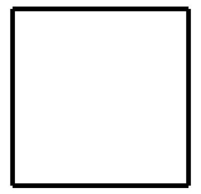
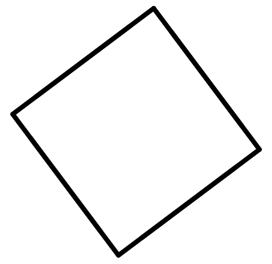
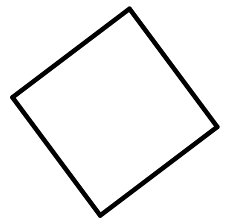
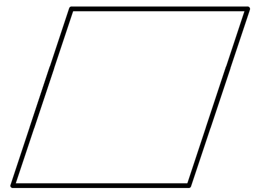
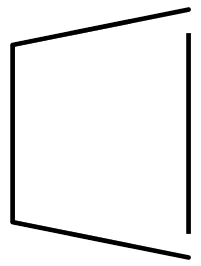
2D **Transformations**

Transformation	**Matrix**	**# DoF**	**Preserves**	**Icon**
translation	$\left[\begin{array}{c\|c} \boldsymbol{I} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	2	orientation	
rigid (Euclidean)	$\left[\begin{array}{c\|c} \boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	3	lengths	
similarity	$\left[\begin{array}{c\|c} s\boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	4	angles	
affine	$\left[\begin{array}{c} \boldsymbol{A} \end{array}\right]_{2\times 3}$	6	parallelism	
projective	$\left[\begin{array}{c} \tilde{\boldsymbol{H}} \end{array}\right]_{3\times 3}$	8	straight lines	

Projective Transformation

General 3x3 matrix transformation

$$\begin{bmatrix} x_1' \\ y_1' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Projective Transformation

General 3x3 matrix transformation

$$\begin{bmatrix} x'_1 \\ y'_1 \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Lets try an example:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}}_{\text{Transformation}} \underbrace{\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}}_{\text{Points}} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 2 & 1 & 2 \end{bmatrix}}_{\text{Transformed Points}}$$

Projective Transformation

General 3x3 matrix transformation

$$\begin{bmatrix} x'_1 \\ y'_1 \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Lets try an example:

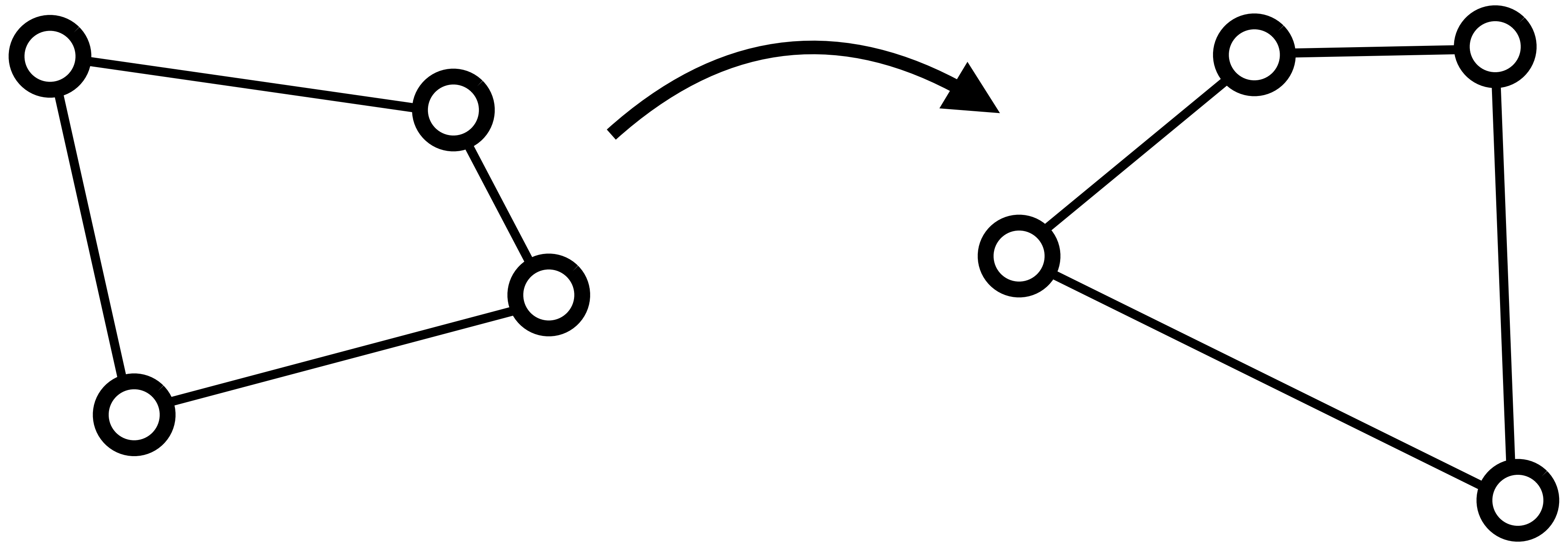
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}}_{\text{Transformation}} \underbrace{\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}}_{\text{Points}} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 2 & 1 & 2 \end{bmatrix}}_{\text{Transformed Points}}$$

Divide by the last row: $\begin{bmatrix} 0 & 0 & 1 & 0.5 \\ 0 & 0.5 & 0 & 0.5 \\ 1 & 1 & 1 & 1 \end{bmatrix}$

Compute **H** from Correspondences

Each match gives 2 equations to solve for **8** parameters

$$\begin{bmatrix} x'_1 \\ y'_1 \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$



→ 4 correspondences to solve for **H** matrix

Solution uses **Singular Value Decomposition** (SVD)

In **Assignment 4** you can compute this using `cv2.findHomography`

Example 1: Fitting a Line

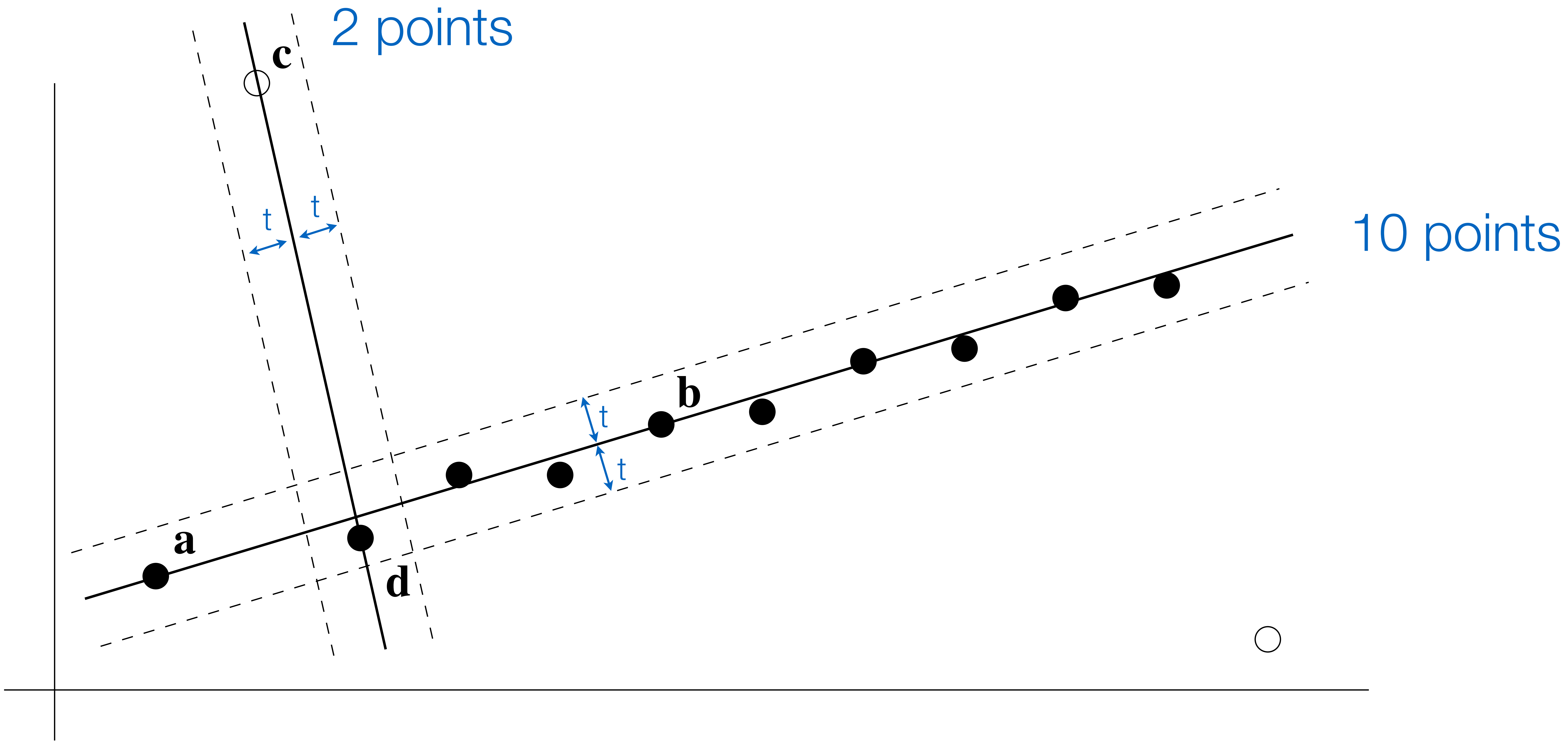
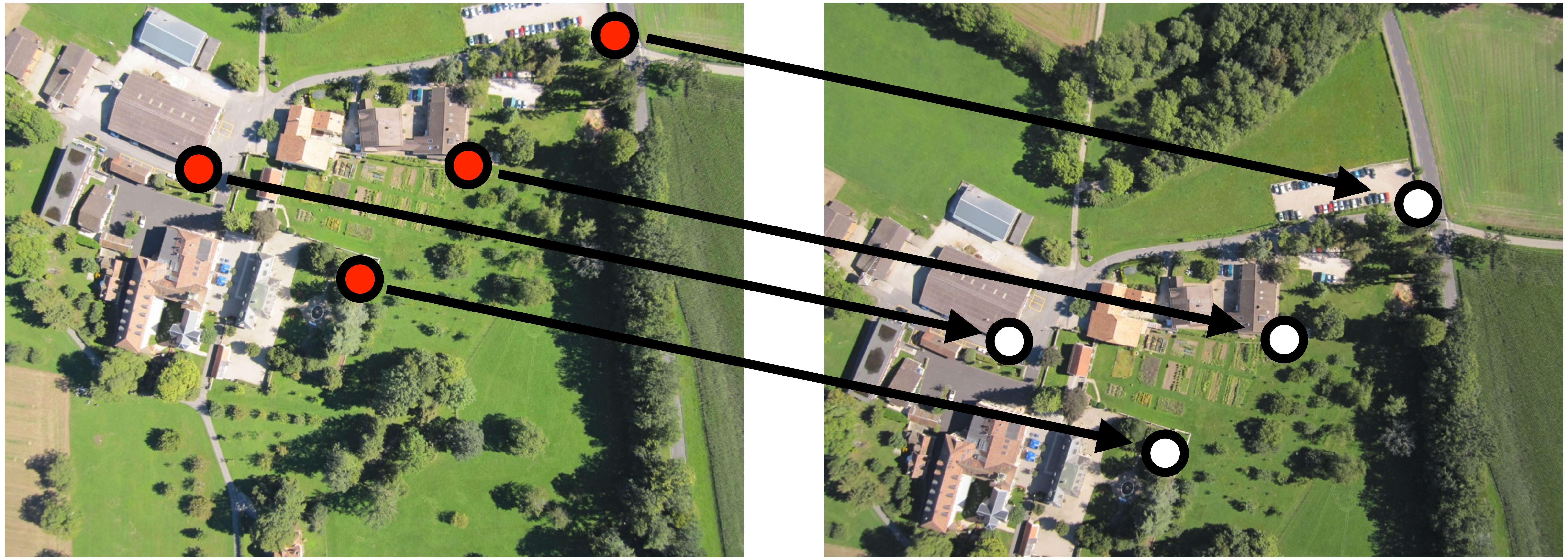


Figure Credit: Hartley & Zisserman

Image **Alignment**

Find **corresponding** (matching) points between the image



$$\mathbf{u} = \mathbf{H}\mathbf{x}$$

2 points for Similarity

3 for Affine

4 for Homography

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)

4 inliers (red, yellow, orange, brown),

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)

4 outliers (blue, light blue, purple, pink)

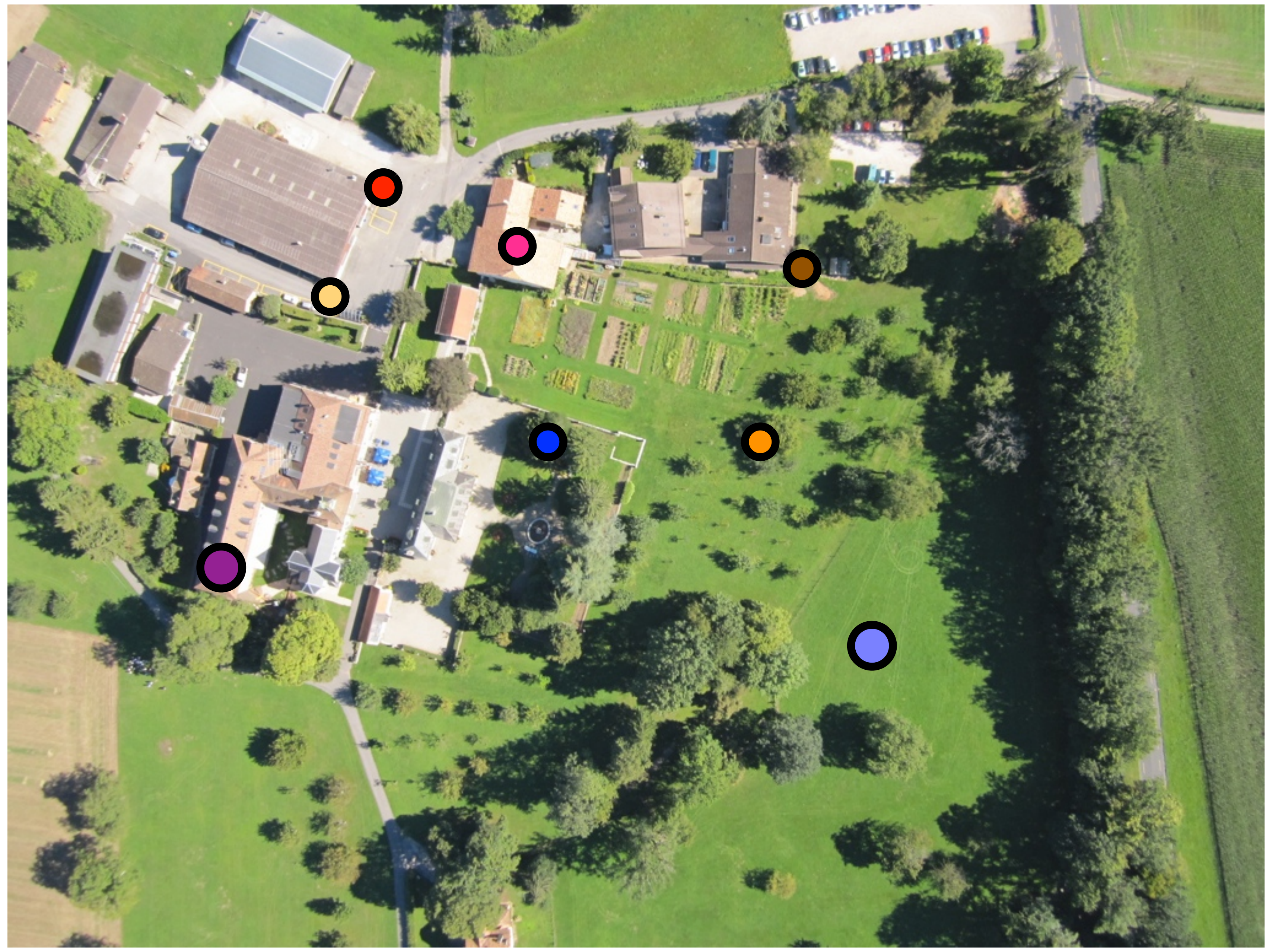
Image **Alignment + RANSAC**

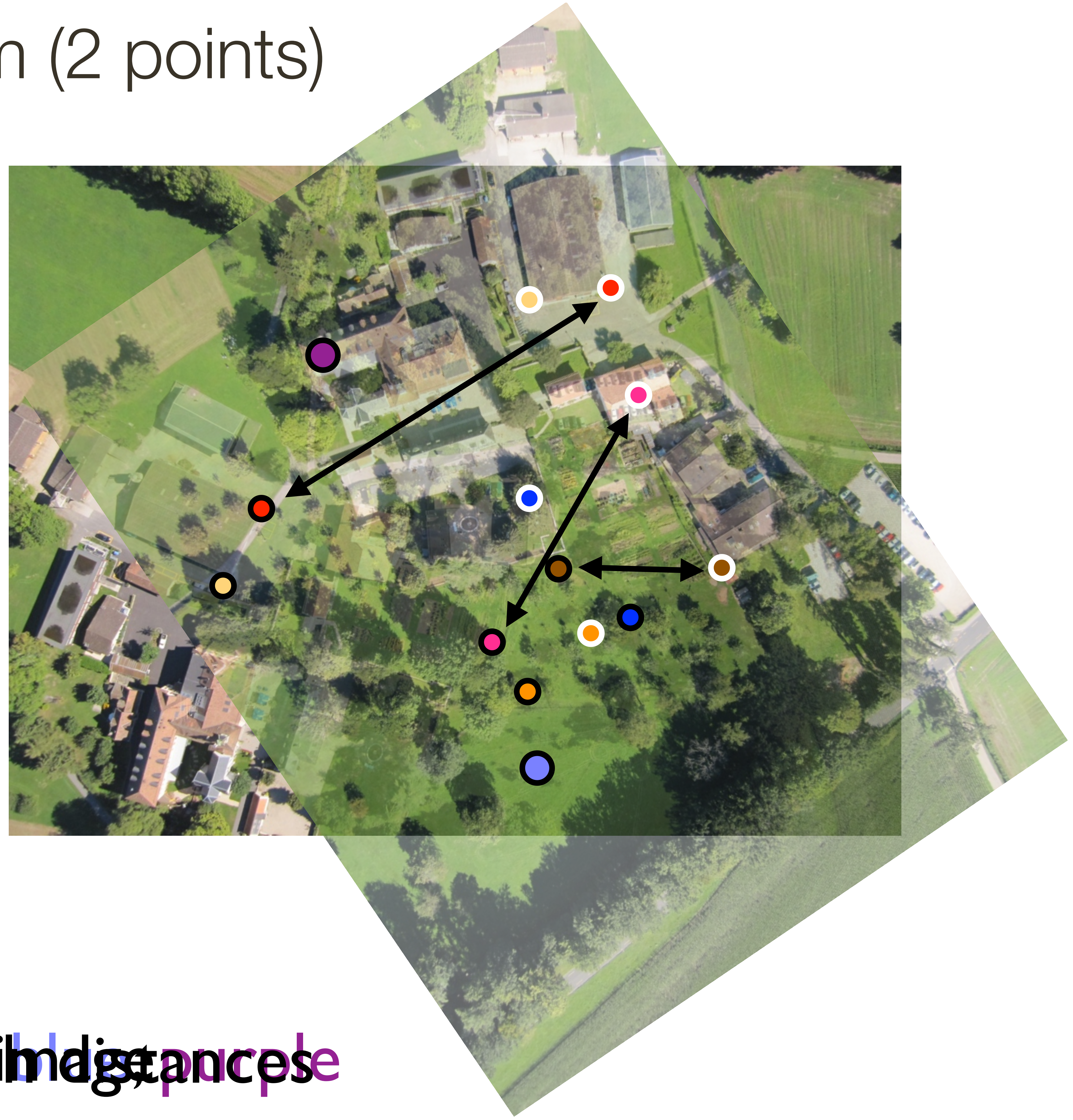
RANSAC solution for Similarity Transform (2 points)

4 inliers (red, yellow, orange, brown),
4 outliers (blue, light blue, purple, pink)

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)





check match distances

warp image, light blue, purple

#inliers = 2

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)

check distances

choose pink, blue

warp image

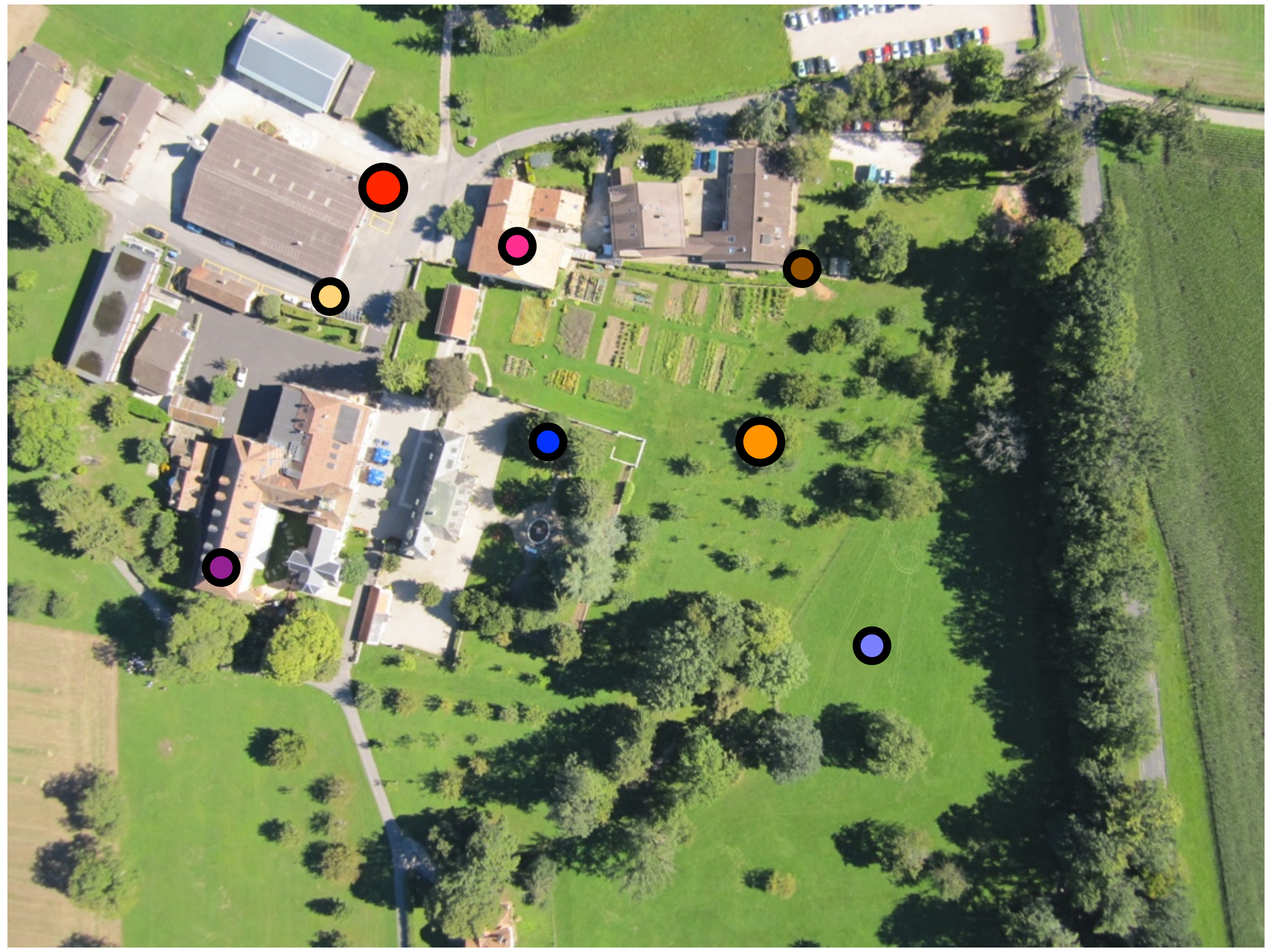
#inliers = 2

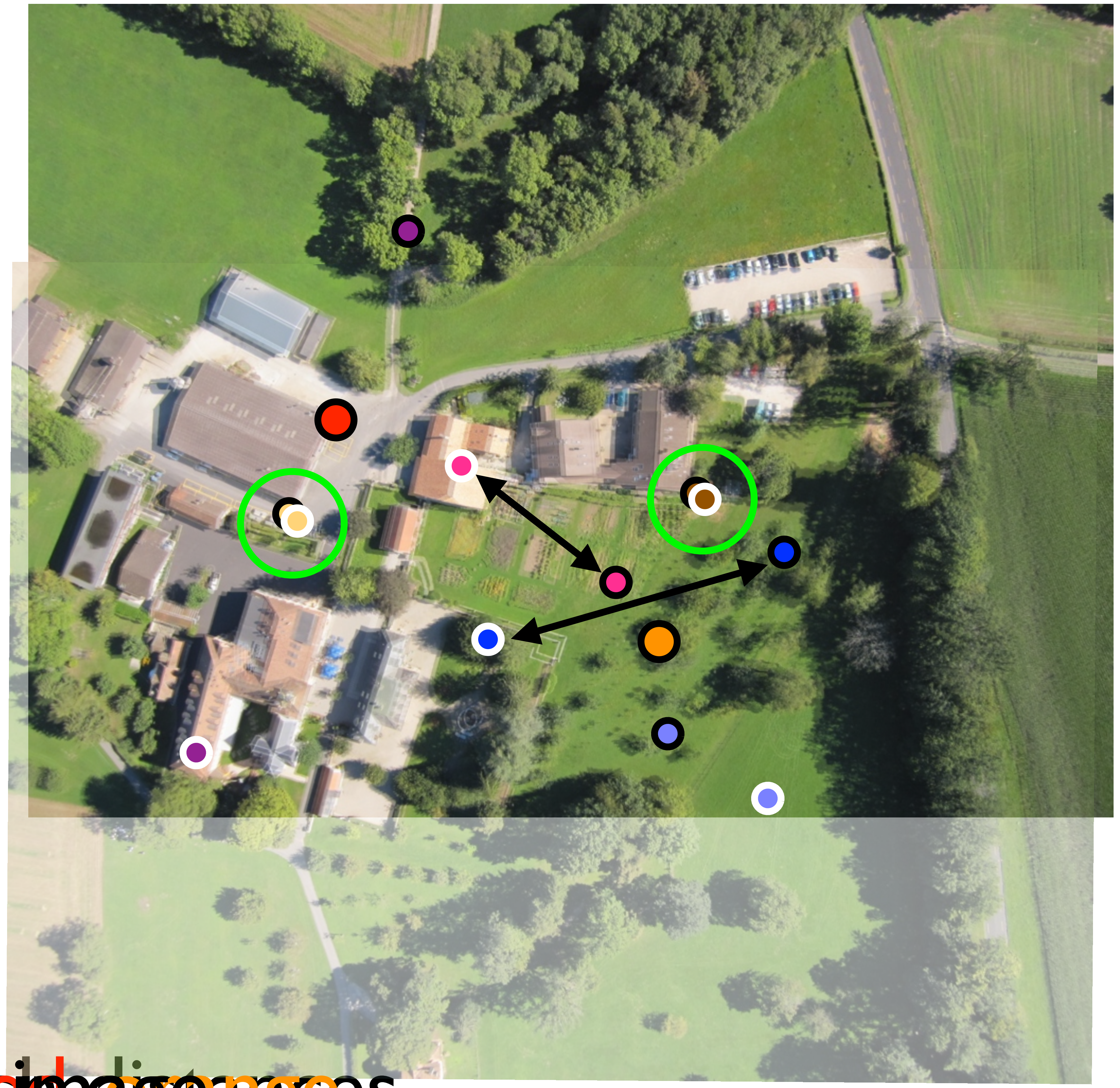
Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)





check distances
check red, orange
warp image

#inliers = 4

Image **Alignment + RANSAC**

RANSAC solution for Similarity Transform (2 points)

Image **Alignment + RANSAC**

Assignment 4

1. Match feature points between 2 views

2. Select minimal subset of matches*

3. Compute transformation T using minimal subset

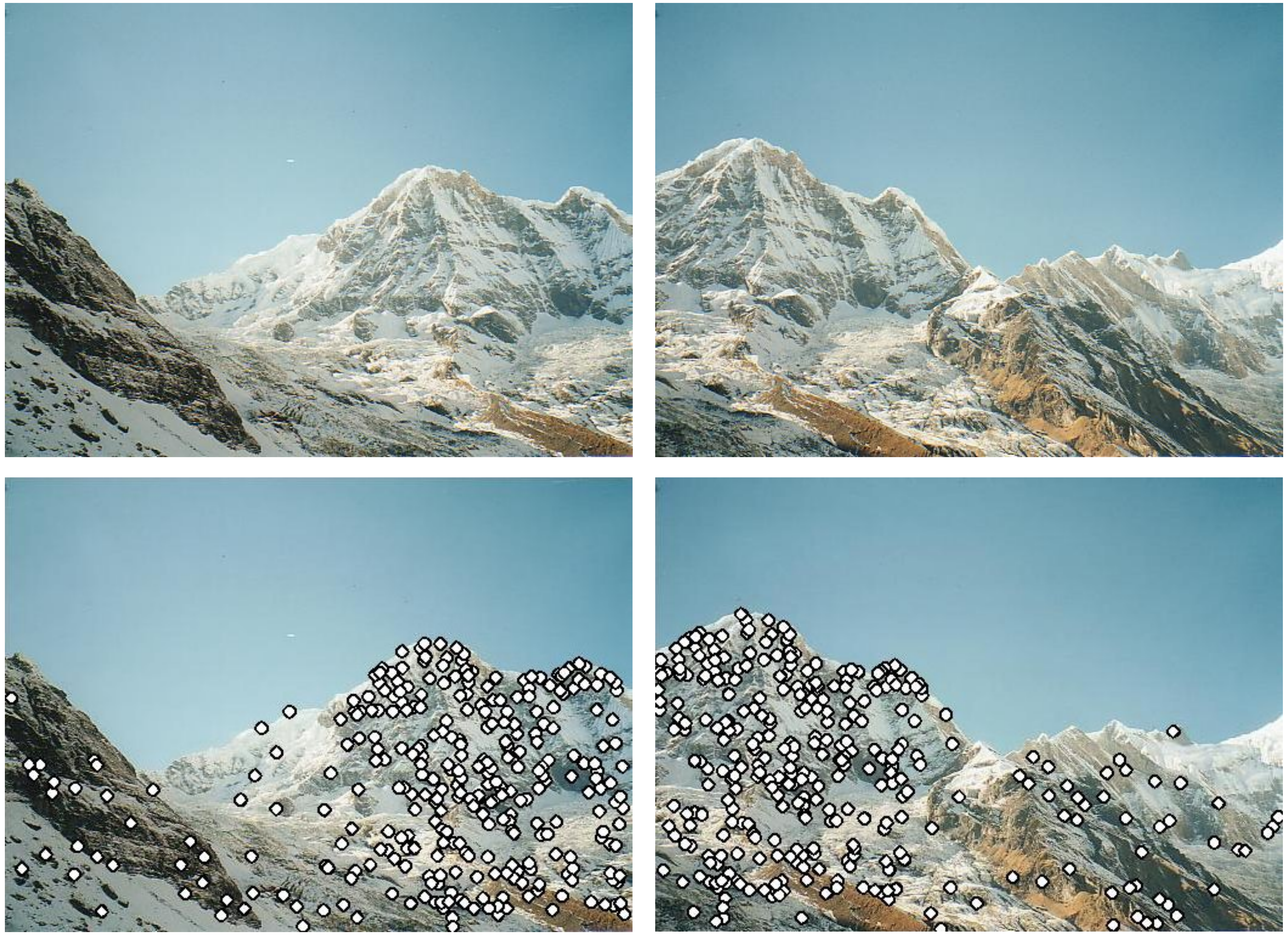
4. Check consistency of all points with T — compute projected position and count #inliers with distance < threshold

5. Repeat steps 2-4 to maximize #inliers

* Similarity transform = 2 points, Affine = 3, Homography = 4

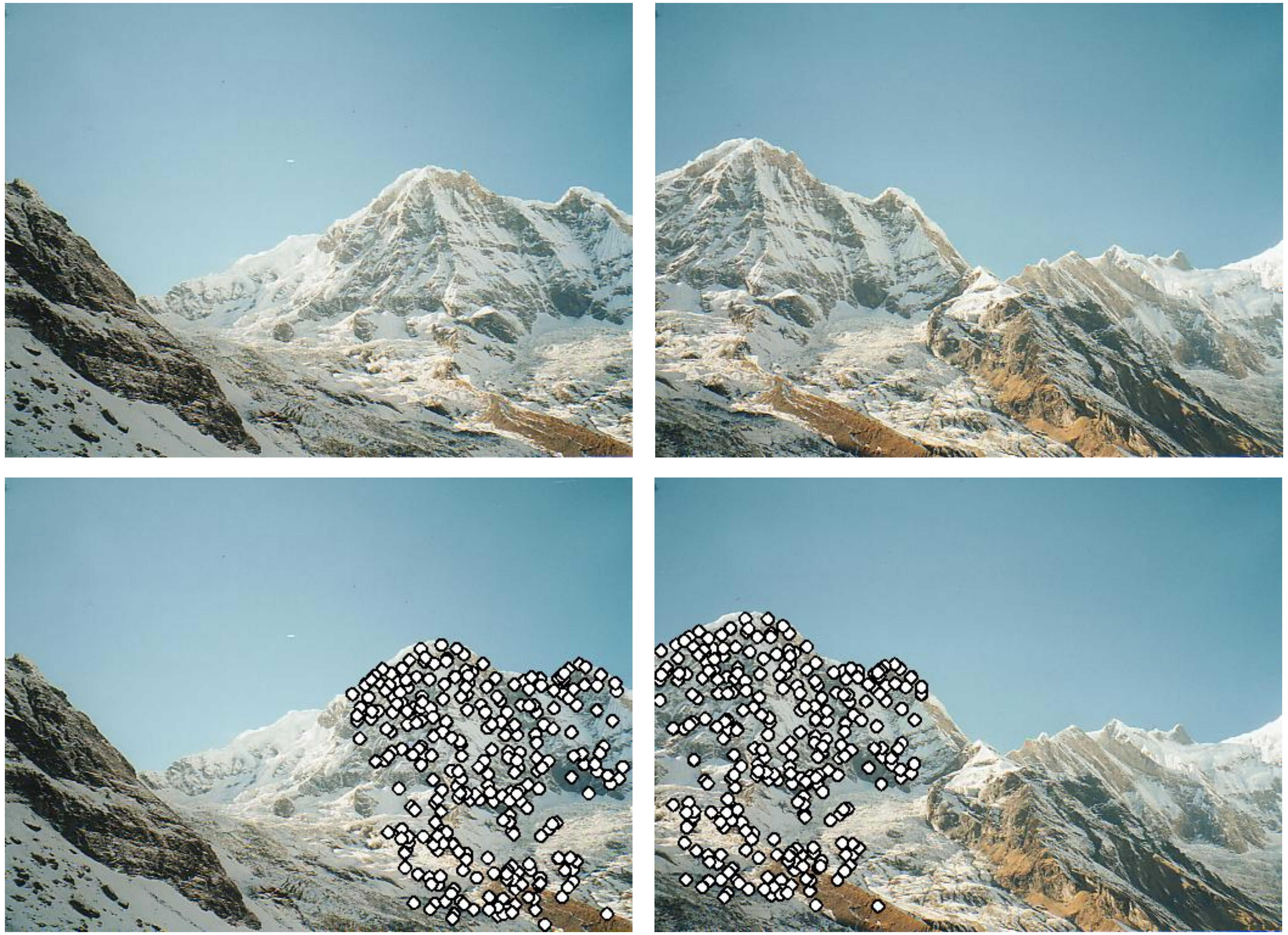
2-view **Rotation** Estimation

Find features + raw matches, use RANSAC to find Similarity



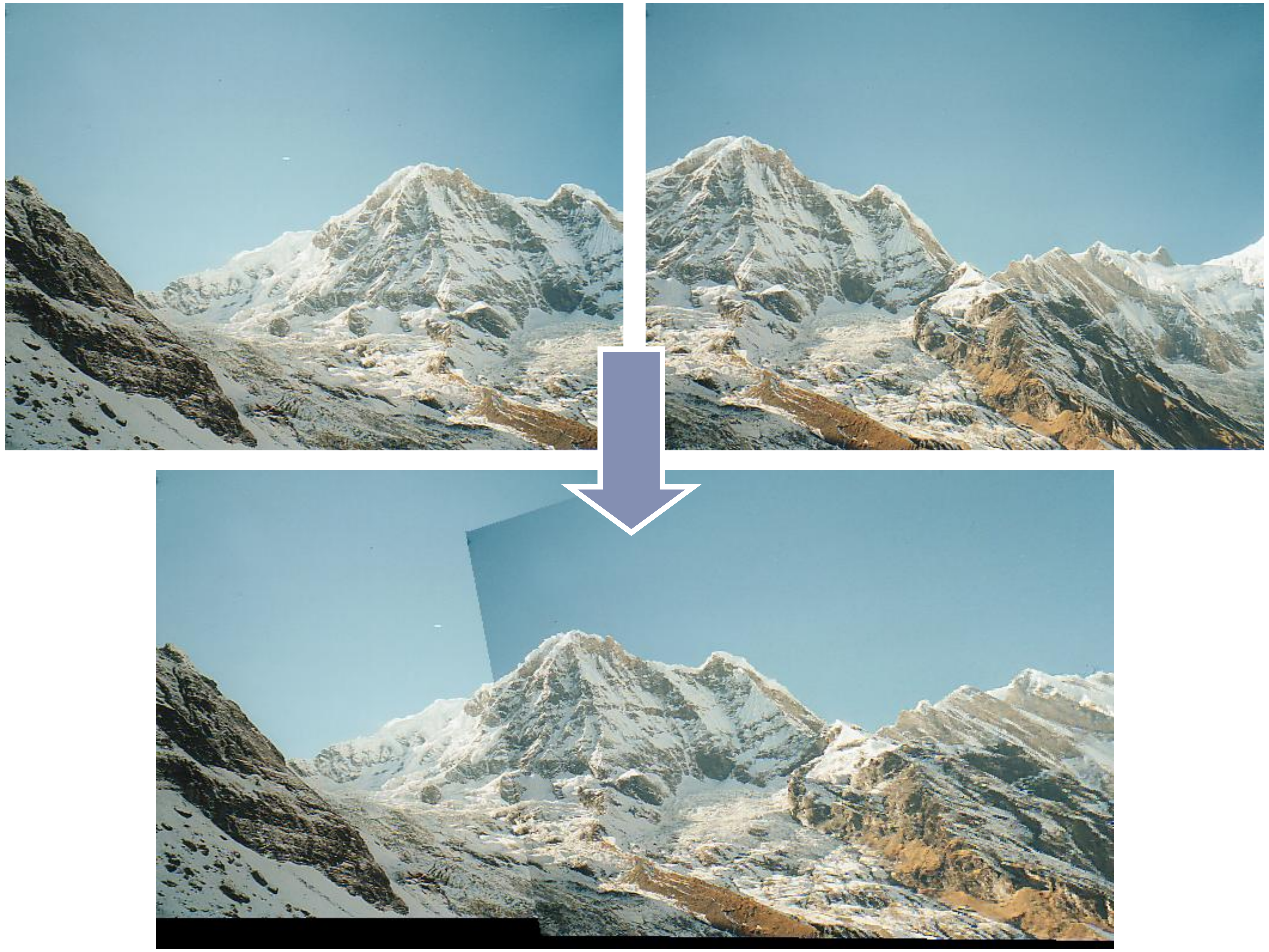
2-view **Rotation** Estimation

Remove outliers, can now solve for R using least squares



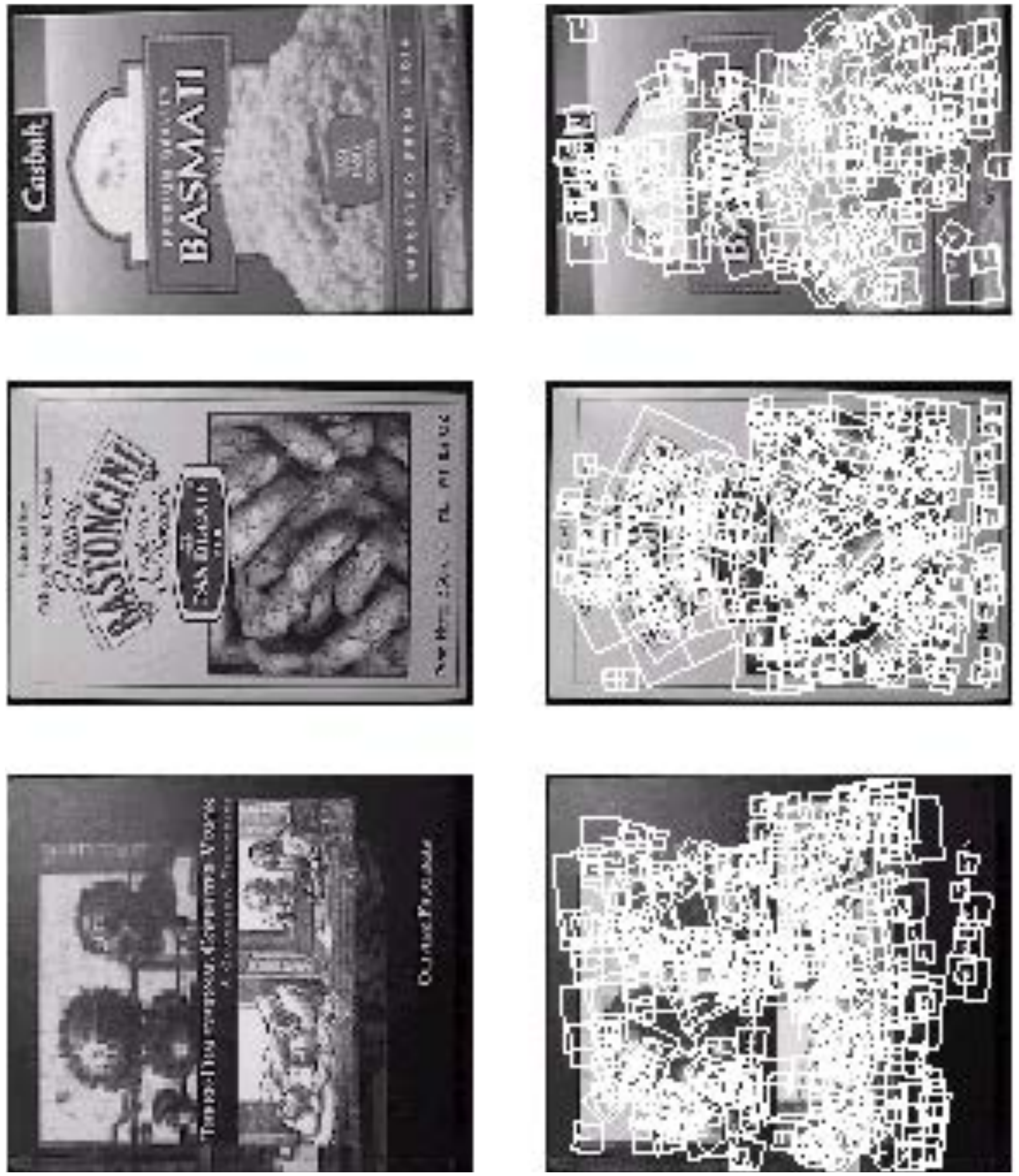
2-view **Rotation** Estimation

Final rotation estimation



Object **Instance Recognition**

Database of planar objects



Instance recognition

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Object **Instance Recognition** with SIFT

Match SIFT descriptors between **query image** and a database of known keypoints extracted from **training examples**

— use fast (approximate) nearest neighbour matching

— threshold based on ratio of distances between 1NN and 2NN

Use **RANSAC** to find a **subset of matches** that all agree on an object and geometric transform (e.g., **affine transform**)

Optionally **refine pose estimate** by recomputing the transformation using all the RANSAC 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, what fraction of pairs will consist entirely of ‘good’ data points (inliers)?

RANSAC: How many samples?

Let p_0 be the fraction of outliers (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

How many samples do we need to find a good solution?

13.7

RANSAC: How many samples? (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

In practice…

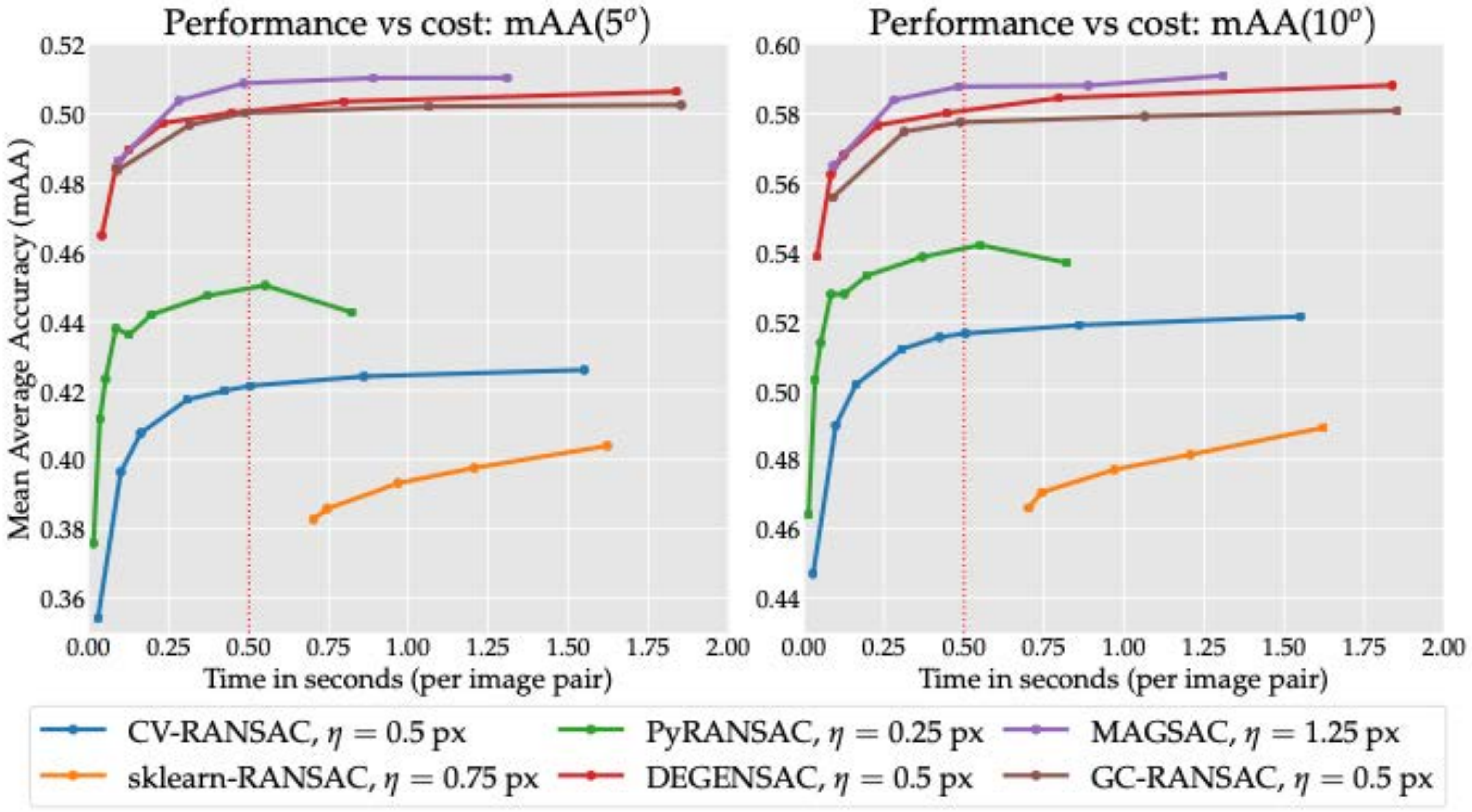


Fig. 9 Validation – Performance vs. cost for RANSAC. We evaluate six RANSAC variants, using 8k SIFT features with "both" matching and a ratio test threshold of r=0.8. The inlier threshold η and iterations limit Γ are variables – we plot only the best η for each method, for clarity, and set a budget of 0.5 seconds per image pair (dotted red line). For each RANSAC variant, we pick the largest Γ under this time "limit" and use it for all validation experiments. Computed on 'n1-standard-2' VMs on Google Compute (2 vCPUs, 7.5 GB).

[Jin et al., 2021]

Re-cap: RANSAC

RANSAC is a technique to fit data to a model

— divide data into inliers and outliers

— estimate model from minimal set of inliers

— improve model estimate using all inliers

— alternate fitting with re-classification as inlier/outlier

RANSAC is a general method suited for a wide range of model fitting problems

— easy to implement

— easy to estimate/control failure rate

RANSAC only handles a moderate percentage of outliers without cost blowing up

Menu for Today

Topics:

- **Planar** Geometry
- **Image Alignment**, Object Recognition
- **RANSAC**

Readings:

- **Today's** Lecture: Szeliski 2.1, 8.1, Forsyth & Ponce 10.4.2

Reminders:

- **Assignment 3:** Due **TODAY**!

THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision

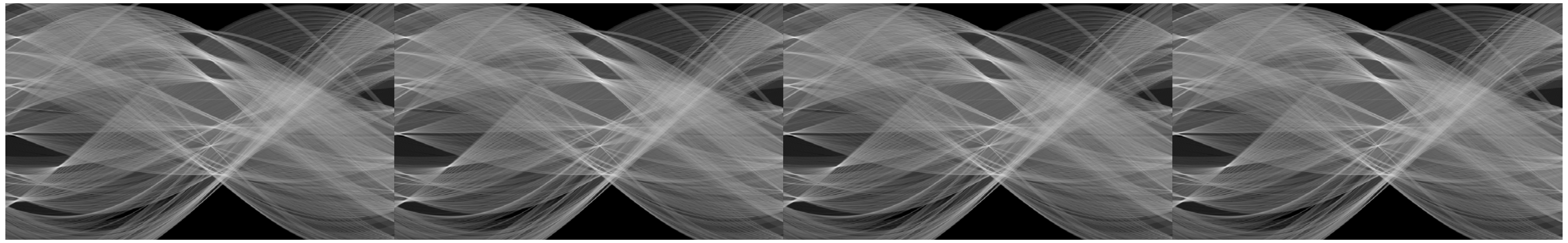


Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 14: Hough Transform

Menu for Today

Topics:

— **Hough** Transform

— Transformation Space Voting

— **Line** Detection

Readings:

— **Today's** Lecture: Szeliski 7.4, Forsyth & Ponce 10.1

Reminders:

— **Assignment 4**: RANSAC and Panorama Stitching — **now available**

— **ECCV** conference deadline is in **1 day**

Learning **Goals**

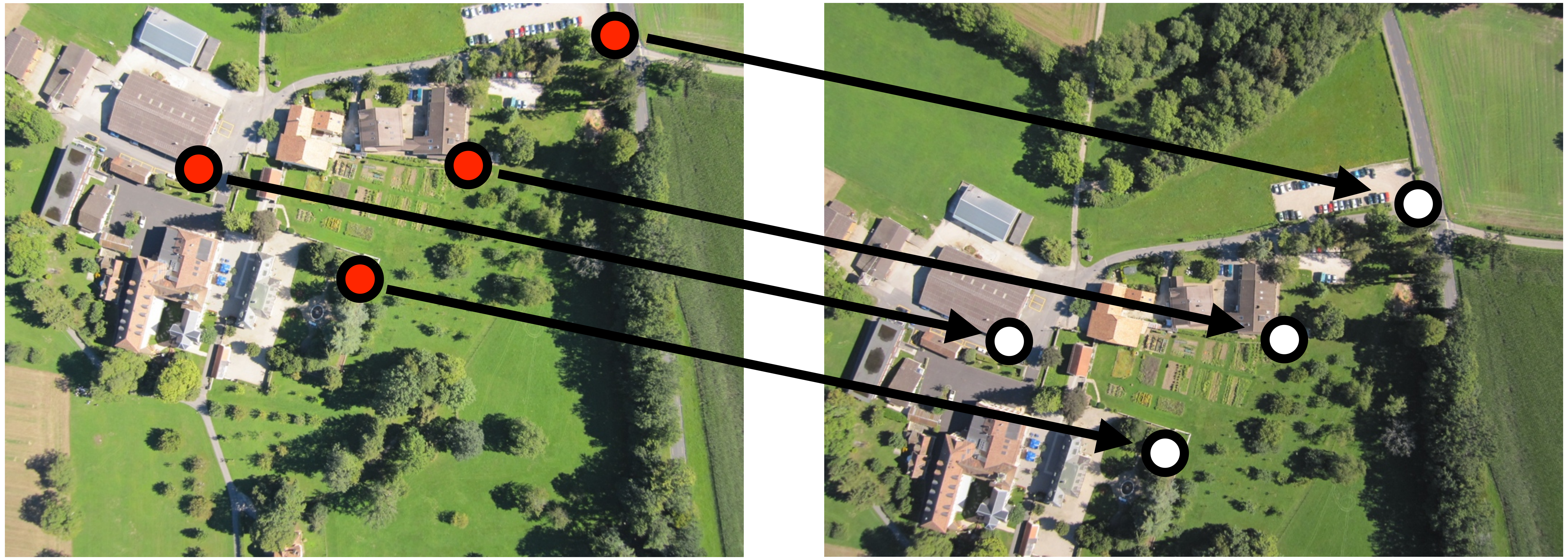
1. How to get **multiple** hypothesis
2. **Voting**-based strategies are useful

Image **Alignment**

Aim: Warp one image to align with another <u>using a 2D transformation</u>

Image **Alignment**

Step 1: Find correspondences (matching points) across two images



$$\mathbf{u} = \mathbf{H}\mathbf{x}$$

2 points for Similarity

3 for Affine

4 for Homography

Image **Alignment**

Step 2: Compute the transformation to align the two images

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**

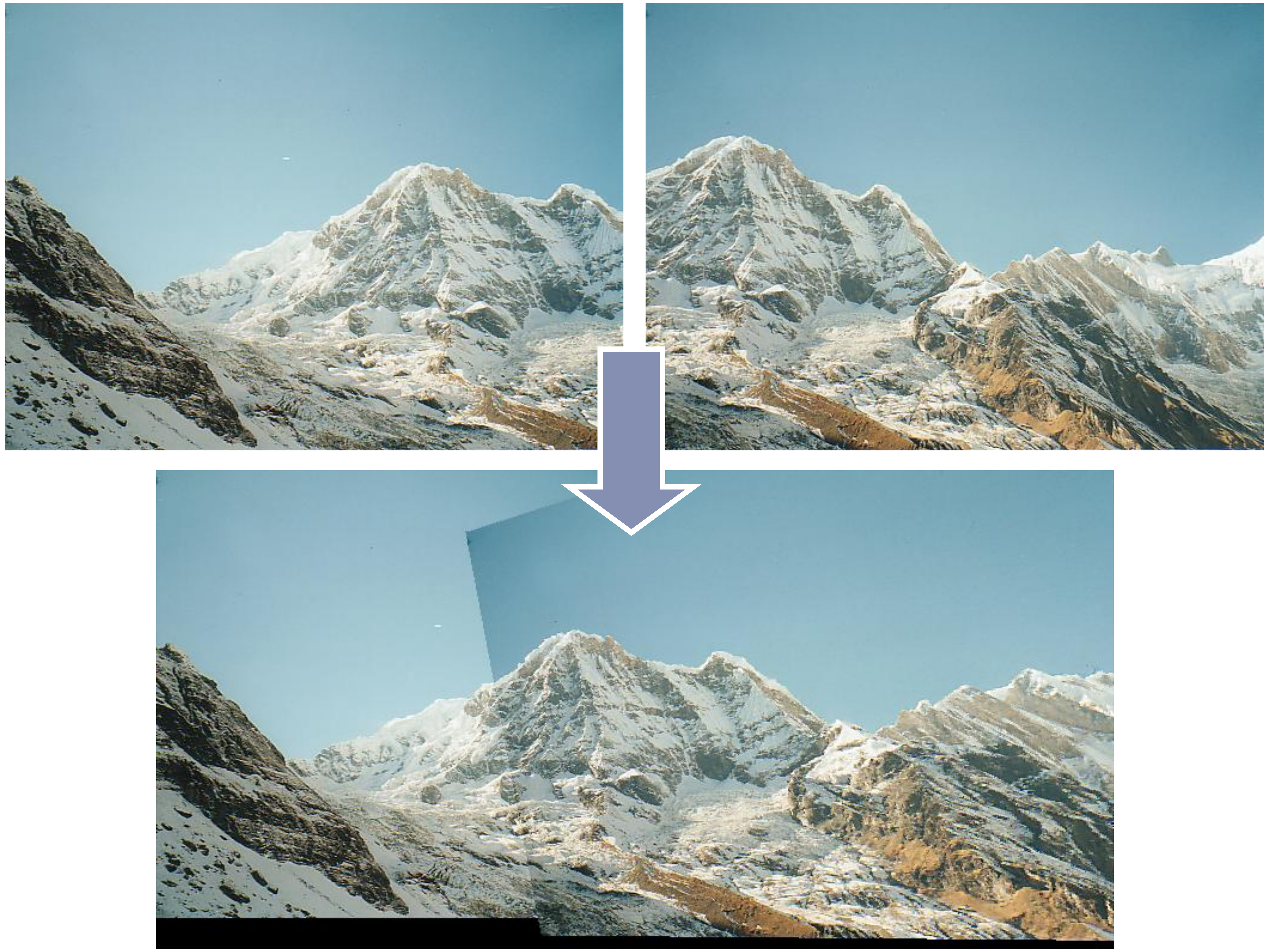
3. Repeat for N samples; model with biggest support is most robust fit
   — Points within distance t of best model are inliers
   — Fit final model to all inliers

RANSAC is very useful for variety of applications

Slide Credit: Christopher Rasmussen

2-view **Rotation** Estimation

Final rotation estimation



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

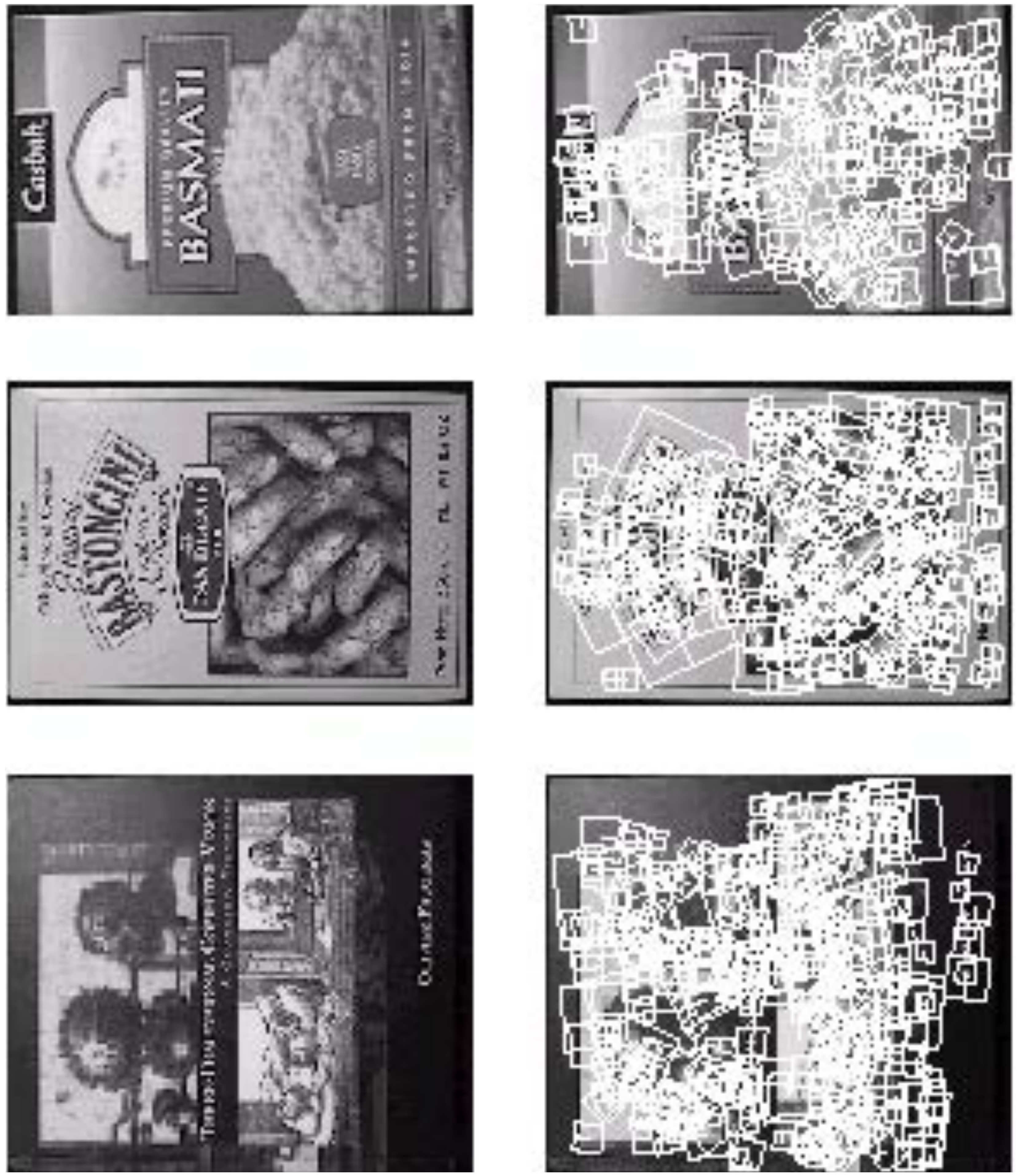
Uses both SIFT and RANSAC

Example: Photo Tourism

[Agarwal, Furukawa, Snavely, Curless, Seitz, Szeliski, 2010]

Object **Instance Recognition**

Database of planar objects



Instance recognition

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

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)

— Hard to deal with multiple solutions (e.g., object detection with many objects)

The **Hough transform** can handle high percentage of outliers

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)~~

— Hard to deal with multiple solutions (e.g., object detection with many objects)

The **Hough transform** can handle high percentage of outliers

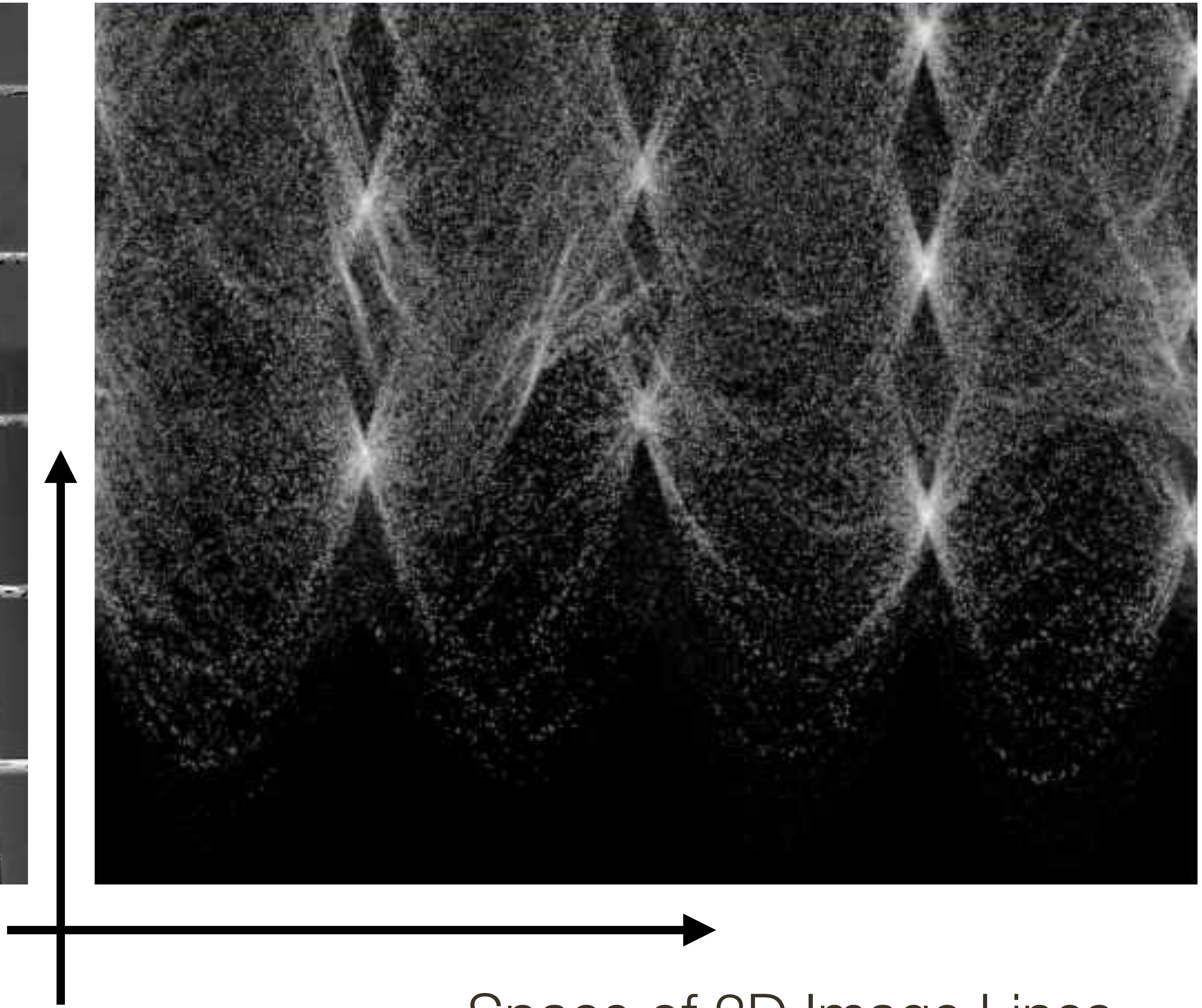
Hough Transform: Motivation

How to find lines in this image?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Motivation

Votes / Probability Distribution



Space of 2D Image Lines

Hough Transform

Idea of **Hough transform**:

— For each token / data point vote for all models to which it could belong

— Return models that get many votes / distribution of possible models

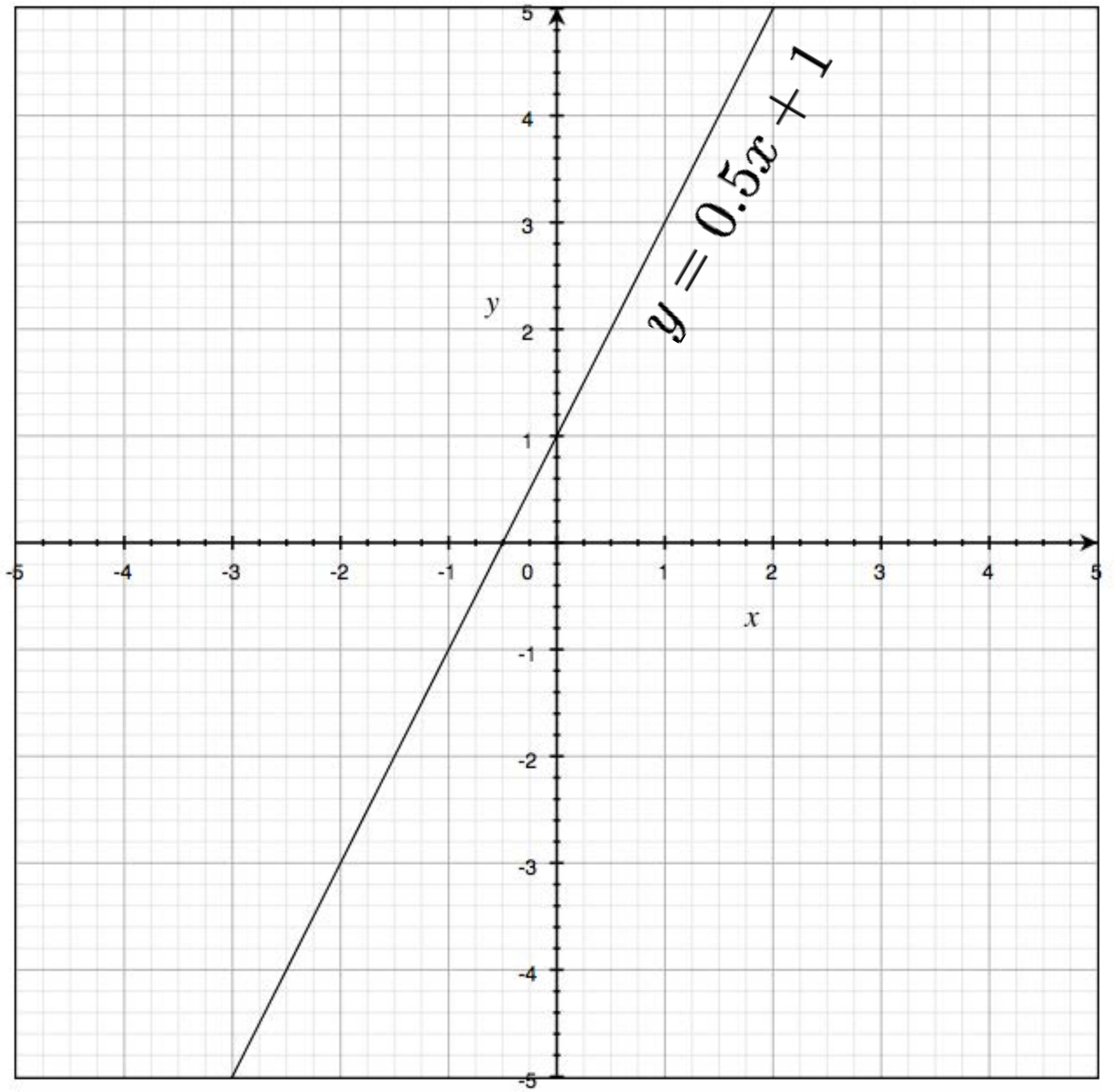
Example: For each point, vote for all lines that could pass through it; the true lines will pass through many points and so receive many votes

c.f. **RANSAC** which optimizes a **single hypothesis** by maximizing the number of inliers (though modifications exist to find multiple instances of a model)

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

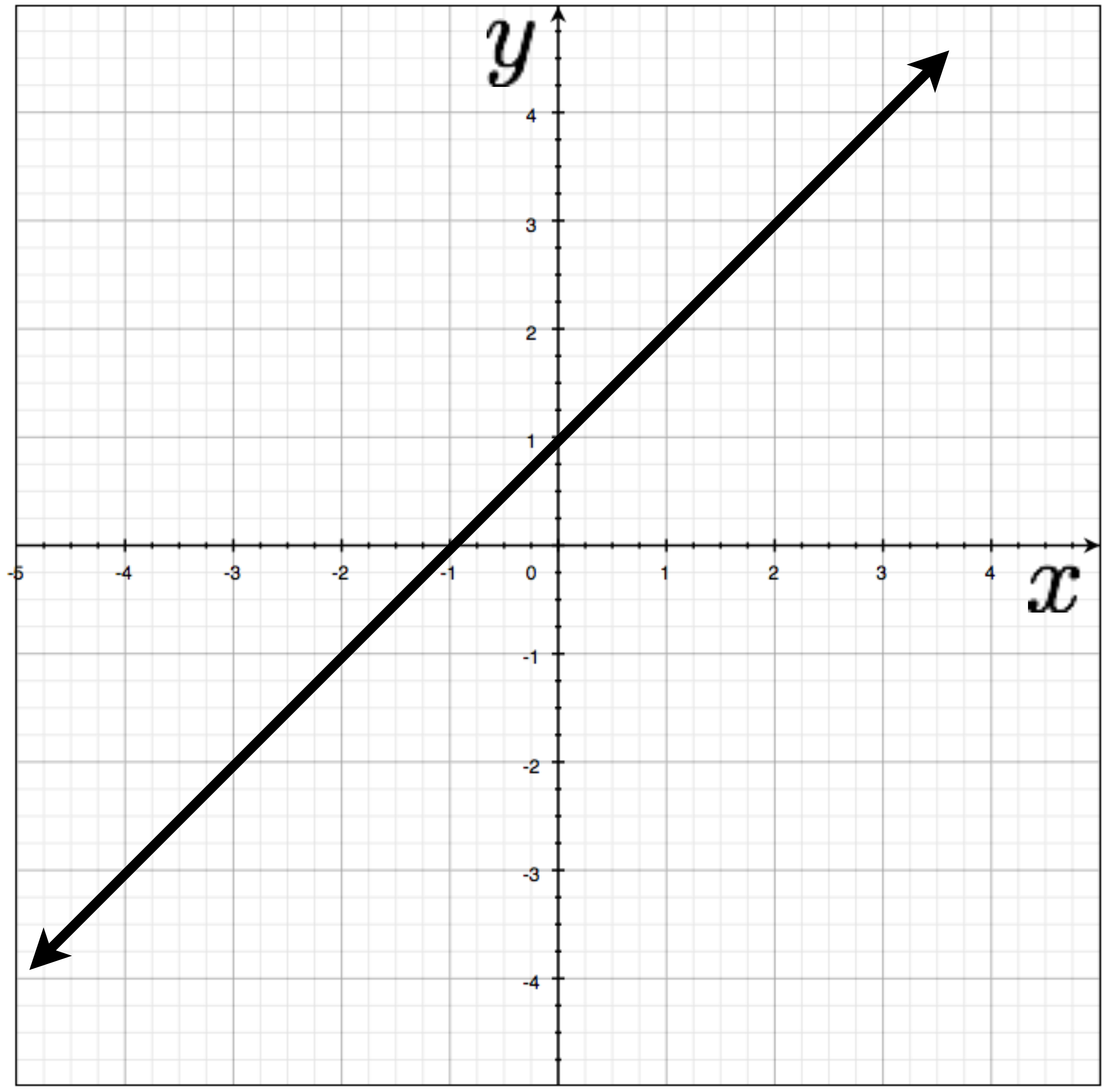


Image space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Image and Parameter Space

variables

$$y = mx + b$$

parameters

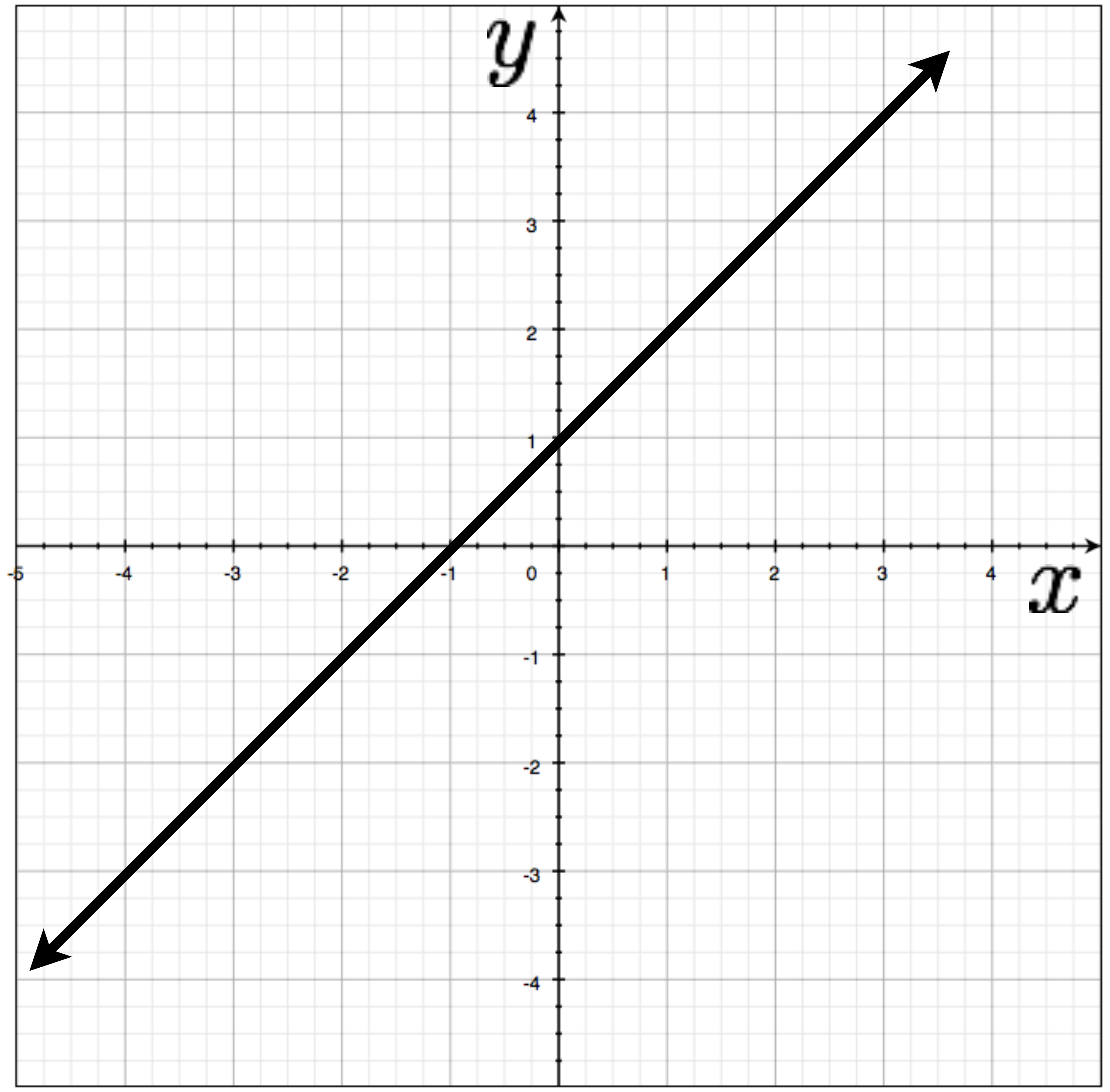


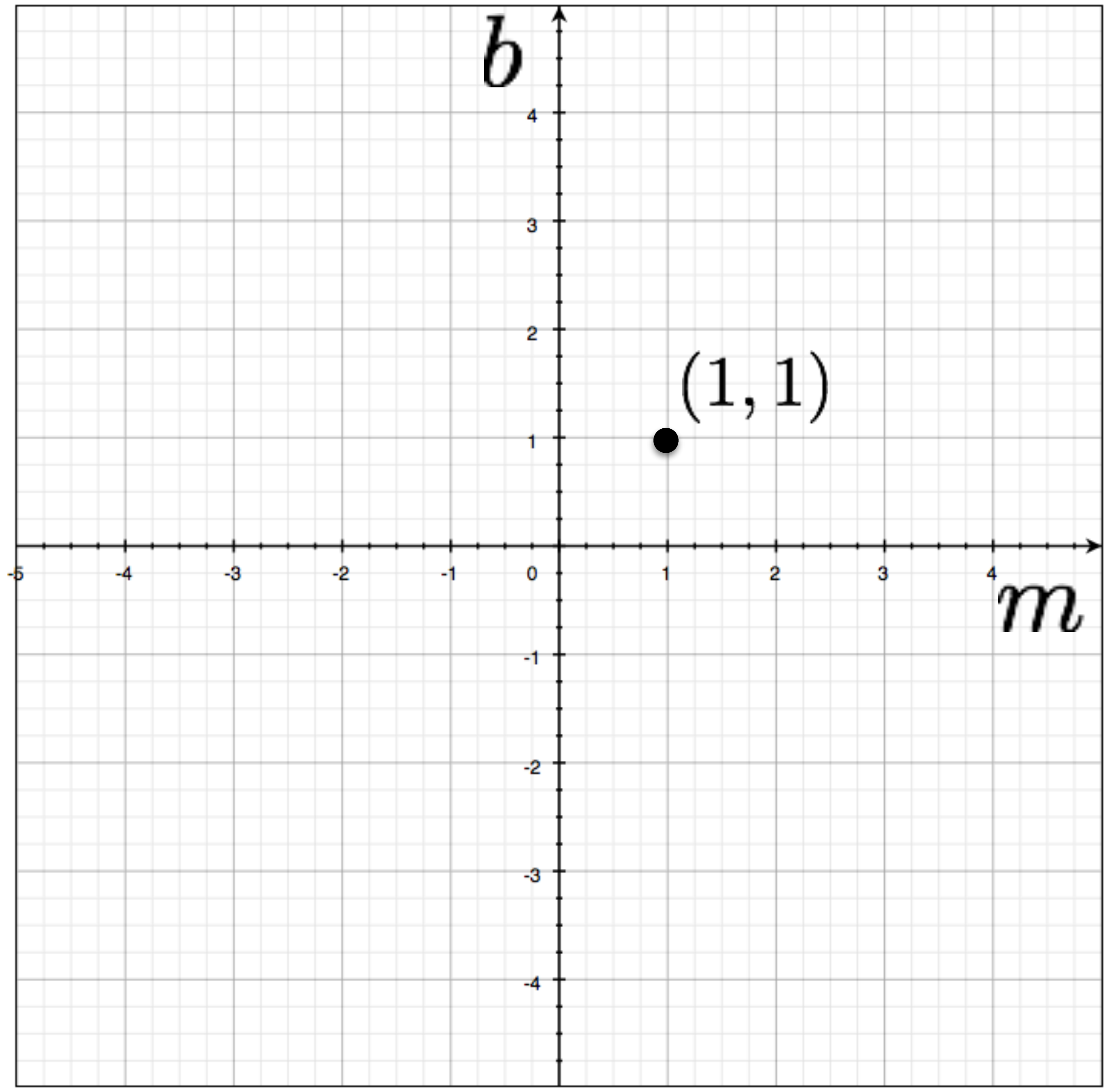
Image space

a line becomes a point

variables

$$y - mx = b$$

parameters



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Image and Parameter Space

variables

$$y = mx + b$$

parameters

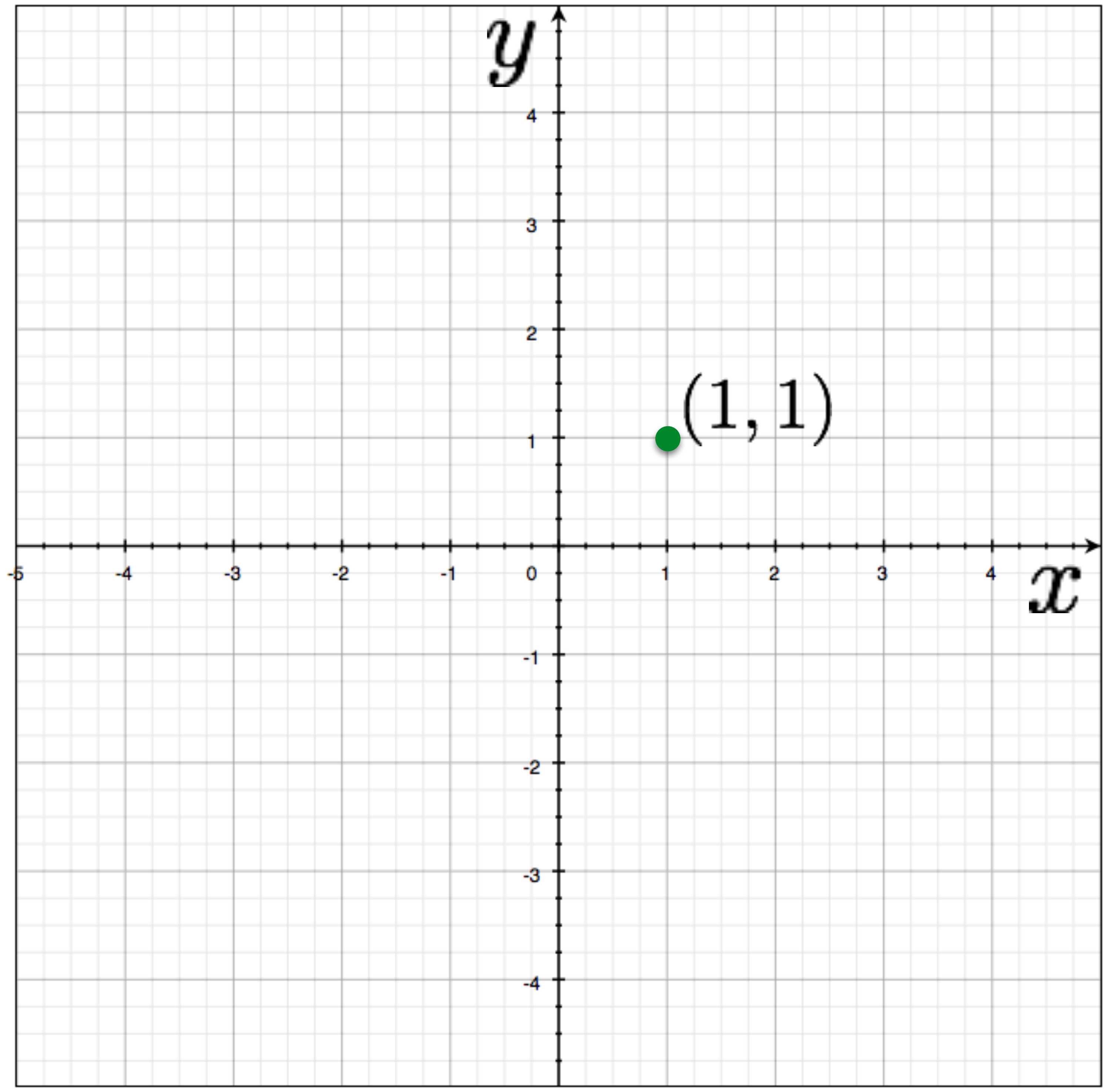


Image space

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

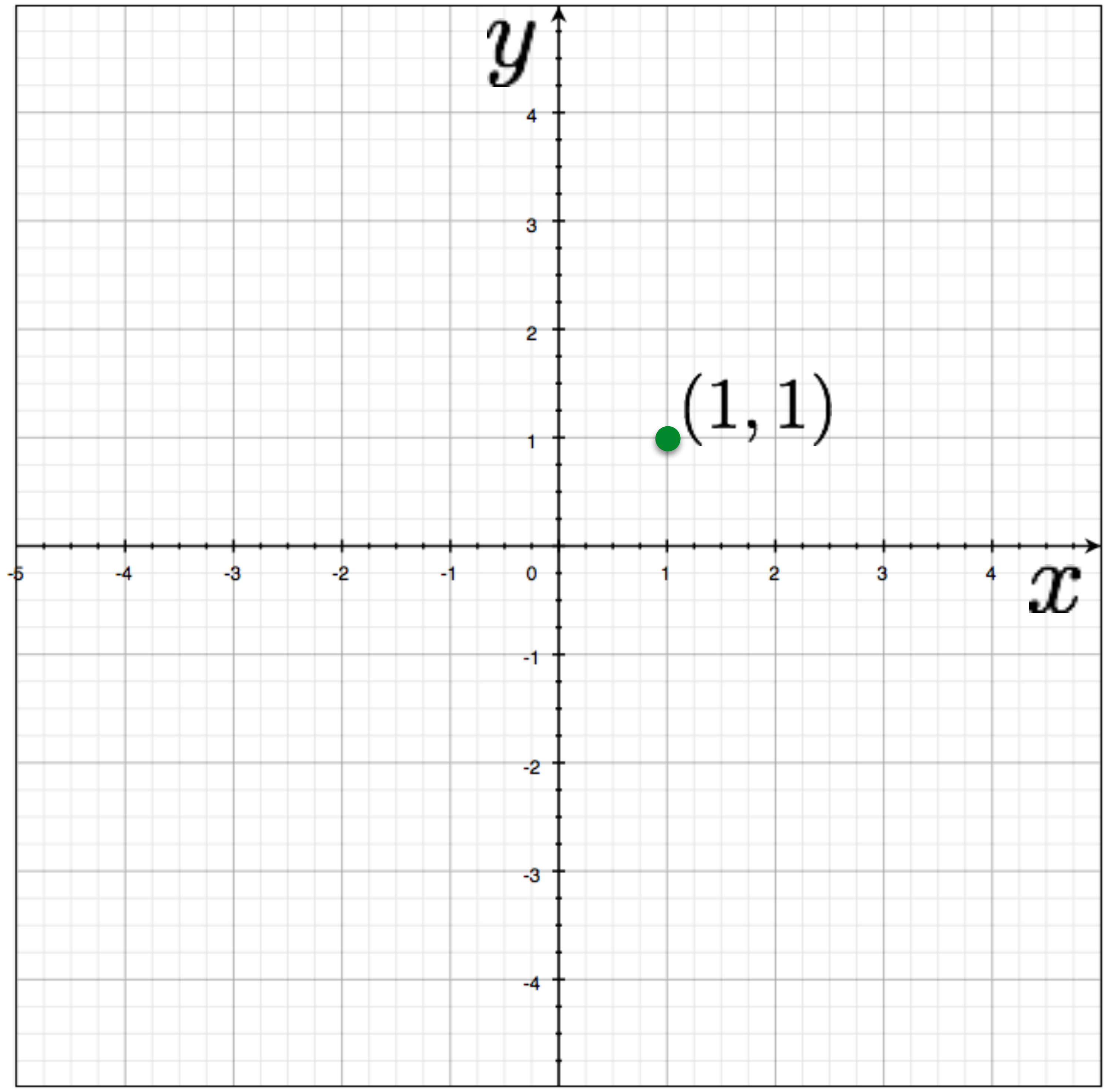


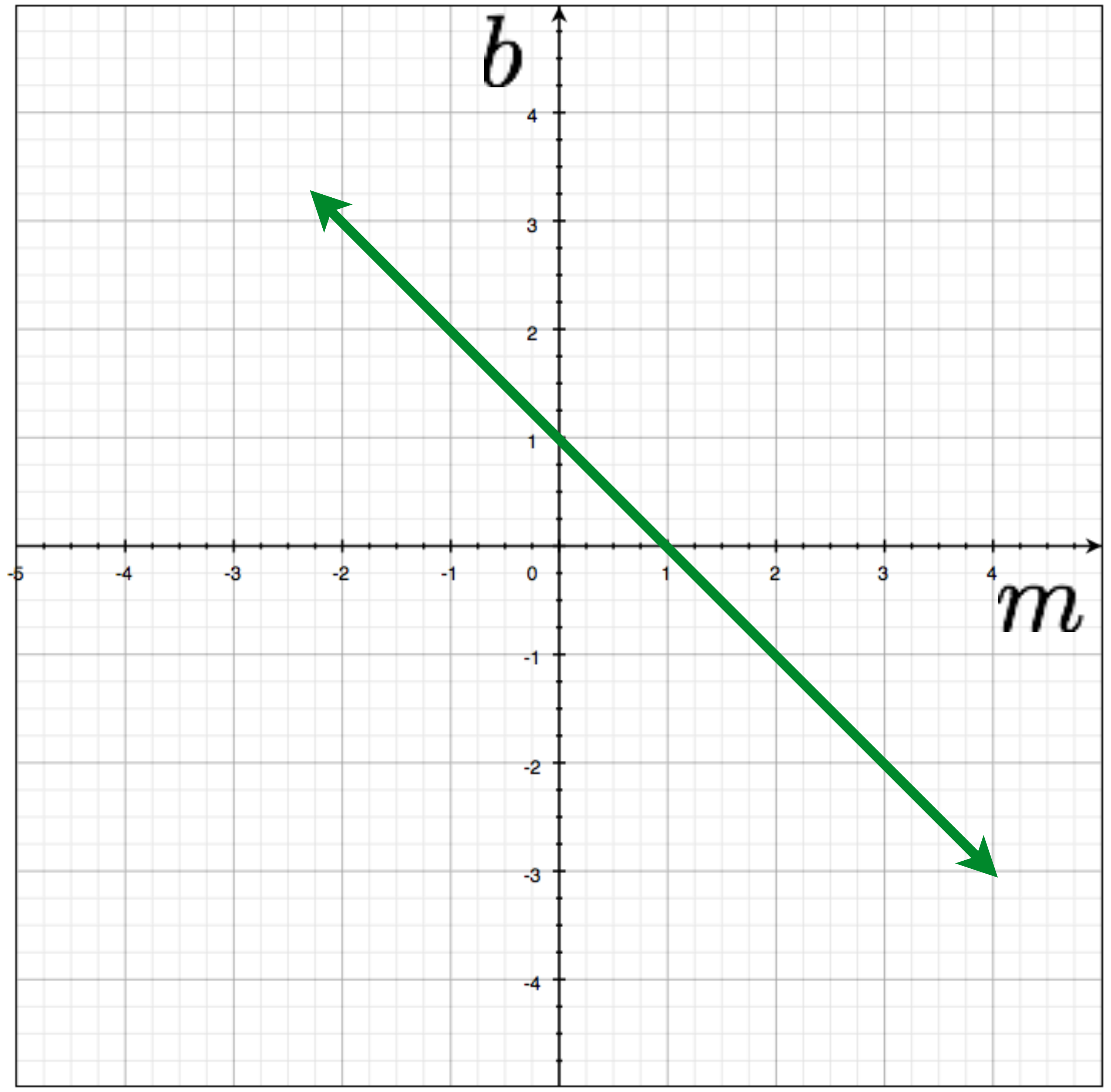
Image space

a point becomes a line

variables

$$y - mx = b$$

parameters



Parameter space

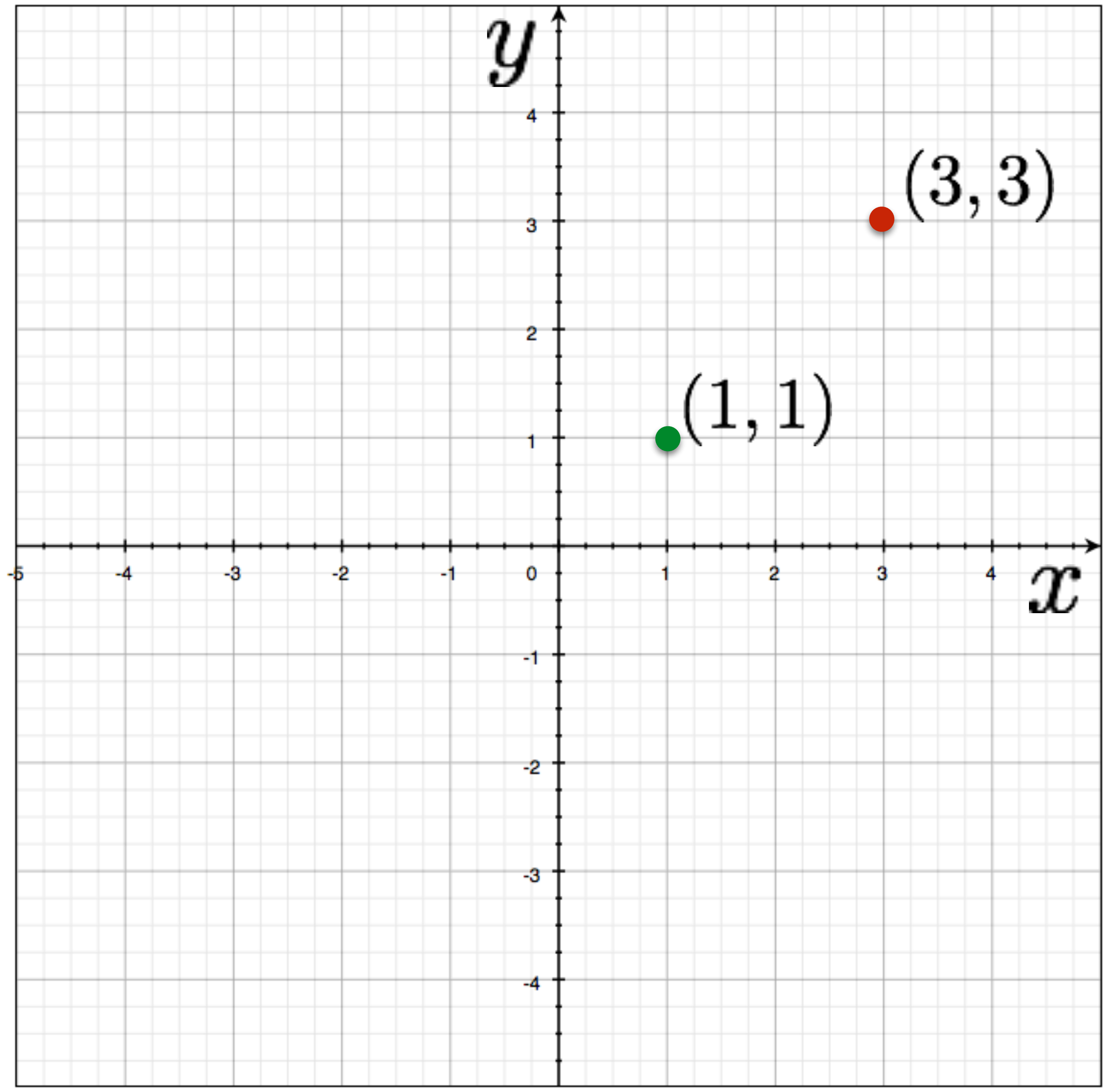
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

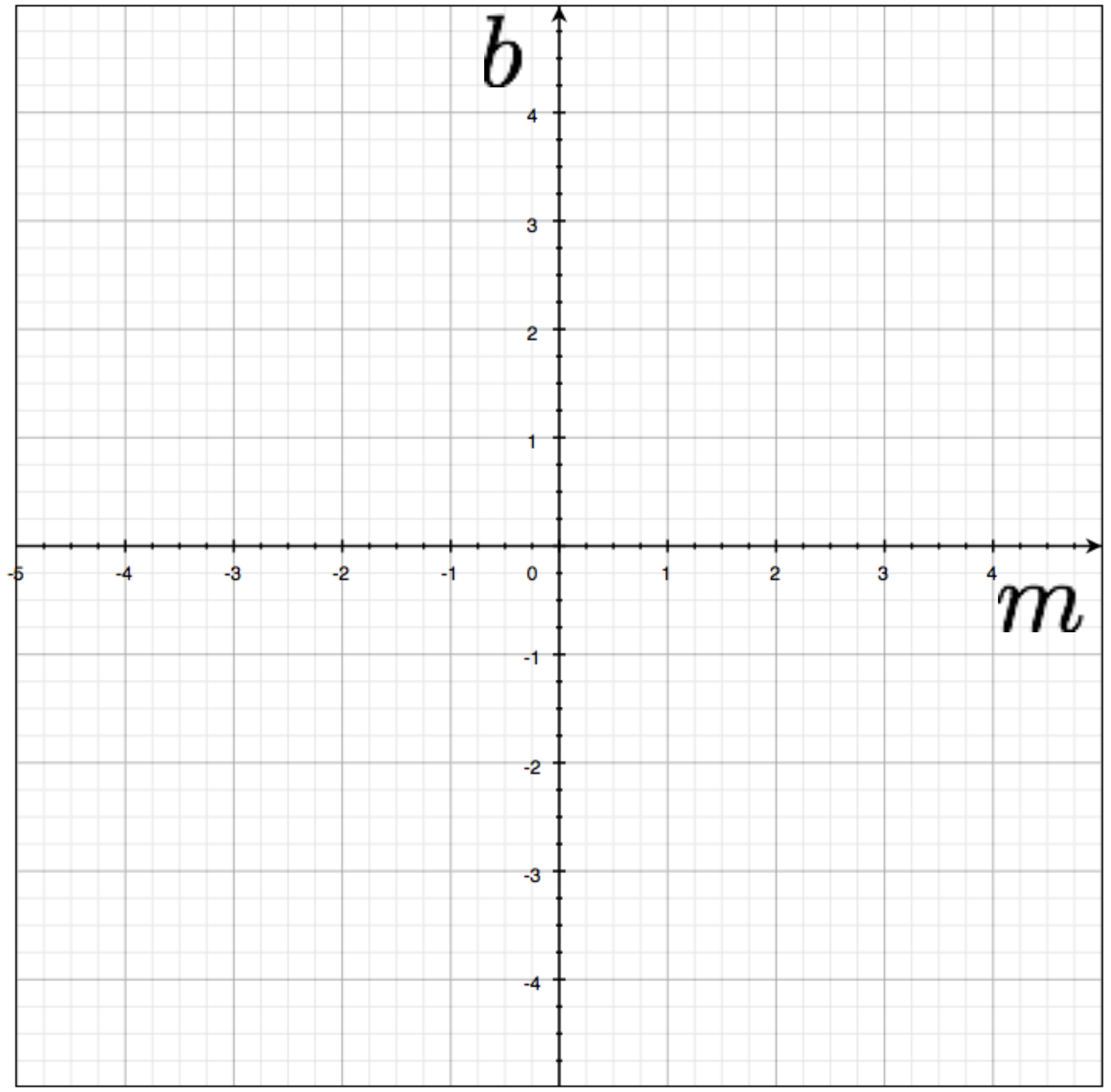
Hough Transform: Lines

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

variables

$$y - mx = b$$

parameters

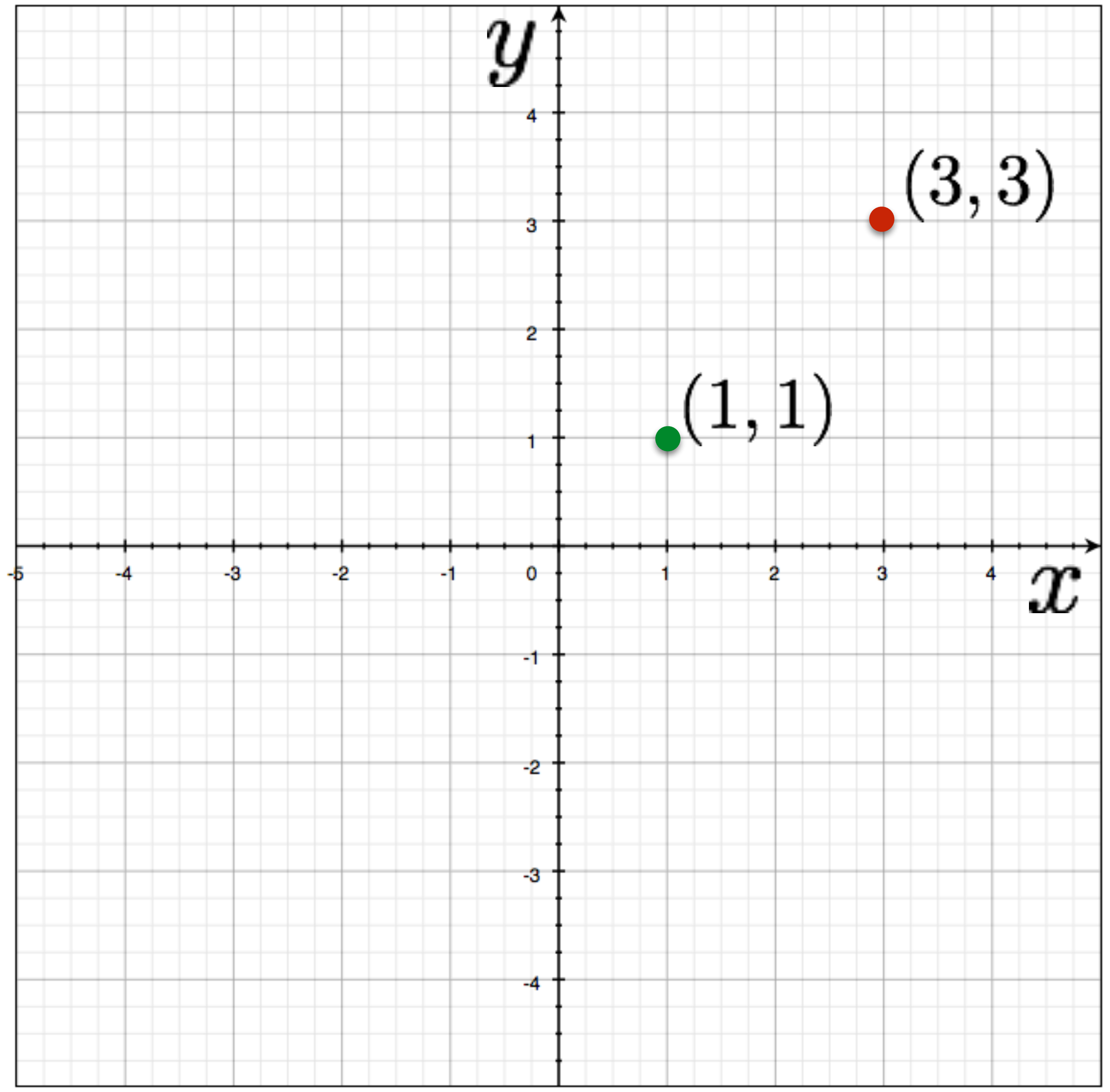
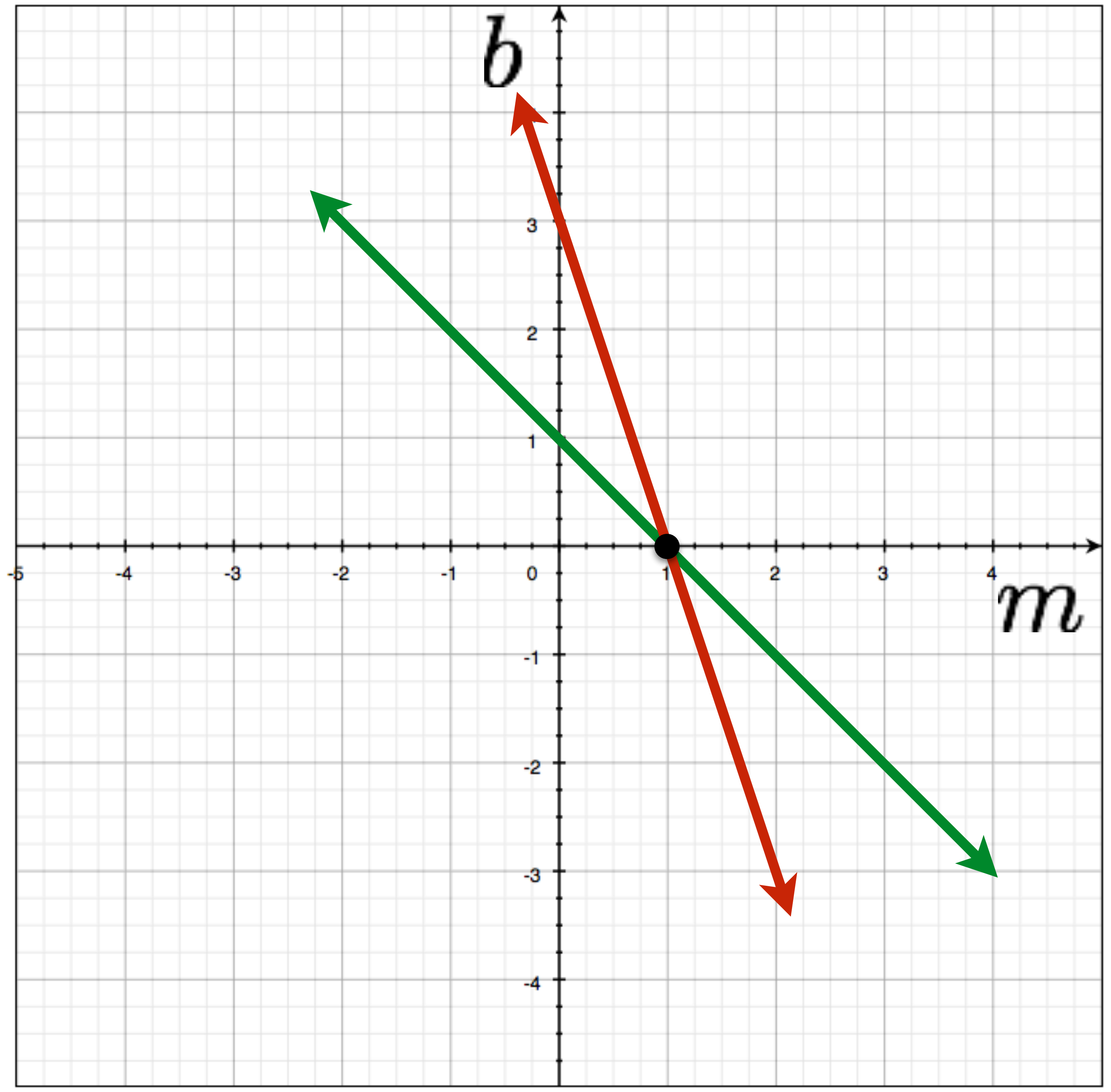


Image space

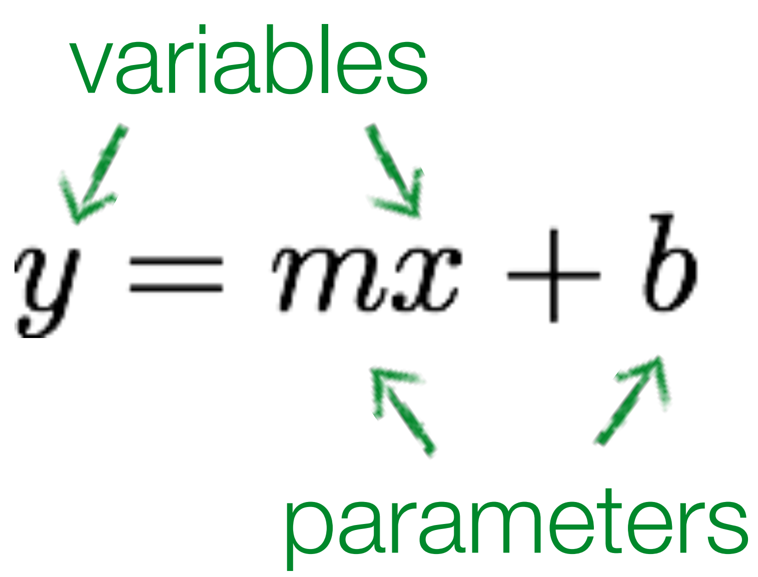
two
points?

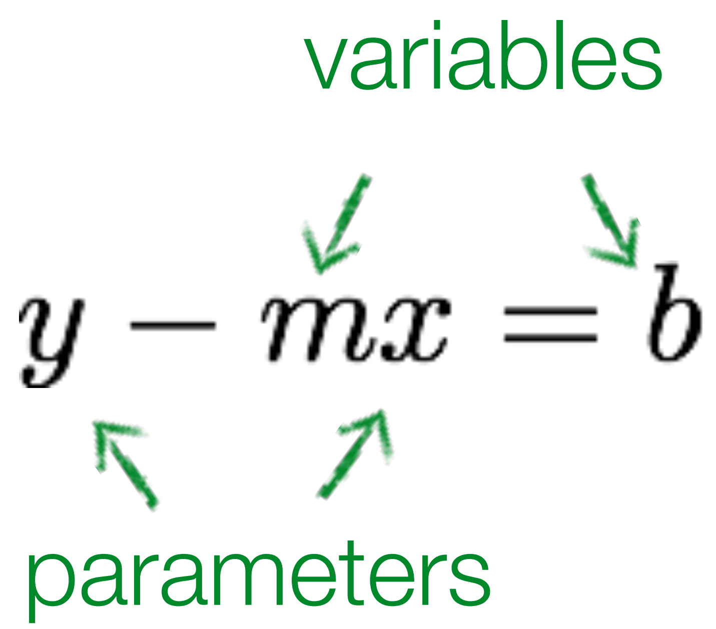


Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines





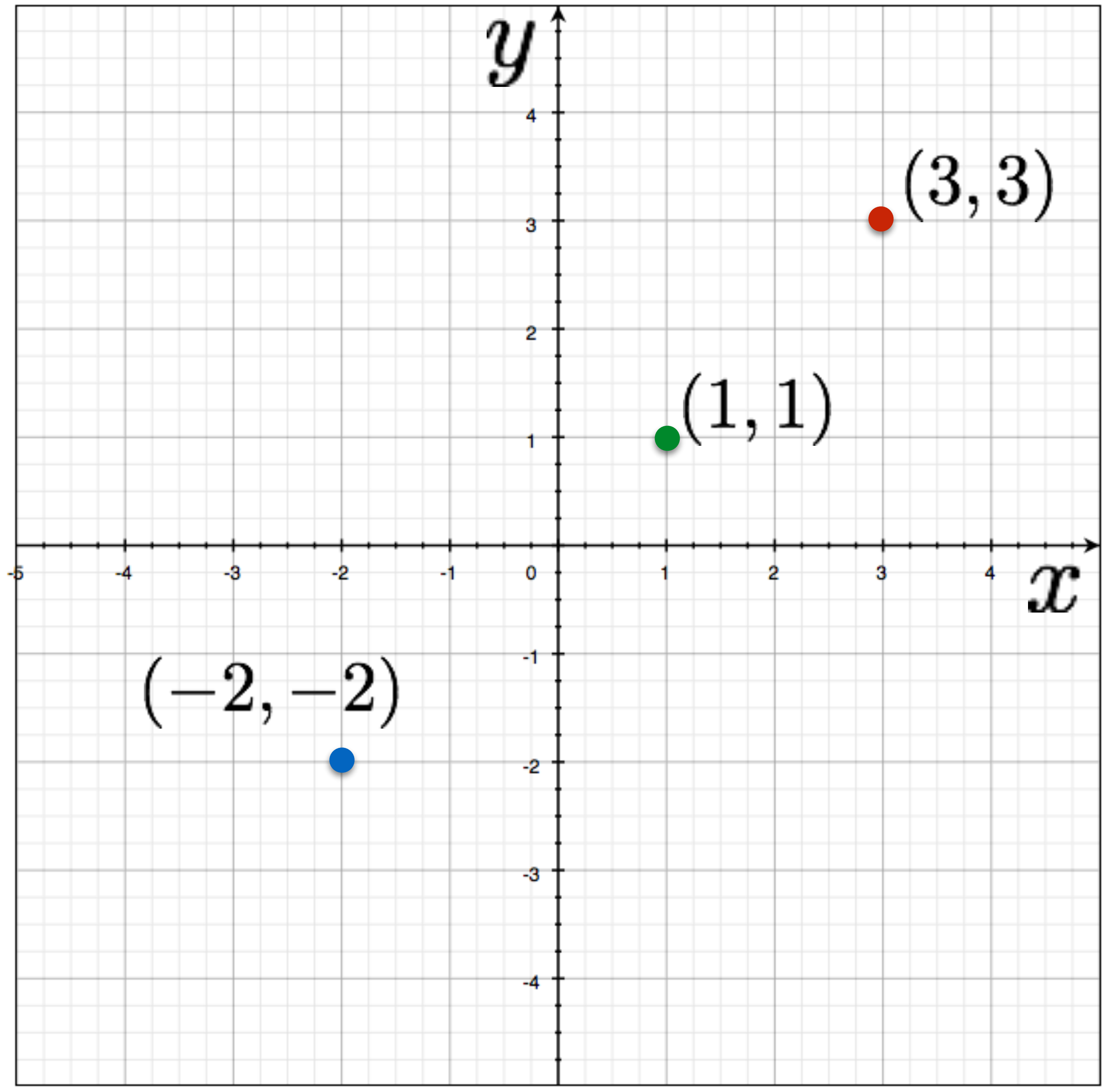
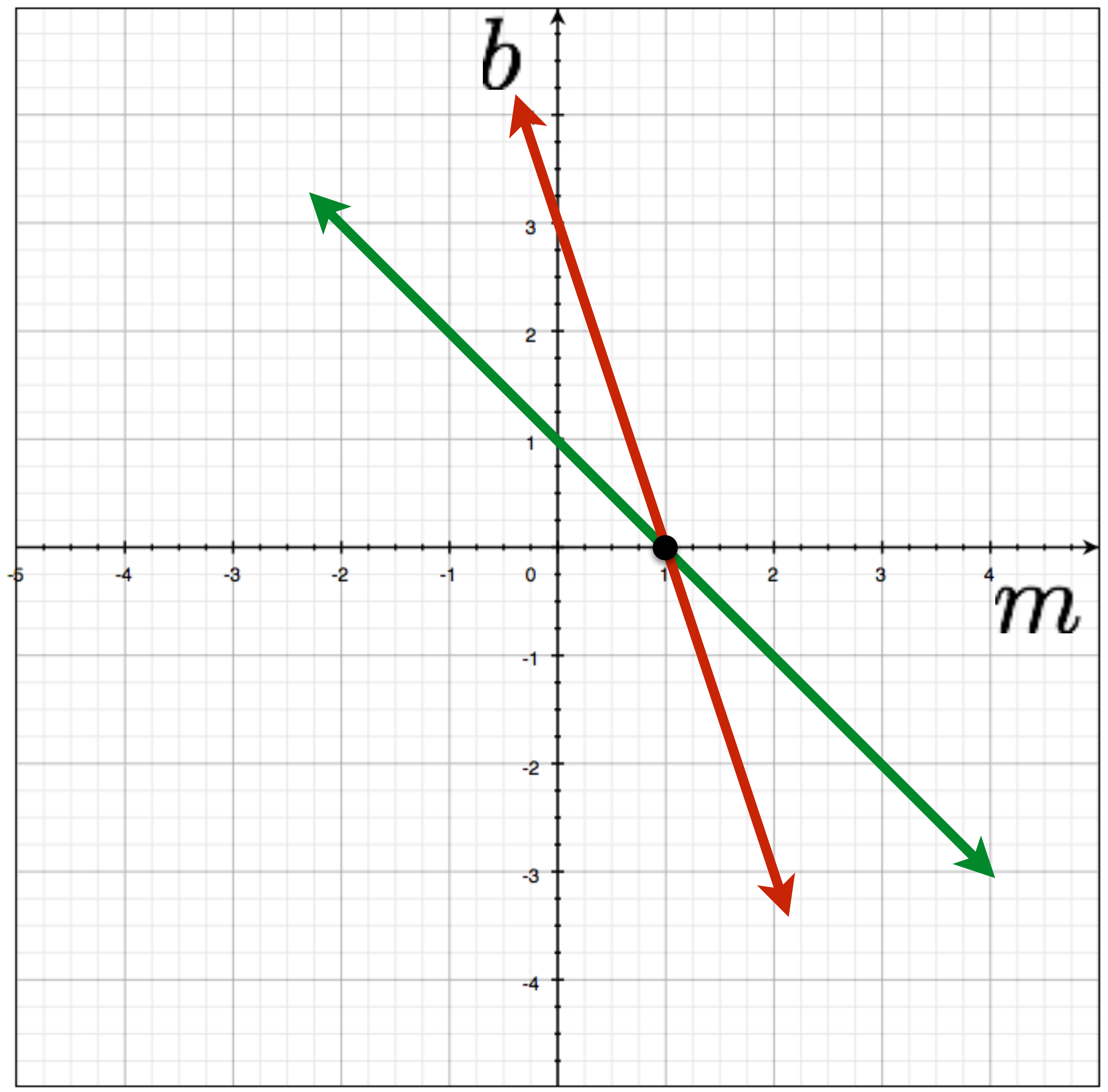


Image space



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

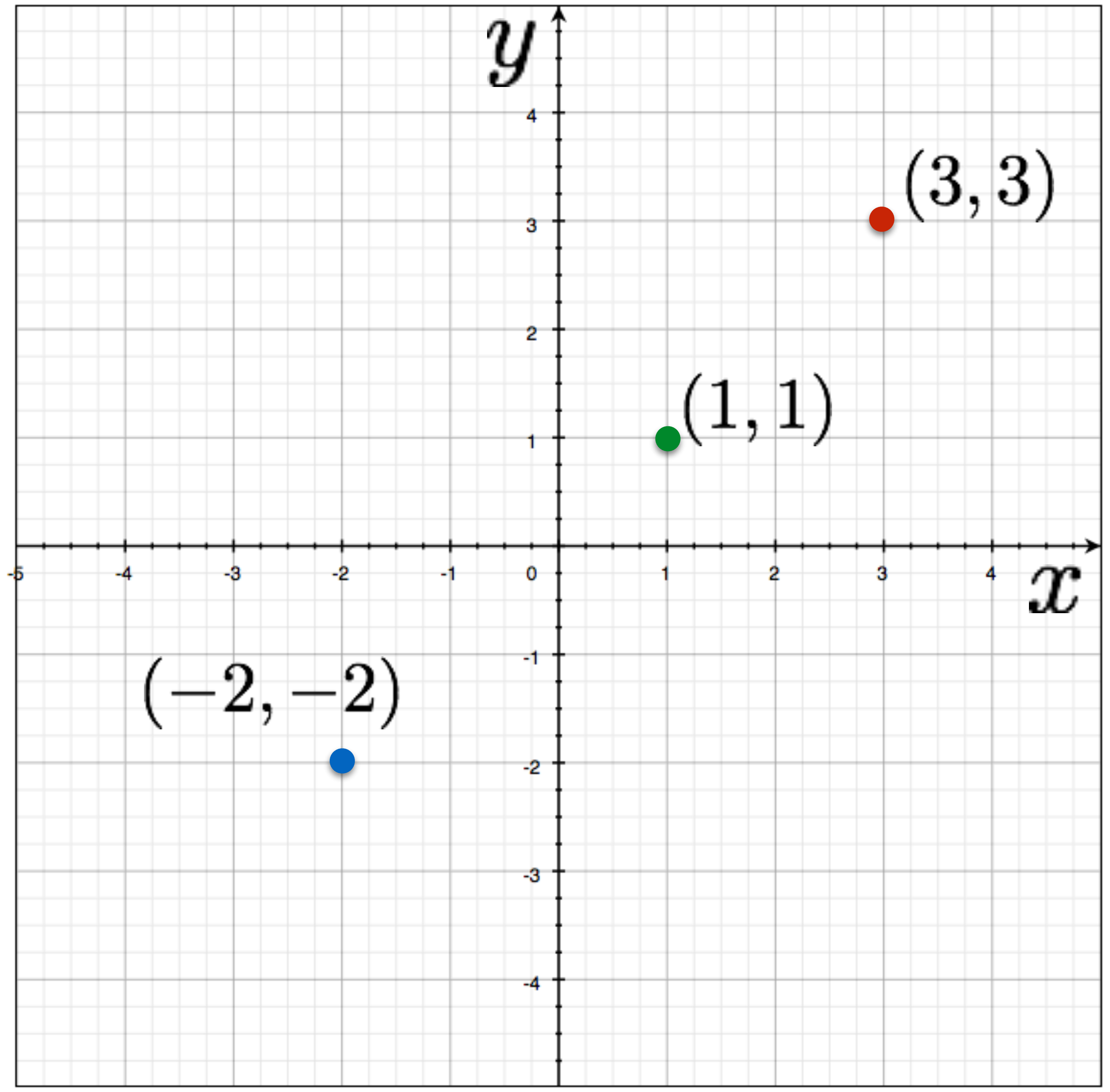


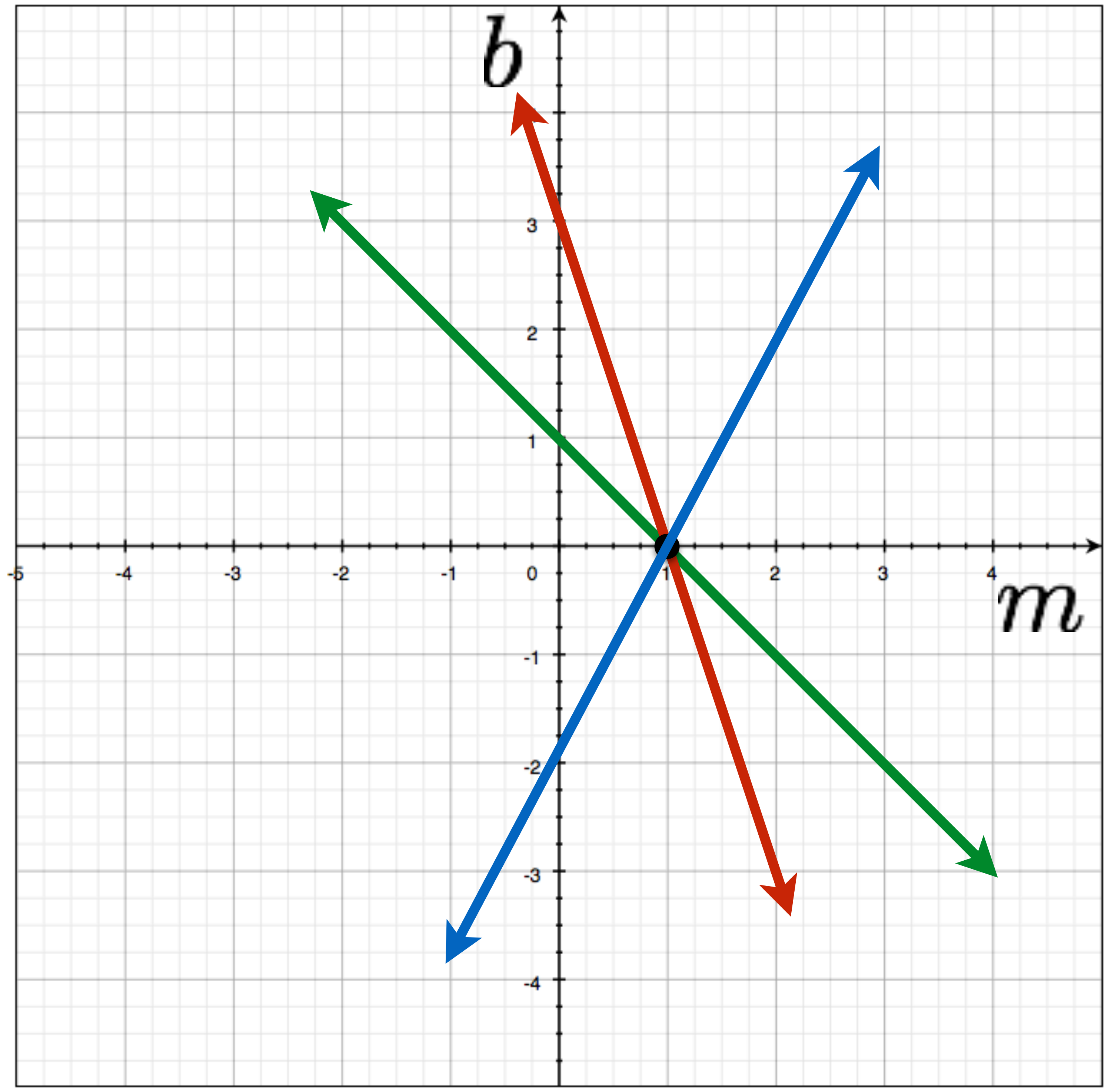
Image space

three
points?

variables

$$y - mx = b$$

parameters



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

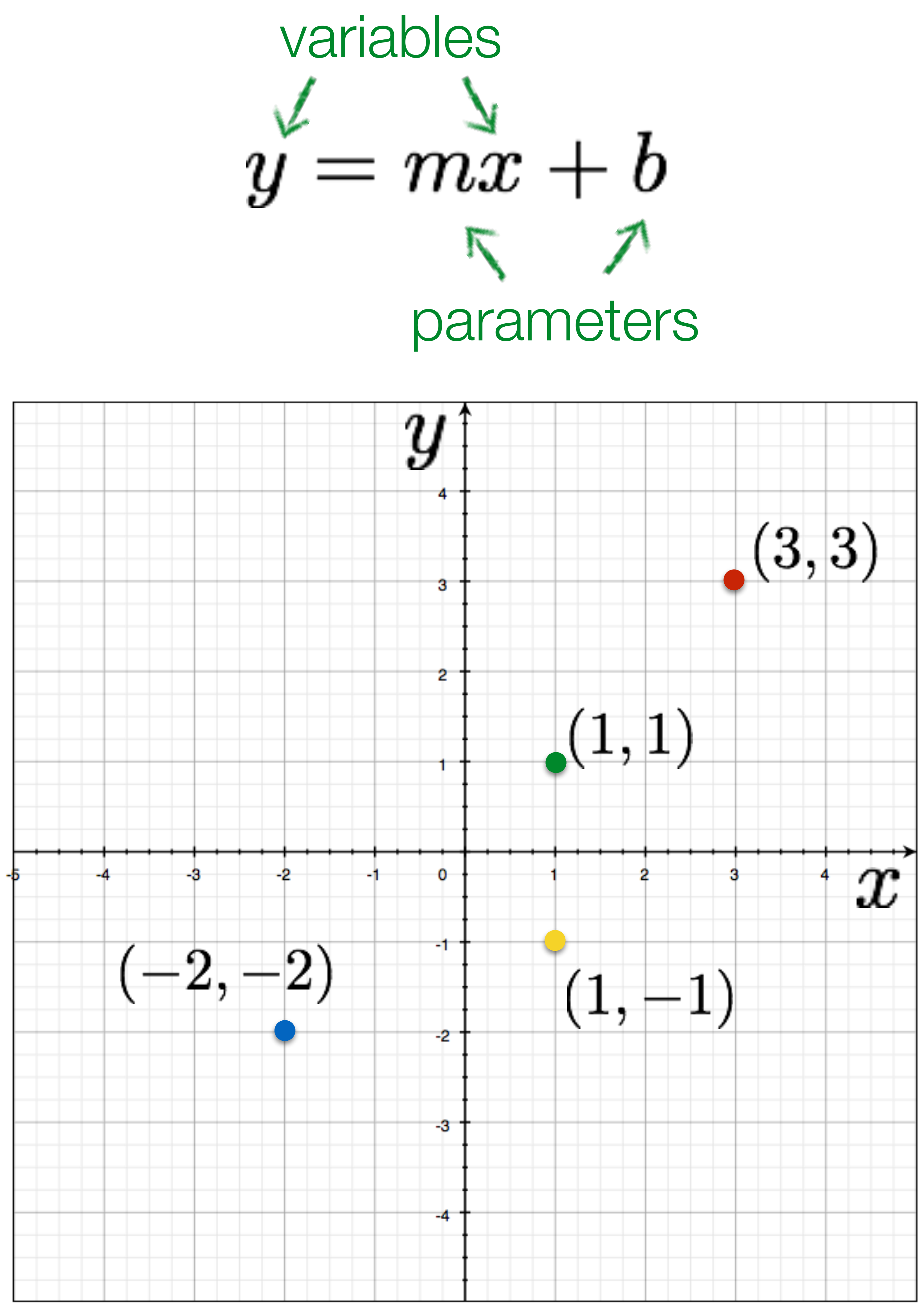
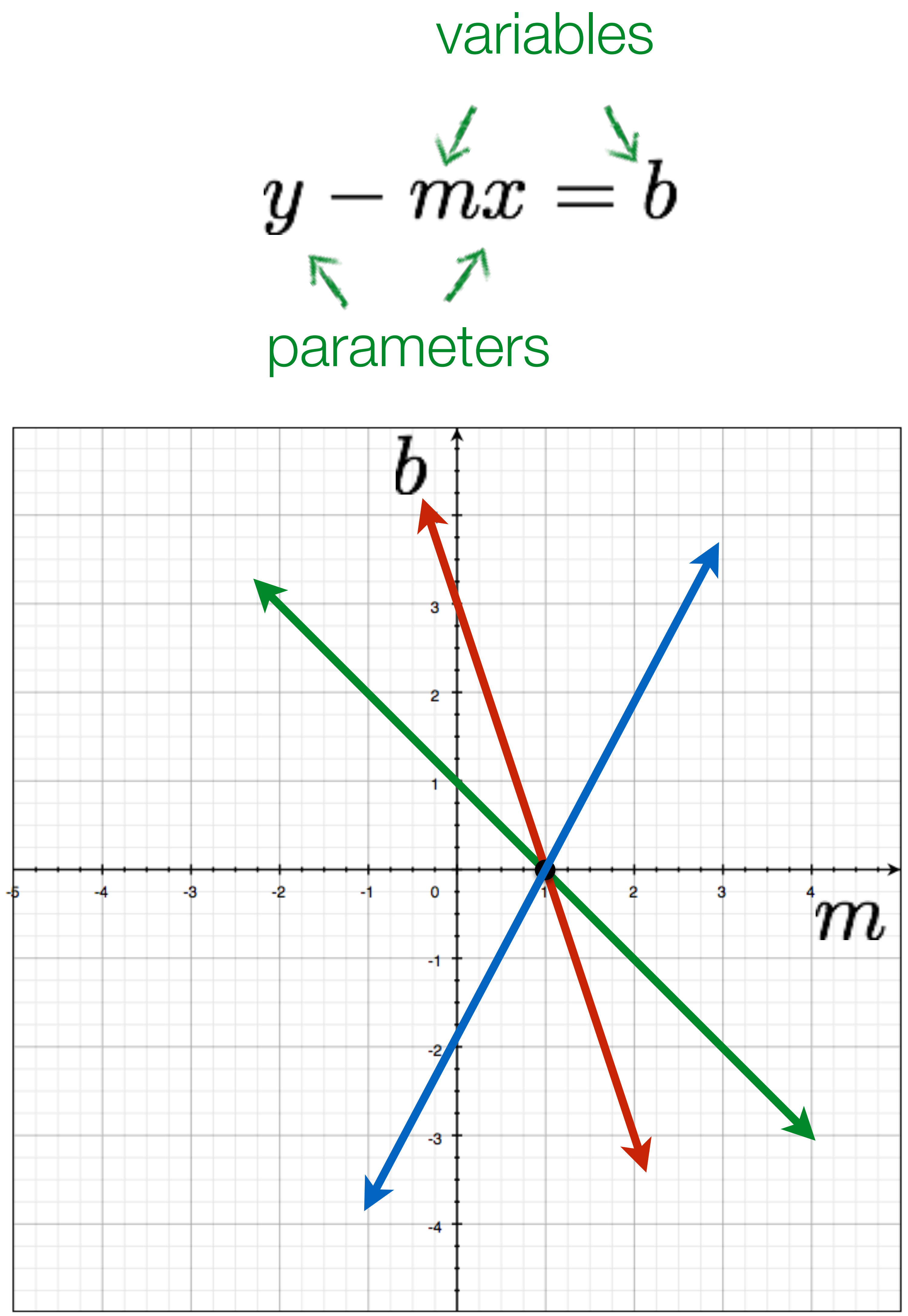


Image space

four points?



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

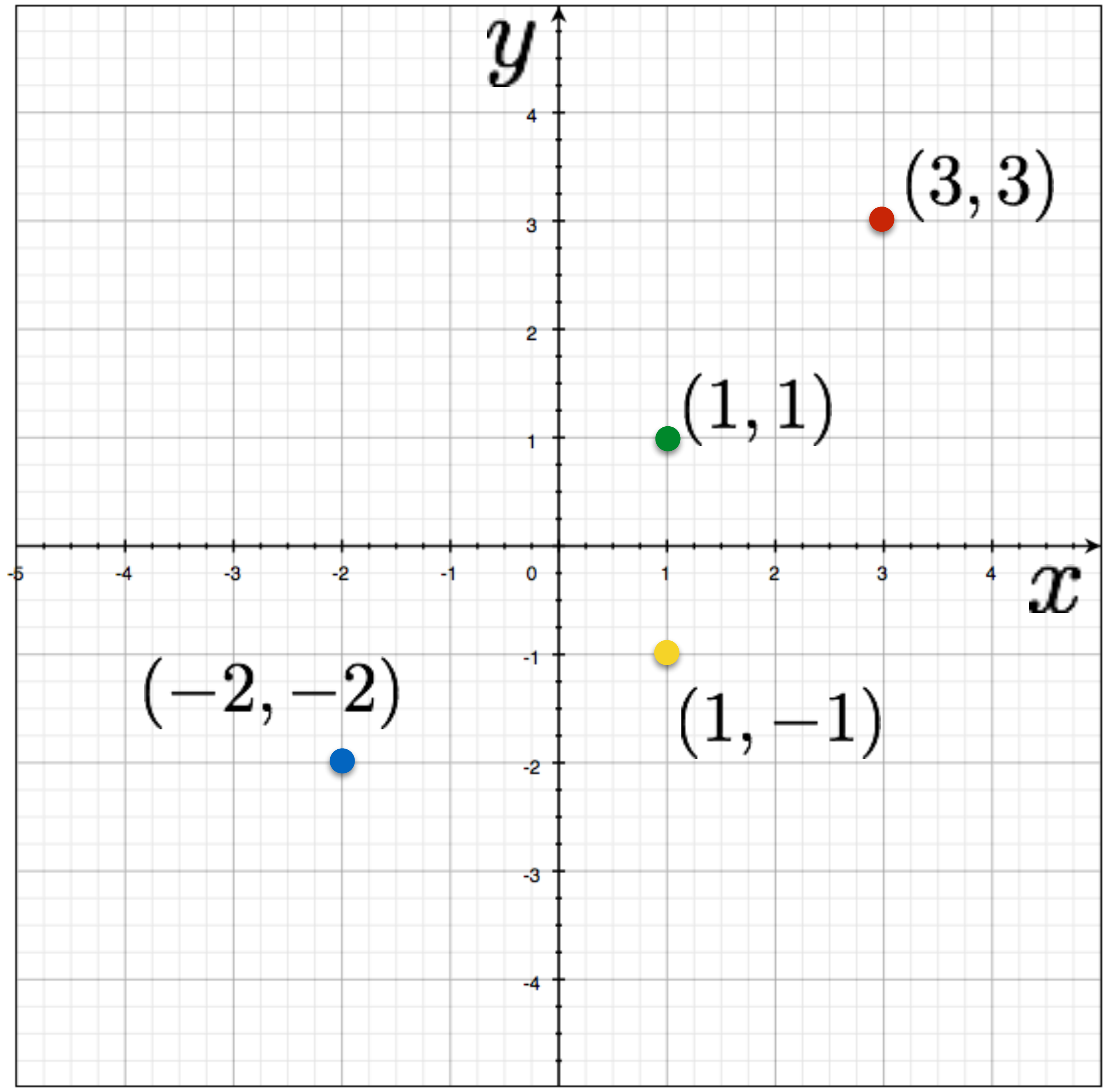


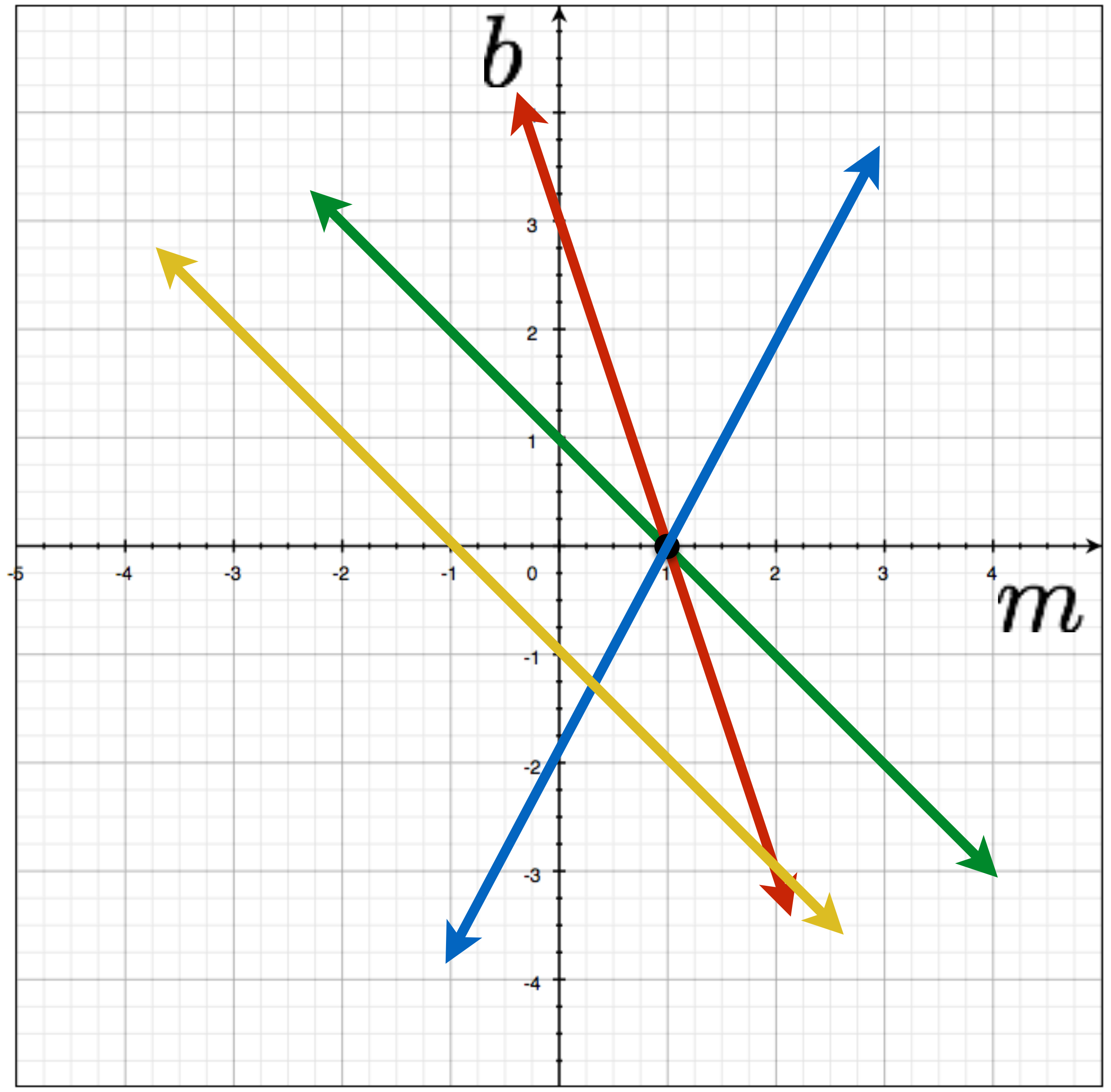
Image space

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?

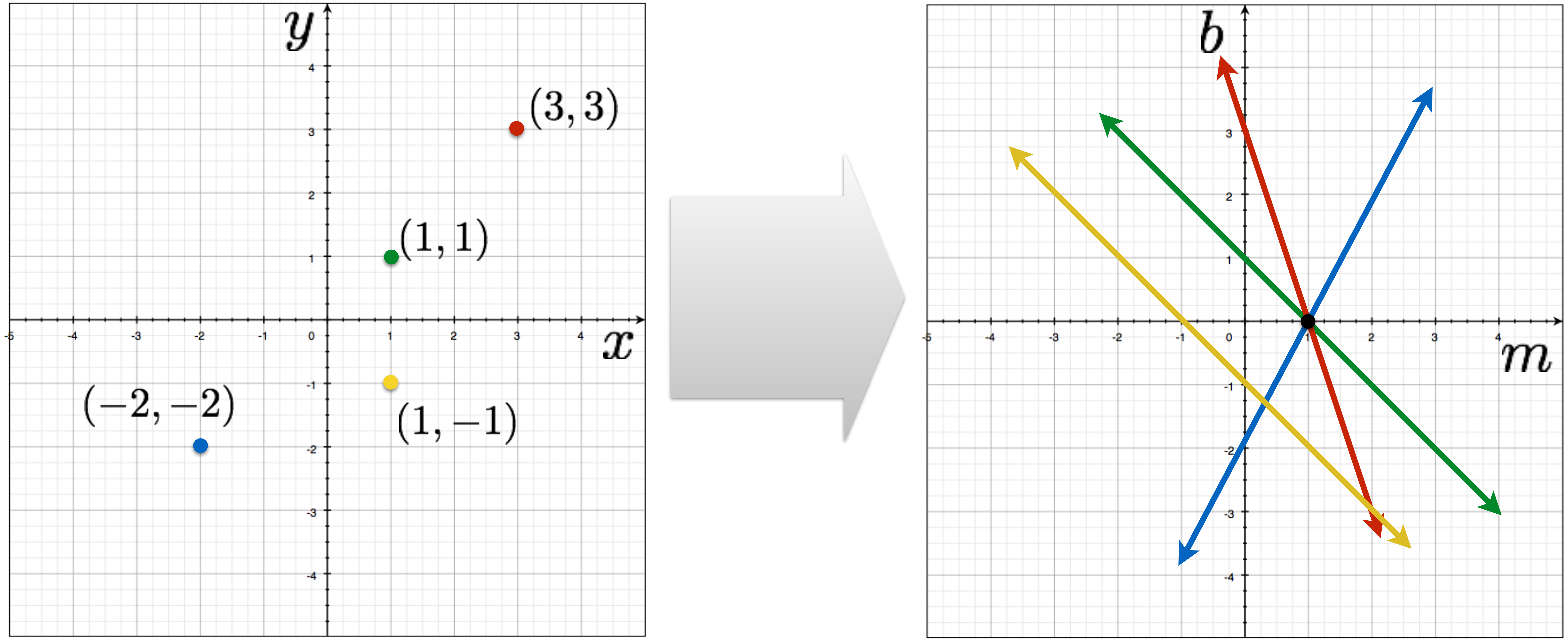


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

Is this method robust to measurement noise? clutter?

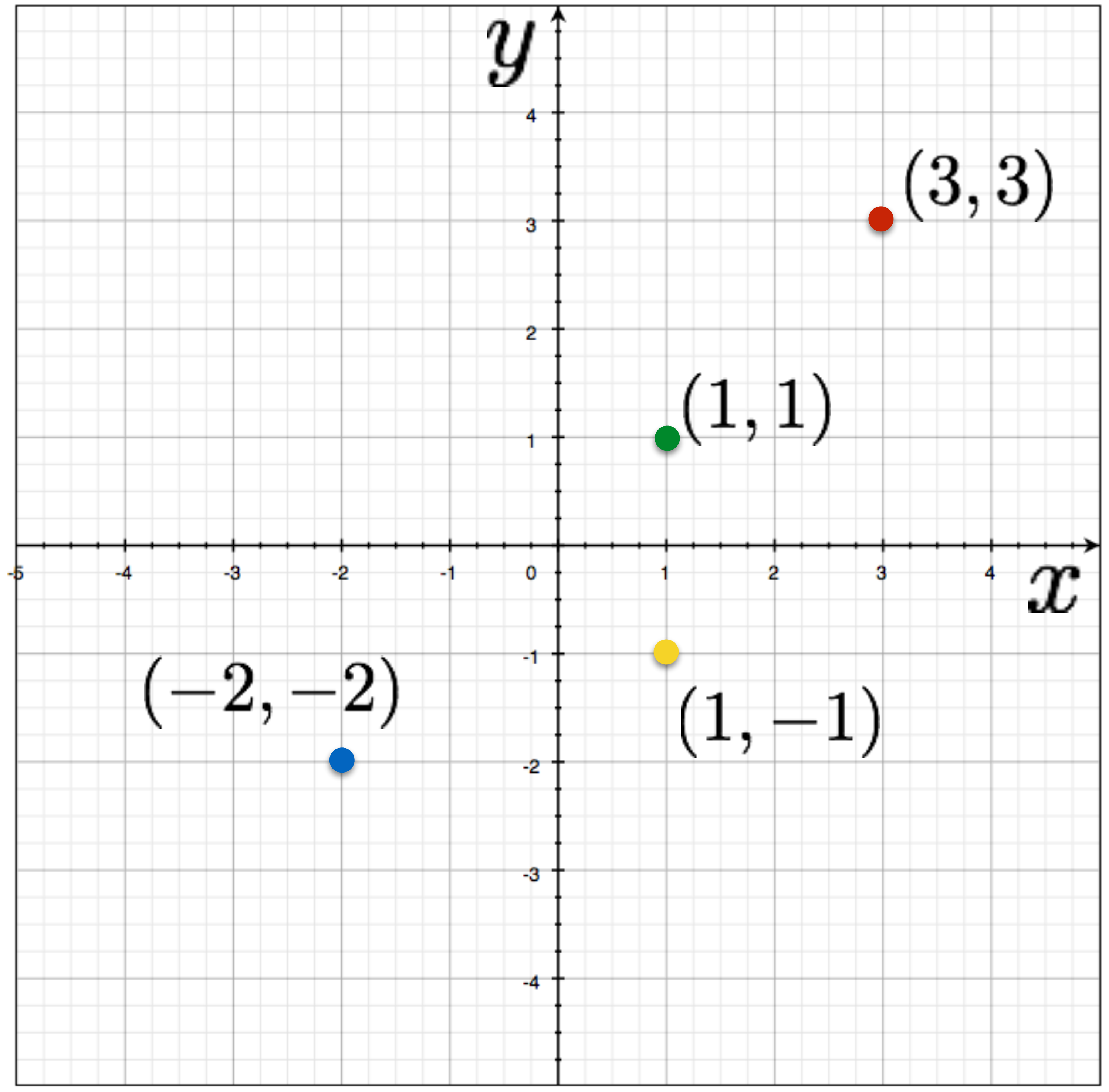
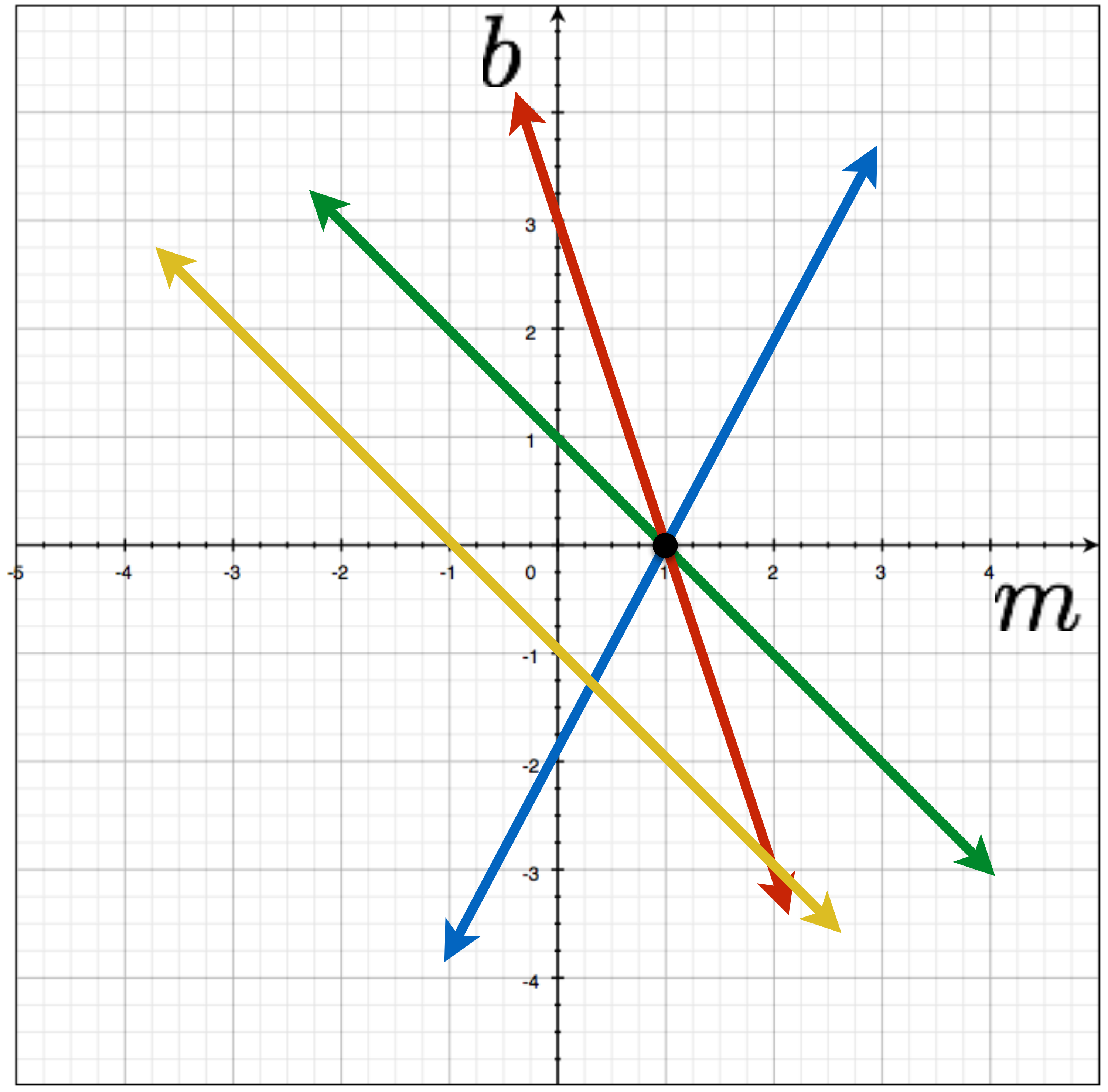


Image space



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Line Detection by Hough Transform

Algorithm:

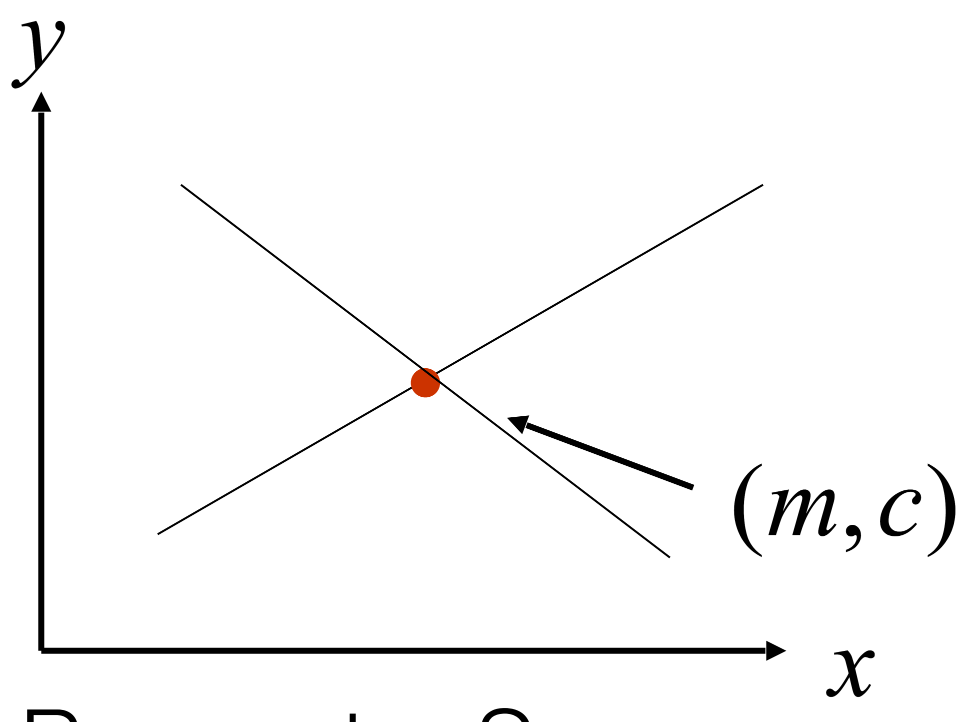
1. Quantize Parameter Space (m,c)

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!

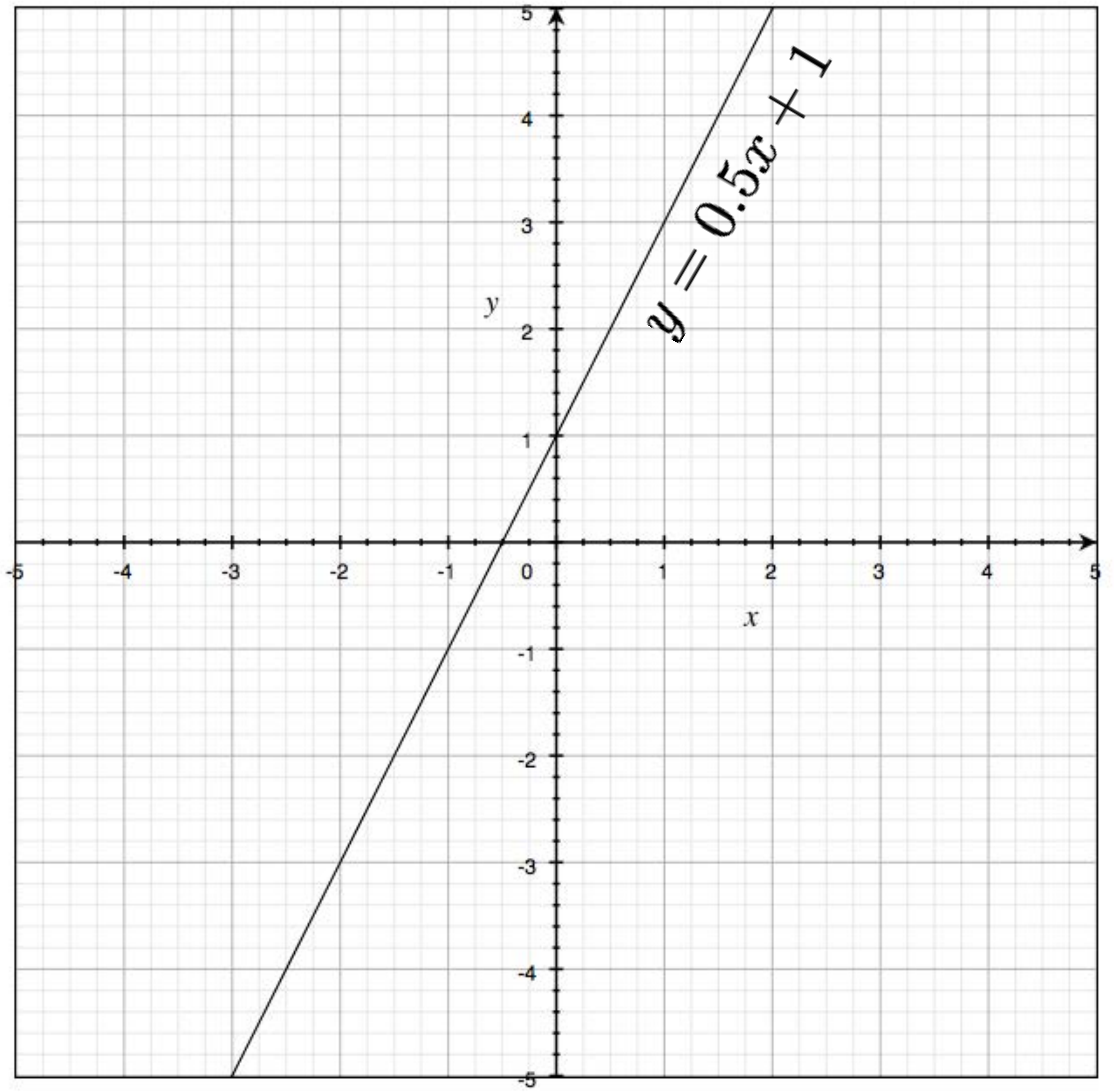
$$-\infty \leq c \leq \infty$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lines: Slope intercept form

$$y = mx + b$$

slope y-intercept



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lines: Normal form

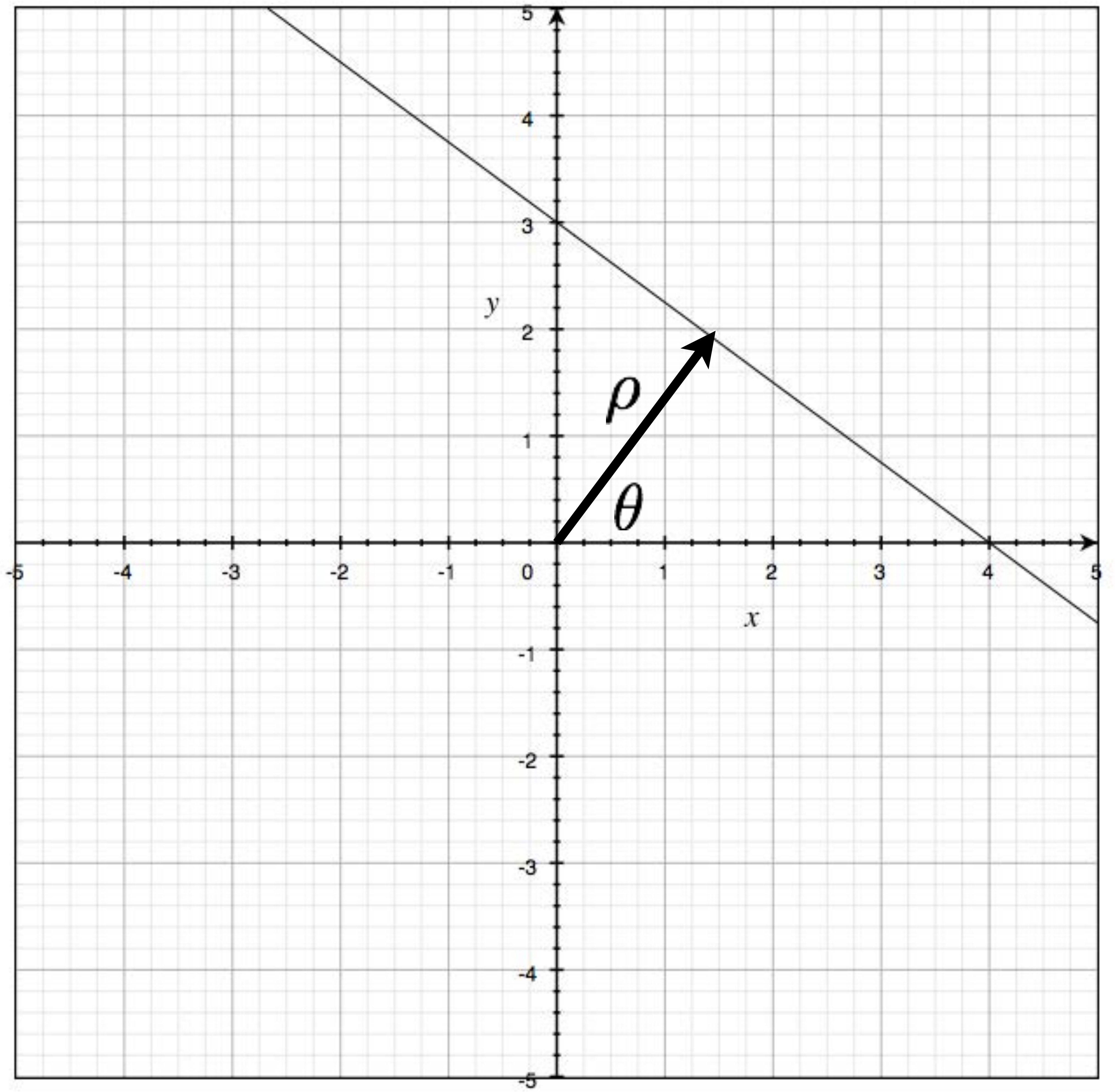
$$x\cos(\theta) + y\sin(\theta) = \rho$$

Forsyth/Ponce convention

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

$$r \geq 0$$

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



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

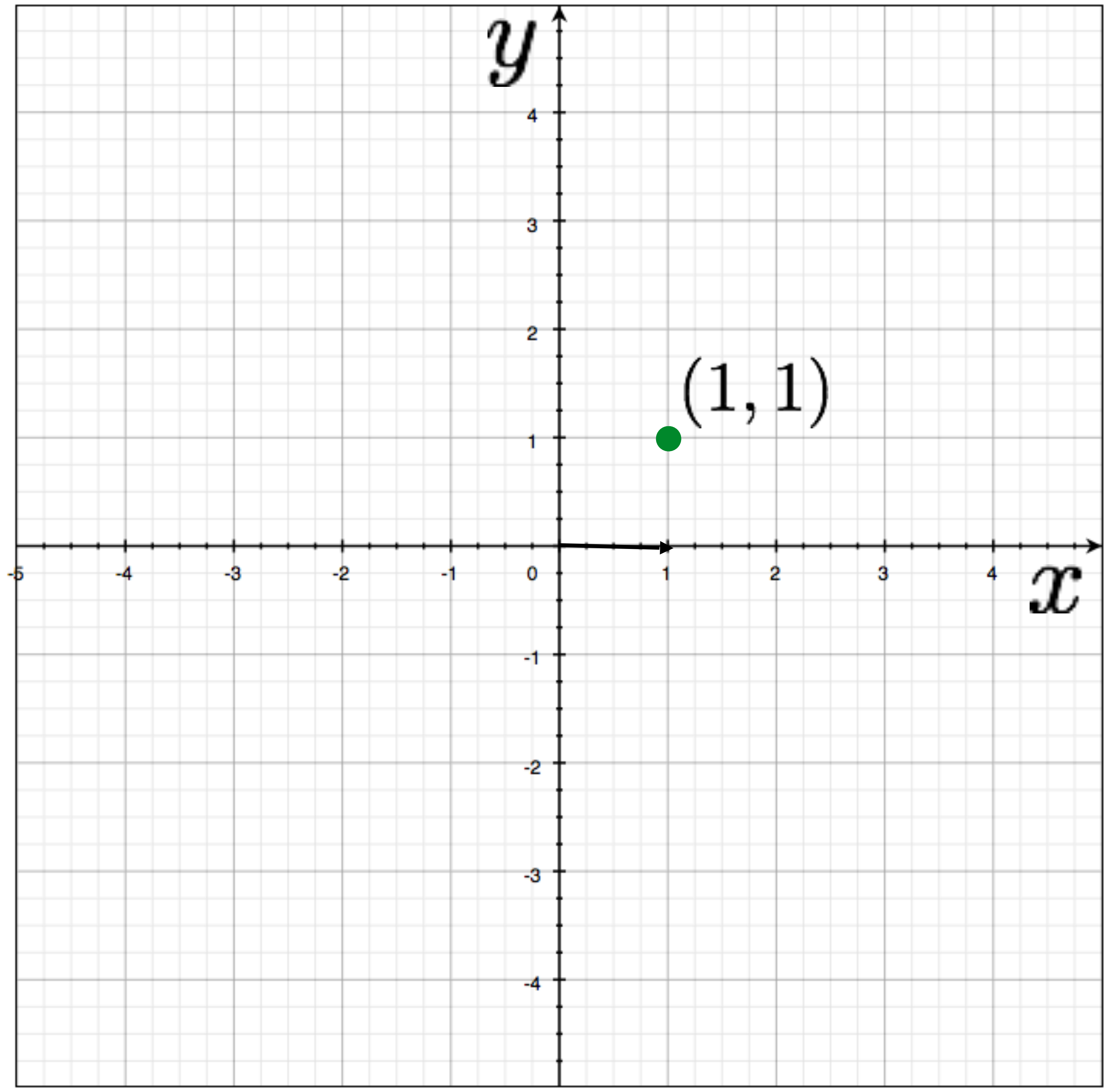
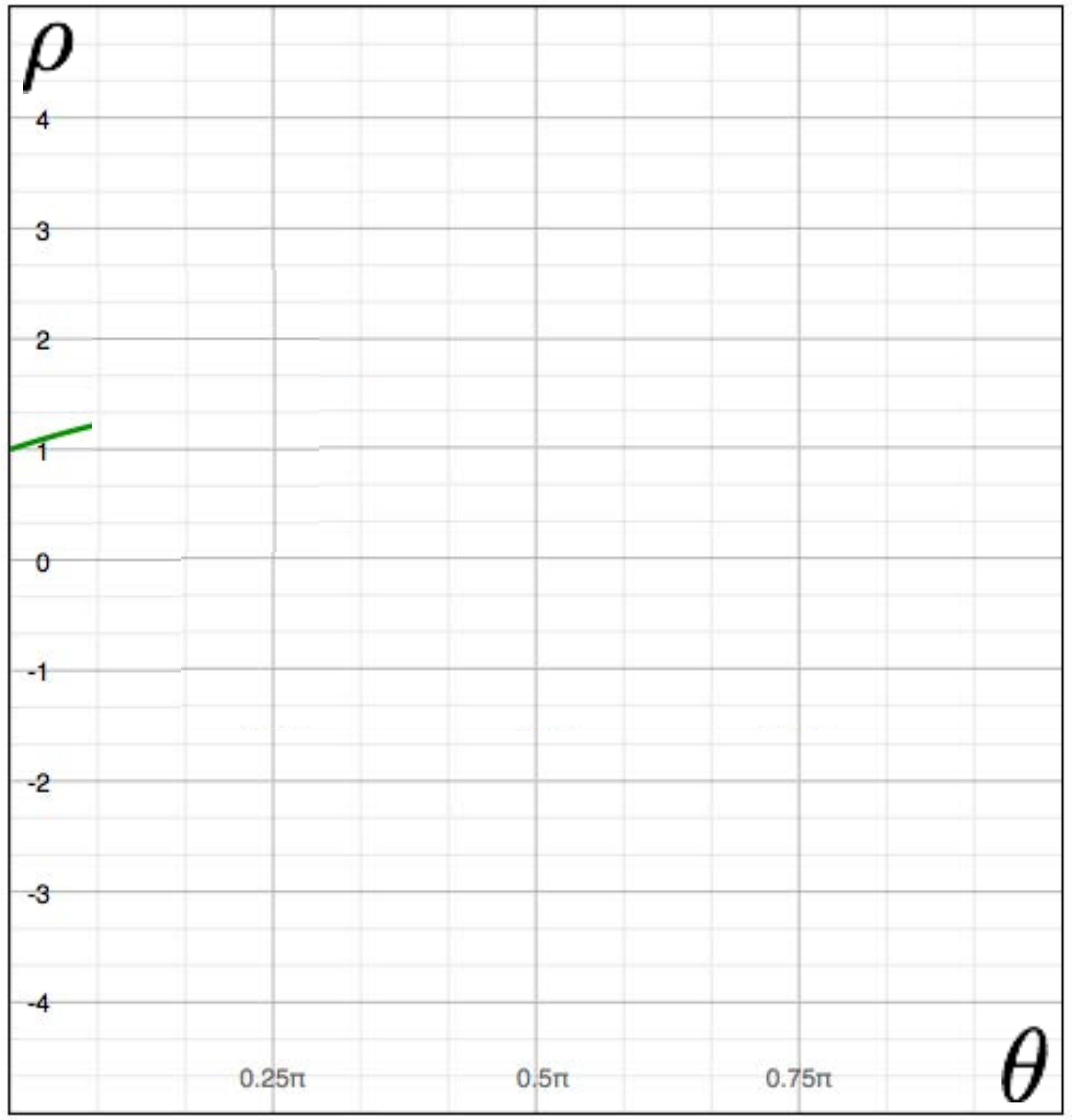


Image space

a point becomes?



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

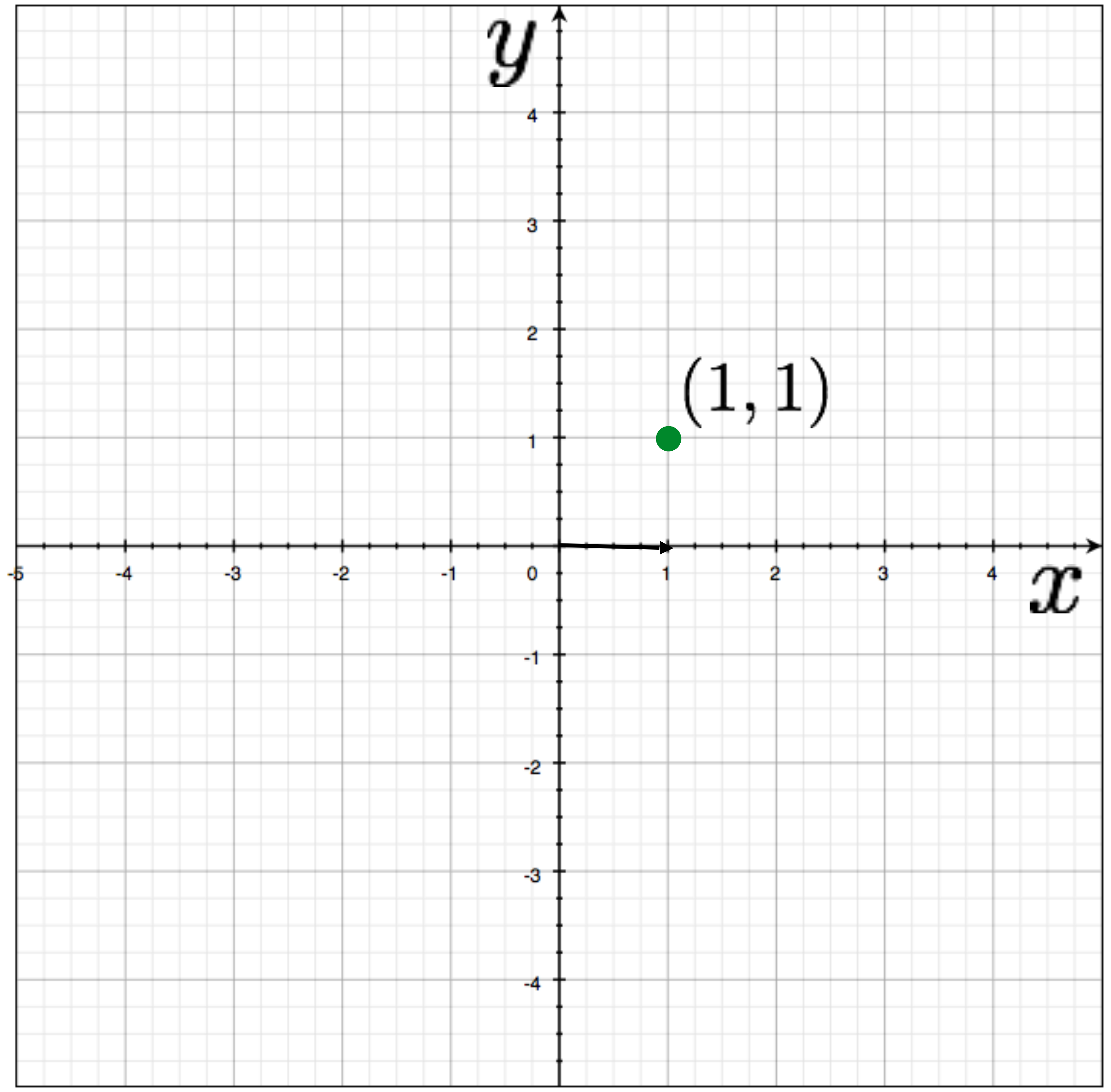


Image space

a point
becomes
a wave

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

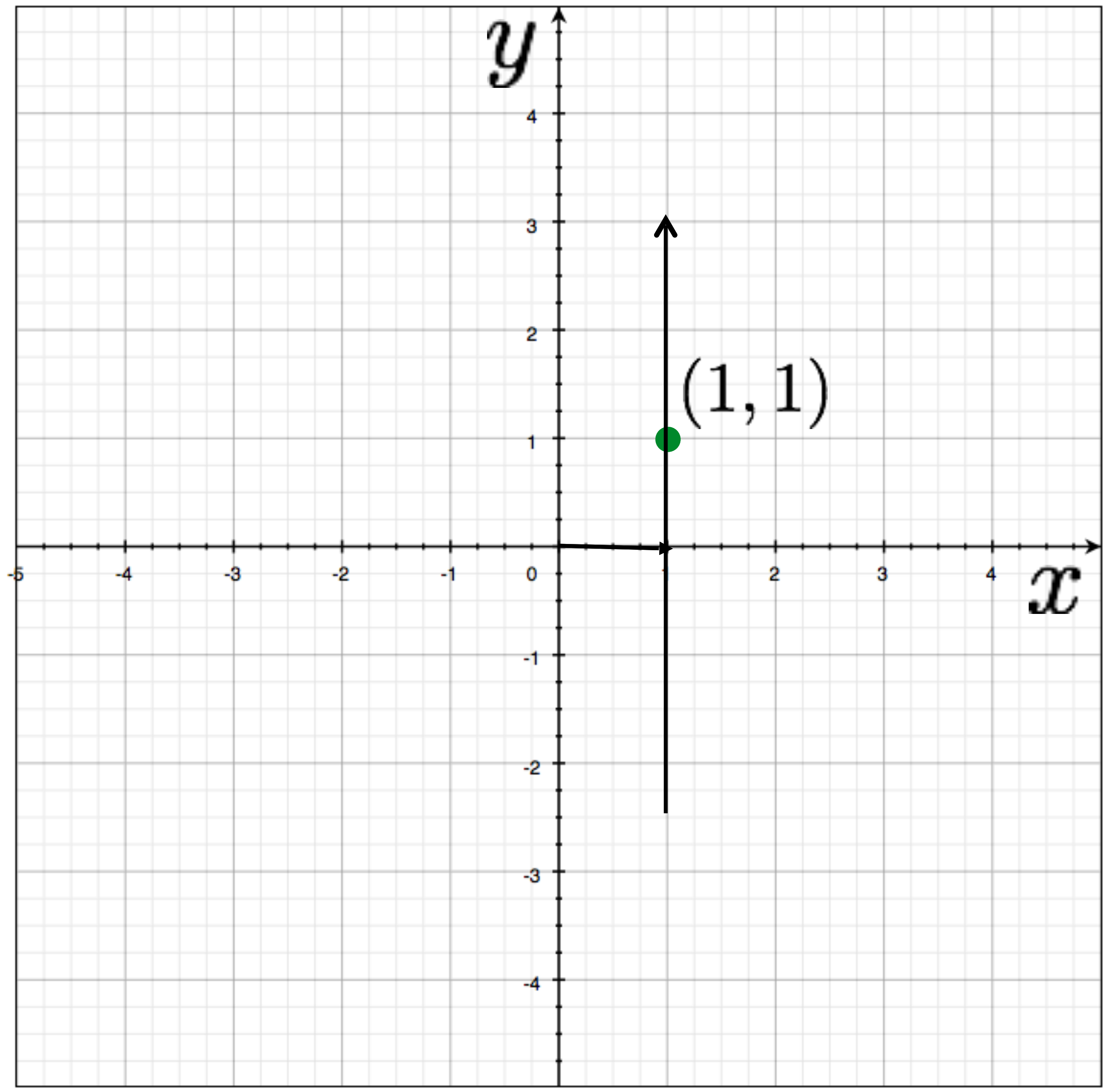


Image space

a line becomes?

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

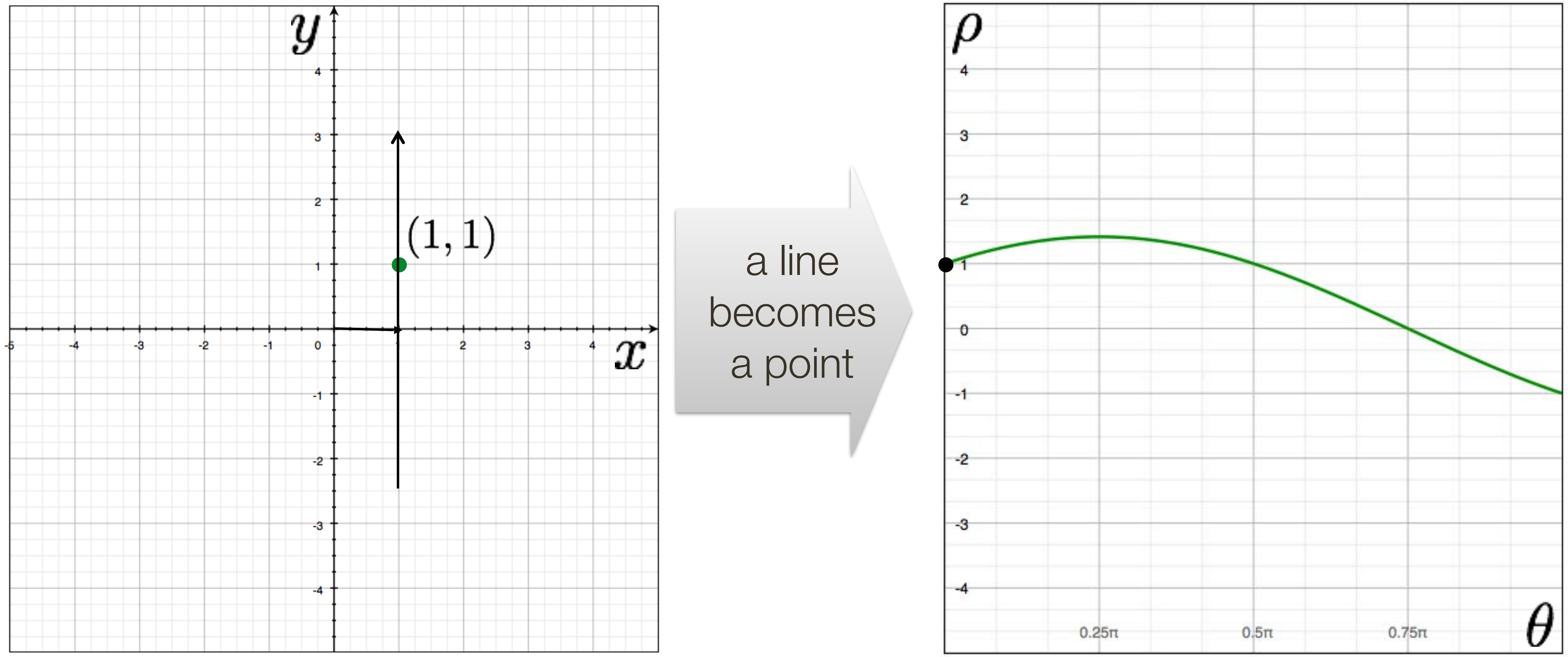


Image space

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

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

— 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

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

— 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

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

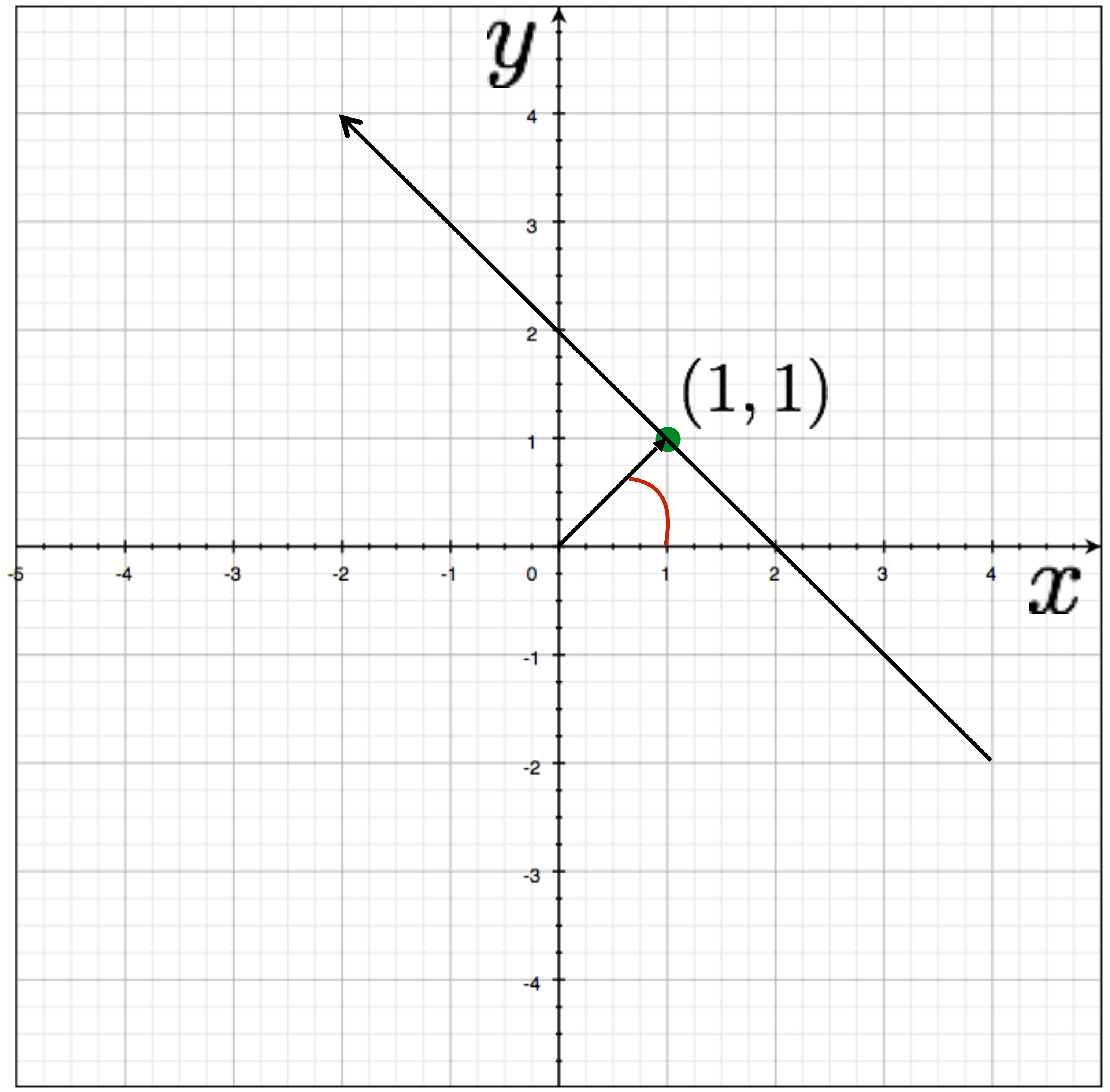


Image space

a line
becomes?

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

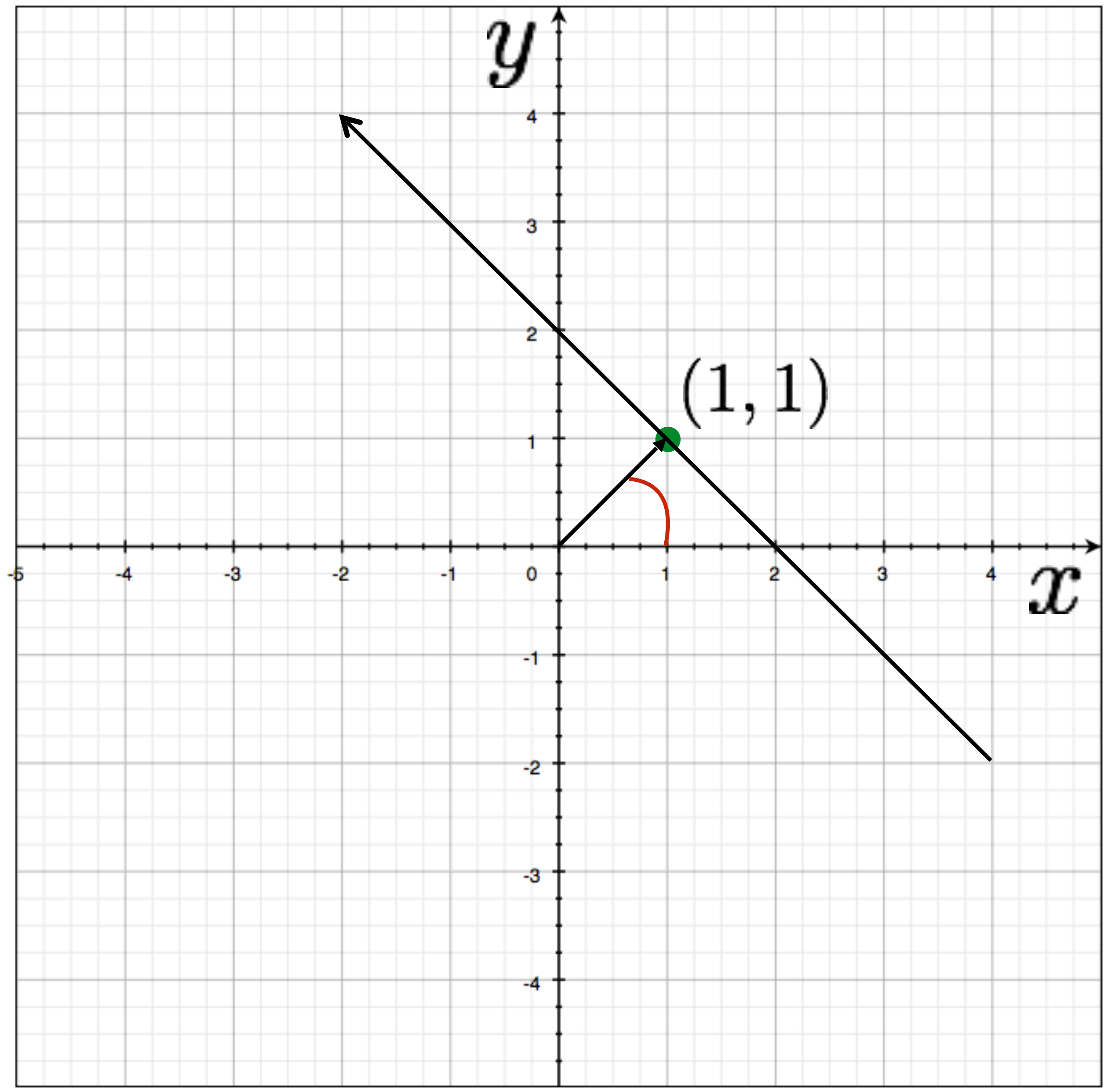
$$y = mx + b$$

parameters

parameters

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

variables



a line becomes a point

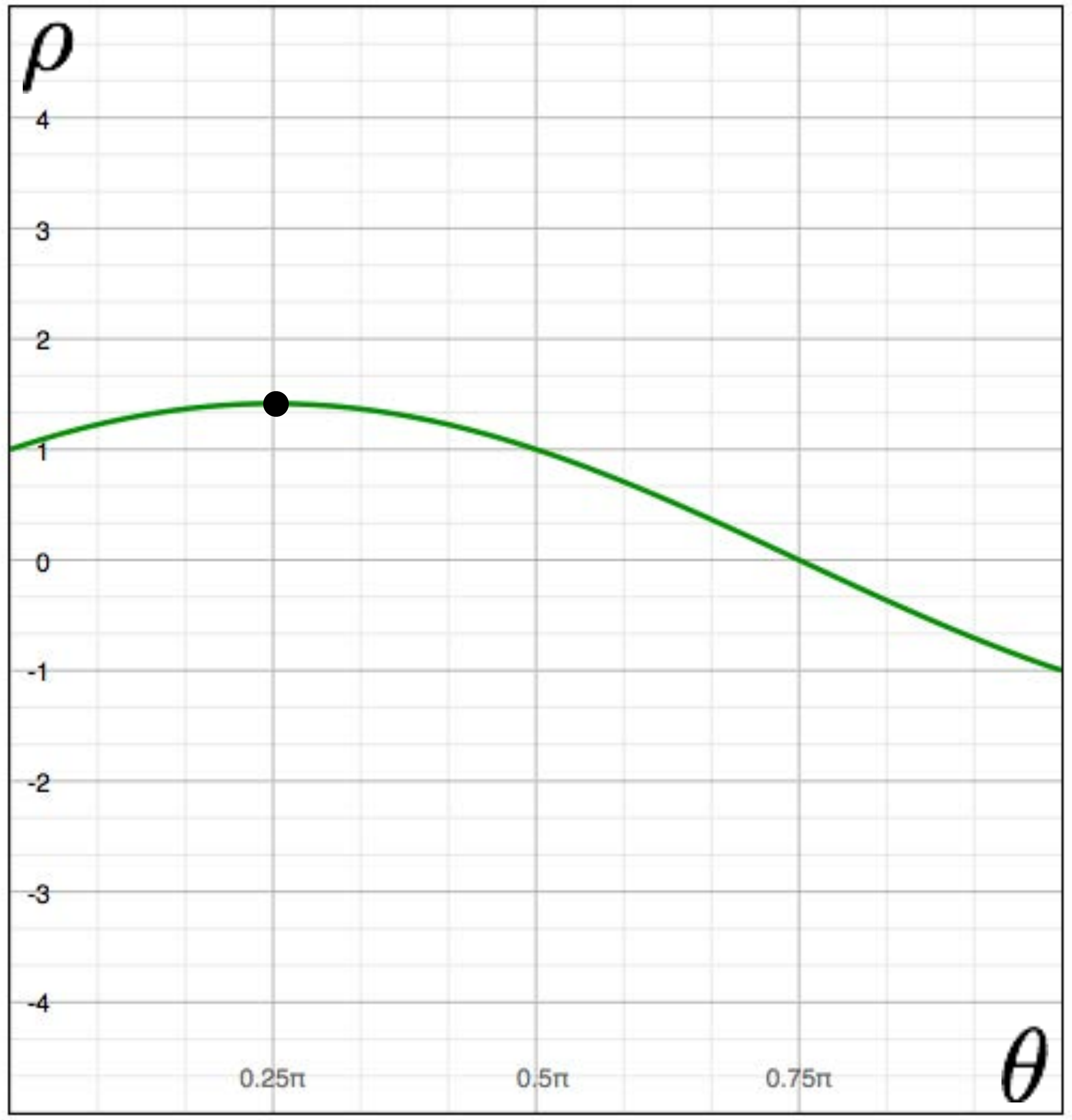


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

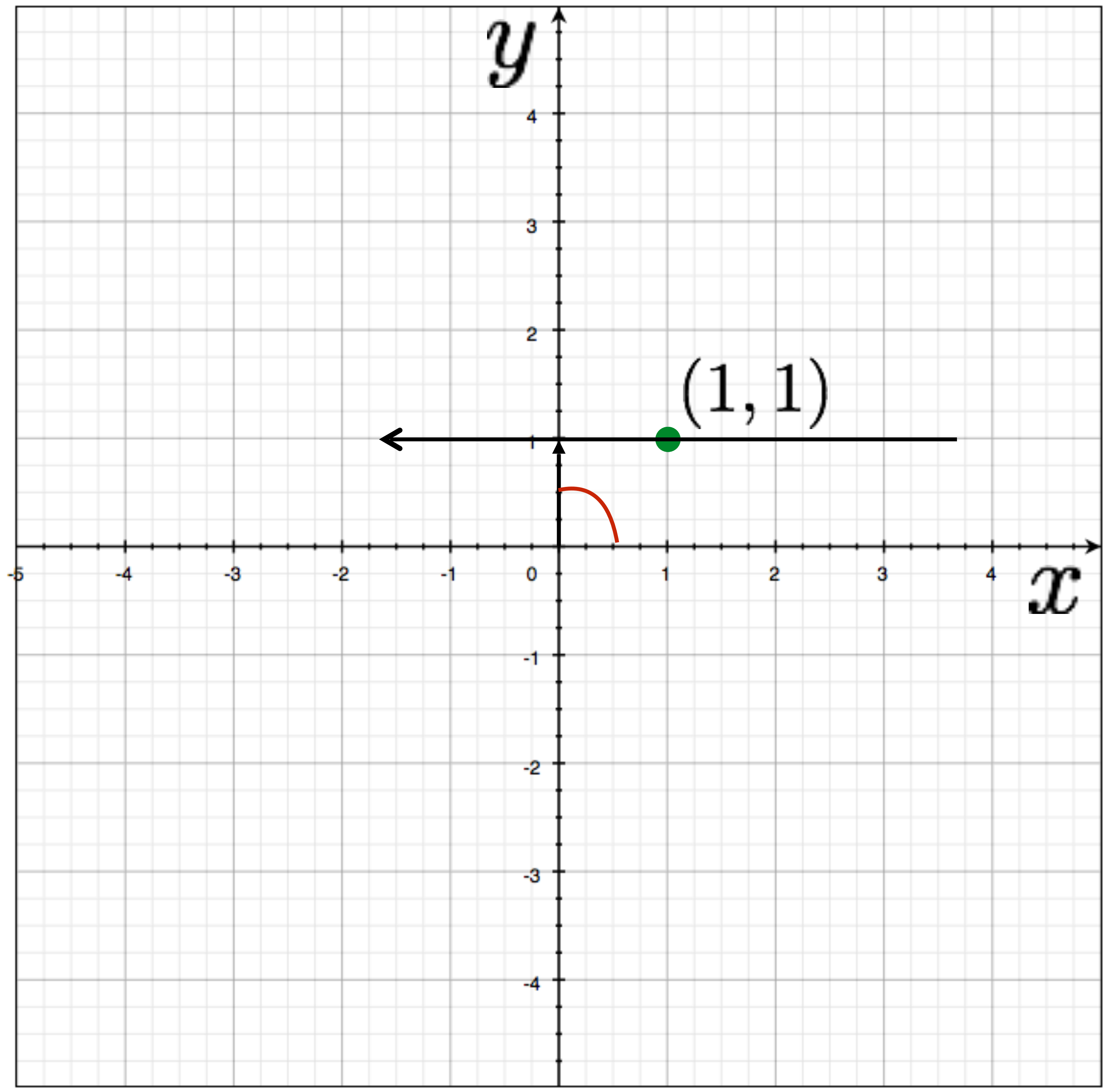
$$y = mx + b$$

parameters

parameters

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

variables



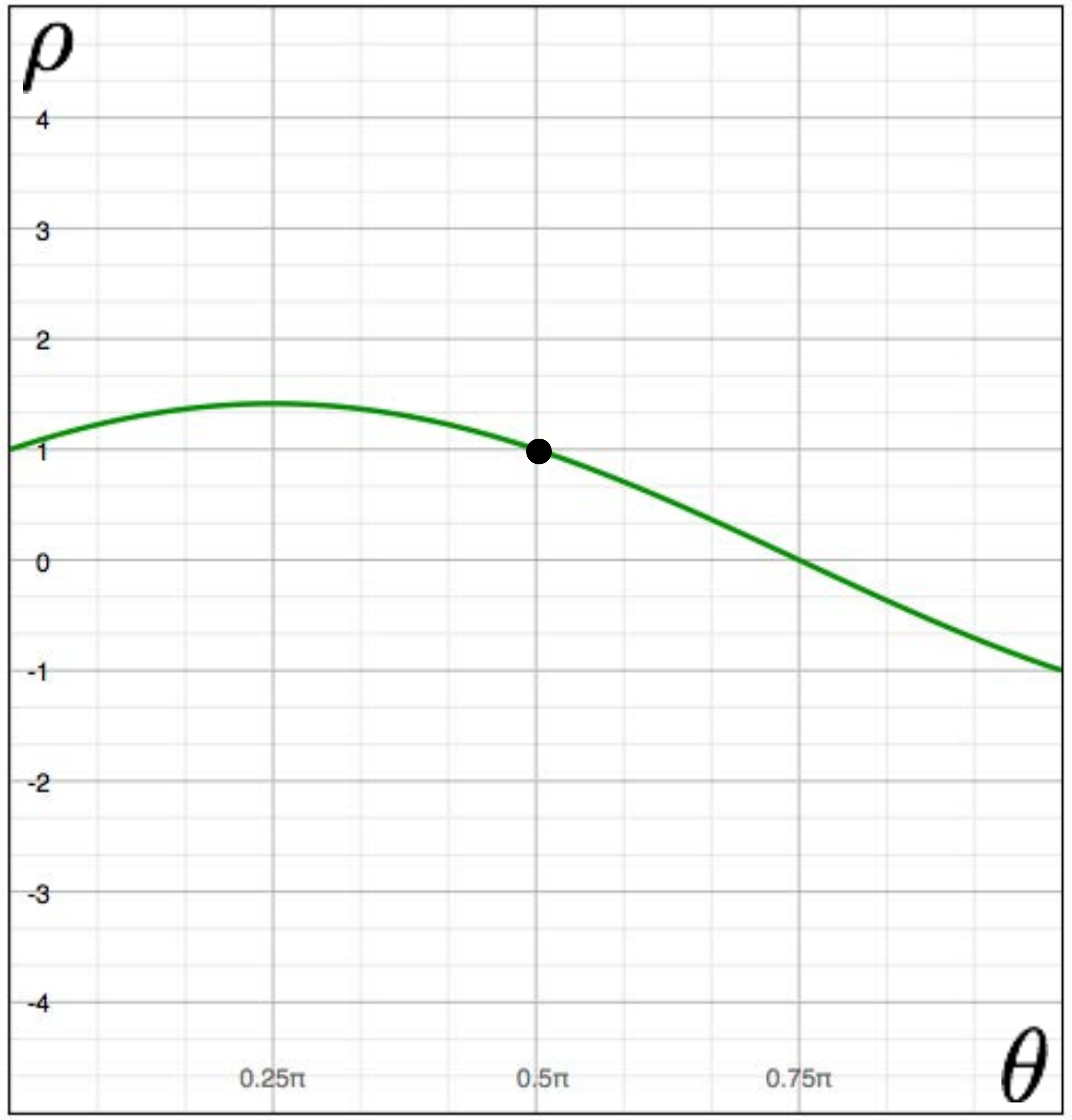


Image space

Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

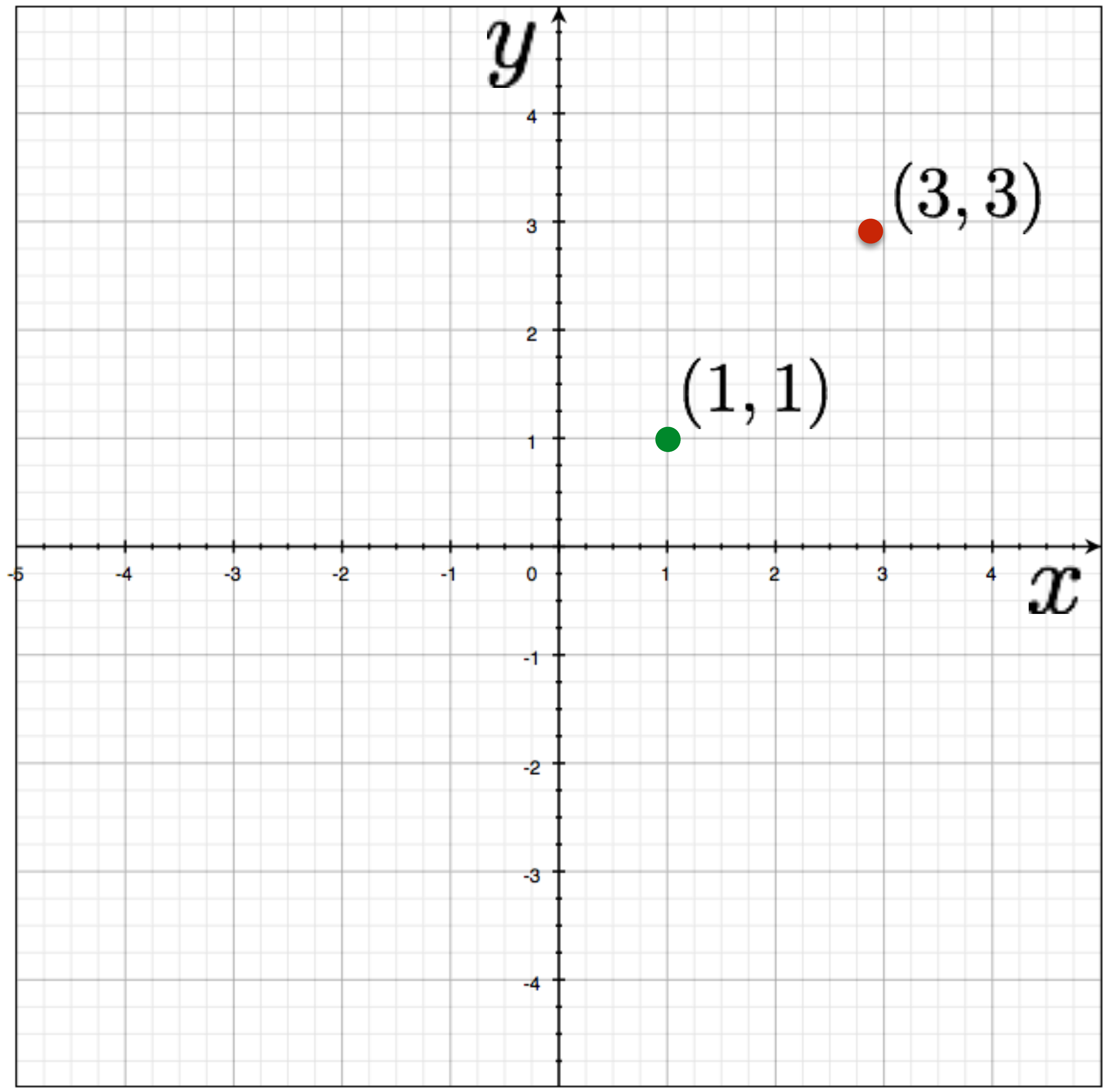


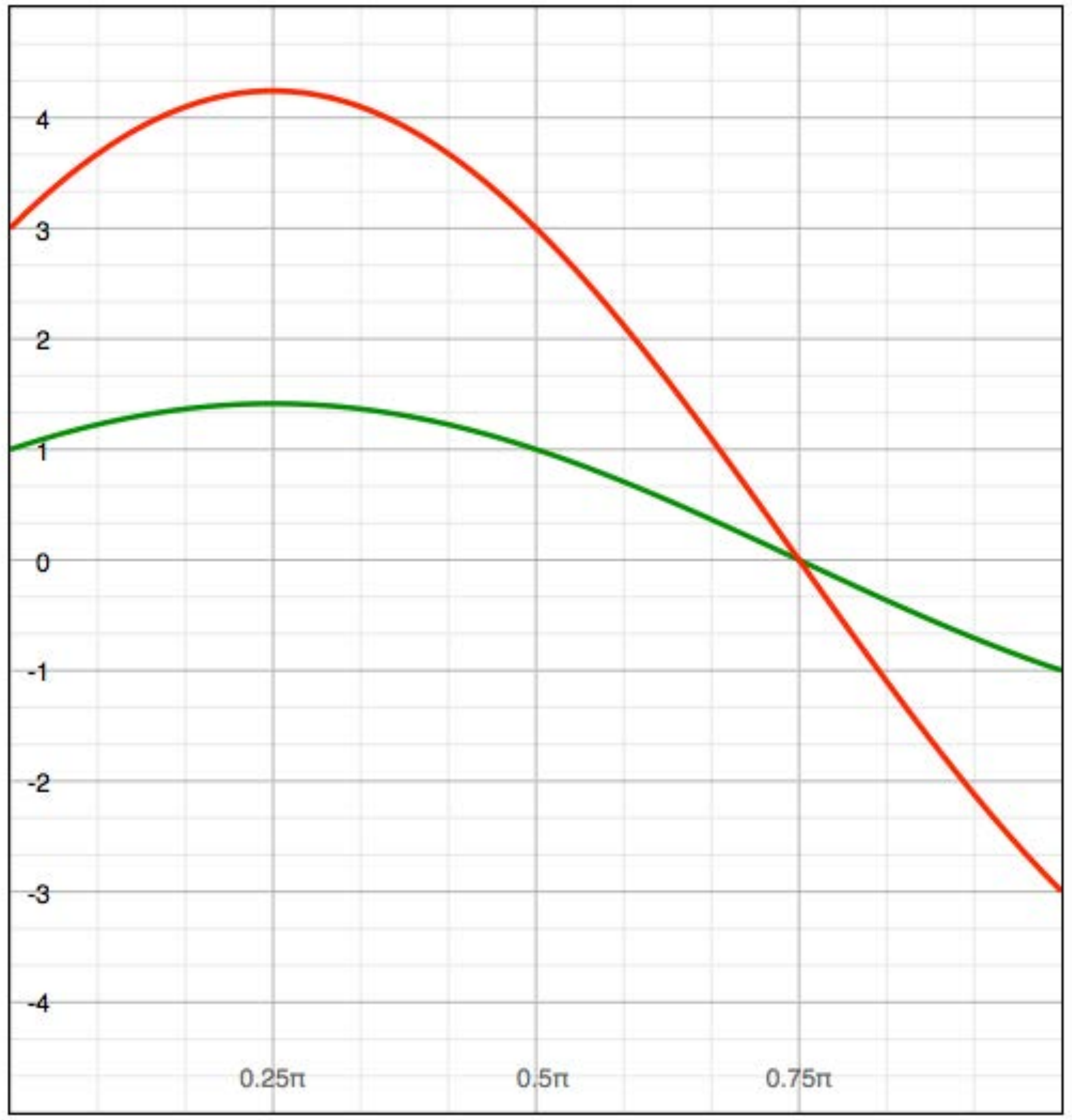
Image space

two points become?

parameters

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

variables



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

parameters

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

variables

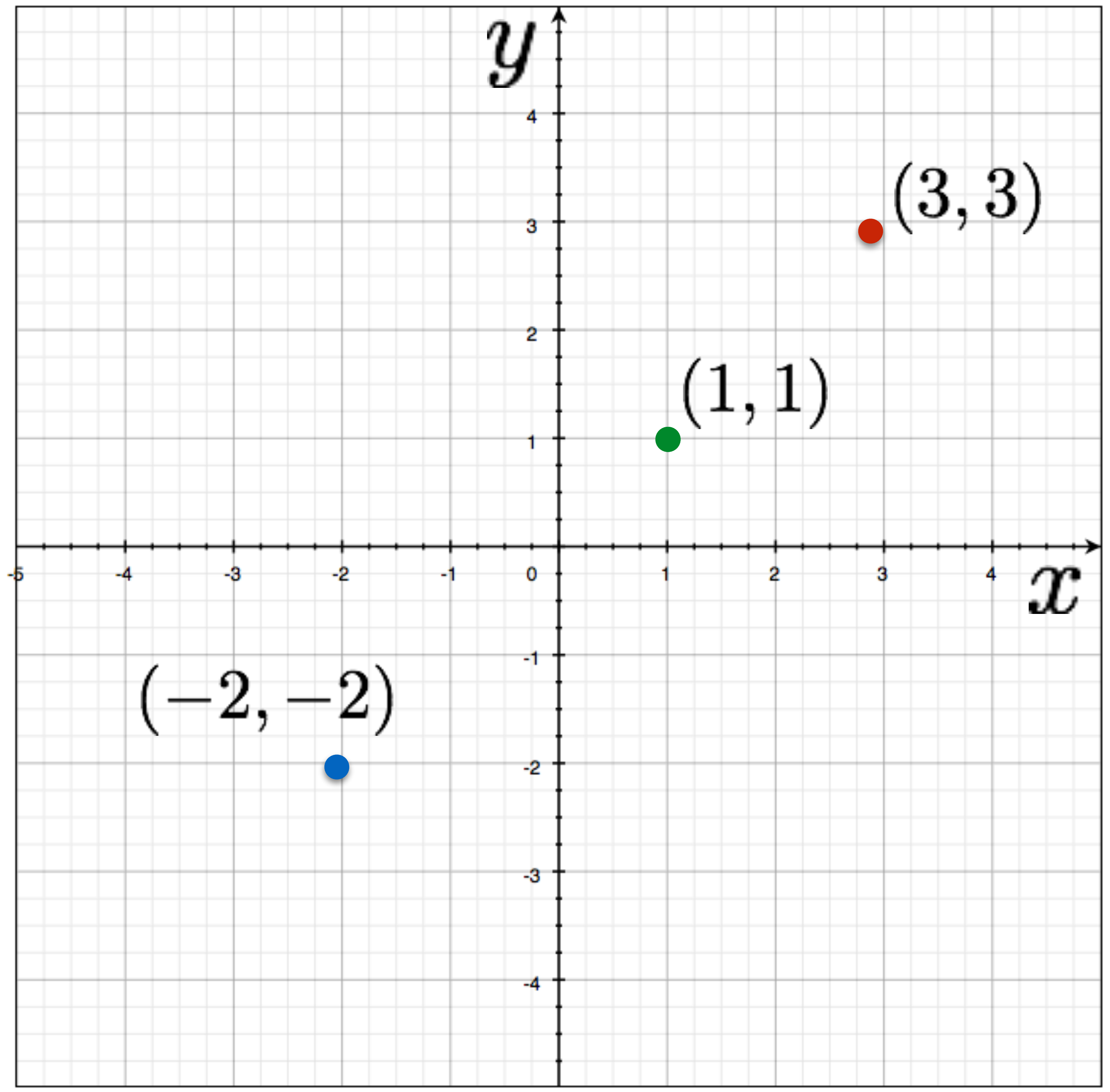
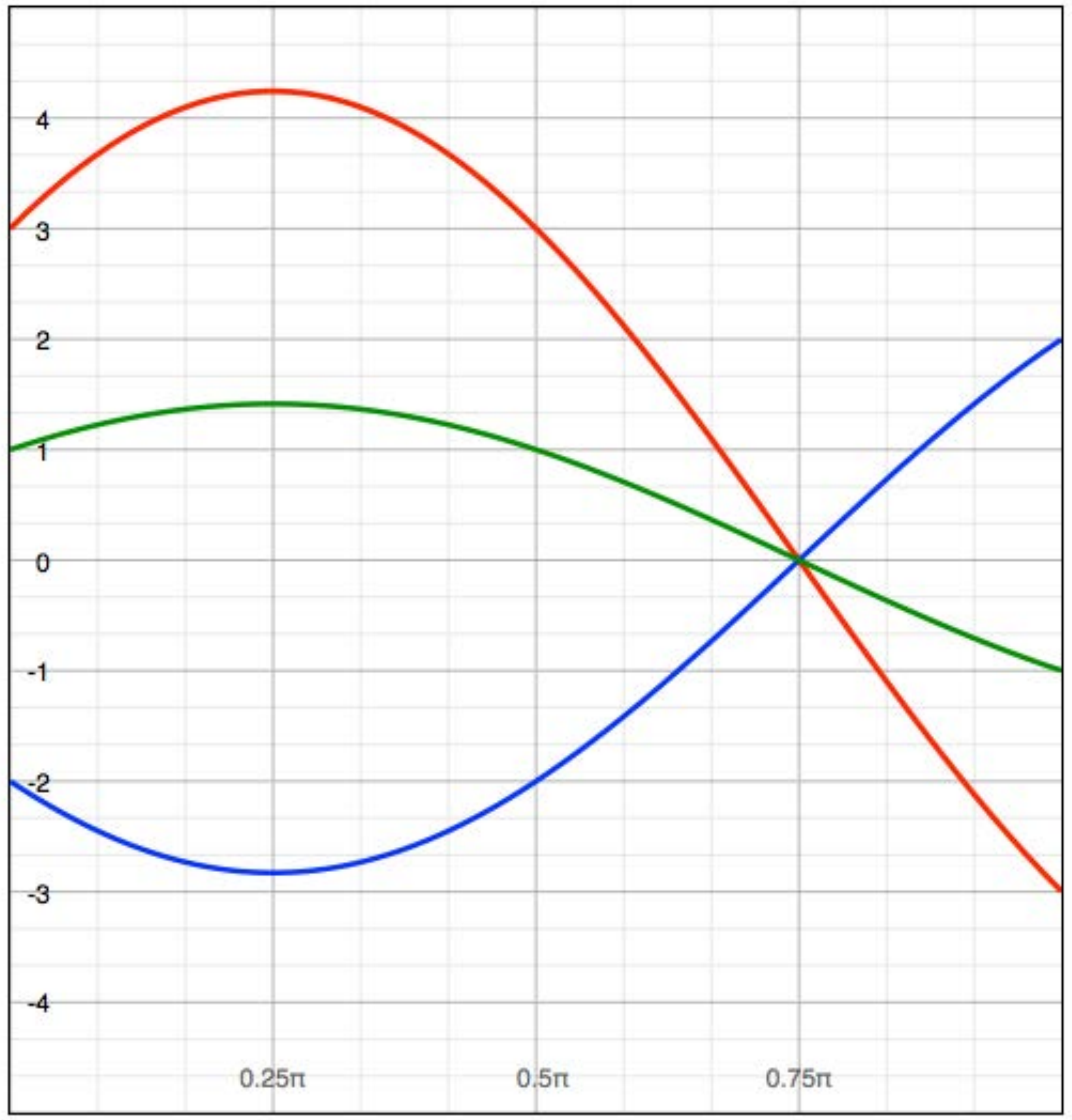


Image space

three points become?



Parameter space

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Hough Transform: Lines

variables

$$y = mx + b$$

parameters

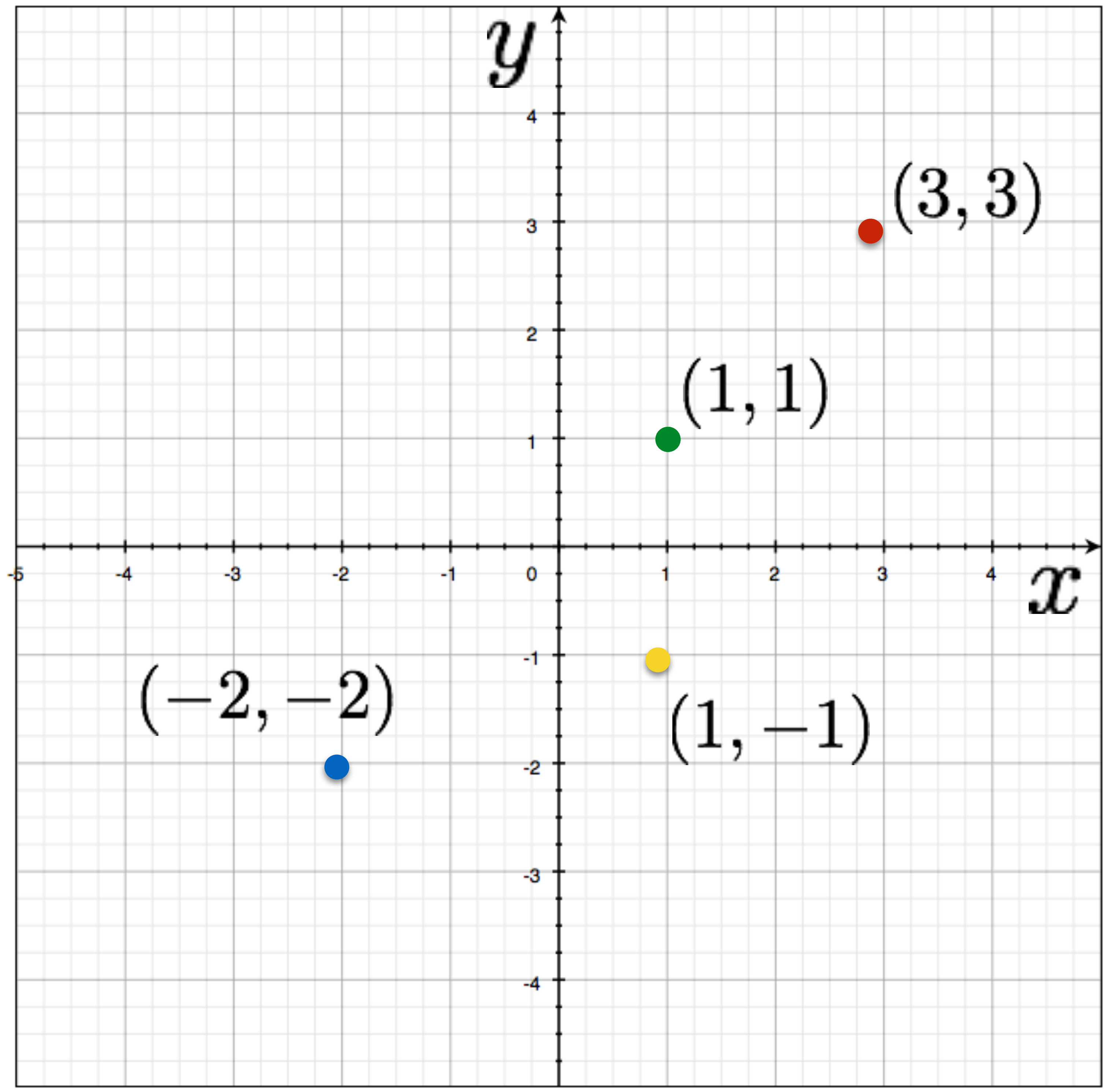


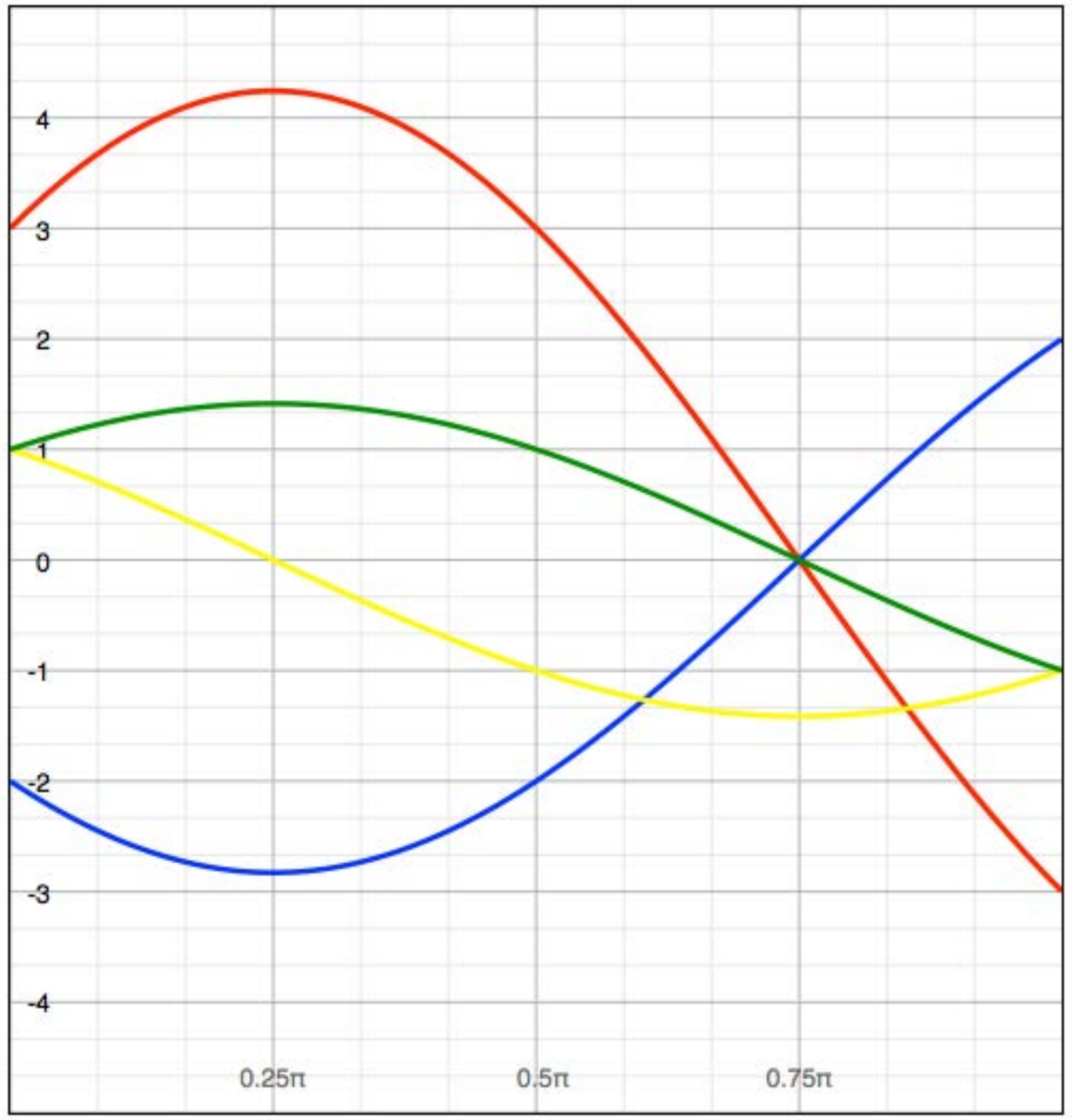
Image space

four points become?

parameters

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

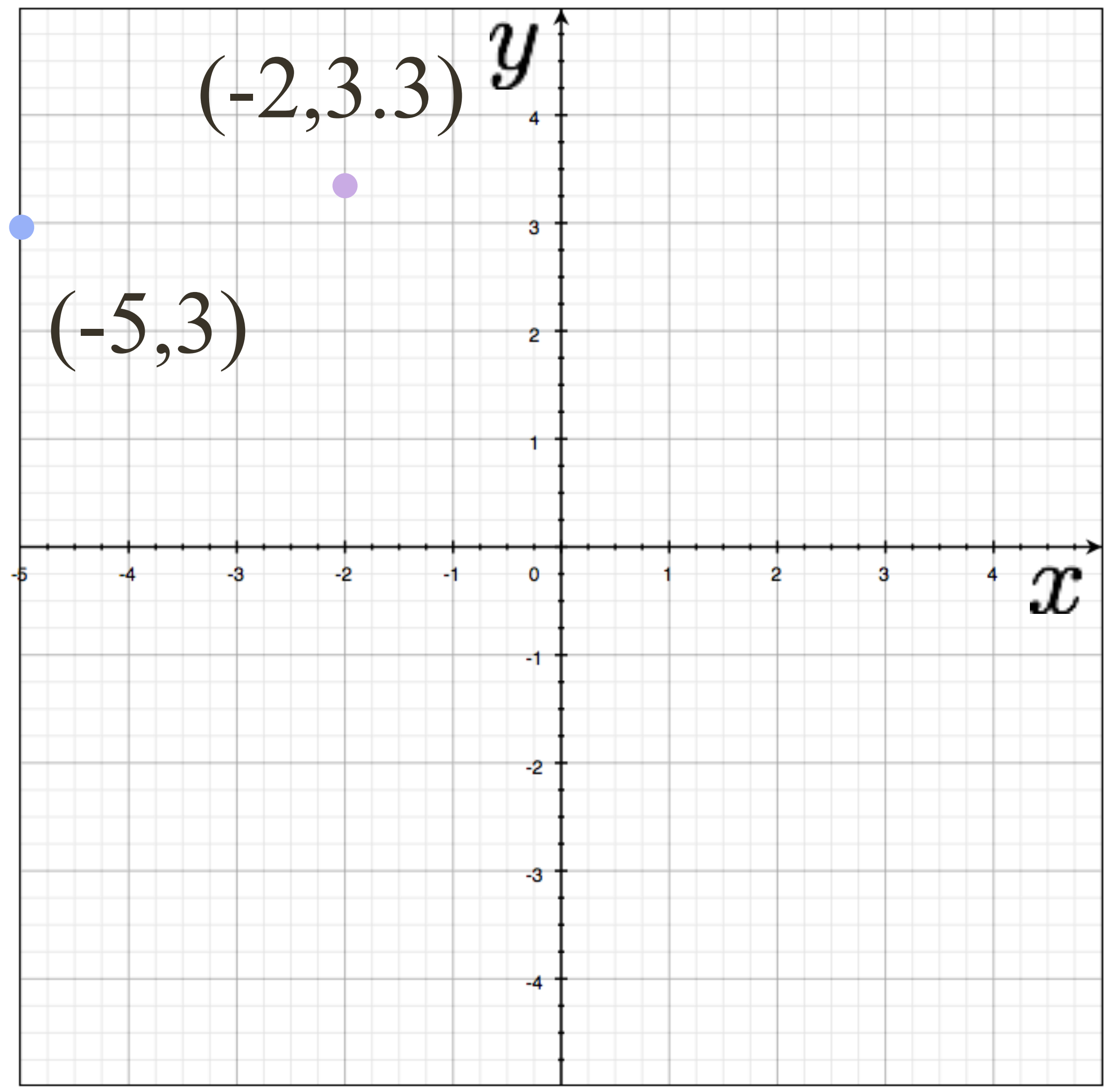
variables

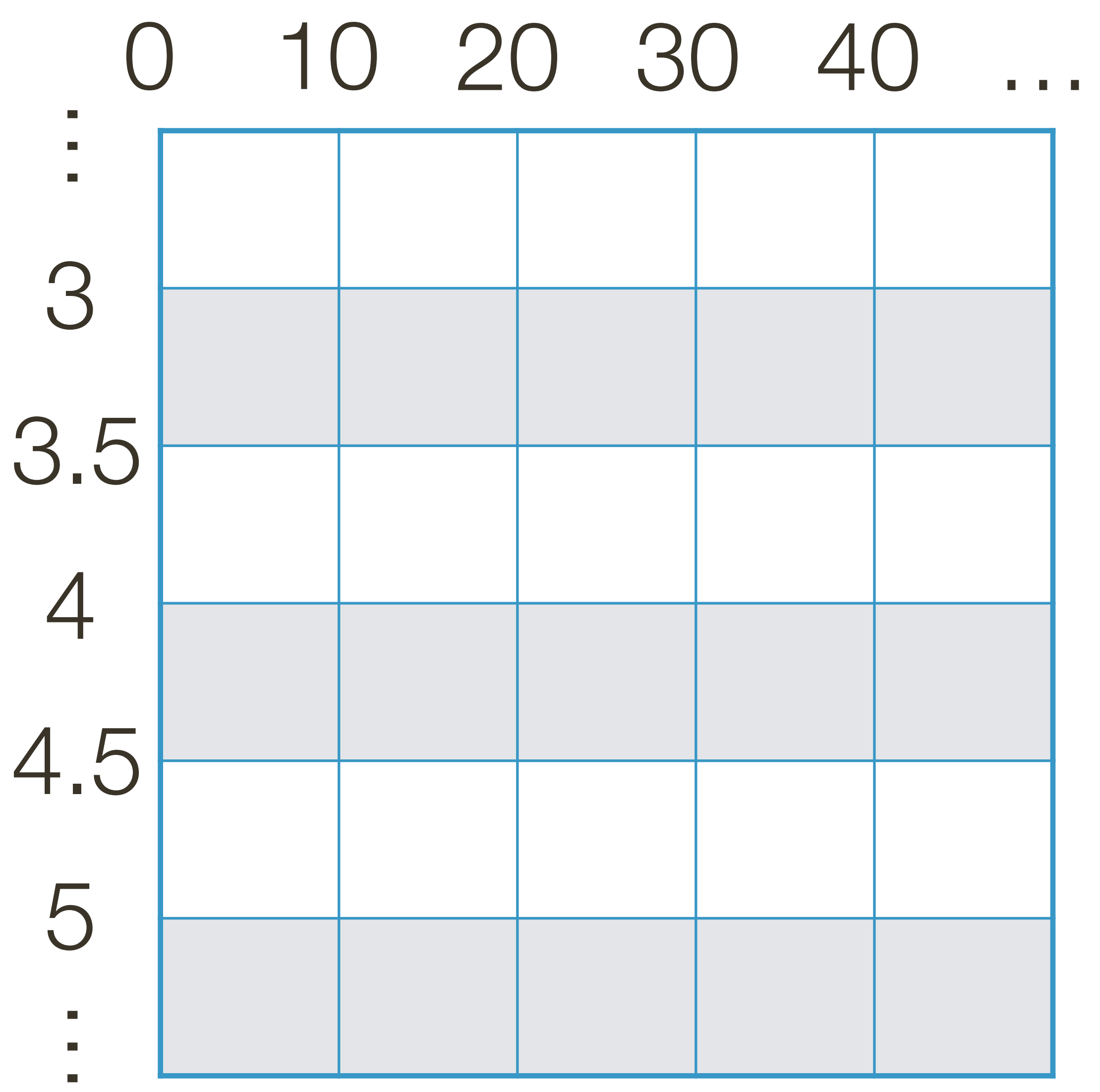


Parameter space

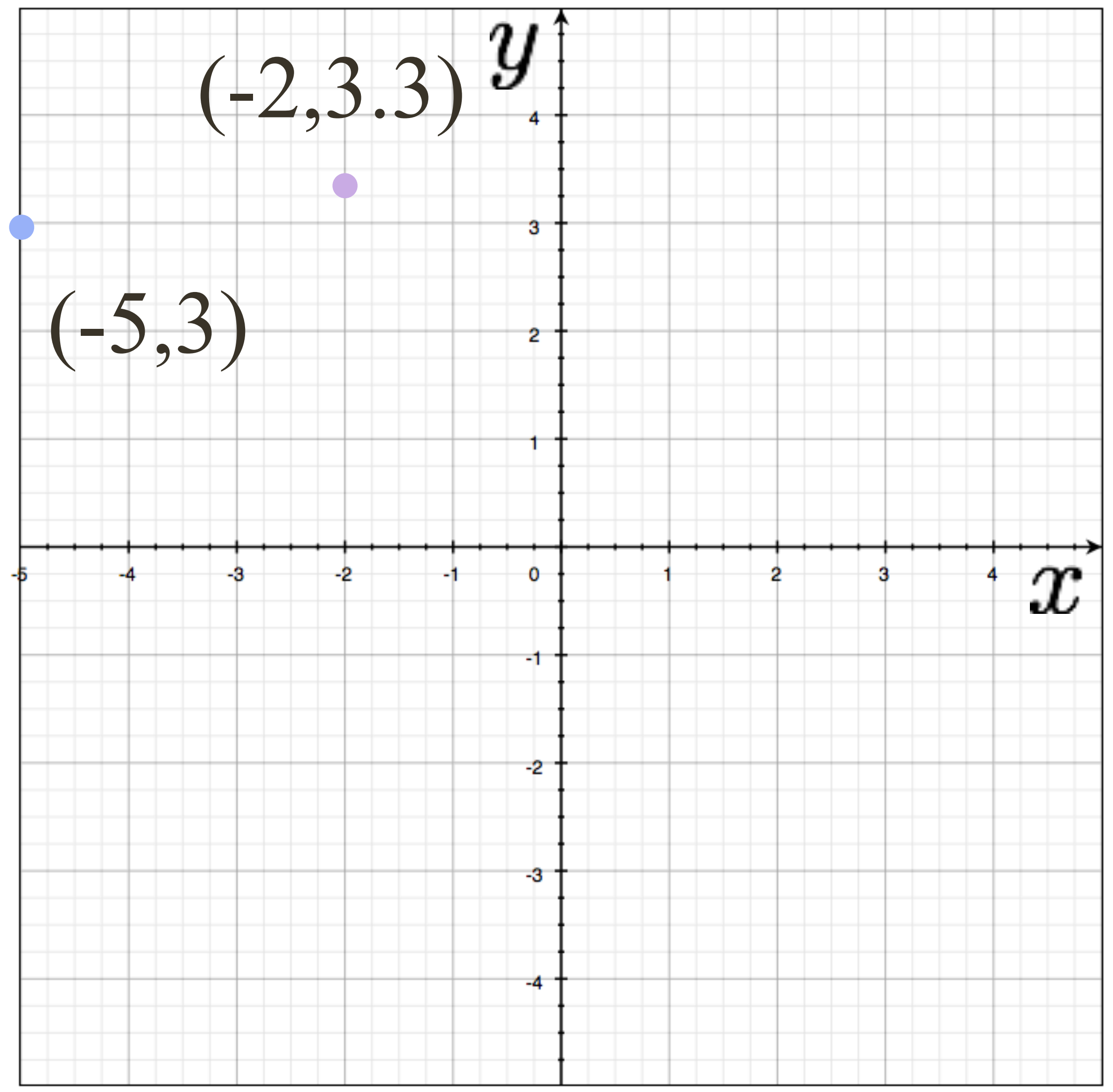
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

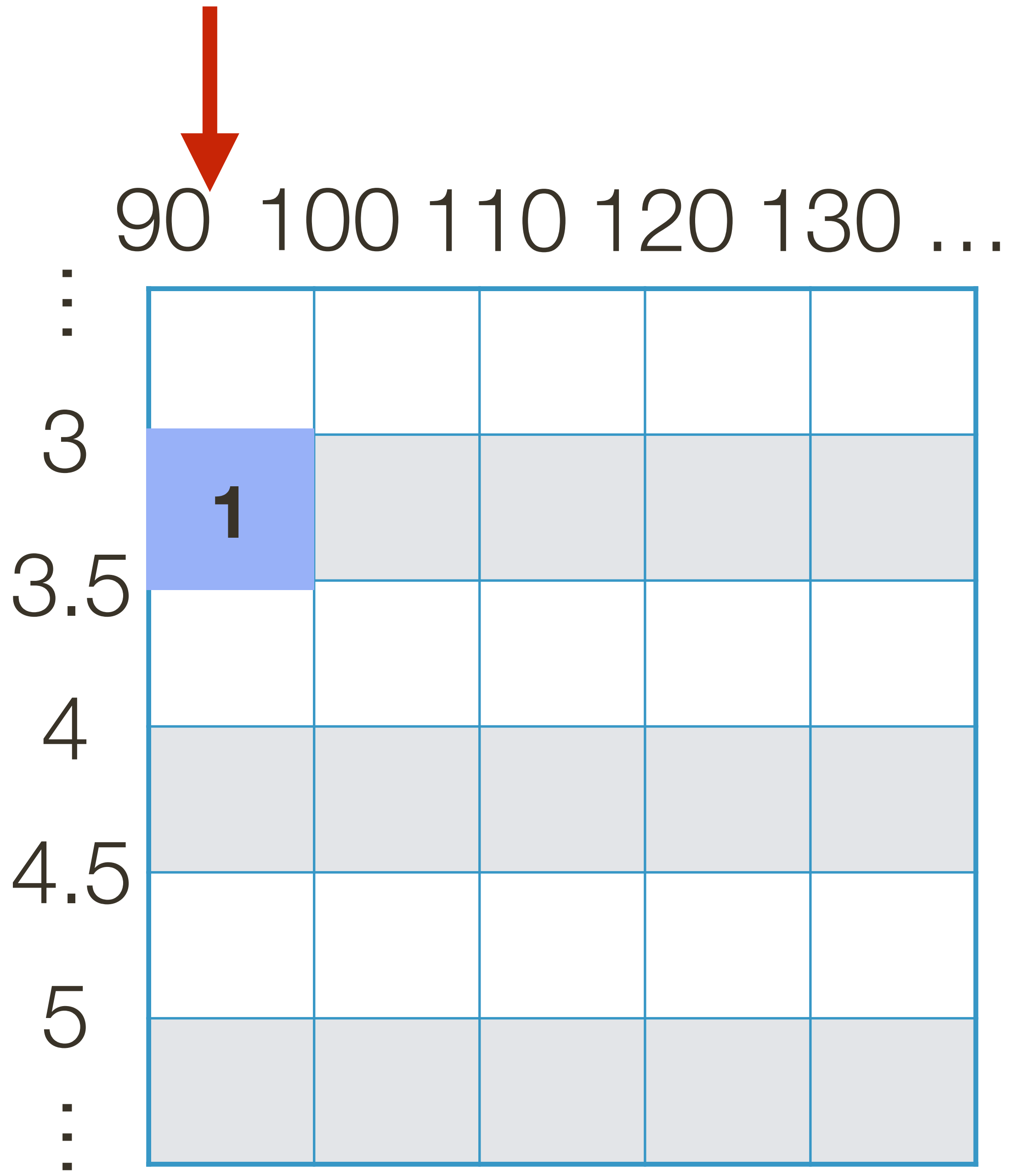
Example: Hough Transform for Lines





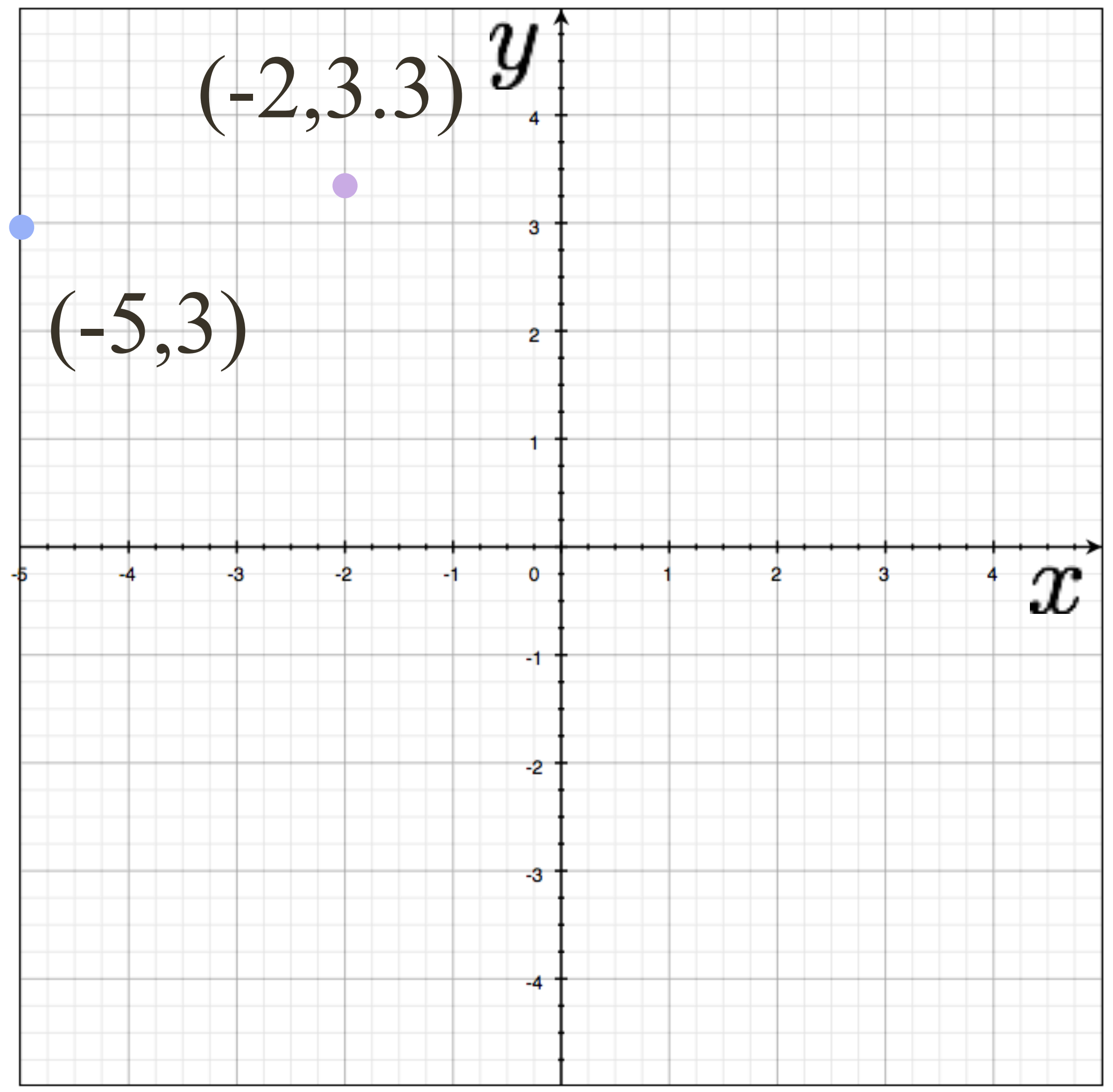
Example: Hough Transform for Lines

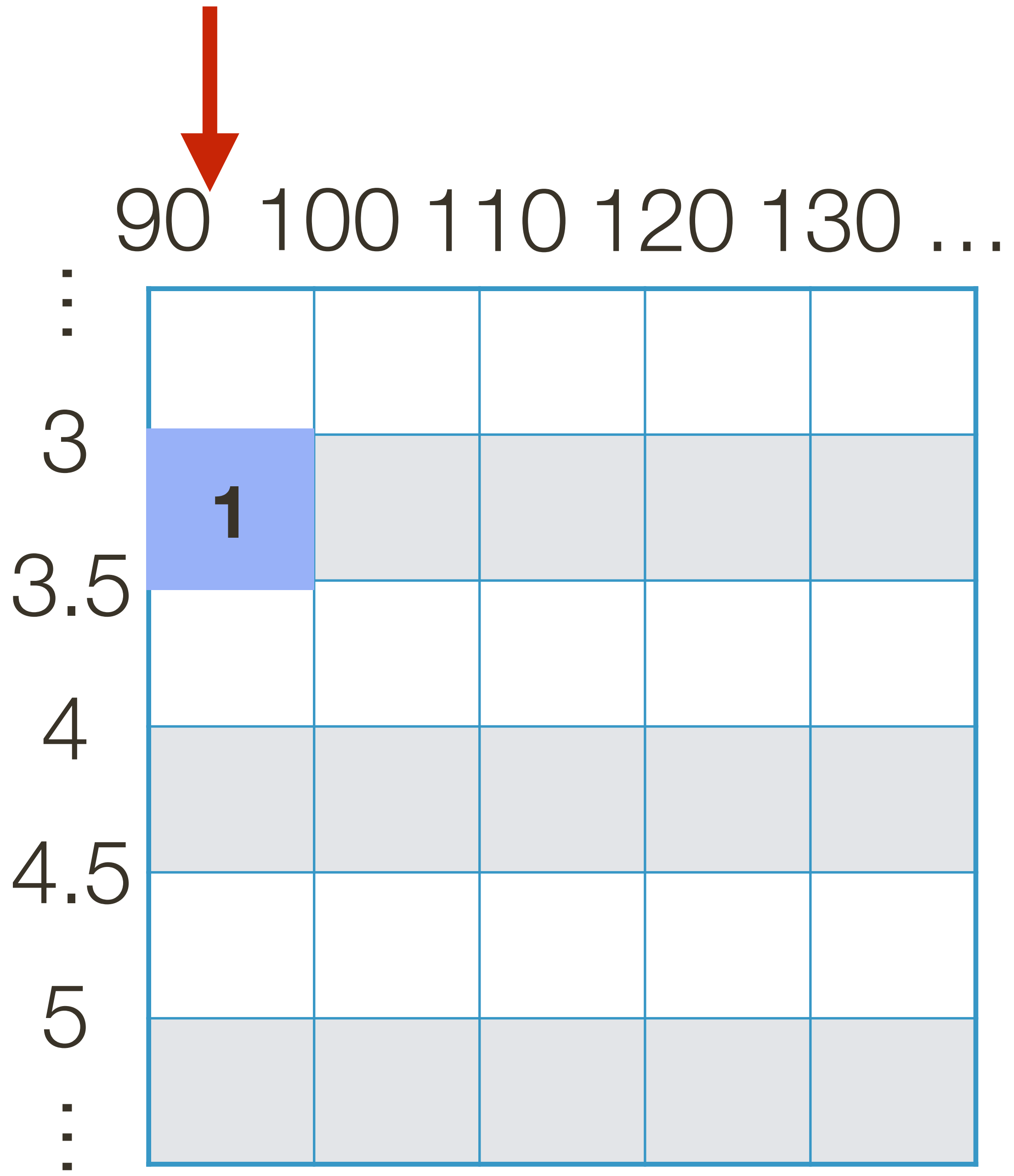




$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

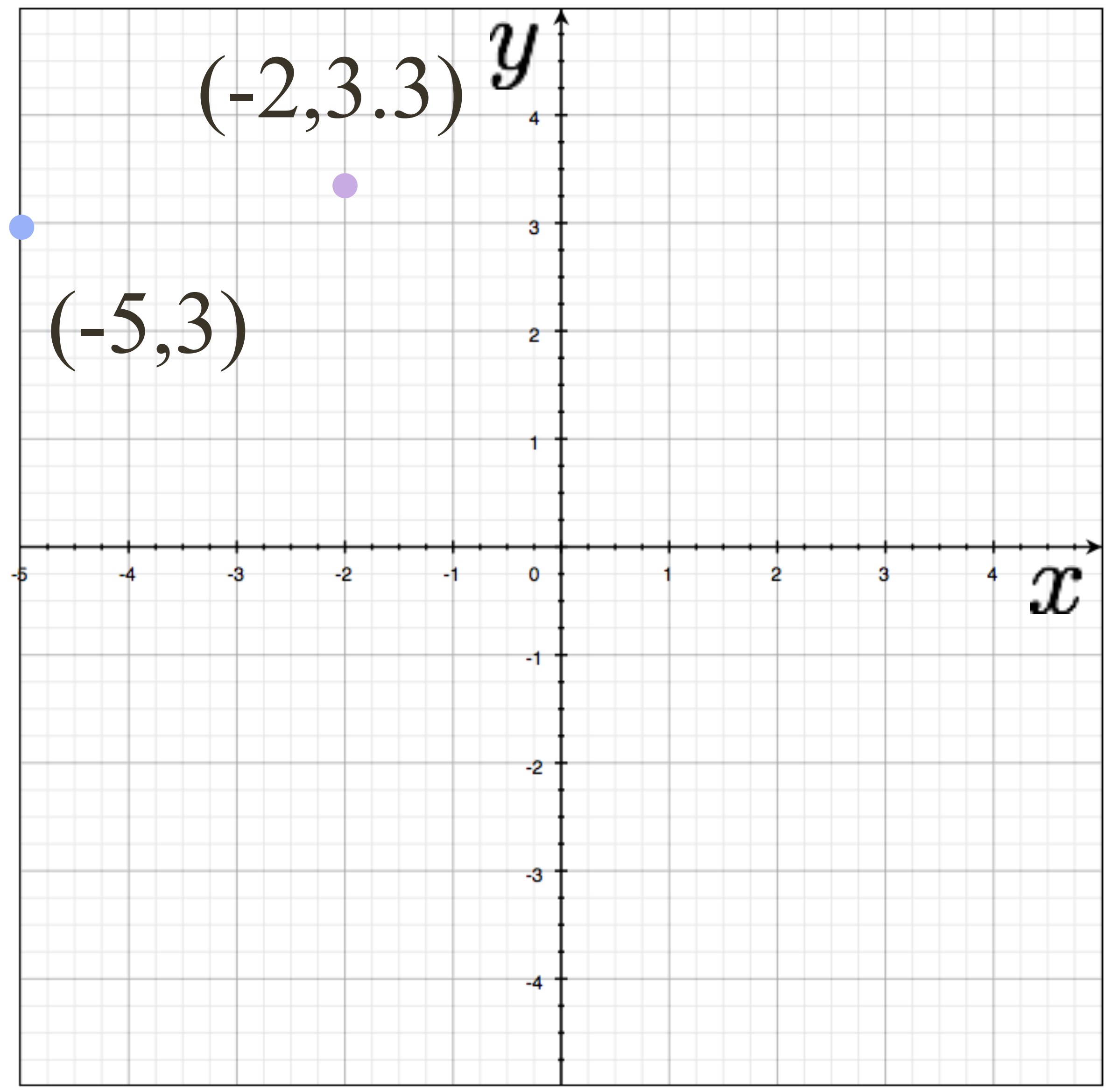
Example: Hough Transform for Lines





$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

Example: Hough Transform for Lines

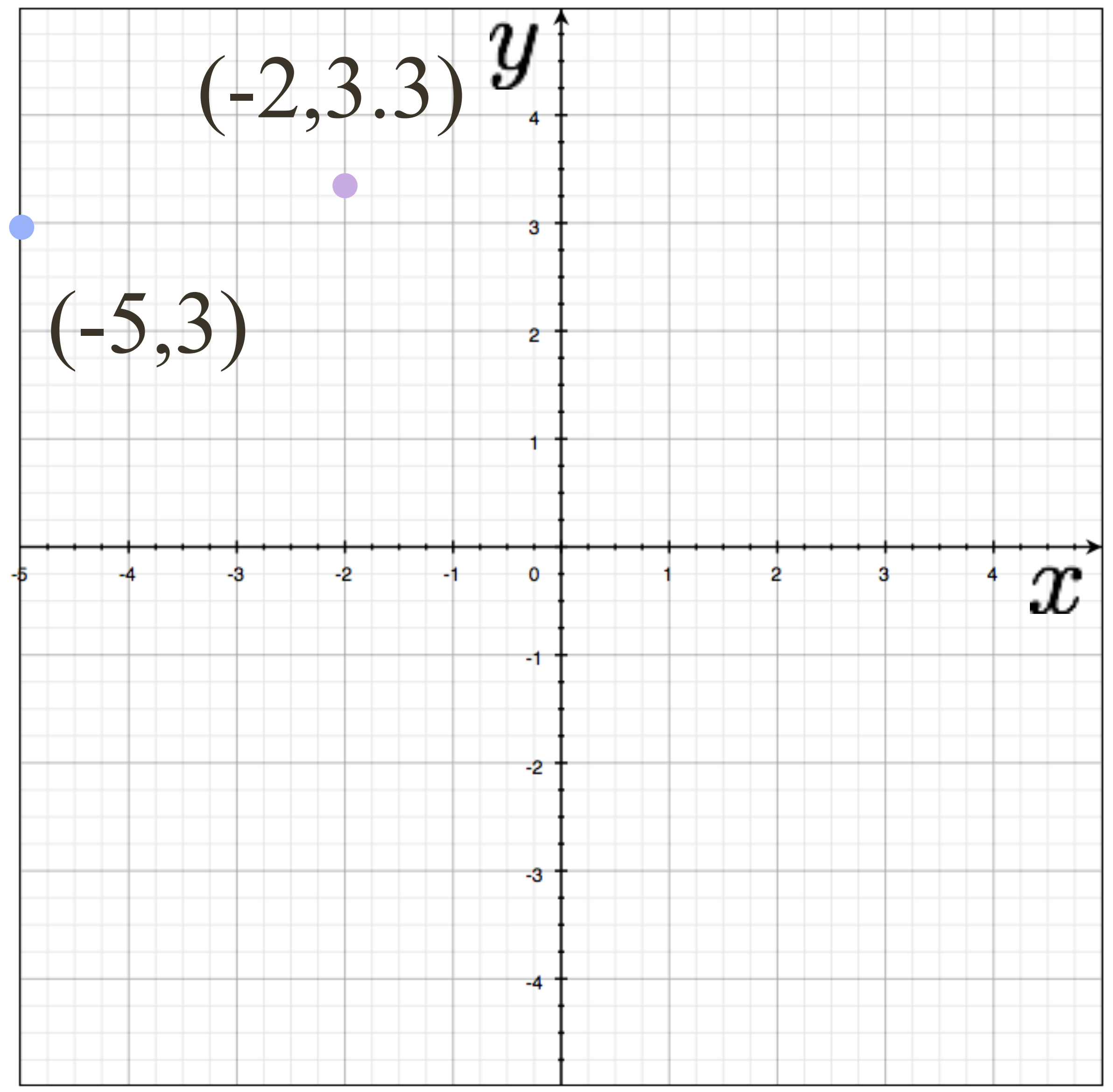


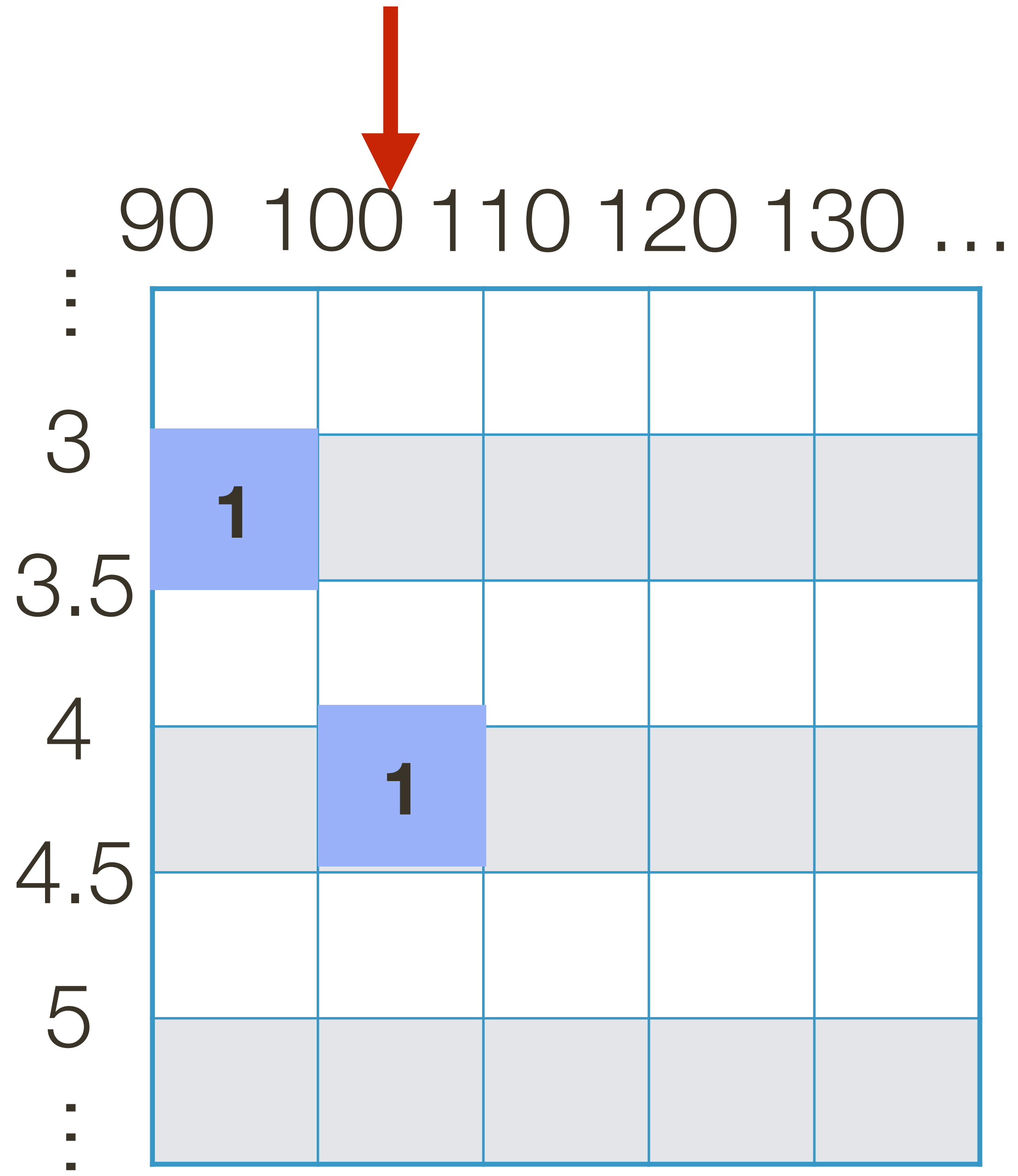
	90	100	110	120	130	…
⋮						
3	1					
3.5						
4						
4.5						
5						
⋮						

$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

Example: Hough Transform for Lines

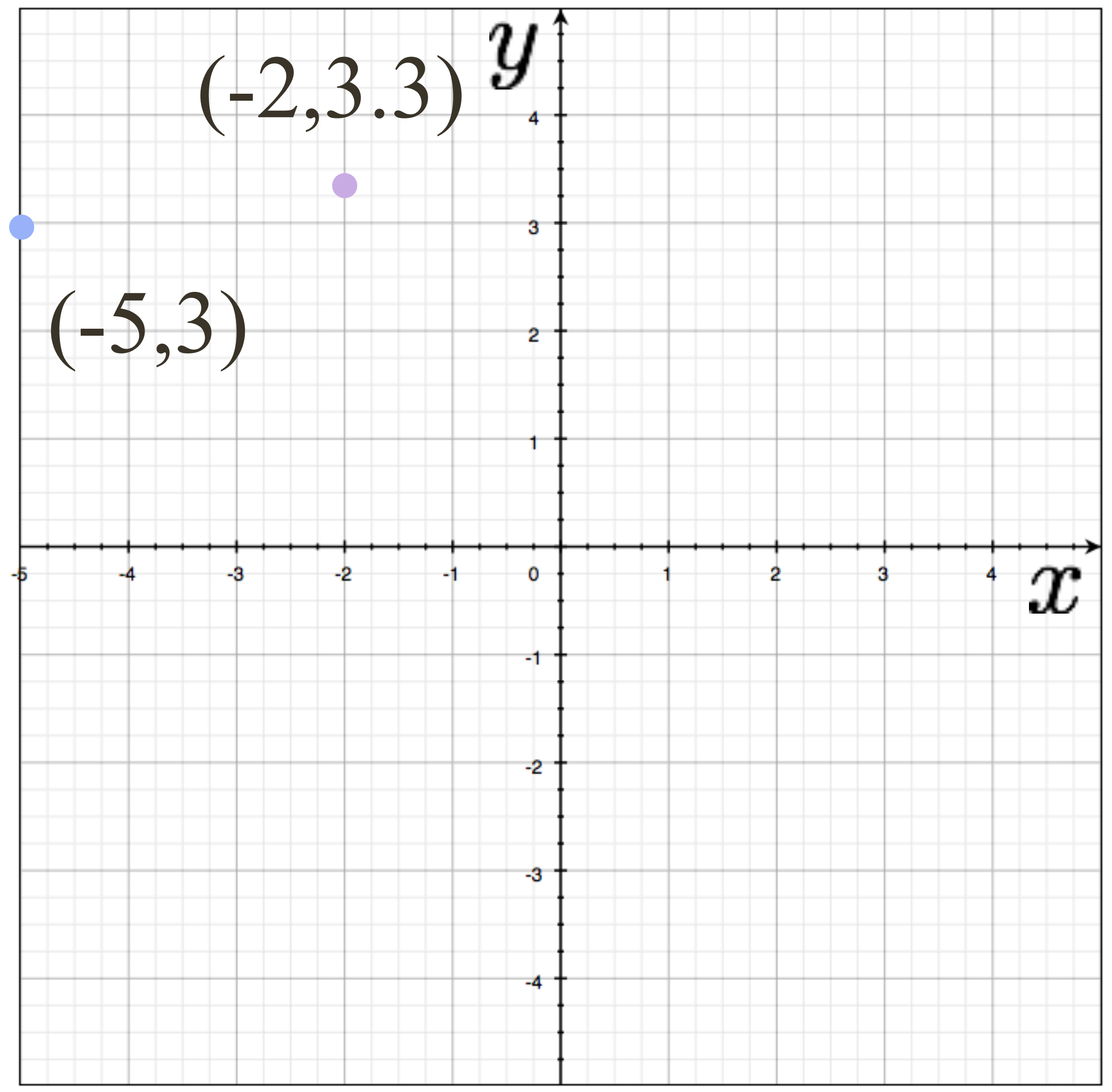


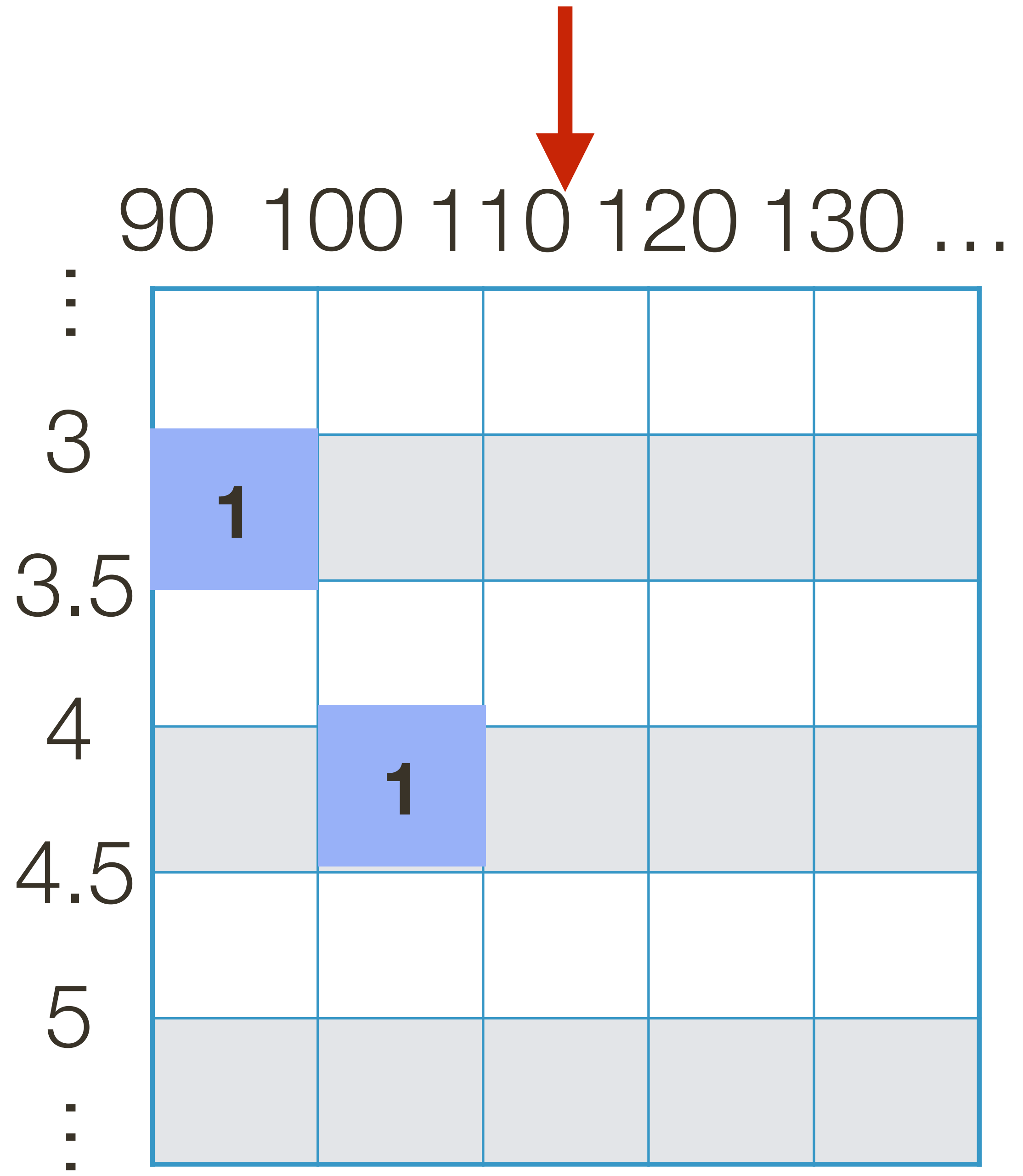


$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

Example: Hough Transform for Lines



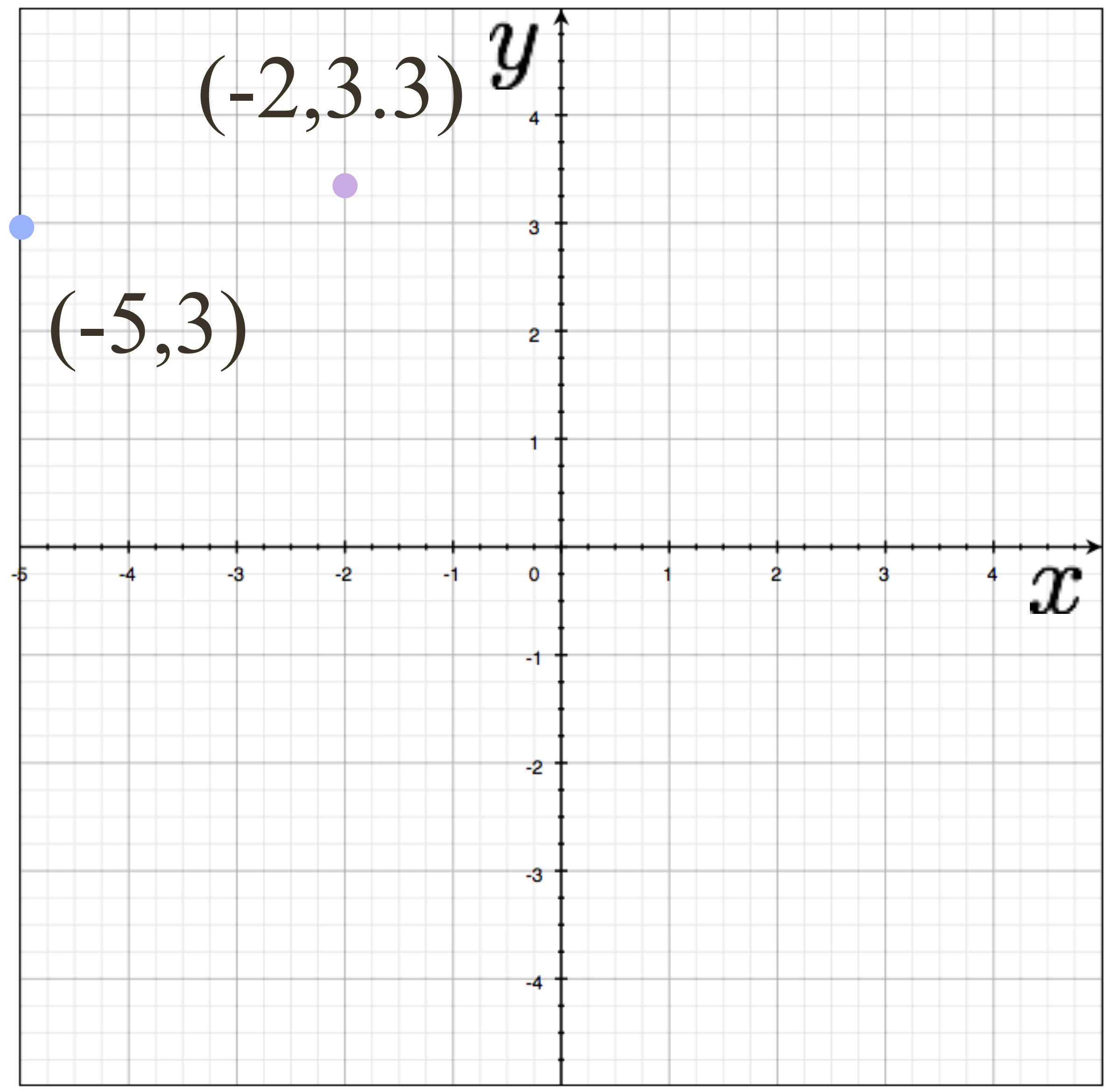


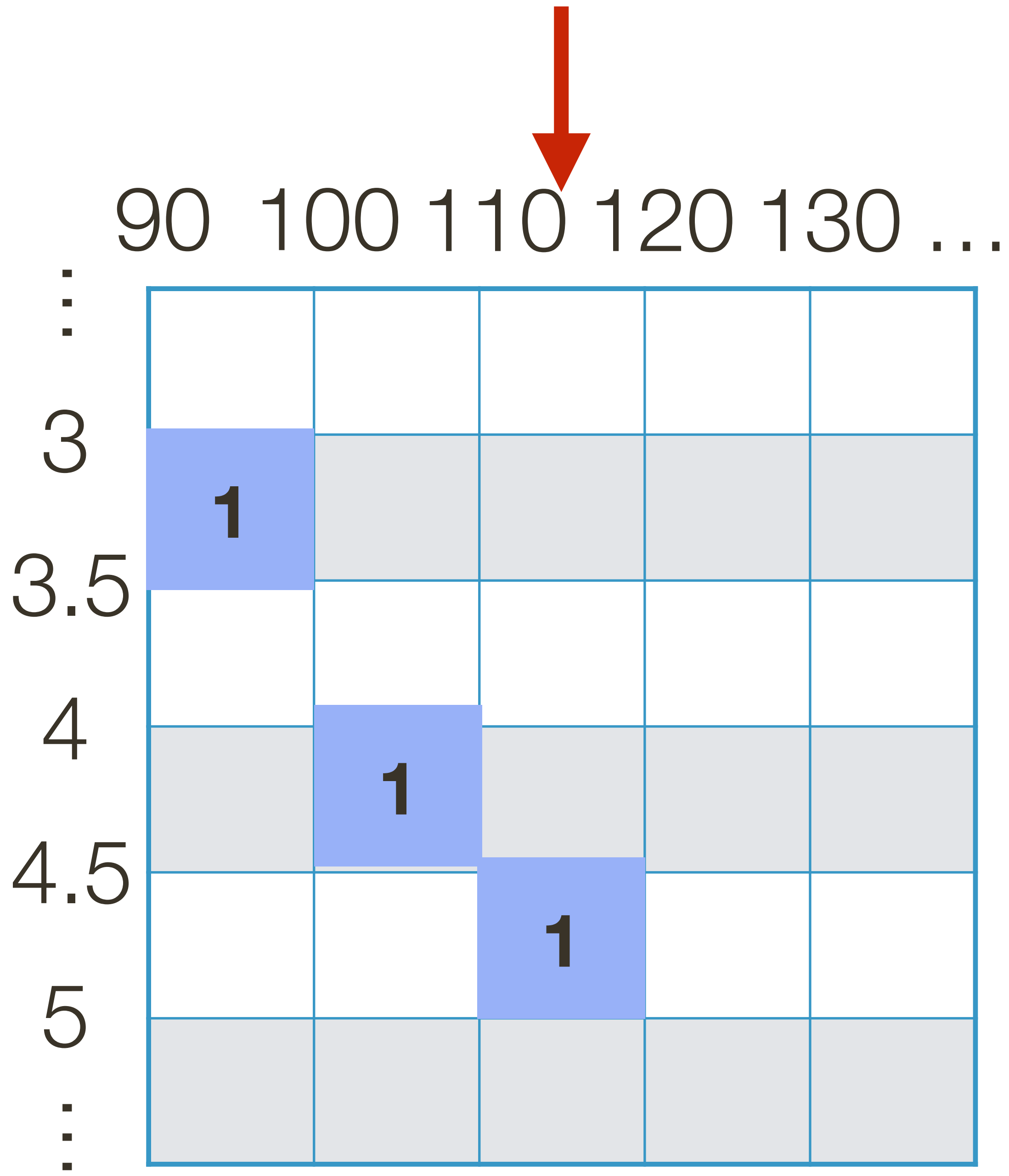
$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

$$-5\cos(115^\circ) + 3\sin(115^\circ) + r = 0 \rightarrow r \approx 4.83$$

Example: Hough Transform for Lines



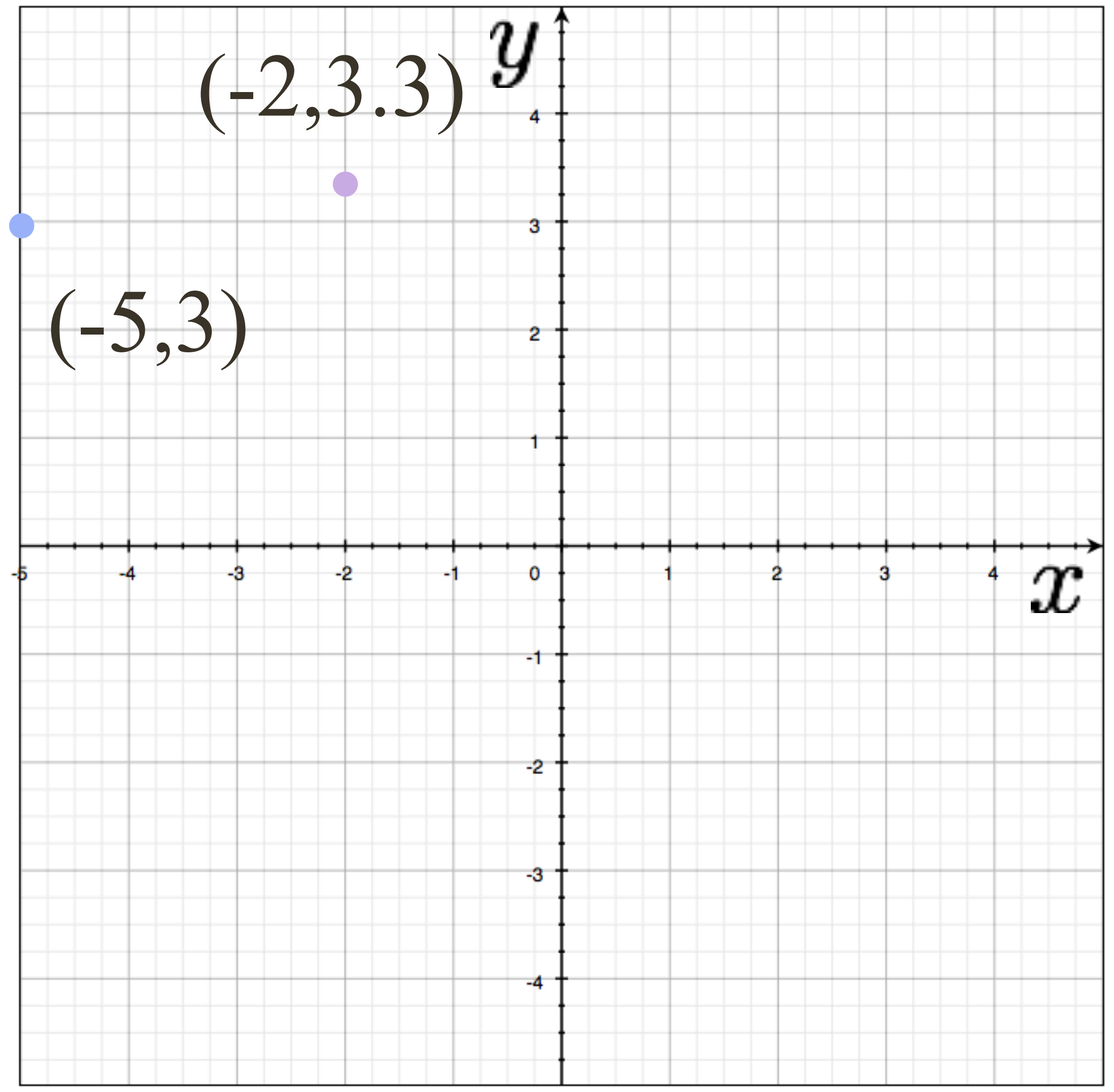


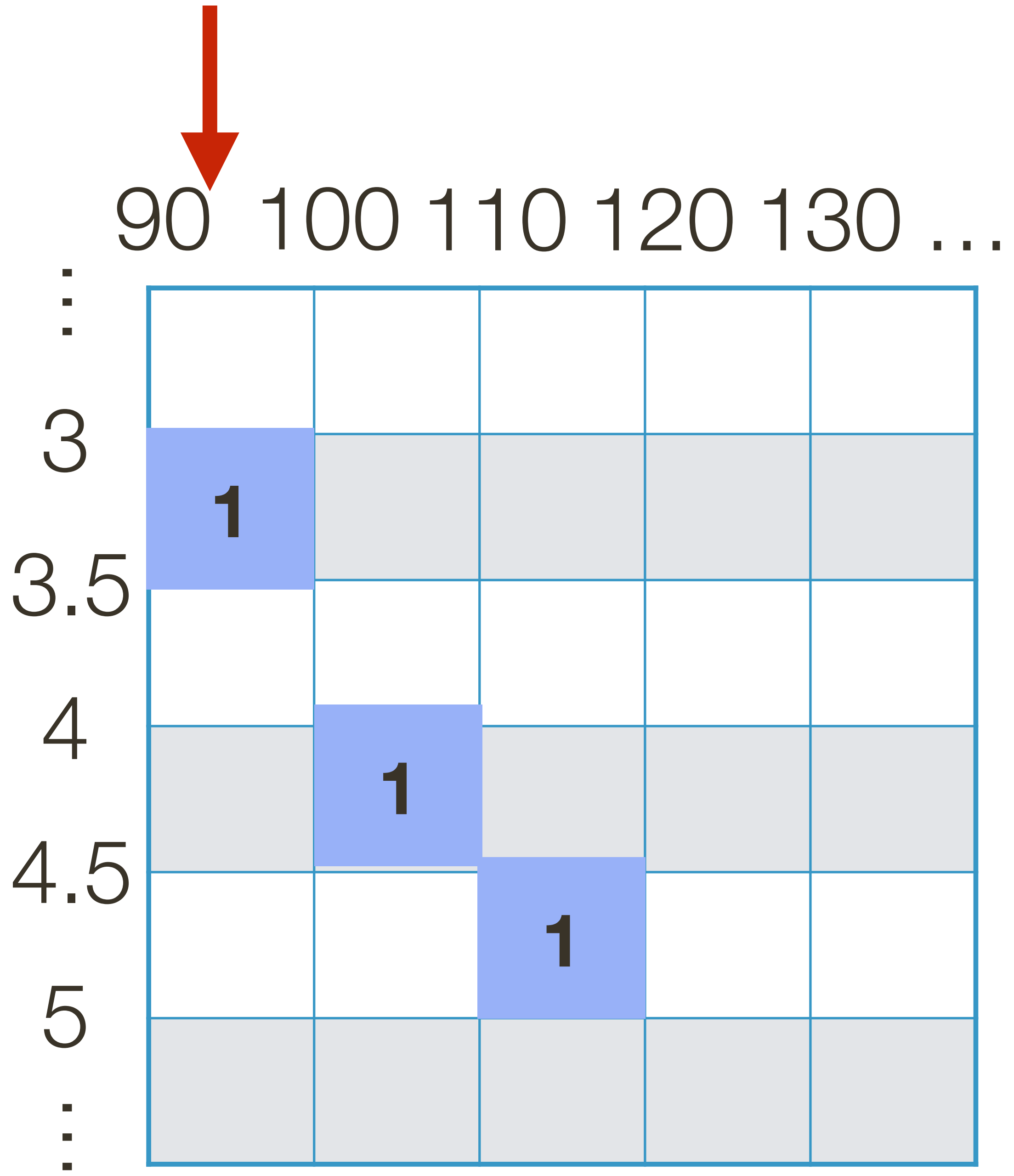
$$-5\cos(95°) + 3\sin(95°) + r = 0 \rightarrow r \approx 3.42$$

$$-5\cos(105°) + 3\sin(105°) + r = 0 \rightarrow r \approx 4.18$$

$$-5\cos(115°) + 3\sin(115°) + r = 0 \rightarrow r \approx 4.83$$

Example: Hough Transform for Lines





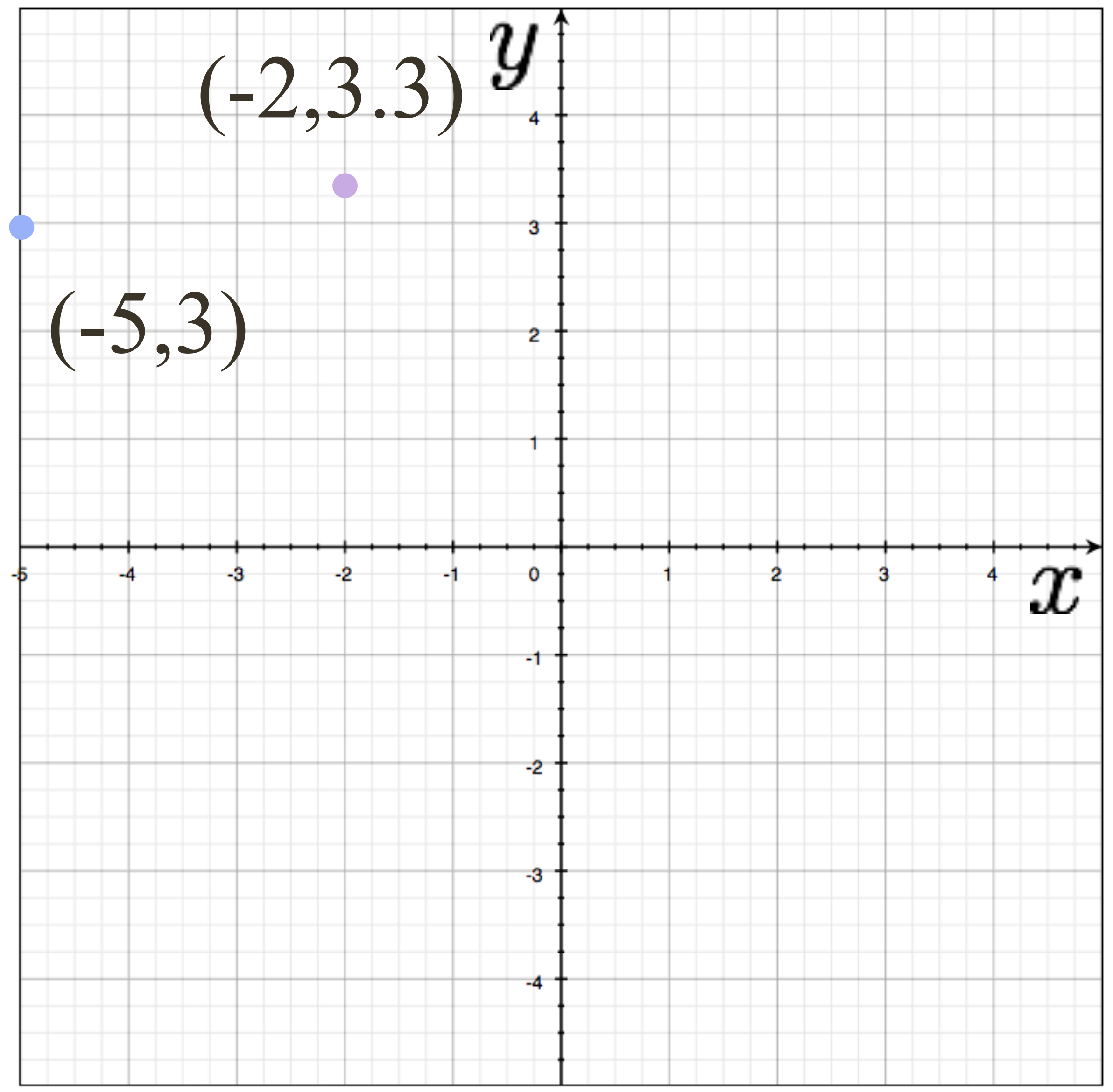
$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

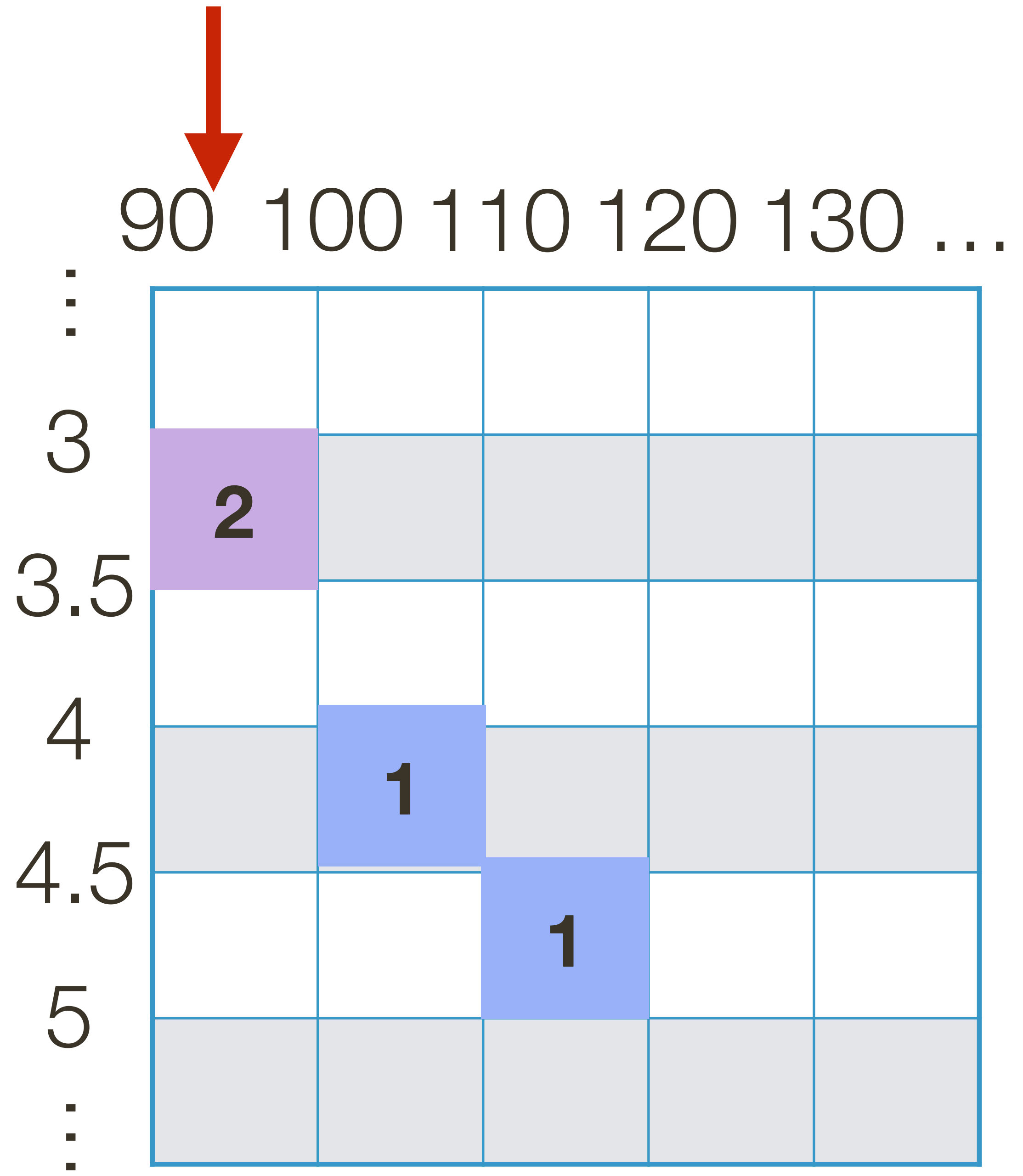
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

$$-5\cos(115^\circ) + 3\sin(115^\circ) + r = 0 \rightarrow r \approx 4.83$$

$$-2\cos(95^\circ) + 3.3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.46$$

Example: Hough Transform for Lines





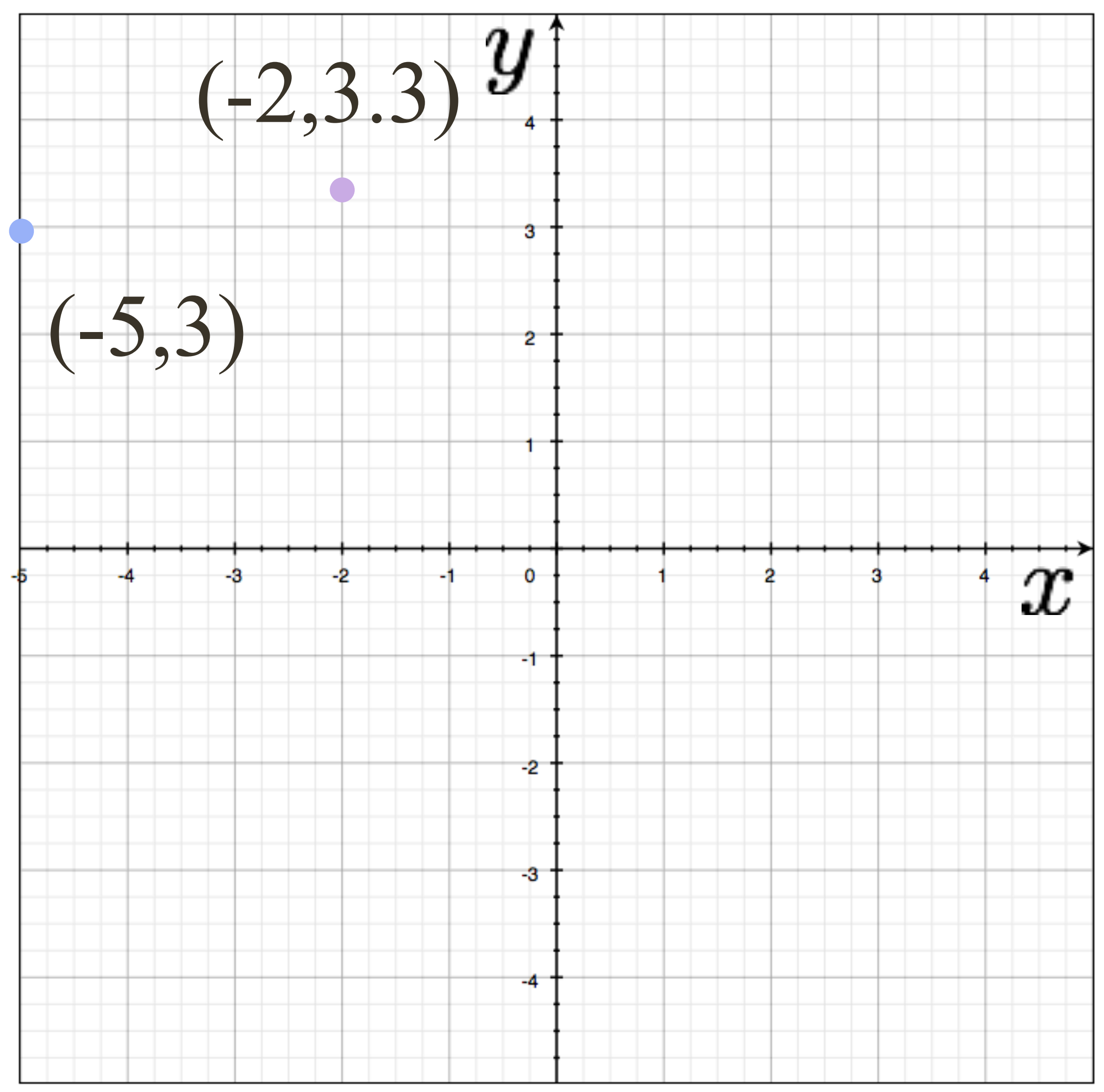
$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

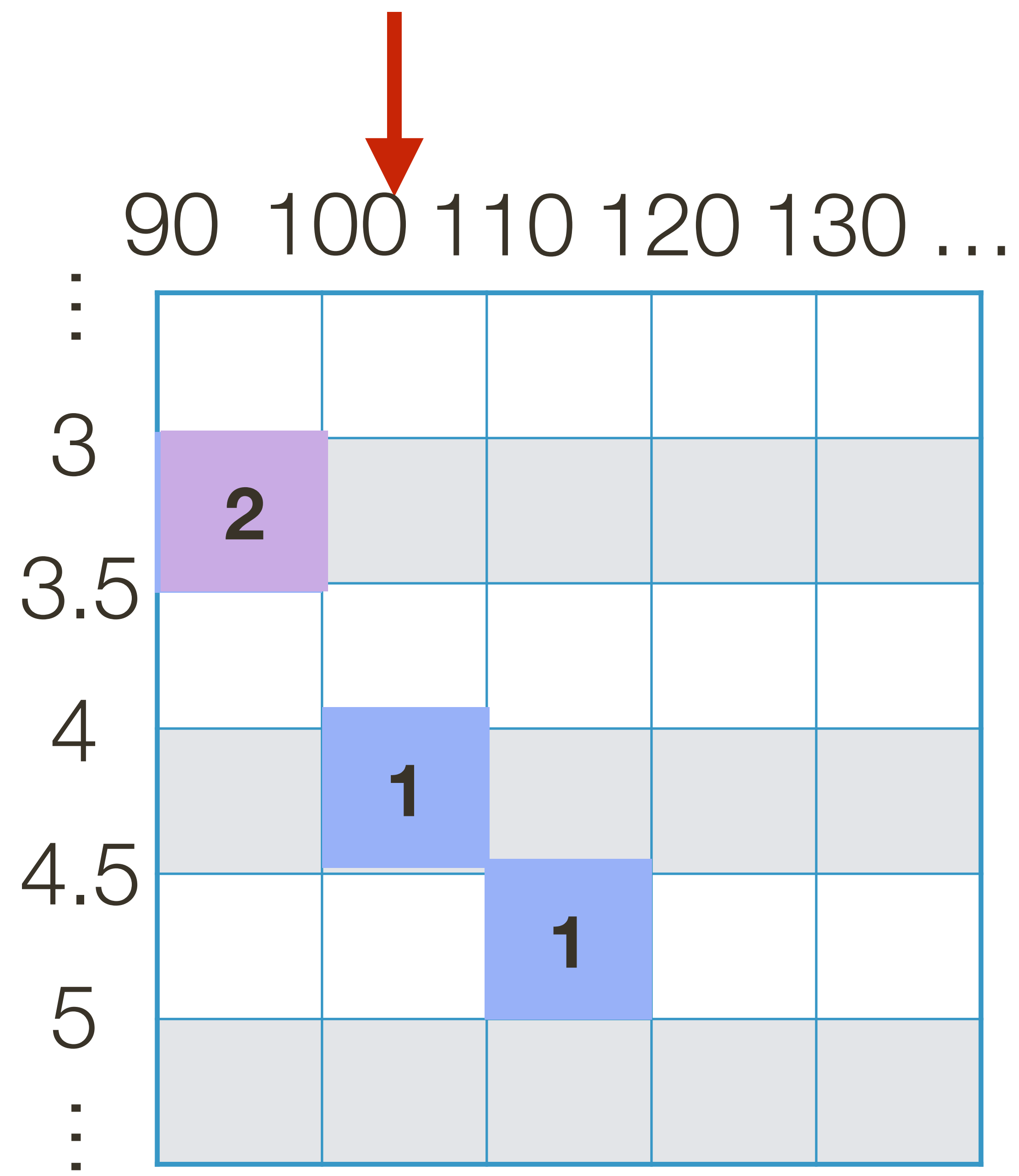
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

$$-5\cos(115^\circ) + 3\sin(115^\circ) + r = 0 \rightarrow r \approx 4.83$$

$$-2\cos(95^\circ) + 3.3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.46$$

Example: Hough Transform for Lines





$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

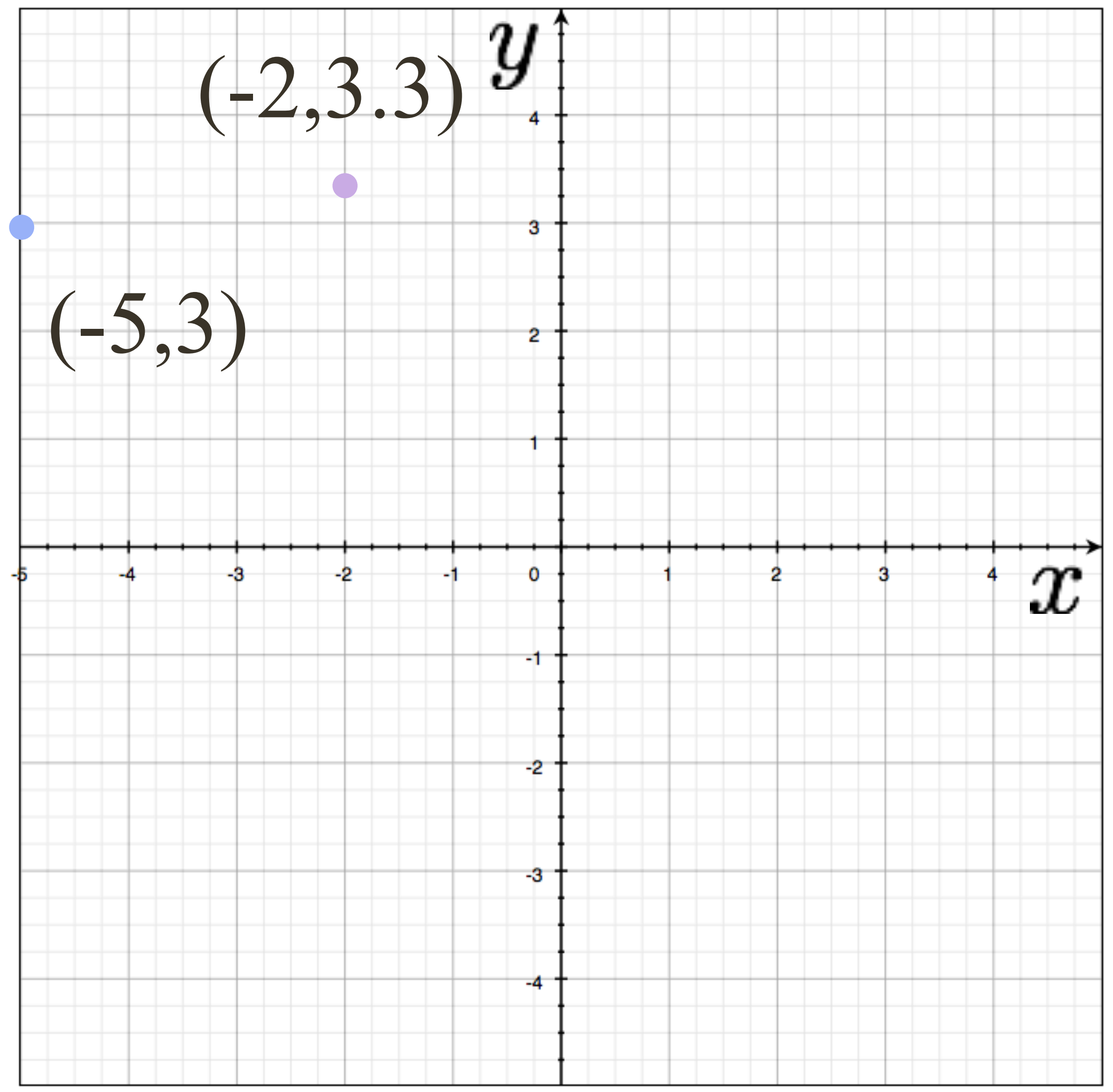
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

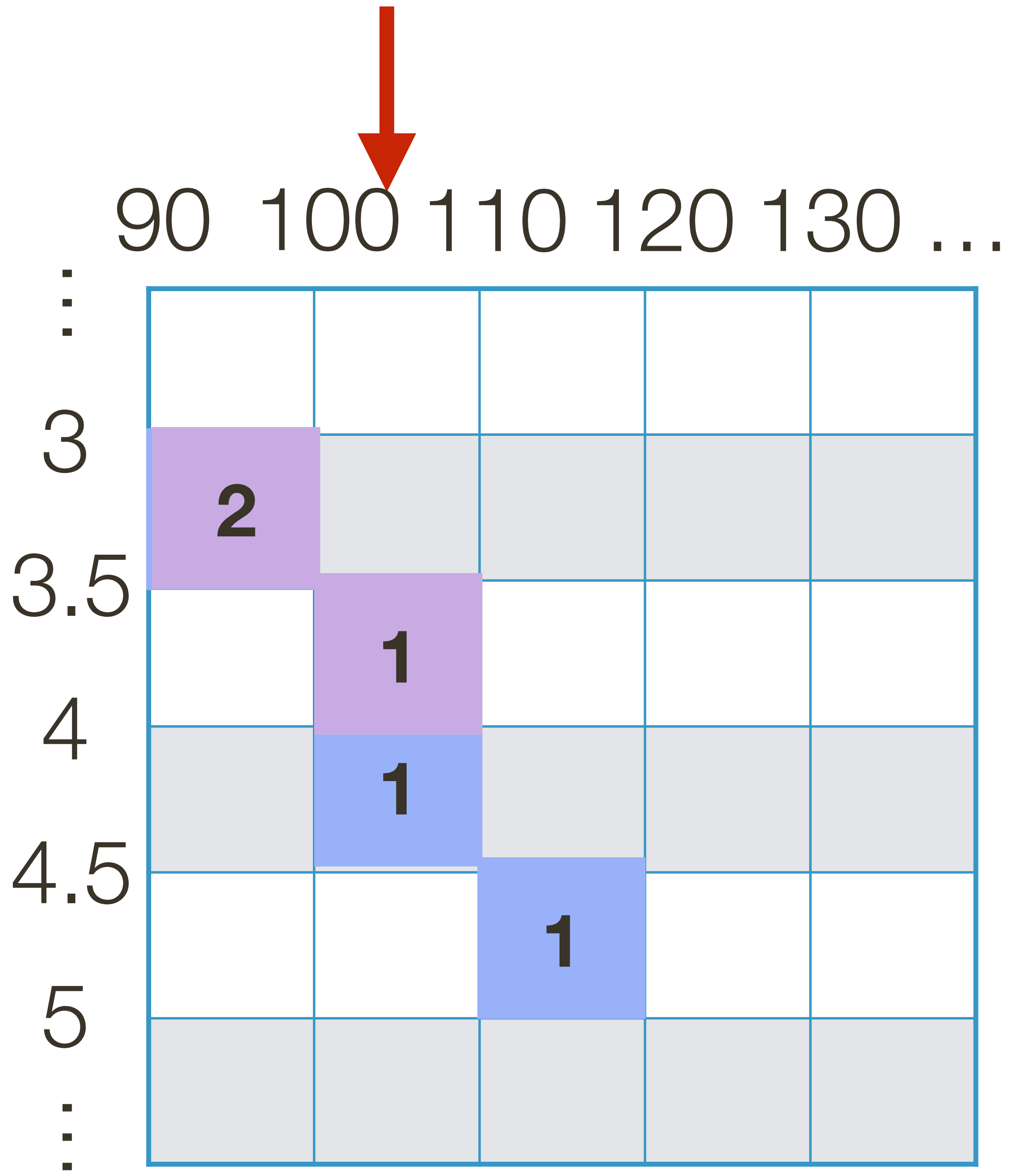
$$-5\cos(115^\circ) + 3\sin(115^\circ) + r = 0 \rightarrow r \approx 4.83$$

$$-2\cos(95^\circ) + 3.3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.46$$

$$-2\cos(105^\circ) + 3.3\sin(105^\circ) + r = 0 \rightarrow r \approx 3.71$$

Example: Hough Transform for Lines





$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

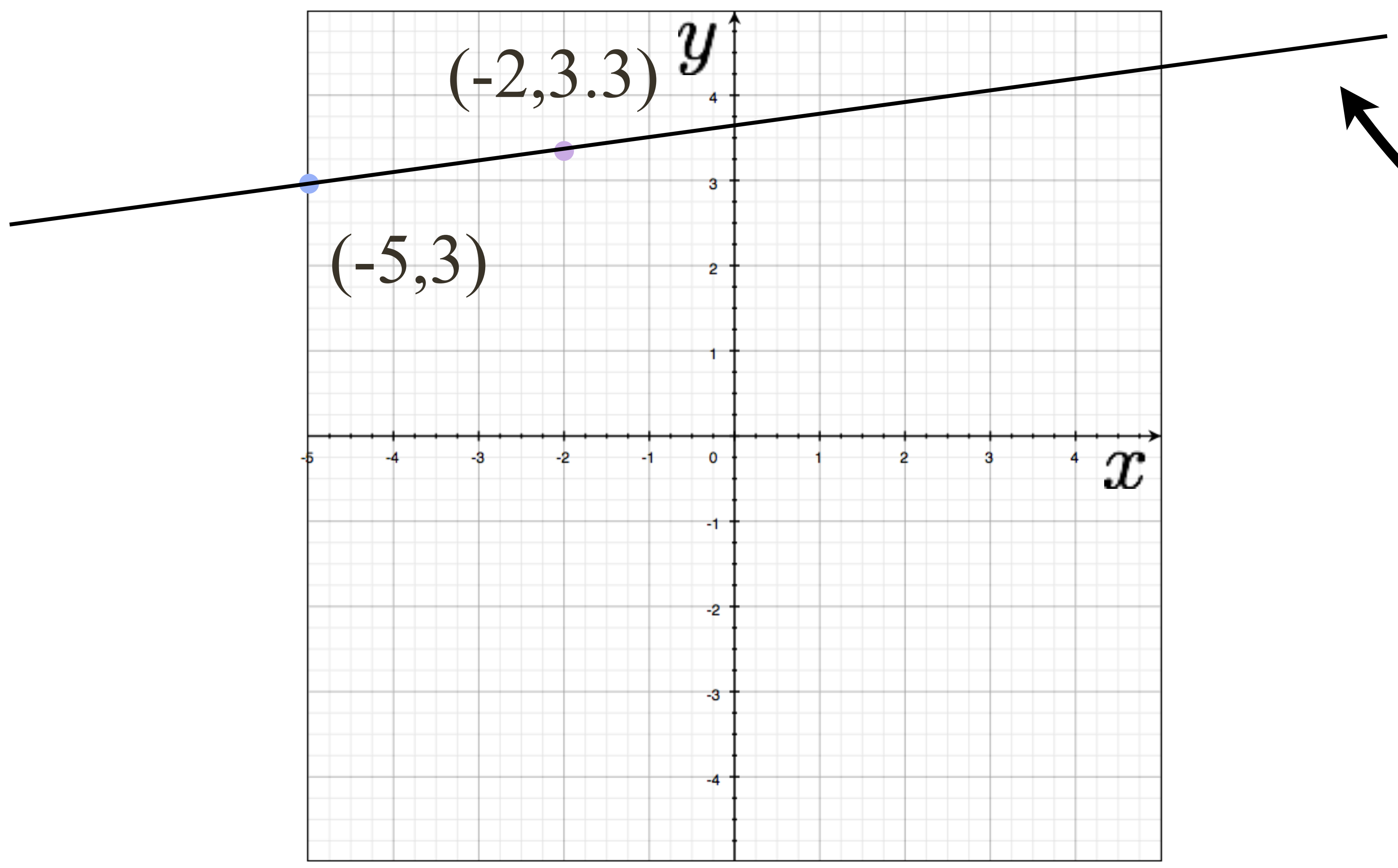
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

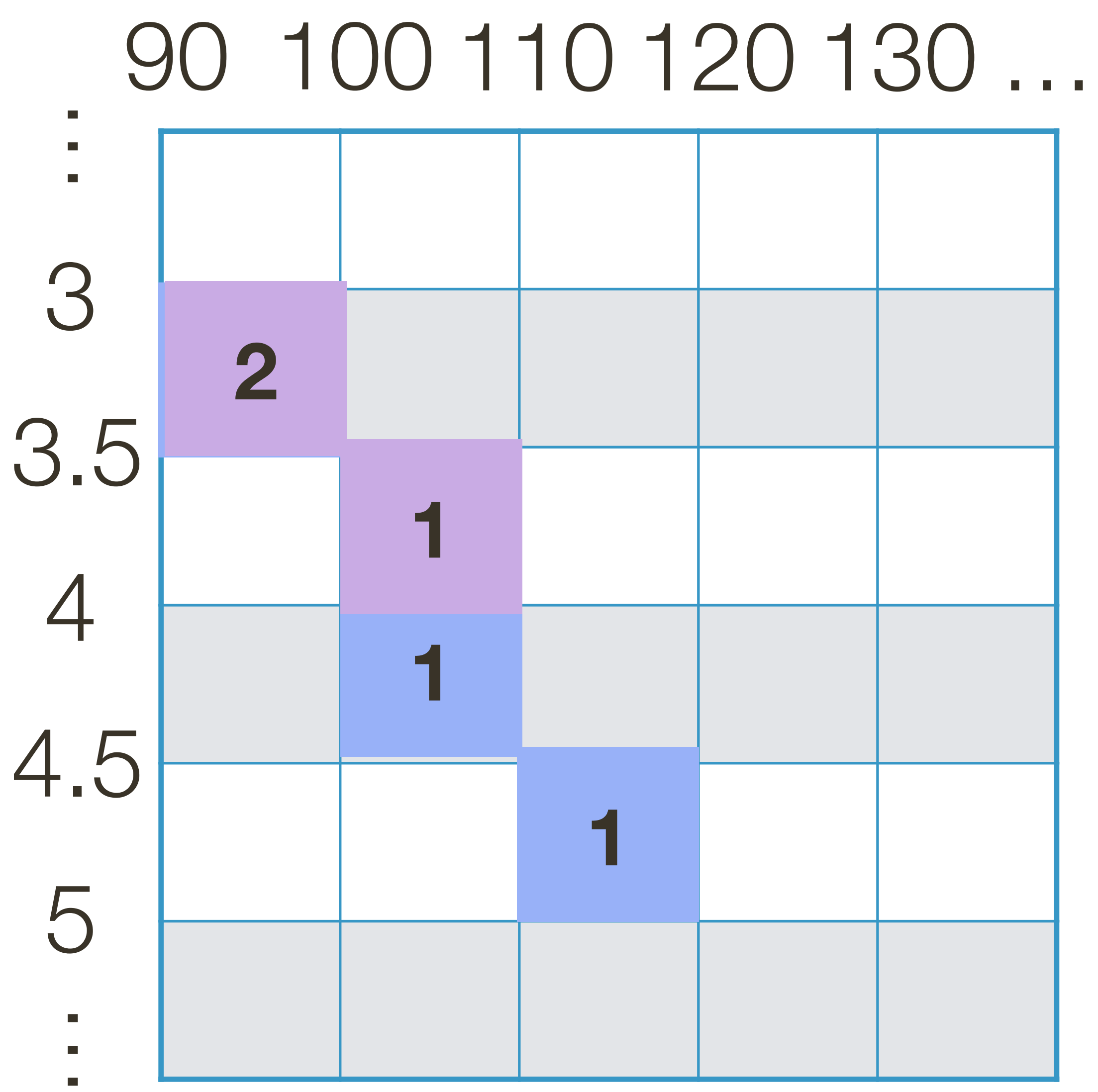
$$-5\cos(115^\circ) + 3\sin(115^\circ) + r = 0 \rightarrow r \approx 4.83$$

$$-2\cos(95^\circ) + 3.3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.46$$

$$-2\cos(105^\circ) + 3.3\sin(105^\circ) + r = 0 \rightarrow r \approx 3.71$$

Example: Hough Transform for Lines





$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

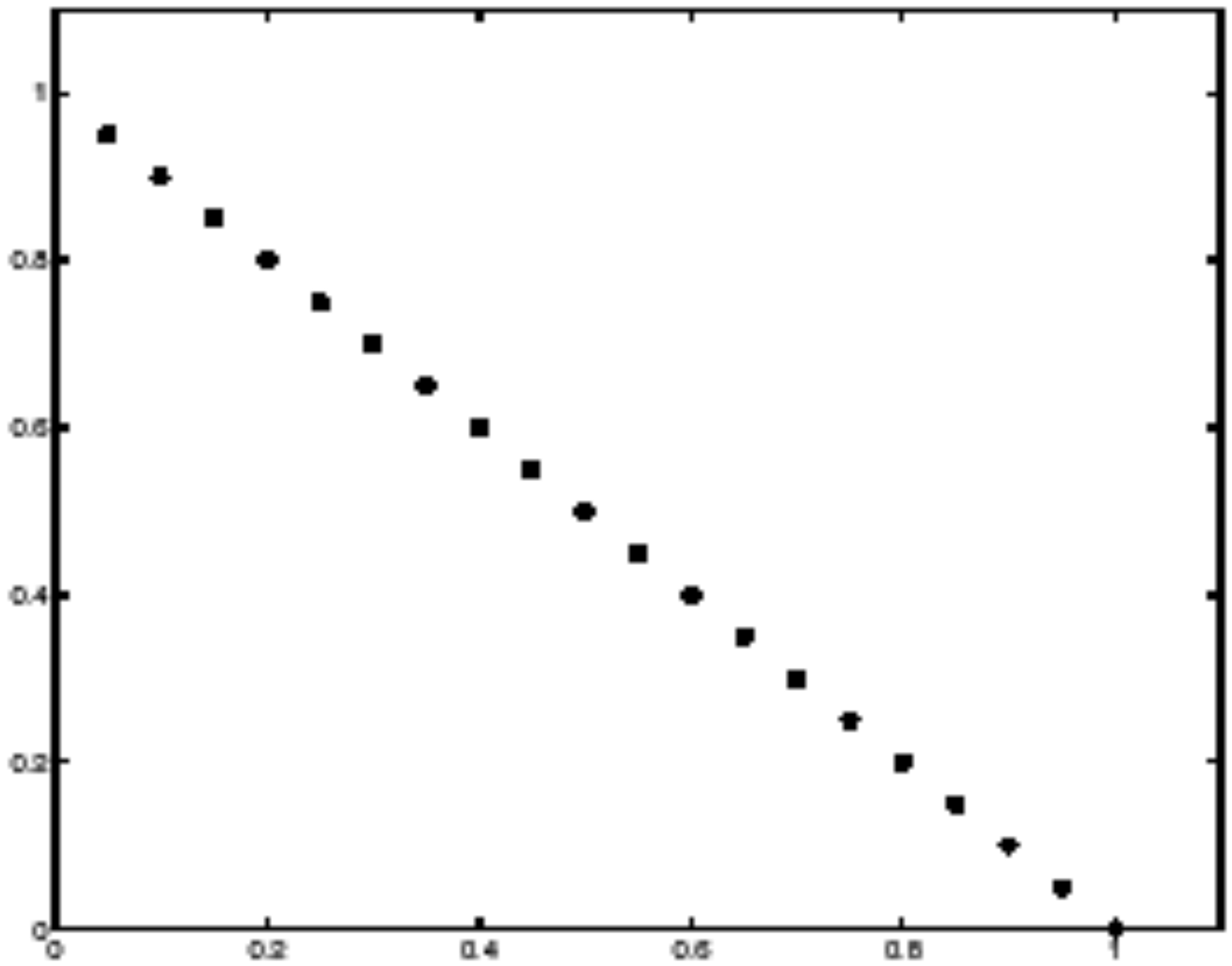
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

$$-5\cos(115^\circ) + 3\sin(115^\circ) + r = 0 \rightarrow r \approx 4.83$$

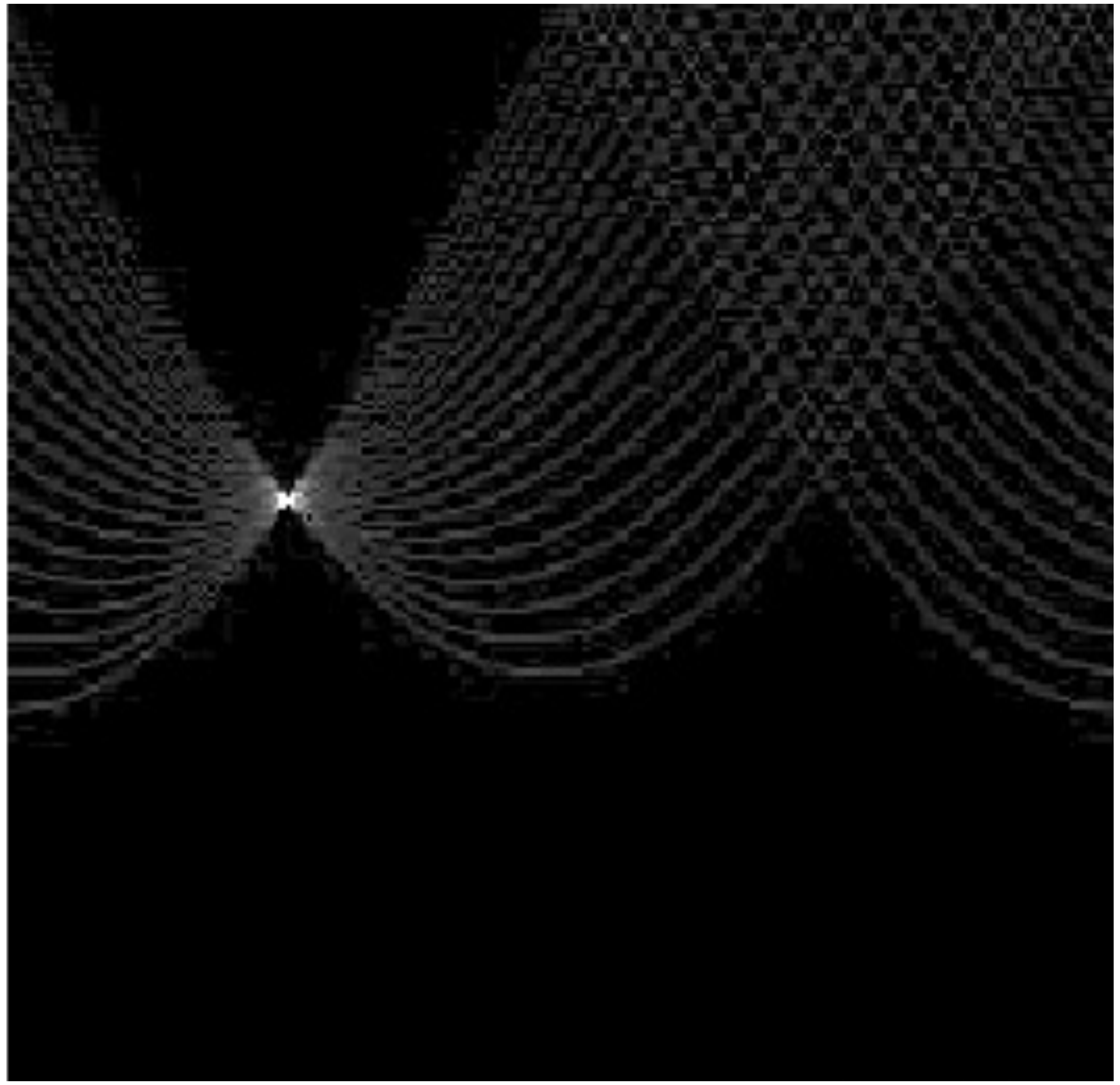
$$-2\cos(95^\circ) + 3.3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.46$$

$$-2\cos(105^\circ) + 3.3\sin(105^\circ) + r = 0 \rightarrow r \approx 3.71$$

Example: Clean Data



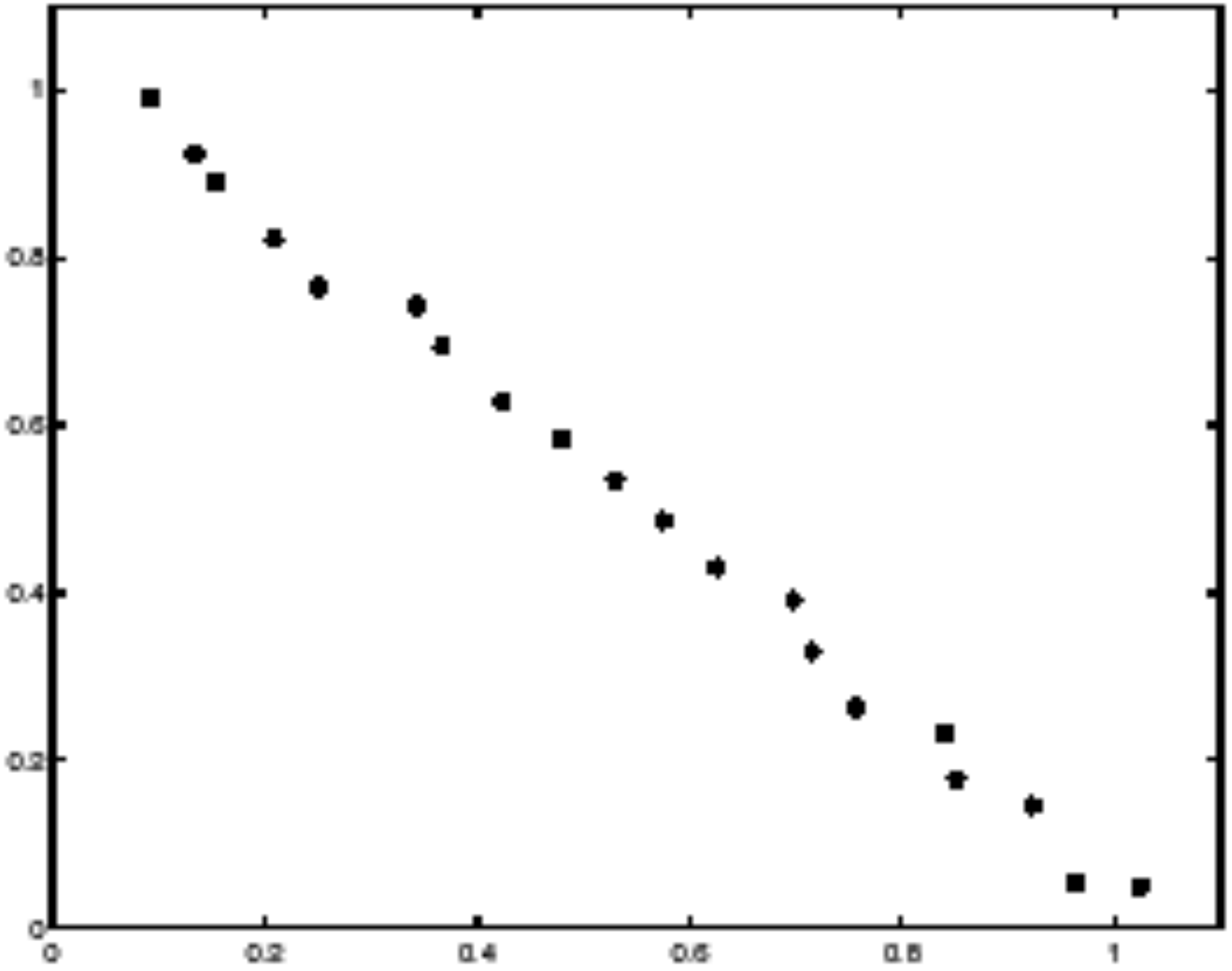
Tokens



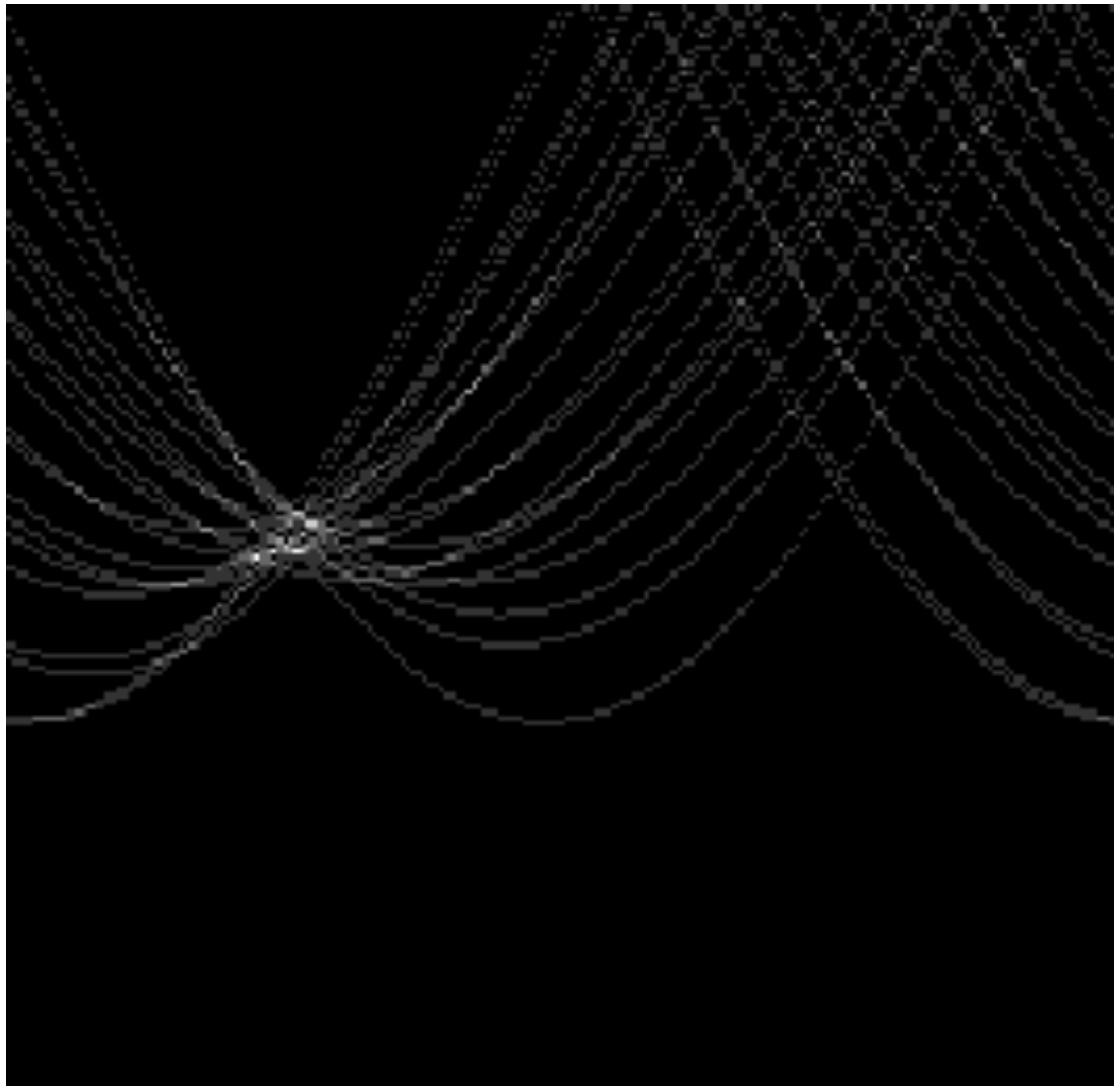
Votes
Horizontal axis is θ
Vertical Axis is r

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

Example: Some Noise



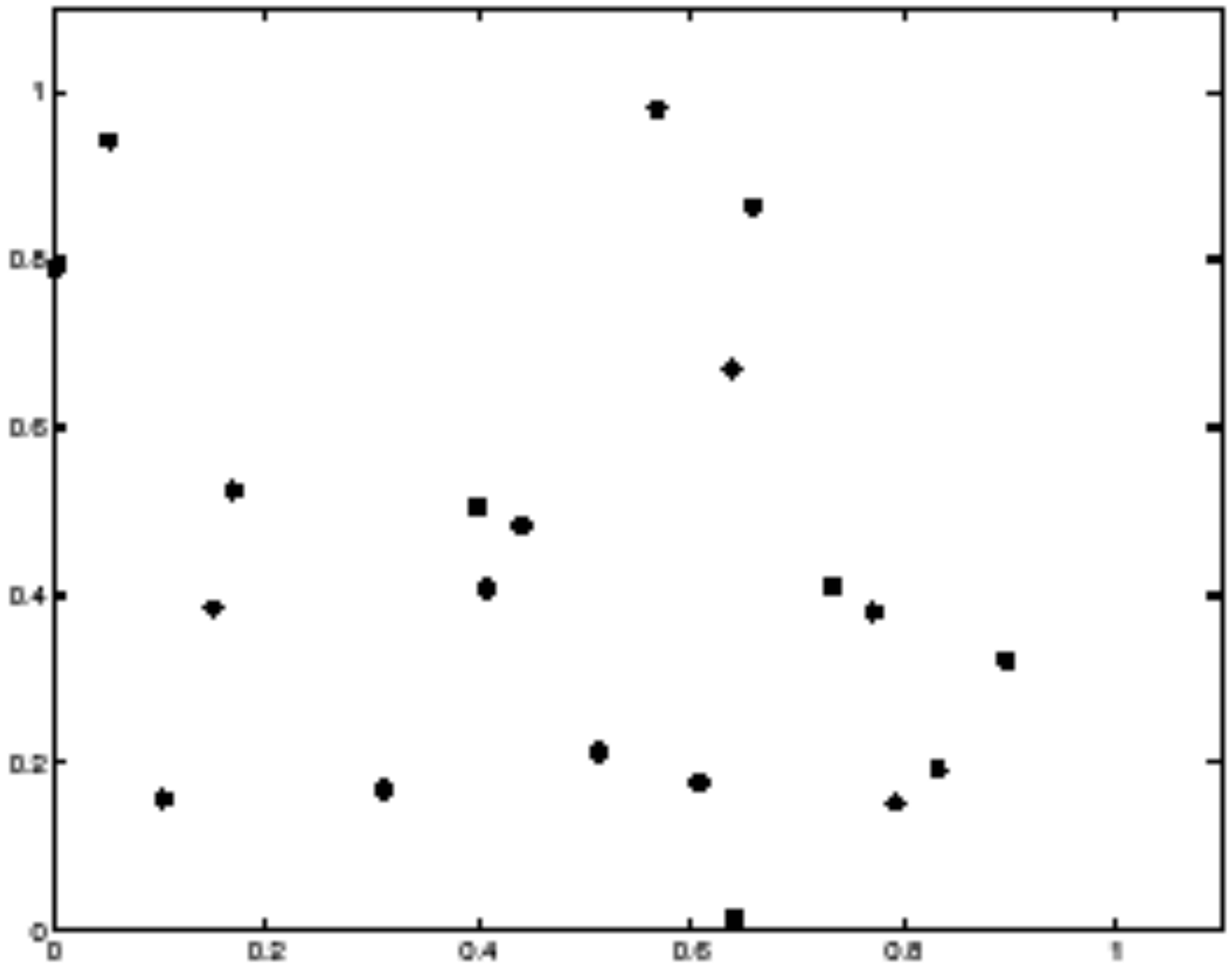
Tokens



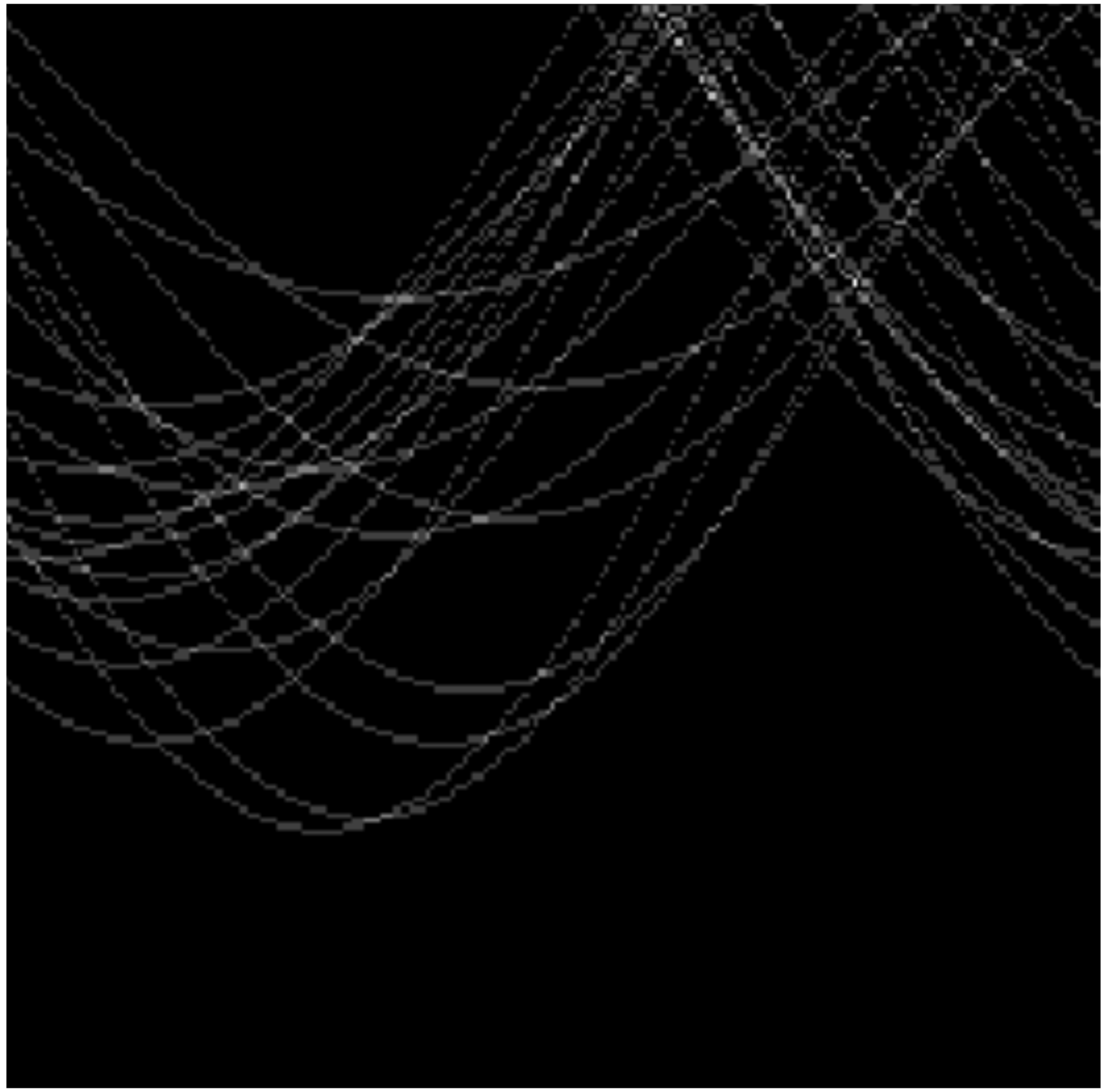
Votes
Horizontal axis is θ
Vertical Axis is r

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

Example: Too Much Noise



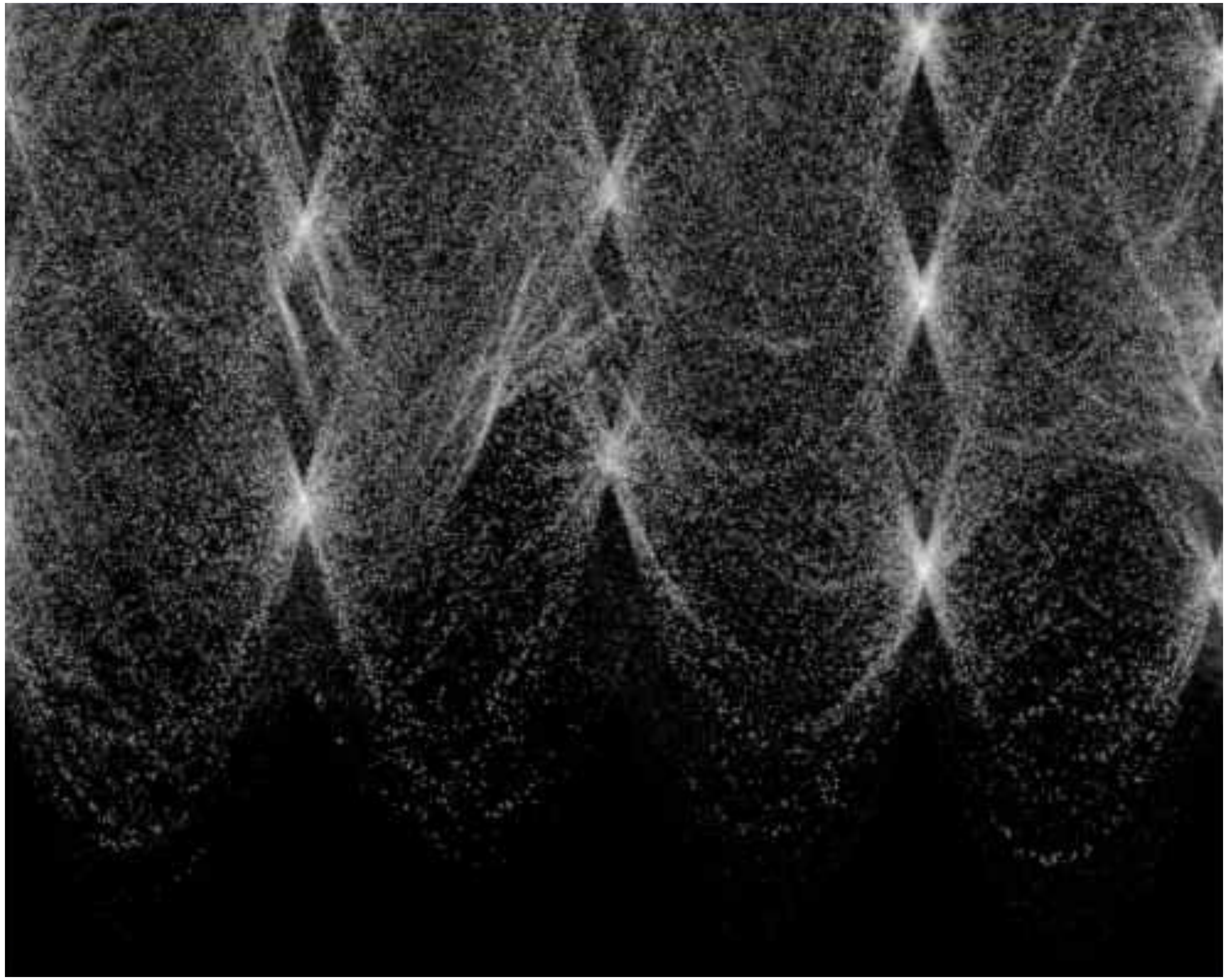
Tokens



Votes
Horizontal axis is θ
Vertical Axis is r

Forsyth & Ponce (2nd ed.) Figure 10.2

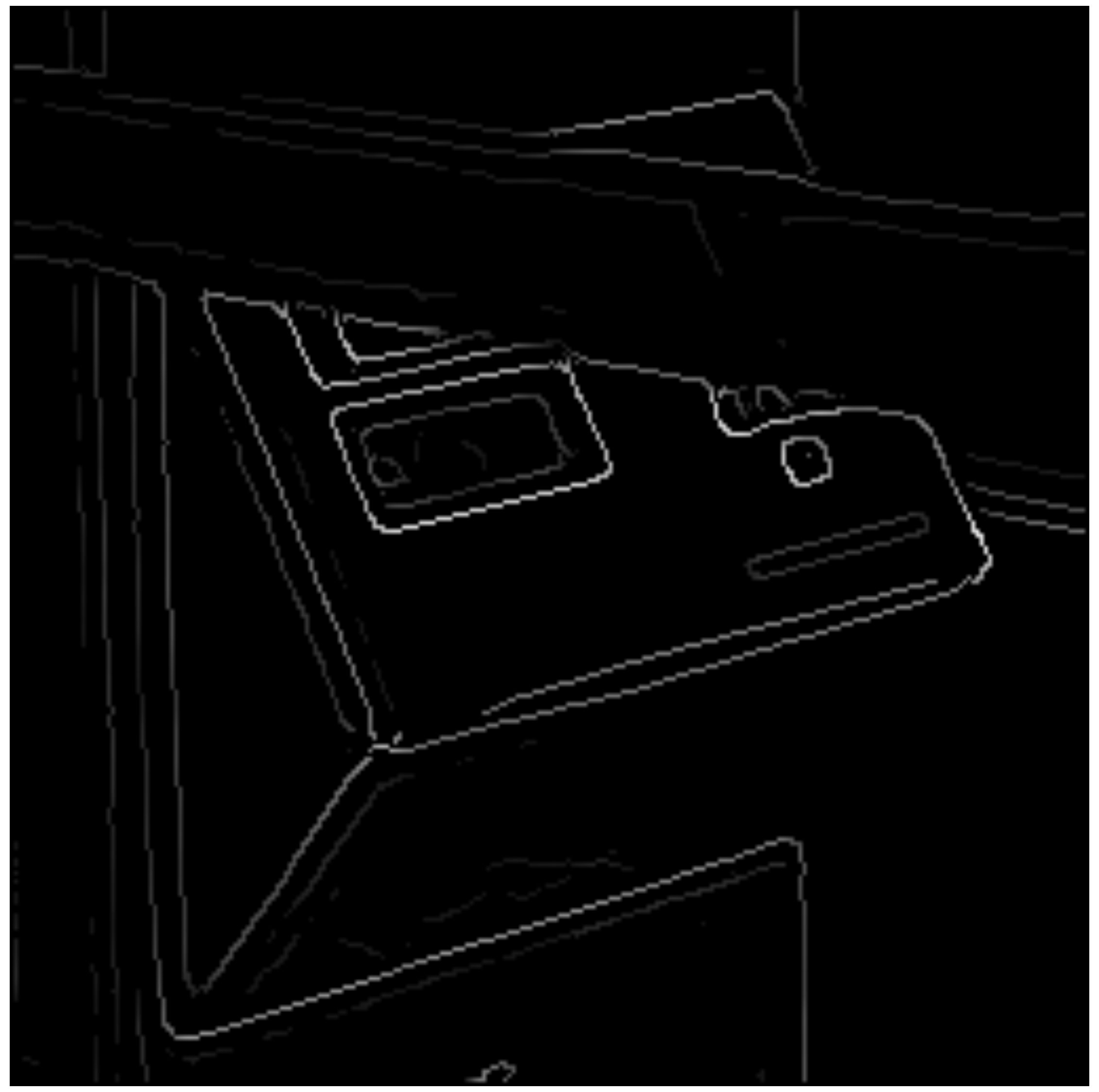
Real World **Example**



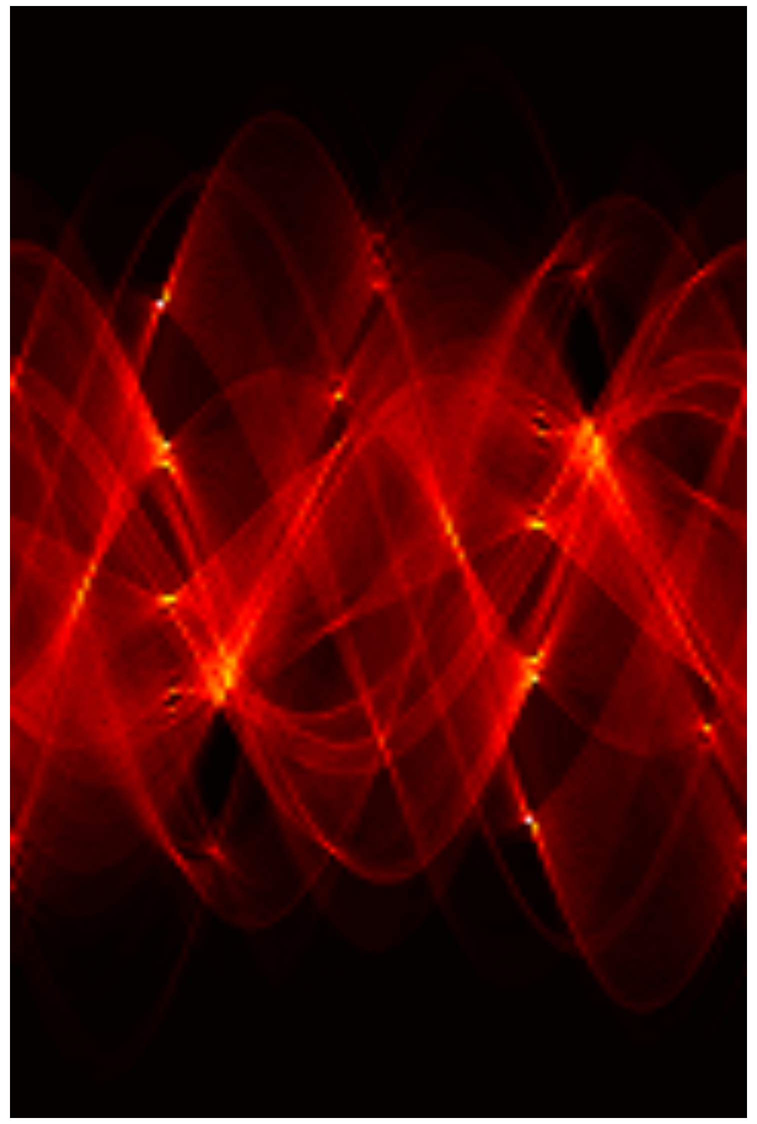
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Real World **Example**

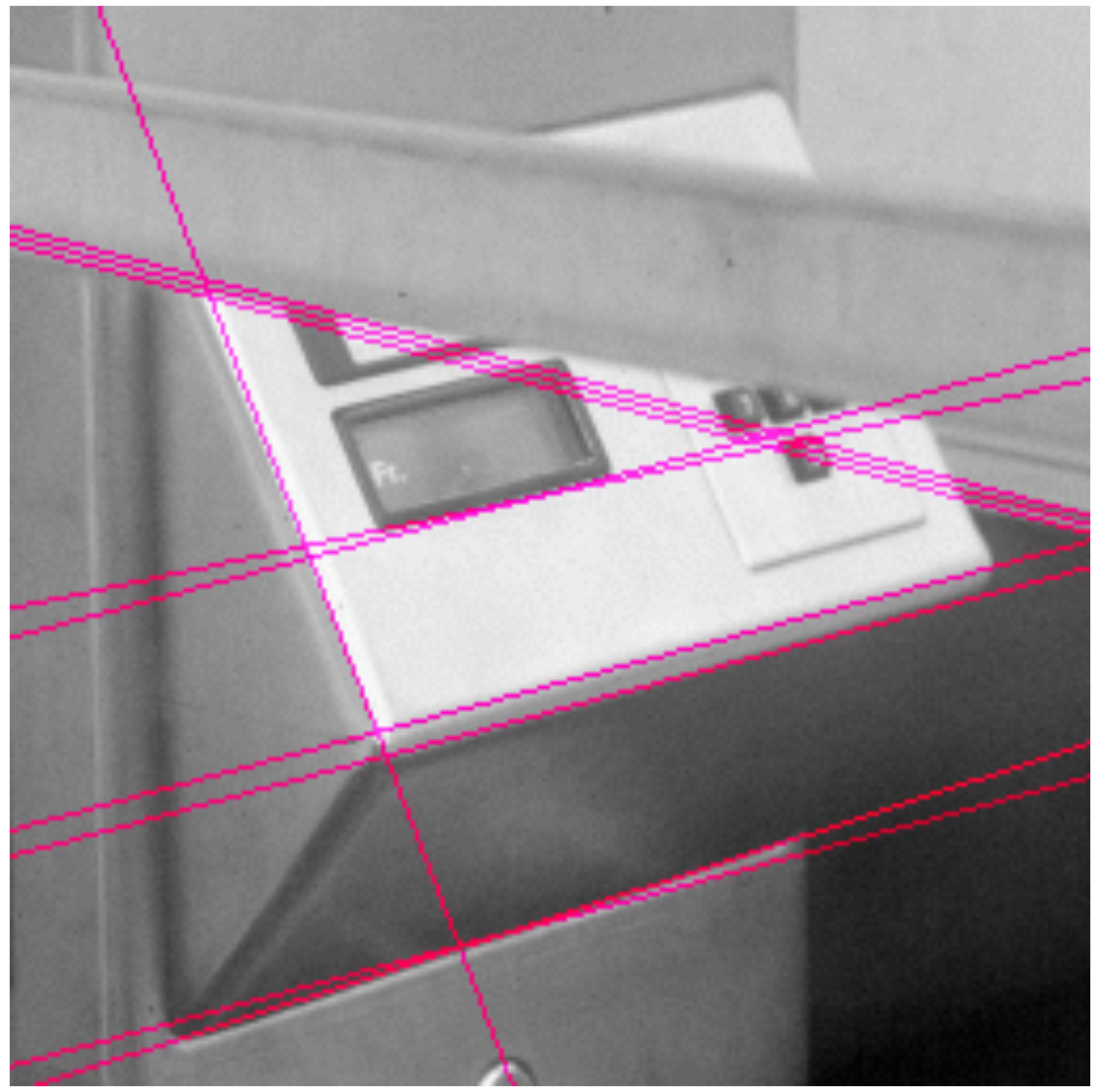
Original



Edges



Parameter space



Hough Lines

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Mechanics of Hough Transform

1. Construct a quantized array to represent θ and r

2. For each point, render curve (θ, r) into this array adding one vote at each cell

Difficulties:
— How big should the cells be? (too big, and we merge quite different lines; too small, and noise causes lines to be missed)

How many lines?

— Count the peaks in the Hough array

— Treat adjacent peaks as a single peak

Some **Practical Details** of Hough Transform

It is best to **vote** for the two closest bins in each dimension, as the locations of the bin boundaries are arbitrary

— This means that peaks are “blurred” and noise will not cause similar votes to fall into separate bins

Can use a **hash table** rather than an array to store the votes

— This means that no effort is wasted on initializing and checking empty bins

— It avoids the need to predict the maximum size of the array, which can be non-rectangular

Hough Transform: **Transformation Space** Voting

Sometimes a single point / measurement can vote on the entire transformation

e.g., **SIFT** keypoint matches with **location, scale and orientation** vote on the 4 parameters of a **similarity transform** (x,y,s,theta)

In this case, the votes of each sample can be seen as a **distribution in the parameter space** of the transformation

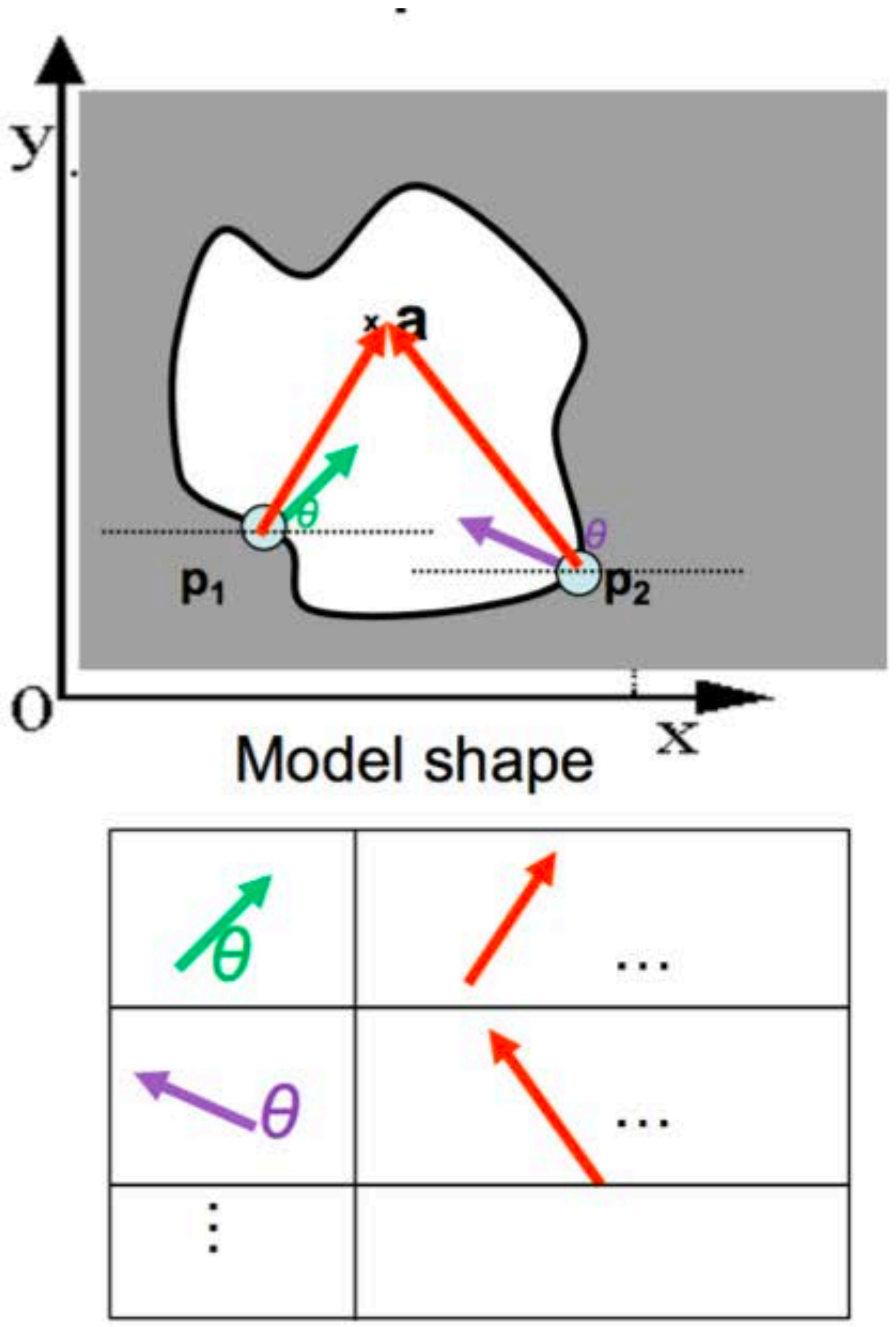
This can be effective in preventing noise in the distribution, e.g., edge detections with orientation can vote on **single lines** rather than **all lines** that pass through a point.

Generalized Hough Transform

What if we want to detect an **arbitrary** geometric shape?

Generalized Hough Transform

What if we want to detect an **arbitrary** geometric shape?



Offline procedure:

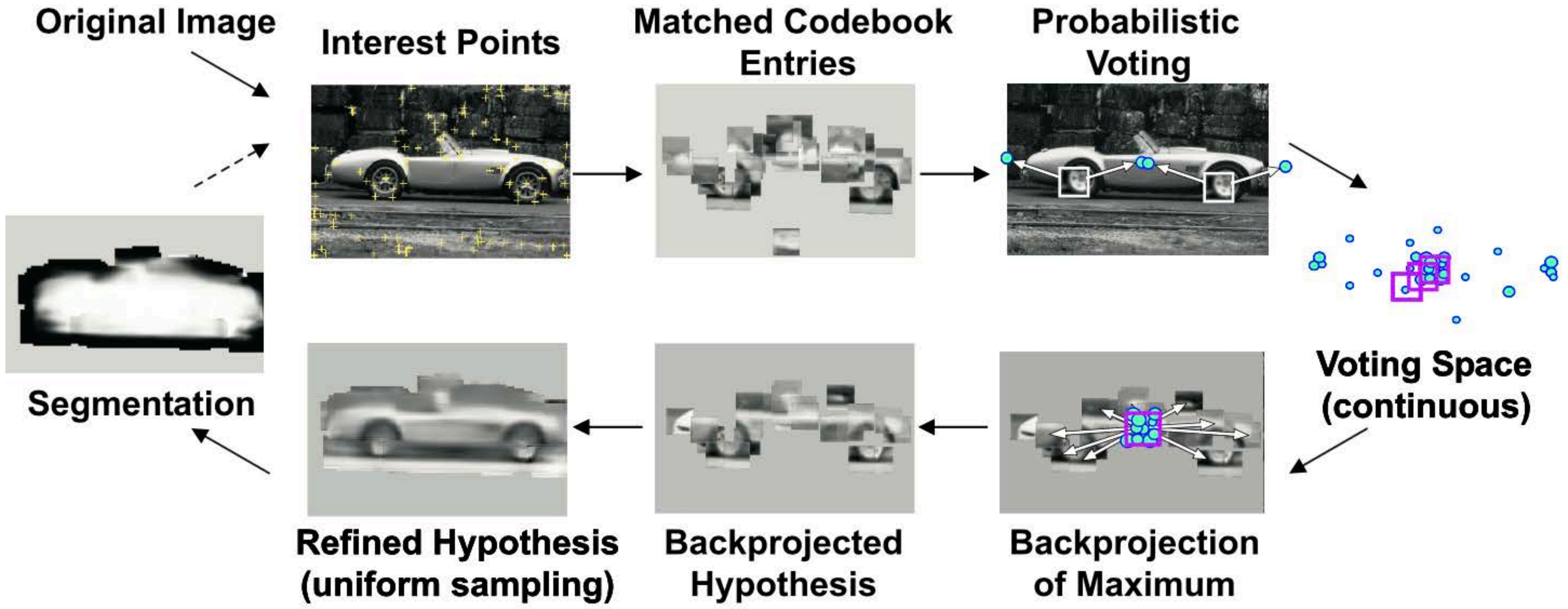
At each boundary point, compute displacement vector: $\mathbf{r} = \mathbf{a} - \mathbf{p}_i$.

Store these vectors in a table indexed by gradient orientation θ.

Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980

Example 1: Object Recognition — Implicit Shape Model

Combined object detection and segmentation using an implicit shape model.
Image patches cast weighted votes for the object centroid.



B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Example 1: Object Recognition — Implicit Shape Model

Basic Idea:

— Find **interest points/keypoints** in an image (e.g., SIFT Keypoint detector or Corners)

— **Match patch** around each interest point to a training patch (e.g., SIFT Descriptor)

— **Vote** for object center given that training instances

— Find the patches that voted for the peaks (**back-project**)

Example 1: Object Recognition — Implicit Shape Model

"**Training**" images of cows

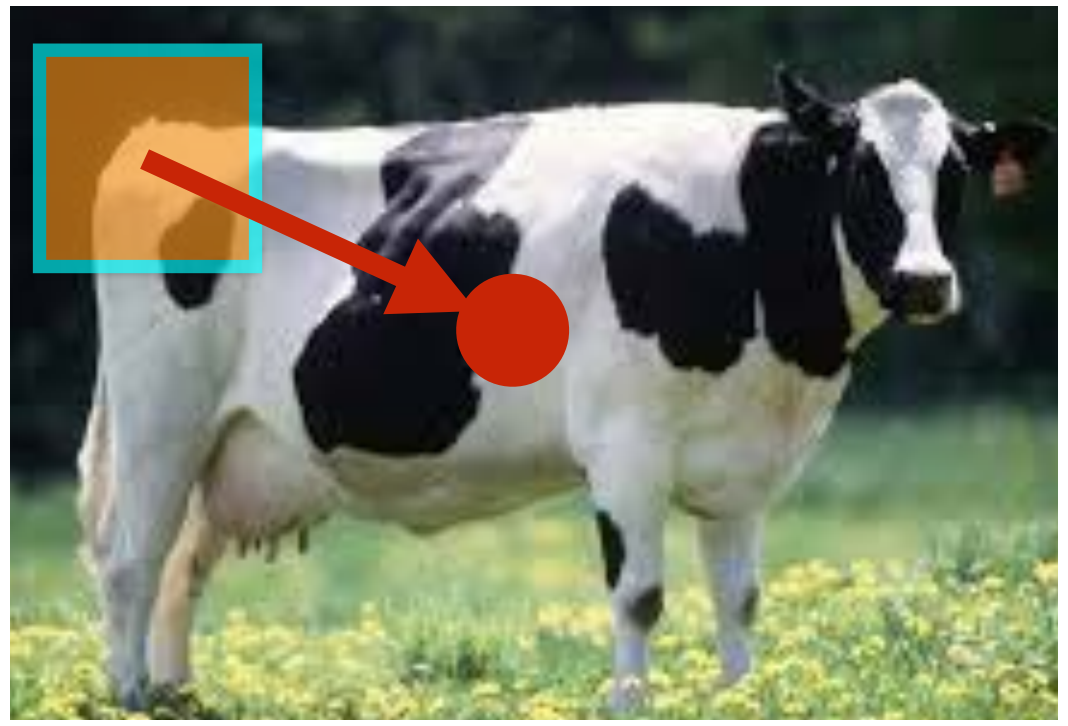


Image Index	Keypoint Index	Keypoint Detection (4D)	Keypoint Description (128D)	Offset to Centroid
Image 1	1	[x, y, s, Theta]	[…]	[x,y]

Example 1: Object Recognition — Implicit Shape Model

"**Training**" images of cows

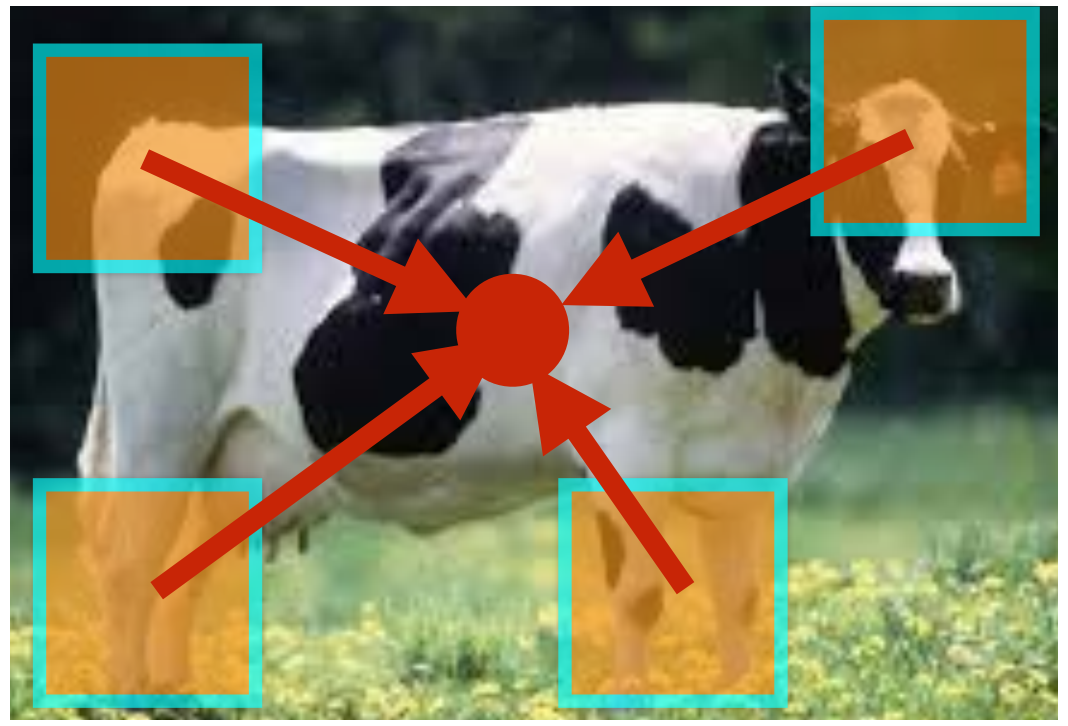
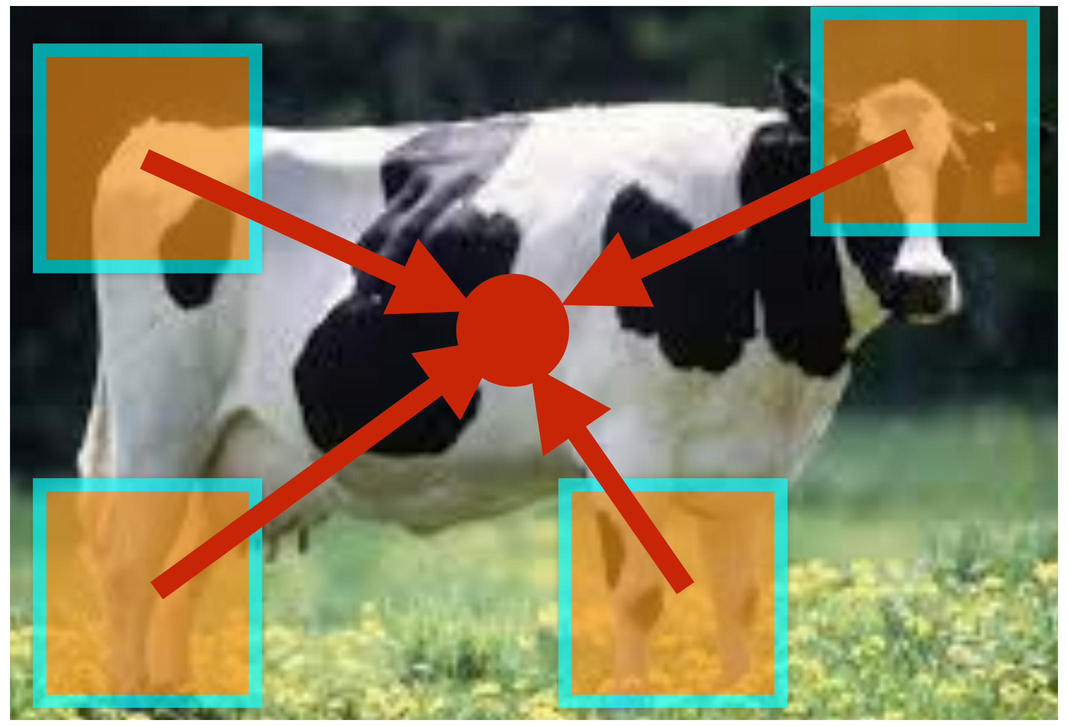


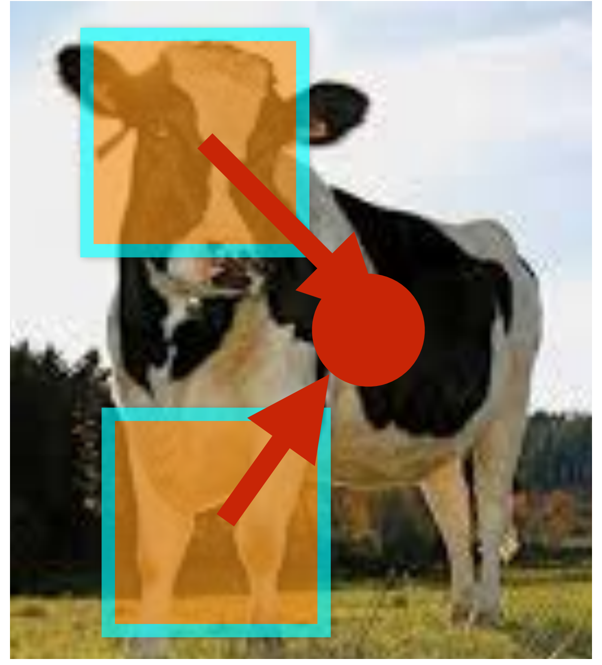
Image Index	Keypoint Index	Keypoint Detection (4D)	Keypoint Description (128D)	Offset to Centroid
Image 1	1	[x, y, s, Theta]	[…]	[x,y]
Image 1	2	[x, y, s, Theta]	[…]	[x,y]
….	….	….	…	…
Image 1	265	[x, y, s, Theta]	[…]	[x,y]

* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows





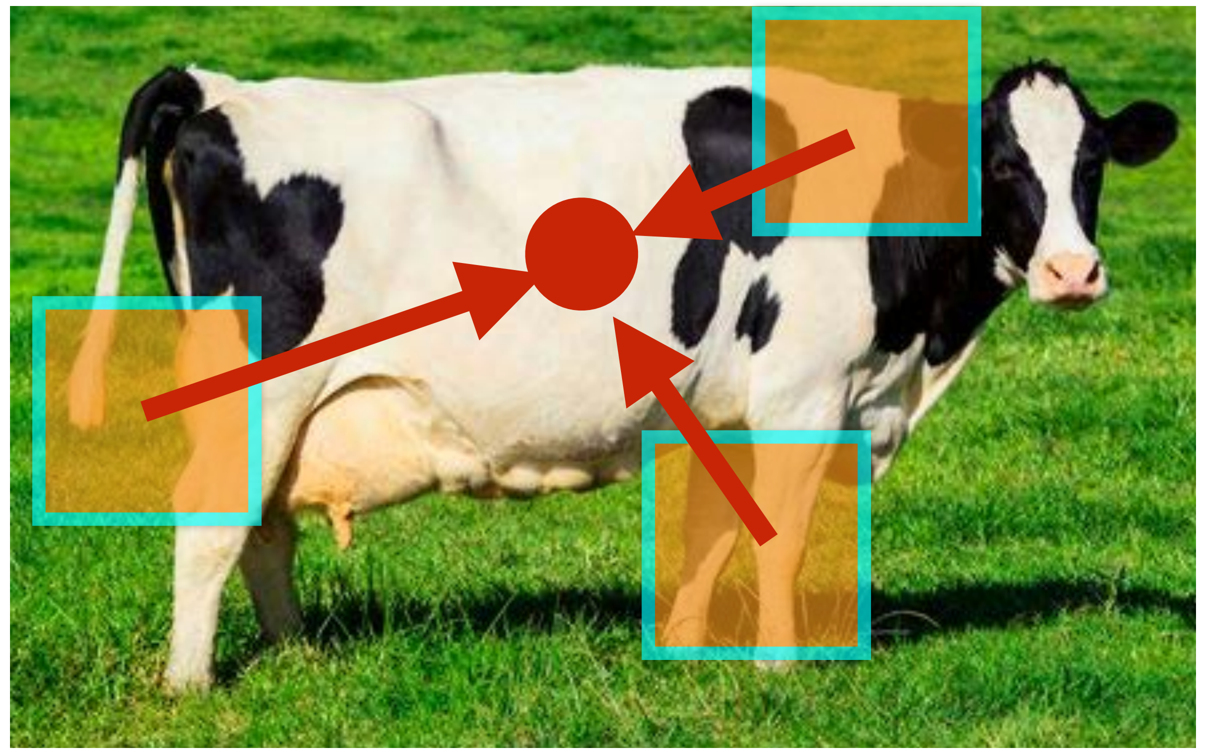


Image Index	Keypoint Index	Keypoint Detection (4D)	Keypoint Description (128D)	Offset to Centroid
Image 1	1	[x, y, s, Theta]	[…]	[x,y]
Image 1	2	[x, y, s, Theta]	[…]	[x,y]
….	….	….	…	…
Image 1	265	[x, y, s, Theta]	[…]	[x,y]
Image 2	1	[x, y, s, Theta]	[…]	[x,y]
Image 2	2	[x, y, s, Theta]	[…]	[x,y]
…	…	…	…	…
Image 2	645	[x, y, s, Theta]	[…]	[x,y]
Image K	1	[x, y, s, Theta]	[…]	[x,y]
Image K	2	[x, y, s, Theta]	[…]	[x,y]
…	…	…	…	…
Image K	134	[x, y, s, Theta]	[…]	[x,y]

* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows

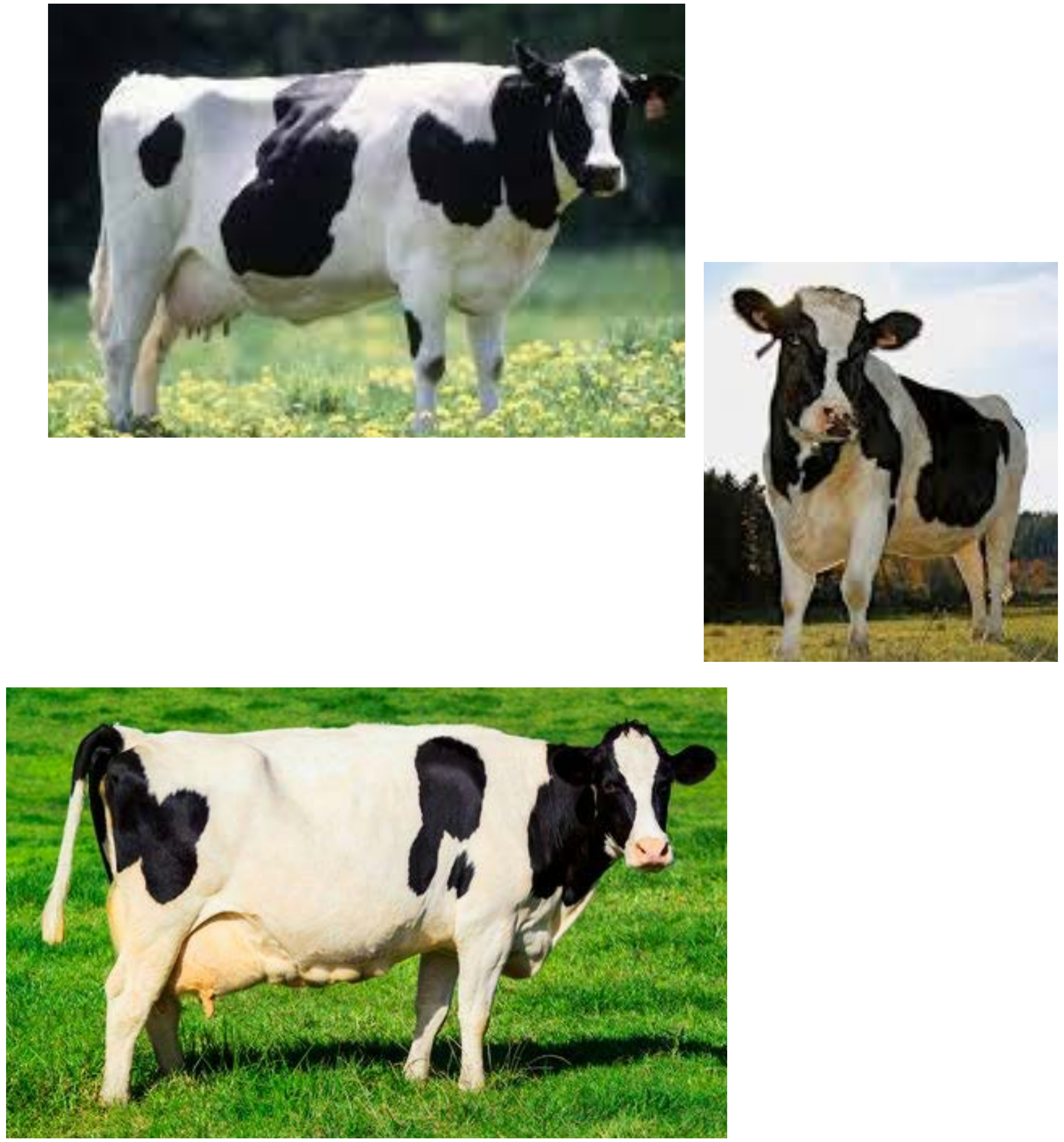


“**Testing**” image

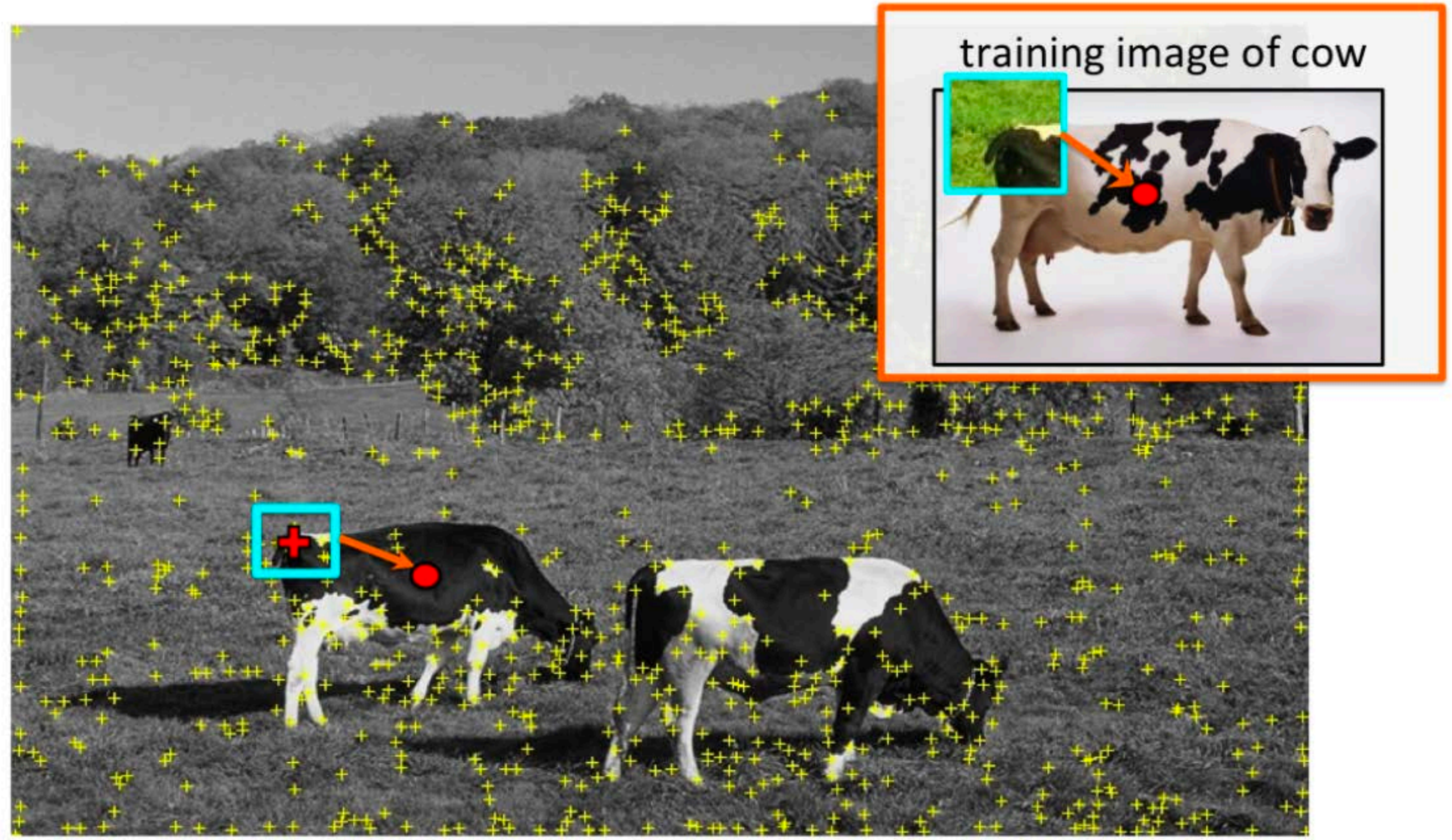
* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



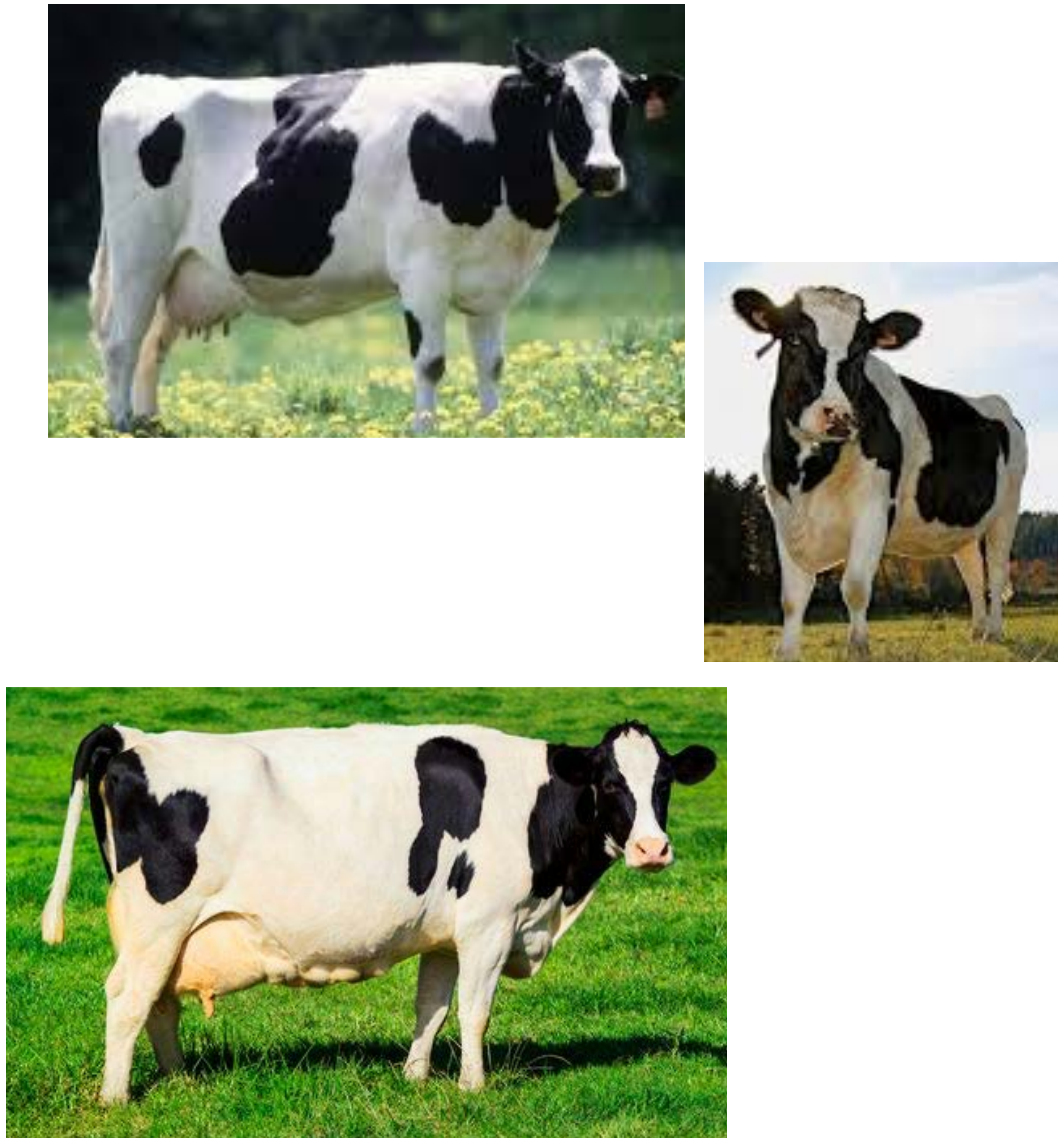
“**Testing**” image



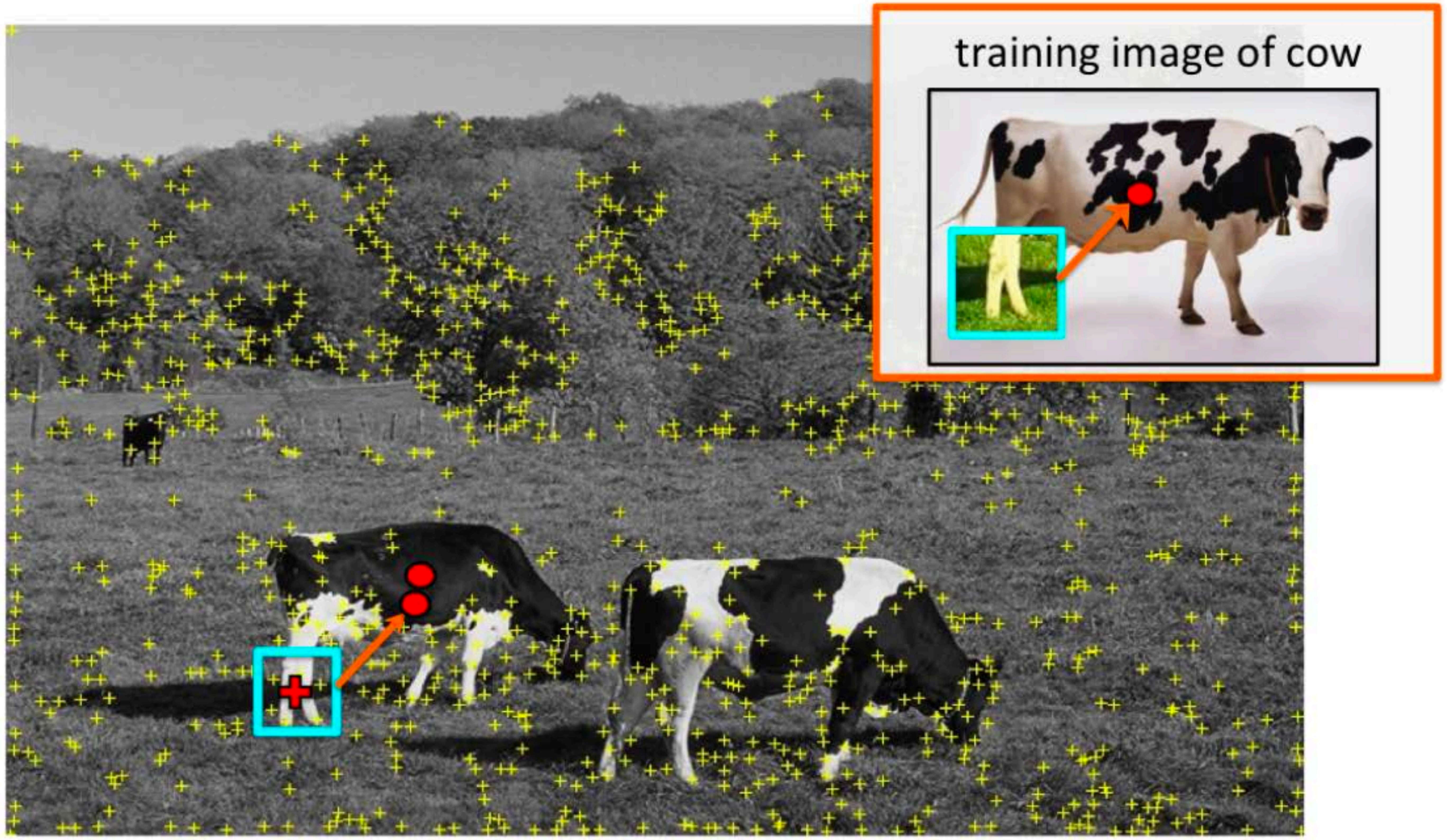
Vote for center of object

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



“**Testing**” image

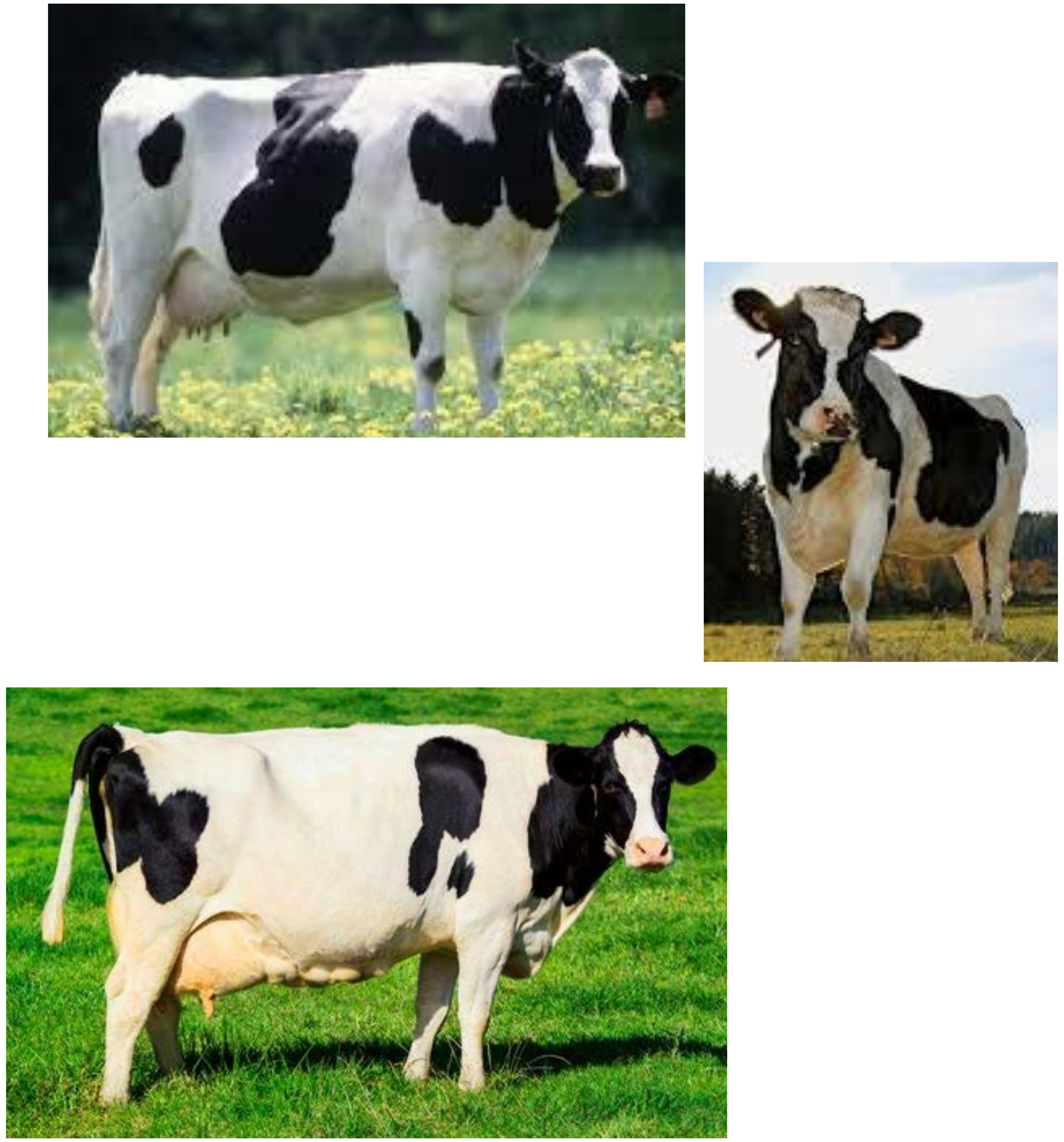


Vote for center of object

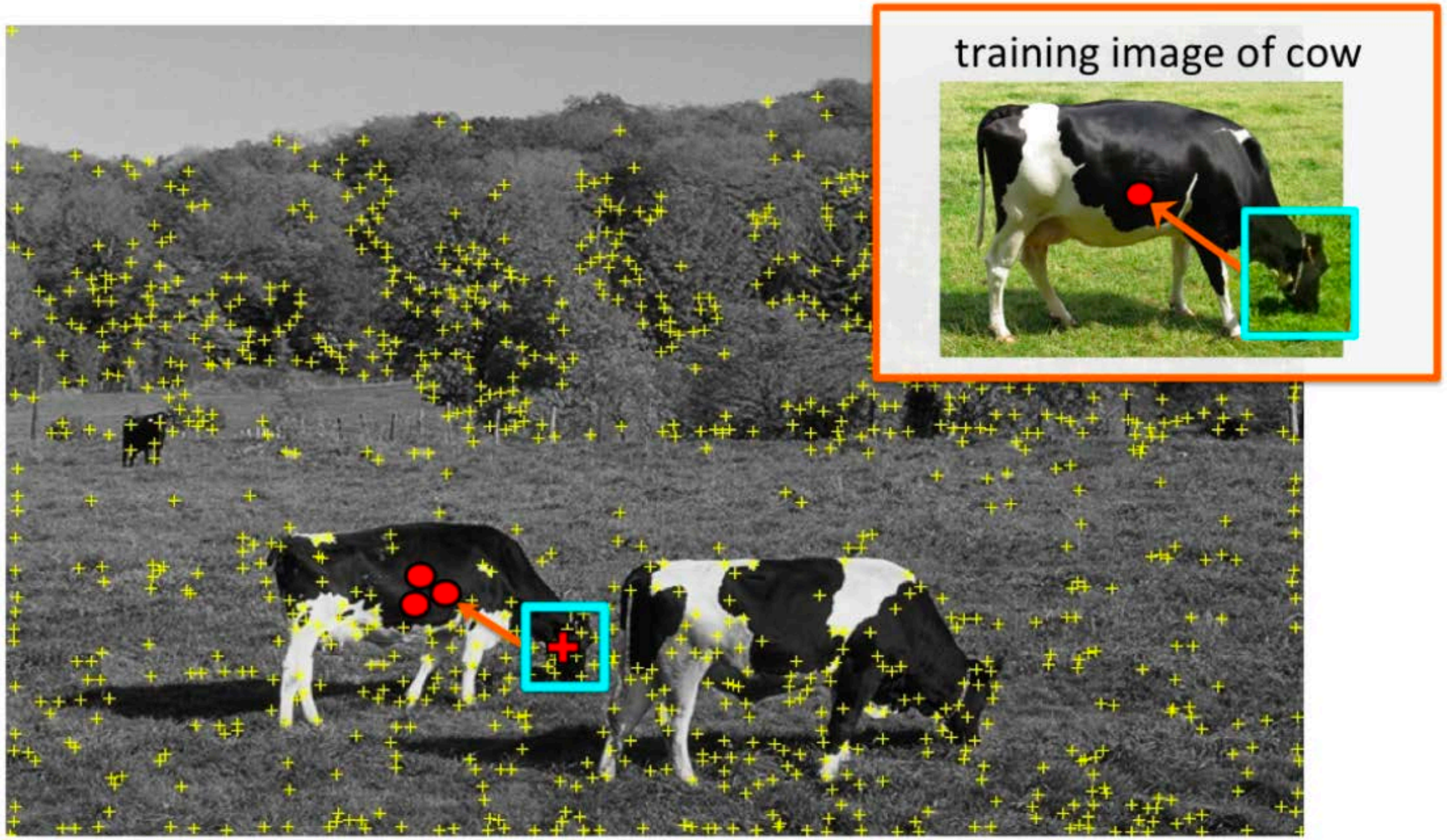
* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



“**Testing**” image

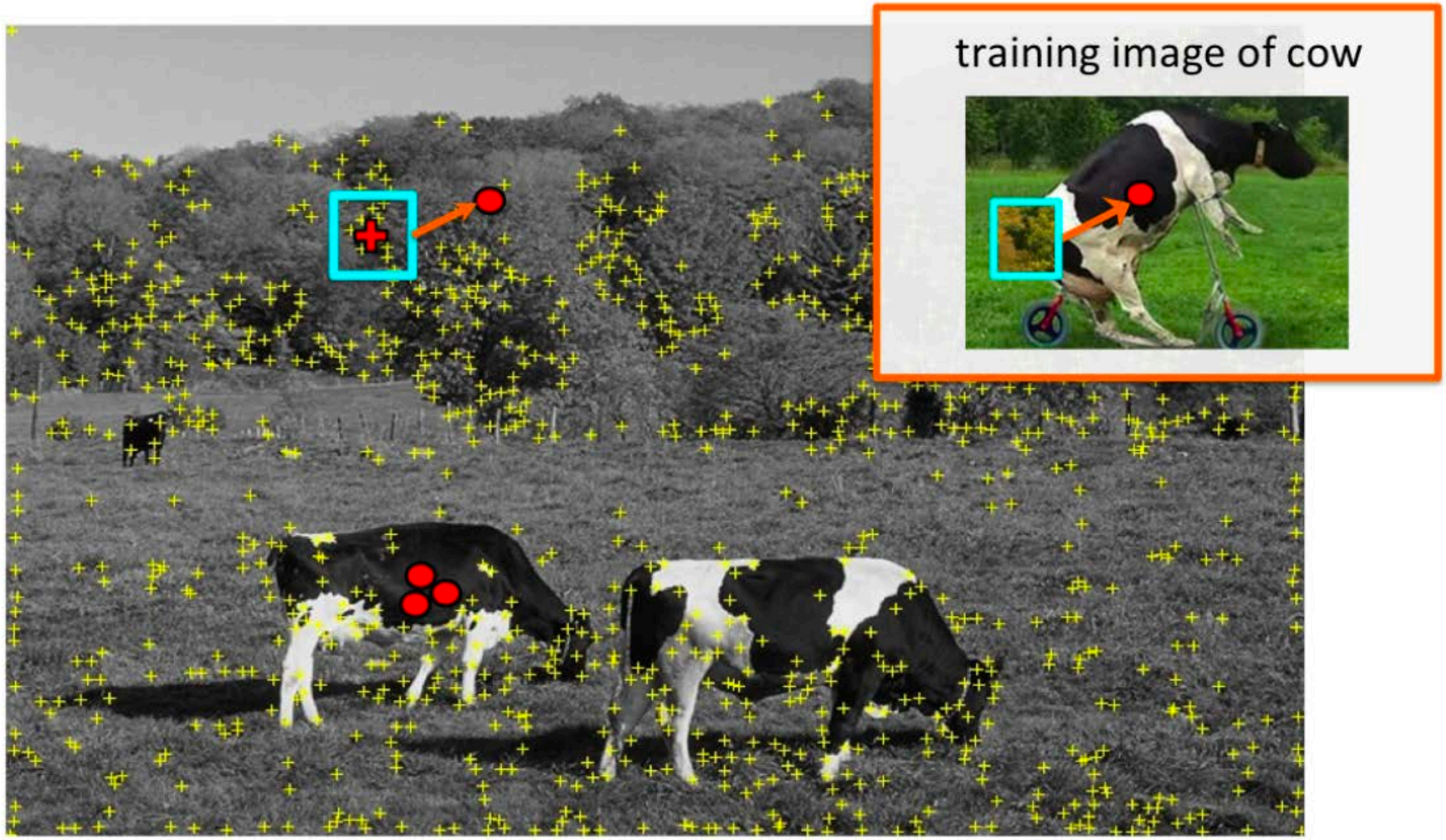


Vote for center of object

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows

“**Testing**” image



of course sometimes wrong votes are bound to happen

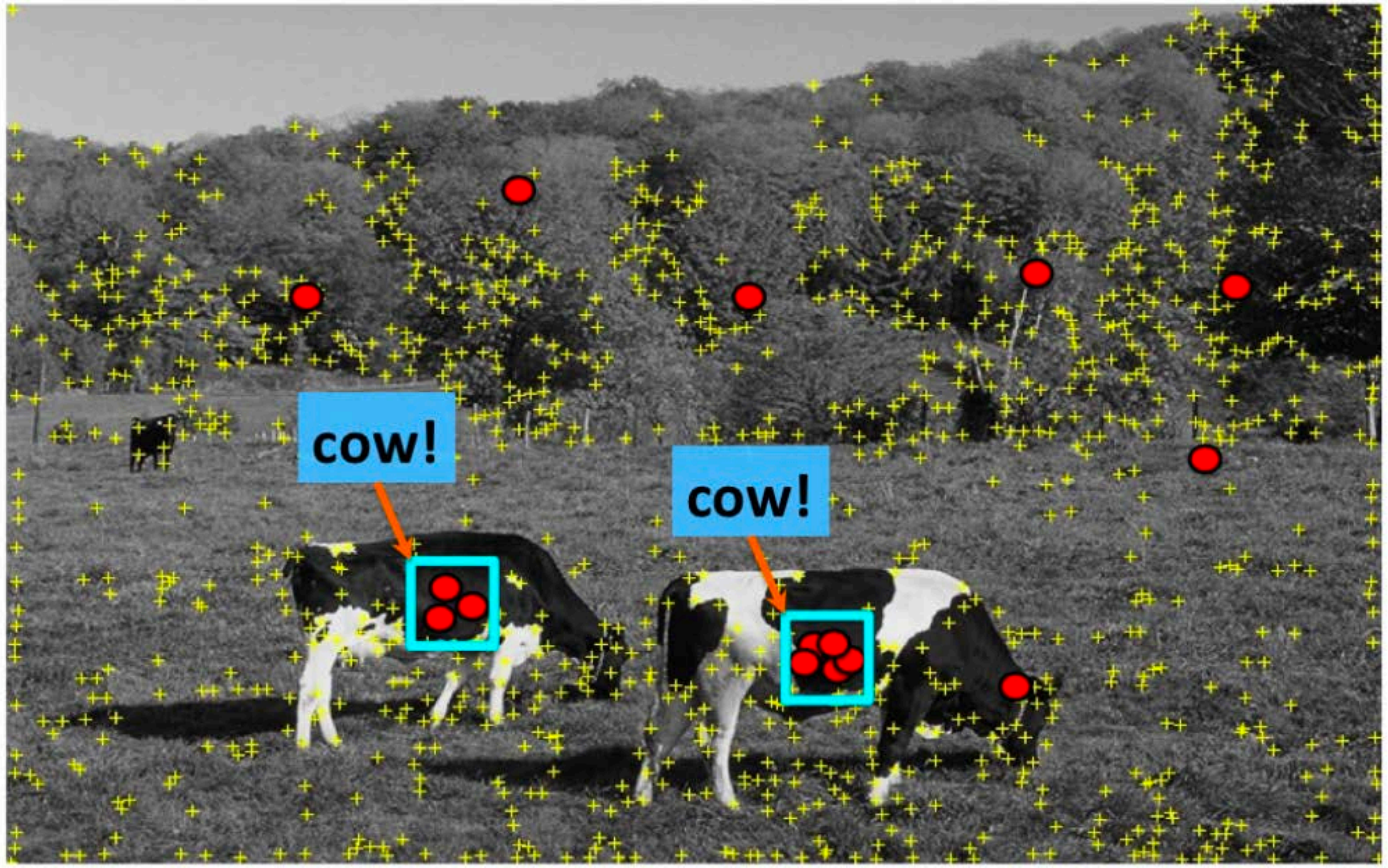
* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



“**Testing**” image



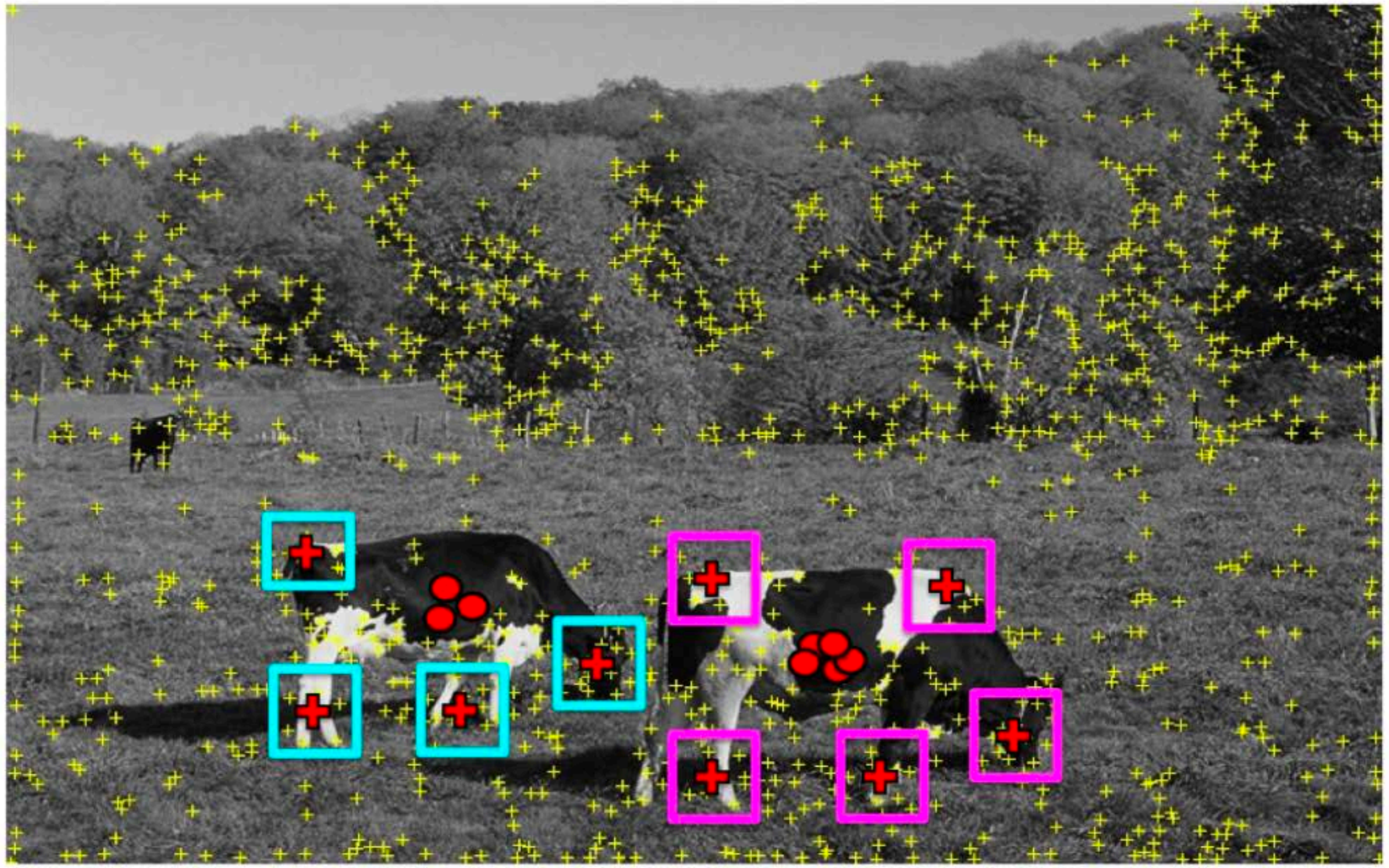
That’s ok. We want only **peaks** in voting space.

* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows

“**Testing**” image

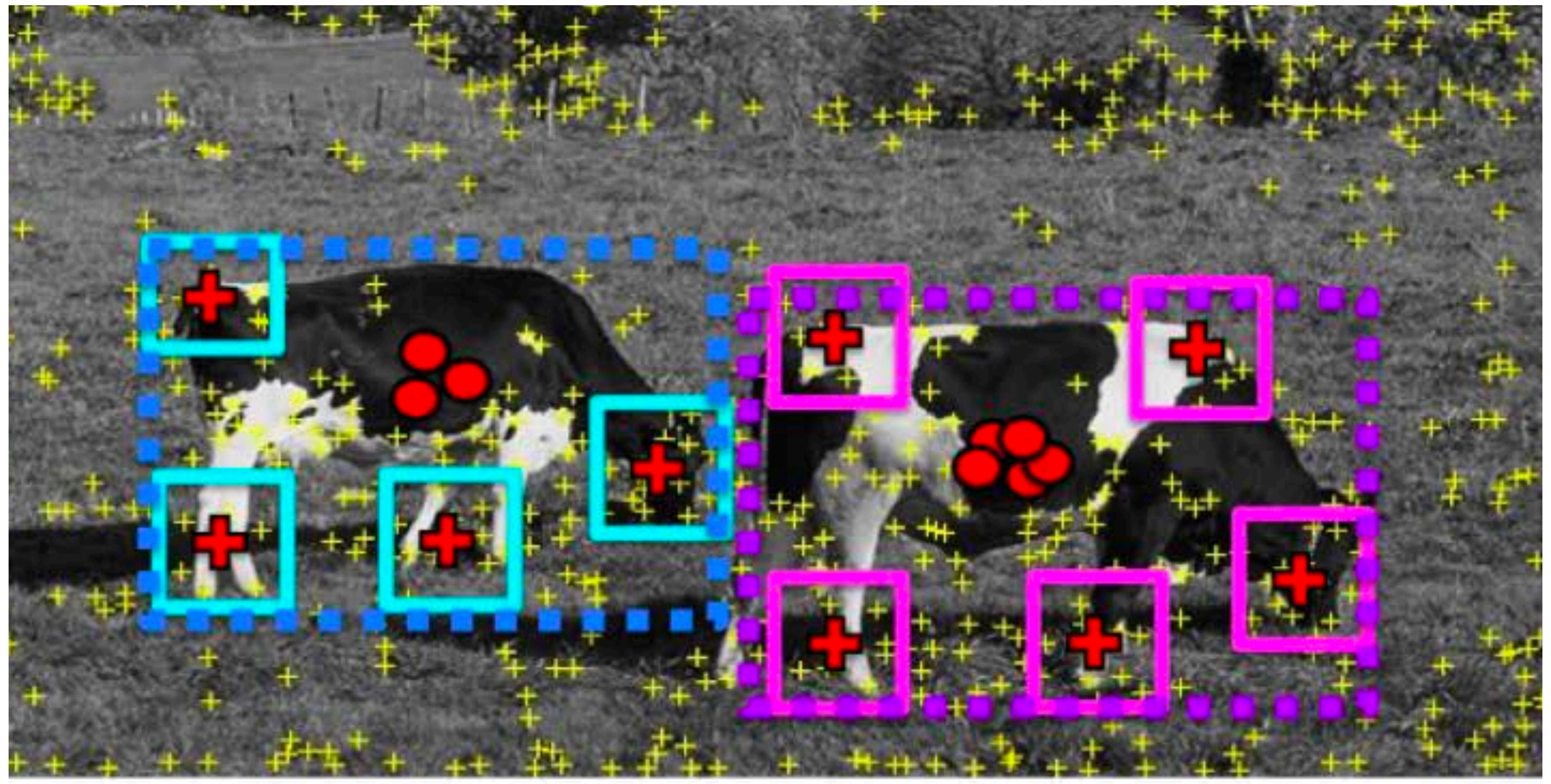


Find patches that voted for the peaks (back-project)

* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

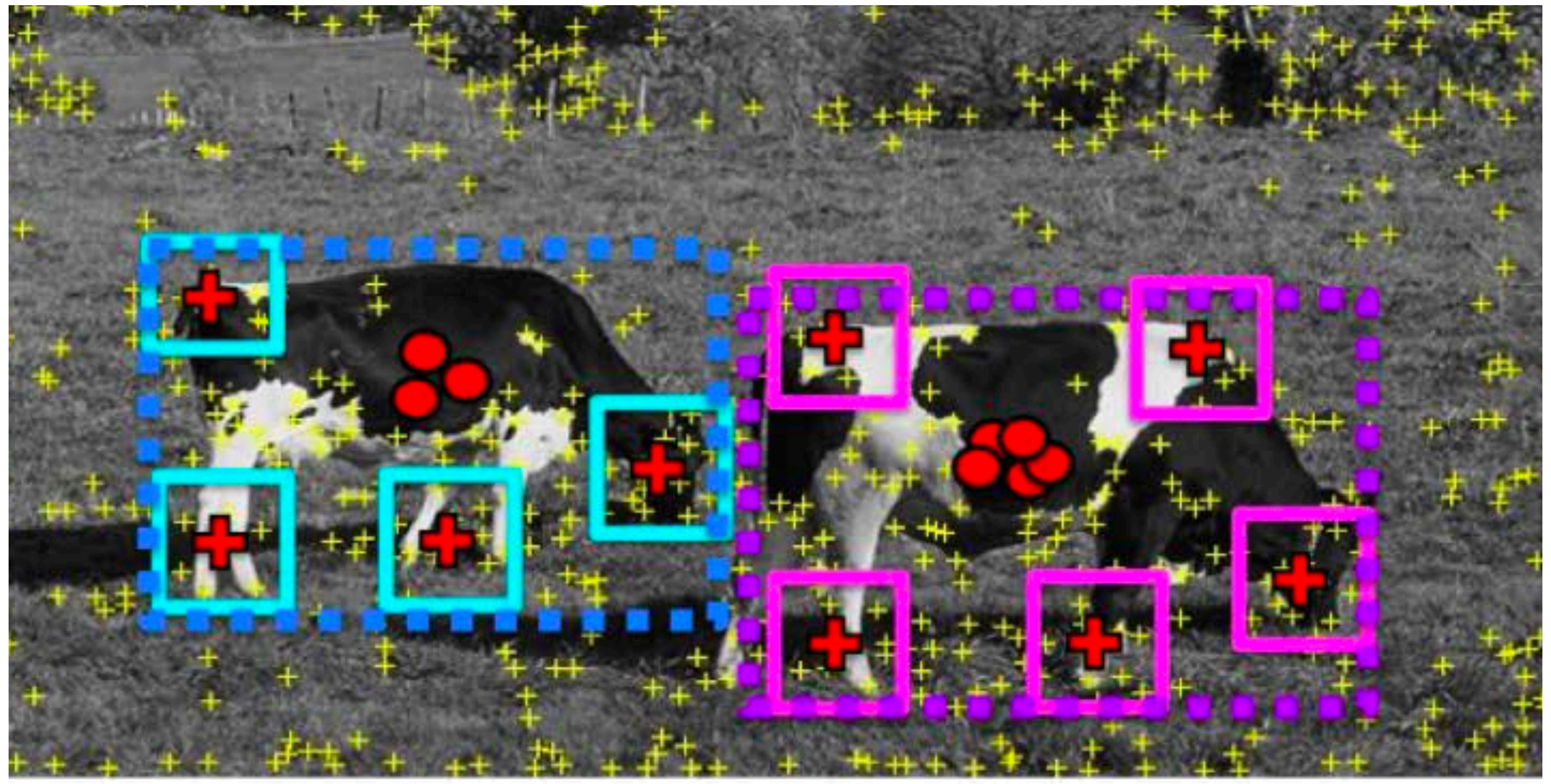
Image Index	Keypoint Index	Keypoint Detection (4D)	Keypoint Description (128D)	Offset to Centroid
Image 1	1	[x, y, s, Theta]	[…]	[x,y]
Image 1	2	[x, y, s, Theta]	[…]	[x,y]
….	….	….	…	…
Image 1	265	[x, y, s, Theta]	[…]	[x,y]
Image 2	1	[x, y, s, Theta]	[…]	[x,y]
Image 2	2	[x, y, s, Theta]	[…]	[x,y]
…	…	…	…	…
Image 2	645	[x, y, s, Theta]	[…]	[x,y]
Image K	1	[x, y, s, Theta]	[…]	[x,y]
Image K	2	[x, y, s, Theta]	[…]	[x,y]
…	…	…	…	…
Image K	134	[x, y, s, Theta]	[…]	[x,y]



* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

Image Index	Keypoint Index	Keypoint Detection (4D)	Keypoint Description (128D)	Offset to Centroid
Image 1	1	[x, y, s, Theta]	[…]	[x,y]
Image 1	2	[x, y, s, Theta]	[…]	[x,y]
….	….	….	…	…
Image 1	265	[x, y, s, Theta]	[…]	[x,y]
Image 2	1	[x, y, s, Theta]	[…]	[x,y]
Image 2	2	[x, y, s, Theta]	[…]	[x,y]
…	…	…	…	…
Image 2	645	[x, y, s, Theta]	[…]	[x,y]
Image K	1	[x, y, s, Theta]	[…]	[x,y]
Image K	2	[x, y, s, Theta]	[…]	[x,y]
…	…	…	…	…
Image K	134	[x, y, s, Theta]	[…]	[x,y]



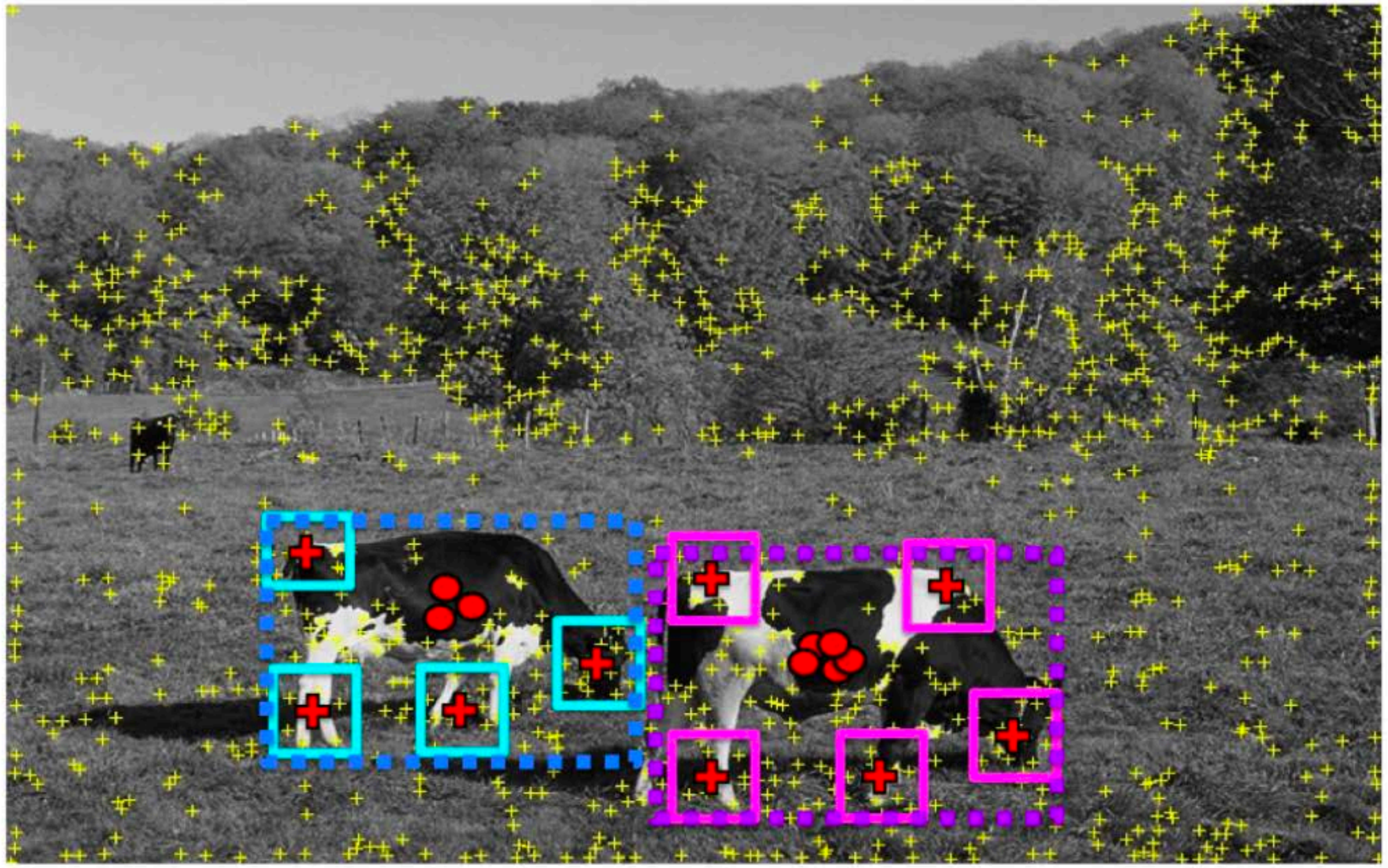
* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows

“**Testing**” image

box around patches = object



Find objects based on the back projected patches

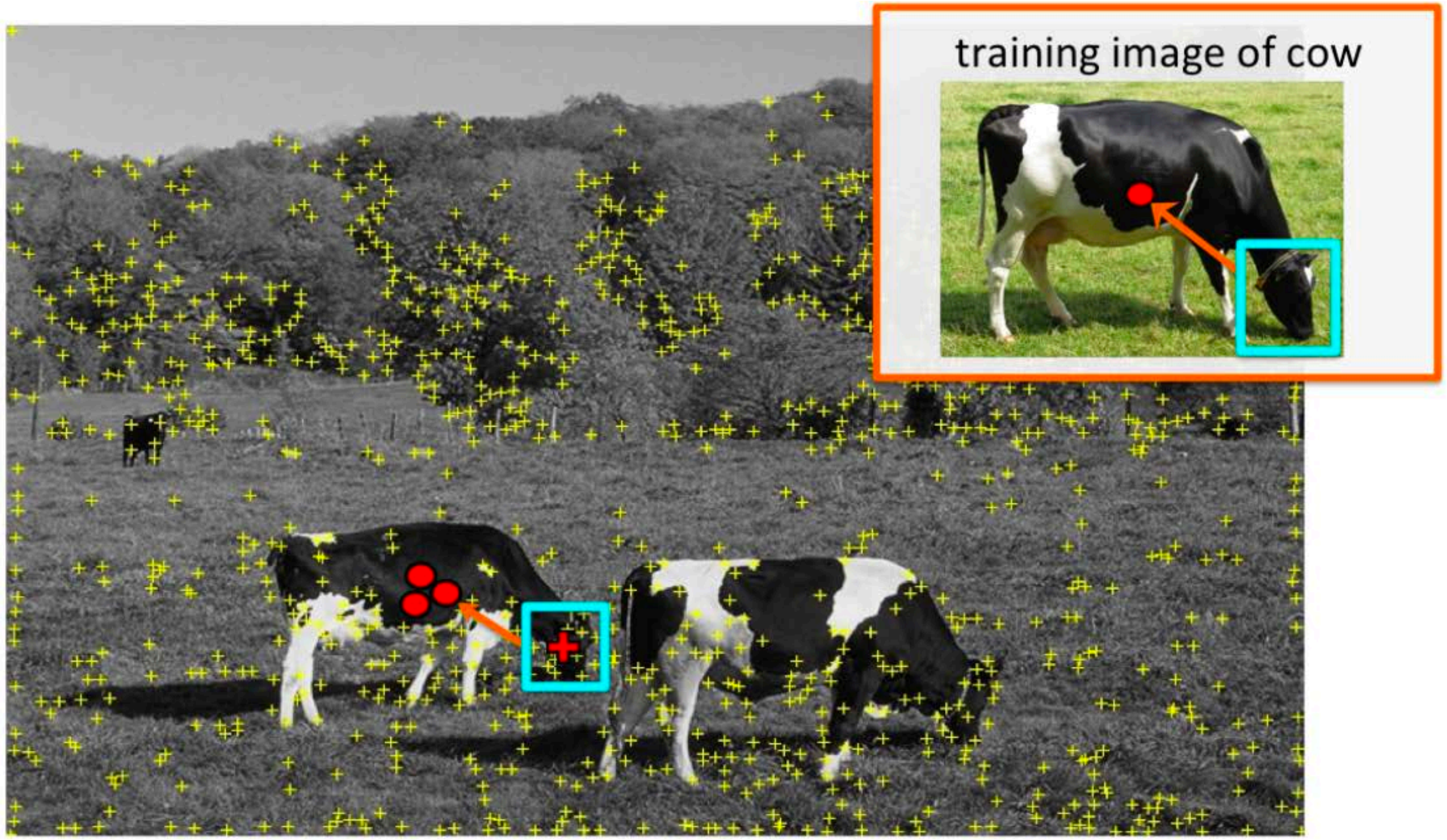
* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

"**Training**" images of cows

"**Testing**" image

Really easy … but slow … how do we make it fast?

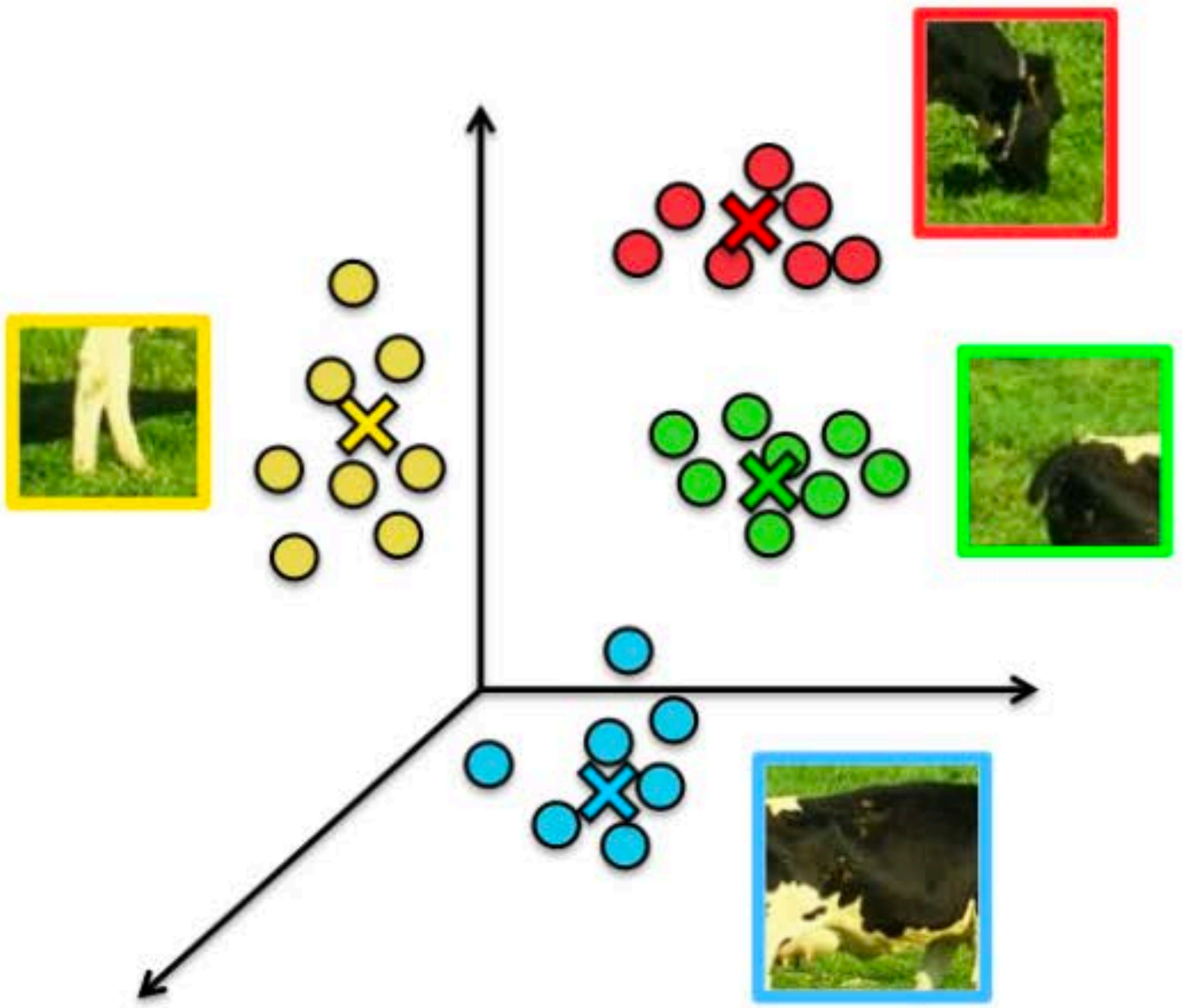


We need to match a patch around each yellow keypoint to all patches in all training images (**slow**)

* Slide from Sanja Fidler

Visual Words

- **Visual vocabulary** (we saw this for retrieval)
- Compare each patch to a small set of visual words (clusters)

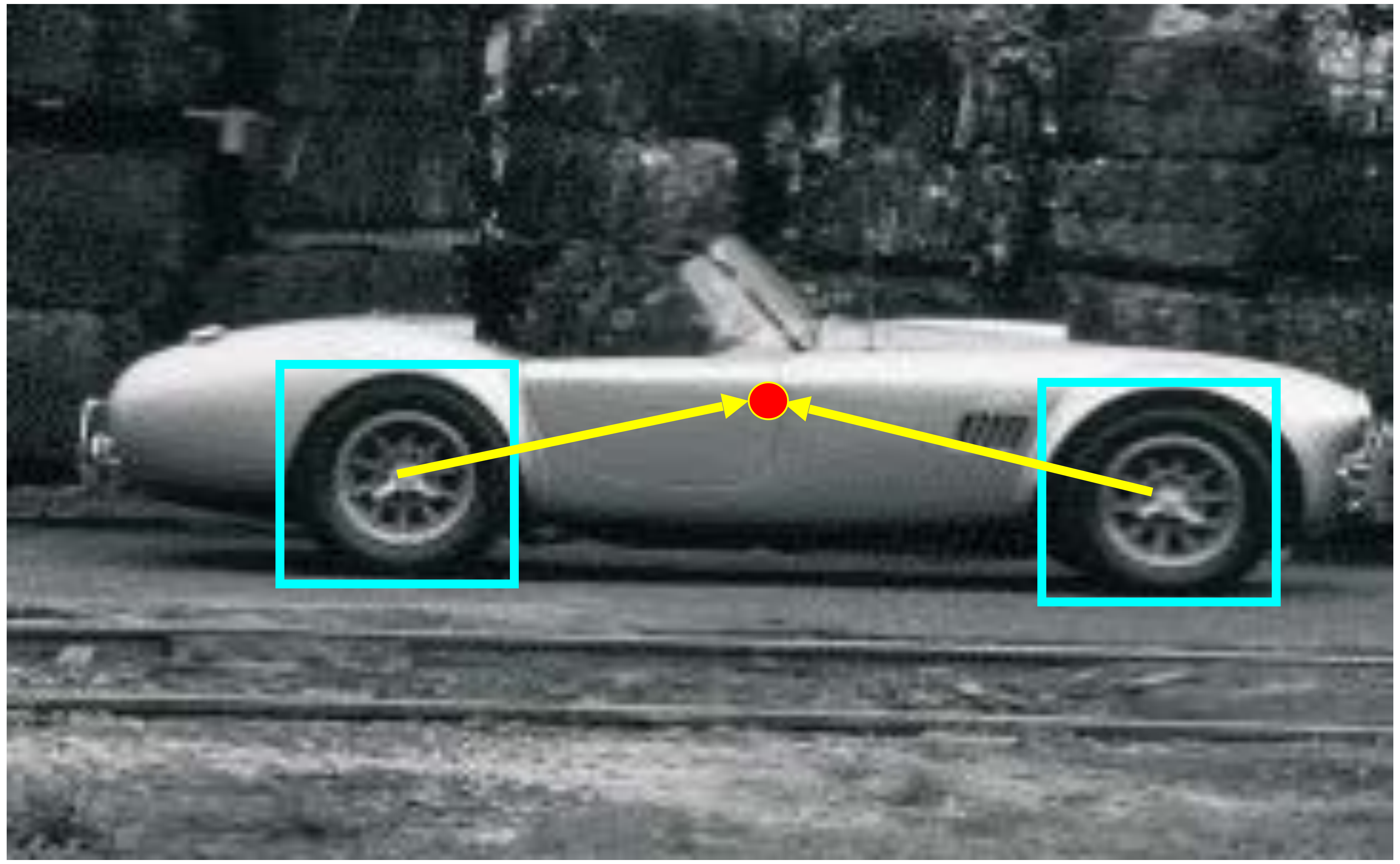


Visual words (visual codebook)!

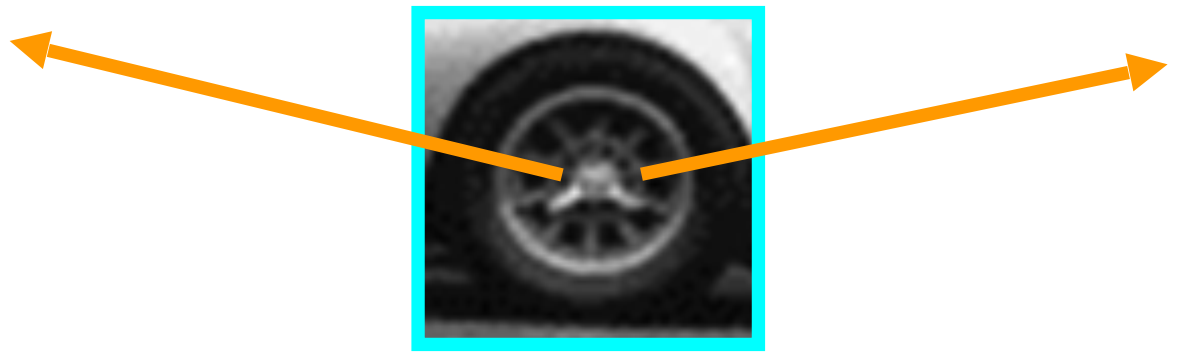
* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

Index displacements by "visual codeword"



training image



visual codeword with
displacement vectors

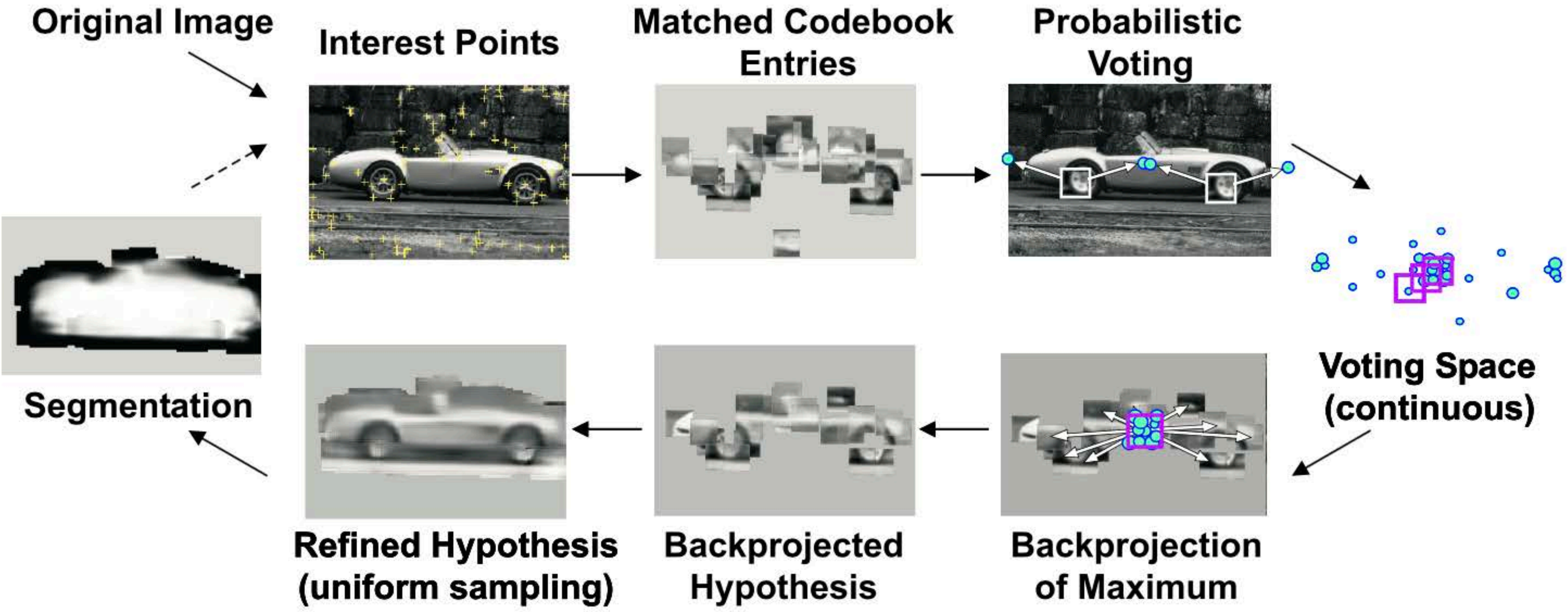
B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model,
ECCV Workshop on Statistical Learning in Computer Vision 2004

Example 1: Object Recognition — Implicit Shape Model

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model,
ECCV Workshop on Statistical Learning in Computer Vision 2004

Inferring Other Information: **Segmentation**

Combined object detection and segmentation using an implicit shape model.
Image patches cast weighted votes for the object centroid.



B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows

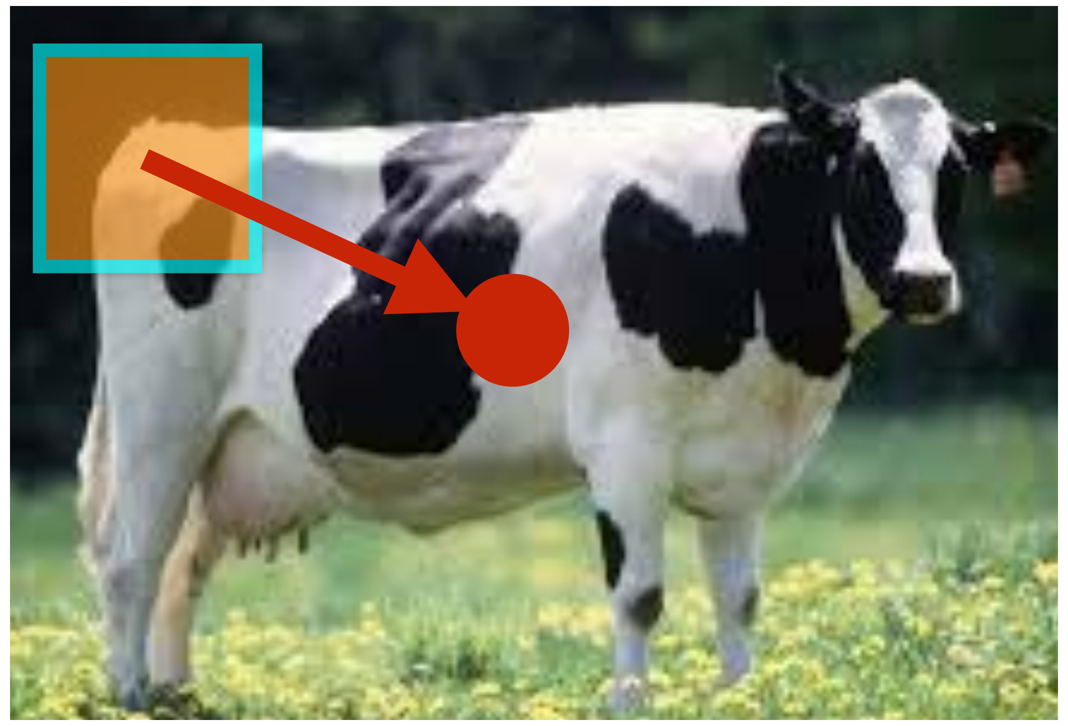
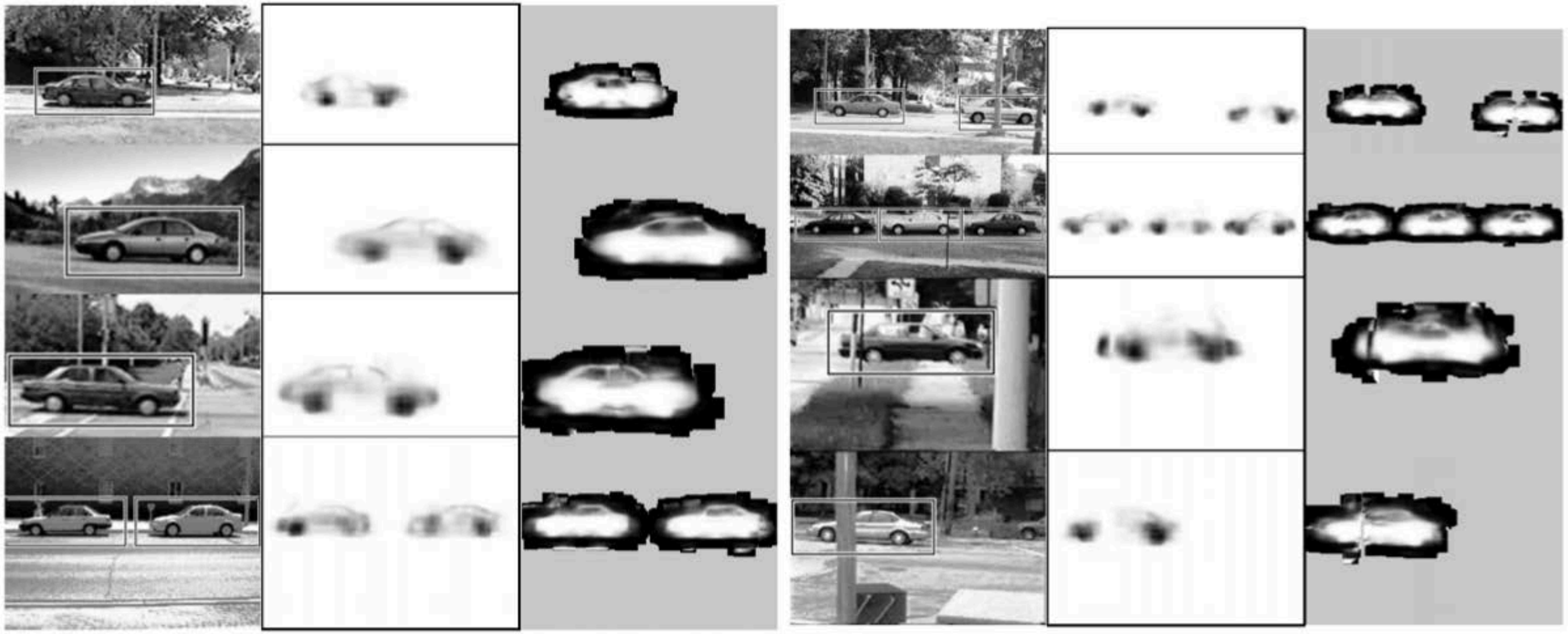


Image Index	Keypoint Index	Keypoint Detection (4D)	Keypoint Description (128D)	Offset to Centroid	Segment
Image 1	1	[x, y, s, Theta]	[…]	[x,y]	
Image 1	2	[x, y, s, Theta]	[…]	[x,y]	
….	….	….	…	…	
Image 1	265	[x, y, s, Theta]	[…]	[x,y]	
Image 2	1	[x, y, s, Theta]	[…]	[x,y]	
Image 2	2	[x, y, s, Theta]	[…]	[x,y]	
…	…	…	…	…	
Image 2	645	[x, y, s, Theta]	[…]	[x,y]	
Image K	1	[x, y, s, Theta]	[…]	[x,y]	
Image K	2	[x, y, s, Theta]	[…]	[x,y]	
…	…	…	…	…	
Image K	134	[x, y, s, Theta]	[…]	[x,y]	

* Slide from Sanja Fidler

Inferring Other Information: **Segmentation**

Idea: When back-projecting, back-project labeled segmentations per training patch

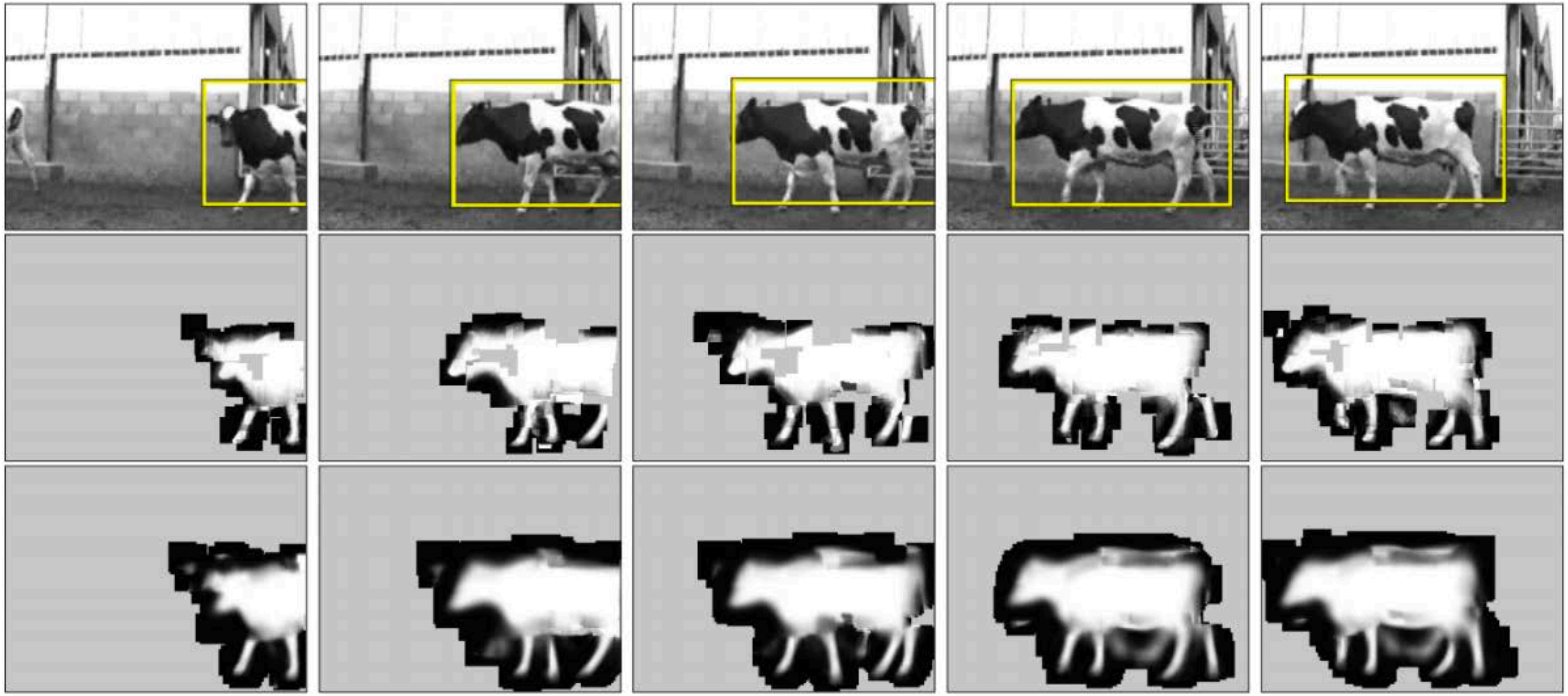


(a) detections (b) p(figure) (c) segmentation (a) detections (b) p(figure) (c) segmentation

[Source: B. Leibe]

* Slide from Sanja Fidler

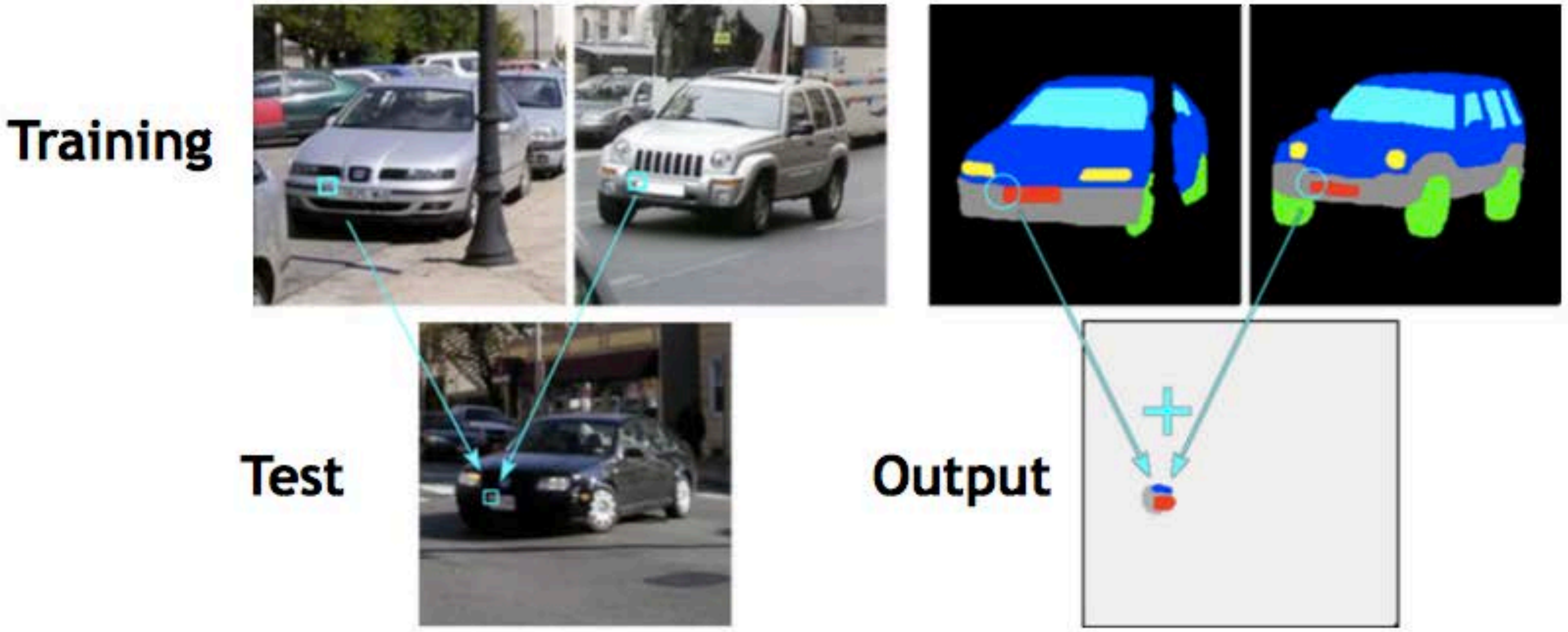
Inferring Other Information: **Segmentation**



[Source: B. Leibe]

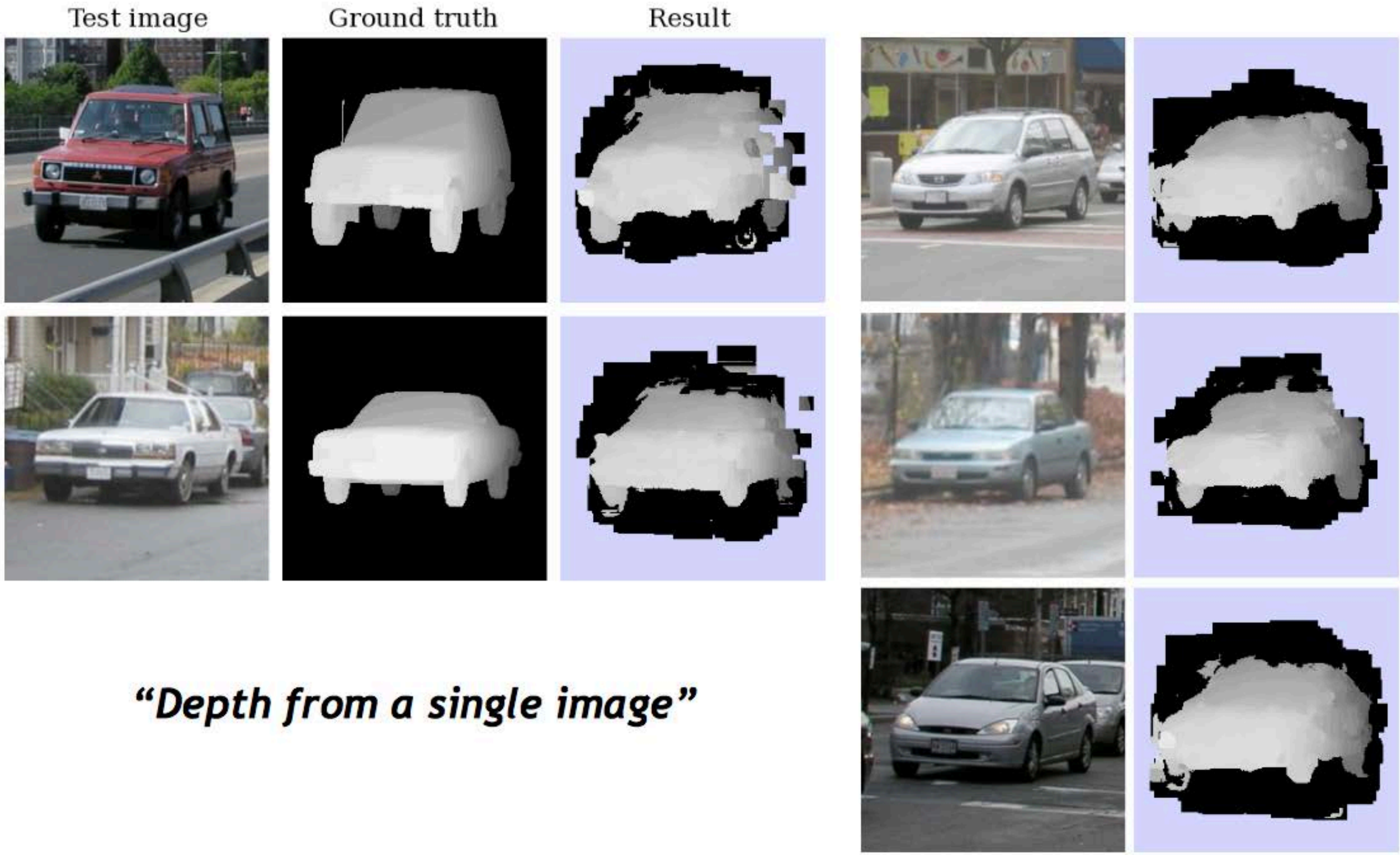
* Slide from Sanja Fidler

Inferring Other Information: **Part Labels**



* Slide from Sanja Fidler

Inferring Other Information: **Depth**



"Depth from a single image"

* Slide from Sanja Fidler

Example 2: Object Recognition — Boundary Fragments

Boundary fragments cast weighted votes for the object centroid. Also obtains an estimate of the object's contour.

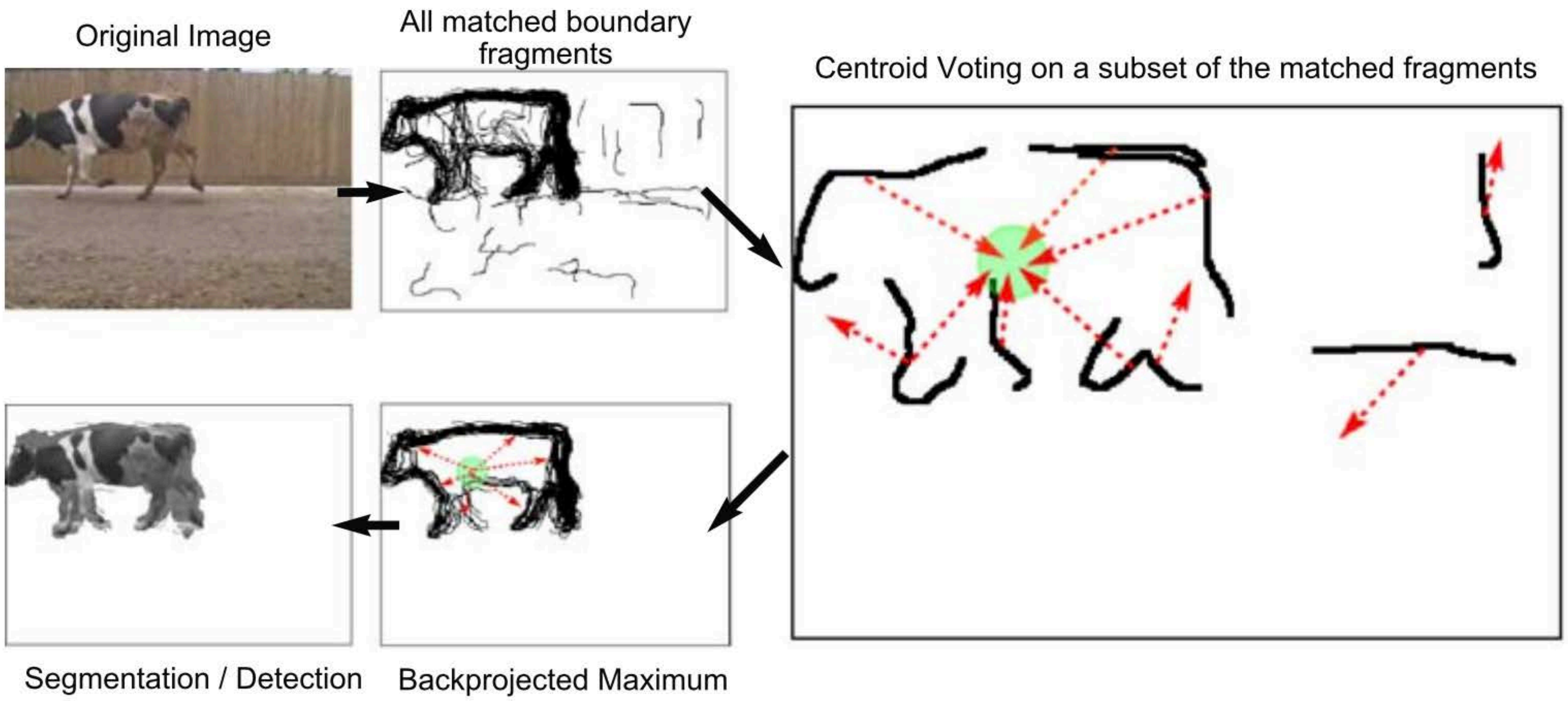


Image credit: Opelt et al., 2006

Example 2: Object Recognition — Boundary Fragments

Boundary fragments cast weighted votes for the object centroid. Also obtains an estimate of the object's contour.

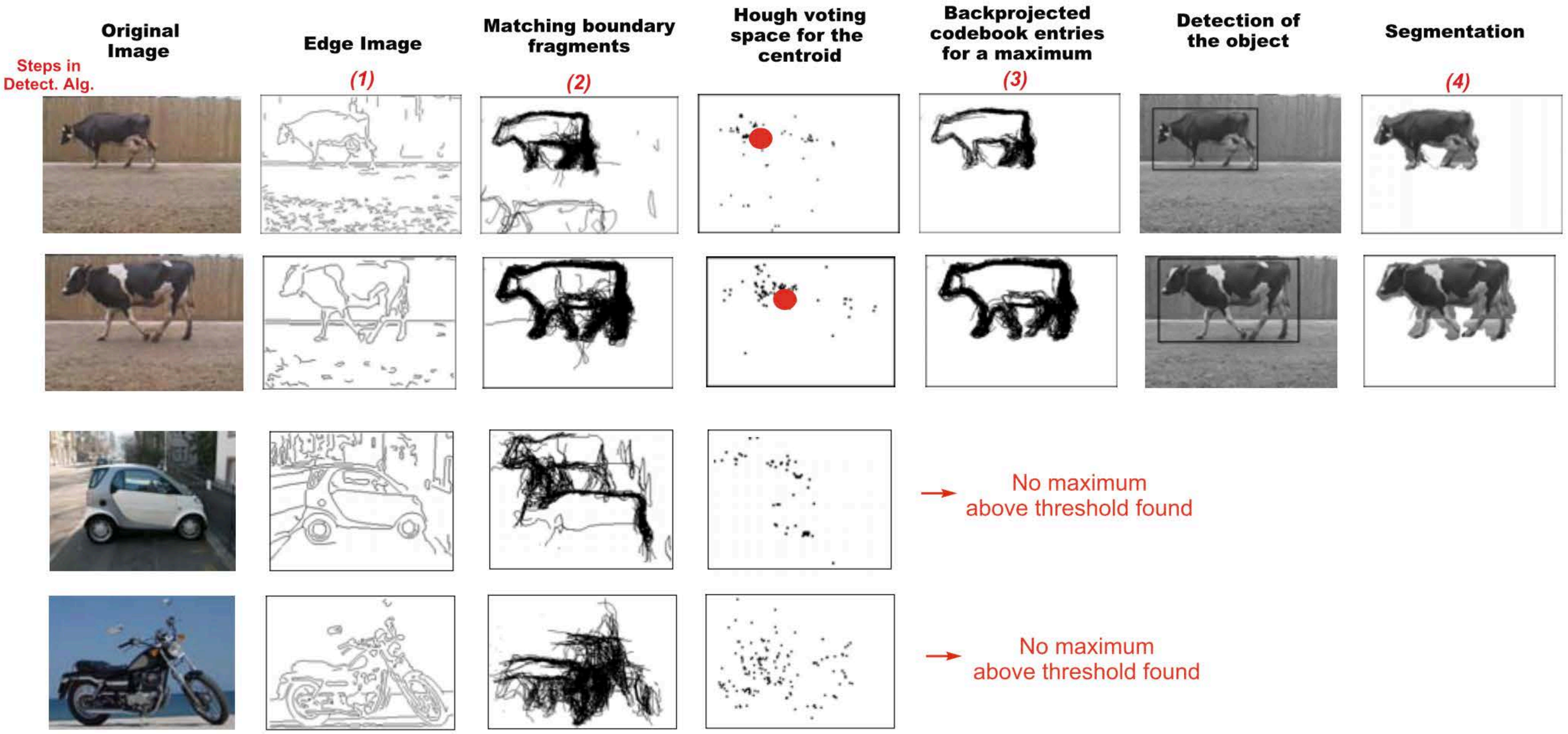


Image credit: Opelt et al., 2006

Example 3: Deep Hough Voting

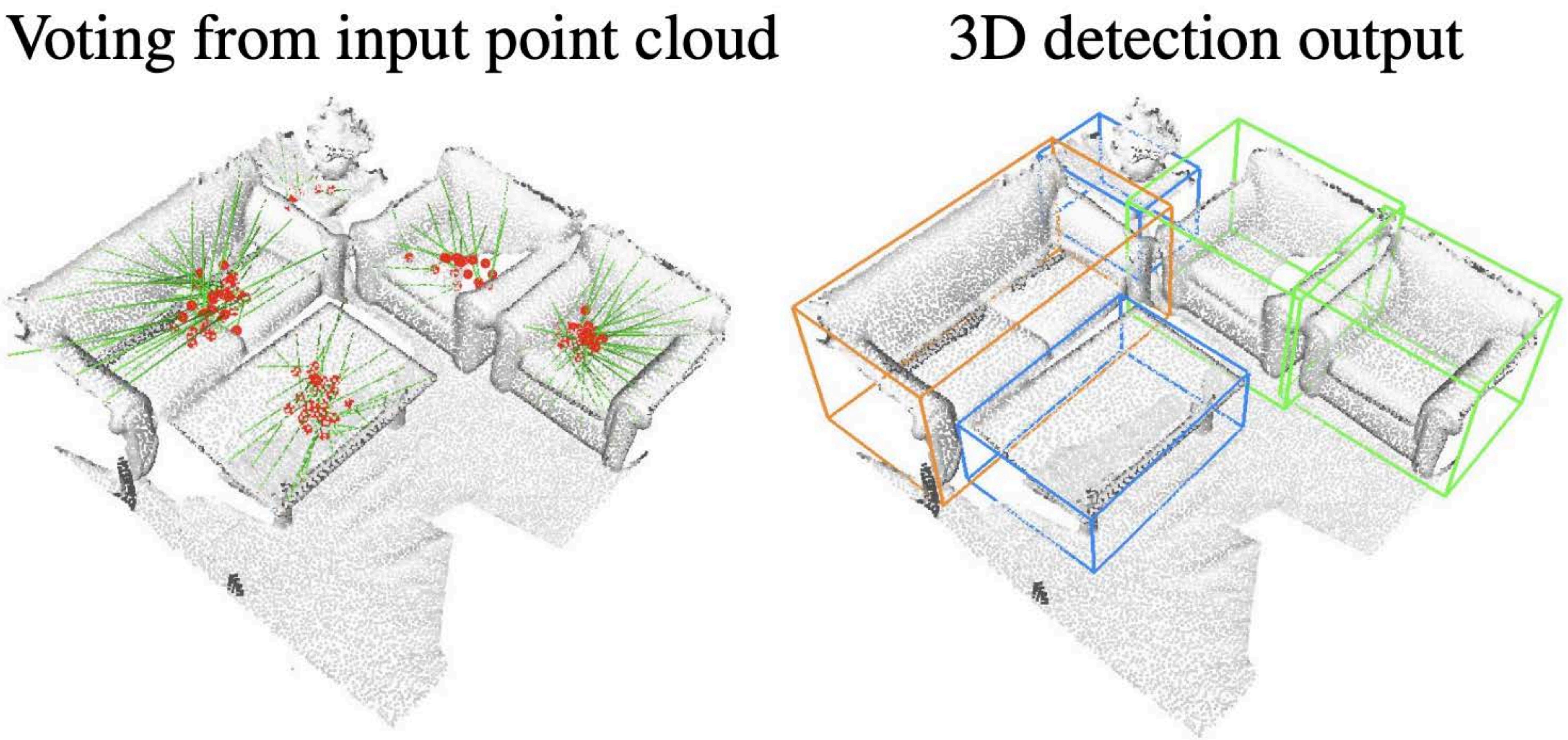


Figure 1. **3D object detection in point clouds with a deep Hough voting model.** Given a point cloud of a 3D scene, our VoteNet votes to object centers and then groups and aggregates the votes to predict 3D bounding boxes and semantic classes of objects.

[Qi et al., 2019, ICCV]

Summary of 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

e.g., 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

Advantages:

— Can handle high percentage of outliers: each point votes separately

— Can detect multiple instances of a model in a single pass

Disadvantages:

— Search time increases exponentially with the number of model parameters

— Can be tricky to pick a good bin size