Hough Transform

Computer Vision

Adduru U G Sankararao, IIIT Sri City

Line Fitting

- We have already seen a couple of line fitting algorithms: *Least squares fit* and *RANSAC*
- How do they perform when multiple lines are present?

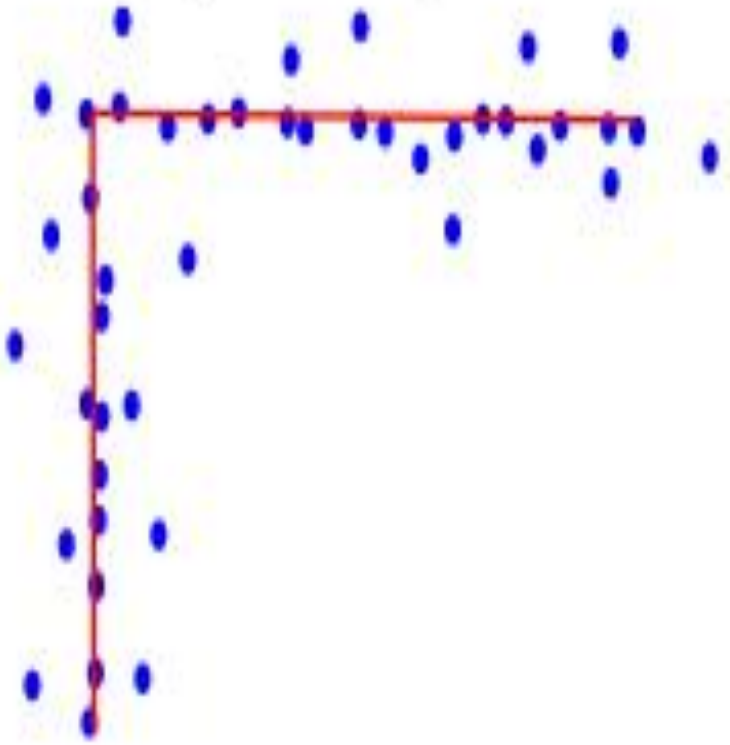
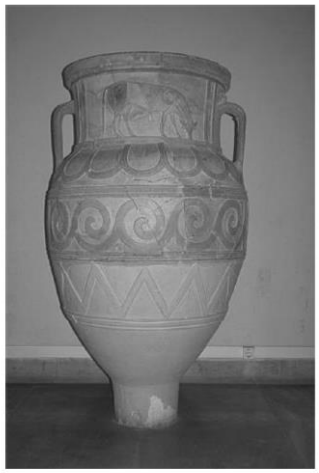
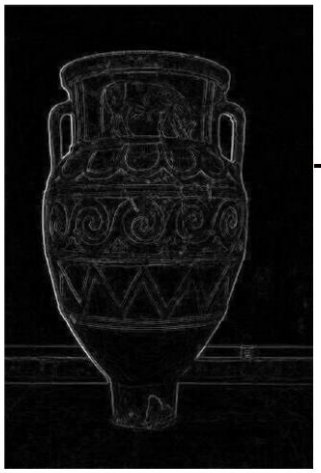


Figure 1: Example line configuration in an image.

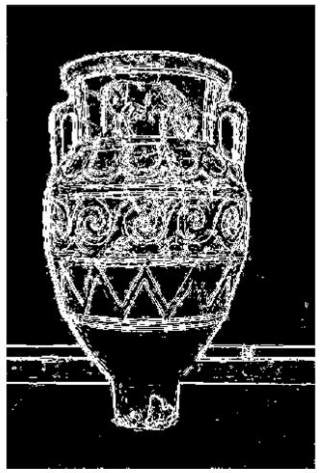
Preprocessing Edge Images



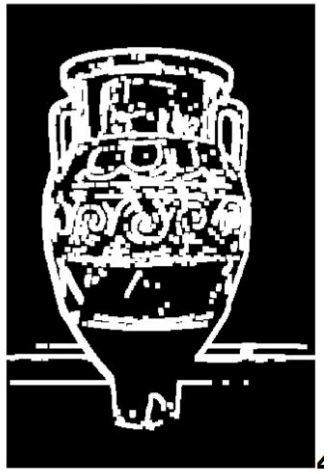
Edge Detection



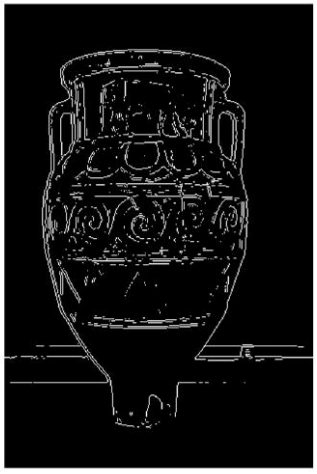
Thresholding



Shrink & Expand



Thinning



Boundary Detection

Manually Sketched

Difficulties for Fitting Approaches

- Extraneous Data: Which points to fit to?
- Incomplete Data: Only part of the model is visible.
- Noise

Solution: Hough Transform

Line Fitting: Hough Transform

- Hough, *Method and means for recognizing complex patterns*, U.S. Patent No. 3,069,654, Dec 1962
- Line equation in Cartesian co-ordinates is:
  - $y = mx + c$ m is slope, c is y-intercept
- Rearranging it slightly, we get:
  - $c = (-x)m + y$ which for a specific point (x_i, y_i) becomes $c = (-x_i)m + y_i$
- This can be thought of as the equation of line in parameter space; i.e in the (m, c) coordinate system with slope $-x_i$ and c-intercept y
- Each point in parameter space is a model

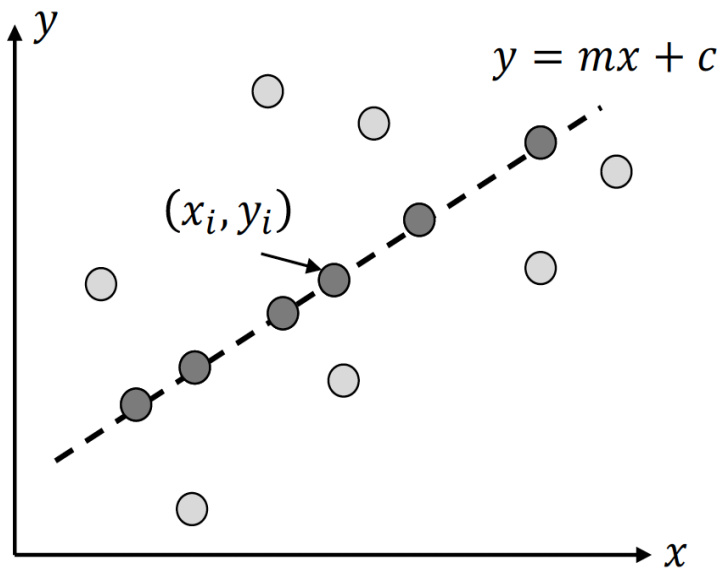
Source: Alper Yilmaz, Mubarak Shah, Fall 2011 UCF

Hough Transform: Line Detection

Given: Edge Points (x_i, y_i)

Task: Detect line

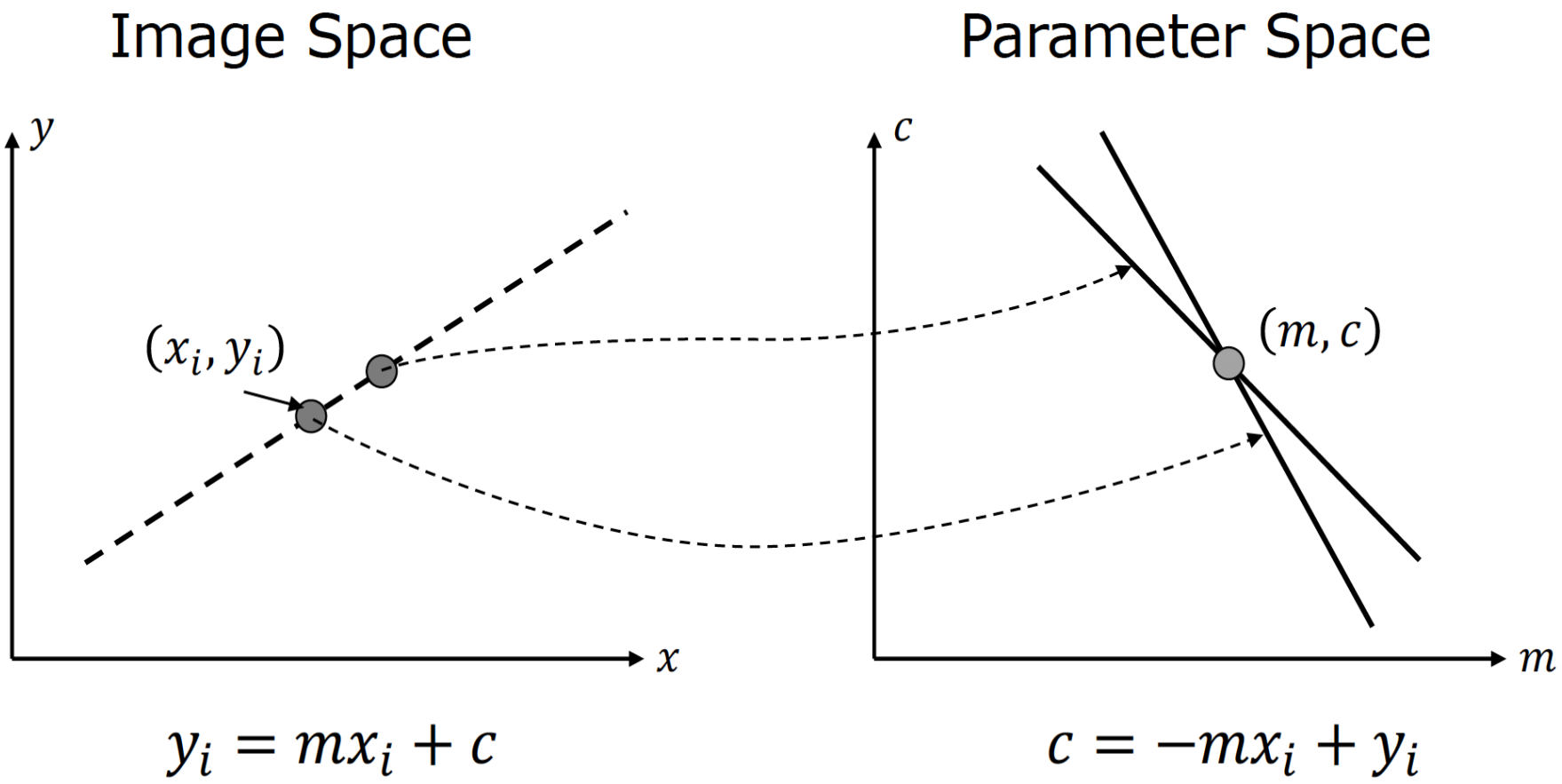
$$y = mx + c$$



Consider point (x_i, y_i)

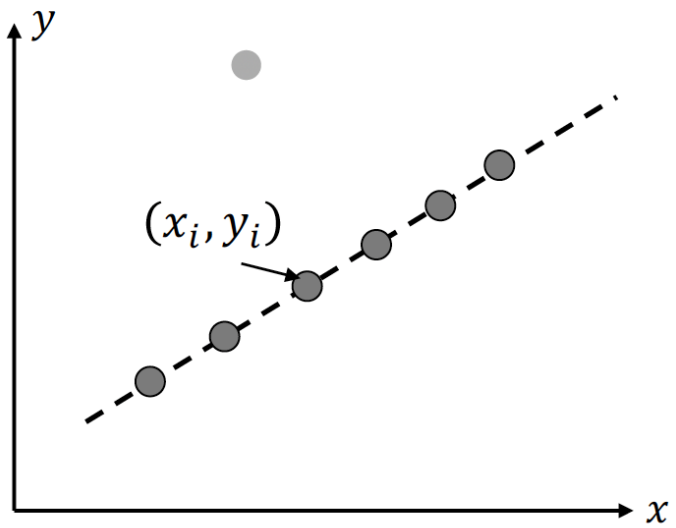
$$y_i = mx_i + c \quad \Longleftrightarrow \quad c = -mx_i + y_i$$

Hough Transform

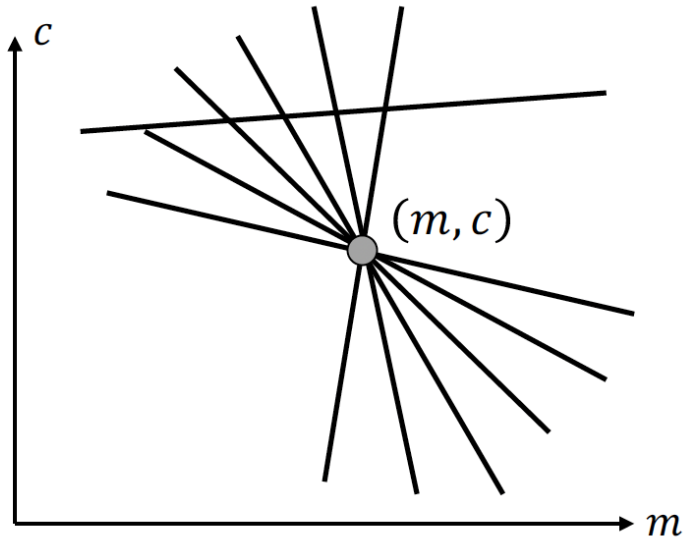


Hough Transform (HT)

Image Space



Parameter Space



$$y_i = mx_i + c$$

$$c = -mx_i + y_i$$

Point ⟷ Line

Line ⟷ Point

HT: Line Detection Algorithm

Step 1. Quantize parameter space (m, c)

Step 2. Create accumulator array $A(m, c)$

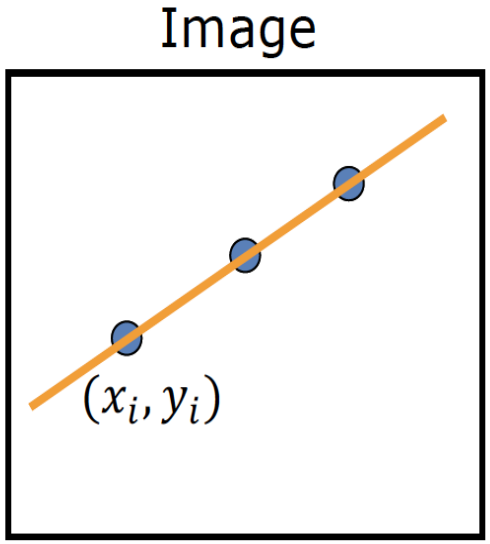
Step 3. Set $A(m, c) = 0$ for all (m, c)

Step 4. For each edge point (x_i, y_i),

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

if (m, c) lies on the line: $c = -mx_i + y_i$

Step 5. Find local maxima in $A(m, c)$



$A(m, c)$

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

m

Hough Transform

- N samples needed to fit a model (2 points to fit a line)
- But even one sample brings some information
- In the space of all possible models, vote for ones that satisfy a given sample
- Collect votes for all samples, and seek for consensus

Multiple Line Detection

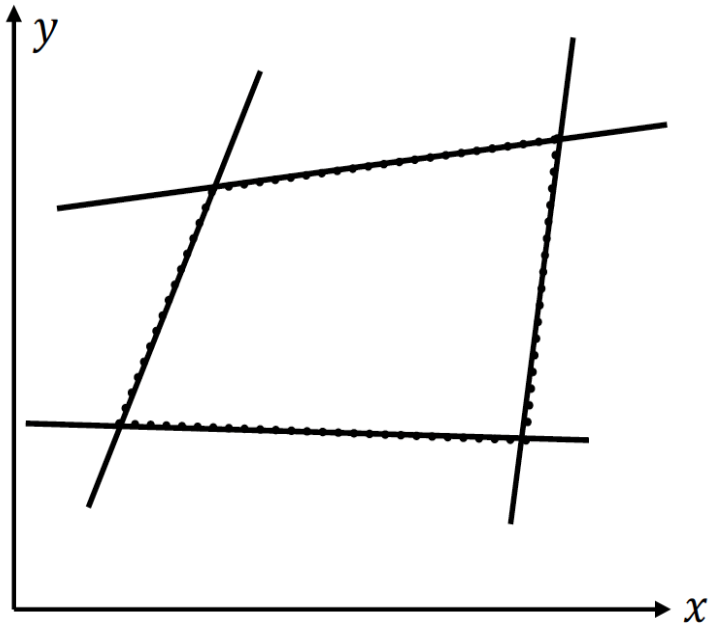
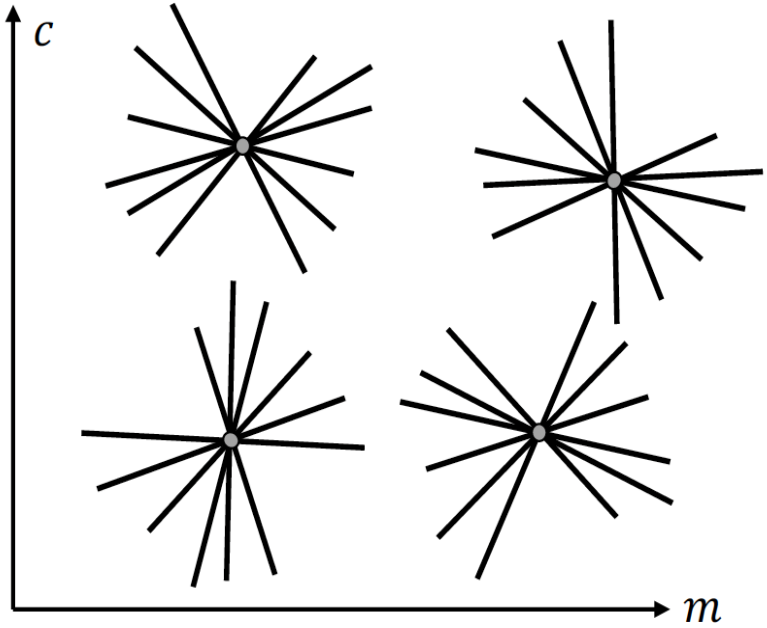


Image Space



Parameter Space

HT: Better Parameterization

Issue: Slope of the line $-\infty \leq m \leq \infty$

- Large Accumulator
- More Memory and Computation

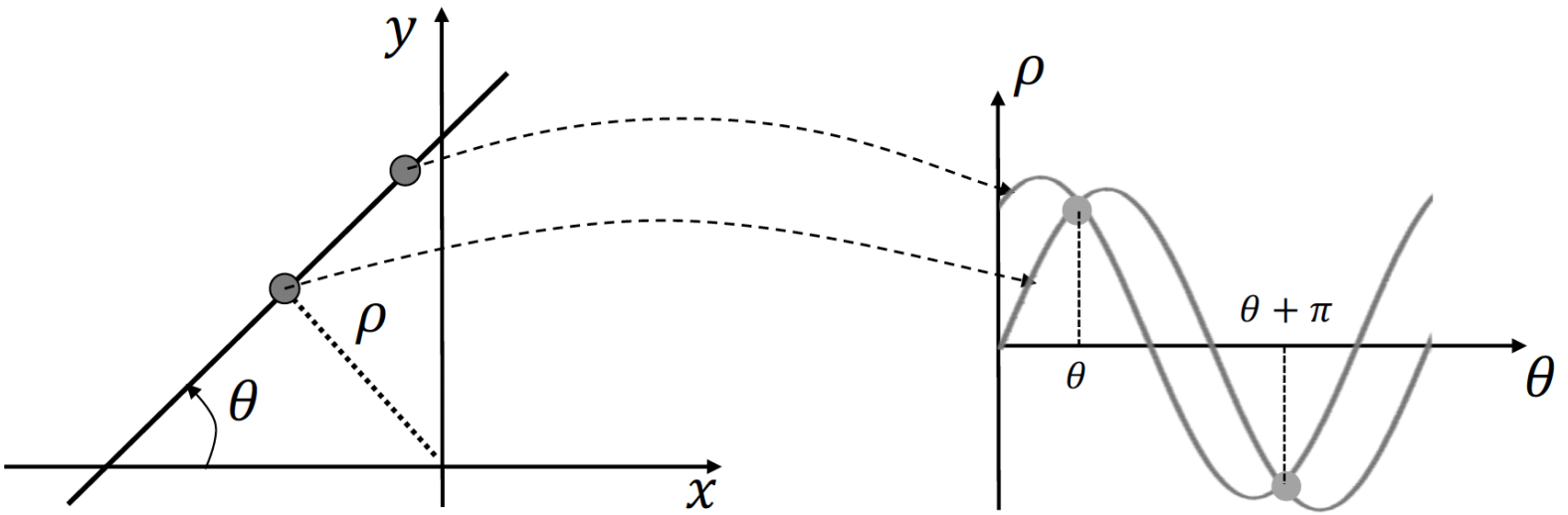
Solution: Use $x \sin\theta - y \cos\theta + \rho = 0$

- Orientation θ is finite: $0 \leq \theta < \pi$
- Distance ρ is finite

HT: Better Parameterization

Image Space

Parameter Space



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

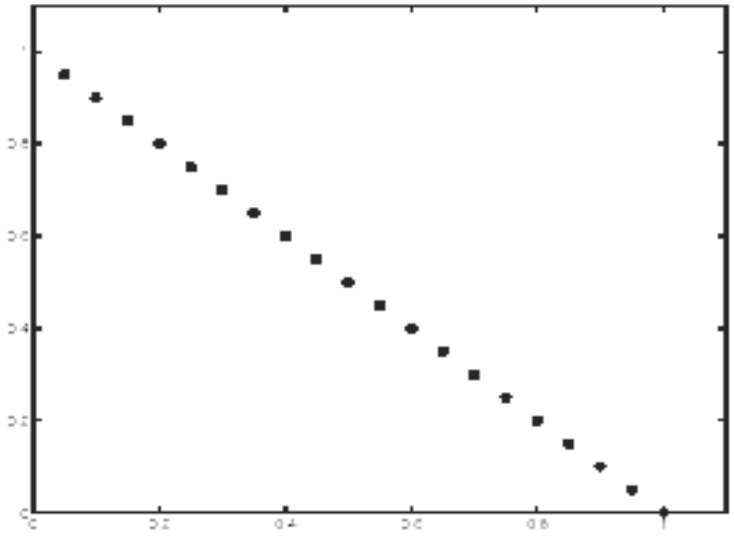
$x \sin\theta - y \cos\theta + \rho = 0$

For images: $0 \leq \theta < \pi$ and $|\rho| \leq$ Image Diagonal

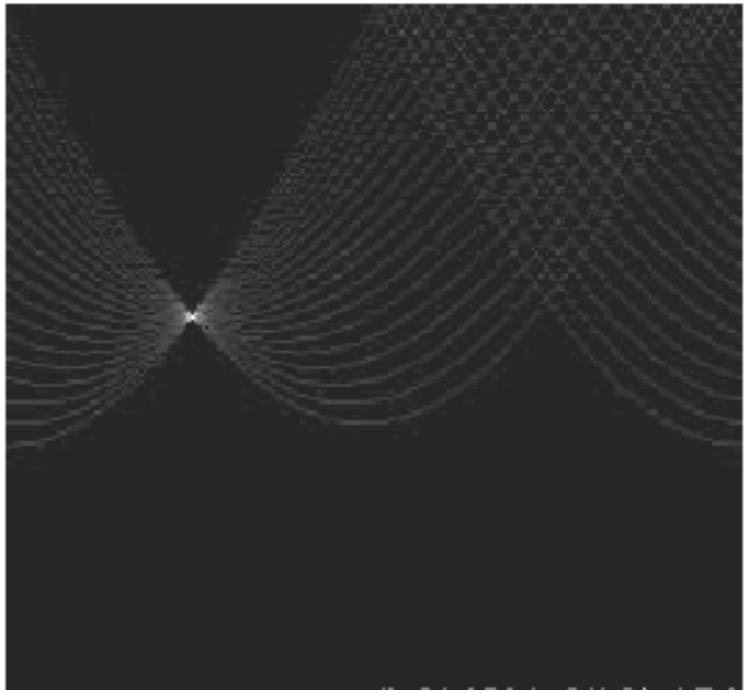
Hough Transform Mechanics

- How big should the accumulator cells be?
  - Too big, and different lines may be merged
  - Too small, and noise causes lines to be missed
- How many lines?
  - Count the peaks in the accumulator array
- Handling inaccurate edge locations:
  - Increment patch in accumulator rather than single point

Example

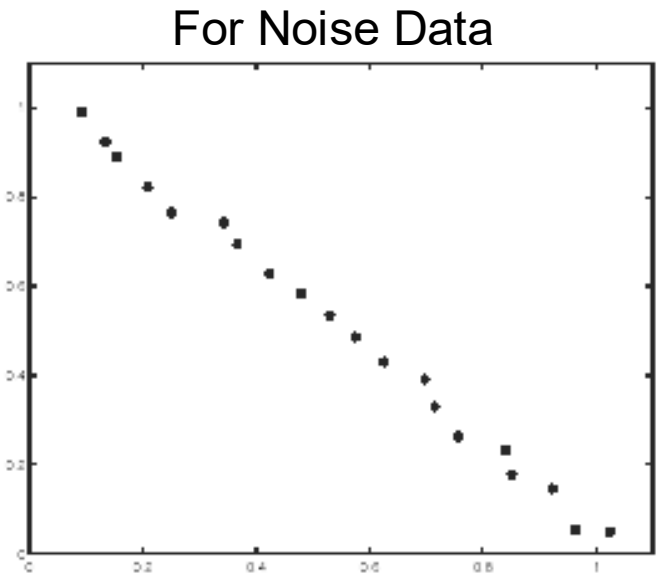


features

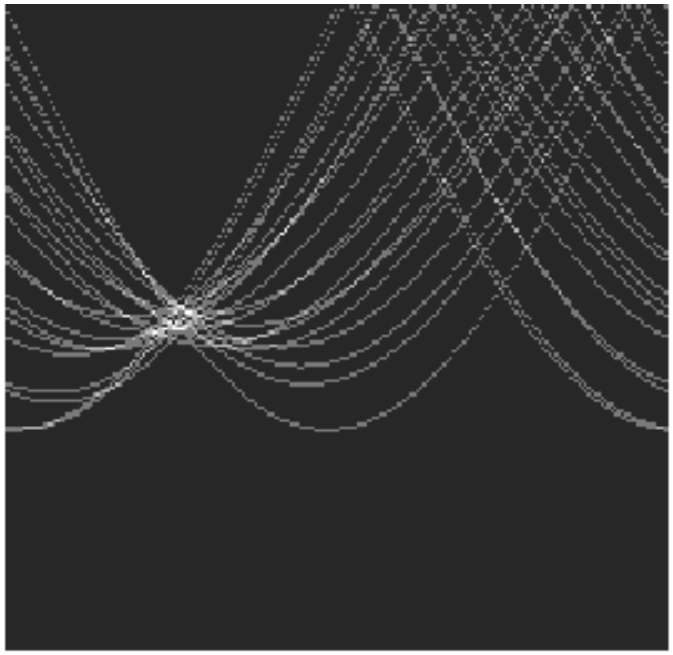


votes

For Noise Data



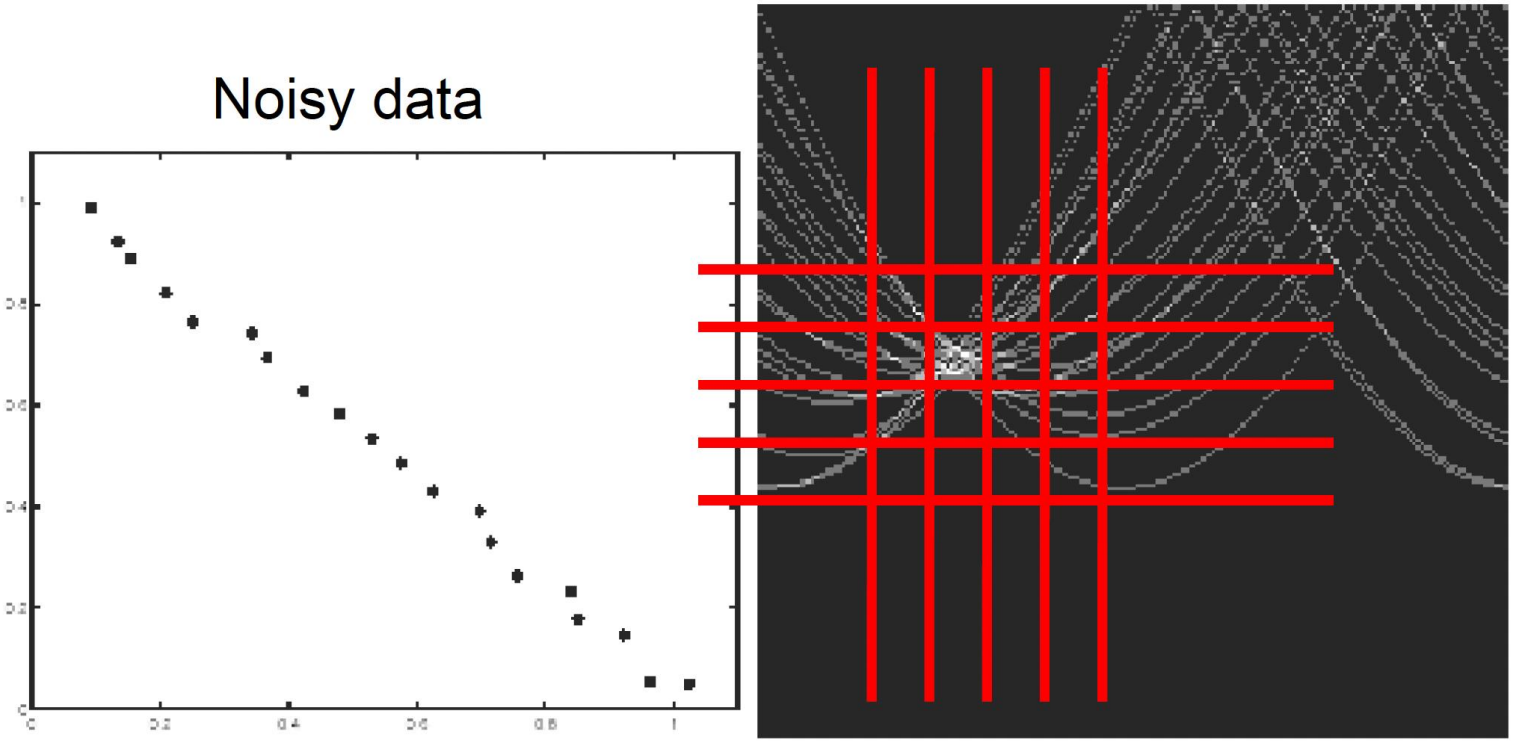
features



votes

Need to adjust grid size or smooth

For Noise Data



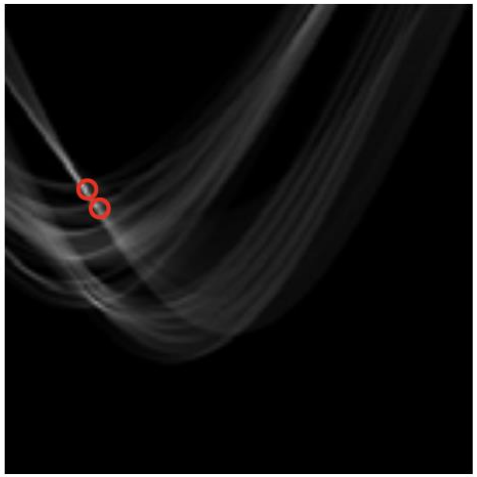
Need to adjust grid size or smooth

Line Detection Results

Original Image

Gradient

Edge (Threshold)

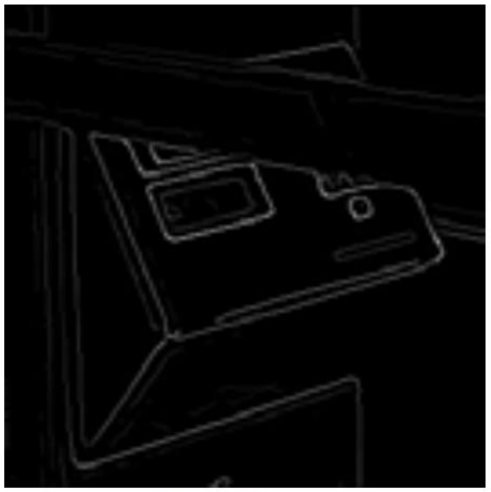


Hough Transform $A(\rho, \theta)$

Detected Lines

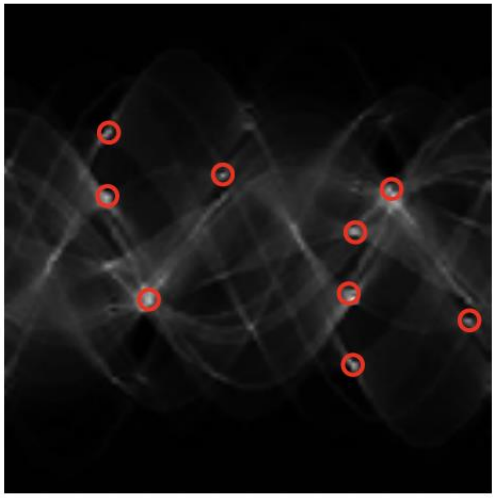
Line Detection Results

Original Image

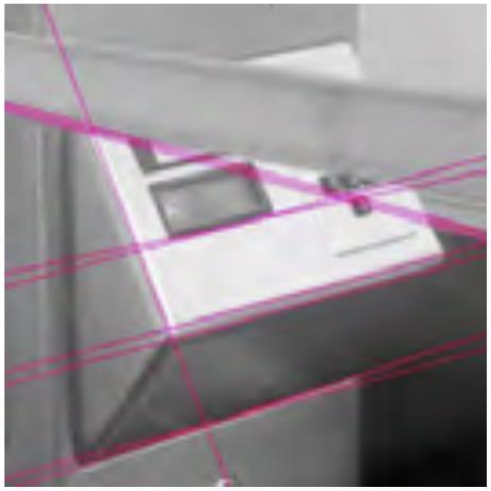


Gradient

Edge (Threshold)

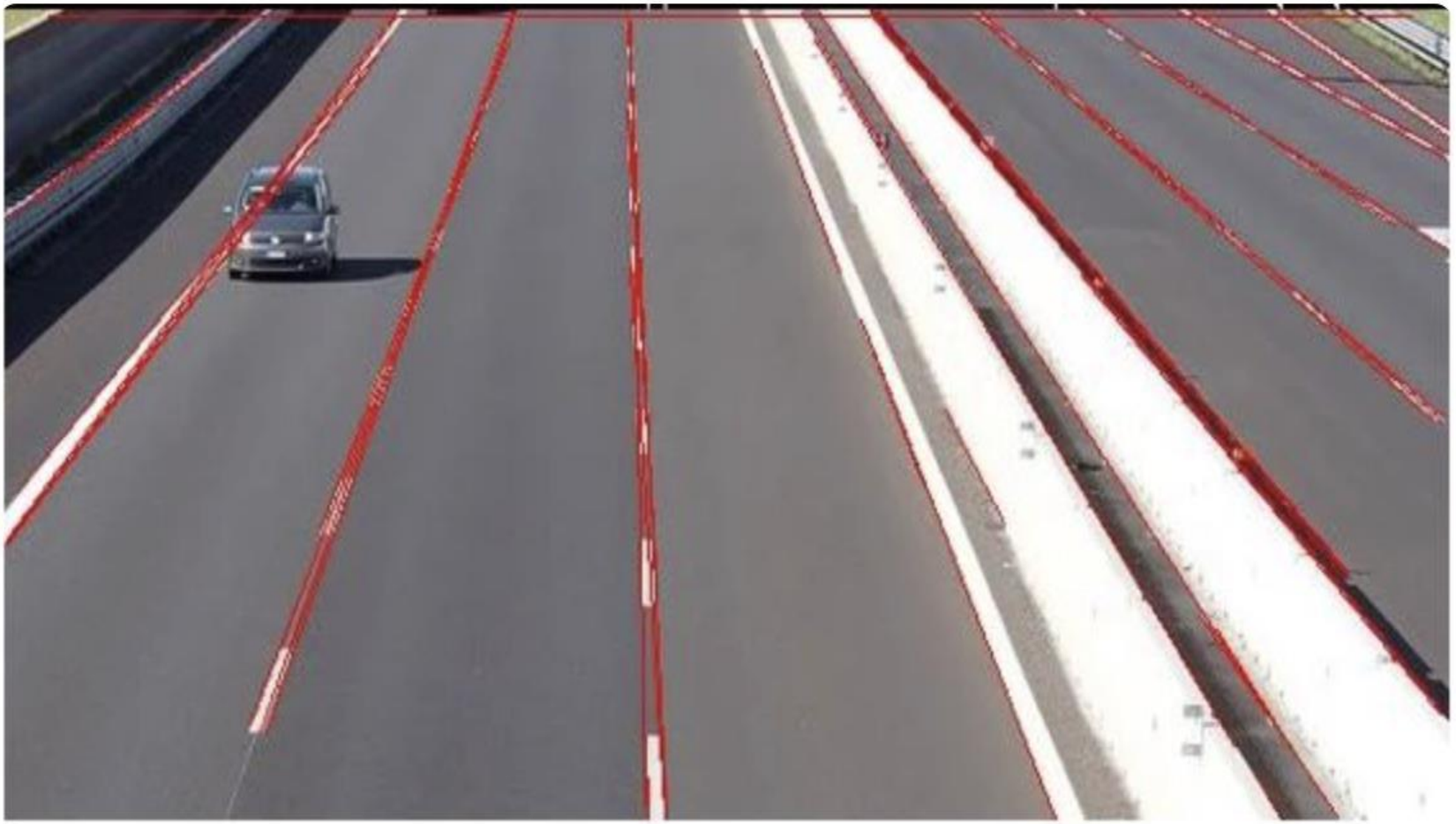


Hough Transform $A(\rho, \theta)$

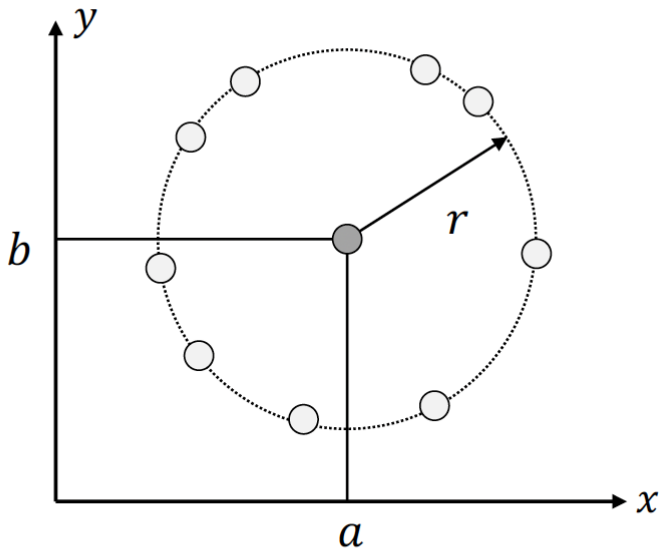


Detected Lines

Line Detection



Hough Transform: Circle Detection



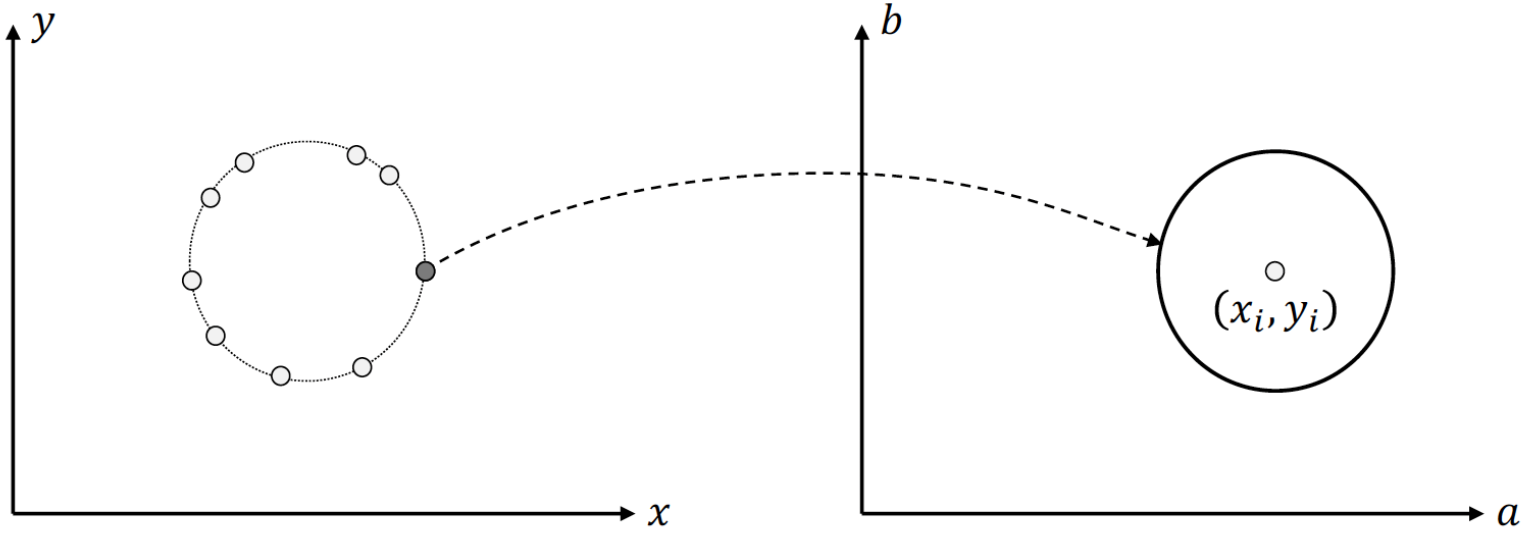
Equation of Circle: $(x_i - a)^2 + (y_i - b)^2 = r^2$

Hough Transform: Circle Detection

If radius r is known: Accumulator Array: $A(a, b)$

Image Space

Parameter Space



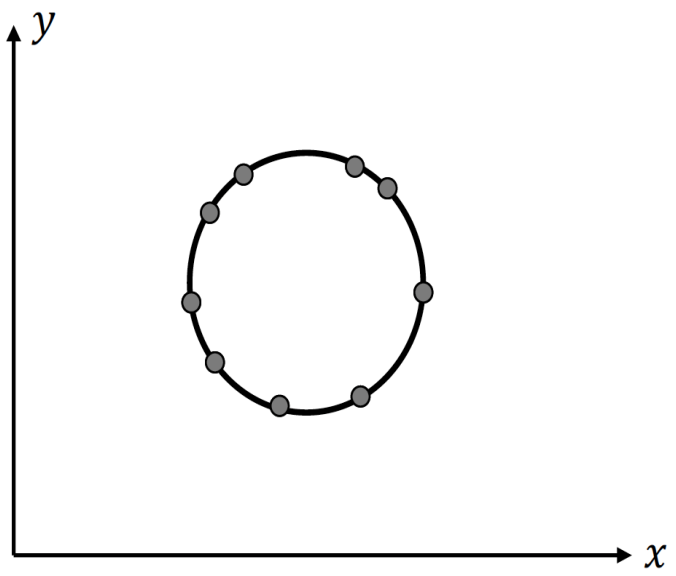
$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

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

Hough Transform: Circle Detection

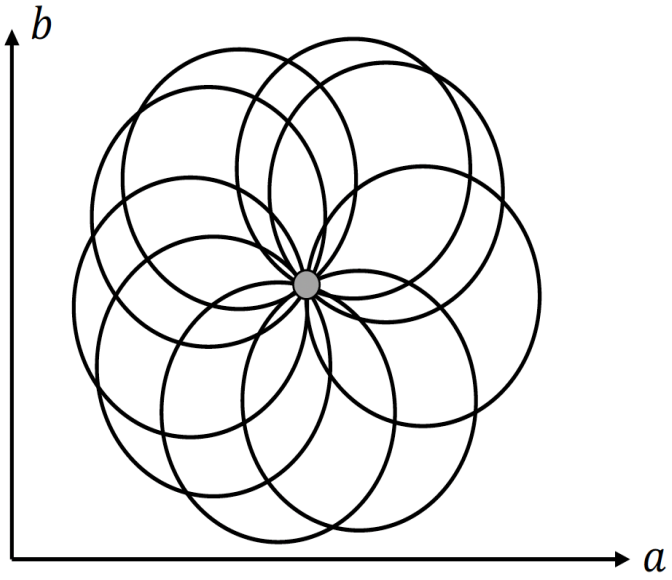
If radius r is known: Accumulator Array: $A(a, b)$

Image Space



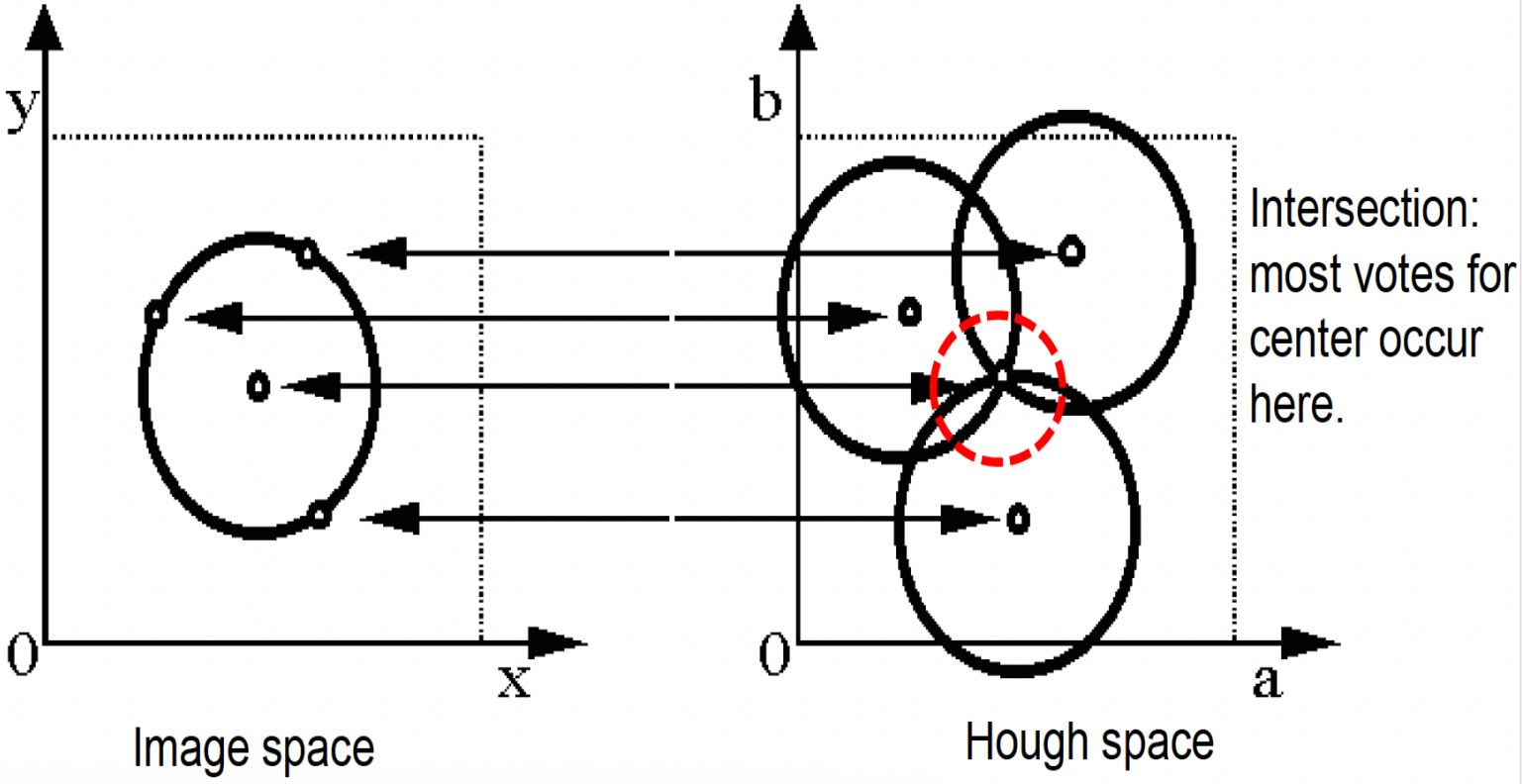
$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

Parameter Space



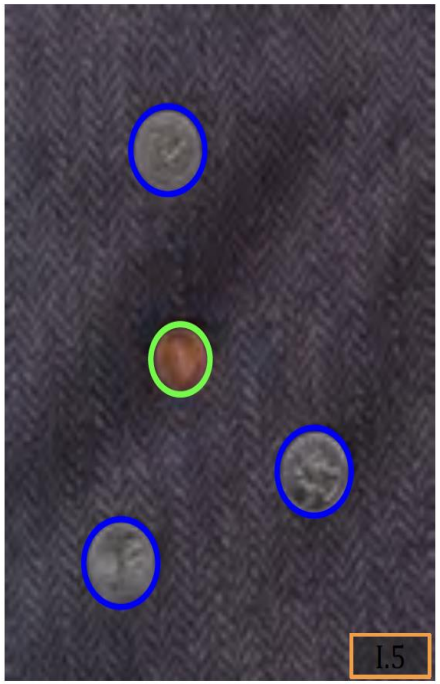
$$(a - x_i)^2 + (b - y_i)^2 = r^2$$

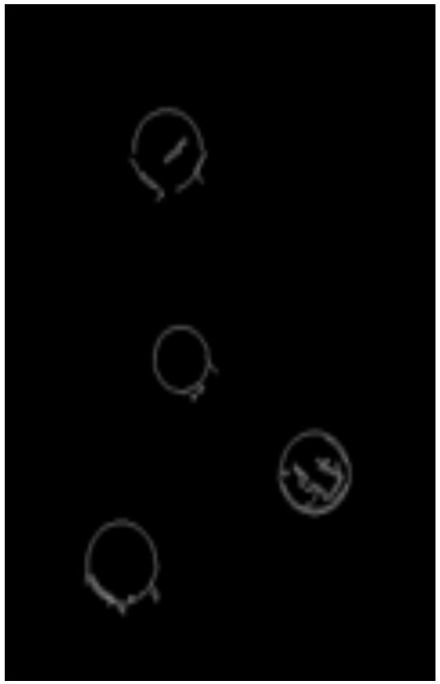
Hough Transform: Circle Detection

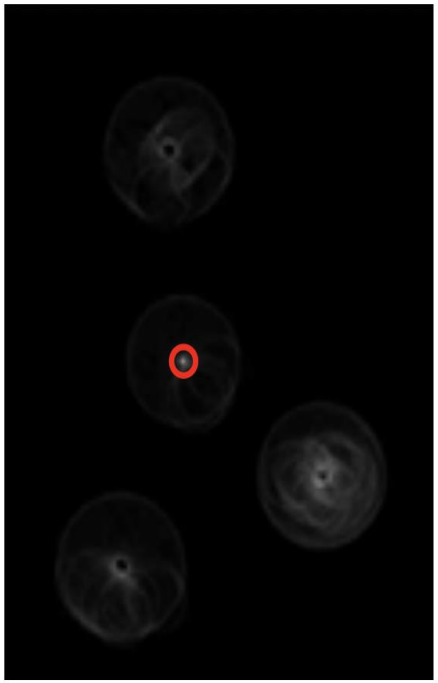


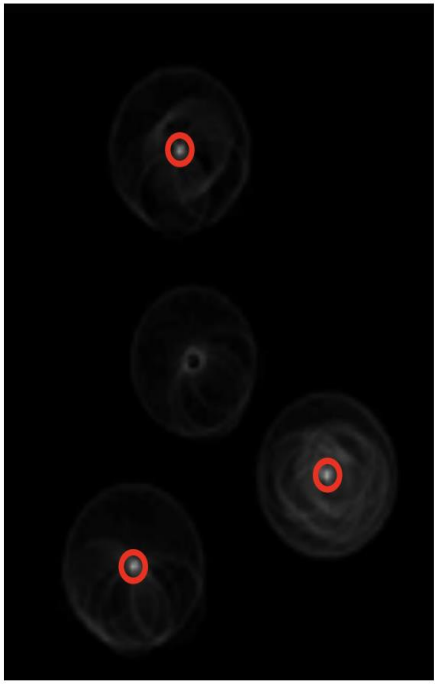
HT: Circle Detection Results

Penny $(r = r_1)$ Quarter $(r = r_2)$









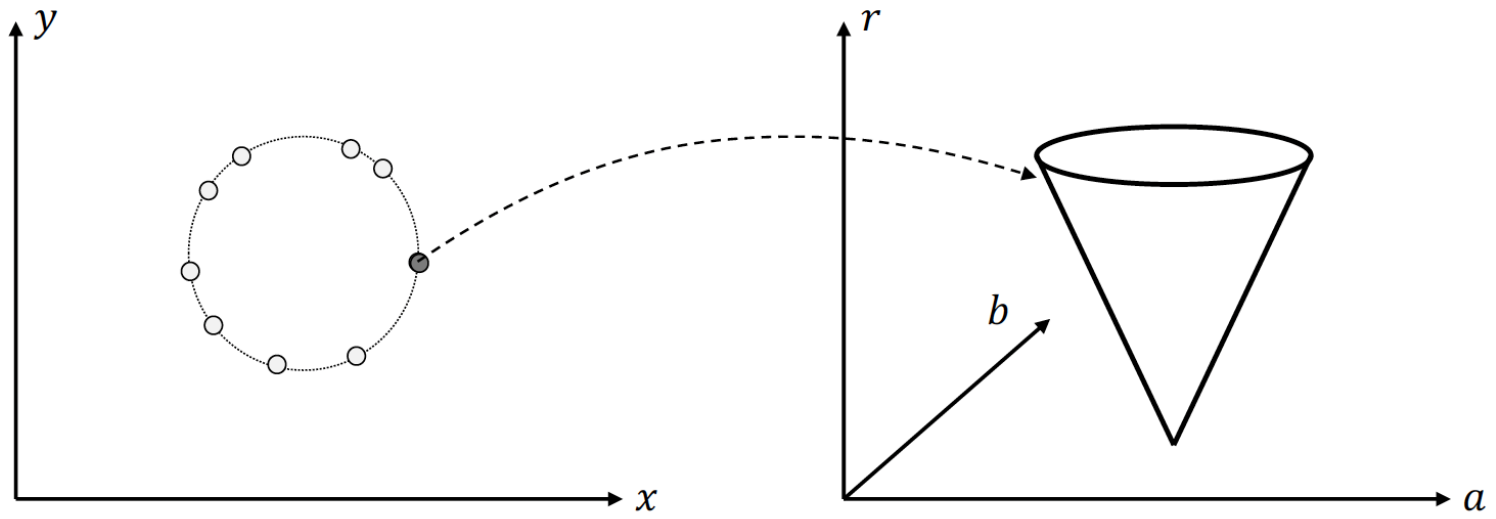
Original Image | Edge (Threshold) | Hough Transform $A_1(a, b)$ | Hough Transform $A_2(a, b)$

Hough Transform: Circle Detection

If radius r is NOT known: Accumulator Array: $A(a, b, r)$

Image Space

Parameter Space



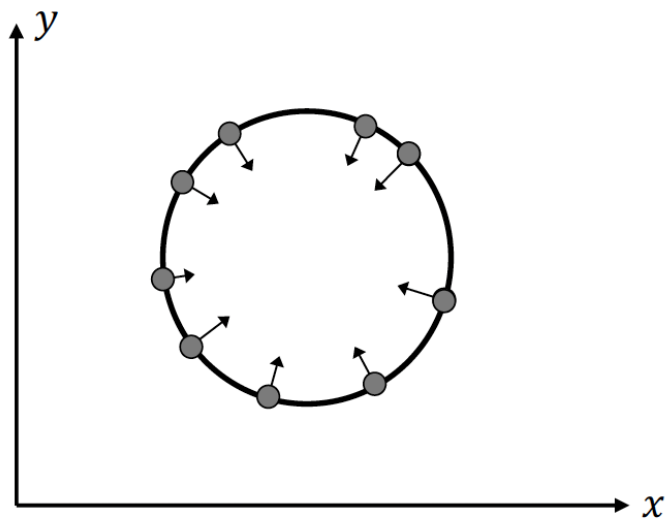
$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

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

Hough Transform: Using Gradient Information

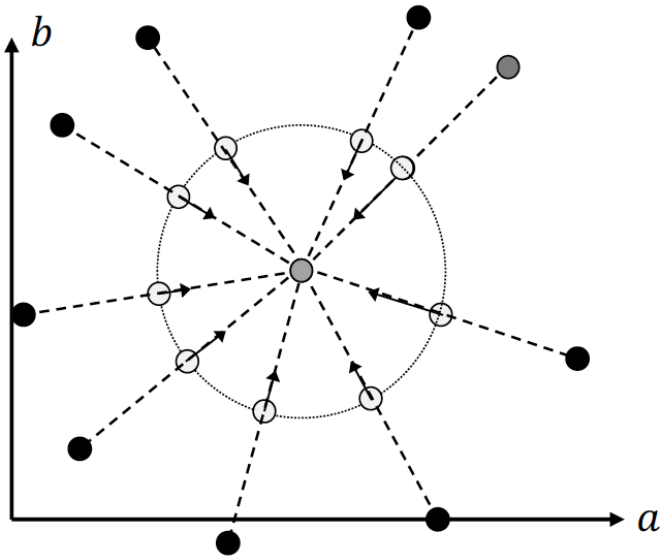
Given: Edge Location (x_i, y_i), Edge Direction φ_i and Radius r

Image Space



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

Parameter Space

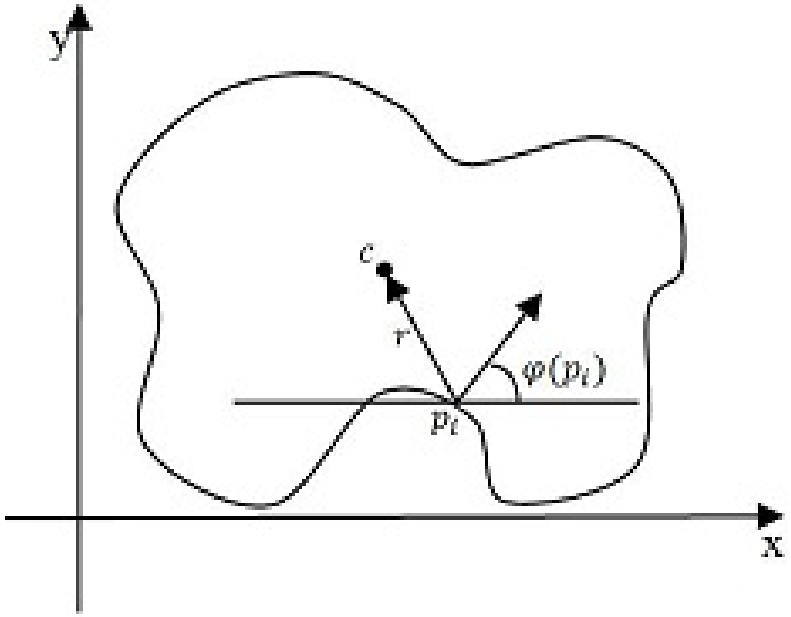


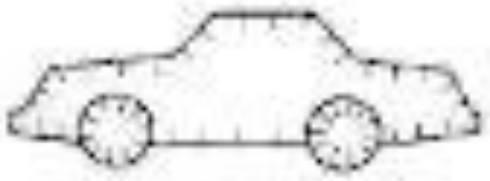
$$a = x_i \pm r\cos\varphi_i$$

$$b = y_i \pm r\sin\varphi_i$$

Need to increment only TWO points in $A(a, b)$

Generalized Hough Transform: For Arbitrary Shapes





Reading Assignment

Hough Transform: Conclusion

Advantages:

- Works on disconnected edges
- Relatively insensitive to occlusion and noise
- Robust to outliers: each point votes separately
- Effective for simple shapes (lines, circles, etc.), multiple instances

Disadvantages:

- Not suitable for more than a few parameters (grid size grows exponentially)
- Bin size trades off between noise tolerance, precision, and speed/memory. Can be hard to find sweet spot
- Setting parameters is not easy

Applications:

- Line fitting (also circles, ellipses, etc.), boundary detection
- Object instance recognition, Object category recognition

Hough Transform: Applications

Model

Model Detected

Hough Transform $A(x_c, y_c)$

Eiffel tower detection

model image

test image

Ballard. PR 1981. Generalizing the Hough Transform to Detect Arbitrary shapes.

Eiffel tower detection

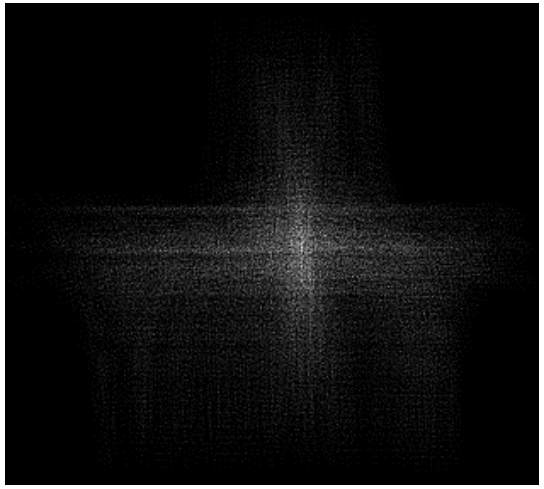
model image points

test image points

Ballard. PR 1981. Generalizing the Hough Transform to Detect Arbitrary shapes.

Eiffel tower detection

model image points



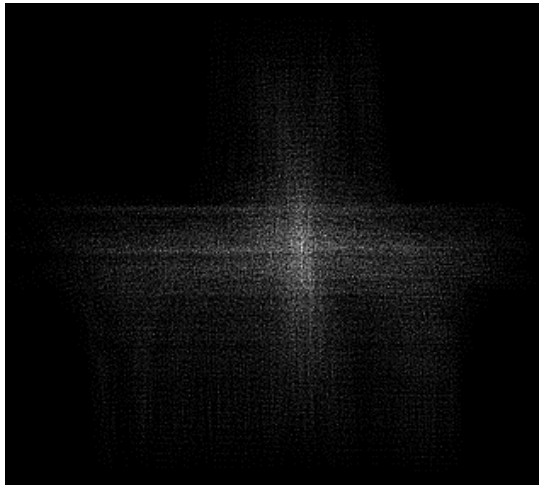
accumulator

test image points

Ballard. PR 1981. Generalizing the Hough Transform to Detect Arbitrary shapes.

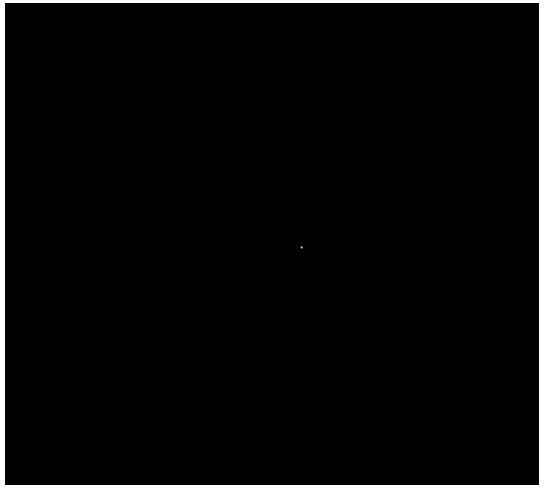
Eiffel tower detection

model image points



accumulator

test image points

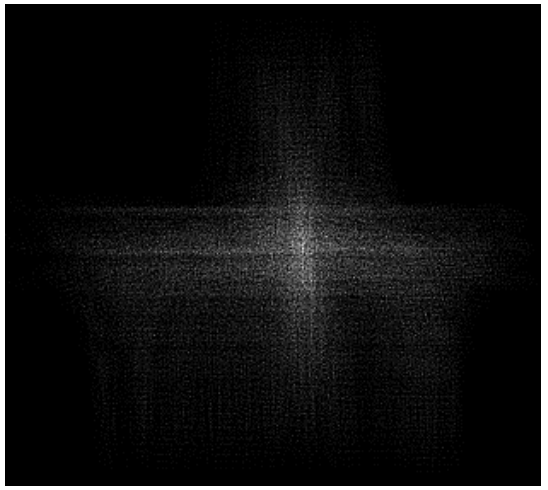


local maxima

Ballard. PR 1981. Generalizing the Hough Transform to Detect Arbitrary shapes.

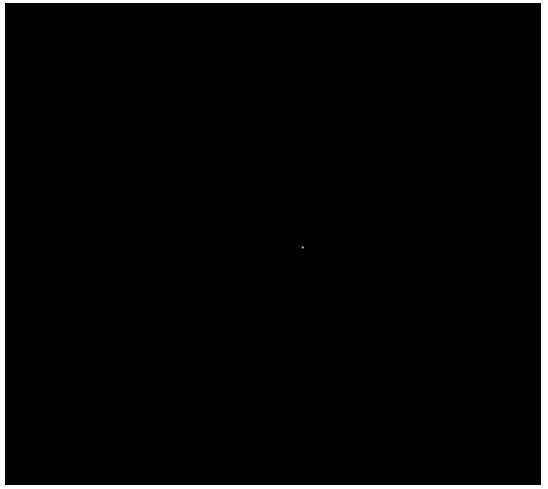
Eiffel tower detection

model image points



accumulator

detected location



local maxima

Ballard. PR 1981. Generalizing the Hough Transform to Detect Arbitrary shapes.

Acknowledgements

- Thanks to the following researchers for making their teaching/research material online
  - Shree K. Nayar
  - Vineeth N
  - Shivram Dubey