THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision

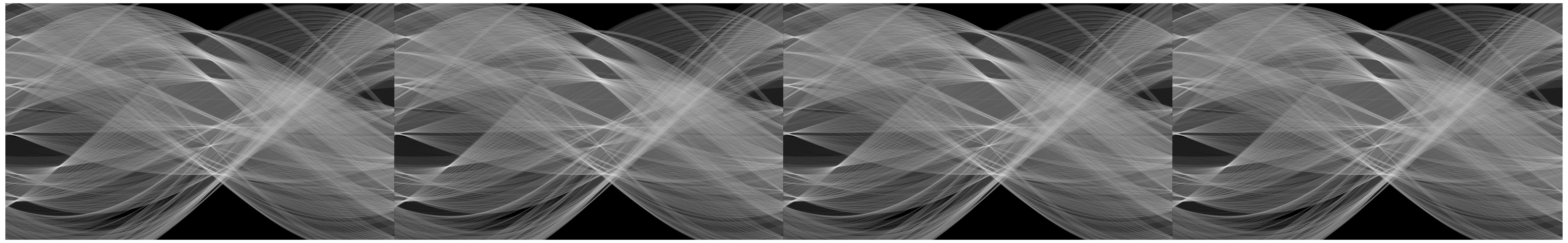


Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 14: Object Recognition, RANSAC, Hough Transform

Menu for Today (**March 3, 2020**)

Topics:

— Object Detection
— Model Fitting
— RANSAC
— Hough Transform

Readings:

— **Today's** & **Next** Lecture: Forsyth & Ponce (2nd ed.) 10.1, 10.2

Reminders:

— **Assignment 3**: is due today

— **Midterm** is being graded (grades are expected next week)

— **Assignment 4**: will be available tonight / tomorrow

Today's "**fun**" Example: Everybody Dance Now

Today's "**fun**" Example: Everybody Dance Now

DwNet: Dense warp-based network for pose-guided human video generation

Polina Zablotskaia, Aliaksandr Siarohin,
Bo Zhao and Leonid Sigal

Lecture 13: Re-Cap

— We motivated SIFT for identifying locally distinct keypoints in an image (**detection**)

— SIFT features (**description**) are invariant to translation, rotation, and scale; robust to 3D pose and illumination

1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

Lecture 13: Re-Cap

Keypoint is an image location at which a descriptor is computed

— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.

Lecture 13: Re-Cap

Keypoint is an image location at which a descriptor is computed

— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.

Lecture 13: Re-Cap

Keypoint is an image location at which a descriptor is computed

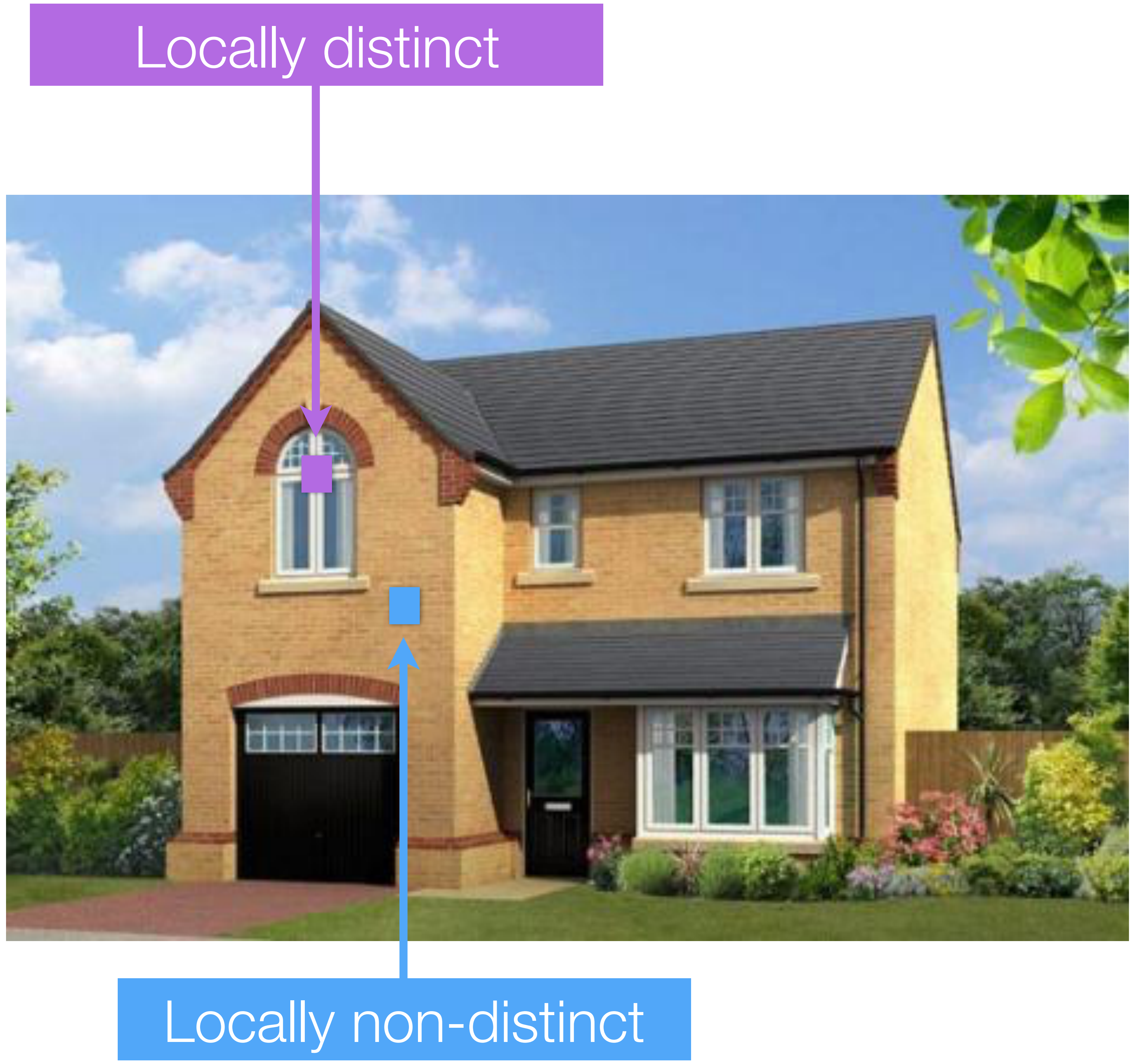
— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.



Lecture 13: Re-Cap

Keypoint is an image location at which a descriptor is computed

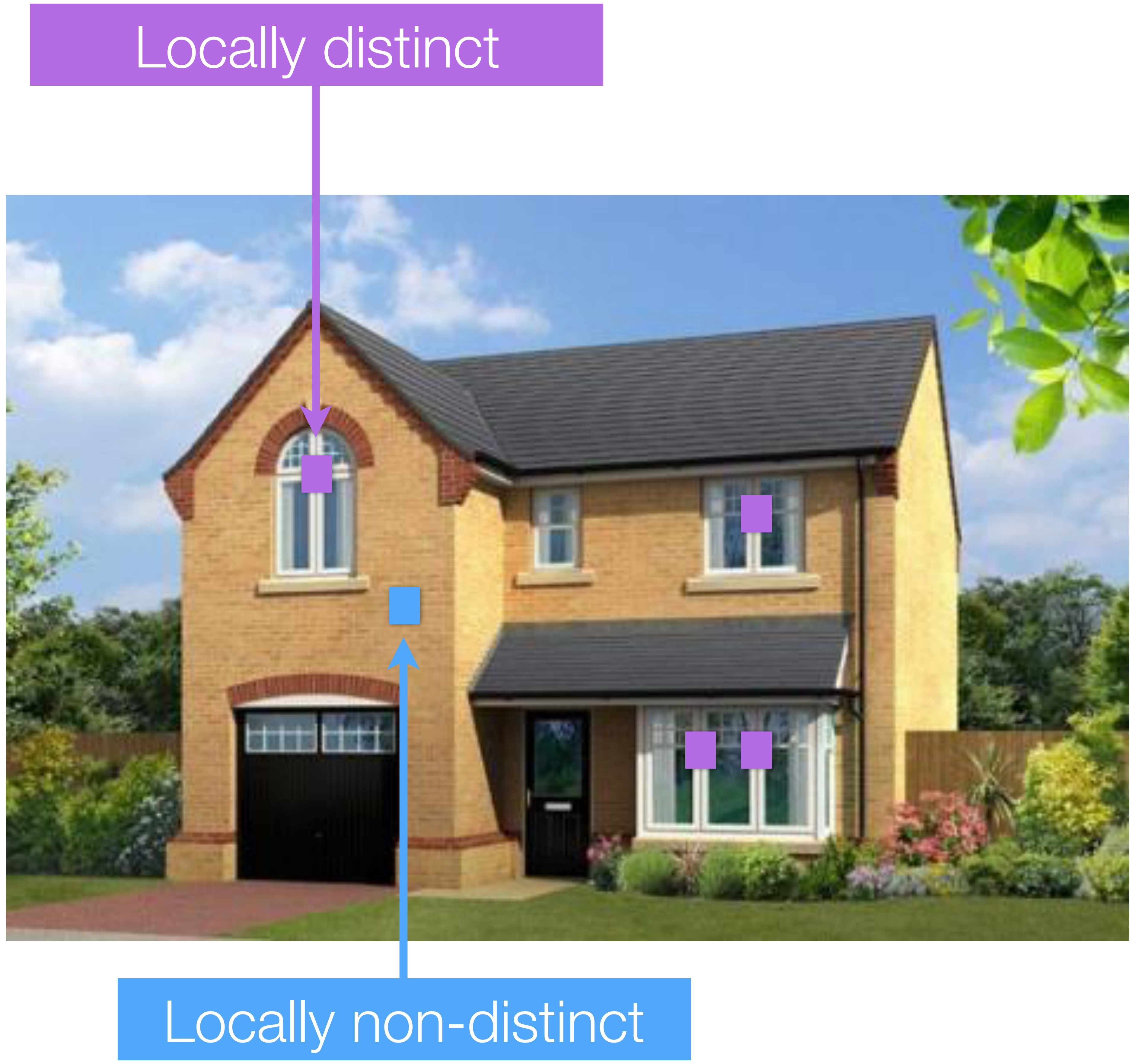
— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.



Lecture 13: Re-Cap

Keypoint is an image location at which a descriptor is computed

— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.

Lecture 13: Re-Cap

— We motivated SIFT for identifying locally distinct keypoints in an image (**detection**)

— SIFT features (**description**) are invariant to translation, rotation, and scale; robust to 3D pose and illumination

1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

Lecture 13: Re-Cap

Four steps to SIFT feature generation:

1. **Scale-space representation and local extrema detection**
   — use DoG pyramid
   — 3 scales/octave, down-sample by factor of 2 each octave
2. **Keypoint localization**
   — select stable keypoints (threshold on magnitude of extremum, ratio of principal curvatures)
3. **Keypoint orientation assignment**
   — based on histogram of local image gradient directions
4. **Keypoint descriptor**
   — histogram of local gradient directions — vector with $8 \times (4 \times 4) = 128$ dim
   — vector normalized (to unit length)

Lecture 13: Histogram of Oriented Gradients (**HOG**)

Pedestrian detection

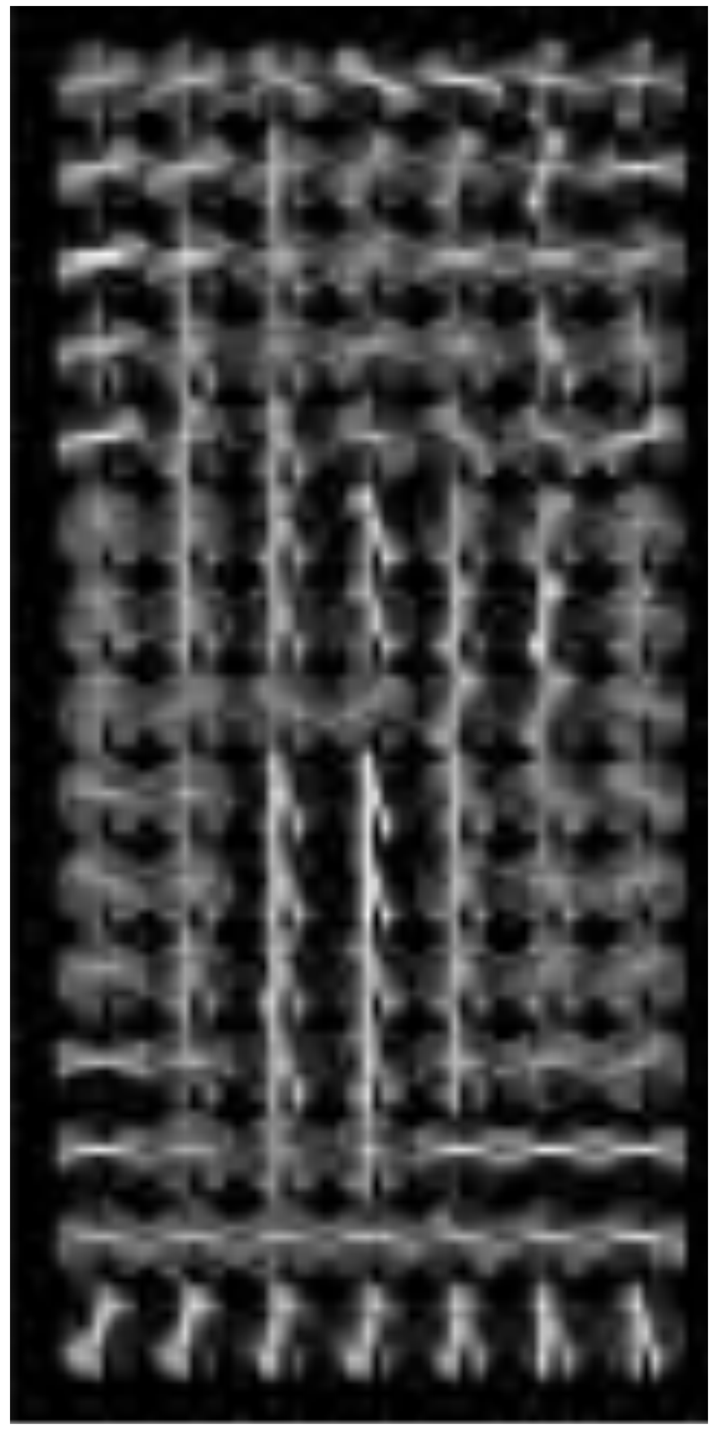
1 cell step size

128 pixels
16 cells
15 blocks

64 pixels
8 cells
7 blocks

15 x 7 x 4 x 9 =
3780

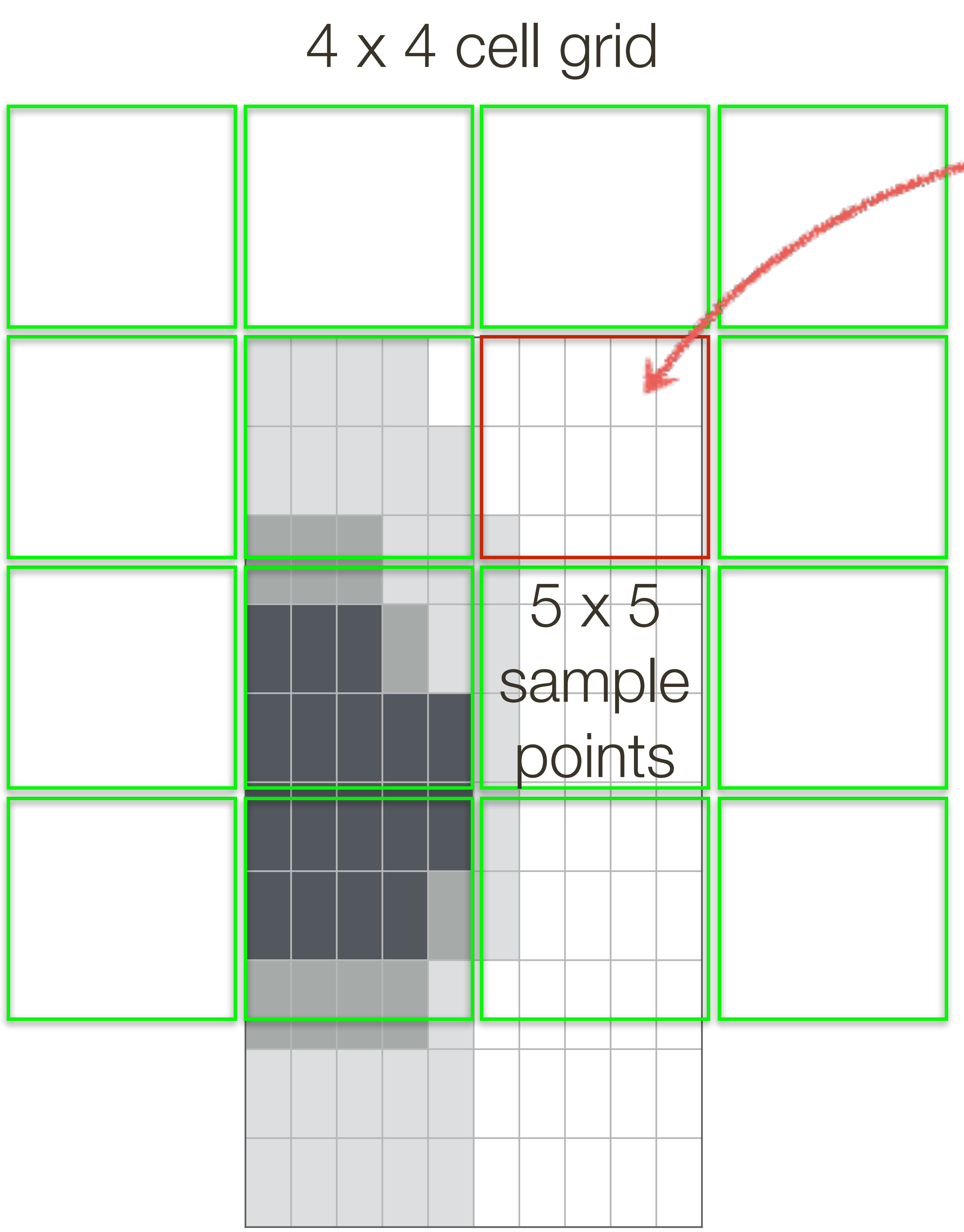
visualization



Redundant representation due to overlapping blocks

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

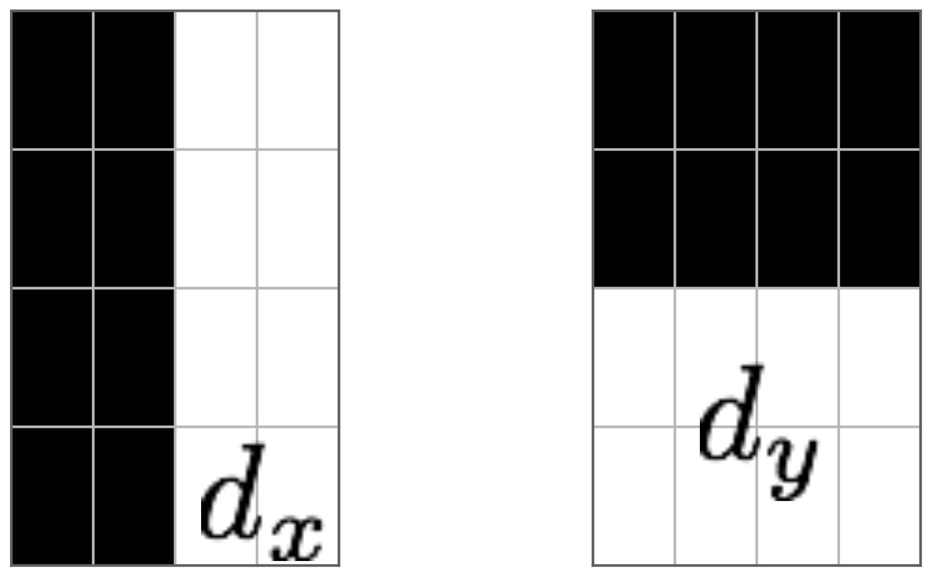
Lecture 13: ‘Speeded’ Up Robust Features



Each cell is represented by 4 values:

$$\left[\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right]$$

Haar wavelets filters
(Gaussian weighted from center)



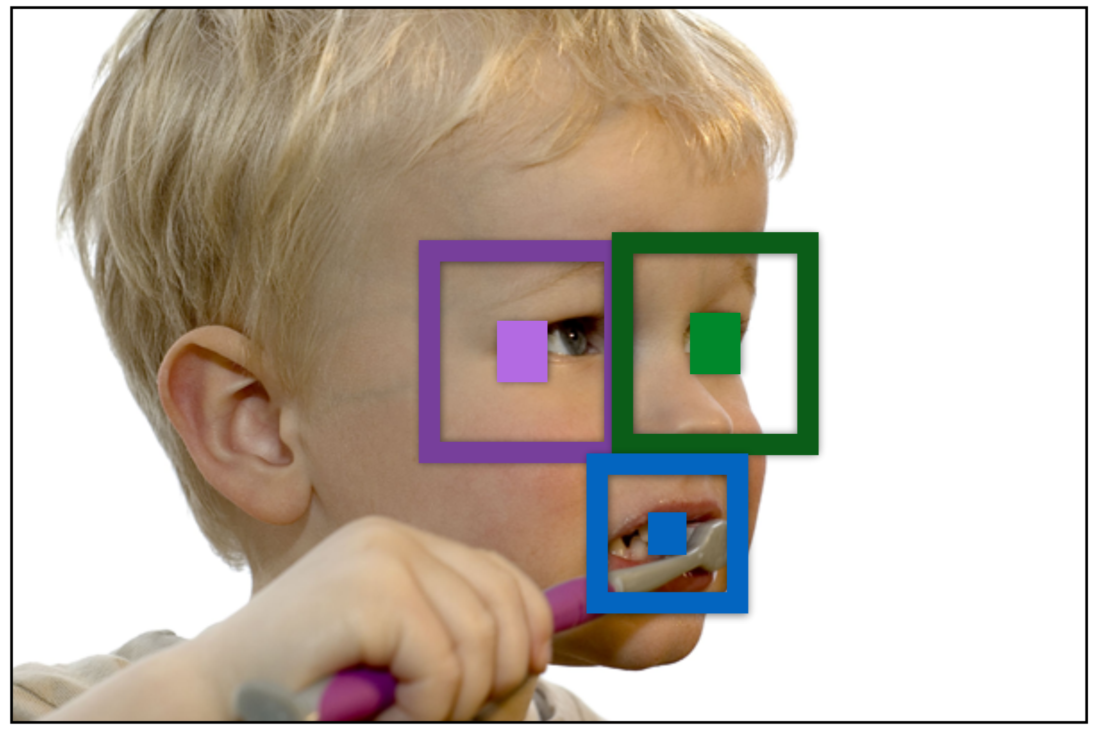
How big is the SURF descriptor?

64 dimensions

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

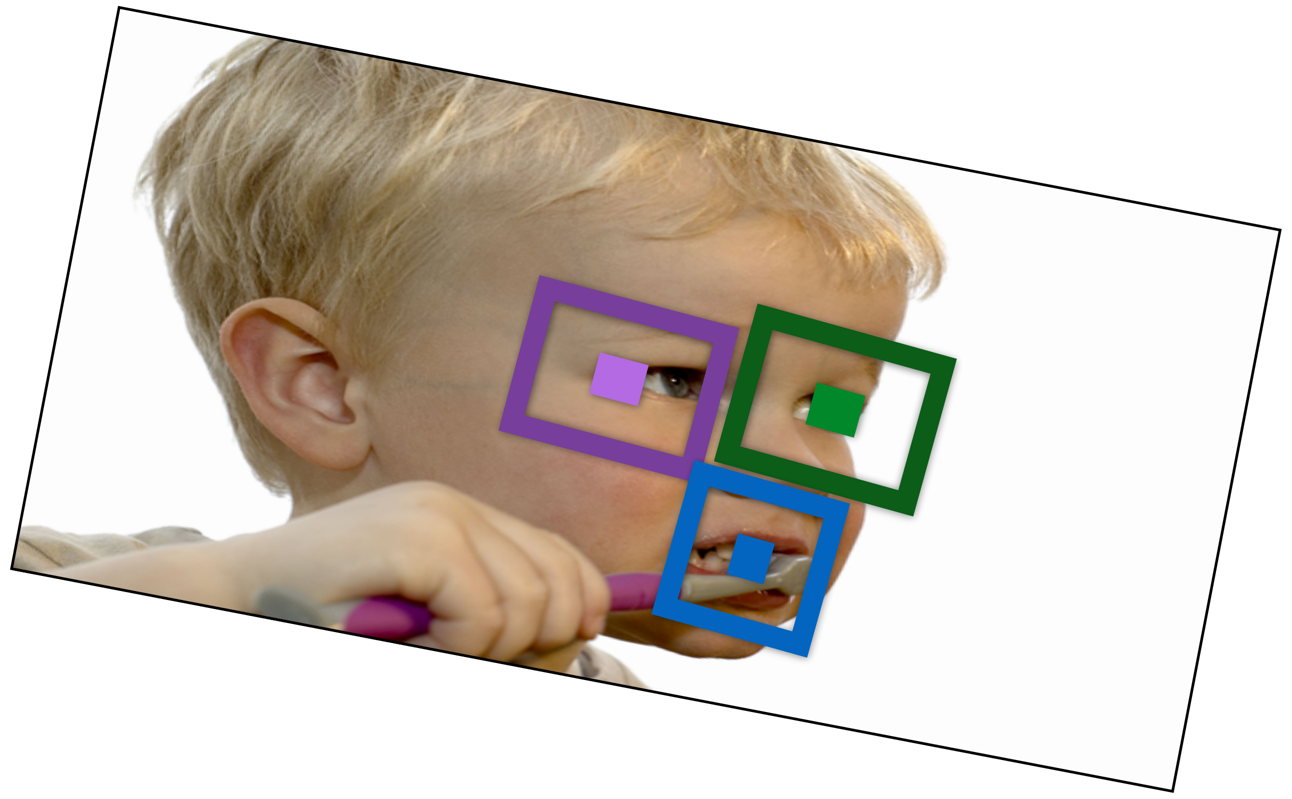
What types of **transformations** can we do?

$I(X,Y)$



Warping

$I'(X,Y)$

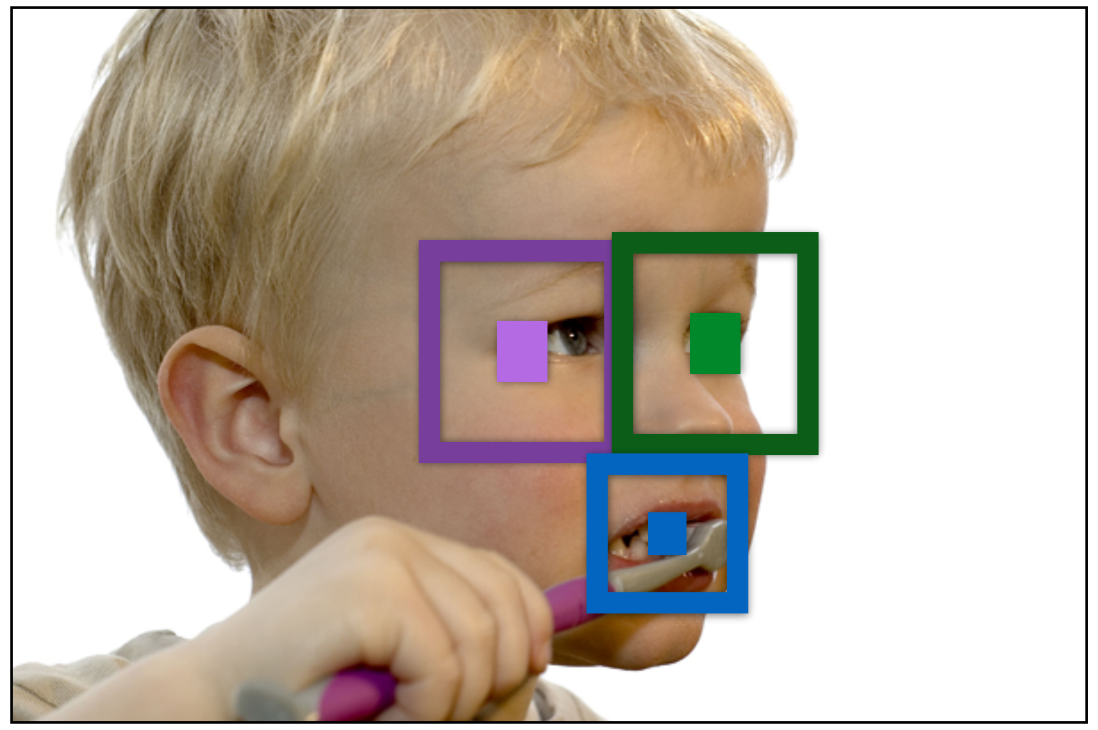


changes domain of image function

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

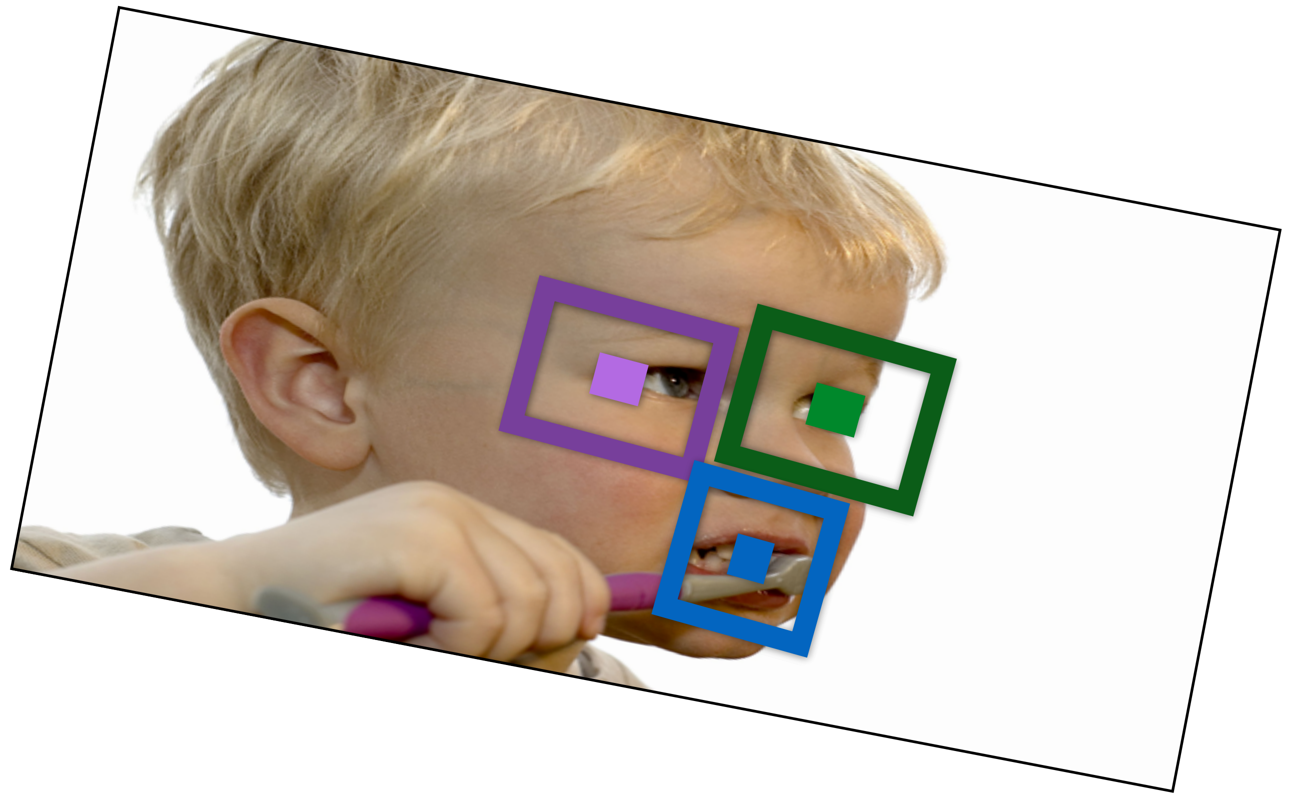
What types of **transformations** can we do?

$I(X,Y)$



Warping

$I'(X,Y)$



changes domain of image function

We will call this
"**Warping**" a "**Model**"

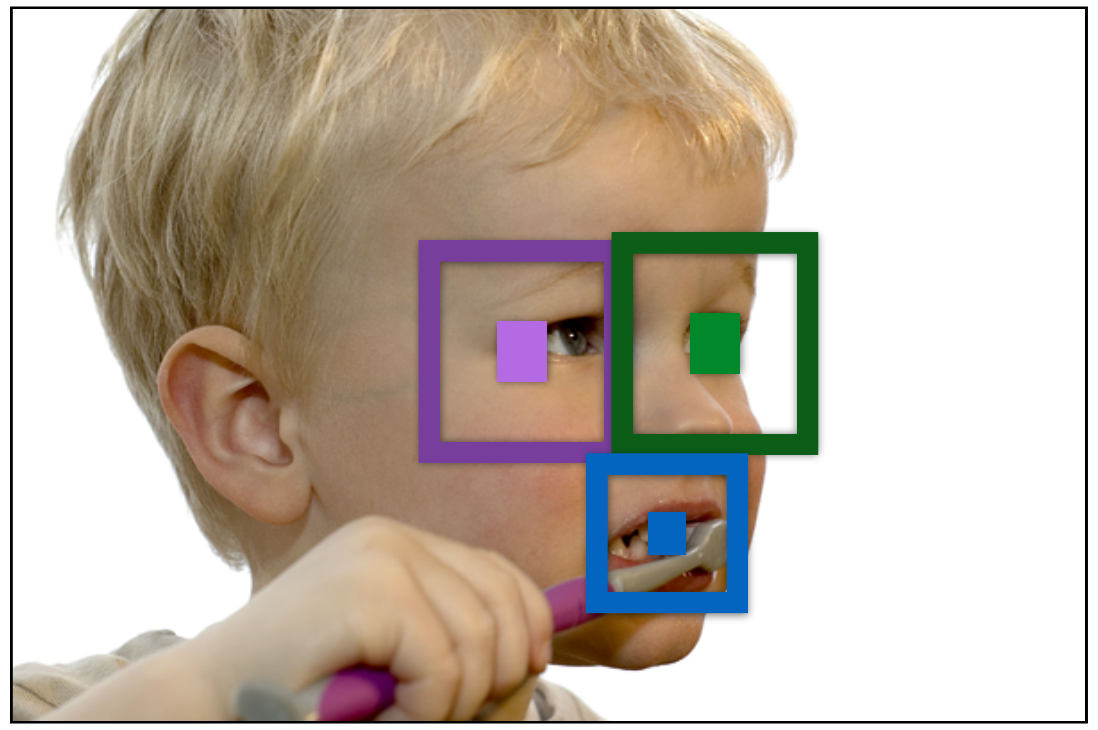
$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = M \begin{bmatrix} X \\ Y \end{bmatrix}$$

$$I'(X',Y') = I(X,Y)$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

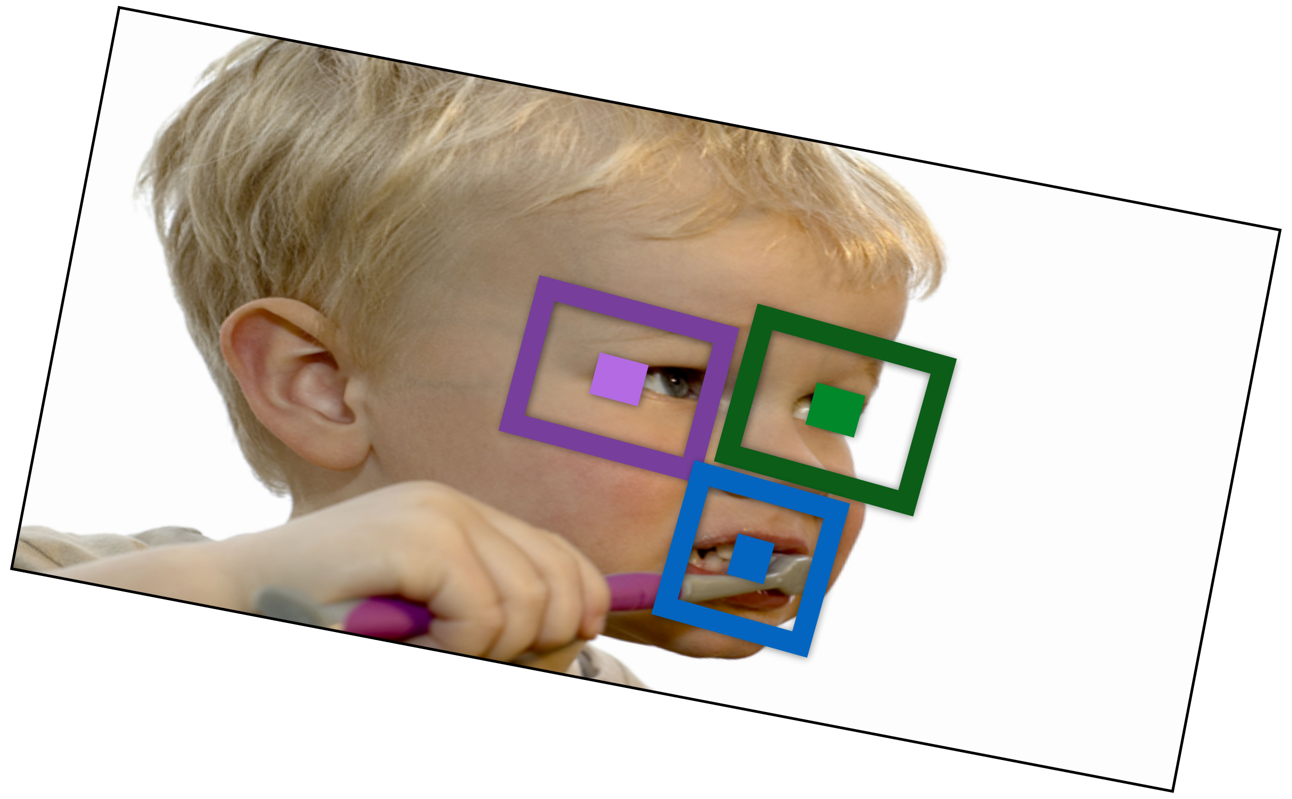
What types of **transformations** can we do?

$I(X,Y)$



Warping

$I'(X,Y)$



changes domain of image function

We will call this
"**Warping**" a "**Model**"

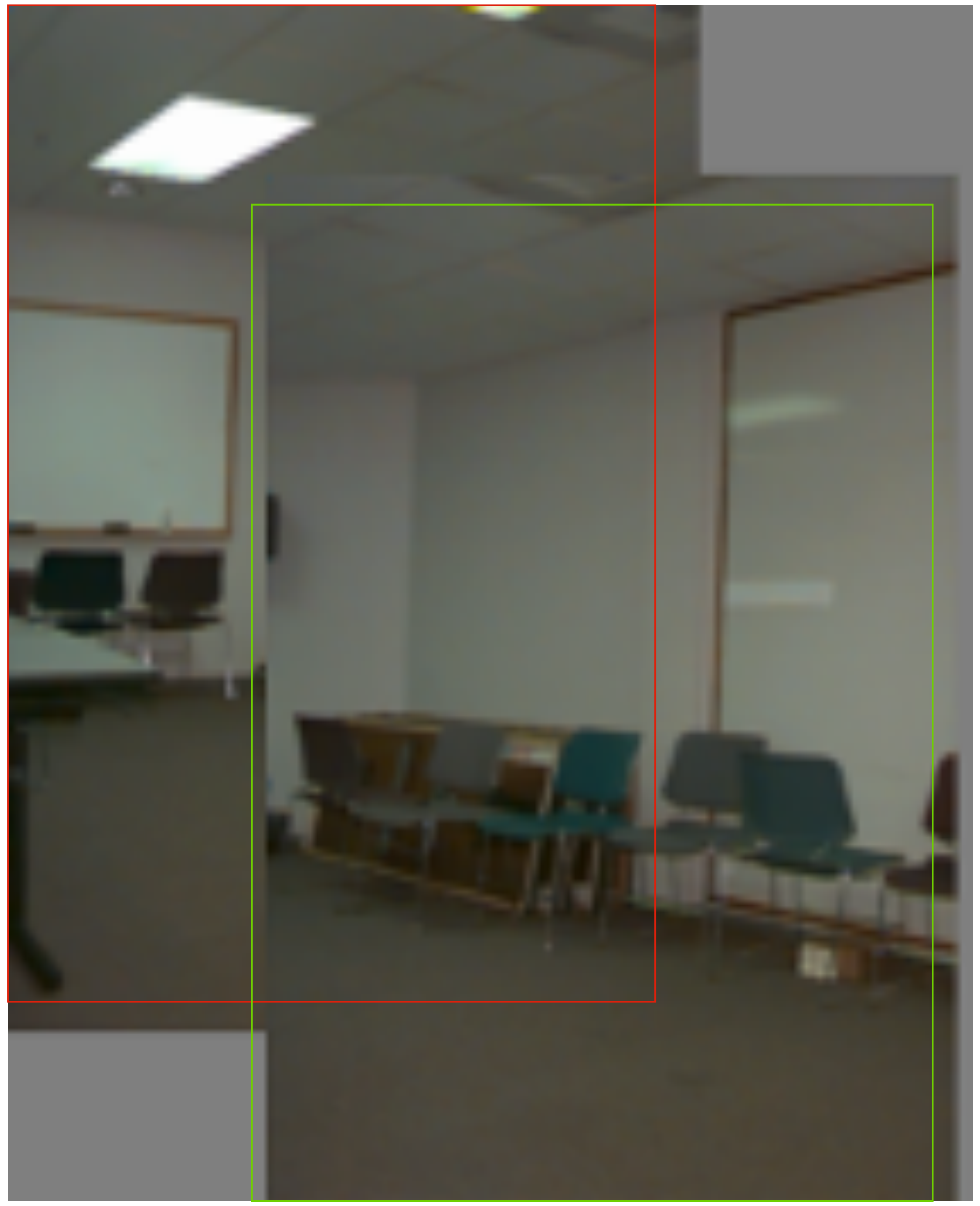
$$\begin{bmatrix} X' \\ Y' \\ 1 \end{bmatrix} = M \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$I'(X',Y') = I(X,Y)$$

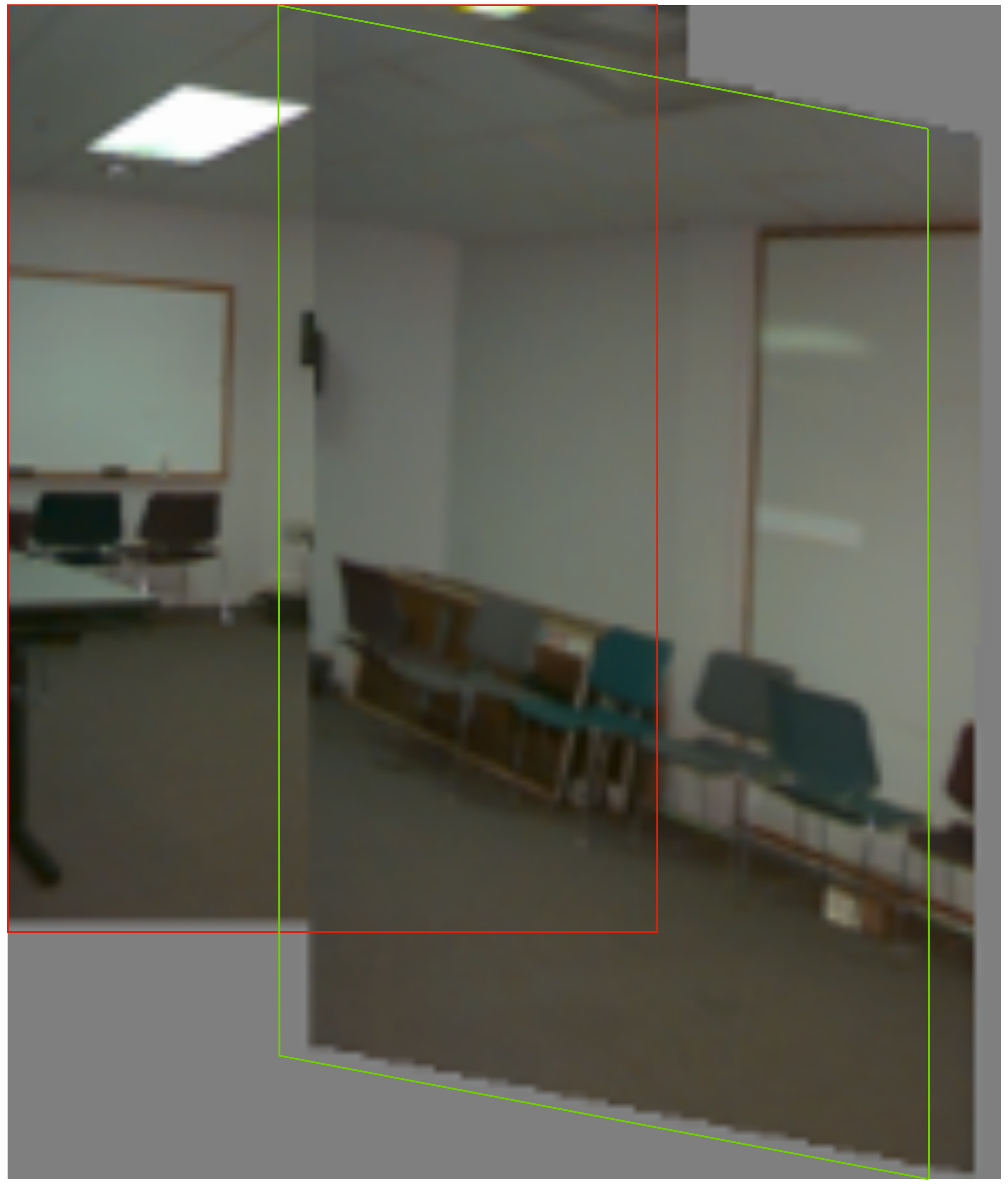
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: Warping with Different Transformations

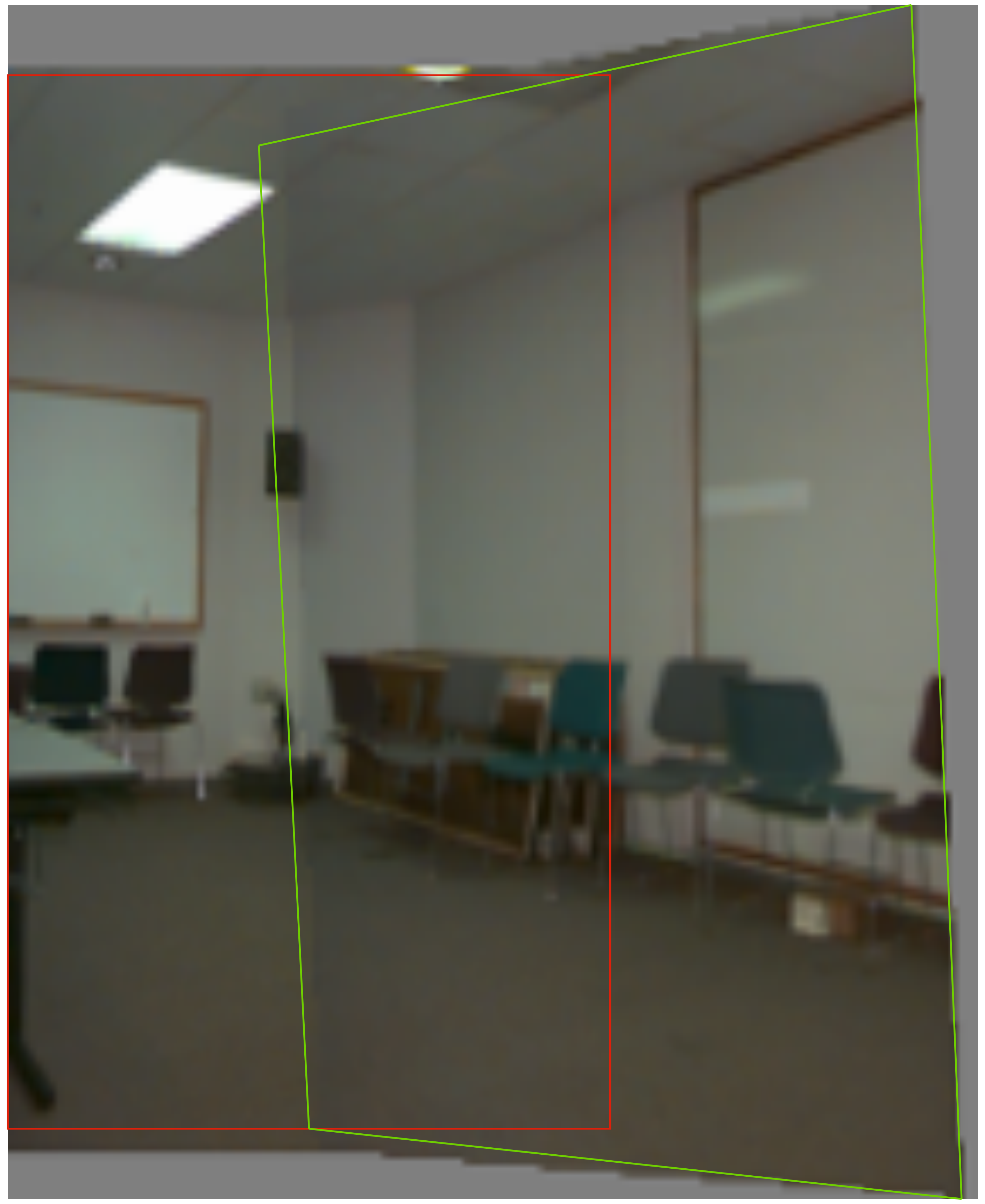
Translation



Affine

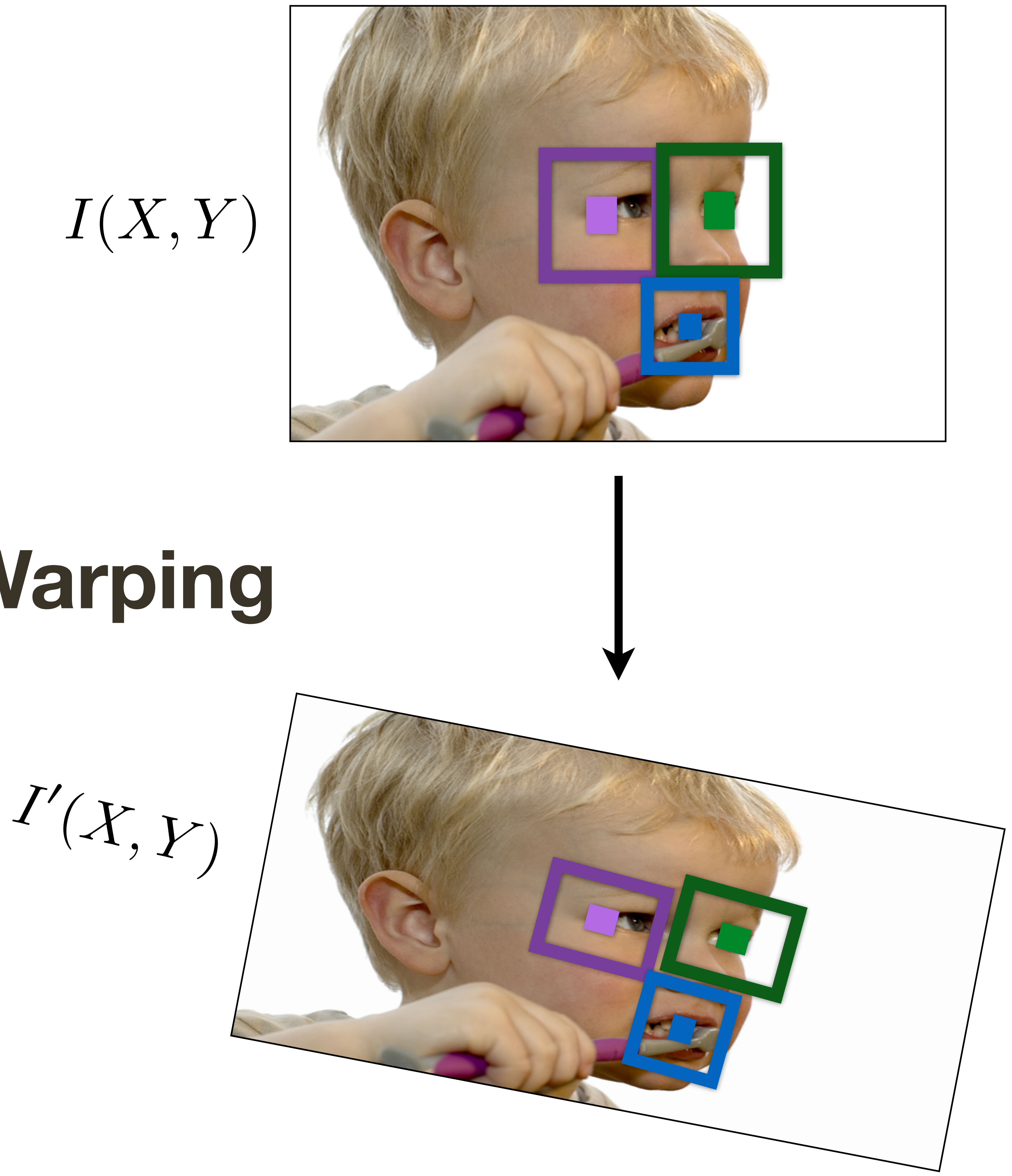


Projective
(homography)



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

What types of **transformations** can we do?



changes domain of image function

We will call this
"**Warping**" a "**Model**"

$$\begin{bmatrix} X' \\ Y' \\ 1 \end{bmatrix} = M \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Affine "Model"

$$M = \begin{bmatrix} m_1 & m_2 & t_x \\ m_3 & m_4 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Solution for **Affine** Parameters

Affine transform of $[x, y]$ to $[u, v]$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Rewrite to solve for **transformation** parameters:

$$\begin{bmatrix} x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_1 & y_1 & 0 & 1 \\ x_2 & y_2 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_2 & y_2 & 0 & 1 \\ & & \cdots & \cdots & & \\ & & \cdots & \cdots & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \cdots \\ \cdots \end{bmatrix}$$

(6 equations 6 unknowns)

Solution for **Affine** Parameters

Suppose we have $k \geq 3$ matches, $[x_i, y_i]$ to $[u_i, v_i]$, $i = 1, 2, \cdots, k$

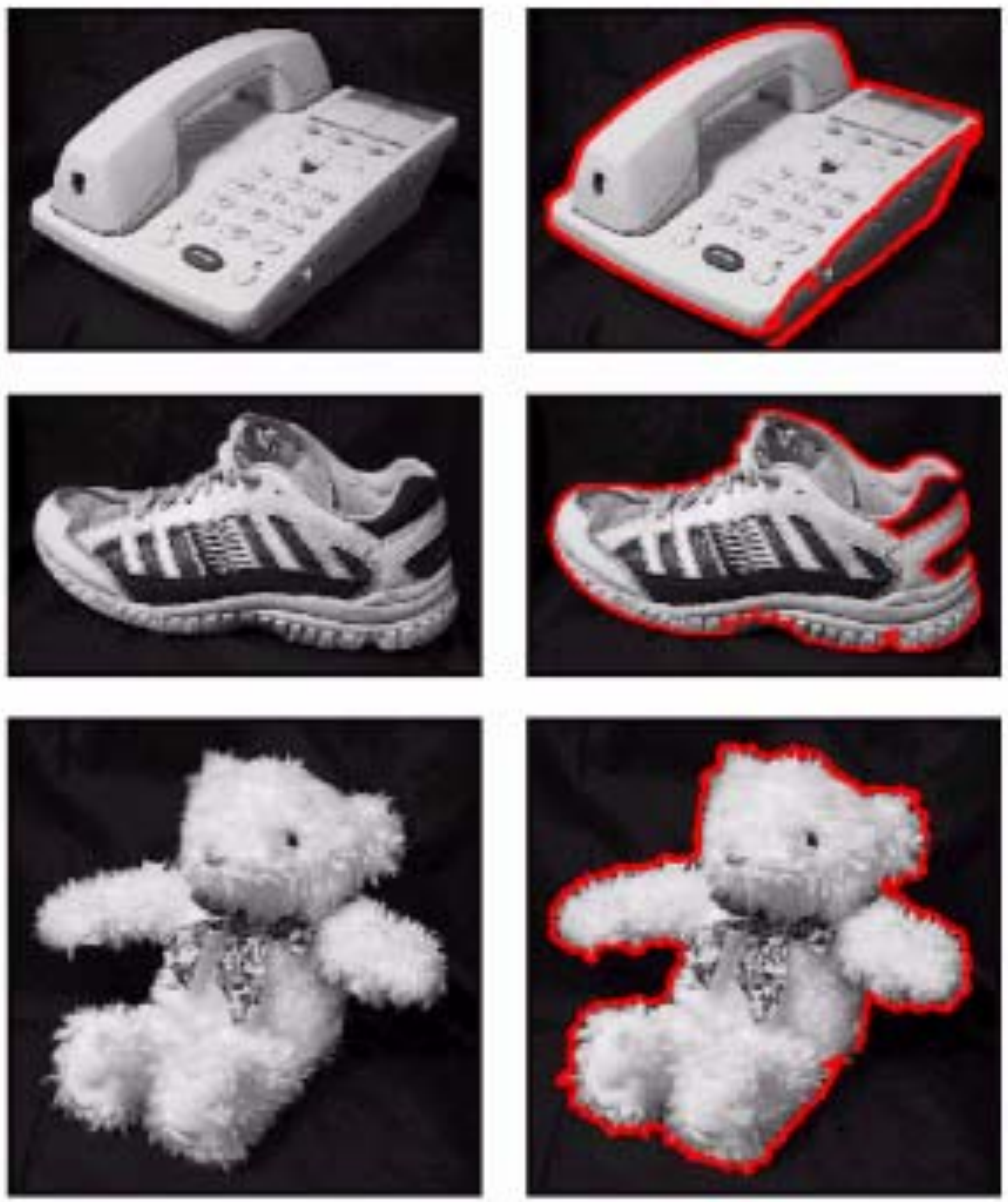
Then,

$$\begin{bmatrix} x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_1 & y_1 & 0 & 1 \\ x_2 & y_2 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_2 & y_2 & 0 & 1 \\ & & \cdots & \cdots & & \\ & & \cdots & \cdots & & \\ x_k & y_k & 0 & 0 & 1 & 0 \\ 0 & 0 & x_k & y_k & 0 & 1 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \cdots \\ \cdots \\ u_k \\ v_k \end{bmatrix}$$

Limitation of this …

We need to have **exact** matches

3D **Object Recognition**

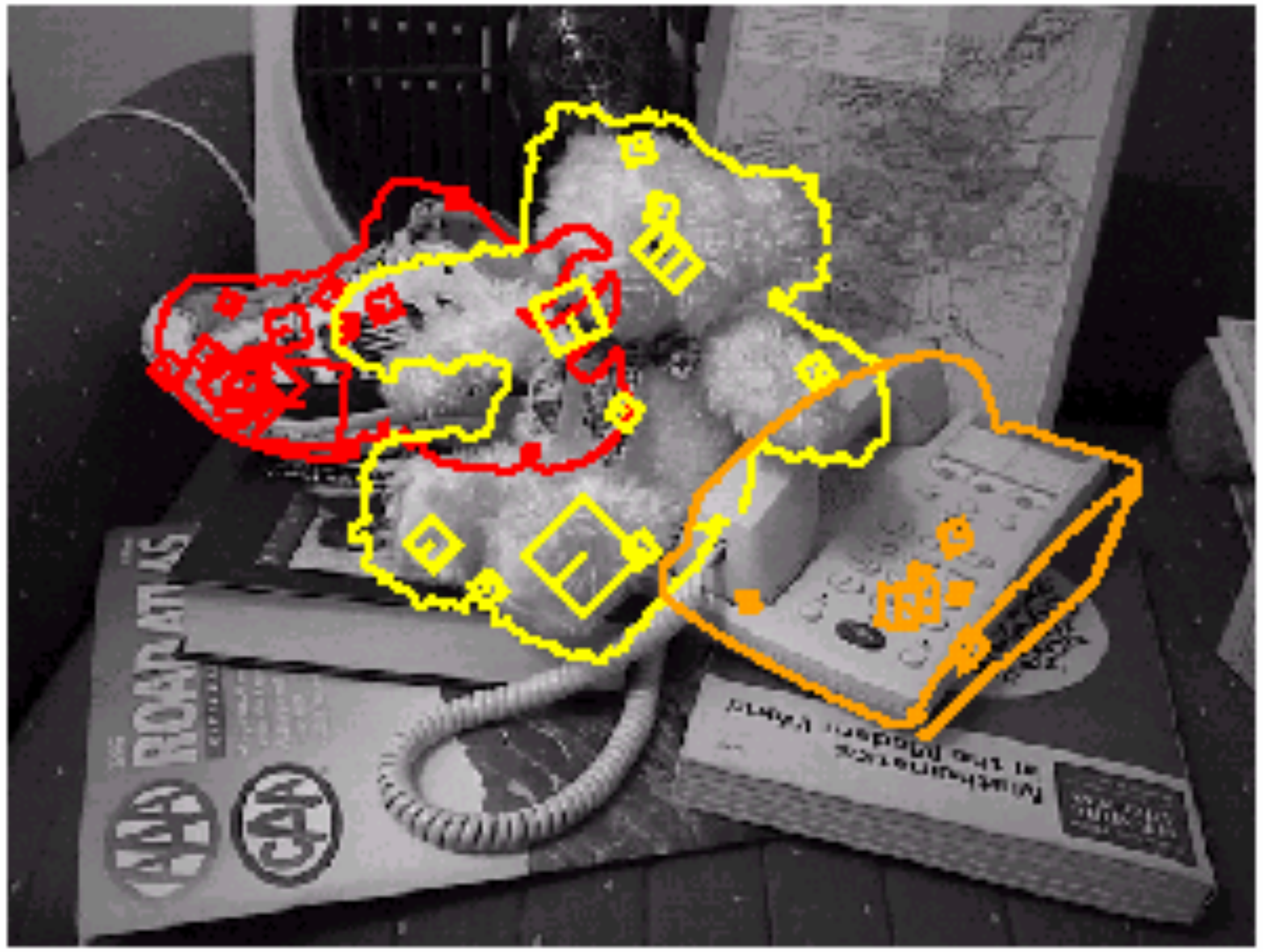


Extract outlines with background subtraction

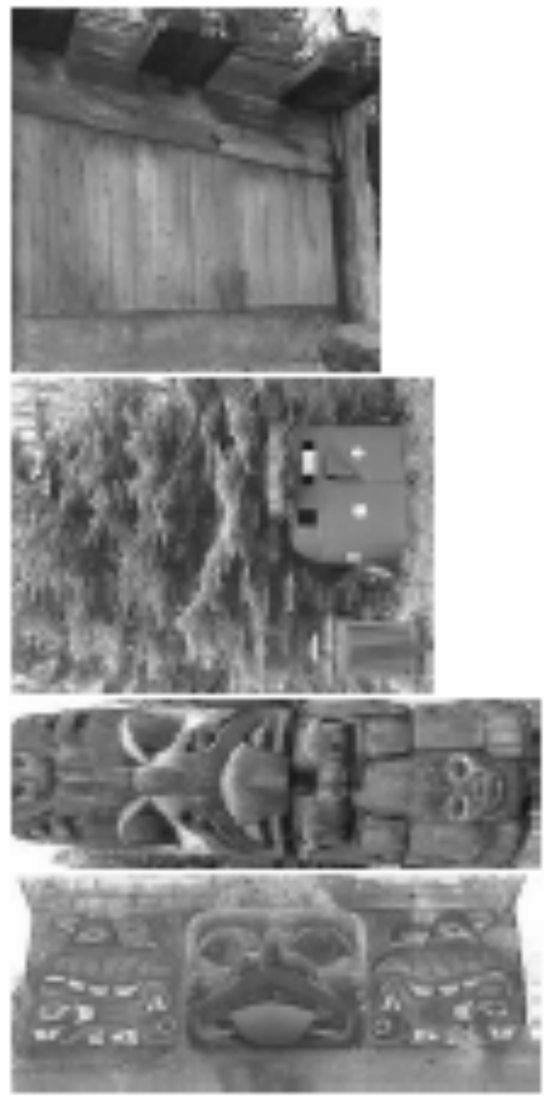
3D **Object Recognition**

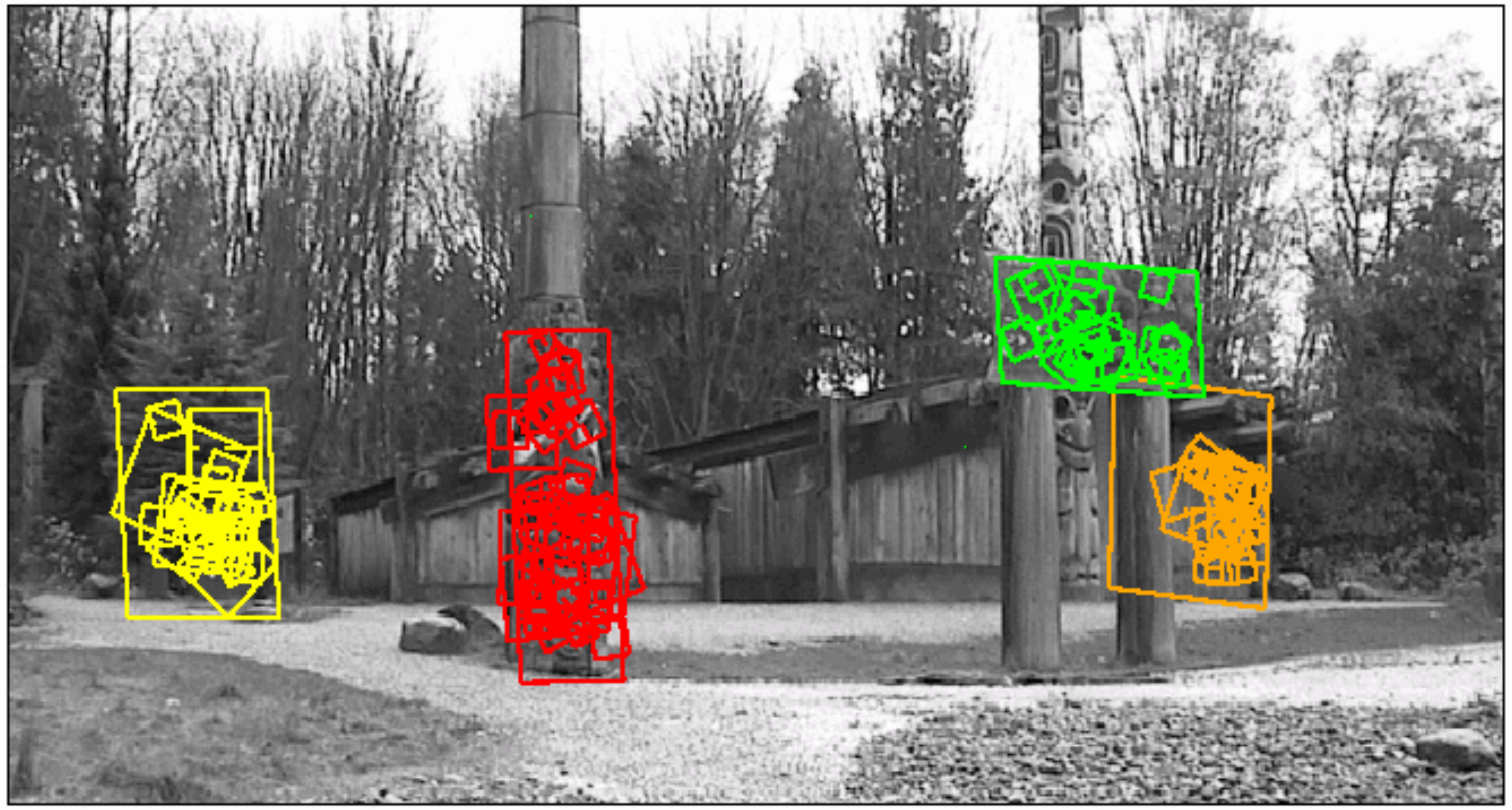
Only 3 keypoints are needed for recognition, so extra keypoints provide robustness

Recognition Under **Occlusion**



Location Recognition





Example 1: Sony Aibo

SIFT Usage

- Recognize charging station
- Communicate with visual cards

Summary of Object Recognition with SIFT

Match each keypoint independently to database of known keypoints extracted from “training” examples

— use fast (approximate) nearest neighbour matching

— threshold based on ratio of distances to best and to second best match

Identify **clusters of (at least) 3 matches** that agree on an object and a similarity pose

— use generalized Hough transform

Check each cluster found by performing detailed geometric fit of affine transformation to the model

— accept/reject interpretation accordingly

Limitation of this …

We need to have <u>**exact**</u> matches

Fitting a Model to Noisy Data

Suppose we are **fitting a line** to a dataset that consists of 50% outliers

We can fit a line using two points

If we draw pairs of points uniformly at random, what fraction of pairs will consist entirely of 'good' data points (inliers)?

Fitting a Model to Noisy Data

Suppose we are **fitting a line** to a dataset that consists of 50% outliers

We can fit a line using two points

— If we draw pairs of points uniformly at random, then about 1/4 of these pairs will consist entirely of ‘good’ data points (inliers)

— We can identify these good pairs by noticing that a large collection of other points lie close to the line fitted to the pair

— A better estimate of the line can be obtained by refitting the line to the points that lie close to the line

RANSAC (**RAN**dom **SA**mple **C**onsensus)

1. Randomly choose minimal subset of data points necessary to fit model (a **sample**)

2. Points within some distance threshold, t, of model are a **consensus set**. Size of consensus set is model's **support**

3. Repeat for N samples; model with biggest support is most robust fit
   - Points within distance t of best model are inliers
   - Fit final model to all inliers

Slide Credit: Christopher Rasmussen

RANSAC (**RAN**dom **SA**mple **C**onsensus)

1. Randomly choose minimal subset of data points necessary to fit model (a **sample**)

2. Points within some distance threshold, t, of model are a **consensus set**. Size of consensus set is model's **support**

3. Repeat for N samples; model with biggest support is most robust fit
   — Points within distance t of best model are inliers
   — Fit final model to all inliers

RANSAC is very useful for variety of applications

Slide Credit: Christopher Rasmussen

RANSAC (**RAN**dom **SA**mple **C**onsensus)

1. Randomly choose minimal subset of data points necessary to fit model (a **sample**)

Fitting a Line: 2 points

2. Points within some distance threshold, t, of model are a **consensus set**. Size of consensus set is model's **support**

3. Repeat for N samples; model with biggest support is most robust fit
   — Points within distance t of best model are inliers
   — Fit final model to all inliers

Slide Credit: Christopher Rasmussen

Example 1: Fitting a Line

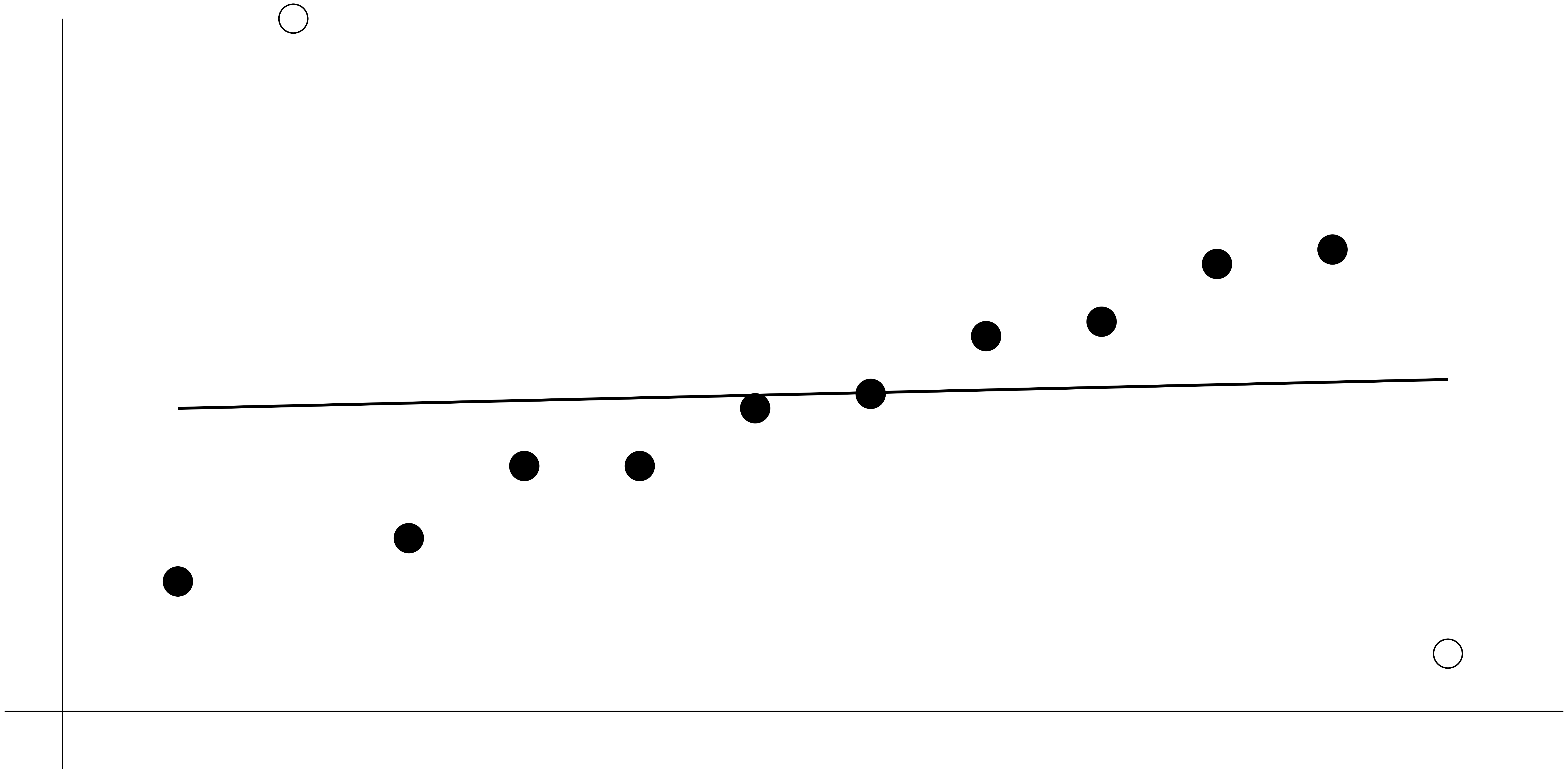


Figure Credit: Hartley & Zisserman

Example 1: Fitting a Line

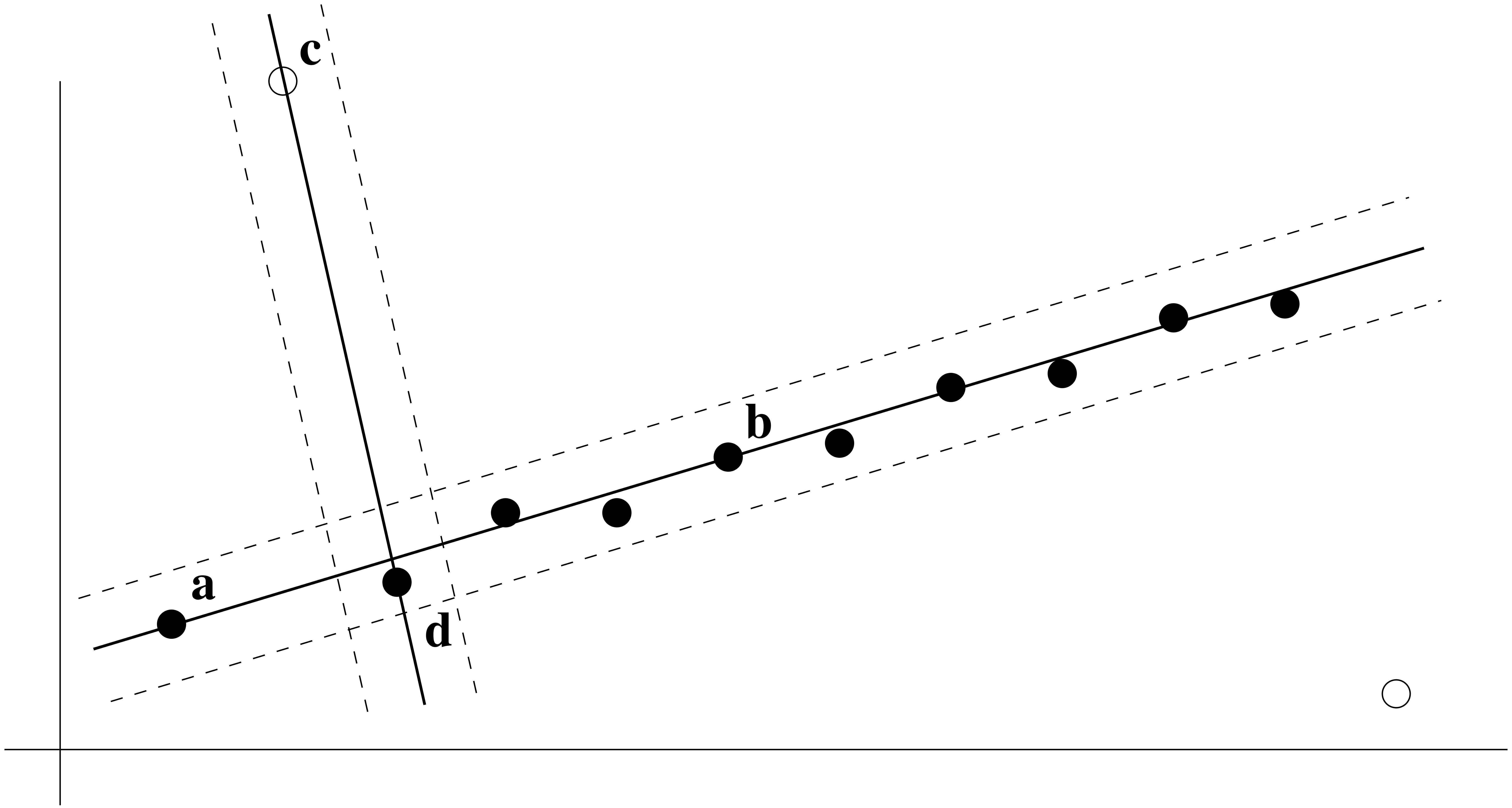


Figure Credit: Hartley & Zisserman

Example 1: Fitting a Line

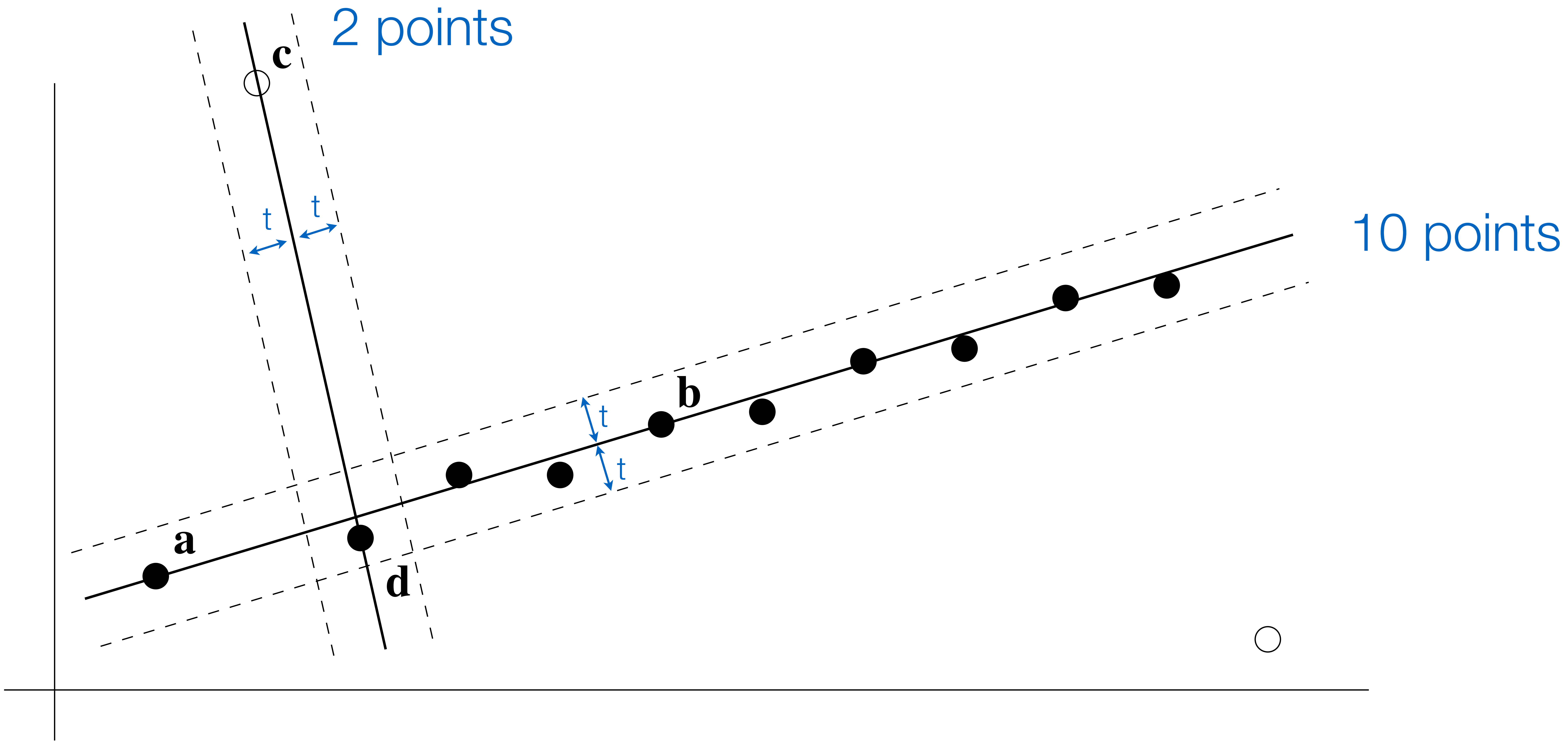


Figure Credit: Hartley & Zisserman

Algorithm 10.4

This was Algorithm 15.4 in Forsyth & Ponce (1st ed.)

Algorithm 15.4: RANSAC: fitting lines using random sample consensus

```
Determine:
    n — the smallest number of points required
    k — the number of iterations required
    t — the threshold used to identify a point that fits well
    d — the number of nearby points required
      to assert a model fits well
Until k iterations have occurred
    Draw a sample of n points from the data
      uniformly and at random
    Fit to that set of n points
    For each data point outside the sample
      Test the distance from the point to the line
        against t; if the distance from the point to the line
        is less than t, the point is close
    end
    If there are d or more points close to the line
      then there is a good fit. Refit the line using all
      these points.
end
Use the best fit from this collection, using the
  fitting error as a criterion
```

RANSAC: Fitting Lines Using Random Sample Consensus

RANSAC: How many samples?

Let ω be the fraction of inliers (i.e., points on line)

Let n be the number of points needed to define hypothesis
$(n = 2$ for a line in the plane)

Suppose k samples are chosen

The probability that a single sample of n points is correct (all inliers) is

RANSAC: How many samples?

Let ω be the fraction of inliers (i.e., points on line)

Let n be the number of points needed to define hypothesis
($n = 2$ for a line in the plane)

Suppose k samples are chosen

The probability that a single sample of n points is correct (all inliers) is

$$\omega^n$$

The probability that all k samples fail is

RANSAC: How many samples?

Let ω be the fraction of inliers (i.e., points on line)

Let n be the number of points needed to define hypothesis
$(n = 2$ for a line in the plane)

Suppose k samples are chosen

The probability that a single sample of n points is correct (all inliers) is

$$\omega^n$$

The probability that all k samples fail is

$$(1 - \omega^n)^k$$

Choose k large enough (to keep this below a target failure rate)

RANSAC: *k* Samples Chosen (p = 0.99)

Sample size	Proportion of outliers						
n	**5%**	**10%**	**20%**	**25%**	**30%**	**40%**	**50%**
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Figure Credit: Hartley & Zisserman

After RANSAC

RANSAC divides data into inliers and outliers and yields estimate computed from minimal set of inliers

Improve this initial estimate with estimation over all inliers (e.g., with standard least-squares minimization)

But this may change inliers, so alternate fitting with re-classification as inlier/outlier

Example 2: Fitting a Line

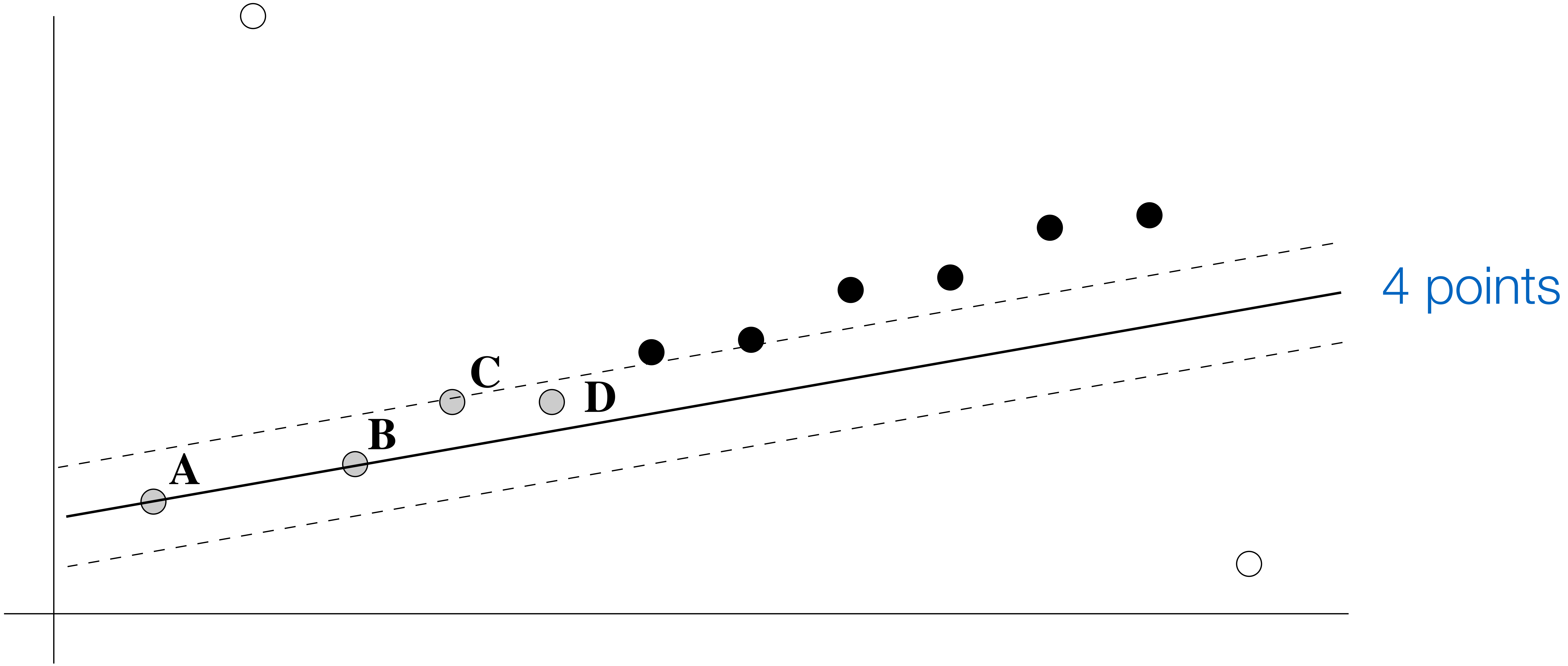


Figure Credit: Hartley & Zisserman

Example 2: Fitting a Line

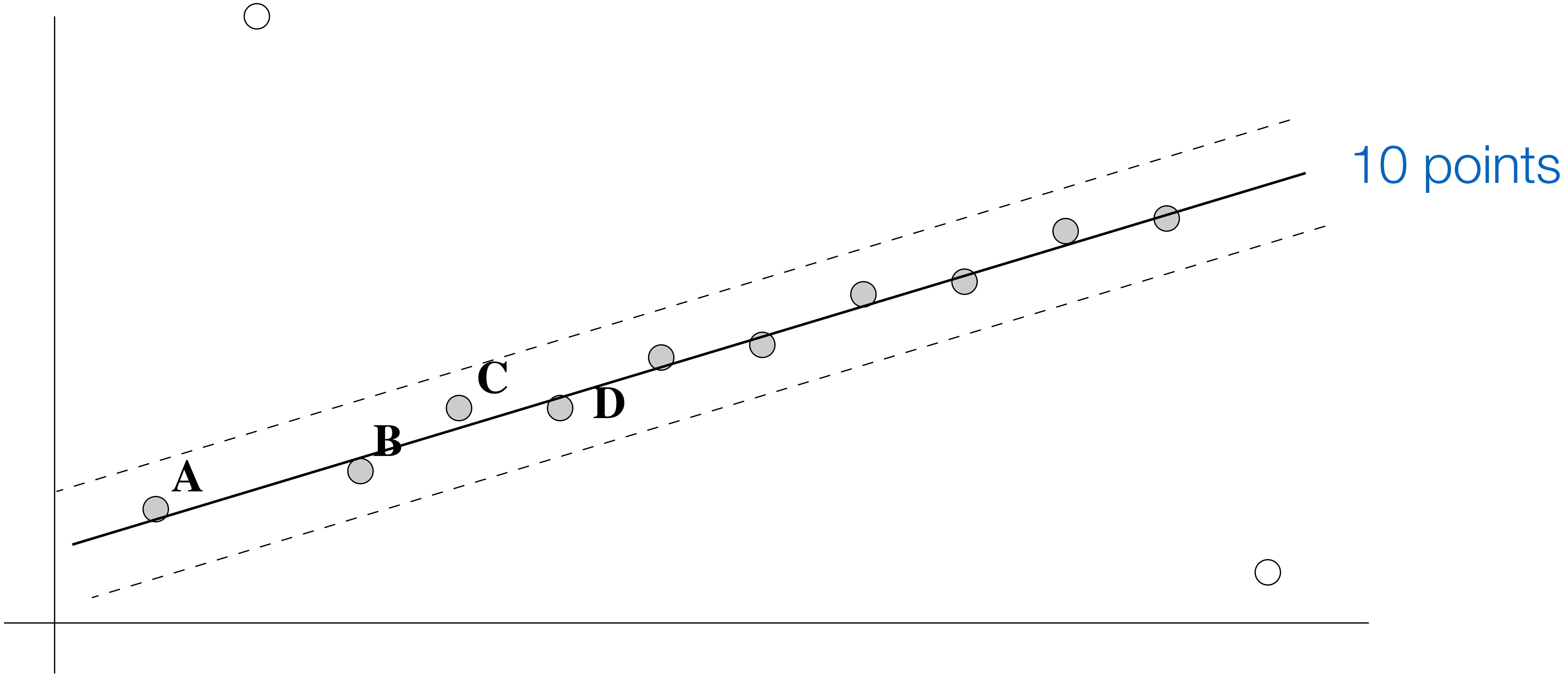


Figure Credit: Hartley & Zisserman

Example 3: Automatic Matching of Images

— How to get correct correspondences without human intervention?
— Can be used for image stitching or automatic determination of epipolar geometry

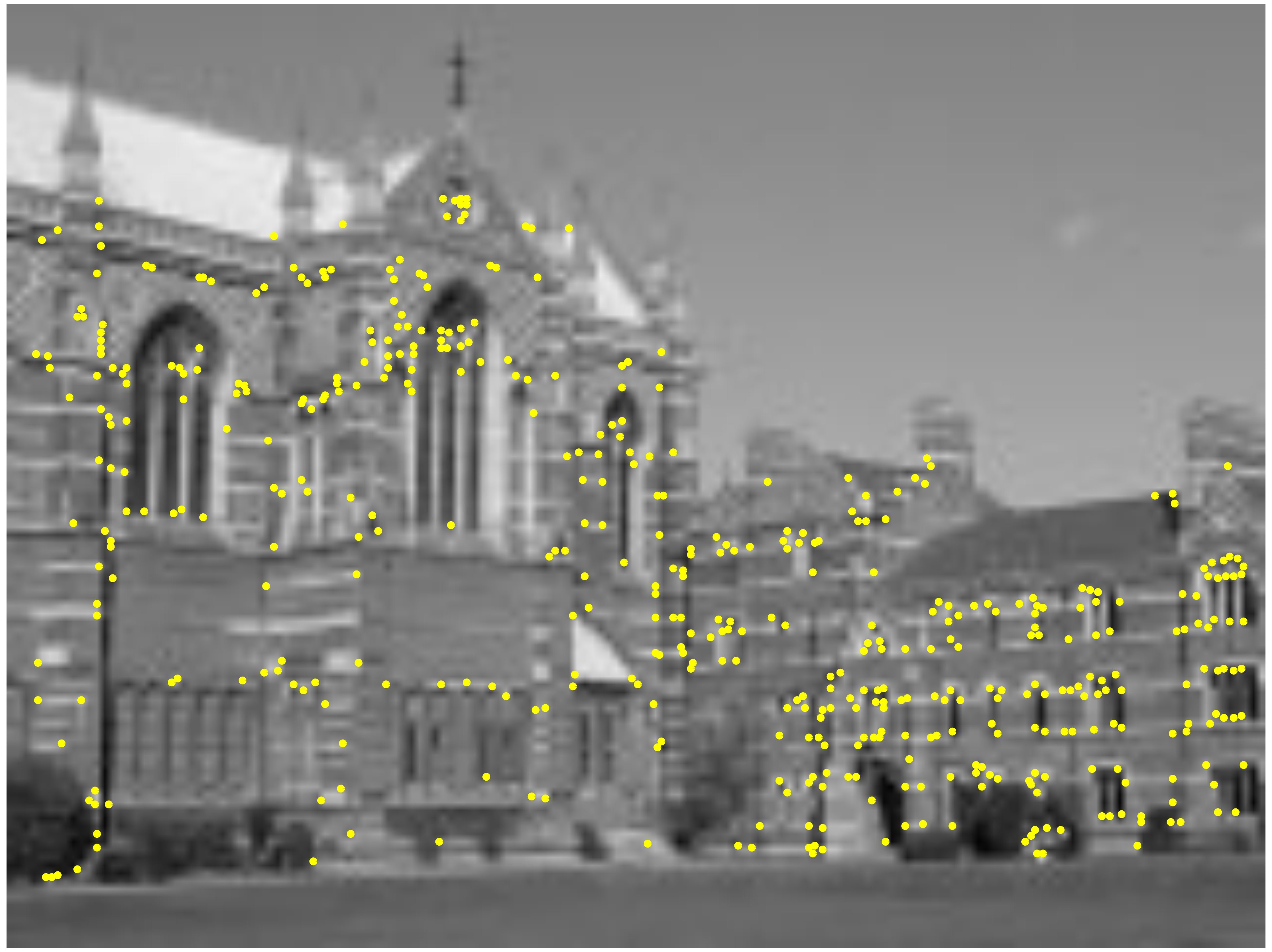
 

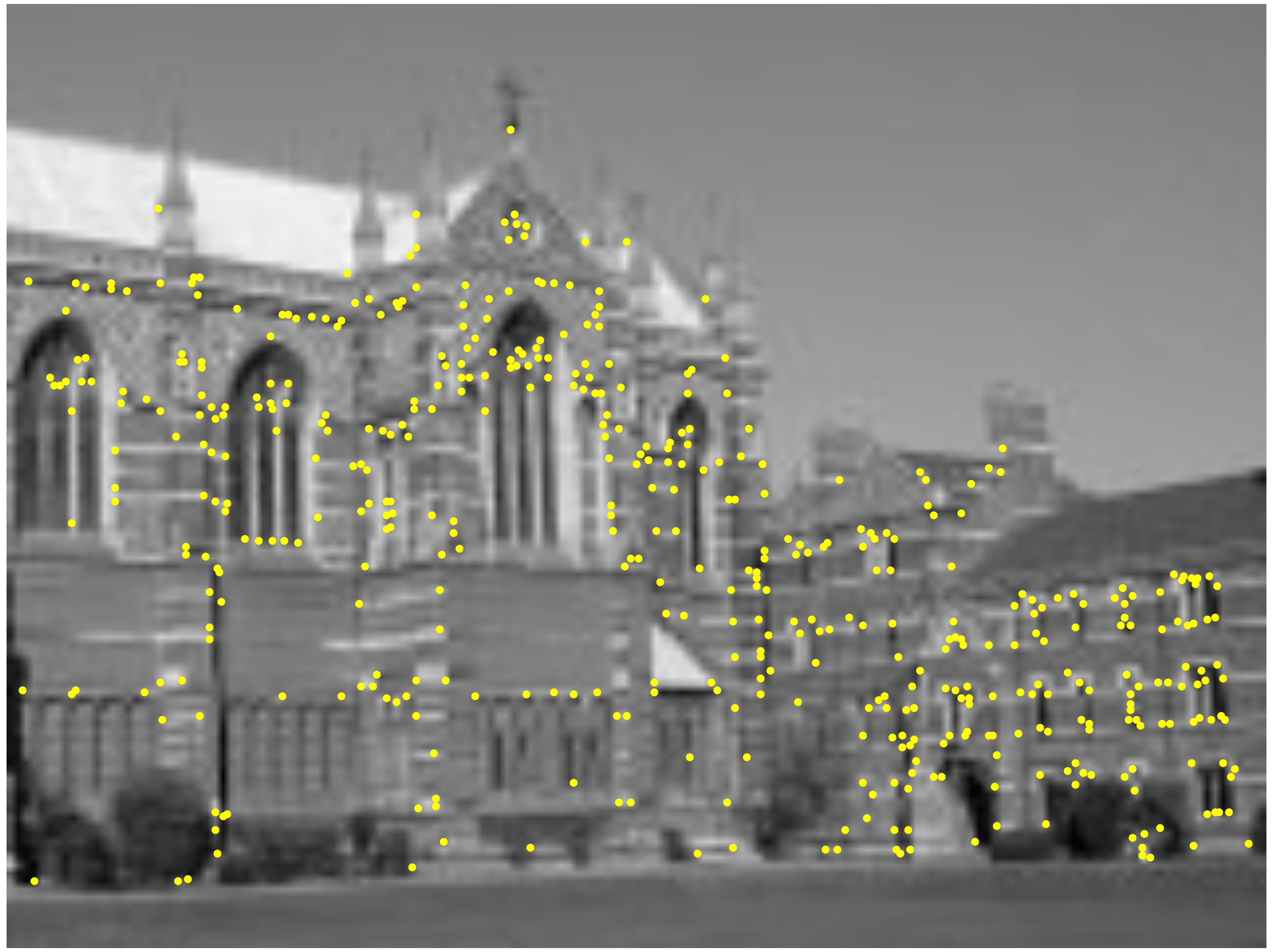
Figure Credit: Hartley & Zisserman

Example 3: Feature Extraction

— Find features in pair of images using Harris corner detector

— Assumes images are roughly the same scale



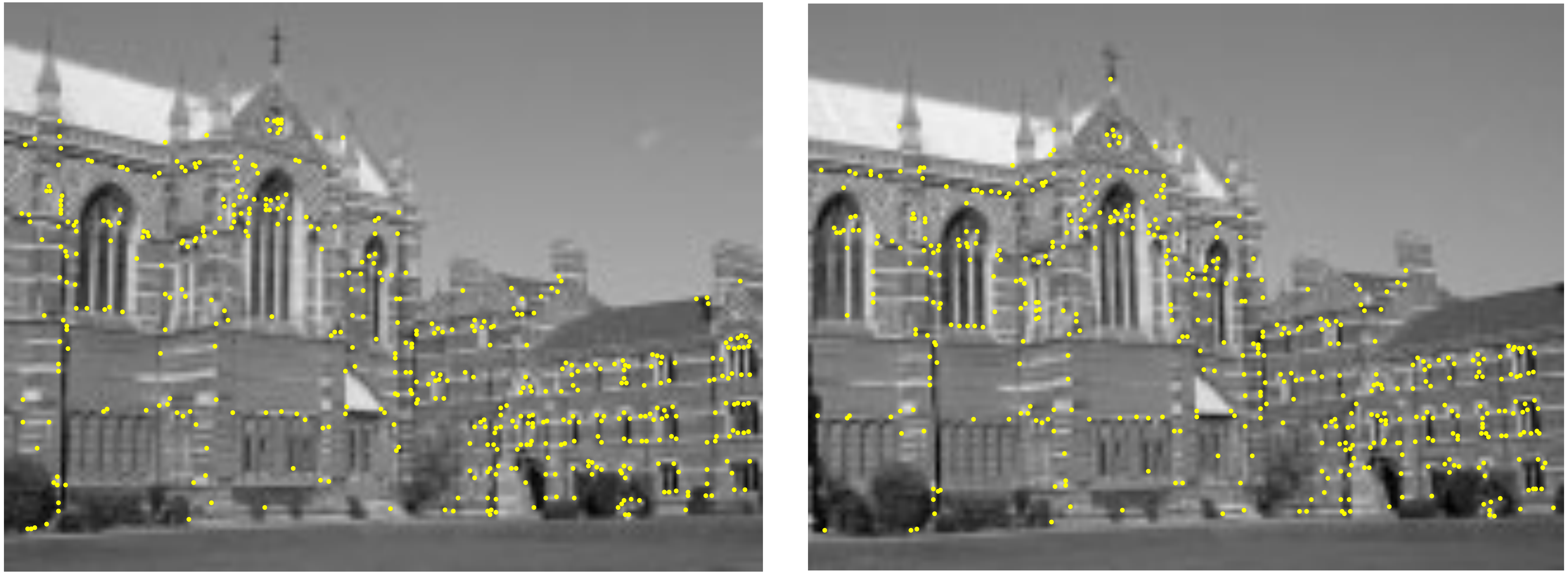


≈ 500 corner features found in each image

Figure Credit: Hartley & Zisserman

Example 3: Finding Feature Matches

Select best match over threshold within a square search window (here ±320 pixels) using SSD or (normalized) cross-correlation for small patch around the corner



≈ 500 corner features found in each image

Figure Credit: Hartley & Zisserman

Example 3: Initial Match Hypothesis

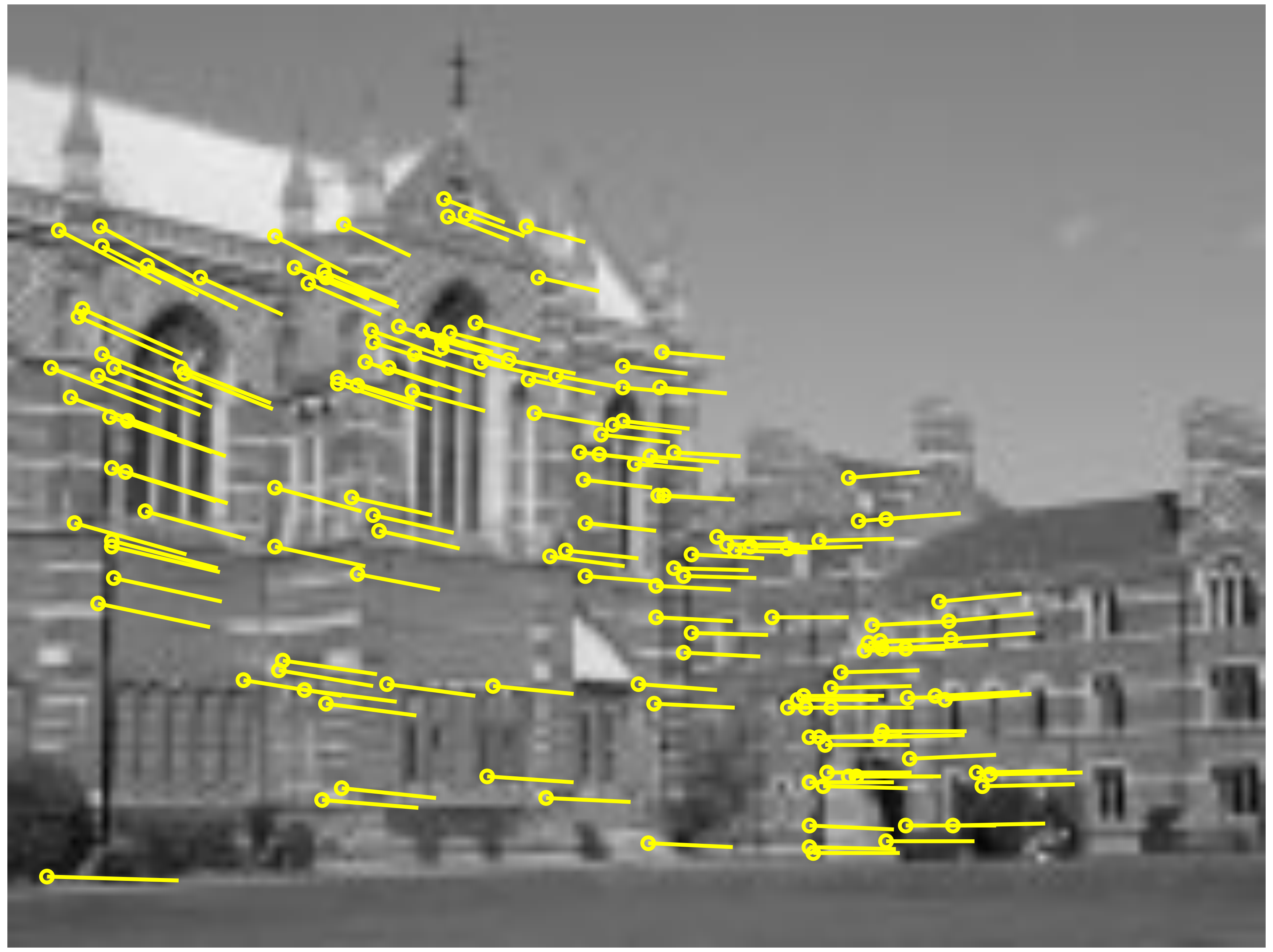
268 matched features (over SSD threshold) superimposed on left image (pointing to locations of corresponding feature in right image)

Figure Credit: Hartley & Zisserman

Example 3: Outliers & Inliers after RANSAC

- *n* is 4 for this problem (a homography relating 2 images)
- Assume up to 50% outliers
- 43 samples used with t = 1.25 pixels

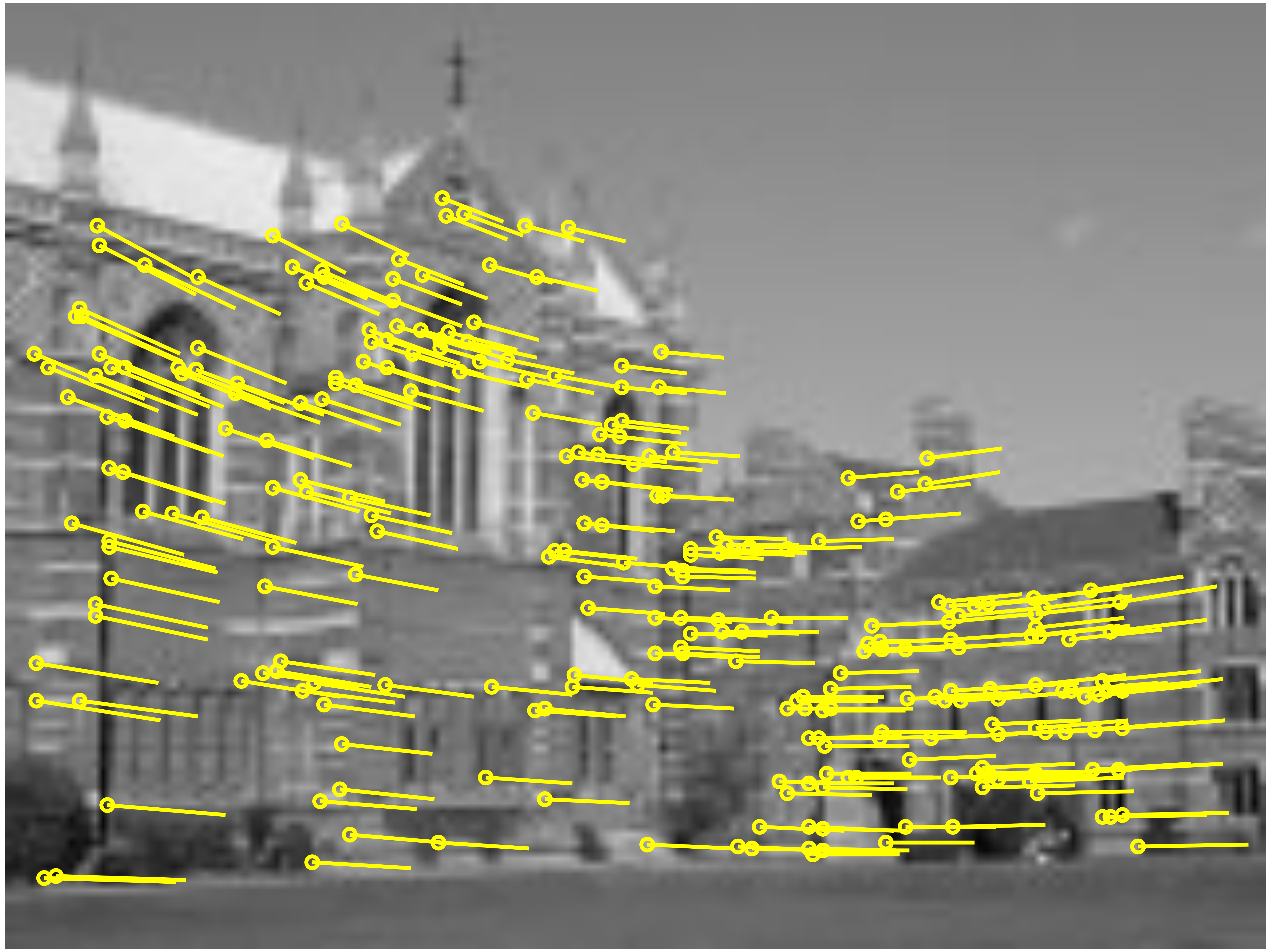
117 **outliers**



151 **inliers**

Figure Credit: Hartley & Zisserman

Example 3: Final Matches



final set of 262 matches

Figure Credit: Hartley & Zisserman

Discussion of RANSAC

Advantages:

— General method suited for a wide range of model fitting problems

— Easy to implement and easy to calculate its failure rate

Disadvantages:

— Only handles a moderate percentage of outliers without cost blowing up

— Many real problems have high rate of outliers (but sometimes selective choice of random subsets can help)

The Hough transform can handle high percentage of outliers

Example: Photo Tourism

Figure credit: Snavely et al. 2006

Takes as input unstructured collections of photographs and reconstructs each photo's viewpoint and a sparse 3D model of the scene

Uses both SIFT and RANSAC

Fitting a Model

Suppose we want to fit a model to a set of tokens

— e.g. A line fits well to a set of points. This is unlikely to be due to chance, so we represent the points as a line.

— e.g. A 3D model can be scaled, rotated and translated to closely fit a set of points or line segments. If it fits well, the object is recognized.

Fitting a Model is Difficult

Difficulties arise owing to:

Extraneous data: clutter or multiple models

— We do not know what is part of the model

— Can we fit models with a few parts when there is significant background clutter?

Missing data: only some parts of model are present Noise

Computational cost:

— Not feasible to check all combinations of features by fitting a model to each possible subset

Hough Transform

Idea of **Hough transform**:

— For each token vote for all models to which the token could belong

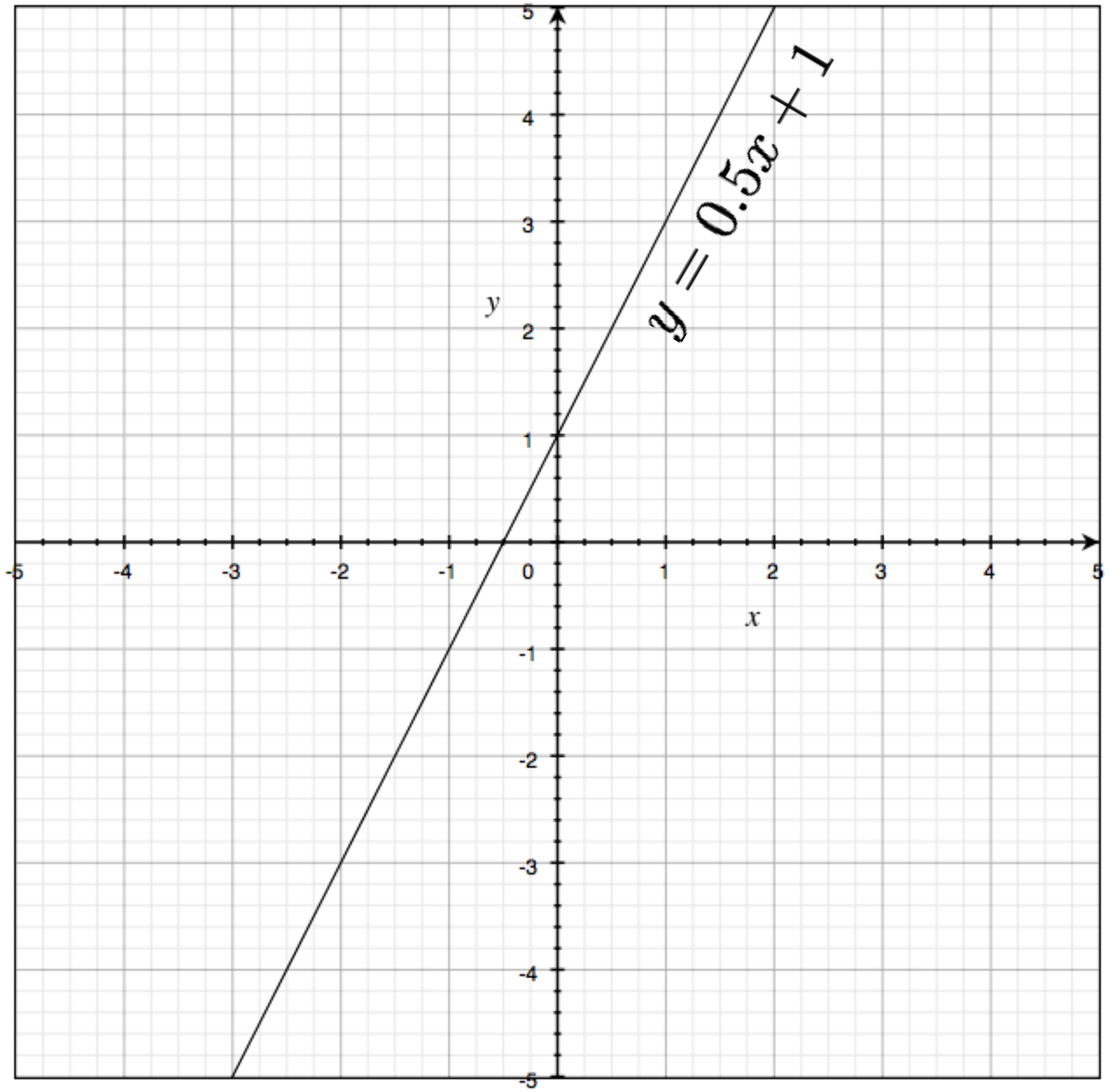
— Return models that get many votes

Example: For each point, vote for all lines that could pass through it; the true lines will pass through many points and so receive many votes

Lines: Slope intercept form

$$y = mx + b$$

slope y-intercept



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Image and Parameter Space

variables

$$y = mx + b$$

parameters

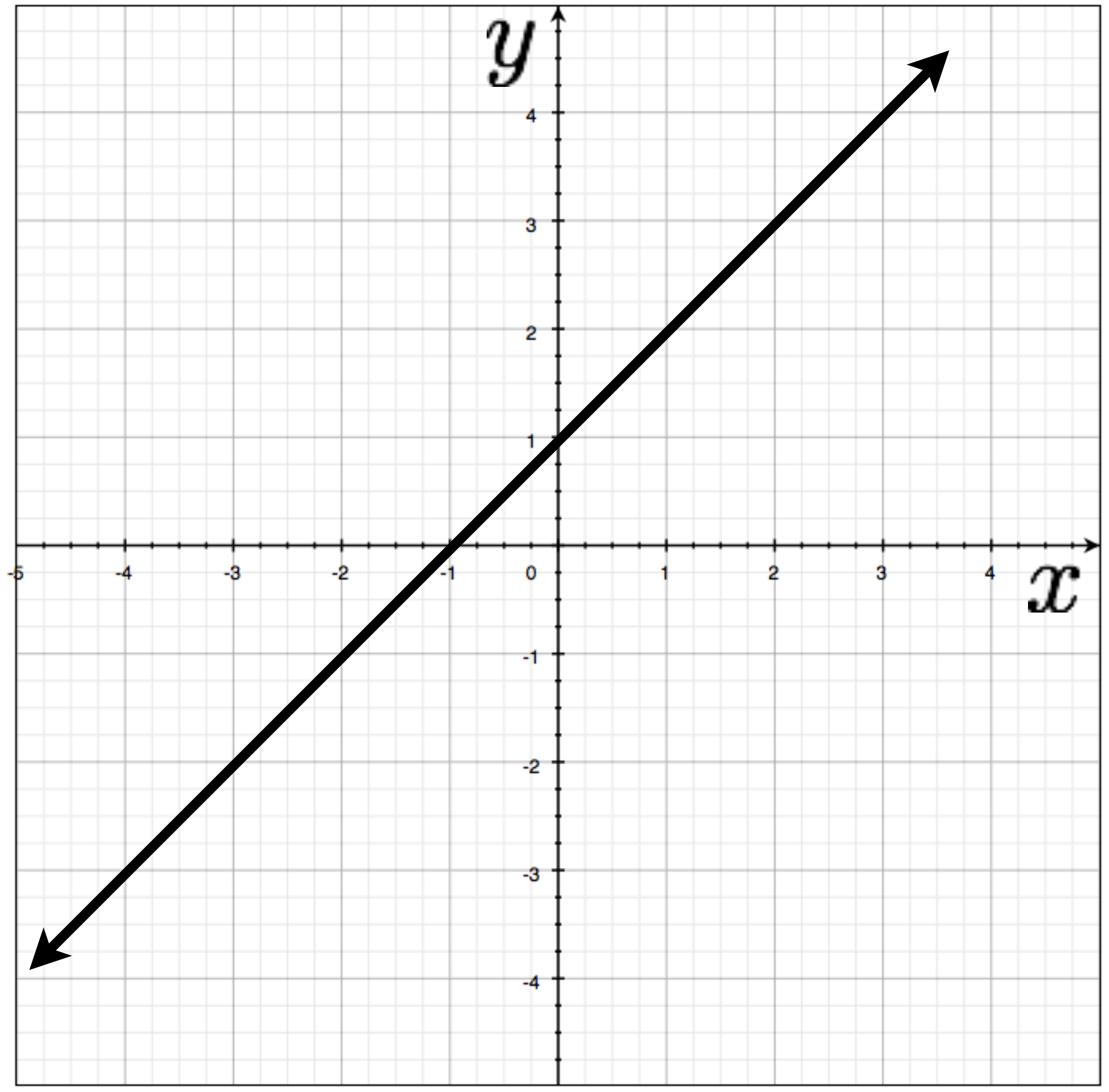


Image space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Image and Parameter Space

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

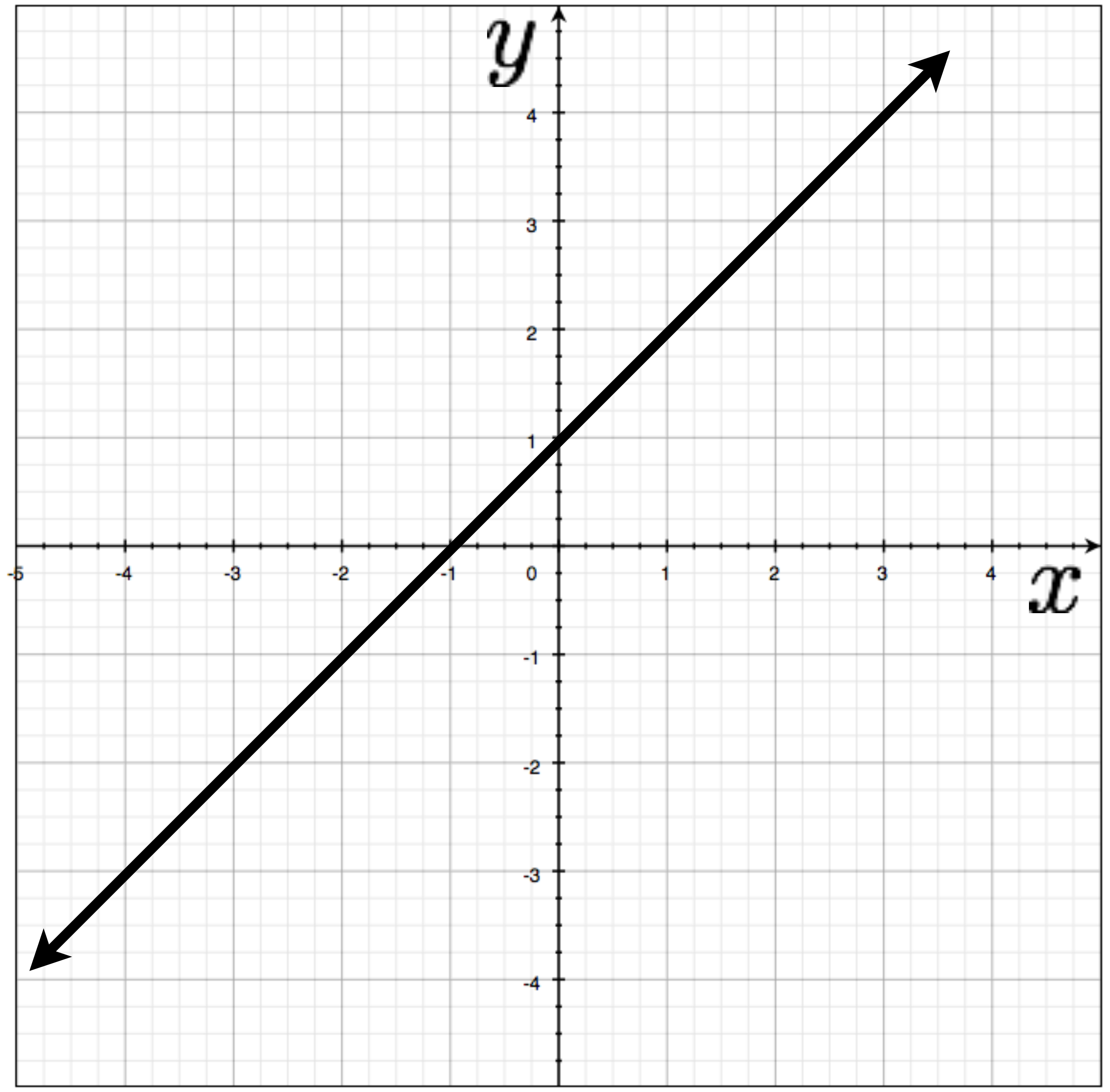
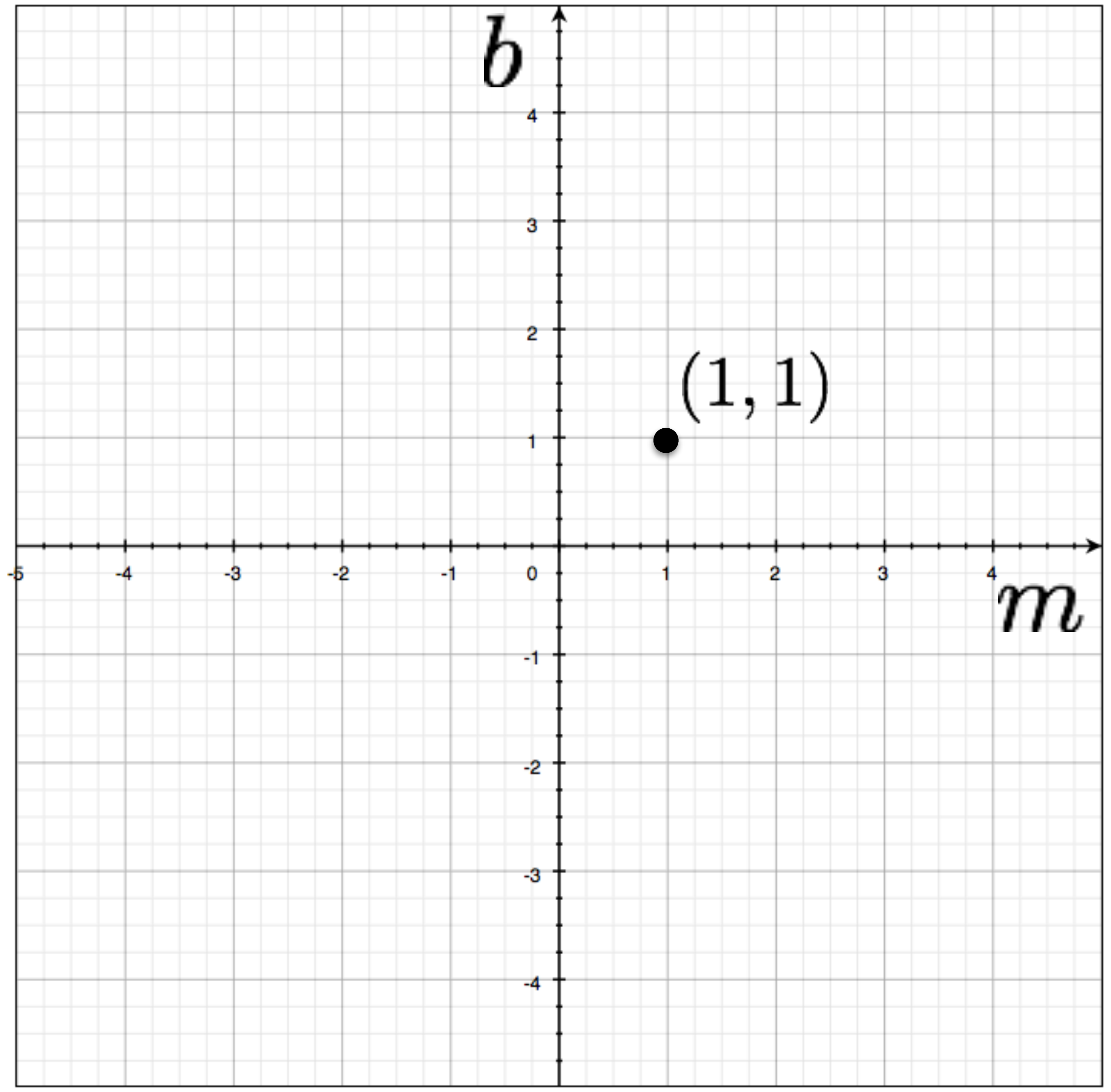


Image space

a line becomes a point



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Image and Parameter Space

variables

$$y = mx + b$$

parameters

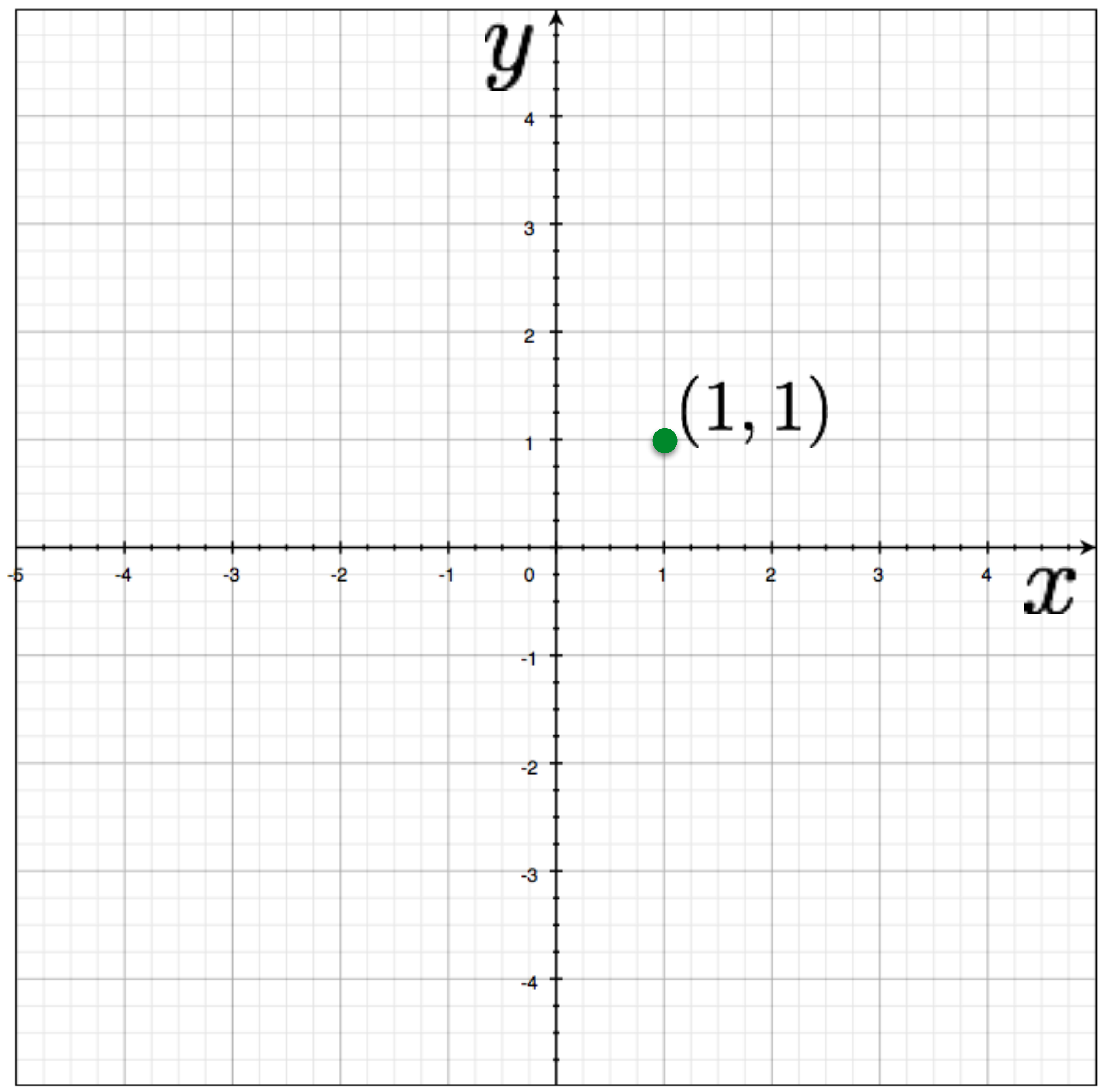


Image space

What would a **point** in image space become in **parameter space**?

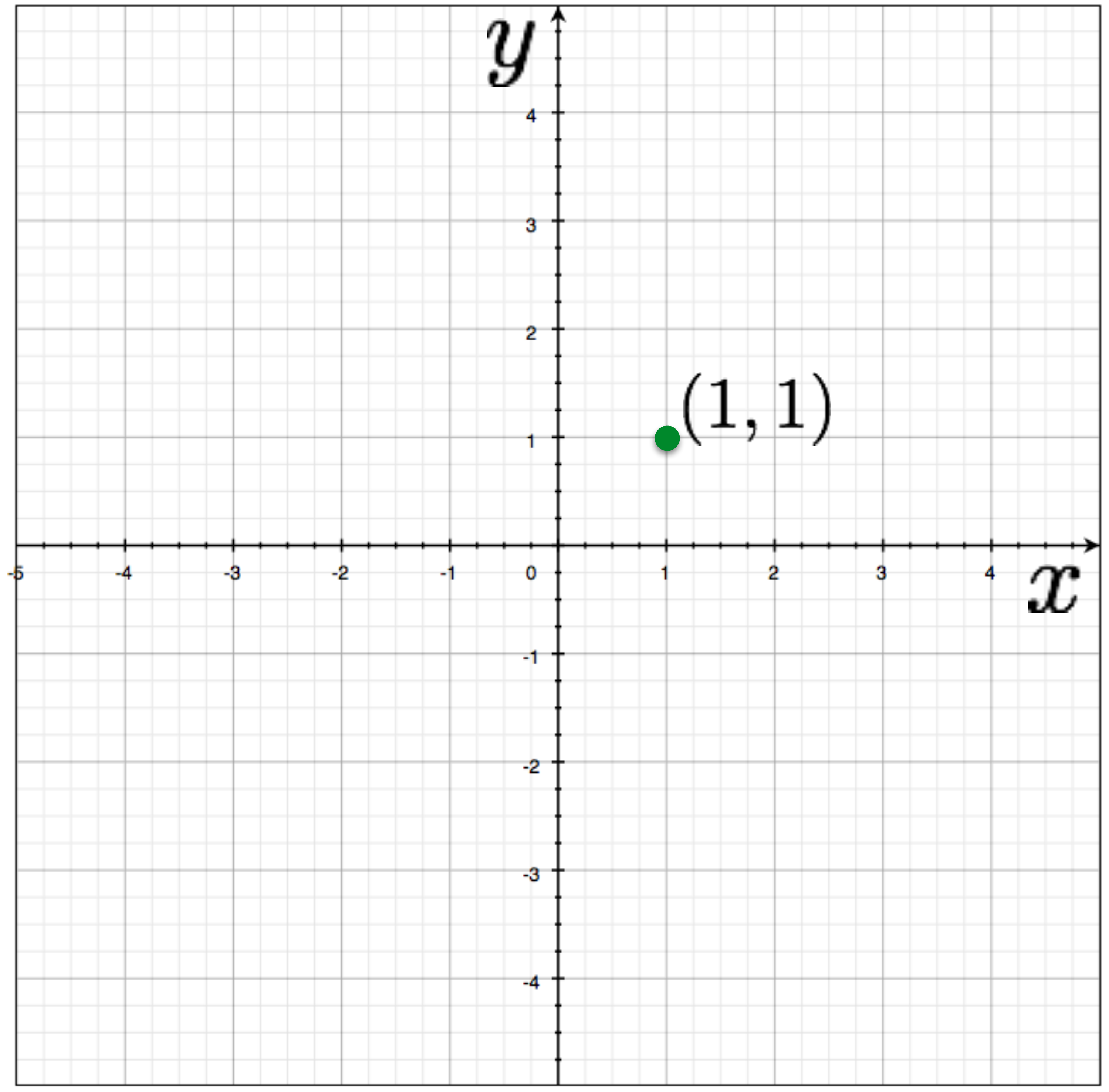
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters



a point becomes a line

variables

$$y - mx = b$$

parameters

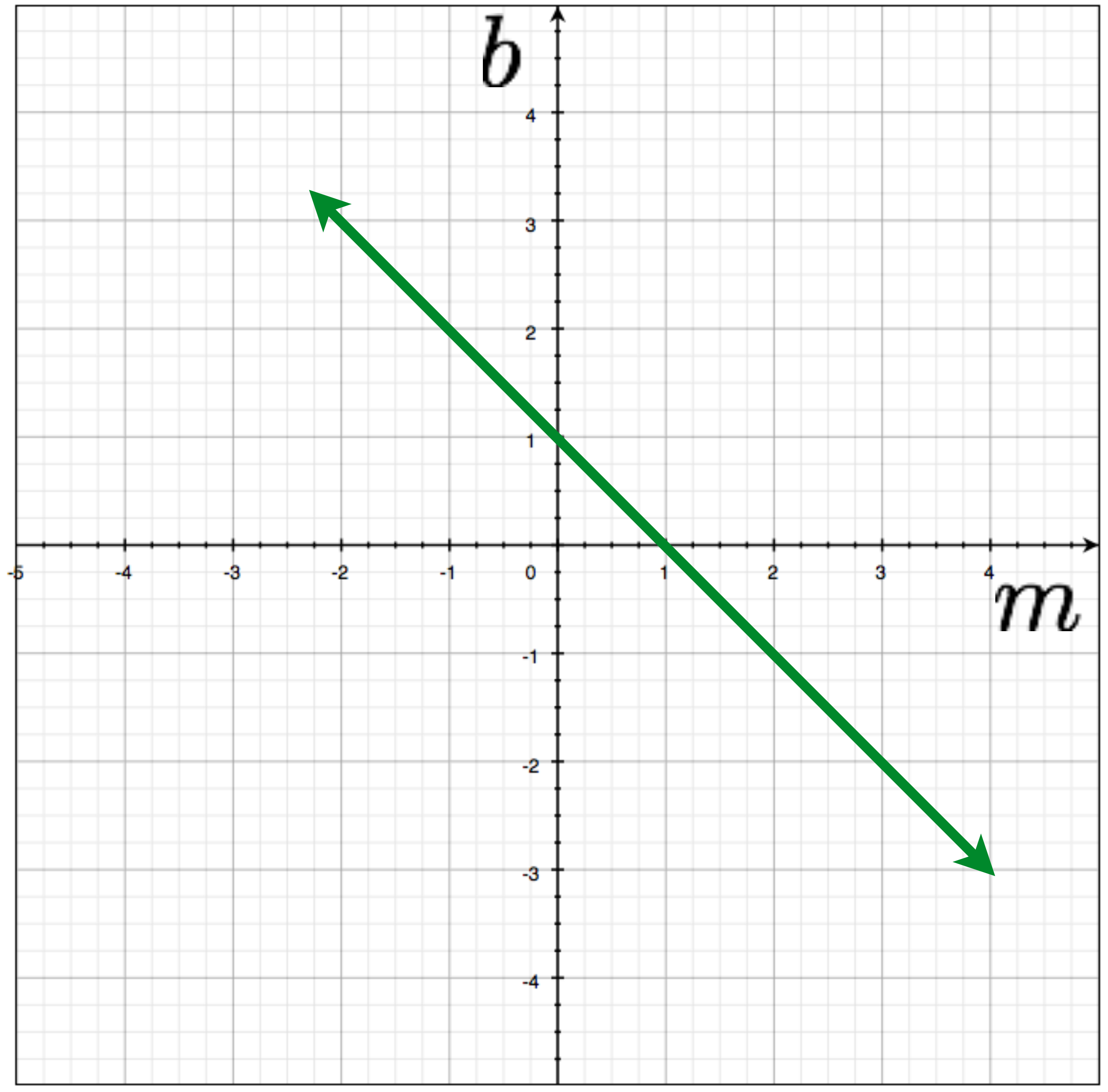


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

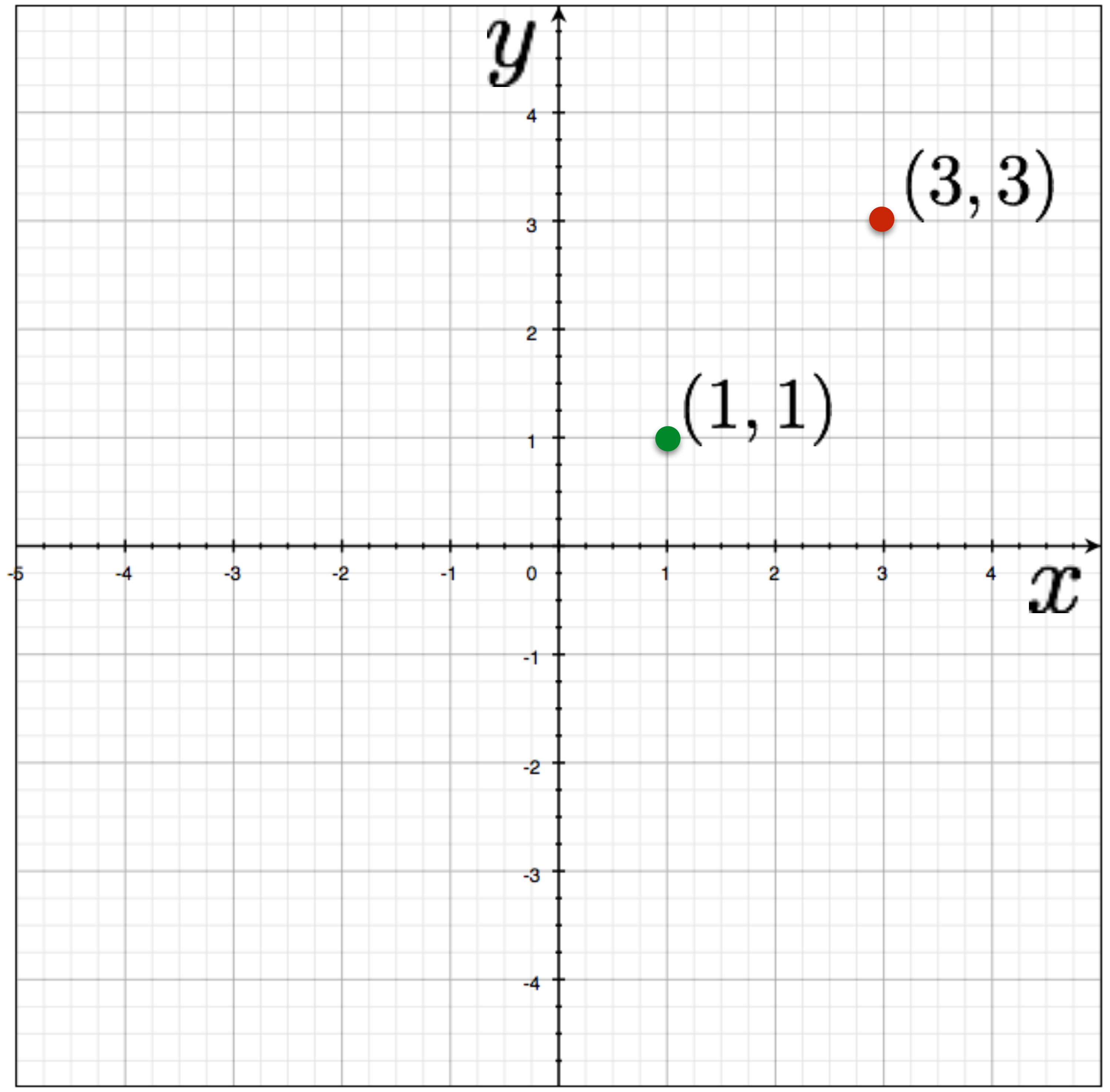
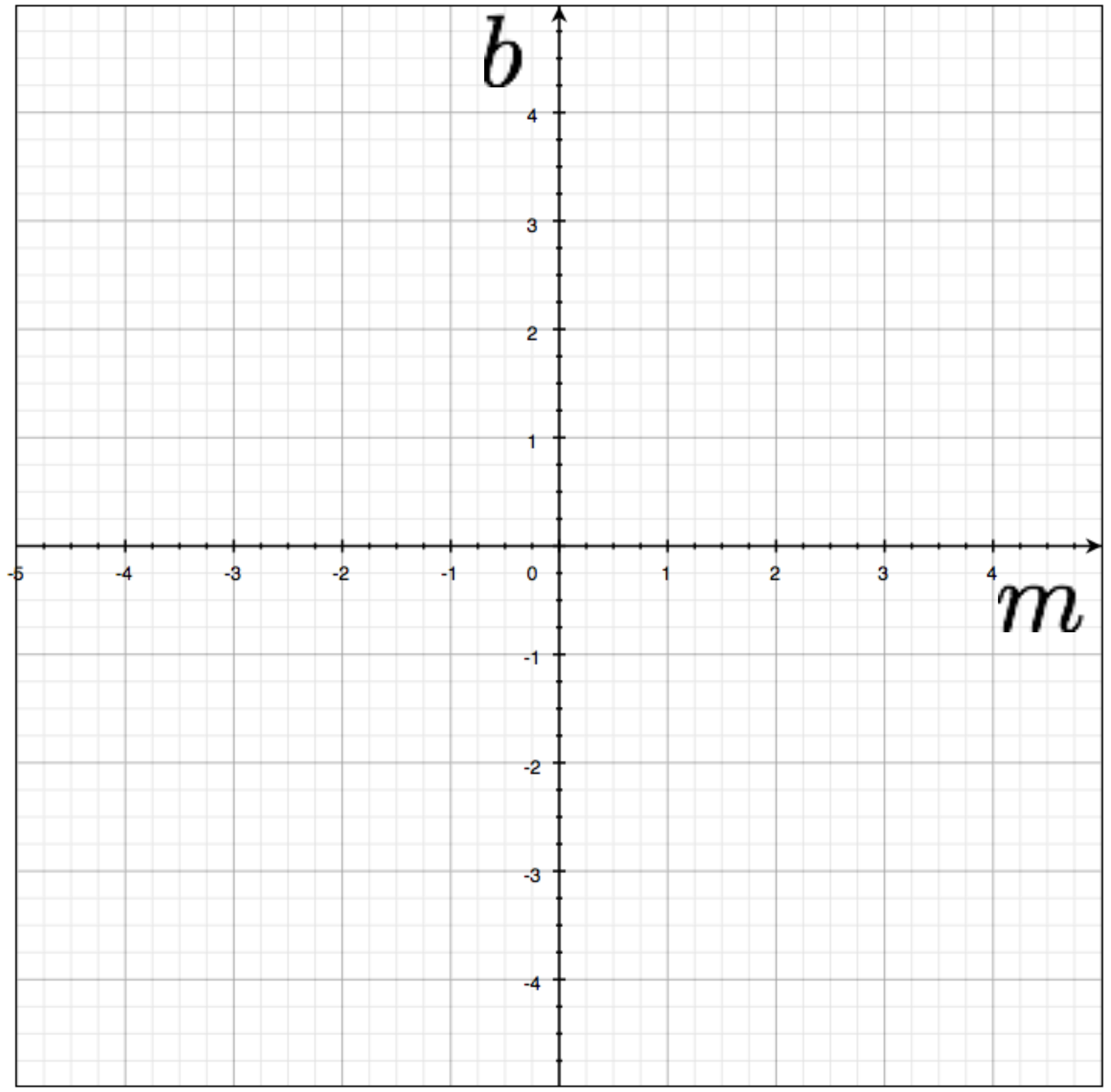


Image space

two points?



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

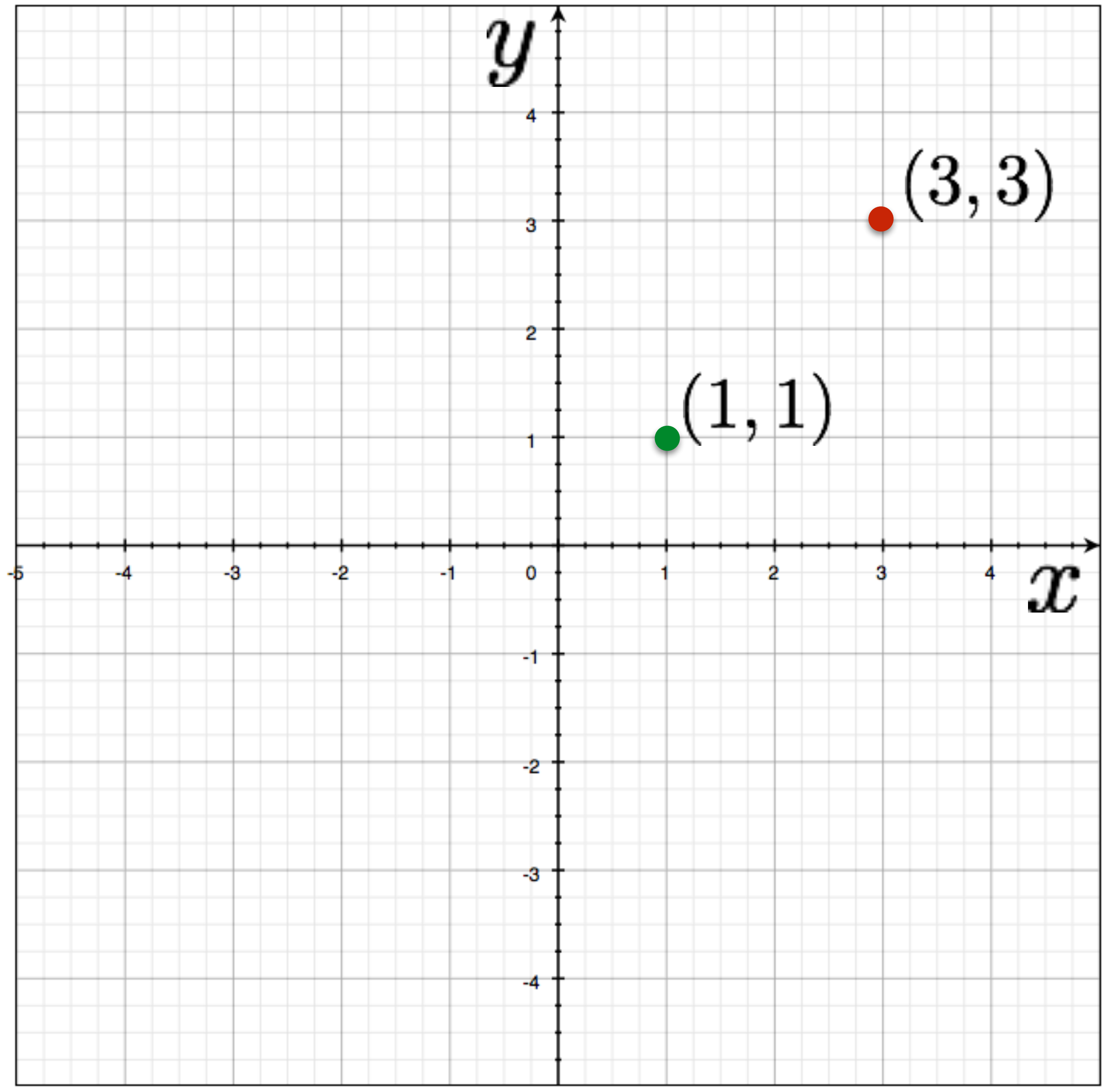


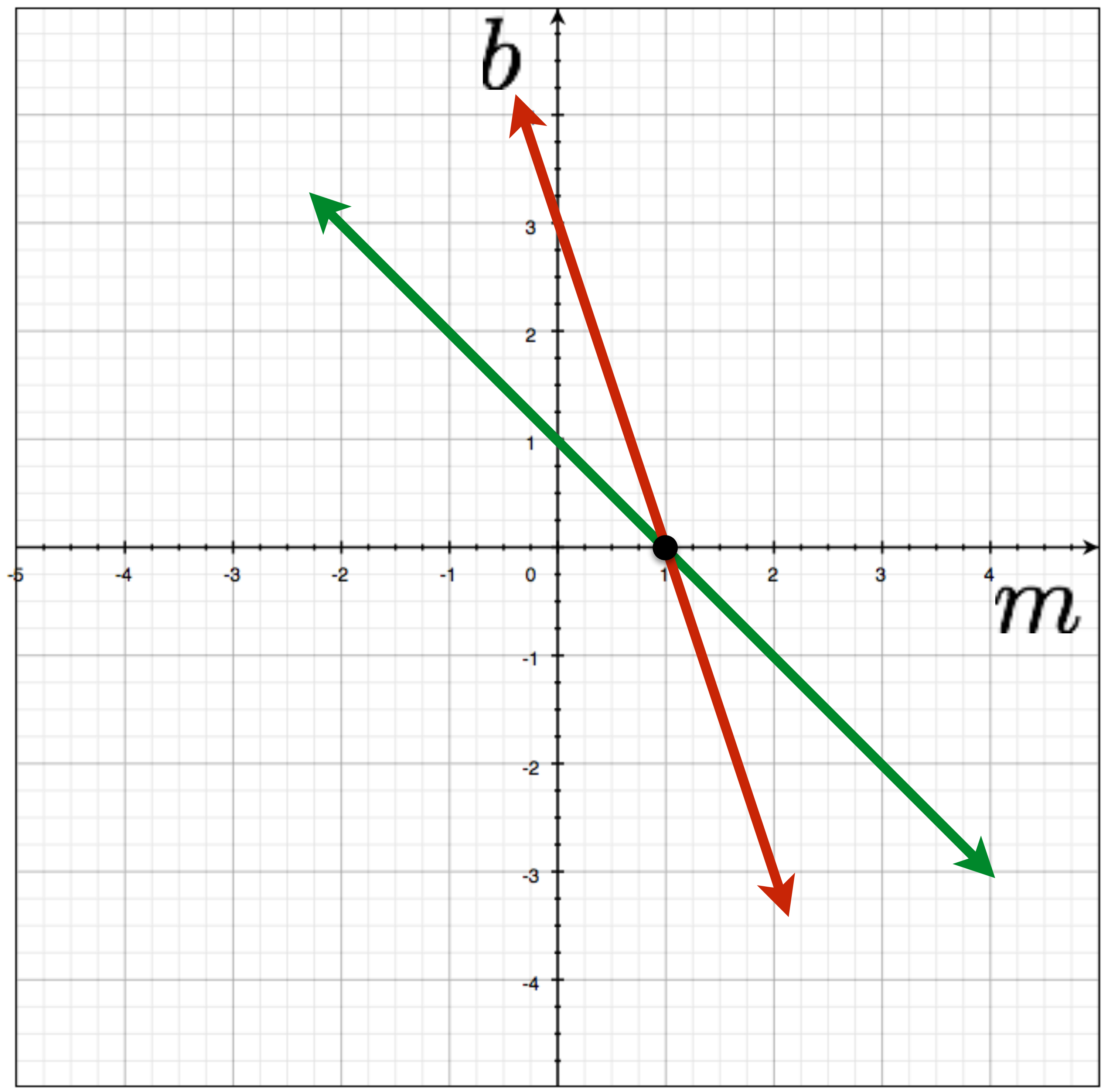
Image space

two points?

variables

$$y - mx = b$$

parameters



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

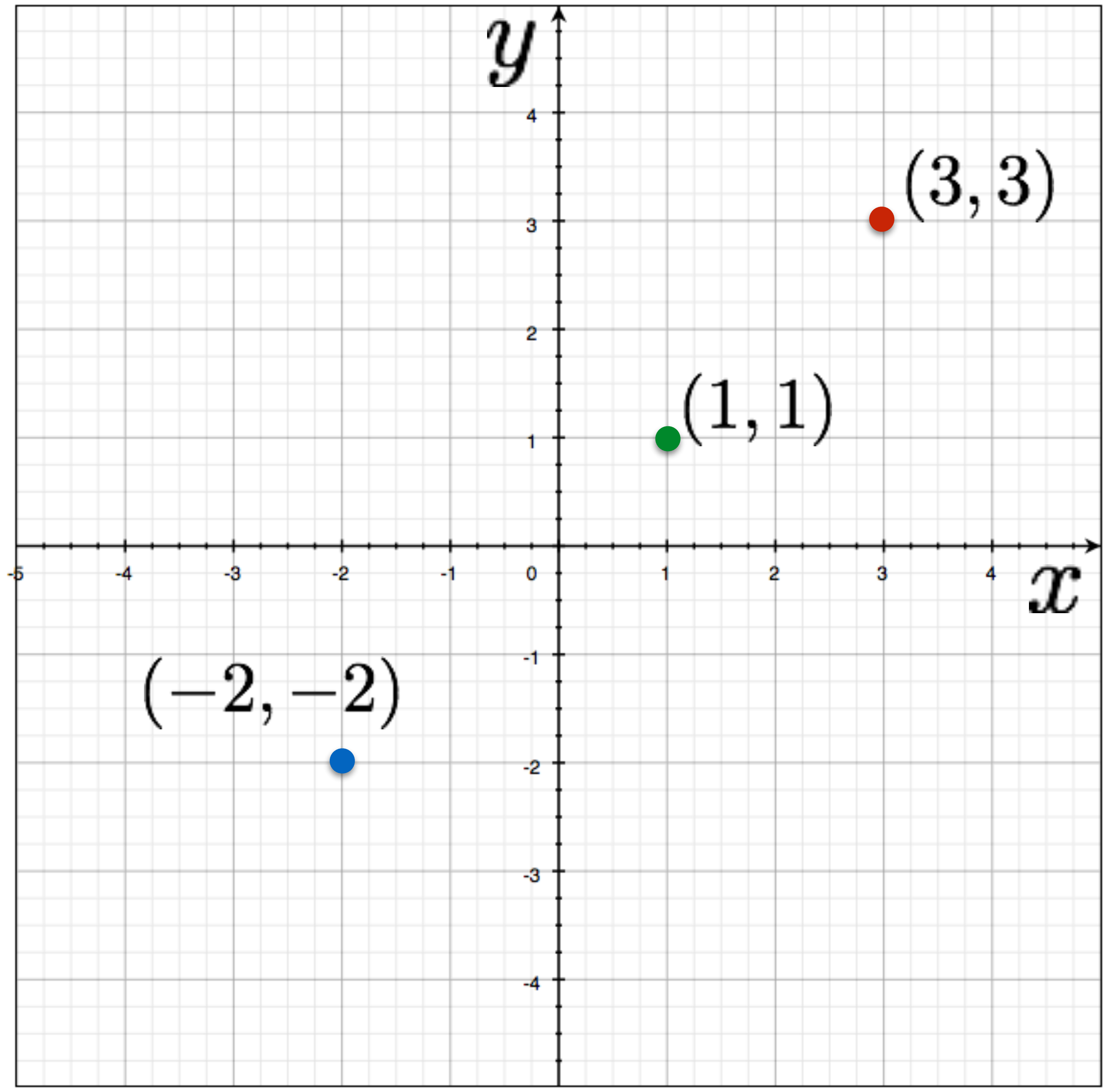


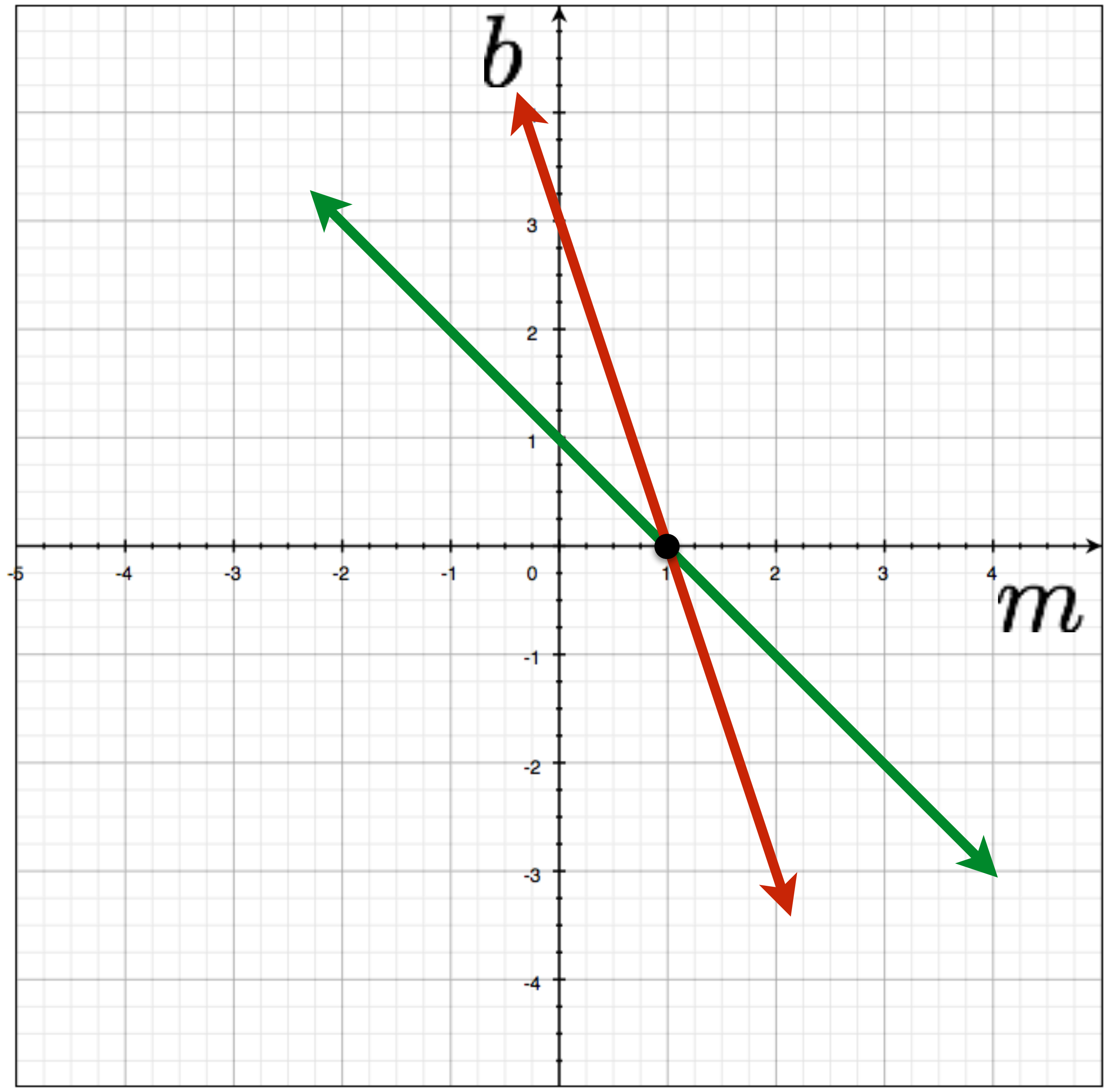
Image space

three points?

variables

$$y - mx = b$$

parameters



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

variables

$$y - mx = b$$

parameters

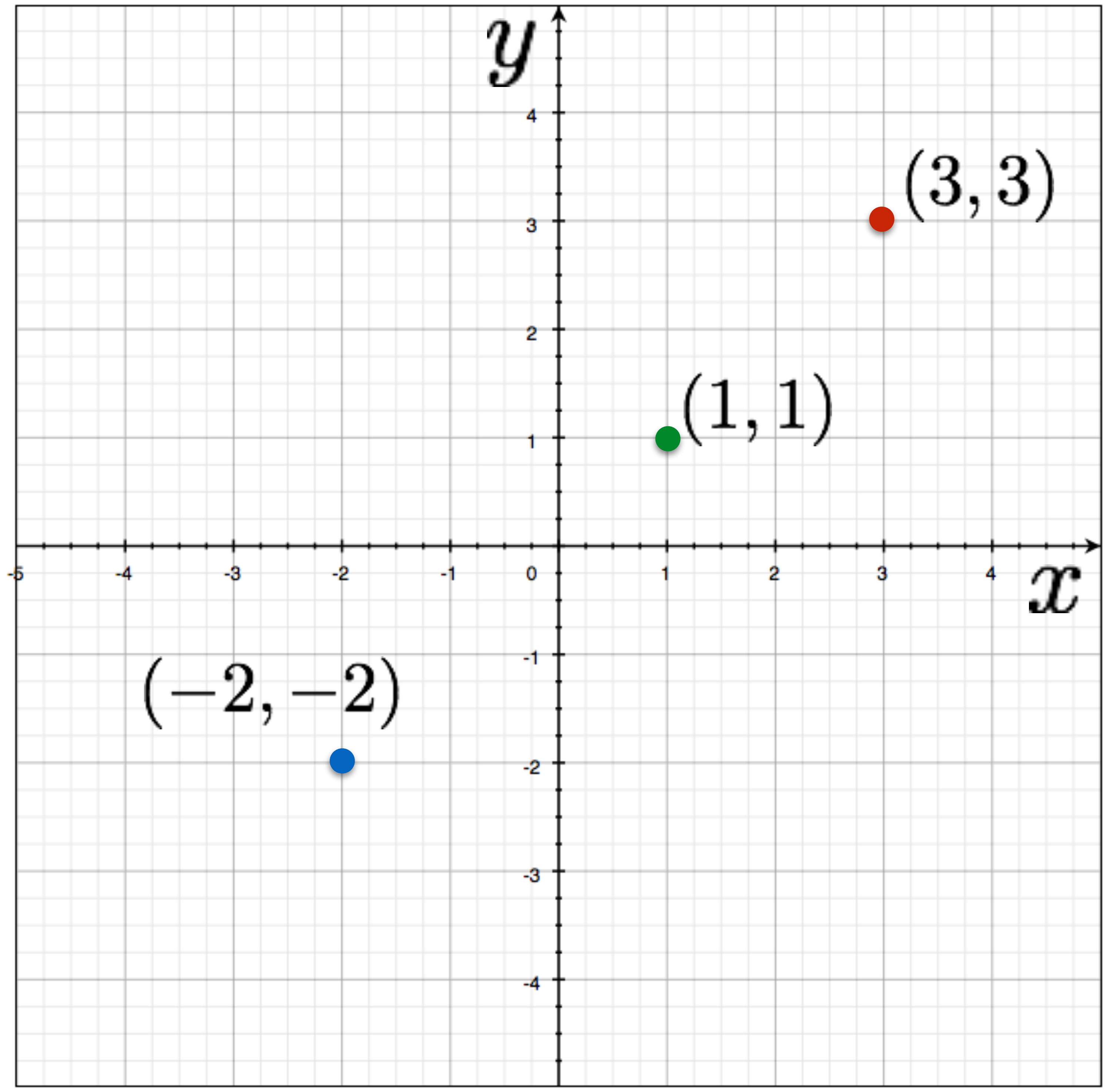
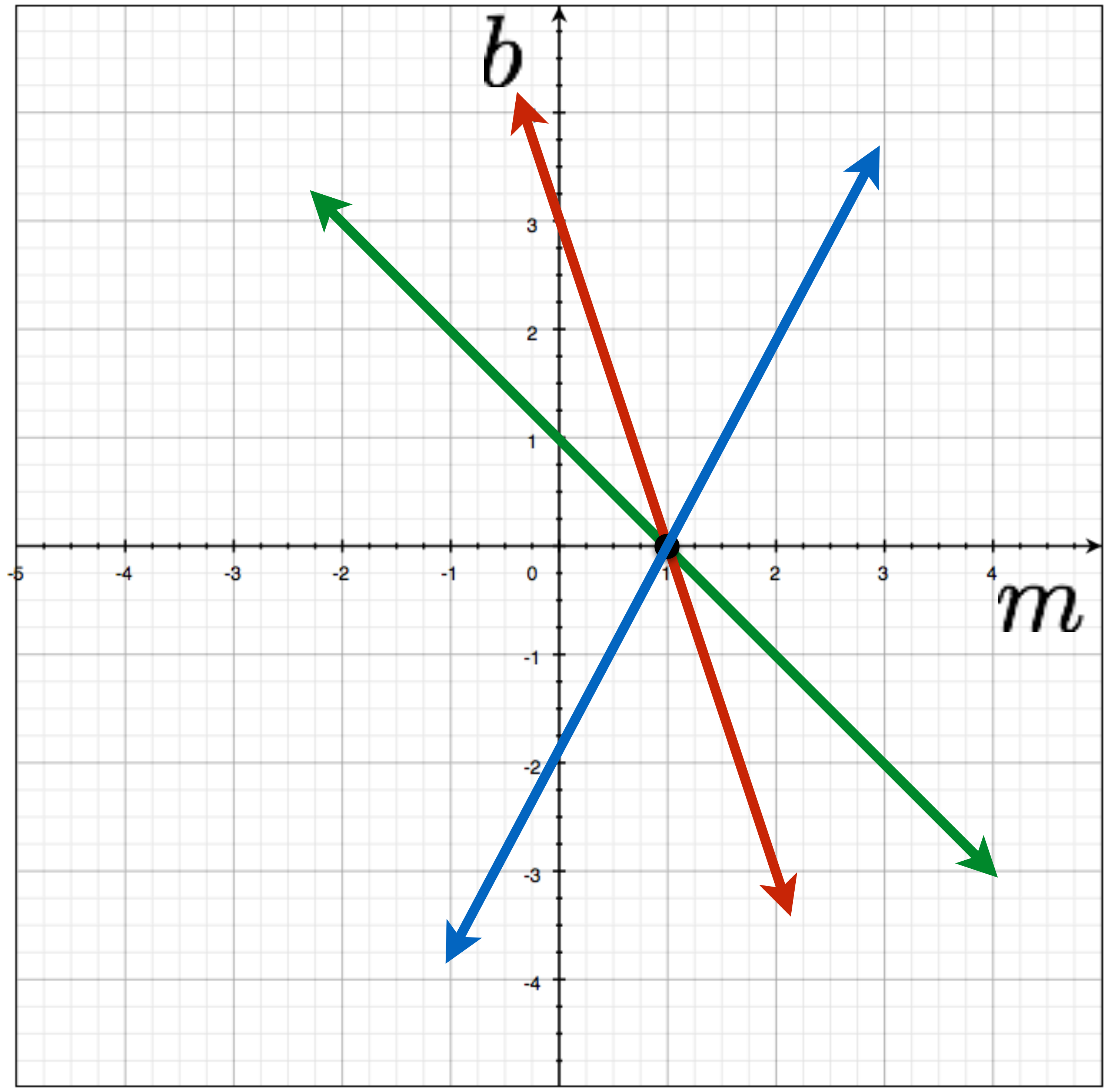


Image space



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

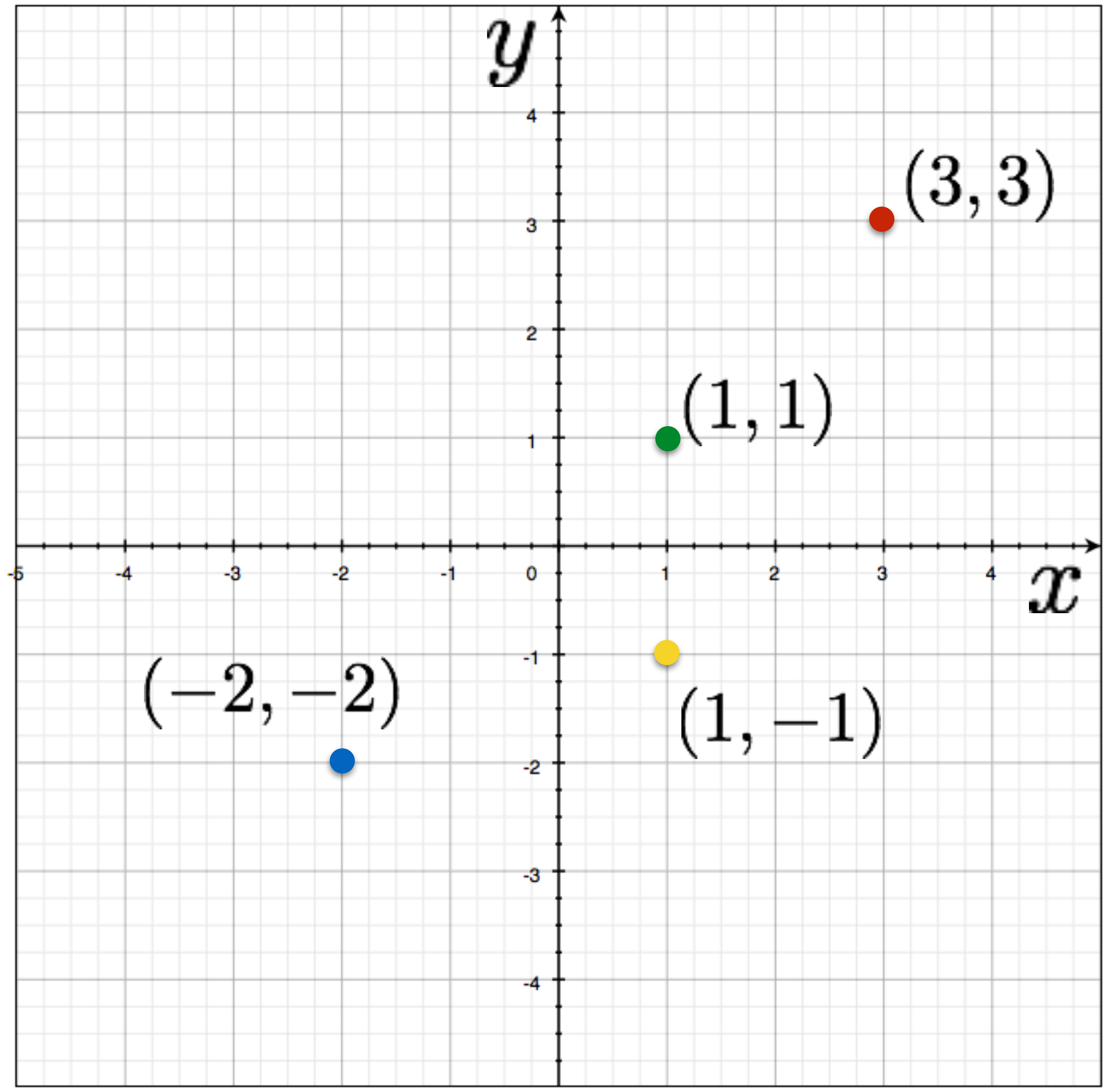


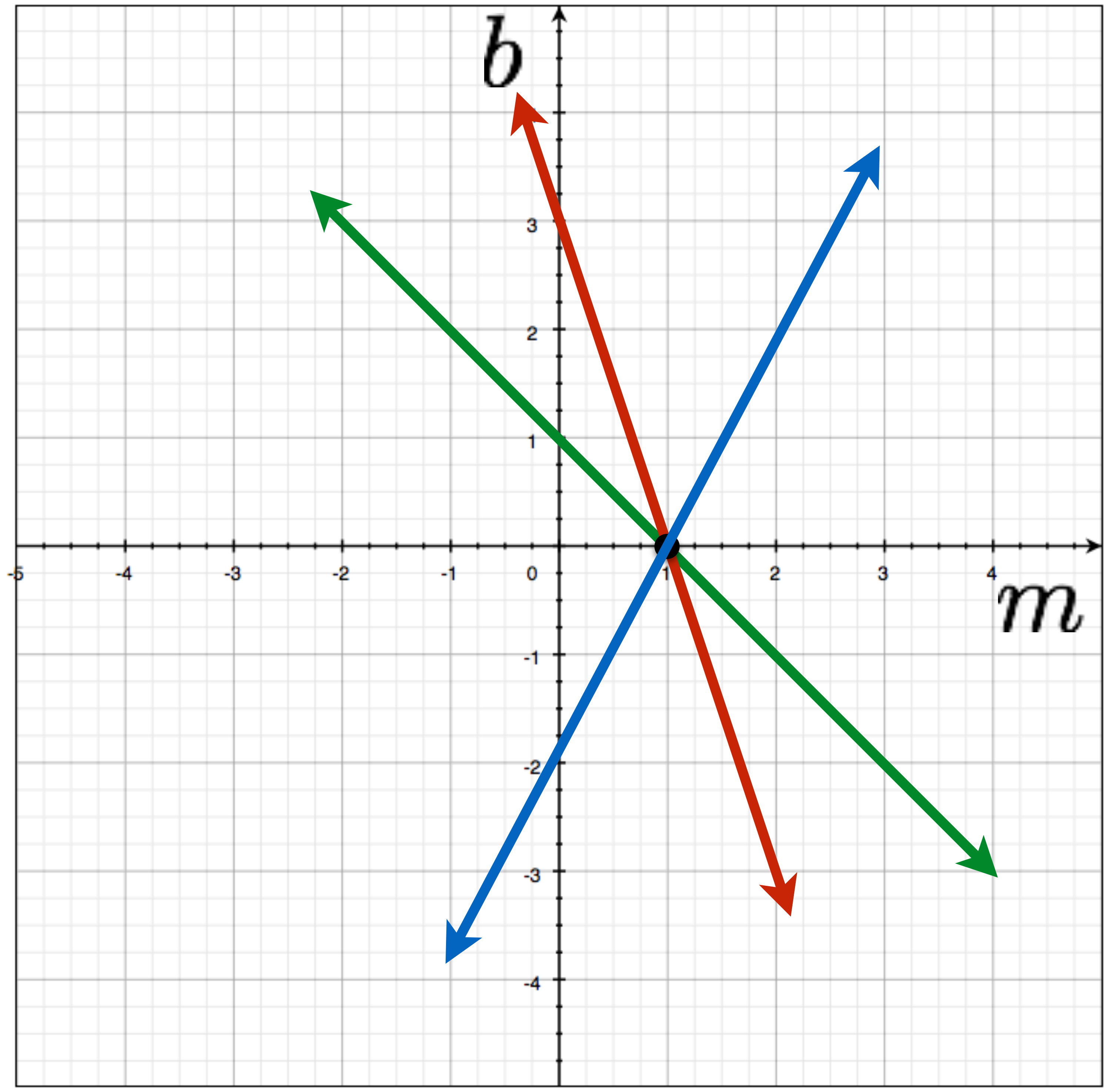
Image space

four points?

variables

$$y - mx = b$$

parameters



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

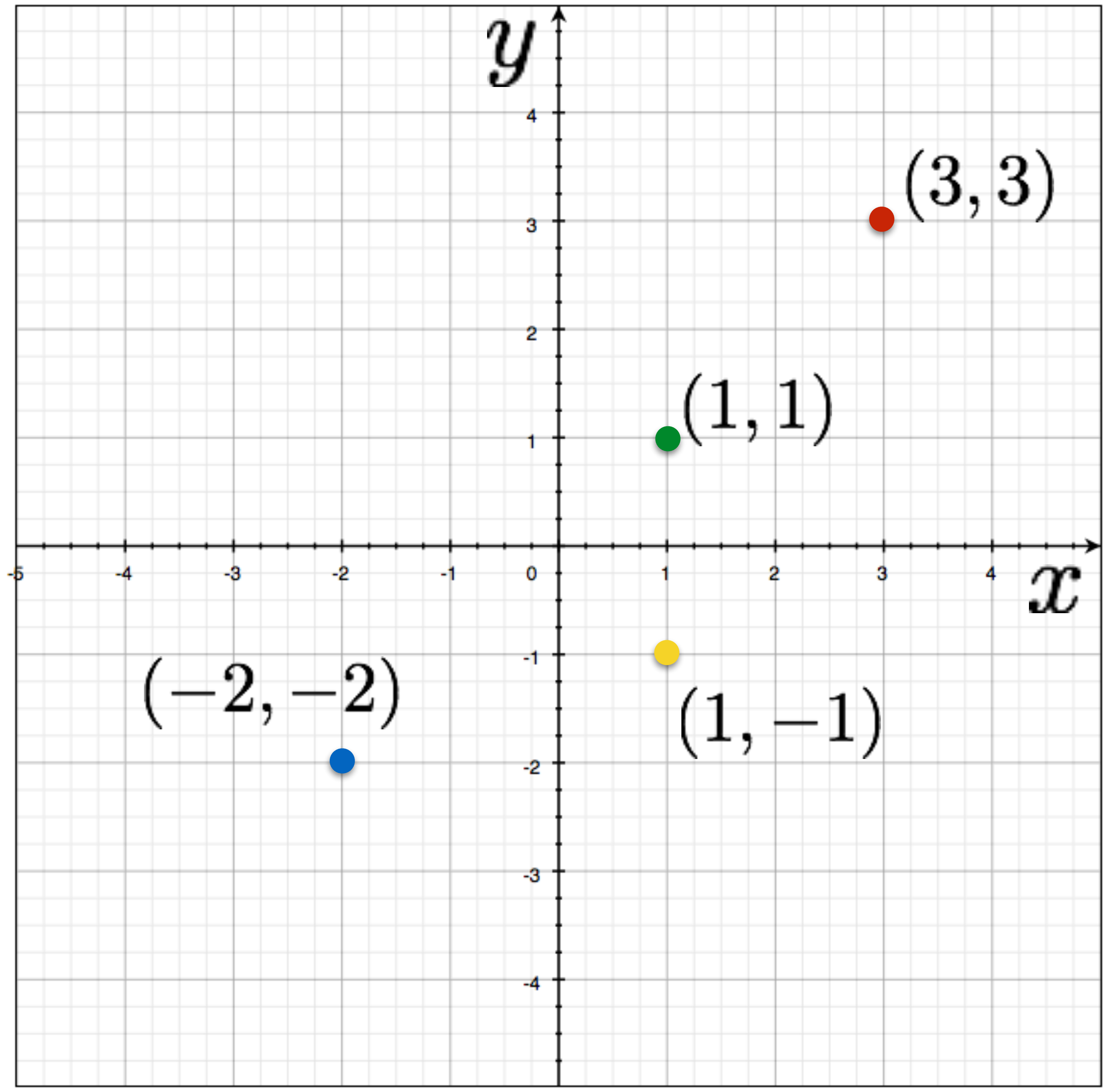


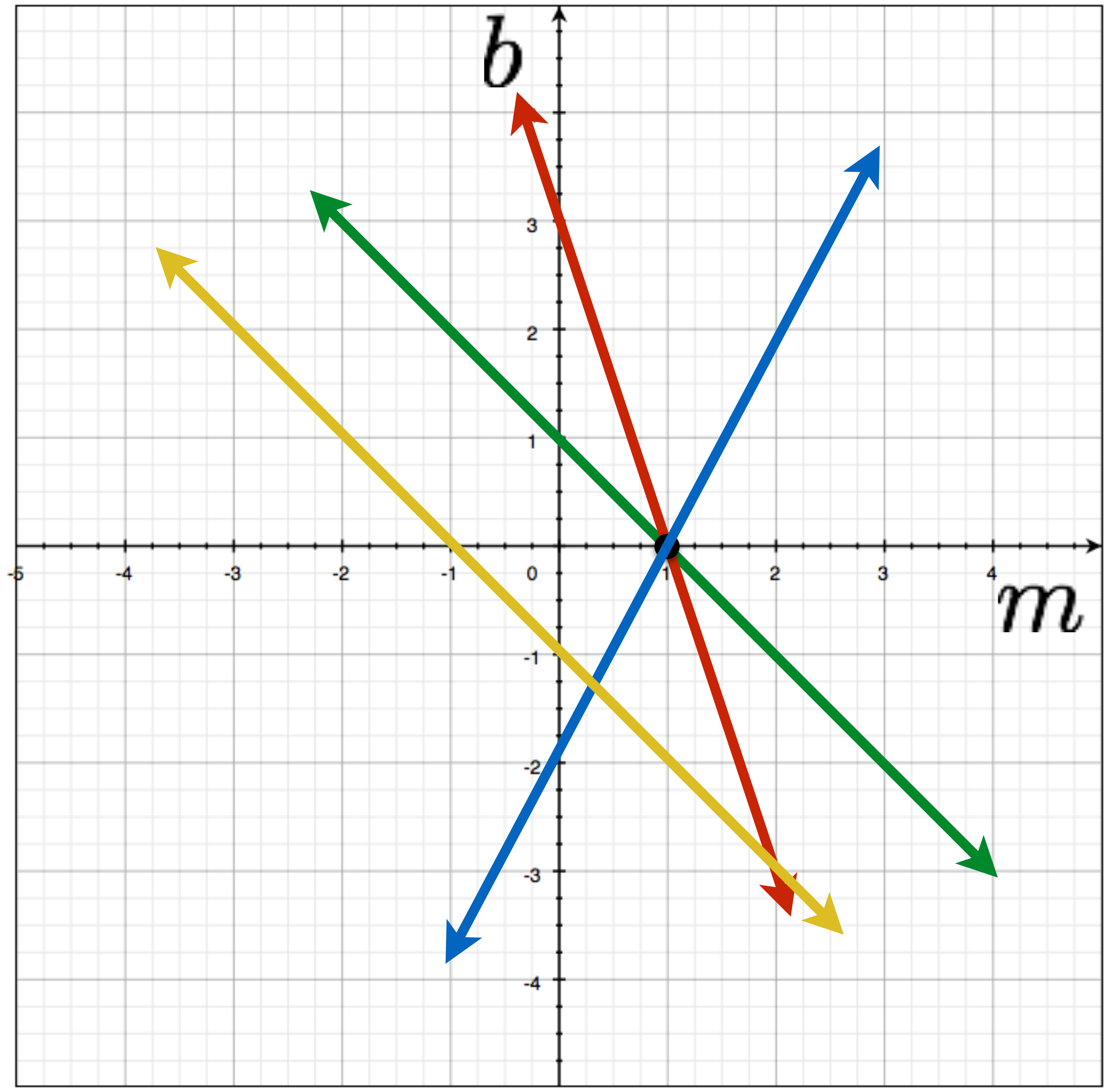
Image space

four points?

variables

$$y - mx = b$$

parameters



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

How would you find the best fitting line?

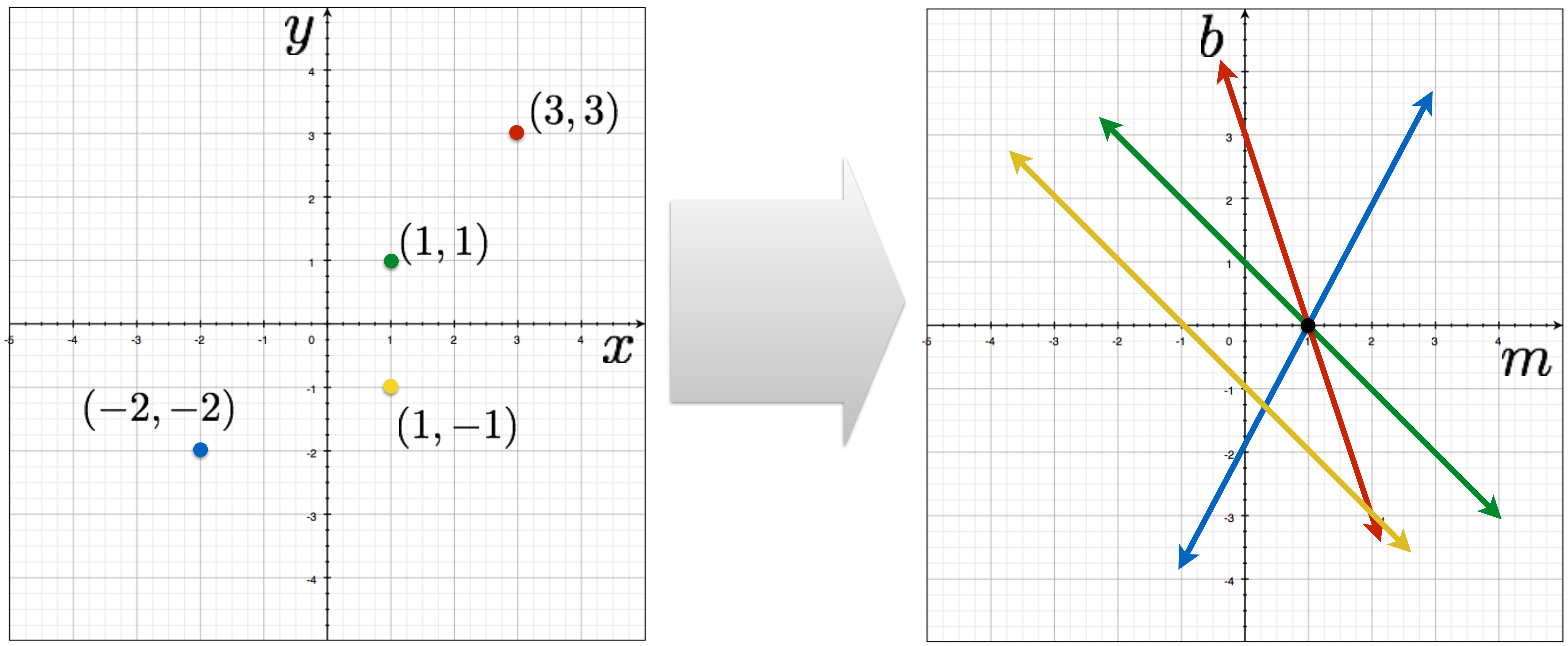


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

Is this method robust to measurement noise? clutter?

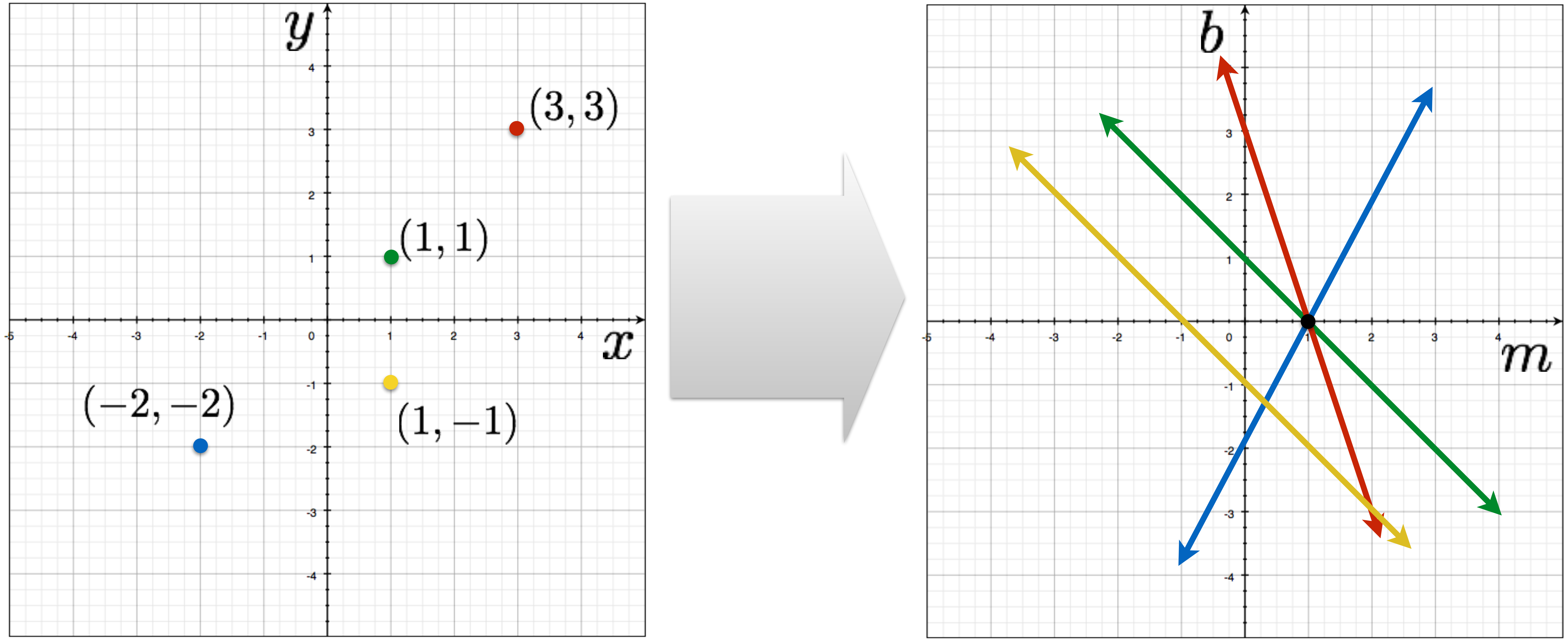


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Line Detection by Hough Transform

Algorithm:

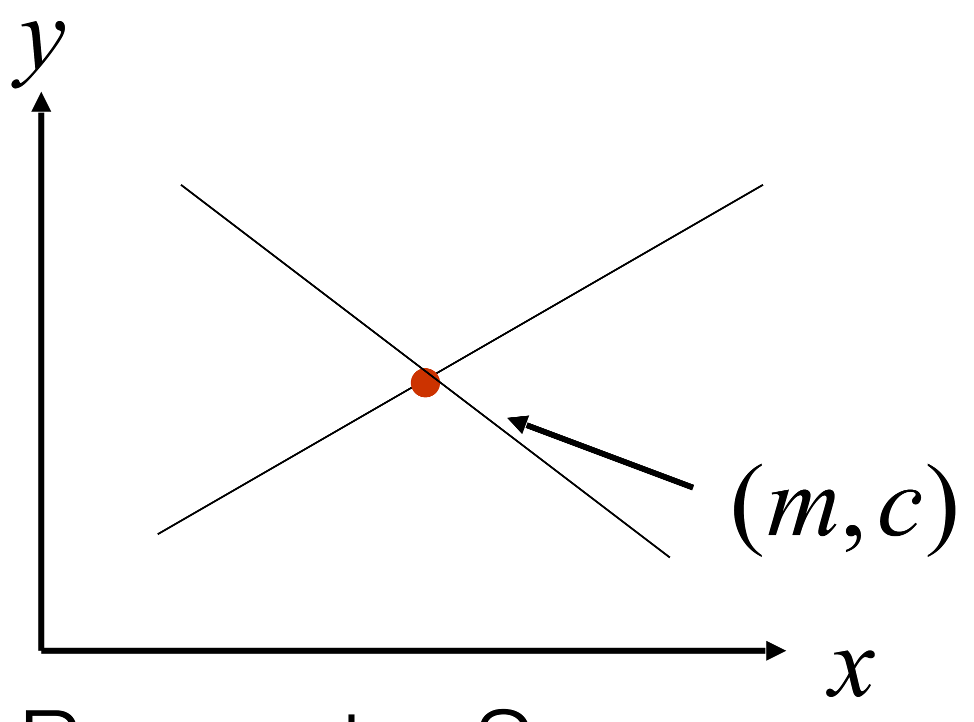
1. Quantize Parameter Space (m,c)

2. Create Accumulator Array $A(m,c)$

3. Set $A(m,c)=0 \quad \forall m,c$

4. For each image edge (x_i, y_i)
   For each element in $A(m,c)$
   If (m,c) lies on the line: $c=-x_i m+y_i$
   Increment $A(m,c)=A(m,c)+1$

5. 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		

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Problems with **Parametrization**

How big does the accumulator need to be for the parameterization (m, c)?

$A(m, c)$

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

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Problems with **Parametrization**

How big does the accumulator need to be for the parameterization (m, c)?

$A(m, c)$

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

The space of m is huge!

$$-\infty \leq m \leq \infty$$

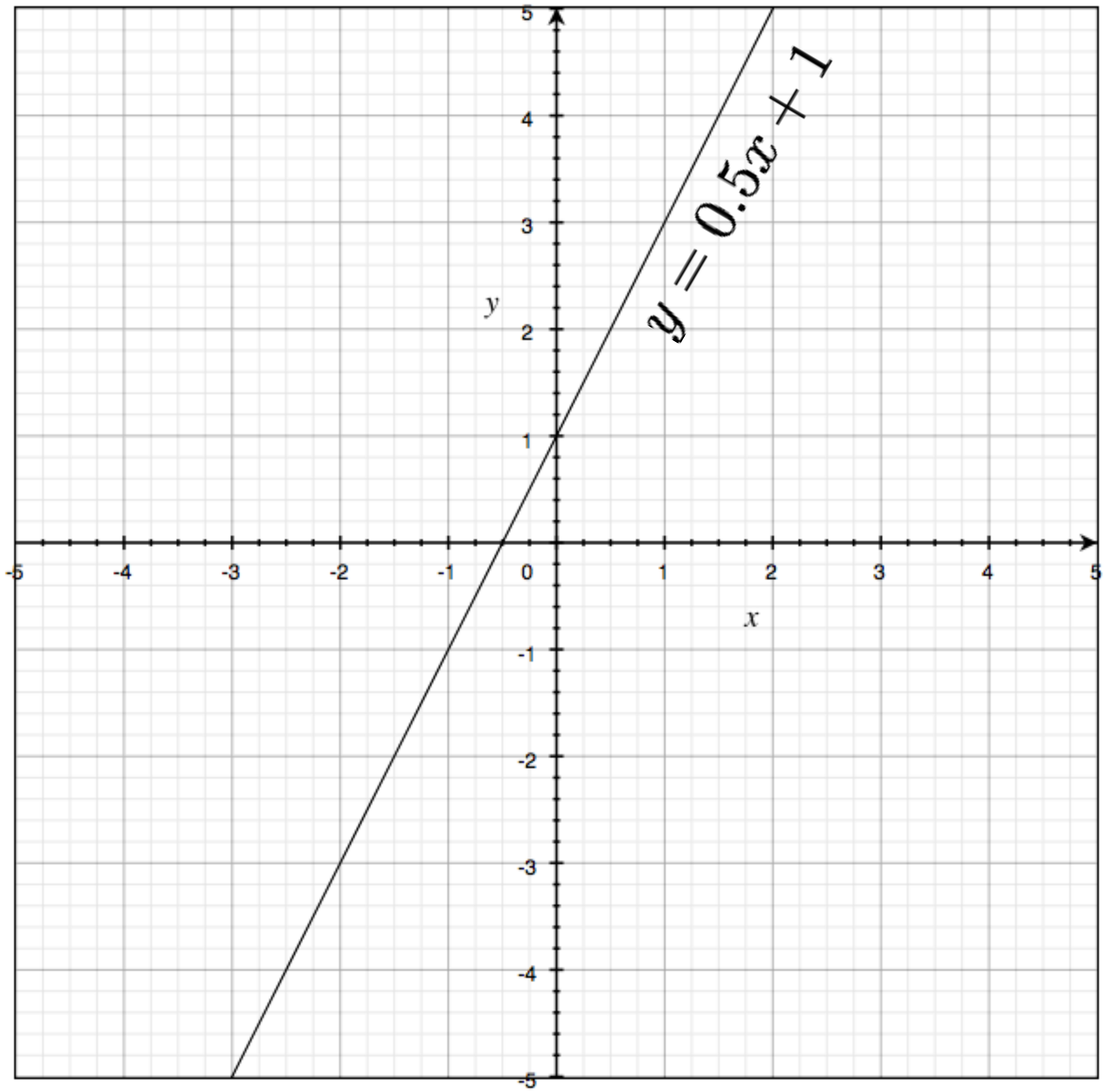
The space of c is huge!

$$-\infty \leq c \leq \infty$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lines: Slope intercept form

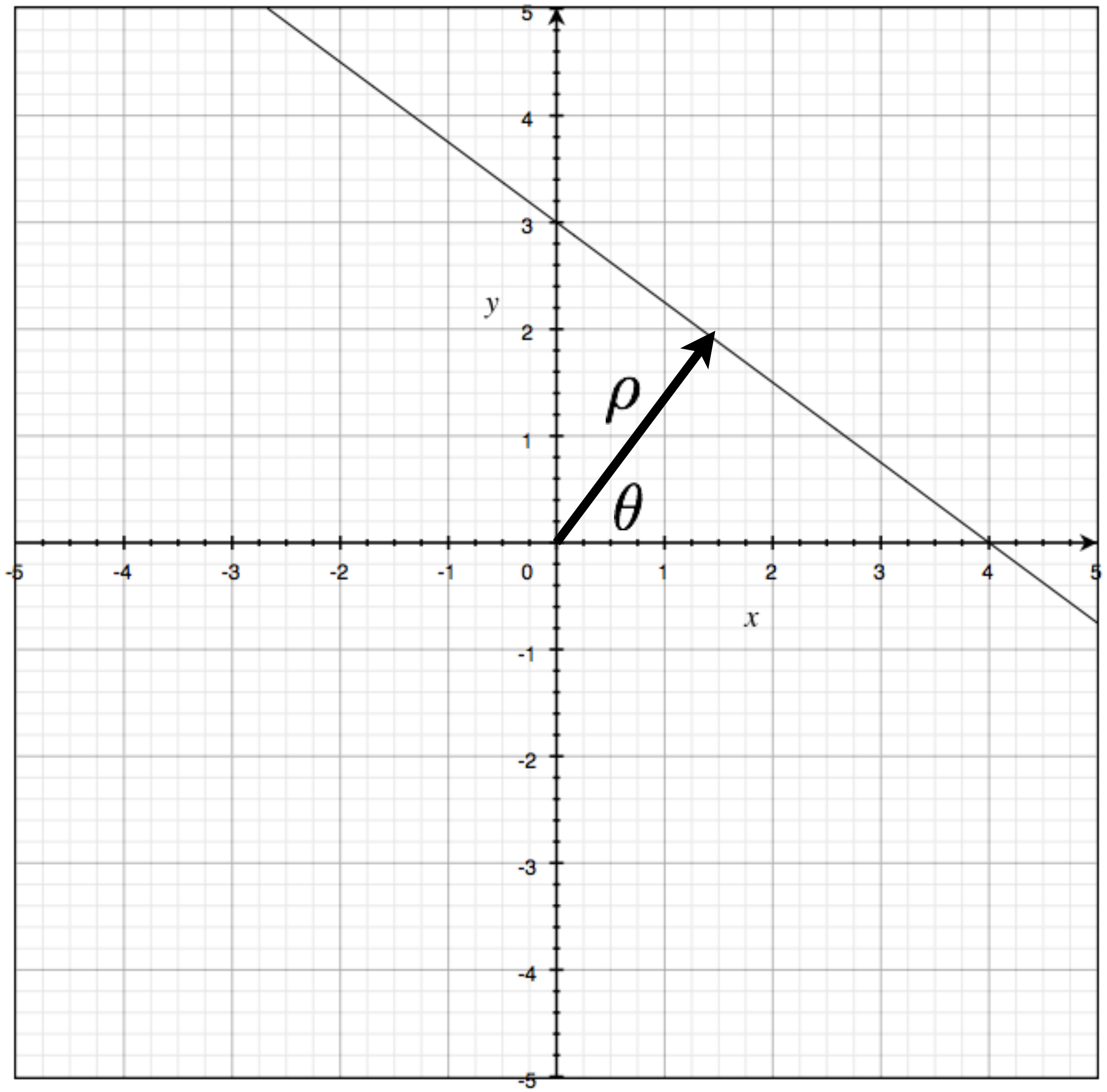
$$y = \underbrace{m}_{\text{slope}}x + \underbrace{b}_{\text{y-intercept}}$$



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lines: Normal form

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



Book's convention

$$x \sin\theta + y \cos\theta + r = 0$$

$$r \geq 0$$

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

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

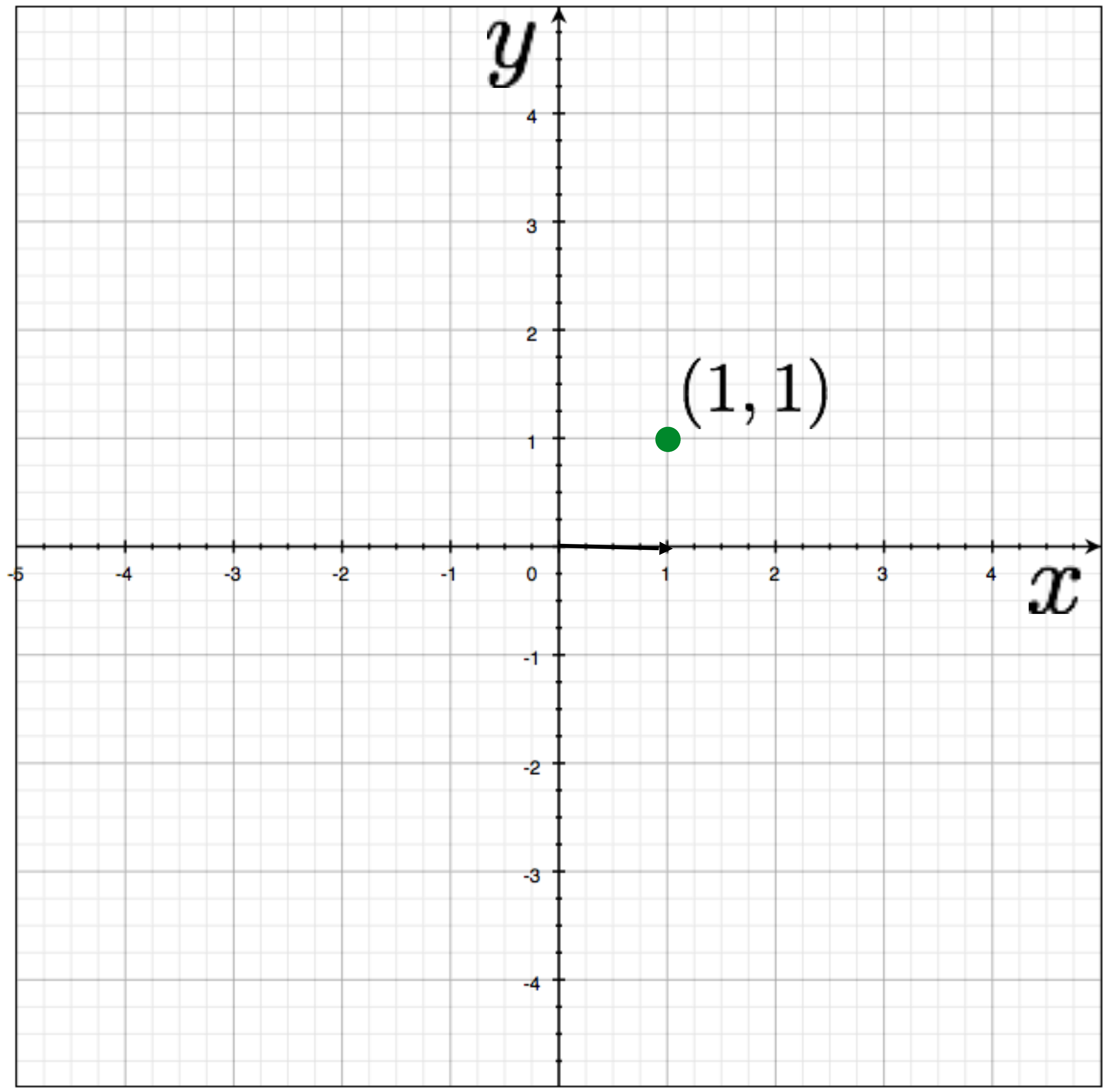
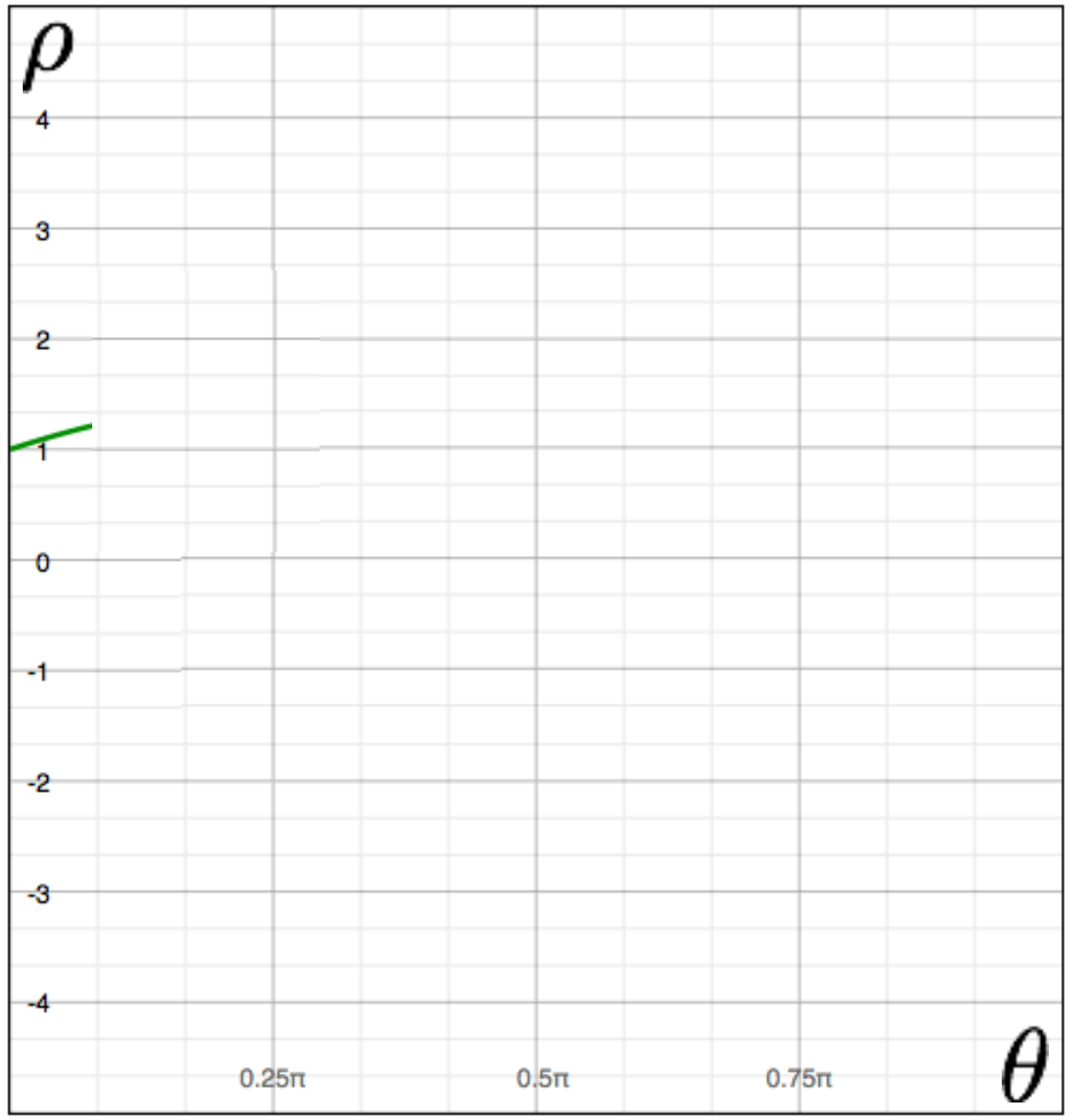


Image space

a point becomes?



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

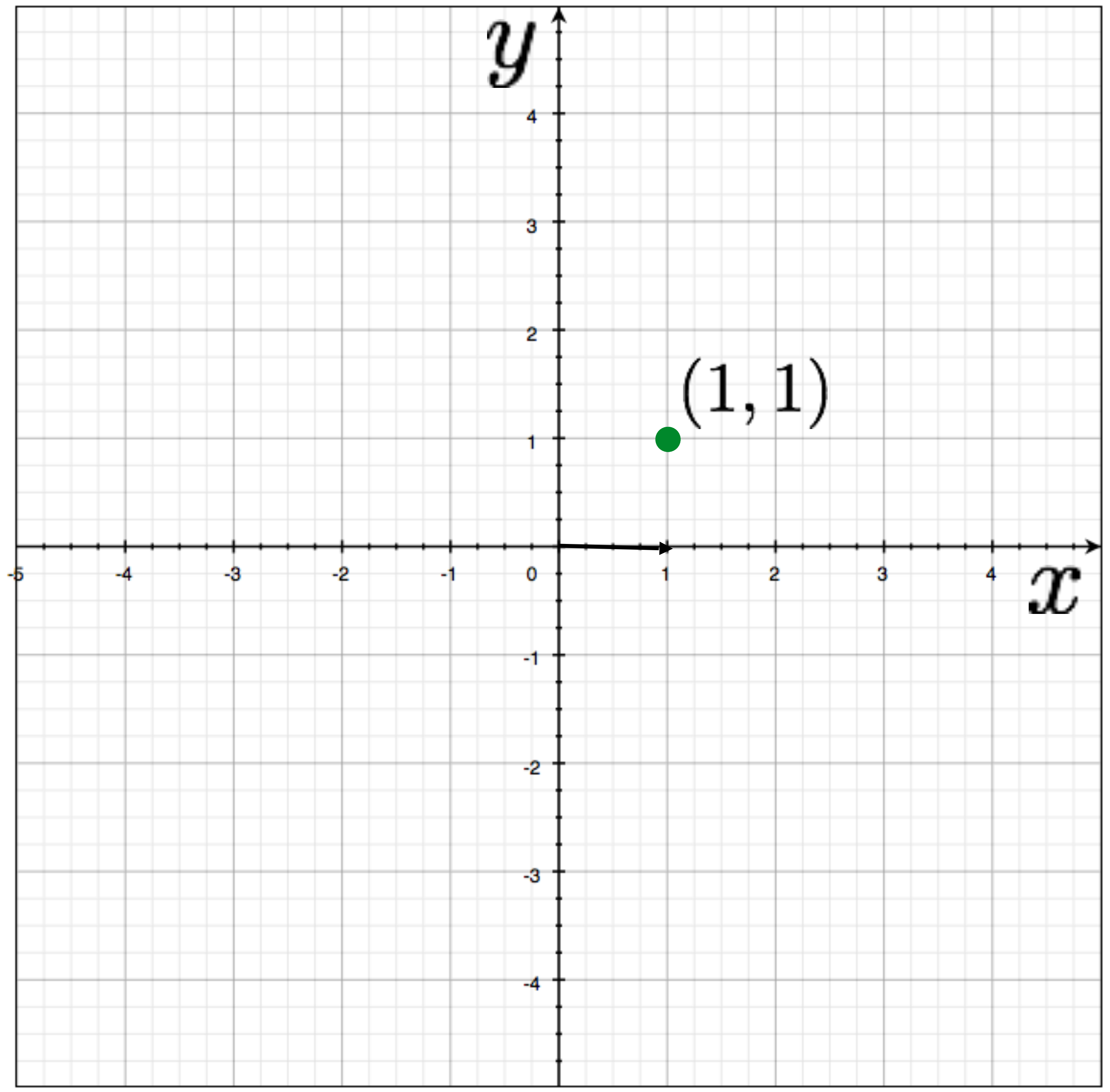
$$y = mx + b$$

parameters

parameters

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

variables



a point becomes a wave

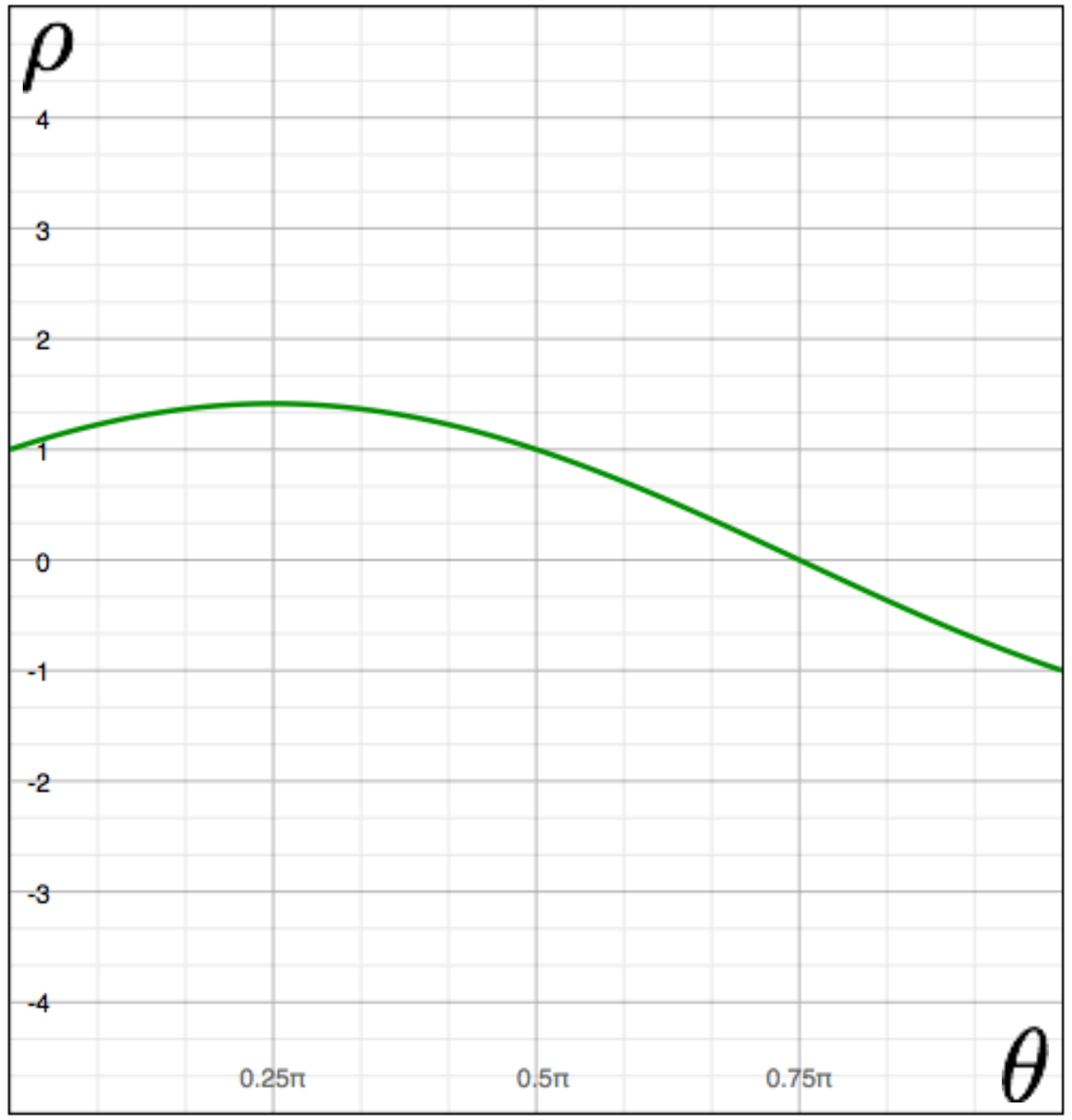


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

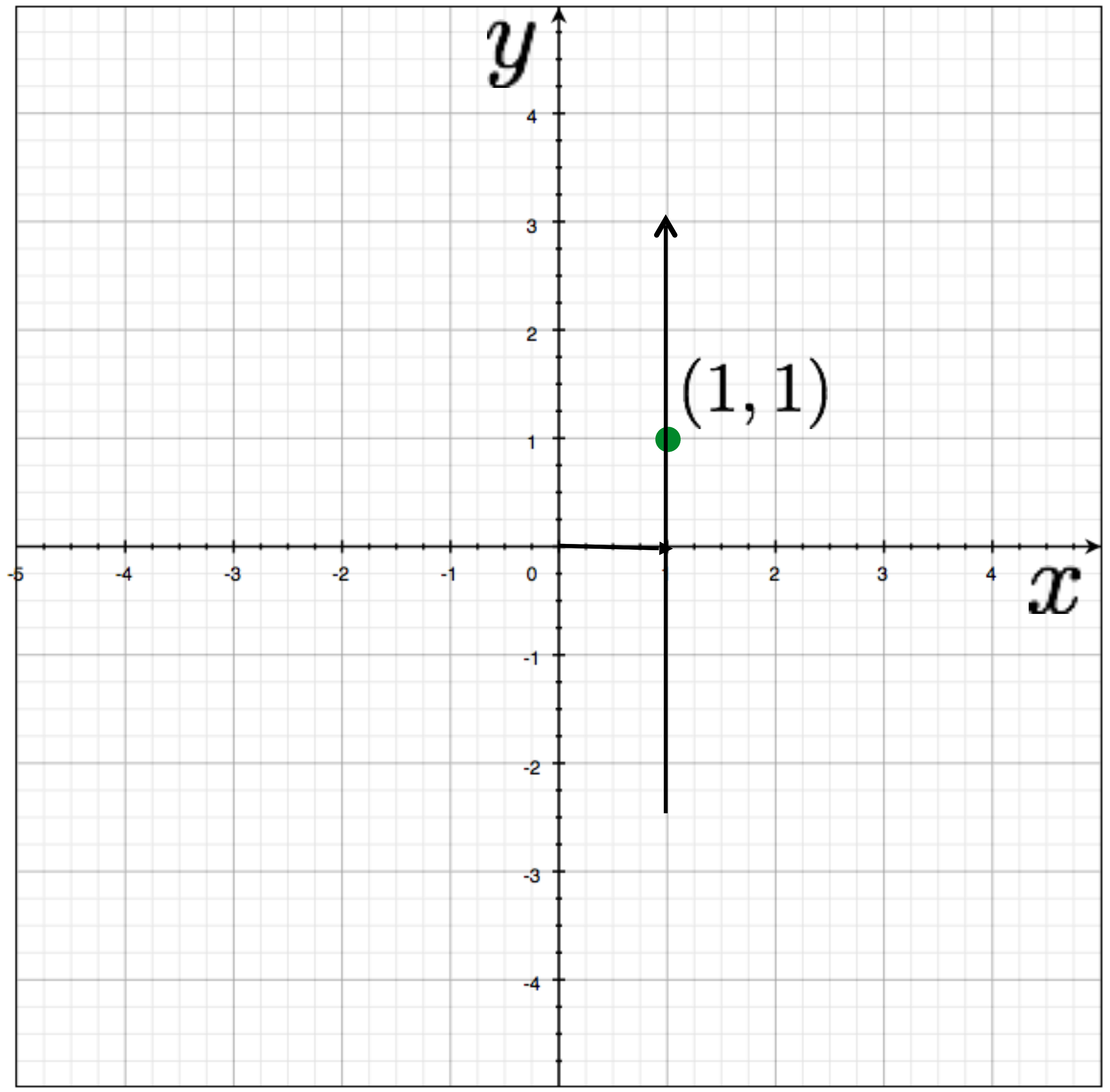
$$y = mx + b$$

parameters

parameters

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

variables



a line becomes?

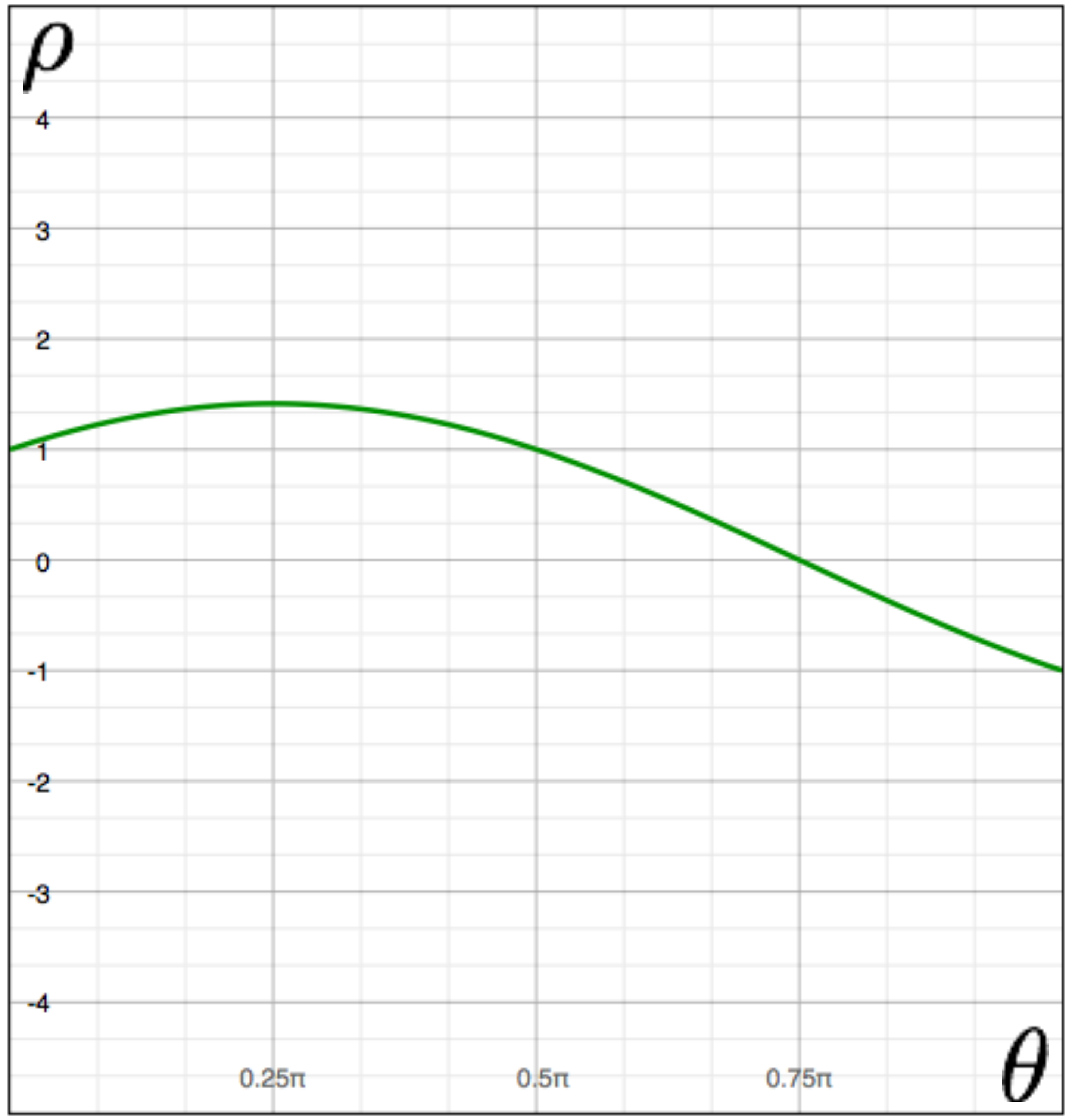


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

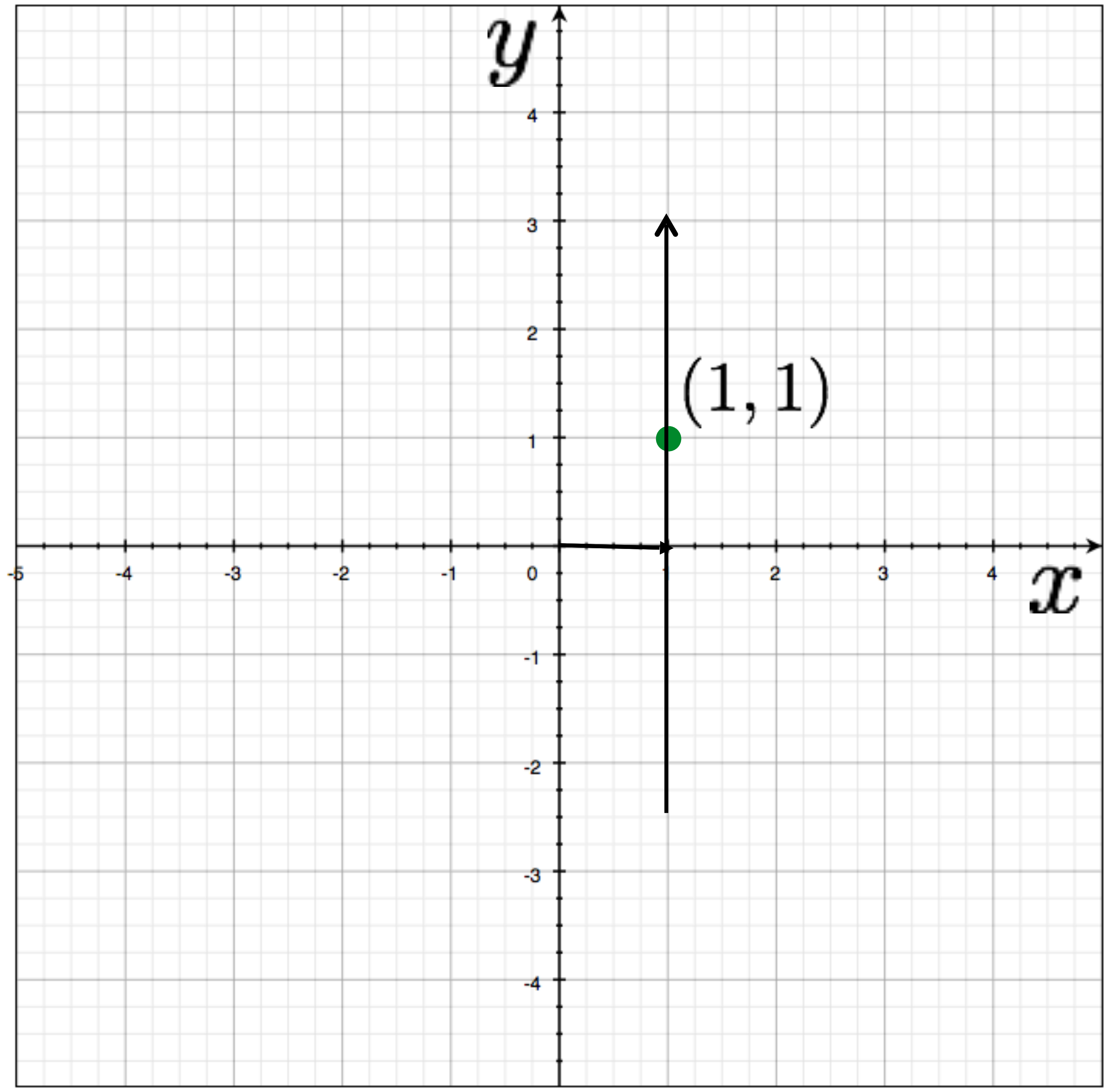
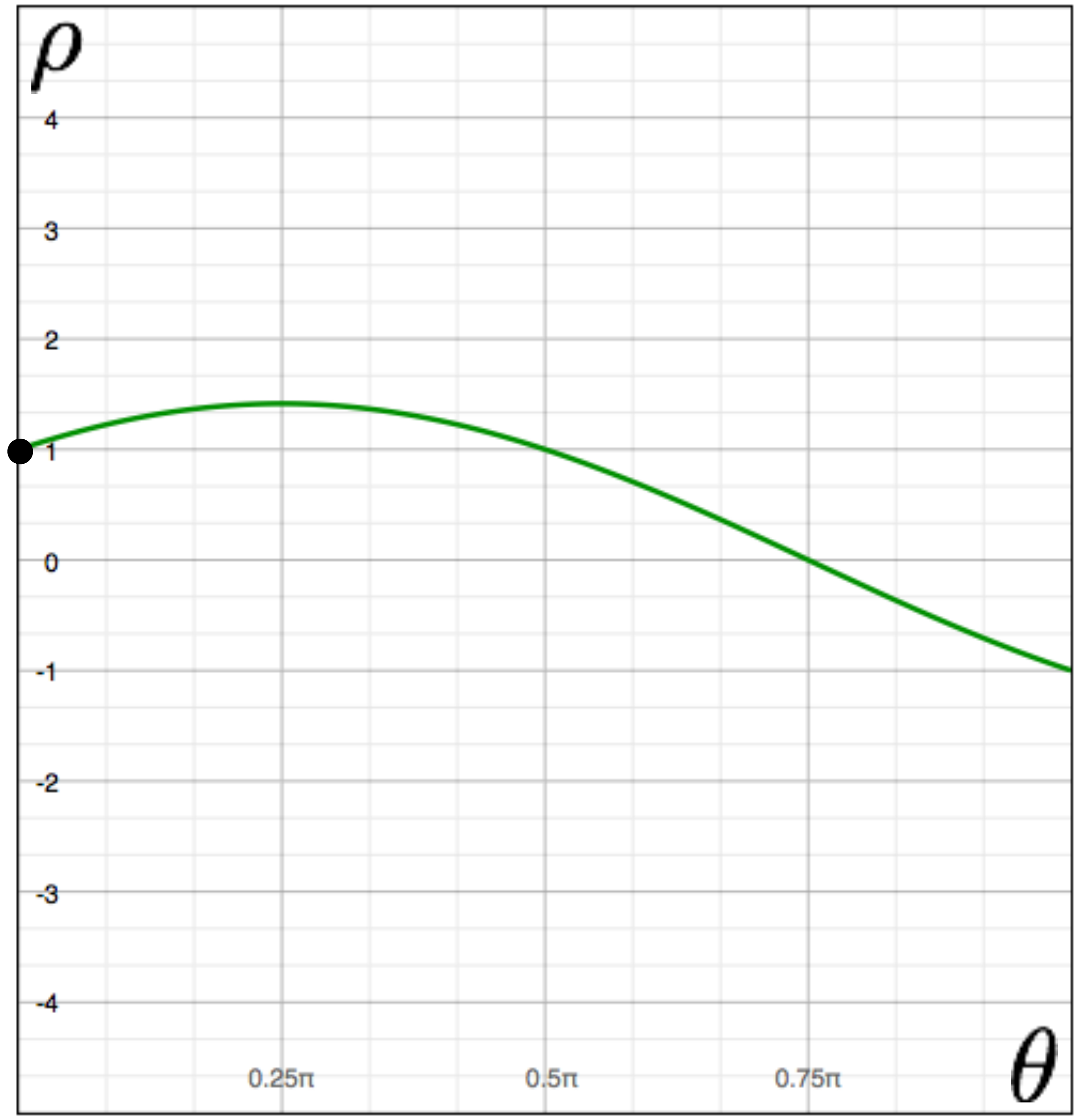


Image space

a line becomes a point



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

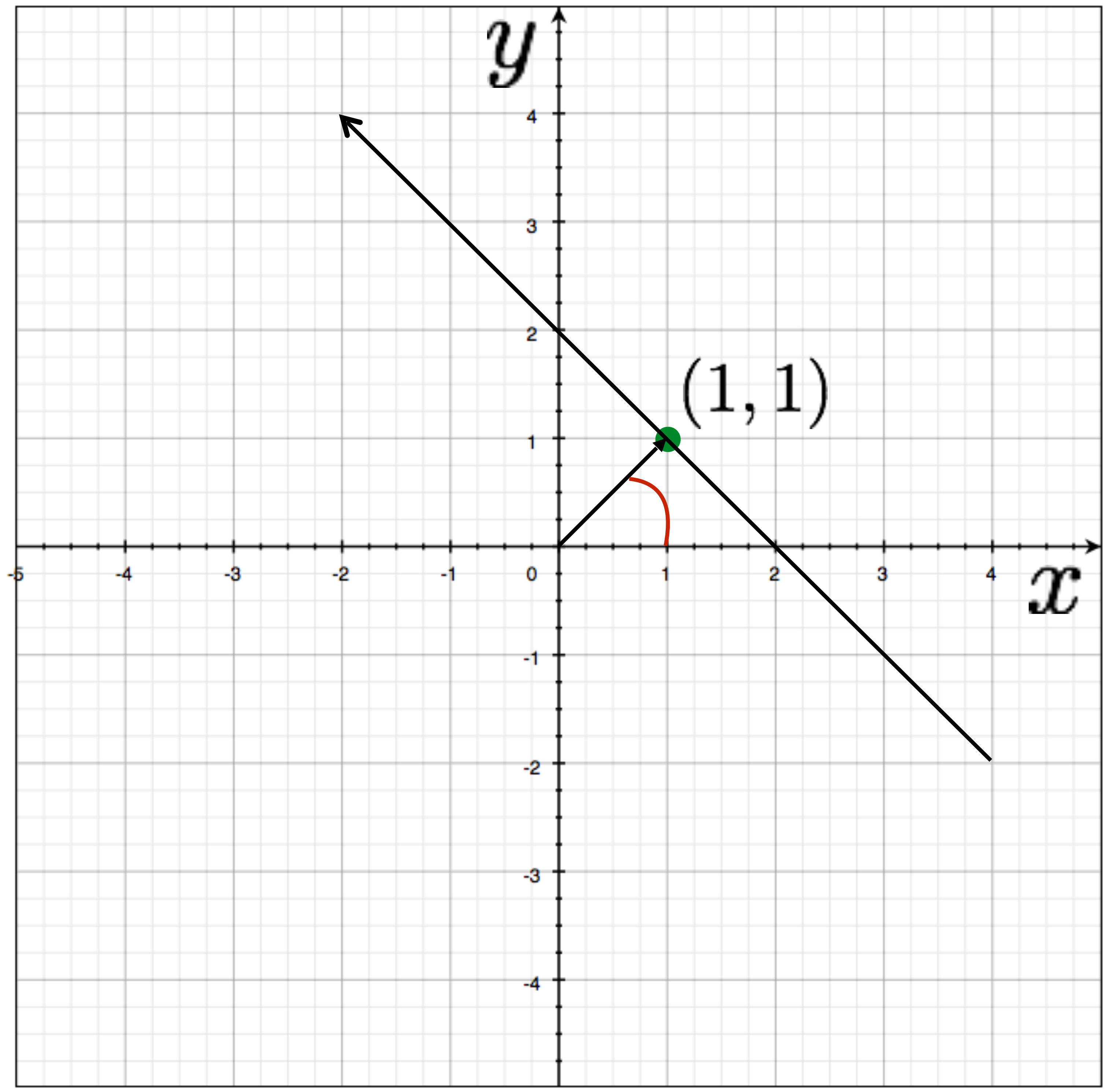
$$y = mx + b$$

parameters

parameters

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

variables



a line becomes?

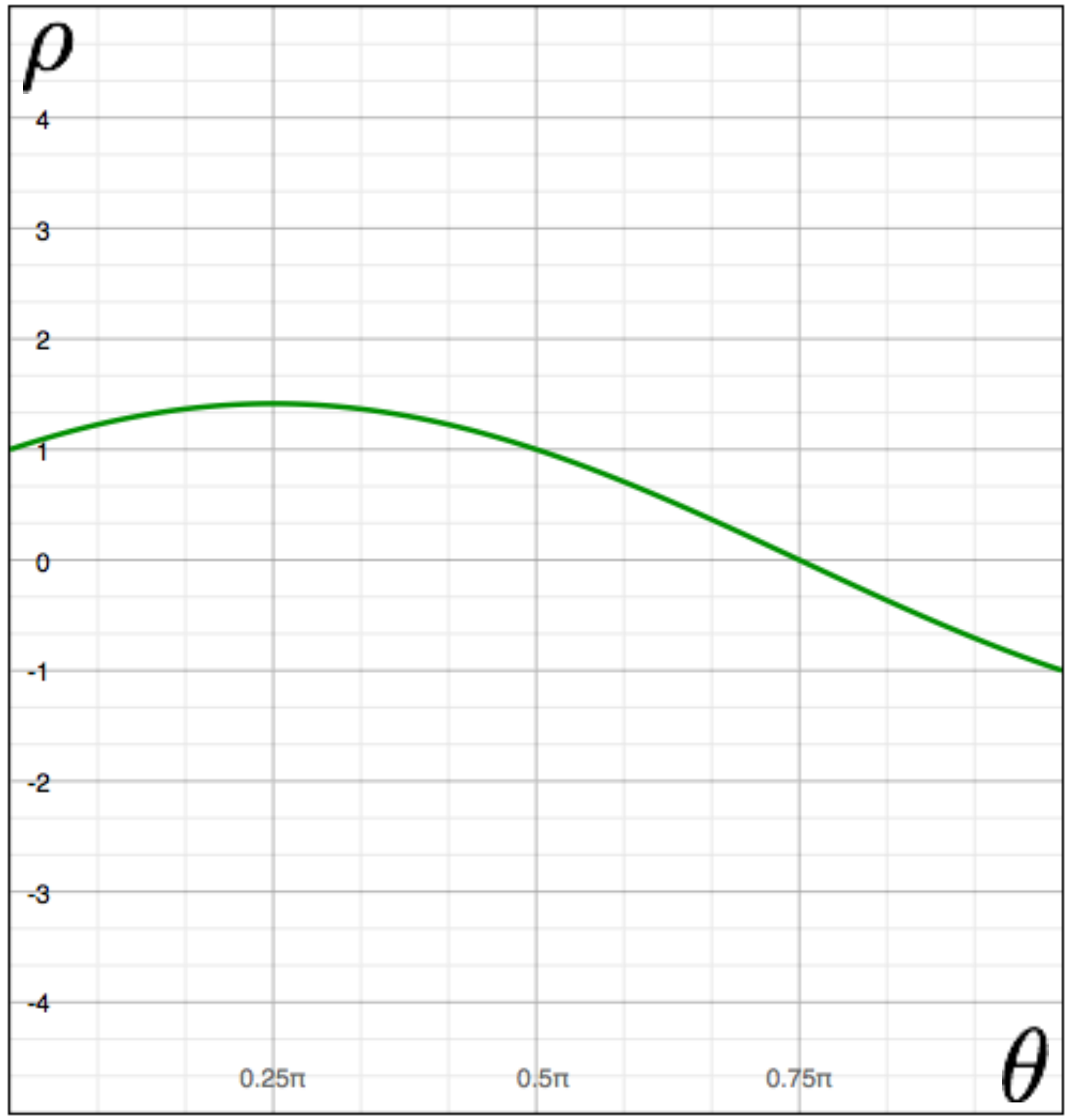


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

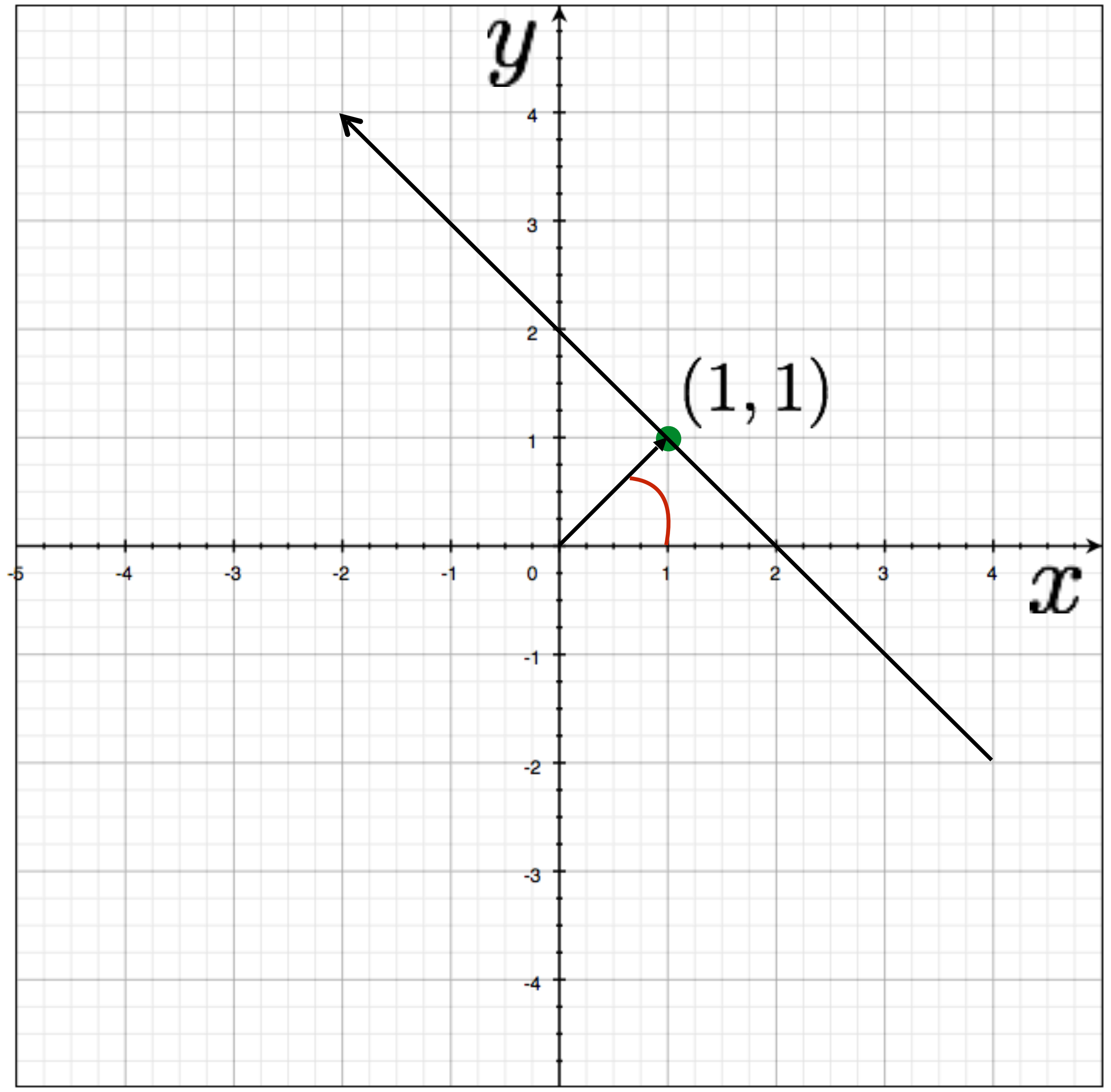
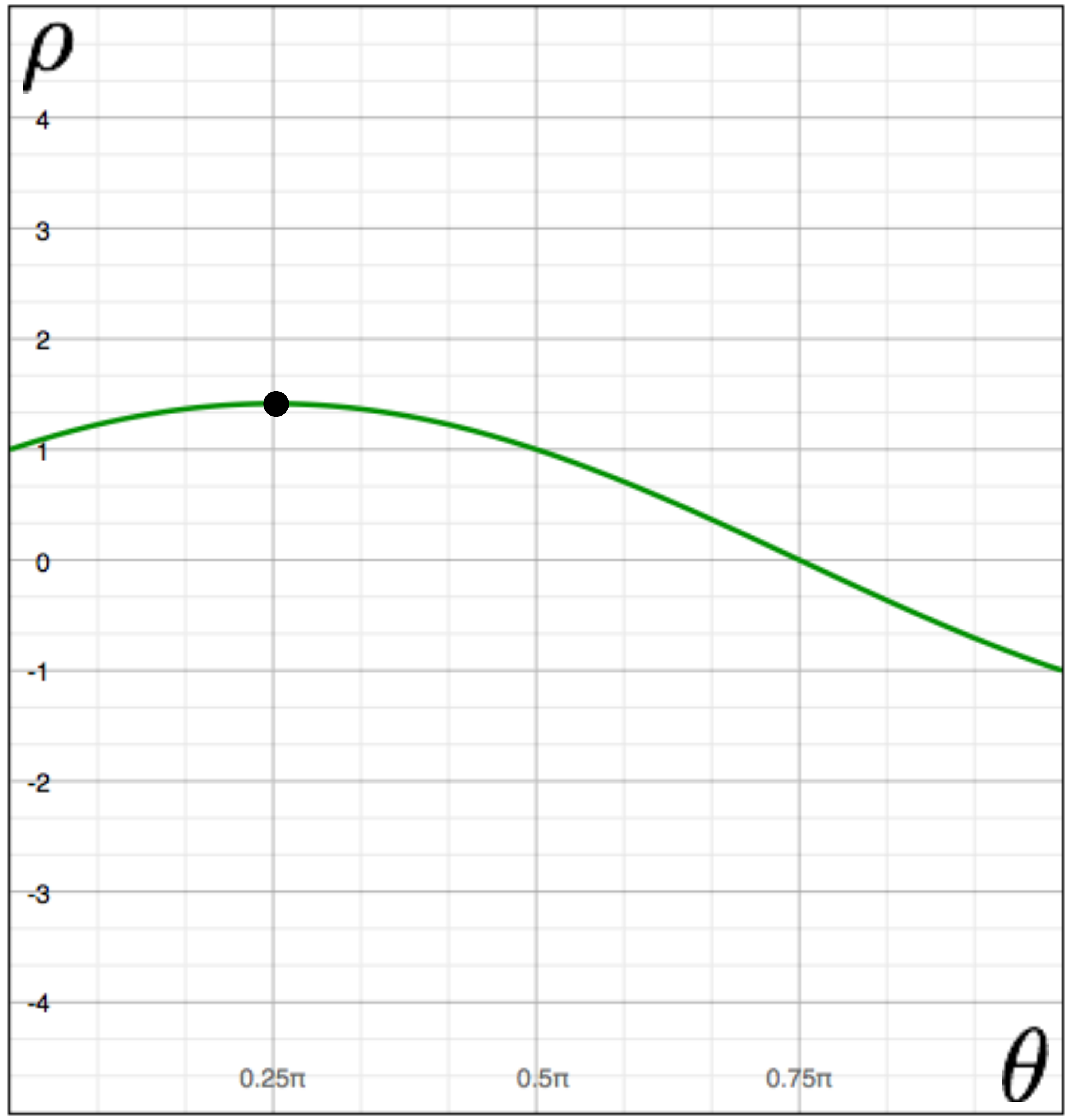


Image space

a line
becomes
a point



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

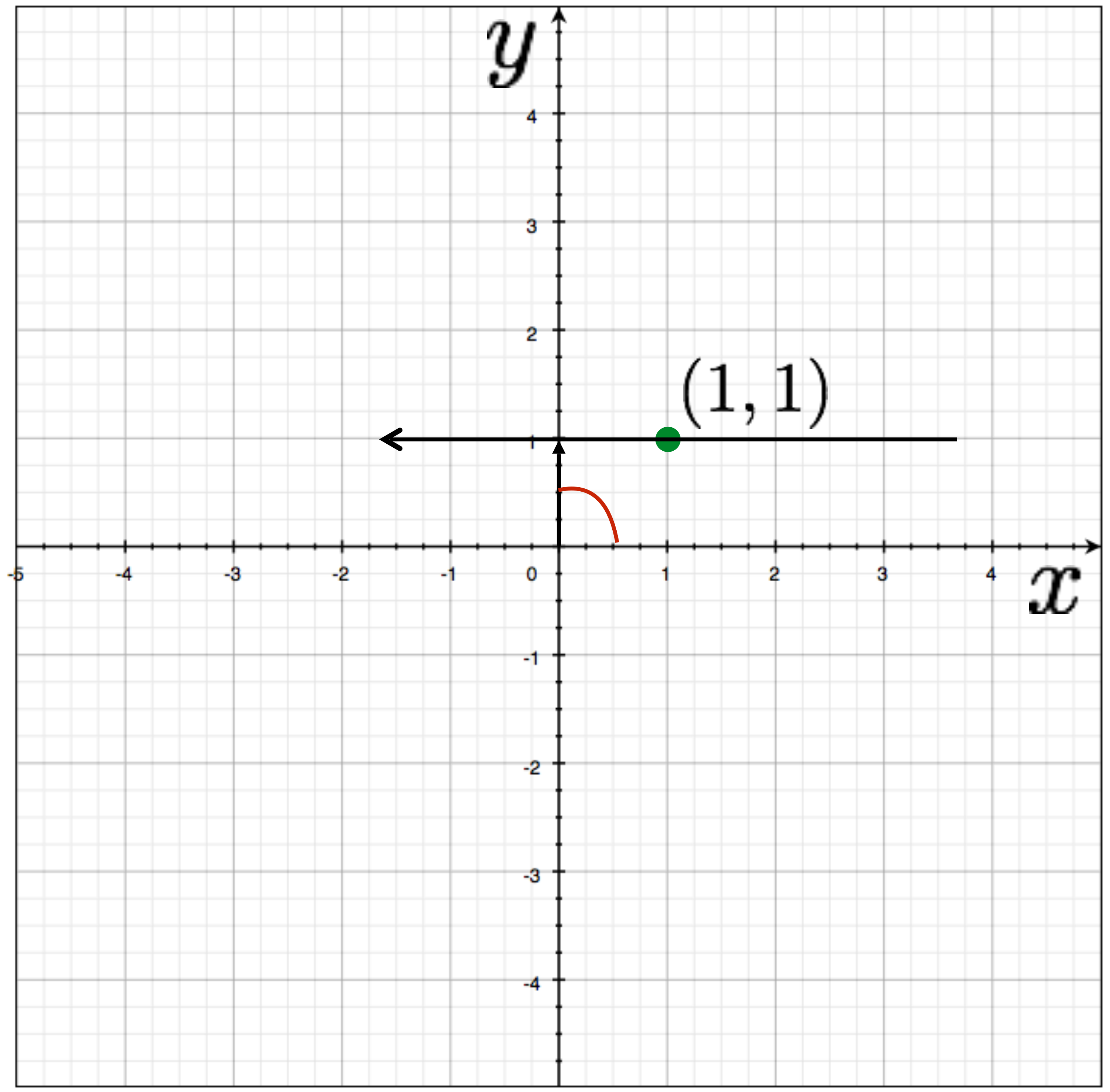
$$y = mx + b$$

parameters

parameters

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

variables



a line becomes a point

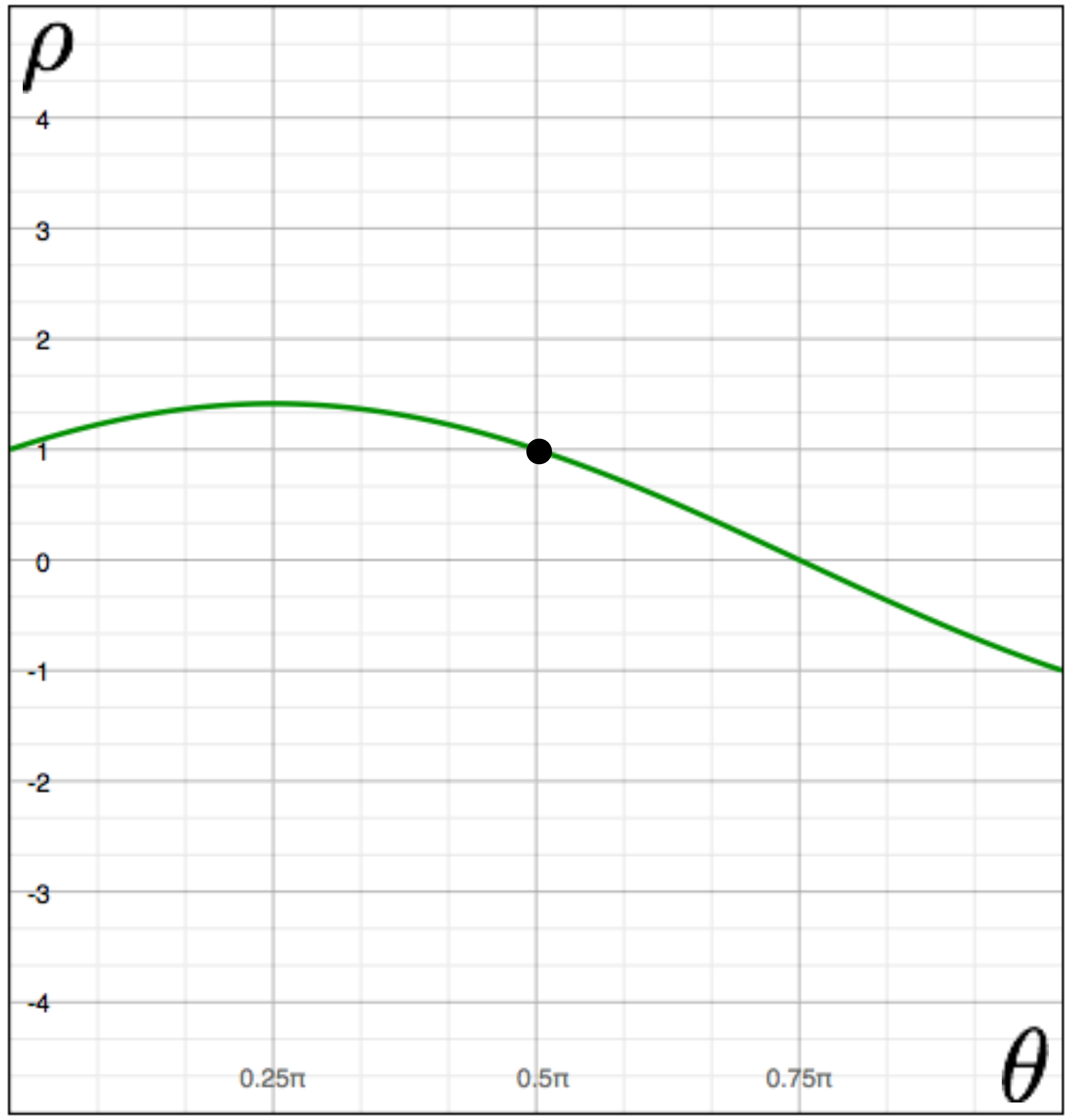


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

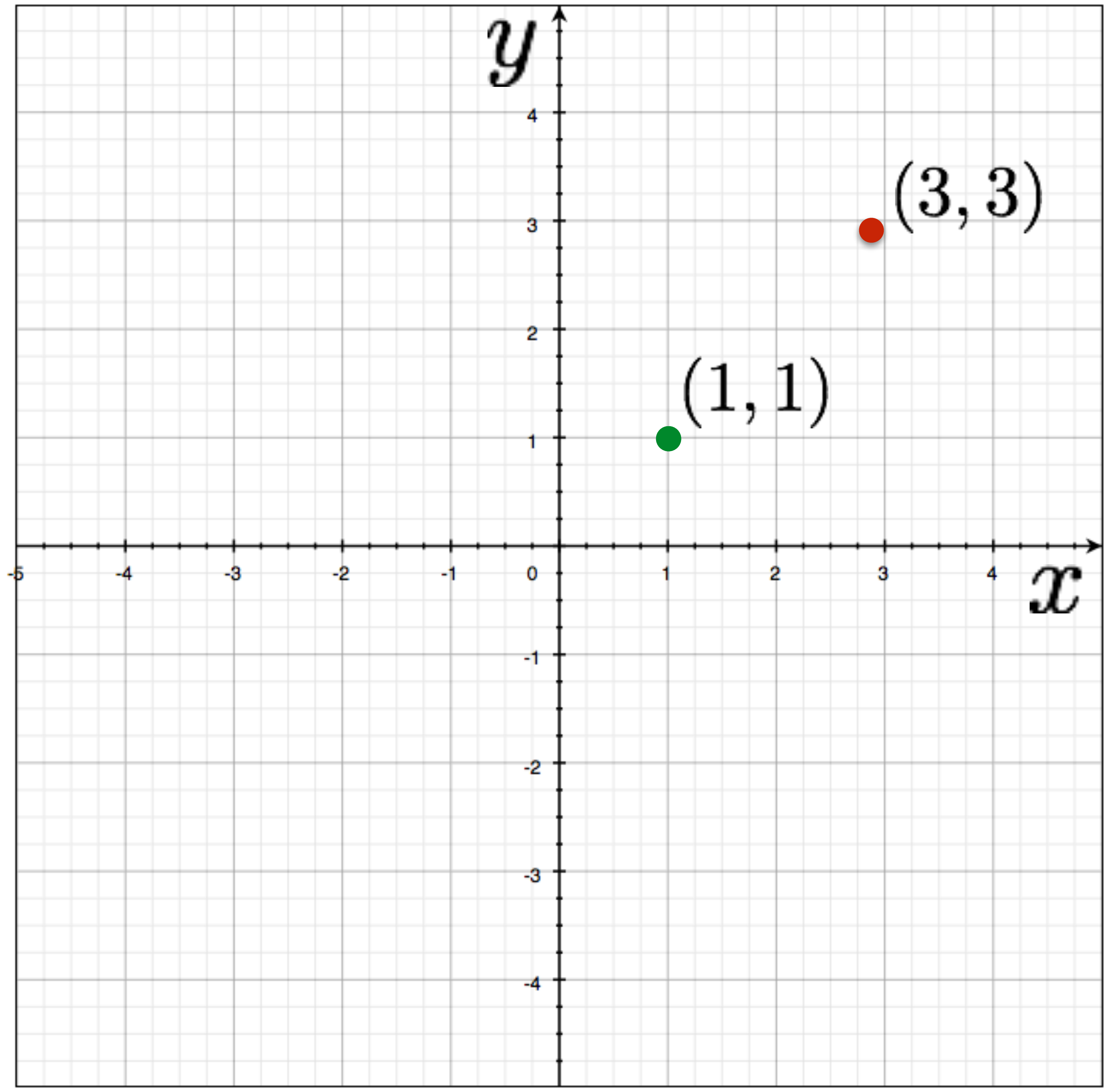
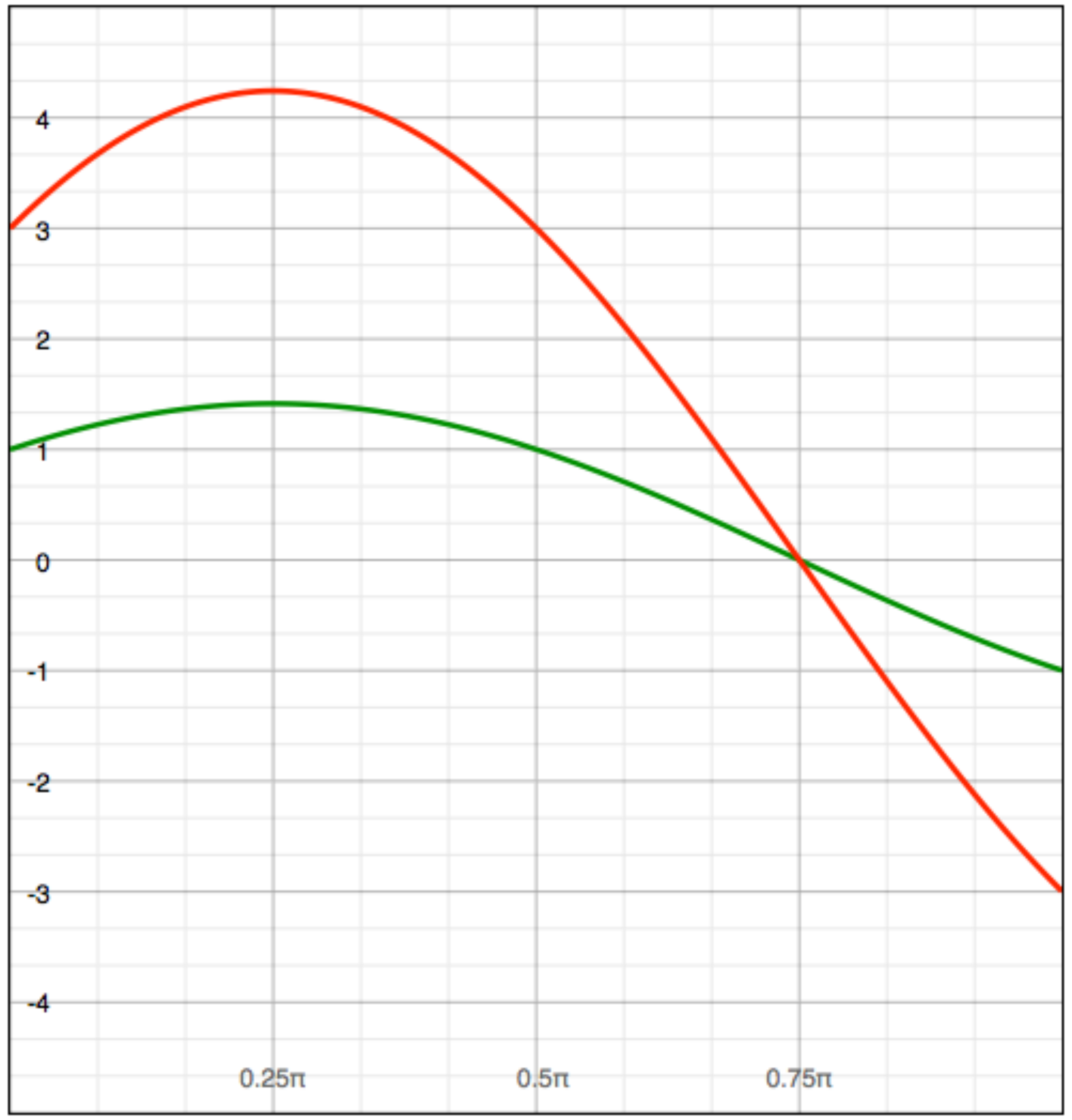


Image space

two points become?



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

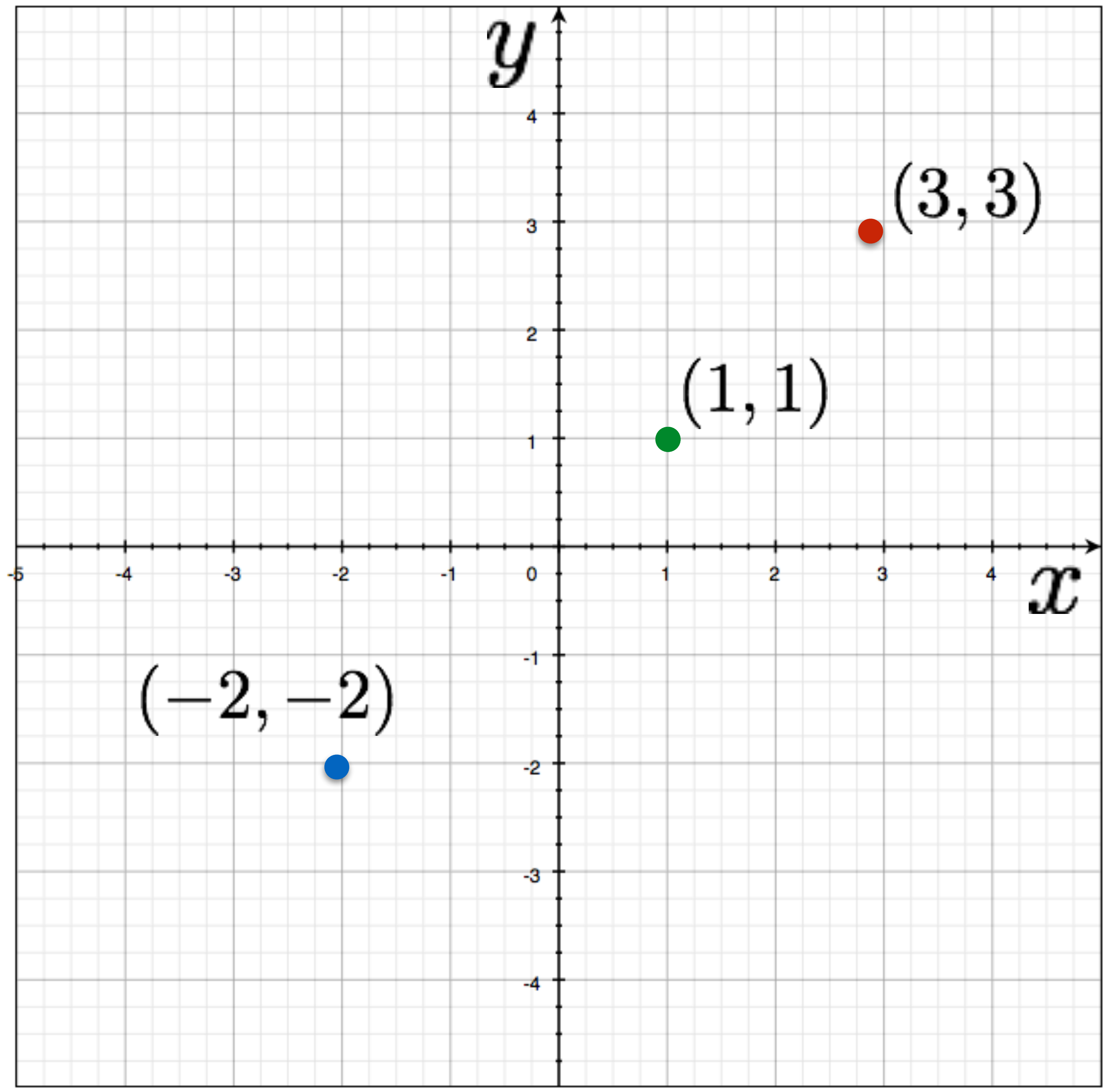
$$y = mx + b$$

parameters

parameters

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

variables



three points become?

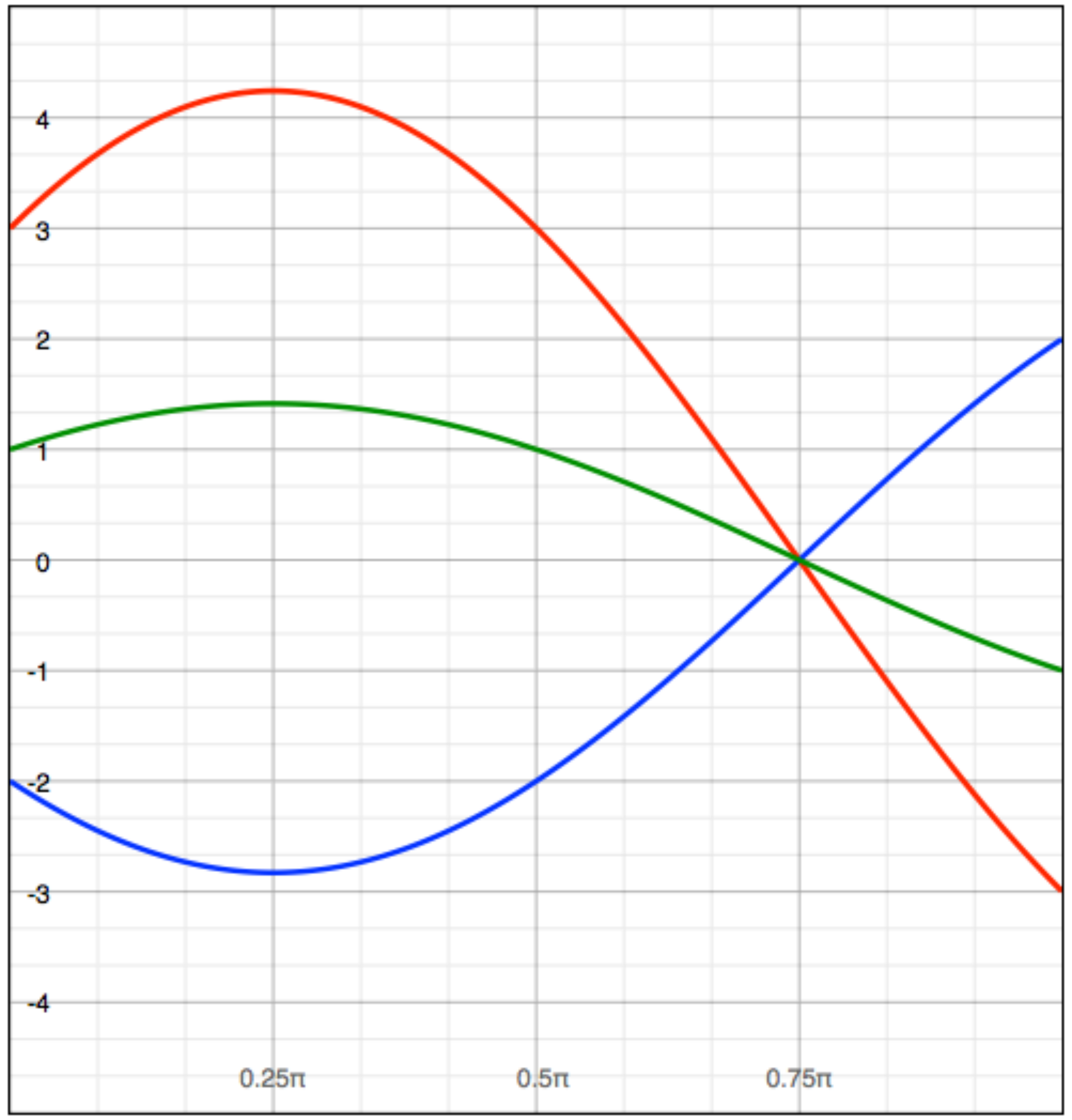


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

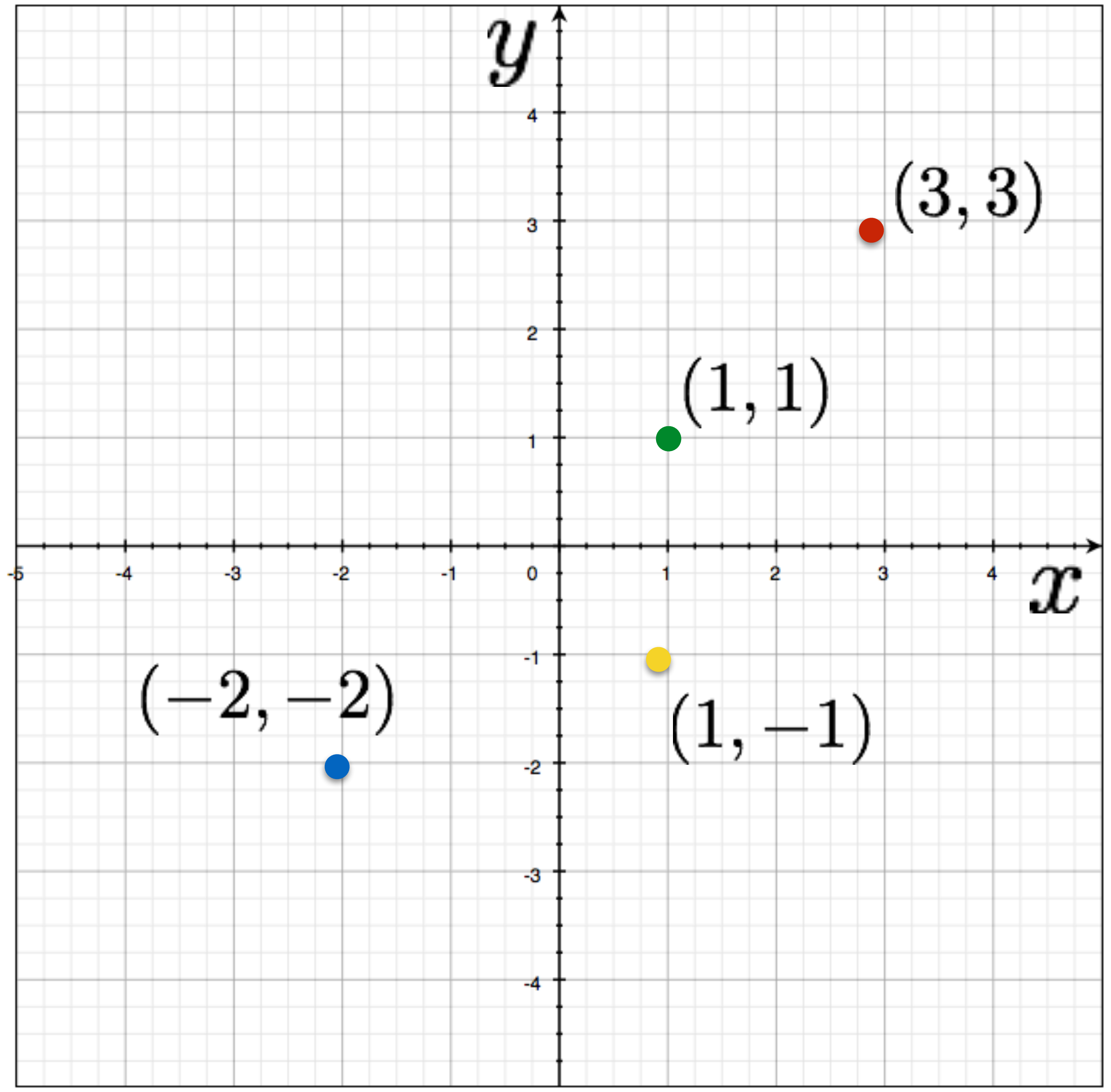
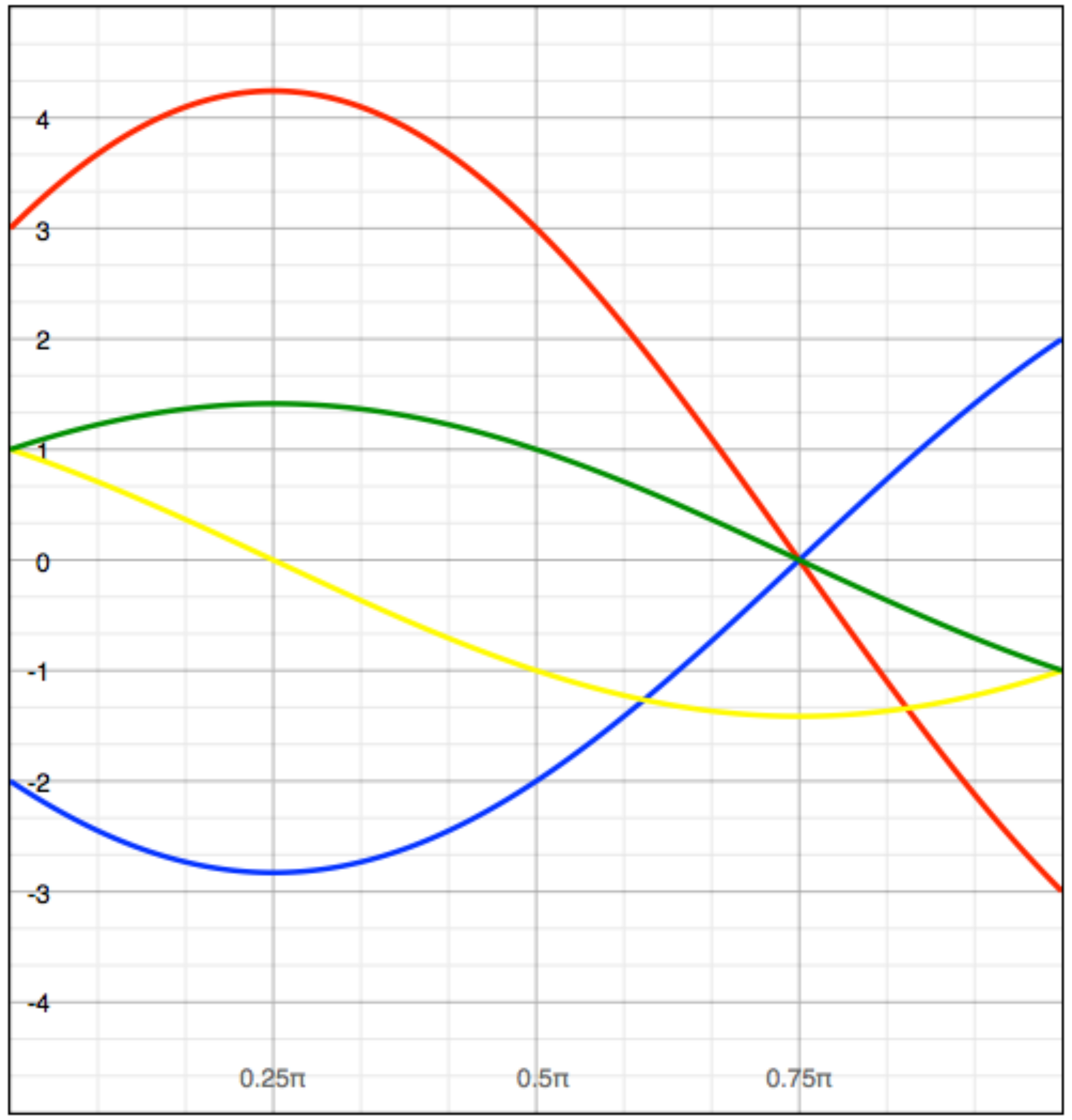


Image space

four points become?



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform for Lines (switching to books notation)

Idea: Each point votes for the lines that pass through it

— A line is the set of points, (x, y), such that

$$x \sin\theta + y \cos\theta + r = 0$$

— Different choices of θ, r give different lines

Hough Transform for Lines (switching to books notation)

Idea: Each point votes for the lines that pass through it

— A line is the set of points, (x, y), such that

$$x \sin\theta + y \cos\theta + r = 0$$

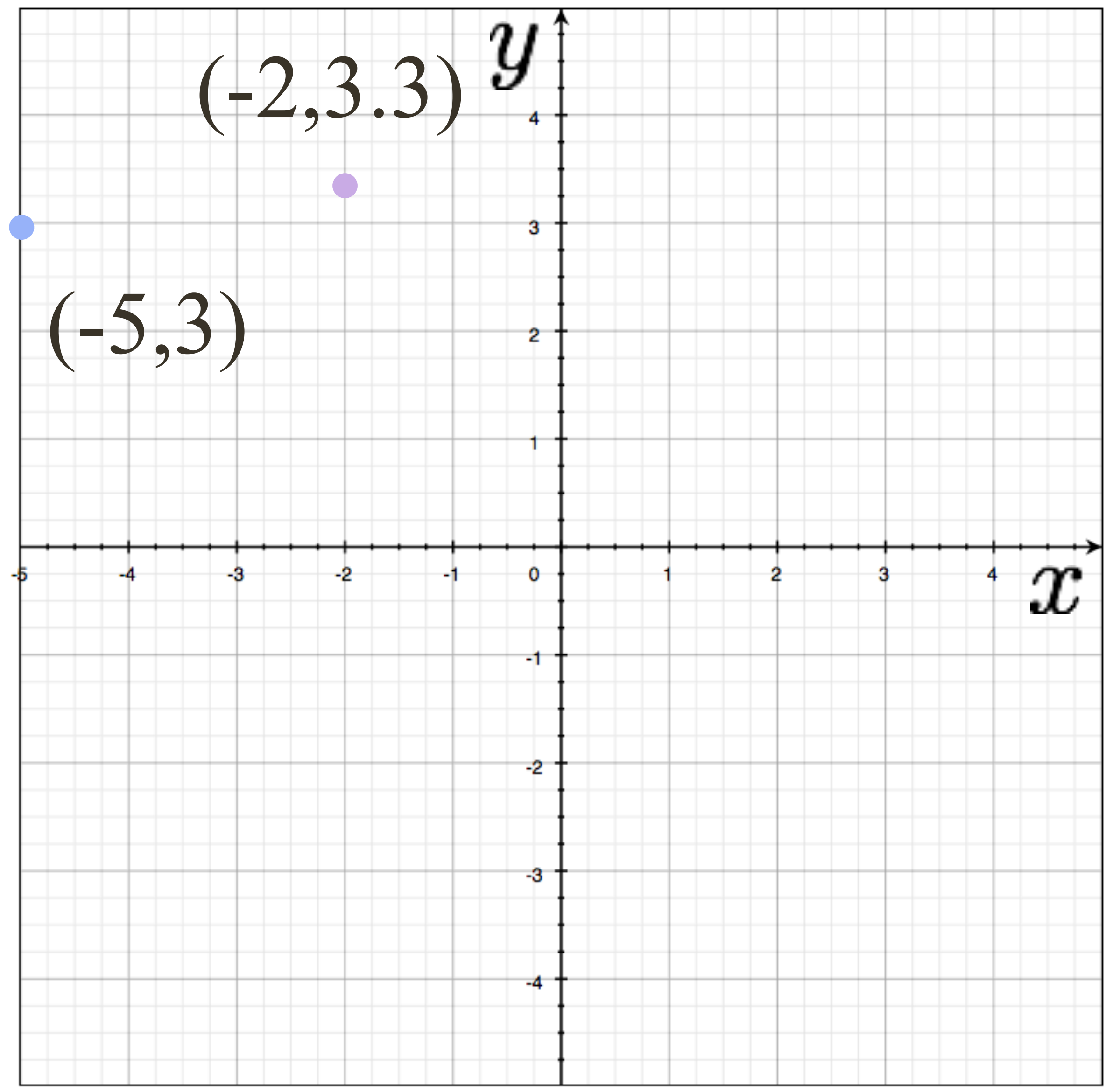
— Different choices of θ, r give different lines

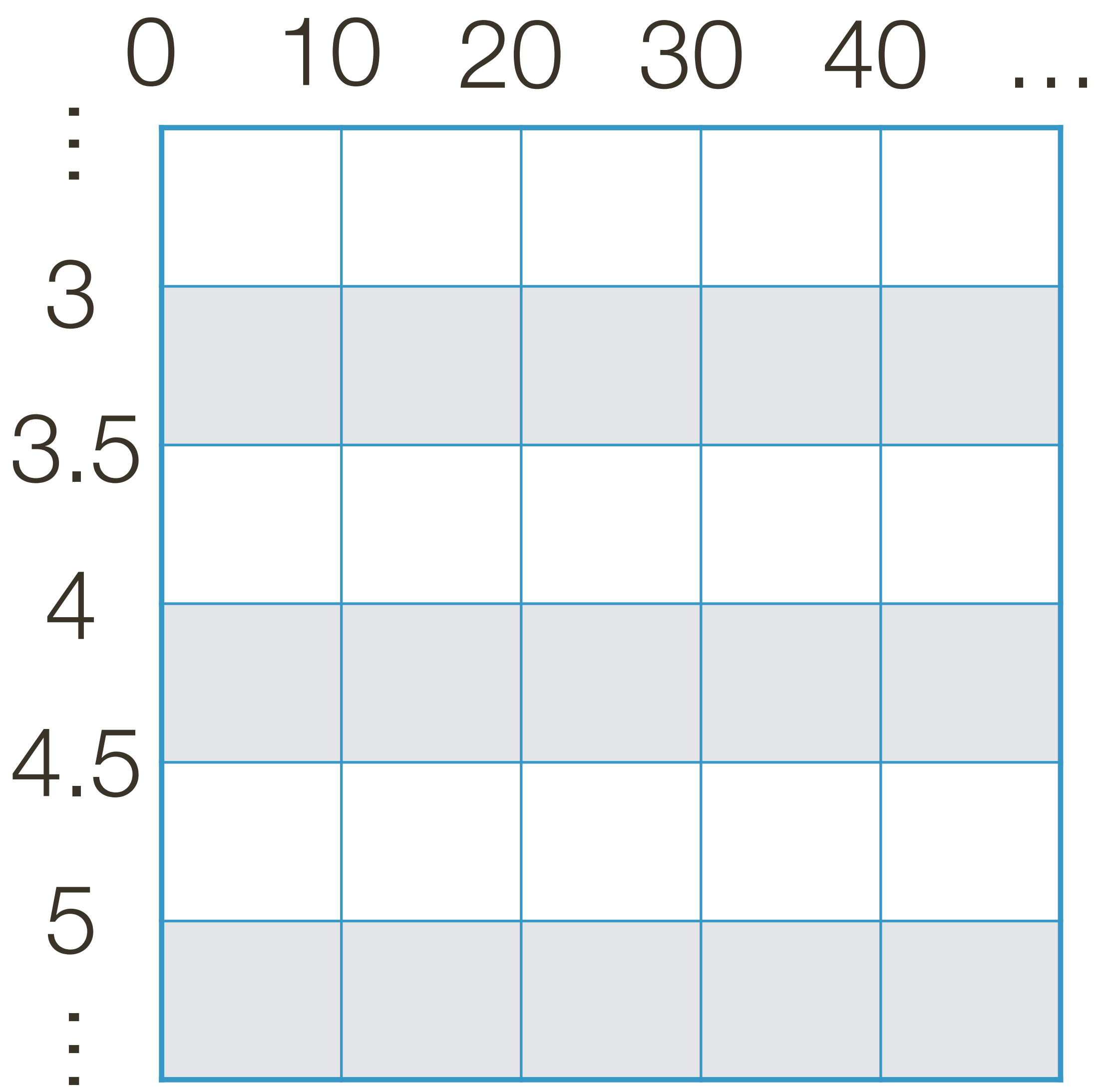
— For any (x, y) there is a one parameter family of lines through this point. Just let (x, y) be constants and for each value of θ the value of r will be determined

— Each point enters votes for each line in the family

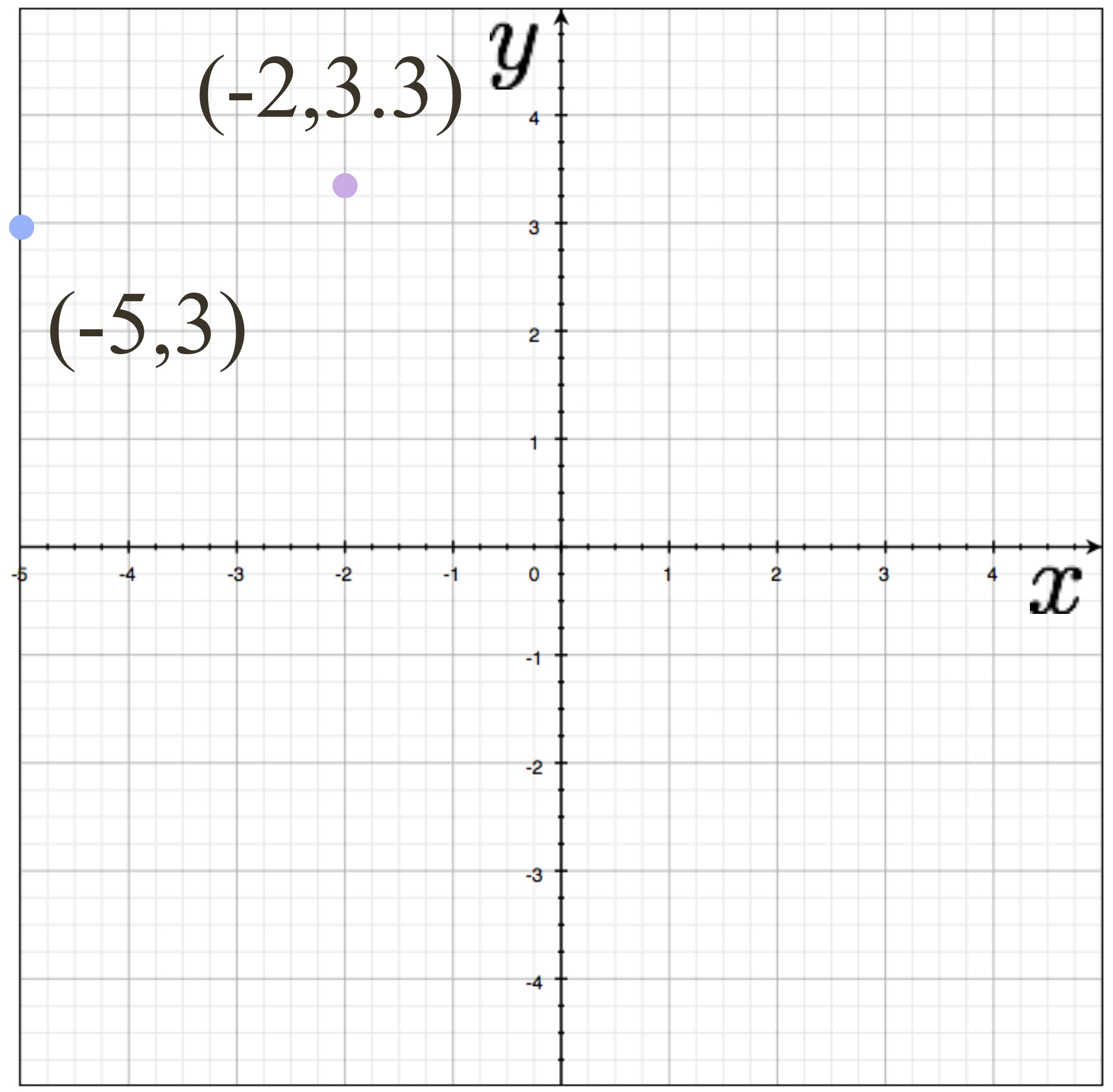
— If there is a line that has lots of votes, that will be the line passing near the points that voted for it

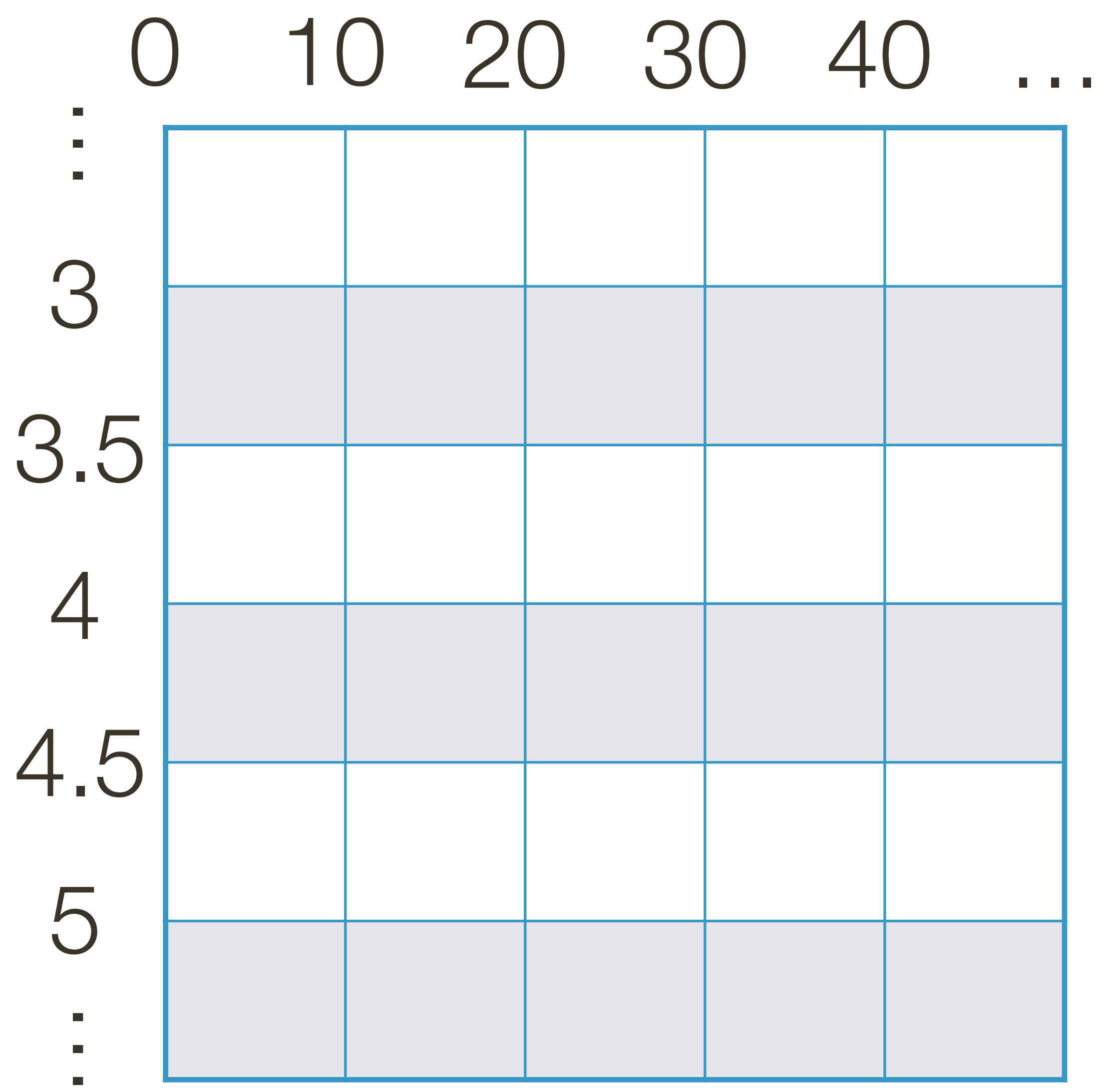
Example: Hough Transform for Lines





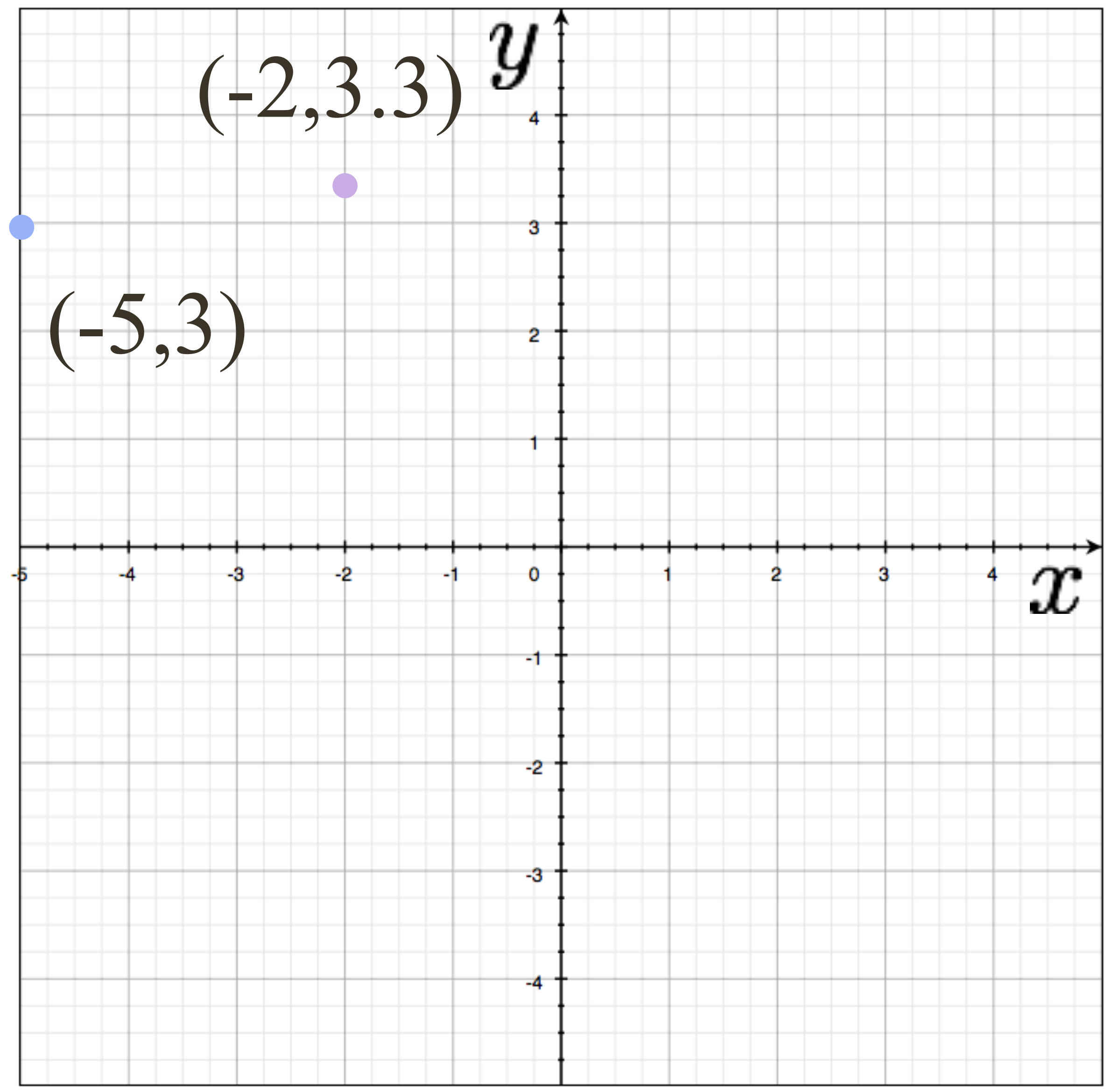
Example: Hough Transform for Lines





$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$

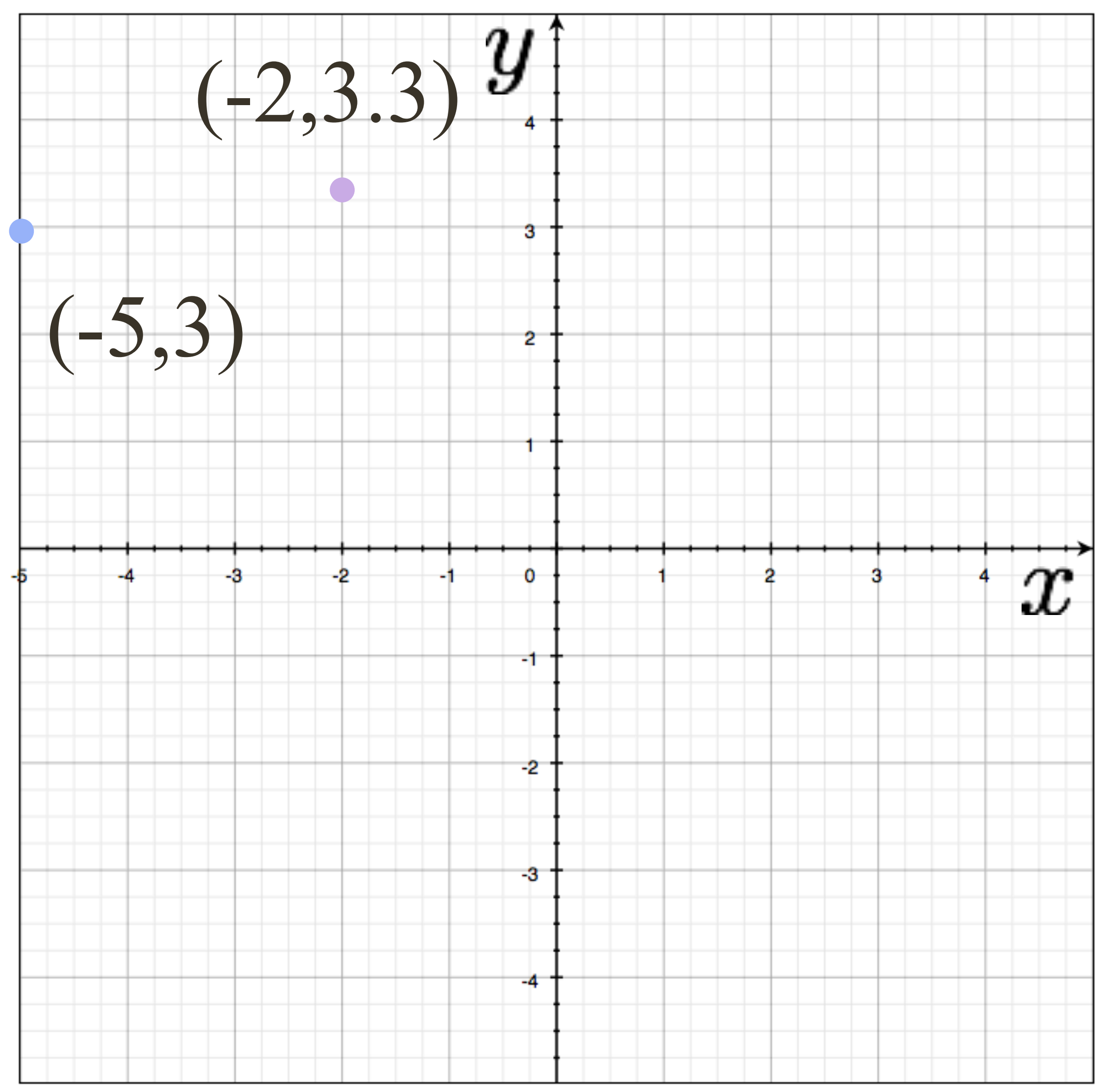
Example: Hough Transform for Lines

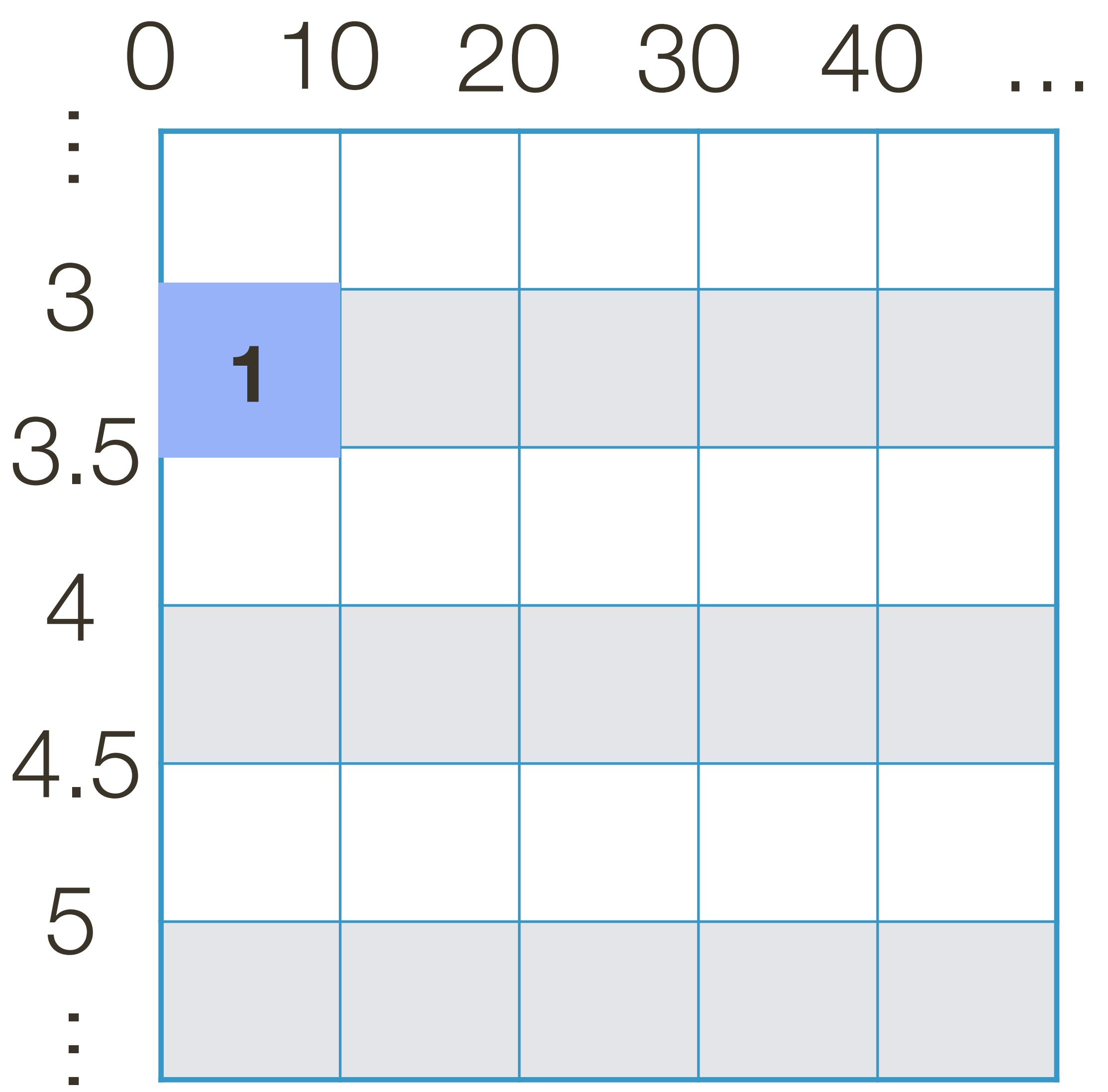


	0	10	20	30	40	…
⋮						
3	**1**					
3.5						
4						
4.5						
5						
⋮						

$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$

Example: Hough Transform for Lines

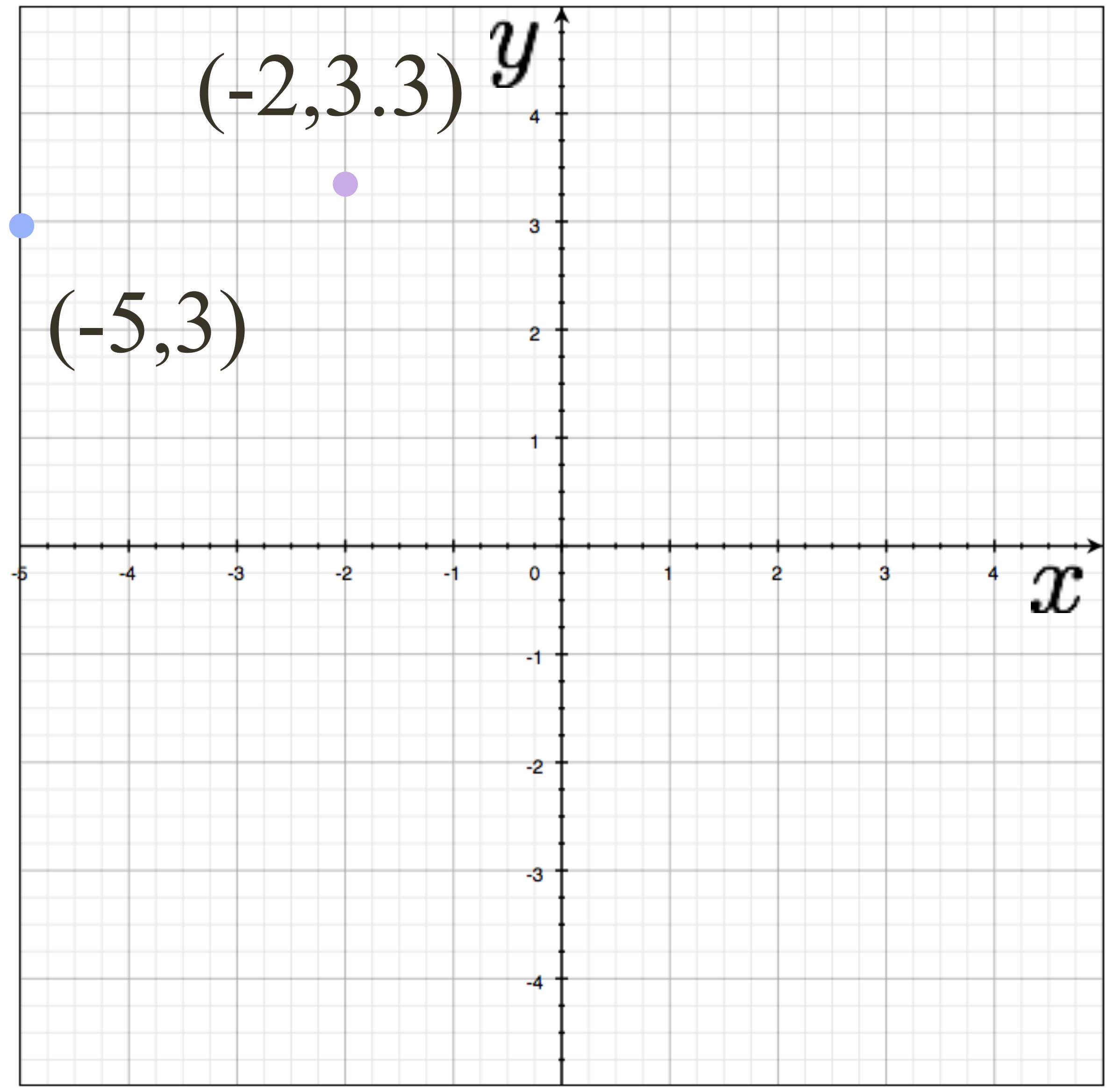


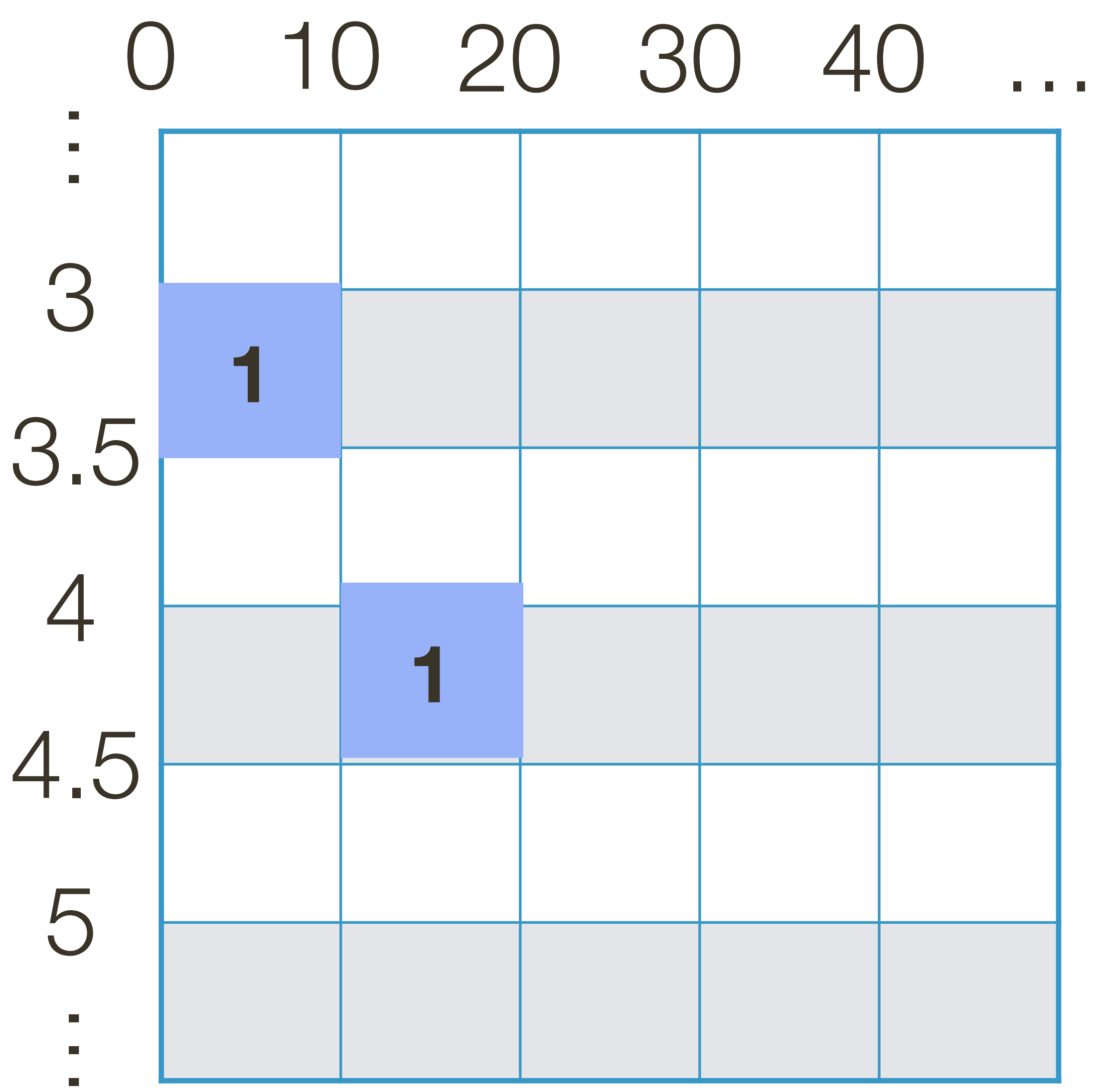


$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$

$$-5\sin(15^\circ) - 3\cos(15^\circ) + r = 0 => r = 4.18$$

Example: Hough Transform for Lines

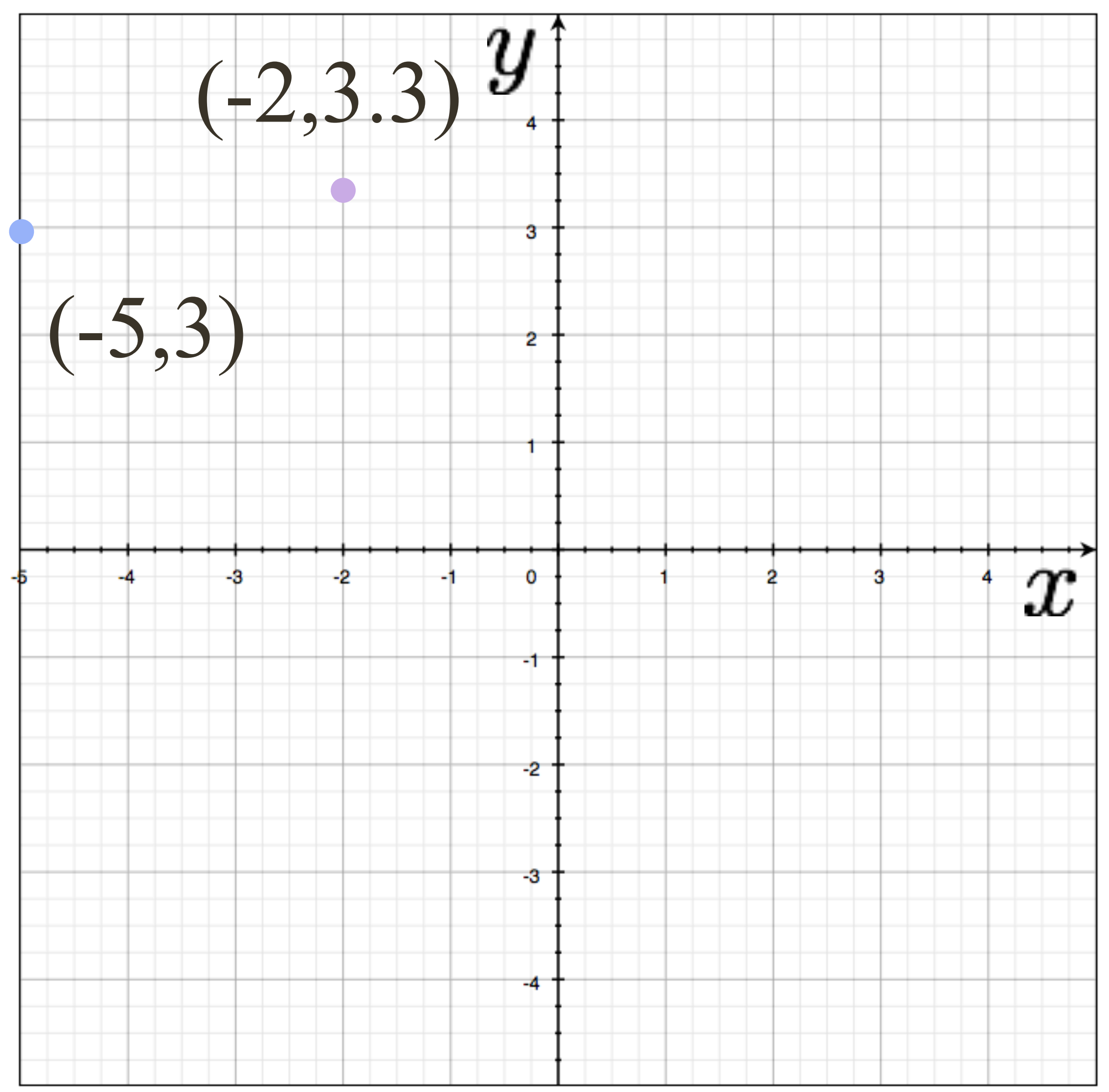


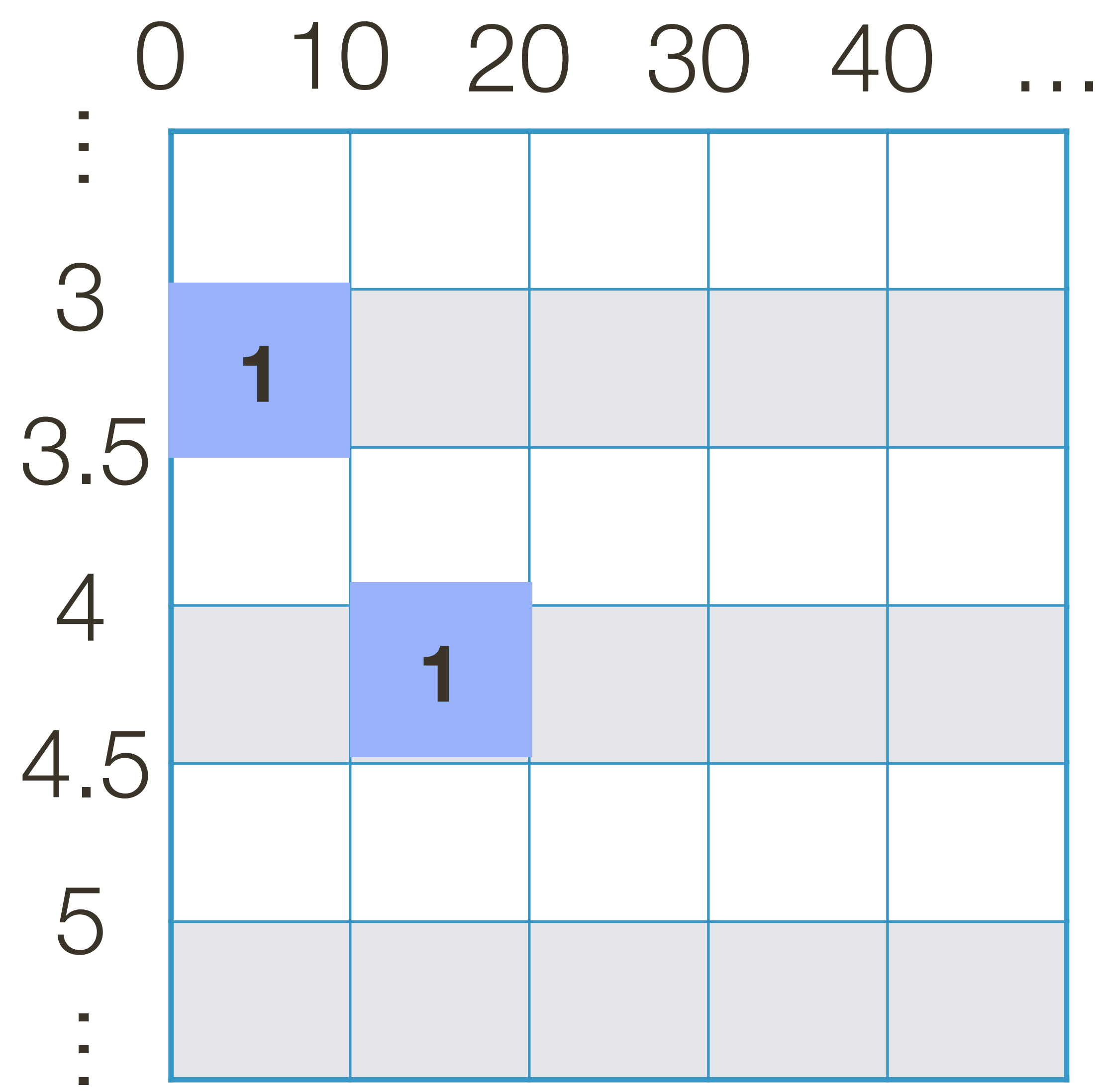


$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$

$$-5\sin(15^\circ) - 3\cos(15^\circ) + r = 0 => r = 4.18$$

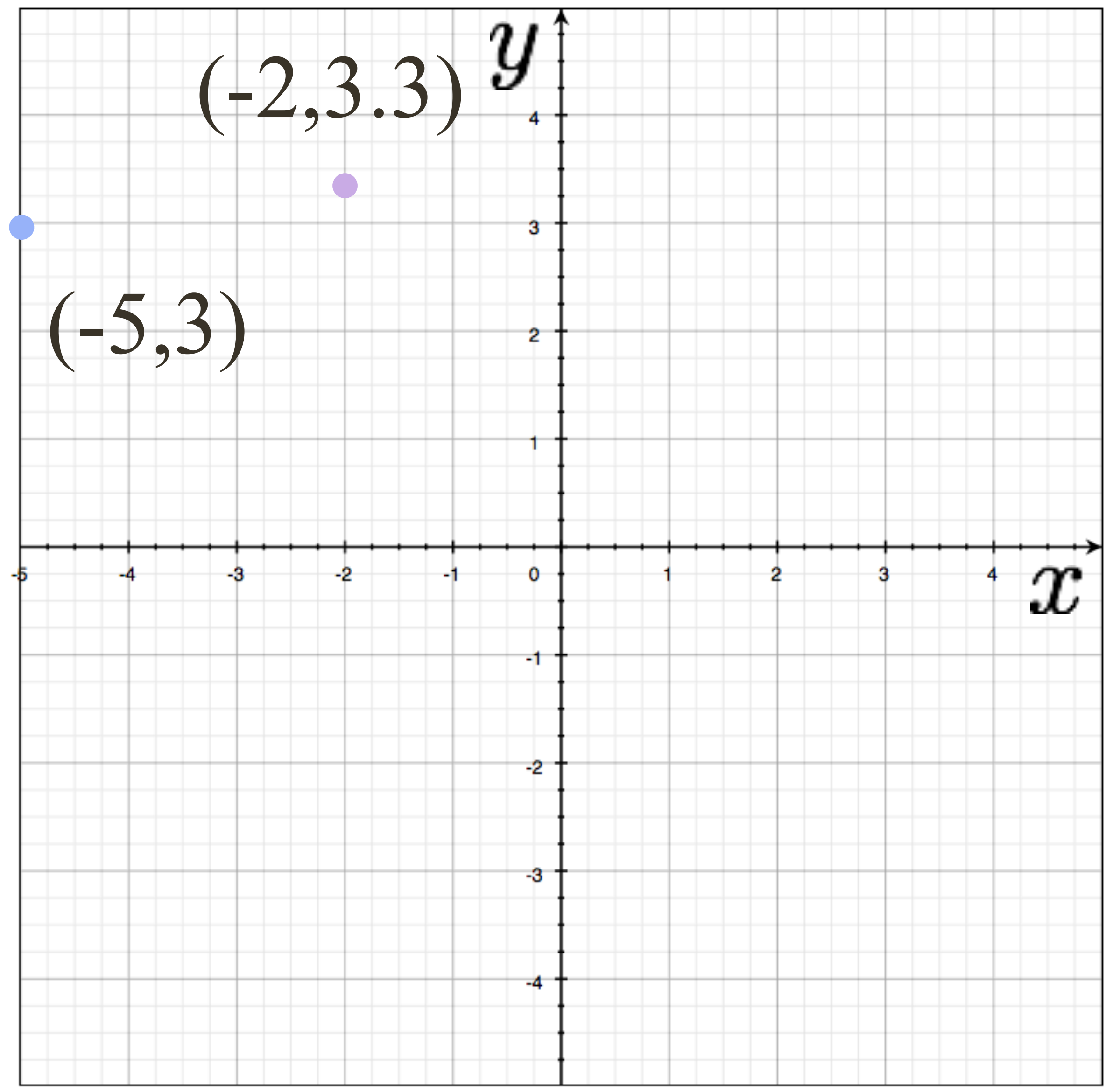
Example: Hough Transform for Lines

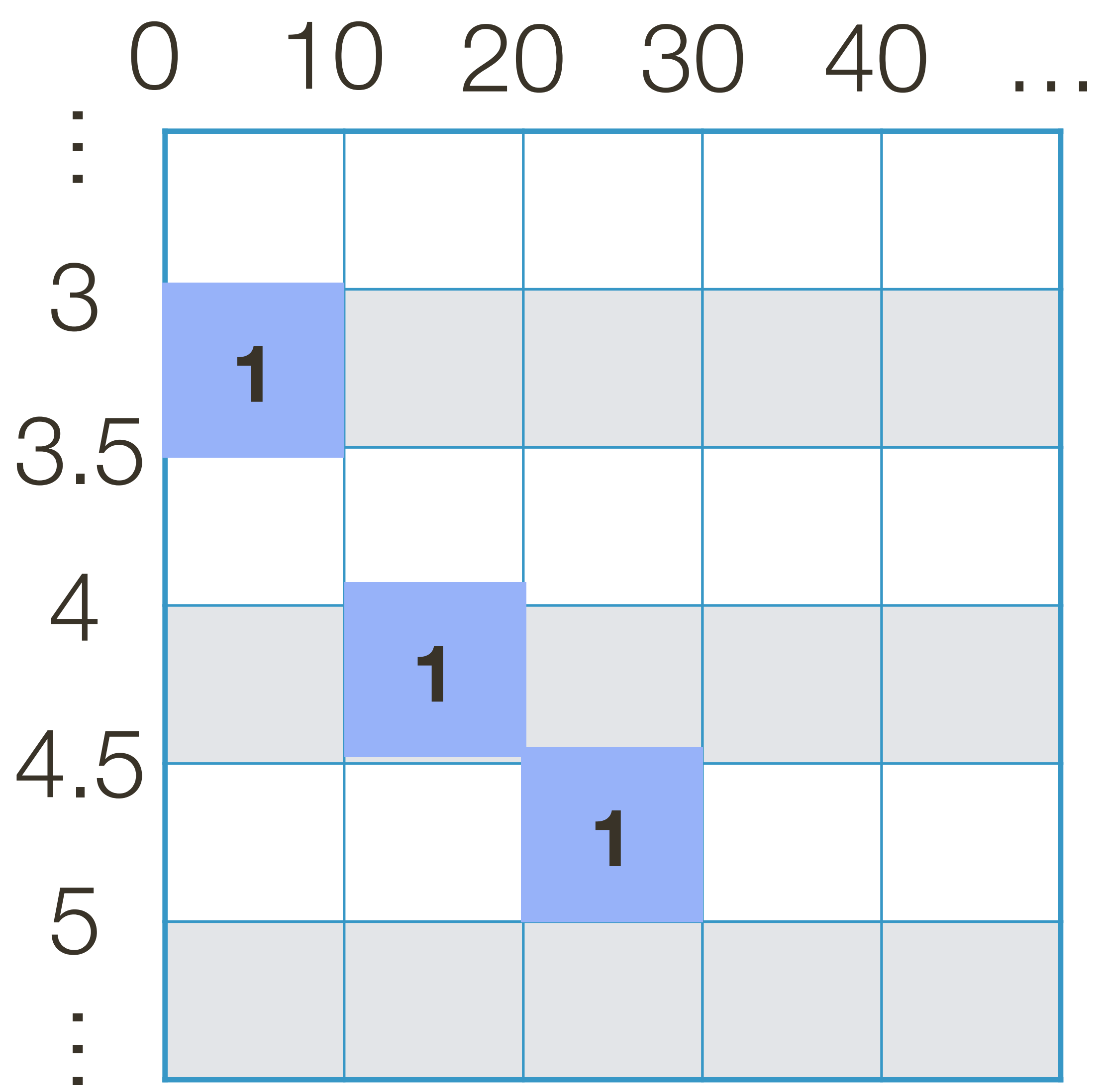




$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$
$$-5\sin(15^\circ) - 3\cos(15^\circ) + r = 0 => r = 4.18$$
$$-5\sin(25^\circ) - 3\cos(25^\circ) + r = 0 => r = 4.83$$

Example: Hough Transform for Lines



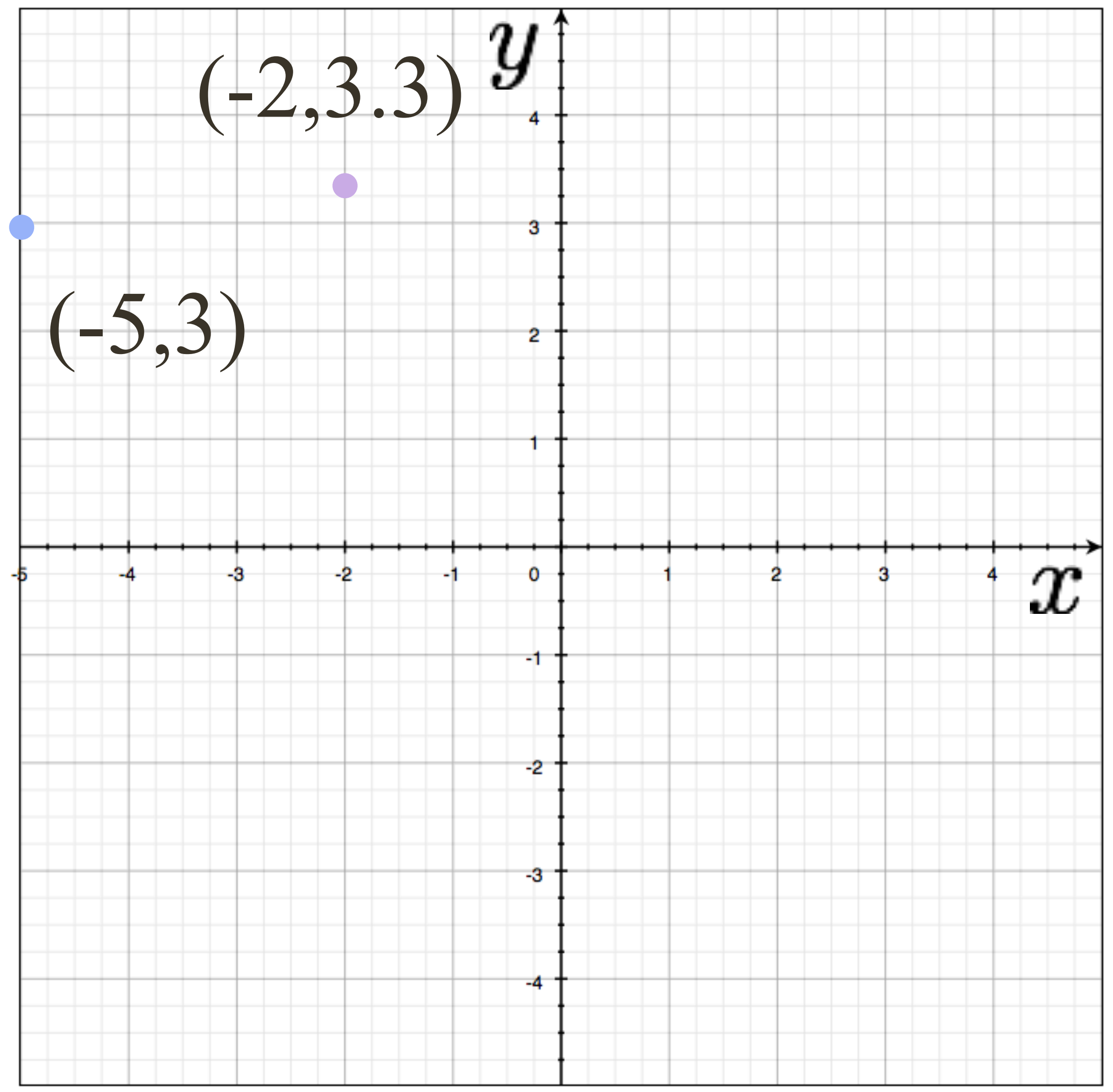


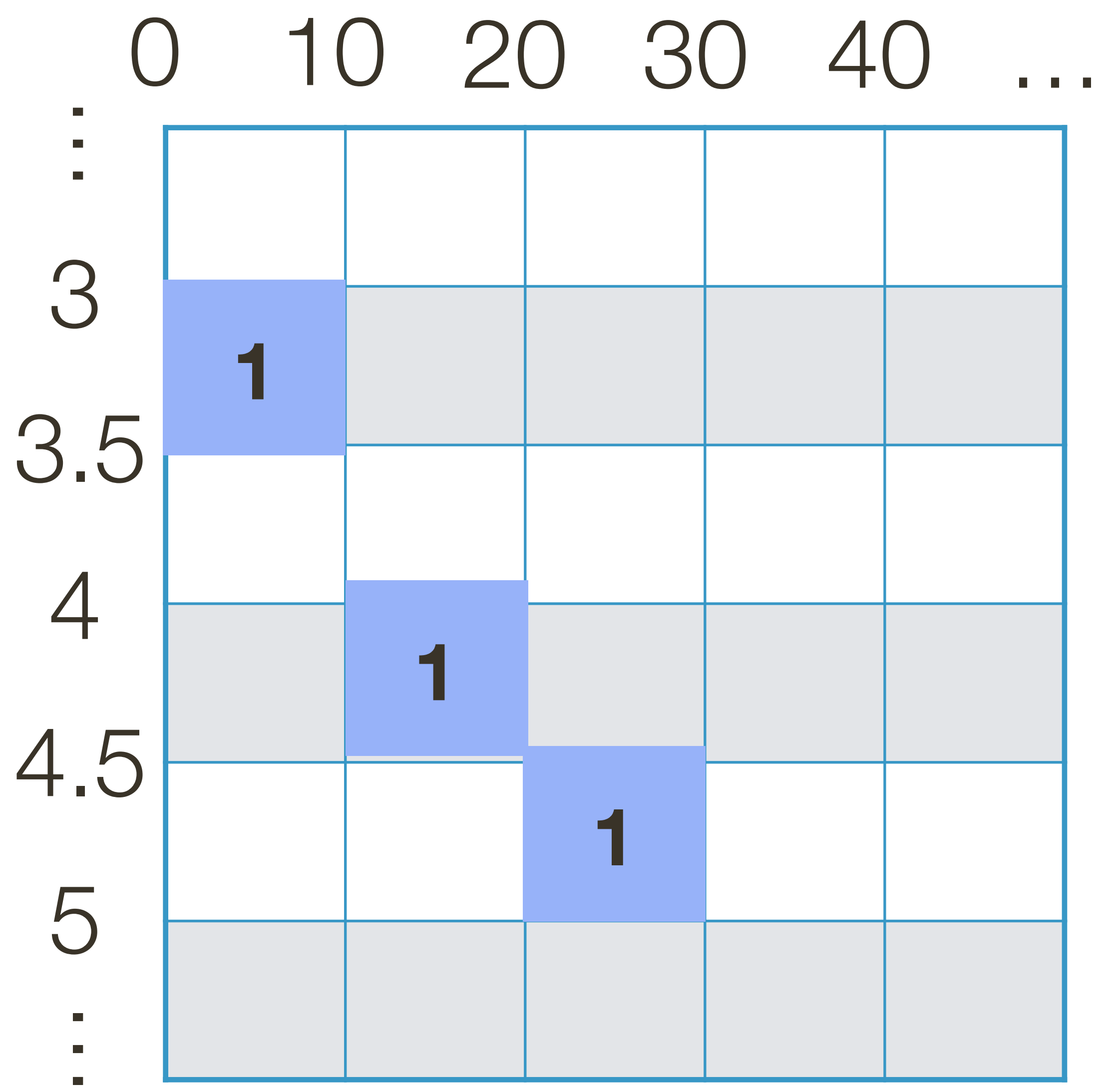
$$-5\sin(5°) - 3\cos(5°) + r = 0 => r = 3.42$$

$$-5\sin(15°) - 3\cos(15°) + r = 0 => r = 4.18$$

$$-5\sin(25°) - 3\cos(25°) + r = 0 => r = 4.83$$

Example: Hough Transform for Lines





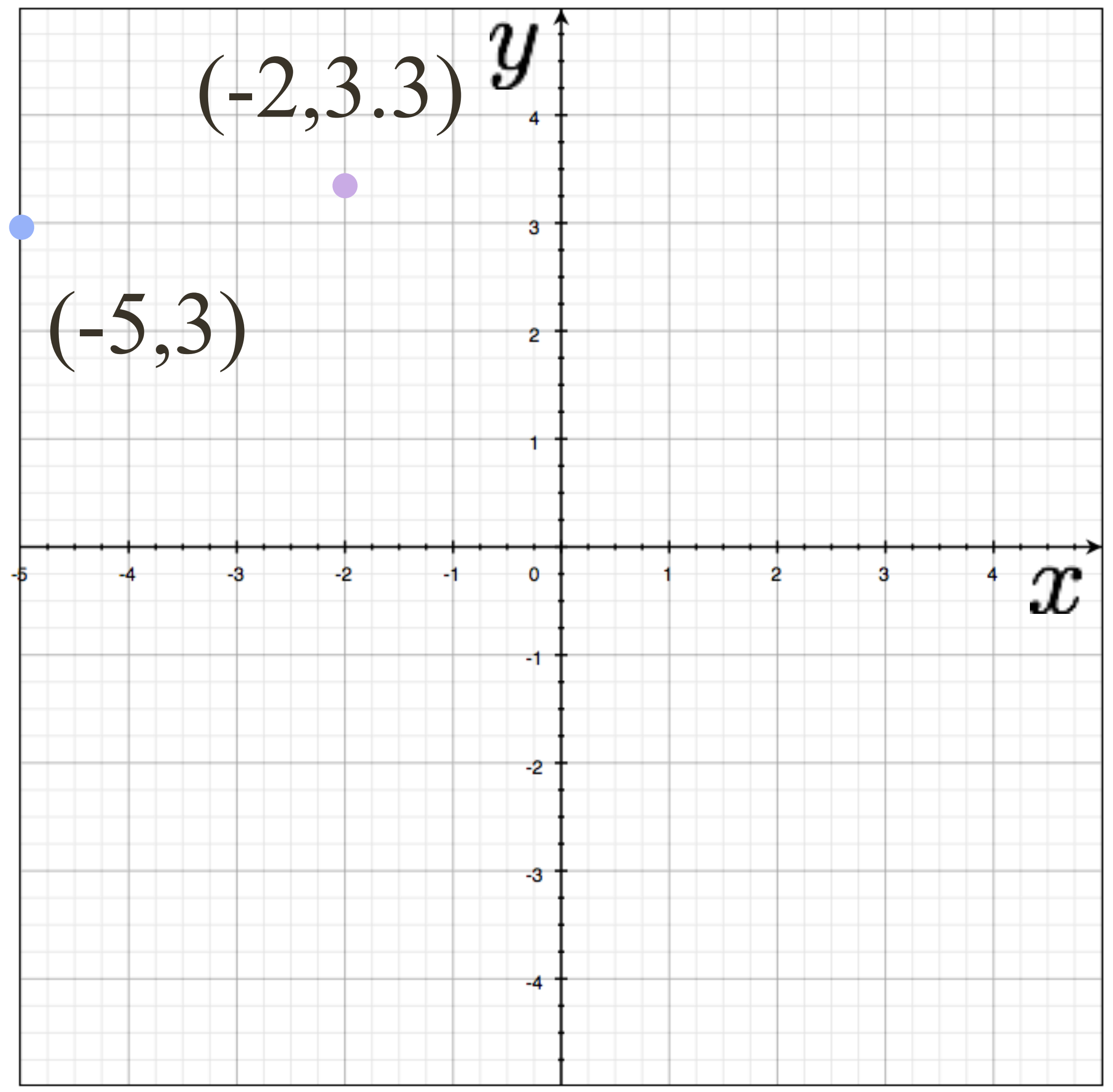
$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$

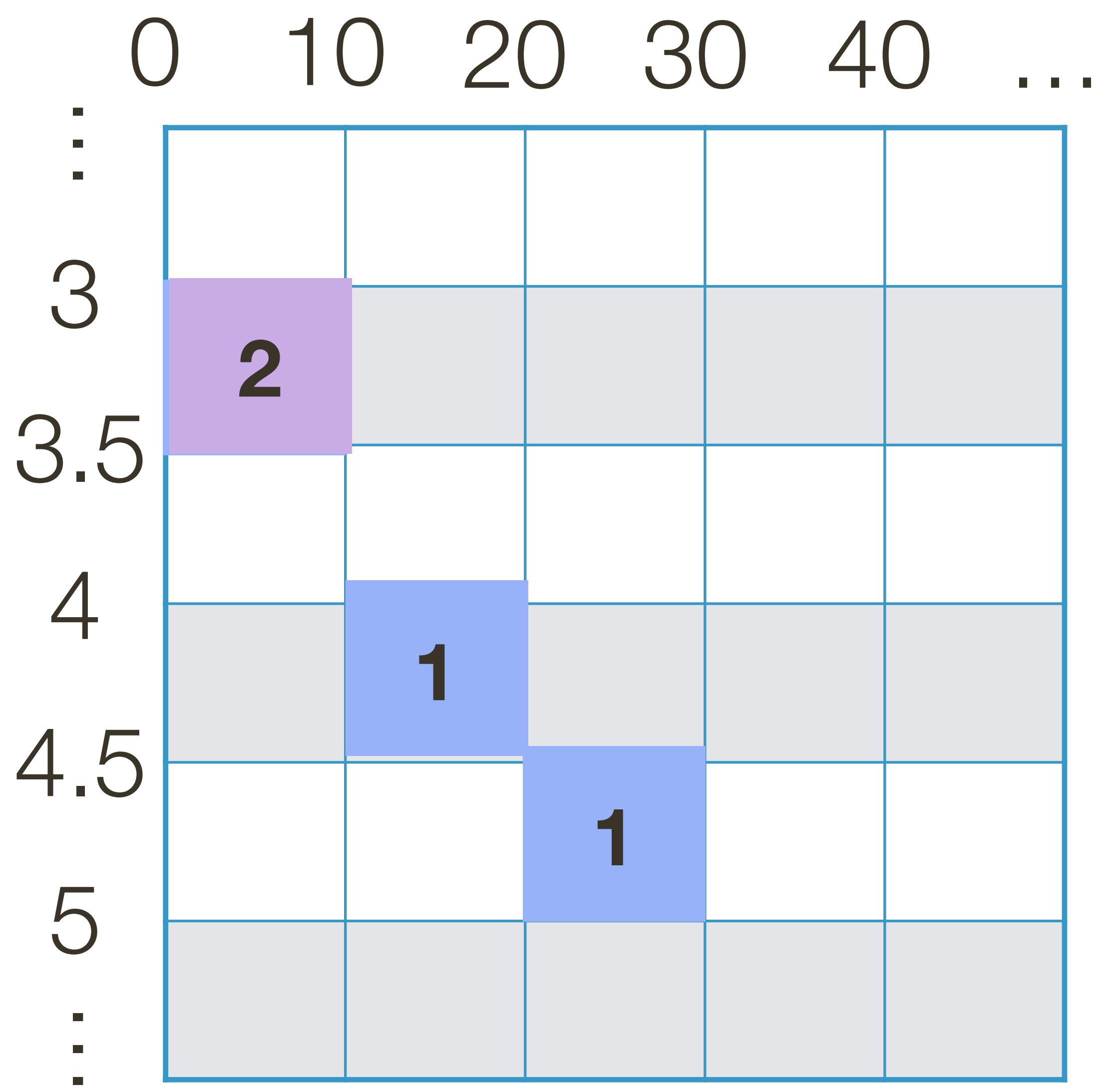
$$-5\sin(15^\circ) - 3\cos(15^\circ) + r = 0 => r = 4.18$$

$$-5\sin(25^\circ) - 3\cos(25^\circ) + r = 0 => r = 4.83$$

$$-2\sin(5^\circ) - 3.3\cos(5^\circ) + r = 0 => r = 3.46$$

Example: Hough Transform for Lines





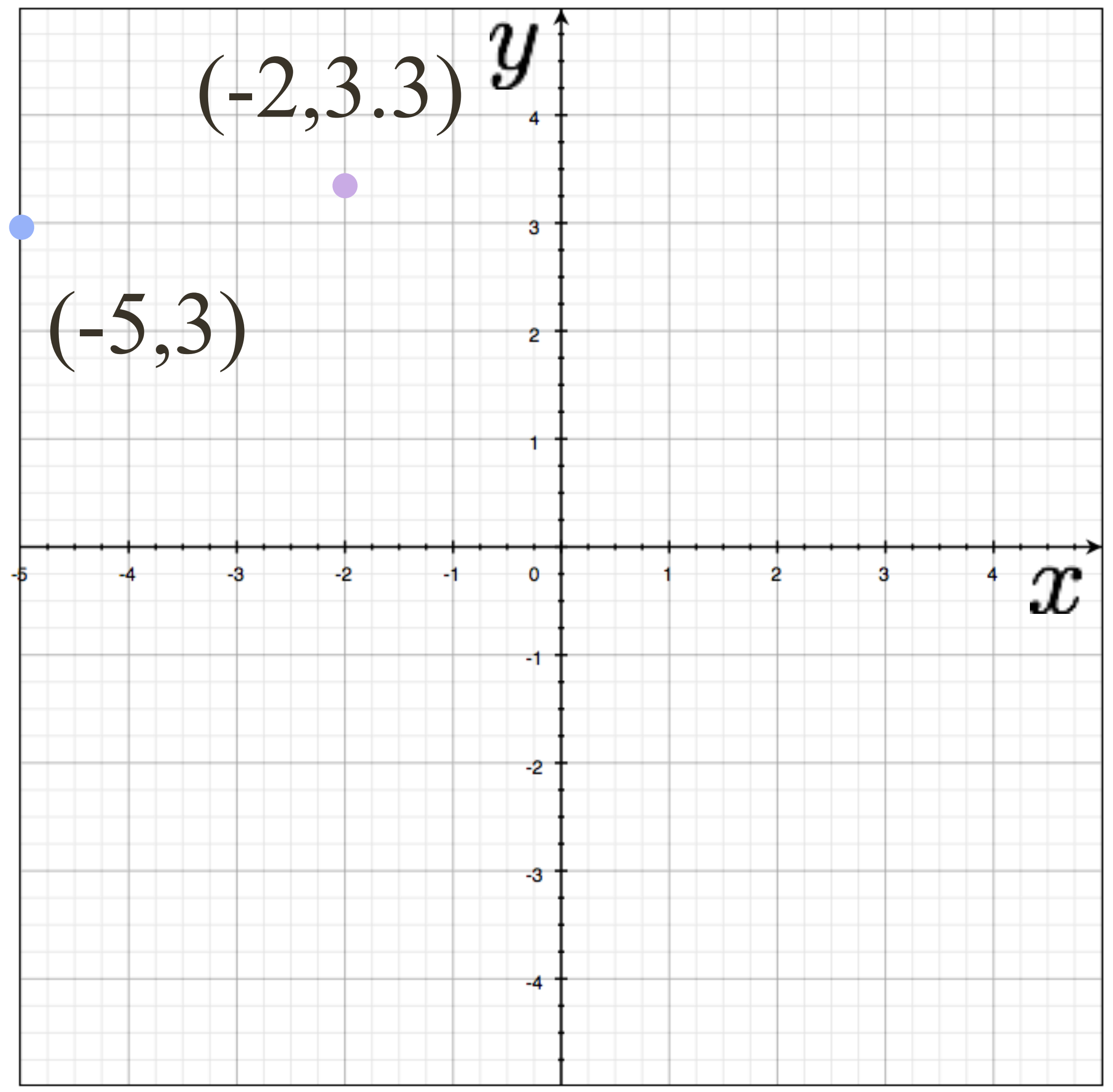
$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$

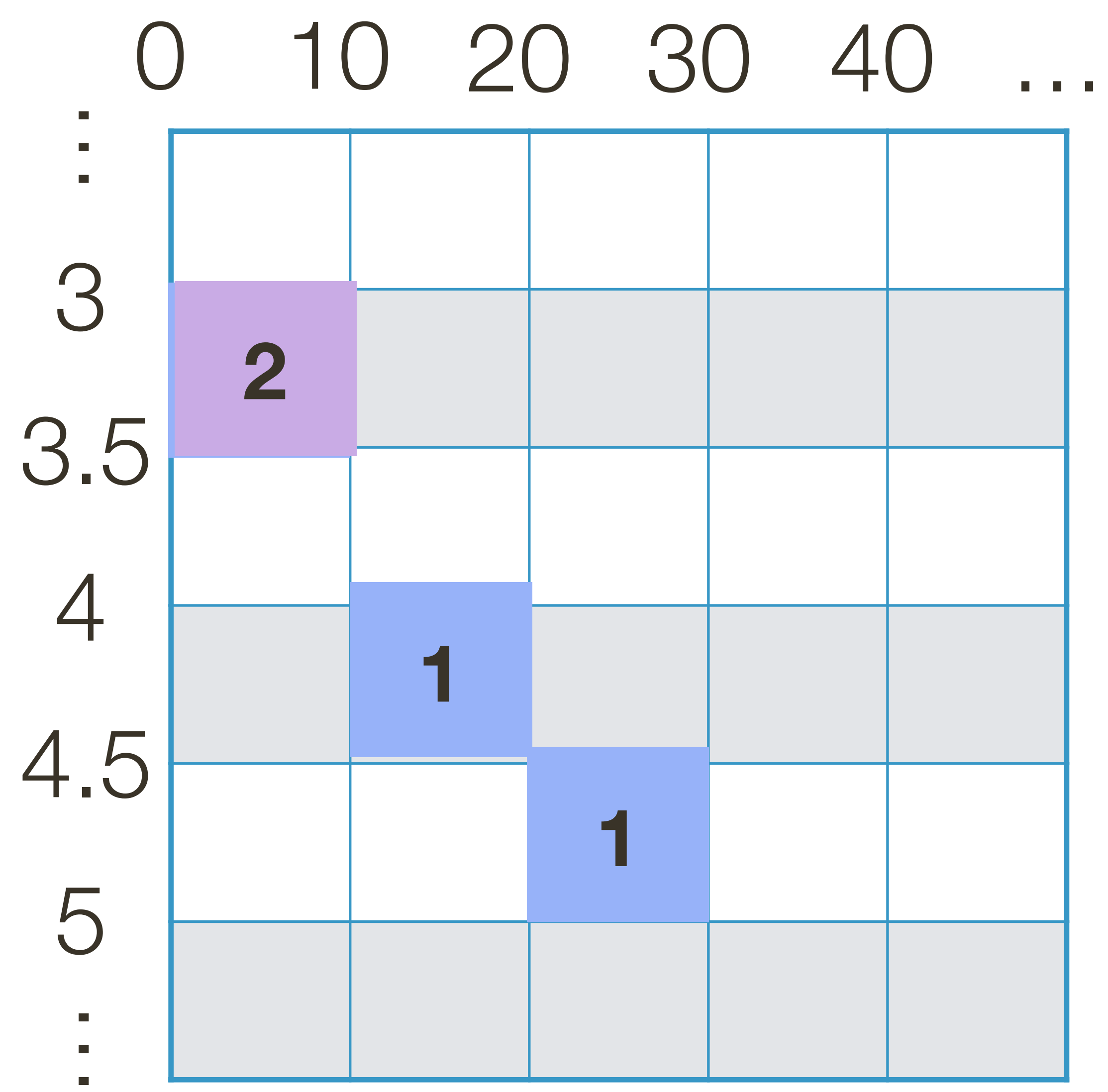
$$-5\sin(15^\circ) - 3\cos(15^\circ) + r = 0 => r = 4.18$$

$$-5\sin(25^\circ) - 3\cos(25^\circ) + r = 0 => r = 4.83$$

$$-2\sin(5^\circ) - 3.3\cos(5^\circ) + r = 0 => r = 3.46$$

Example: Hough Transform for Lines

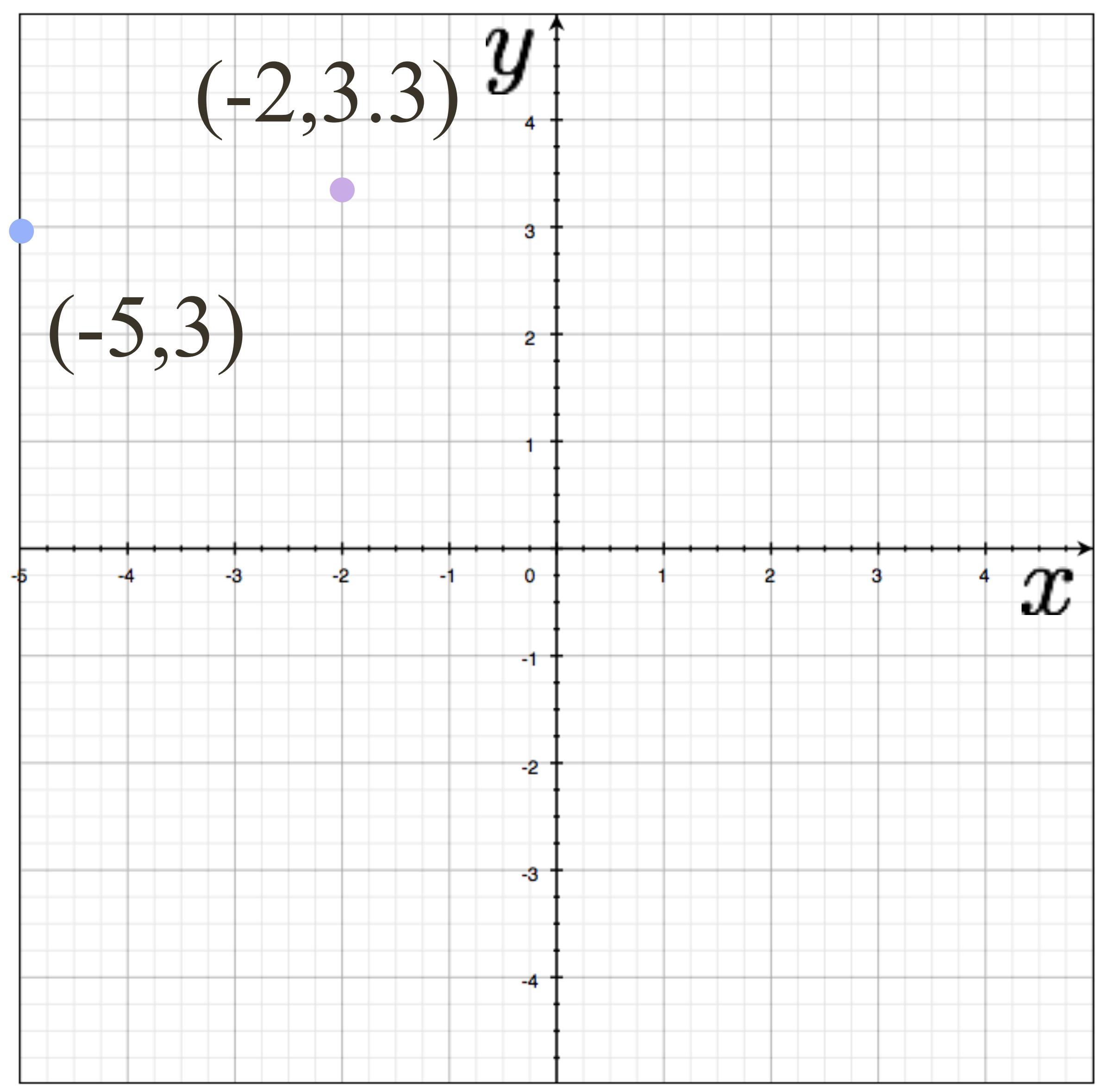


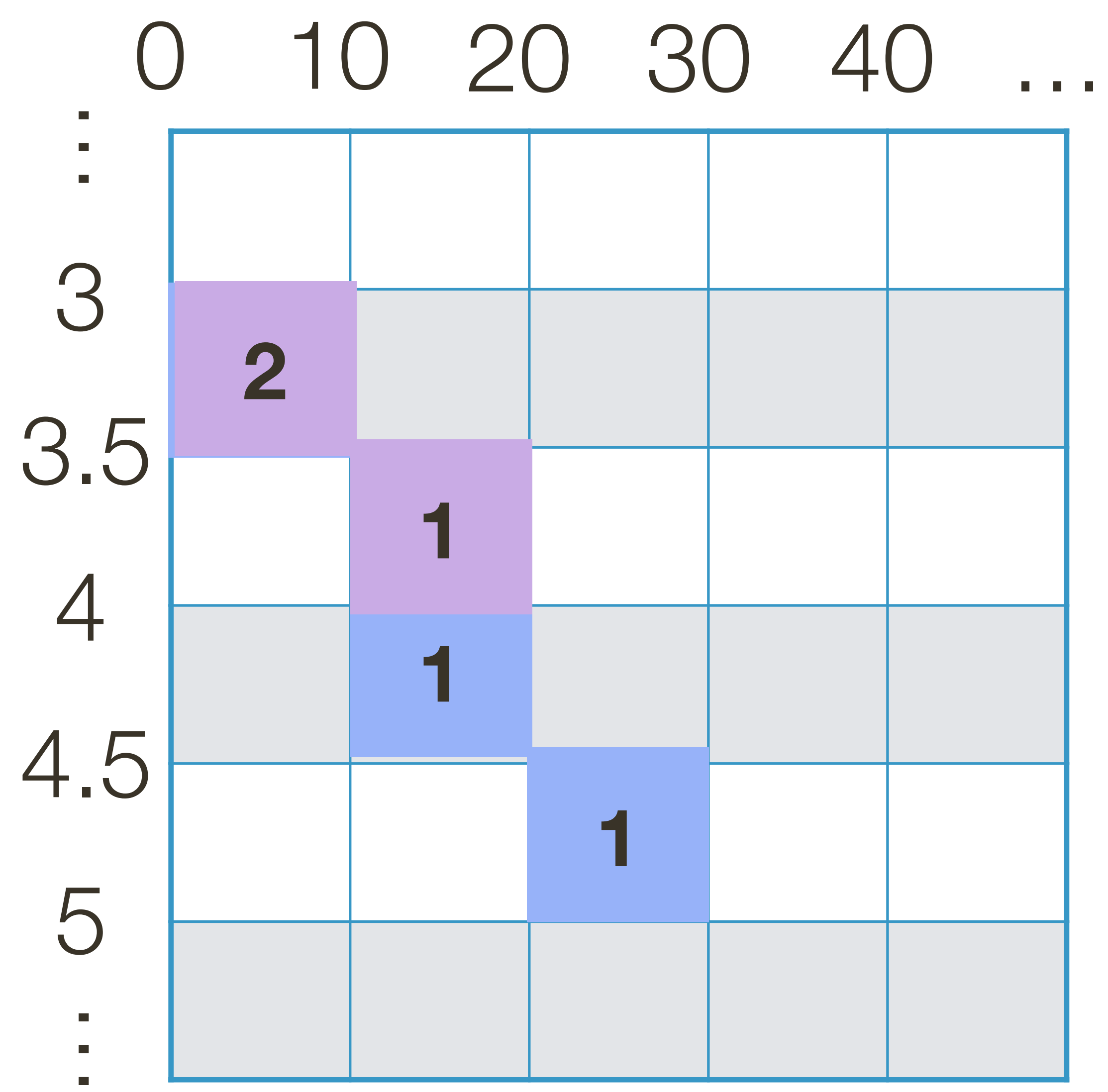


$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$
$$-5\sin(15^\circ) - 3\cos(15^\circ) + r = 0 => r = 4.18$$
$$-5\sin(25^\circ) - 3\cos(25^\circ) + r = 0 => r = 4.83$$

$$-2\sin(5^\circ) - 3.3\cos(5^\circ) + r = 0 => r = 3.46$$
$$-2\sin(15^\circ) - 3.3\cos(15^\circ) + r = 0 => r = 3.71$$

Example: Hough Transform for Lines





$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$

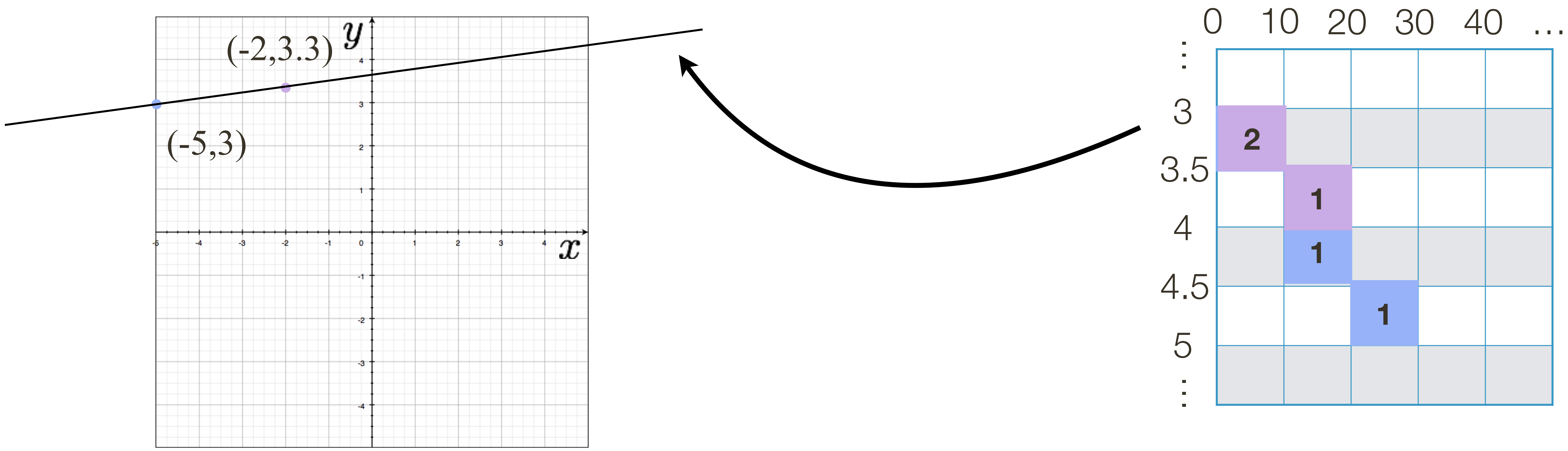
$$-5\sin(15^\circ) - 3\cos(15^\circ) + r = 0 => r = 4.18$$

$$-5\sin(25^\circ) - 3\cos(25^\circ) + r = 0 => r = 4.83$$

$$-2\sin(5^\circ) - 3.3\cos(5^\circ) + r = 0 => r = 3.46$$

$$-2\sin(15^\circ) - 3.3\cos(15^\circ) + r = 0 => r = 3.71$$

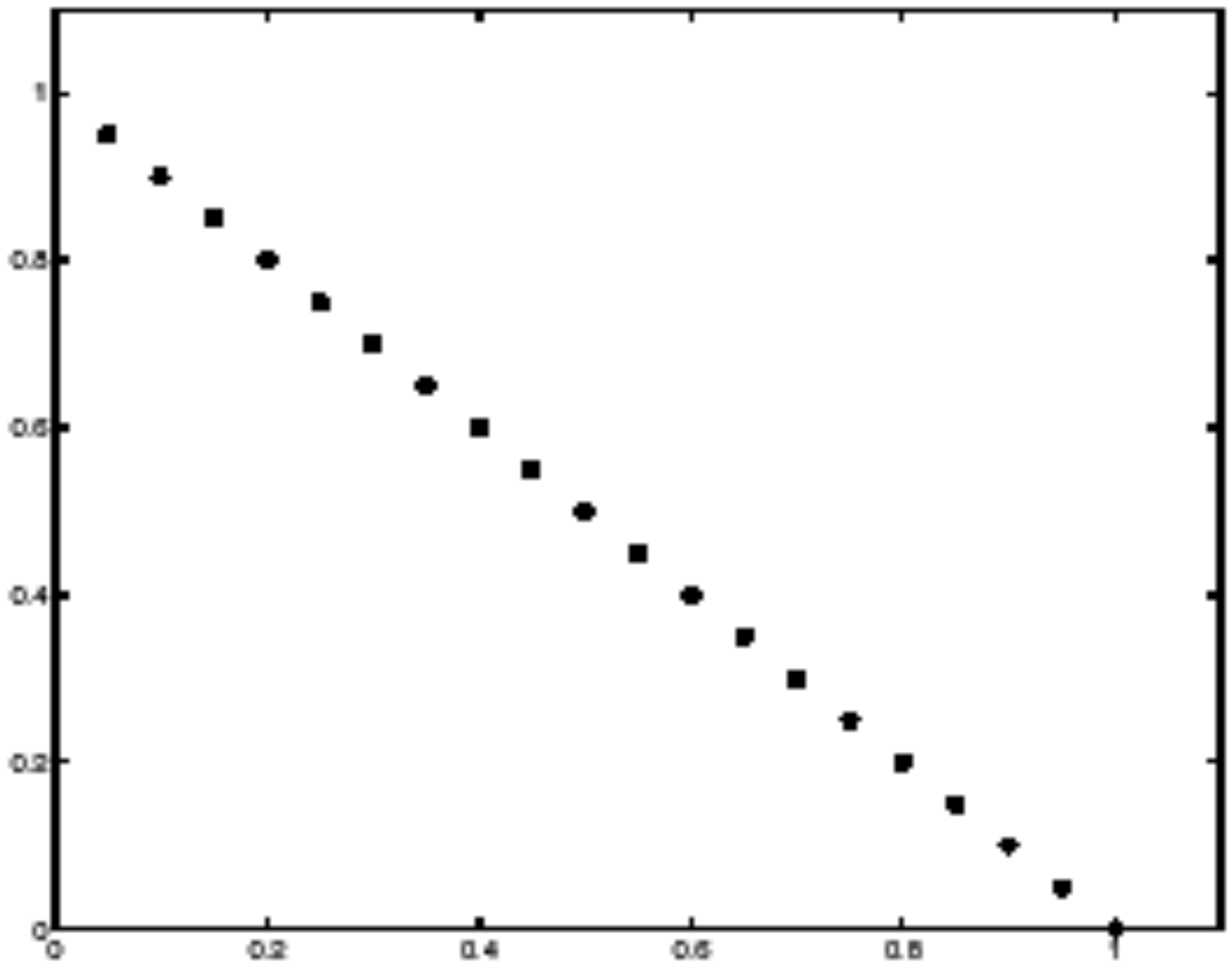
Example: Hough Transform for Lines



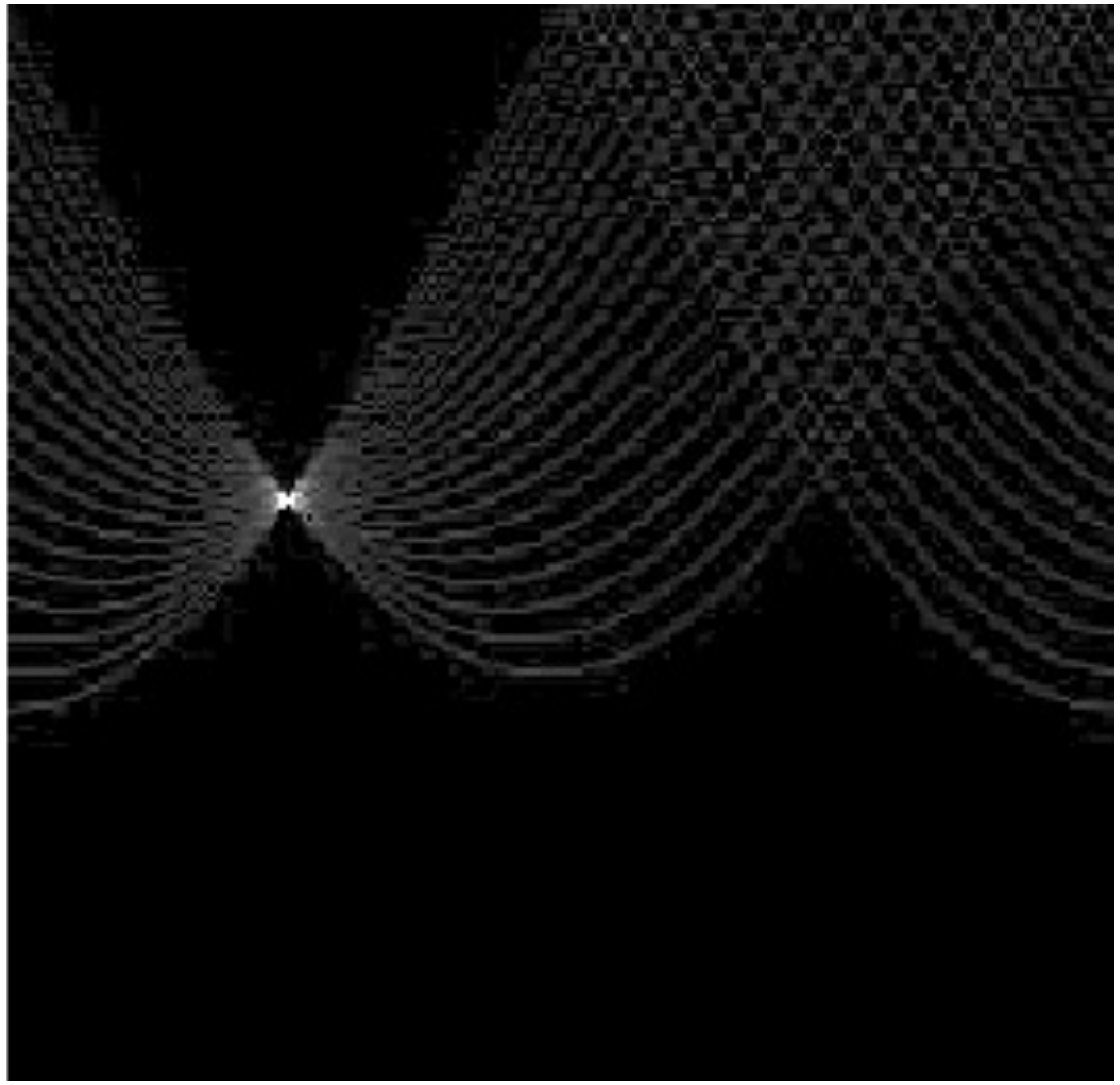
$$-5\sin(5^\circ) - 3\cos(5^\circ) + r = 0 => r = 3.42$$
$$-5\sin(15^\circ) - 3\cos(15^\circ) + r = 0 => r = 4.18$$
$$-5\sin(25^\circ) - 3\cos(25^\circ) + r = 0 => r = 4.83$$

$$-2\sin(5^\circ) - 3.3\cos(5^\circ) + r = 0 => r = 3.46$$
$$-2\sin(15^\circ) - 3.3\cos(15^\circ) + r = 0 => r = 3.71$$

Example: Clean Data



Tokens



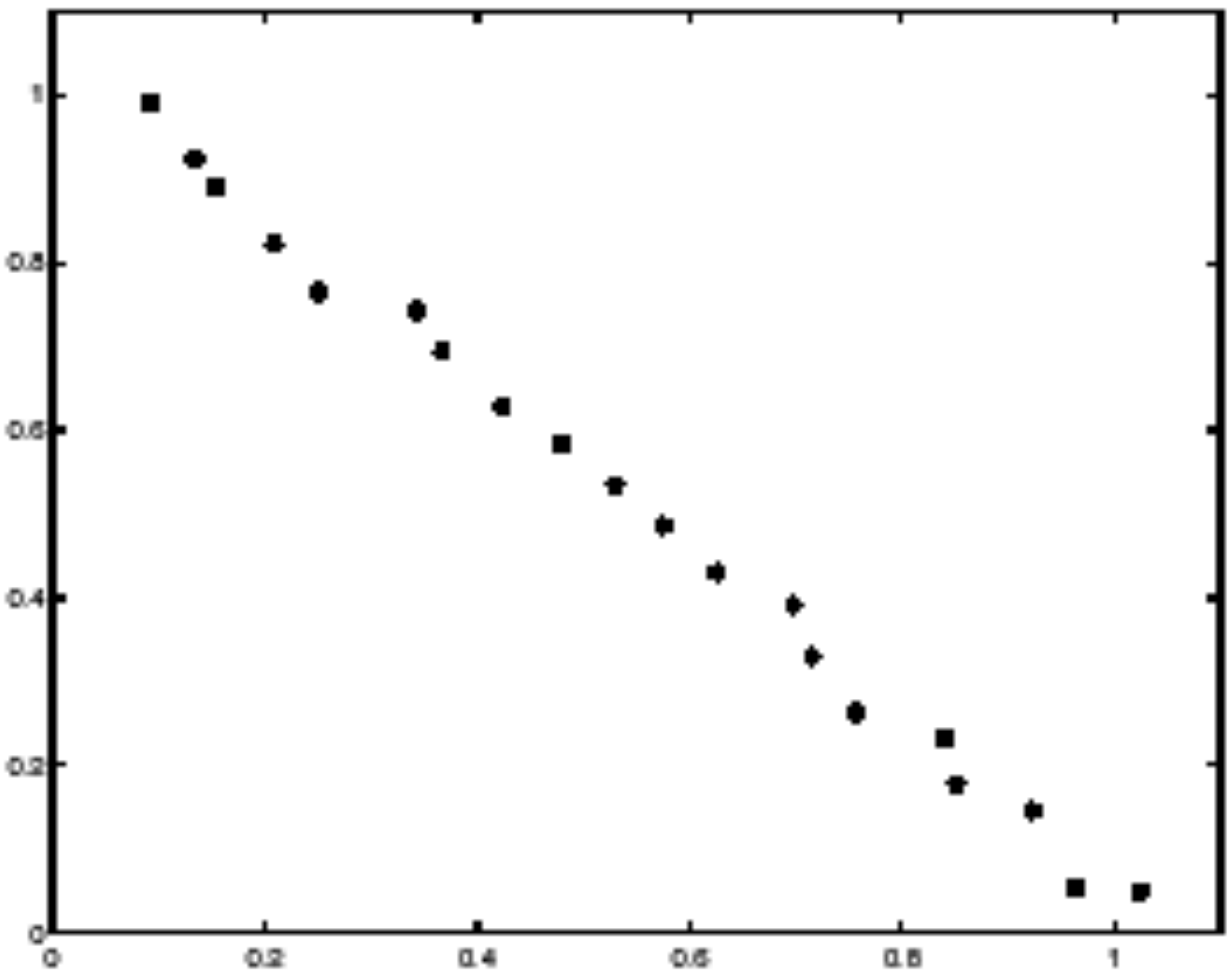
Votes

Horizontal axis is θ

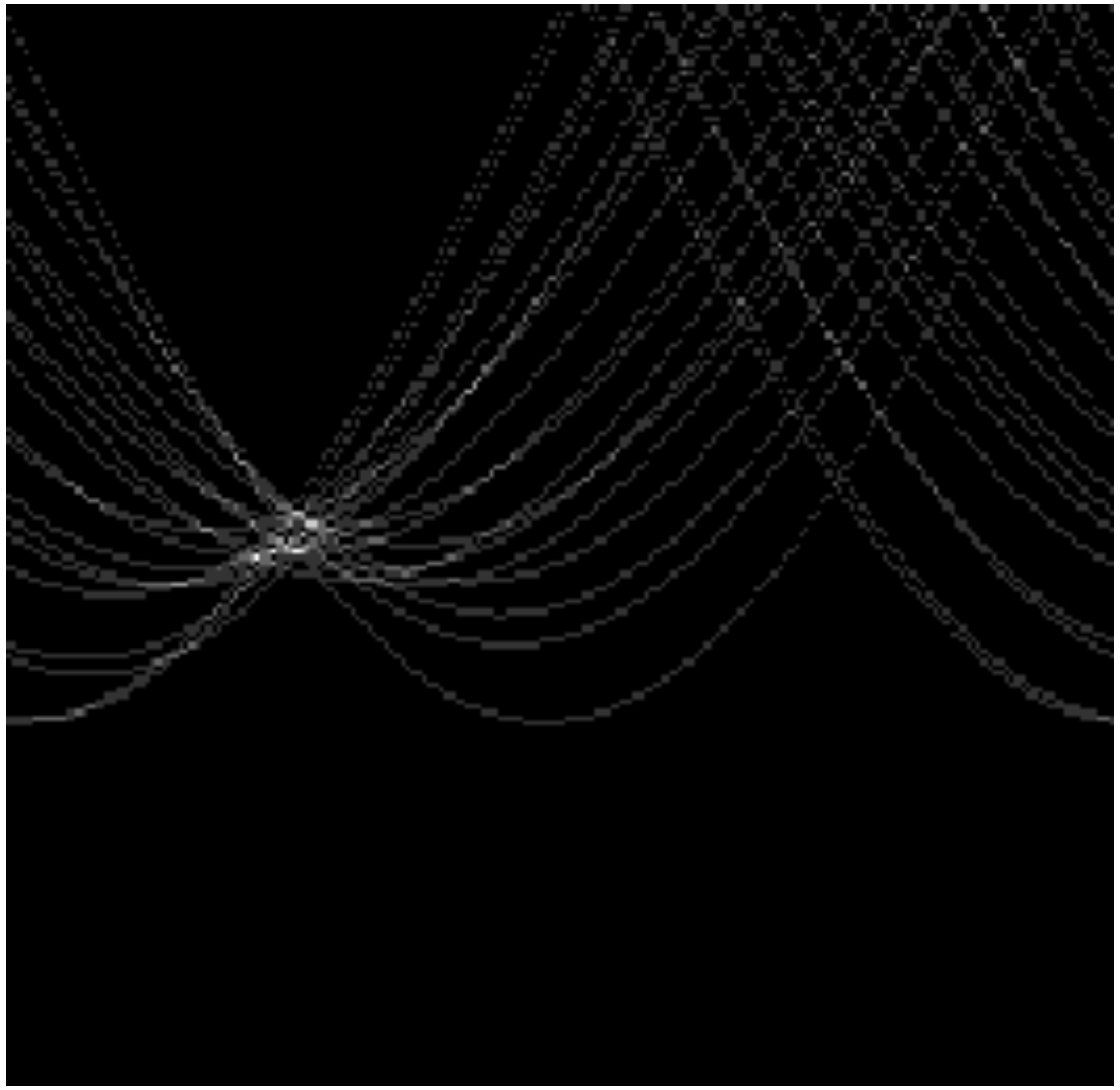
Vertical Axis is r

Forsyth & Ponce (2nd ed.) Figure 10.1 (Top)

Example: Some Noise



Tokens



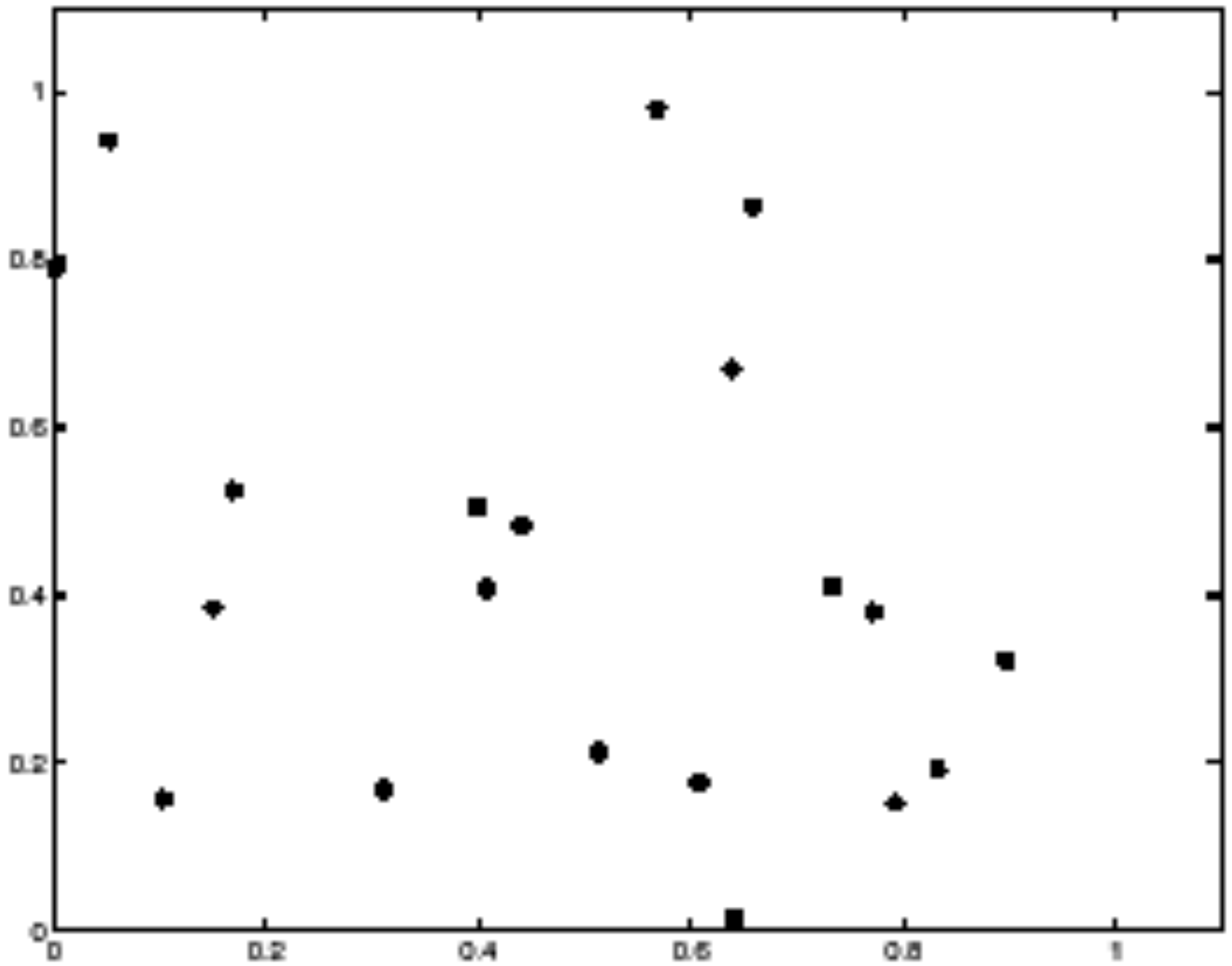
Votes

Horizontal axis is θ

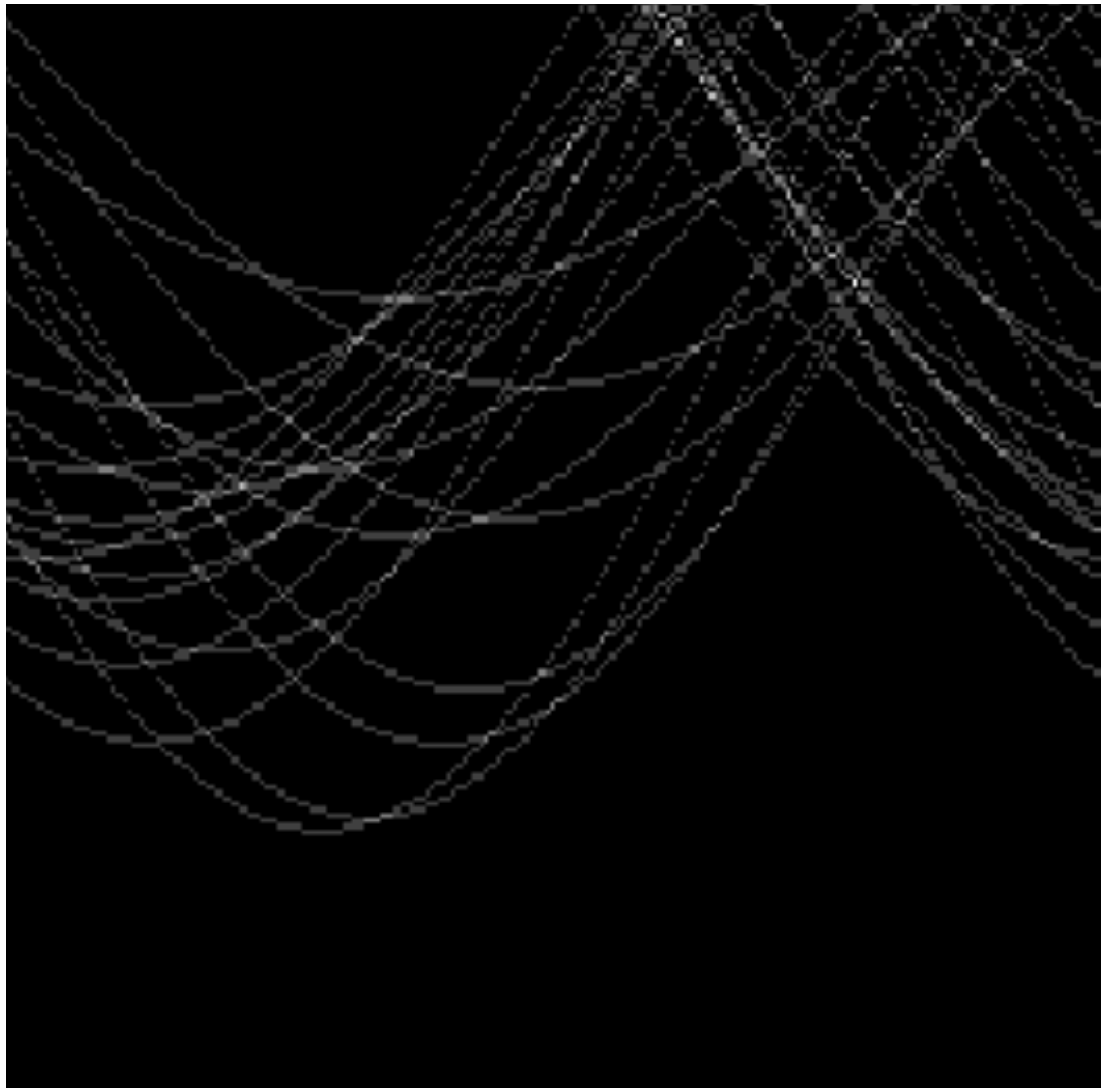
Vertical Axis is r

Forsyth & Ponce (2nd ed.) Figure 10.1 (Bottom)

Example: Too Much Noise



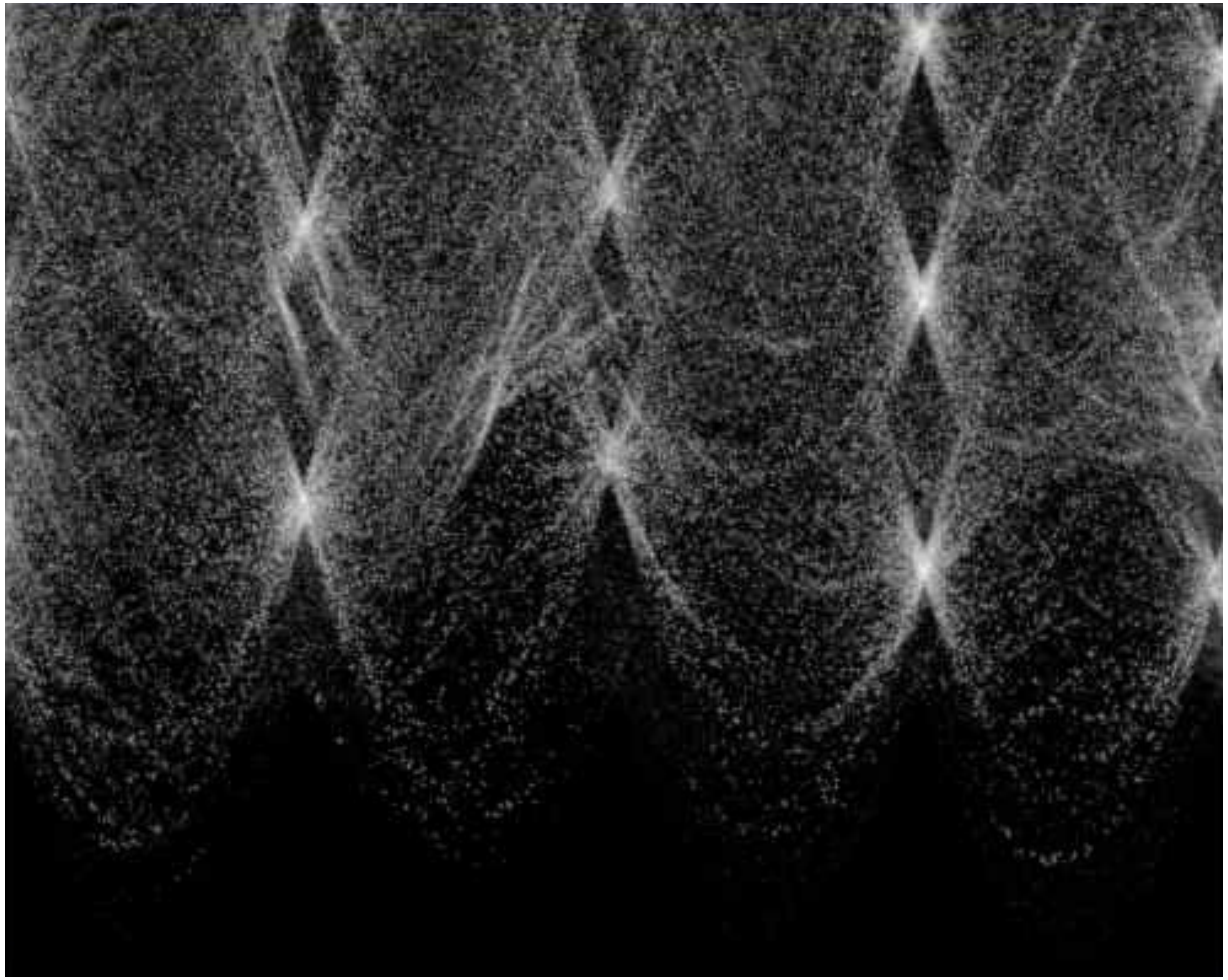
Tokens



Votes
Horizontal axis is θ
Vertical Axis is r

Forsyth & Ponce (2nd ed.) Figure 10.2

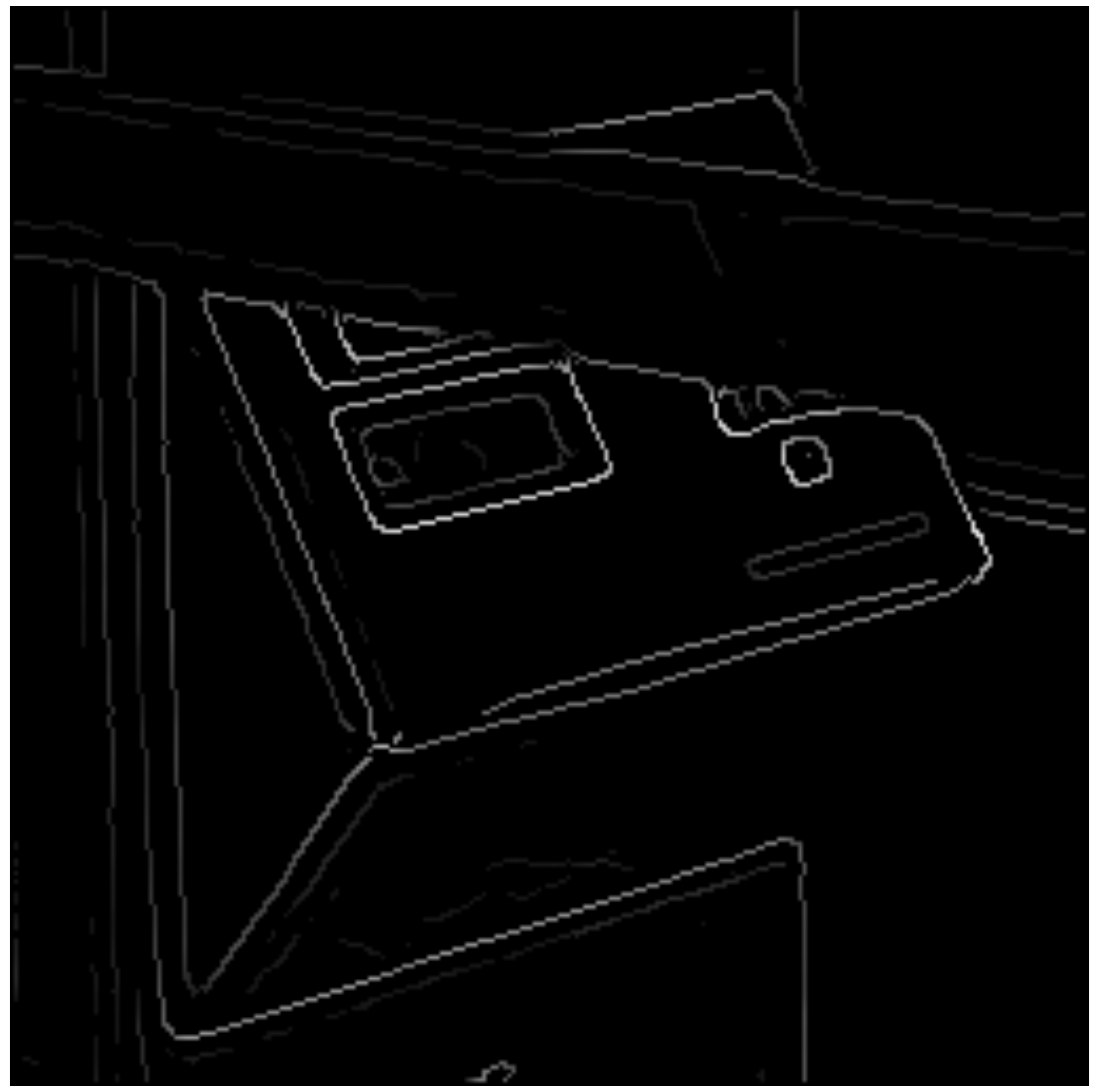
Real World **Example**



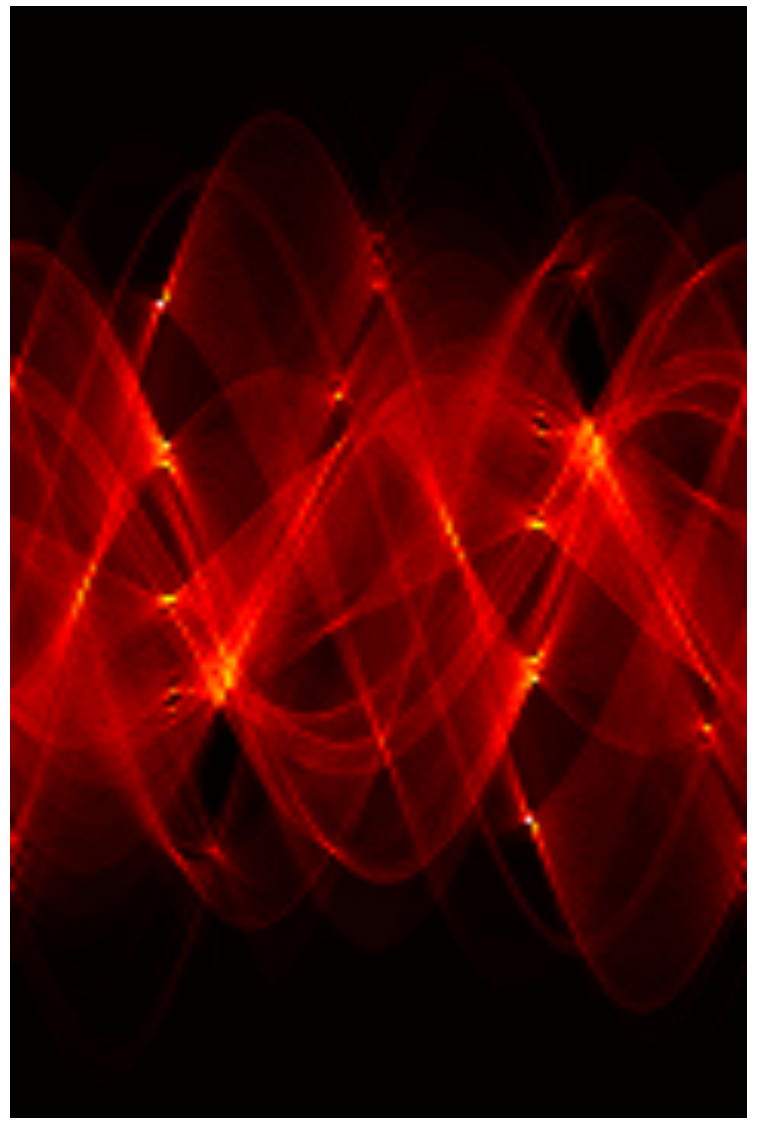
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Real World **Example**

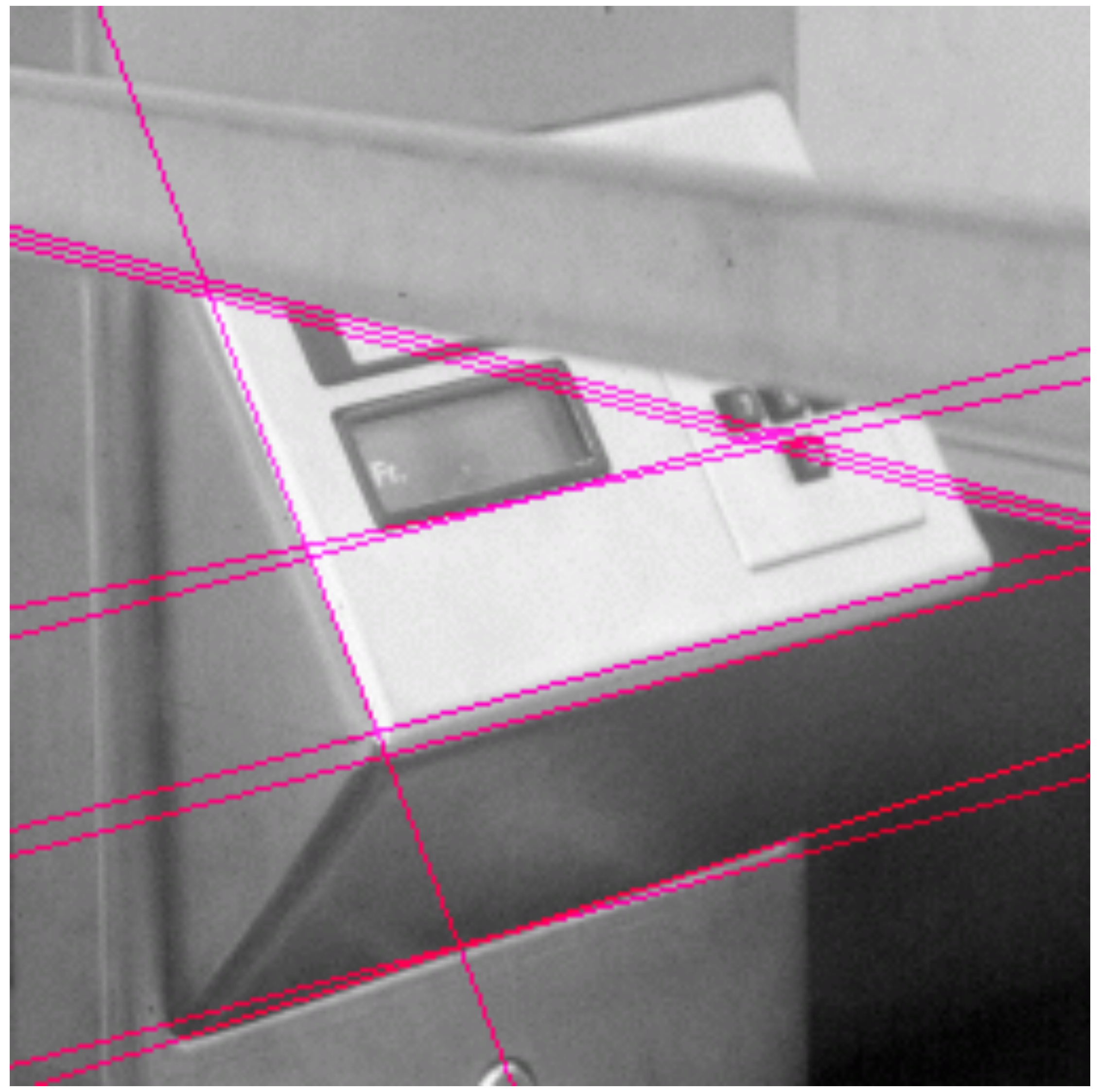
Original



Edges



Parameter space



Hough Lines

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)