THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision

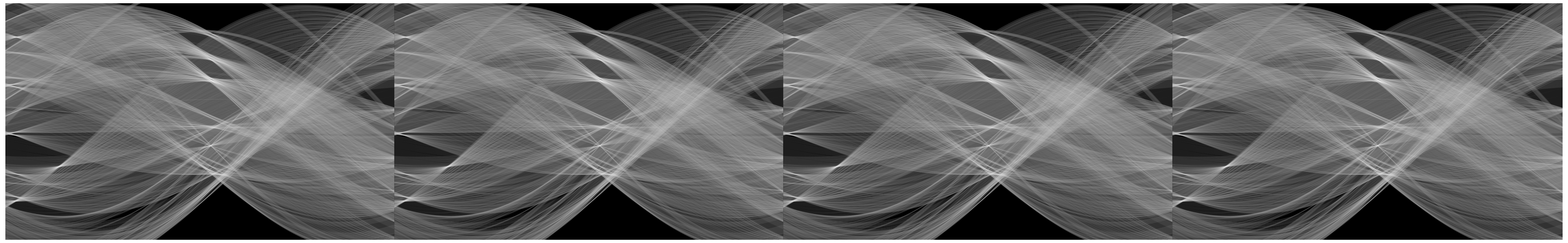


Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 14: Object Recognition, RANSAC, Hough Transform

Menu for Today (**October 28, 2020**)

Topics:

— Object Detection

— Model Fitting

— RANSAC

— Hough Transform

Readings:

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

Reminders:

— **Midterm** is still being graded (we lost grades due to Canvas mishap)

— **Assignment 4**: please start working on it!

— **Final Exam** date is set to December 16th @ noon.

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

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

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

Polina Zablotskaia, Aliaksandr Siarohin,
Bo Zhao and Leonid Sigal

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

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

Polina Zablotskaia, Aliaksandr Siarohin,
Bo Zhao and Leonid Sigal

Lecture 19: Re-Cap

Keypoint is an image location at which a descriptor is computed

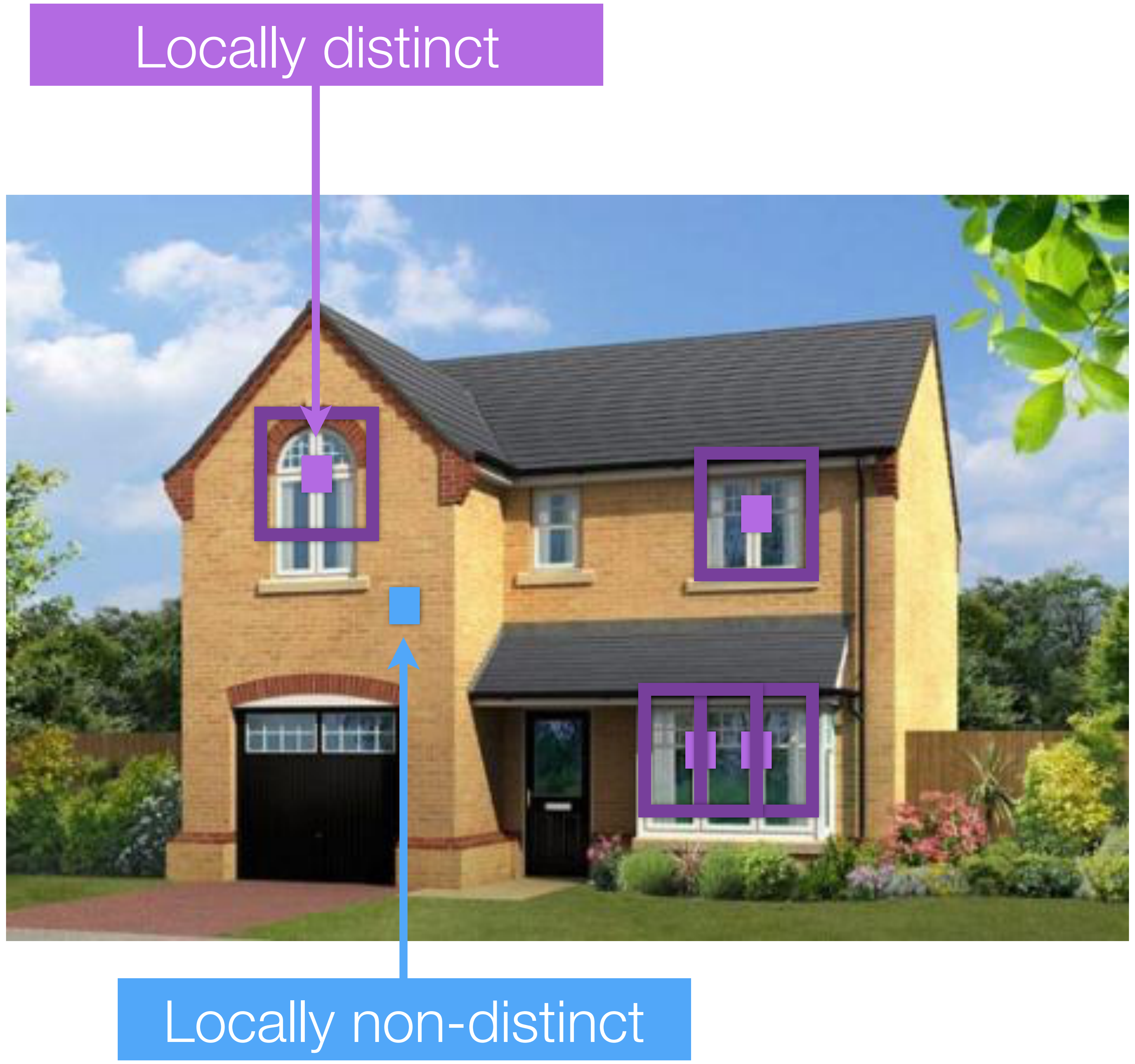
— Locally distinct points

— Easily localizable and identifiable

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

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

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



Lecture 19: Re-Cap

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

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

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

Lecture 19: Re-Cap

Four steps to SIFT feature generation:

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

Lecture 19: Histogram of Oriented Gradients (**HOG**)

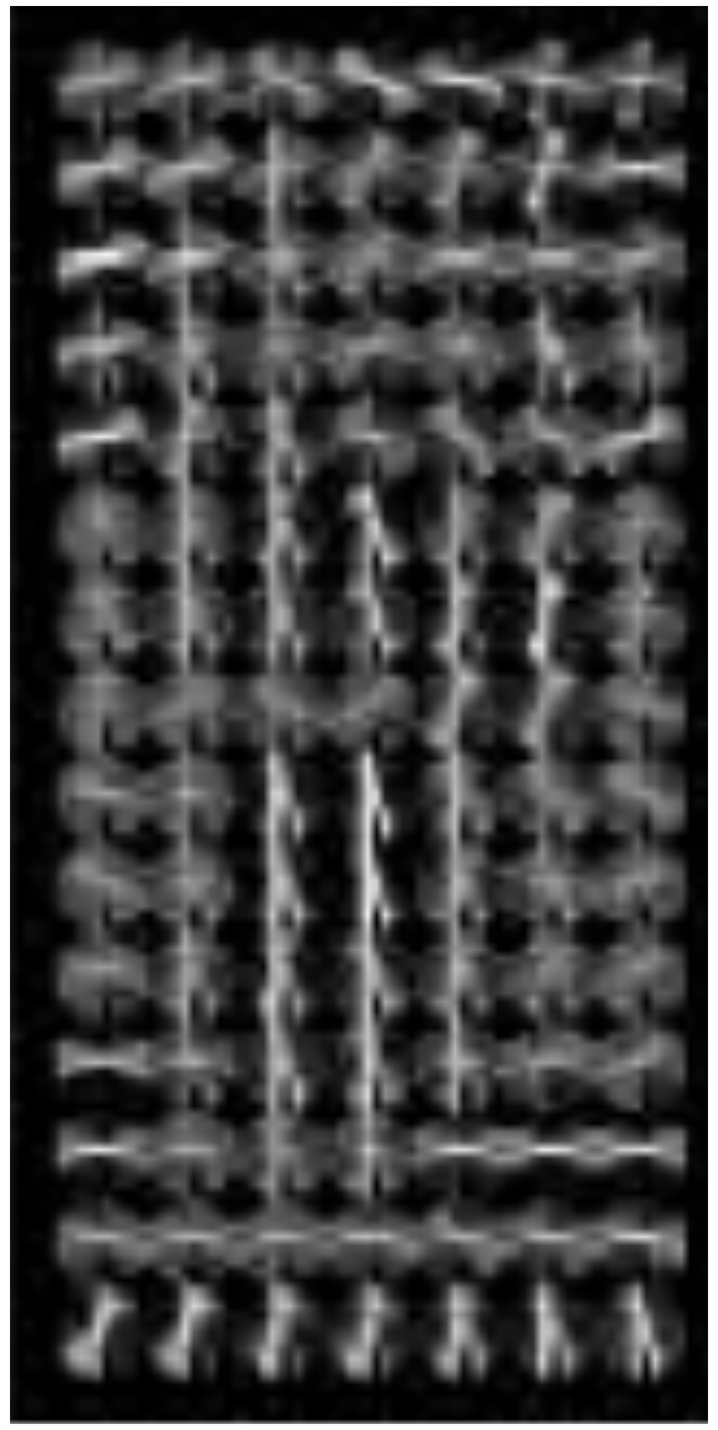
1 cell step size

visualization

Pedestrian detection

128 pixels
16 cells
15 blocks

15 x 7 x 4 x 9 =
3780

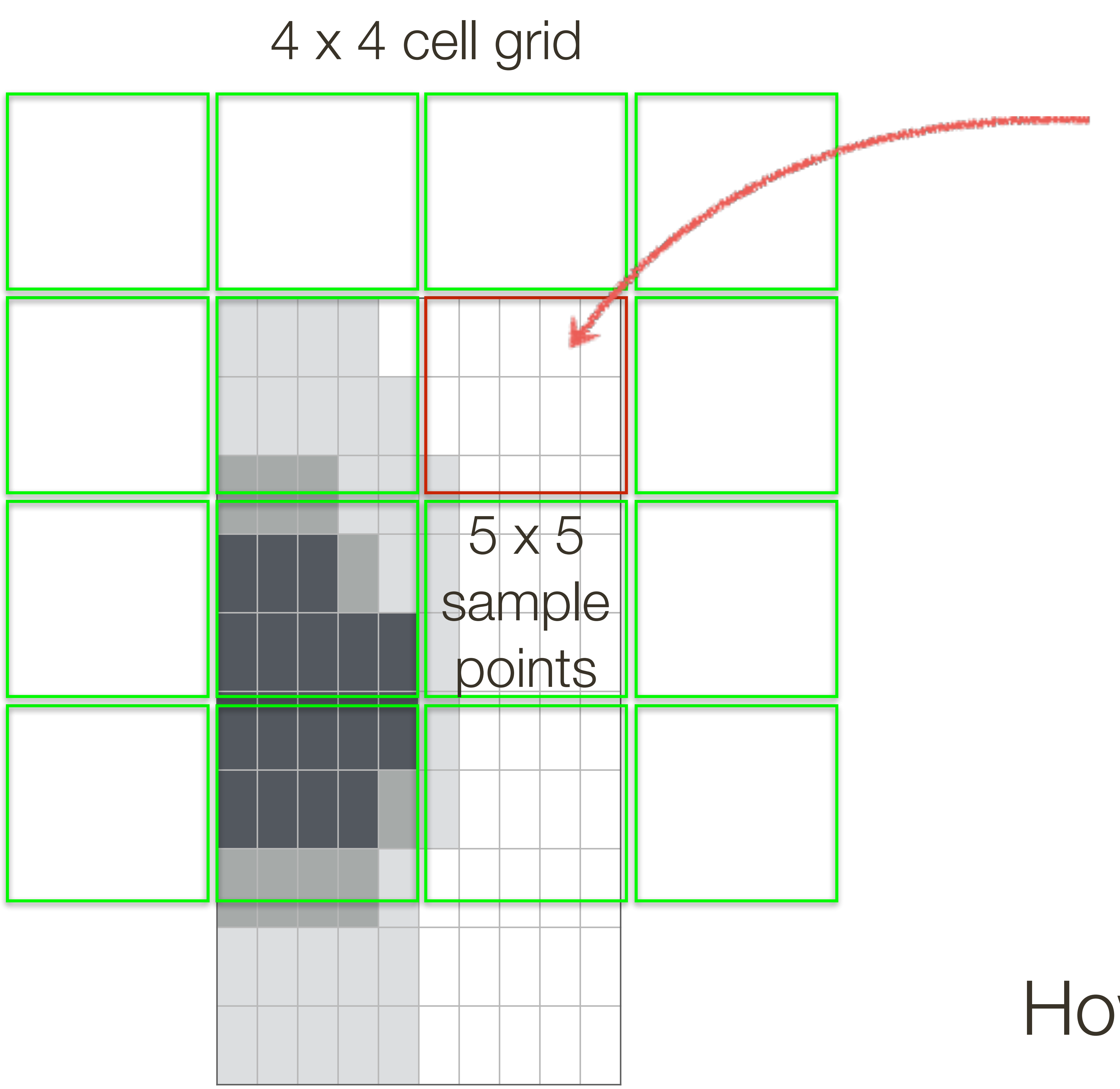


64 pixels
8 cells
7 blocks

Redundant representation due to overlapping blocks

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

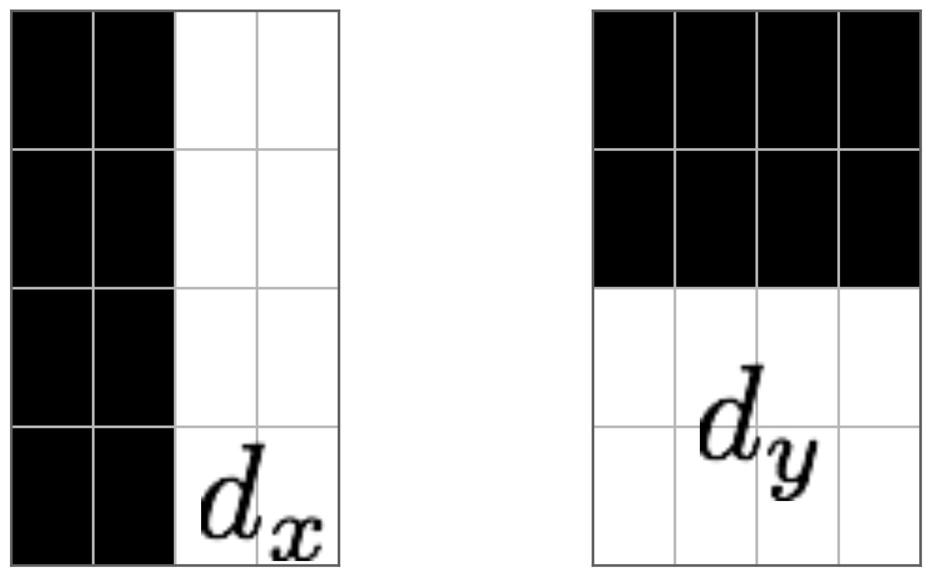
Lecture 19: 'Speeded' Up Robust Features



Each cell is represented by 4 values:

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

Haar wavelets filters (Gaussian weighted from center)



How big is the SURF descriptor?

64 dimensions

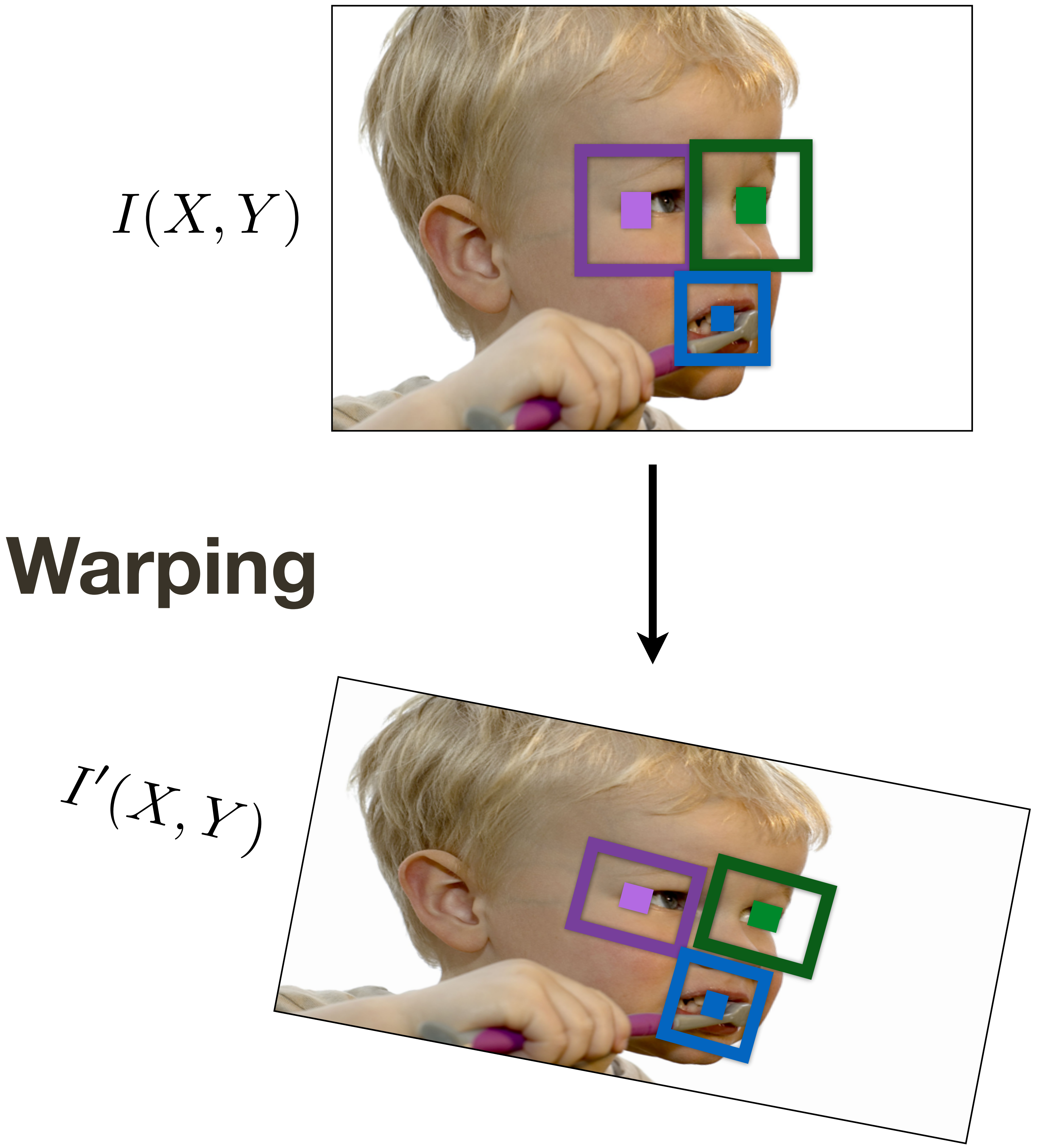
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Summary

Keypoint Detection Algorithms	**Representation**
Harris Corners	(x,y,s)
LoG / Blobs	(x,y,s)
SIFT	(x,y,s,theta)

Keypoint Description Algorithms	**Representation**
SIFT	128D
Histogram of Oriented Gradients	3780D
SURF	64D

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

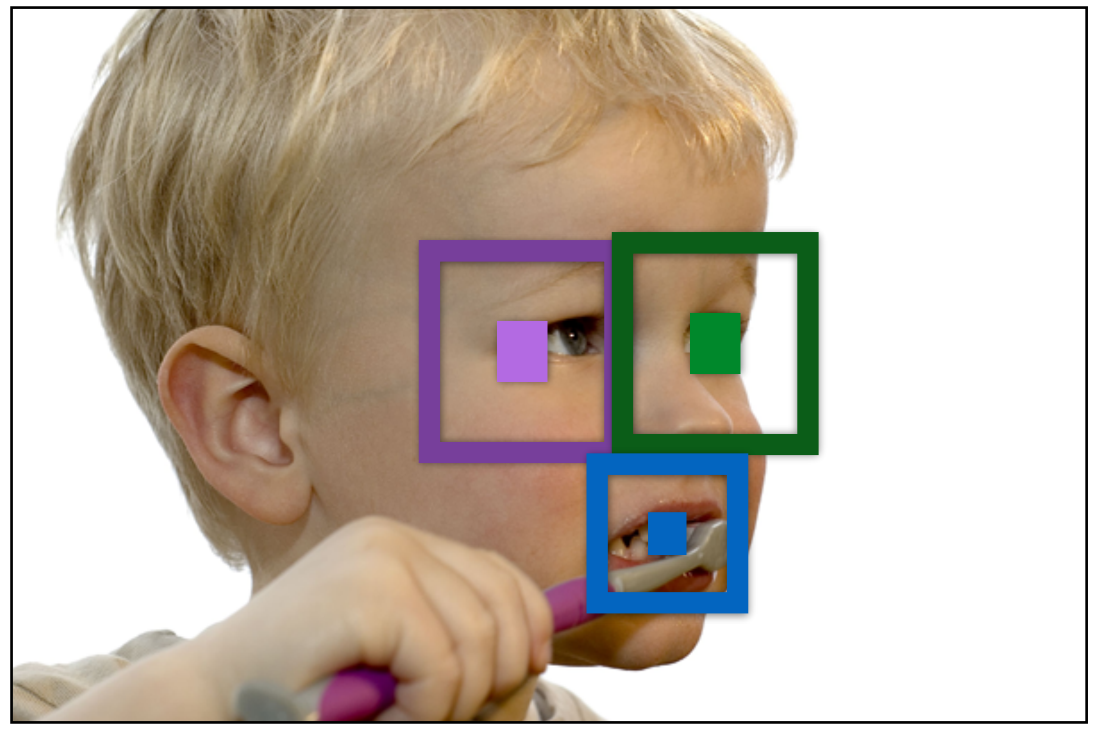


changes domain of image function

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

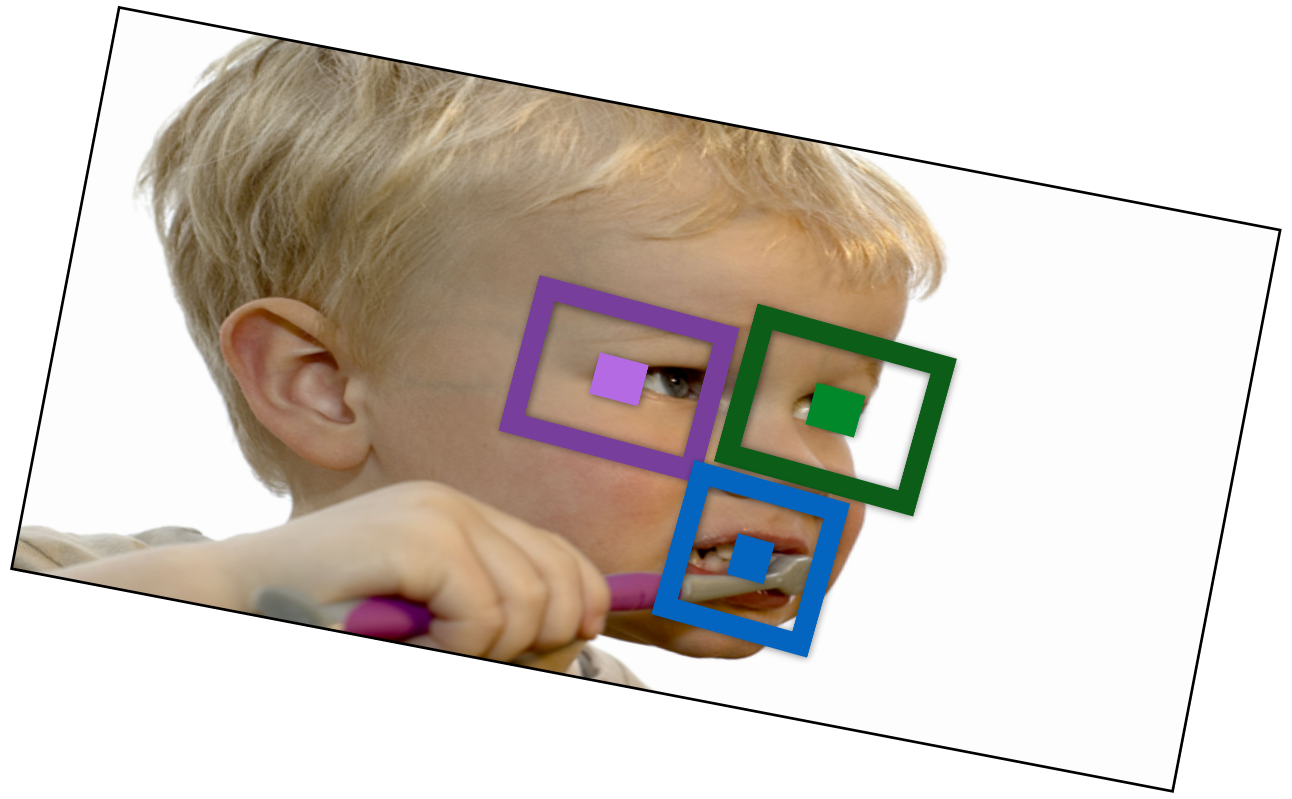
What types of **transformations** can we do?

$I(X, Y)$



Warping

$I'(X, Y)$



changes domain of image function

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

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

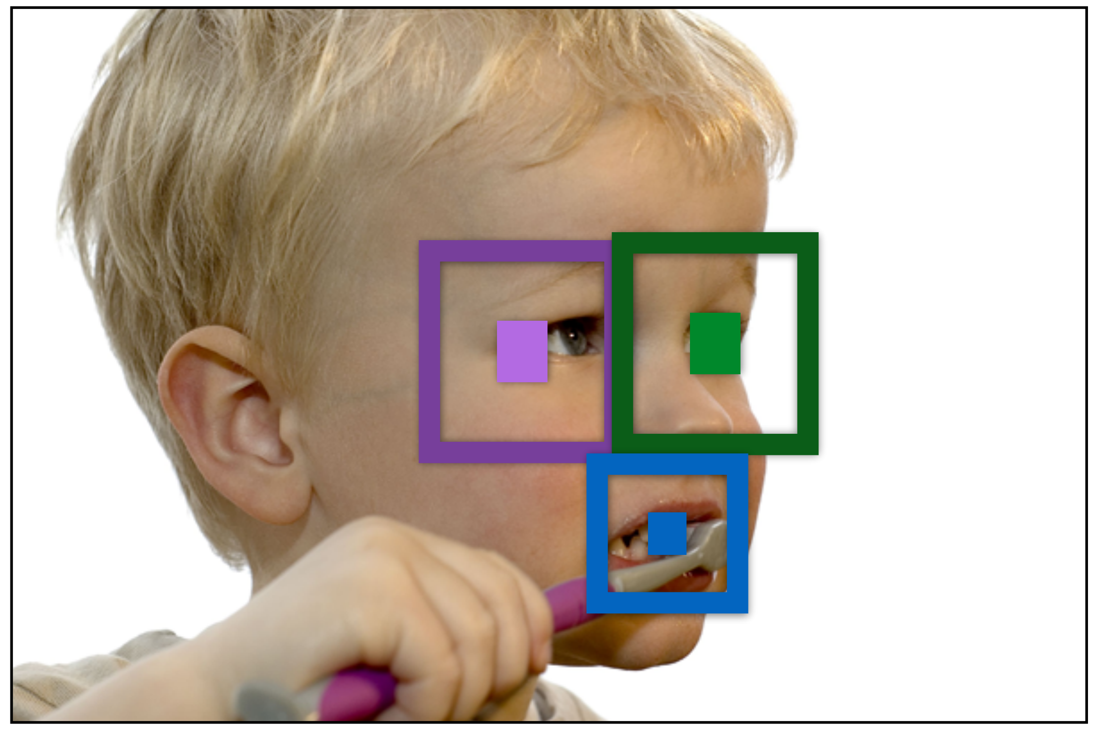
$$I'(X', Y') = I(X, Y)$$

Note: The "model" / "warping" gives you a way to transform any pixel in the original image to the corresponding image

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

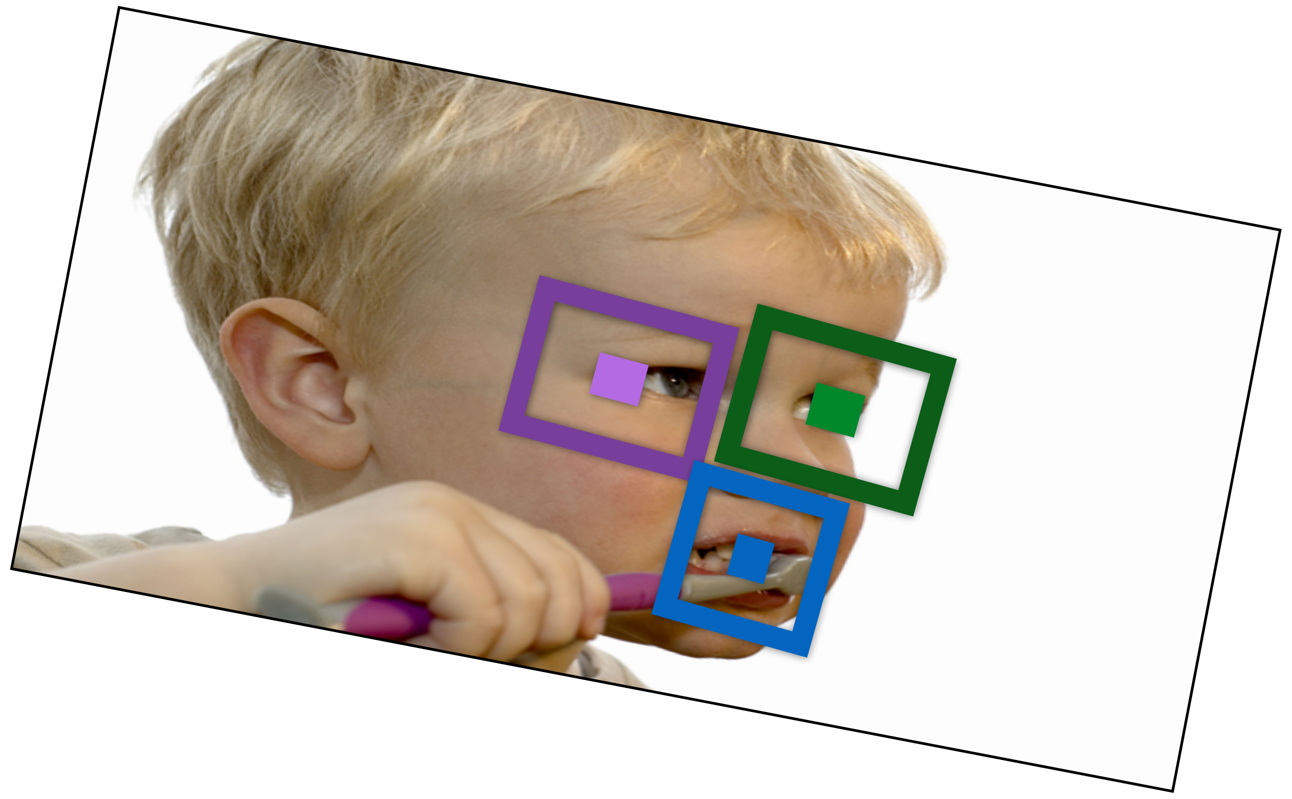
What types of **transformations** can we do?

$I(X,Y)$



Warping

$I'(X,Y)$



changes domain of image function

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

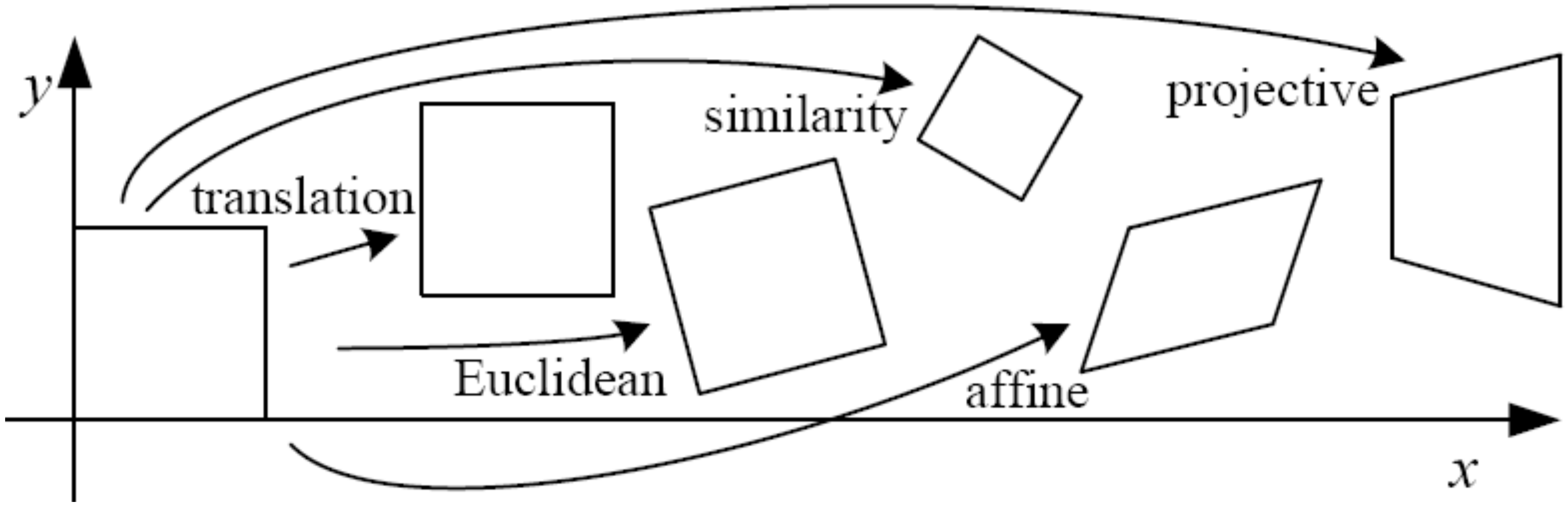
$$\begin{bmatrix} X' \\ Y' \\ 1 \end{bmatrix} = M \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

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

Note: The "model" / "warping" gives you a way to transform any pixel in the original image to the corresponding image

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Forms of the **“Model”**

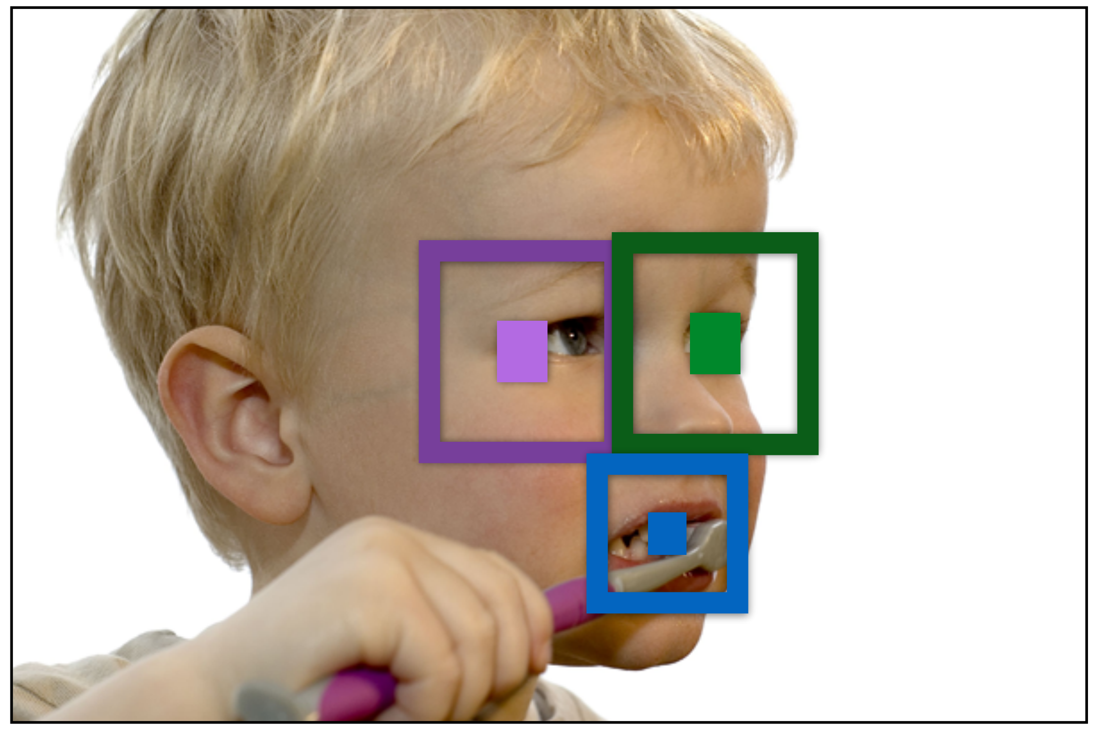


Name	Matrix	# D.O.F.
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$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

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

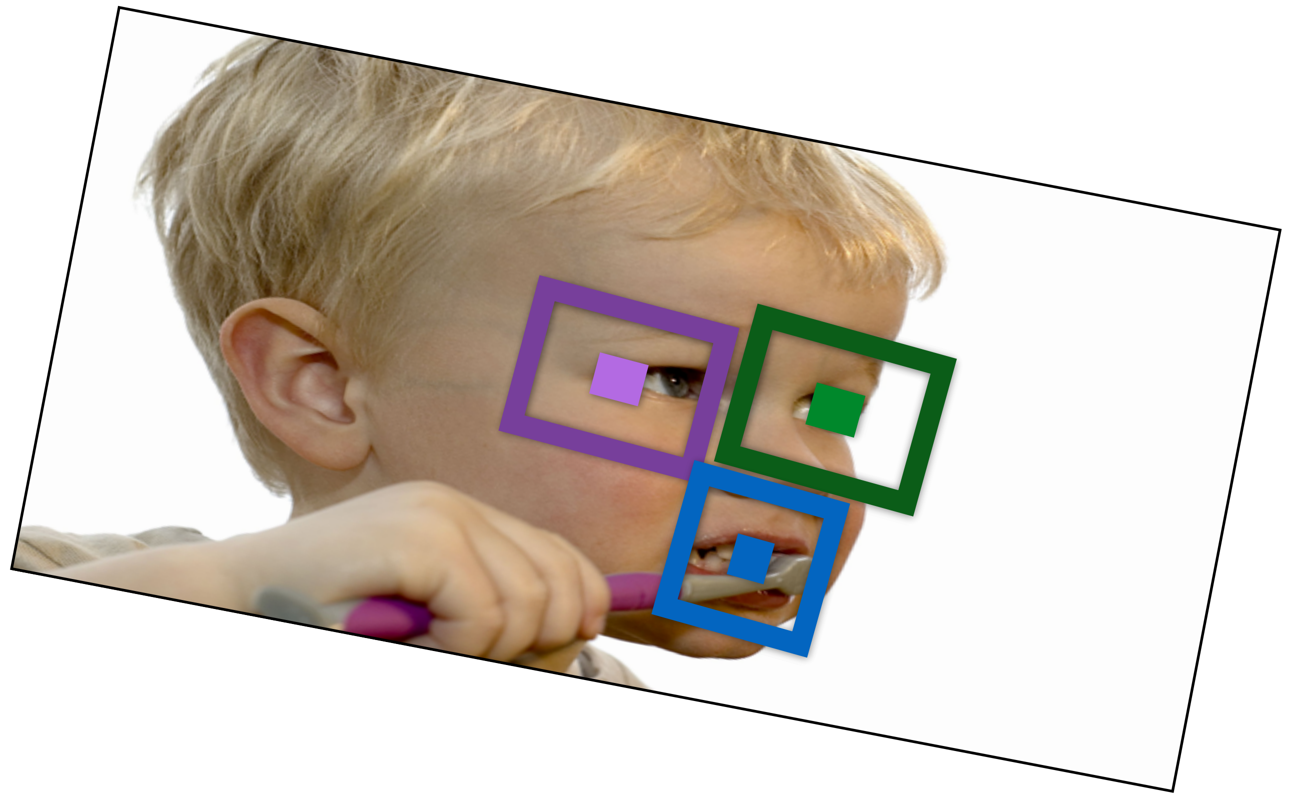
What types of **transformations** can we do?

$I(X,Y)$



Warping

$I'(X,Y)$



changes domain of image function

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

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

Affine "Model"

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

Note: The "model" / "warping" gives you a way to transform any pixel in the original image to the corresponding image

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Solution for **Affine** Parameters

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

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

Rewrite to solve for **transformation** parameters:

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

(6 equations 6 unknowns)

Solution for **Affine** Parameters

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

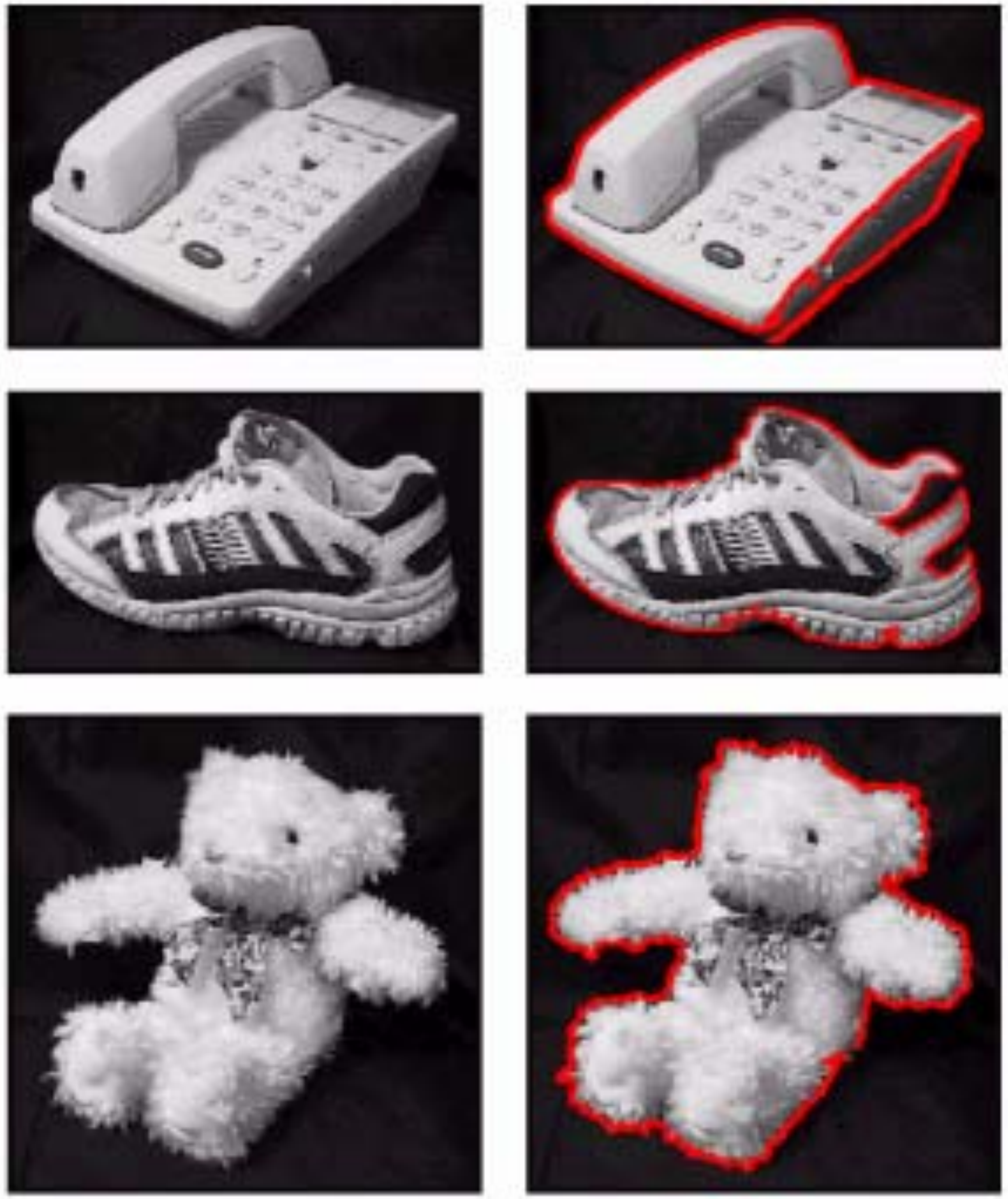
Then,

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

Limitation of this …

We need to have **exact** matches

3D **Object Recognition**

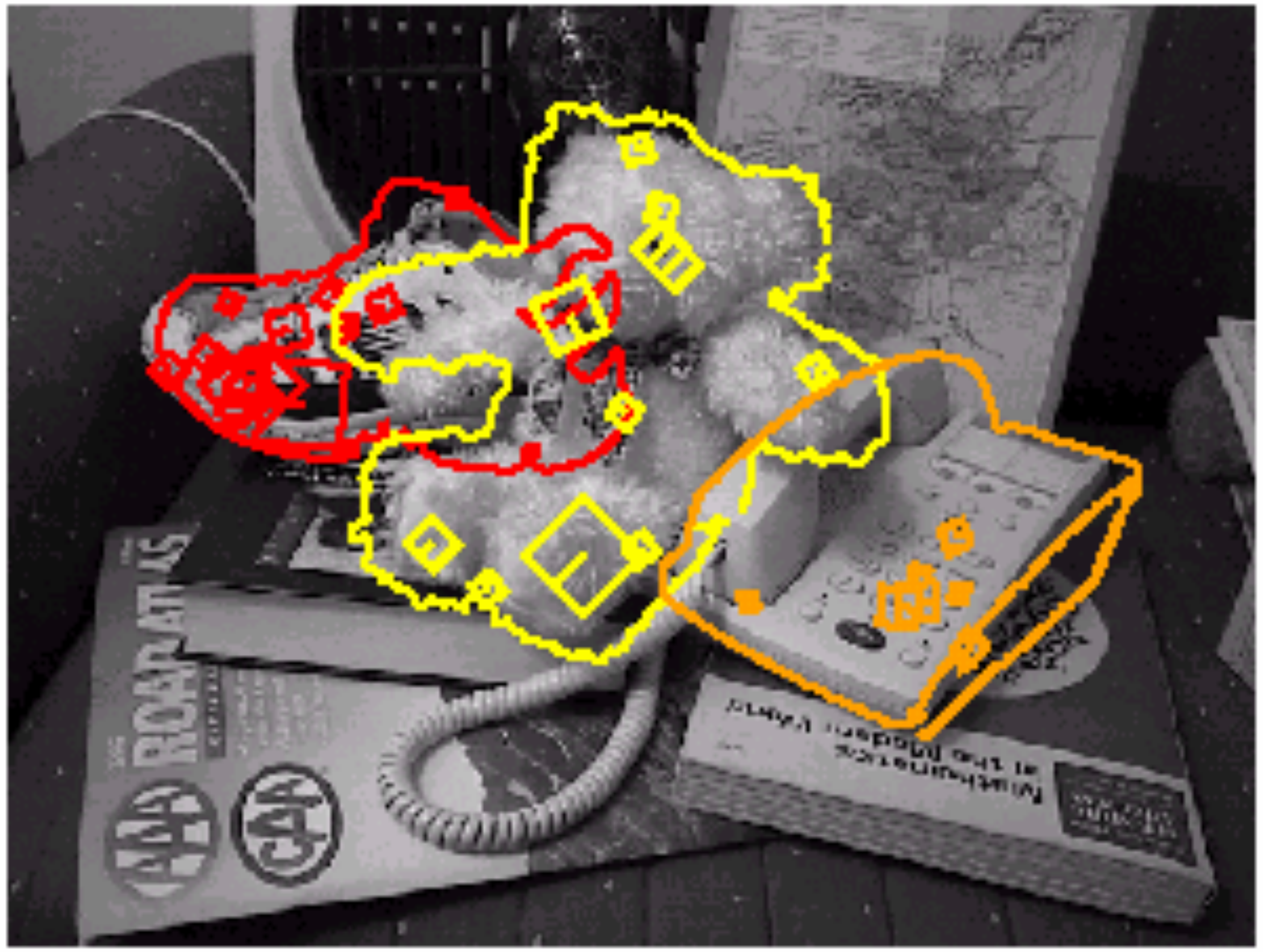


Extract outlines with background subtraction

3D **Object Recognition**

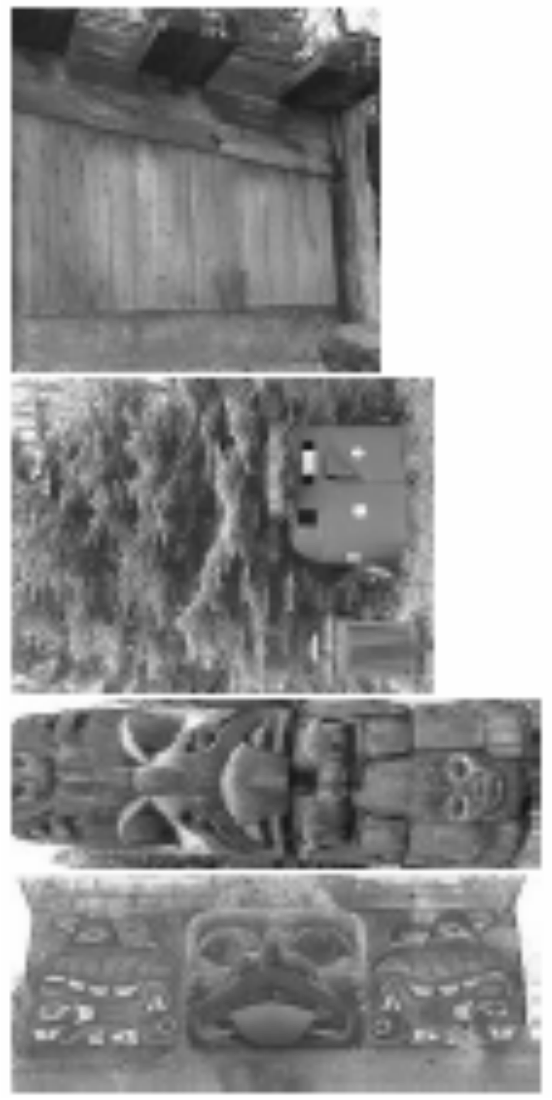
Only 3 keypoints are needed for recognition, so extra keypoints provide robustness

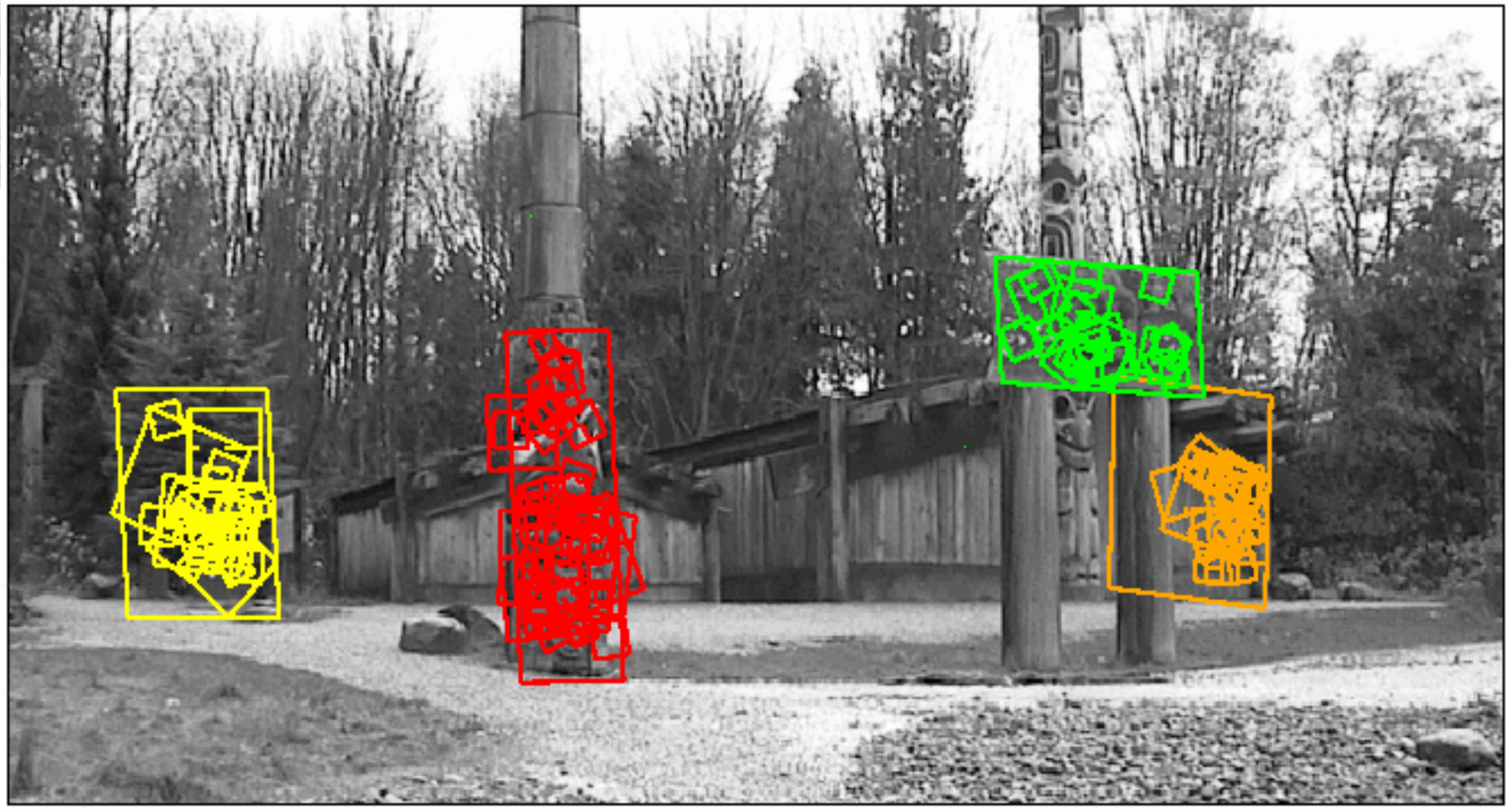
Recognition Under **Occlusion**



Strategy: Solve for M for each object by leveraging SIFT matches of keypoints for that object, then apply M to the outline of that object

Location Recognition

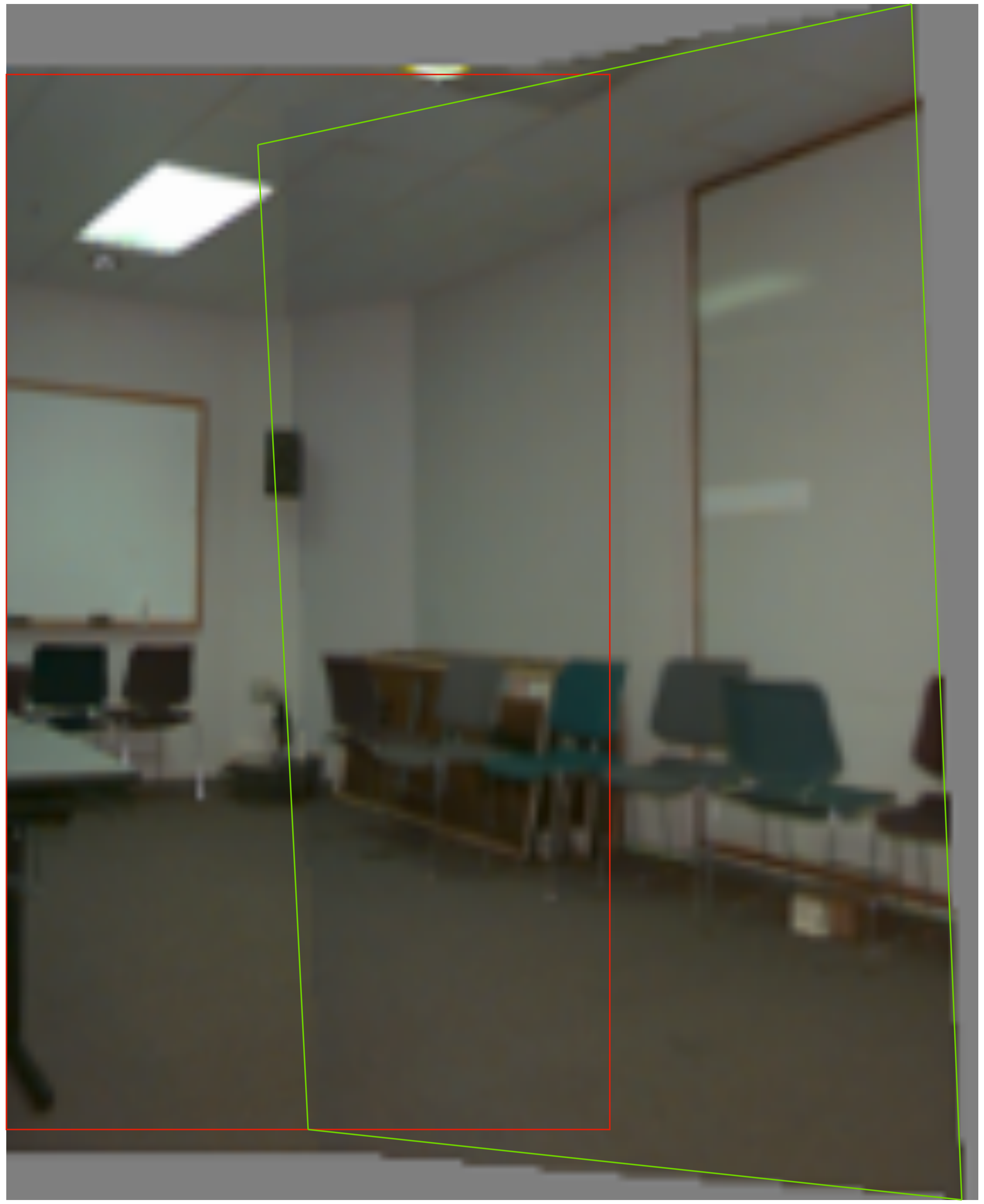




Strategy: Solve for M for each object by leveraging SIFT matches of keypoints for that object, then apply M to the bounding box of that object

Stitching **Panoramas**

Strategy: Solve for M across two views by leveraging SIFT matches of keypoints, then apply M to the entire image for alignment … and then blend



Example 1: Sony Aibo

SIFT Usage

— Recognize charging station

— Communicate with visual cards

Limitation of this …

We need to have **exact** matches

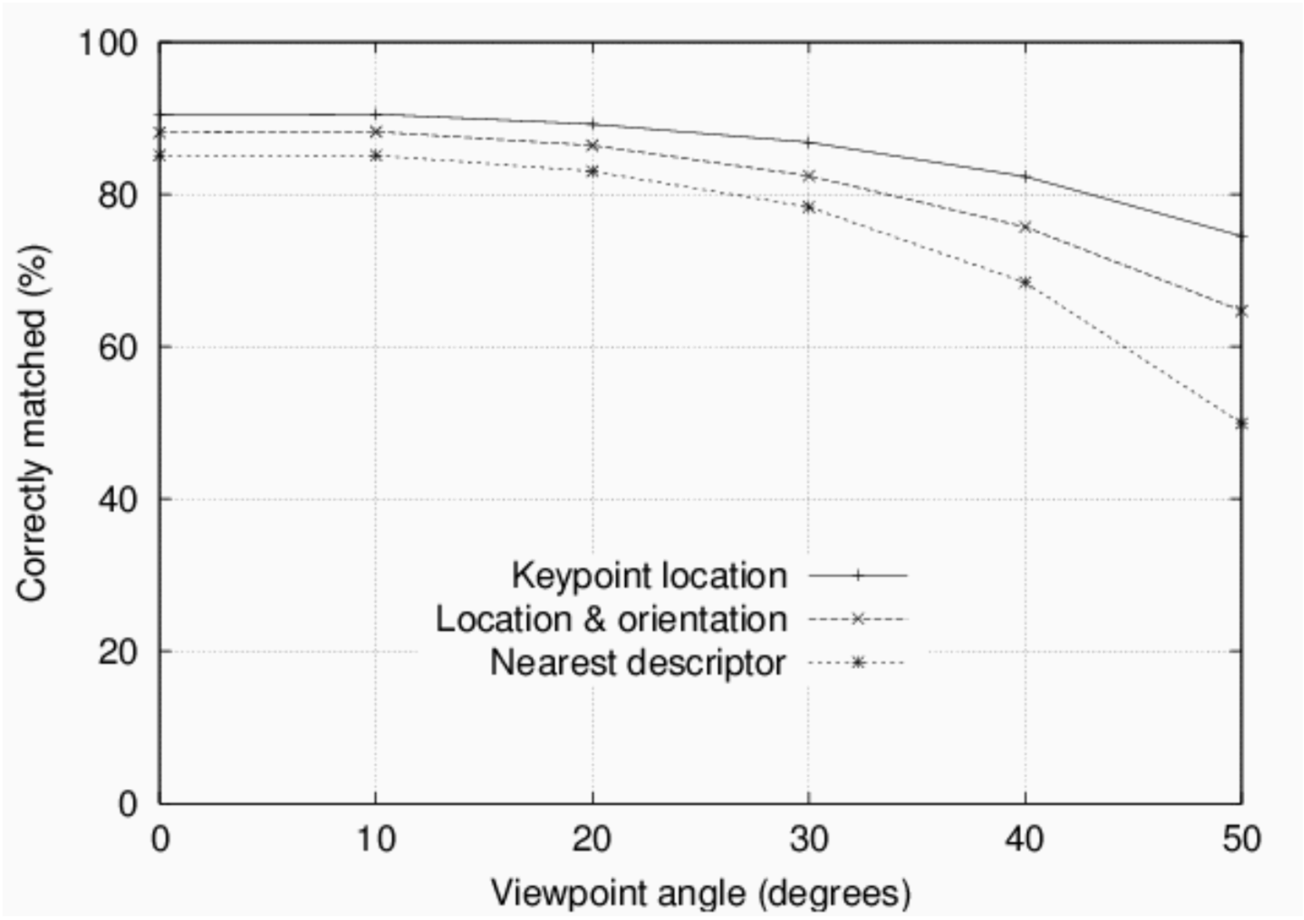
Limitation of this …

Despite all efforts this is **very** difficult …

1. If we can find exact match **80%** of the time, we can find 3 matches correctly only about **50%** of the time.

2. Image **noise**, **deformations**, will make this worse (e.g., if finding exact match drops to 50%, the probability of finding 3 exact matches will drop to 12.5%)

3. Multiple object instances will make this impossible



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 (RANdom SAmple Consensus)

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

Fitting a Line: 2 points

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

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

Slide Credit: Christopher Rasmussen

Example 1: Fitting a Line

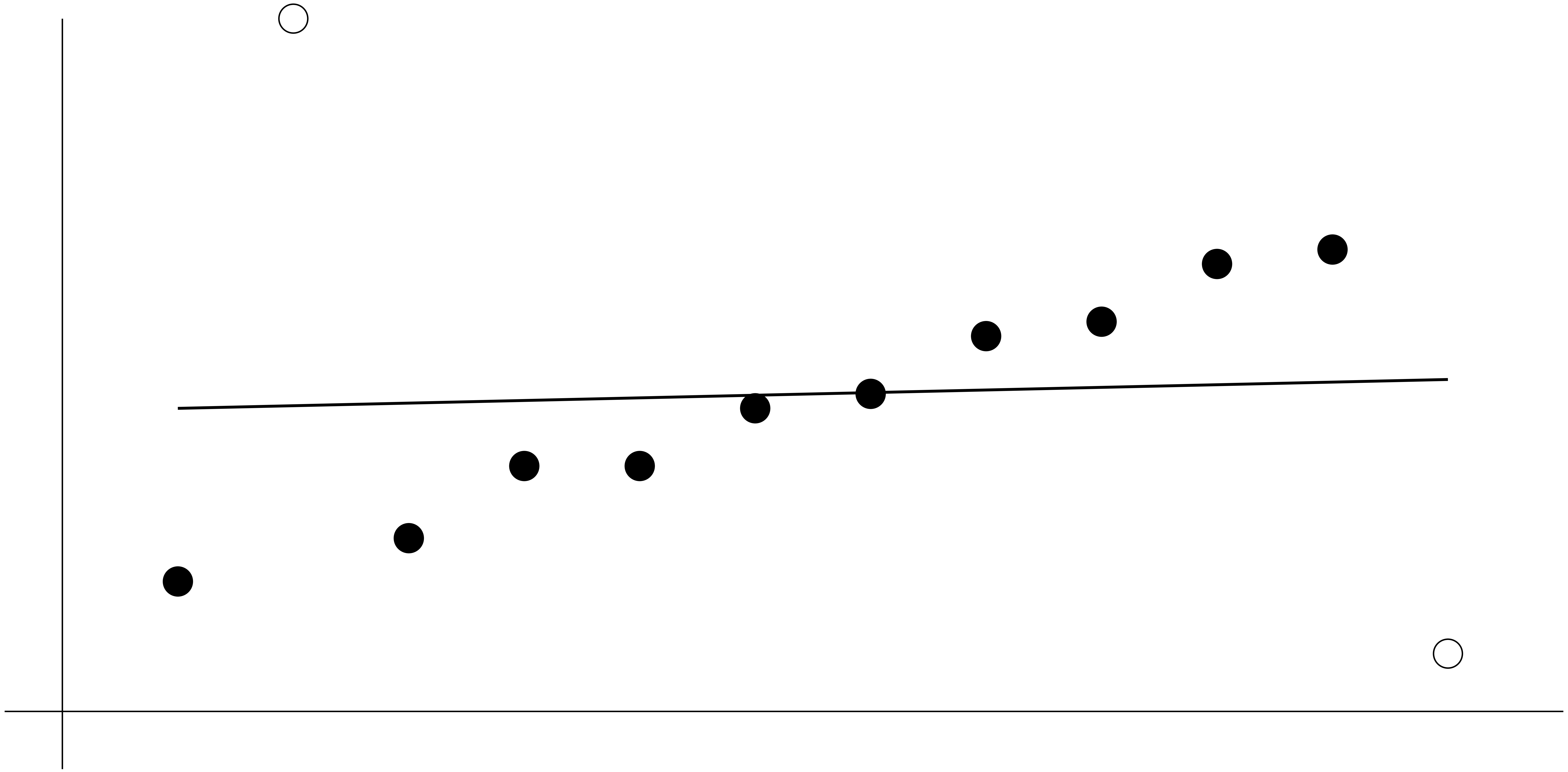


Figure Credit: Hartley & Zisserman

Example 1: Fitting a Line

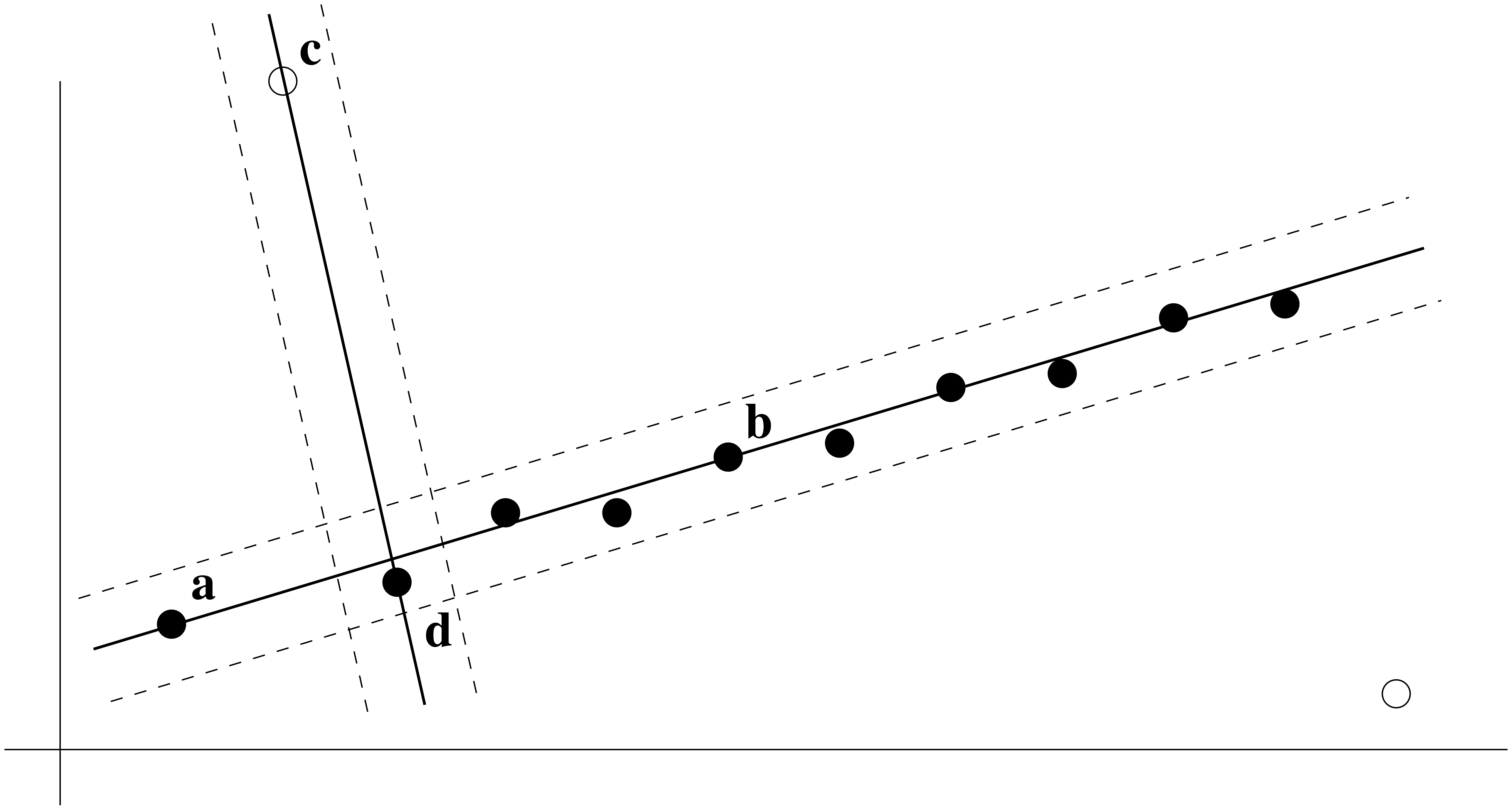


Figure Credit: Hartley & Zisserman

Example 1: Fitting a Line

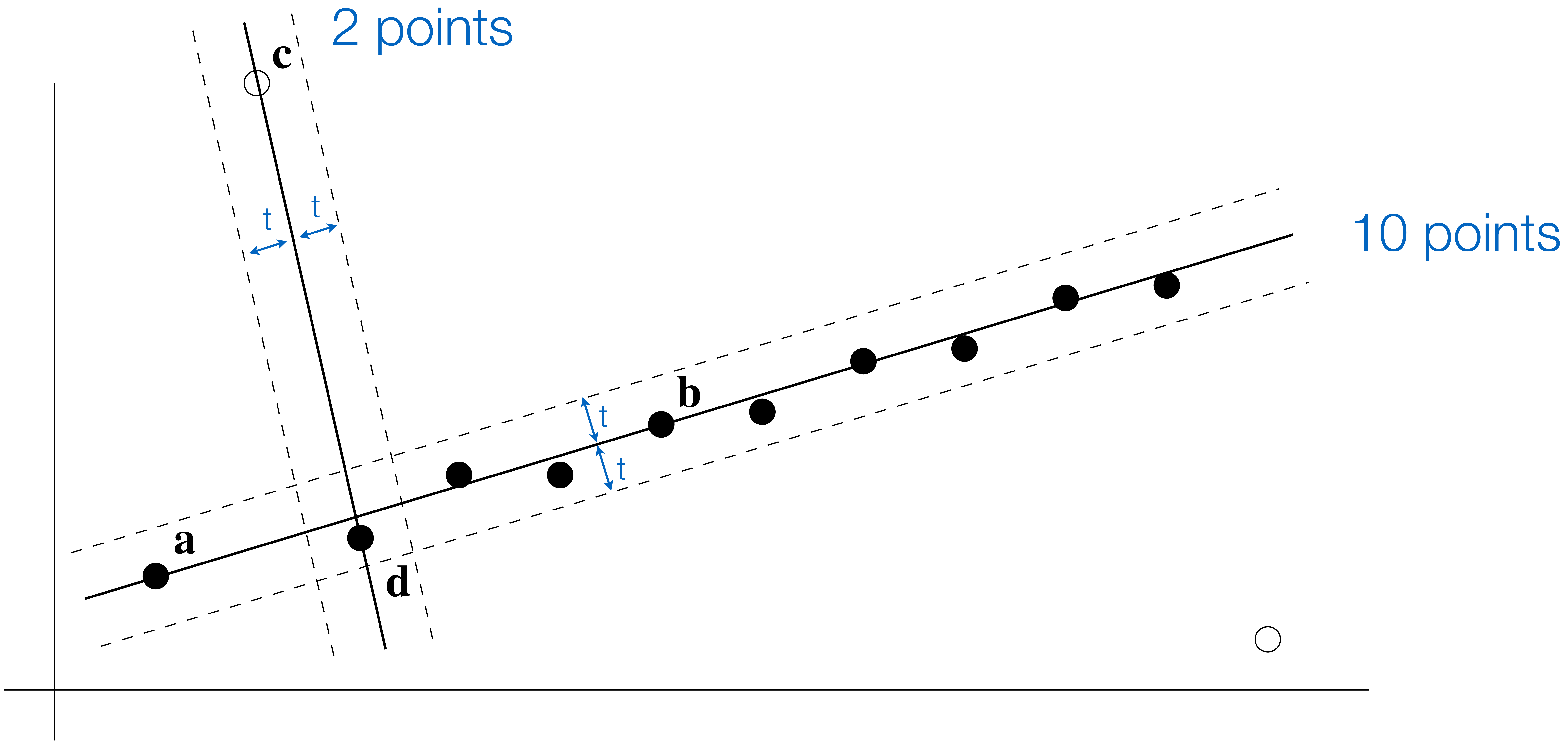


Figure Credit: Hartley & Zisserman

Algorithm 10.4

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

Algorithm 15.4: RANSAC: fitting lines using random sample consensus

Determine:
- n — the smallest number of points required
- k — the number of iterations required
- t — the threshold used to identify a point that fits well
- d — the number of nearby points required to assert a model fits well

Until k iterations have occurred
- Draw a sample of n points from the data uniformly and at random
- Fit to that set of n points
- For each data point outside the sample
  - Test the distance from the point to the line against t; if the distance from the point to the line is less than t, the point is close
- end
- If there are d or more points close to the line then there is a good fit. Refit the line using all these points.

end

Use the best fit from this collection, using the fitting error as a criterion

RANSAC: Fitting Lines Using Random Sample Consensus

RANSAC: How many samples?

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

Let n be the number of points needed to define hypothesis

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

Suppose k samples are chosen

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

RANSAC: How many samples?

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

Let n be the number of points needed to define hypothesis

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

Suppose k samples are chosen

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

$$\omega^n$$

The probability that all k samples fail is

RANSAC: How many samples?

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

Let n be the number of points needed to define hypothesis

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

Suppose k samples are chosen

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

$$\omega^n$$

The probability that all k samples fail is

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

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

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

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

Figure Credit: Hartley & Zisserman

After RANSAC

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

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

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

Example 2: Fitting a Line

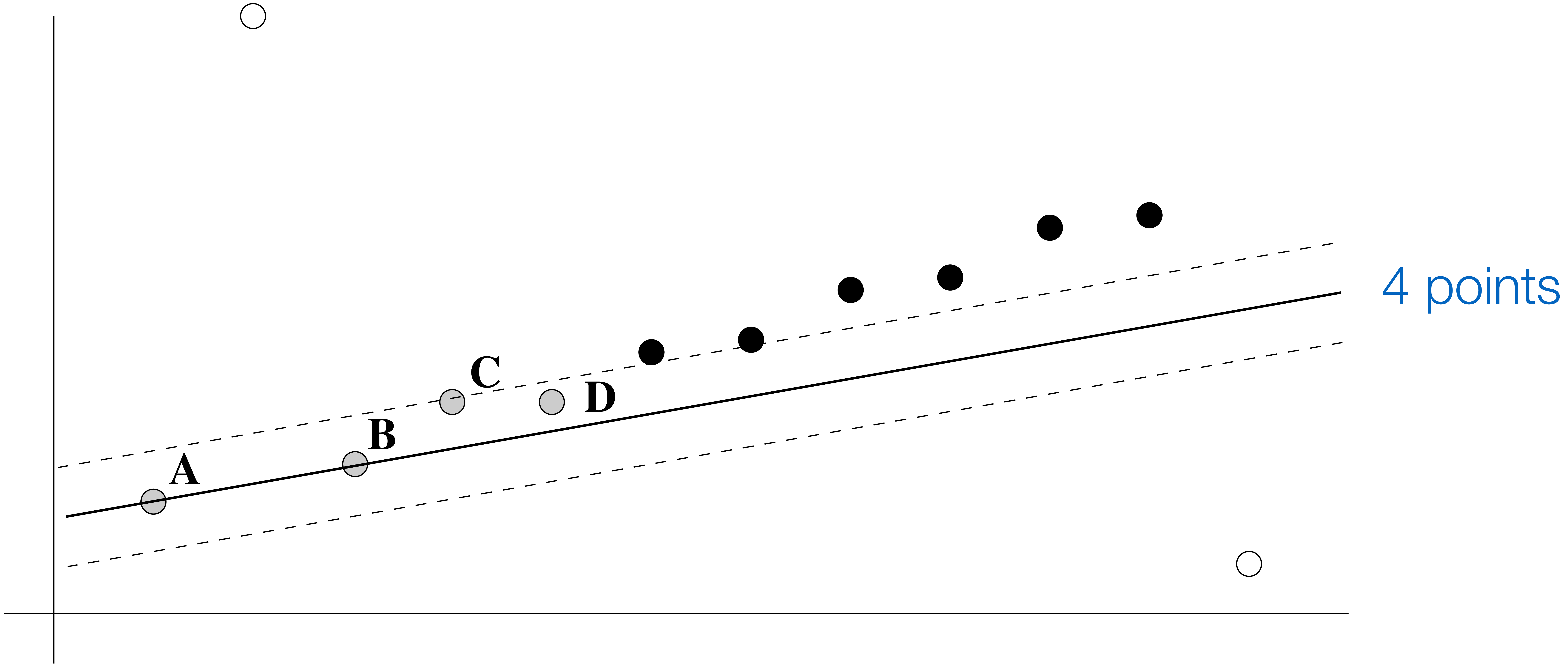


Figure Credit: Hartley & Zisserman

Example 2: Fitting a Line

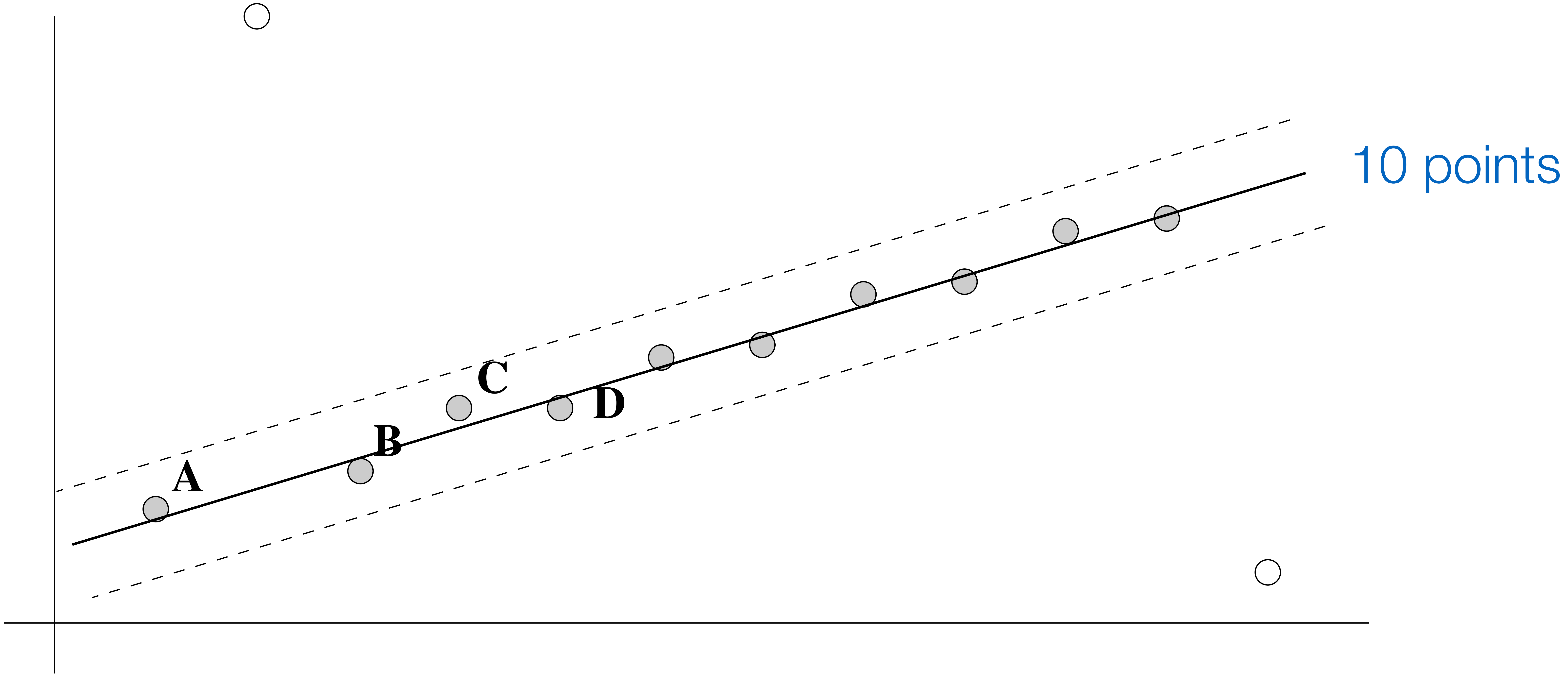


Figure Credit: Hartley & Zisserman

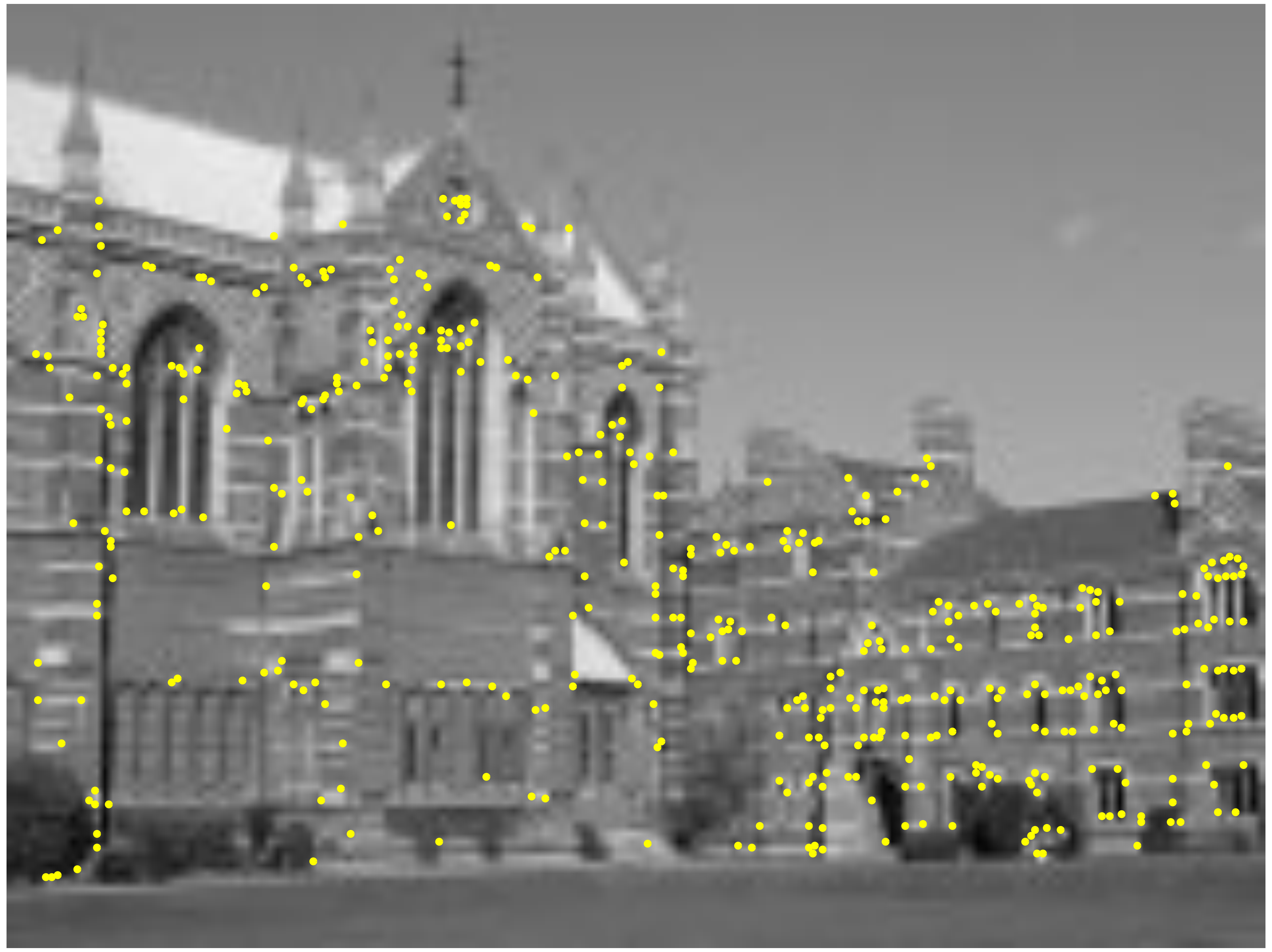
Example 3: Automatic Matching of Images

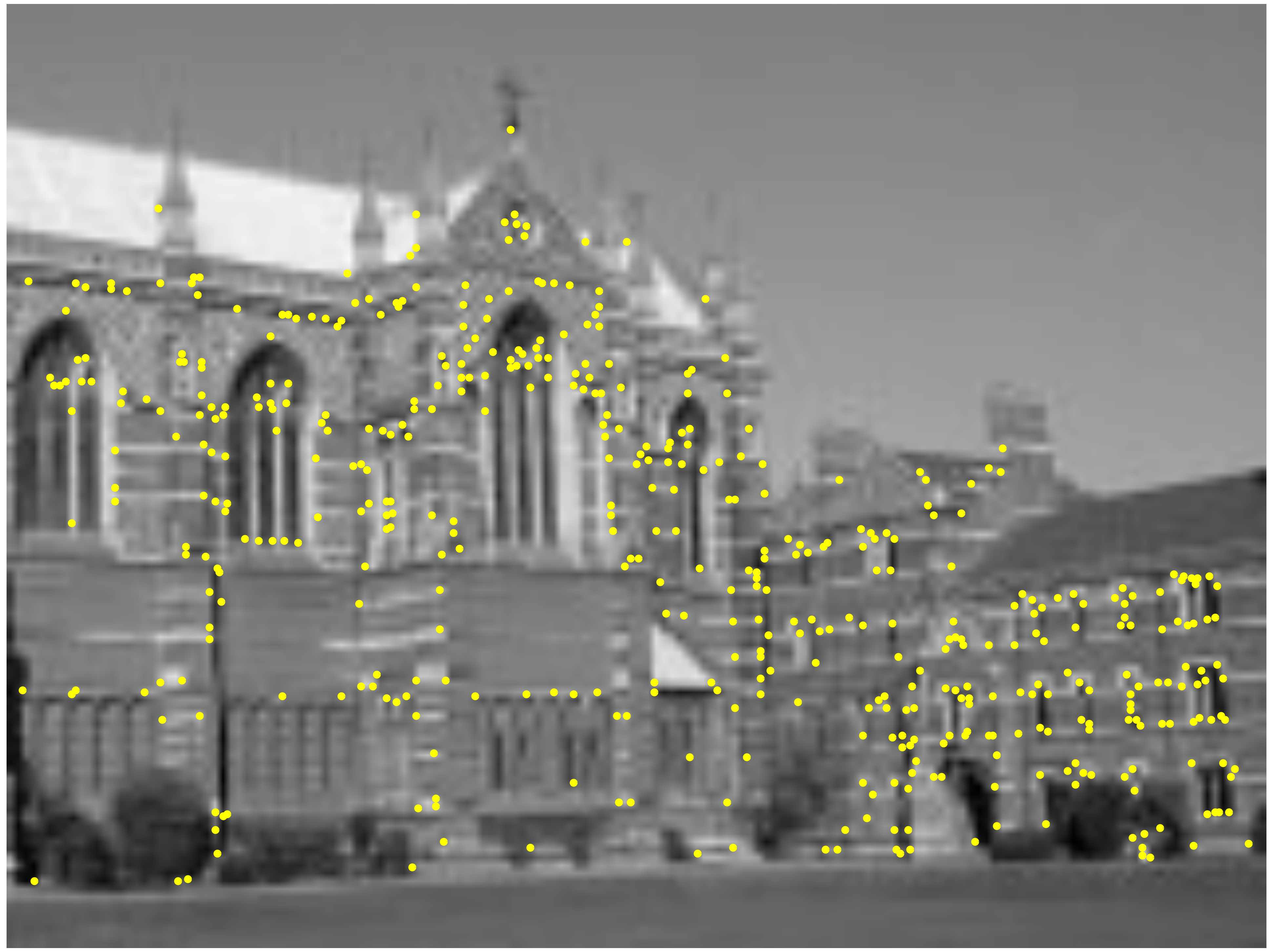
— How to get correct correspondences without human intervention?
— Can be used for image stitching or automatic determination of epipolar geometry

Figure Credit: Hartley & Zisserman

Example 3: Feature Extraction

- Find features in pair of images using Harris corner detector
- Assumes images are roughly the same scale



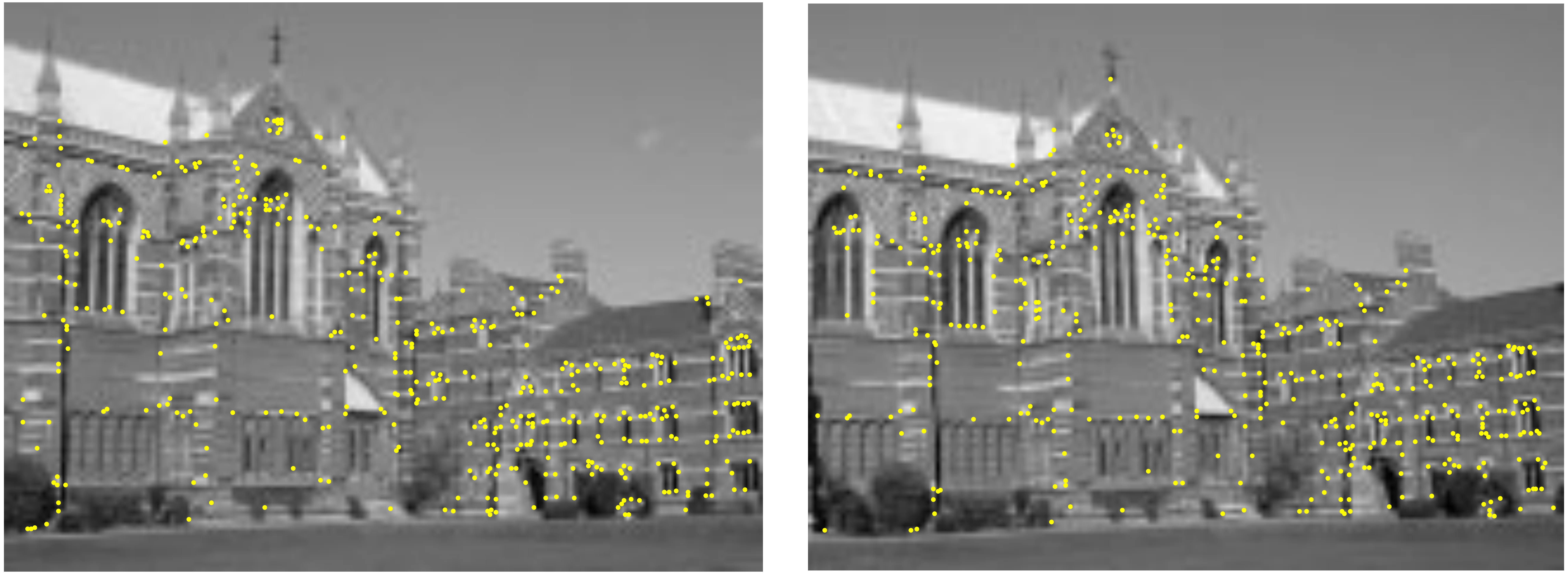


≈ 500 corner features found in each image

Figure Credit: Hartley & Zisserman

Example 3: Finding Feature Matches

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



≈ 500 corner features found in each image

Figure Credit: Hartley & Zisserman

Example 3: Initial Match Hypothesis

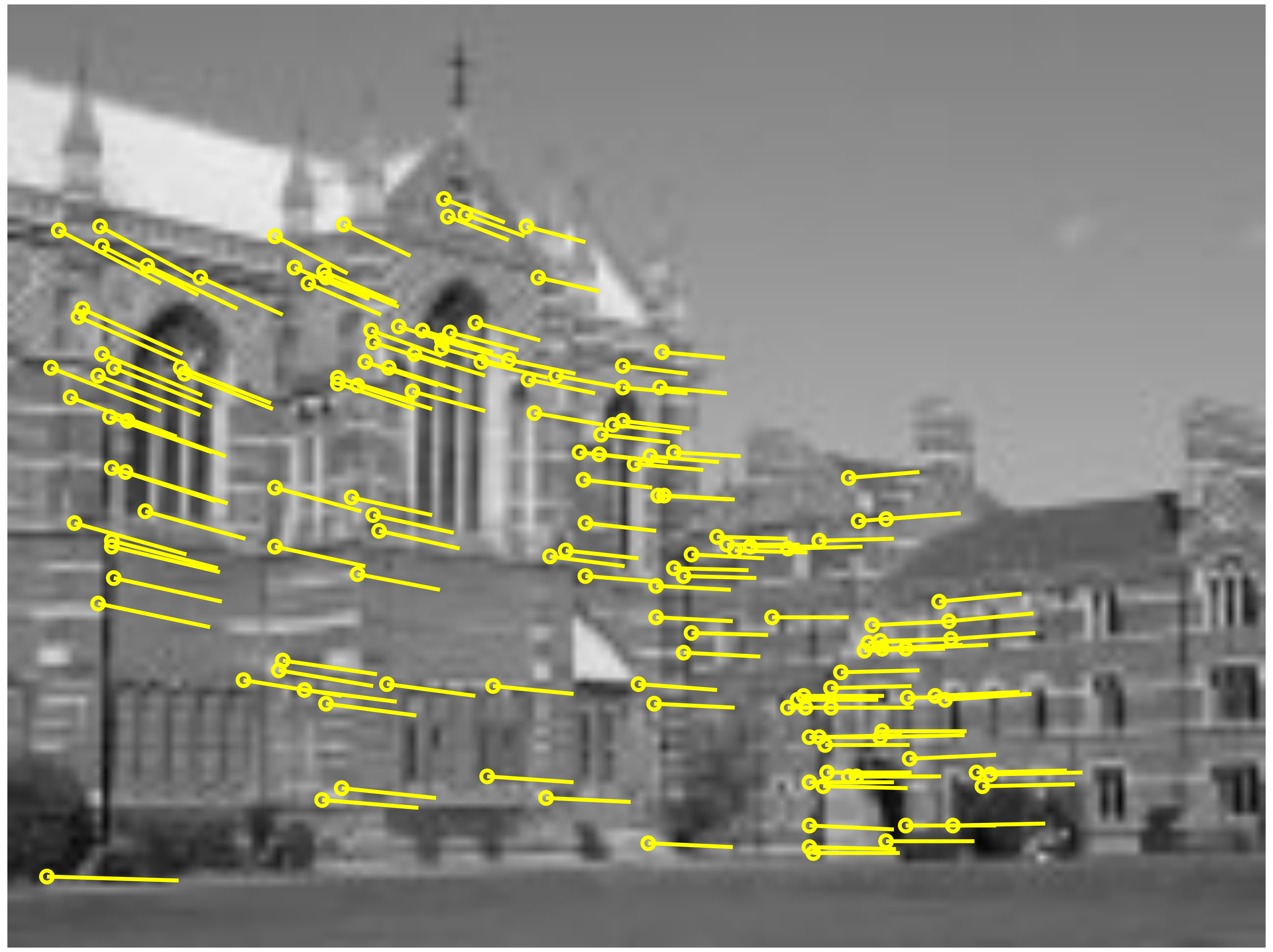
268 matched features (over SSD threshold) superimposed on left image (pointing to locations of corresponding feature in right image)

Figure Credit: Hartley & Zisserman

Example 3: Outliers & Inliers after RANSAC

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

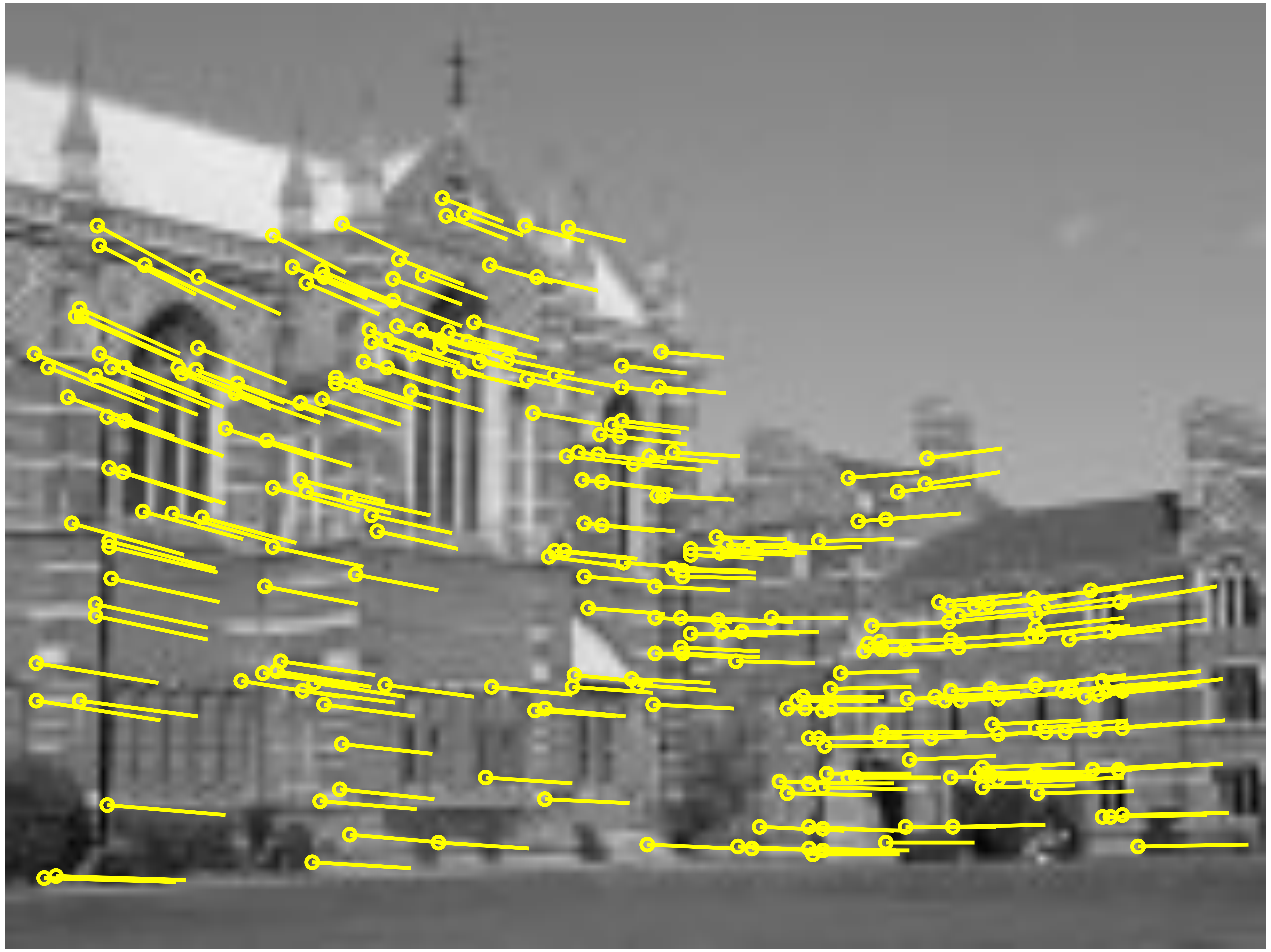
117 **outliers**



151 **inliers**

Figure Credit: Hartley & Zisserman

Example 3: Final Matches



final set of 262 matches

Figure Credit: Hartley & Zisserman

Discussion of RANSAC

Advantages:

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

— Easy to implement and easy to calculate its failure rate

Disadvantages:

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

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

The Hough transform can handle high percentage of outliers

Example: Photo Tourism

Figure credit: Snavely et al. 2006

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

Uses both SIFT and RANSAC

Example: Photo Tourism

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

Example: Photo Tourism

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