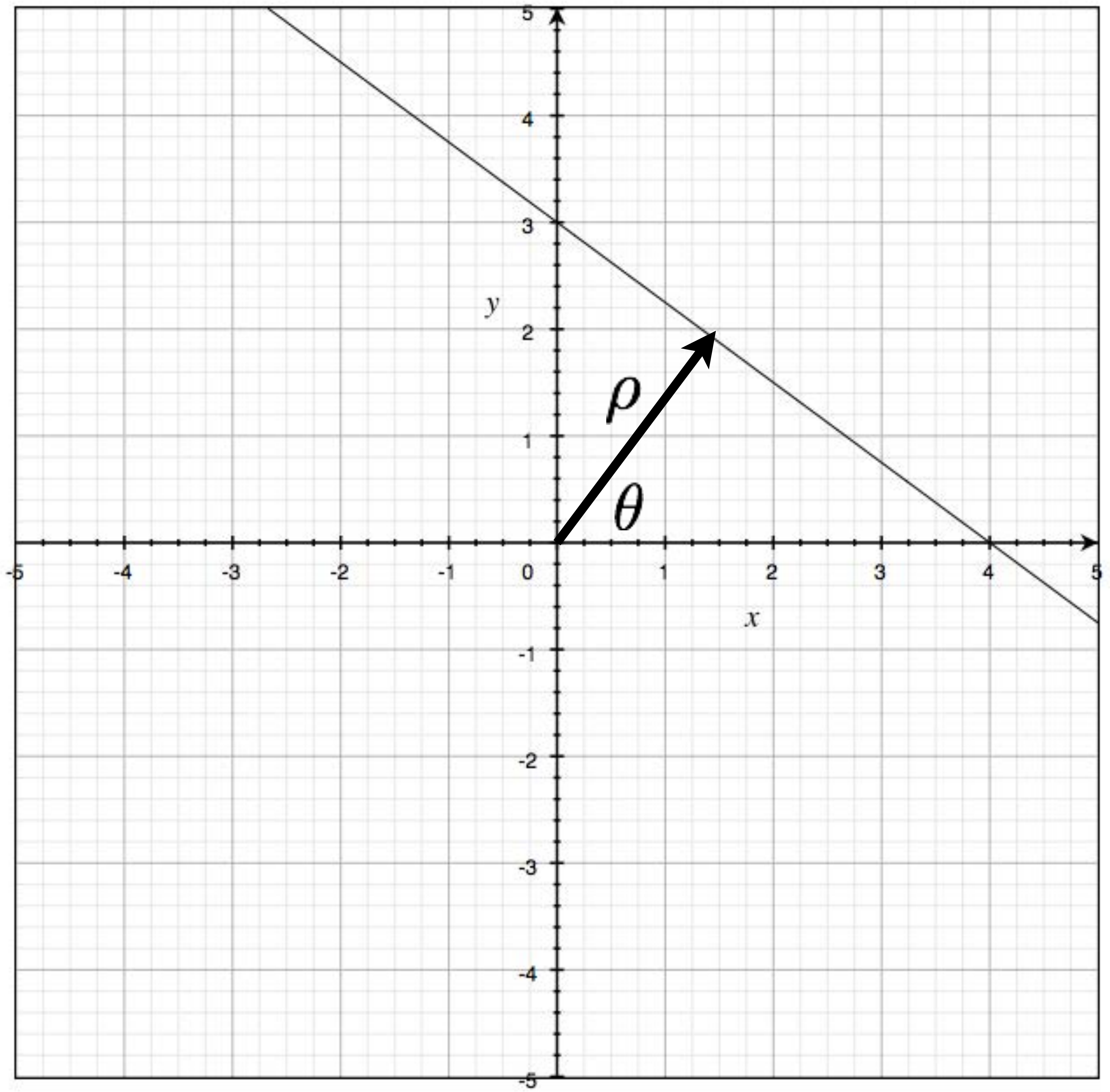
Lines: Normal form

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



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

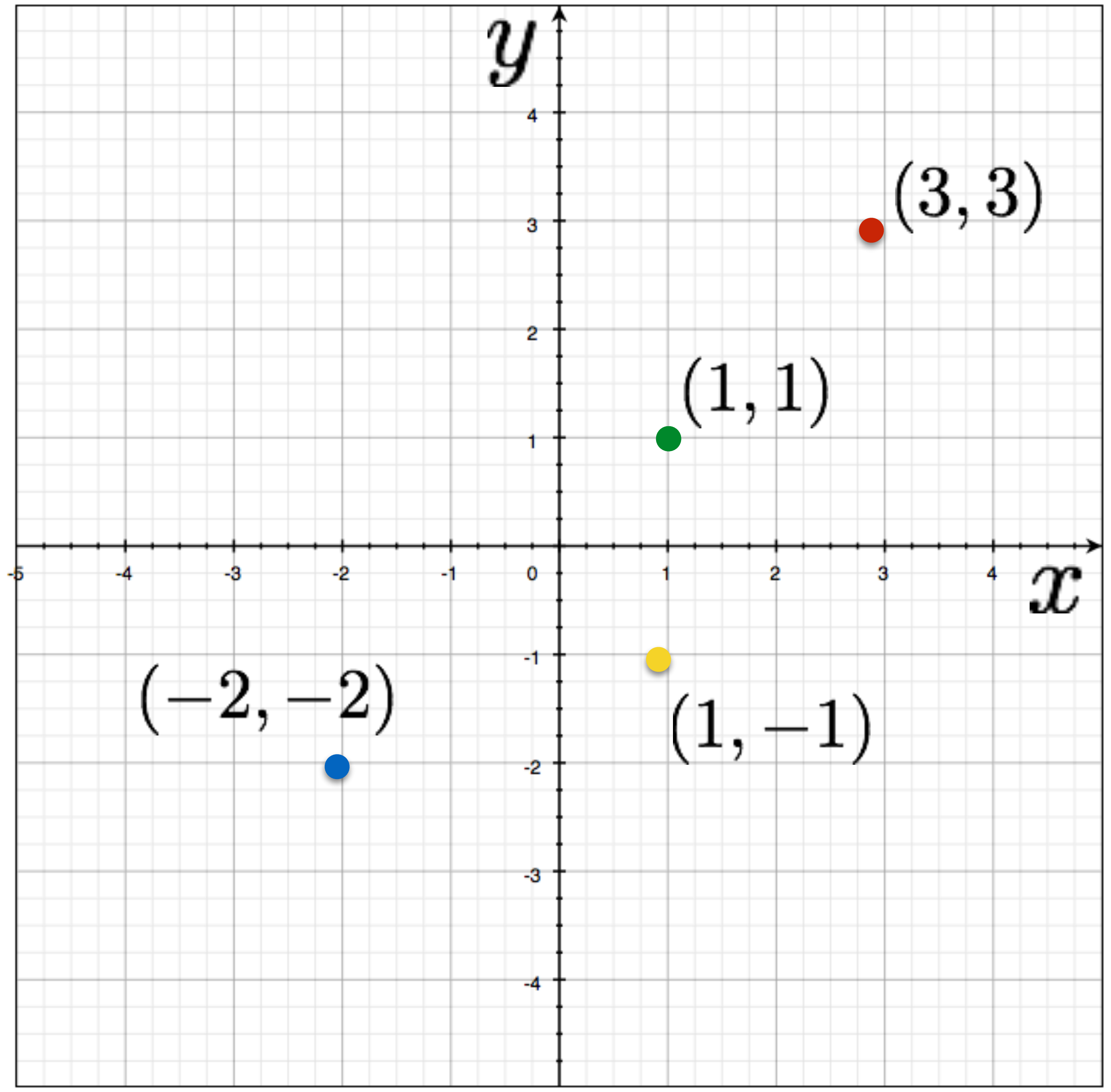
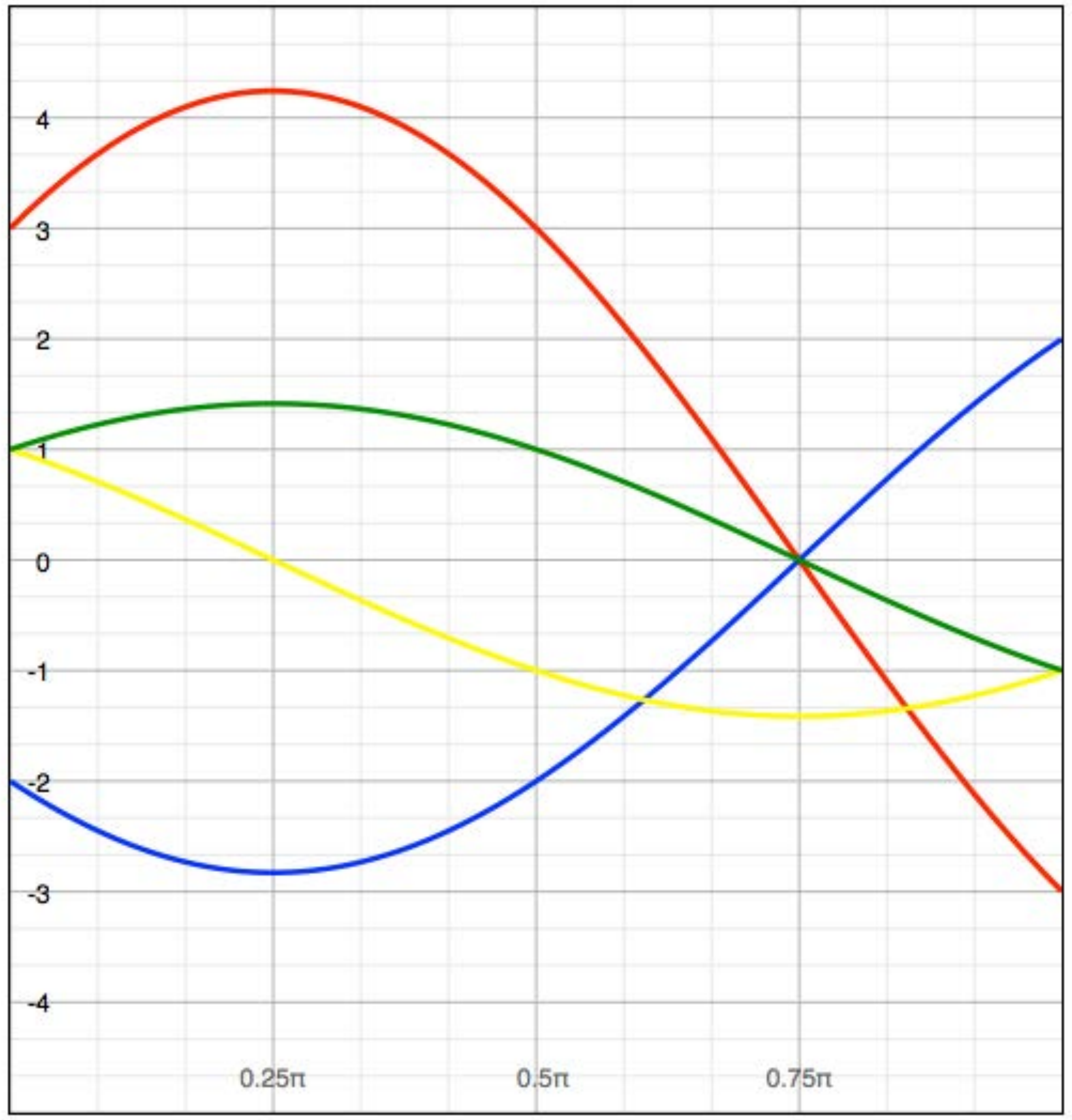


Image space

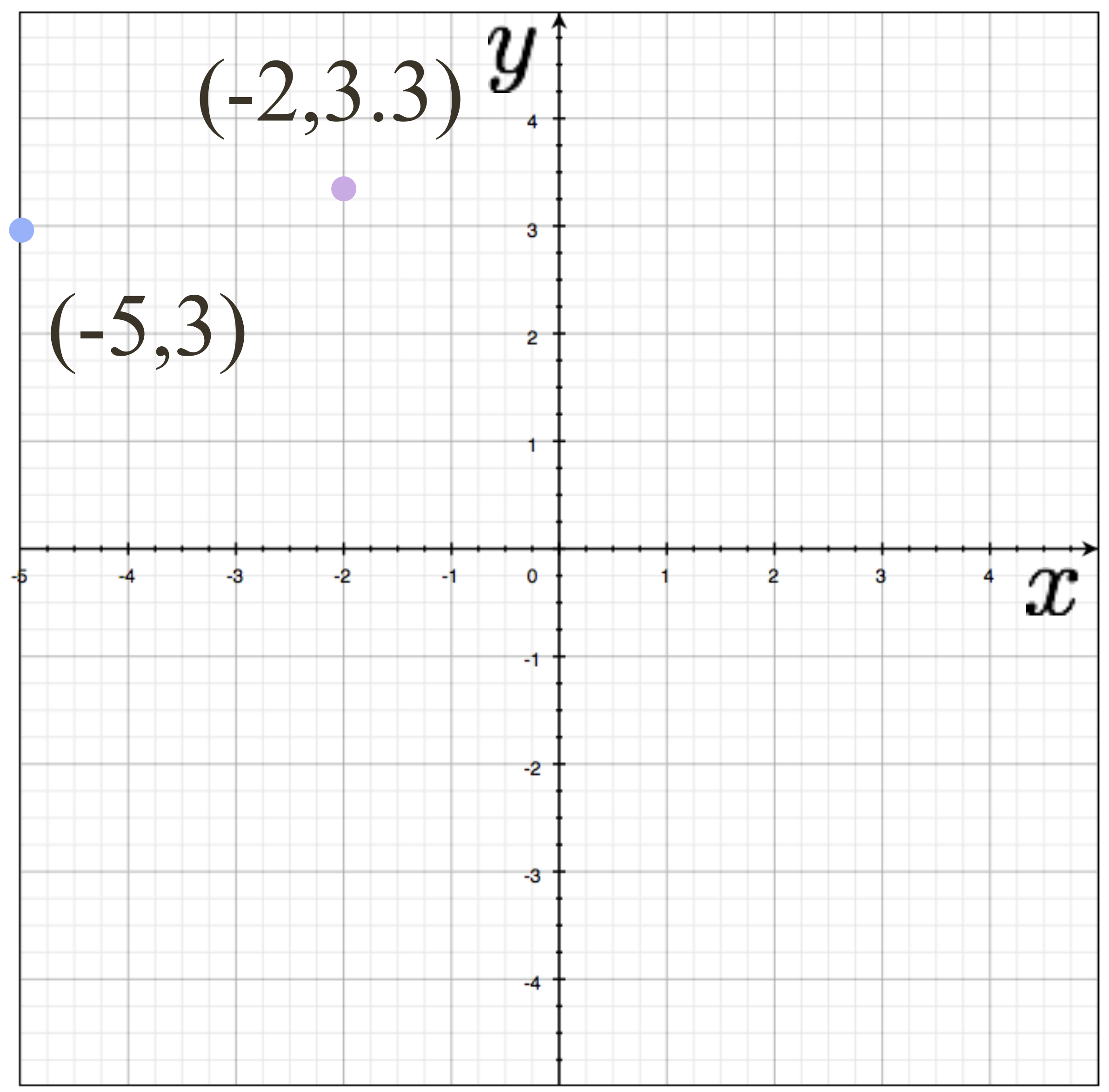
four points become?

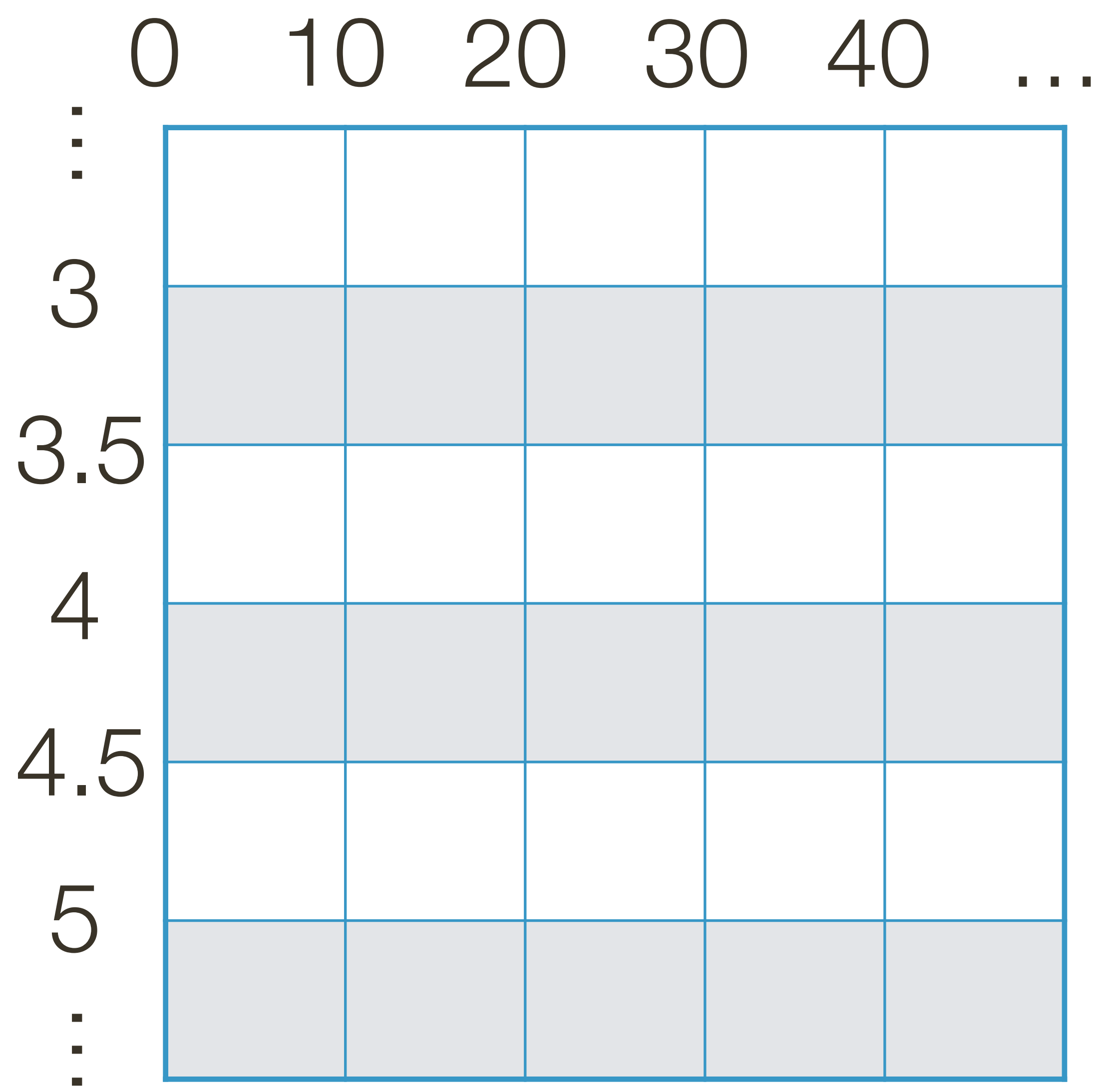


Parameter space

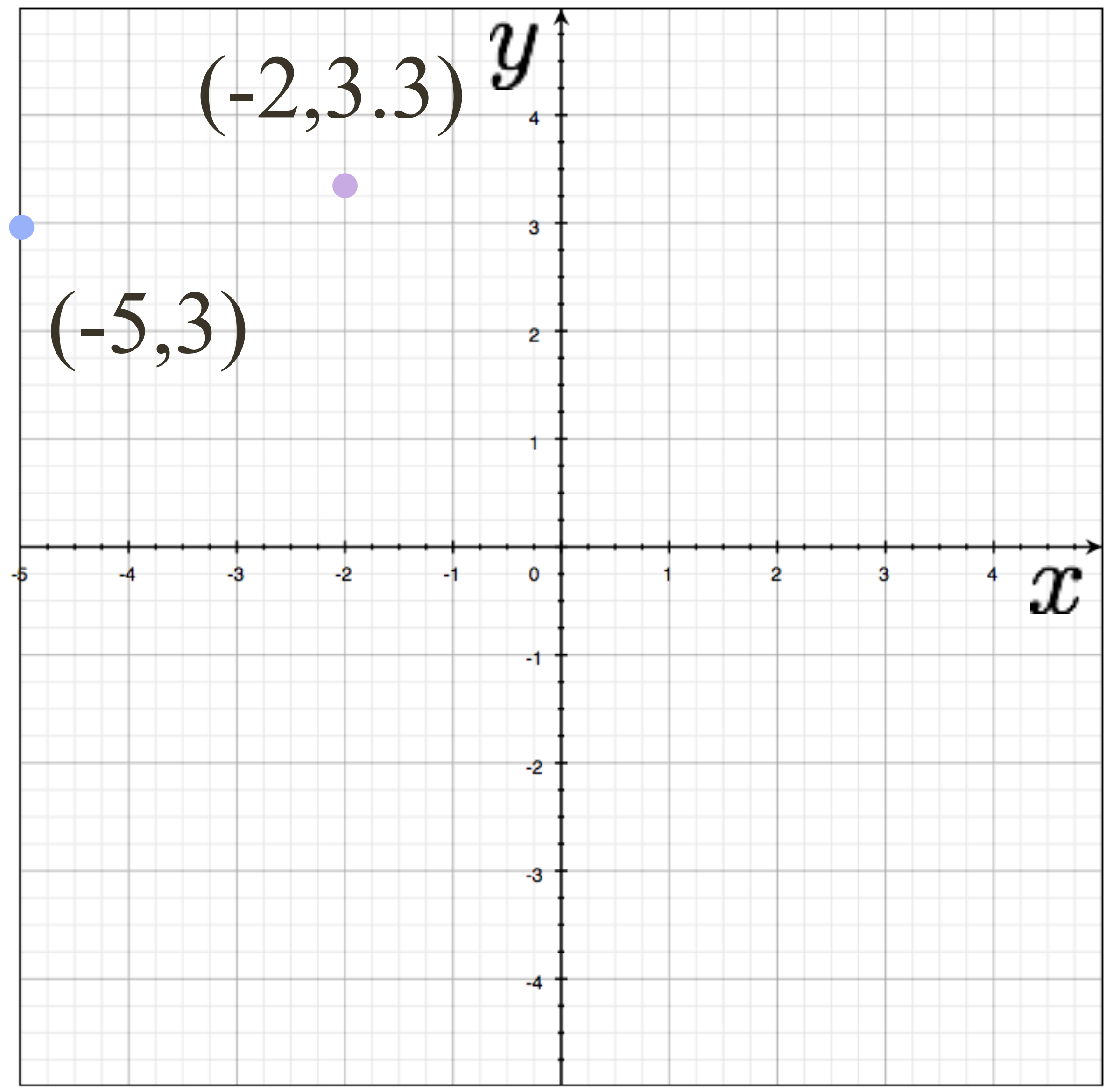
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

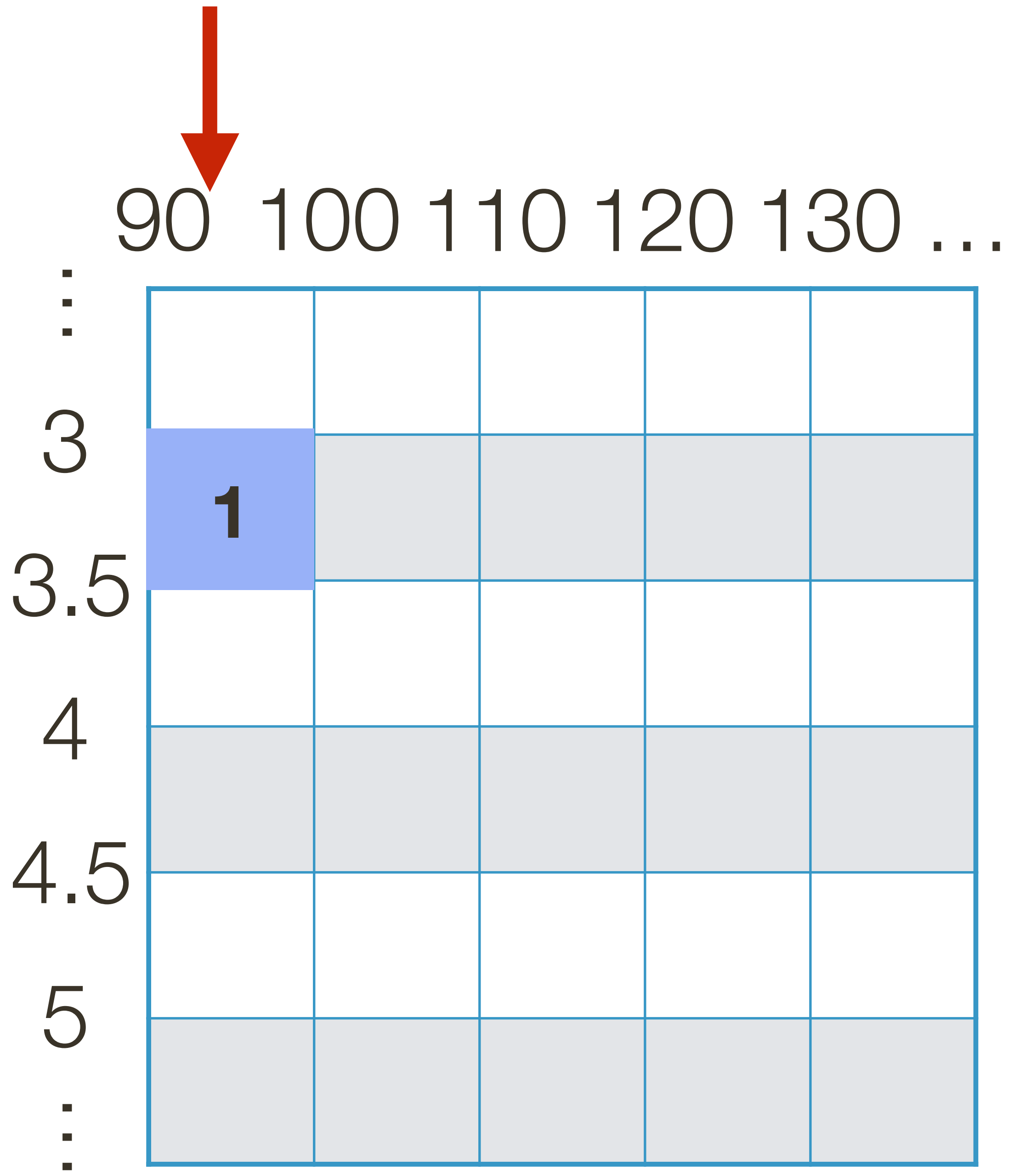
Example: Hough Transform for Lines





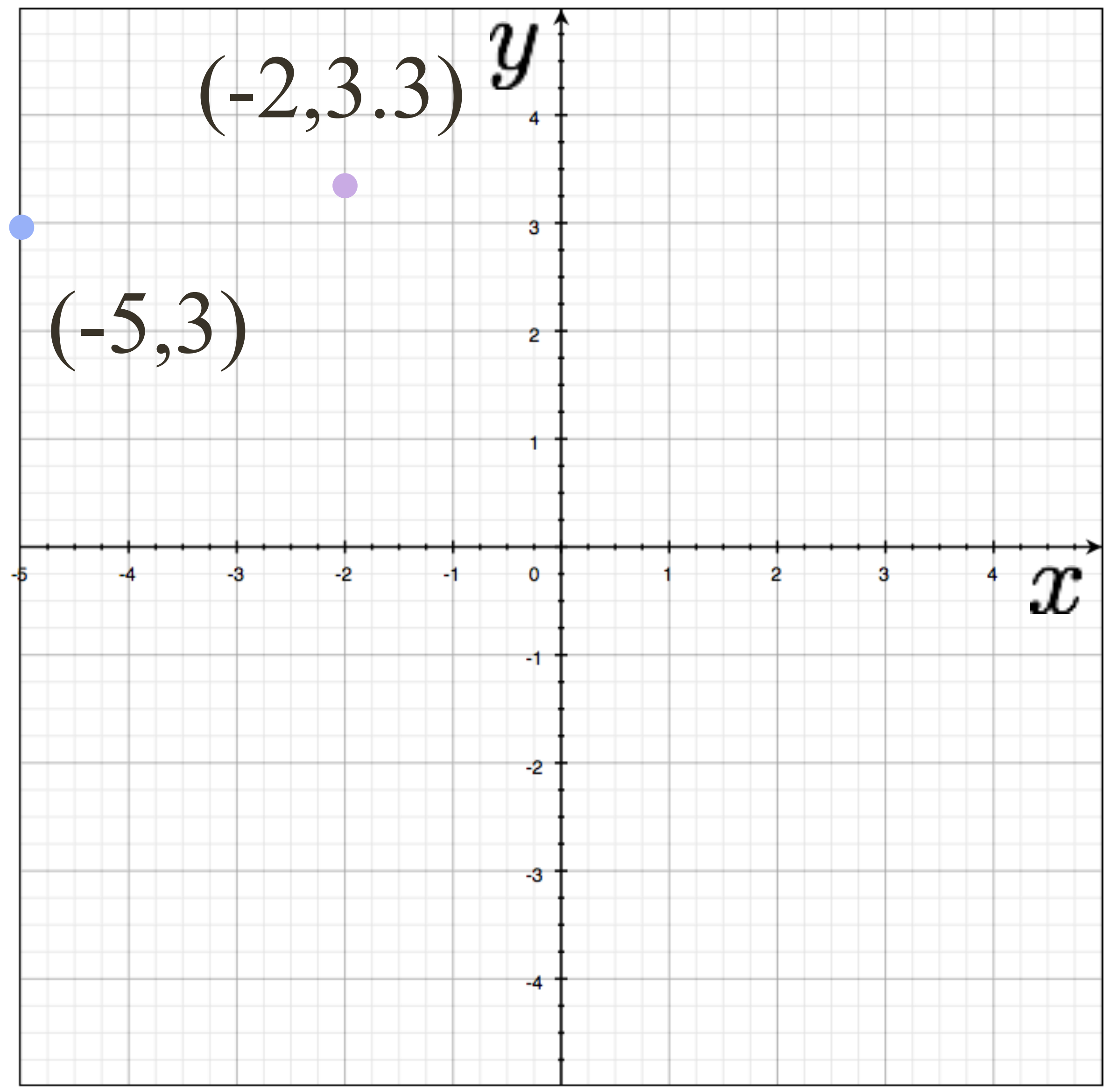
Example: Hough Transform for Lines

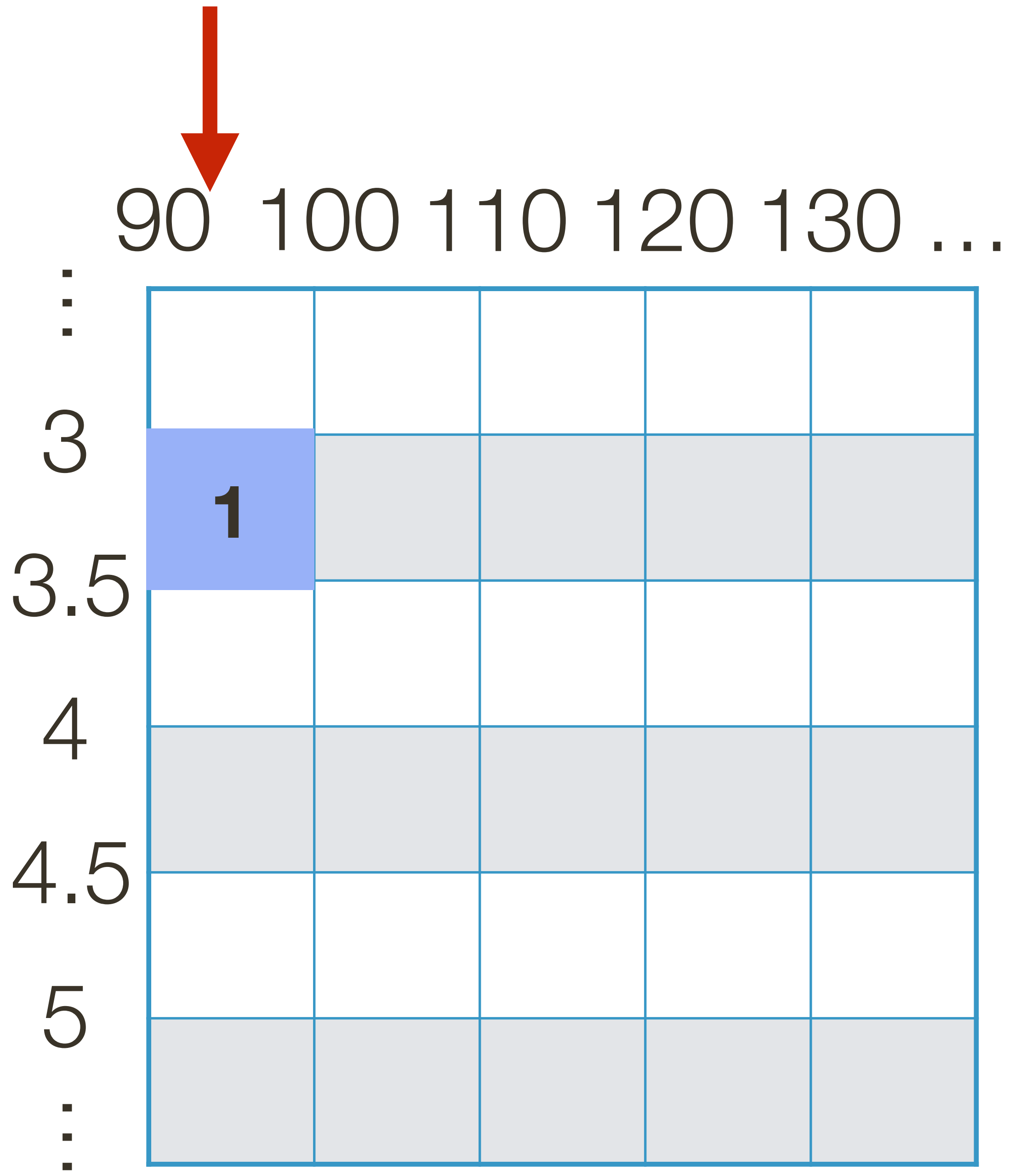




$$-5\cos(95°) + 3\sin(95°) + r = 0 \rightarrow r \approx 3.42$$

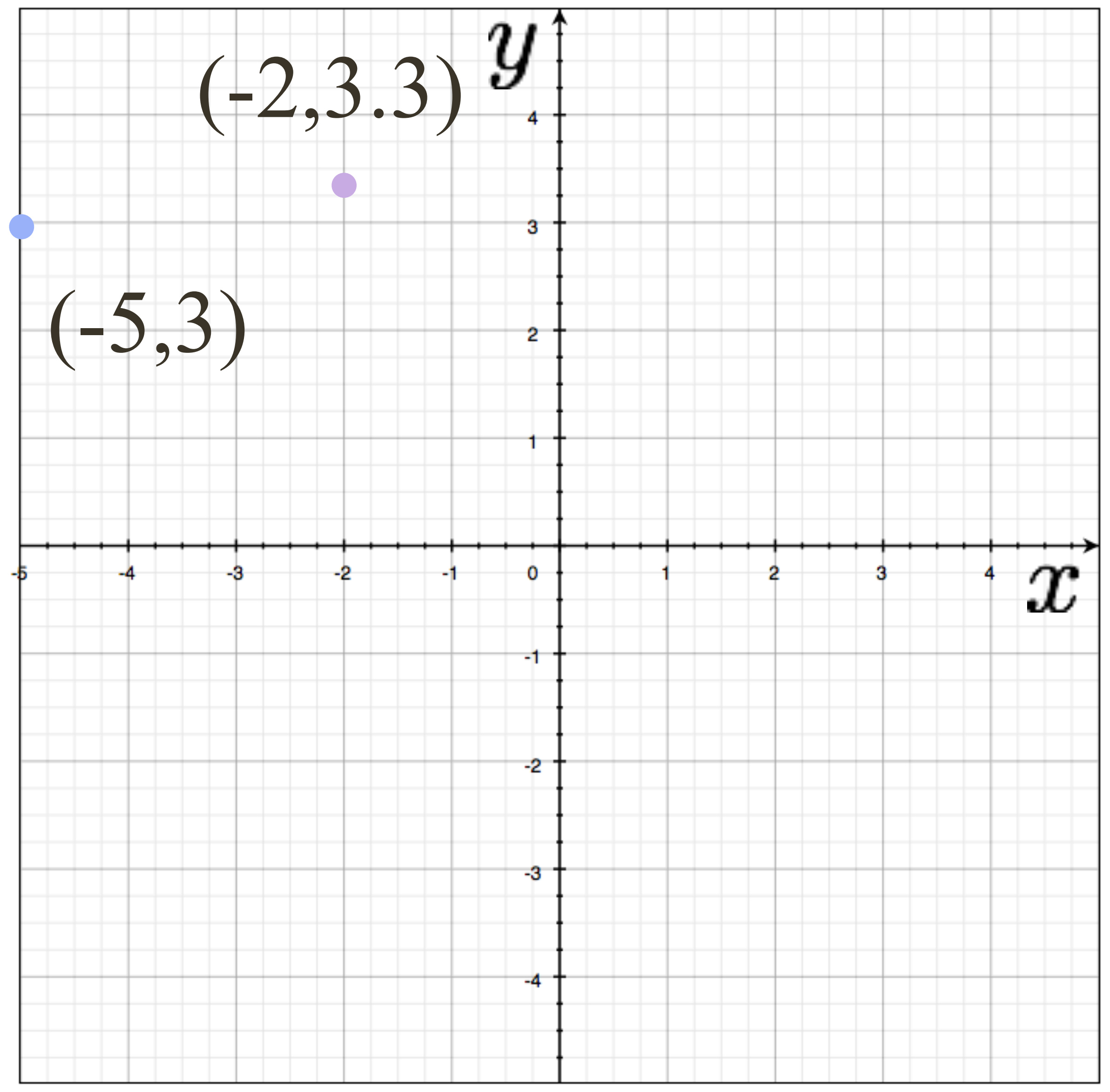
Example: Hough Transform for Lines

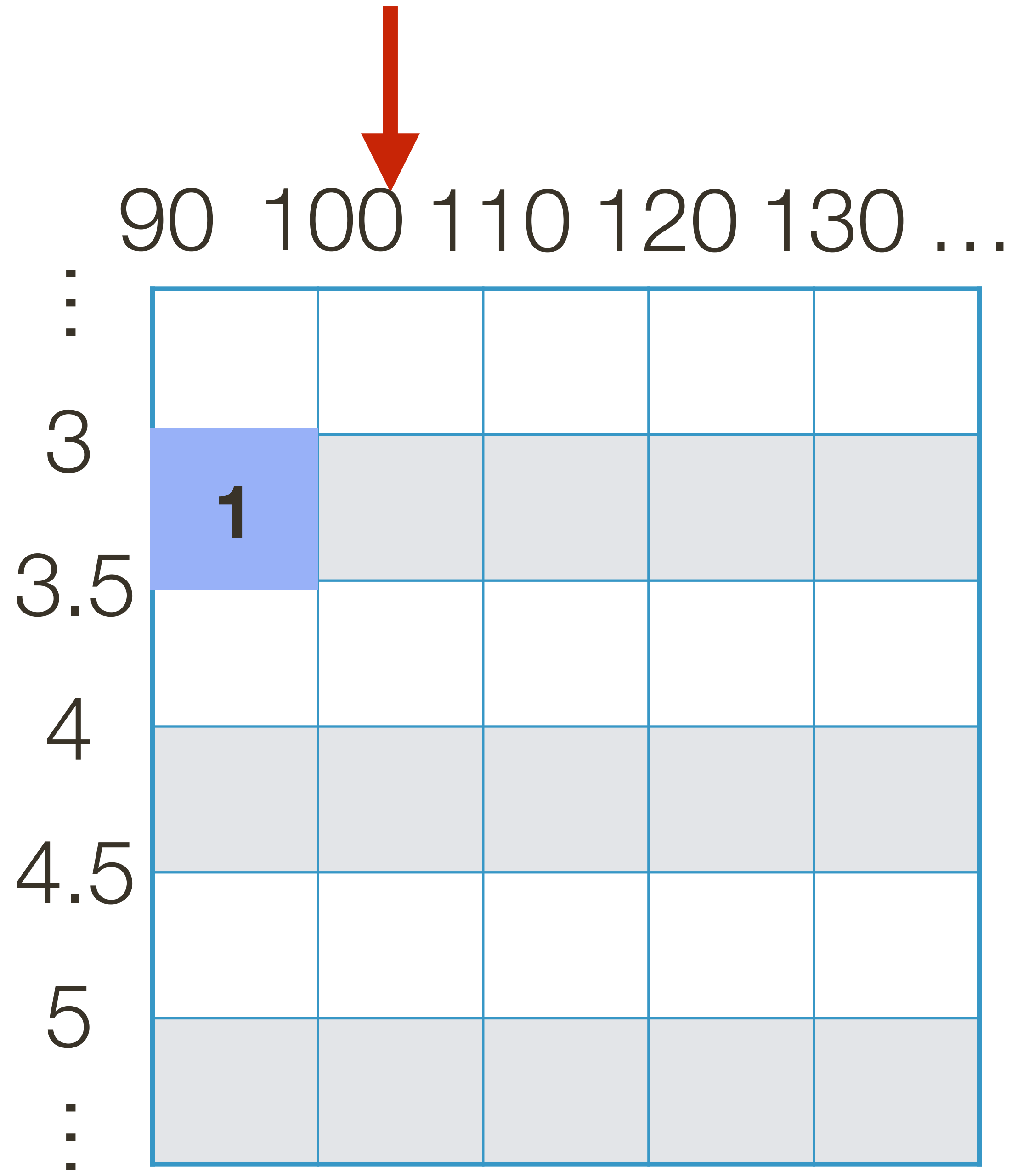




$$-5\cos(95^{\circ}) + 3\sin(95^{\circ}) + r = 0 \rightarrow r \approx 3.42$$

Example: Hough Transform for Lines

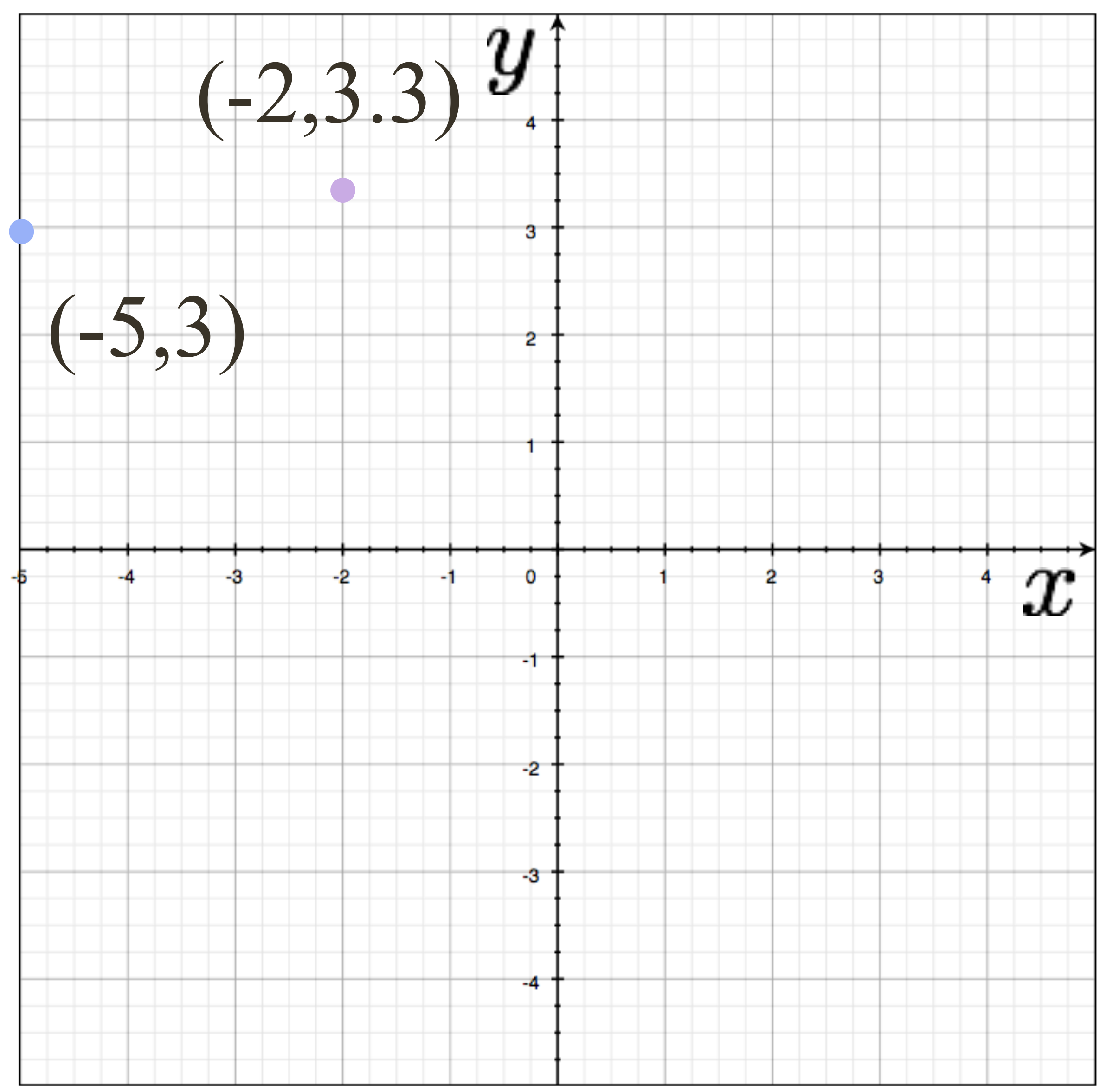


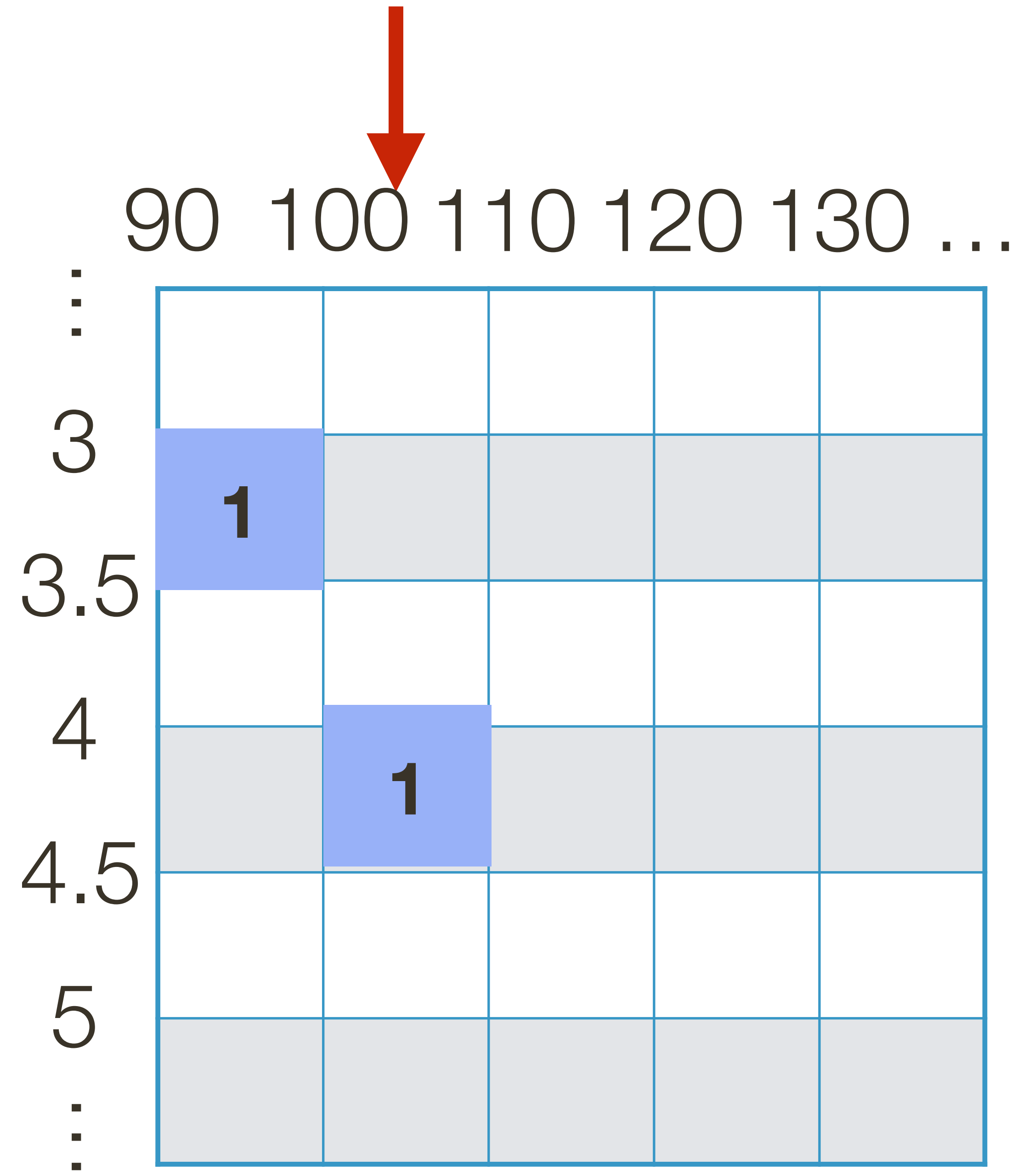


$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

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

Example: Hough Transform for Lines

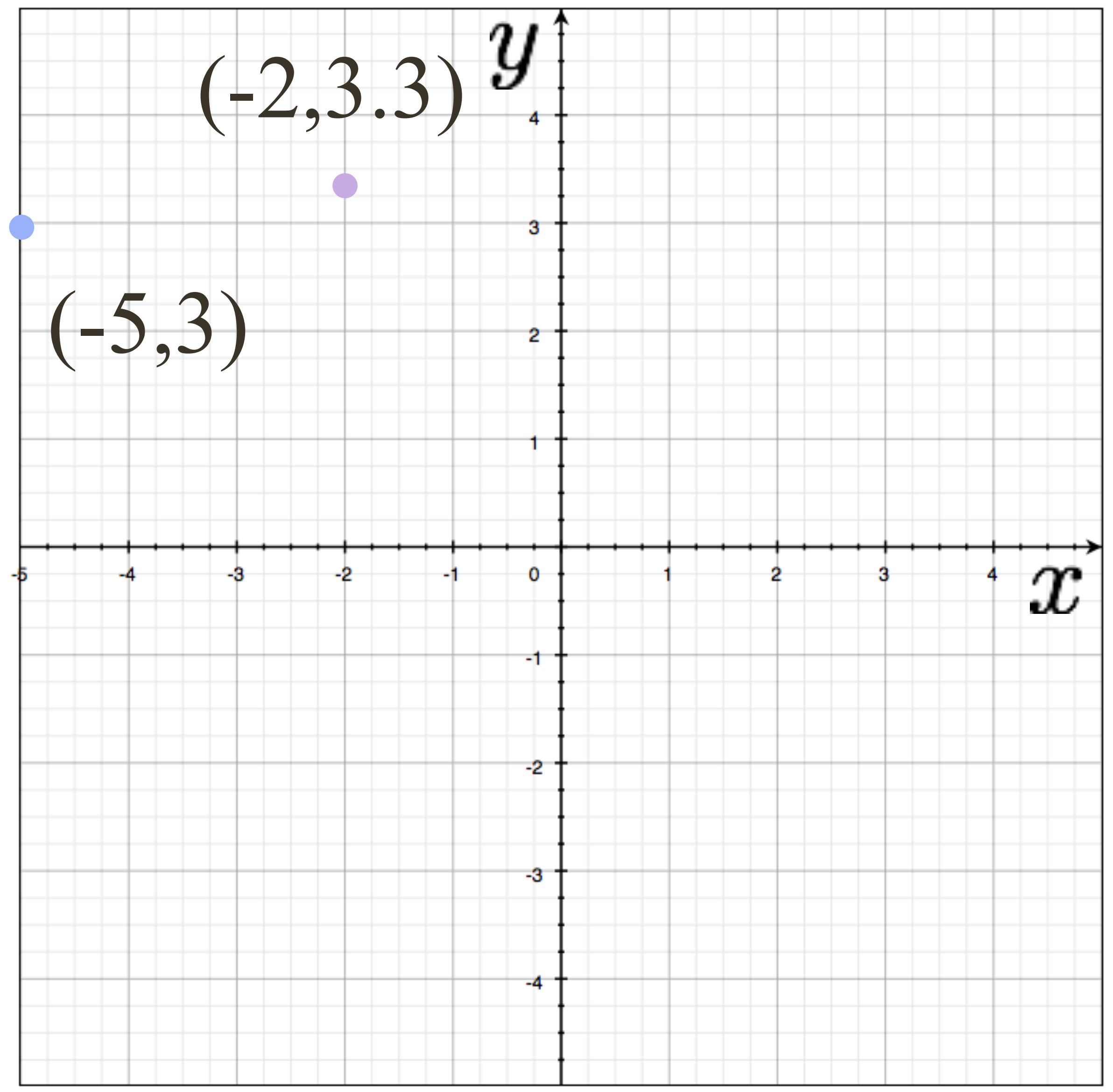


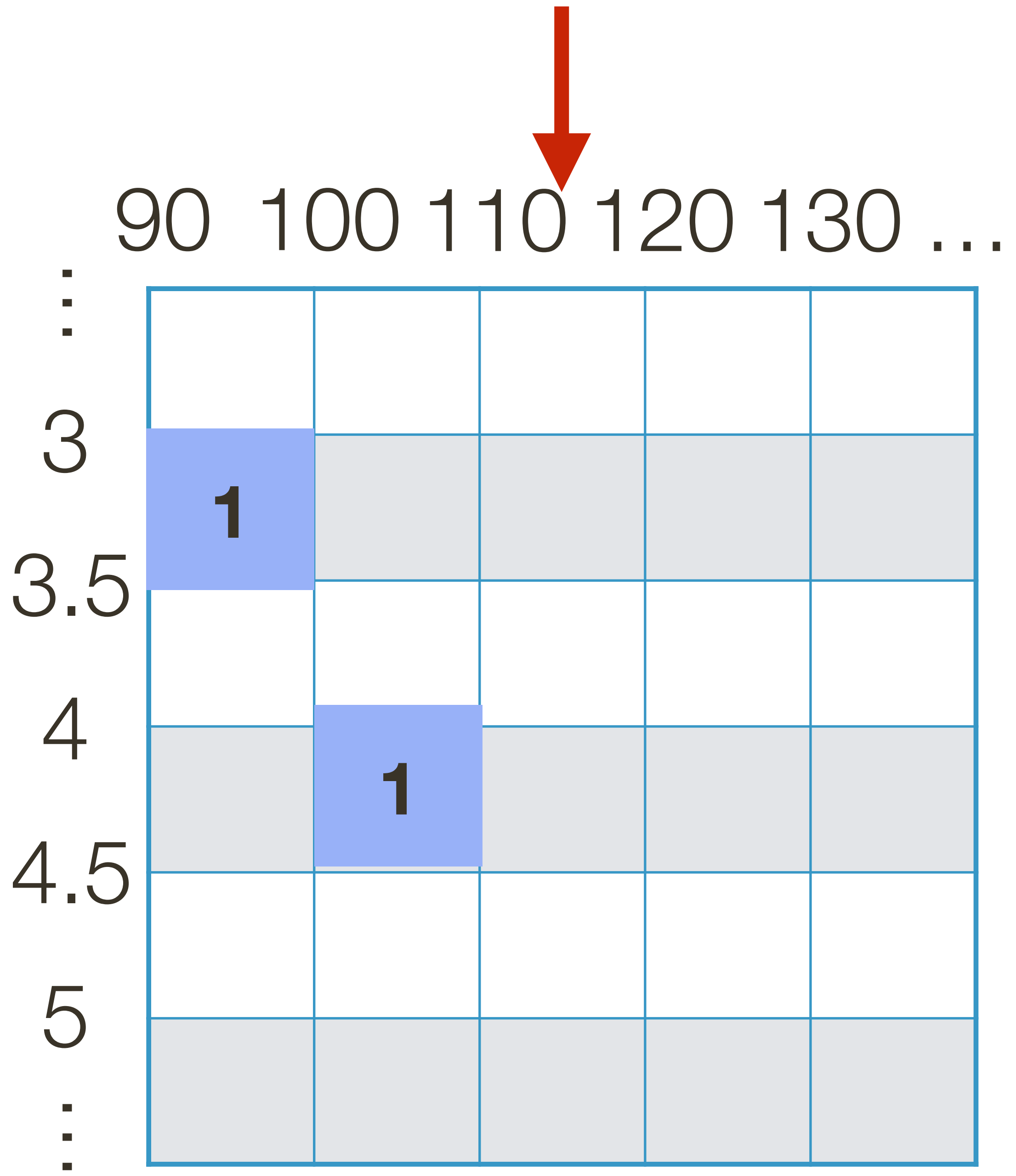


$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

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

Example: Hough Transform for Lines



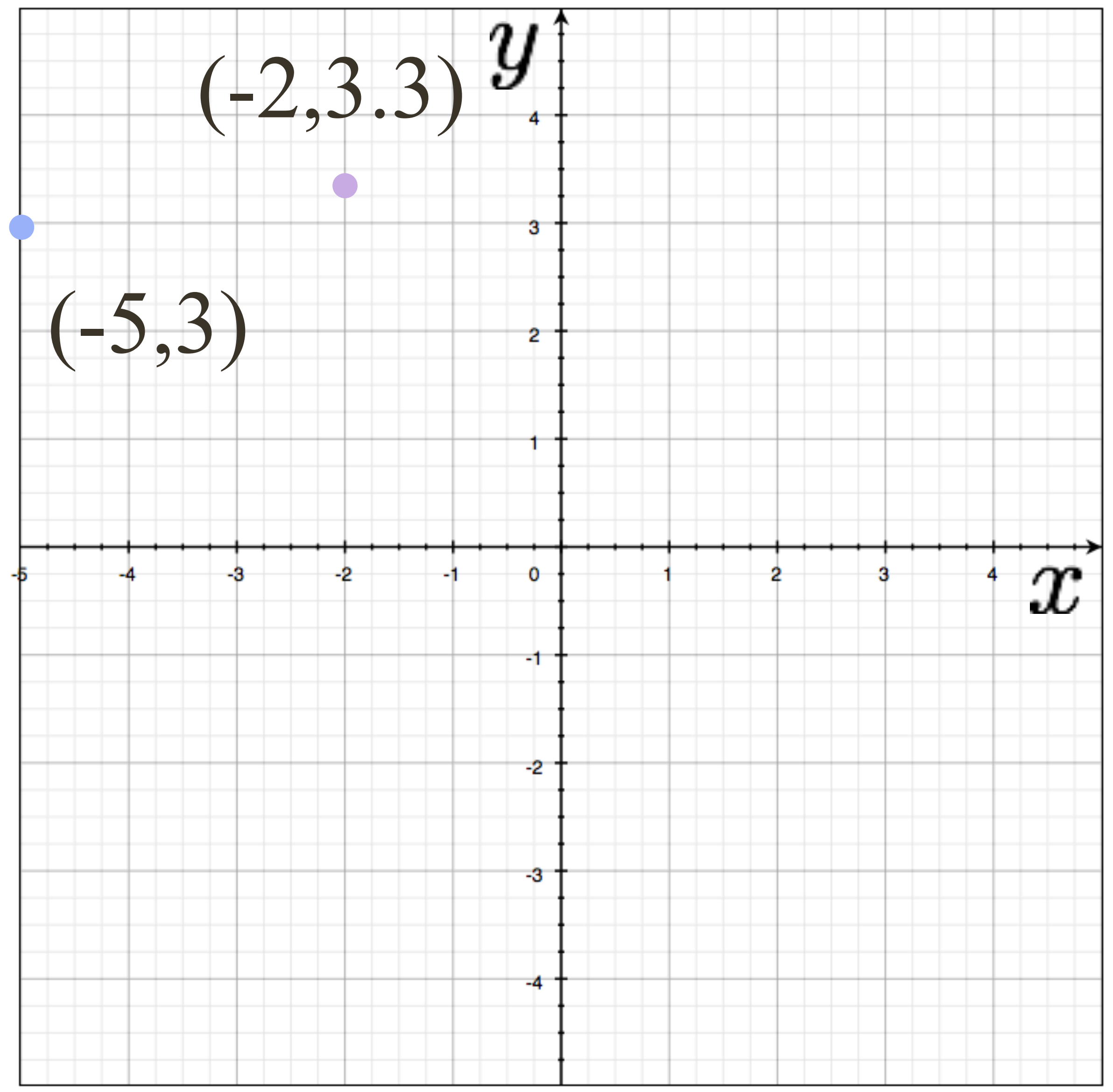


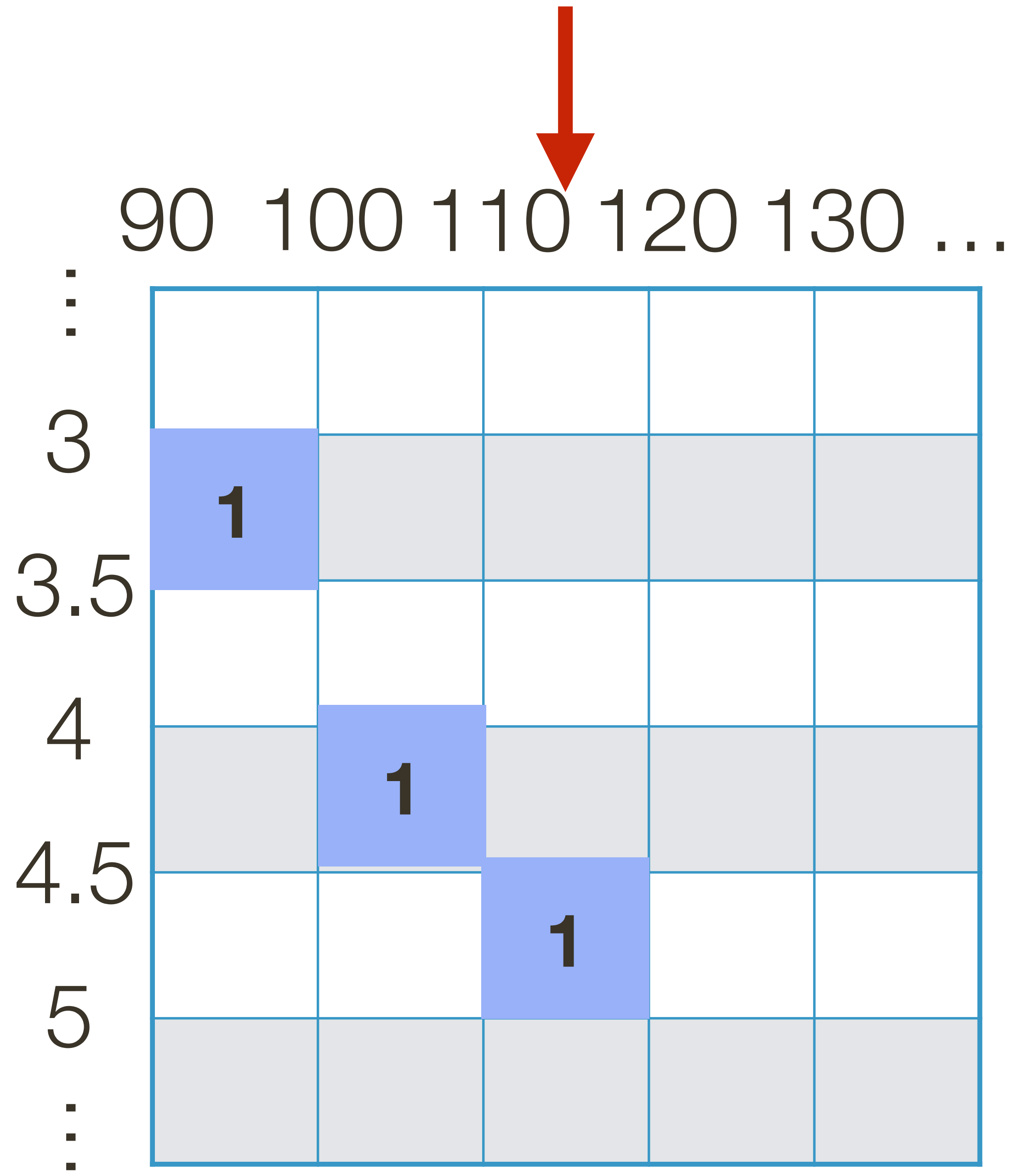
$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

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

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

Example: Hough Transform for Lines



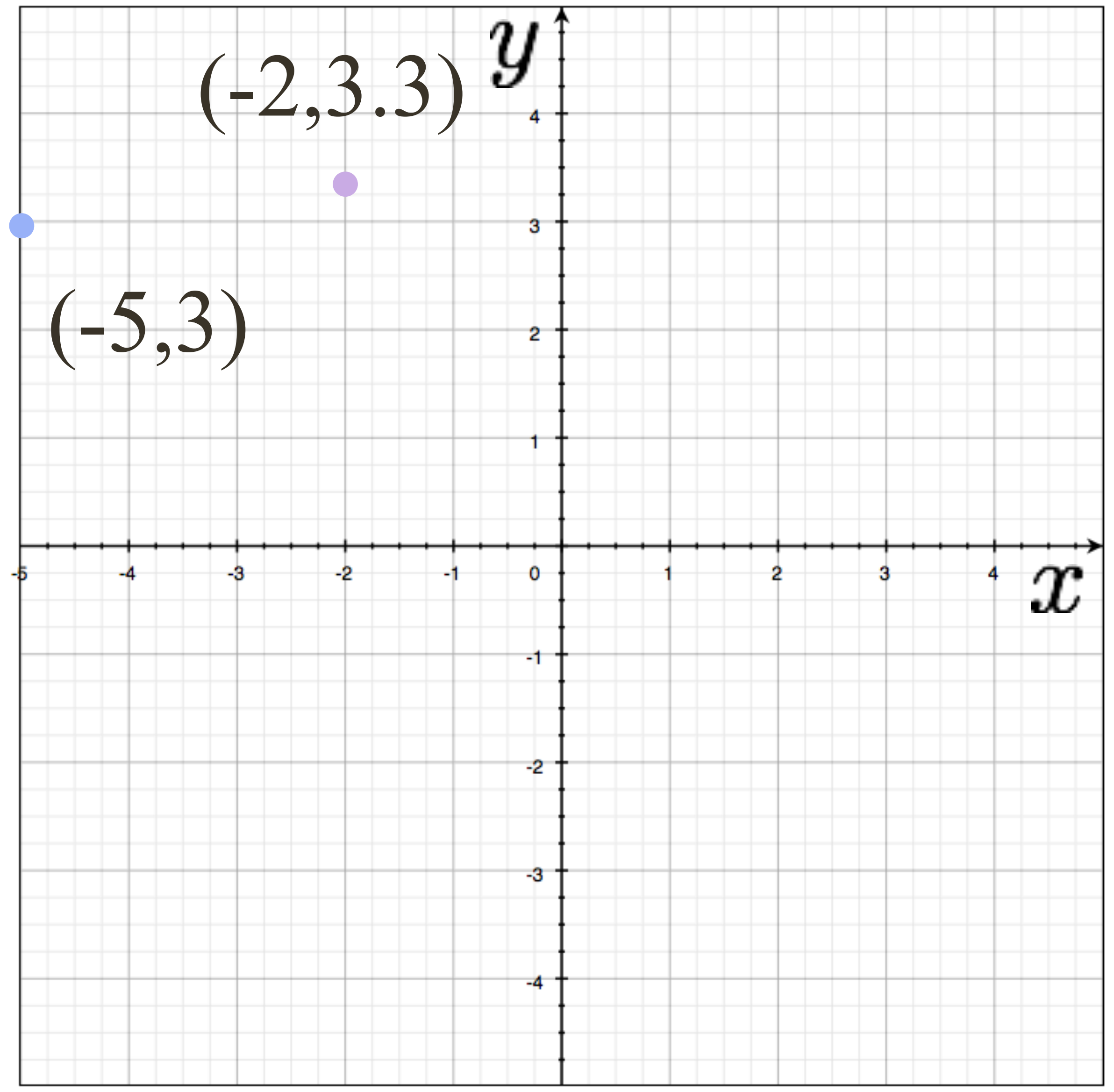


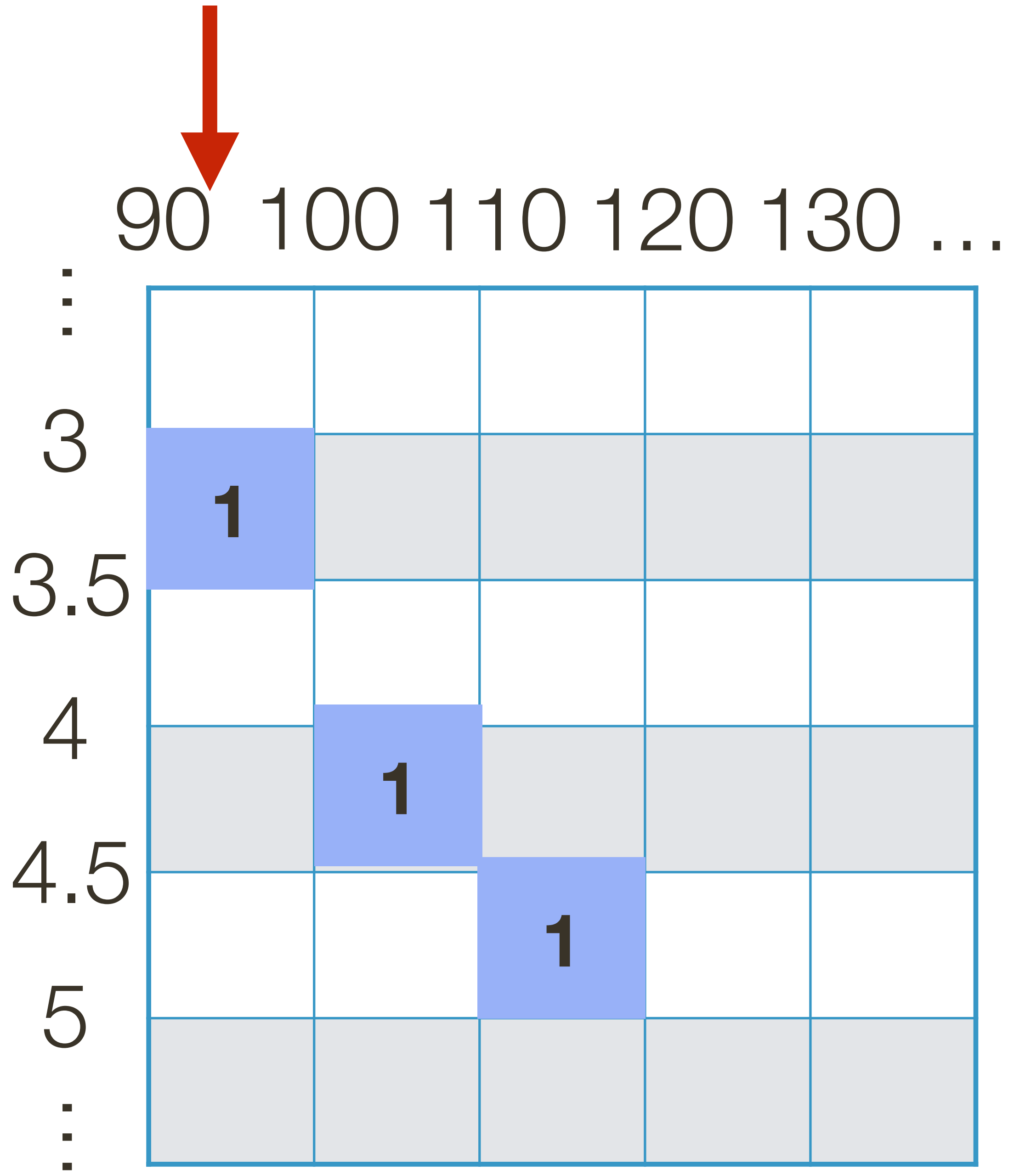
$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

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

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

Example: Hough Transform for Lines





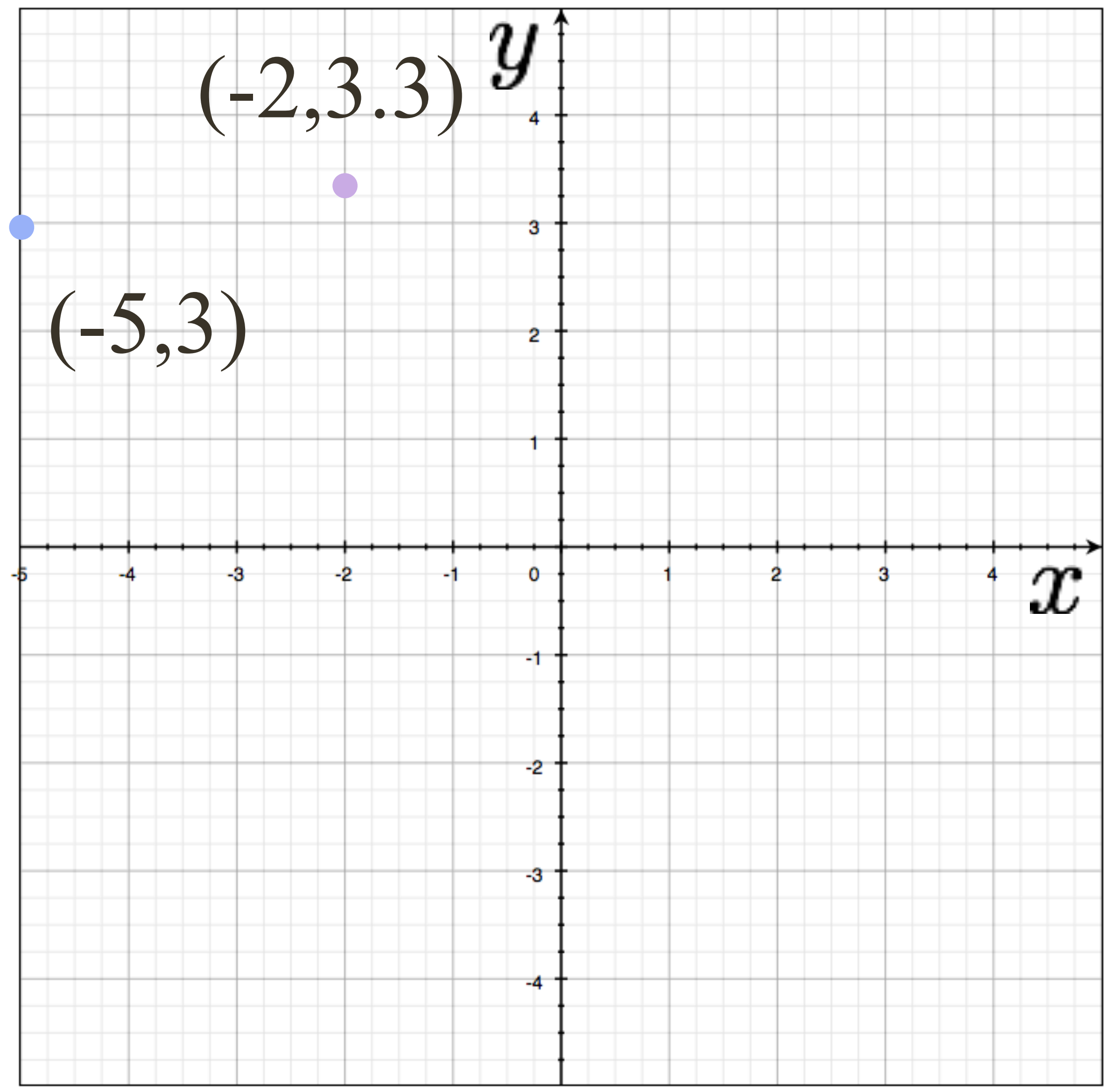
$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

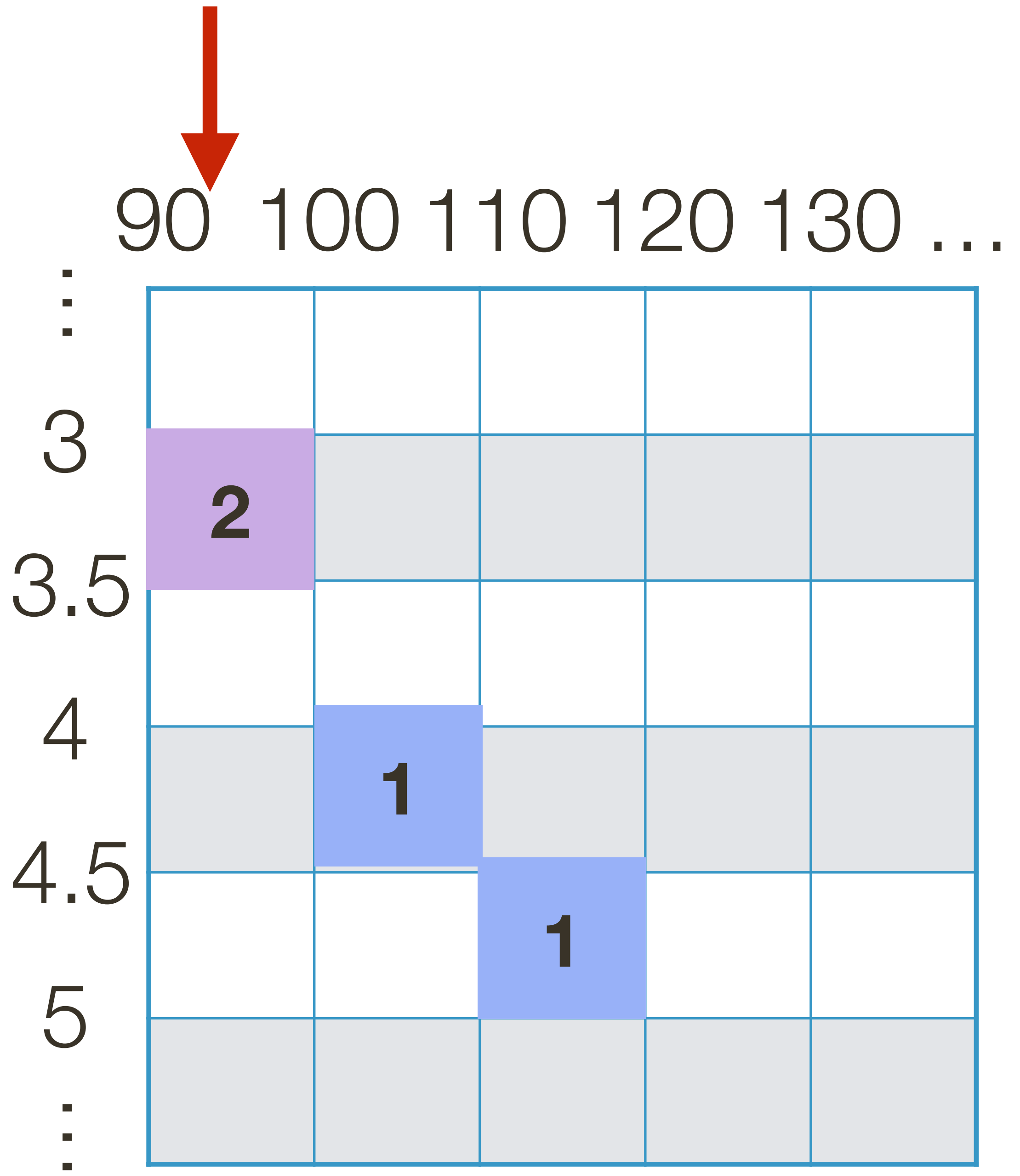
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

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

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

Example: Hough Transform for Lines





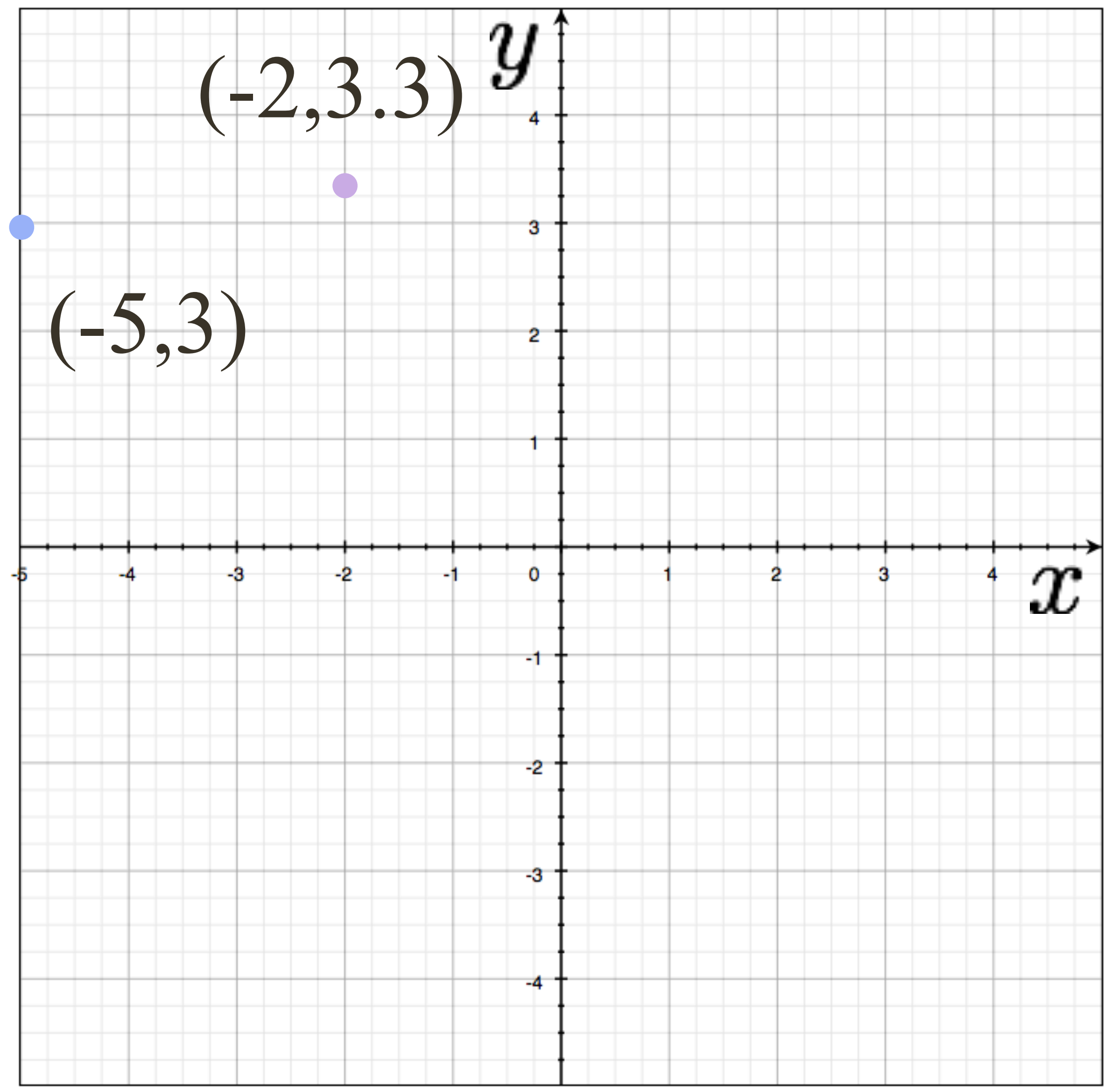
$$-5\cos(95^\circ) + 3\sin(95^\circ) + r = 0 \rightarrow r \approx 3.42$$

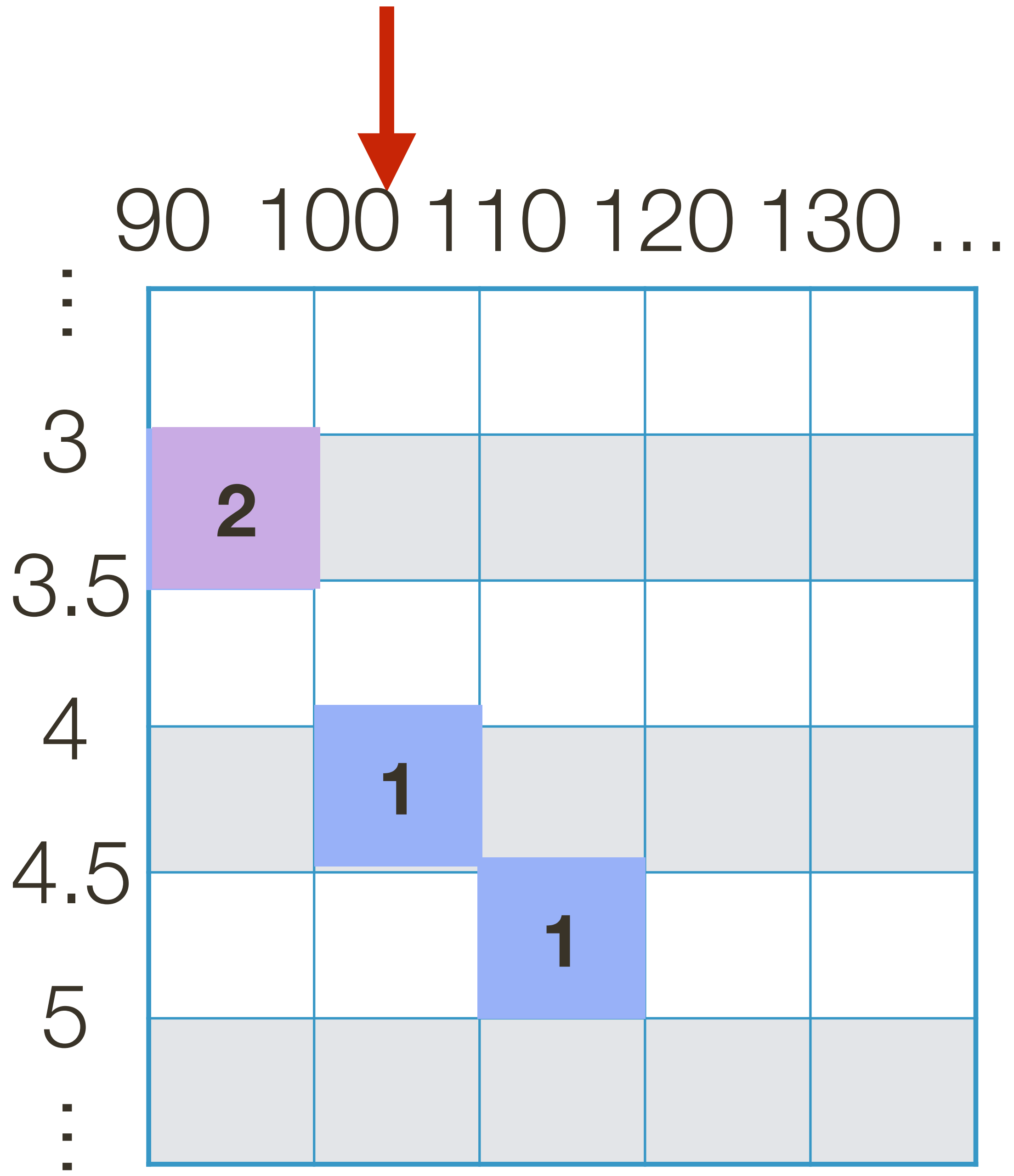
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

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

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

Example: Hough Transform for Lines





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

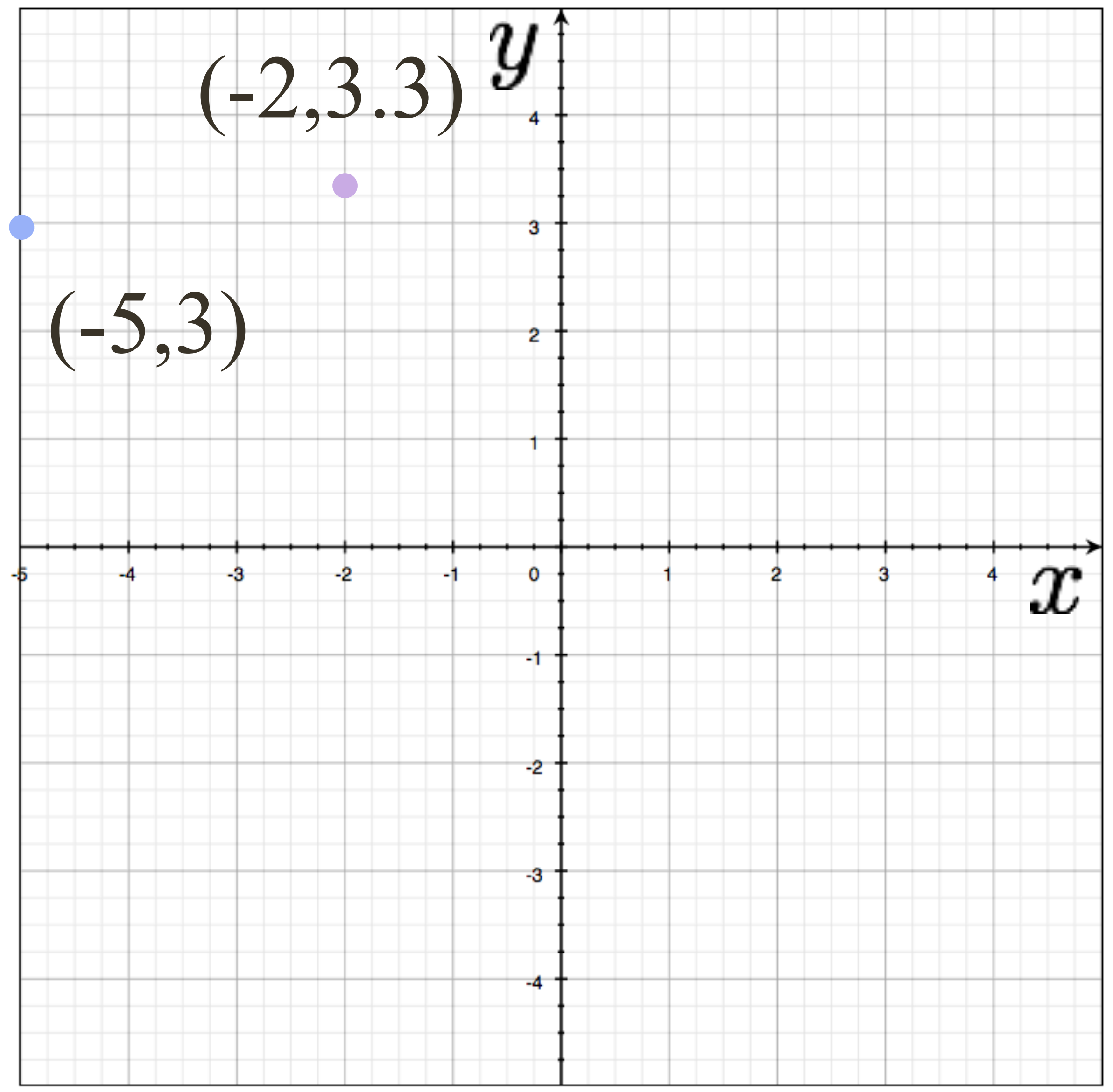
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

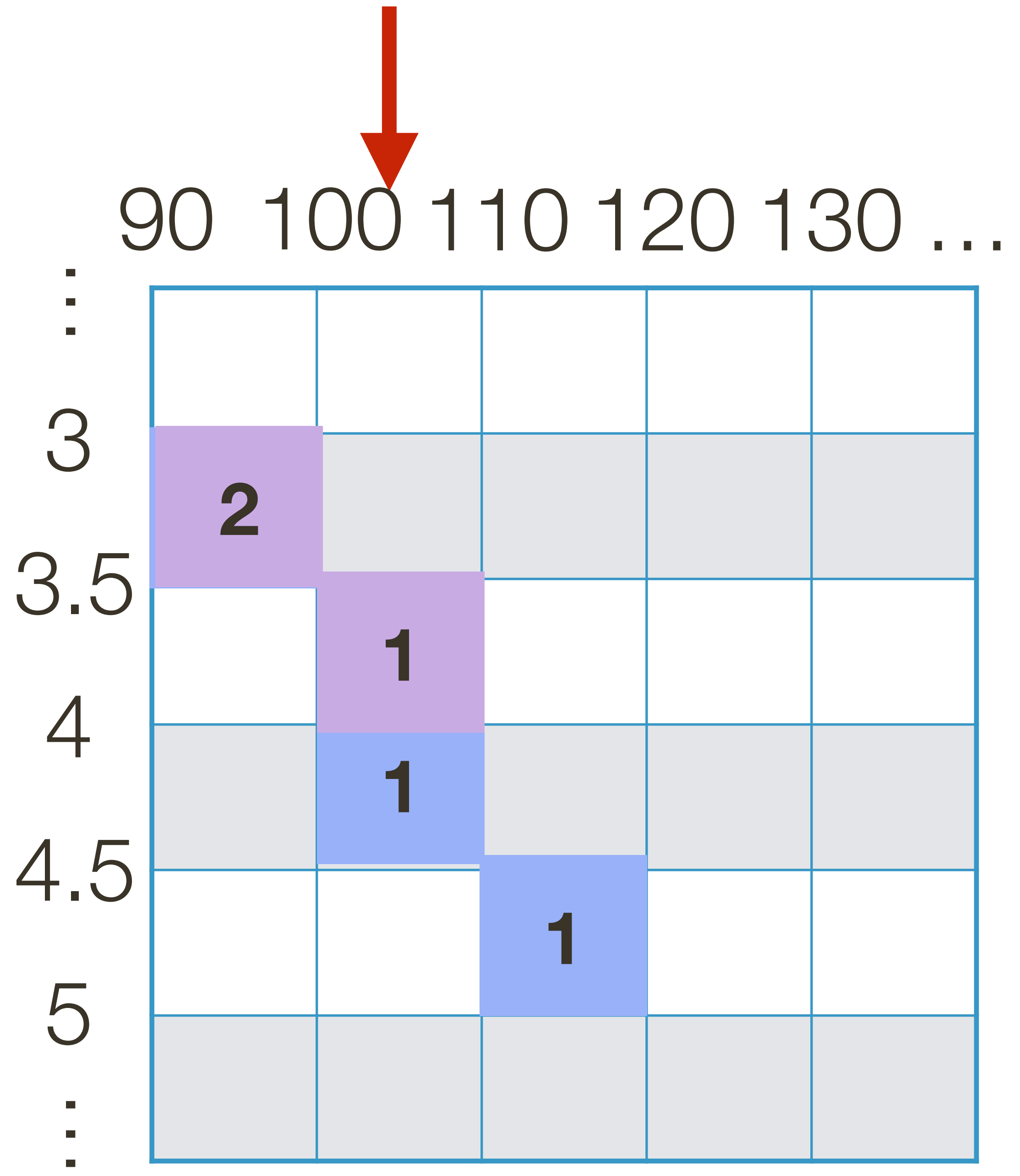
$$-5\cos(115^\circ) + 3\sin(115^\circ) + r = 0 \rightarrow r \approx 4.83$$

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

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

Example: Hough Transform for Lines





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

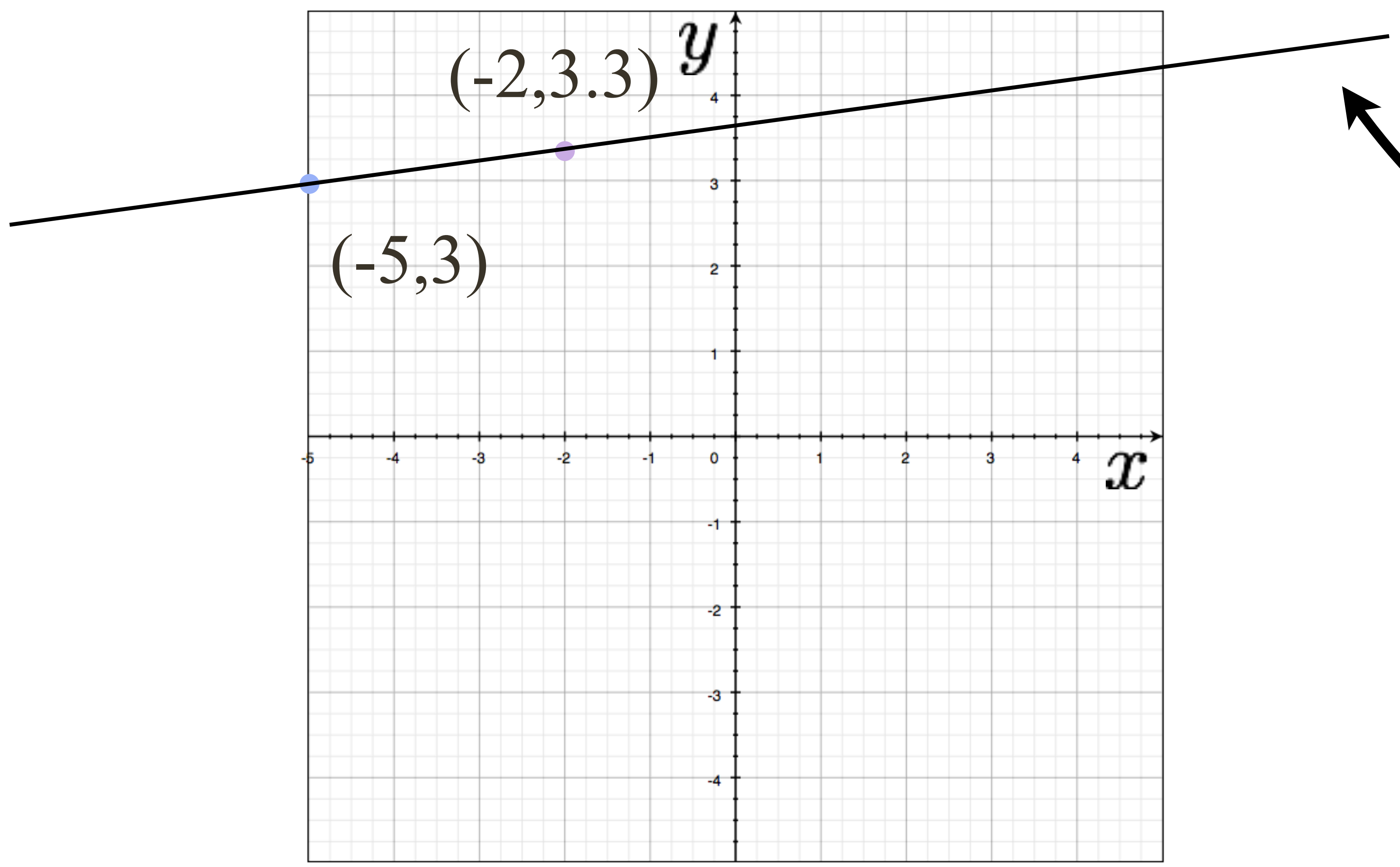
$$-5\cos(105^\circ) + 3\sin(105^\circ) + r = 0 \rightarrow r \approx 4.18$$

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

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

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

Example: Hough Transform for Lines



	90	100	110	120	130 …
⋮					
3	2				
3.5		1			
4		1			
4.5			1		
5 ⋮					

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

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

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

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

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

Example 1: Object Recognition — Implicit Shape Model

"**Training**" images of cows

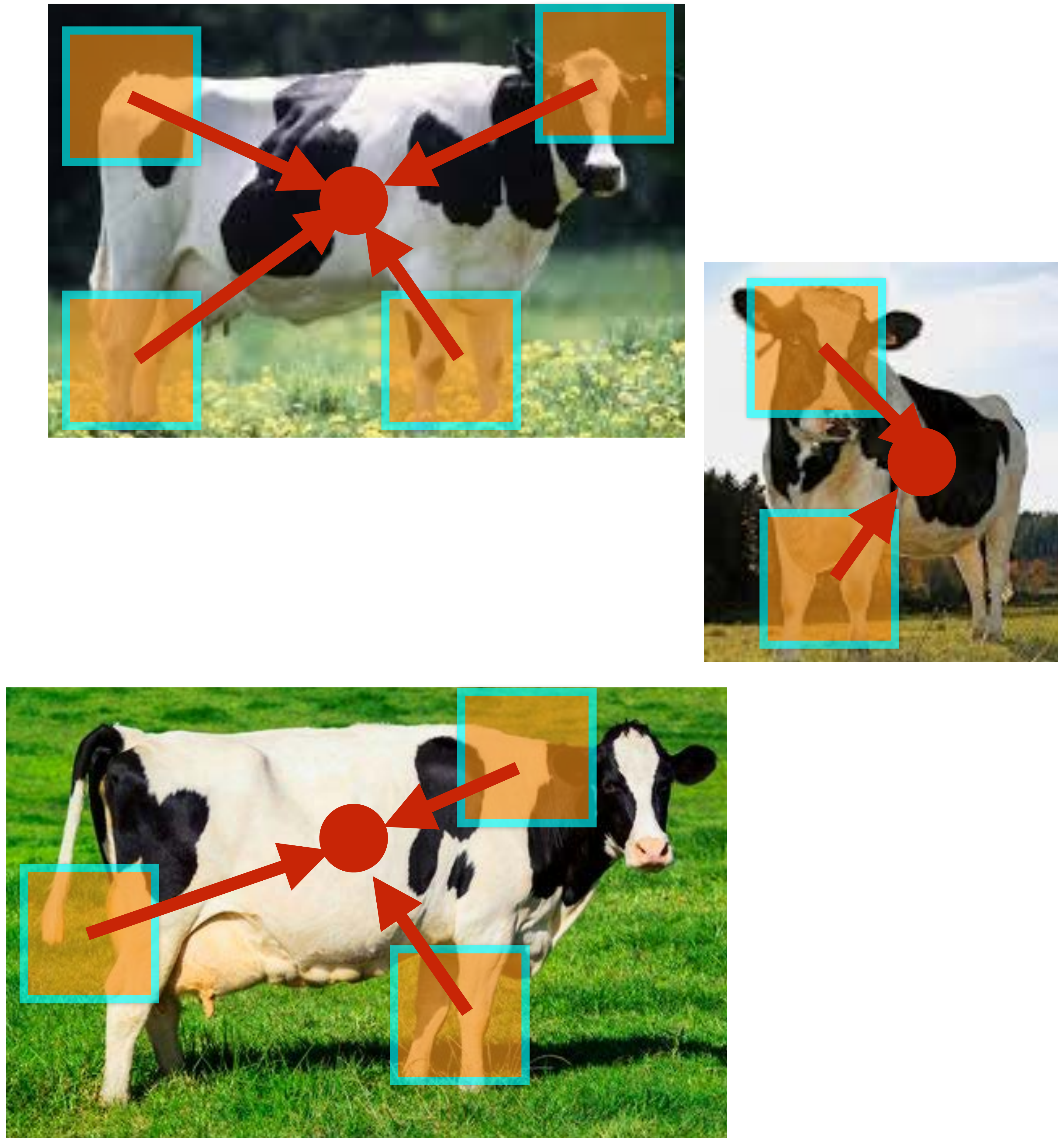
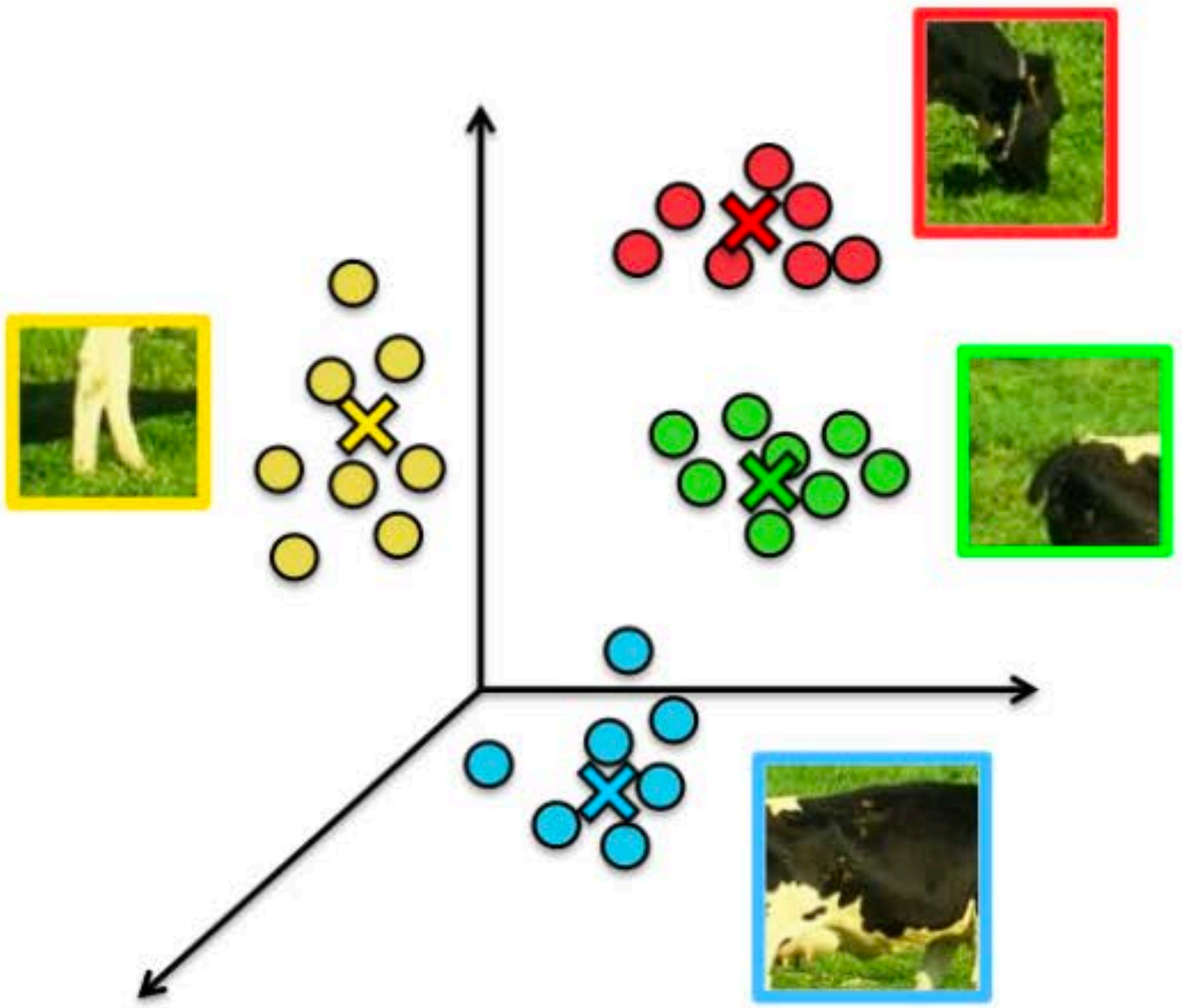


Image Index	Keypoint Index	Keypoint Detection (4D)	Keypoint Description (128D)	Offset to Centroid
Image 1	1	[x, y, s, Theta]	[…]	[x,y]
Image 1	2	[x, y, s, Theta]	[…]	[x,y]
….	….	….	…	…
Image 1	265	[x, y, s, Theta]	[…]	[x,y]
Image 2	1	[x, y, s, Theta]	[…]	[x,y]
Image 2	2	[x, y, s, Theta]	[…]	[x,y]
…	…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

Visual Words

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

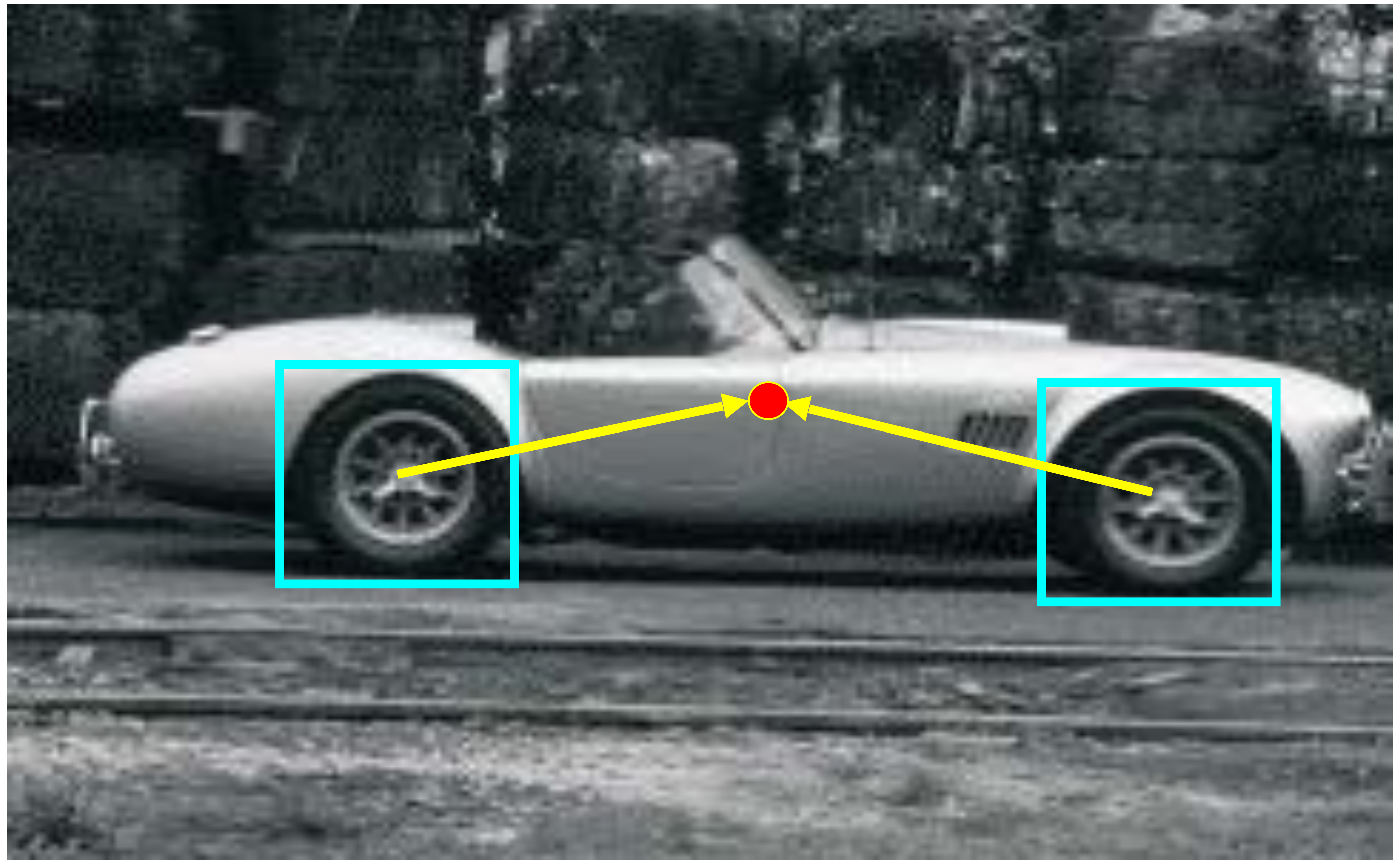


Visual words (visual codebook)!

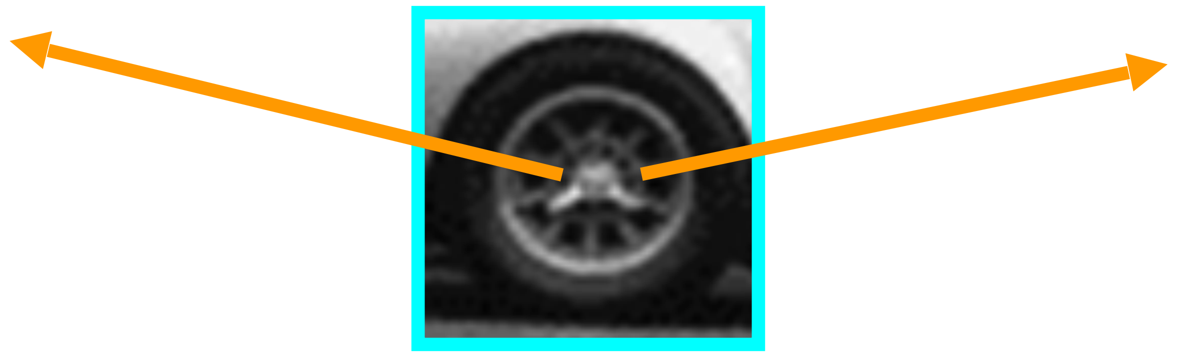
* Slide from Sanja Fidler

Example 1: Object Recognition — Implicit Shape Model

Index displacements by “visual codeword”



training image



visual codeword with
displacement vectors

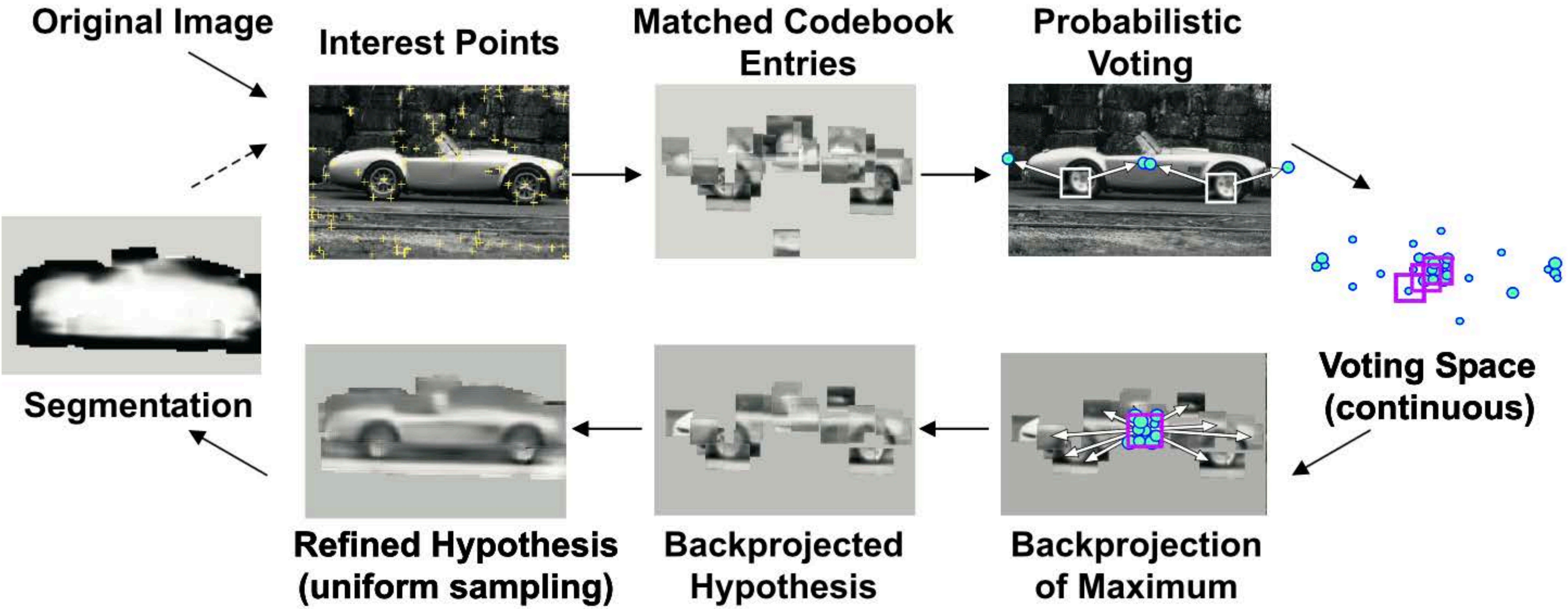
B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model,
ECCV Workshop on Statistical Learning in Computer Vision 2004

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

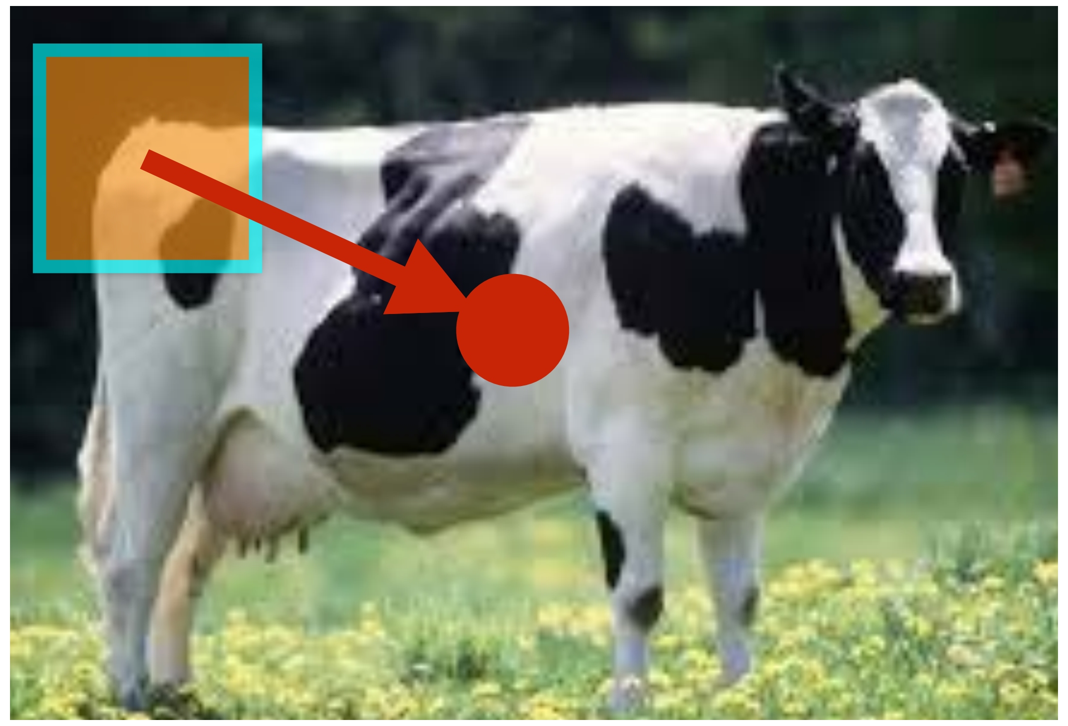
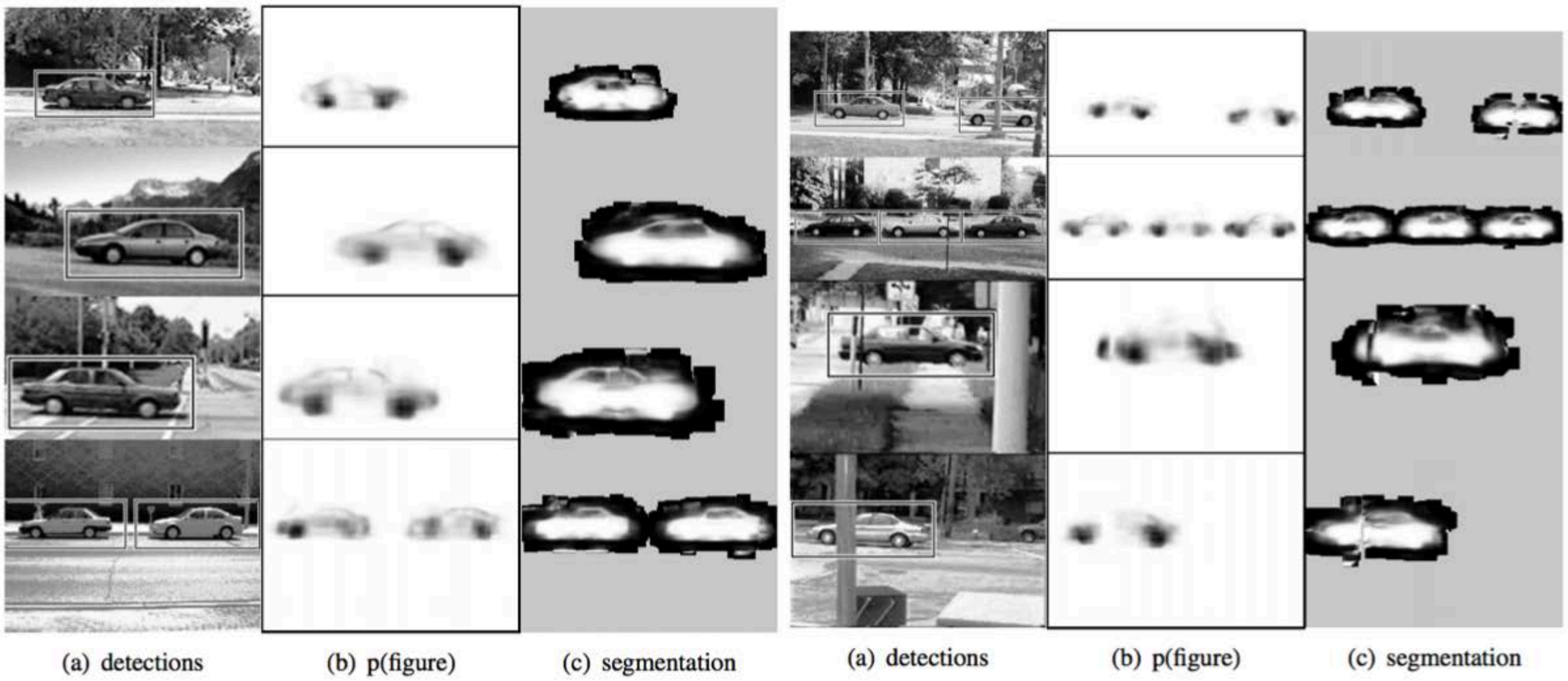


Image Index	Keypoint Index	Keypoint Detection (4D)	Keypoint Description (128D)	Offset to Centroid	Segment
Image 1	1	[x, y, s, Theta]	[…]	[x,y]	
Image 1	2	[x, y, s, Theta]	[…]	[x,y]	
….	….	….	…	…	
Image 1	265	[x, y, s, Theta]	[…]	[x,y]	
Image 2	1	[x, y, s, Theta]	[…]	[x,y]	
Image 2	2	[x, y, s, Theta]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

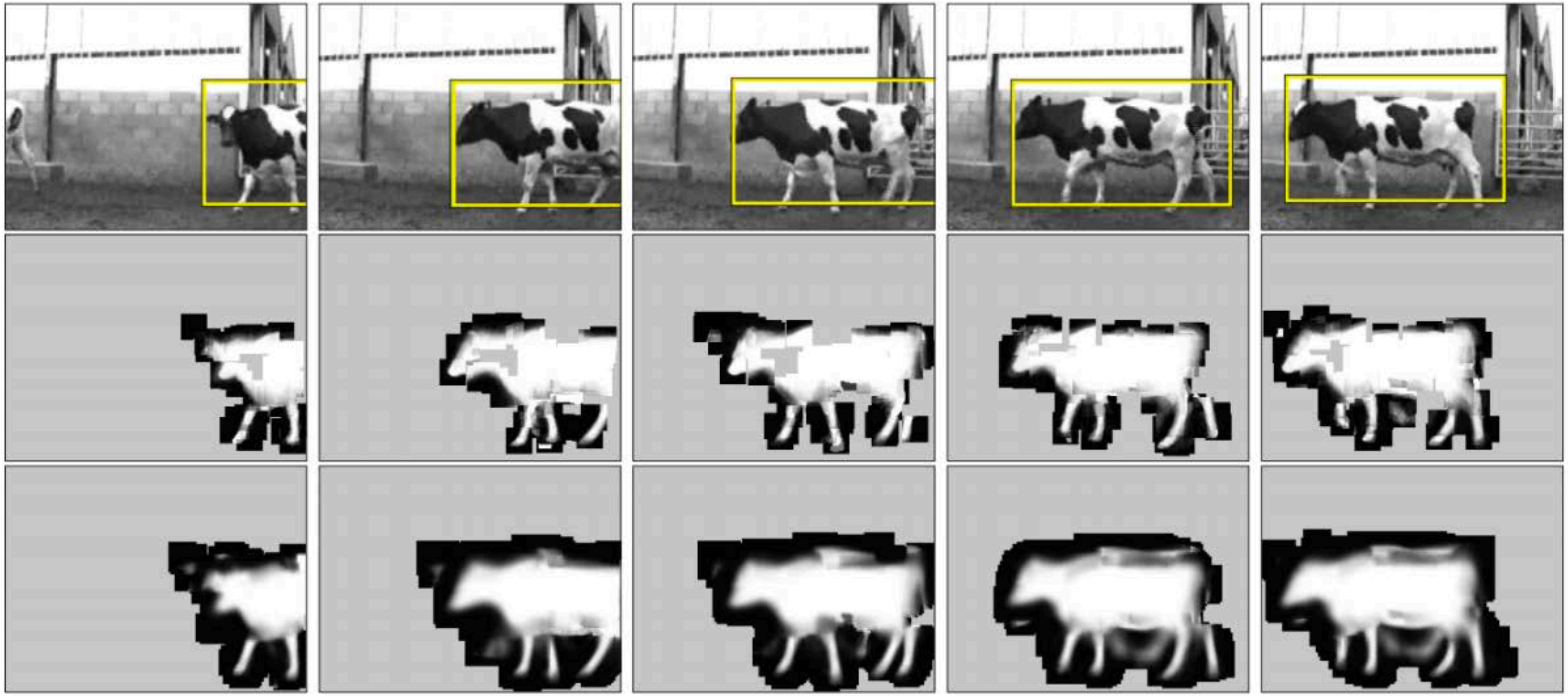
Idea: When back-projecting, back-project labeled segmentations per training patch



[Source: B. Leibe]

* Slide from Sanja Fidler

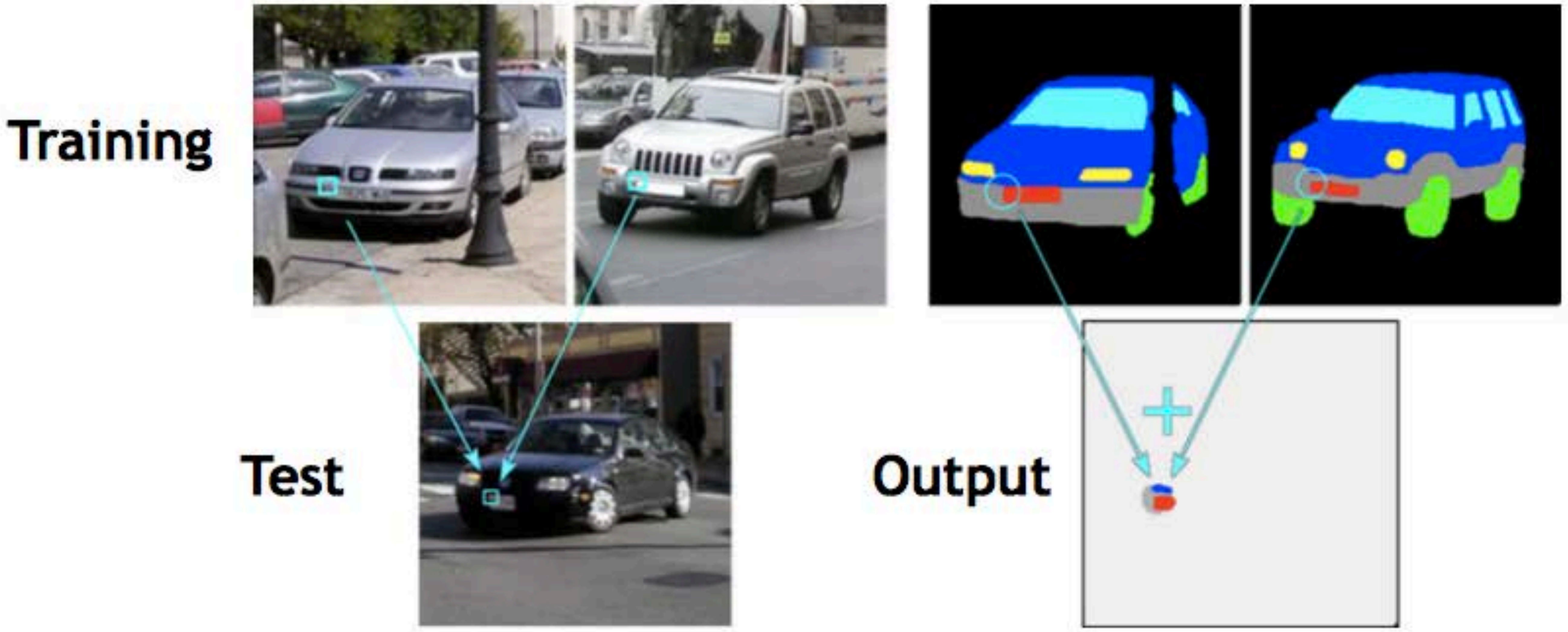
Inferring Other Information: **Segmentation**



[Source: B. Leibe]

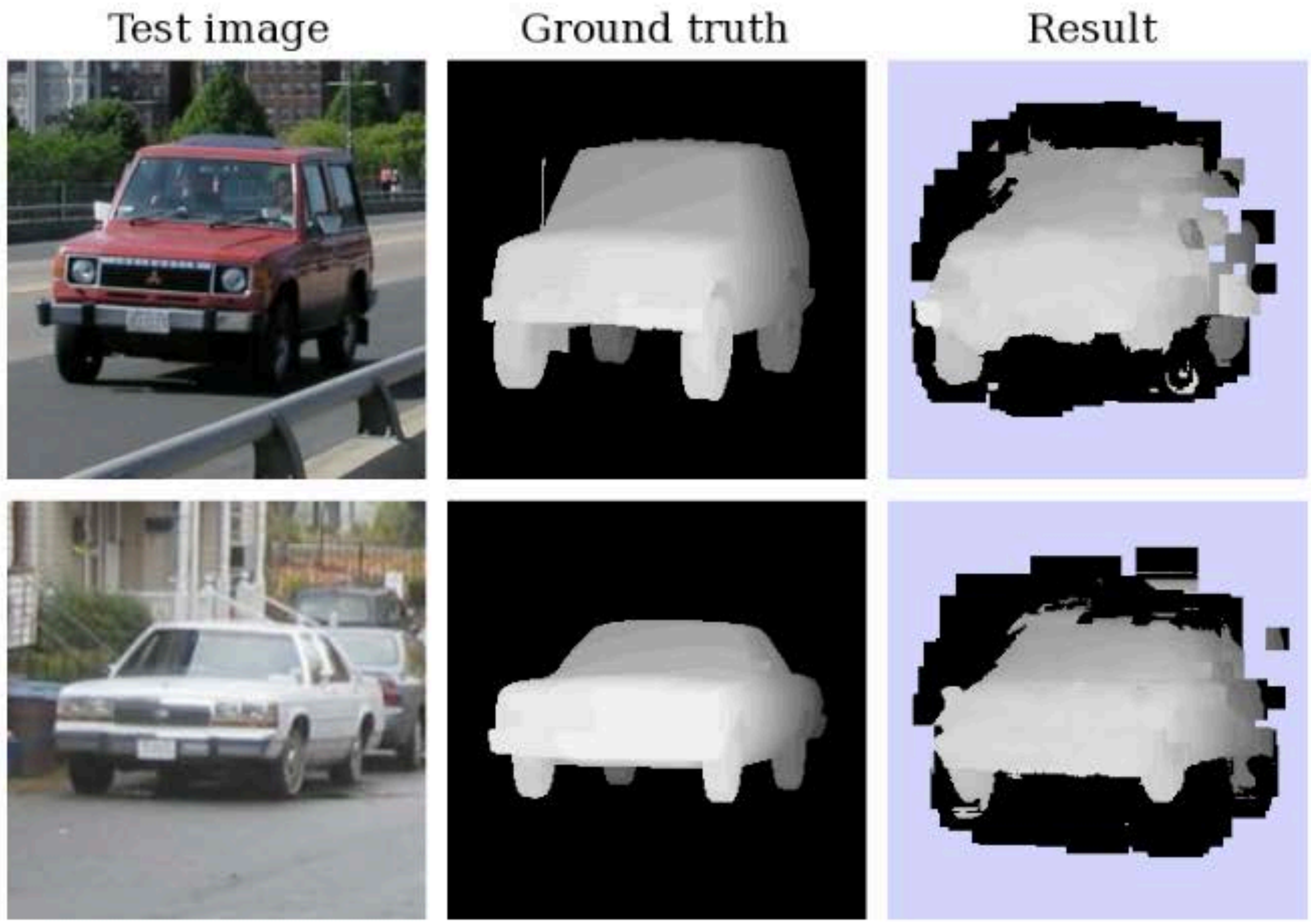
* Slide from Sanja Fidler

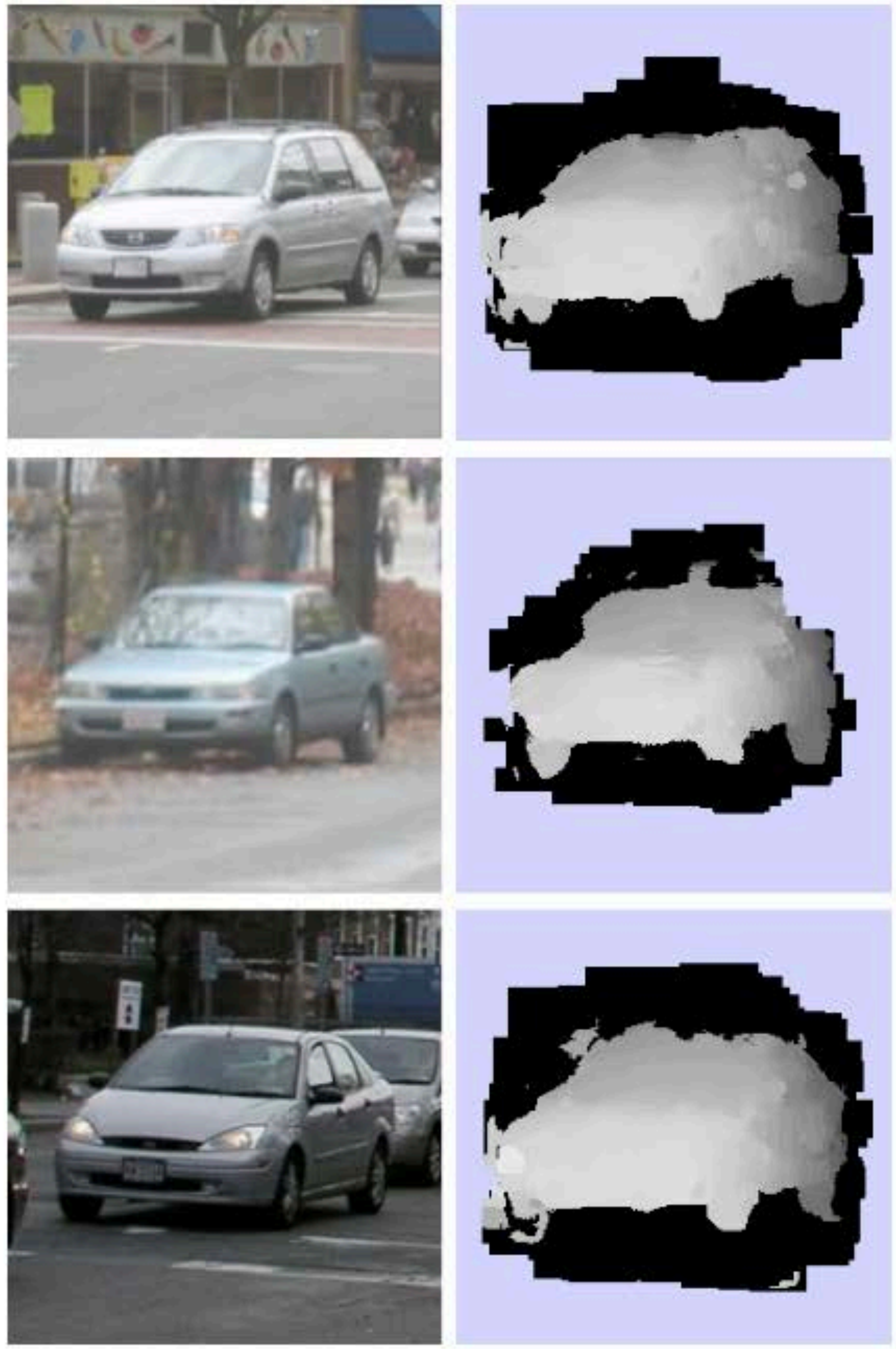
Inferring Other Information: **Part Labels**



* Slide from Sanja Fidler

Inferring Other Information: **Depth**





"Depth from a single image"

Example 2: Object Recognition — Boundary Fragments

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

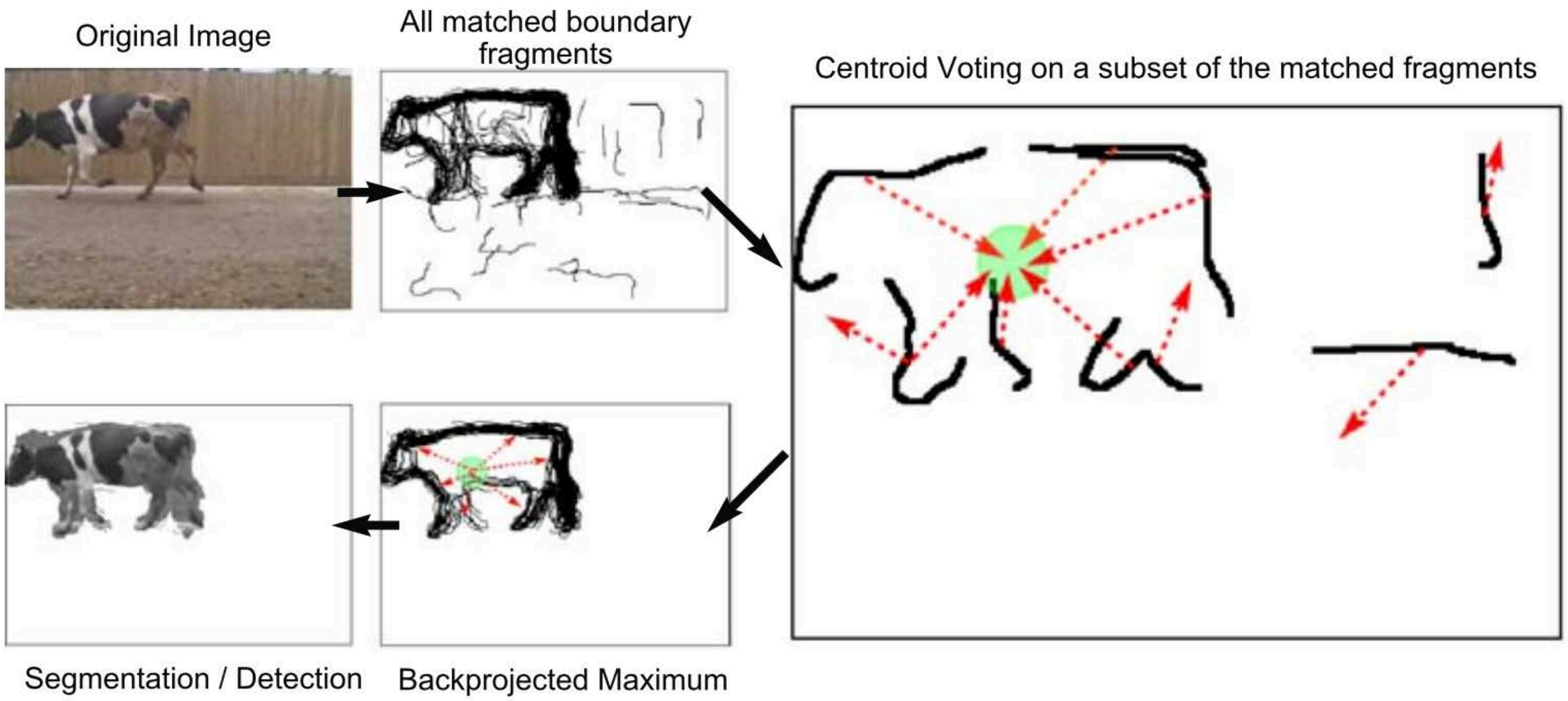


Image credit: Opelt et al., 2006

Example 2: Object Recognition — Boundary Fragments

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

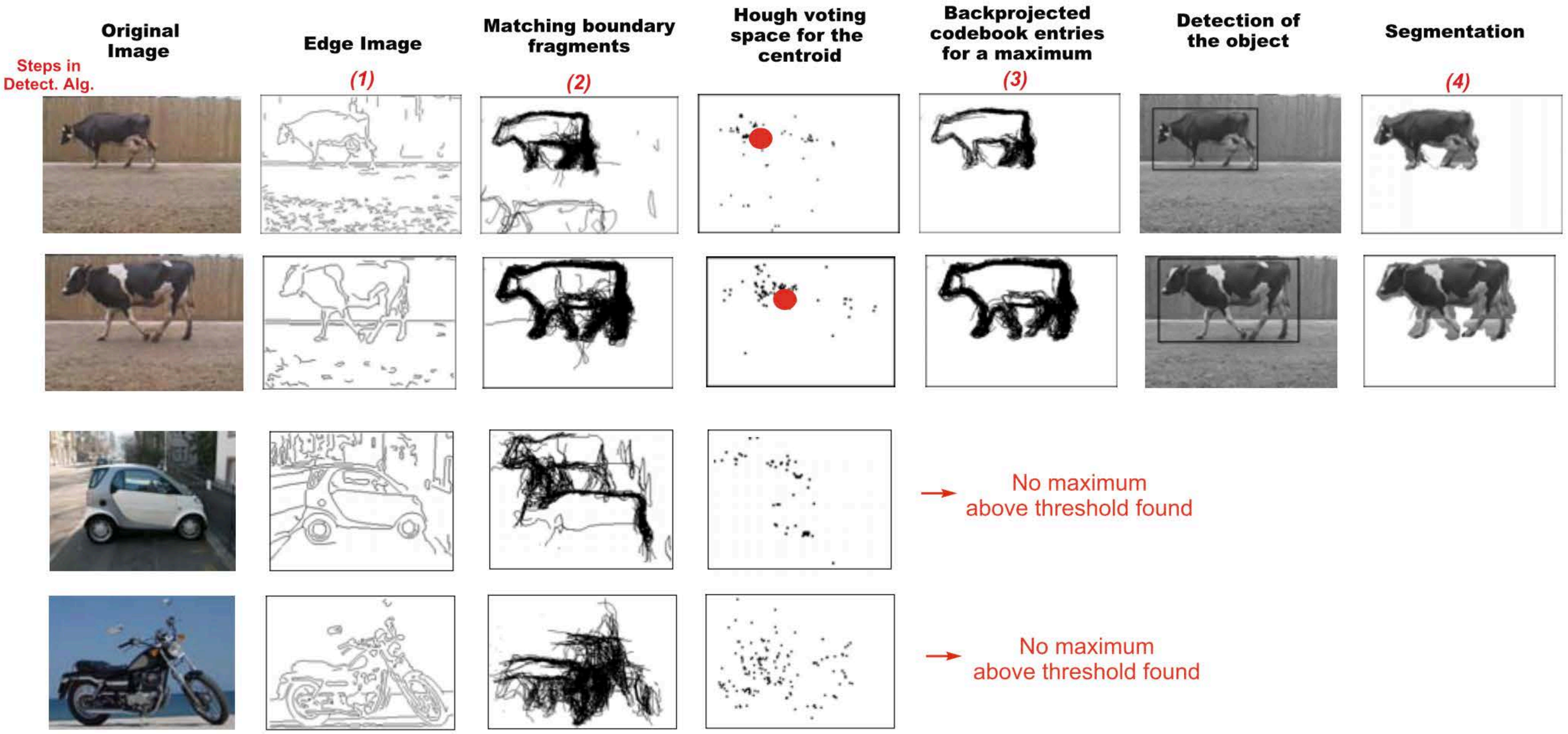


Image credit: Opelt et al., 2006

Example 3: Deep Hough Voting

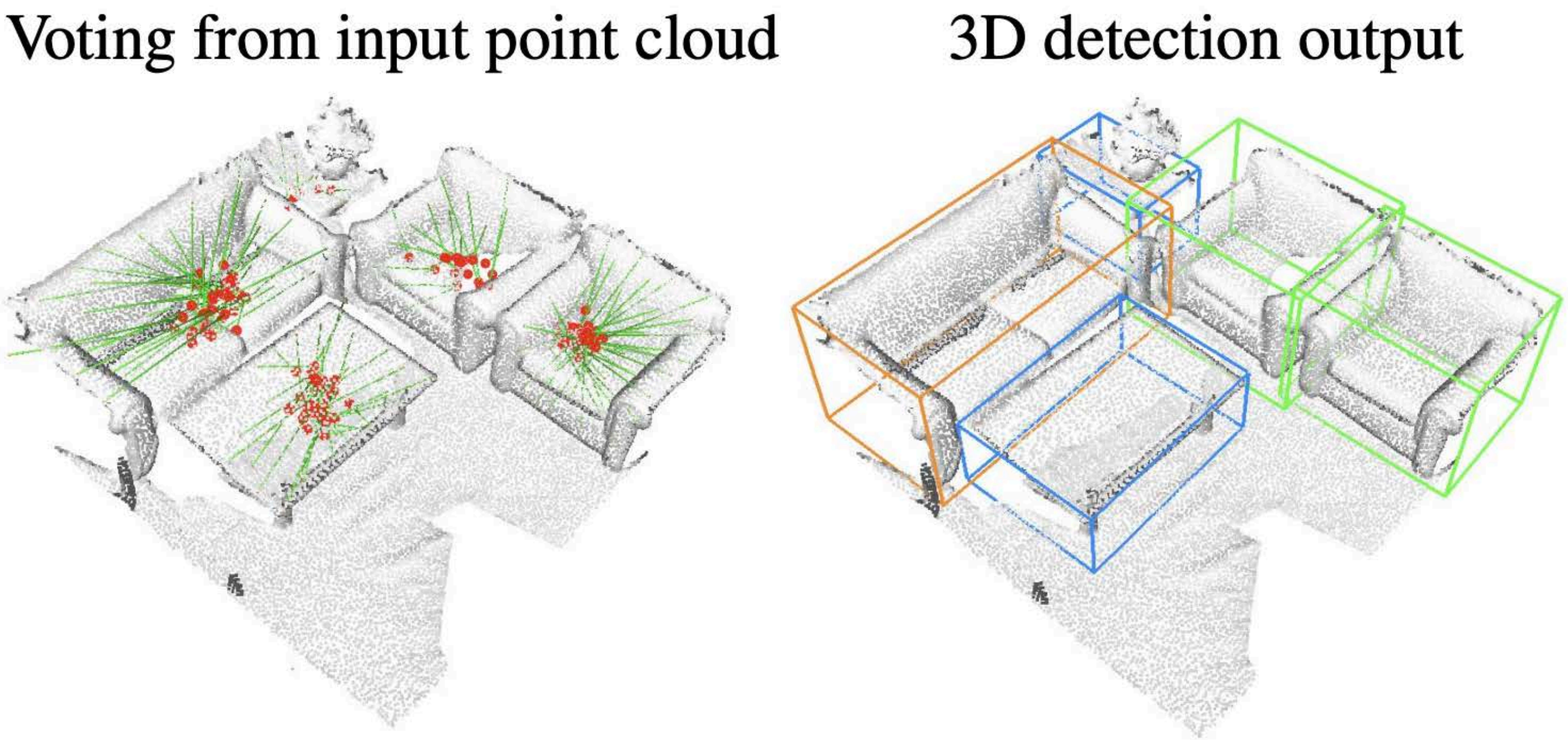


Figure 1. **3D object detection in point clouds with a deep Hough voting model.** Given a point cloud of a 3D scene, our VoteNet votes to object centers and then groups and aggregates the votes to predict 3D bounding boxes and semantic classes of objects.

[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

THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision

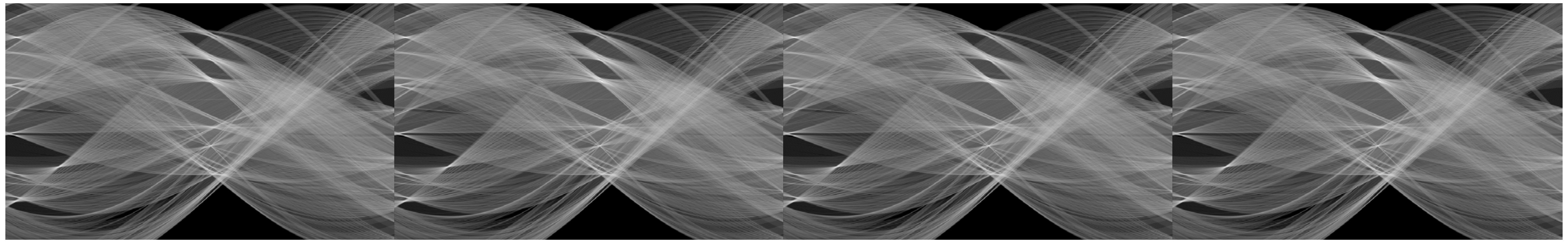


Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 15: Stereo

Menu for Today

Topics:

— 3D Correspondence, **Epipolar** Geometry

— **Stereo** Vision

Readings:

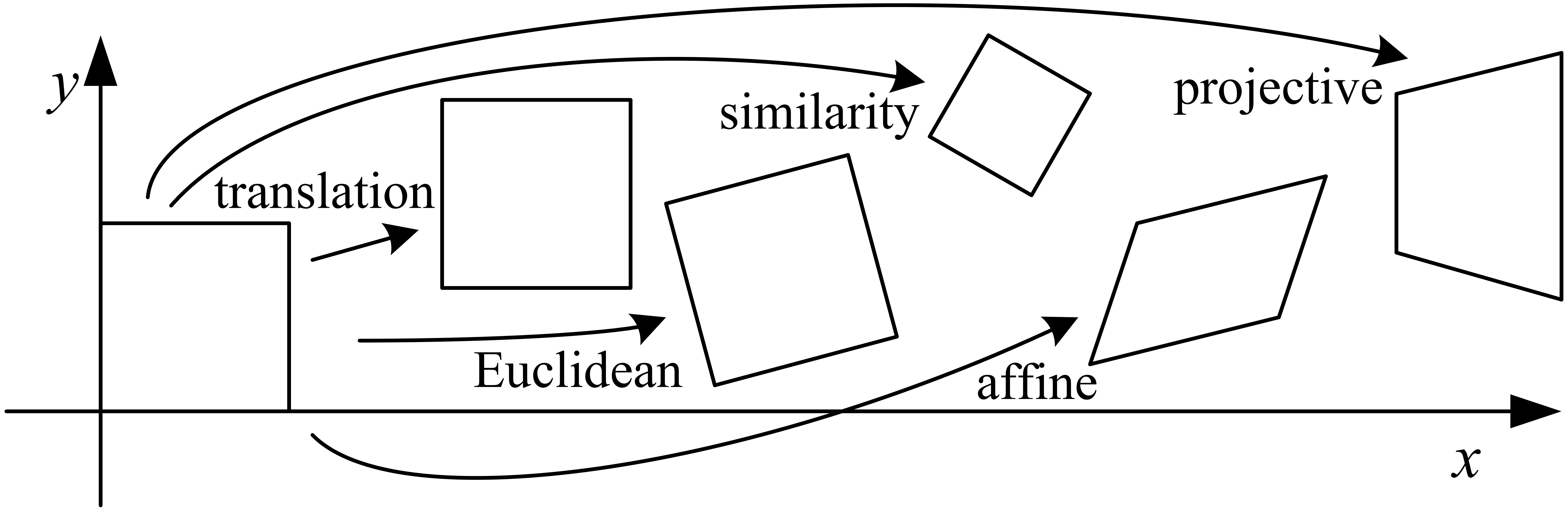
— **Today's** Lecture: Szeliski 12.1, 12.3-12.4, 9.3

Reminders:

— **Assignment 4**: RANSAC and Panoramas due **March 20th**

Recap: 2D **Transformations**

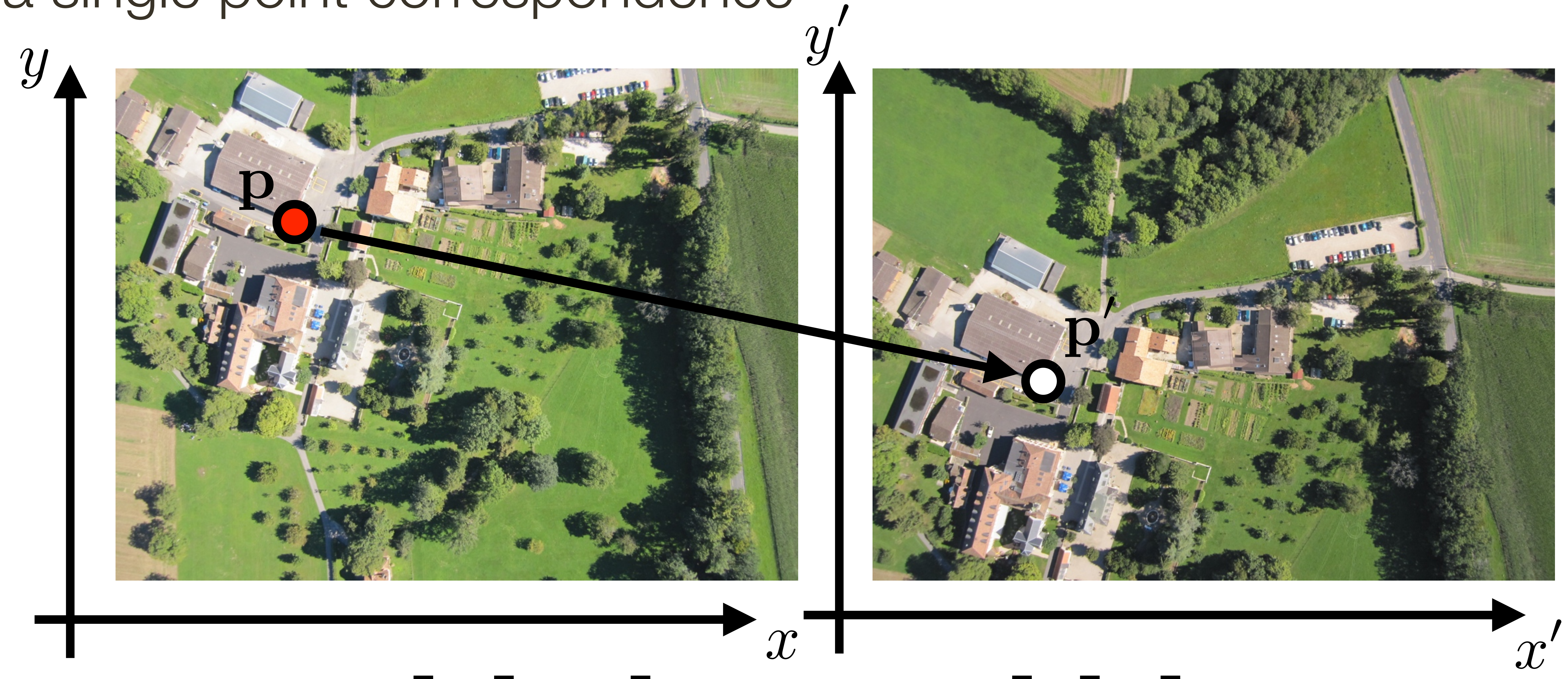
— We will look at a family that can be represented by 3x3 matrices



— This group represents perspective projections of **planar surfaces**

Recap: **Linear** (or Affine) Transformations

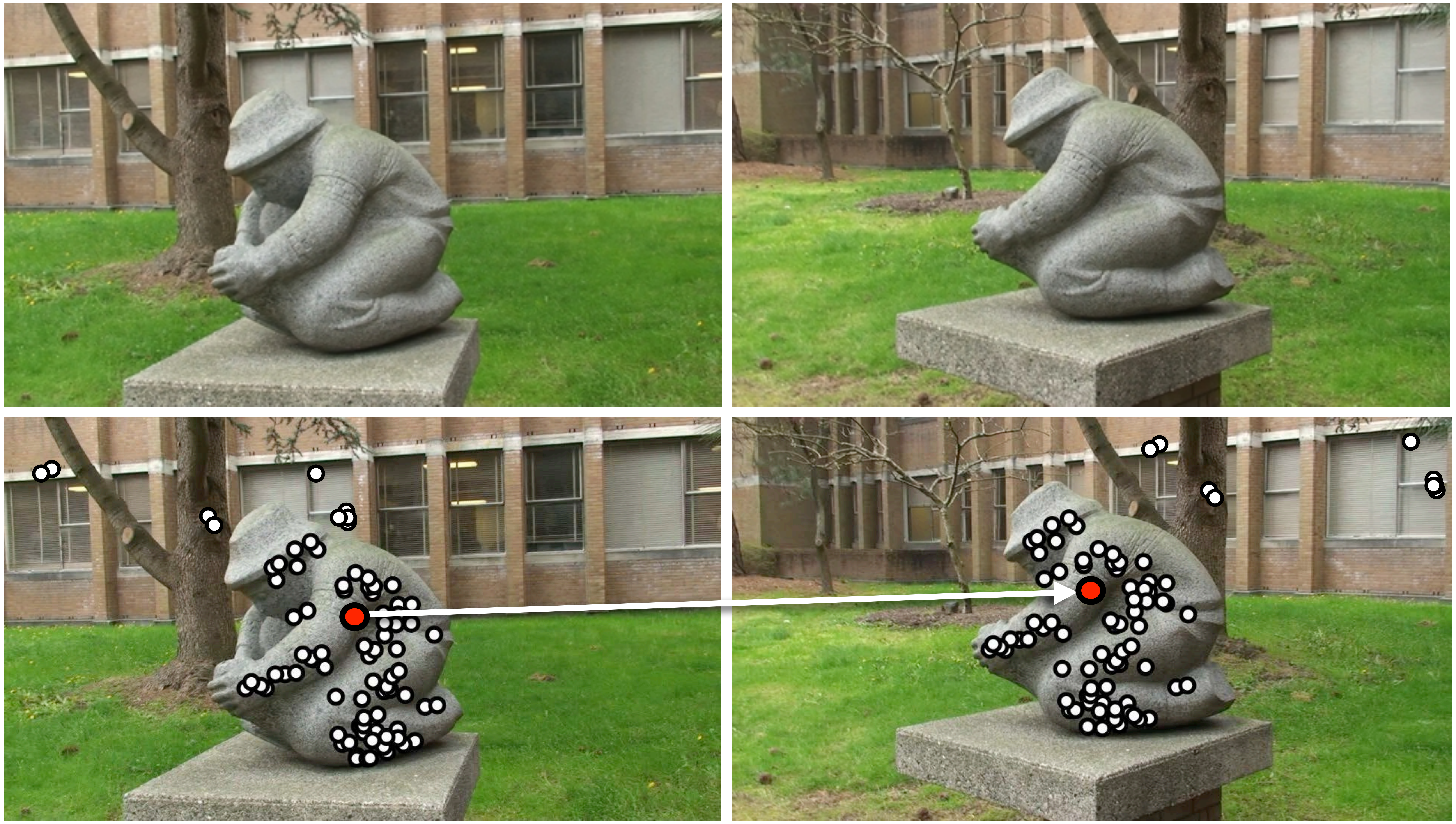
Consider a single point correspondence



$$\begin{bmatrix} x_1' \\ y_1' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

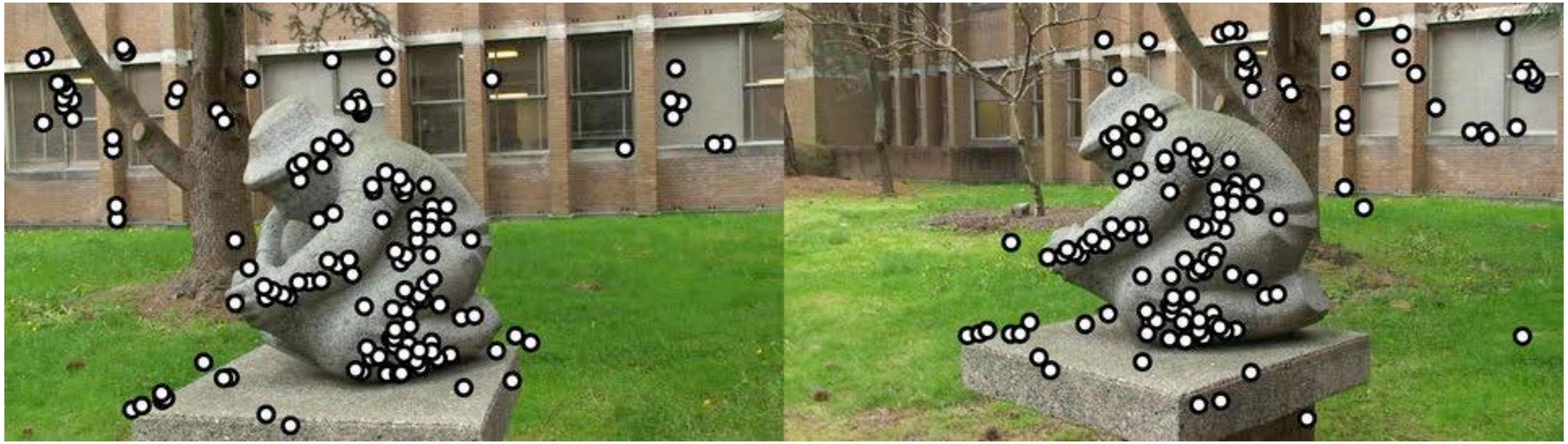
Correspondences in **3D**

Find all matches between views



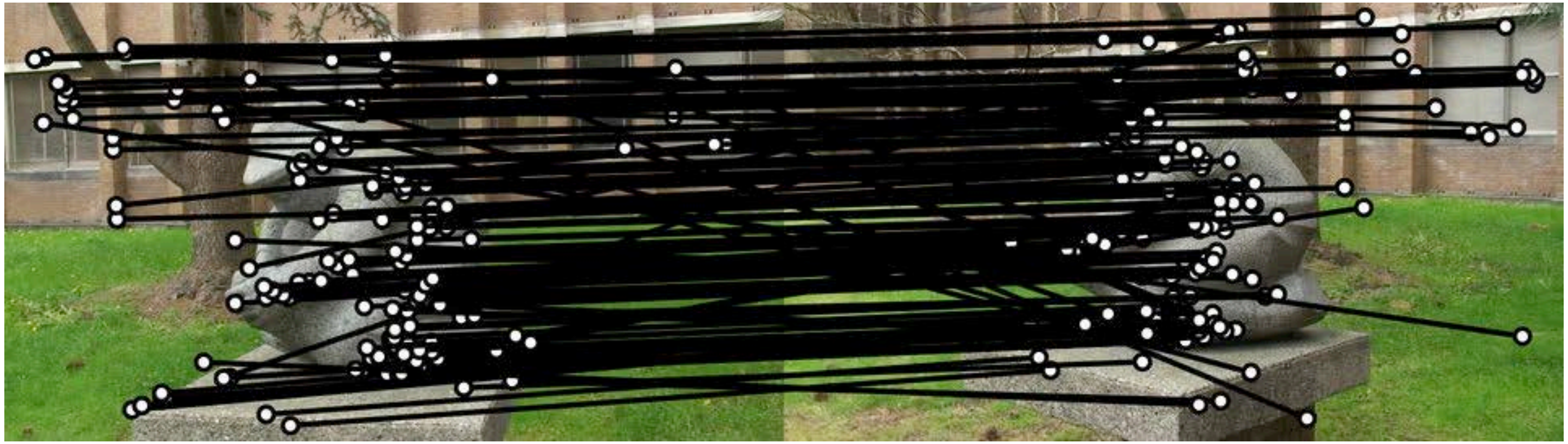
Correspondences in **3D**

Find subset of matches that are consistent with a geometric transformation



Correspondences in **3D**

Find subset of matches that are consistent with a geometric transformation

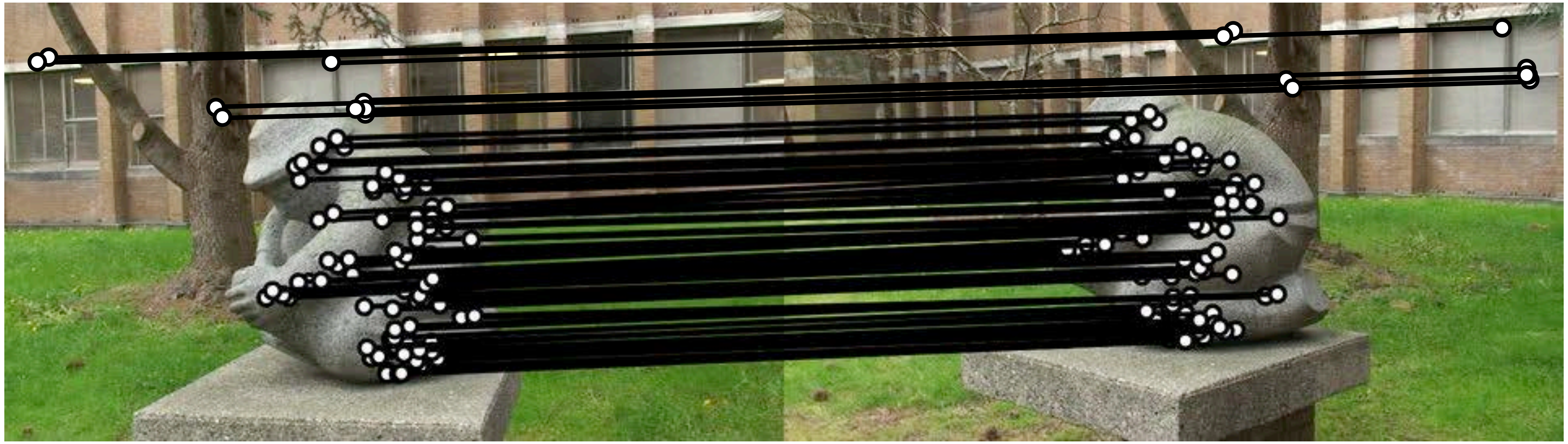


Correspondences in **3D**

Find subset of matches that are consistent with a geometric transformation

Correspondences in **3D**

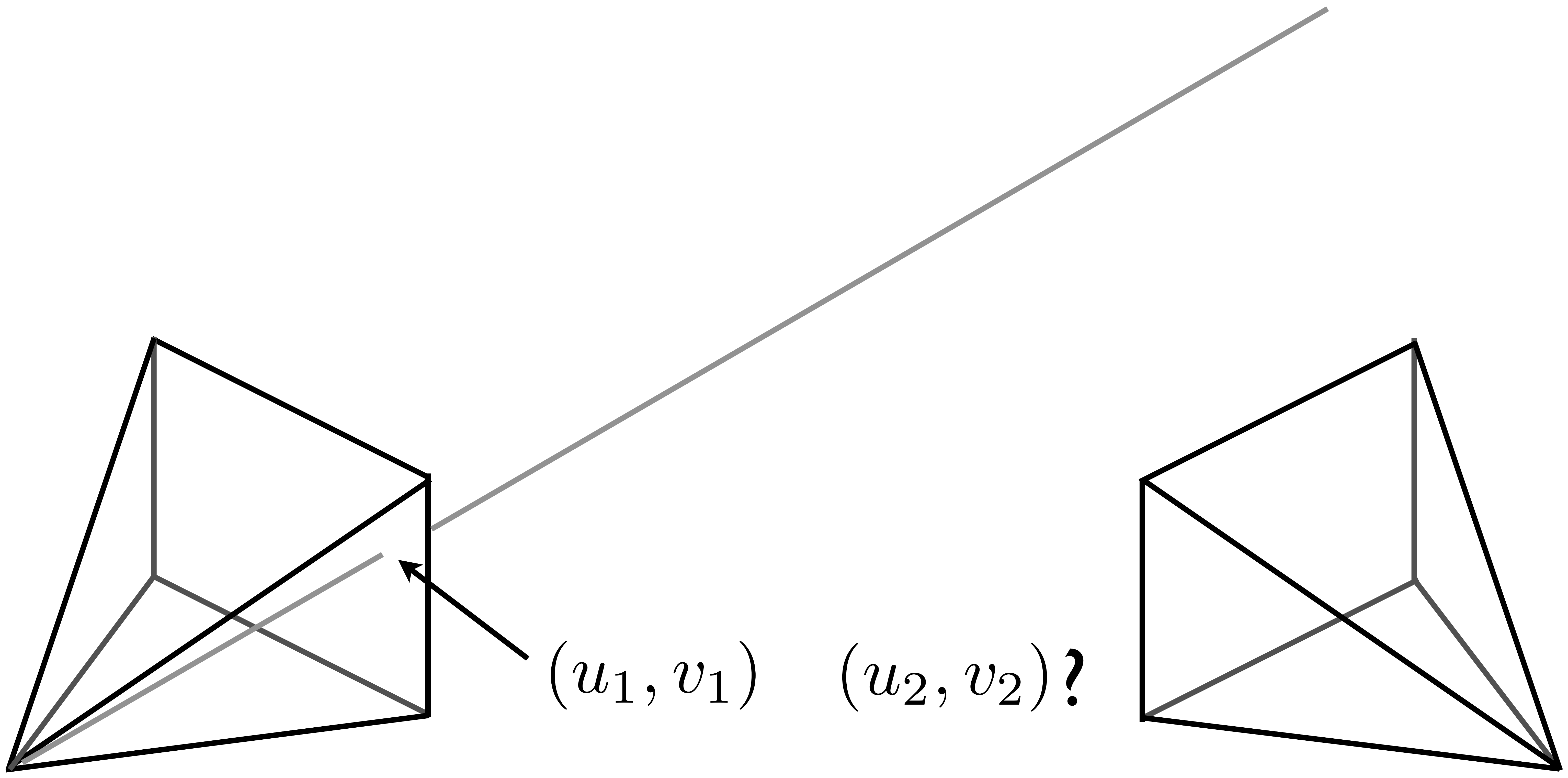
Find subset of matches that are consistent with a geometric transformation



Consistent matches can be used for subsequent stages, e.g., 3D reconstruction, object recognition etc.

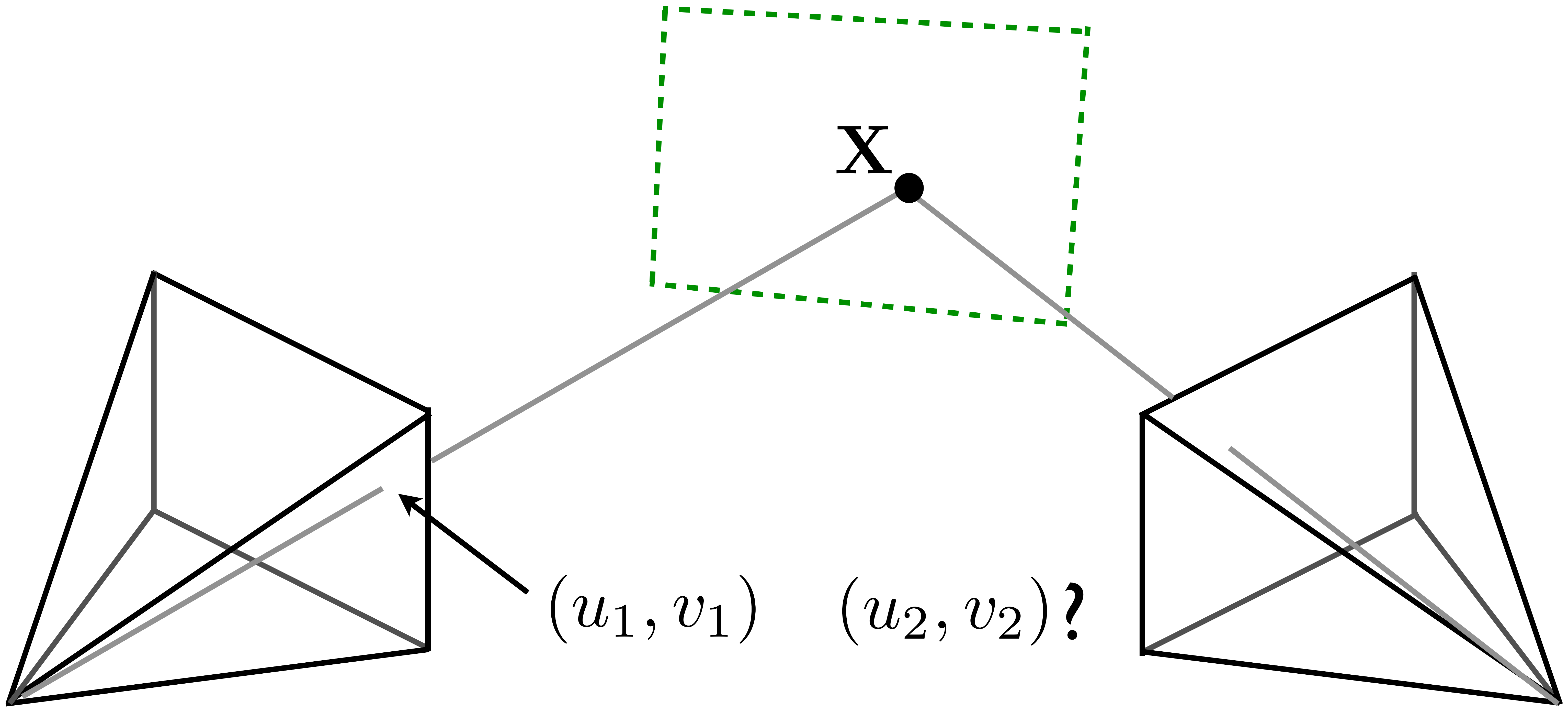
2-view **Geometry**

How do we find correspondences between two views?



2-view **Geometry**

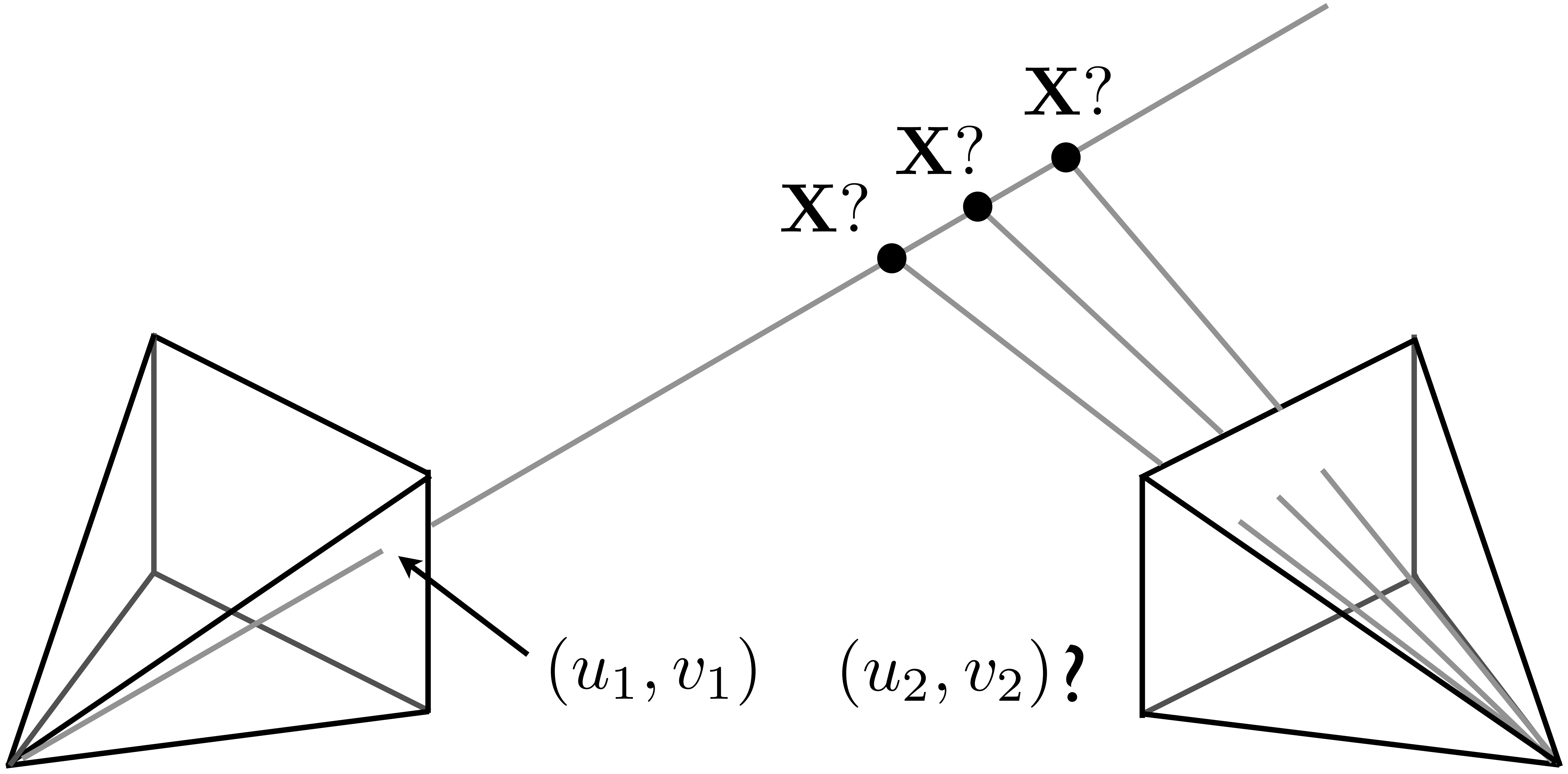
How do we find correspondences between two views?



Planar case: the mapping can be obtained by a homography

2-view **Geometry**

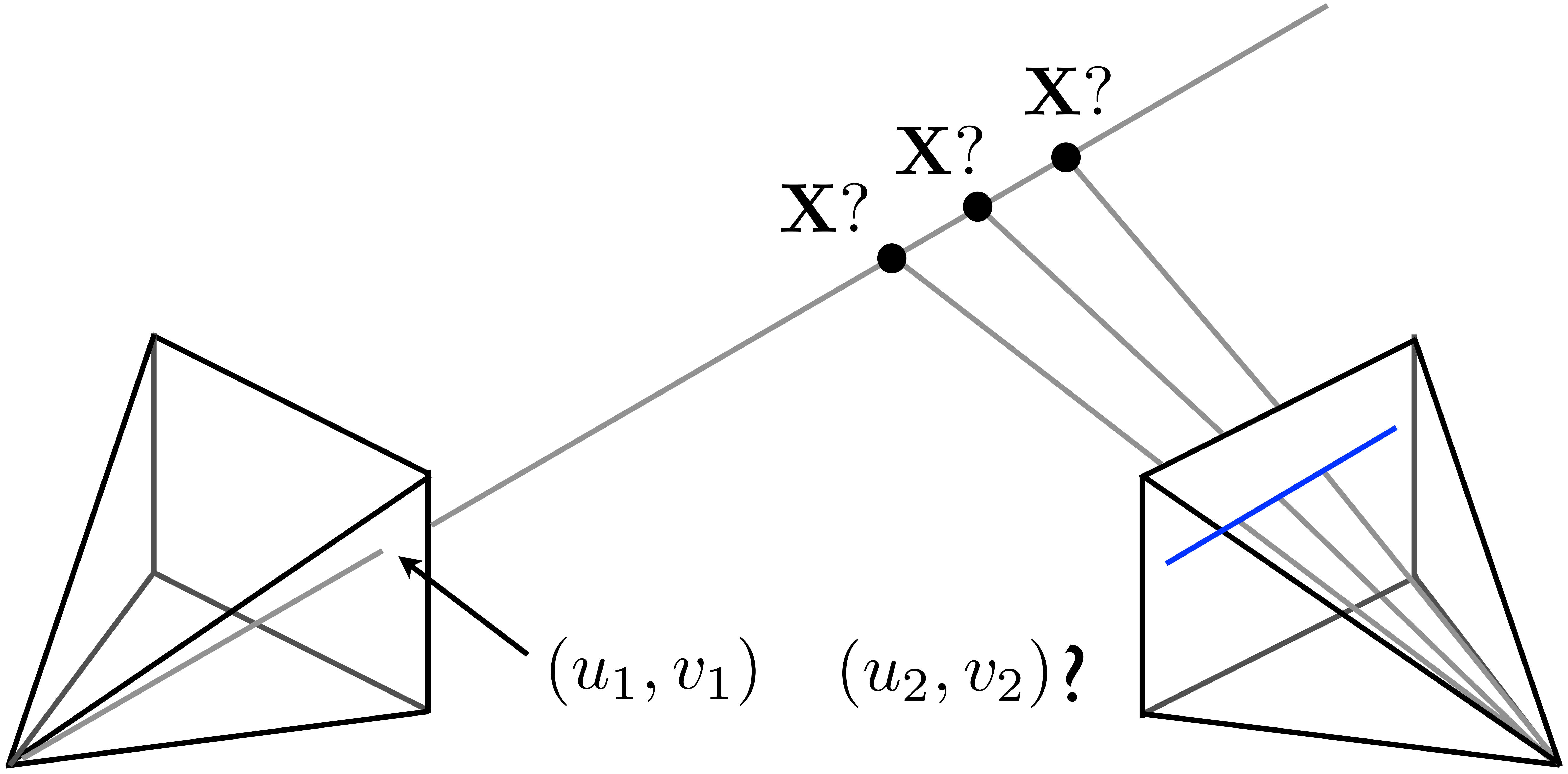
How do we find correspondences between two views?



Non-planar case: depends on the depth of the 3D point

Epipolar Line

How do we find correspondences between two views?



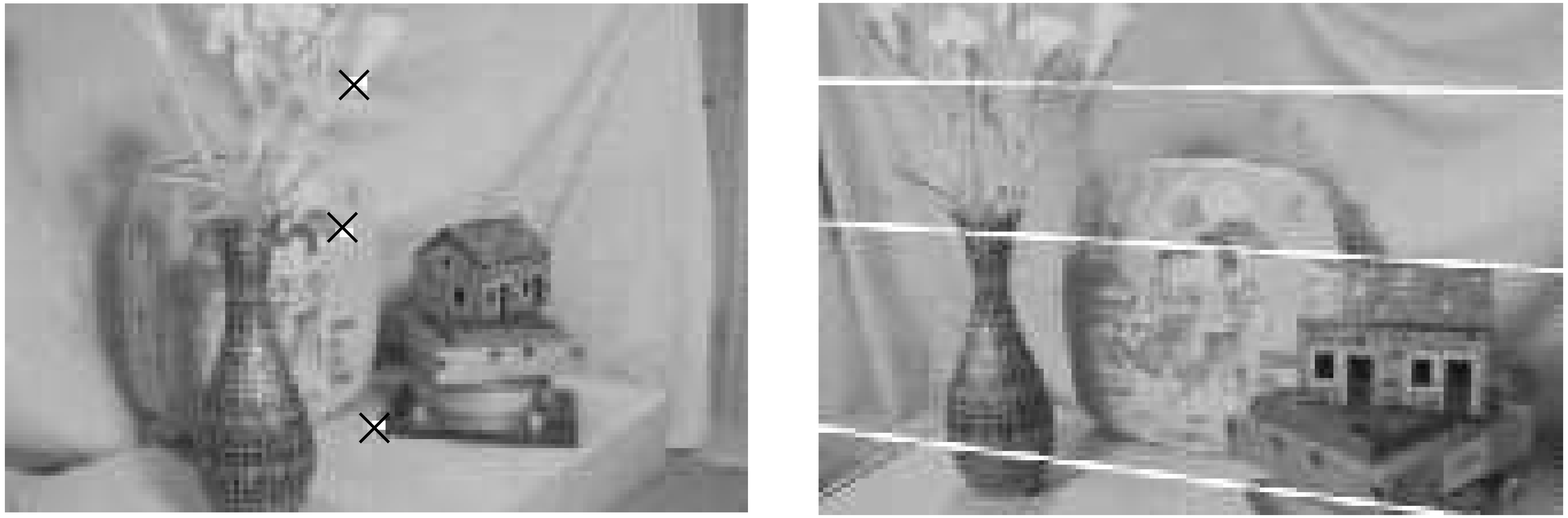
A point in Image 1 must lie along the **line** in Image 2

2-view **Stereo**

Search over matches constrained to (epipolar) line

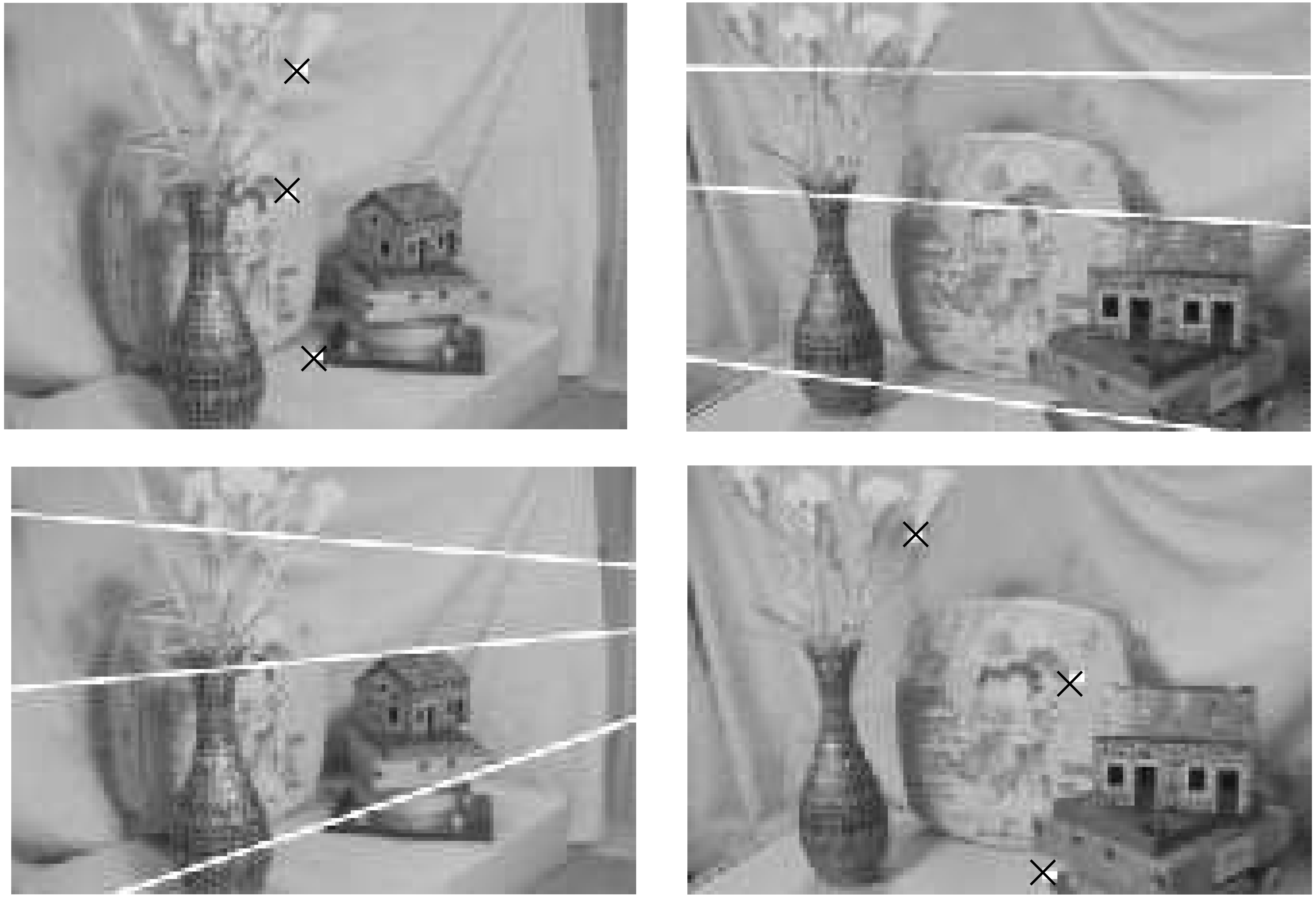
(reduces to 1d search)

Visualization of **Epipolar Lines**



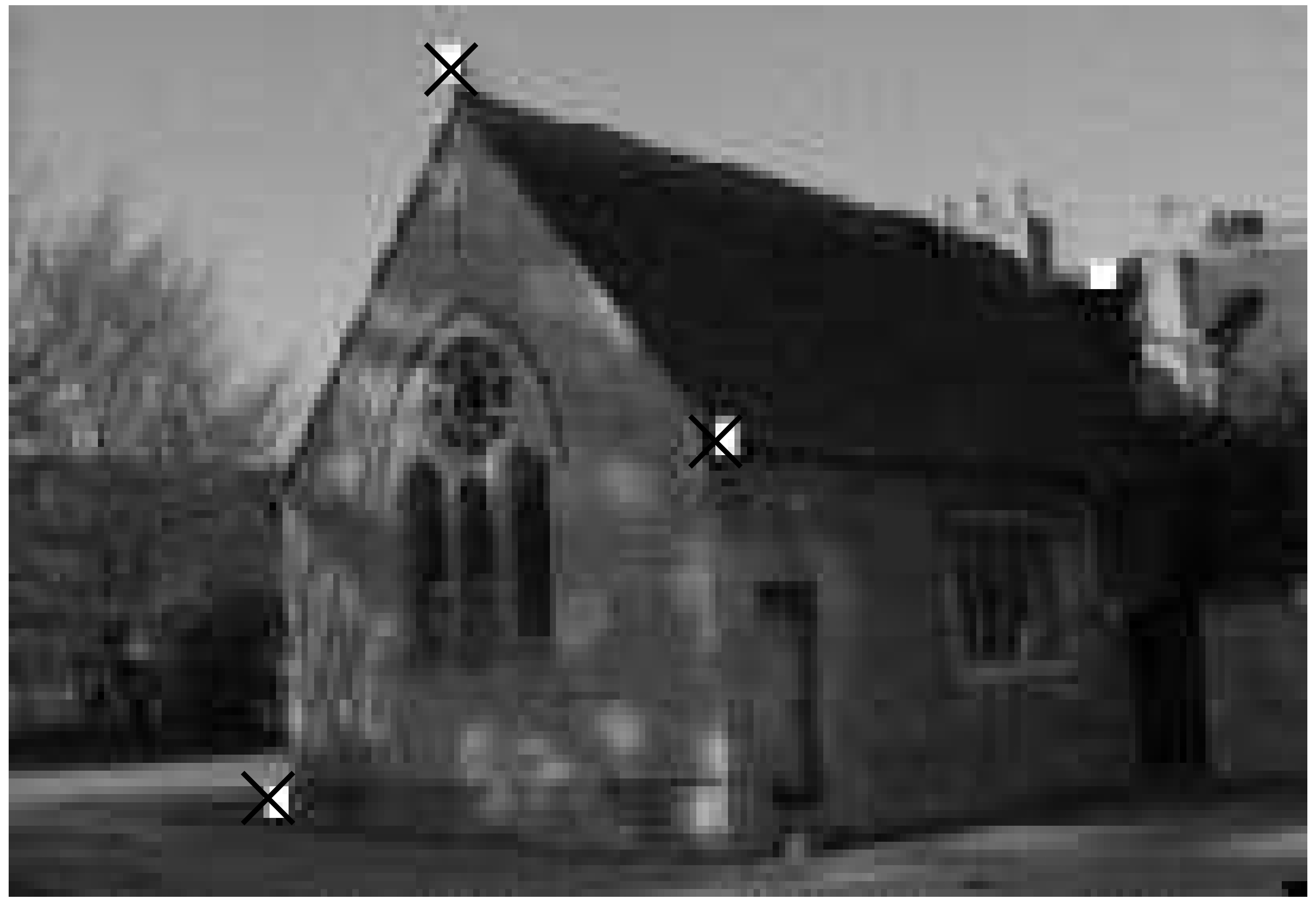
[R. Cipolla]

Visualization of **Epipolar Lines**



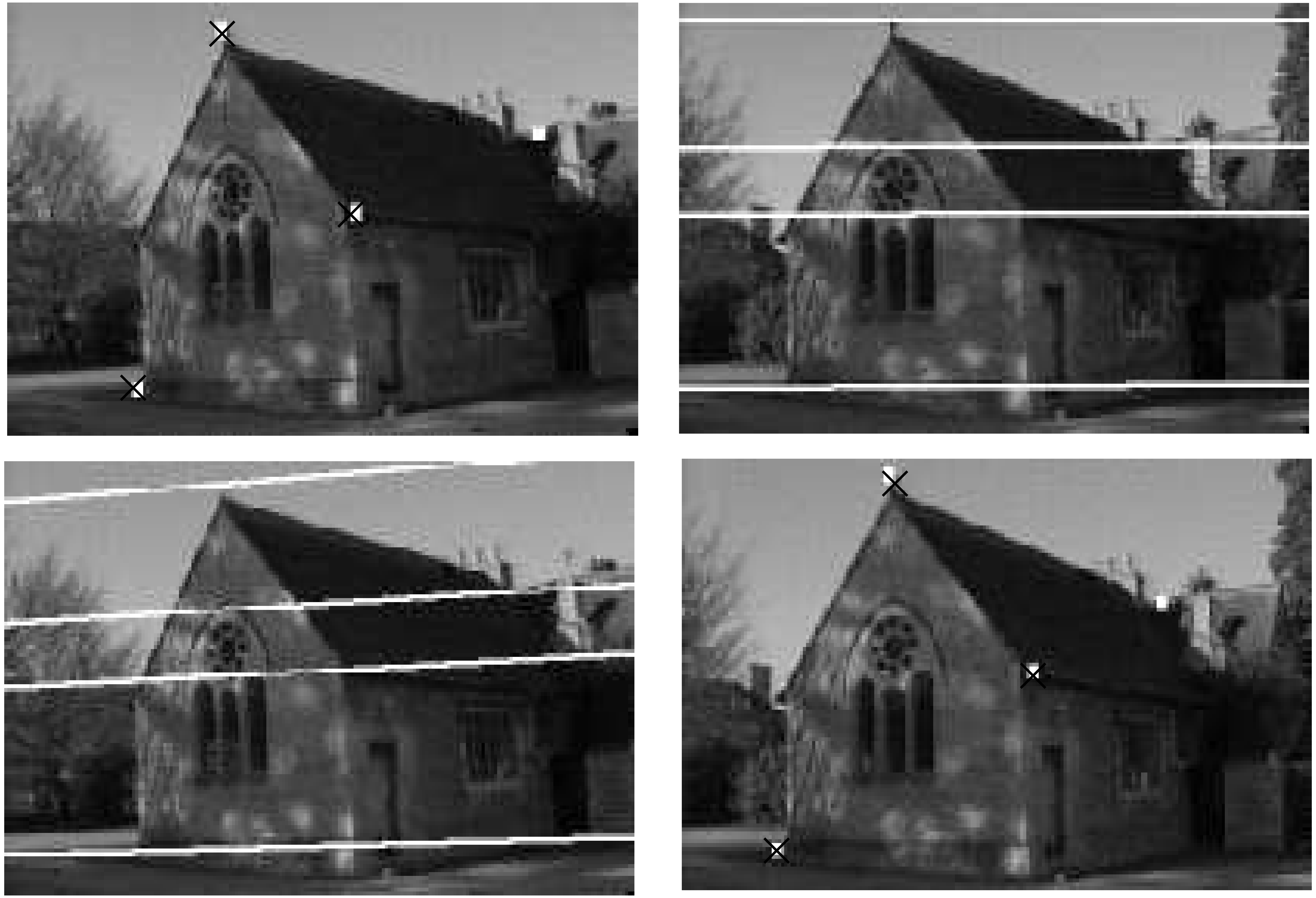
[R. Cipolla]

Visualization of **Epipolar Lines**



[R. Cipolla]

Visualization of **Epipolar Lines**

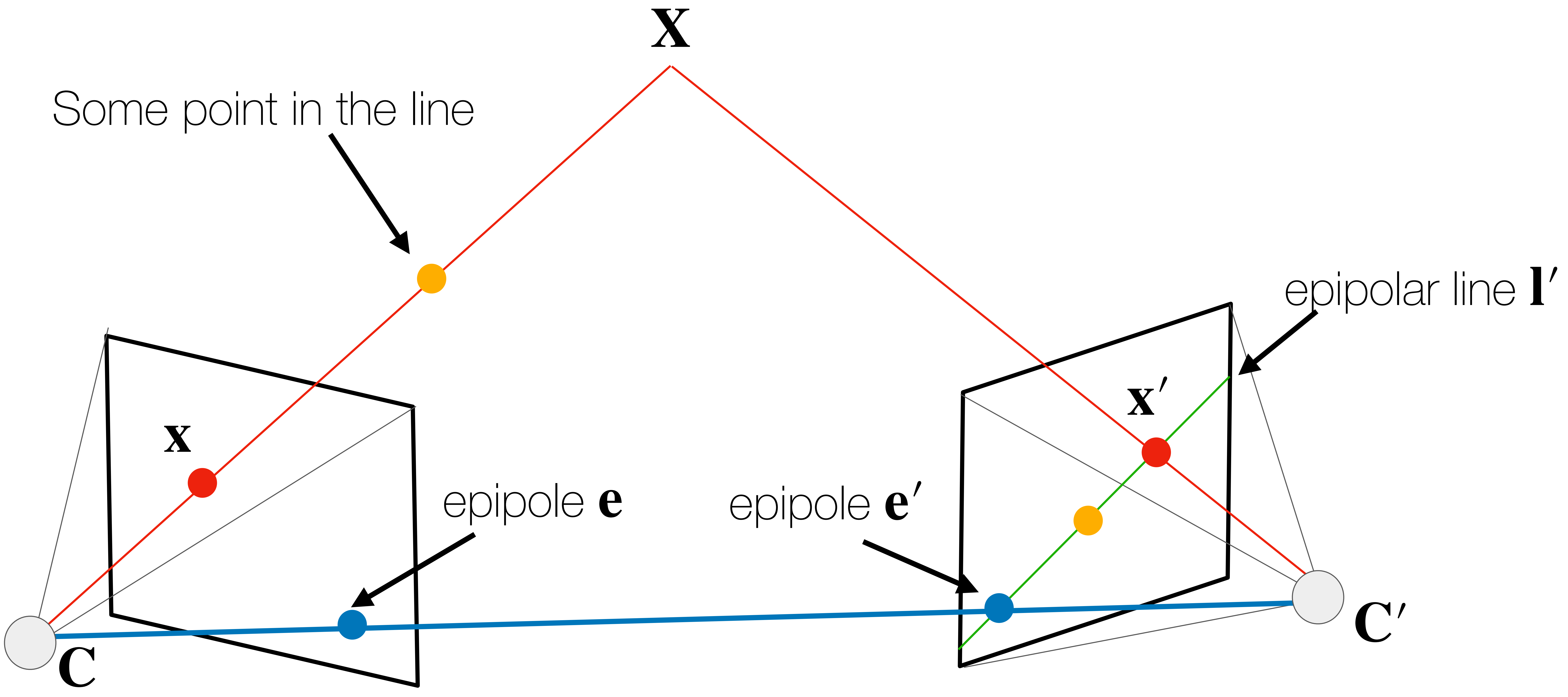


[R. Cipolla]

Aside: The **Epipolar Constraint** — CPSC533Y

For *any* pair of corresponding points $\mathbf{x} \leftrightarrow \mathbf{x}'$ in the two images

$$\mathbf{x}'^{\top}\mathbf{F}\mathbf{x} = 0$$



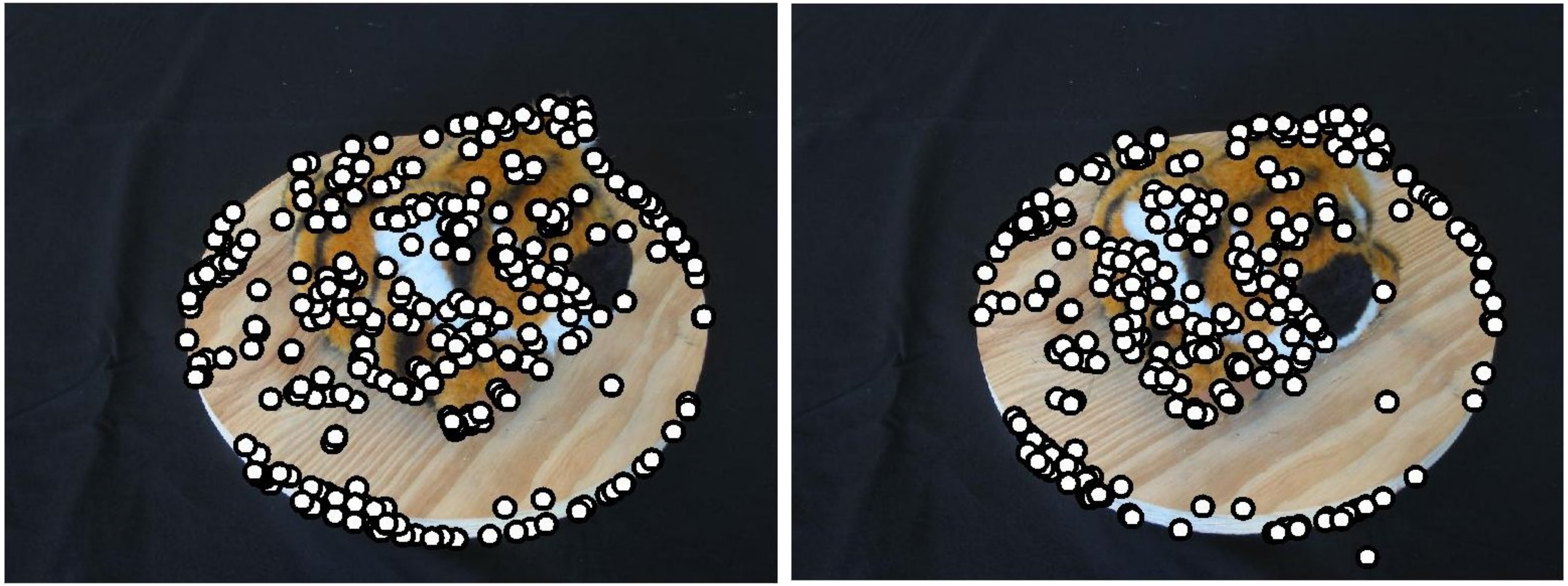
For the motivated: https://www.icloud.com/keynote/0lMsw0TLJioSA-HpXlPpXm2rw#lect_part2_1_epipolar_geom

Improving RANSAC + Alignment with **Epipolar Geometry**

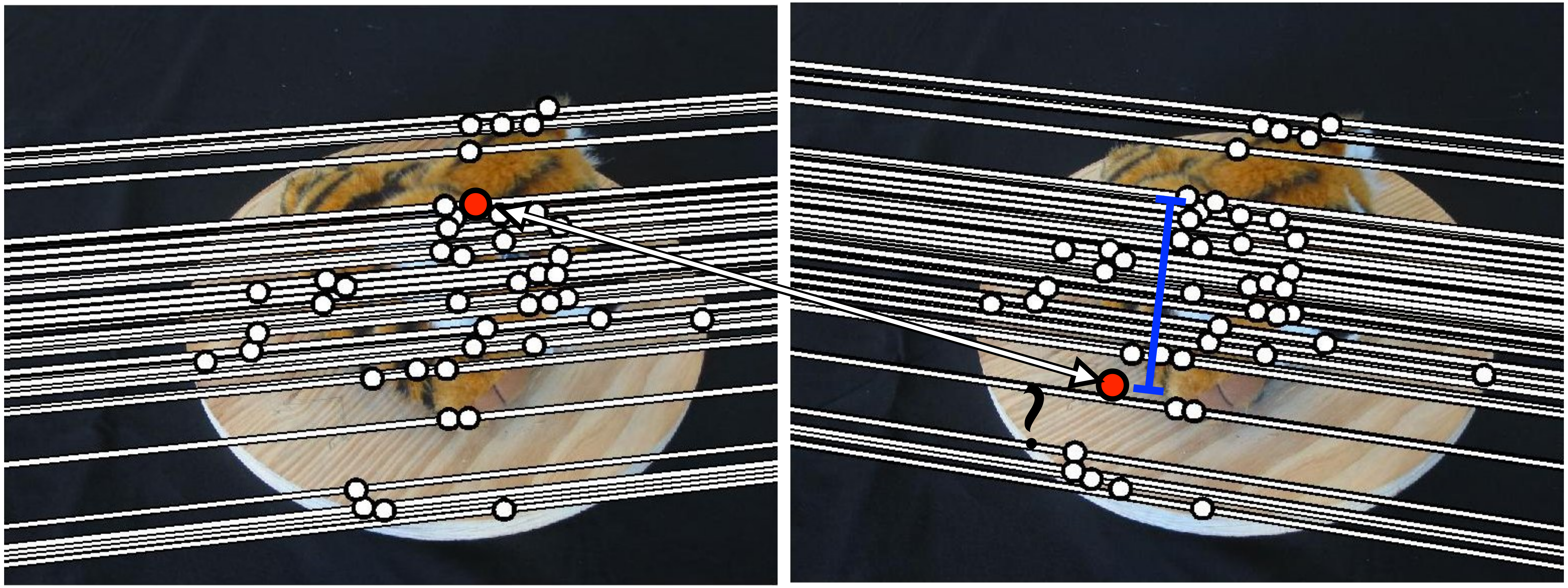
Improving RANSAC + Alignment with **Epipolar Geometry**

Raw SIFT features and their matches



Improving RANSAC + Alignment with **Epipolar Geometry**

Instead of matching purely based on SIFT descriptor, leverage geometry to obtain matches close to epipolar lines



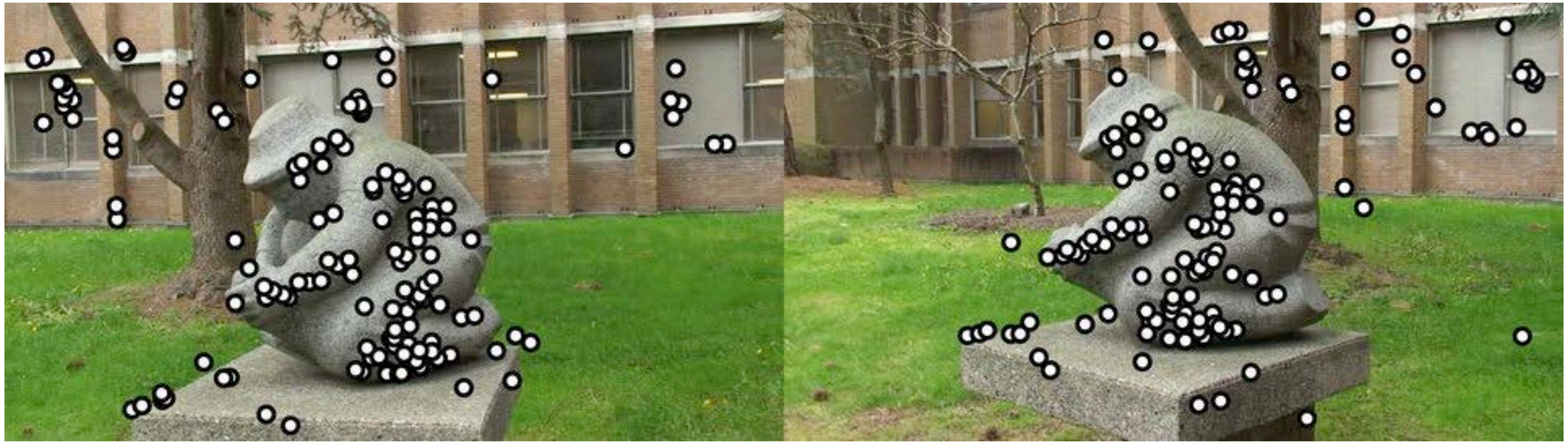
(gives more consistent geometrically valid matches)

Improving RANSAC + Alignment with **Epipolar Geometry**

Better matches lead to fewer iterations of RANSAC

(gives more consistent geometrically valid matches)

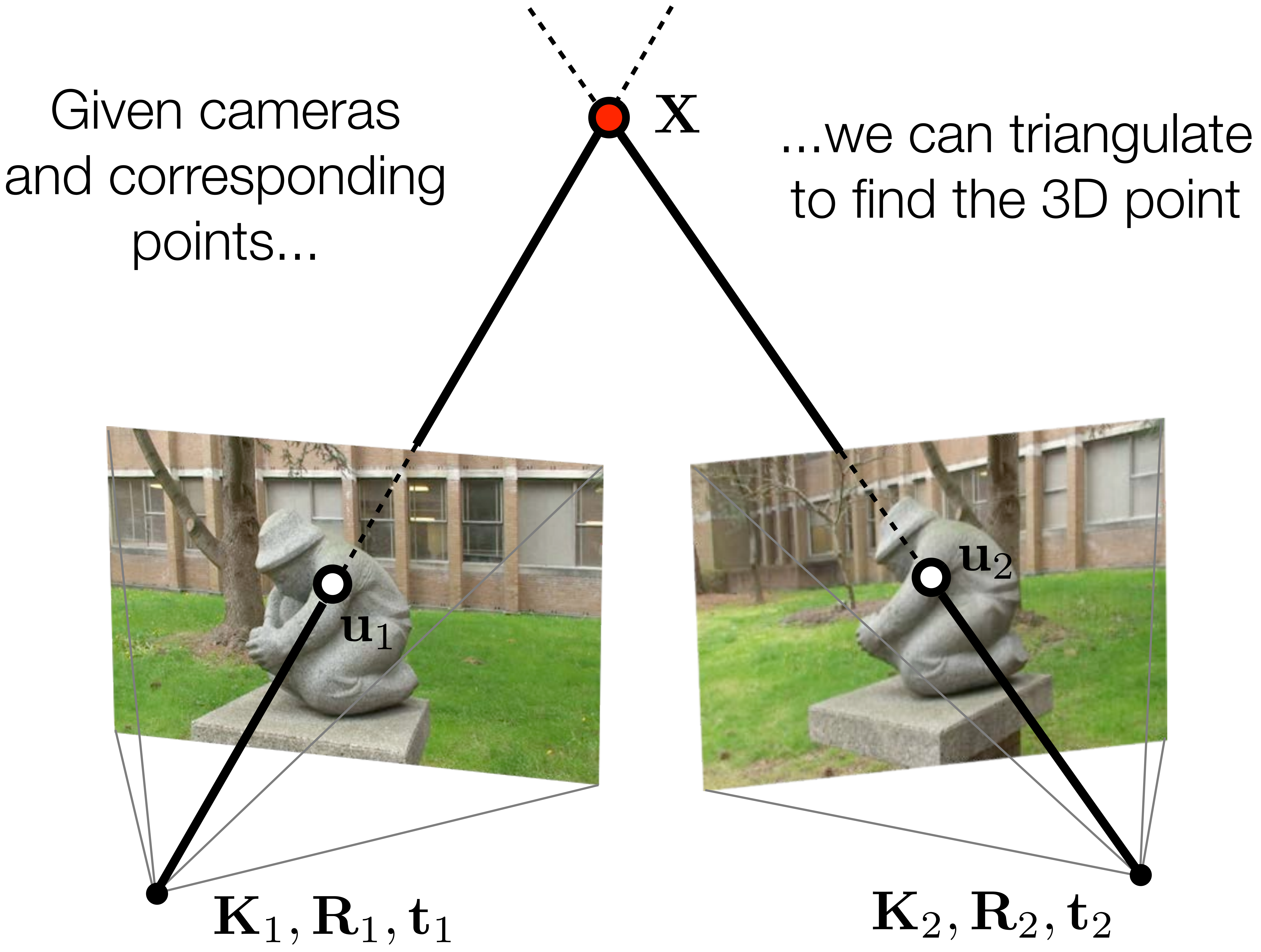
RANSAC for **Epipolar Geometry**



Raw feature matches (after ratio test filtering)

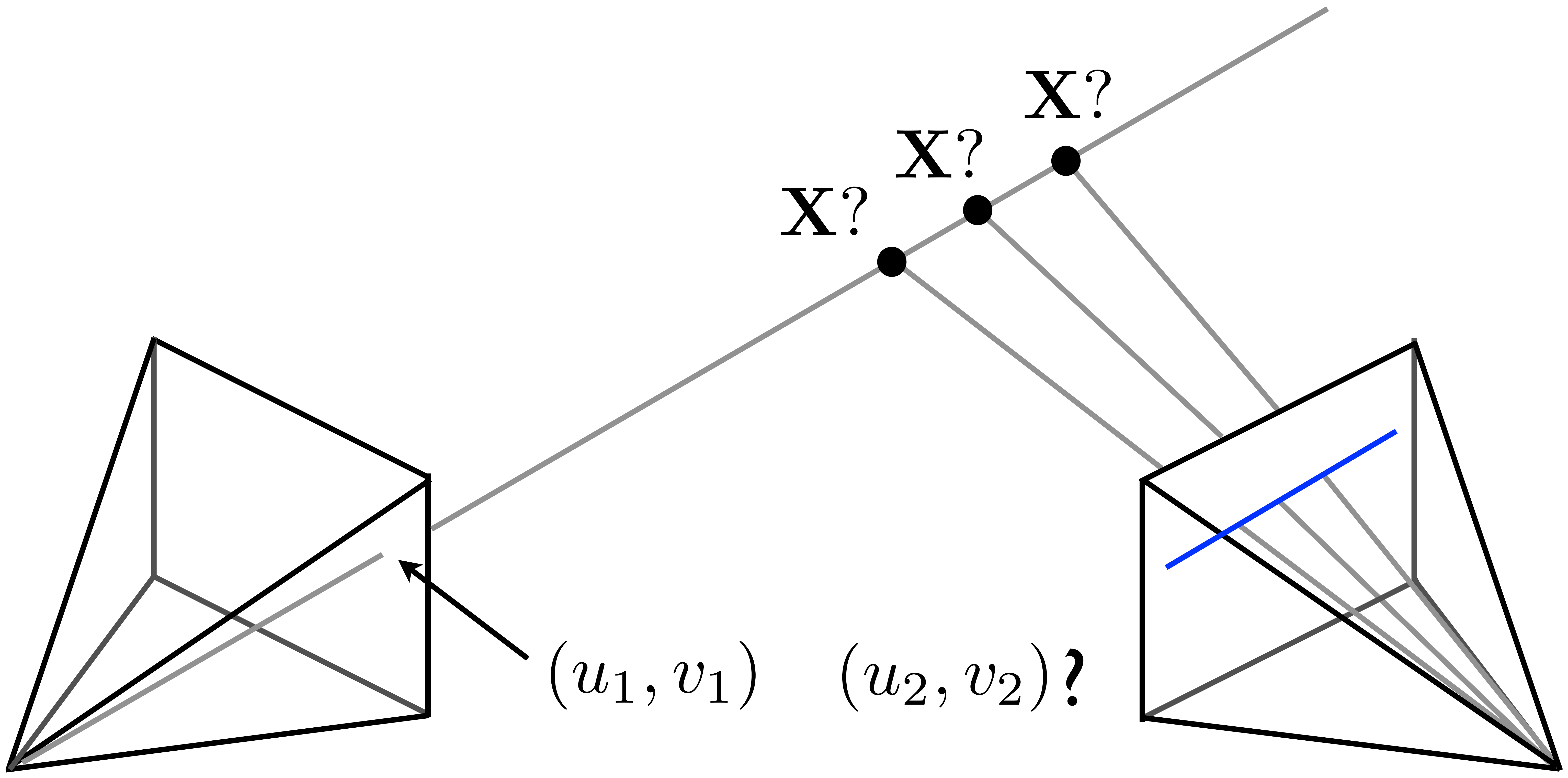
Solve for camera geometry and RANSAC

Triangulation



Going back to **Epipolar** Geometry

How do we find correspondences between two views?



A point in Image 1 must lie along the **line** in Image 2

2-view **Stereo**

Search over matches constrained to (epipolar) line

(reduces to 1d search)

Stereo Camera Configuration

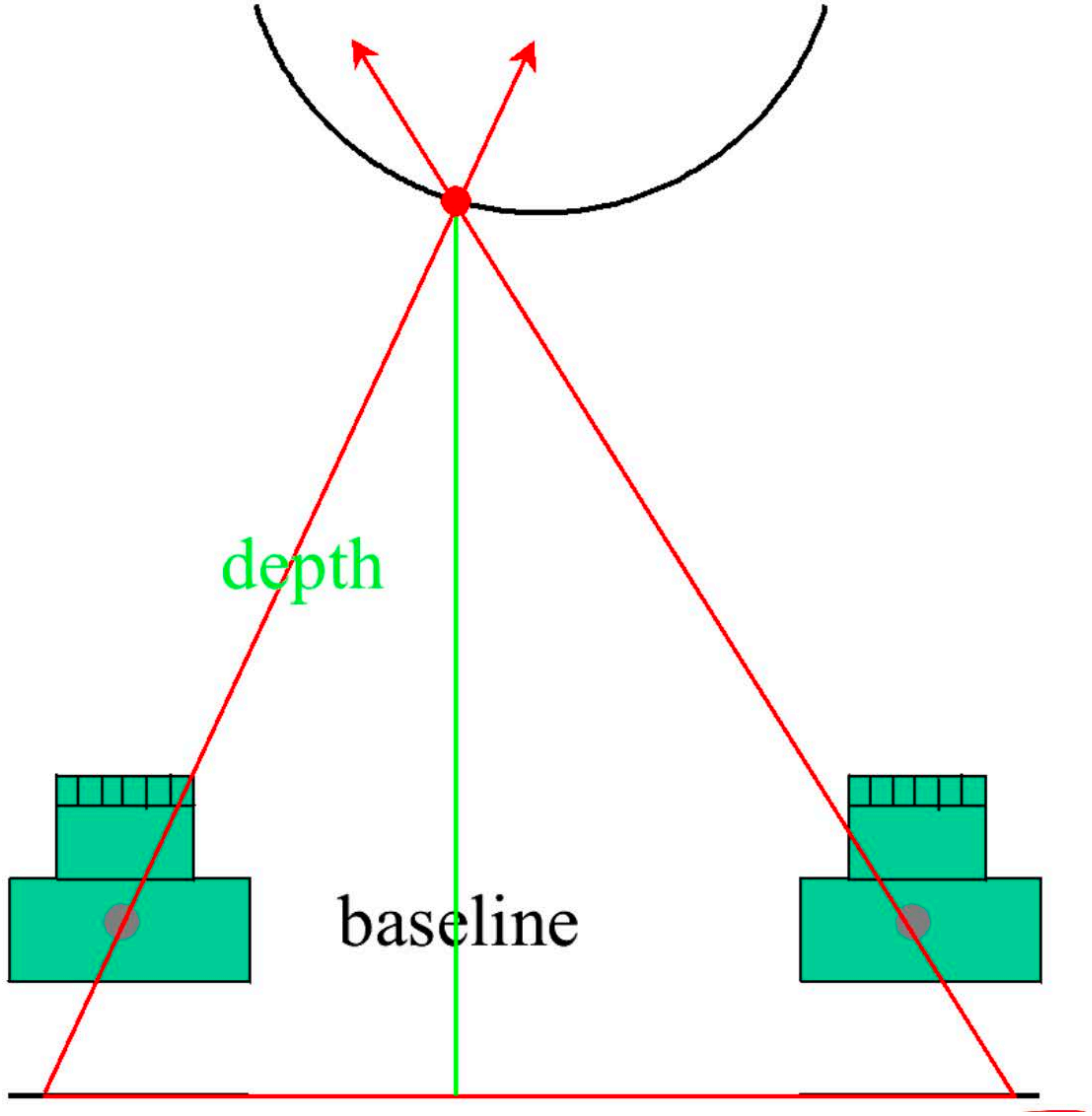
Humans and many stereo cameras have parallel optical axes

[J. Elson]

Axis Aligned **Stereo**

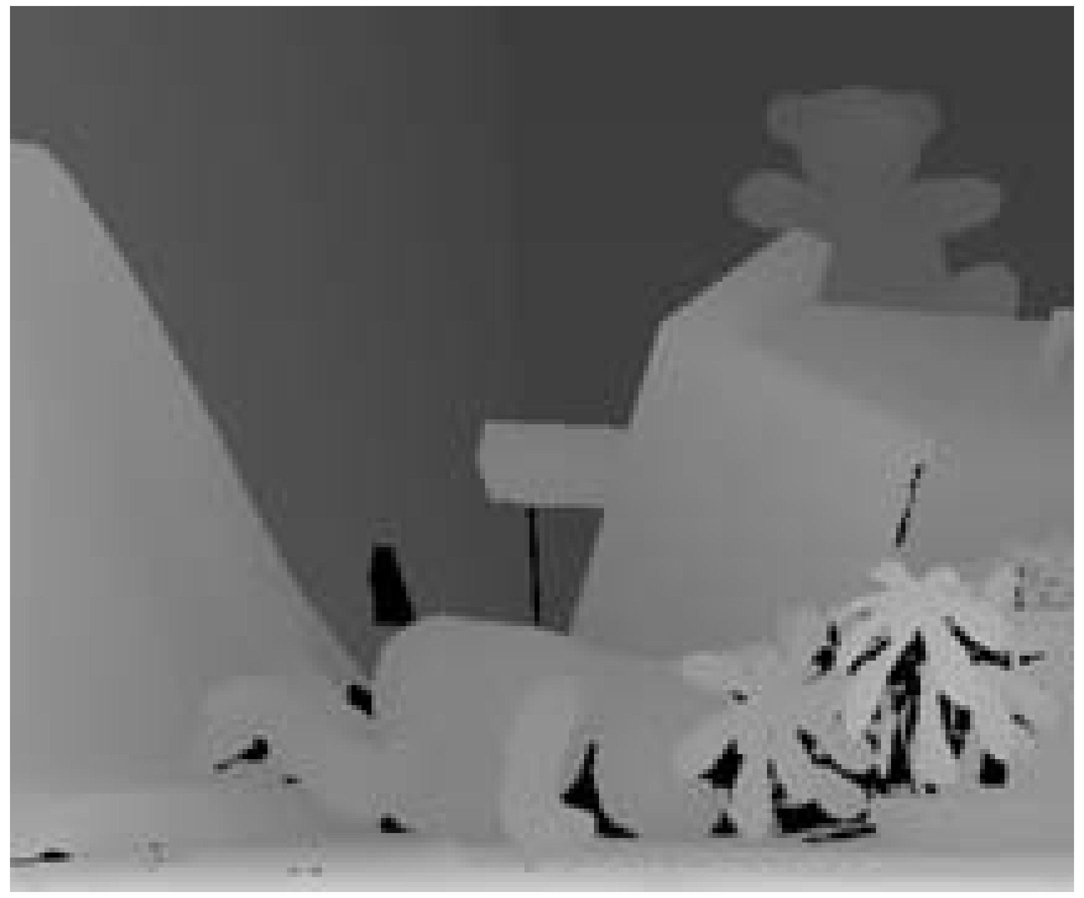
A common stereo configuration has camera optical axes aligned, with cameras related by a translation in the x direction

14.2



Stereo Matching in Rectified Images

— In a standard stereo setup, where cameras are related by translation in the x direction, epipolar lines are horizontal

— Stereo algorithms search along scanlines for matches

— Distance along the scanline (difference in x coordinate) for a corresponding feature is called **disparity**

Stereo Matching in Rectified Images (Left)

[D. Scharstein]

Stereo Matching in Rectified Images (Right)

[D. Scharstein]

Stereo Matching in Rectified Images (Right)

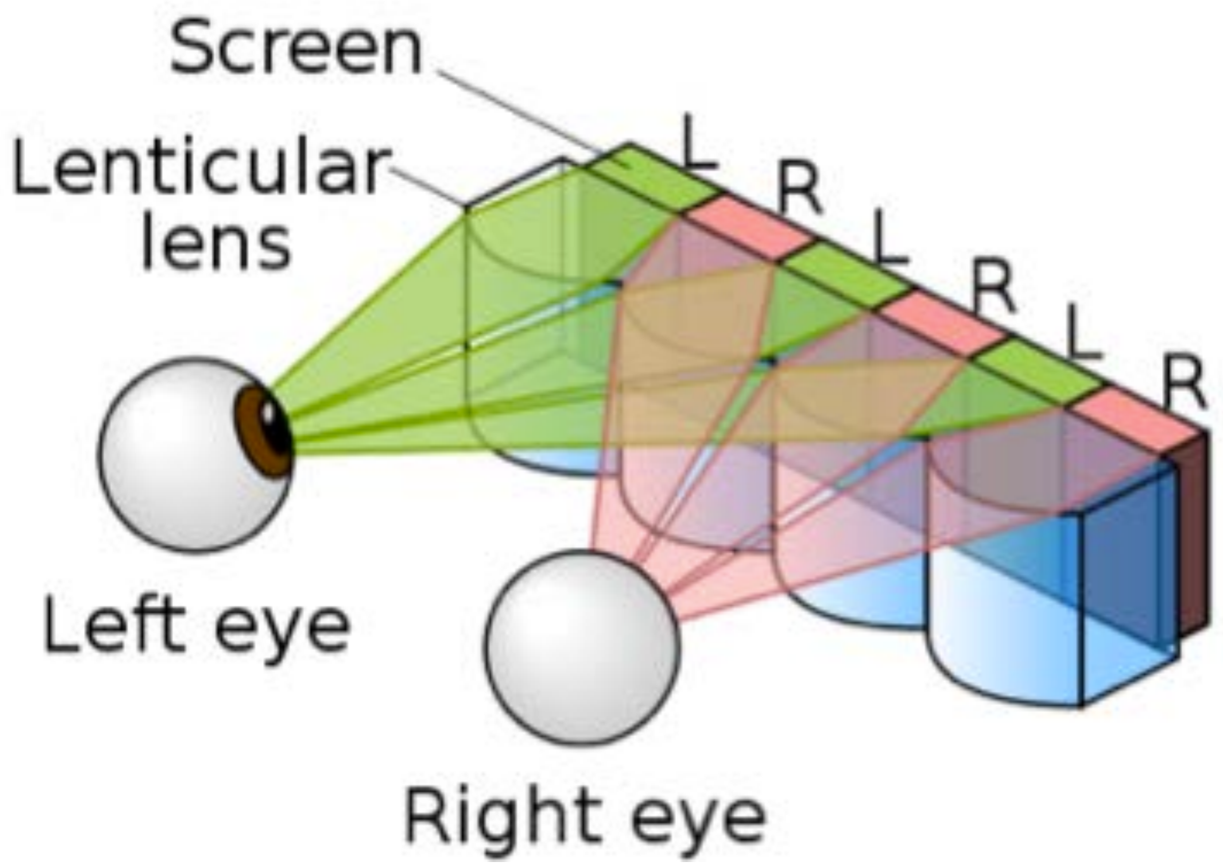
[D. Scharstein]

Anaglyph

Stereo pair with images encoded in different color channels

Stereo Displays

Field sequential (shutter) glasses transmit alternate left/right image at 120Hz

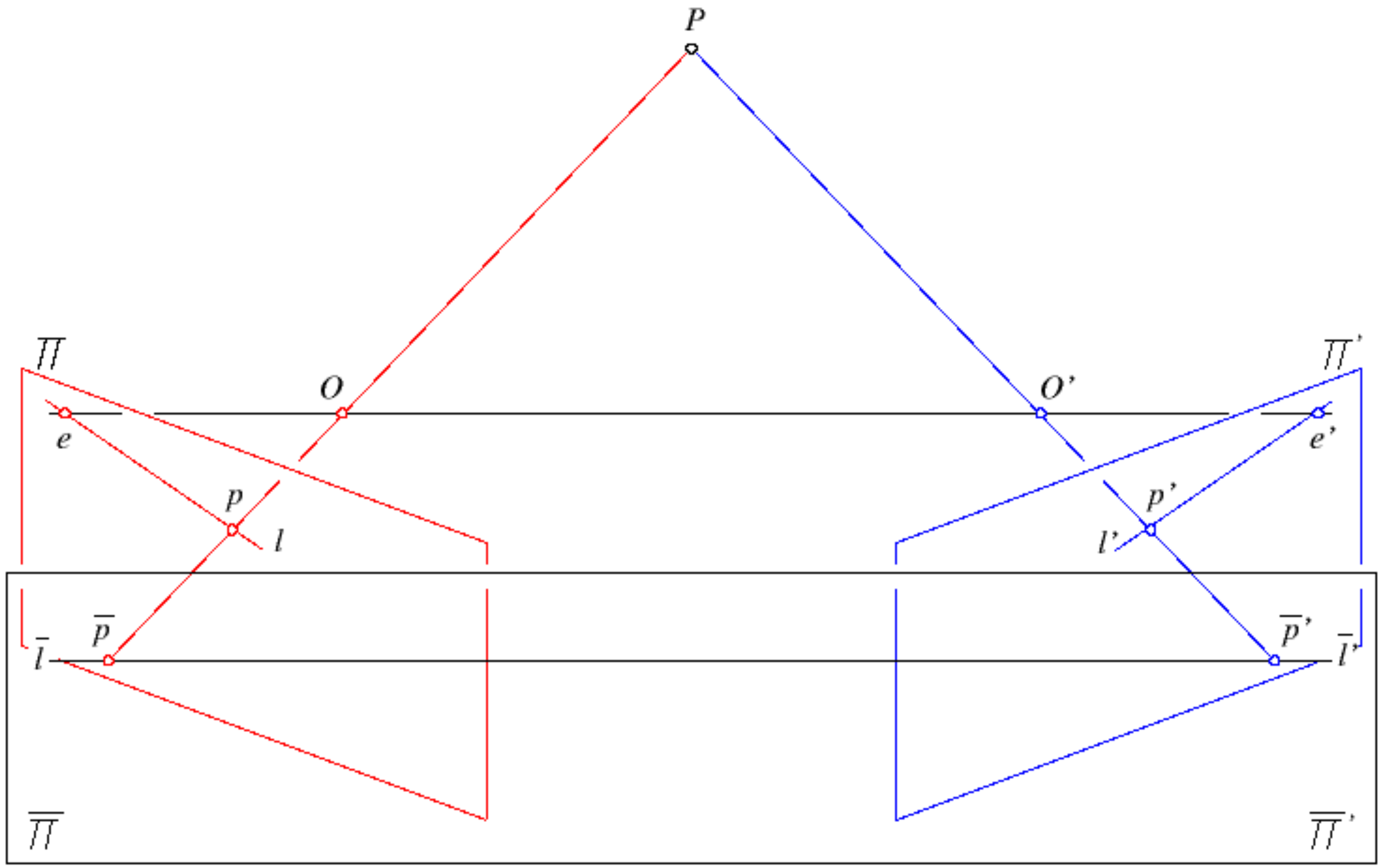
Lenticular lenses send different images directly to each eye, without the need for glasses

Stereo Displays

VR headsets send L/R images directly to each eye

[Google Cardboard]

Rectified Stereo Pair

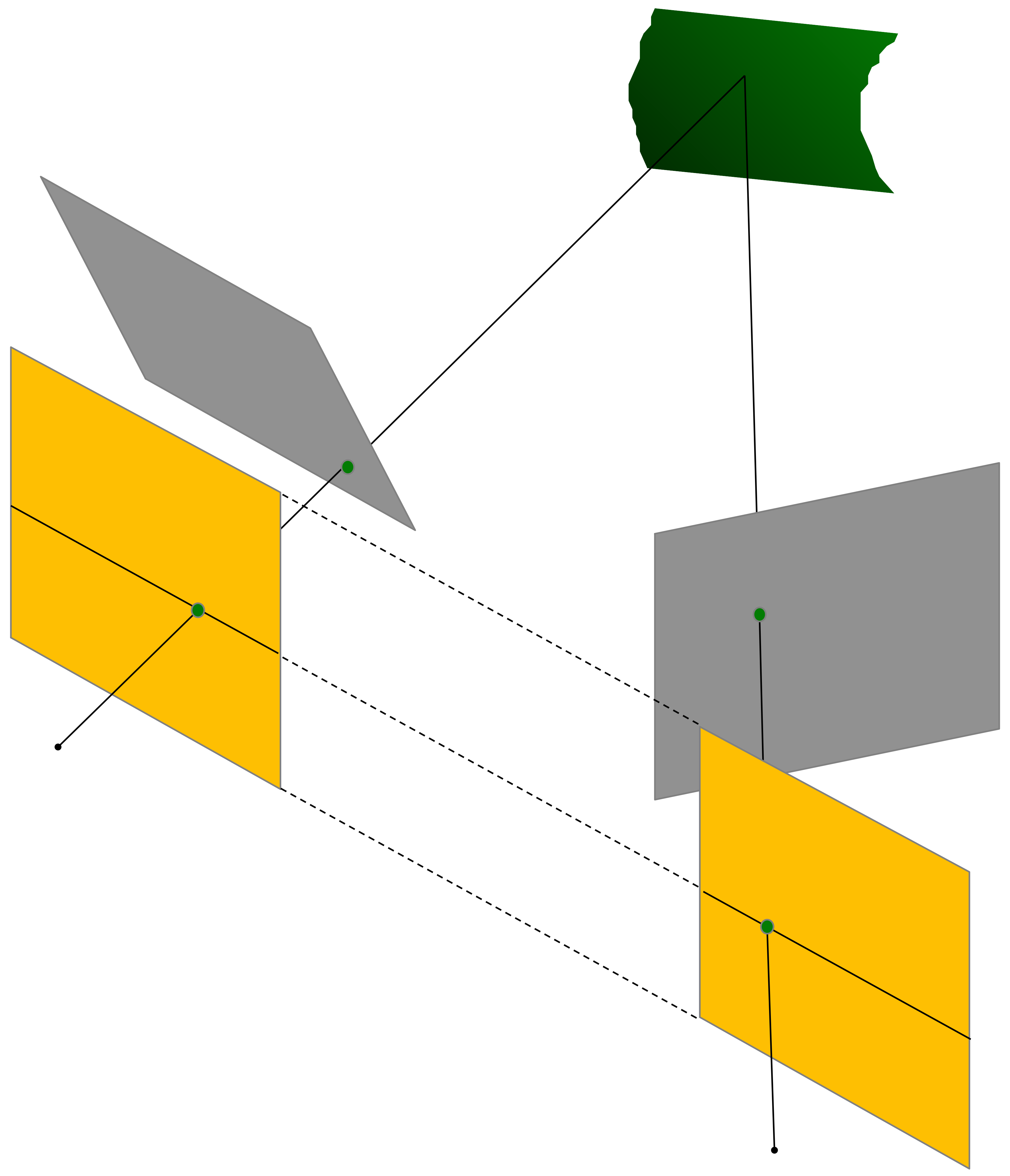


Any two camera views that overlap can be **rectified** so that epipolar lines correspond to scan lines (no special conditions must hold)

Rectified Stereo Pair

Reproject image planes onto a common plane parallel to the line between camera centers

Need two homographies (3x3 transform), one for each input image reprojection

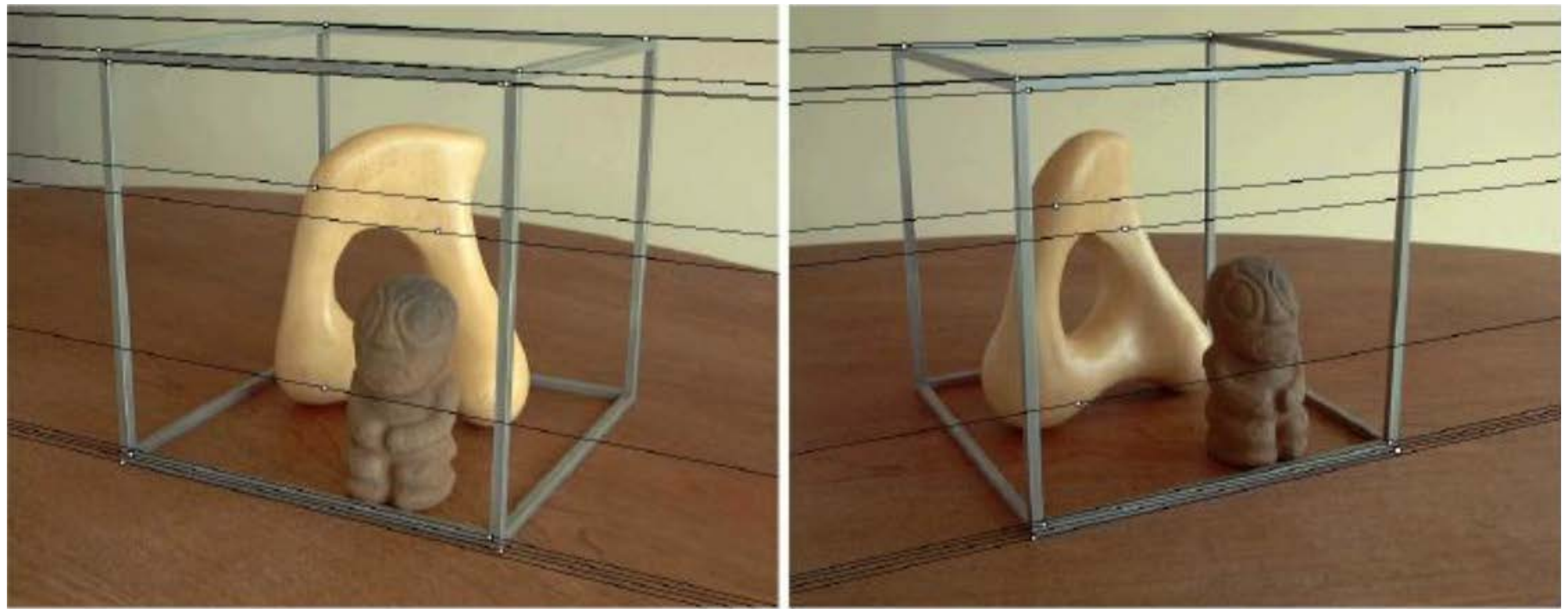


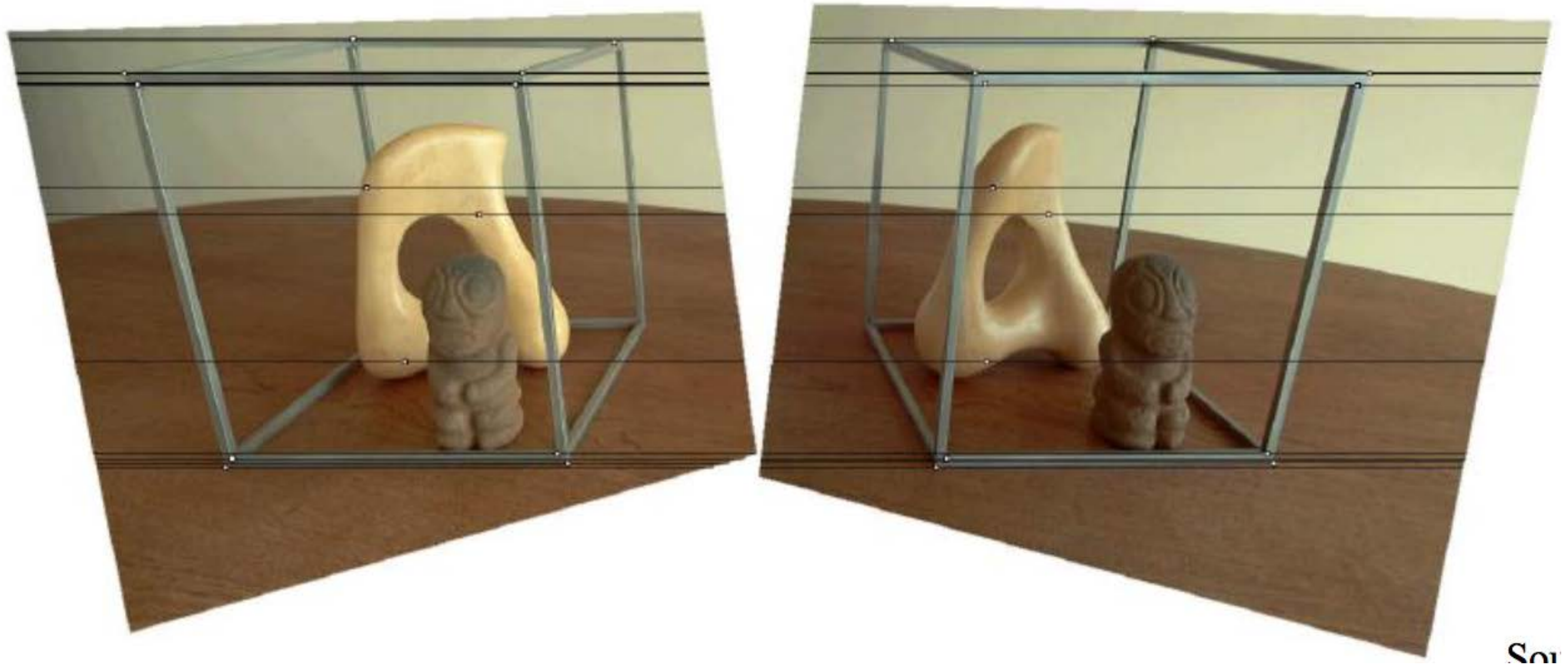
C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision.Computer Vision and Pattern Recognition, 1999.

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Rectified Stereo Pair: Example

Before Rectification





After Rectification

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

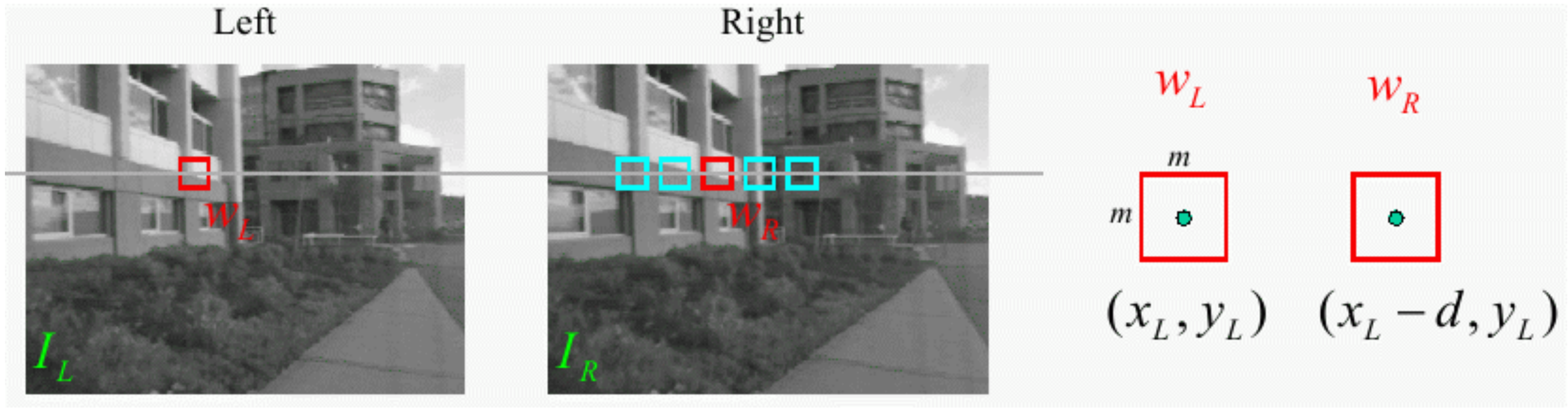
Stereo Matching in Rectified Images

— In a standard stereo setup, where cameras are related by translation in the x direction, epipolar lines are horizontal

— Stereo algorithms search along scanlines for matche

— Distance along the scanline (difference in x coordinate) for a corresponding feature is called **disparity**

Matching along a Scanline



$\mathbf{w}_L$ and $\mathbf{w}_R$ are corresponding $m \times m$ windows of pixels

Define a distance function between image patches, e.g.,

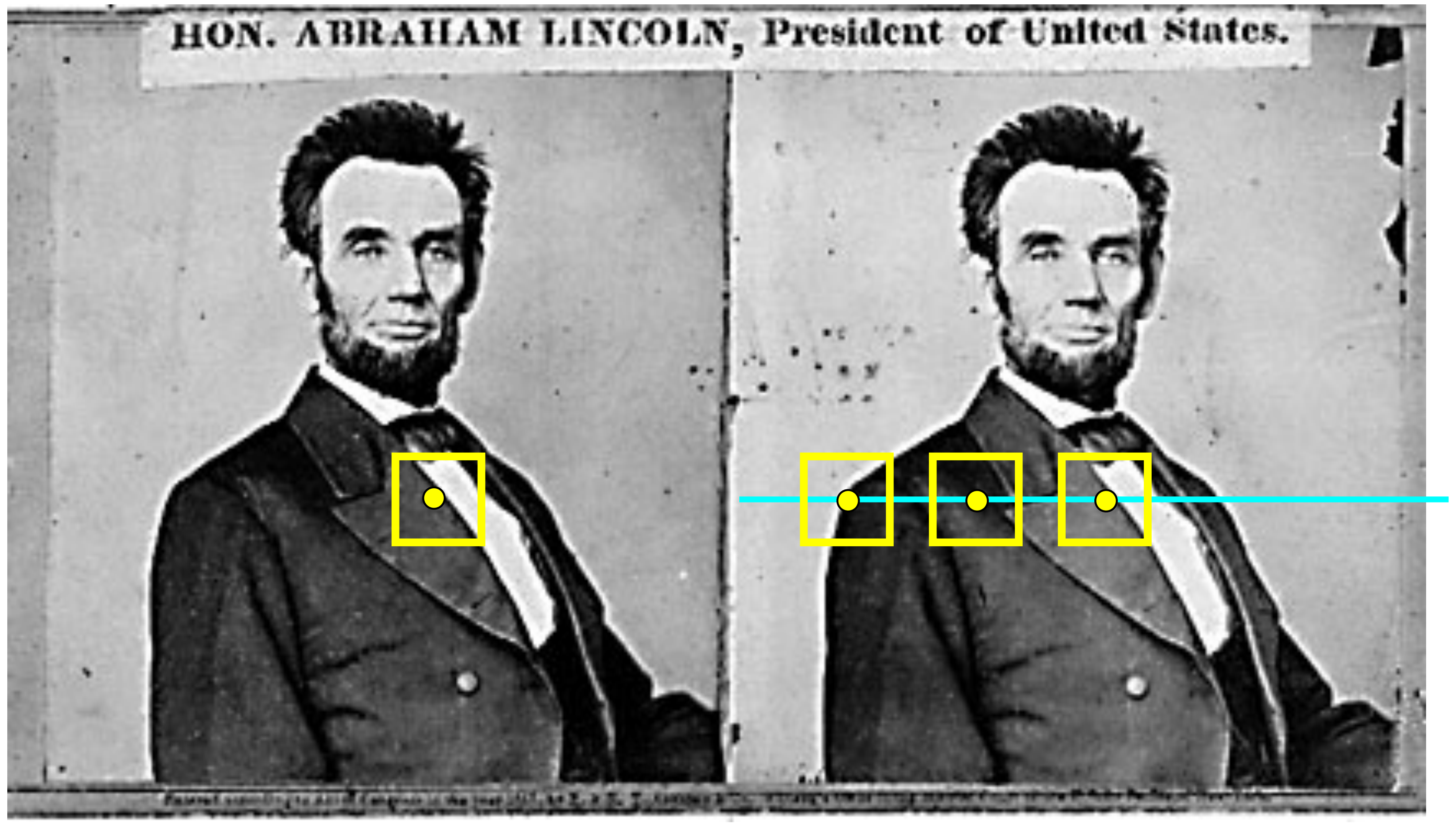
$$\text{SSD} = ||\mathbf{w}_L - \mathbf{w}_R(d)||^2$$

$$\text{correlation} = \mathbf{w}_L \cdot \mathbf{w}_R(d) = \cos\theta$$

Matching along a Scanline

Best match is at minimum of SSD function along a scanline

(simple) **Stereo** Algorithm



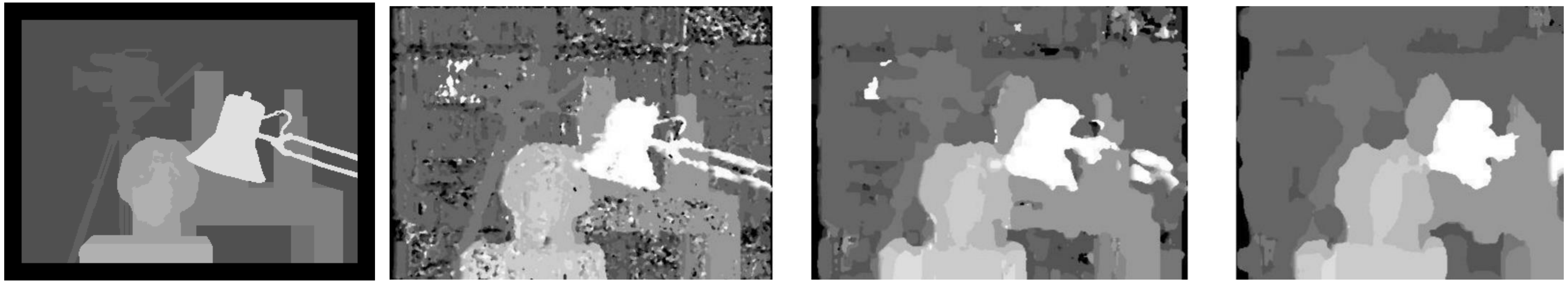
1. Rectify images
   (make epipolar lines horizontal)
2. For each pixel in image 1
   a. Search along epipolar line in image 2
   b. Find best match and record offset = disparity
   c. Compute depth from disparity

$$Z = f\frac{\Delta X}{disp}$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Effect of **Window Size**

Larger windows → smoothed result



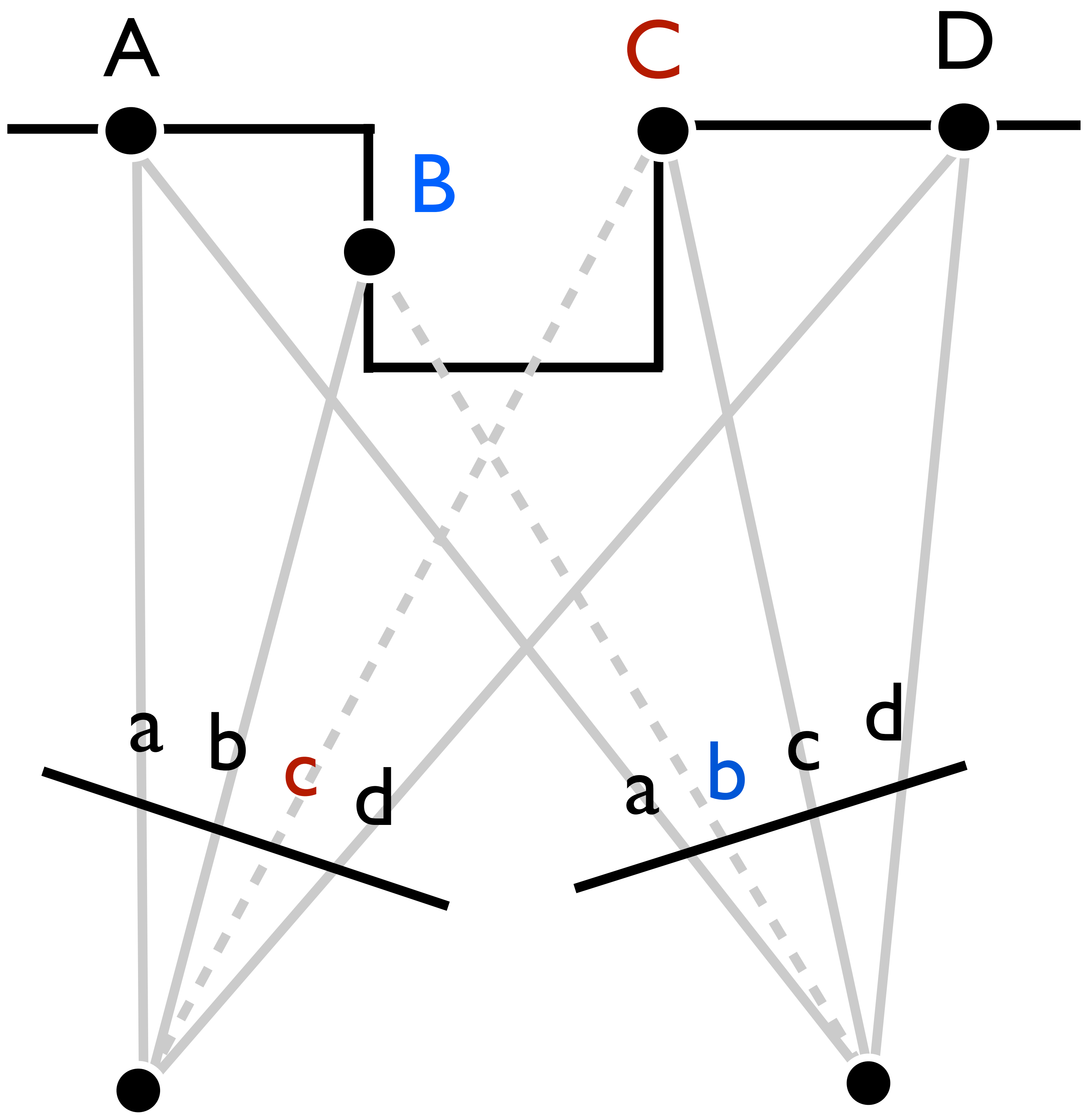
W=3 W=11 W=25

Smaller window
- \+ More detail
- \- More noise

Larger window
- \+ Smoother disparity maps
- \- Less detail
- \- Fails near boundaries

Occlusions

Sometimes a point in image 1 does not appear in image 2, or vice-versa (this is called an **occlusion**)



- Occlusions cause gaps in the stereo reconstruction
- + Matching is difficult nearby as aggregation windows often overlap the occluded region

Edge Aware Stereo

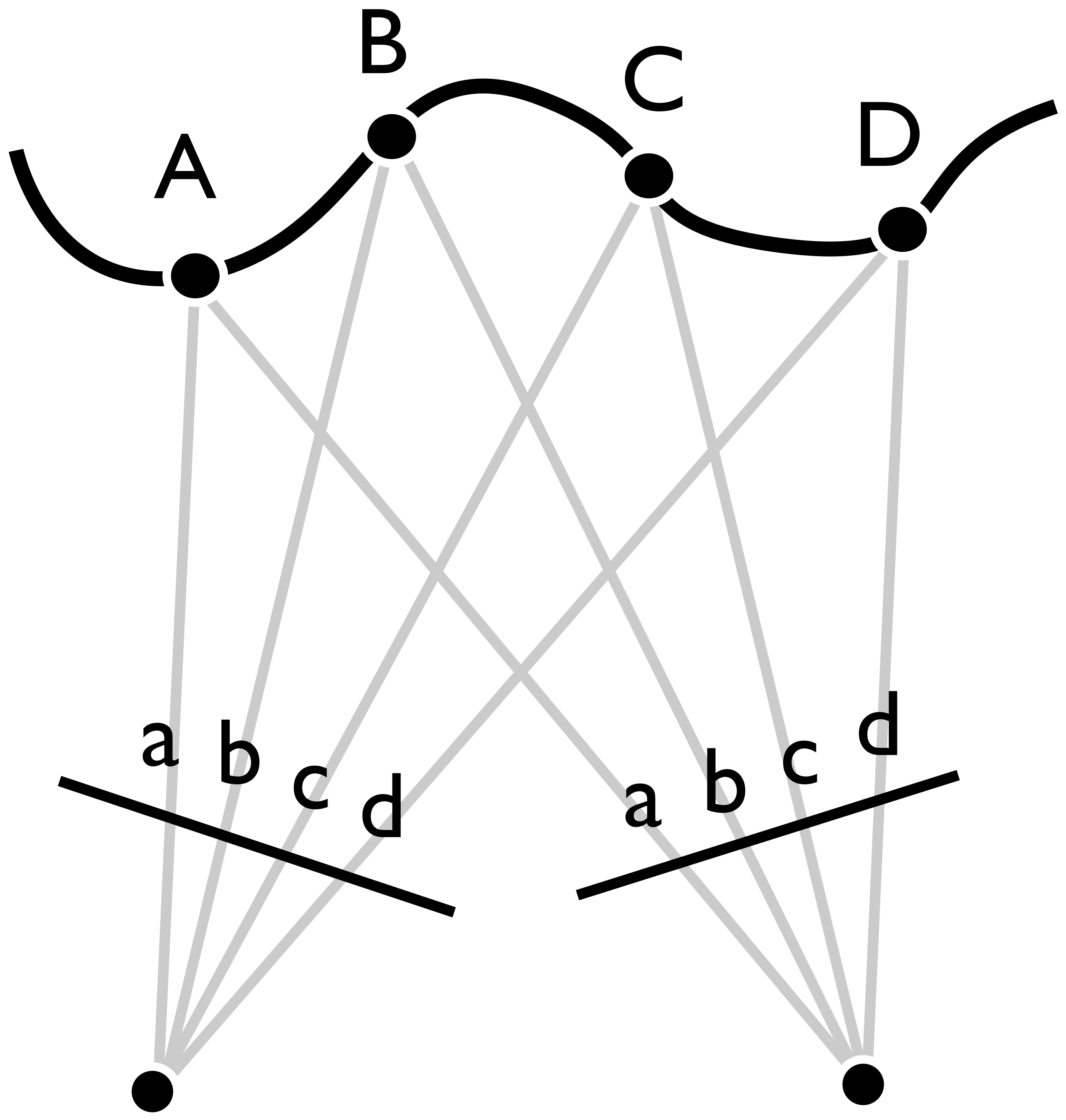
Occlusions and depth discontinuities cause problems for stereo matching, as aggregation windows overlap multiple depths

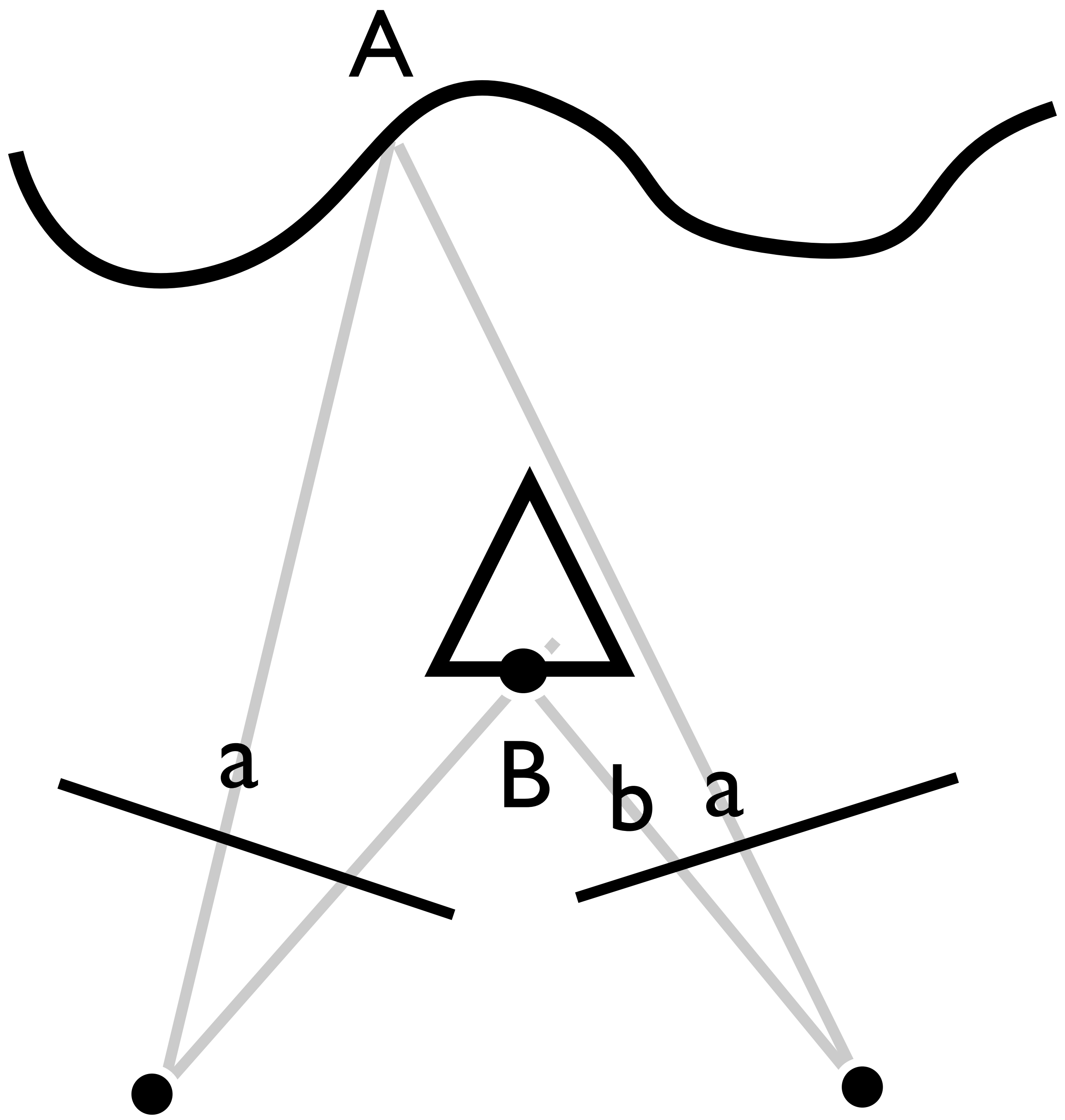
 

- Segmentation-based stereo approaches aim to solve this by trying to guess the depth edges (e.g., joint segmentation and depth estimation [Taguchi et al 2008])

Ordering Constraint

If point B is to the right of point A in image 1, the same is *usually* true in image 2

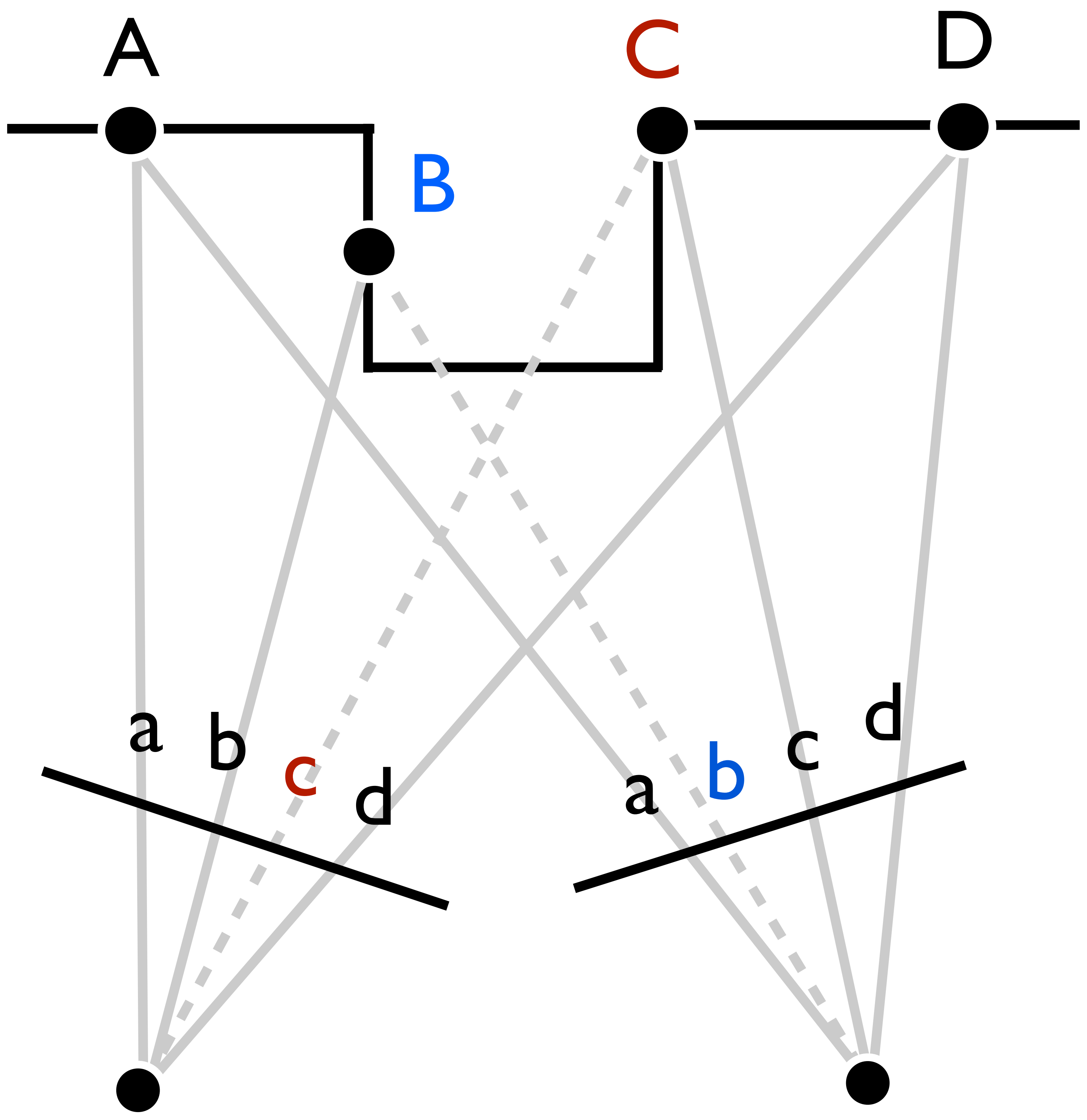




Not always, e.g., if an object is wholly within the ray triangle generated by A

Occlusions + Ordering

Note that the ordering constraint is still maintained in the presence of occlusions (unless there is an object off surface as in the previous slide)



Stereo Cost Functions

- Energy function for stereo matching based on disparity d(x,y)
- Sum of data and smoothness terms

$$E(d) = E_d(d) + \lambda E_s(d)$$

- Data term is cost of pixel x,y allocated disparity d (e.g., SSD)

$$E_d(d) = \sum_{(x,y)} C(x, y, d(x, y))$$

- Smoothness cost penalises disparity changes with robust $\rho(.)$

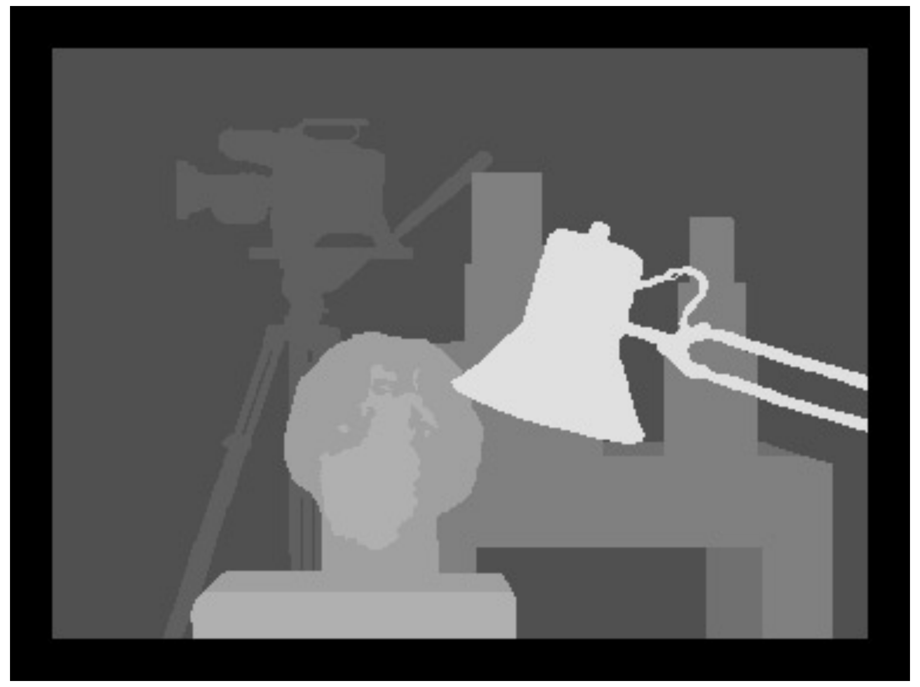
$$E_s(d) = \sum_{(x,y)} \rho(d(x,y) - d(x+1,y)) + \rho(d(x,y) - d(x,y+1))$$

- This is a Markov Random Field (MRF), which can be solved using techniques such as Graph Cuts

[Szeliski B5]

Stereo Comparison

- Global vs Scanline vs Local optimization



Ground truth

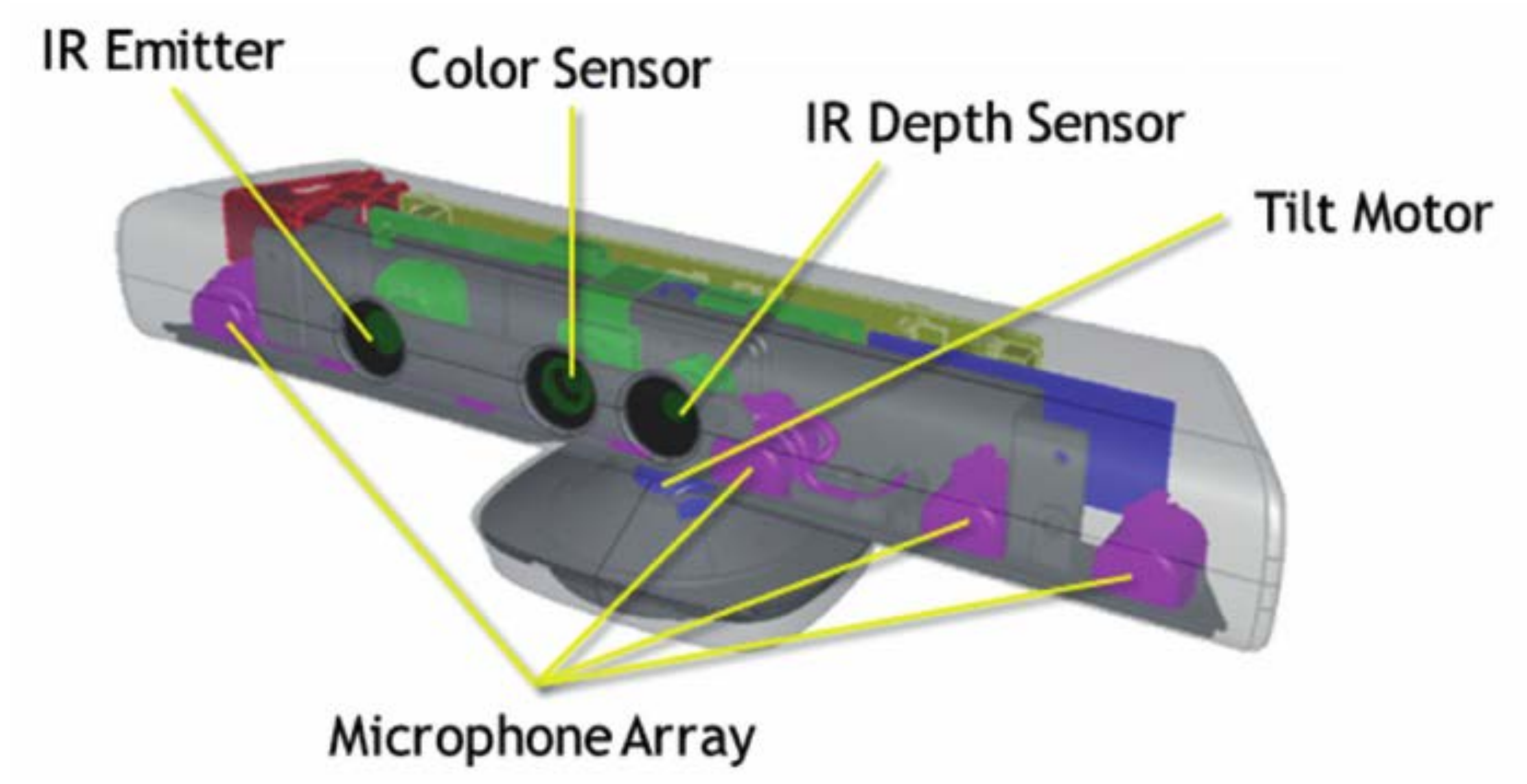
Graph Cuts
[Kolmogorov Zabih 2001]

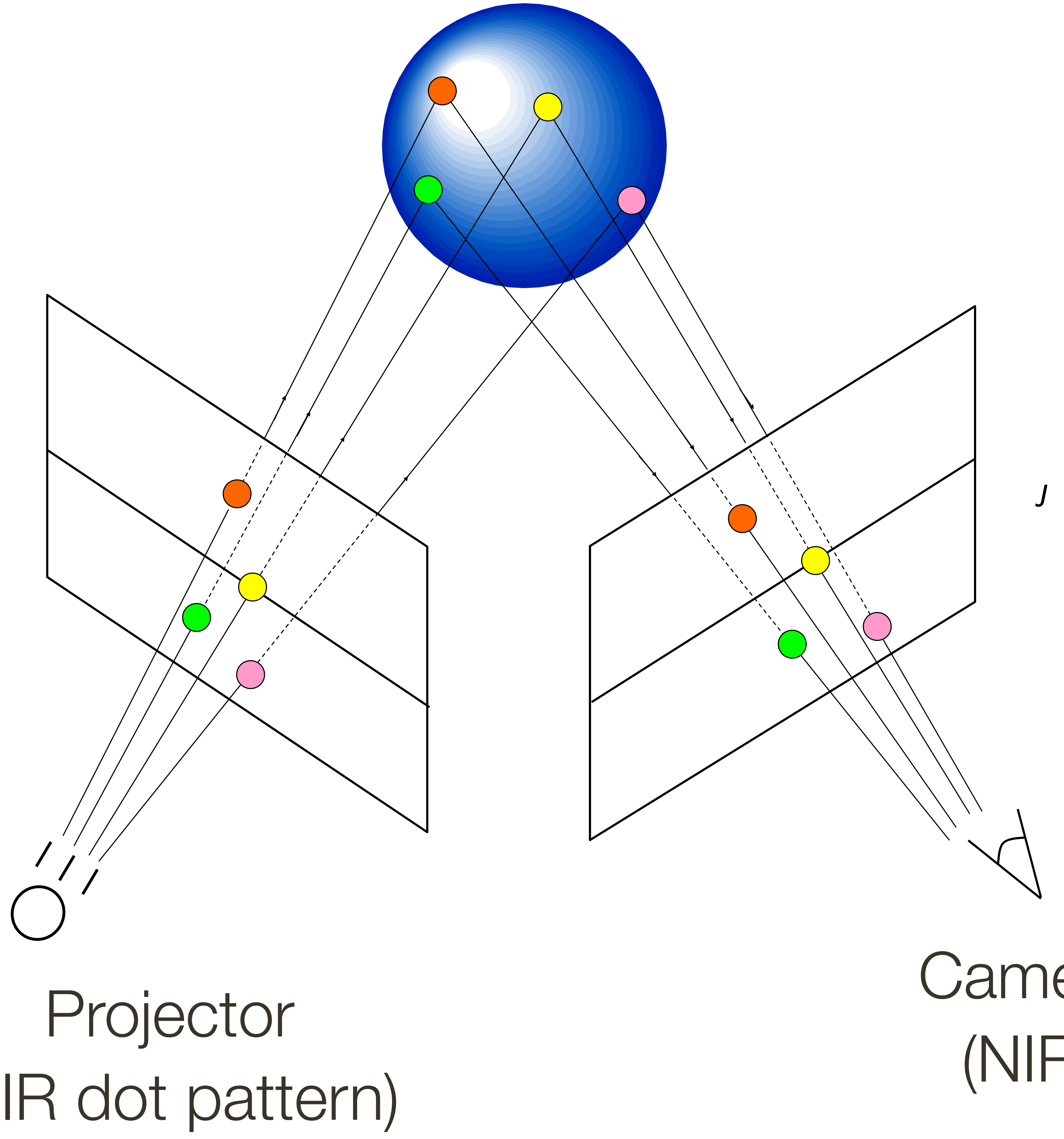
Dynamic Programming

SSD 21px aggregation

[Scharstein Szeliski 2002]

Application: Microsoft **Kinect** v1





Stereo Vision Summary

With two eyes, we acquire images of the world from slightly different viewpoints

We perceive **depth** based on **differences in the relative position of points** in the left image and in the right image

Stereo algorithms work by finding **matches** between points along corresponding lines in a second image, known as epipolar lines.

A point in one image projects to an **epipolar line** in a second image

In an axis-aligned / rectified stereo setup, matches are found along horizontal scanlines