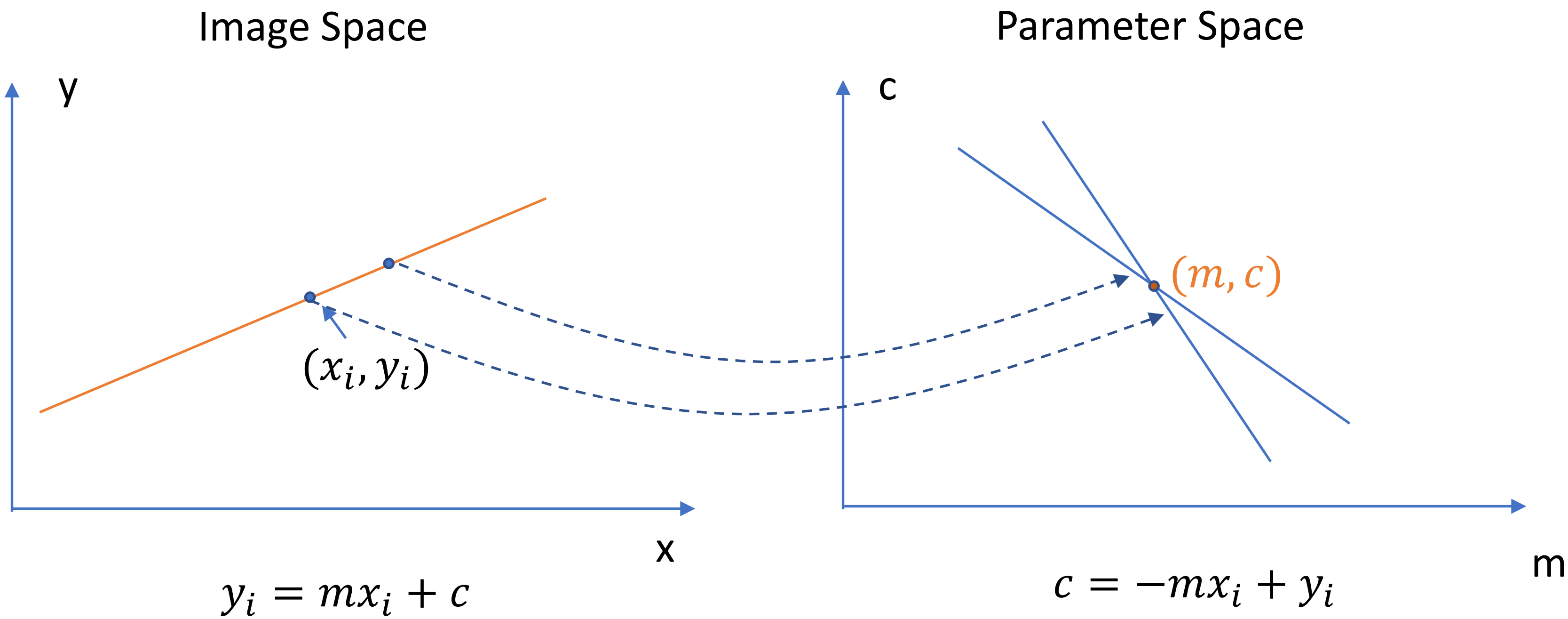
- Given edge points
- Task: detect line $y = mx + c$



- Consider point (x_i, y_i)

$$y_i = mx_i + c \quad \Longleftrightarrow \quad \boldsymbol{c} = -\boldsymbol{m}x_i + y_i$$

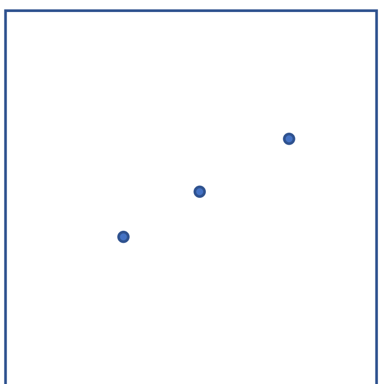
Hough transform: concept



Line detection algorithm

- Step 1: quantize parameter space (m, c)
- Step 2: create accumulator array $A(m, c)$
- Step 3: set $A(m, c)$ = 0 for all (m, c)
- Step 4: for each edge point,
  - $A(m, c) = A(m, c) + 1$
- Step 5: find local maxima in $A(m, c)$

Image



$A(m, c)$

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

$A(m, c)$

1	0	0	0	1
0	1	0	1	0
1	1	3	1	1
0	1	0	1	0
1	0	0	0	1

Multiple line detection

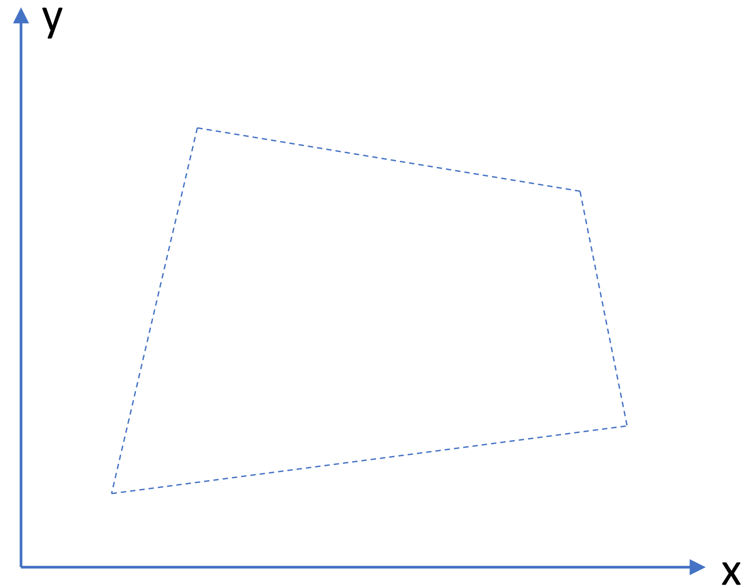
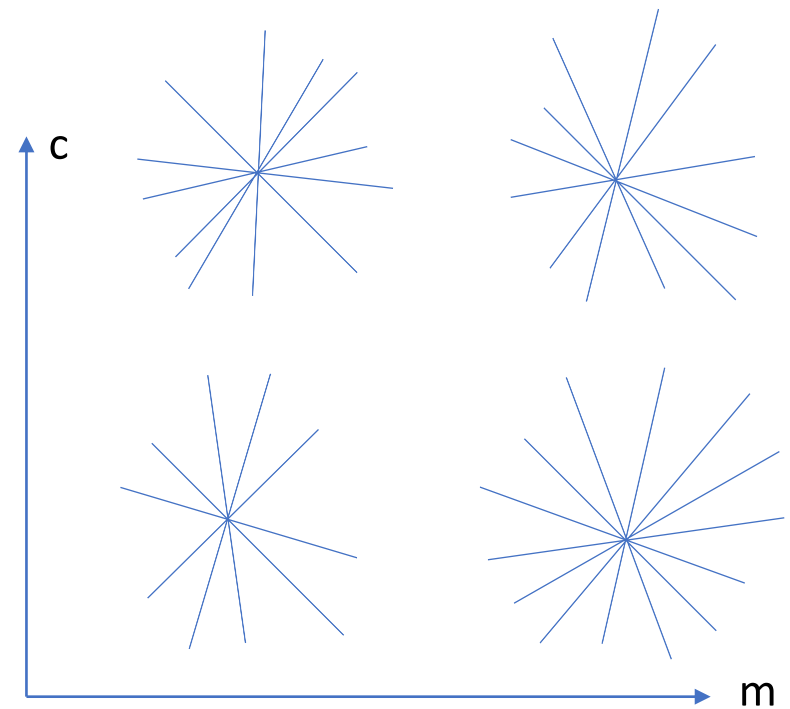


Image space



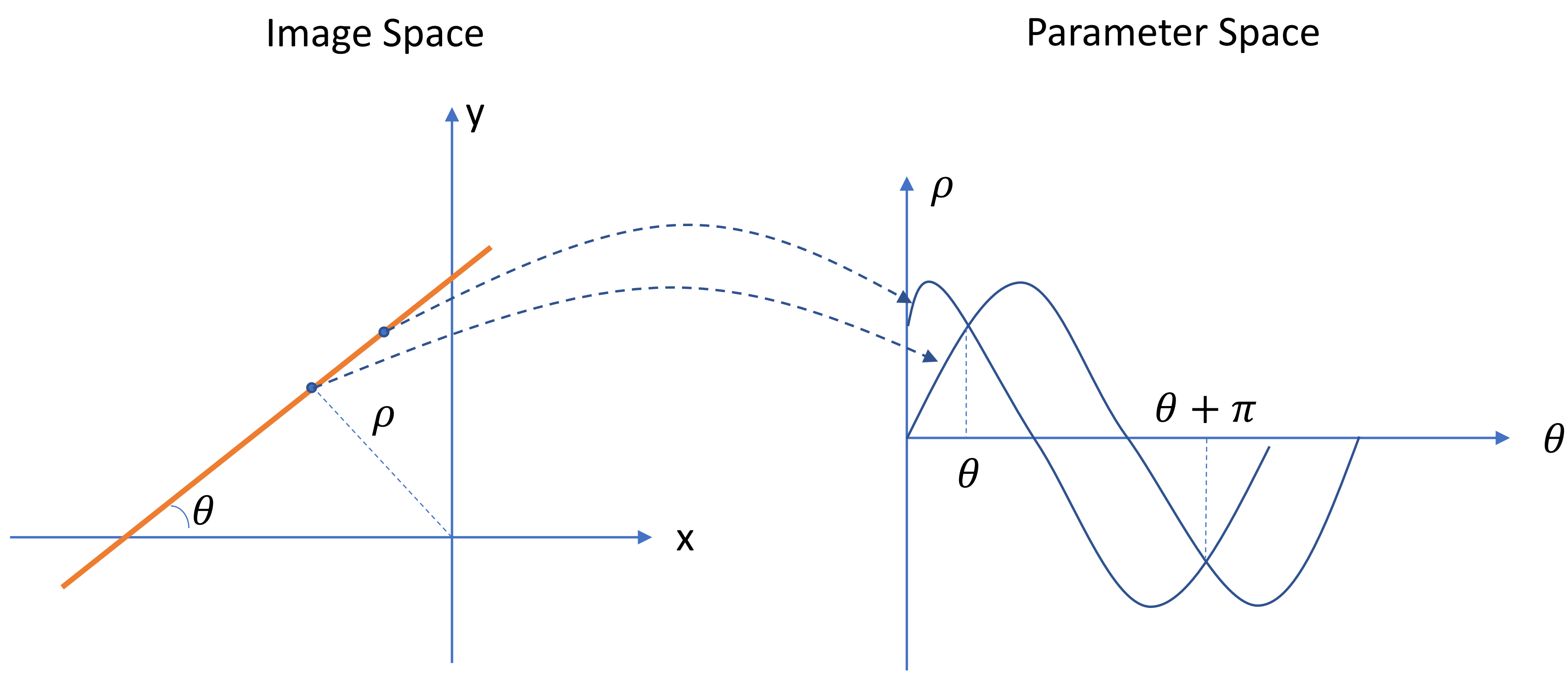
Parameter space

Better parameterization

- Issue: slope of the line $-\infty \leq m \leq \infty$
  - Leads to large accumulator
  - More memory and computation

- Solution: use $x sin\theta - y cos\theta + \rho = 0$
  - Orientation $0 \leq \theta < \pi$
  - Distance ρ is finite

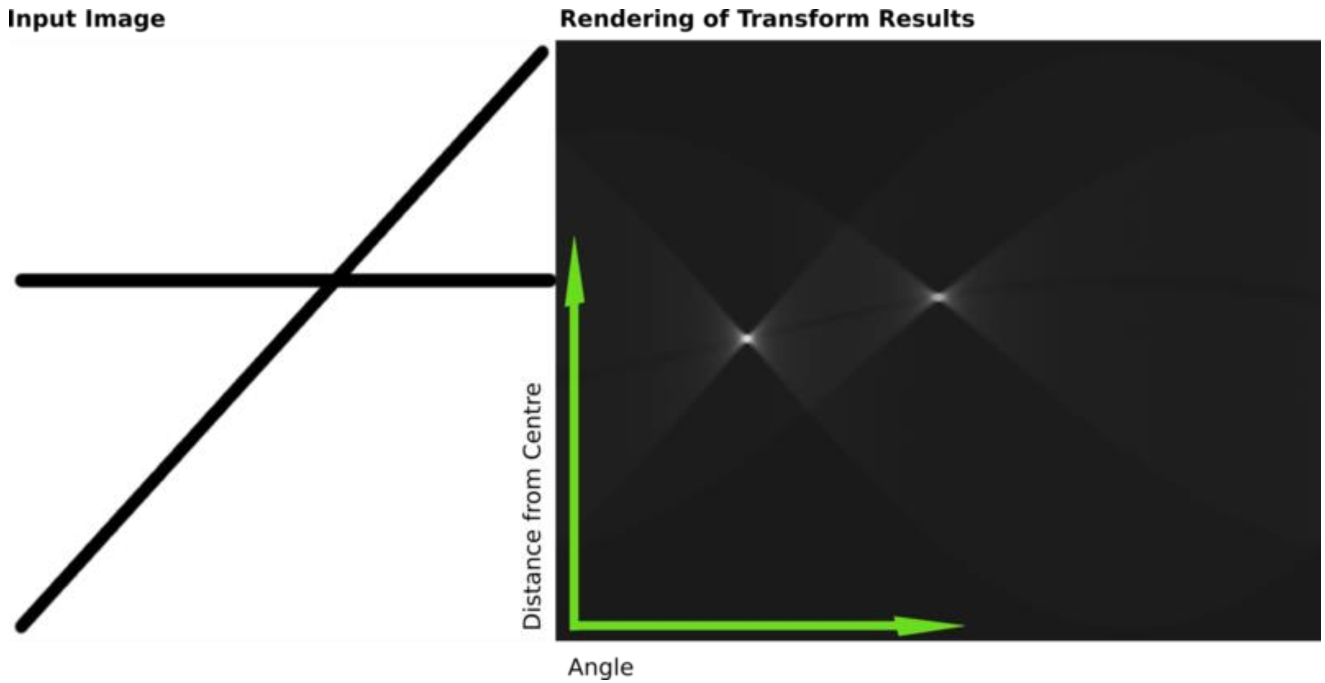
Better parameterization



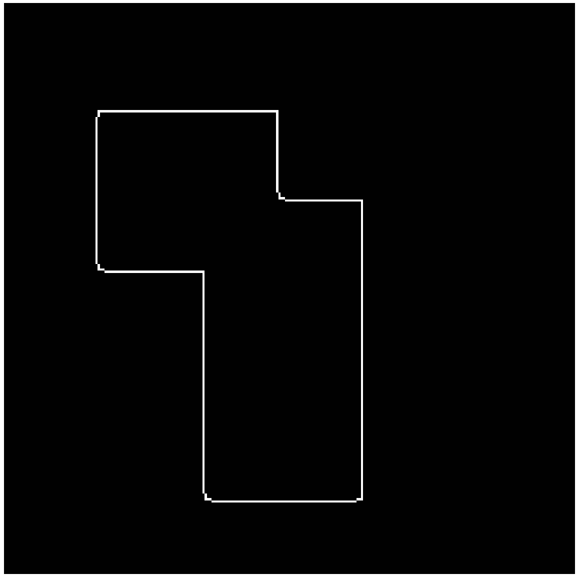
$\boldsymbol{x} sin\theta - \boldsymbol{y} cos\theta + \rho = 0$ $\qquad$ $x sin\boldsymbol{\theta} - y cos\boldsymbol{\theta} + \boldsymbol{\rho} = 0$

Hough transform

- a different example showing the results of a Hough transform on a raster image containing two thick lines.

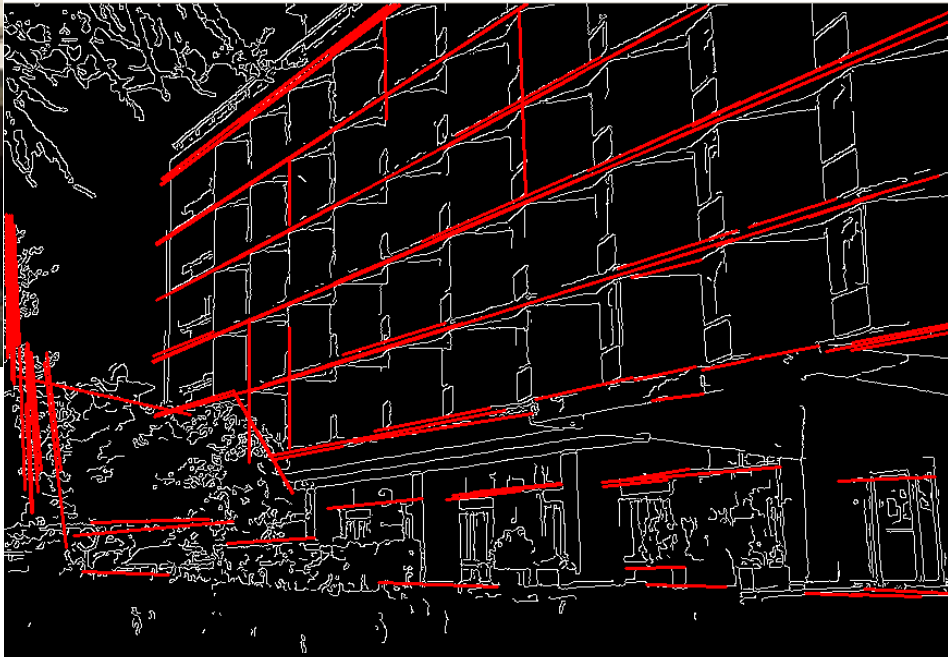


Example of Hough transform

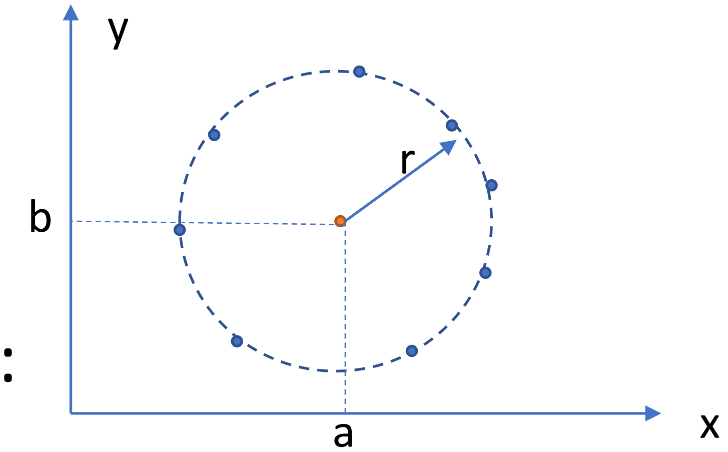


http://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm

Example of Hough transform



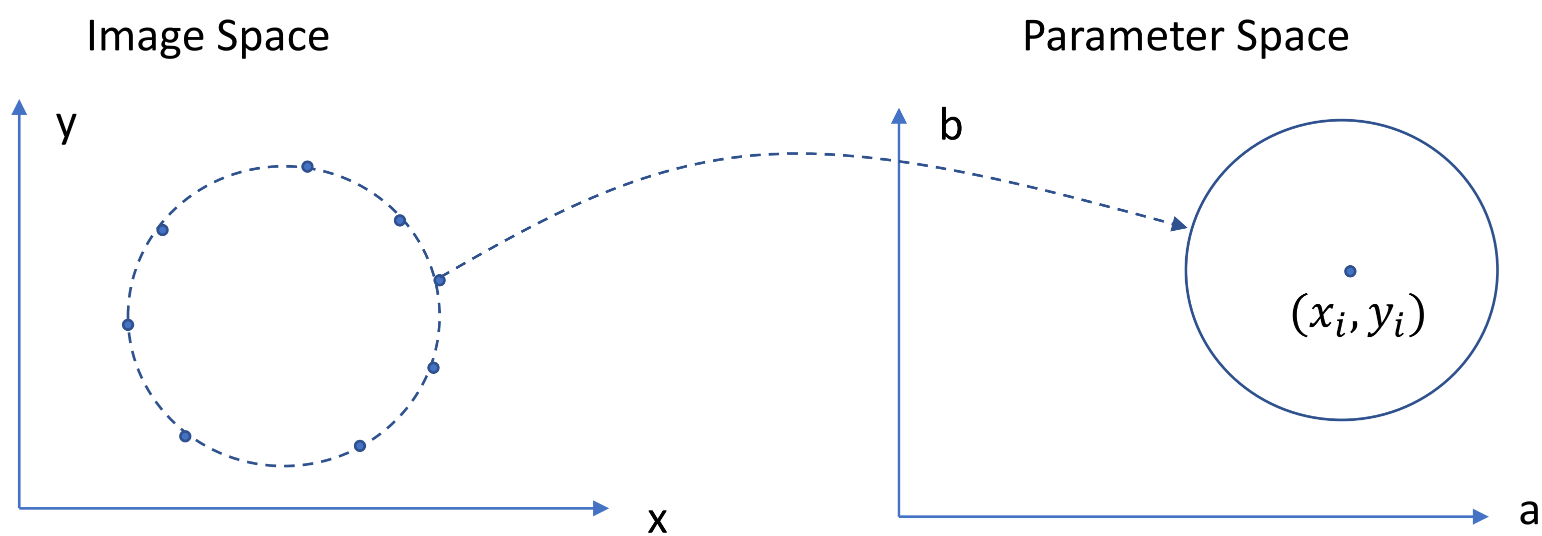
Circle detection



- Parametric equation of a circle:
  $(x-a)^2 + (y-b)^2 = r^2$
- Hough Transform for circles involves a 3D parameter space (center coordinates (a, b) and radius r)
- Explain how each edge point votes for possible circle centers and radii
- Visual example

Hough transform: circle detection

- If radius r is known: accumulator array: A(a,b)



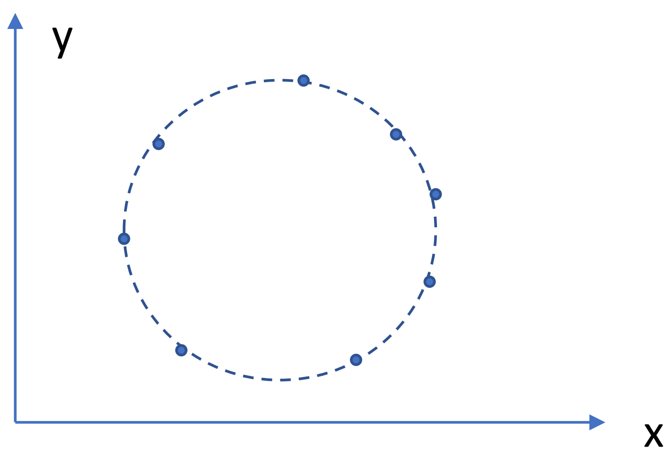
$(x_i - a)^2 + (y_i - b)^2 = r^2$

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

Hough transform: circle detection

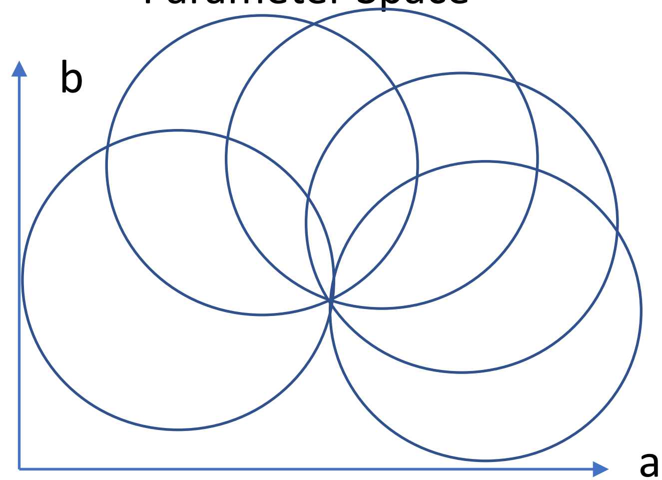
- If radius r is known: accumulator array: A(a,b)

Image Space



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

Parameter Space

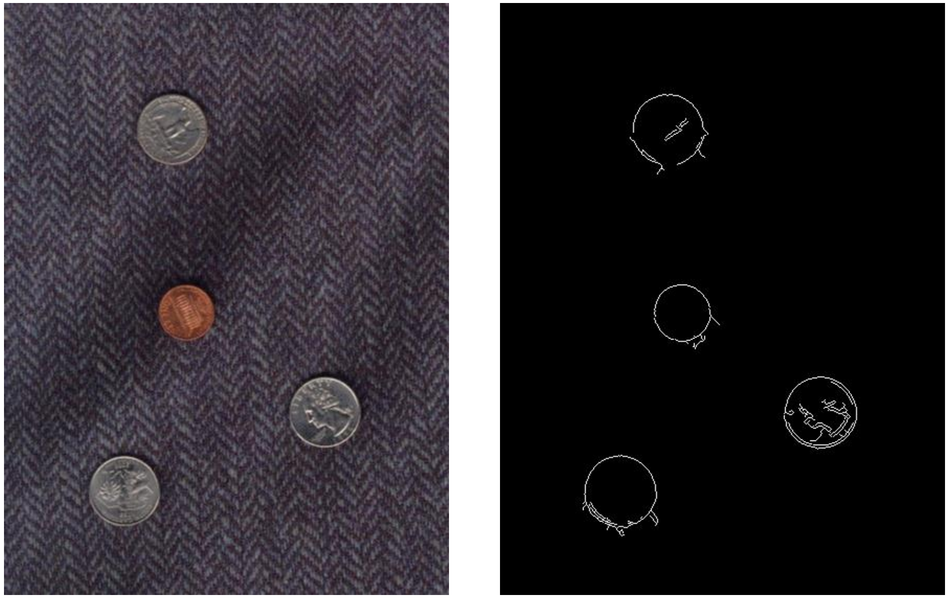


$$(a - x_i)^2 + (b - y_i)^2 = r^2$$

Detecting Circle with given radius

Detecting Coins using Hough Transform

1. Edge detection

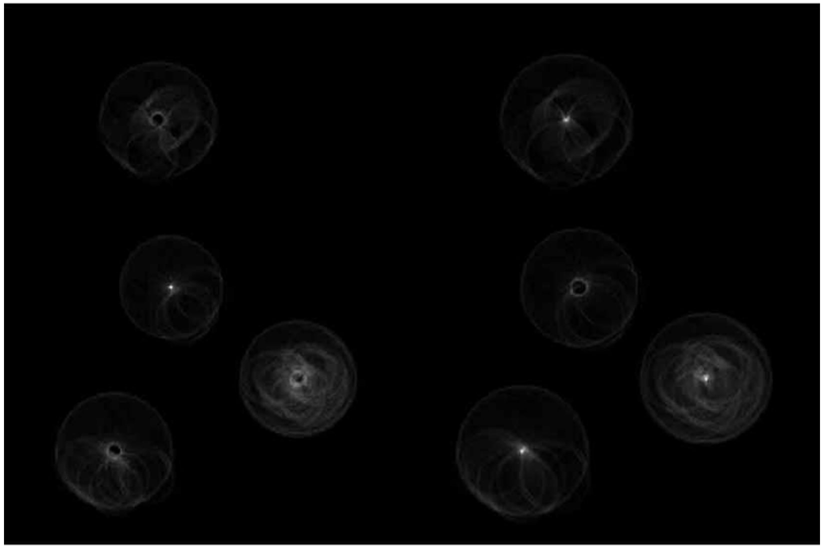


http://www-static.cc.gatech.edu/~kwatra/computer_vision/coins/coins.html

Extension of Hough Transform

2. Hough transform

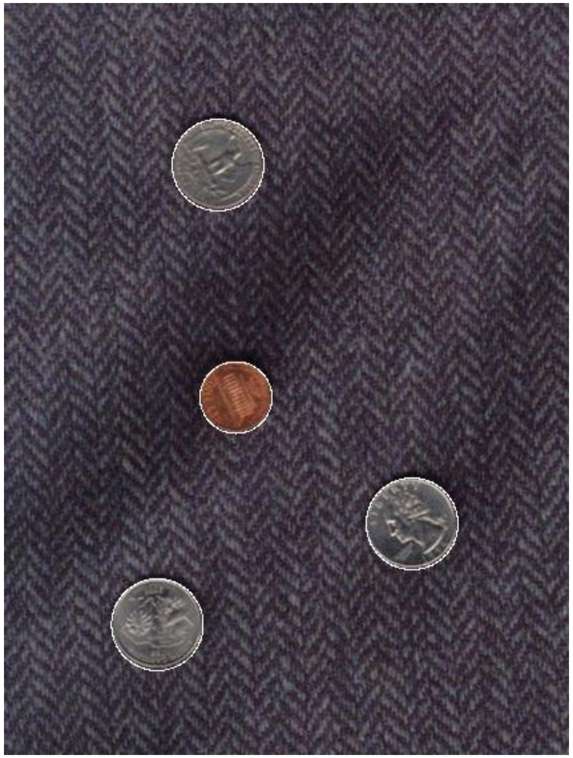
- the parametric equation is $(x-a)^2 + (y-b)^2 = r^2$
- Left for Penny, Right for Quarters



http://www-static.cc.gatech.edu/~kwatra/computer_vision/coins/coins.html

Extension of Hough Transform

3. Detected Coins



http://www-static.cc.gatech.edu/~kwatra/computer_vision/coins/coins.html

Strengths and weaknesses

- Advantages
  - Effective for detecting regular shapes like lines and circles
  - Robust against noise and gaps in edges
- Limitations
  - Computationally expensive, especially for higher dimensional parameter spaces
  - Detection can be sensitive to parameter settings (e.g., thresholds)