



CPSC 425: Computer Vision

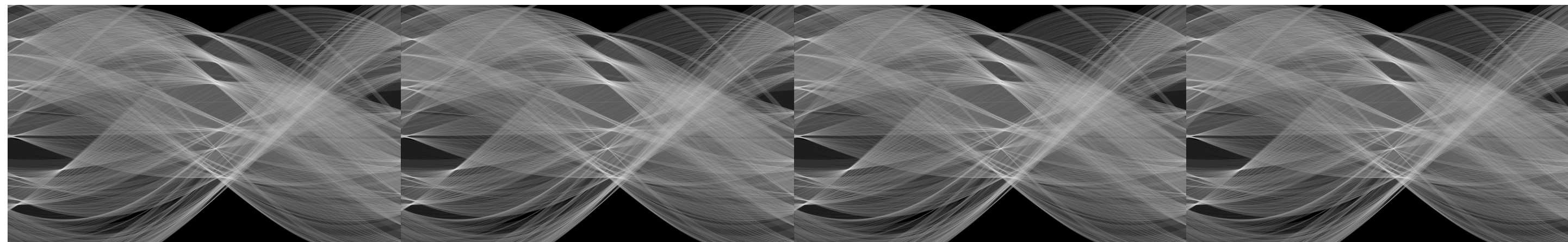


Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 18: Stereo (and finish Hough)

Menu for Today (March 12, 2019)

Topics:

- Hough Transform (examples)
- Stereo Vision

Readings:

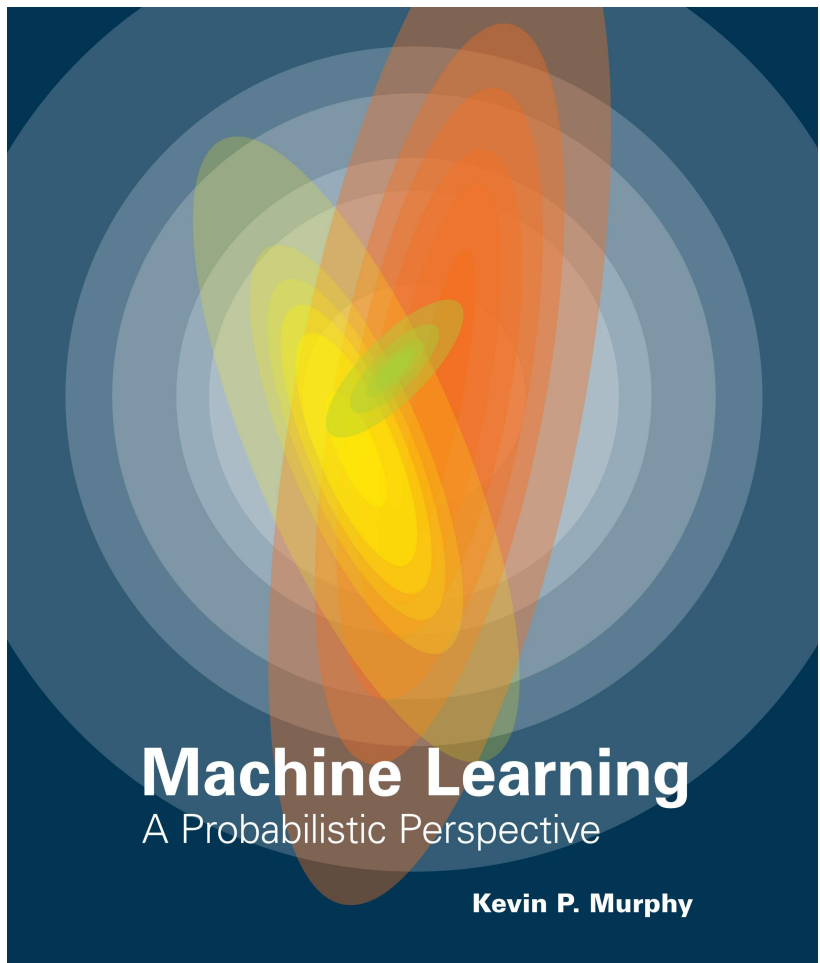
- **Today's** Lecture: Forsyth & Ponce (2nd ed.) 7.1.1, 7.2.1, 7.4, 7.6
- **Next** Lecture: Forsyth & Ponce (2nd ed.) 7.1.1, 7.2.1, 7.4, 7.6

Reminders:

- **Assignment 4** is due **March 19th**
- Tech staff is working on making exams available (there were some issues)

Today's “fun” Example: Im2Calories

ICCV 2015 paper by **Kevin Murphy**
(UBC's former faculty)



Coincidentally Kevin is also author of one of the most prominent ML books

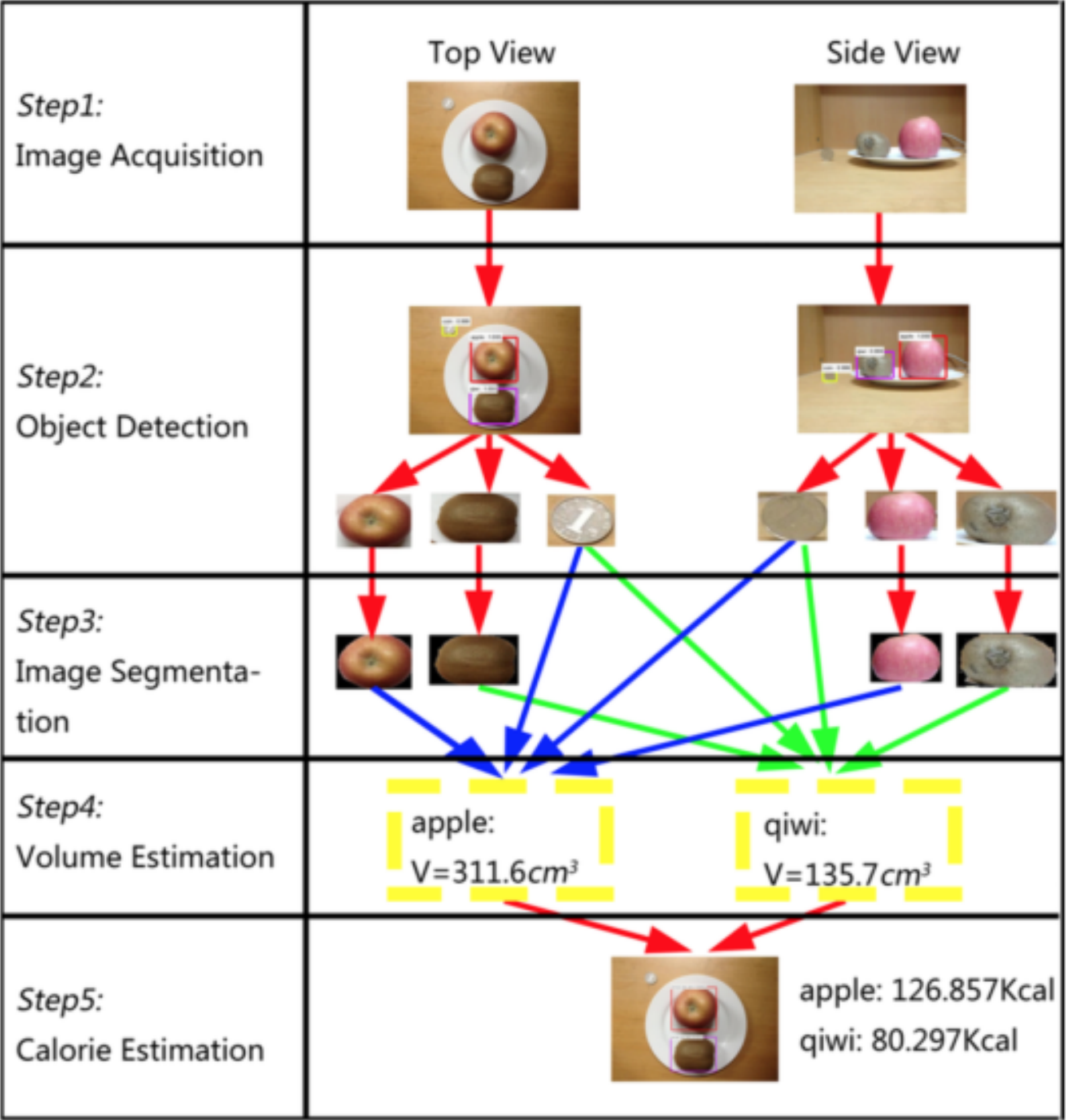


Figure 1: Calorie Estimation Flowchart

Today's “**fun**” Example: Im2Calories

Im2Calories: towards an automated mobile vision food diary

Austin Myers, Nick Johnston, Vivek Rathod, Anoop Korattikara, Alex Gorban Nathan Silberman, Sergio Guadarrama, George Papandreou, Jonathan Huang, Kevin Murphy amyers@umd.edu, (nickj, rathodv, kbanoop, gorban)@google.com (nsilberman, sguada, gpapan, jonathanhuang, kpmurphy)@google.com

Today's “**fun**” Example: Im2Calories

Fun **on-line demo**: <http://www.caloriemama.ai/api>

Lecture 17: Re-cap

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

Lecture 17: Re-cap

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

Lecture 17: Re-cap

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

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

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

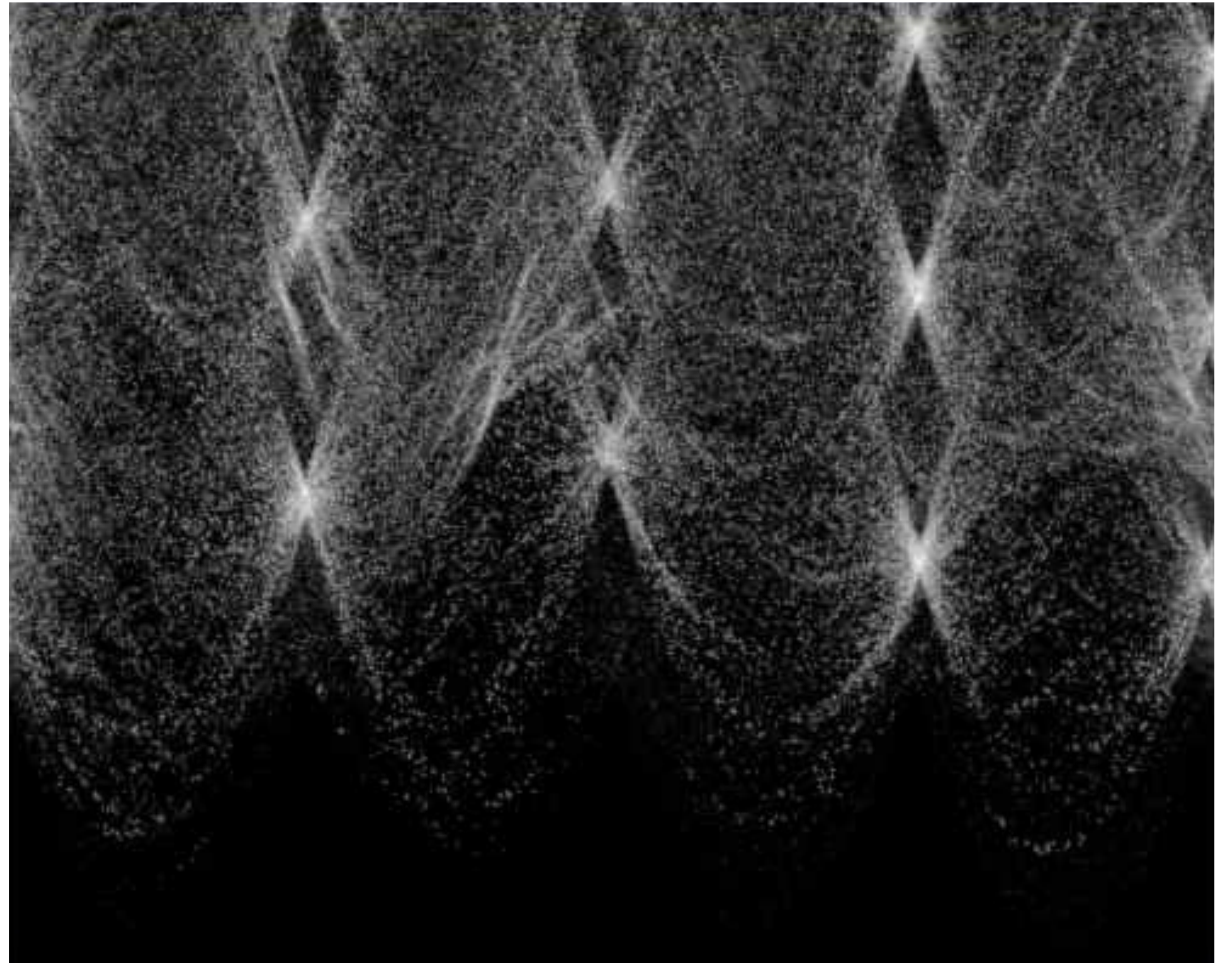
— Different choices of θ, r give different lines

— 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

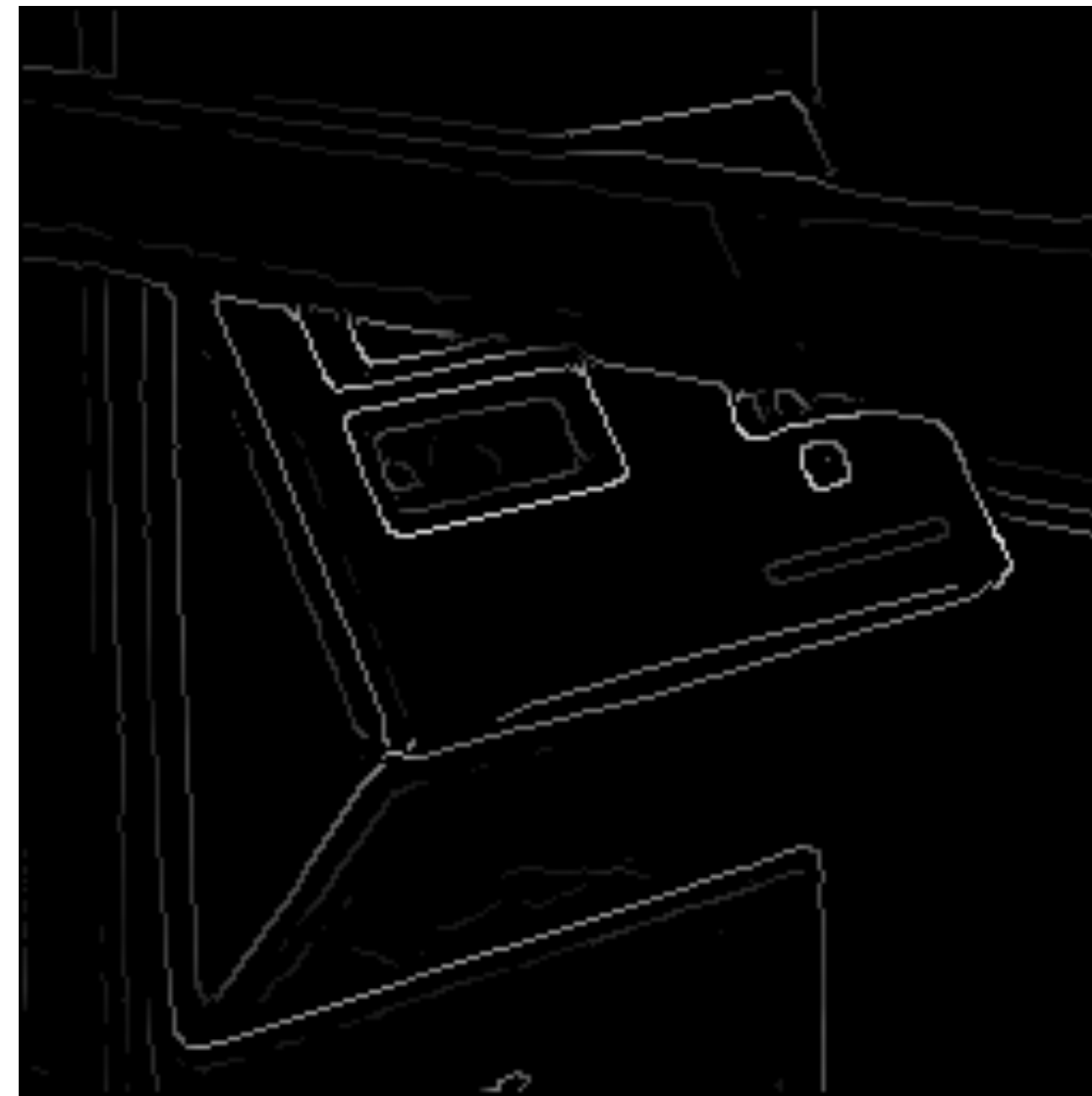
Lecture 17: Re-cap



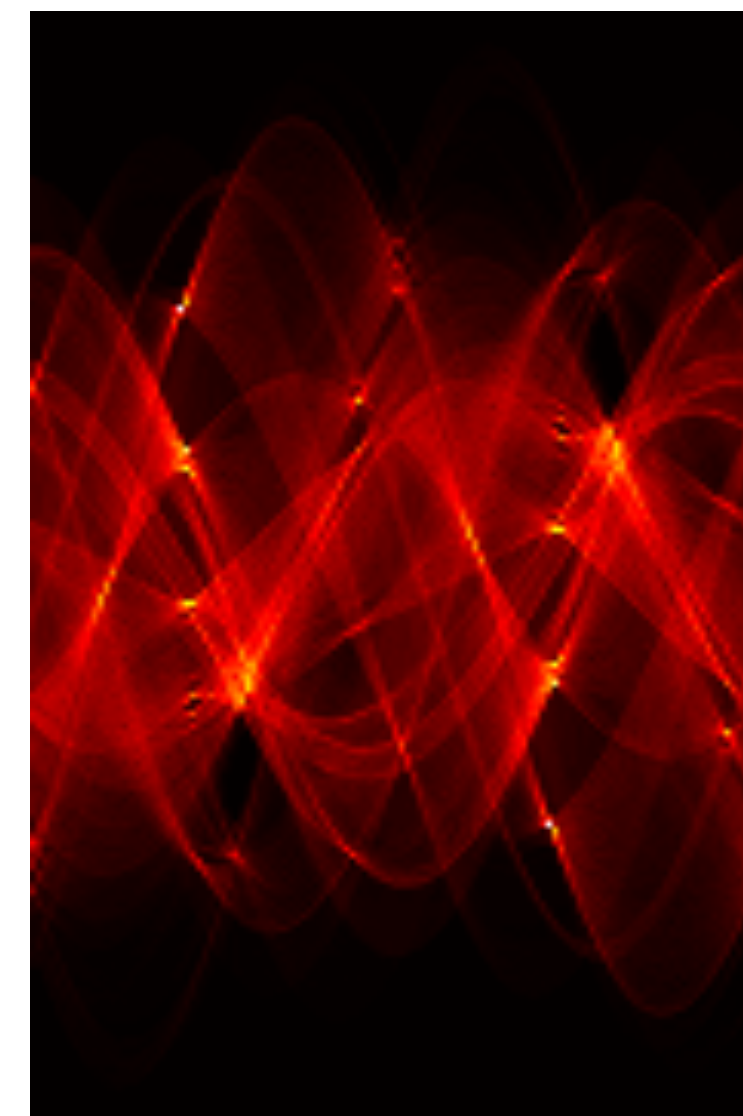
Lecture 17: Re-cap



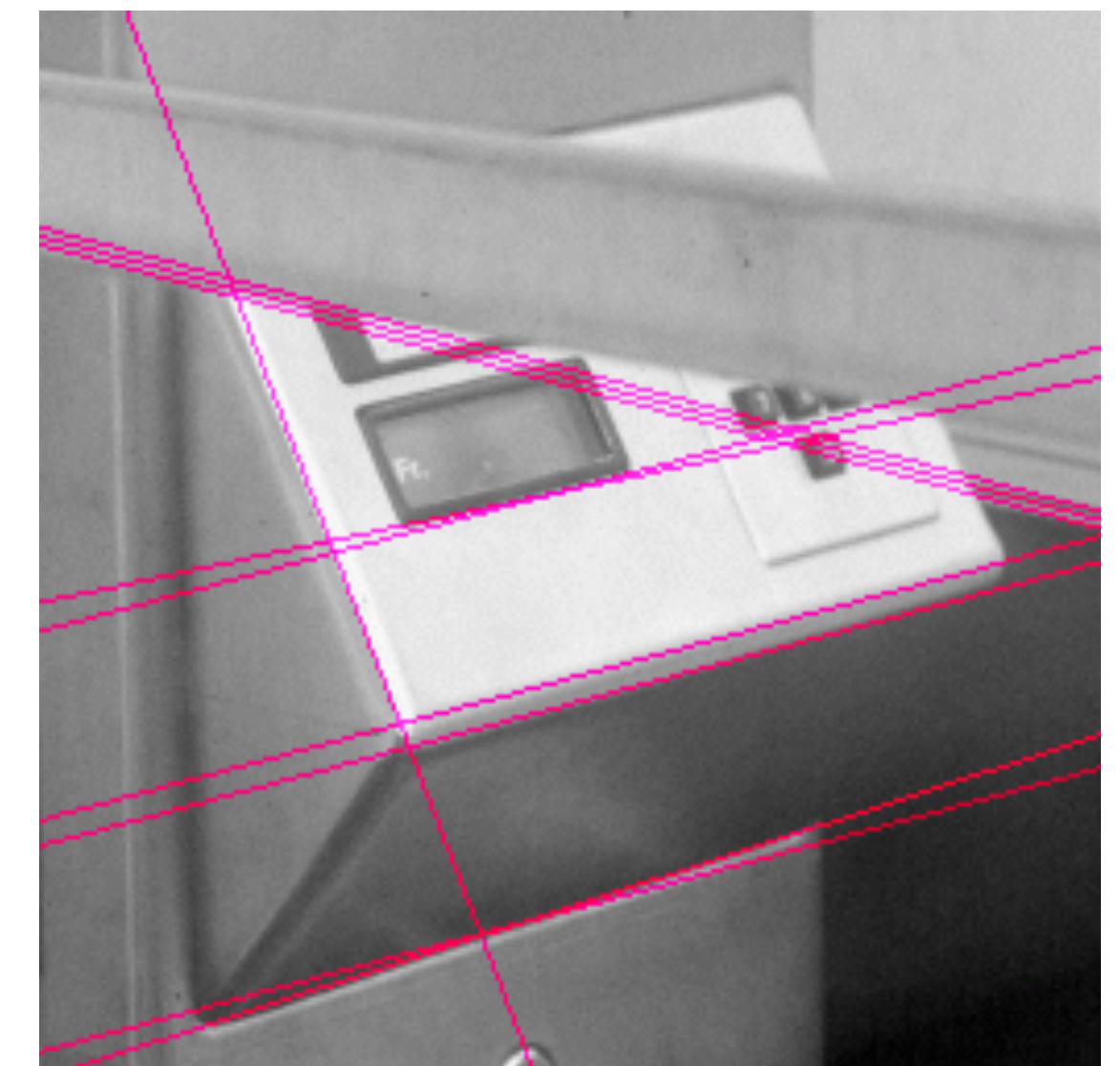
Original



Edges



Parameter
space



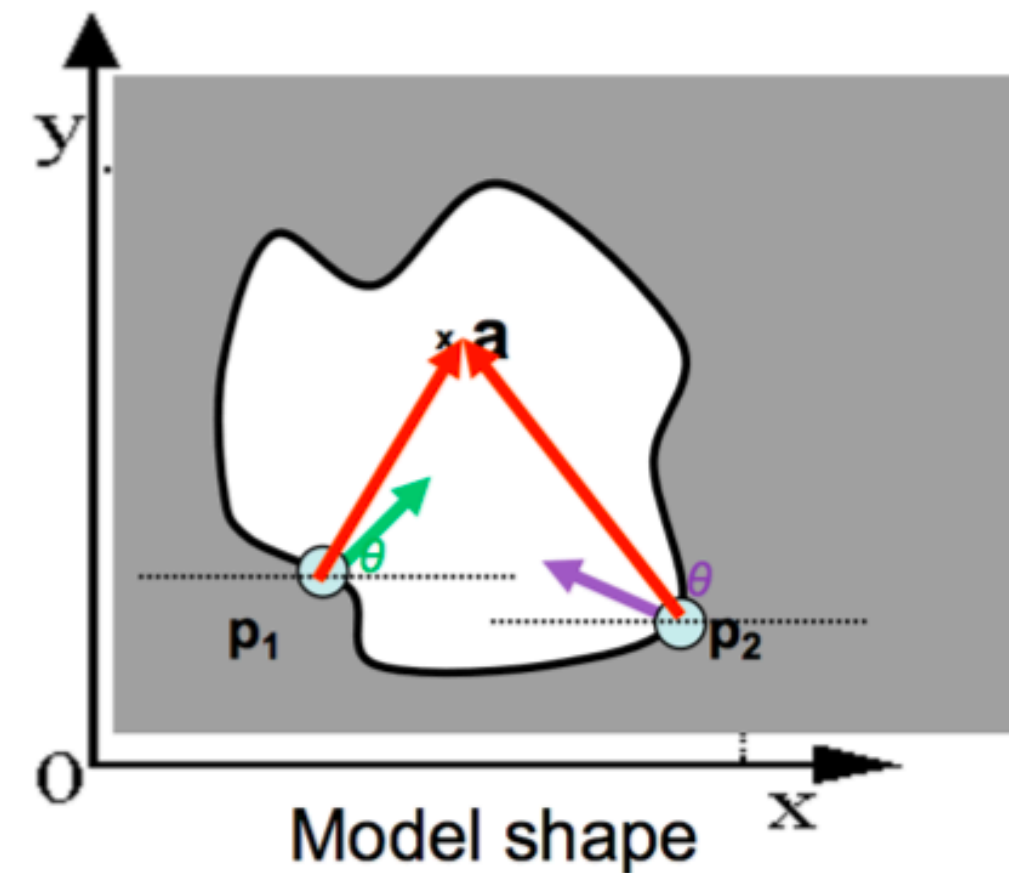
Hough Lines

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:

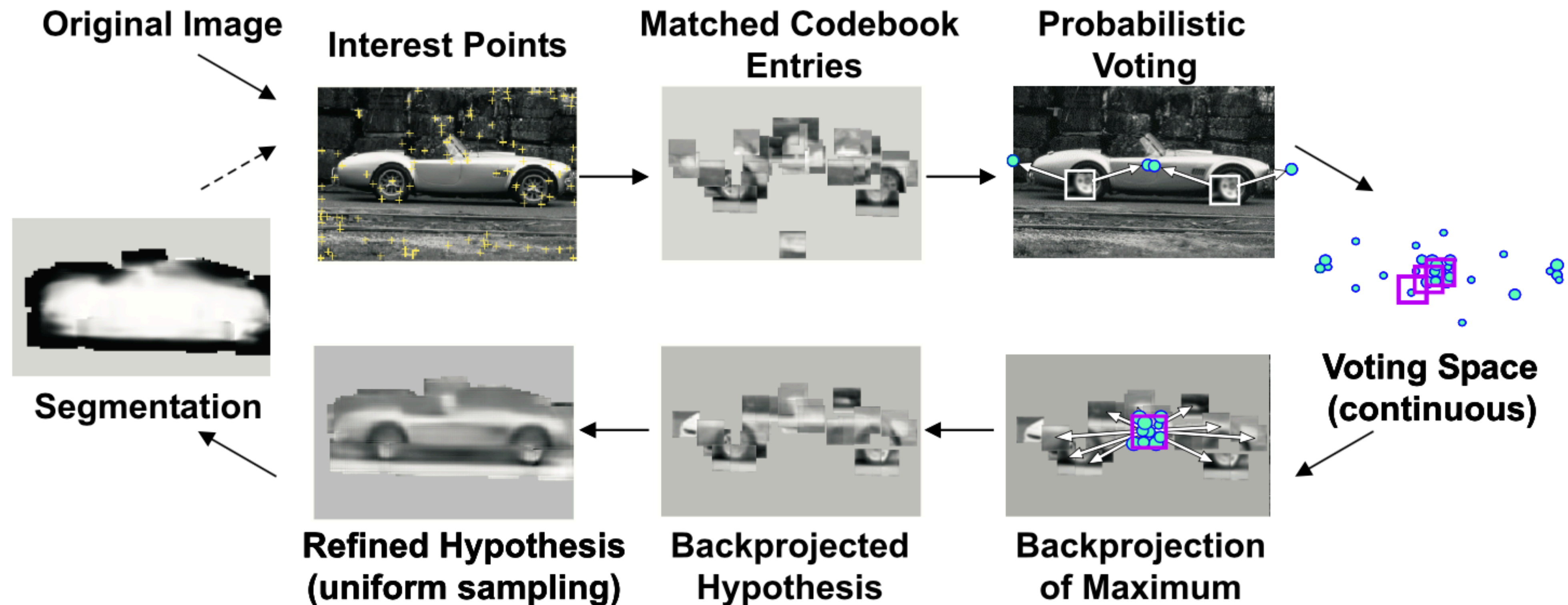
At each boundary point, compute displacement vector: $\mathbf{r} = \mathbf{a} - \mathbf{p}_i$.

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 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 packs (back-project)

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



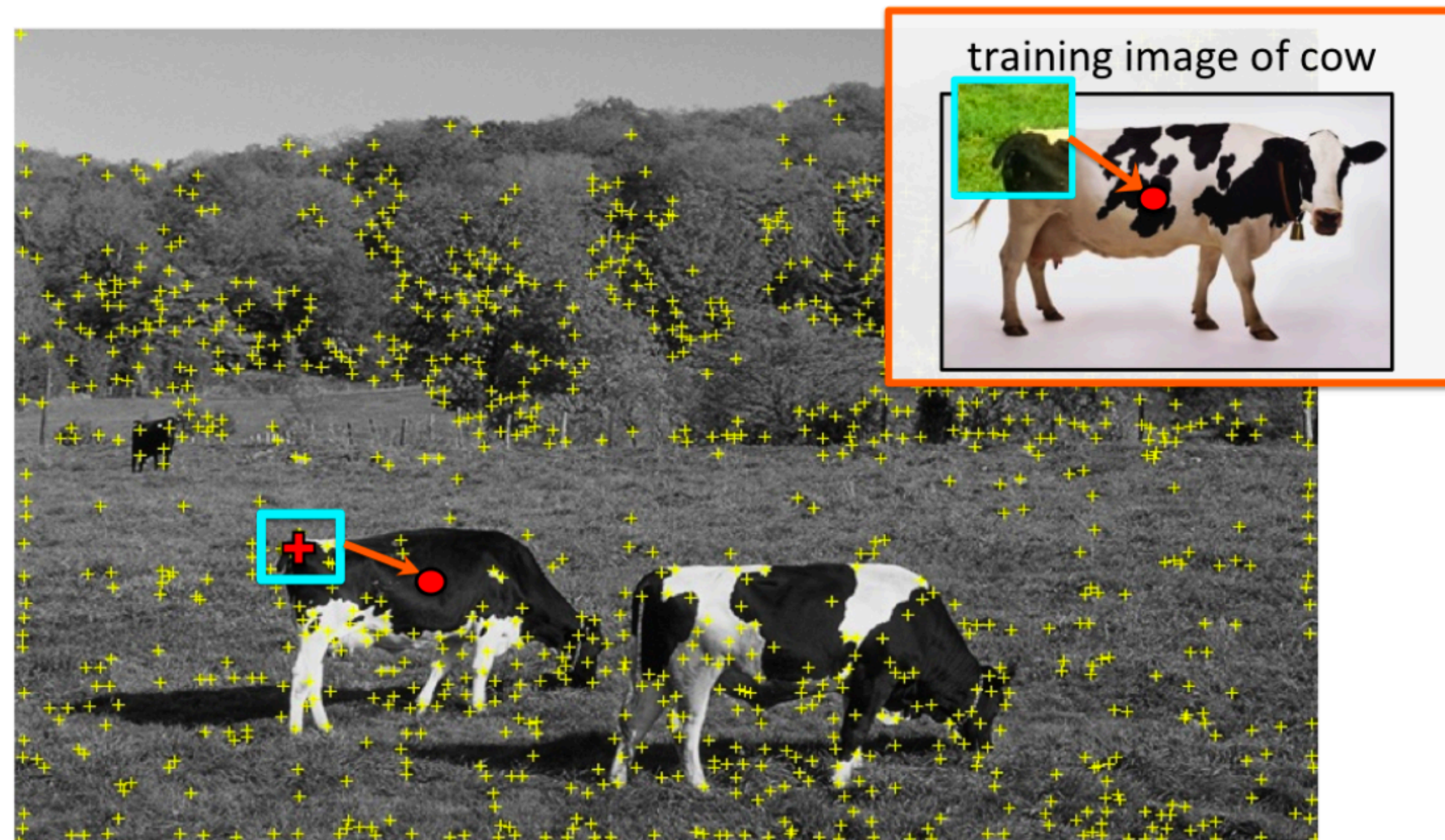
“**Testing**” image



Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows

“**Testing**” image



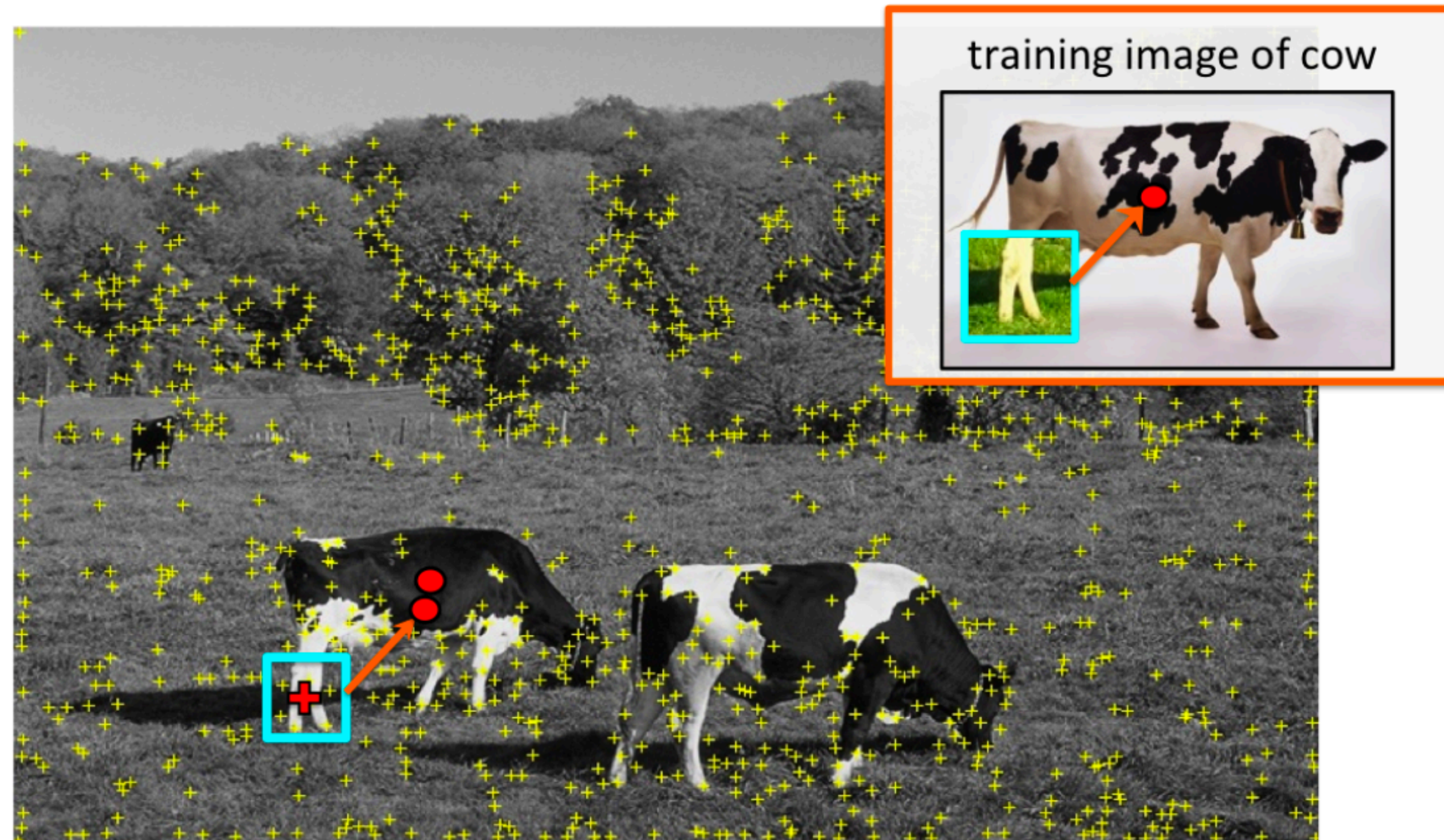
Vote for center of object

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



“**Testing**” image



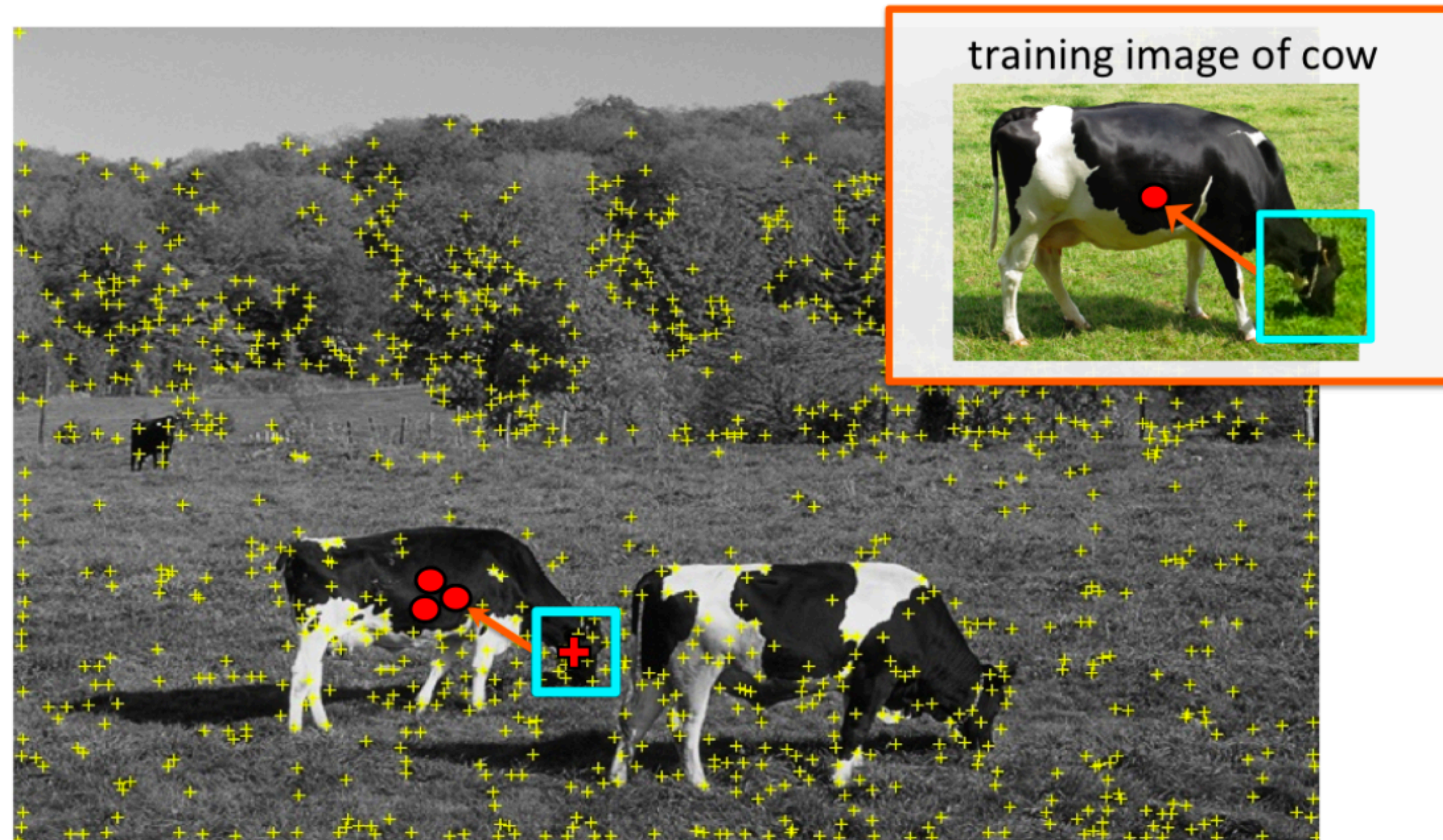
Vote for center of object

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



“**Testing**” image

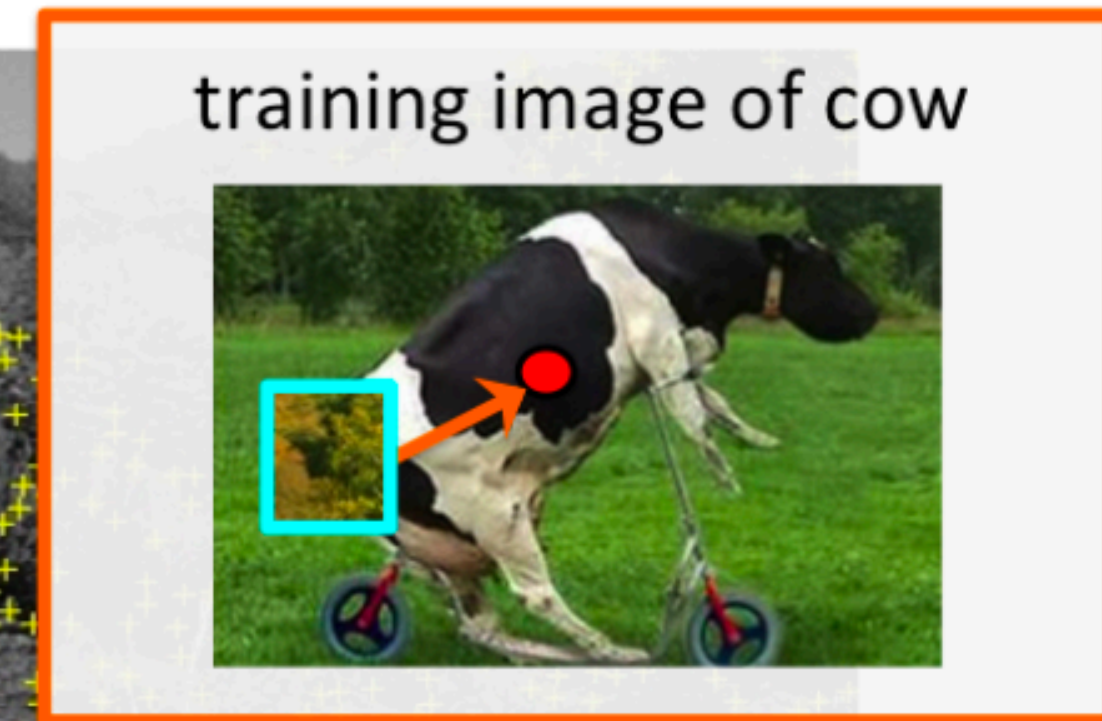
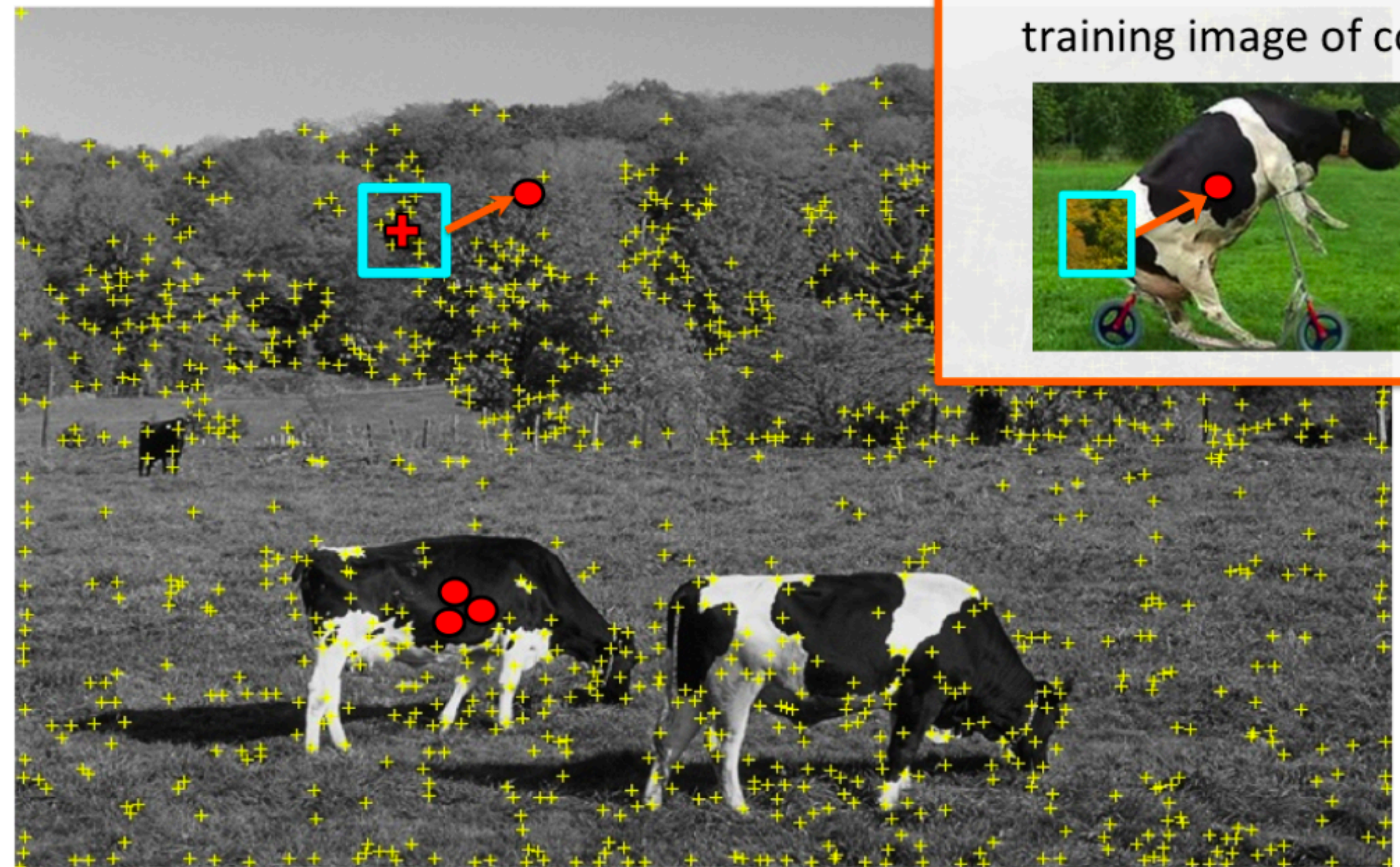


Vote for center of object

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows

“**Testing**” image

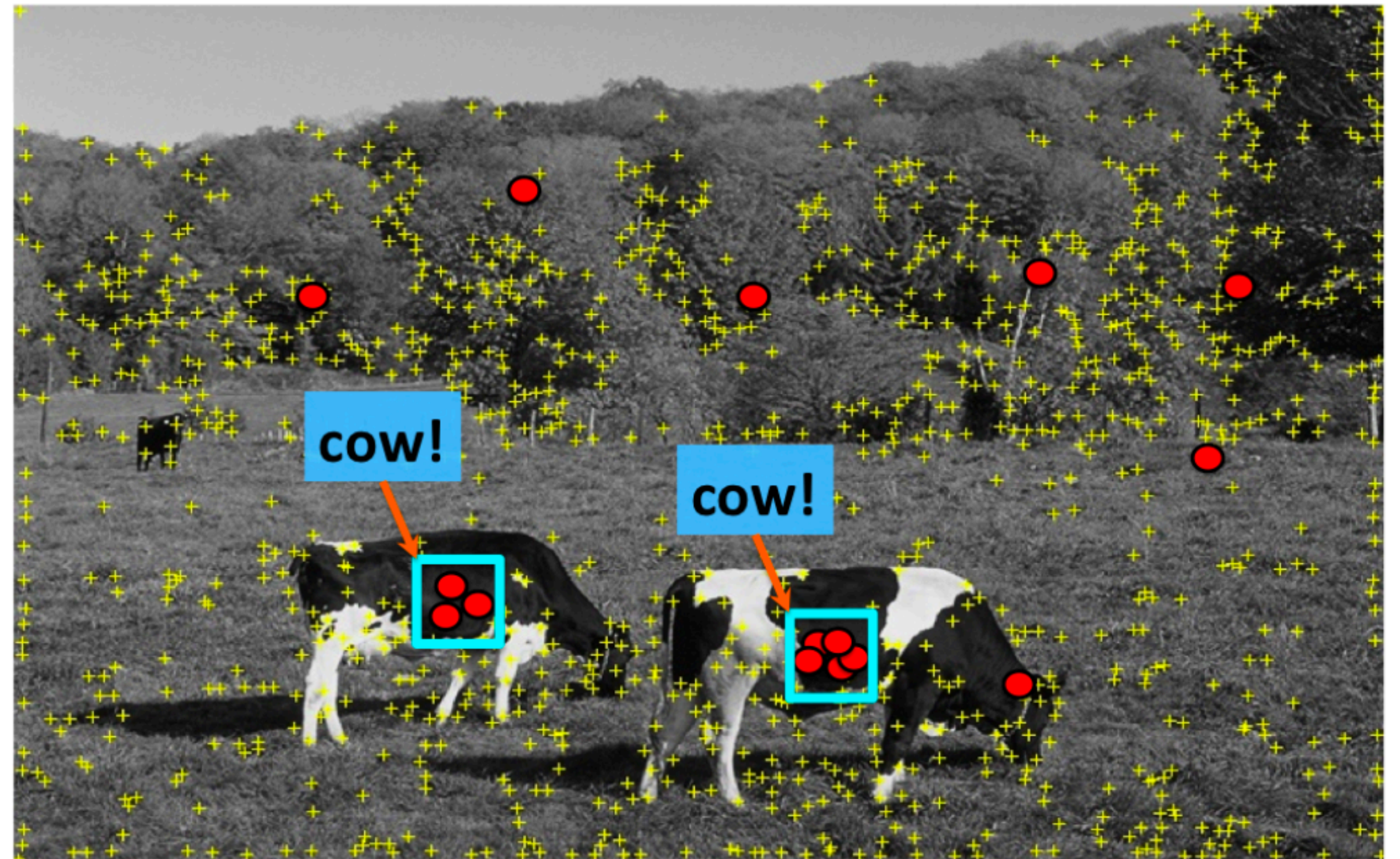


of course sometimes wrong votes are bound to happen

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows

“**Testing**” image



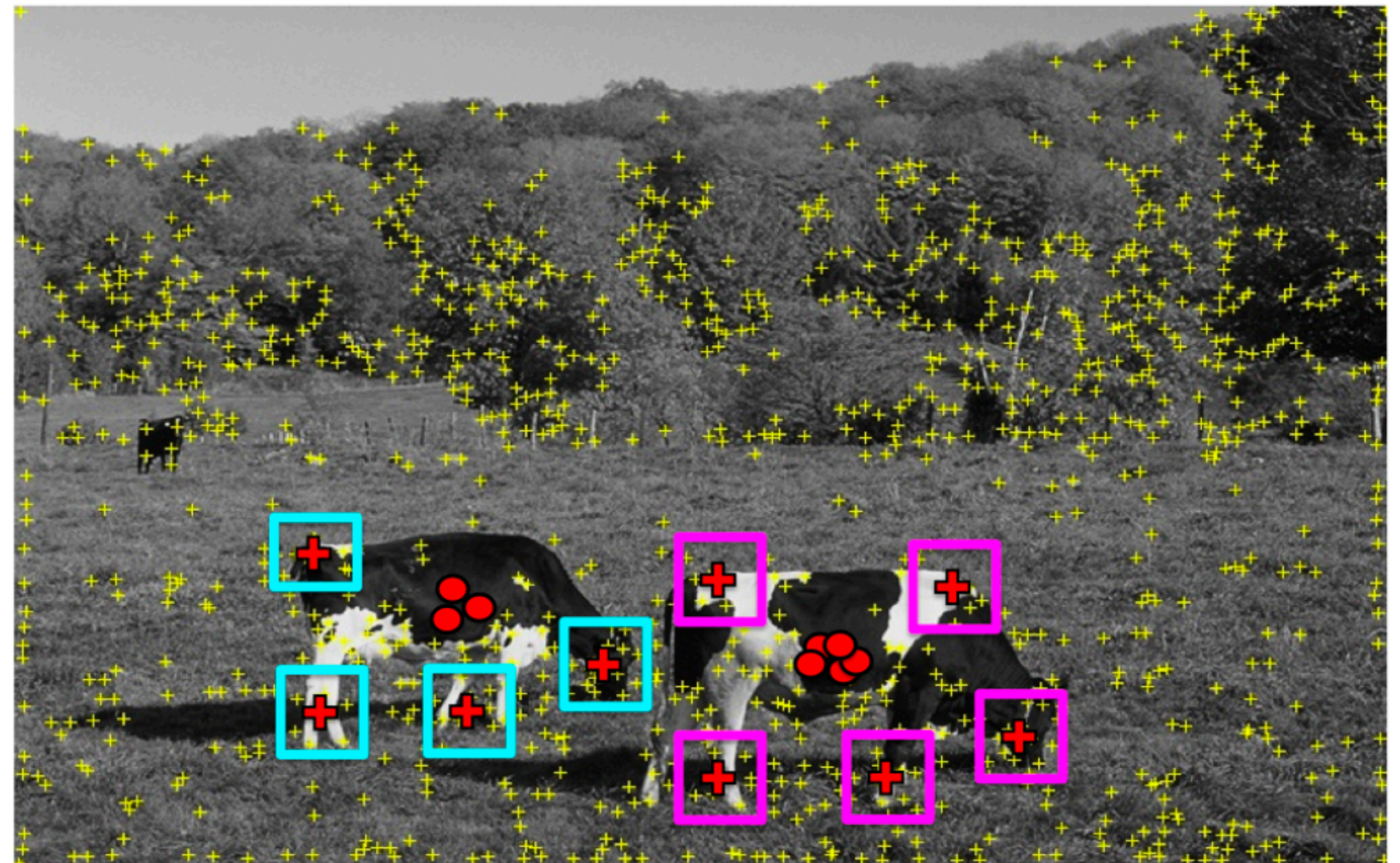
That's ok. We want only **peaks** in voting space.

Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



“**Testing**” image



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

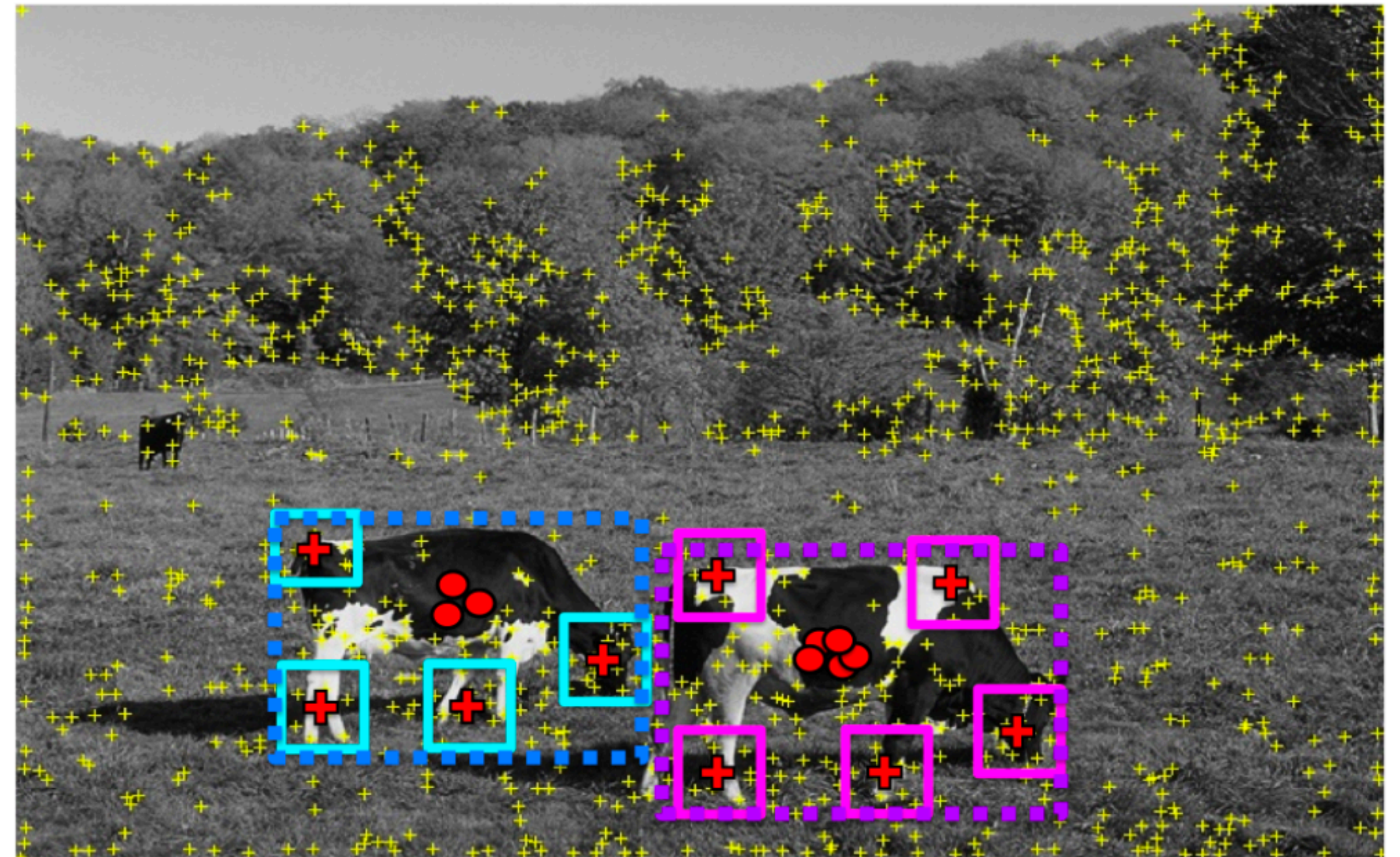
Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



“**Testing**” image

box around patches = object



Find objects based on the back projected patches

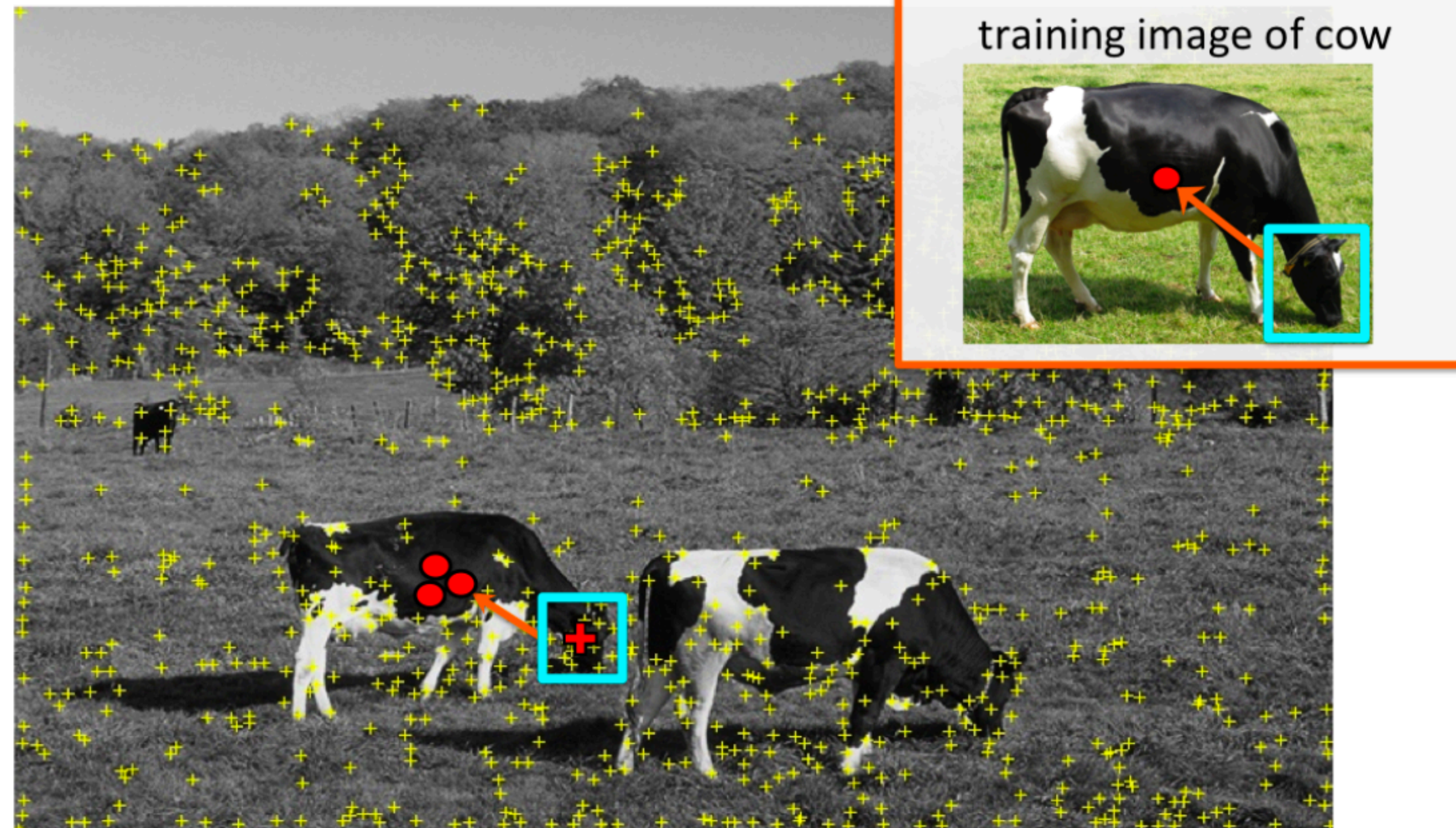
Example 1: Object Recognition — Implicit Shape Model

“**Training**” images of cows



“**Testing**” image

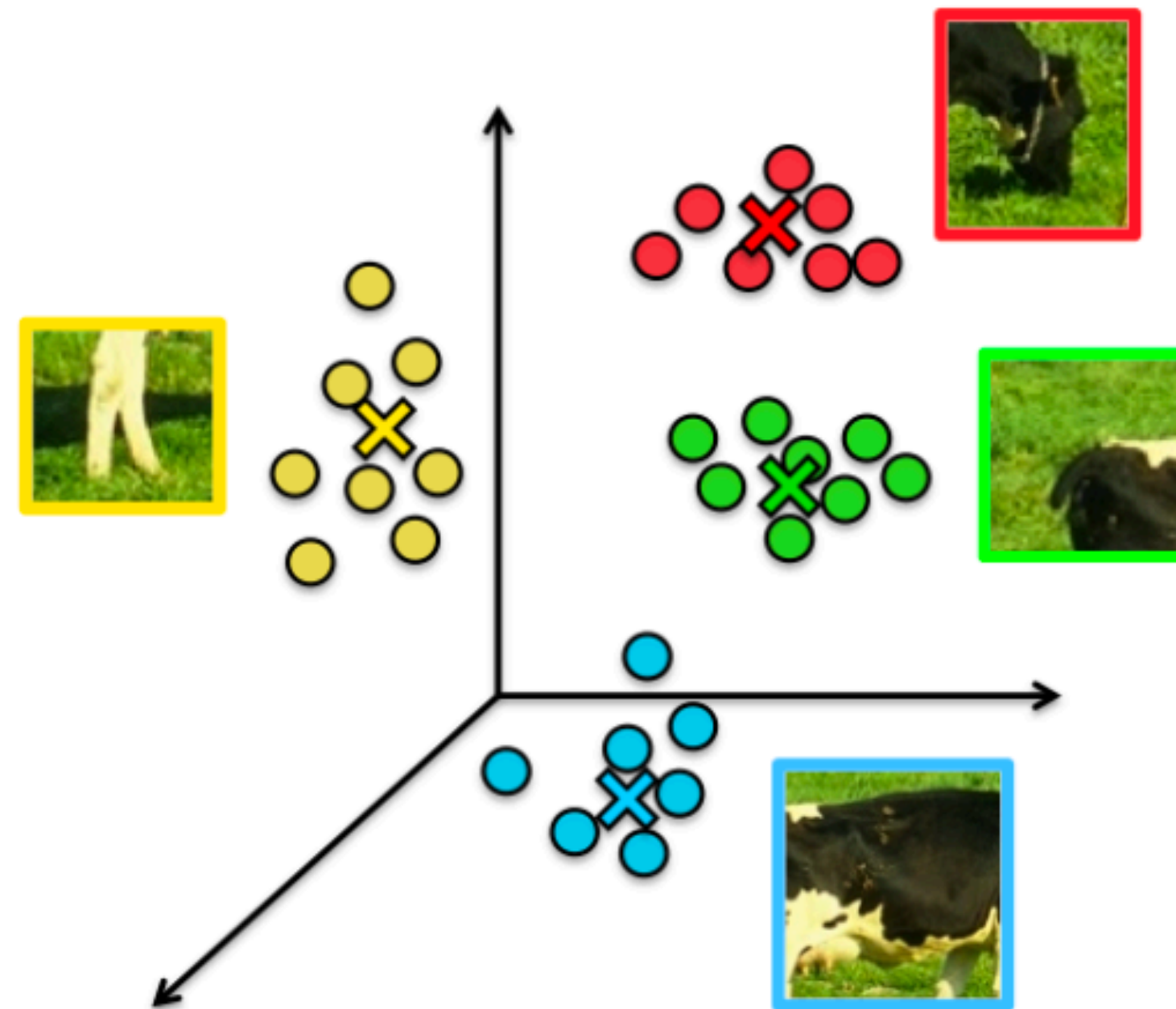
Really easy ... but slow ... how do we make it fast?



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

Visual Words

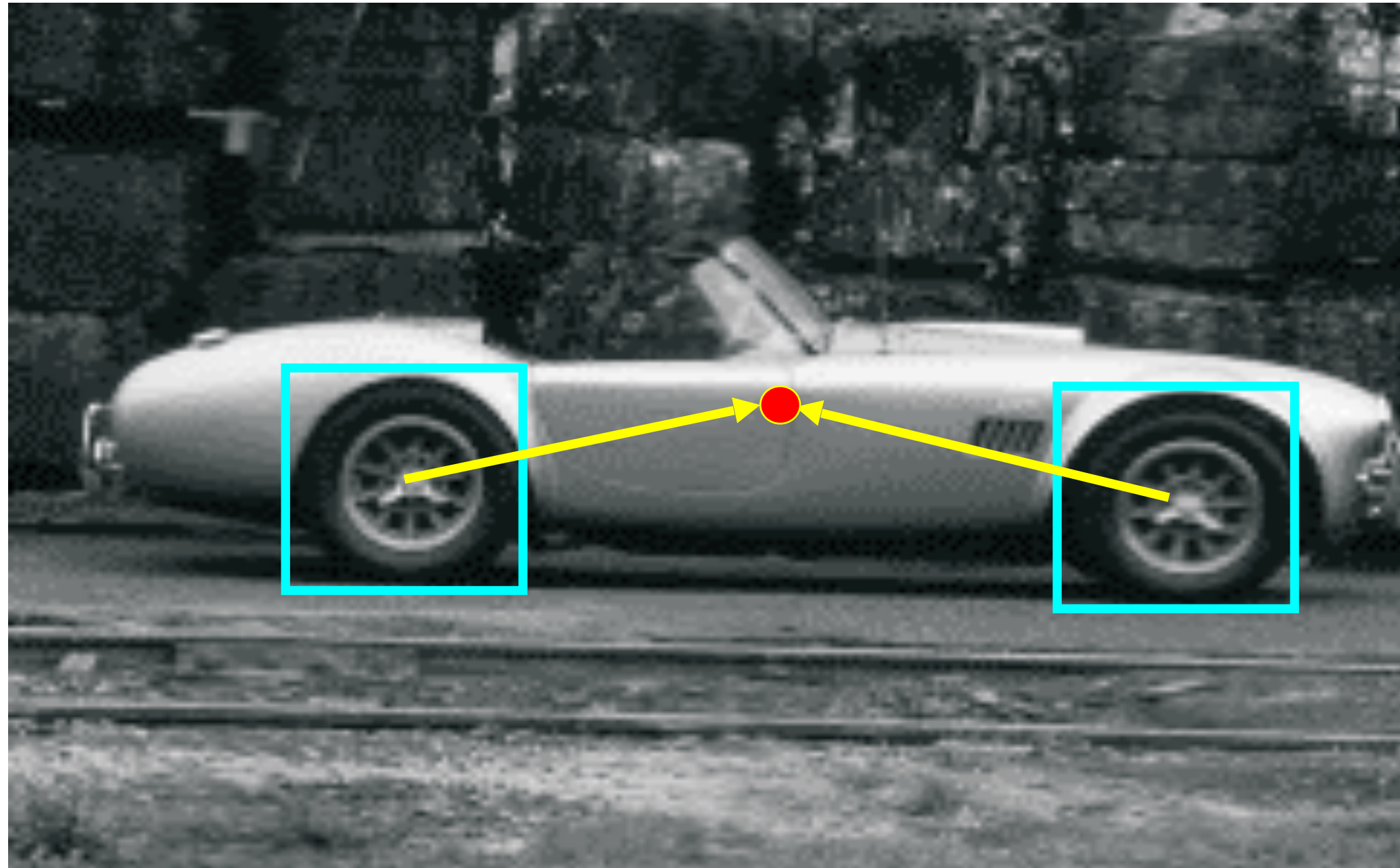
- **Visual vocabulary** (we saw this for retrieval)
- Compare each patch to a small set of visual words (clusters)



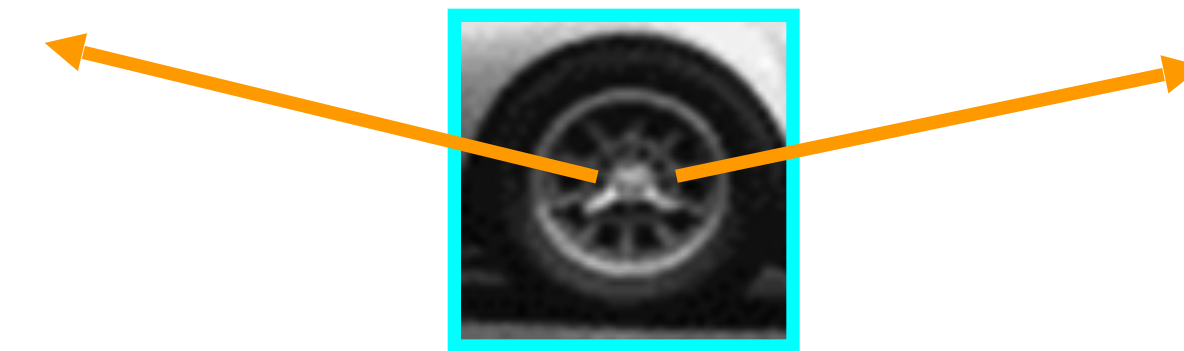
Visual words (visual codebook)!

Example 1: Object Recognition — Implicit Shape Model

Index displacements by “visual codeword”



training image



visual codeword with
displacement vectors

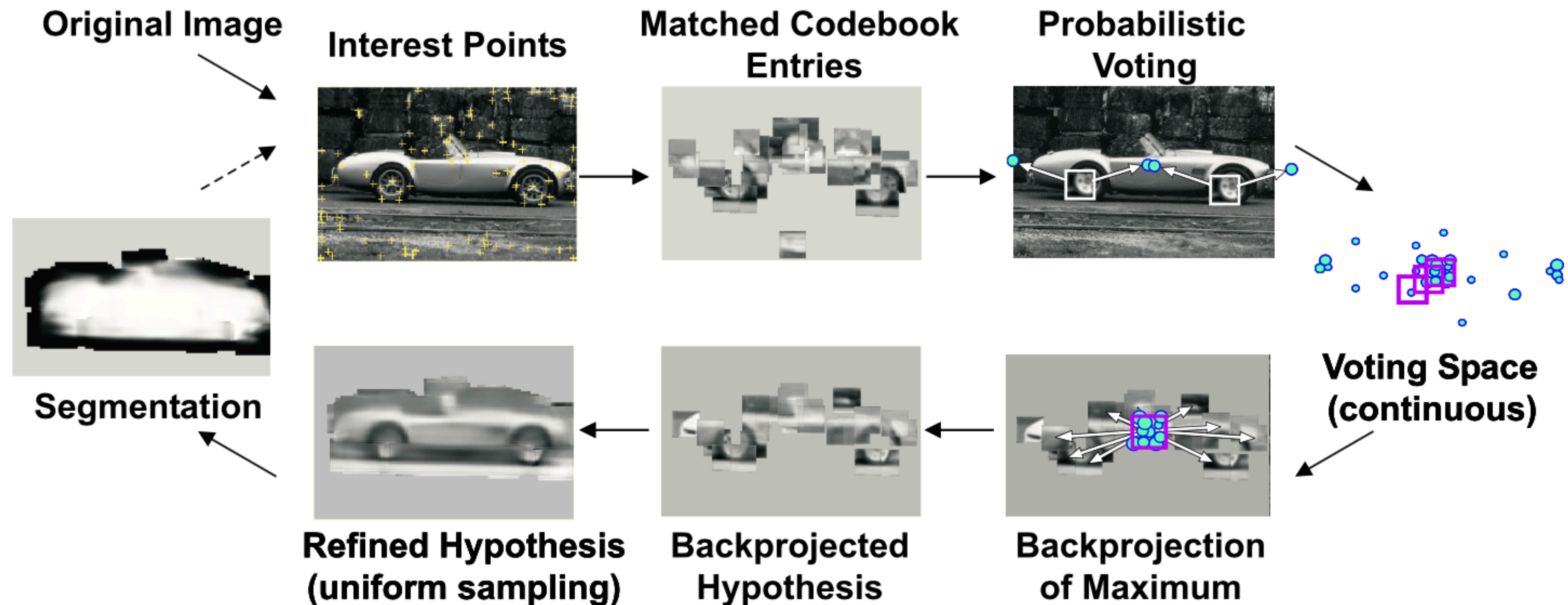
Example 1: Object Recognition — Implicit Shape Model



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

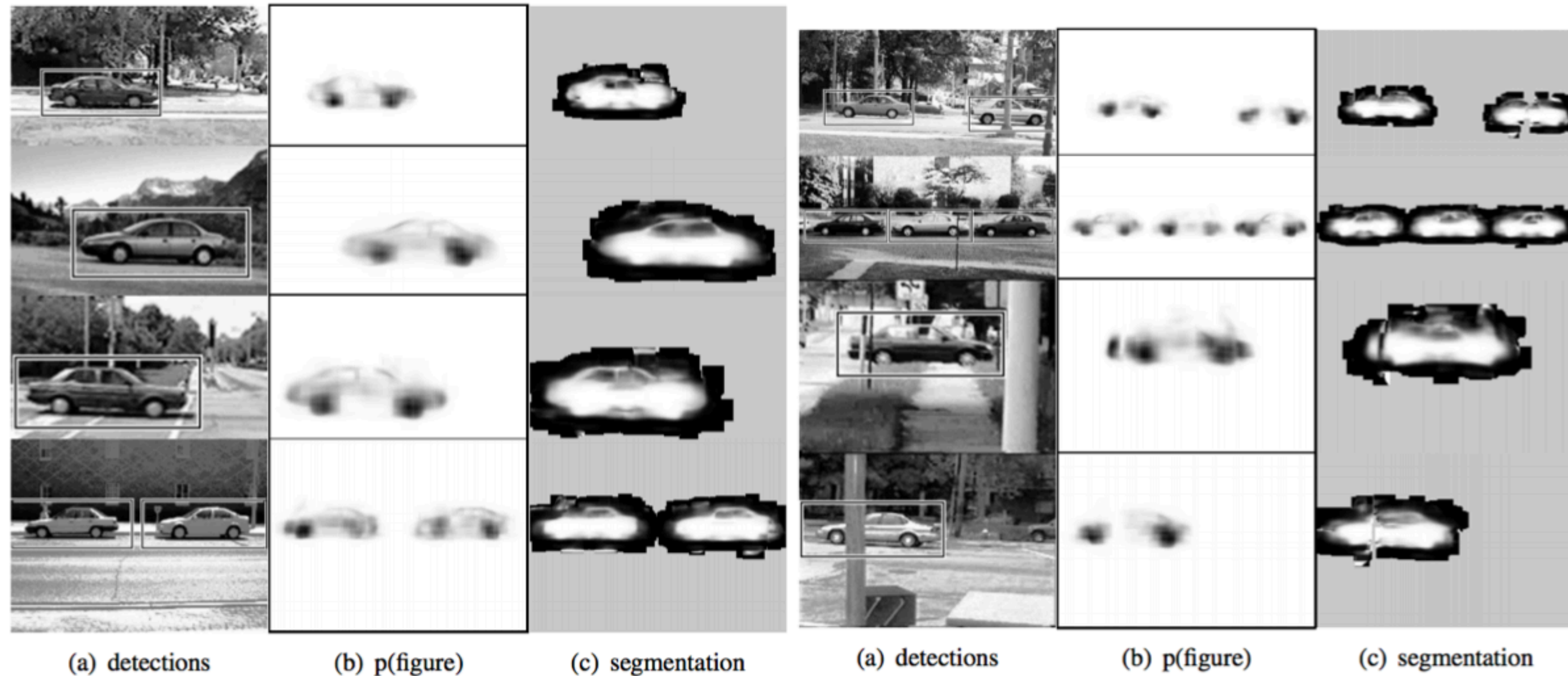
Inferring Other Information: **Segmentation**

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

Inferring Other Information: **Segmentation**



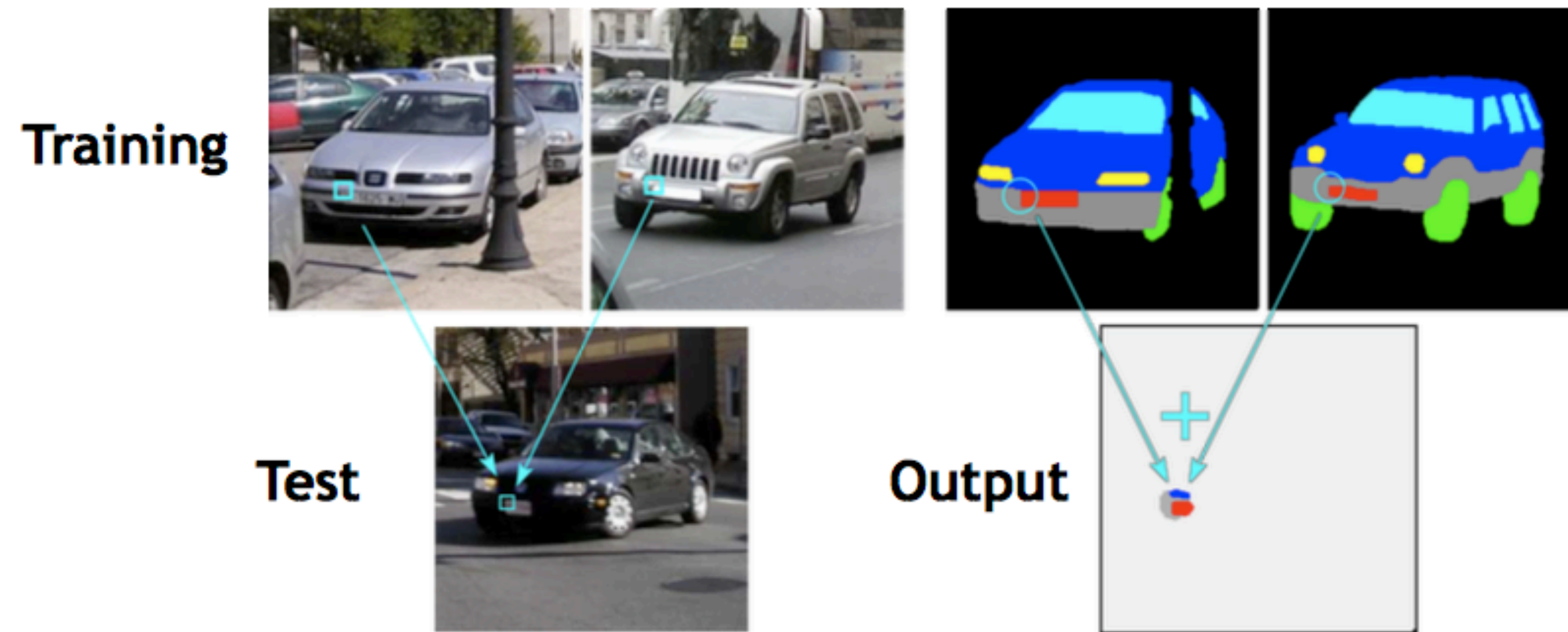
[Source: B. Leibe]

Inferring Other Information: **Segmentation**

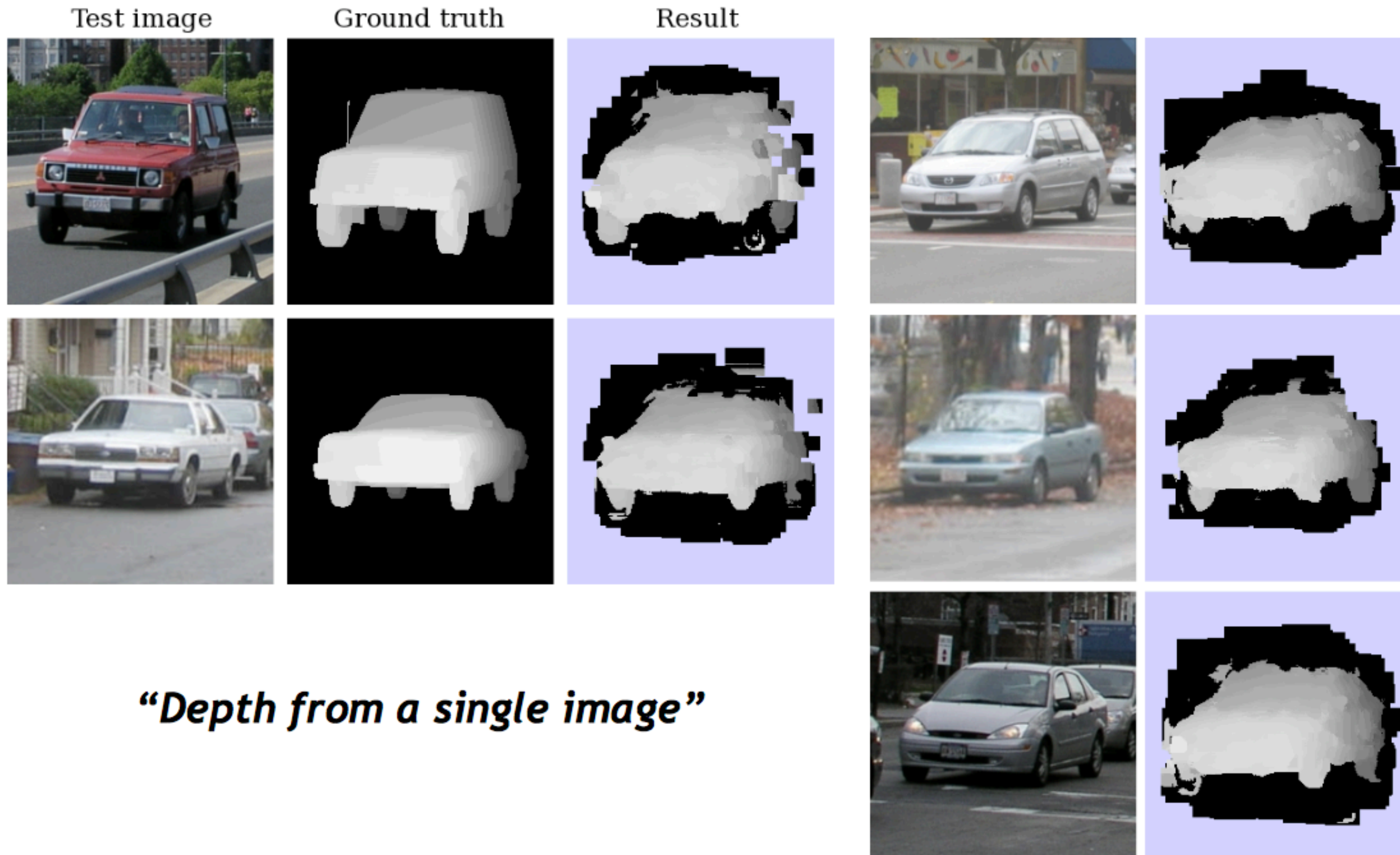


[Source: B. Leibe]

Inferring Other Information: **Part Labels**



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.

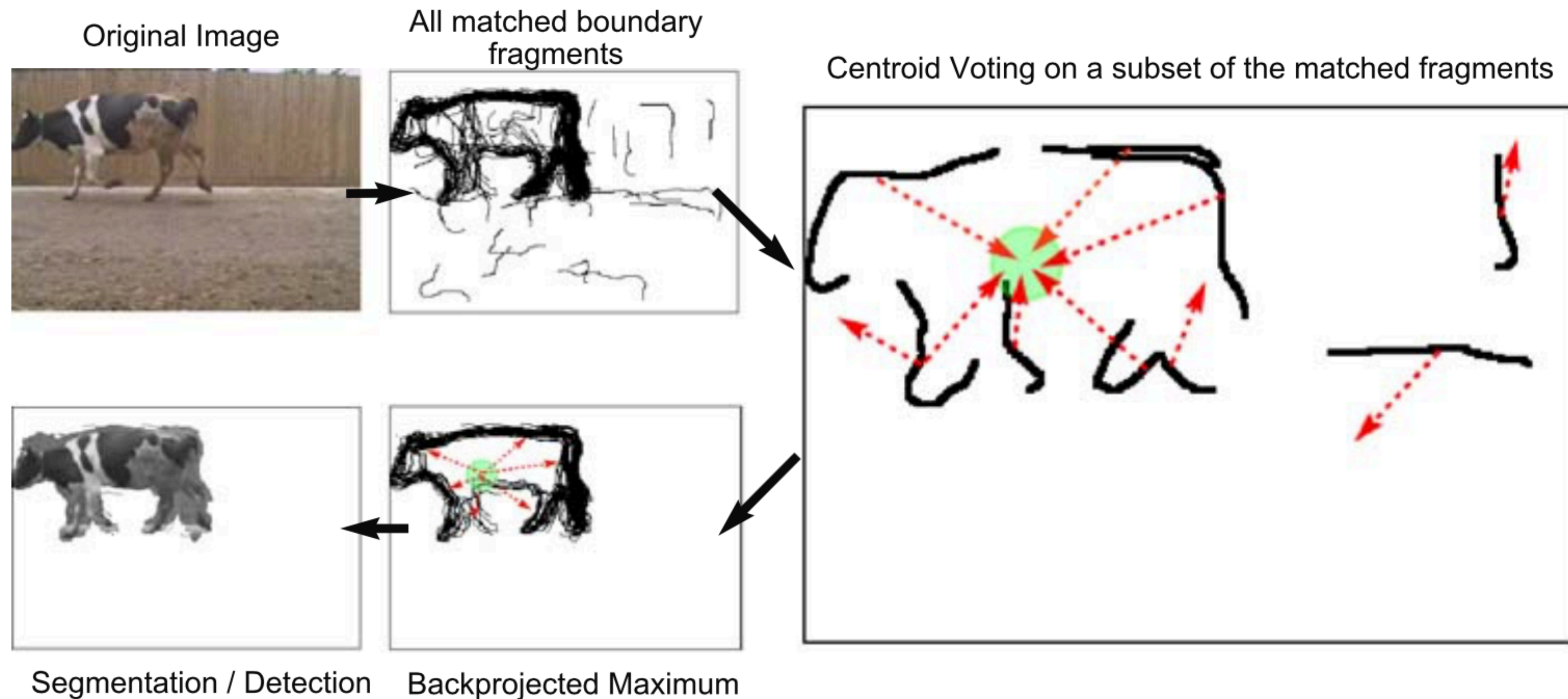


Image credit: Opelt et al., 2006

Example 2: Object Recognition — Boundary Fragments

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

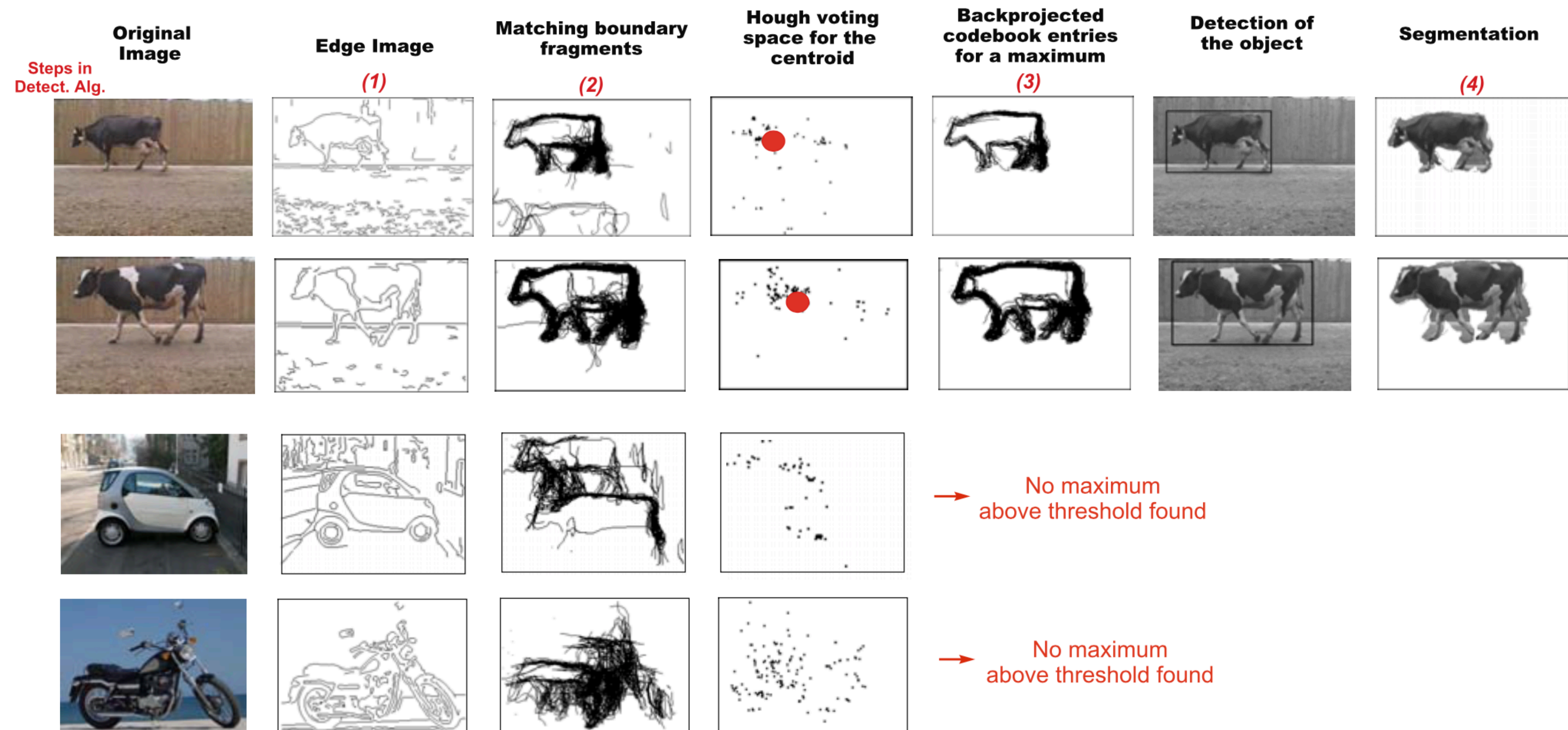


Image credit: Opelt et al., 2006

Example 3: Object Recognition — Poselets

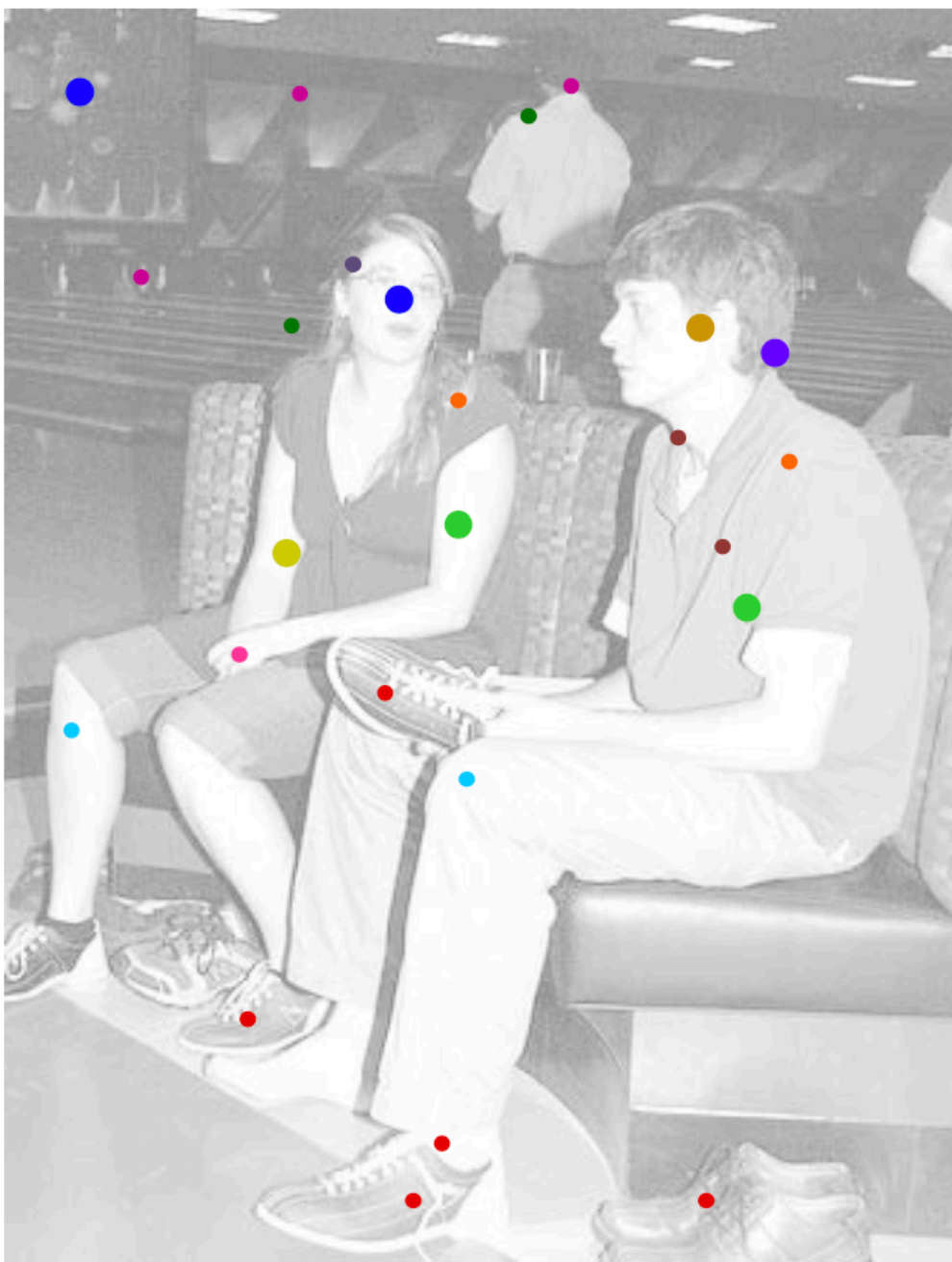
Poselets are image patches that have distinctive appearance and can be used to infer some of the configuration of a parts-based object. Detected poselets vote for the object configuration.



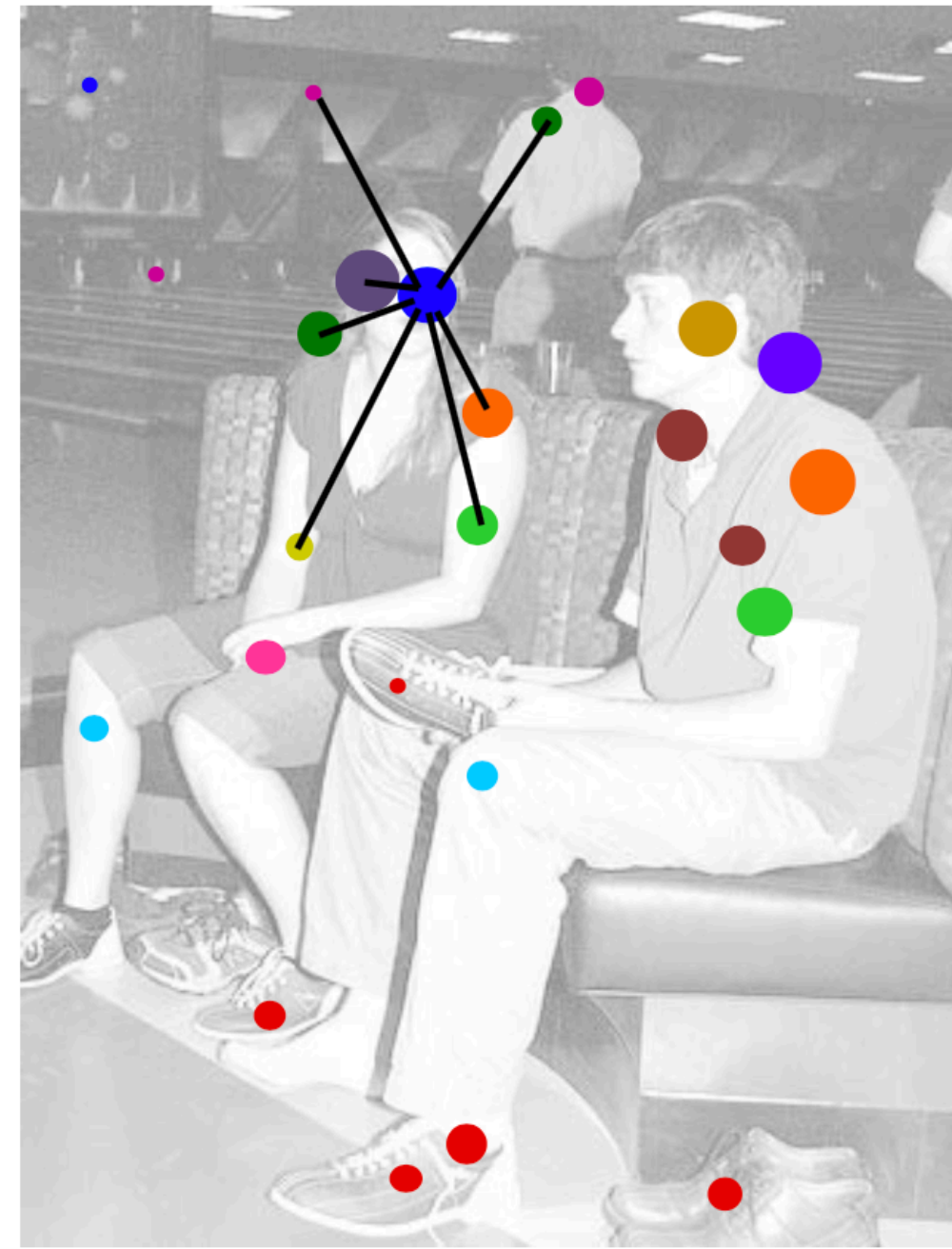
Image credit: Bourdev and Malik, 2009

Example 3: Object Recognition — Poselets

Poselets are image patches that have distinctive appearance and can be used to infer some of the configuration of a parts-based object. Detected poselets vote for the object configuration.



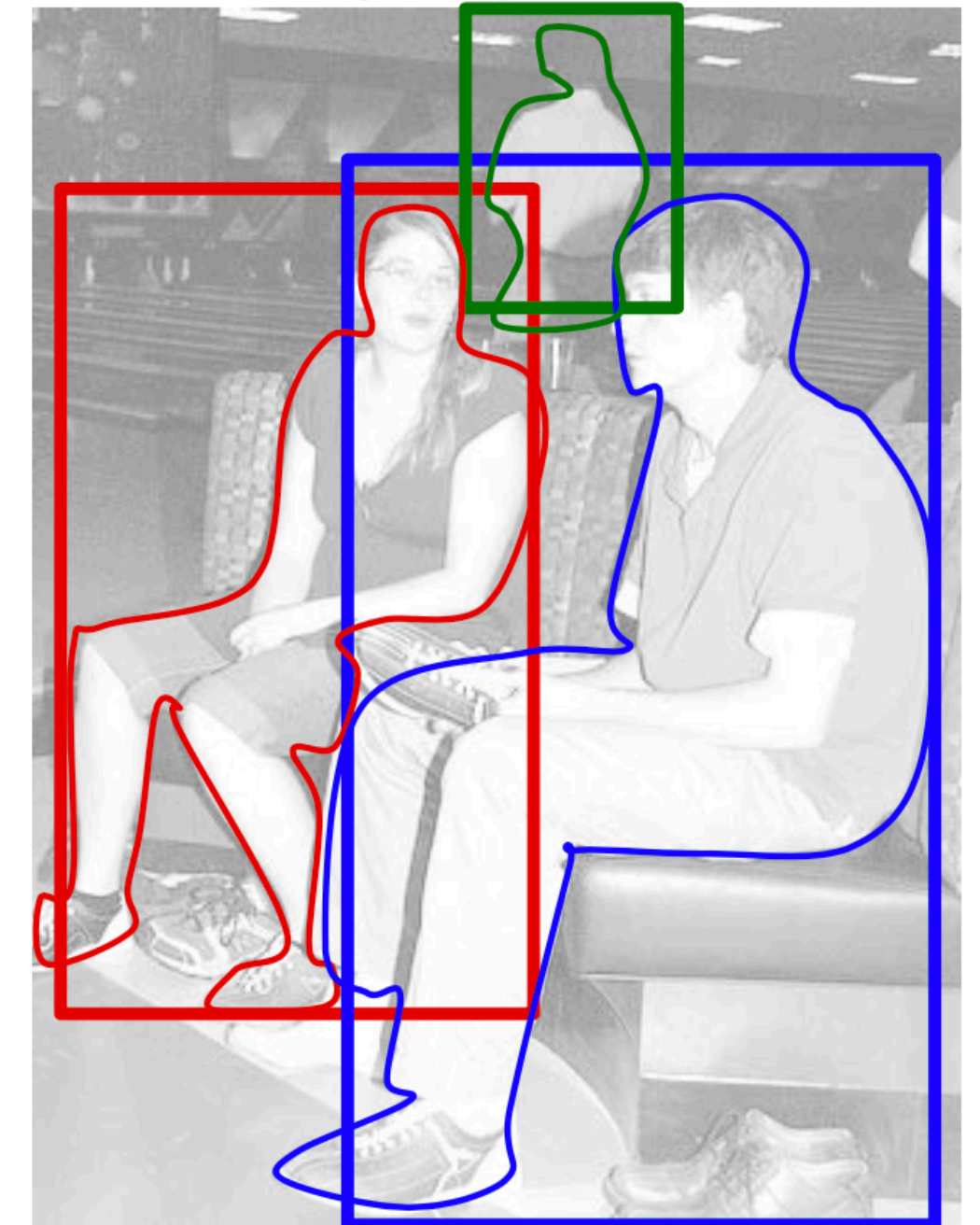
1. **q-scores.** Different colors illustrate different poselet detectors firing in the image. The blob size illustrates the score of the independent poselet classifier.



2. **Q-scores (Section 4).** Evidence from consistent poselet activations leads to a reranking based on mutual activation (Q-scores). Weaker activations consistent with others gain importance, whereas inconsistent ones get damped.



3. **Clustering (Section 5).** Activations are merged in a greedy manner starting with the strongest activation. Merging is based on pairwise consistency.



4. **Bounding boxes (Section 6) and segmentations (Section 7).** We predict the visible bounds and the contour of the person using the poselets within the cluster.

Image credit: Bourdev and Malik, 2009

Discussion of Hough Transform

Advantages:

- Can handle high percentage of outliers: each point votes separately
- Can detect multiple instances of a model in a single pass

Disadvantages:

- Complexity of search time increases exponentially with the number of model parameters
- Can be tricky to pick a good bin size

Summary of Hough Transform

The **Hough transform** is another technique for fitting data to a model

- a voting procedure
- possible model parameters define a quantized accumulator array
- data points “vote” for compatible entries in the accumulator array

A key is to have each data point (token) constrain model parameters as tightly as possible



CPSC 425: Computer Vision

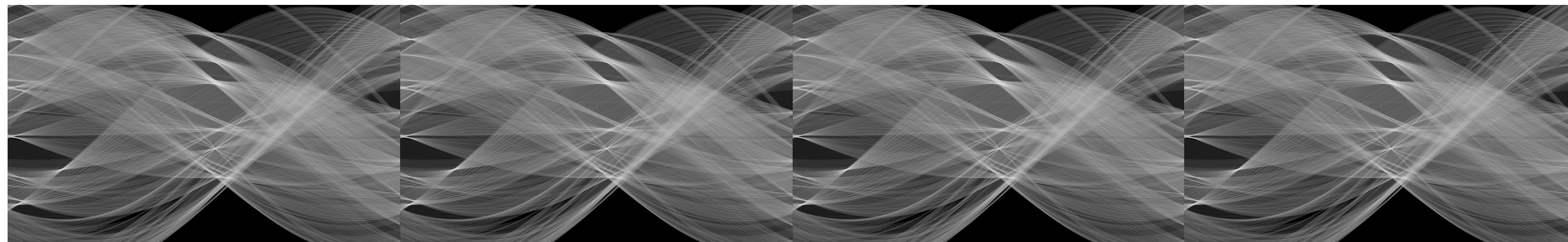


Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 18: Stereo

Stereo Vision

Problem Formulation:

Determine depth using two images acquired from (slightly) different viewpoints

Key Idea(s):

The 3D coordinates of each point imaged are constrained to lie along a ray. This is true also for a second image obtained from a (slightly) different viewpoint.

Rays for the same point in the world intersect at the actual 3D location of that point

Stereo Vision

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

Binoculars

Binoculars enhance binocular depth perception in two distinct ways:

1. magnification
2. longer baseline (i.e., distance between entering light paths) compared to the normal human inter-pupillary distance

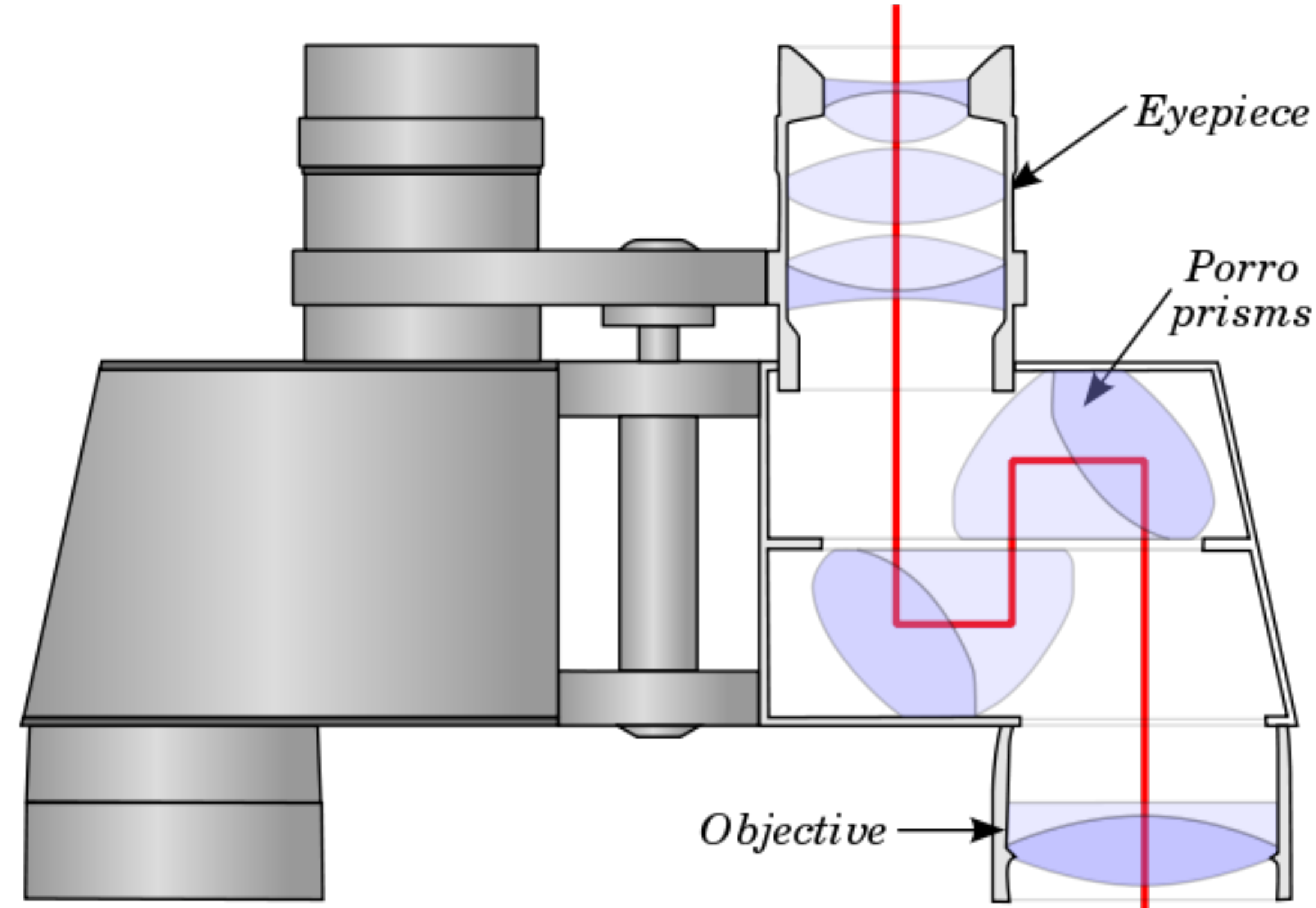


Figure credit: <http://en.wikipedia.org/wiki/Binoculars>

Stereo Vision

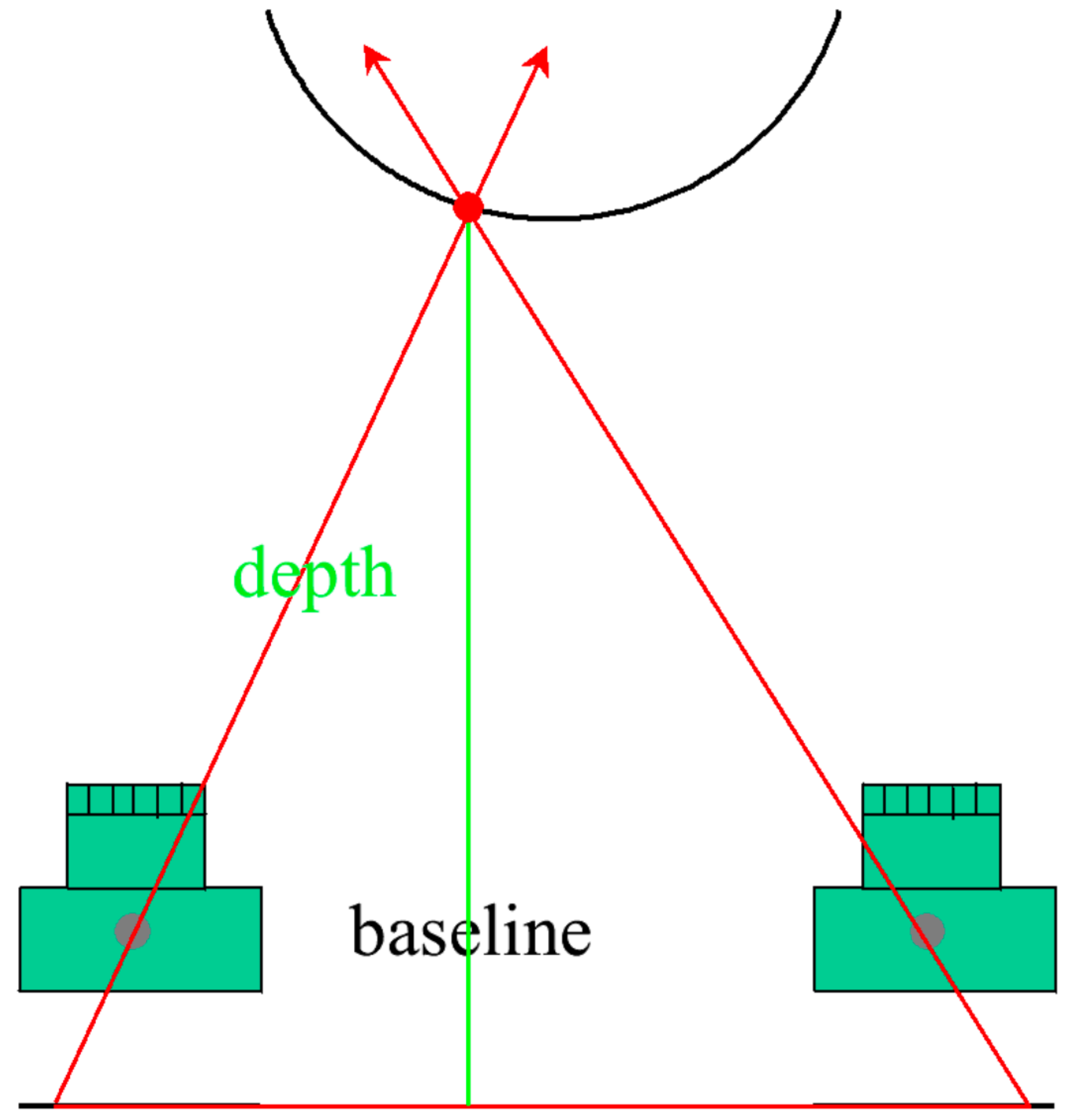
Task: Compute depth from two images acquired from (slightly) different viewpoints

Approach: “Match” locations in one image to those in another

Sub-tasks:

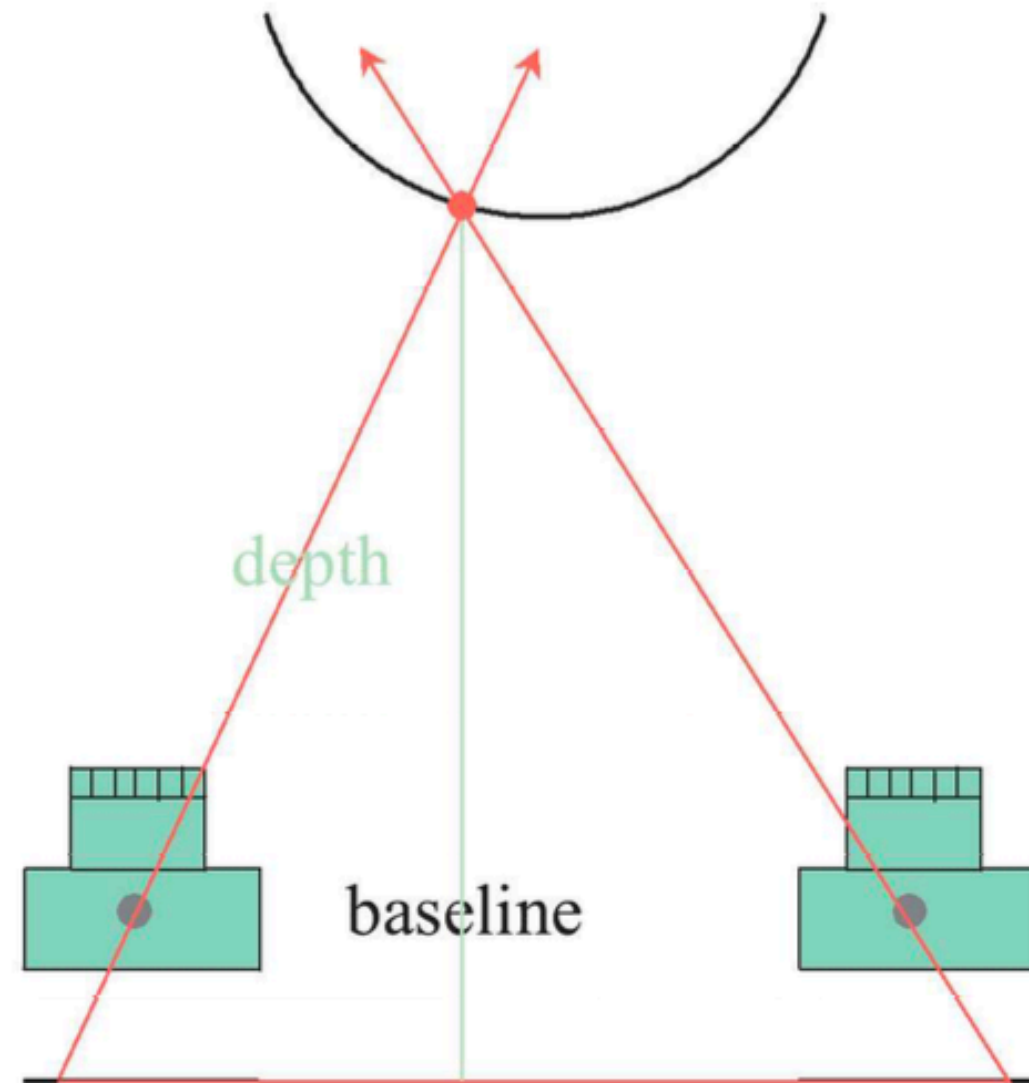
- Calibrate cameras and camera positions
- Find all corresponding points (the hardest part)
- Compute depth and surfaces

Stereo Vision



Slide credit: Trevor Darrell

Stereo Vision

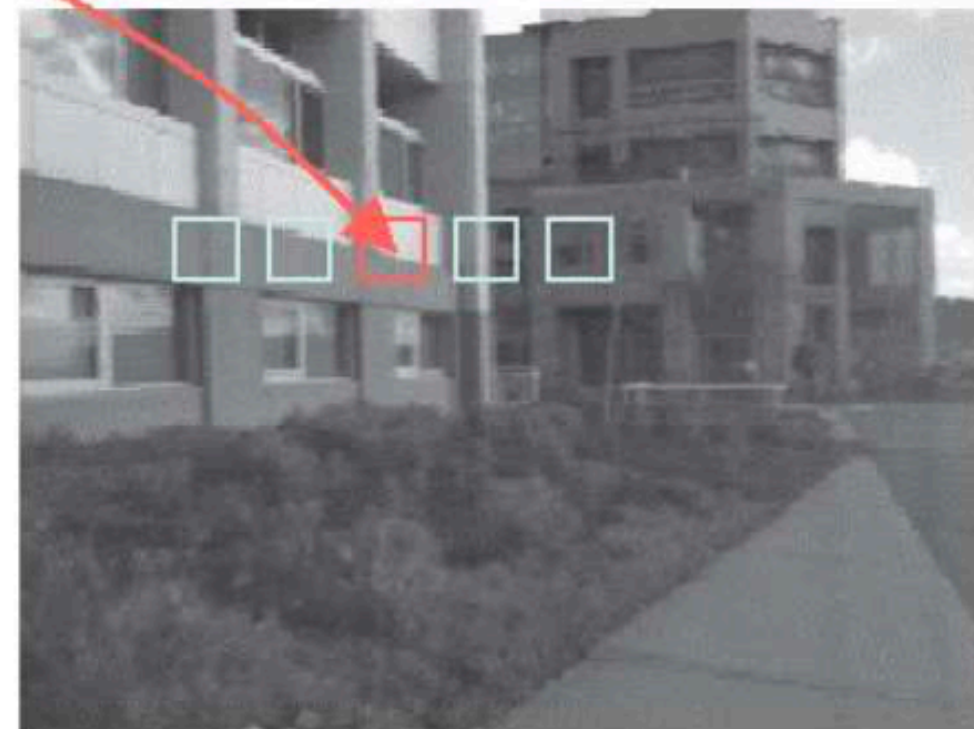


Triangulate on two images of the same point

Left



Right



Match correlation windows
across scan lines

Image credit: Point Grey Research
Slide credit: Trevor Darrell

Point Grey Research **Digiclops**

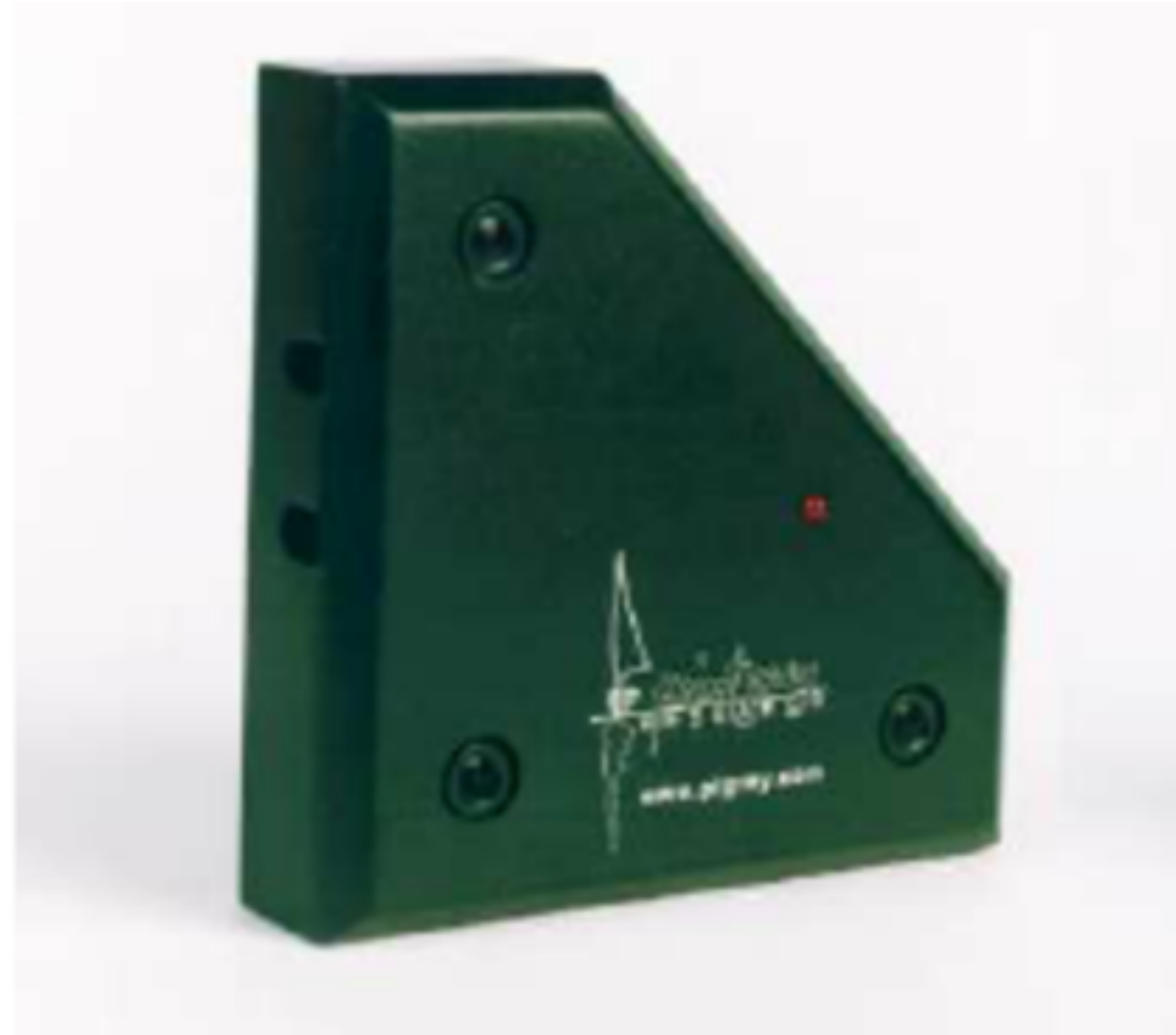
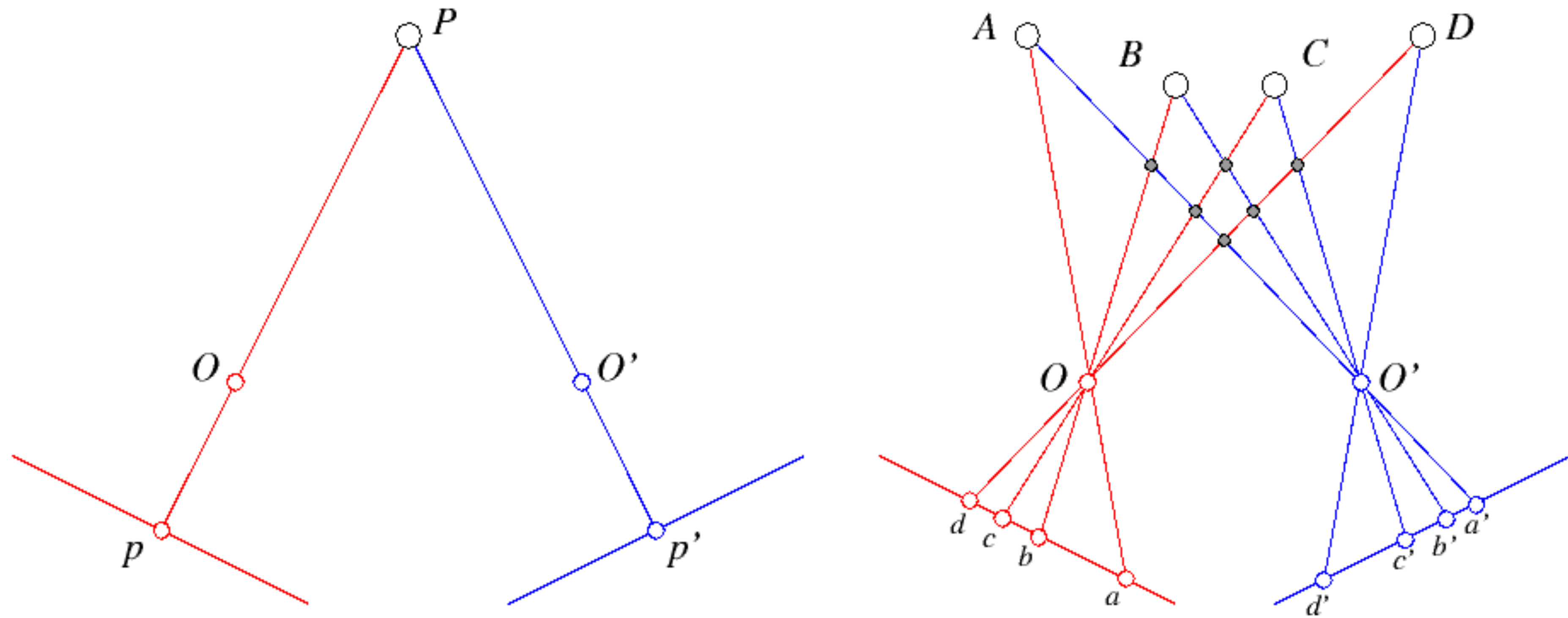


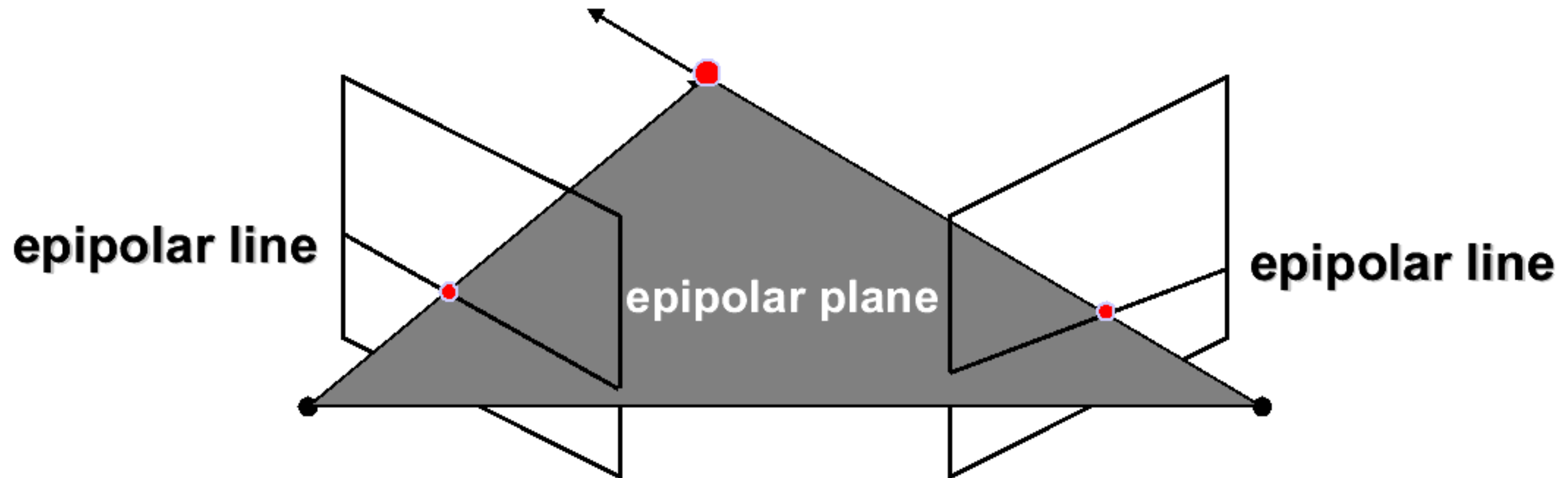
Image credit: Point Grey Research

Correspondence



Forsyth & Ponce (2nd ed.) Figure 7.2

The **Epipolar** Constraint



Matching points lie along corresponding epipolar lines

Reduces correspondence problem to 1D search along conjugate epipolar lines

Greatly reduces cost and ambiguity of matching

Slide credit: Steve Seitz

Simplest Case: **Rectified** Images

Image planes of cameras are **parallel**

Focal **points** are at same height

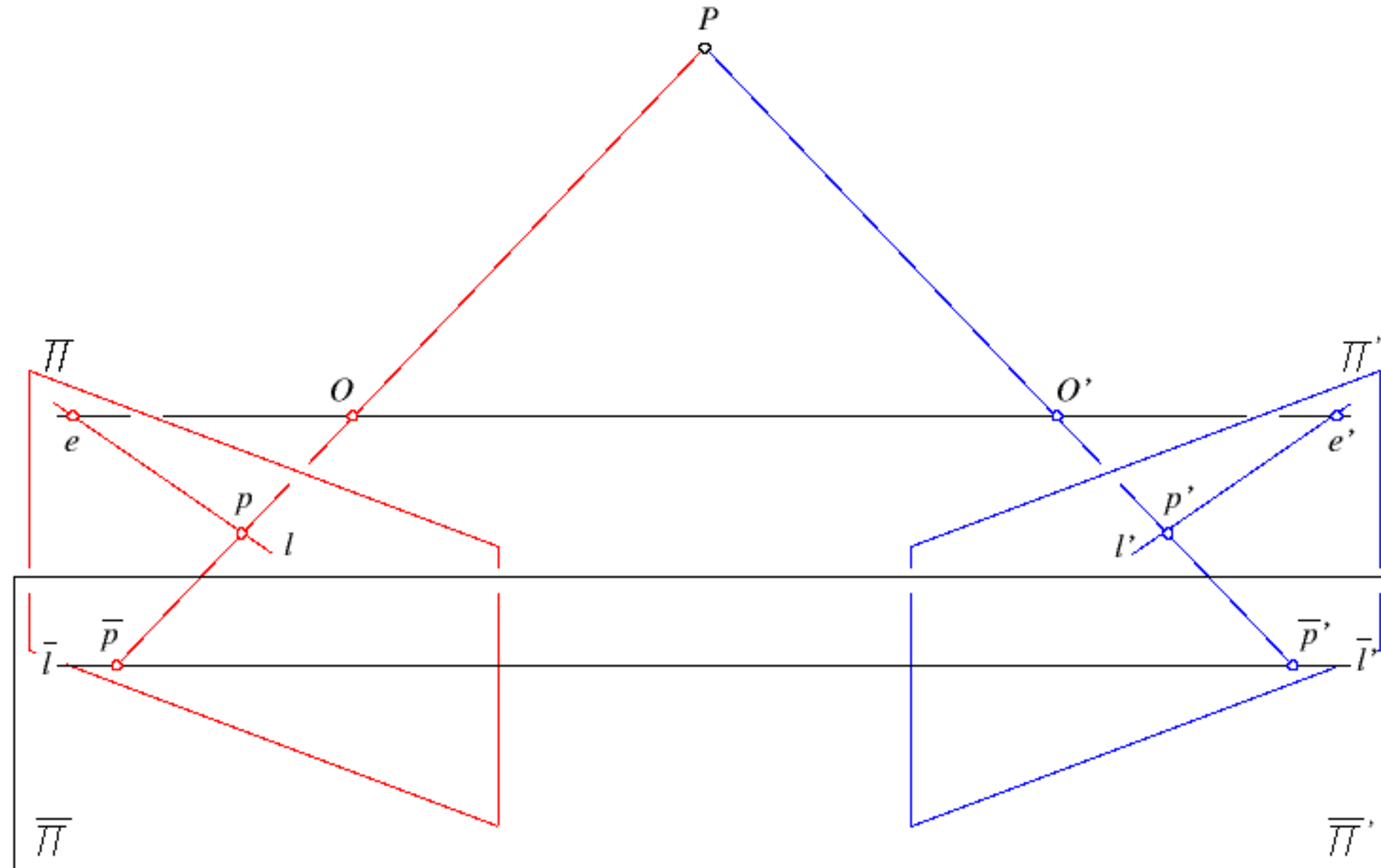
Focal **lengths** same

Then, **epipolar lines** fall along the **horizontal scan lines** of the images

We assume images have been **rectified** so that epipolar lines correspond to scan lines

- Simplifies algorithms
- Improves efficiency

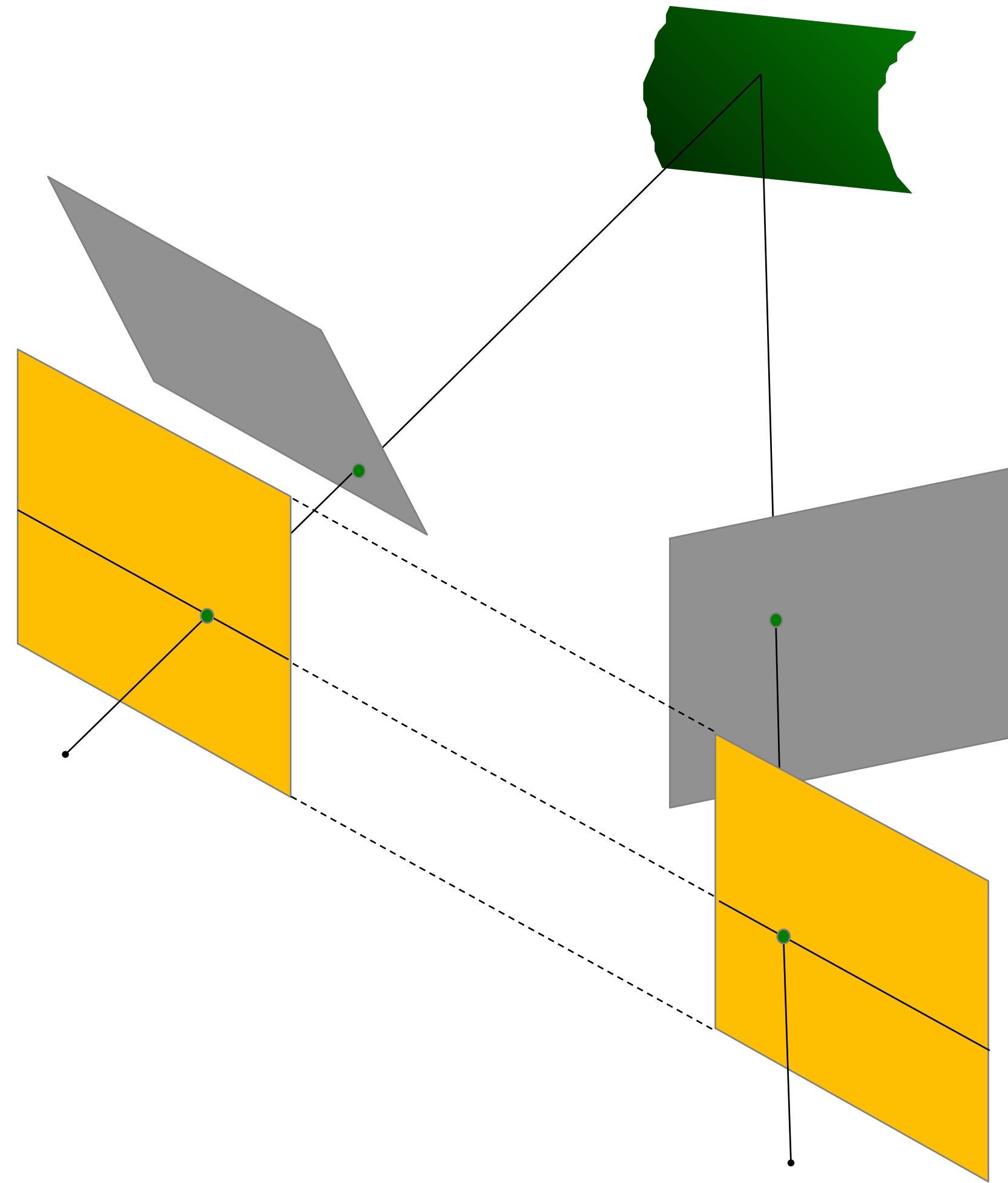
Rectified Stereo Pair



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.

Rectified Stereo Pair: Example

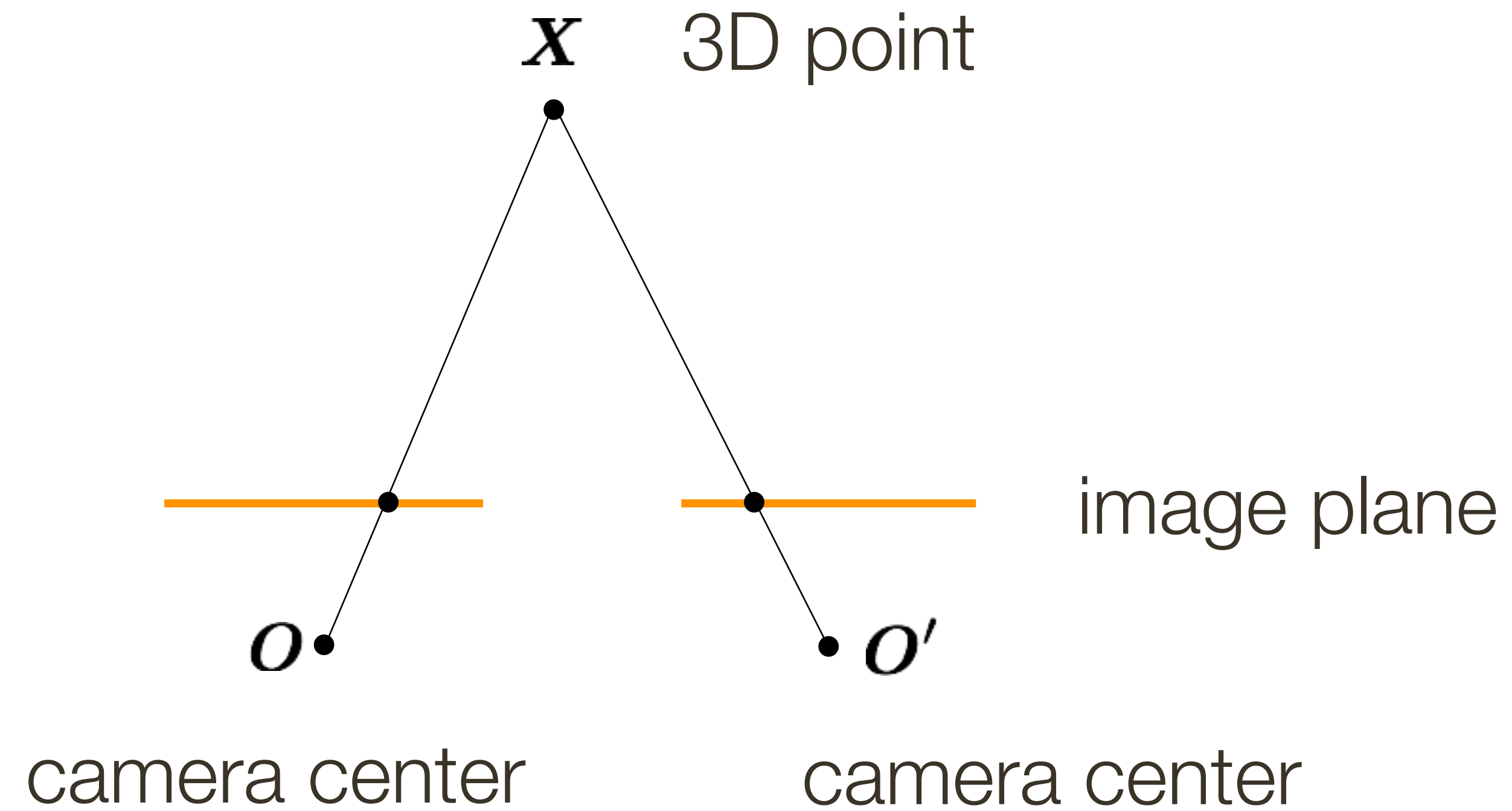
Before Rectification



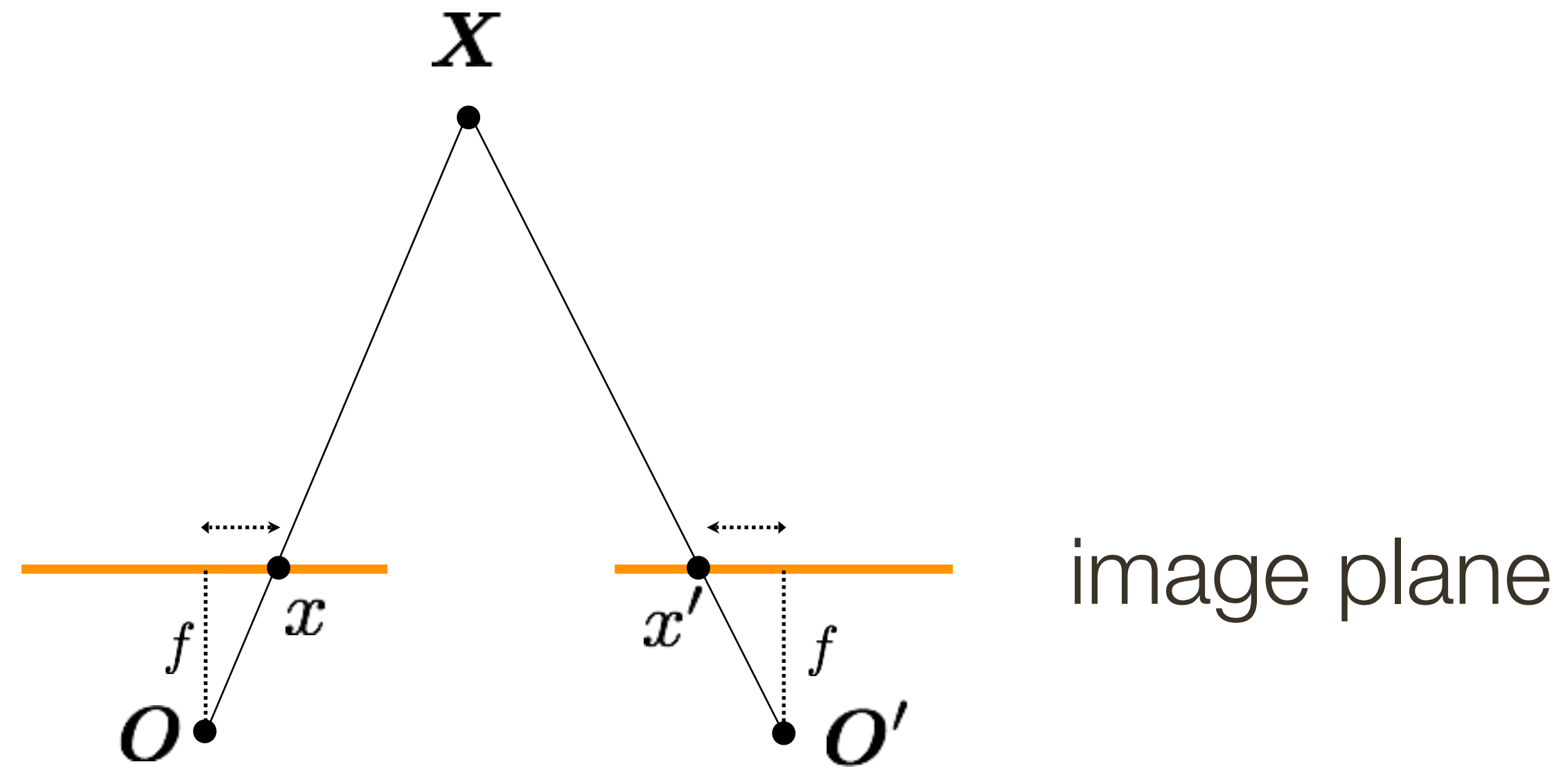
After Rectification

So

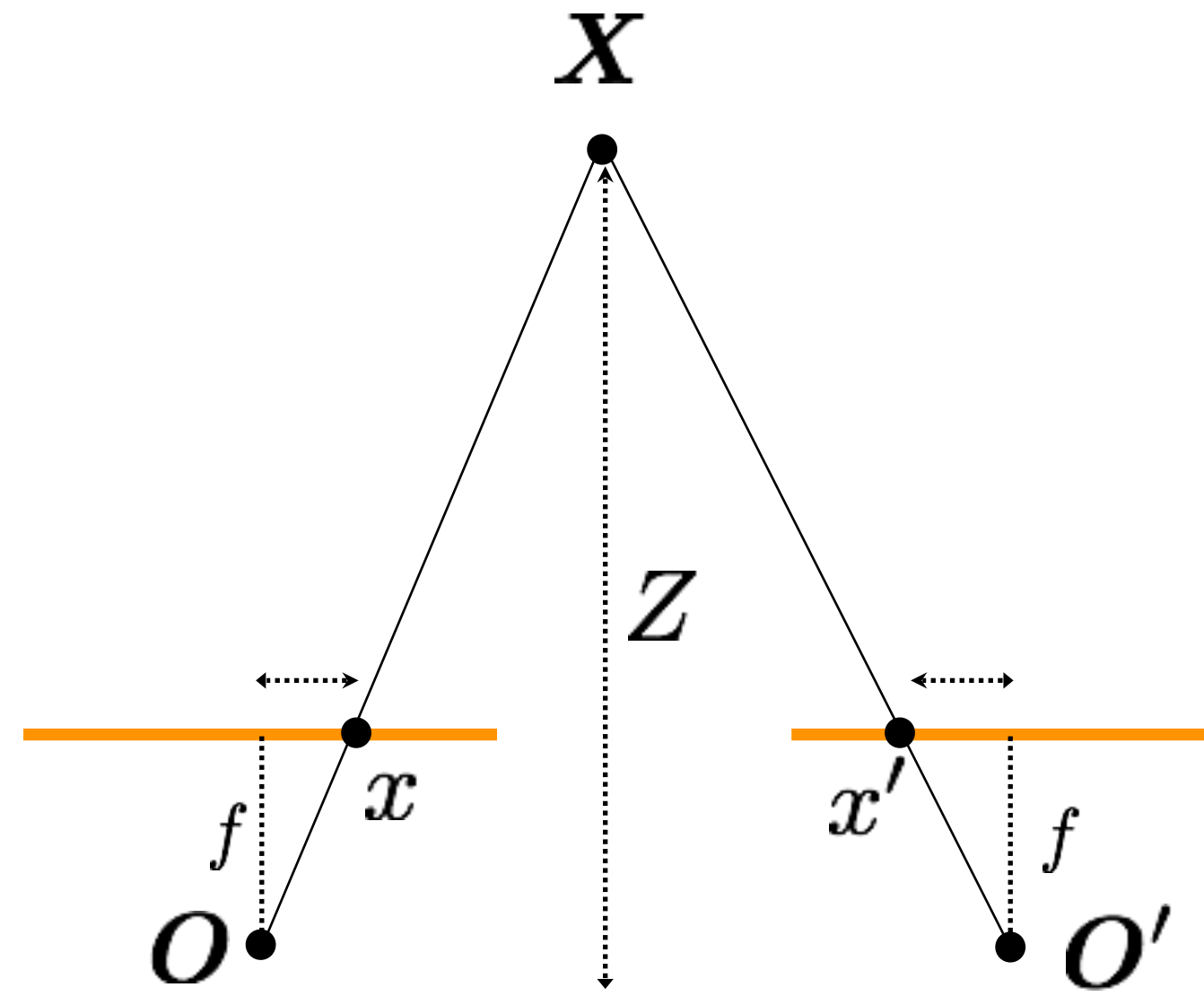
Rectified Stereo Pair: Depth Estimate



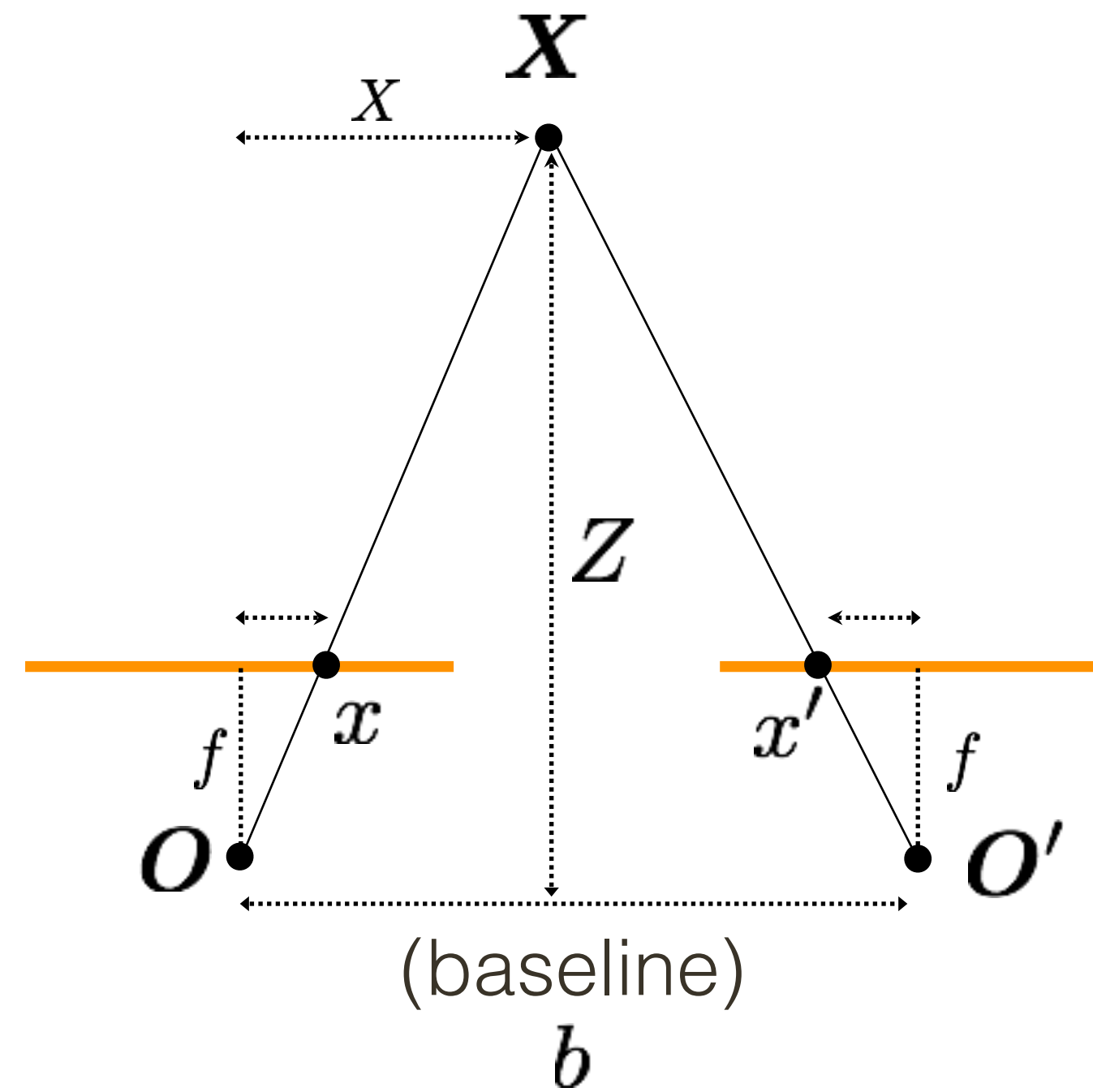
Rectified Stereo Pair: Depth Estimate



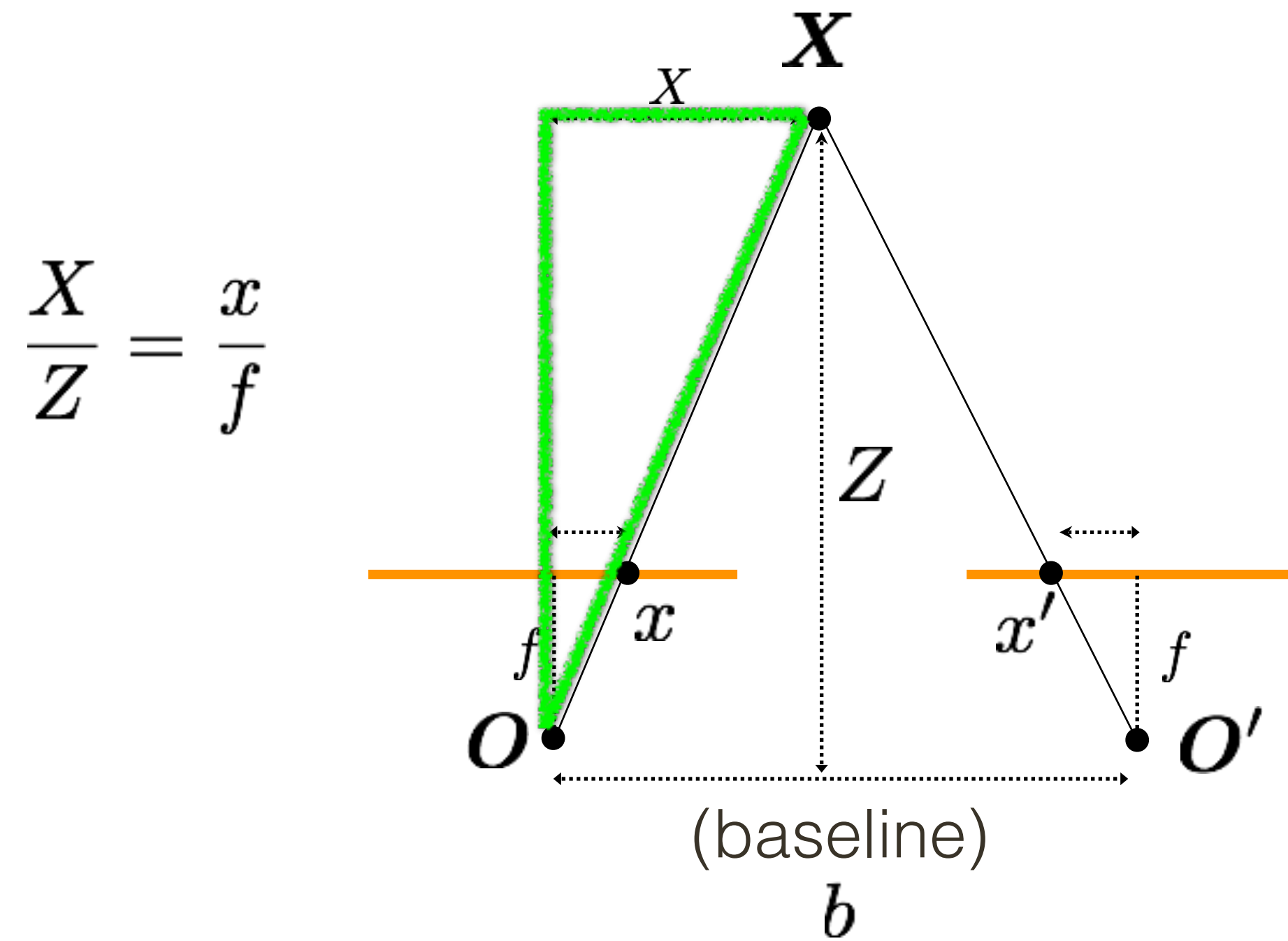
Rectified Stereo Pair: Depth Estimate



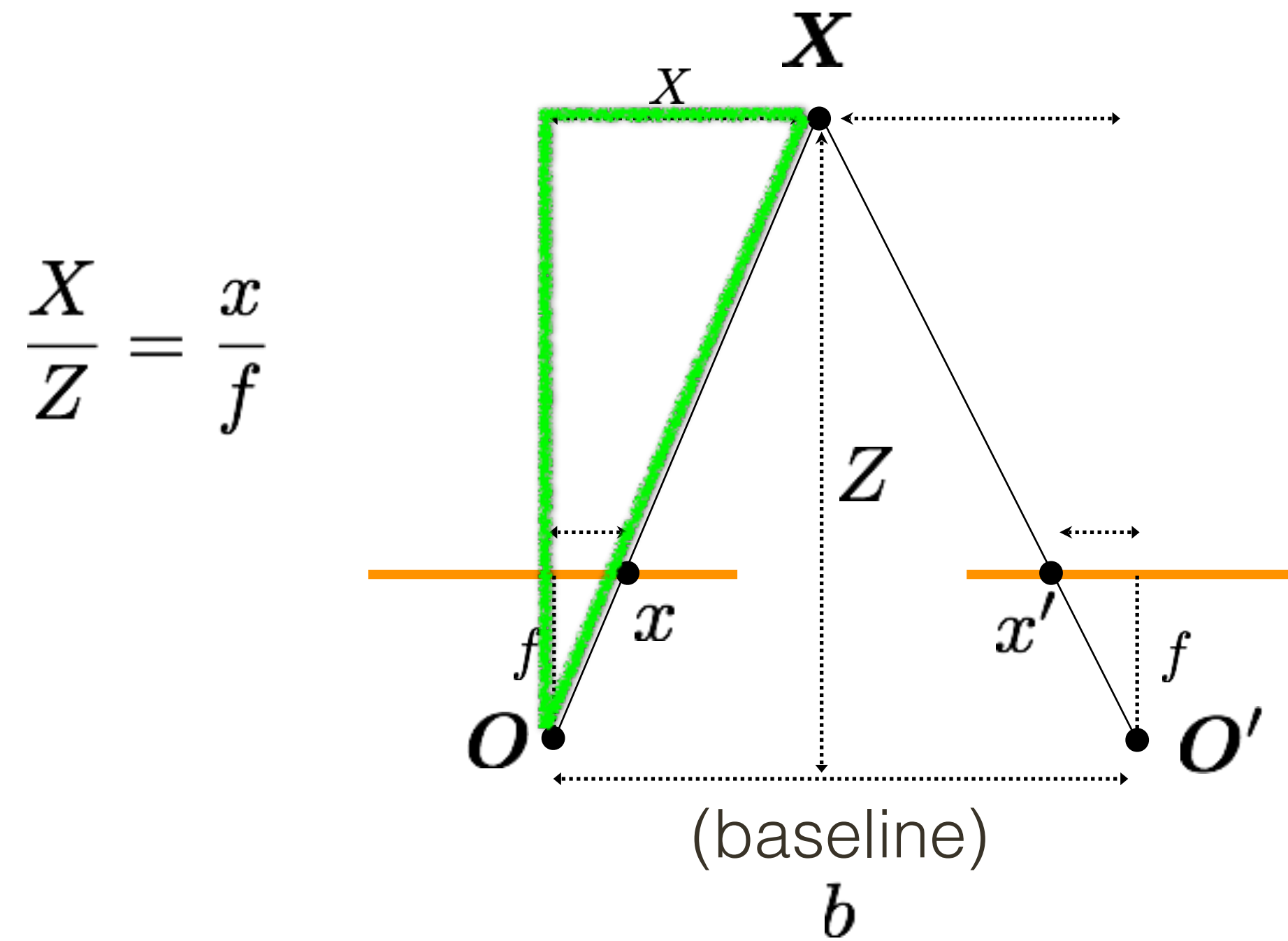
Rectified Stereo Pair: Depth Estimate



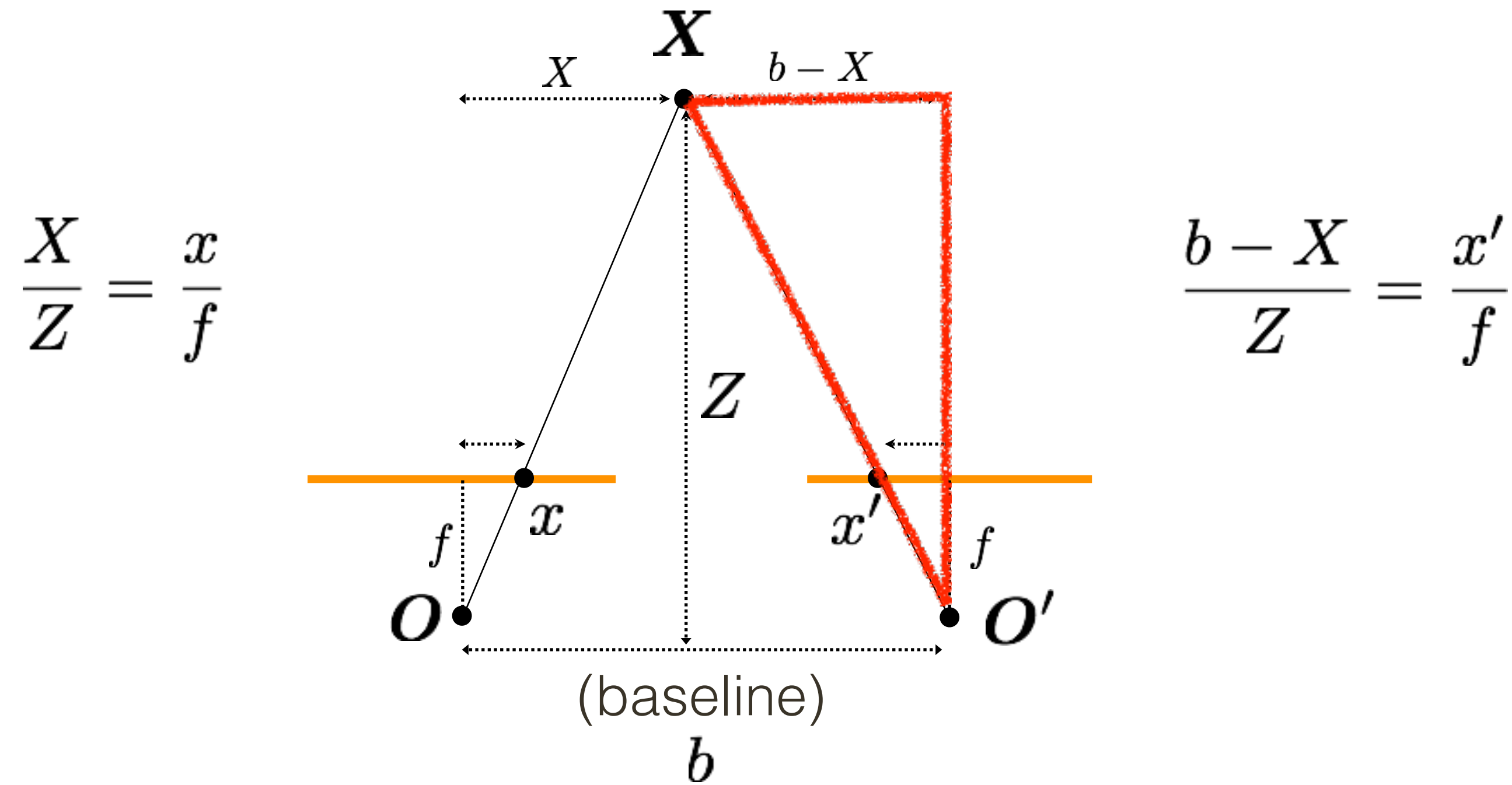
Rectified Stereo Pair: Depth Estimate



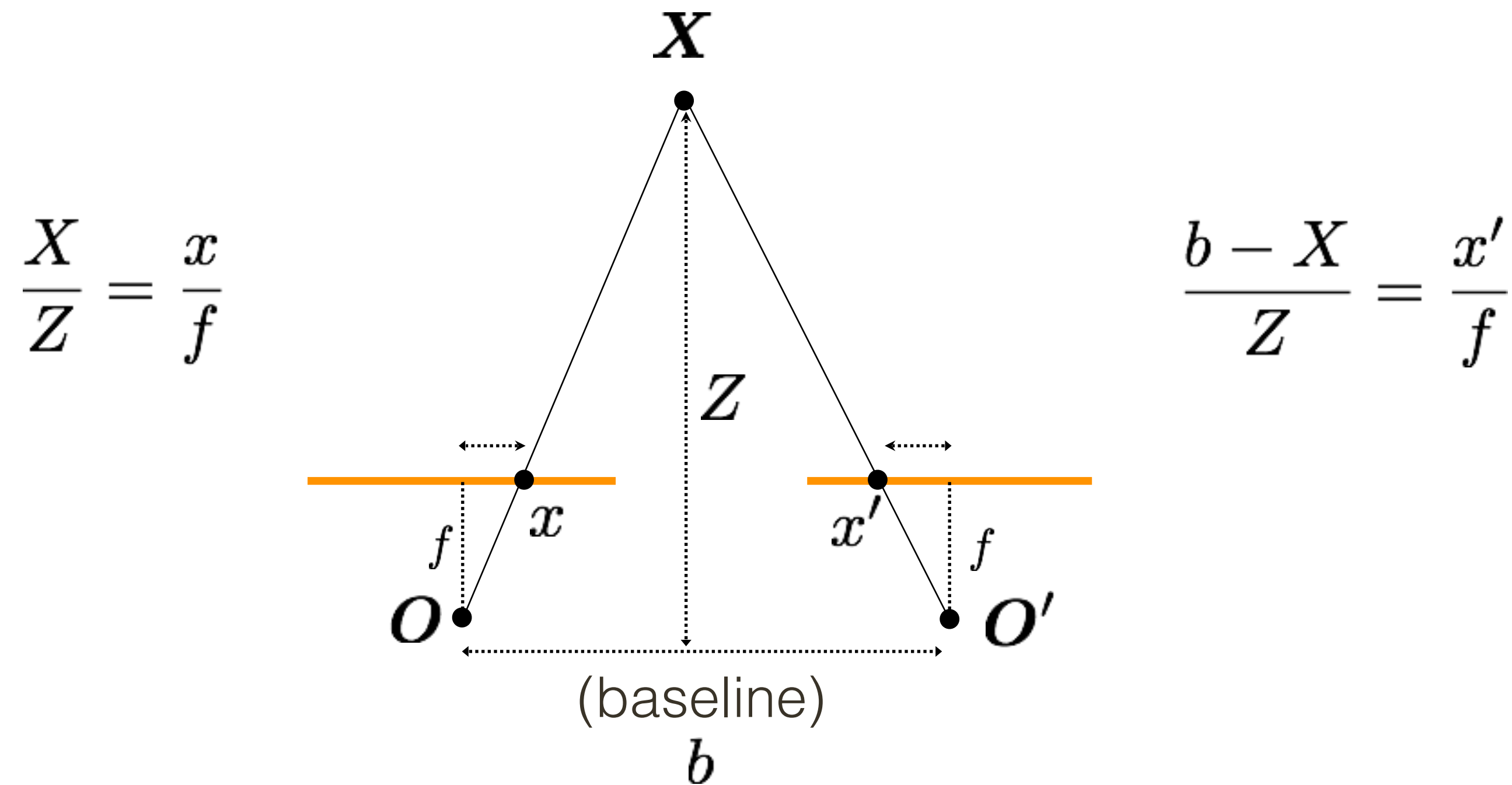
Rectified Stereo Pair: Depth Estimate



Rectified Stereo Pair: Depth Estimate



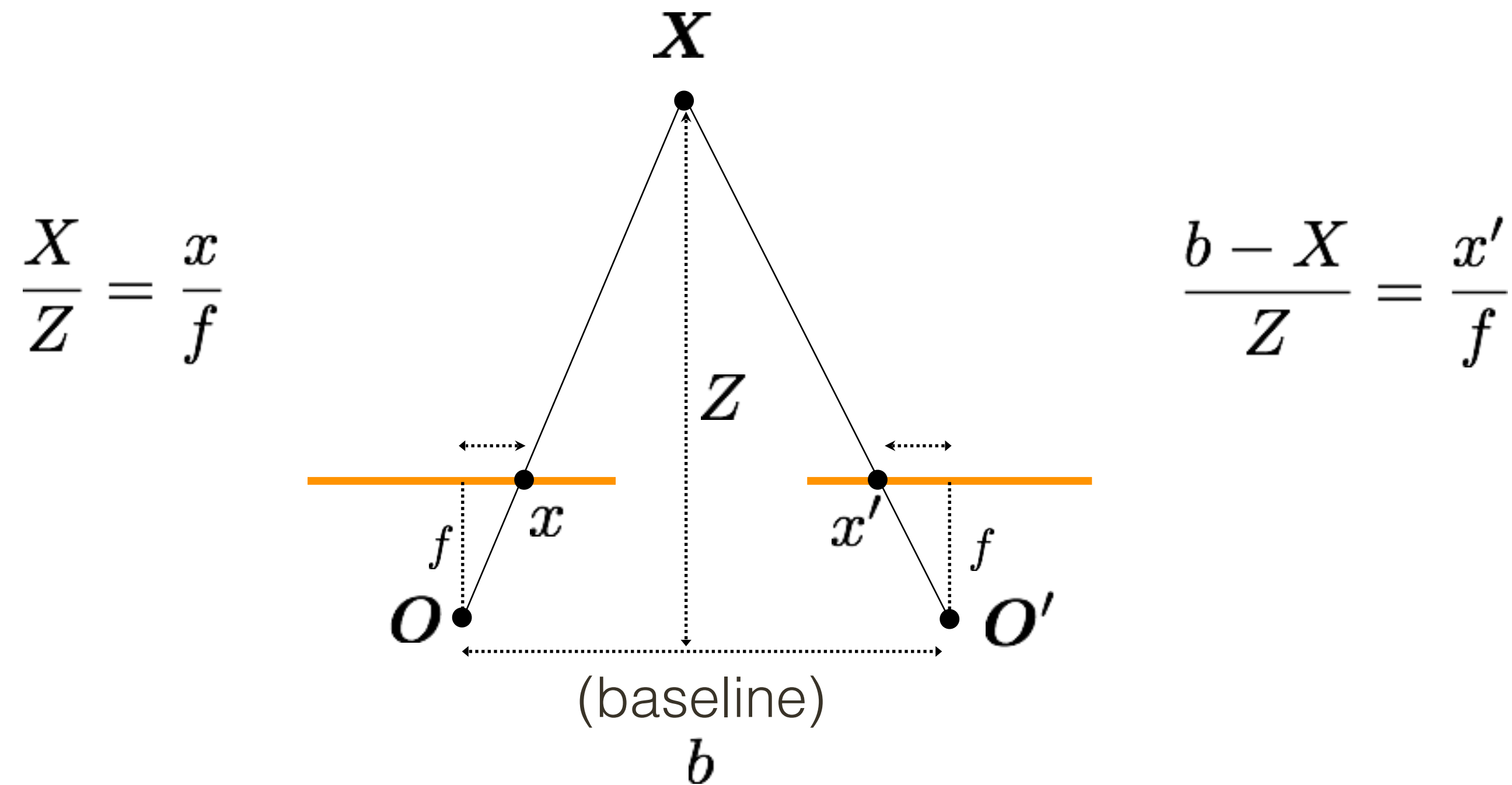
Rectified Stereo Pair: Depth Estimate



Disparity

$$d = x - x' \quad (\text{wrt to camera origin of image plane})$$
$$= \frac{bf}{Z}$$

Rectified Stereo Pair: Depth Estimate



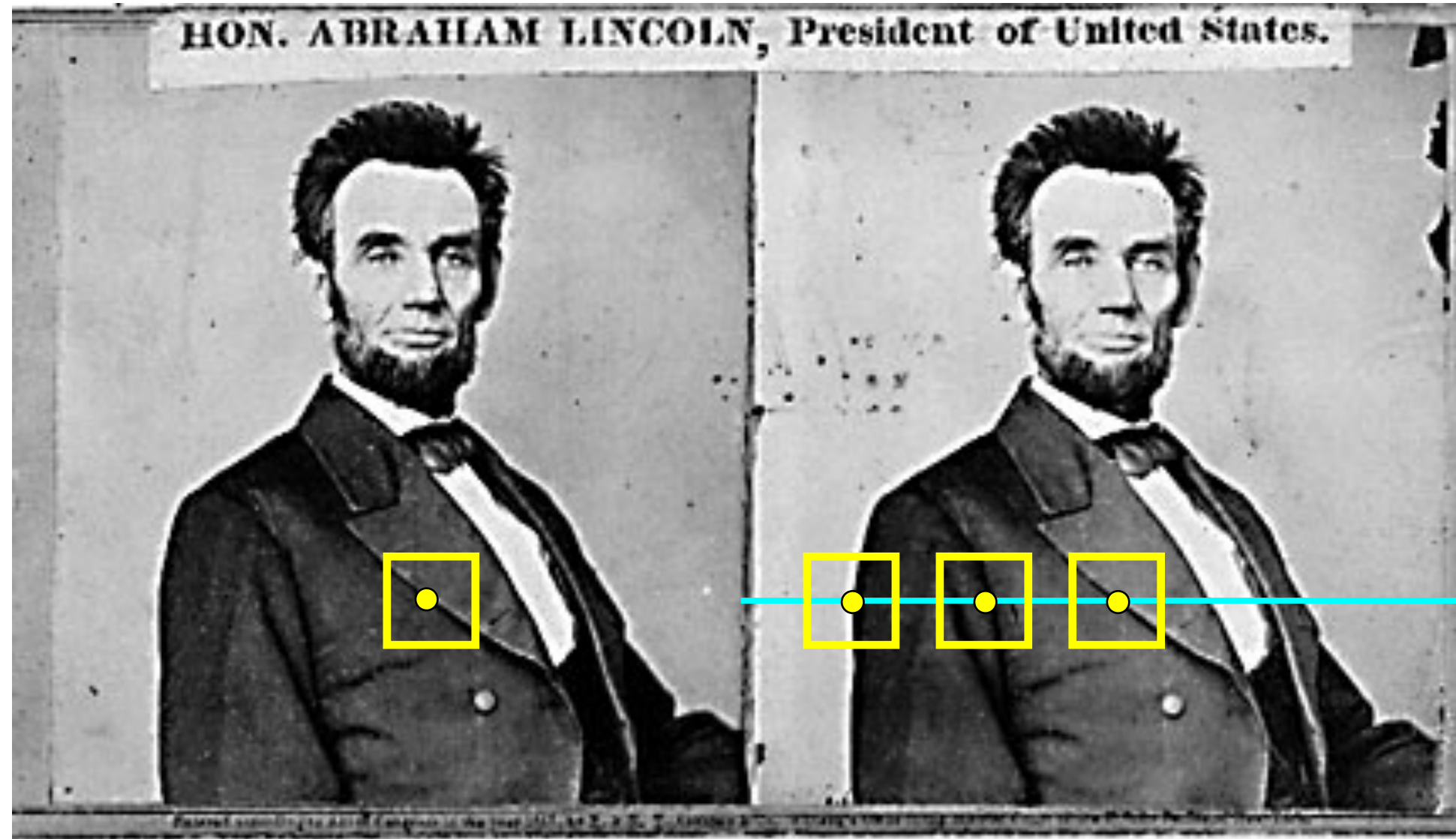
Disparity

$$d = x - x'$$

inversely proportional to depth

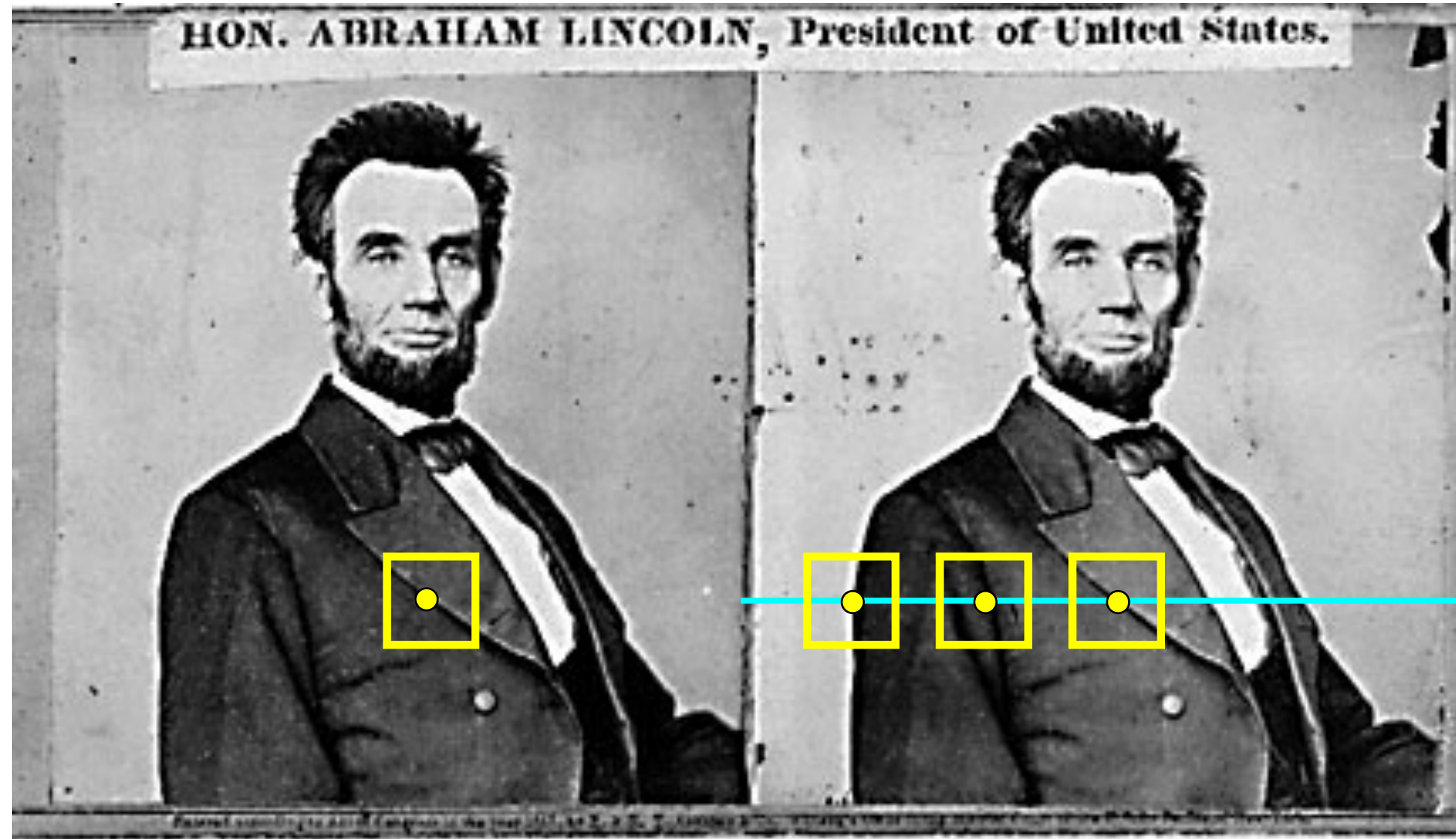
$$= \frac{bf}{Z}$$

(simple) **Stereo** Algorithm



1. Rectify images
(make epipolar lines horizontal)
2. For each pixel
 - a. Find epipolar line
 - b. Scan line for best match
 - c. Compute depth from disparity $Z = \frac{bf}{d}$

(simple) Stereo Algorithm



1. Rectify images
(make epipolar lines horizontal)
2. For each pixel
 - a. Find epipolar line
 - b. Scan line for best match
 - c. Compute depth from disparity $Z = \frac{bf}{d}$

Correspondence: What should we match?

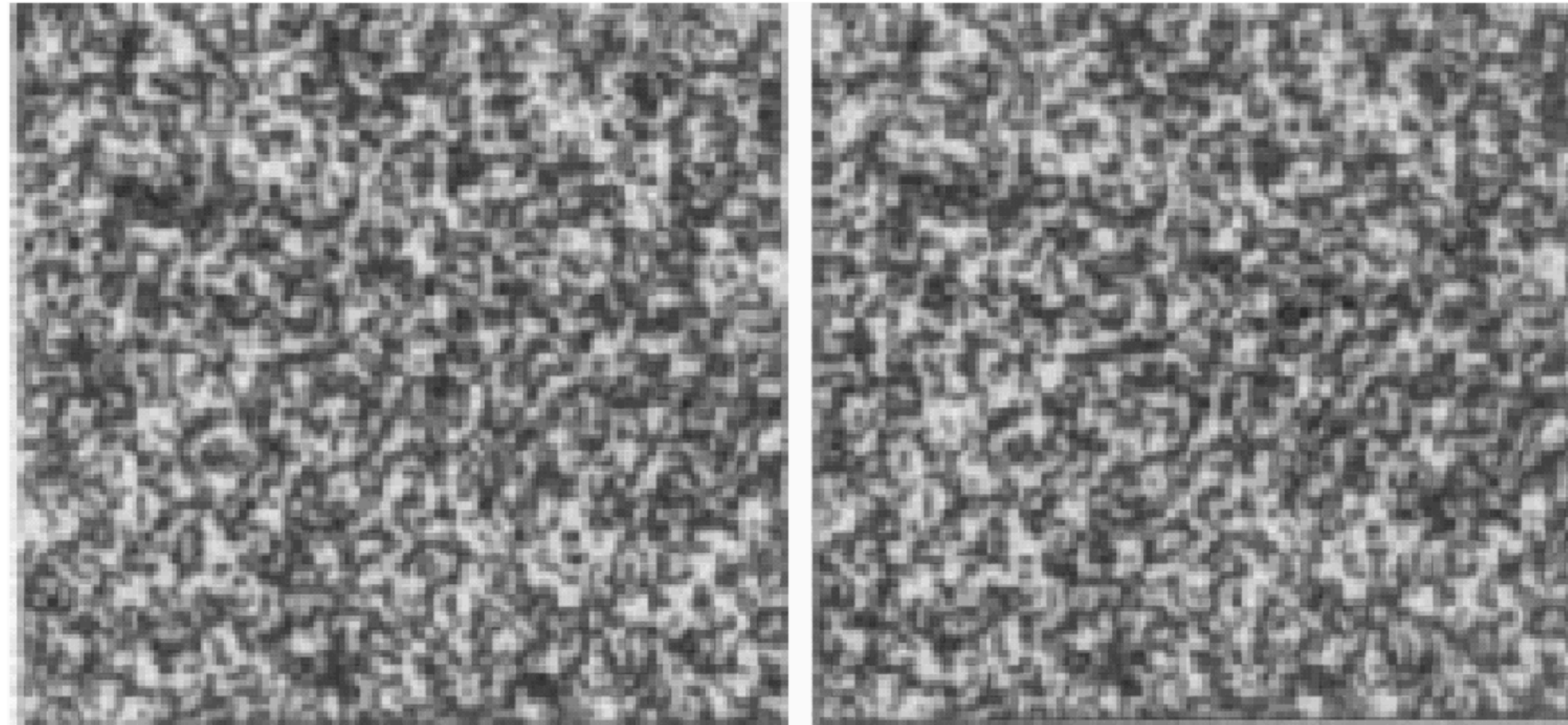
Objects?

Edges?

Pixels?

Collections of pixels?

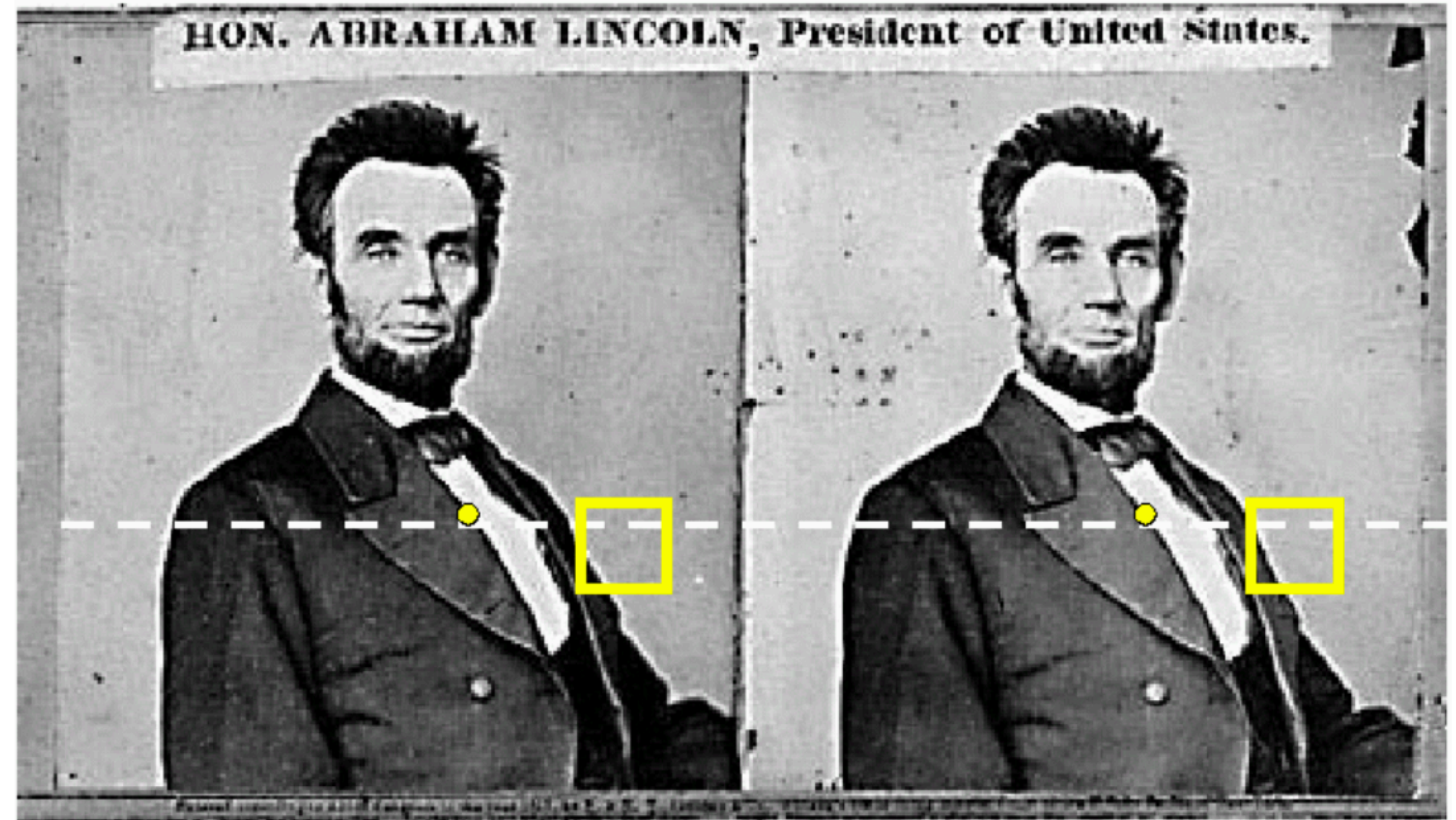
Random Dot Stereograms



Julesz (1960) showed that **recognition is not needed** for stereo

"When viewed monocularly, the images appear completely random. But when viewed stereoscopically, the image pair gives the impression of a square markedly in front of (or behind) the surround."

Method: Pixel Matching



For each **epipolar line**

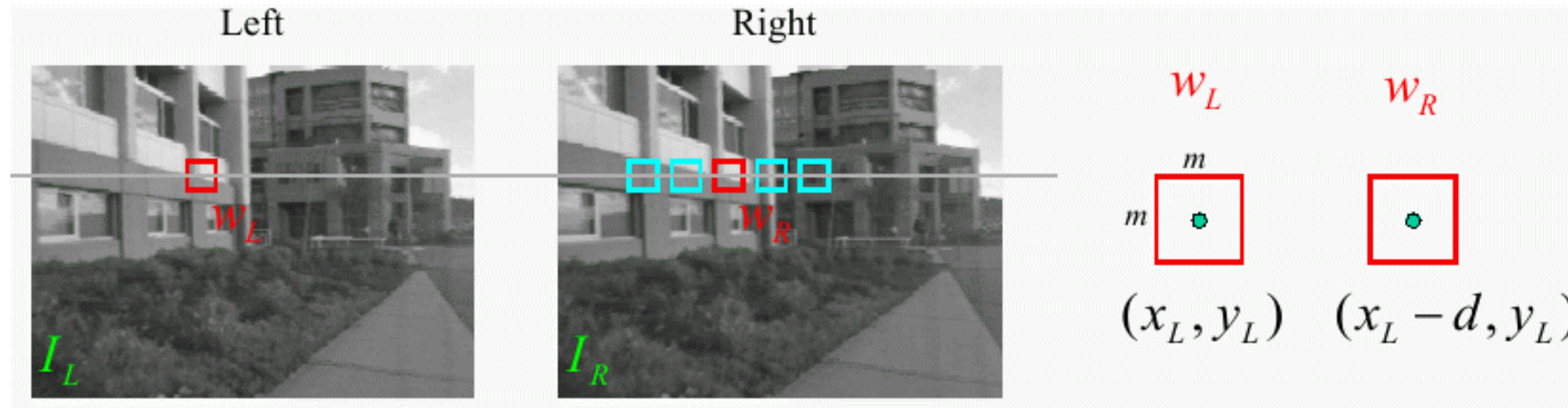
For each **pixel** in the left image

- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

This leaves too much ambiguity!

Slide credit: Steve Seitz

Sum of Squared (Pixel) Differences



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

Define the window function, $\mathbf{W}_m(x, y)$, by

$$\mathbf{W}_m(x, y) = \left\{ (u, v) \mid x - \frac{m}{2} \leq u \leq x + \frac{m}{2}, y - \frac{m}{2} \leq v \leq y + \frac{m}{2} \right\}$$

SSD measures intensity difference as a function of disparity:

$$C_R(x, y, d) = \sum_{(u, v) \in \mathbf{W}_m(x, y)} [I_L(u, v) - I_R(u - d, v)]^2$$

Image Normalization

$$\bar{I} = \frac{1}{|\mathbf{W}_m(x, y)|} \sum_{(u, v) \in \mathbf{W}_m(x, y)} I(u, v)$$

Average Pixel

$$||I||_{\mathbf{W}_m(x, y)} = \sqrt{\sum_{(u, v) \in \mathbf{W}_m(x, y)} [I(u, v)]^2}$$

Window Magnitude

$$\hat{I}(x, y) = \frac{I(x, y) - \bar{I}}{||I - \bar{I}||_{\mathbf{W}_m(x, y)}}$$

Normalized Pixel: subtract the mean, normalize to unit length

Image Metrics

(Normalized) Sum of Squared Differences

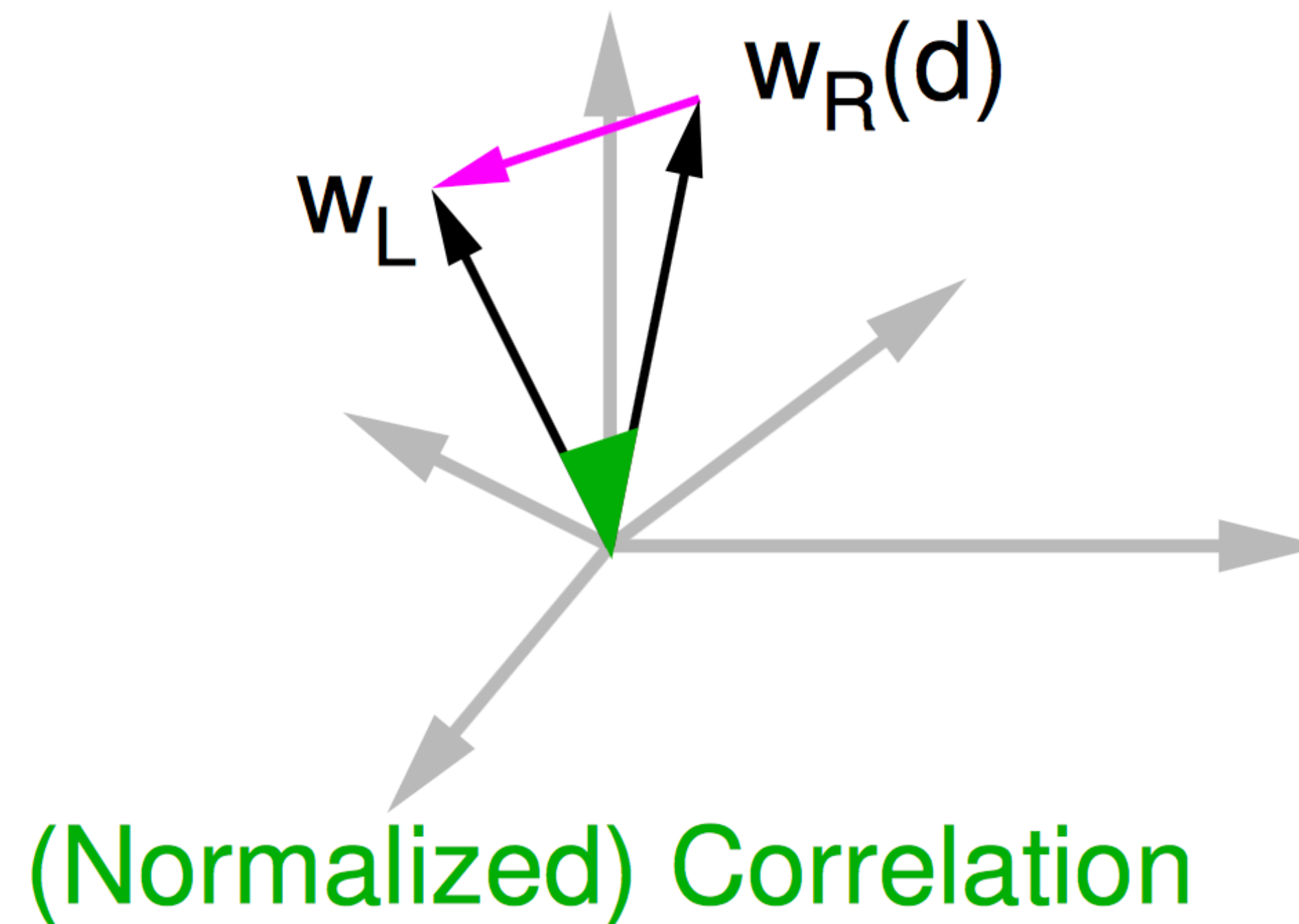


Image **Metrics**

Assume \mathbf{w}_L and $\mathbf{w}_R(d)$ are normalized to unit length (Normalized)

Sum of Squared Differences:

$$\begin{aligned} C_{SSD}(d) &= \sum_{(u,v) \in \mathbf{W}_m(x,y)} \left[\hat{I}_L(u,v) - \hat{I}_R(u-d,v) \right]^2 \\ &= ||\mathbf{w}_L - \mathbf{w}_R(d)||^2 \end{aligned}$$

(Normalized) **Correlation:**

$$\begin{aligned} C_{NC}(d) &= \sum_{(u,v) \in \mathbf{W}_m(x,y)} \hat{I}_L(u,v) \hat{I}_R(u-d,v) \\ &= \mathbf{w}_L \cdot \mathbf{w}_R(d) = \cos \theta \end{aligned}$$

Image **Metrics**

Let d^* be the value of d that minimizes C_{SSD}

Then d^* also is the value of d that minimizes C_{NC}

That is,

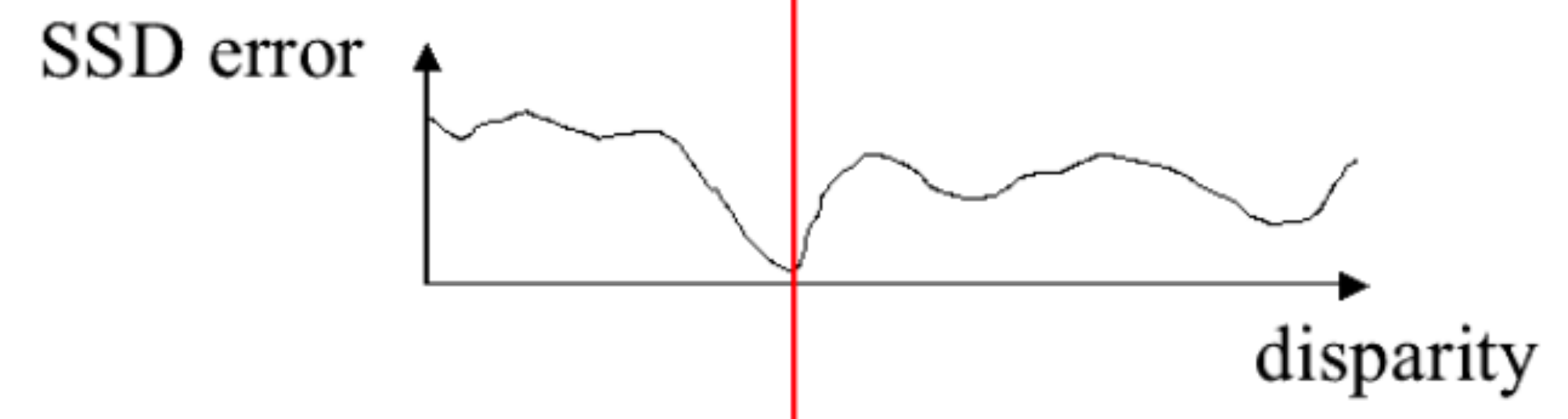
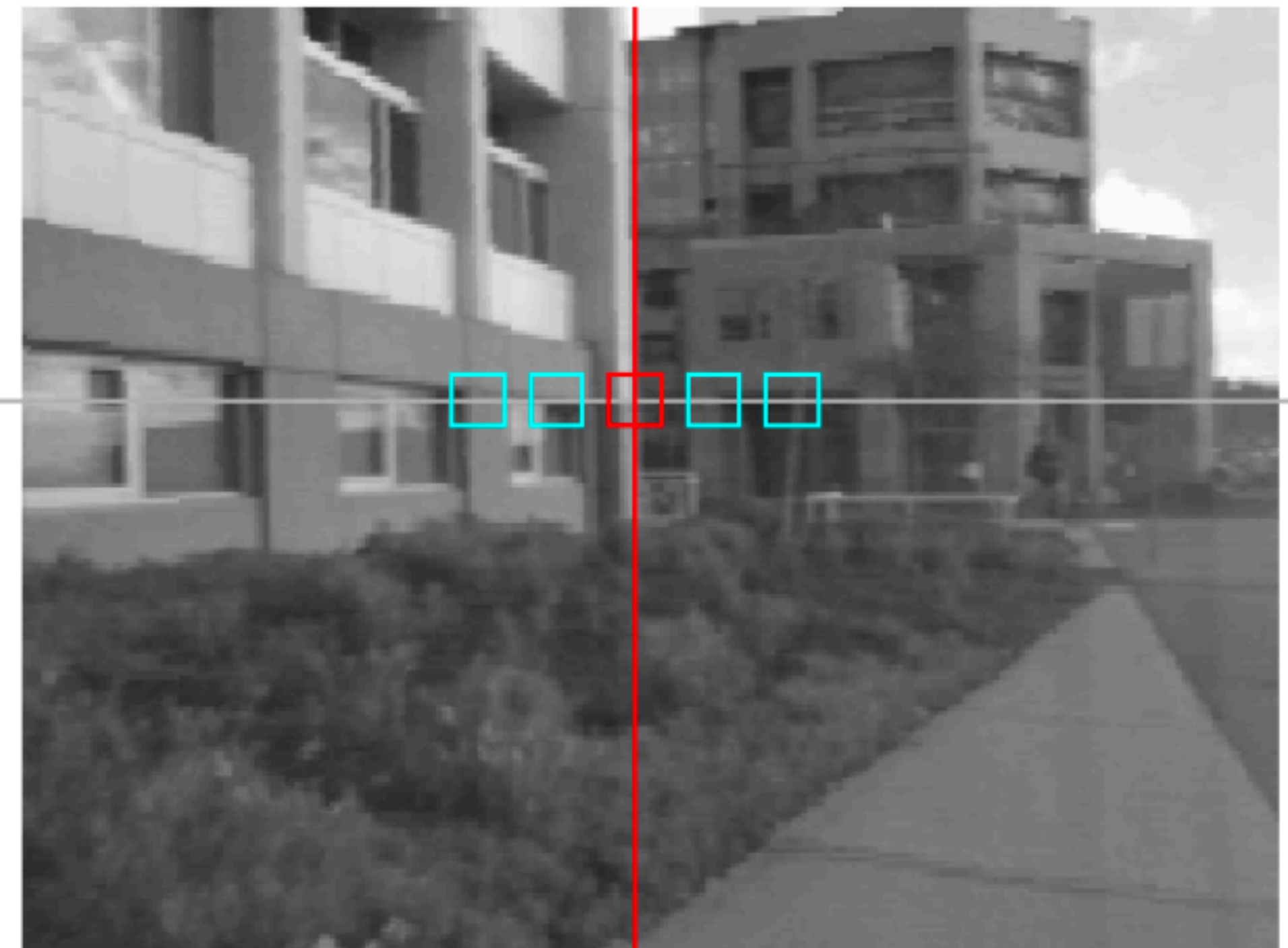
$$d^* = \arg \min_d ||\mathbf{w}_L - \mathbf{w}_R(d)||^2 = \arg \min_d \mathbf{w}_L \cdot \mathbf{w}_R(d)$$

Method: Correlation

Left



Right



Similarity Measure

Sum of Absolute Differences (SAD)

Sum of Squared Differences (SSD)

Zero-mean SAD

Locally scaled SAD

Normalized Cross Correlation (NCC)

Formula

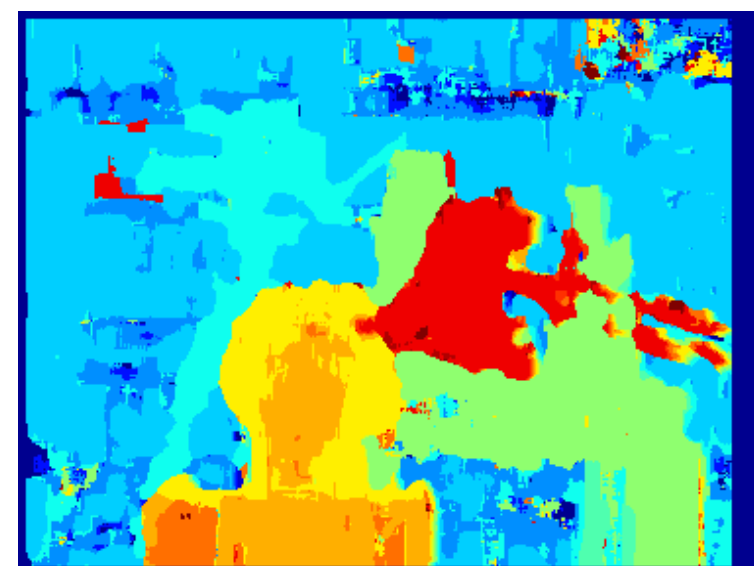
$$\sum_{(i,j) \in W} |I_1(i,j) - I_2(x+i, y+j)|$$

$$\sum_{(i,j) \in W} (I_1(i,j) - I_2(x+i, y+j))^2$$

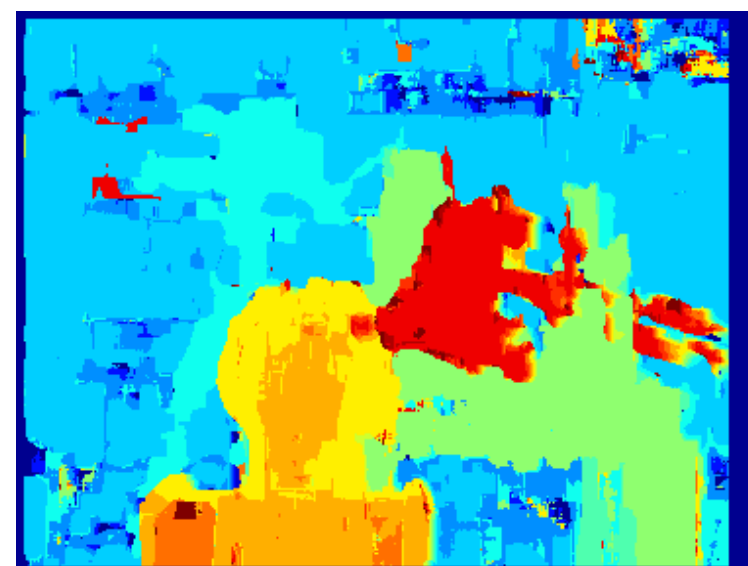
$$\sum_{(i,j) \in W} |I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i, y+j) + \bar{I}_2(x+i, y+j)|$$

$$\sum_{(i,j) \in W} |I_1(i,j) - \frac{\bar{I}_1(i,j)}{\bar{I}_2(x+i, y+j)} I_2(x+i, y+j)|$$

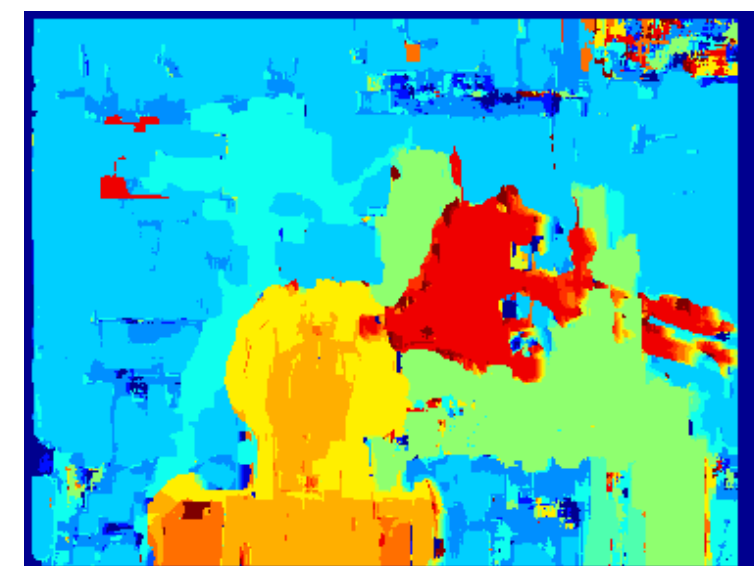
$$\frac{\sum_{(i,j) \in W} I_1(i,j) \cdot I_2(x+i, y+j)}{\sqrt{\sum_{(i,j) \in W} I_1^2(i,j) \cdot \sum_{(i,j) \in W} I_2^2(x+i, y+j)}}$$



SAD



SSD



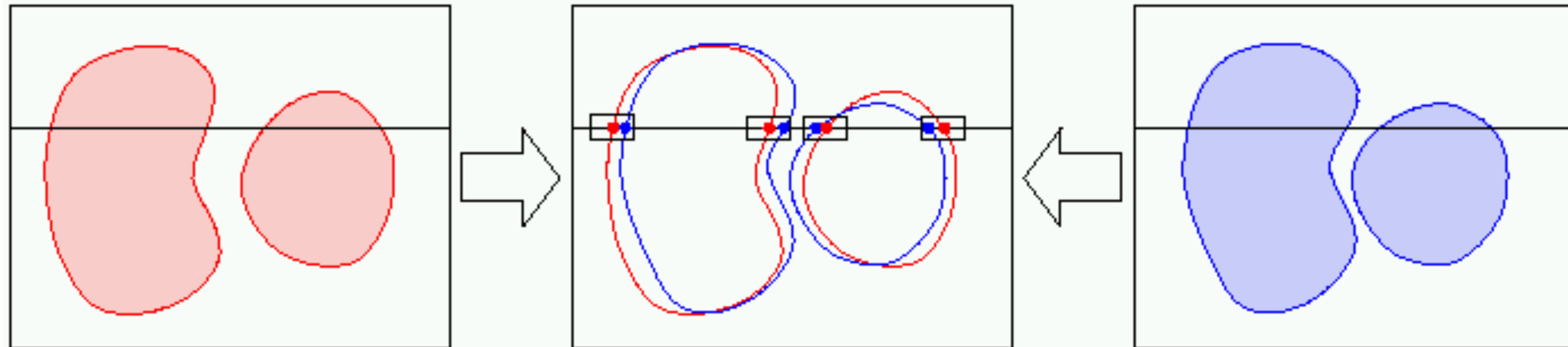
NCC



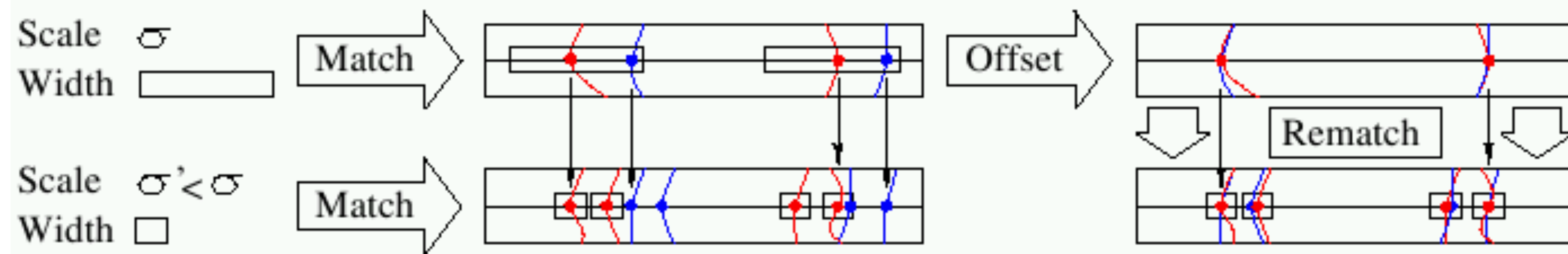
Ground truth

Method: Edges

Matching zero-crossings at a single scale



Matching zero-crossings at multiple scales



Forsyth & Ponce (2nd ed.) Figure 7.12 (Top & Middle)

Method: Edges (aside)

The **Marr/Poggio** (1979) multiscale stereo algorithm:

1. Convolve the two (rectified) images with $\nabla^2 G_\sigma$ filters of increasing $\sigma_1 < \sigma_2 < \sigma_3 < \sigma_4$
2. Find zero crossings along horizontal scanlines of the filtered images
3. For each filter scale σ , match zero crossings with the same parity and roughly equal orientations in a $[-\mathbf{w}_\sigma, +\mathbf{w}_\sigma]$ disparity range, with $\mathbf{w}_\sigma = 2\sqrt{2}\sigma$
4. Use the disparities found at larger scales to control eye vergence and cause unmatched regions at smaller scales to come into correspondence

Which Method is **Better**: Correlation or Edges?

Edges are more “meaningful” [Marr]. but hard to find!

Edges tend to fail in dense texture (outdoors)

Correlation tends to fail in smooth, featureless regions

Note: Correlation-based methods are “dense.” Edge-based methods are “relatively sparse”

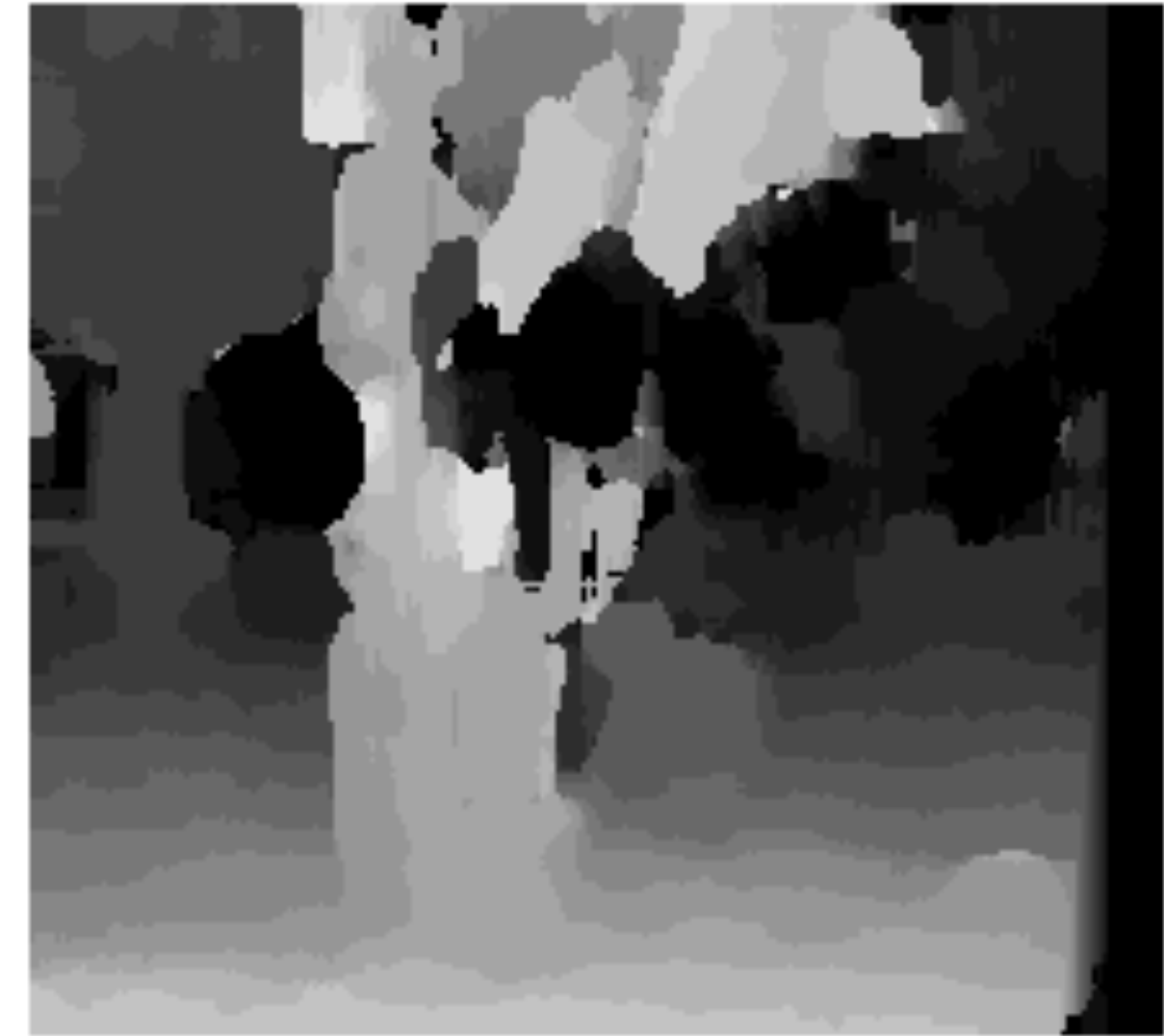
Effect of **Window Size**



$W = 3$

Smaller window

- + More detail
- More noise

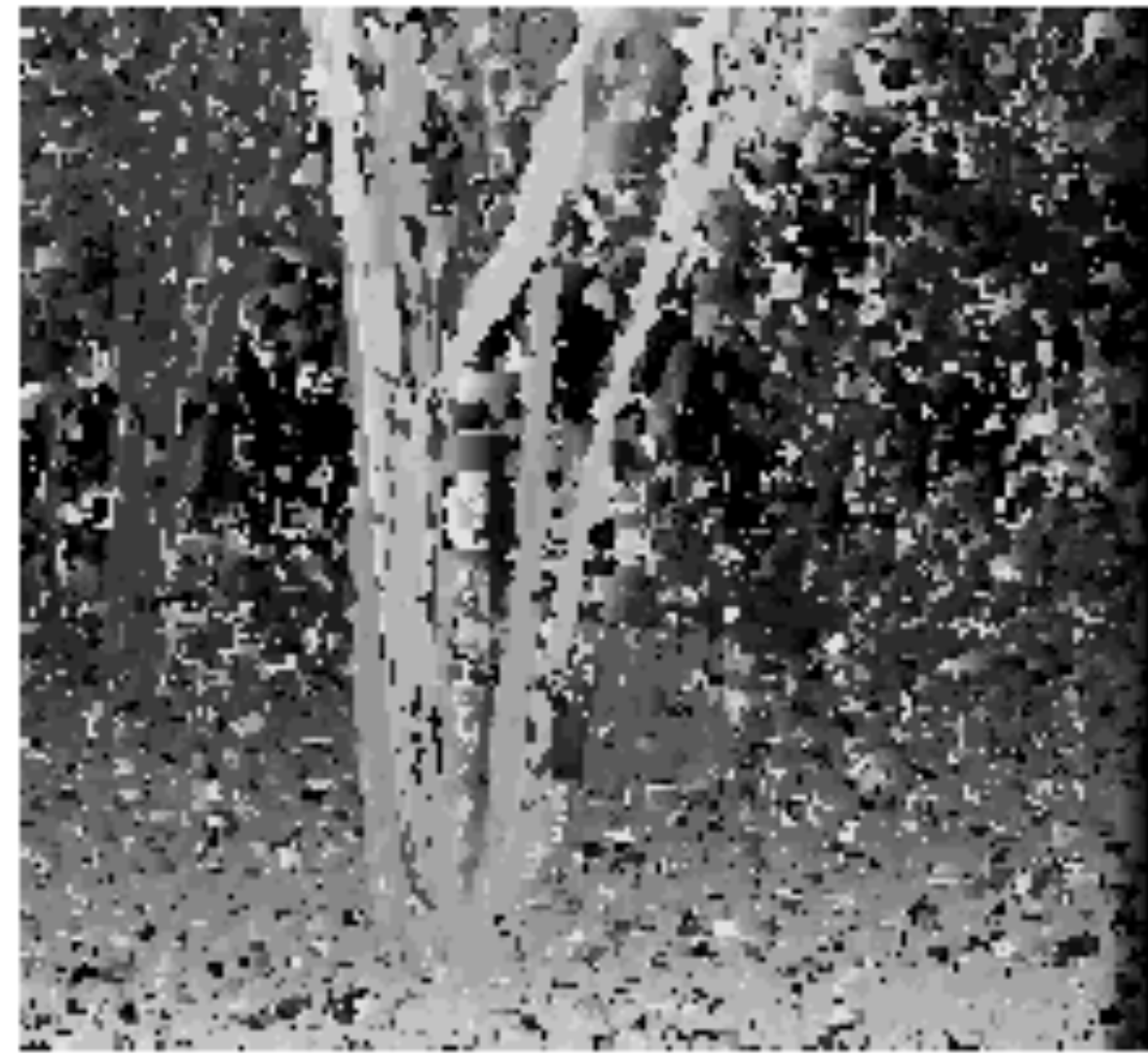


$W = 20$

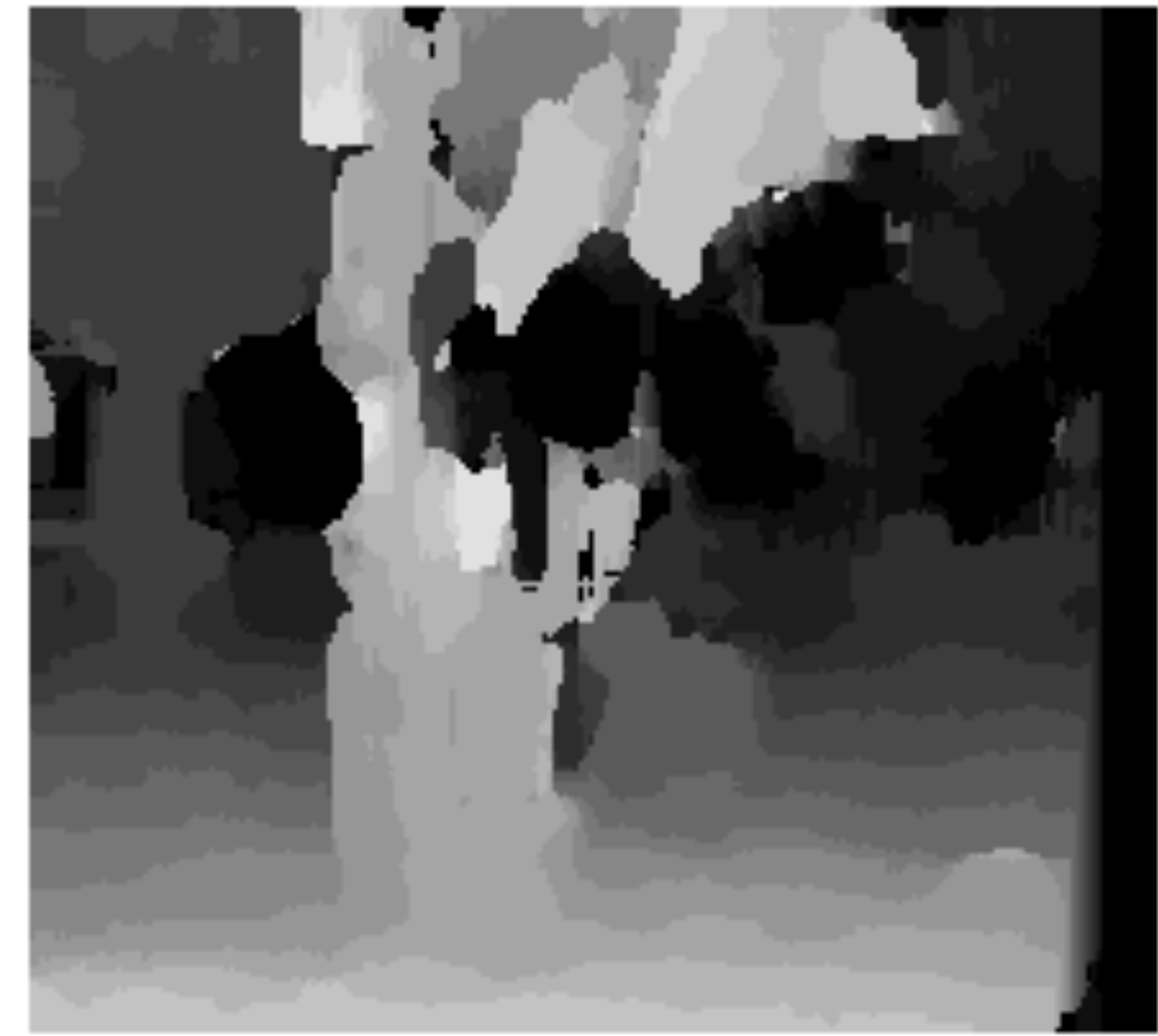
Larger window

- + Smoother disparity maps
- Less detail
- Fails near boundaries

Effect of **Window Size**



$W = 3$

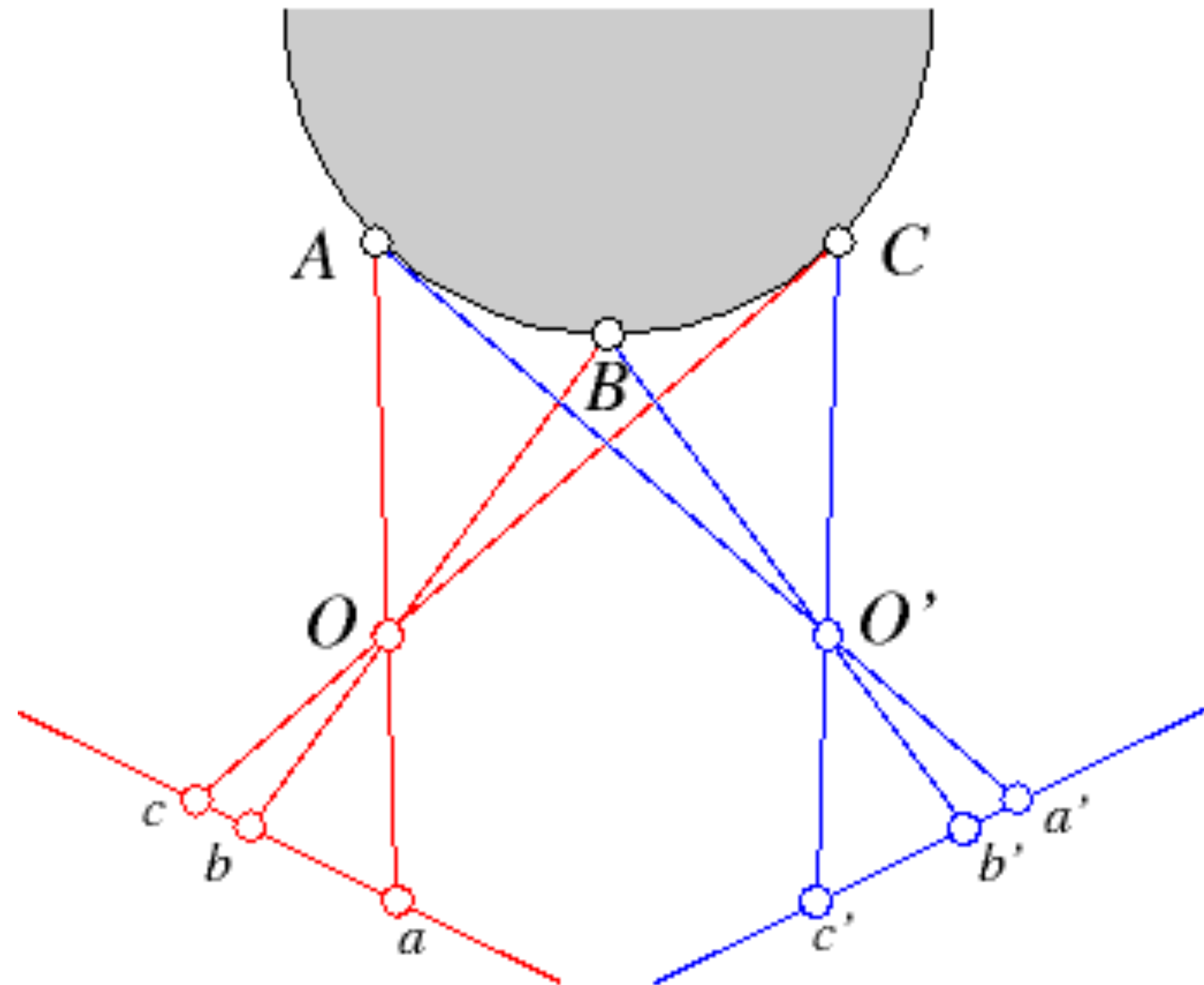


$W = 20$

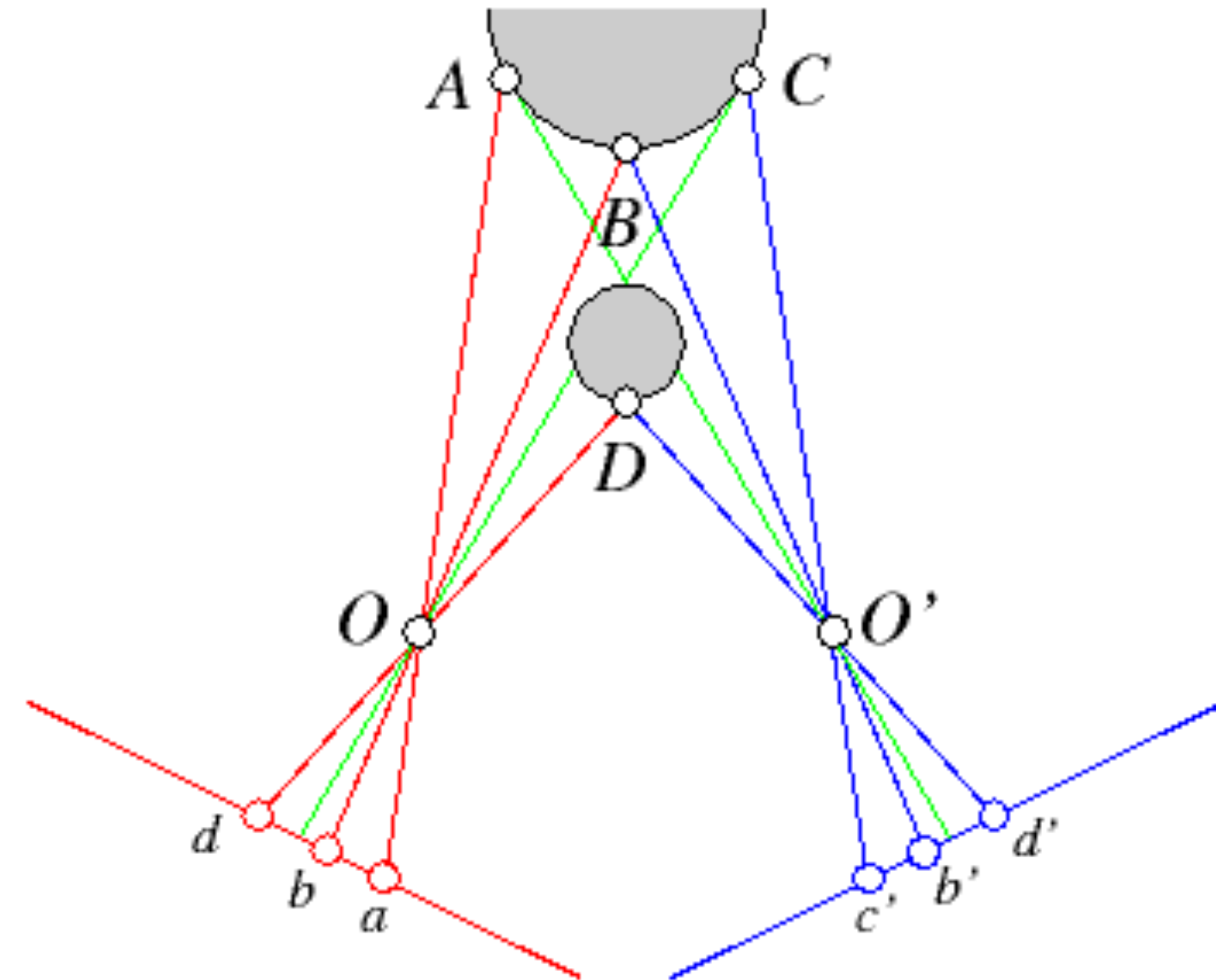
Note: Some approaches use an adaptive window size
— try multiple sizes and select best match

Ordering Constraints

Ordering constraint ...



.... and a **failure** case

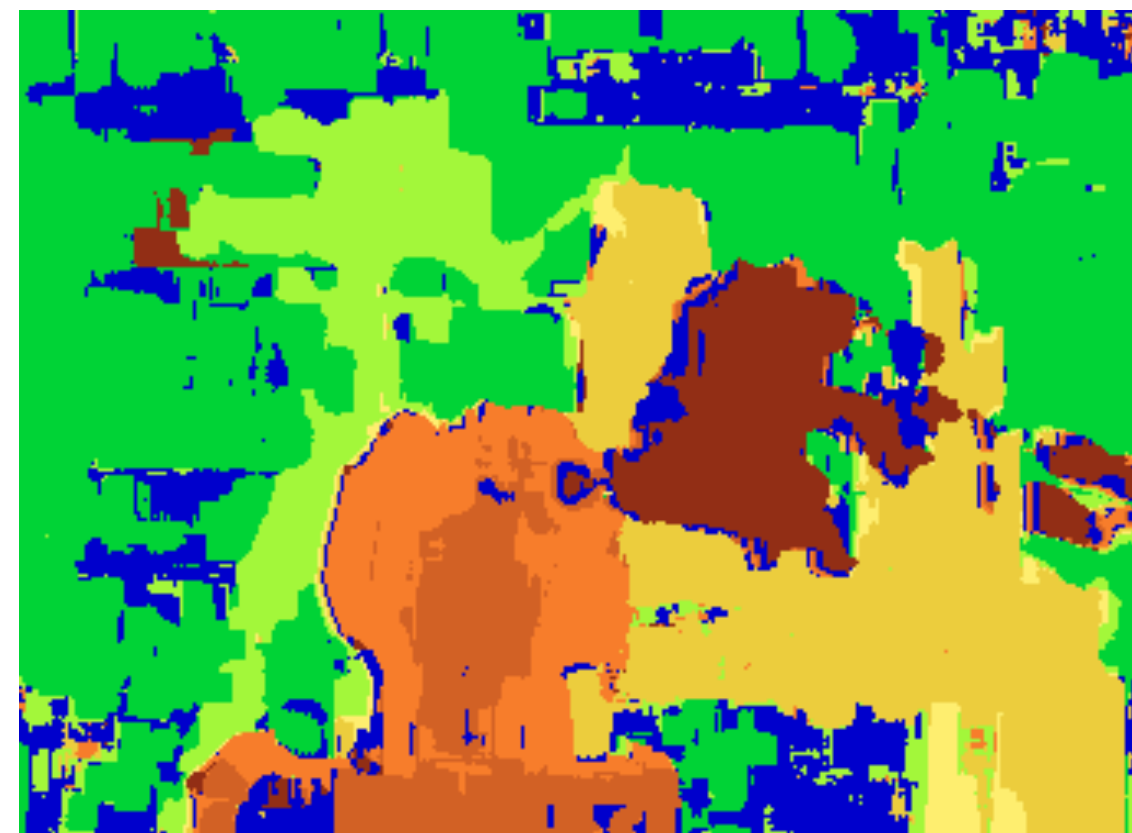


Forsyth & Ponce (2nd ed.) Figure 7.13

Block Matching Techniques: Result



Block matching



Ground truth



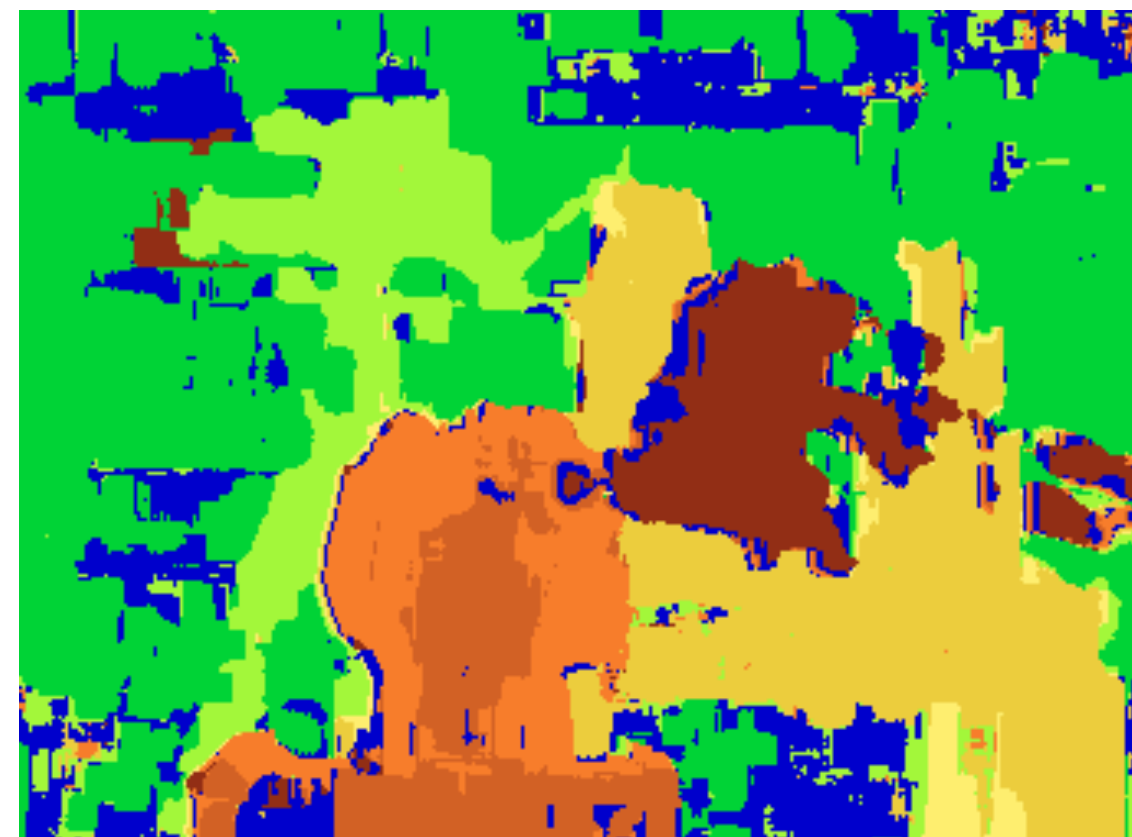
Block Matching Techniques: Result

Too many **discontinuities**.
We expect disparity values to
change slowly.

Let's make an assumption:
depth should change smoothly



Block matching



Ground truth

