THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision

Image Credit: http://cannibal-eshafeeqe.blogspot.com/2014/03/surf-error-free-matching-using-ransac.html

Lecture 20: Object Recognition with SIFT, RANSAC

Menu for Today (**October 24, 2018**)

Topics:

— Object detection with SIFT

— Model fitting: RANSAC

Redings:

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

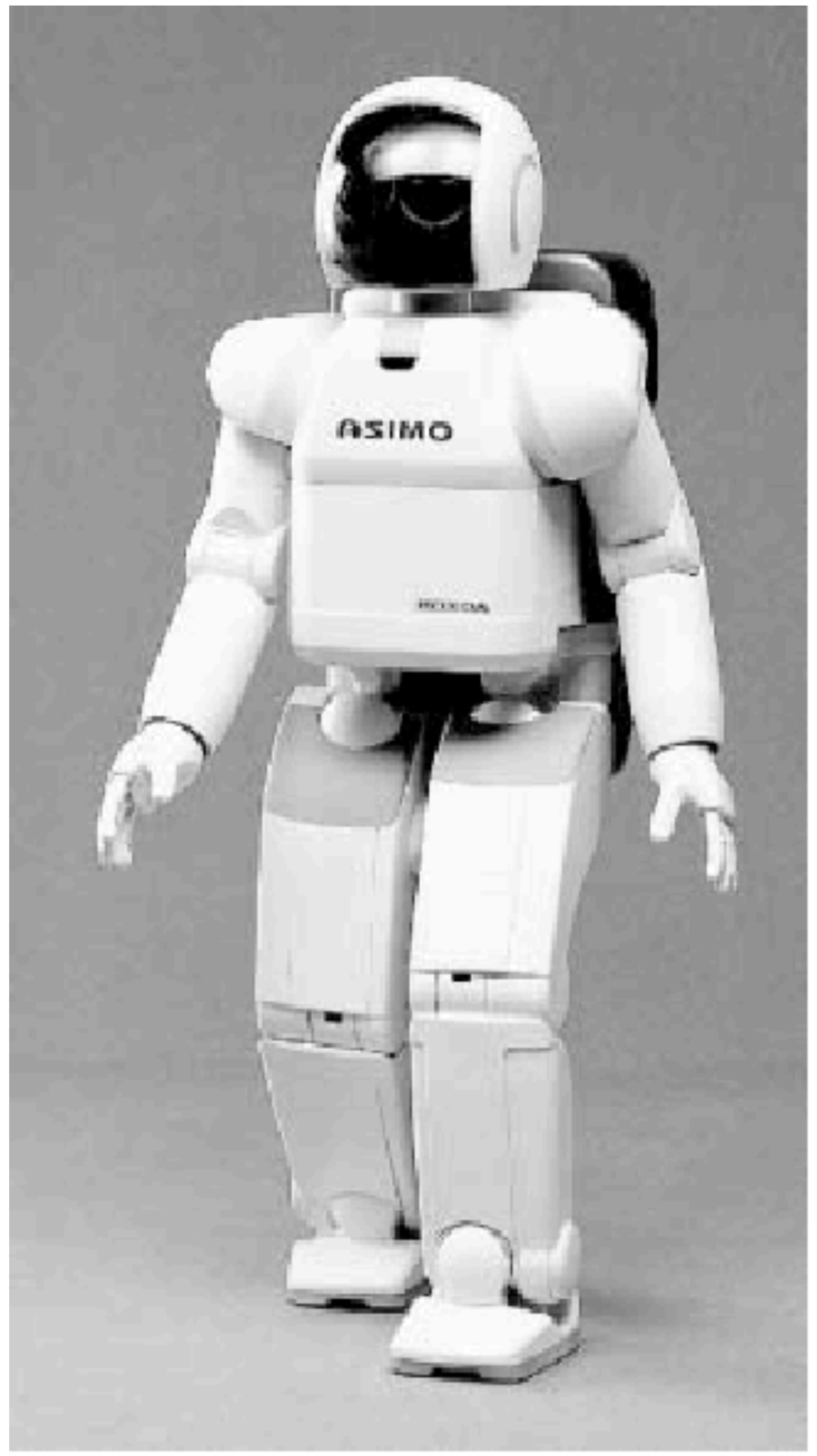
— **Next** Lecture: N/A

Reminders:

— **Assignment 3**: Texture Syntheis is **out,** due on **October 29th**

Today's "**fun**" Example: Honda's ASIMO Robot

Advanced Step in Innovative MObility (ASIMO) Humanoid Robot



height	1200 mm
depth	440 mm
width	450 mm
weight	43 kg
walking speed	0–1.6 km h^{-1}
walking cycle	cycle adjustable, stride adjustable
grasping force	0.5 kg/hand (5-finger hand)

actuator	servomotor + harmonic speed reducer + drive unit
control unit	walking/operating control unit wireless transmission unit
sensors	foot: 6-axis force sensor torso: gyroscope, acceleration and sensor
power system	38V/10AH (Ni-MH)
operating system	workstation and portable controller

Today's "**fun**" Example: Honda's ASIMO Robot

Today's "**fun**" Example: Boston Dynamics' Spot Mini

Lecture 19: Re-cap

Four steps to SIFT feature generation:

1. **Scale-space representation and local extrema detection**

2. **Keypoint localization**
   — select stable keypoints (threshold on magnitude of extremum, ratio of principal curvatures)

3. **Keypoint orientation assignment**

4. **Keypoint descriptor**
   — vector with $8 \times 4 \times 4 = 128$ dimensions

Lecture 19: Re-cap

Histogram of Oriented Gradients (**HoG**)

— Descriptor similar to SIFT

— Focuses on encoding oriented histogram of gradient magnitudes

— Redundant and high dimensional

SURF Descriptor

— Descriptor similar to SIFT

— Characterizes gradient by sums of raw and absolute gradient in x- and y-dir

— Much smaller in size and faster to compute

SIFT and **Object Recognition**

Object recognition requires us to first match each keypoint independently to the database of keypoints

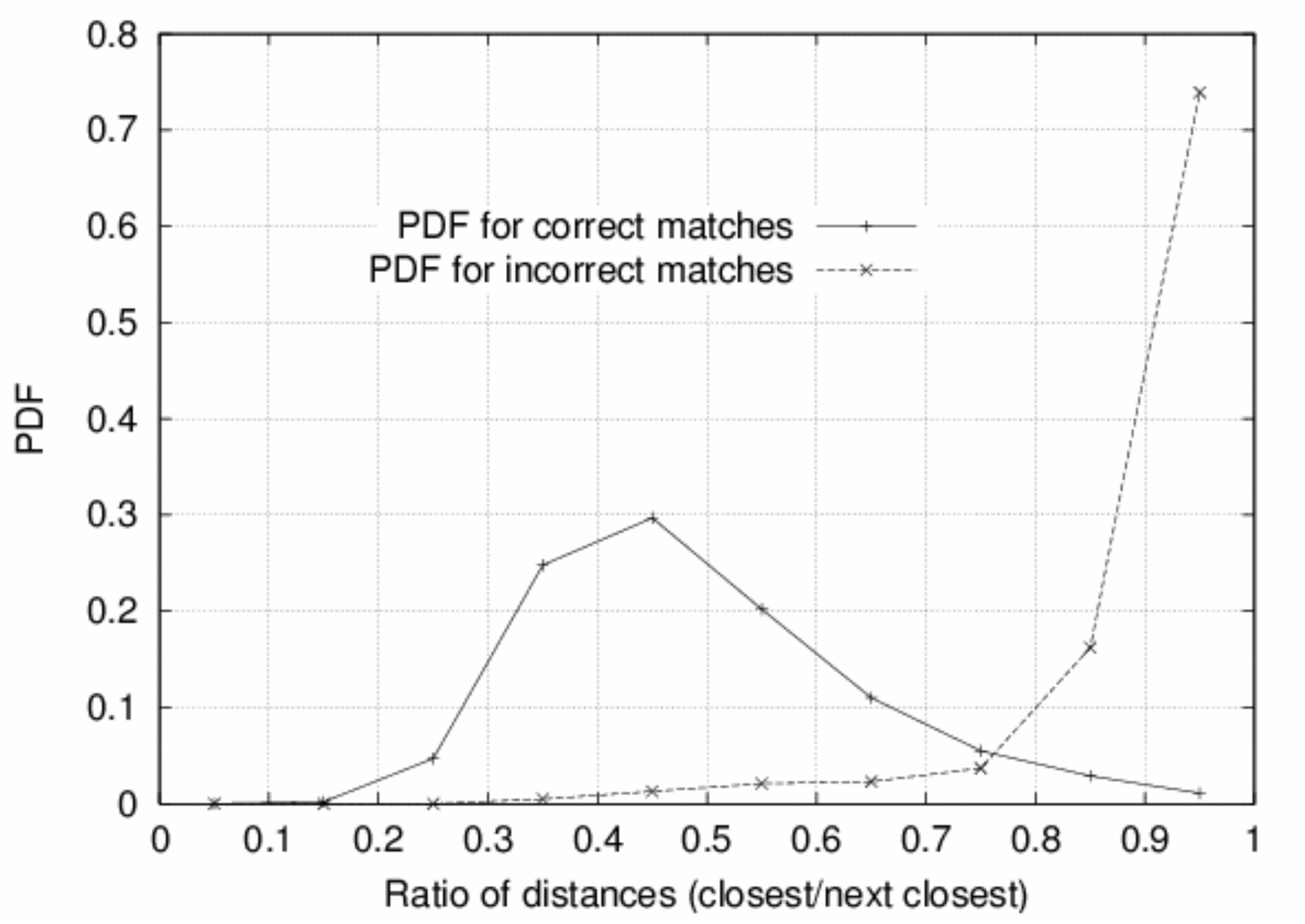
Many features will not have any correct match in the database because they arise from background clutter

It would be useful to have a way to **discard features** that do not have any good match

Probability of **Correct** Match

Compare ratio of distance of **nearest** neighbour to **second** nearest neighbour (from different object)

Threshold of 0.8 provides excellent separation



Nearest-Neighbor Matching to Feature Database

Hypotheses are generated by **approximate nearest neighbour** matching of each feature to vectors in the database

— Use best–bin–first (Beis & Lowe, 97) modification to k-d tree algorithm

— Use heap data structure to identify bins in order by their distance from query point

Result: Can give speedup by factor of 1,000 while finding nearest neighbour (of interest) 95% of the time

Identifying **Consistent** Features

We have matched keypoints to a database of known keypoints extracted from training images

Next we identify **clusters of at least 3 features** that agree on an object and its pose

— a typical image contains 2,000+ features → detecting less than 1% inliers among 99% outliers!

Lowe's solution uses the generalized **Hough transform**

— vote for each potential match according to model ID and pose

— insert into multiple bins to allow for error in similarity approximation

— (more on Hough transforms later)

Model **Verification**

1. Examine all clusters with at least 3 features

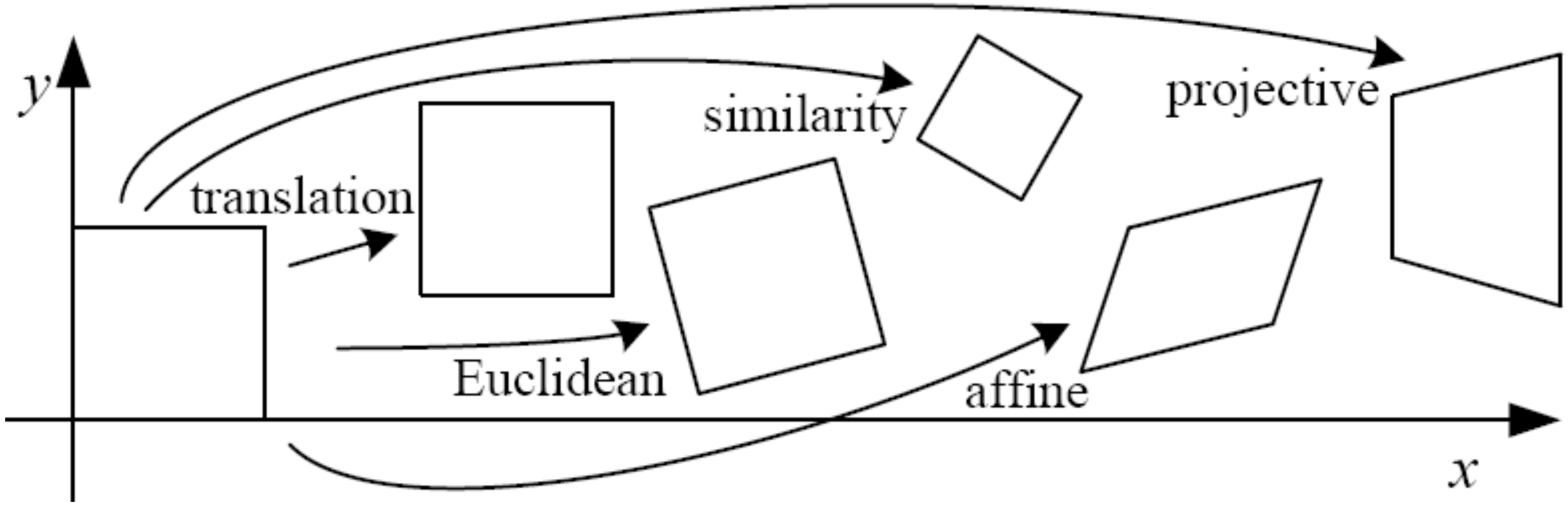
2. Perform least-squares affine **fit to model**

3. **Discard outliers** and perform top-down check for additional features

4. Evaluate probability that match is correct

   — Use Bayesian model, with probability that features would arise by chance if object was not present (Lowe, CVPR 01)

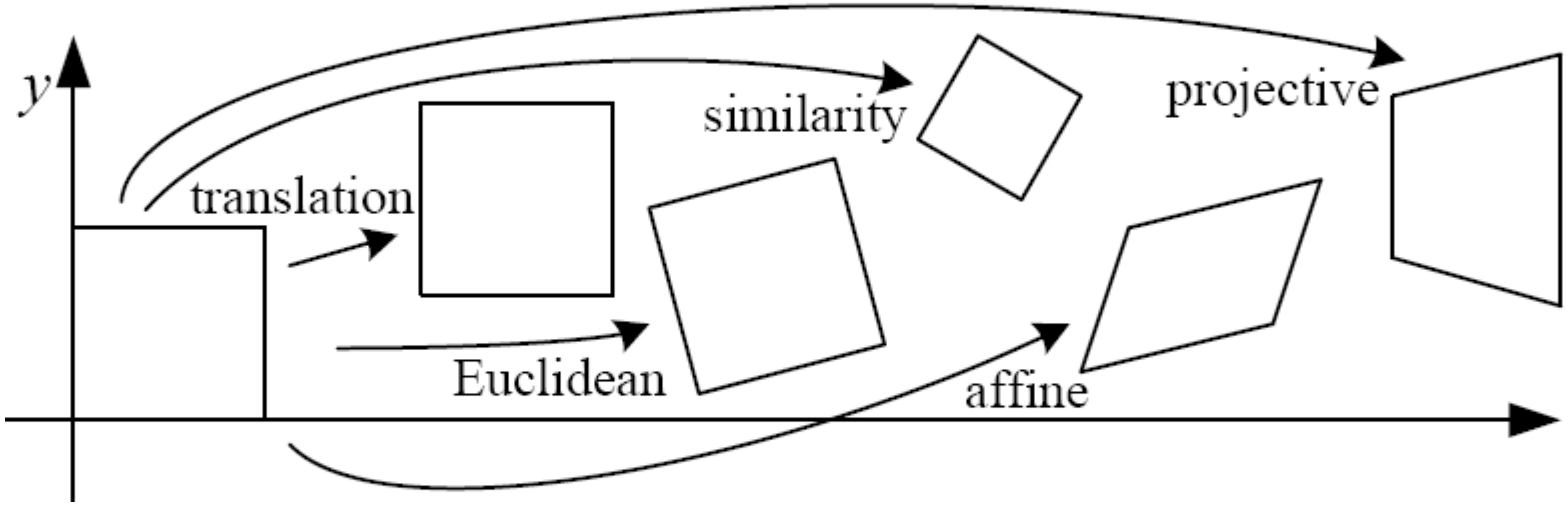
Aside: Classification of 2D Transformations



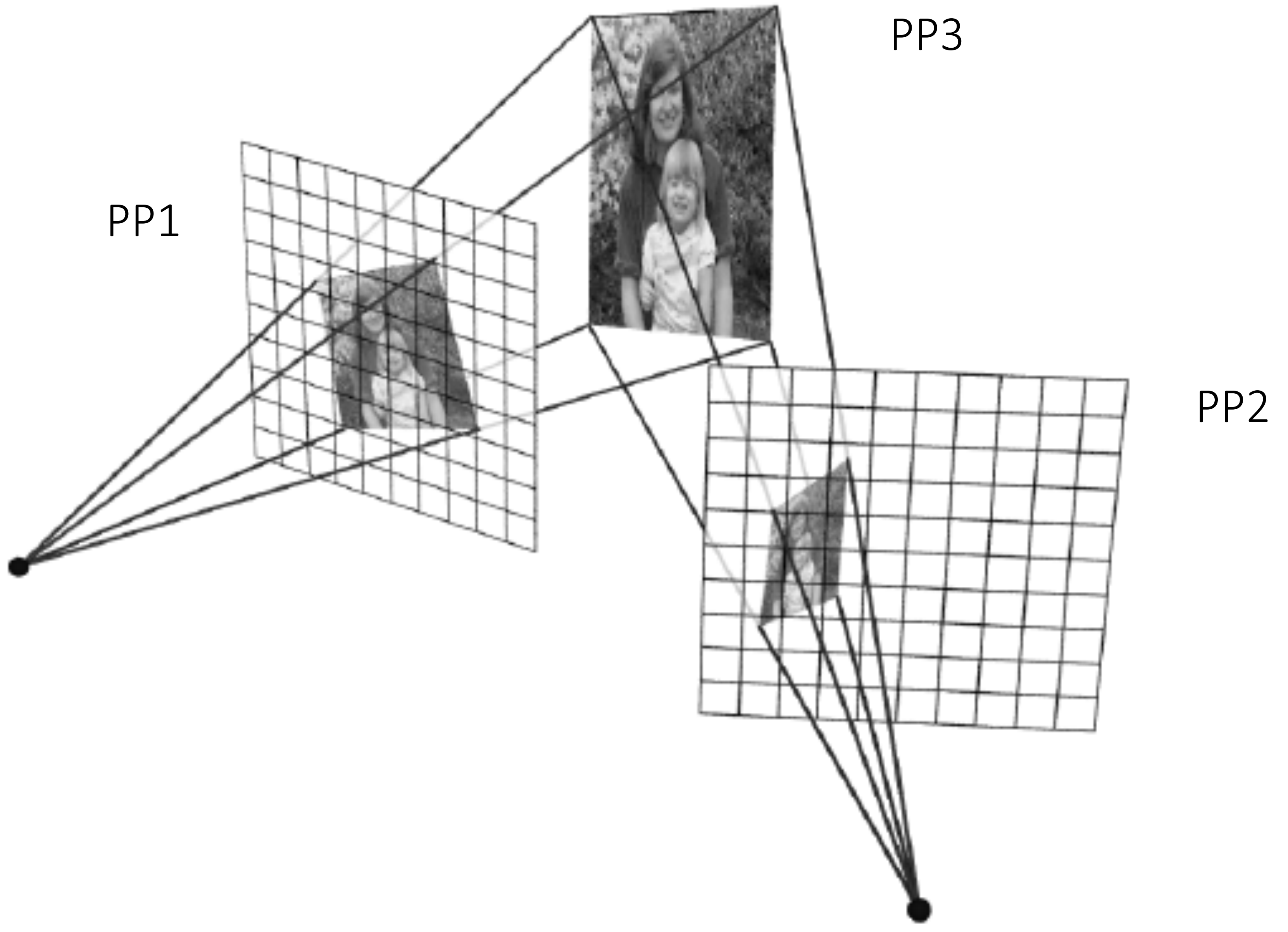
Name	Matrix	# D.O.F.
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	2
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	3
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times3}$	4
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times3}$	6
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times3}$	8

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: Classification of 2D Transformations

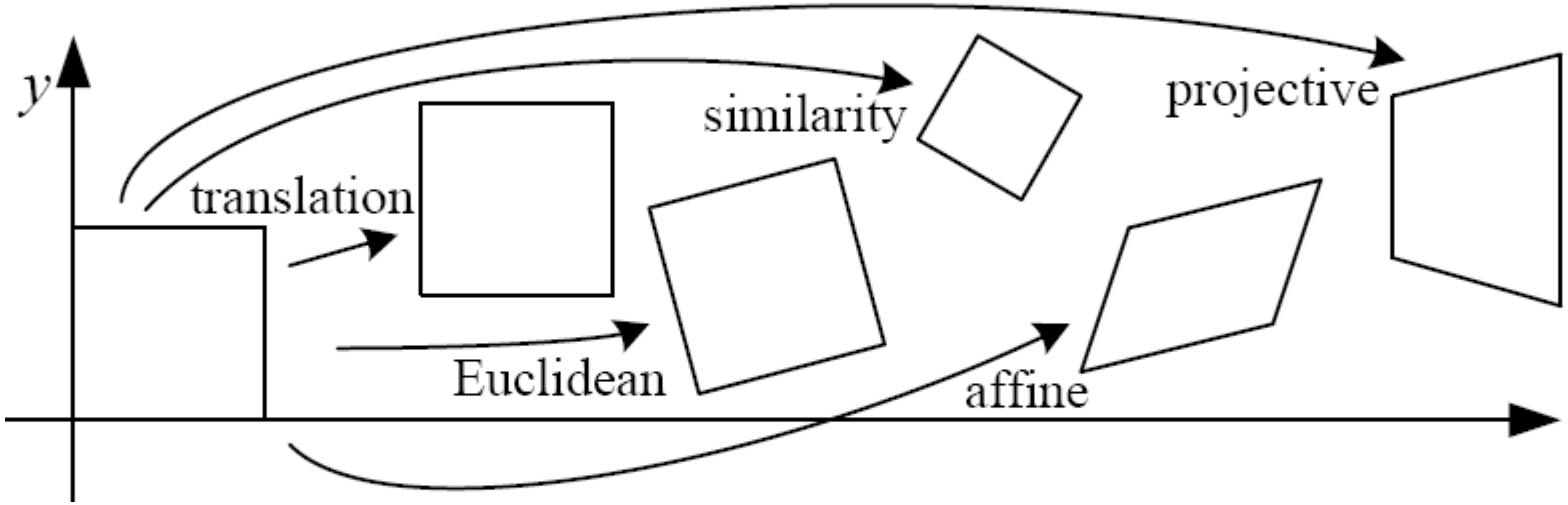


Which kind **transformation** is needed to warp projective plane 1 into projective plane 2?



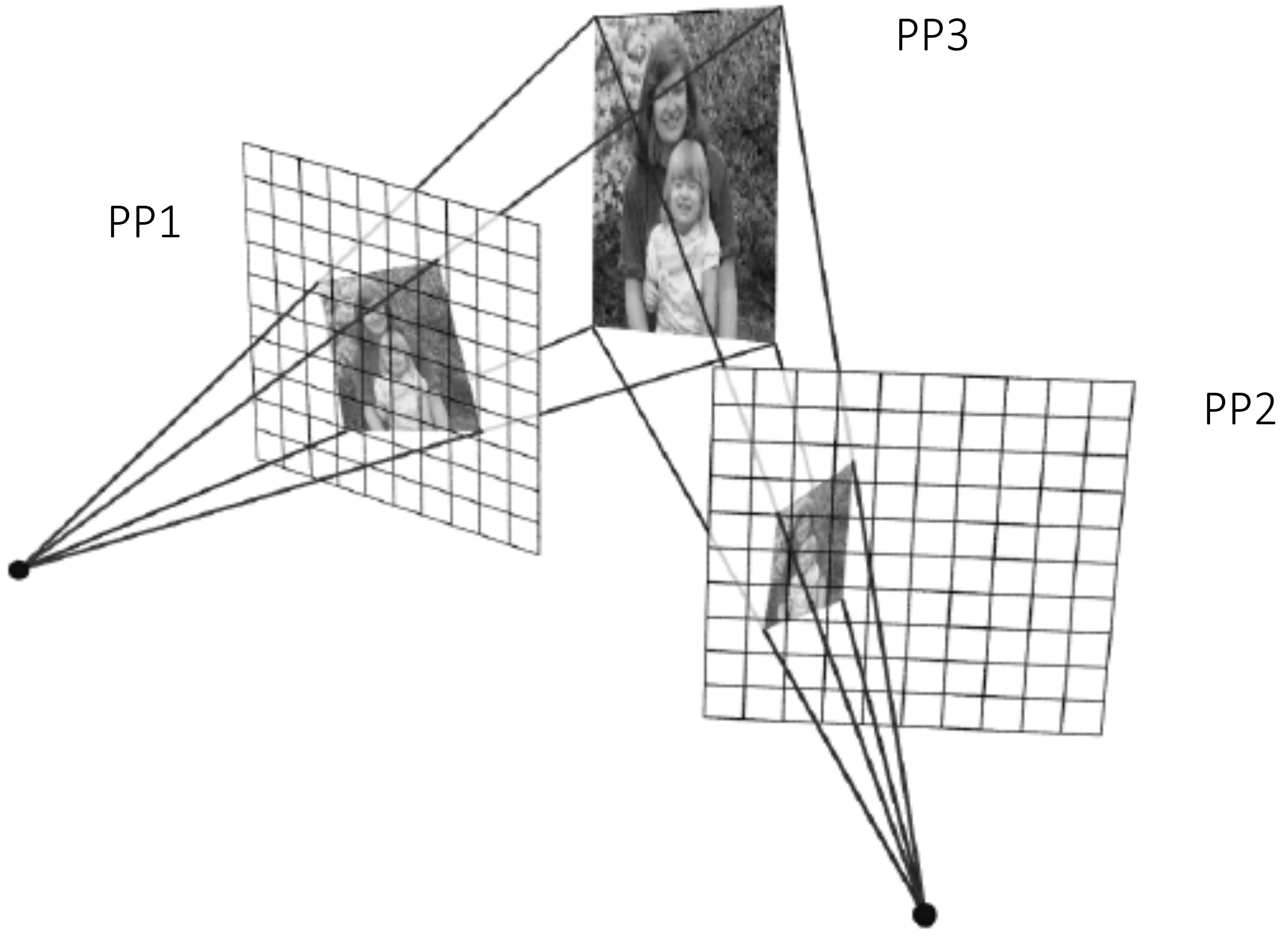
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: Classification of 2D Transformations



Which kind **transformation** is needed to warp projective plane 1 into projective plane 2?

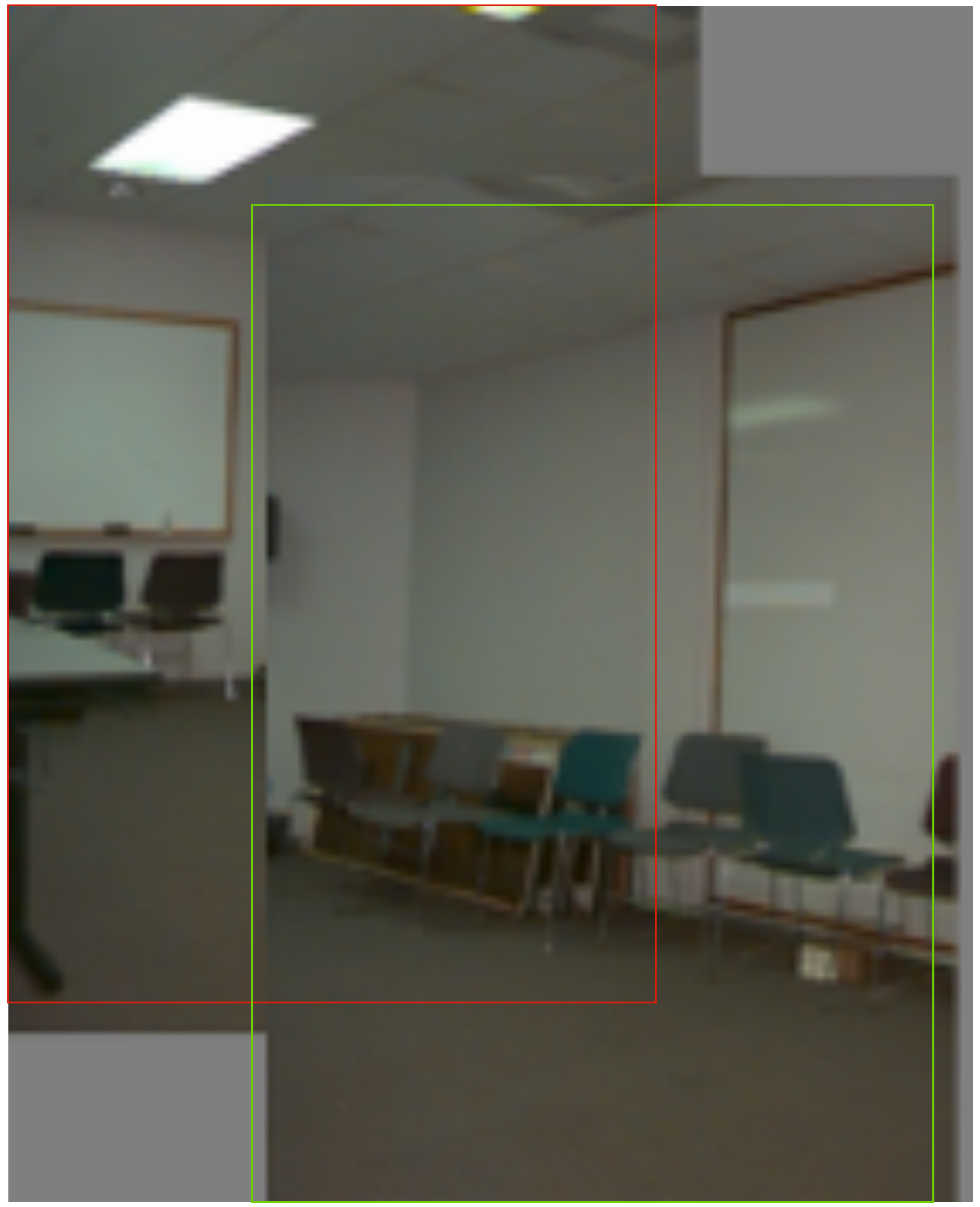
— A **projective** transformation (a.k.a. a homography).



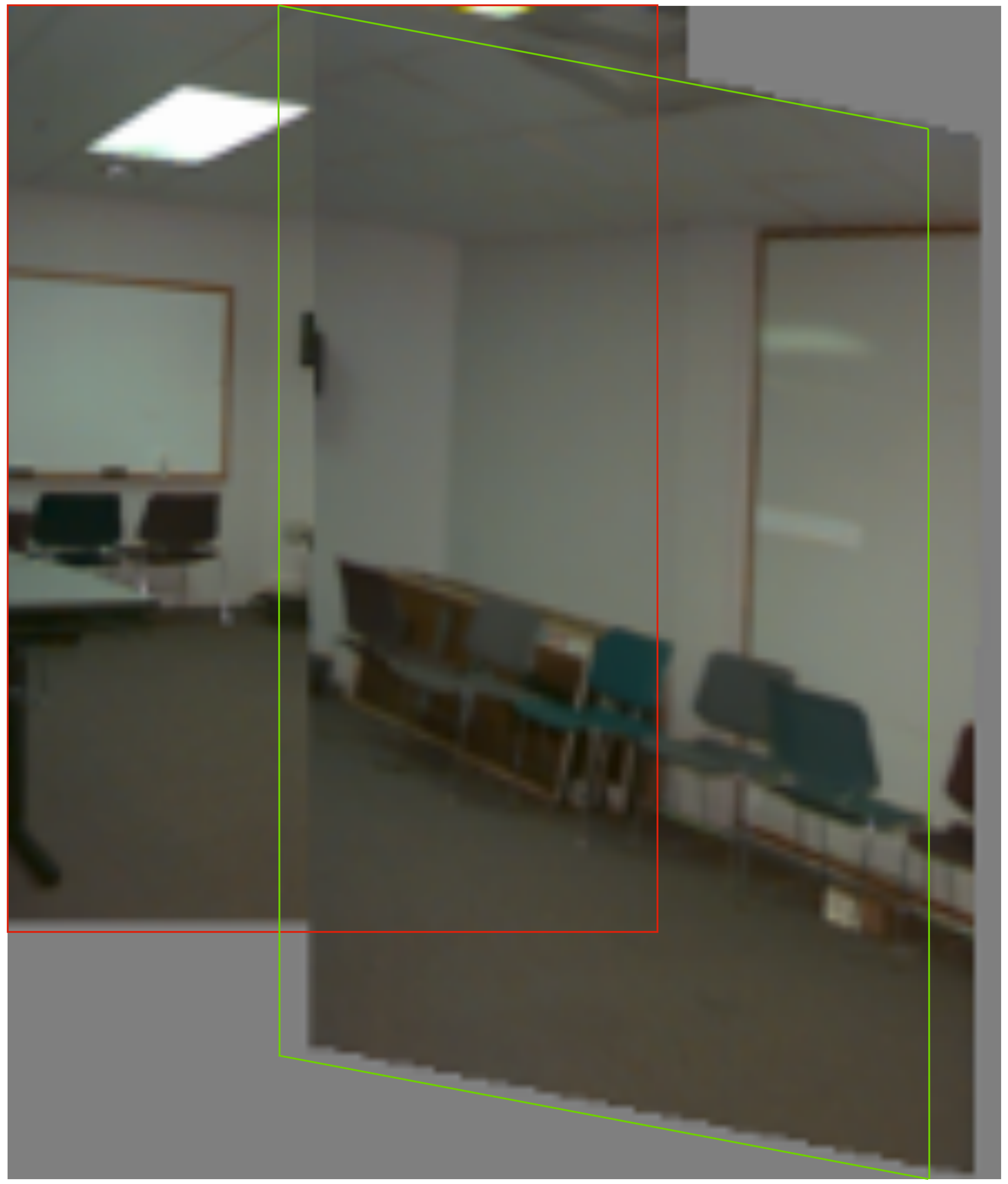
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: Warping with Different Transformations

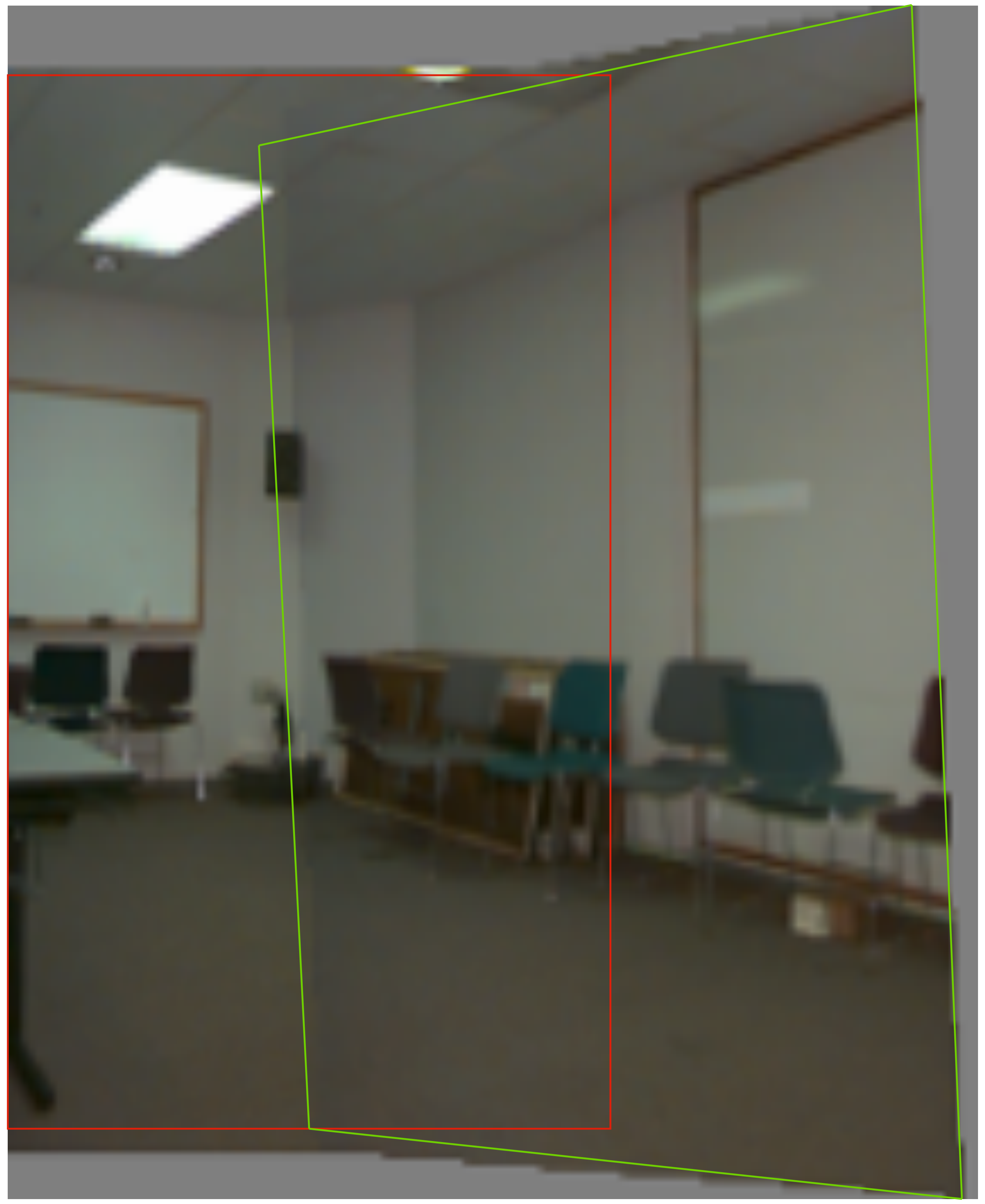
Translation



Affine



Projective
(homography)



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: We can use homographies when …

1.… the scene is planar; or

2.… the scene is very far or has small (relative) depth variation → scene is approximately planar

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: We can use homographies when …

3.… the scene is captured under camera rotation only (no translation or pose change)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Solution for **Affine** Parameters

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

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

Rewrite to solve for **transformation** parameters:

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

(6 equations 6 unknowns)

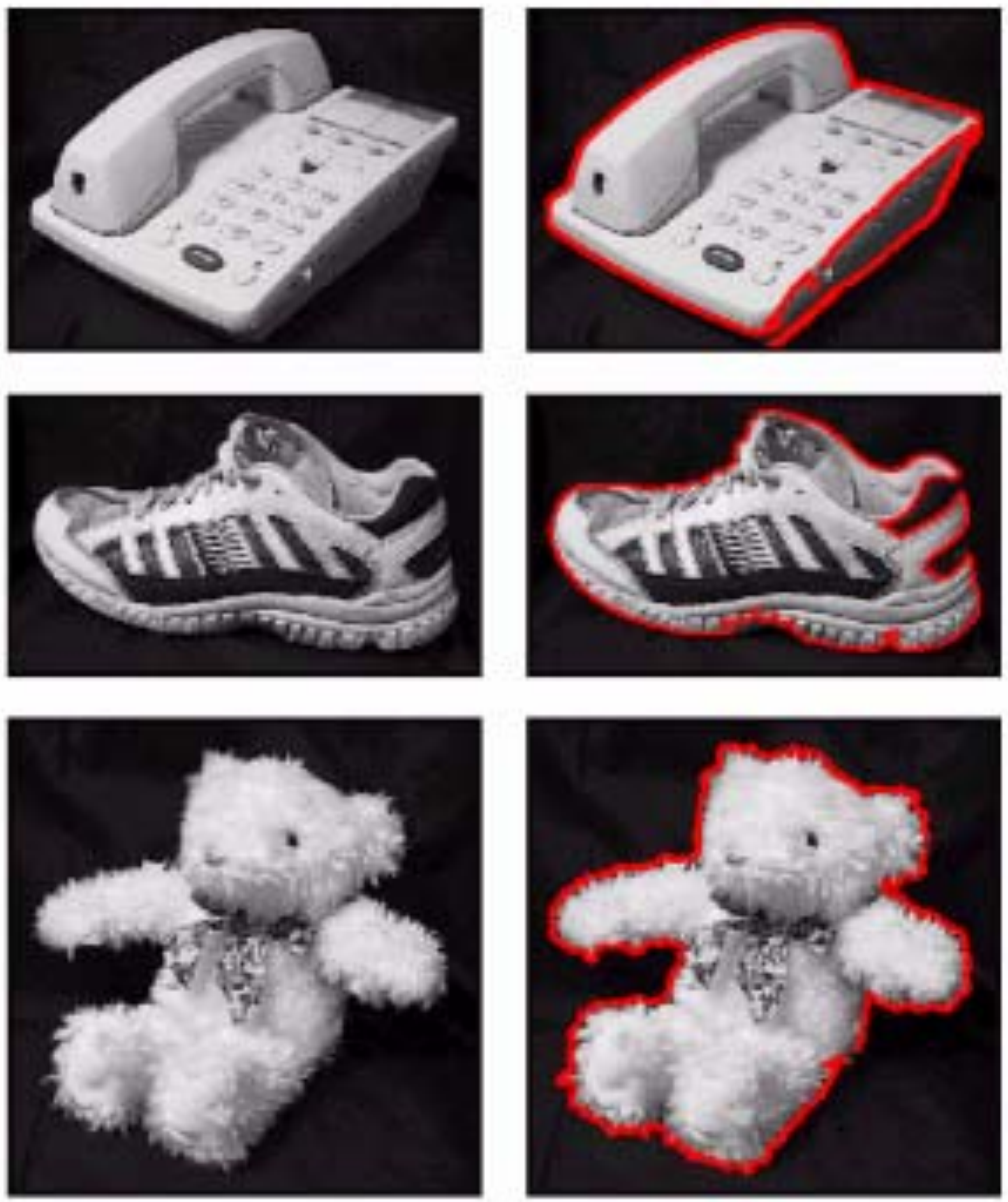
Solution for **Affine** Parameters

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

Then,

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

3D **Object Recognition**

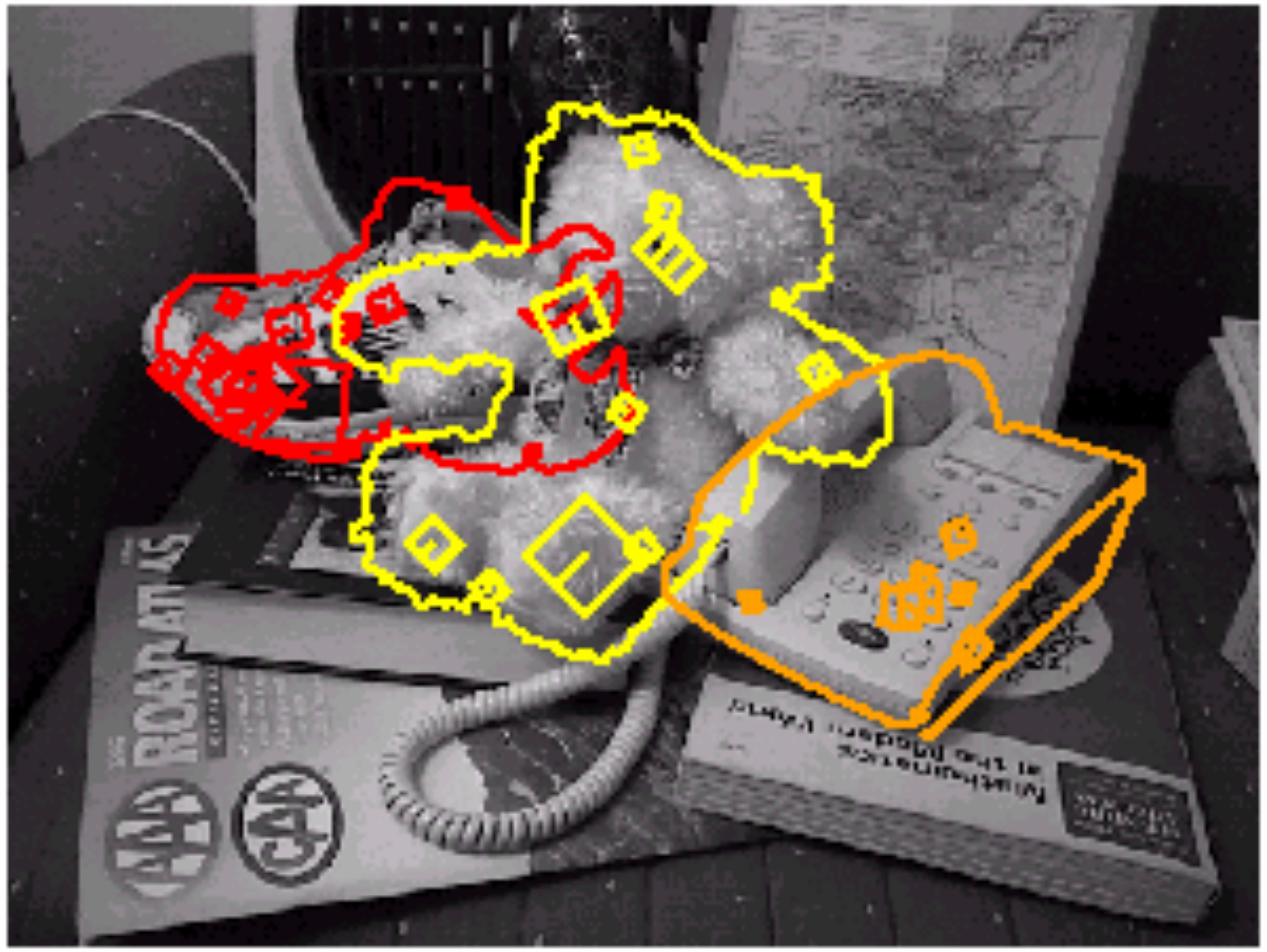


Extract outlines with background subtraction

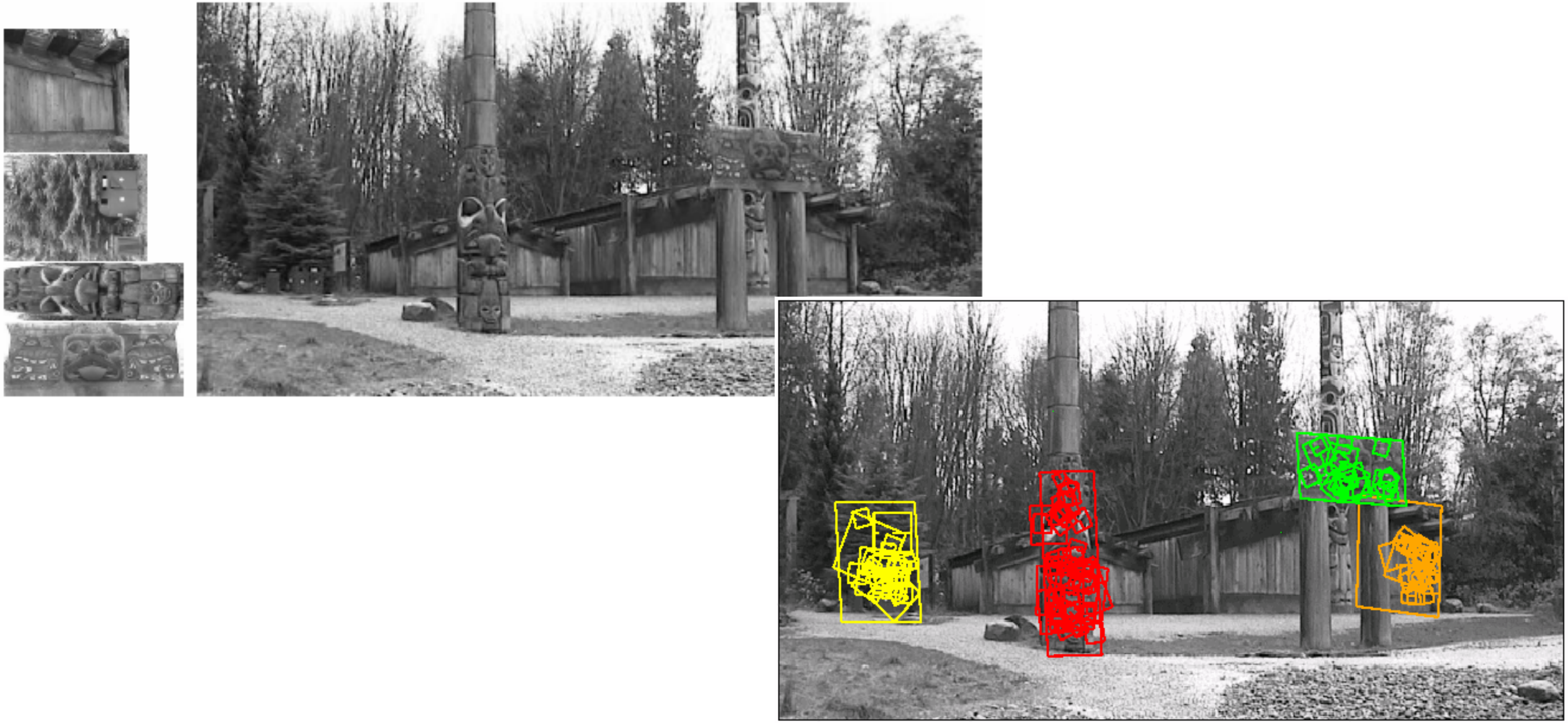
3D **Object Recognition**

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

Recognition Under **Occlusion**



Location Recognition



Example 1: Sony Aibo

SIFT Usage

— Recognize charging station

— Communicate with visual cards

Summary of Object Recognition with SIFT

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

— use fast (approximate) nearest neighbour matching

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

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

— use generalized Hough transform

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

— accept/reject interpretation accordingly

Fitting a Model to Noisy Data

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

We can fit a line using two points

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

Fitting a Model to Noisy Data

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

We can fit a line using two points

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

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

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

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

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

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

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

Slide Credit: Christopher Rasmussen

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

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

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

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

RANSAC is very useful for variety of applications

Slide Credit: Christopher Rasmussen

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

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

Fitting a Line: 2 points

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

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

Slide Credit: Christopher Rasmussen

Example 1: Fitting a Line

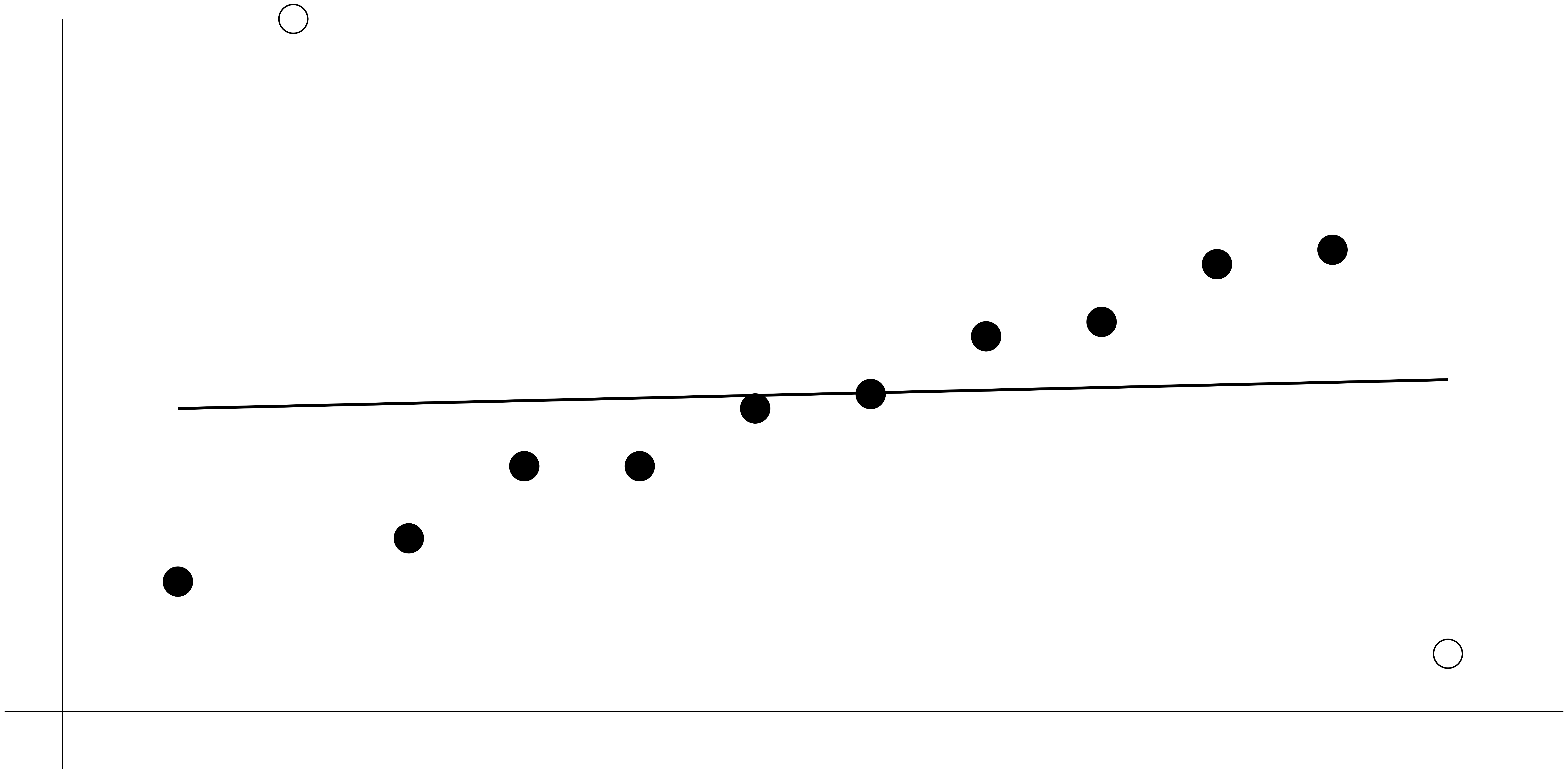


Figure Credit: Hartley & Zisserman

Example 1: Fitting a Line

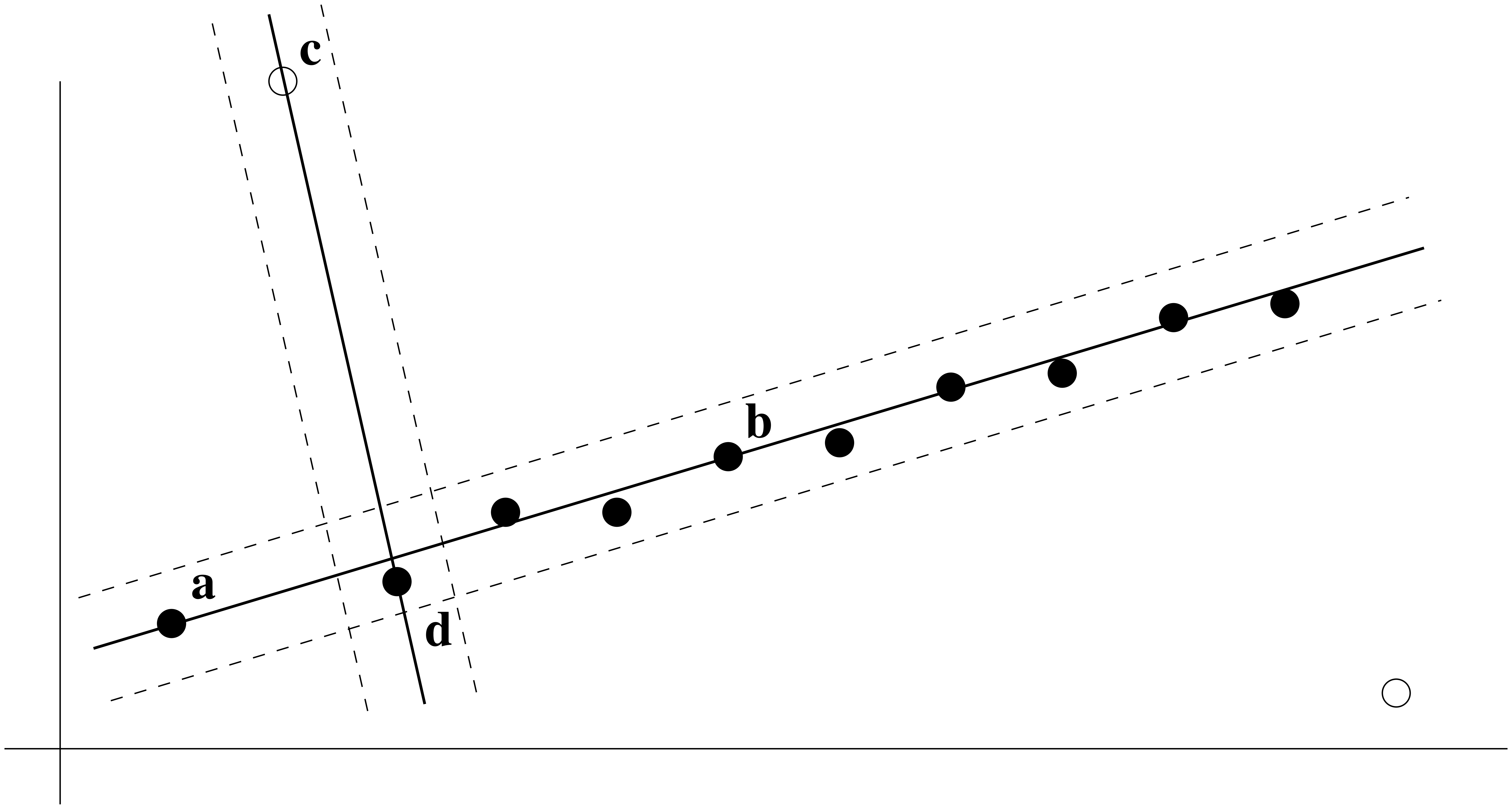


Figure Credit: Hartley & Zisserman

Example 1: Fitting a Line

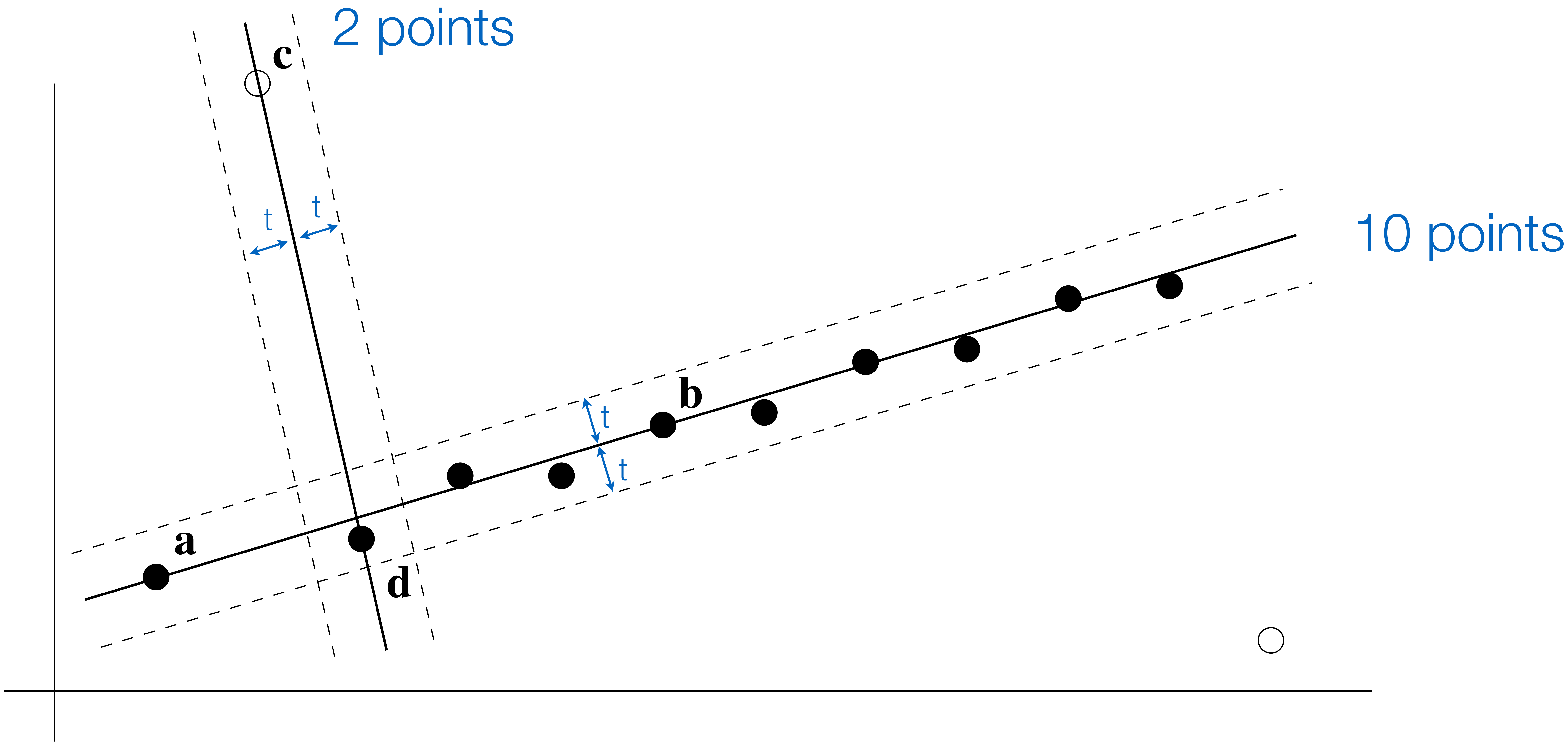


Figure Credit: Hartley & Zisserman

Algorithm 10.4

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

Algorithm 15.4: RANSAC: fitting lines using random sample consensus

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

RANSAC: Fitting Lines Using Random Sample Consensus

RANSAC: How many samples?

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

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

Suppose k samples are chosen

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

RANSAC: How many samples?

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

Let n be the number of points needed to define hypothesis

($n = 2$ for a line in the plane)

Suppose k samples are chosen

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

$$\omega^n$$

The probability that all k samples fail is

RANSAC: How many samples?

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

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

Suppose k samples are chosen

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

$$\omega^n$$

The probability that all k samples fail is

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

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

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

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

Figure Credit: Hartley & Zisserman

After RANSAC

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

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

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

Example 2: Fitting a Line

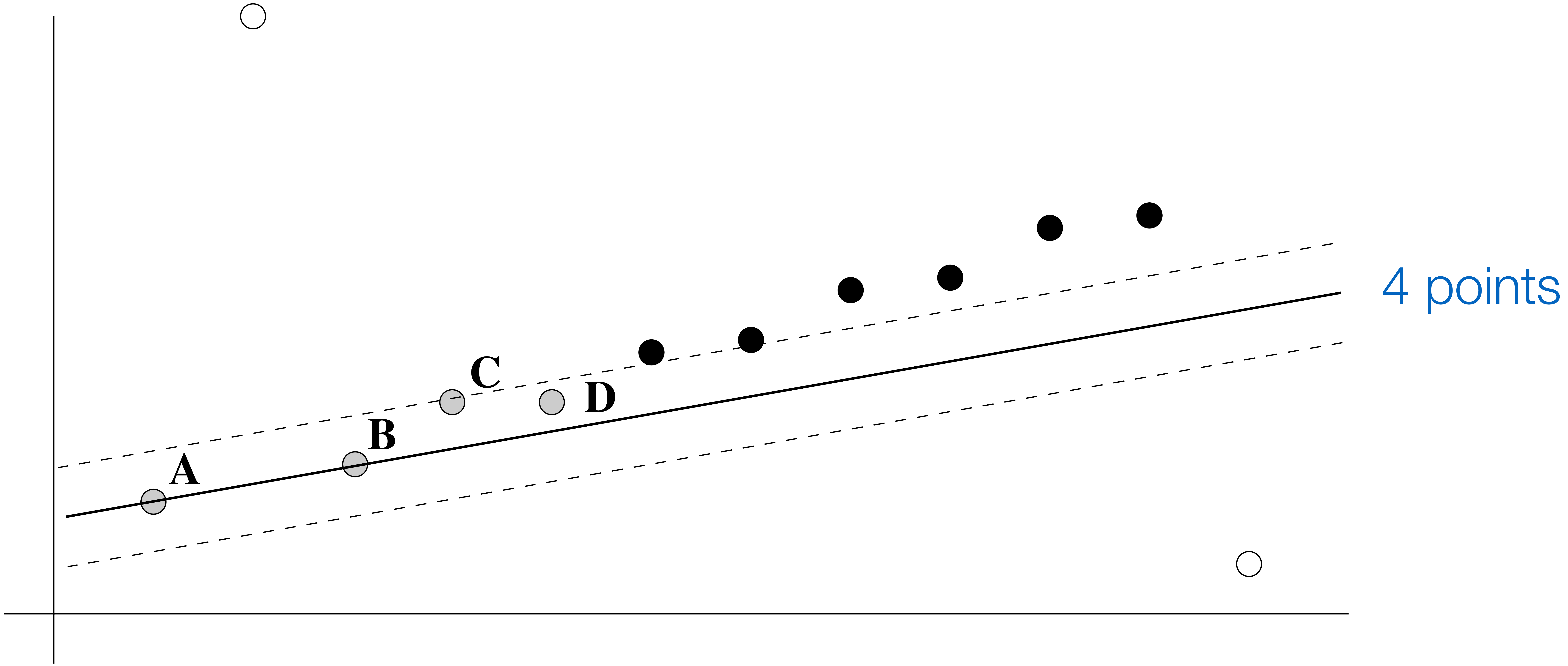


Figure Credit: Hartley & Zisserman

Example 2: Fitting a Line

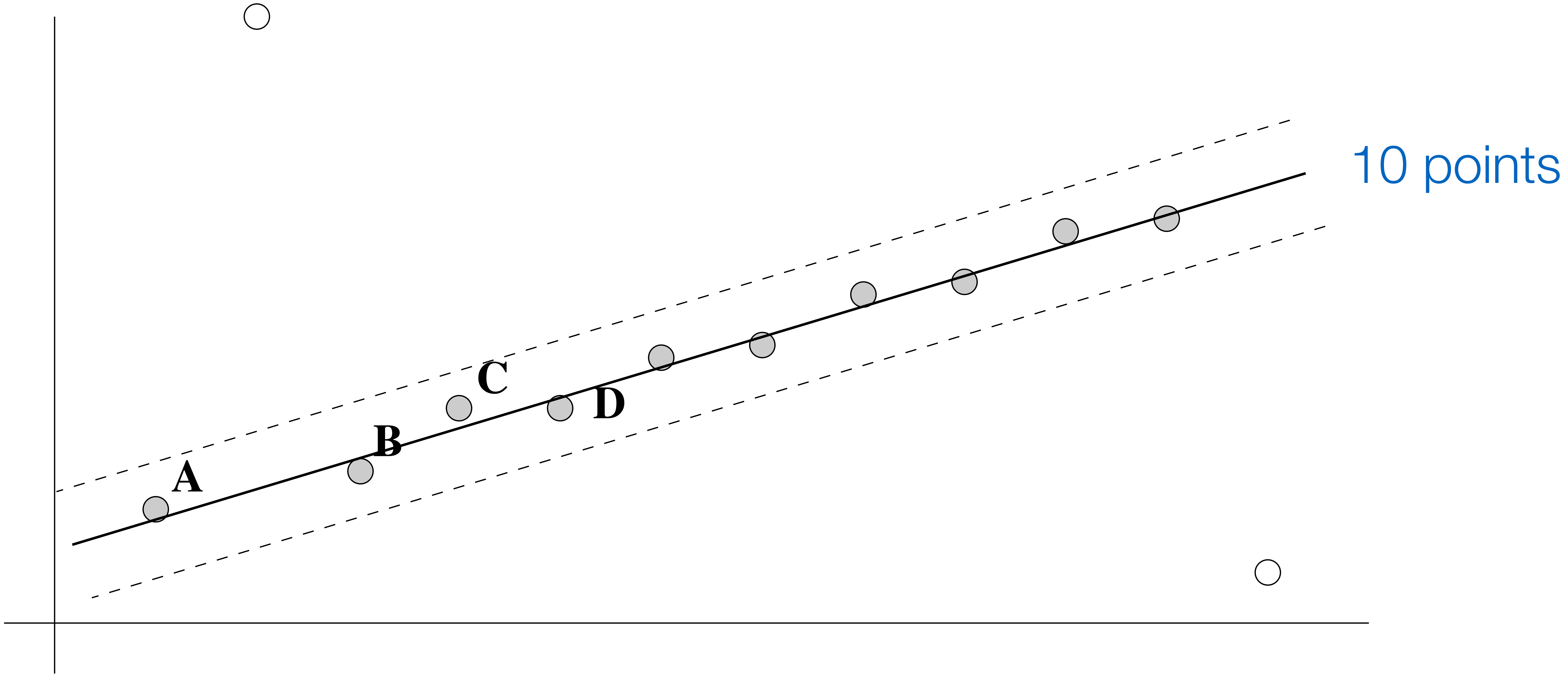


Figure Credit: Hartley & Zisserman