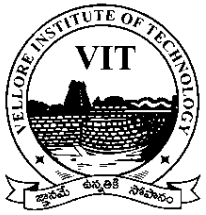


VIT-AP UNIVERSITY

Computer Vision

(Course Code: 4047)

Module-2:Lecture-3: Hough Transform

Gundimeda Venugopal, Professor of Practice, SCOPE

Hough Transform

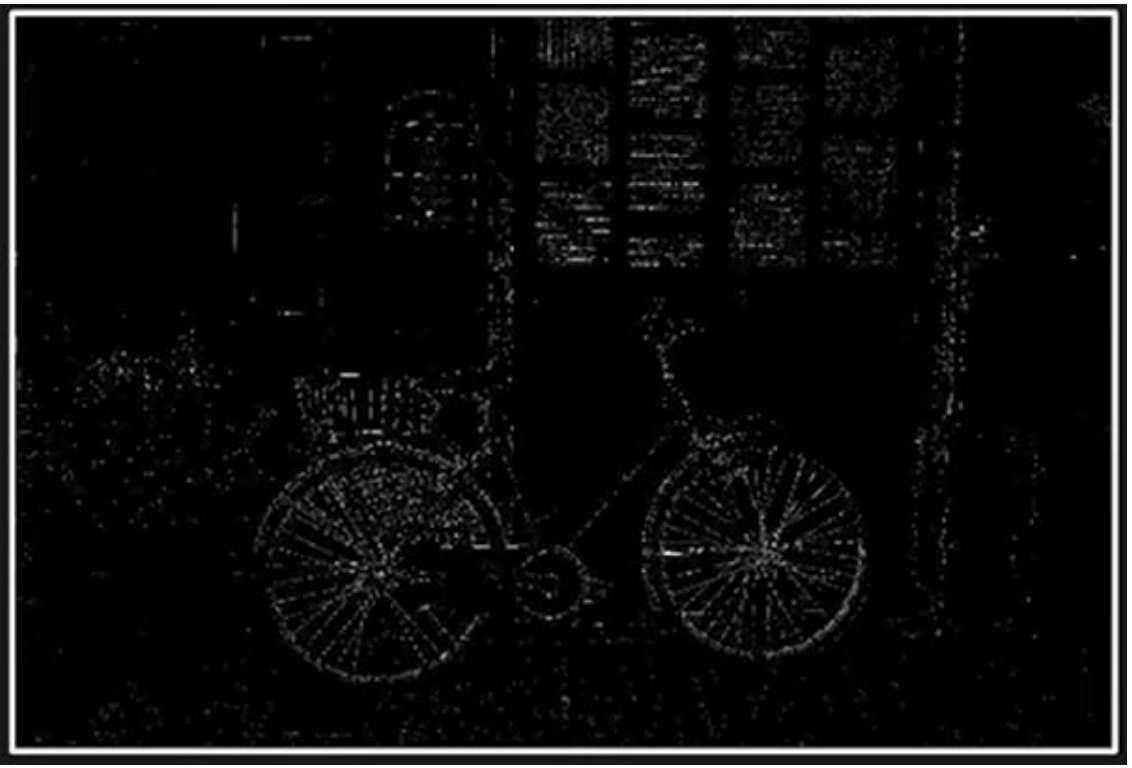
Challenge: One of the problem with boundary detection of an object in an image is knowing which edges in the image that are corresponding to the boundary we are looking for.

- Hough Transform provides an elegant solution to the problem when the boundary can described by a small number of parameters.
- The Hough Transform is an algorithm patented by Paul V. C. Hough and was originally invented to recognize complex lines in photographs (Hough, 1962).
- The algorithm has been modified and enhanced to be able to recognize other shapes such as circles and quadrilaterals of specific types.
- Perform edge detection first on the Input image (using edge detection algorithms: Canny, Sobel, Laplacian etc.) to produce an edge image which will then be used as input into the Hough Transform algorithm.

Difficulties for the Fitting Approach

Input Image

Edge Map



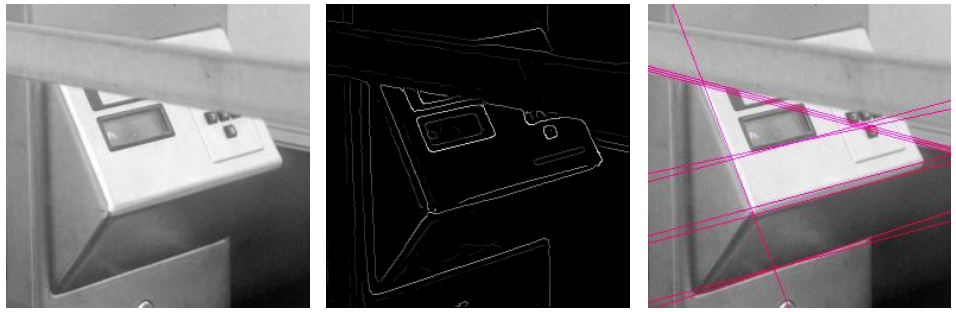
Task: Find the two wheels of the bicycle

- Extraneous Data: Which points to fit to? (e.g., to the bicycle circles,)
- Incomplete Data: Only part of the model is visible (e.g., Occlusion, Part of the wheel is incomplete.)
- Noise (e.g., edges close to the wheels which don't correspond to real edges)

Solution: Hough Transform (1962)

Fit / Associate a model with observed features

❖Want to associate a model with observed features



[Fig from Marszalek & Schmid, 2007]

For example, the model could be a line, a circle, or an arbitrary shape.

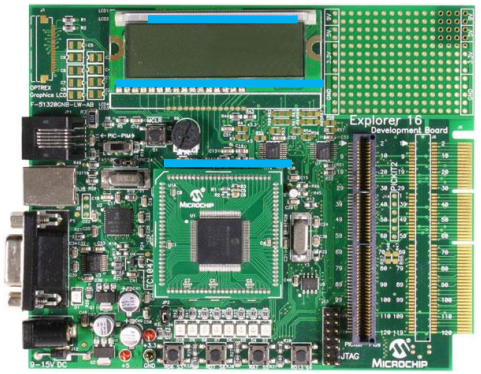
Fitting: Main idea

- Choose a parametric model to represent a set of features
- Membership criterion is not local
  - Can't tell whether a point belongs to a given model just by looking at that point
- Three main questions:
  - What model represents this set of features best?
  - Which of several model instances gets which feature?
  - How many model instances are there?
- Computational complexity is important
  - It is infeasible to examine every possible set of parameters and every possible combination of features

Slide credit: L. Lazebnik

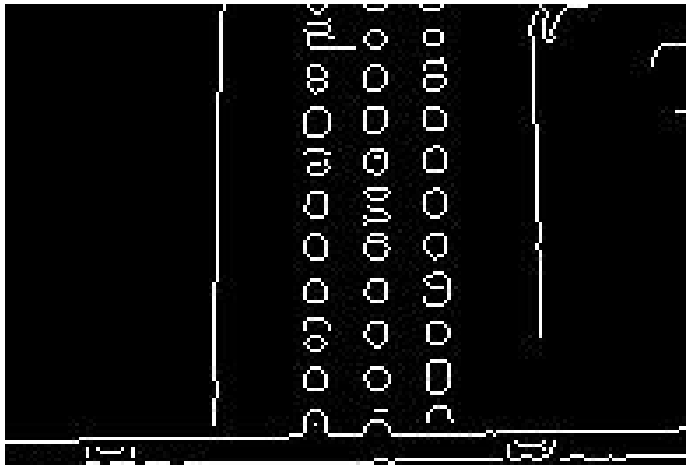
Case study: Line fitting

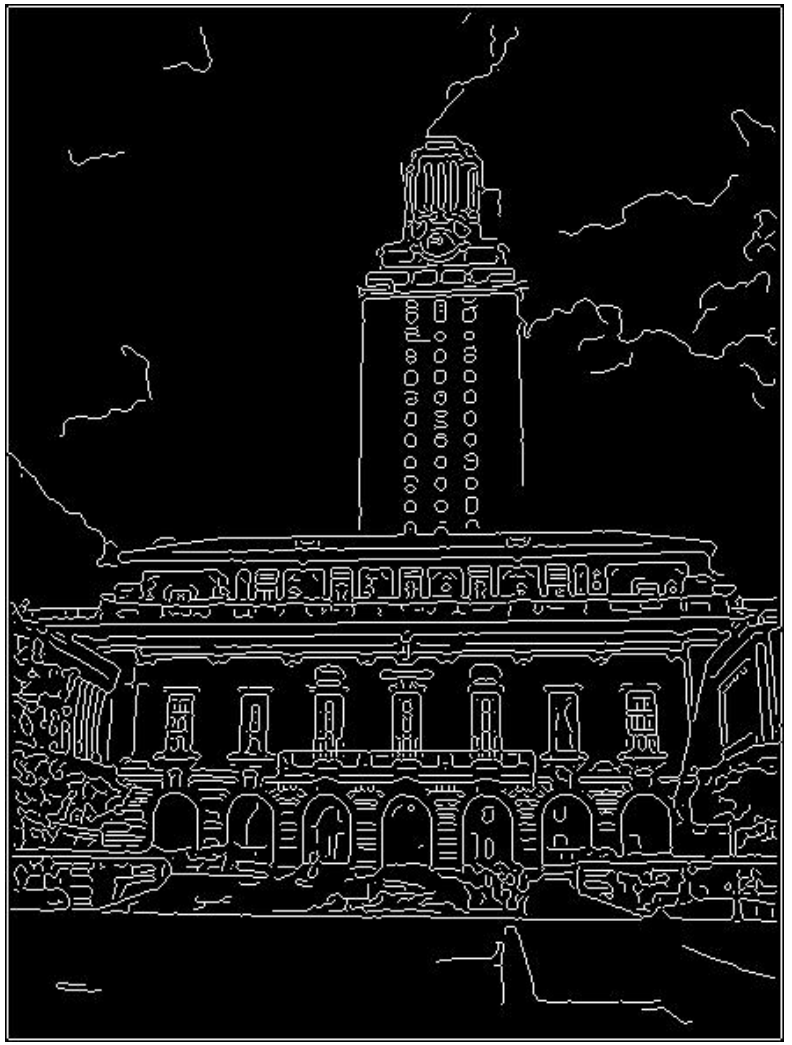
❖Why fit lines?
Many objects characterized by presence of straight lines



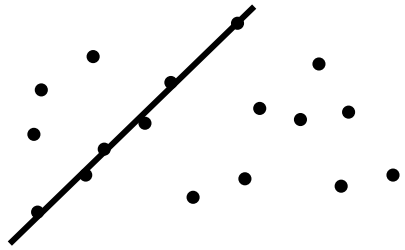
- Wait, why aren't we done just by running edge detection?

Difficulty of Line Fitting





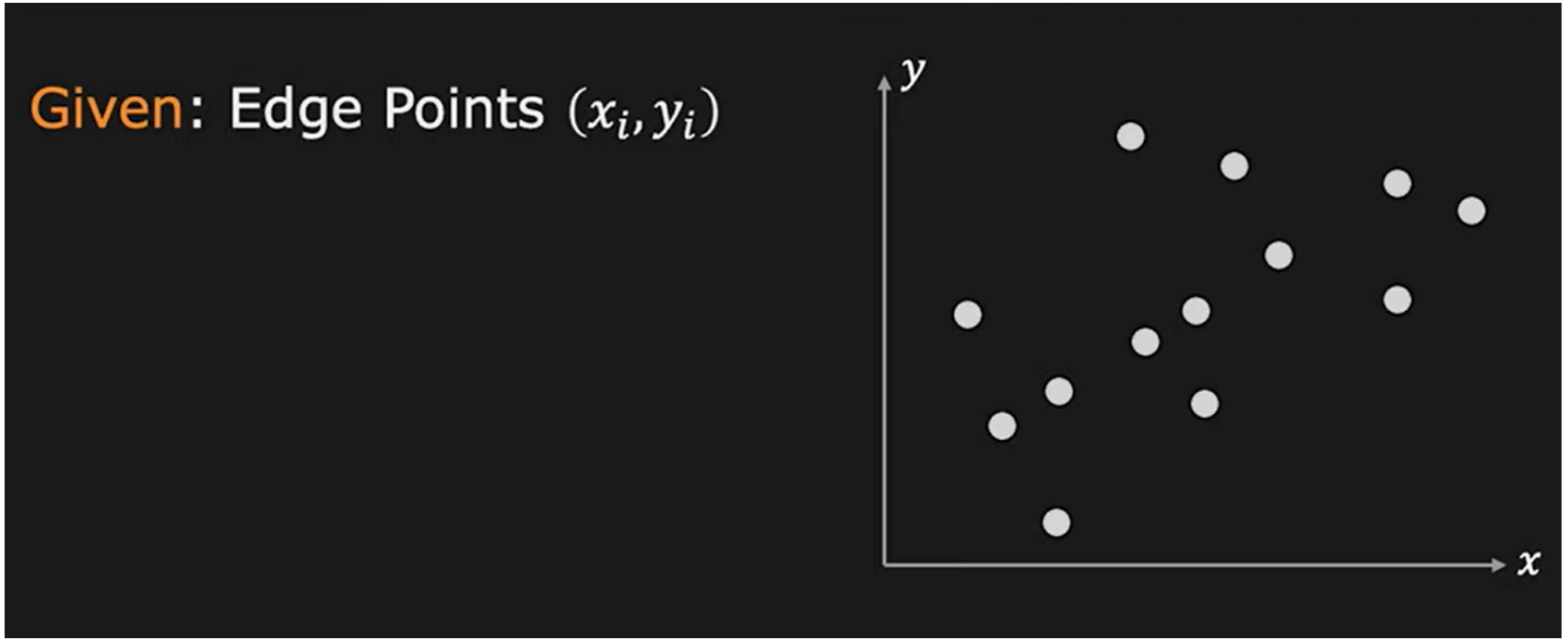
- **Extra** edge points (clutter), multiple models:
  - which points go with which line, if any?
- Only some parts of each line detected, and some parts are **missing:**
  - how to find a line that bridges missing evidence?
- **Noise** in measured edge points, orientations:
  - how to detect true underlying parameters?



Voting

- It's not feasible to check all combinations of features by fitting a model to each possible subset.
- **Voting** is a general technique where we let the features *vote for all models that are compatible with it*.
  - Cycle through features, cast votes for model parameters.
  - Look for model parameters that receive a lot of votes.
- Noise & clutter features will cast votes too, *but* typically their votes should be inconsistent with the majority of "good" features.

Hough Transform: Line Detection

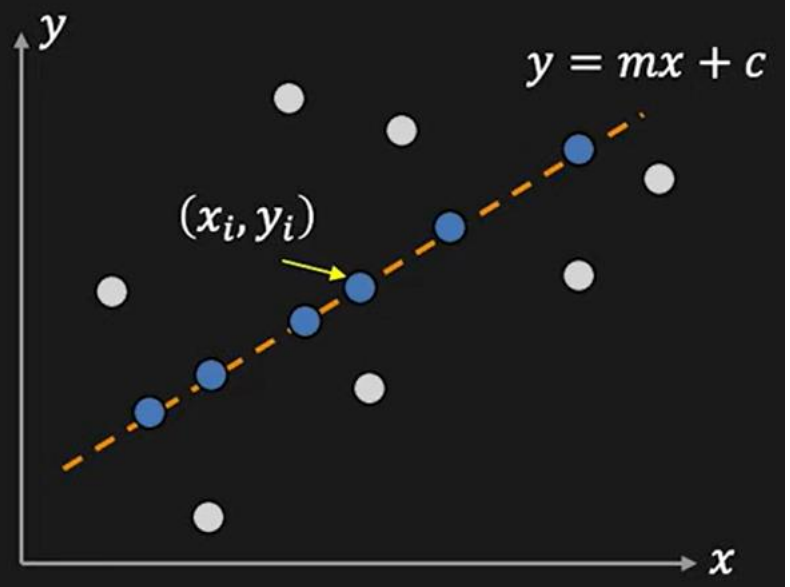


Hough Transform: Line Detection

Given: Edge Points (x_i, y_i)

Task: Detect line

$y = mx + c$



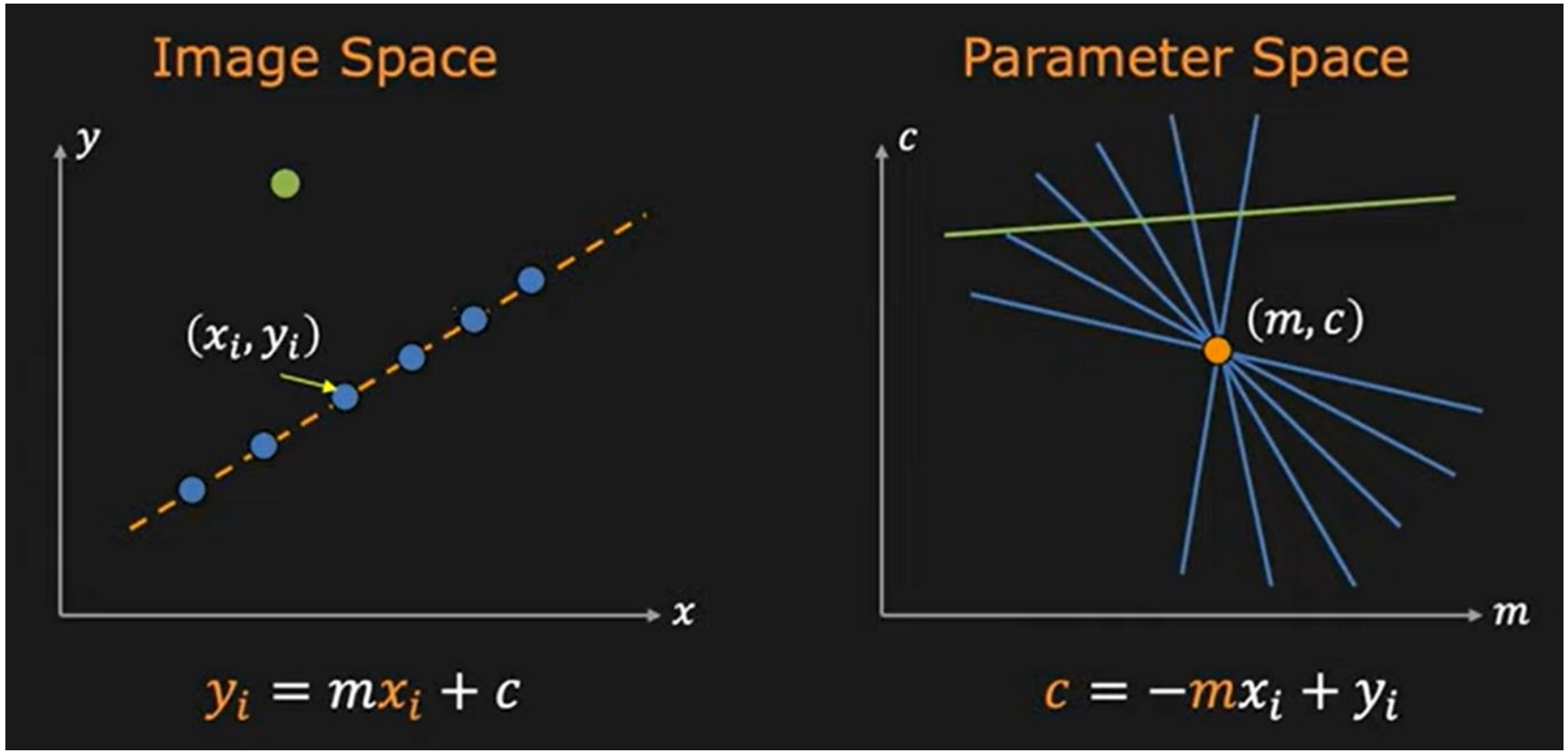
Consider point (x_i, y_i)

$y_i = mx_i + c \iff c = -mx_i + y_i$

Note: Straight line equation for m and c

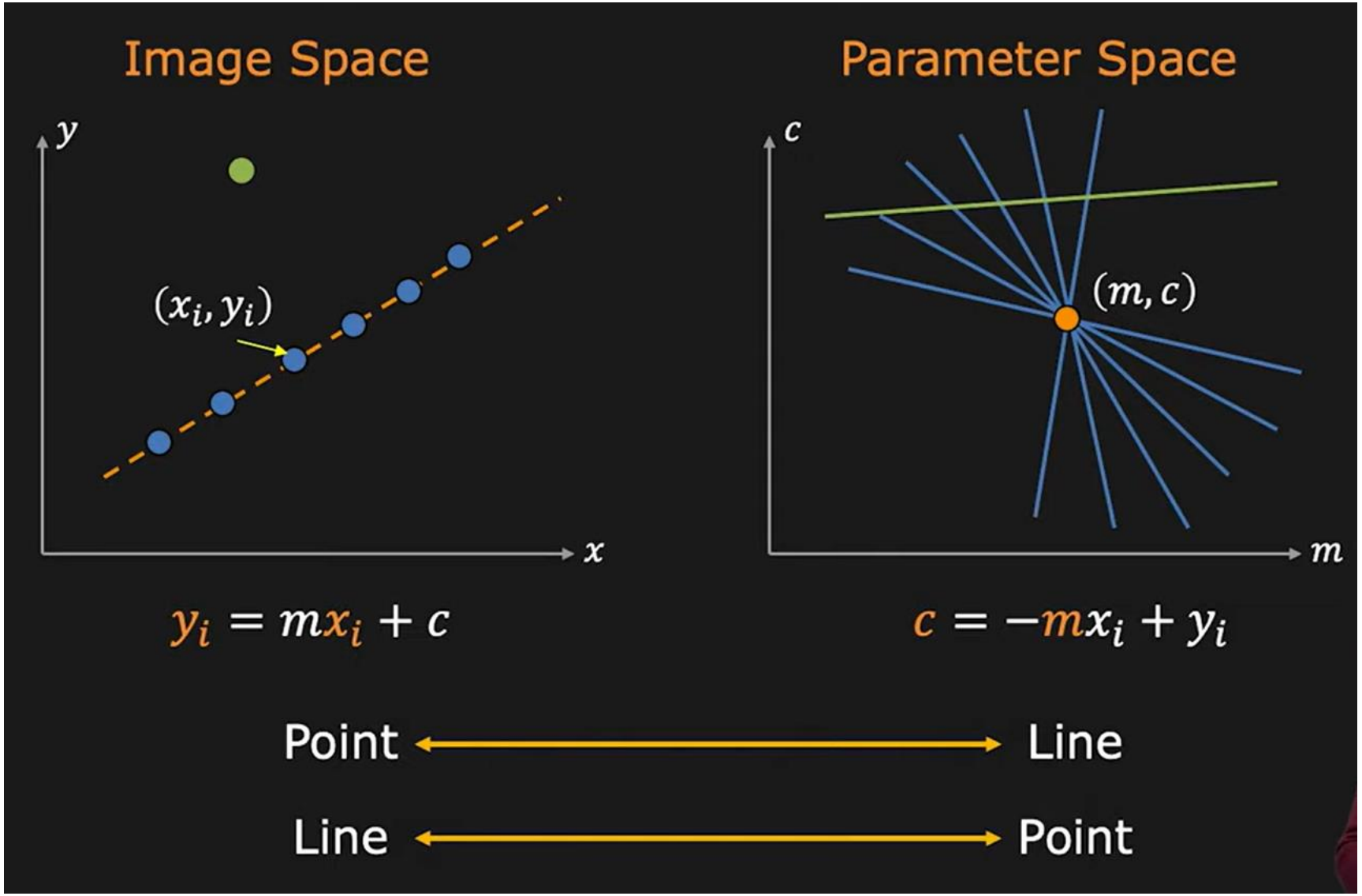
Hough Transform: Line Detection (Hough Parameter Space)

Transform all the image points from Image Space to Hough Parameter Space



Hough Transform: Concept

Transform all the image points from Image Space to Hough Parameter Space



Hough Transform: Line Detection Algorithm (with a Voting scheme)

Step 1. Quantize parameter space (m, c)

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

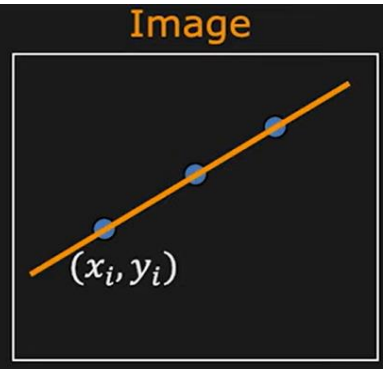
Step 3. Set $A(m, c) = 0$ for all (m, c)

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

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

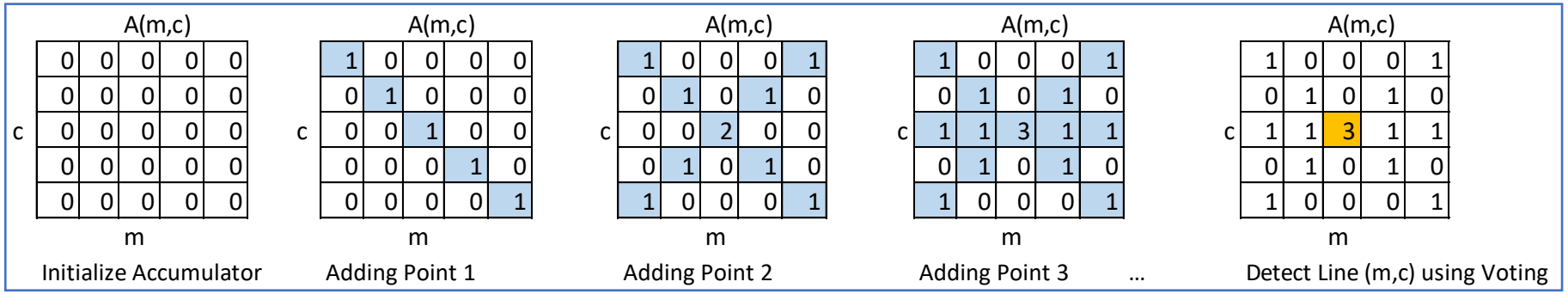
if (m, c) lies on the line: $c = -mx_i + y_i$

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



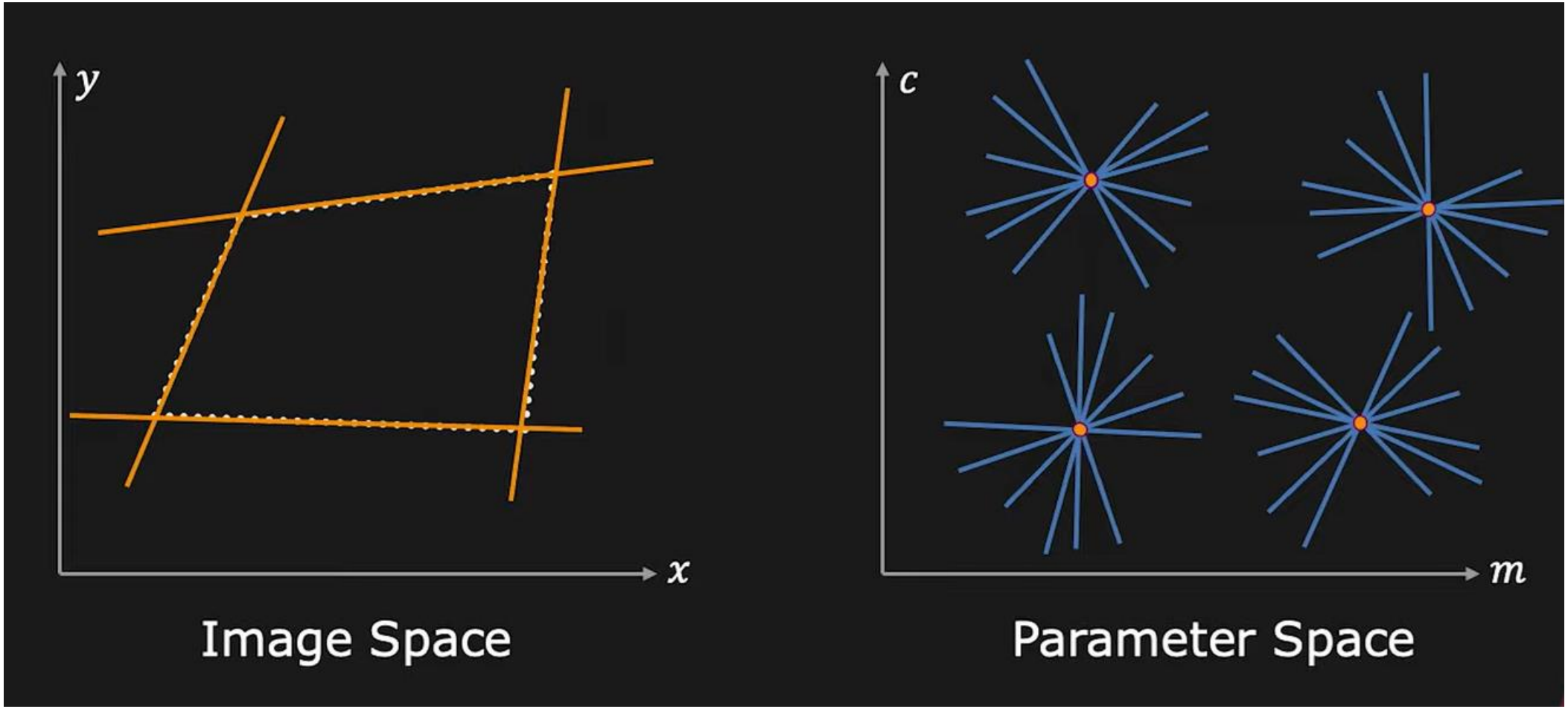
$A(m, c)$

c \ m					
	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



https://fiveko.com/online-tools/hough-circle-detection-demo/

Multiple Lines Detection



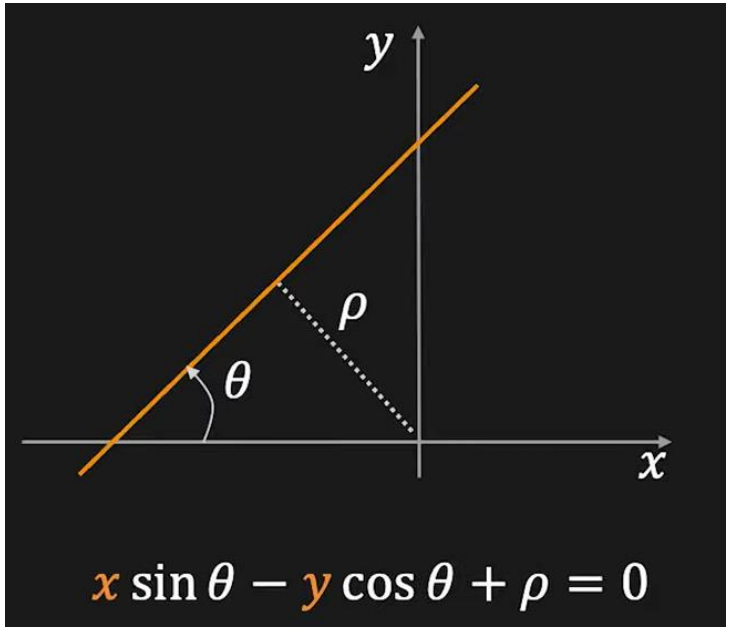
Better Parameterization

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

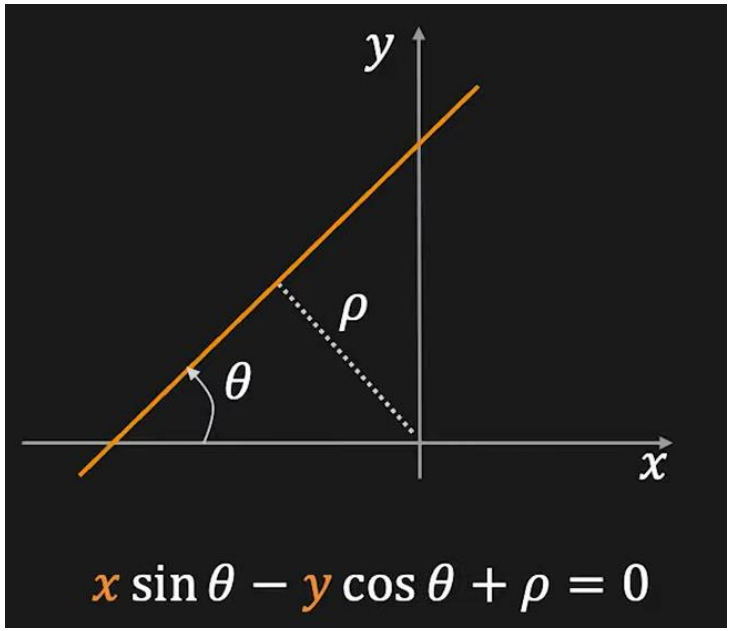
- Large Accumulator
- More Memory and Computation

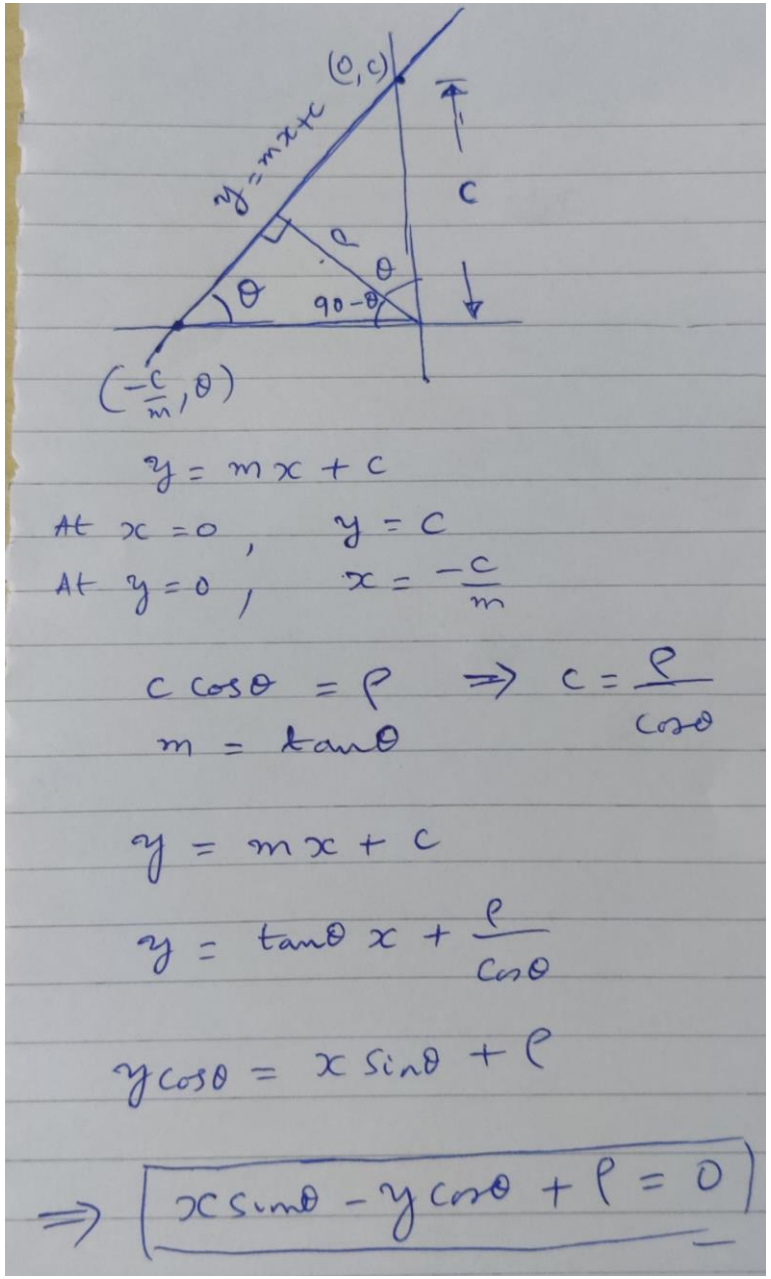
Solution: Use $x \sin\theta - y \cos\theta + \rho = 0$

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

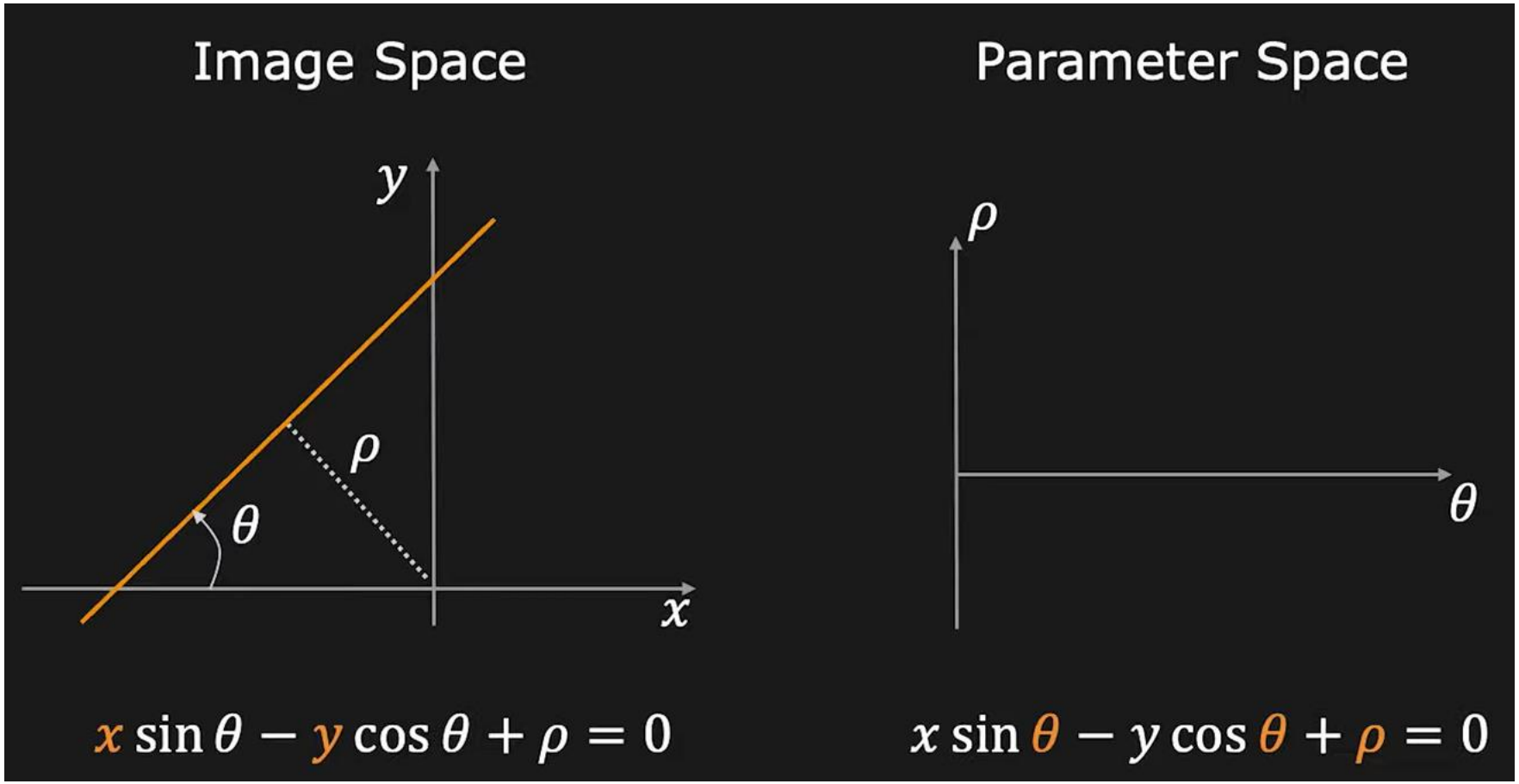


Two Line Equations

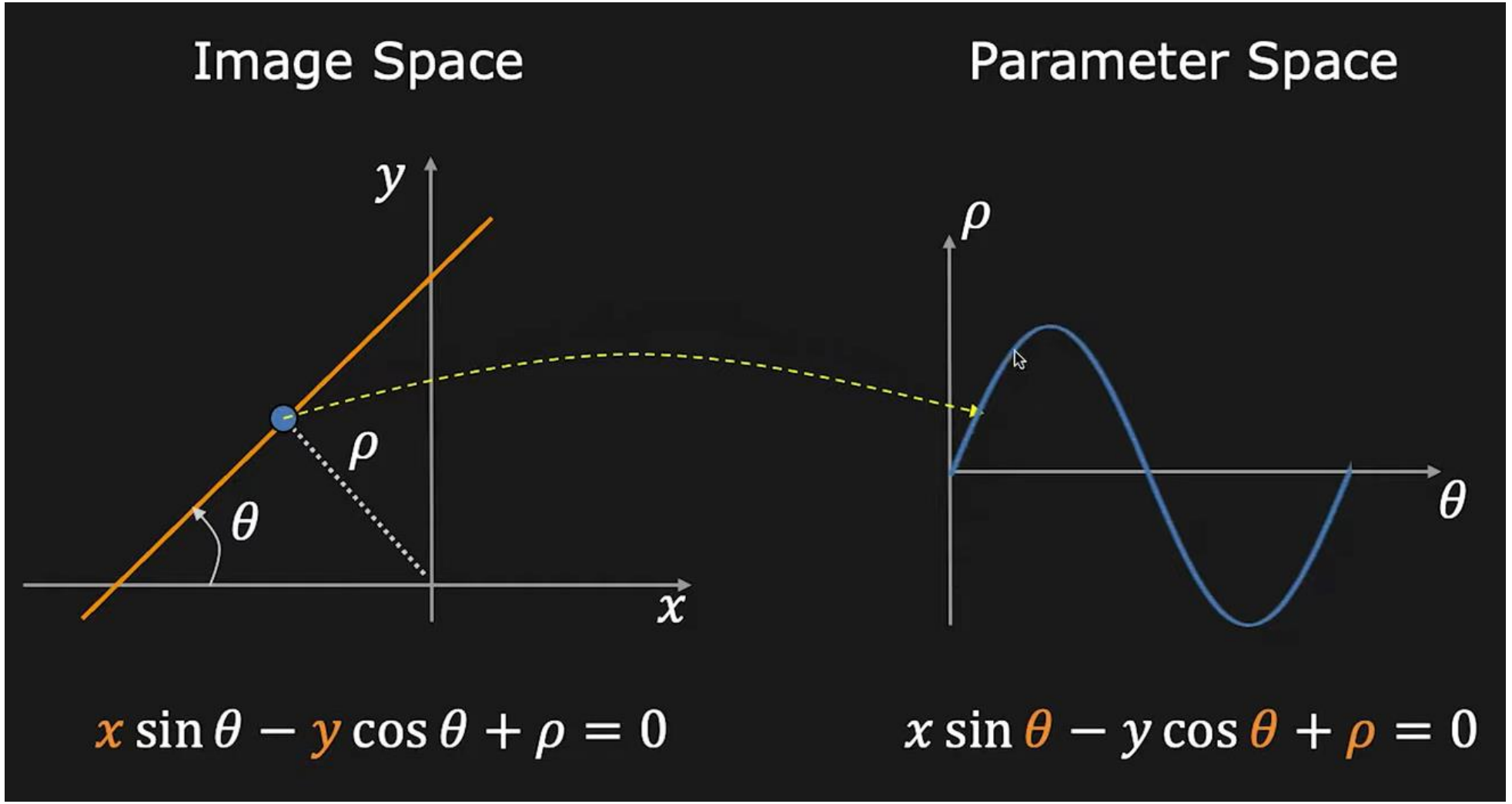




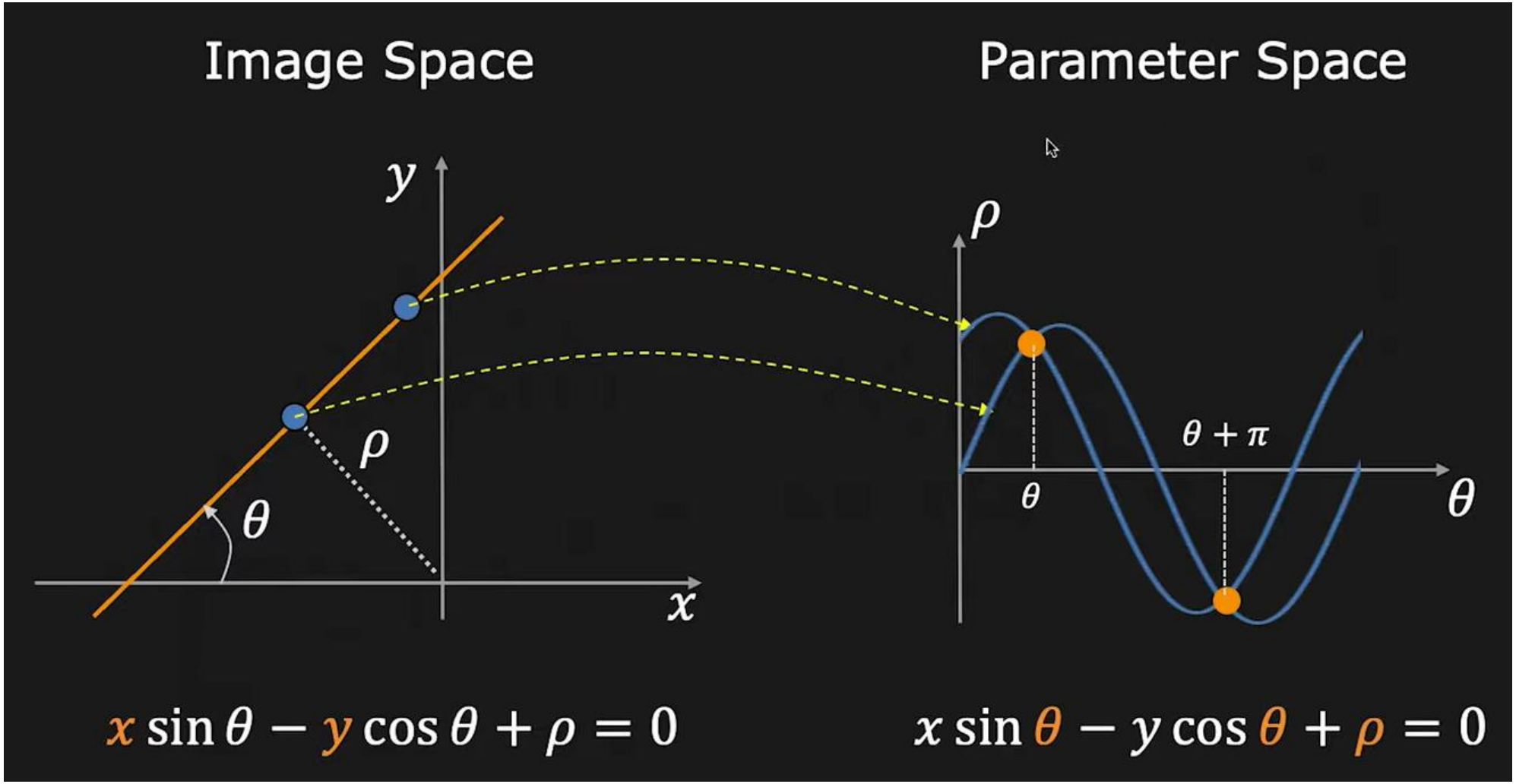
Better Parameterization



Better Parameterization



Better Parameterization



Hugh Transform Mechanics

- How big should the accumulator cells be?
  - Too big: Different lines may be merged.
  - Too small: Noise causes lines to be missed. Miss lines because some points that are not exactly collinear cast votes for different buckets
- How many lines
  - Count the peaks in the accumulator array
- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude
- Handling inaccurate edge locations:
  - Increment patch in accumulator rather than single point
- Strong Edges
  - Give more votes for stronger edges.
  - Examine the surrounding pixels in the chosen cell
  - Edge thinning can be beneficial
- Change the sampling of (ρ, θ) to give more/less resolution
- Increment neighboring bins (smoothing in accumulator array)

Hugh Transform Advantages and Disadvantages

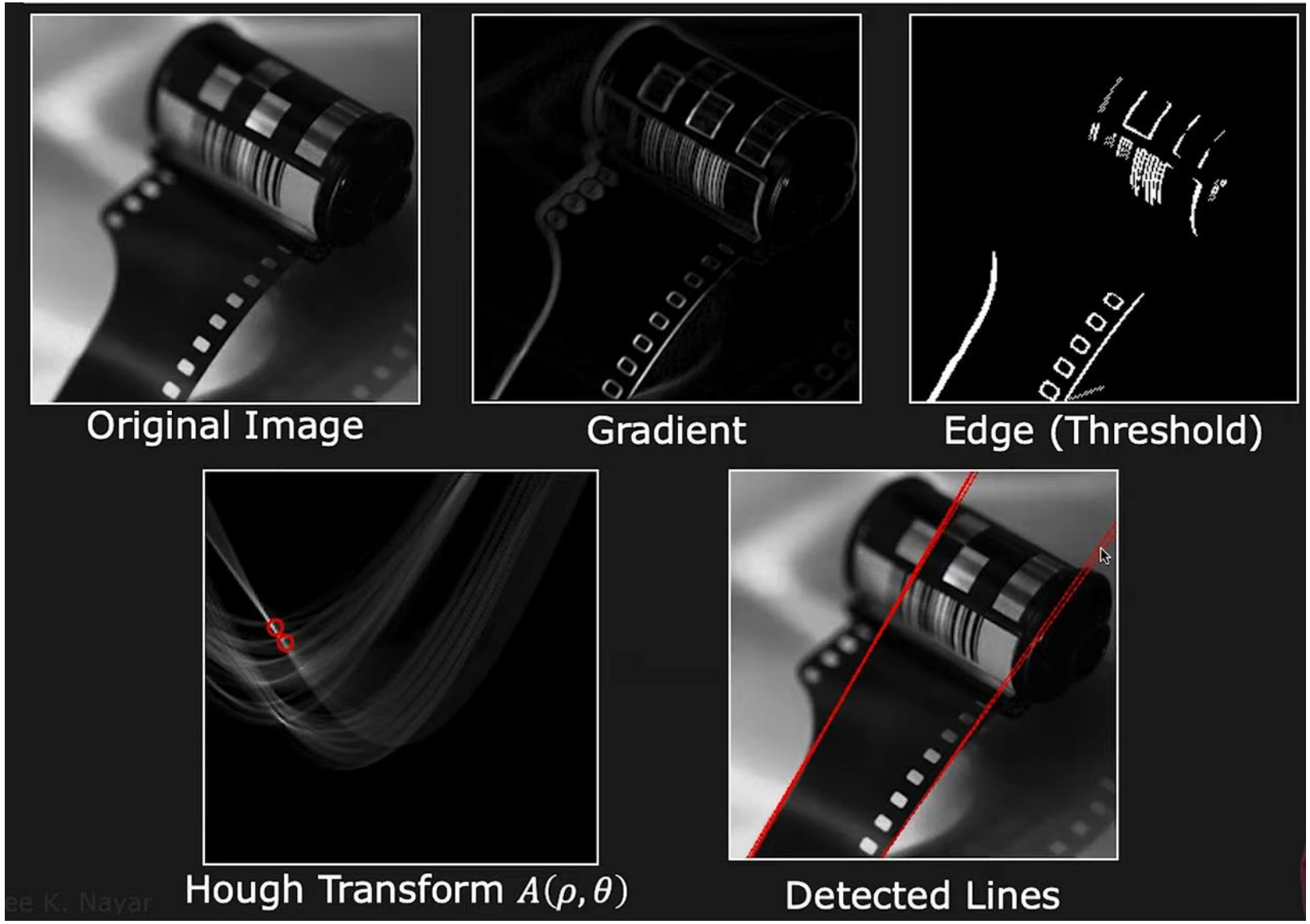
❖ Advantages

- Can deal with non-locality and occlusion
- Can detect multiple instances of a model (e.g., line, circle, …) in a single pass
- Some robustness to noise: noise points unlikely to contribute consistently to any single bin

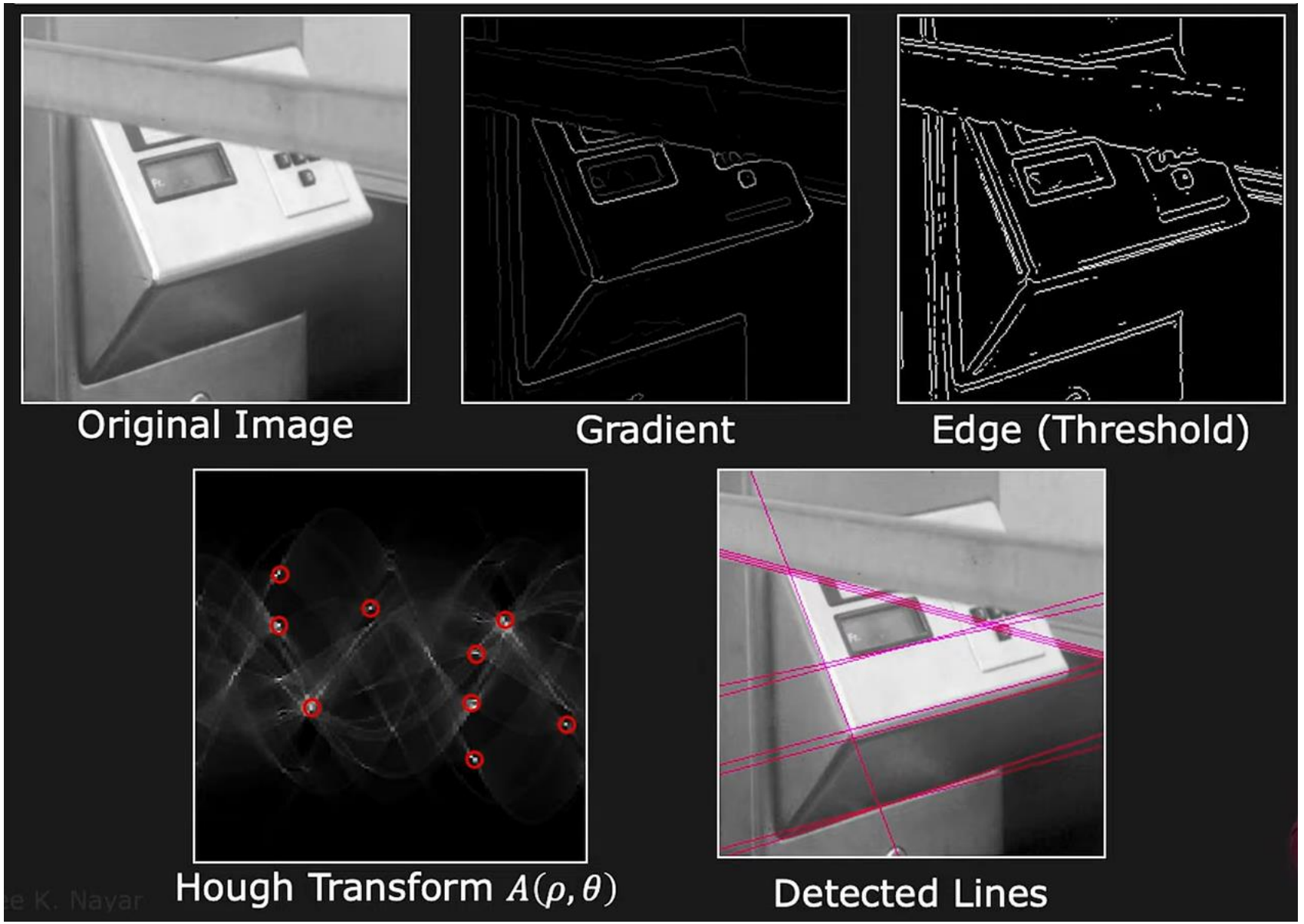
❖ Disadvantages

- ➢ 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 Accumulator (grid) size

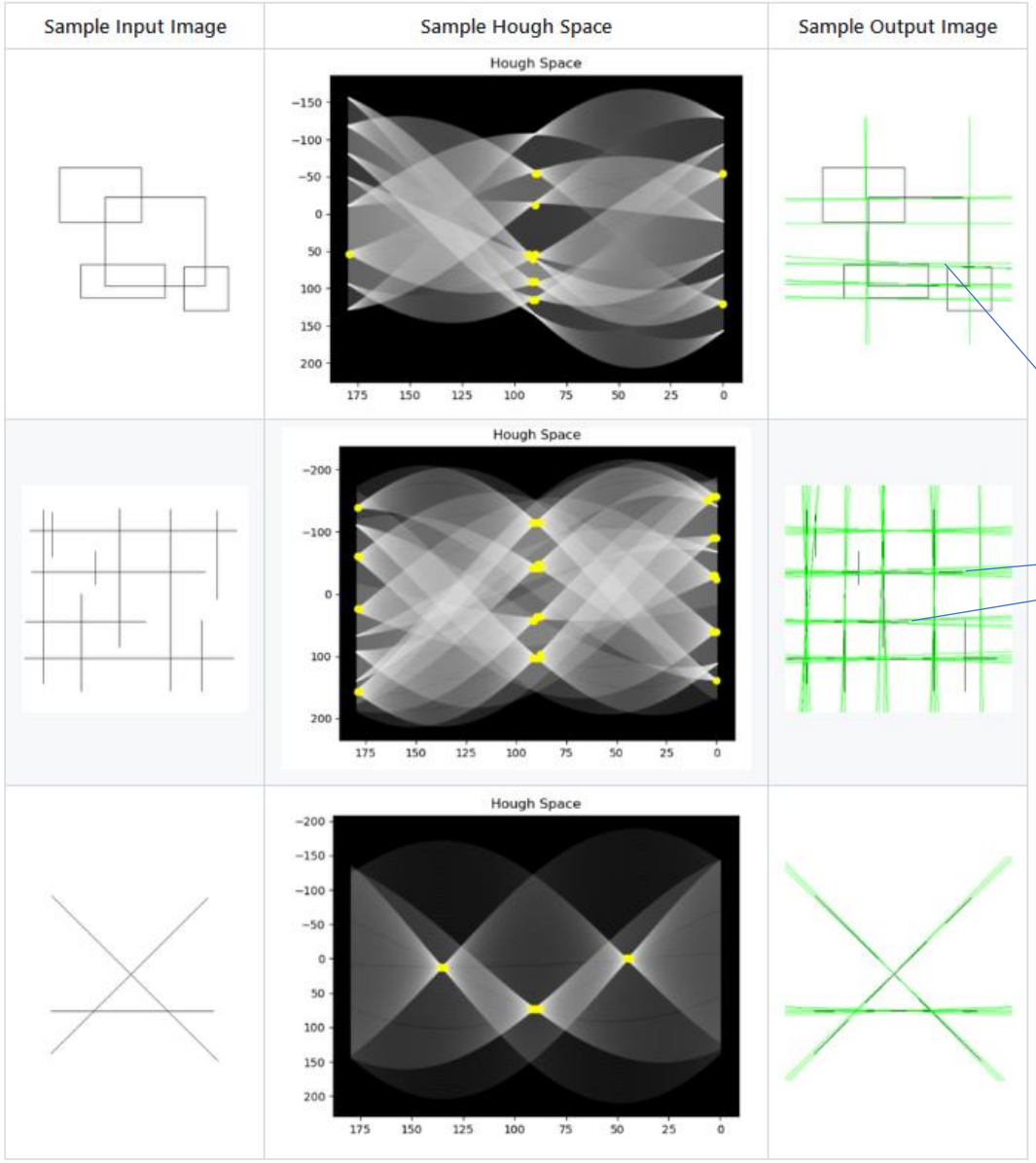
Line Detection Results



Line Detection Results



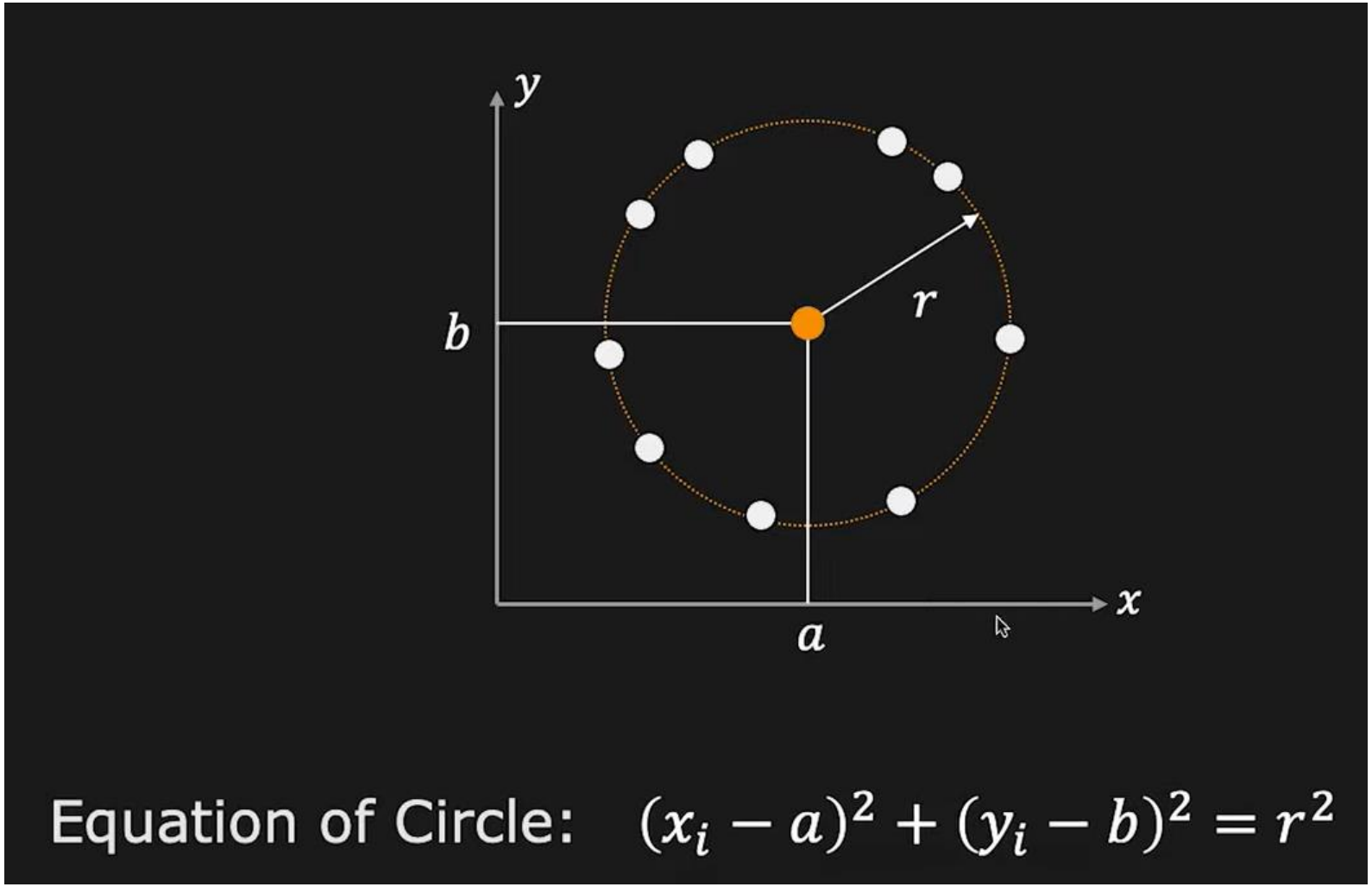
Line Detection Results



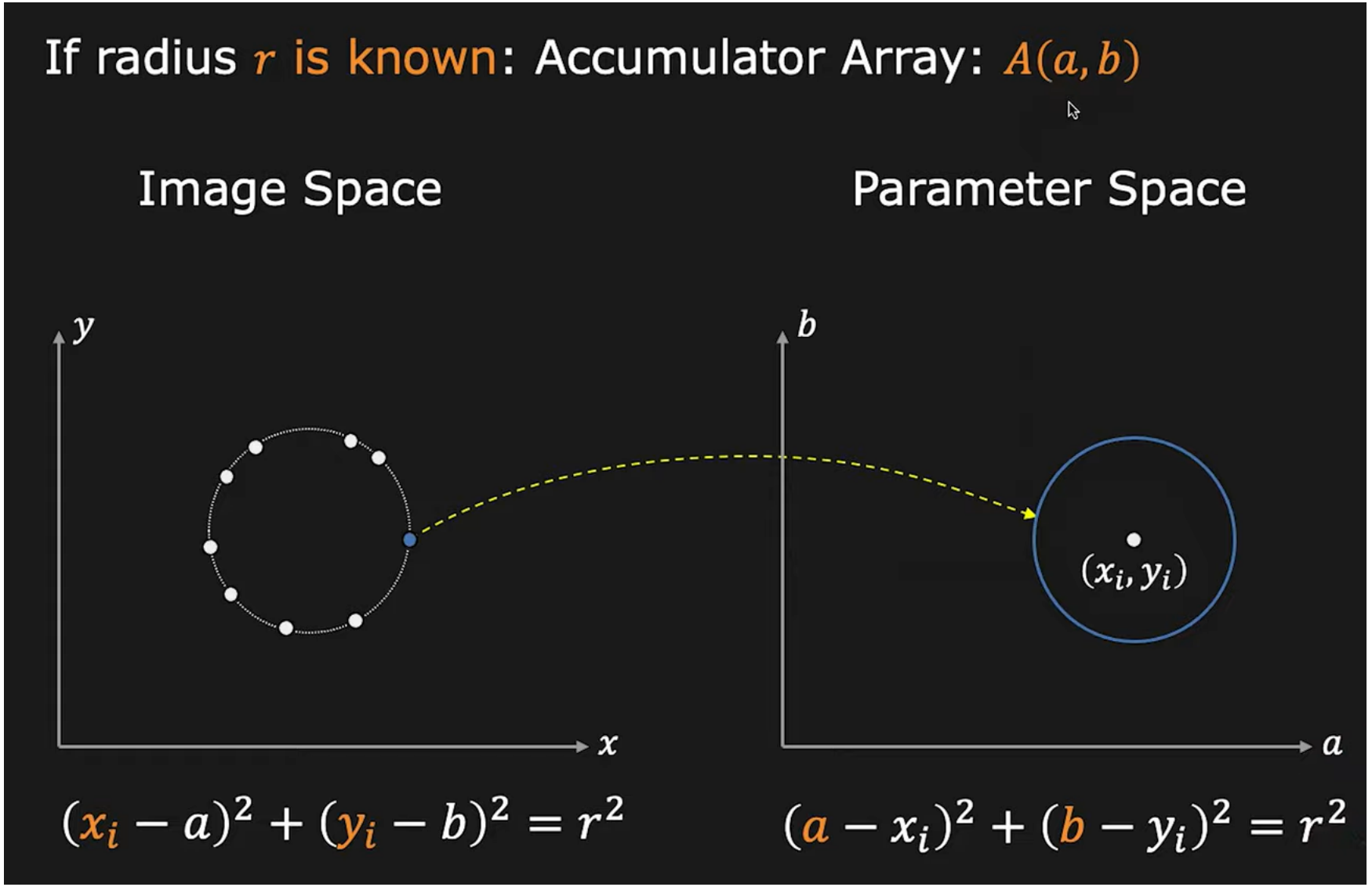
Different lines are merged

https://github.com/adityaintwala/Hough-Line-Detection

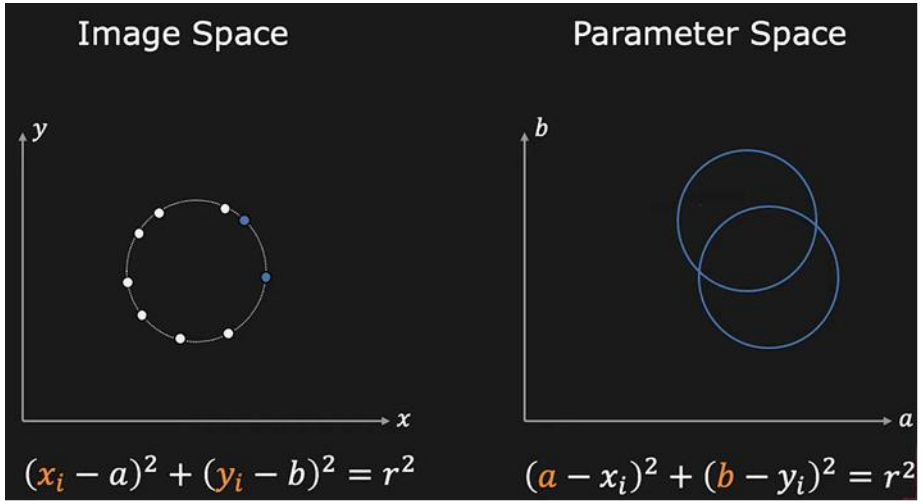
Hugh Transform: Circle Detection

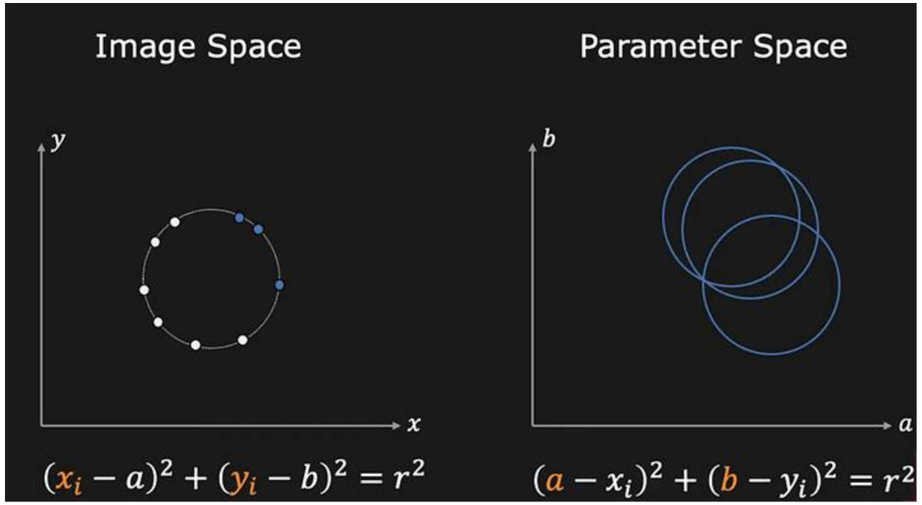


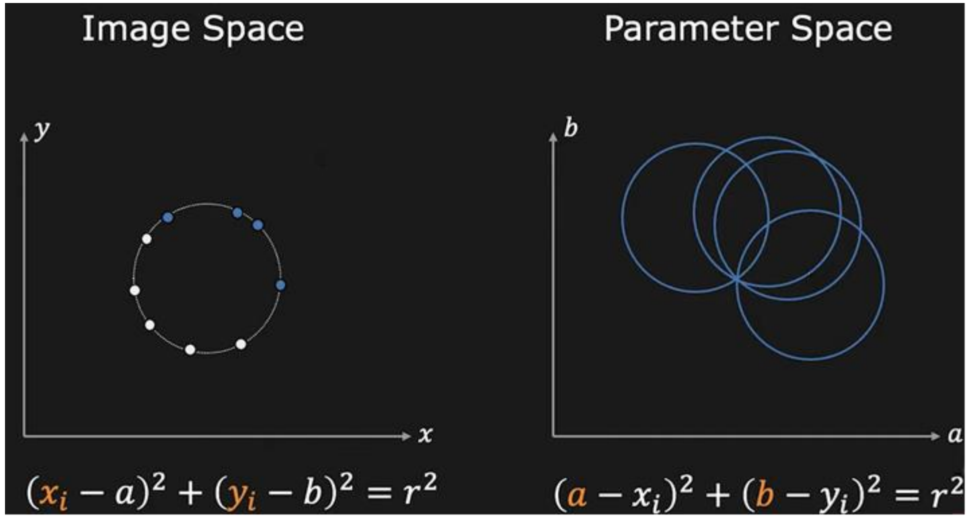
Hugh Transform: Circle Detection

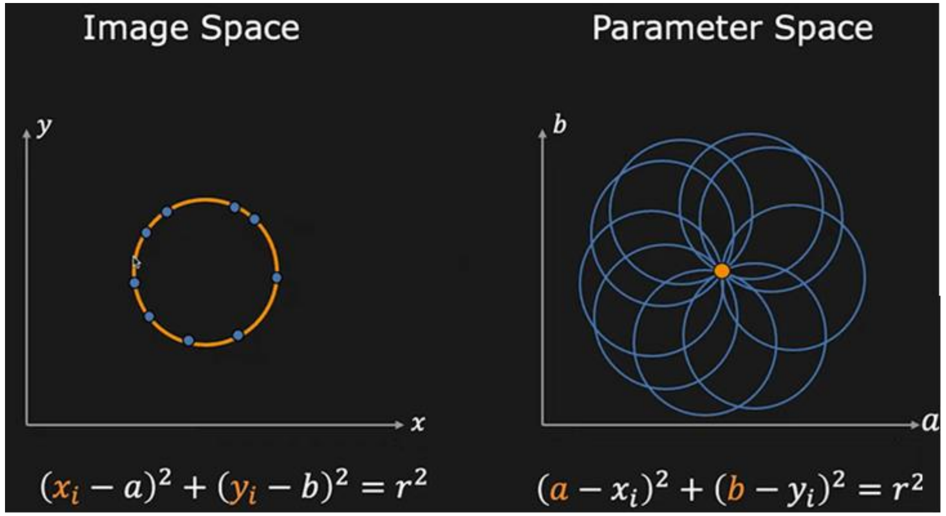


Hugh Transform: Circle Detection

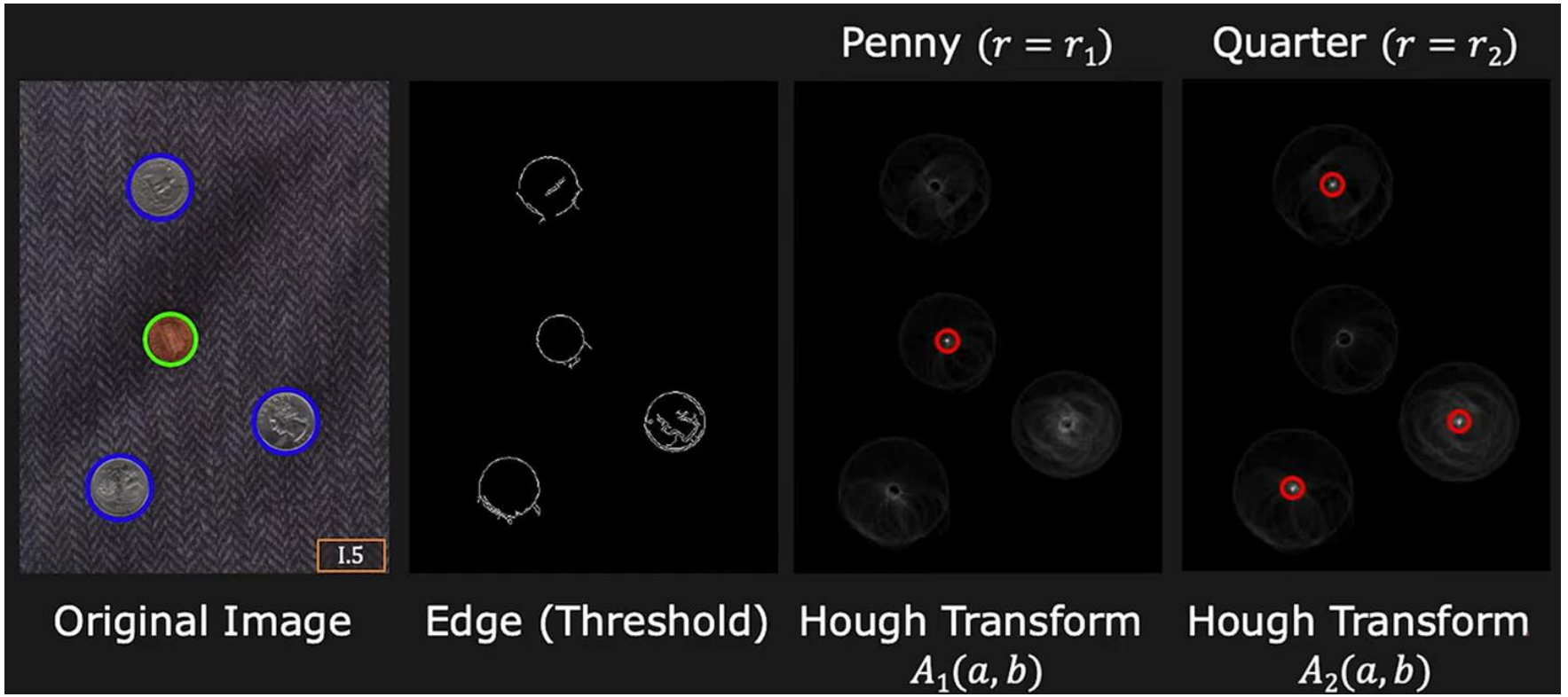








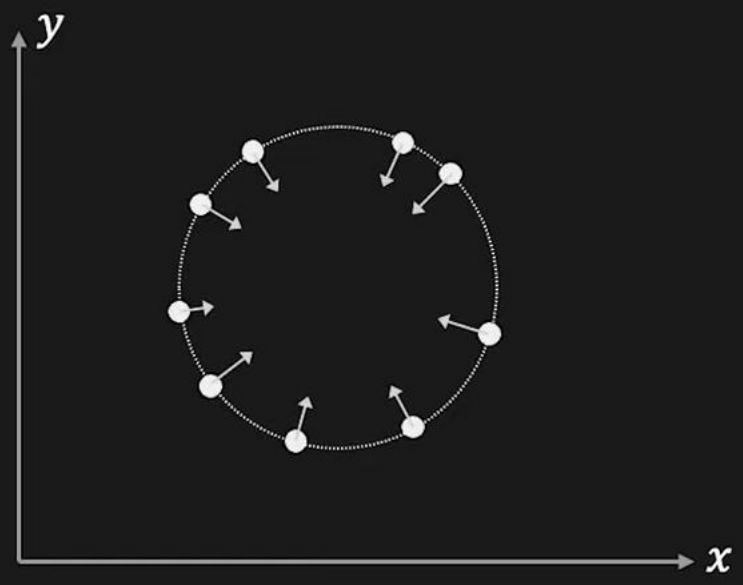
Circle Detection Results



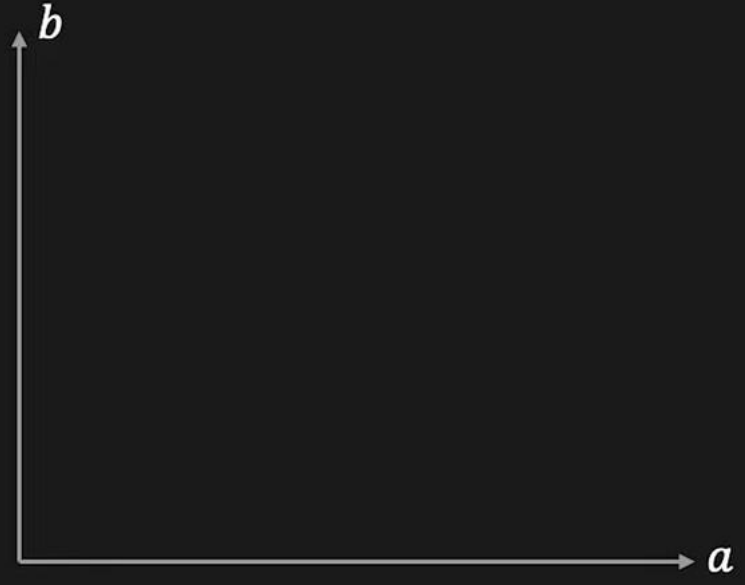
Using Gradient Information

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

Image Space



Parameter Space

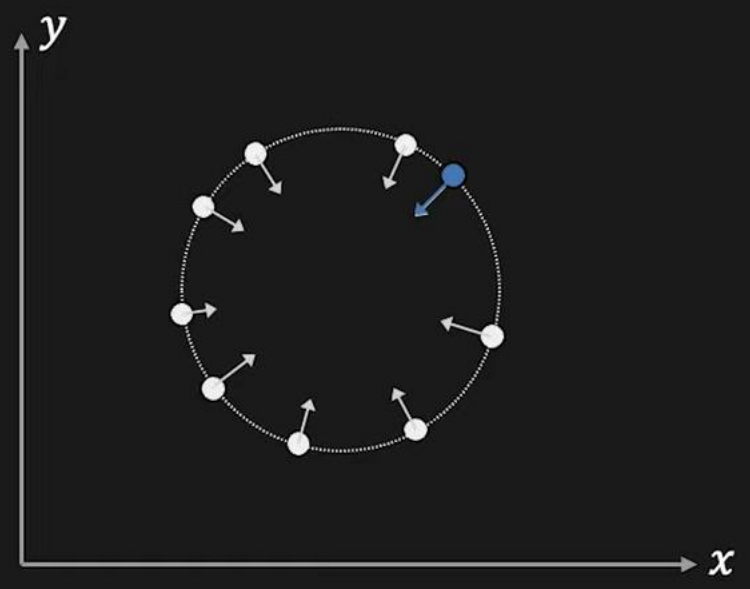


$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

Using Gradient Information

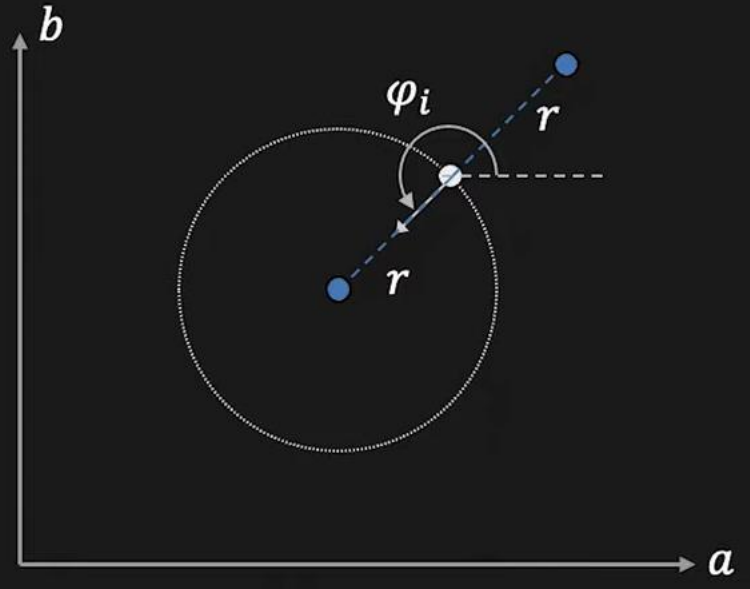
Given: Edge Location (x_i, y_i), Edge Direction φ_i and Radius r

Image Space



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

Parameter Space



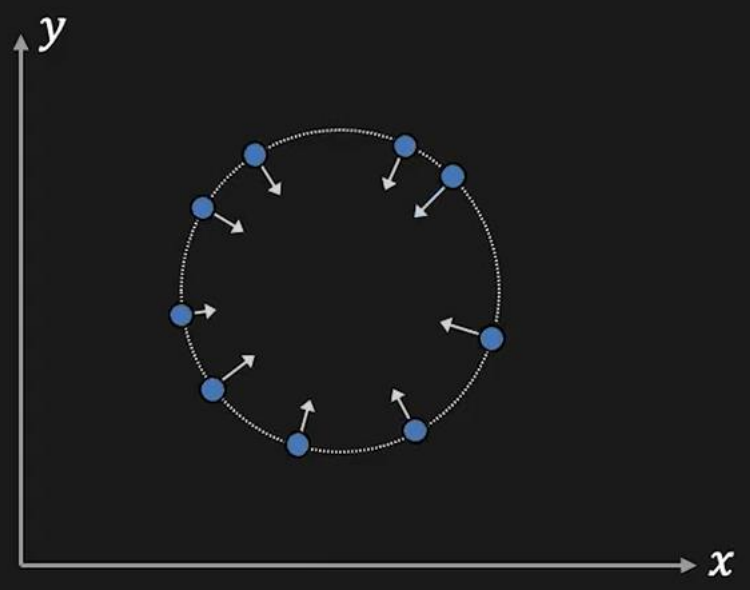
$$a = x_i \pm r \cos \varphi_i$$

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

Using Gradient Information

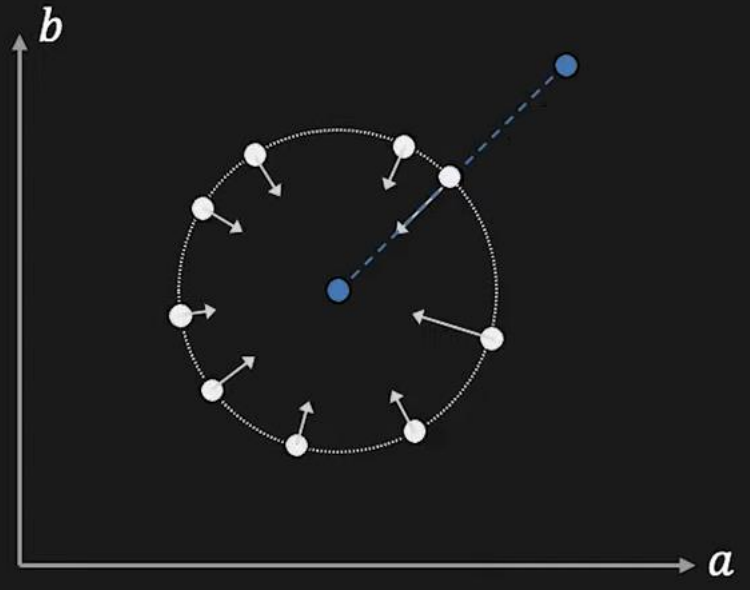
Given: Edge Location (x_i, y_i), Edge Direction φ_i and Radius r

Image Space



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

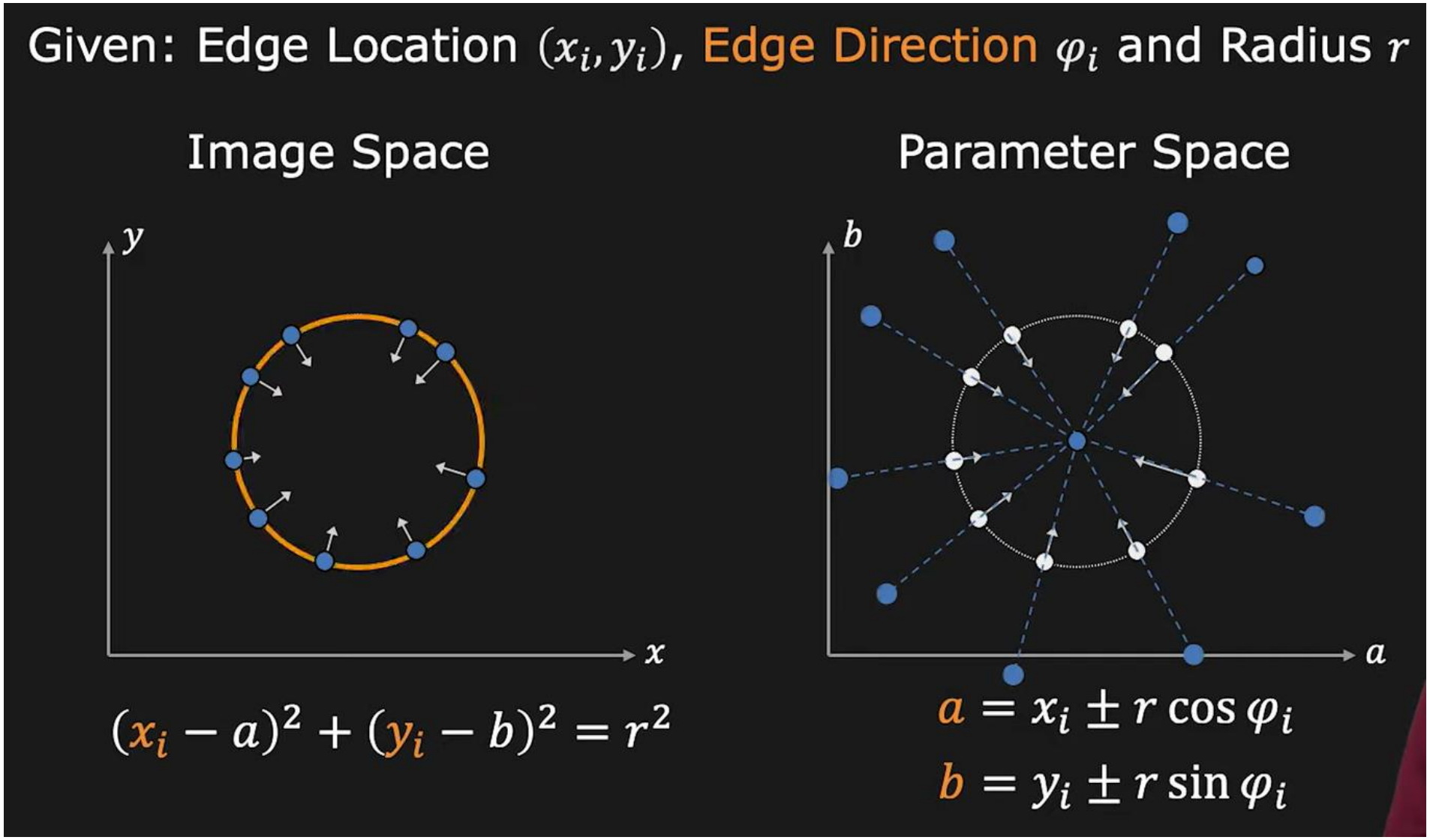
Parameter Space



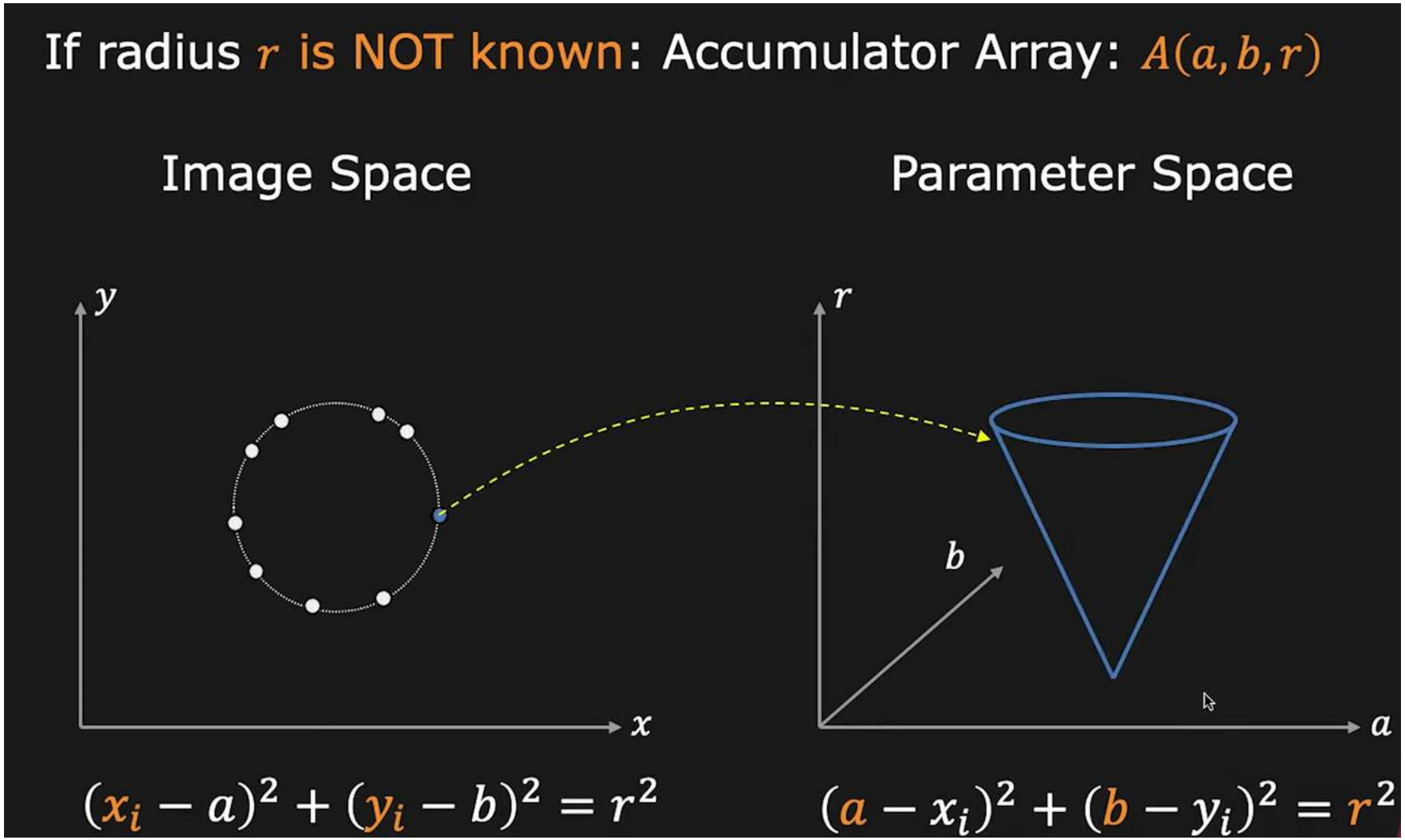
$$a = x_i \pm r \cos \varphi_i$$

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

Using Gradient Information



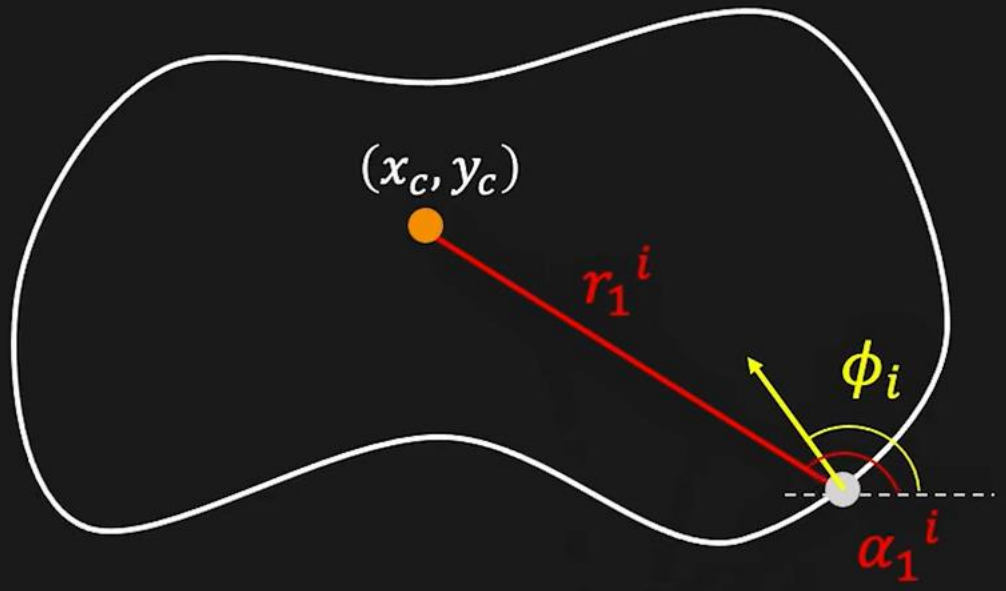
Hough Transform: Circle Detection



Note: Complexity of search time increases exponentially with the number of model parameters

Generalized Hough Transform

Find shapes that cannot be described by equations



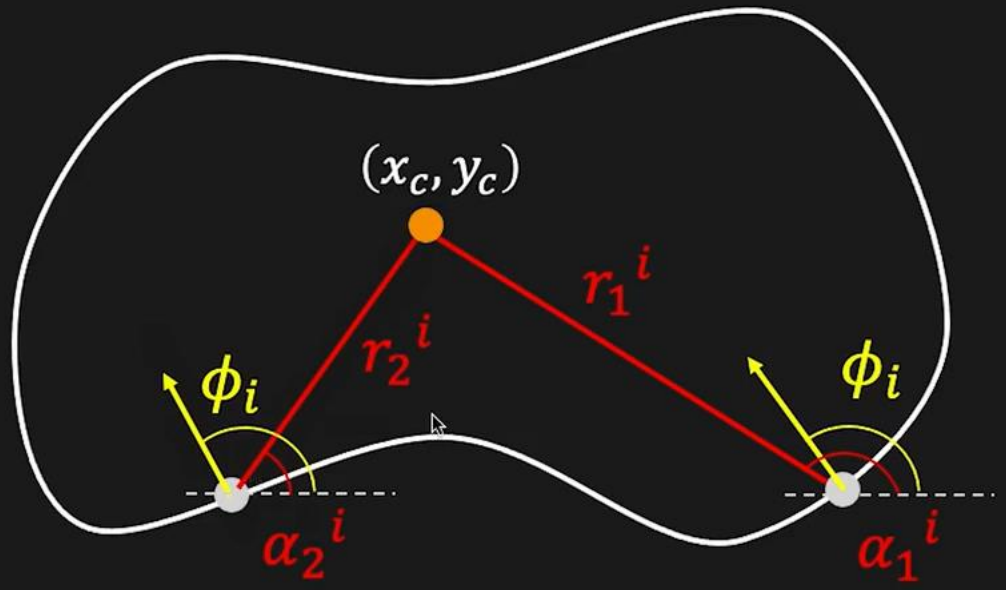
Reference point: (x_c, y_c)

Edge direction: ϕ_i $\quad 0 \le \phi_i < 2\pi$

Edge location: $\vec{r}_k{}^i = (r_k{}^i, \alpha_k{}^i)$

Generalized Hough Transform

Find shapes that cannot be described by equations

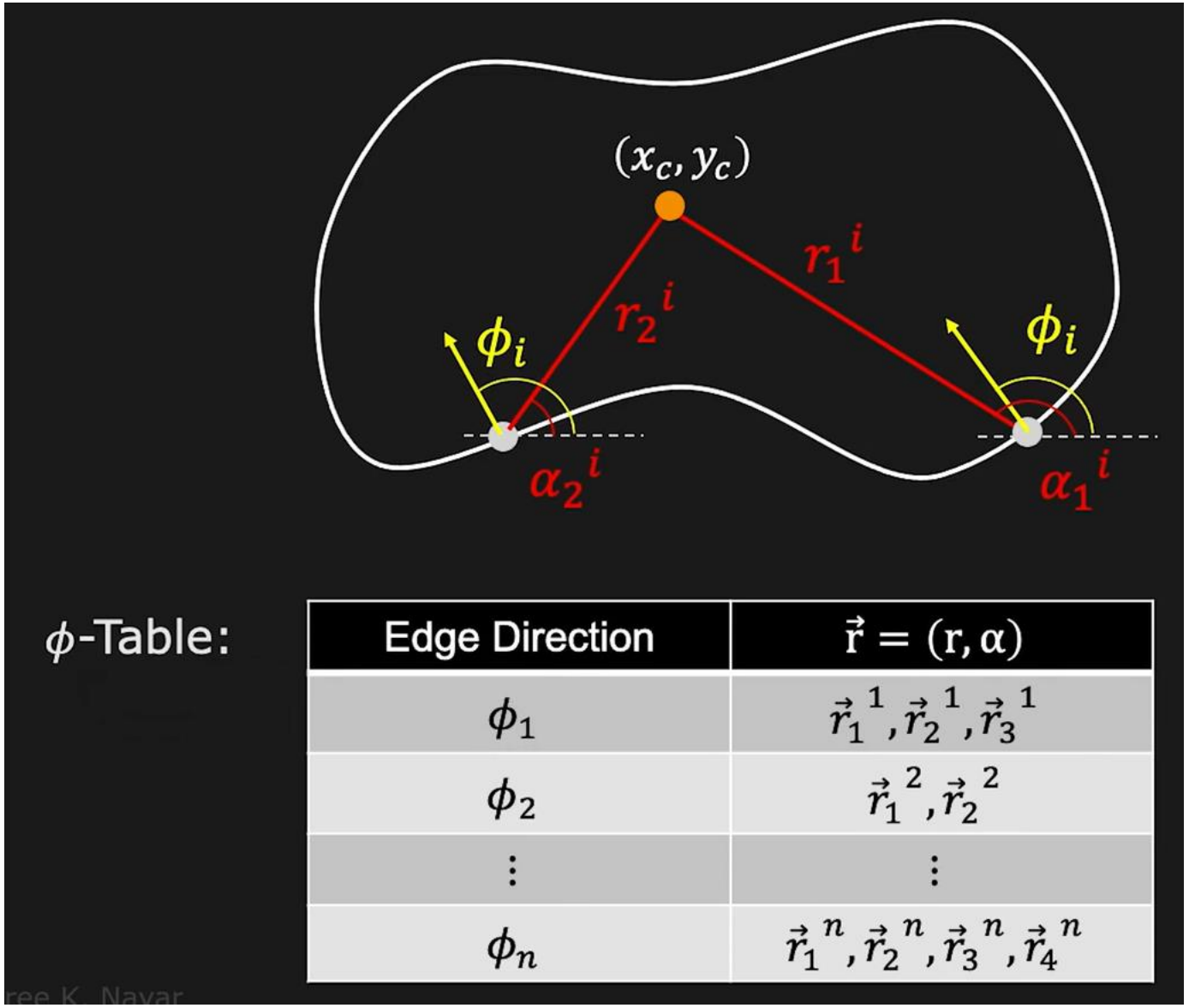


Reference point: (x_c, y_c)

Edge direction: ϕ_i $\quad 0 \leq \phi_i < 2\pi$

Edge location: $\vec{r}_k{}^i = (r_k{}^i, \alpha_k{}^i)$

Hough Model



ϕ-Table:

Edge Direction	$\vec{r} = (r, \alpha)$
ϕ_1	$\vec{r}_1^{\,1}, \vec{r}_2^{\,1}, \vec{r}_3^{\,1}$
ϕ_2	$\vec{r}_1^{\,2}, \vec{r}_2^{\,2}$
⋮	⋮
ϕ_n	$\vec{r}_1^{\,n}, \vec{r}_2^{\,n}, \vec{r}_3^{\,n}, \vec{r}_4^{\,n}$

ree K. Nayar

Generalized Hough Transform

- Create accumulator array $A(x_c, y_c)$
- Set $A(x_c, y_c) = 0$ for all (x_c, y_c)
- For each edge point (x_i, y_i, ϕ_i),

  For each entry $\phi_i \rightarrow \vec{r}_k{}^i$ in ϕ – table,

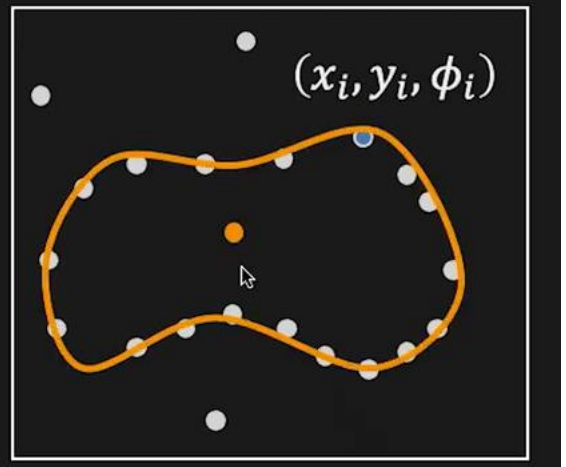
$$x_c = x_i \pm r_k{}^i \cos(\alpha_k{}^i)$$

$$y_c = y_i \pm r_k{}^i \sin(\alpha_k{}^i)$$

$$A(x_c, y_c) = A(x_c, y_c) + 1$$

- Find local maxima in $A(x_c, y_c)$

Image

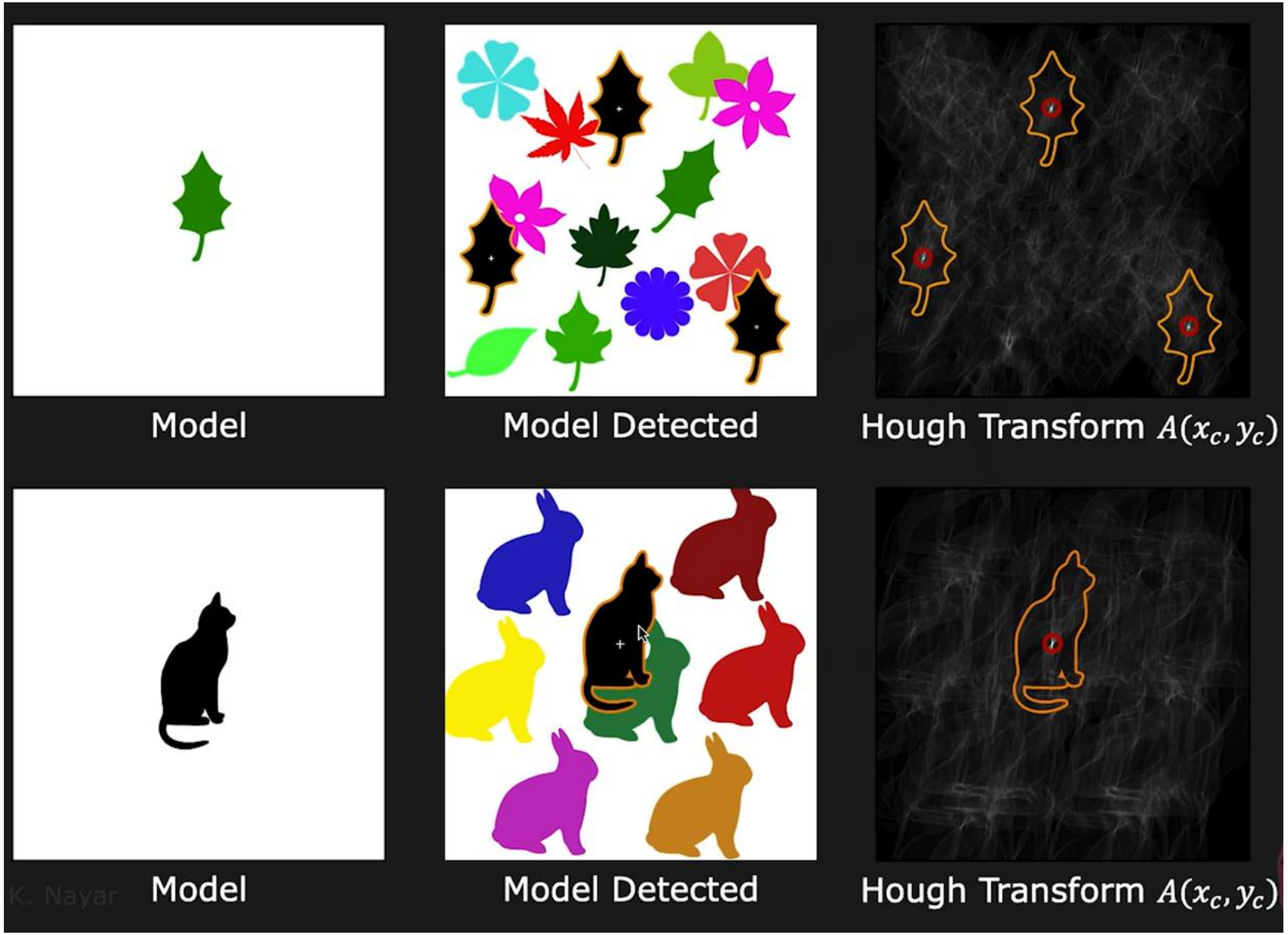


$A(x_c, y_c)$

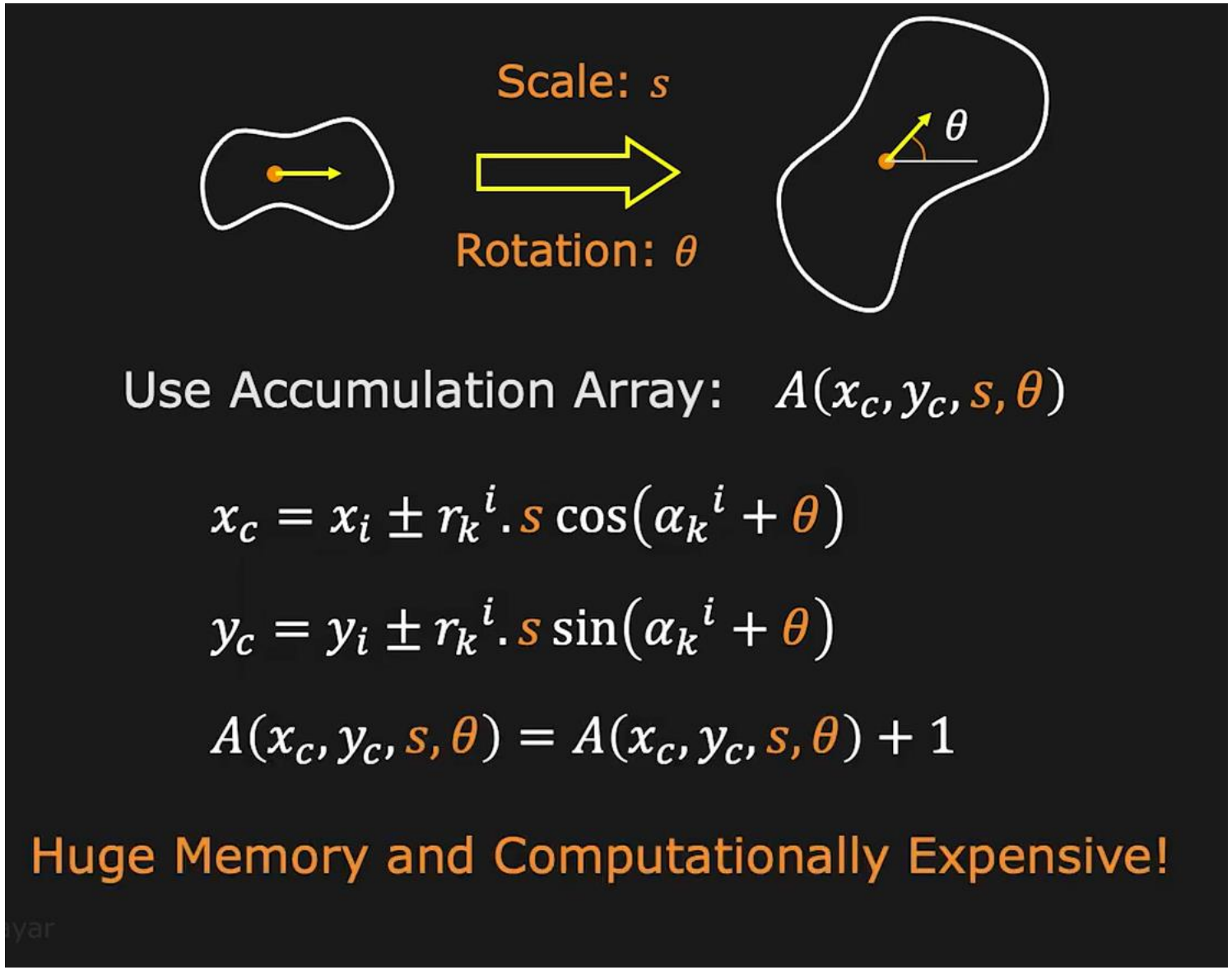
x_c					
	0	0	0	0	0
	0	2	0	1	0
	0	0	4	1	0
	0	2	0	0	0
	0	0	0	1	0

y_c

Results


Model
Model Detected
Hough Transform $A(x_c, y_c)$
Model
Model Detected
Hough Transform $A(x_c, y_c)$
K. Nayar

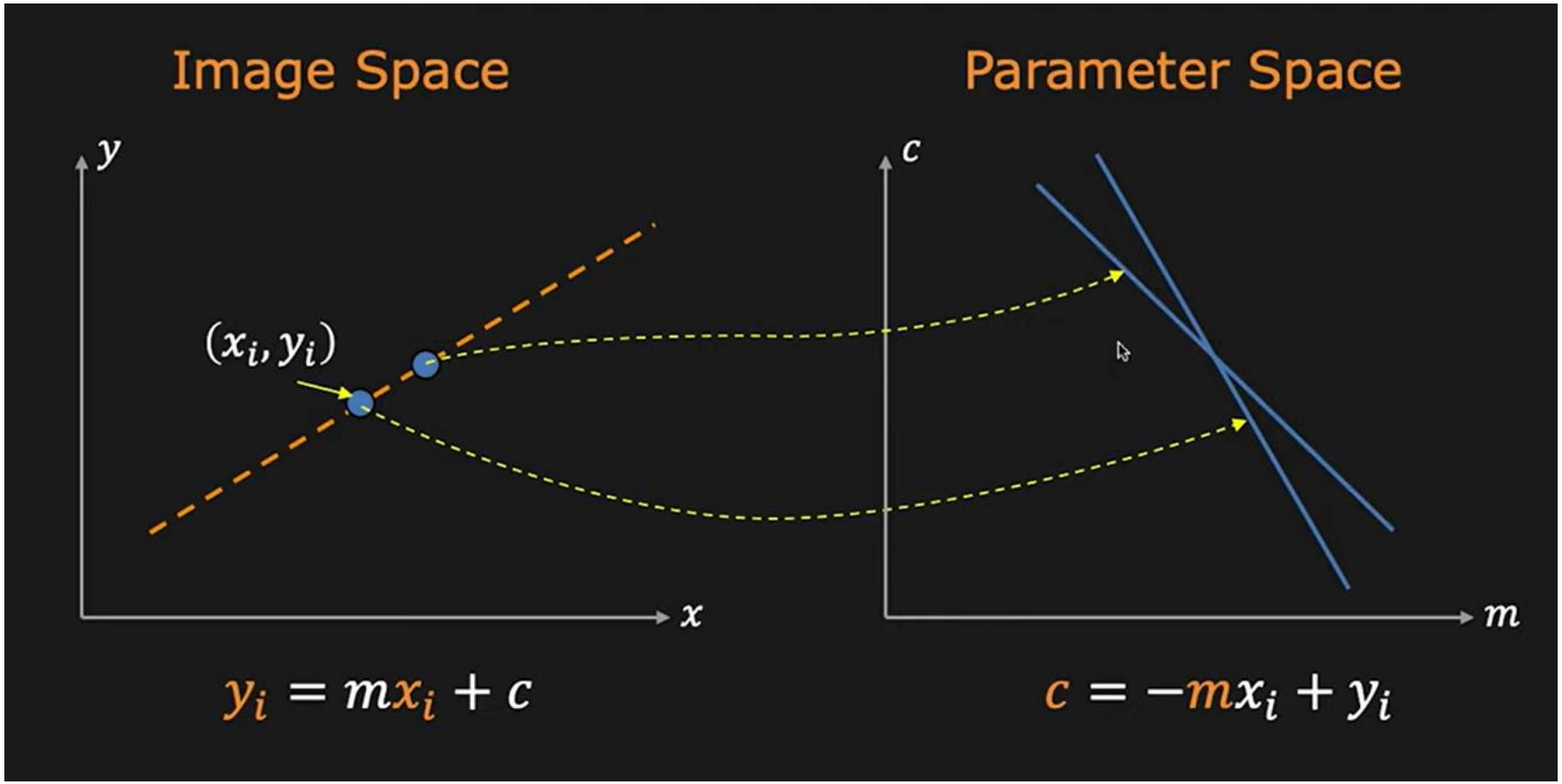
Handling Scale and Rotation



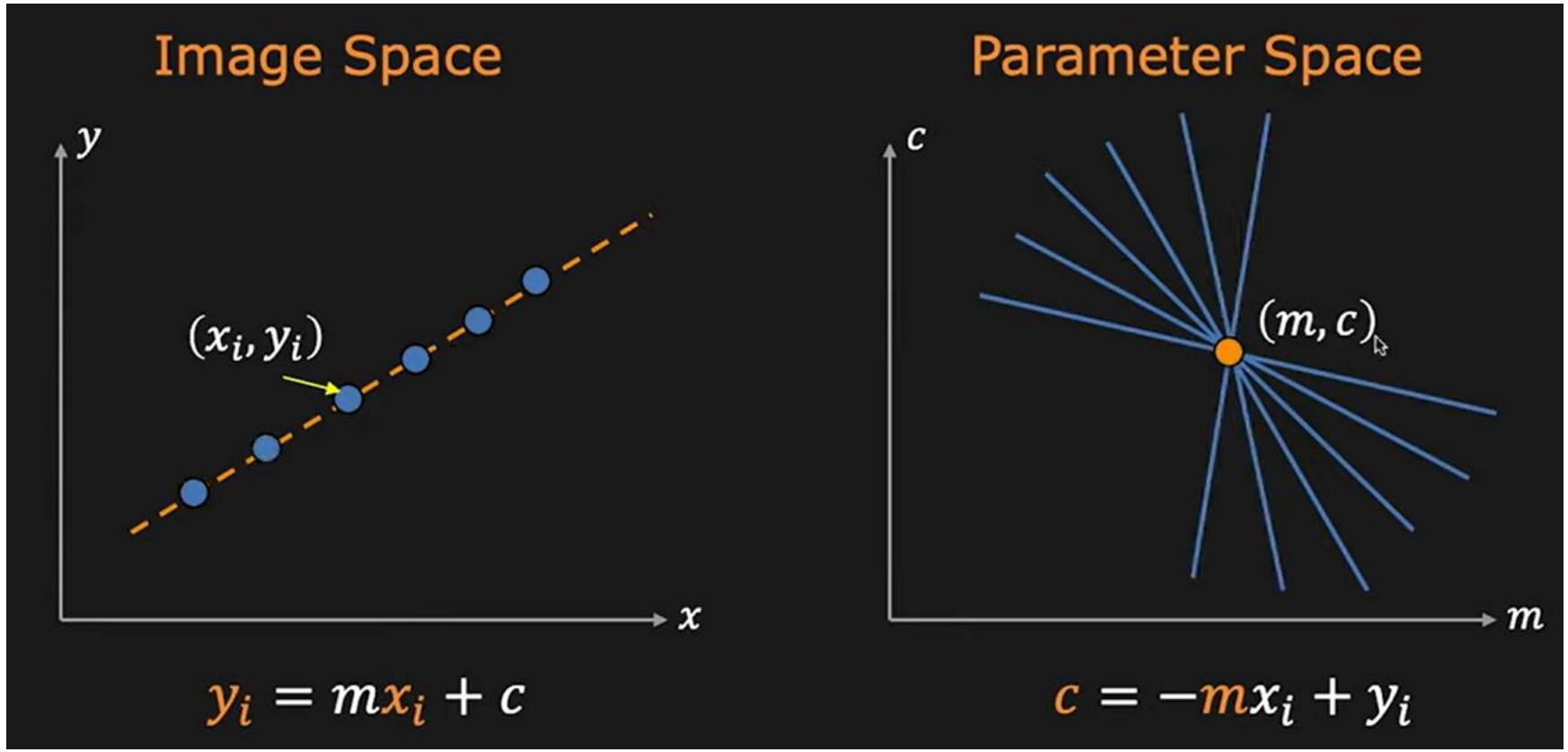
Hough Transform: Comments

- Works on disconnected edges
- Relatively insensitive to occlusion and noise
- Effective for simple shapes (lines, circles, etc.)
- Complex Shapes: Generalized Hough Transform
- Trade-off between work in image space and parameter space

Hough Transform: Line Detection (Parameter Space)



Hough Transform Concept



Hough Transform Concept