Image Processing & Vision

Lecture 06: Model Fitting – RANSAC

Hak Gu Kim

hakgukim@cau.ac.kr

Immersive Reality & Intelligent Systems Lab (IRIS LAB)

Graduate School of Advanced Imaging Science, Multimedia & Film (GSAIM)

Chung-Ang University (CAU)

10 Apr. 2023

Re-cap: Local Feature

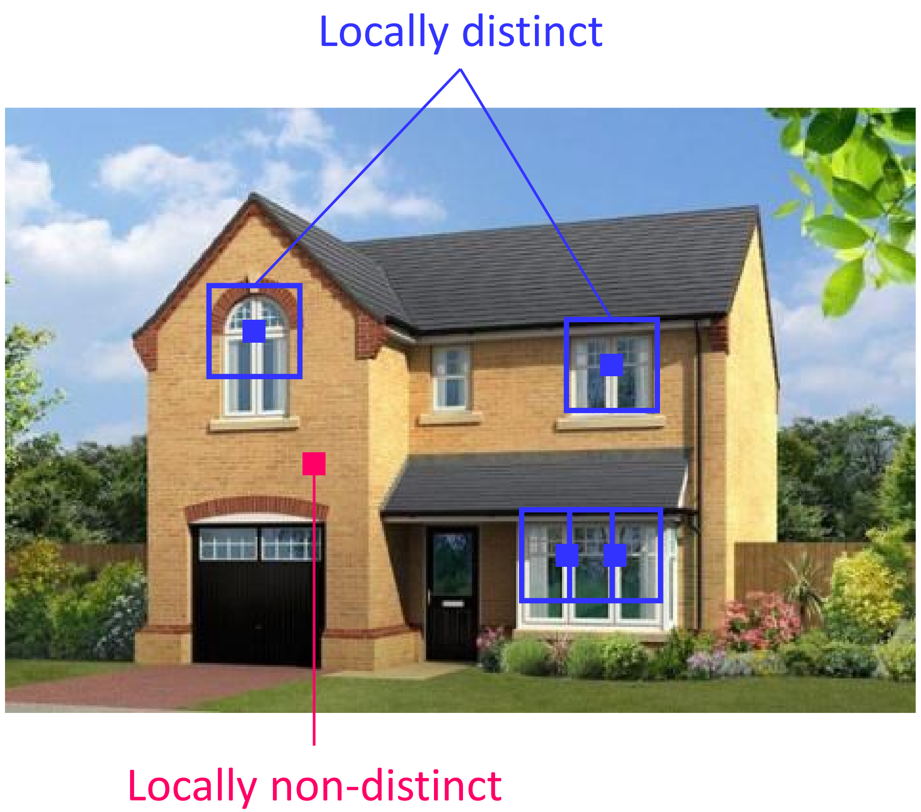
- **Keypoint** is an image location at which a descriptor is computed

— Locally distinct points

— Easily localizable and identifiable

- The feature **descriptor** summarizes the local structure around the keypoint

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



Re-cap: Scale Invariant Feature Transform (SIFT)

- The scale-invariant feature transform (SIFT) is an algorithm to **detect**, **describe**, and **match** local features in images
- SIFT describes both a **detector** and **descriptor**

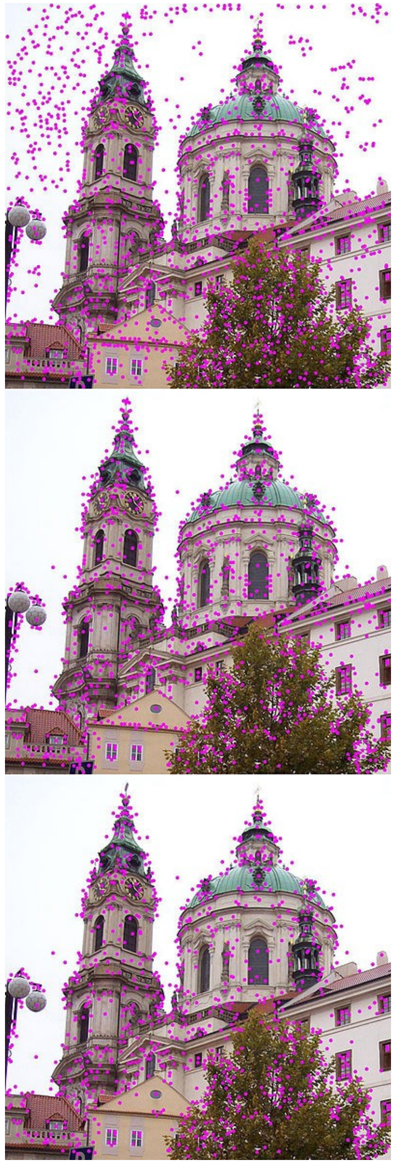
— Applications: object recognition, robotic mapping and navigation, image stitching, 3D modeling, gesture recognition, video tracking, *etc*.

① Multi-scale extrema detection

② Keypoint localization

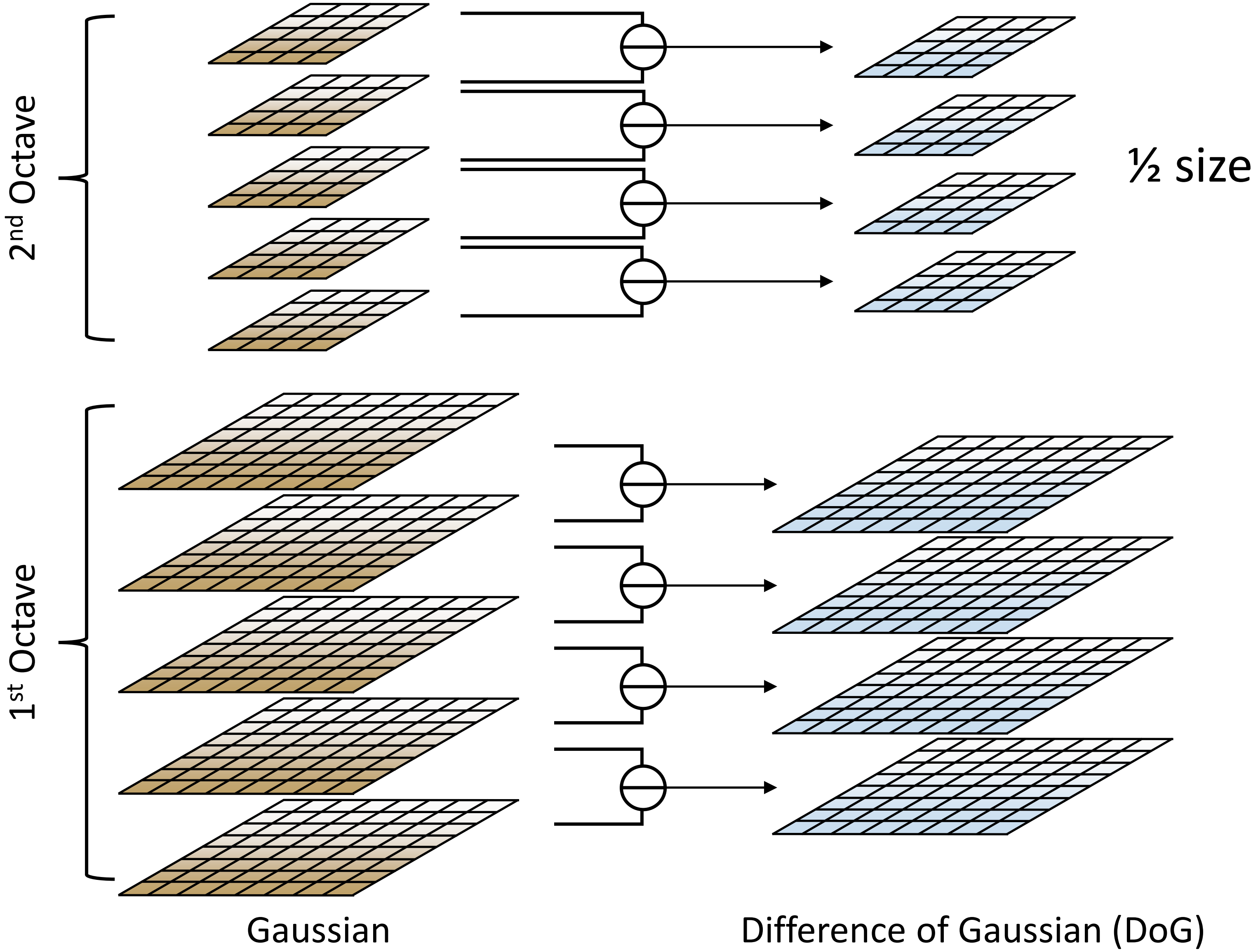
③ Orientation assignment

④ Keypoint descriptor



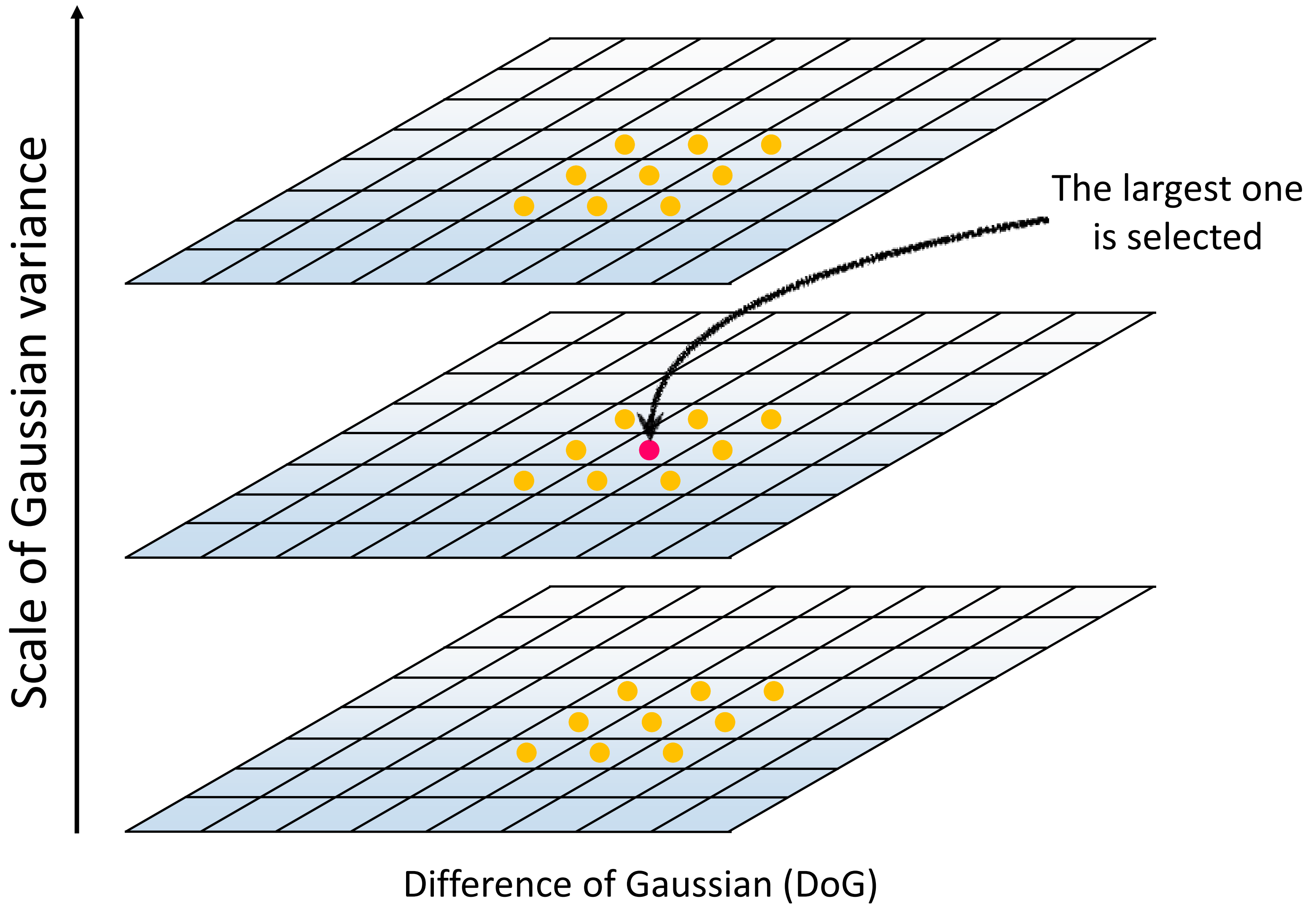
Re-cap: Scale Invariant Feature Transform (SIFT)

- Begin by detecting points of interest (i.e., keypoints)
  - The image is convolved with Gaussian filters at different scales
  - Then the difference of successive Gaussian-blurred images are taken



Re-cap: Scale Invariant Feature Transform (SIFT)

- Begin by detecting points of interest (i.e., keypoints)
  - The image is convolved with Gaussian filters at different scales
  - Then the difference of successive Gaussian-blurred images are taken
  - Keypoints are taken as maxima/minima of the DoG at multiple scales



Re-cap: Scale Invariant Feature Transform (SIFT)

- Scale-space extrema detection produces too many keypoint candidates, some of which are **unstable**
- In keypoint localization, we reject points which are **low contrast** (and are therefore **sensitive to noise**) or **poorly localized** along an edge

- How do we decide whether a keypoint is poorly localized or well-localized?

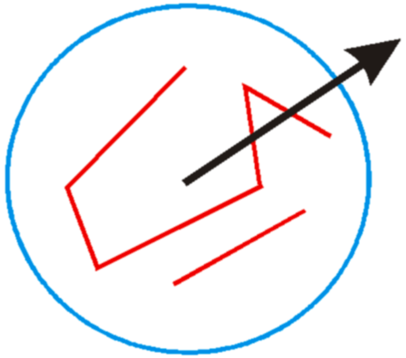
— In SIFT, compute the ratio of the eigenvalues of covariance matrix and check if it is greater than a threshold

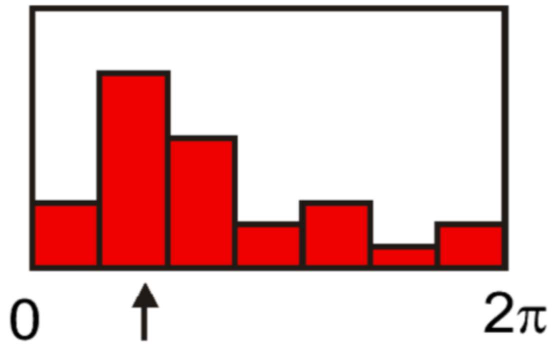
Re-cap: Scale Invariant Feature Transform (SIFT)

- Each keypoint is assigned one or more orientations based on local image gradient directions
- This is the key step in achieving invariance to rotation on the Gaussian-smoothed image, L

$$m(x,y) = \sqrt{\big(L(x+1,y) - L(x-1,y)\big)^2 + \big(L(x,y+1) - L(x,y-1)\big)^2}$$

$$\theta(x,y) = \tan^{-1}\Big[\big(L(x+1,y) - L(x-1,y)\big) / \big(L(x,y+1) - L(x,y-1)\big)\Big]$$





Re-cap: Scale Invariant Feature Transform (SIFT)

- Thresholded image gradients are sampled over 16 x 16 array of locations in scale space (weighted by a Gaussian with sigma half the size of the window)
- Create array of orientation histograms
- 8 Orientations x (4 x 4) histogram array

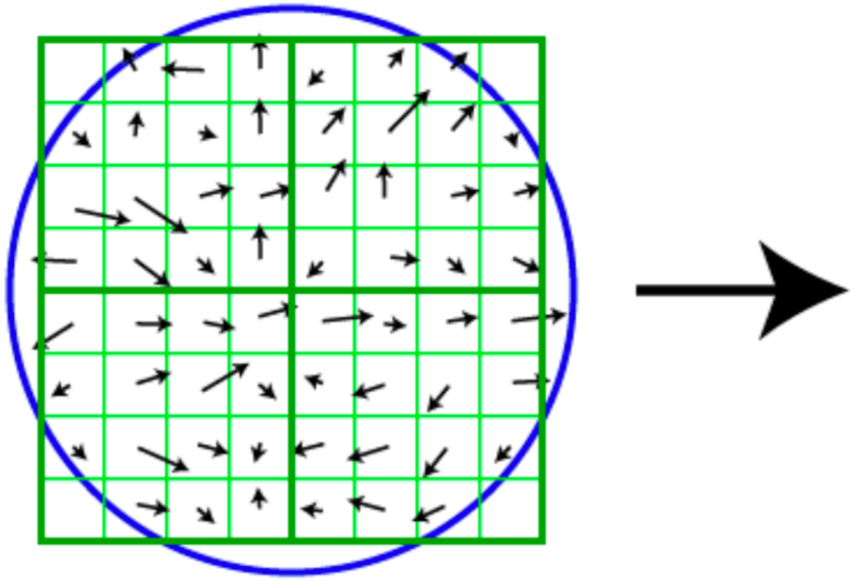
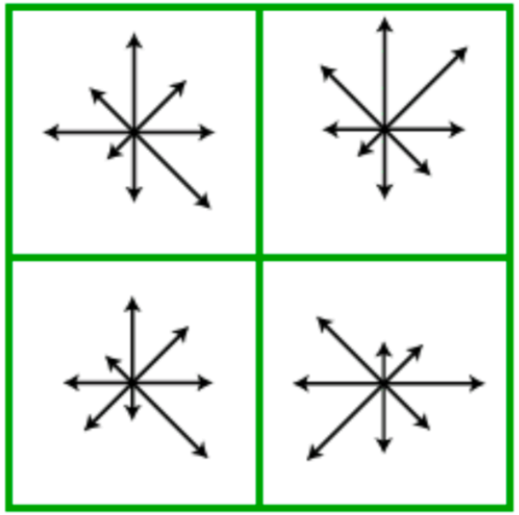


Image gradients



Keypoint descriptor

Re-cap: Scale Invariant Feature Transform (SIFT)

① **Scale-space representation and local extreme detection**

- Use DoG/LoG Pyramid
- 3 scales/octave, down-sample by factor of 2 each octave

② **Keypoint localization**

- Select stable keypoints by thresholding on the magnitude of extrema and ratio of principal curvatures

③ **Keypoint orientation assignment**

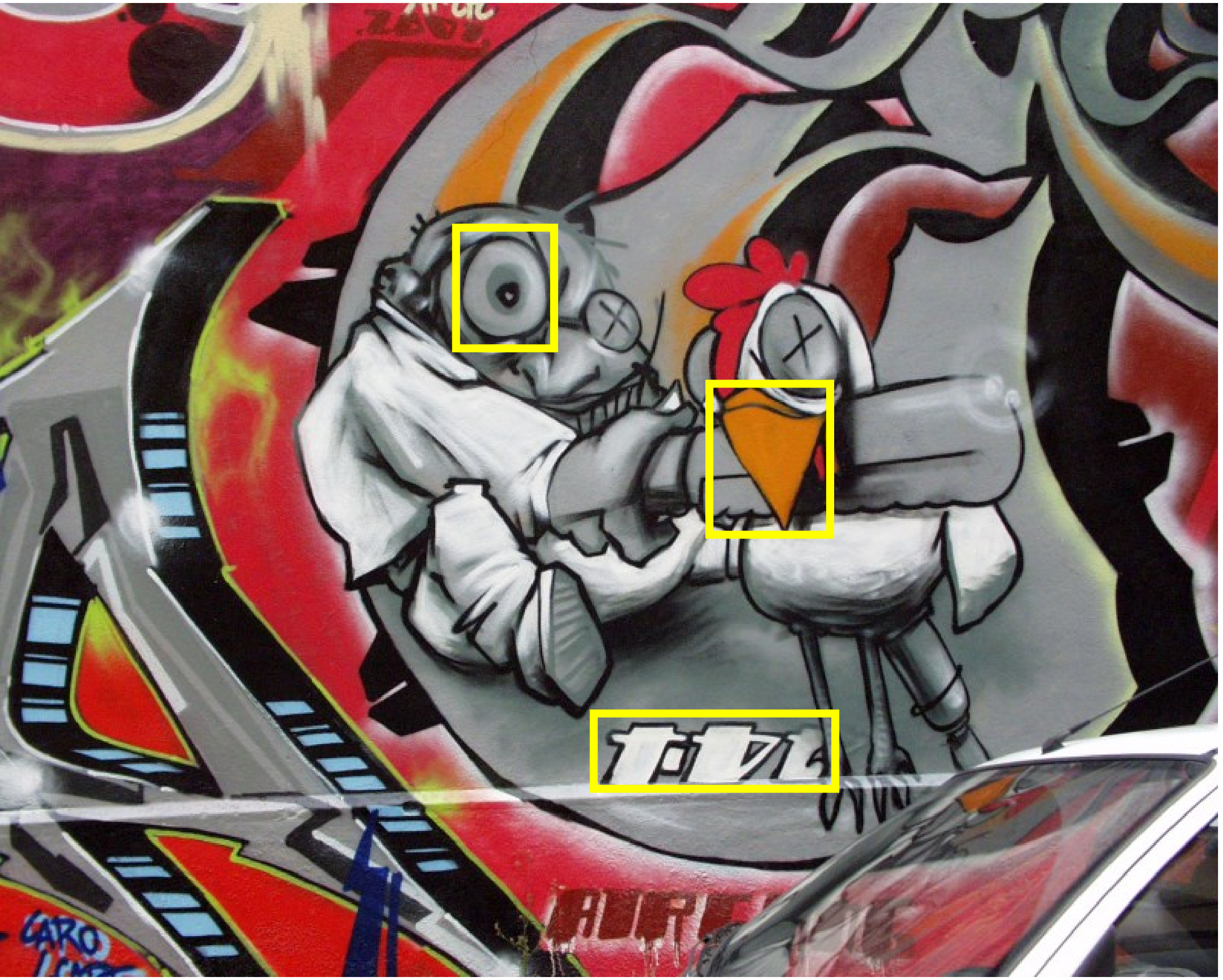
- Based on histogram of local image gradient directions

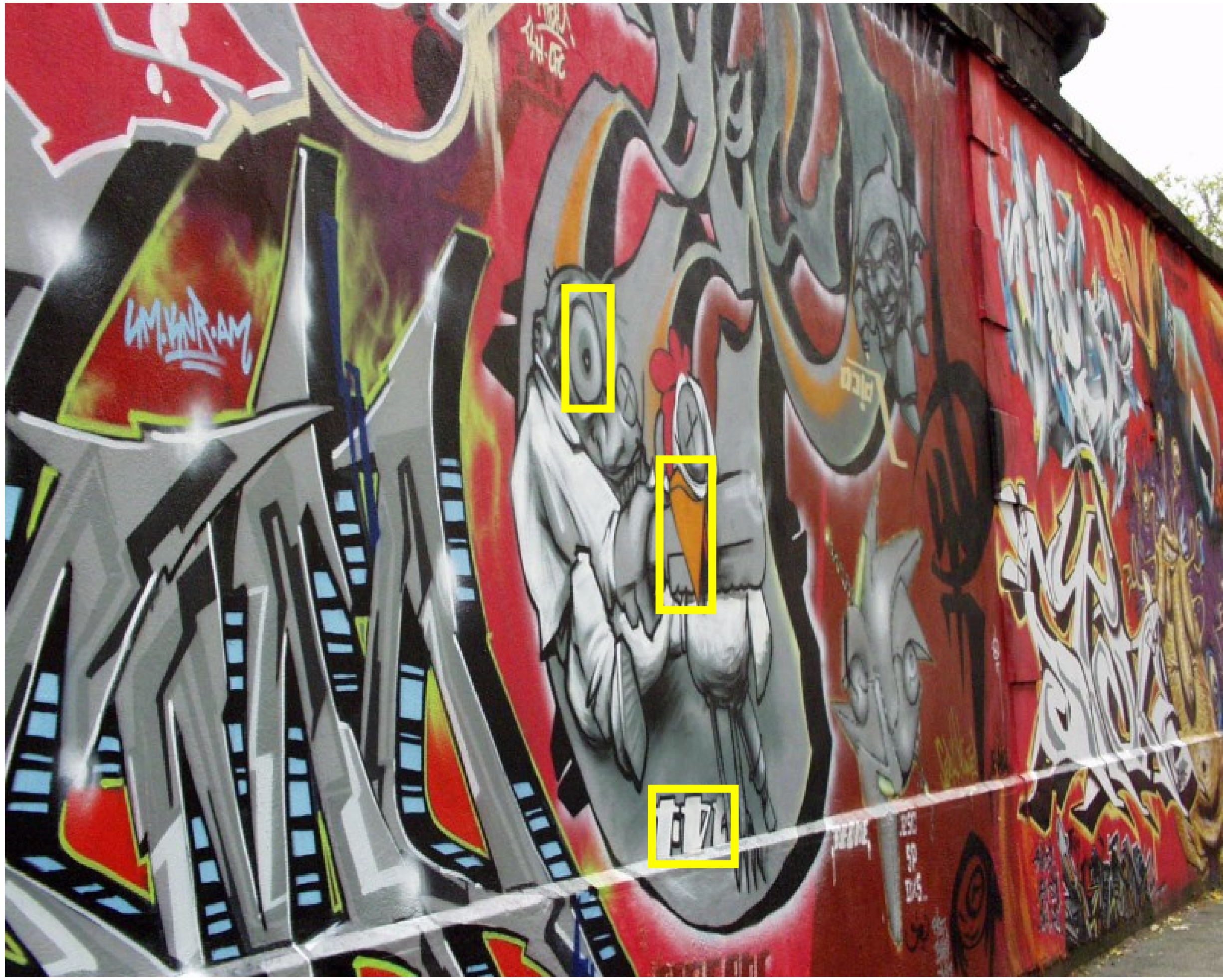
④ **Keypoint descriptor**

- Histogram of local gradient directions – Vector with $8 \times (4 \times 4) = 128$ dimensions
- Vector normalization

Re-cap: Application of Good Local Features

- Feature Matching





Re-cap: Application of Good Local Features

- Example of Feature Matching
  - Composed of **more than 1,000 images** and carefully assembled over the ensuing months

NASA's Curiosity rover captured its highest-resolution panorama of the Martian surface

Re-cap: Application of Good Local Features

- Object Tracking & Recognition

https://www.youtube.com/watch?v=E5cyffF7DNo

Topics

- Model Fitting – RANSAC

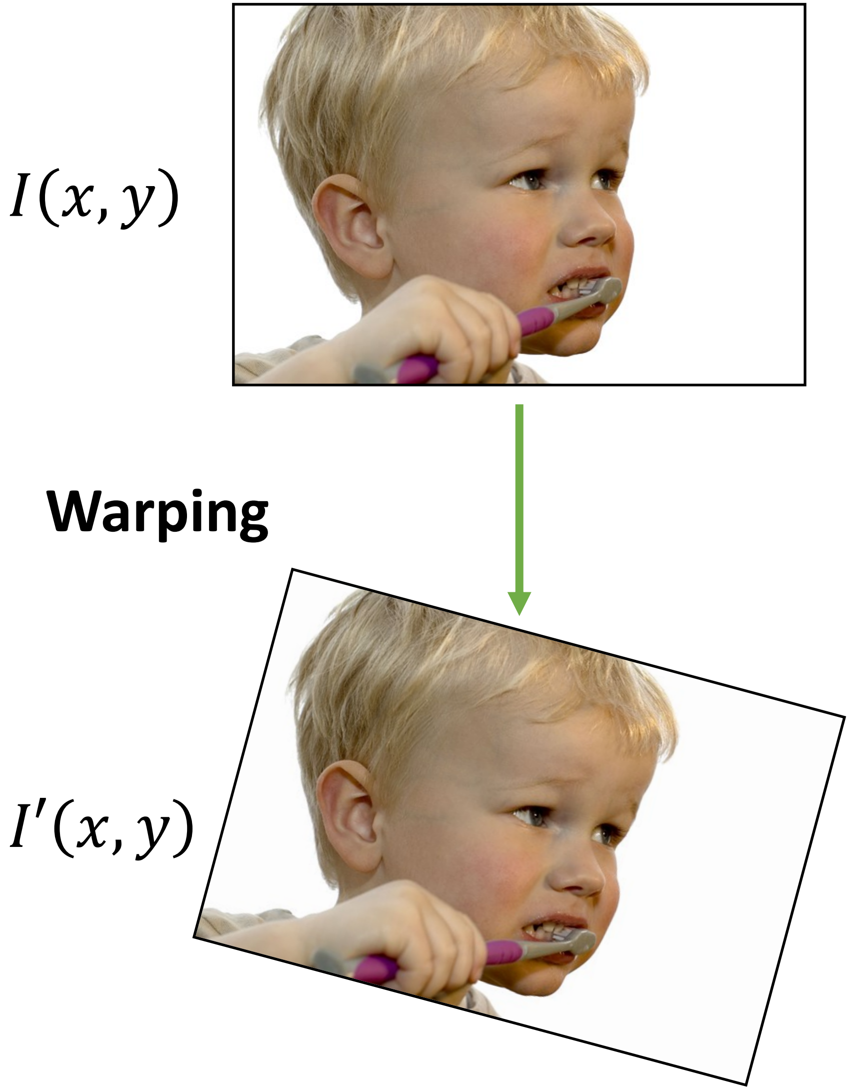
*Note: Many of these slides in this course were adapted from Computer Vision at CMU (16-385) and UBC (CPSC425)

Transformation: **Warping**

- The model (warping) gives us a way to transform any pixel in the original image $I(x, y)$ to the corresponding image $I'(x, y)$
- We will call this "**Warping**" a "**Model**"

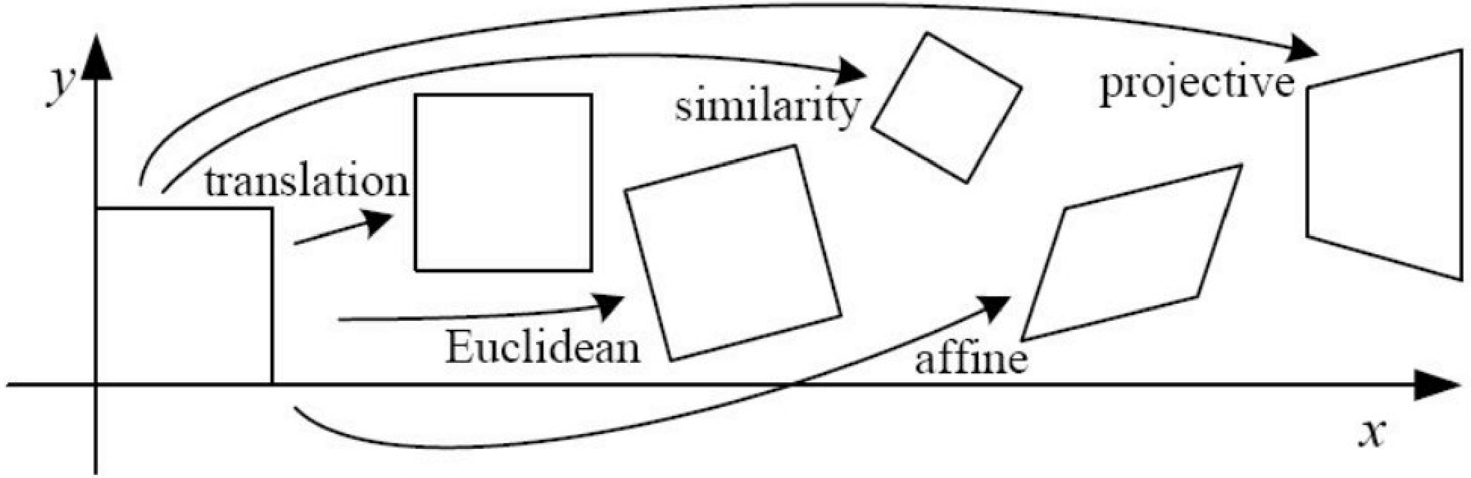
$$\begin{bmatrix} X' \\ Y' \\ 1 \end{bmatrix} = M \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$I(x, y) = I'(x', y')$$



Change domain of image function

Forms of the **Model (Warping)**

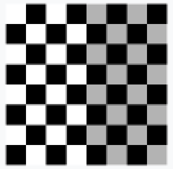
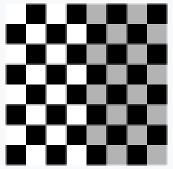
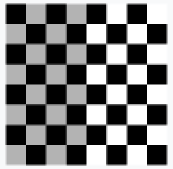


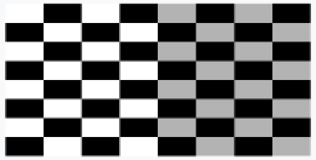
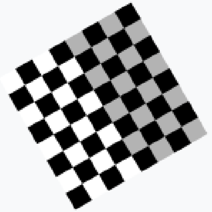
Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$			
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$			
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times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

Affine Model

- In Euclidean geometry, an affine transformation is a geometric transformation that **preserves lines and parallelism**

— But not necessarily distances and angles

Identity (transform to original image)	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	
Translation	$\begin{bmatrix} 1 & 0 & v_x > 0 \\ 0 & 1 & v_y = 0 \\ 0 & 0 & 1 \end{bmatrix}$	
Reflection	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	

Scale	$\begin{bmatrix} c_x = 2 & 0 & 0 \\ 0 & c_y = 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	
Rotate	$\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$	 where $\theta = \frac{\pi}{6}$ =30°
Shear	$\begin{bmatrix} 1 & c_x = 0.5 & 0 \\ c_y = 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	

Affine Model

- 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 the 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

Affine Model

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

- We need to have **exact** matches

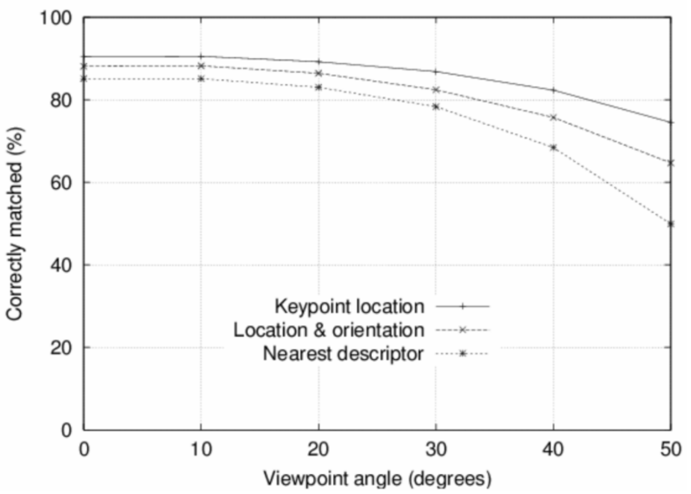
— Only 3 keypoints are needed for recognition, so extra keypoints provide robustness

- It is very **difficult** to find exact match

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

— Image **noise, deformations**, will make this worse

— **Multiple** object instances will make this impossible

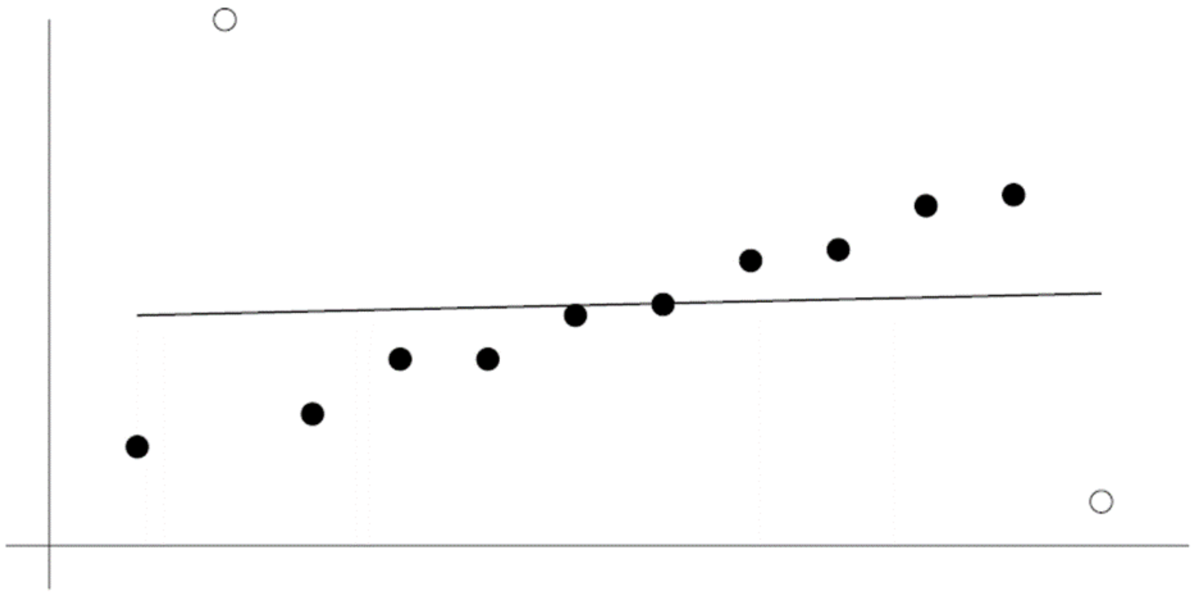


Fitting a Model to Noisy Data

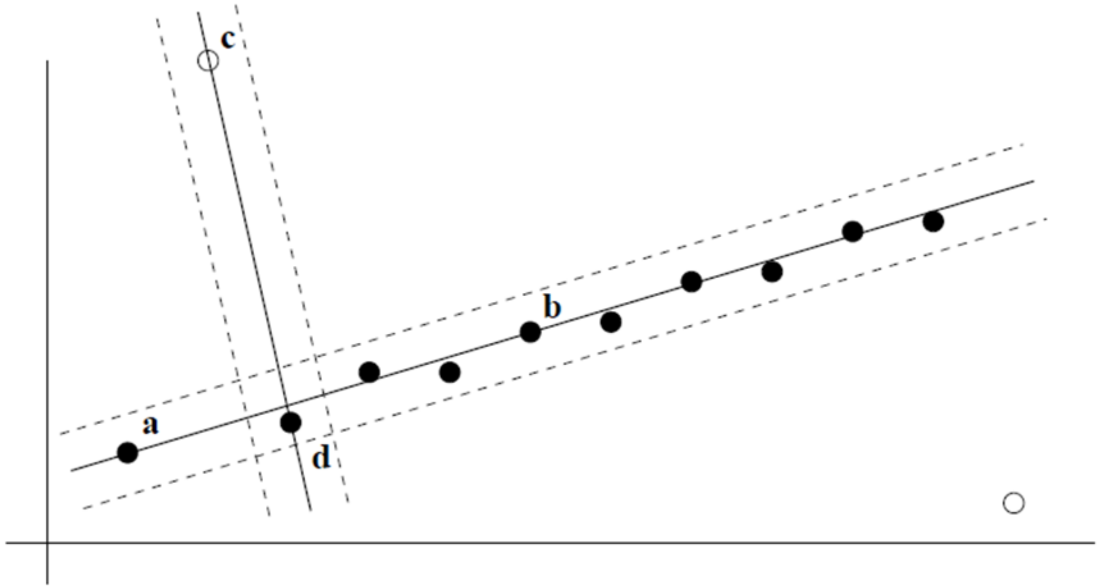
- **Problem of a Least-Square Line Fitting**

— In a line fit to the data

— In a classification of the data into inliers (valid points) and outliers



A least-squares line fitting



RANSAC line fitting

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,
what fraction of pairs will consist entirely of **good data points (inliers)?**

— 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)

- **Objective:** Robust fit of a model to a data set S which contains outliers

- **Algorithm:**

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**

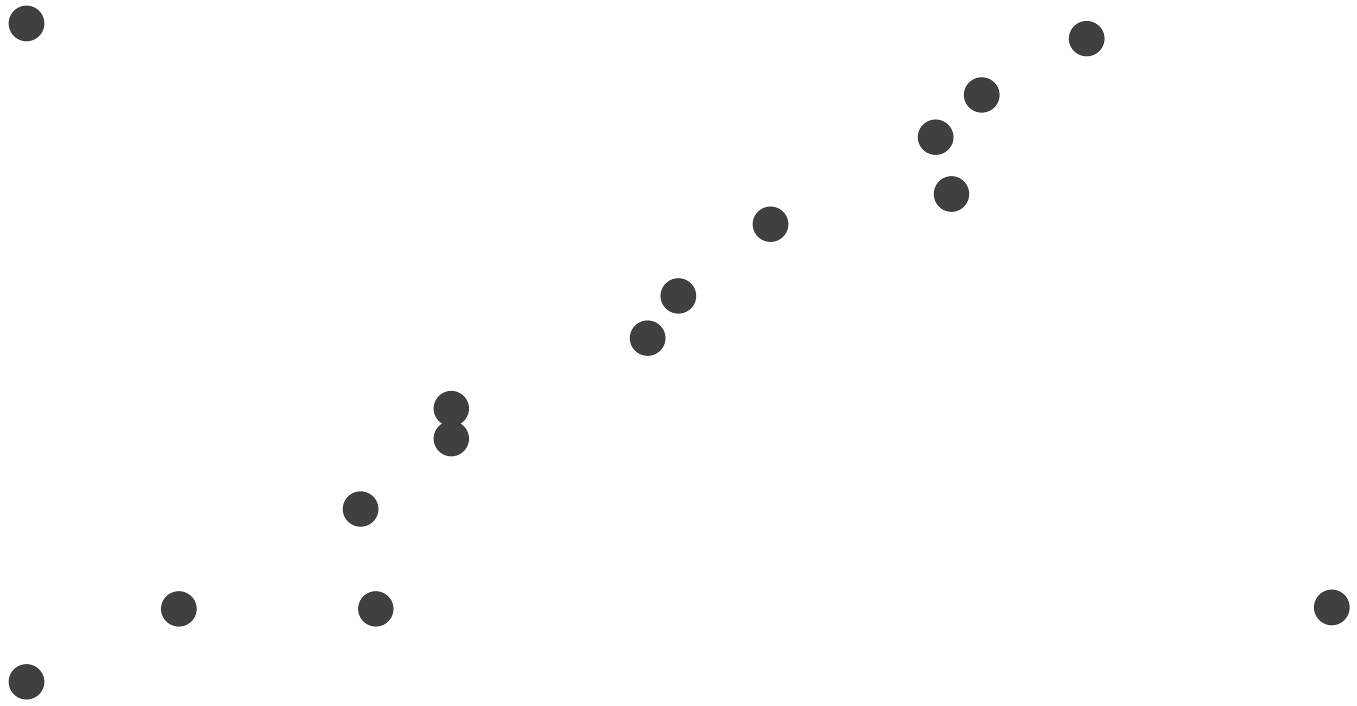
③ Repeat for N samples; model with biggest support is most robust fit.

– Points within distance t of best model are inliers

– Fit final model to all inliers

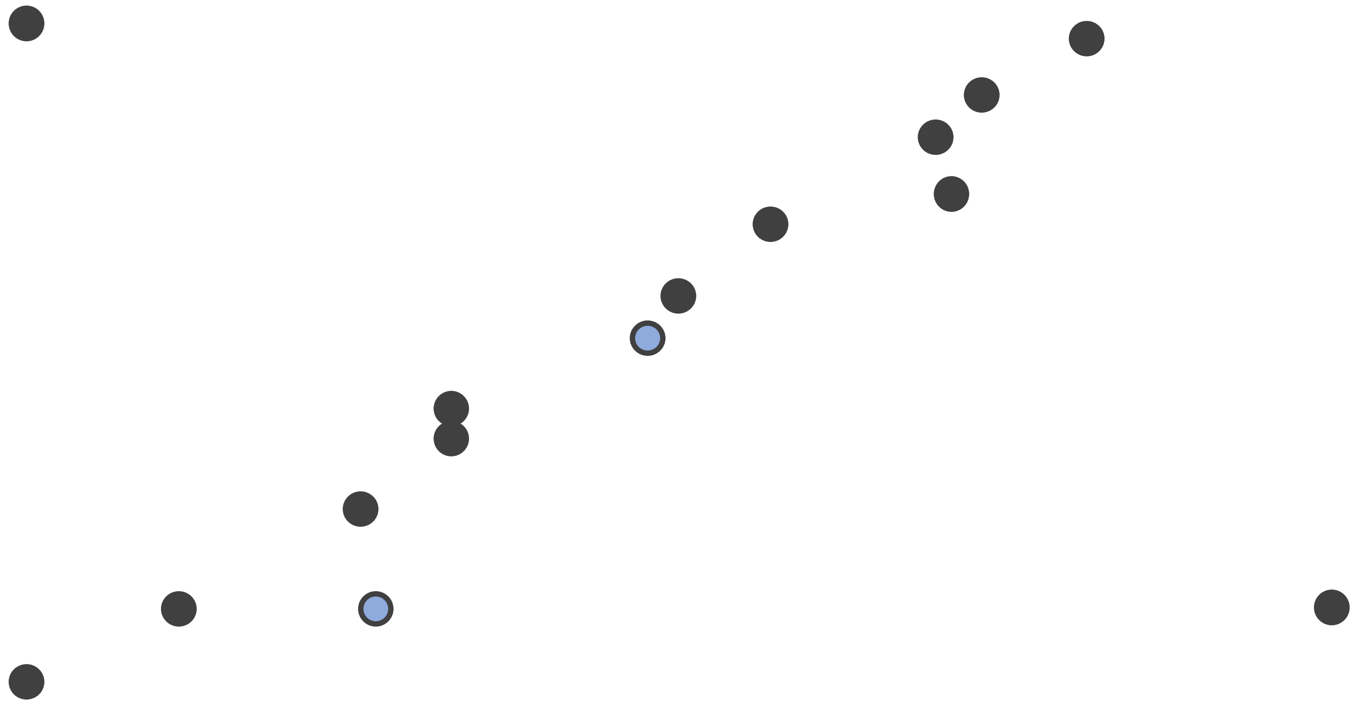
RANSAC (RANdom SAmple Consensus)

- RANSAC Line Fitting Example



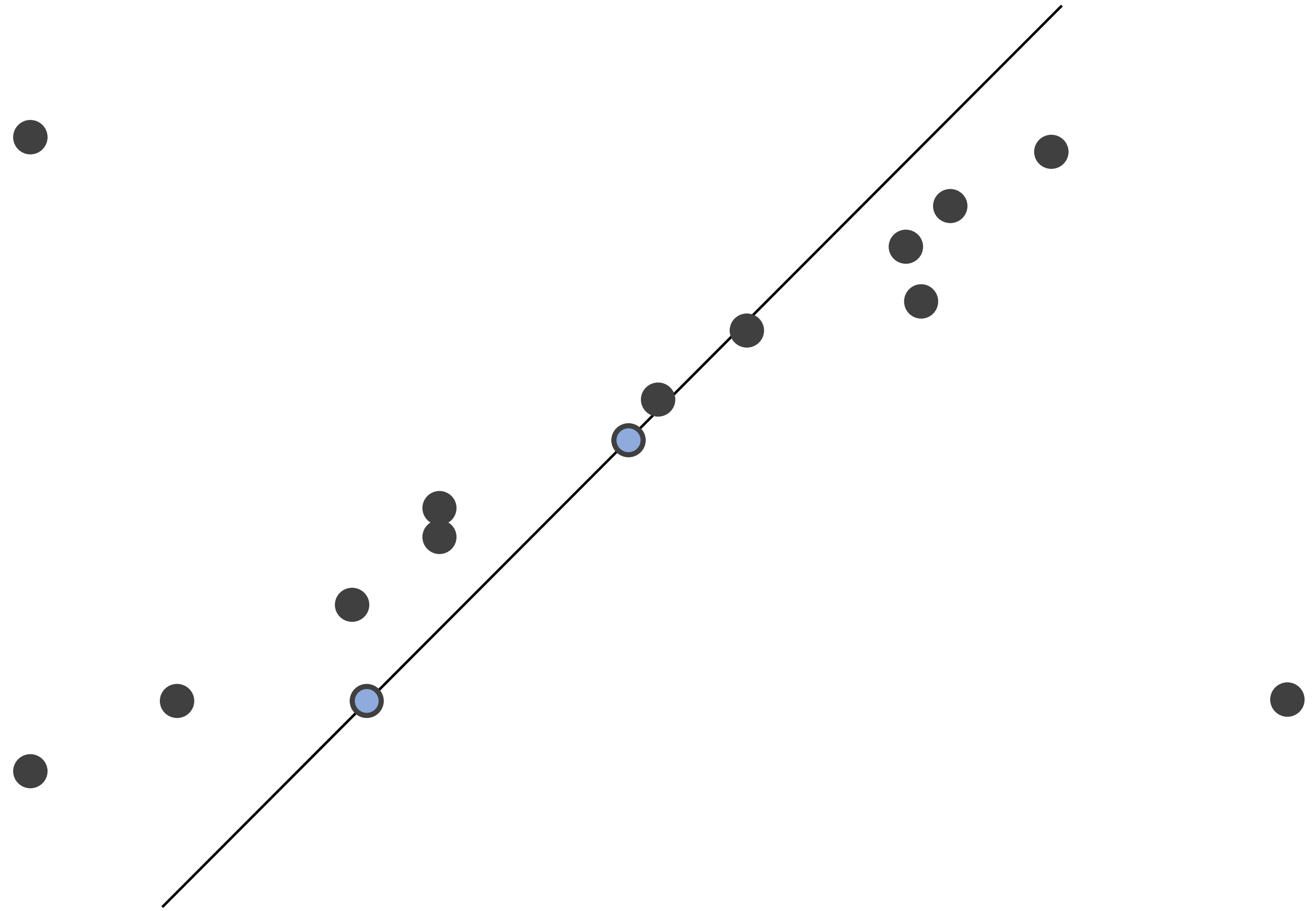
RANSAC (**RAN**dom **SA**mple **C**onsensus)

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



RANSAC (RANdom SAmple Consensus)

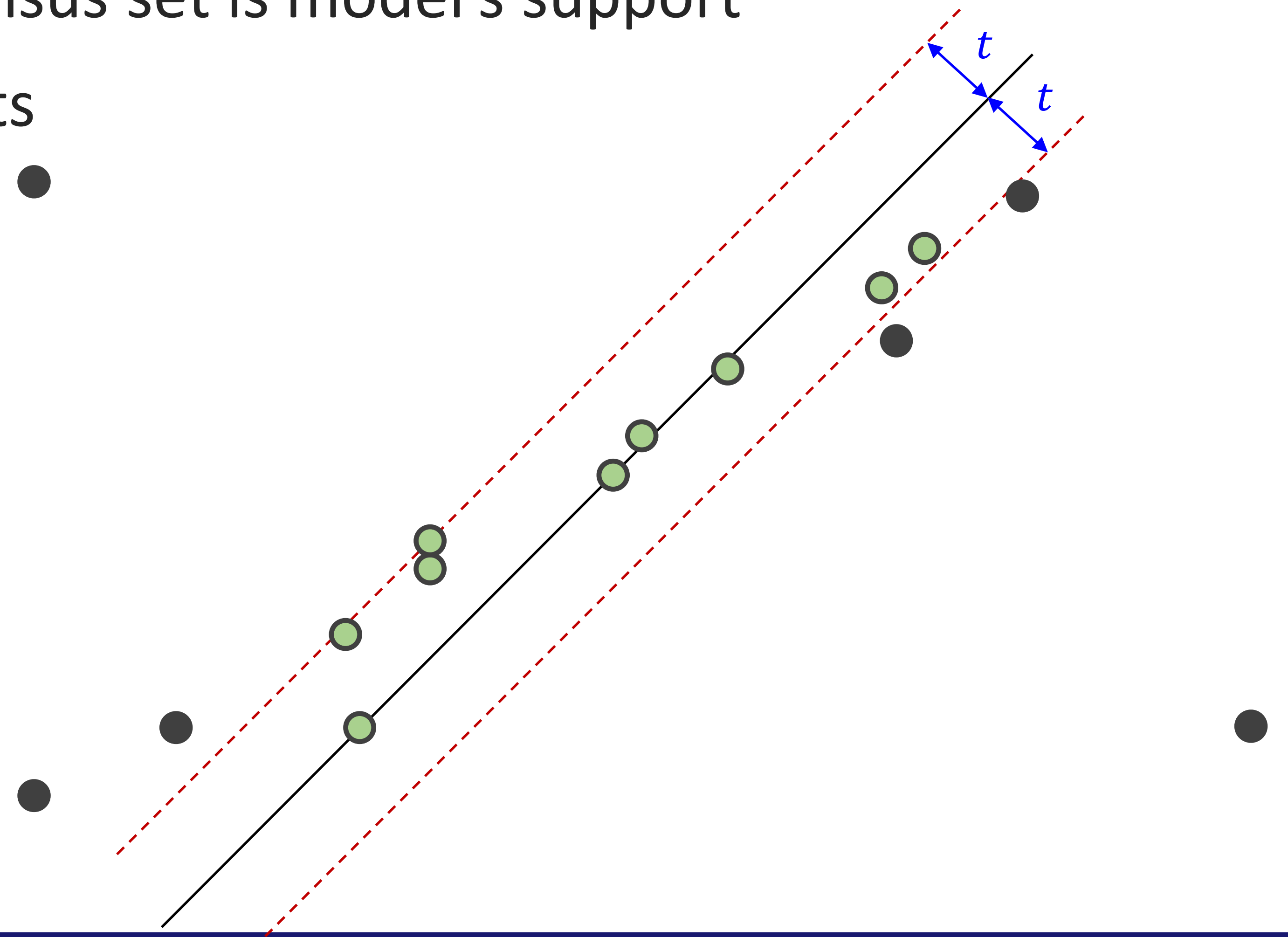
① **Randomly choose minimal subset of data points** necessary to fit model (a sample)



RANSAC (RANdom SAmple Consensus)

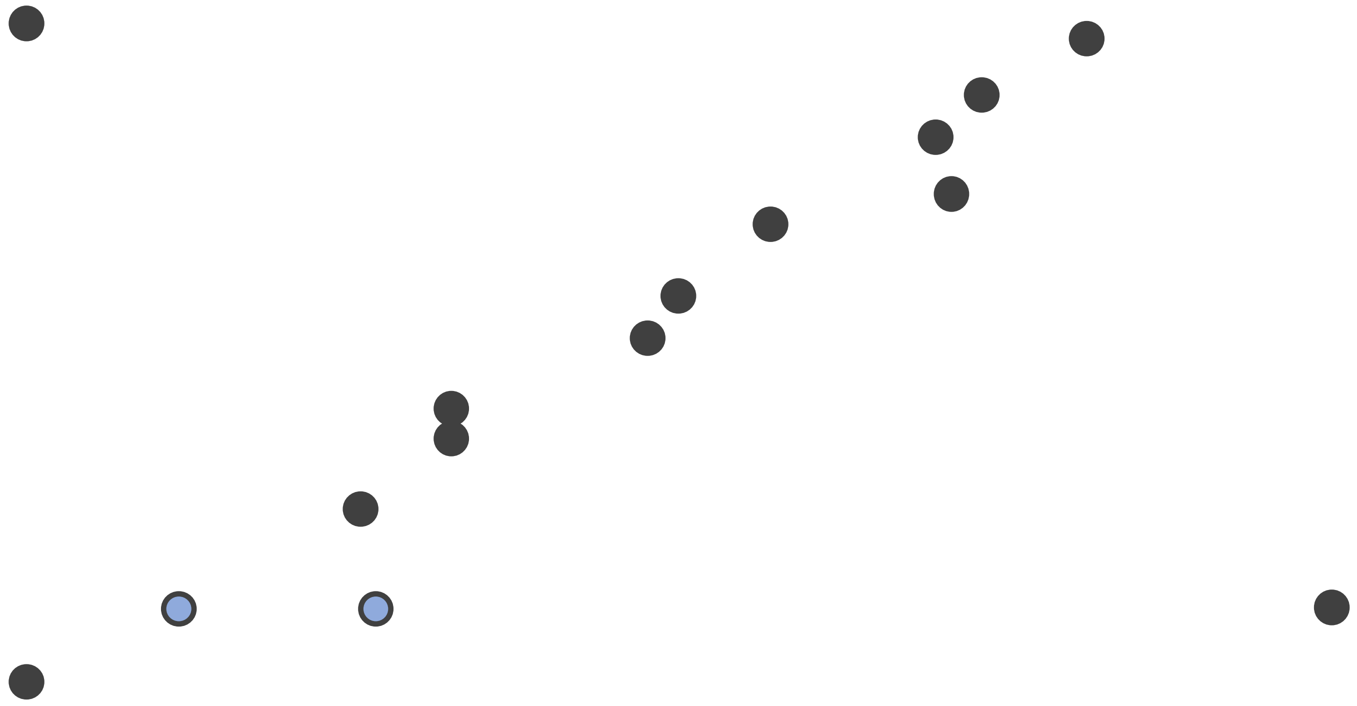
② **Points within some distance threshold**, t, of model are a consensus set.
Size of consensus set is model's support

— **Inliers:** 9 points



RANSAC (RANdom SAmple Consensus)

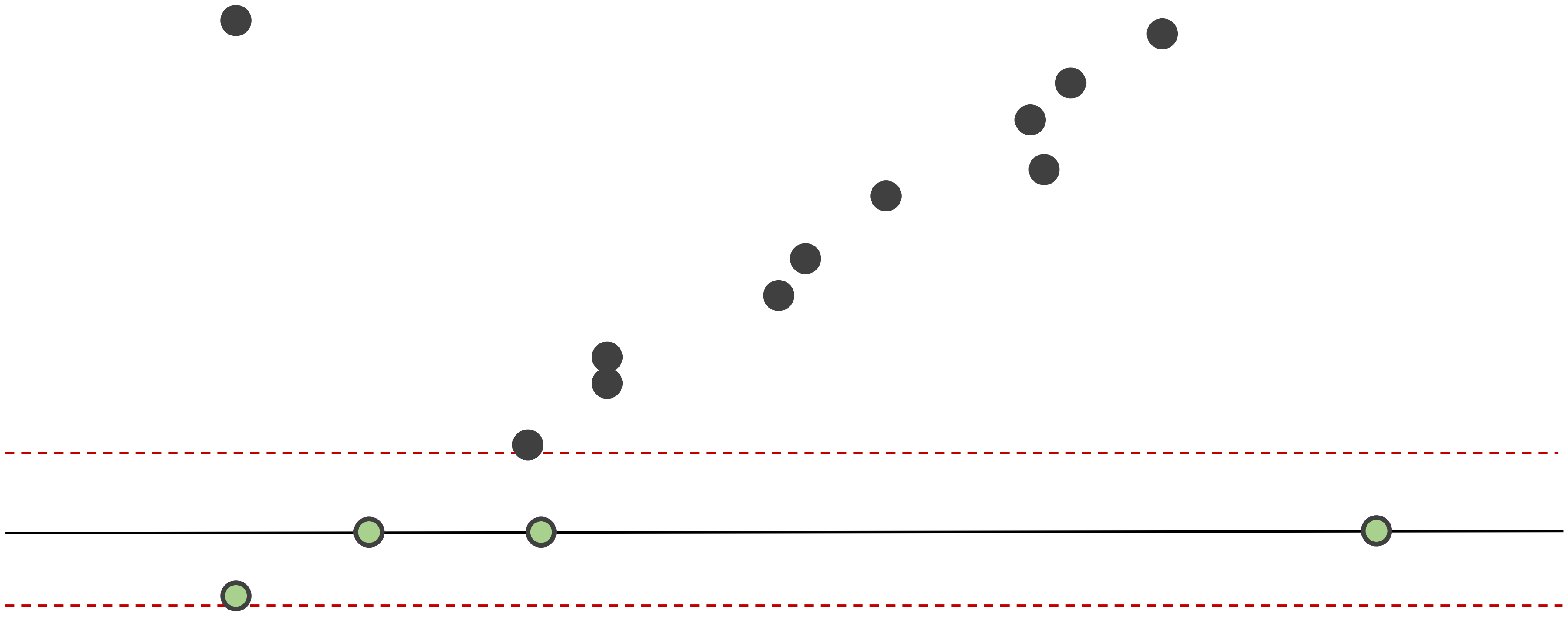
① **Randomly choose minimal subset of data points** necessary to fit model (a sample)



RANSAC (RANdom SAmple Consensus)

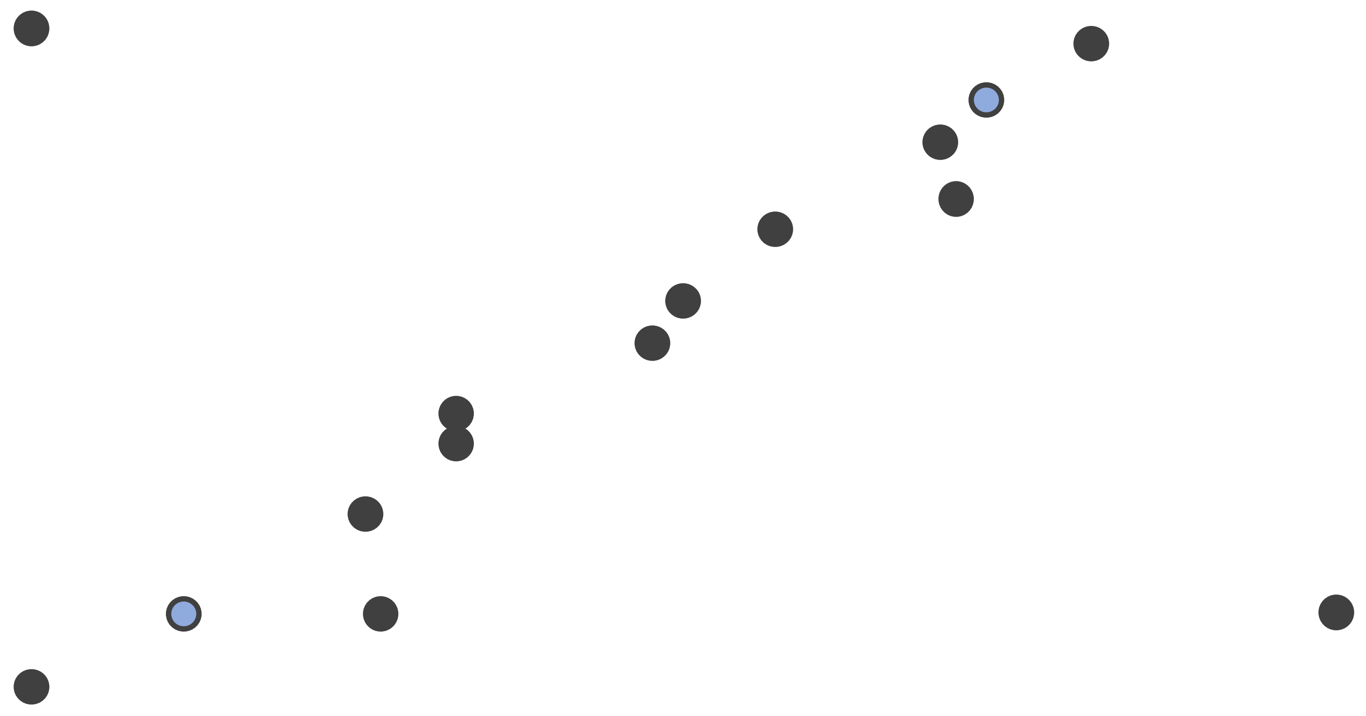
② **Points within some distance threshold**, t, of model are a consensus set.
Size of consensus set is model's support

— **Inliers:** 4 points



RANSAC (RANdom SAmple Consensus)

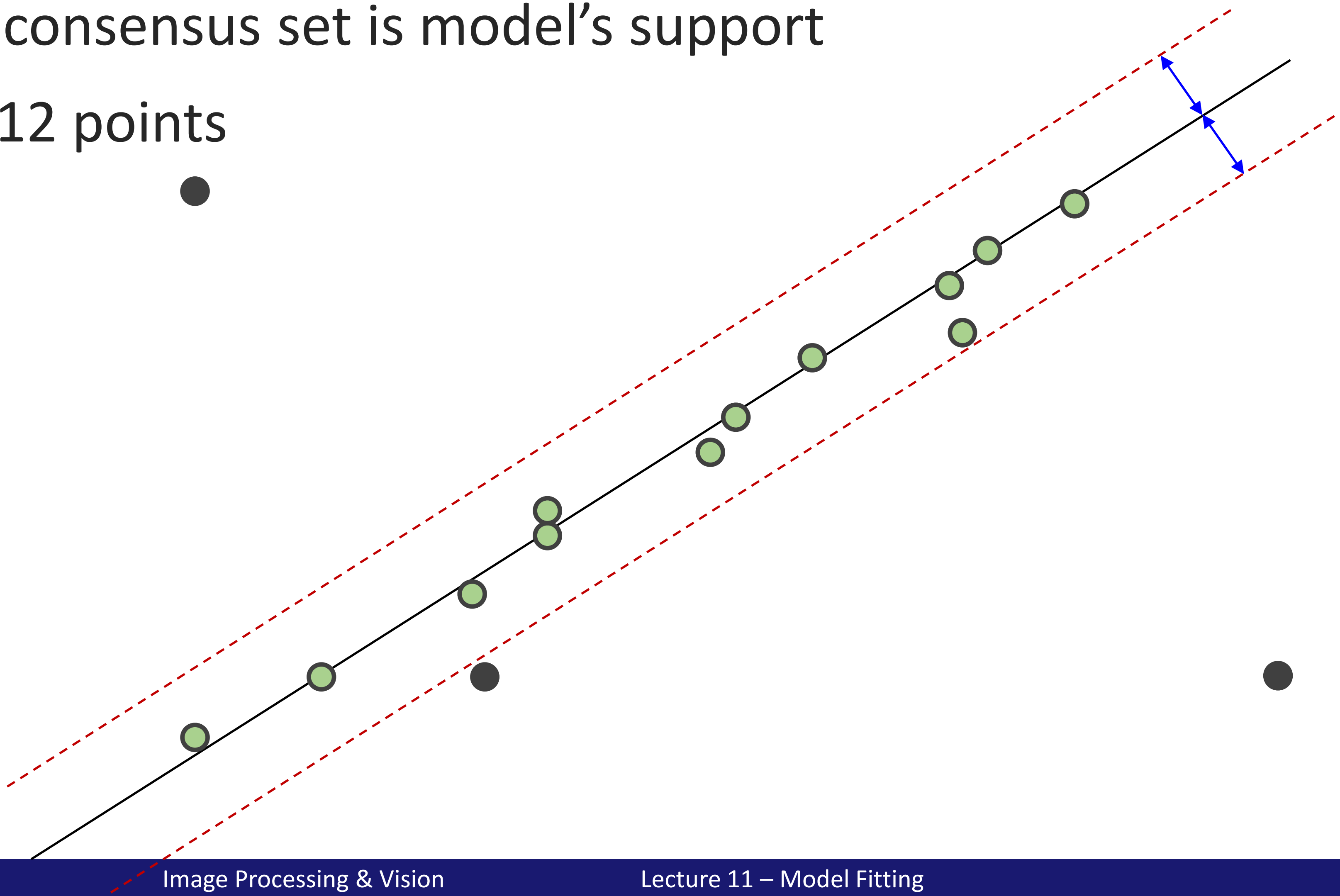
① **Randomly choose minimal subset of data points** necessary to fit model (a sample)



RANSAC (RANdom SAmple Consensus)

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

— **Inliers:** 12 points

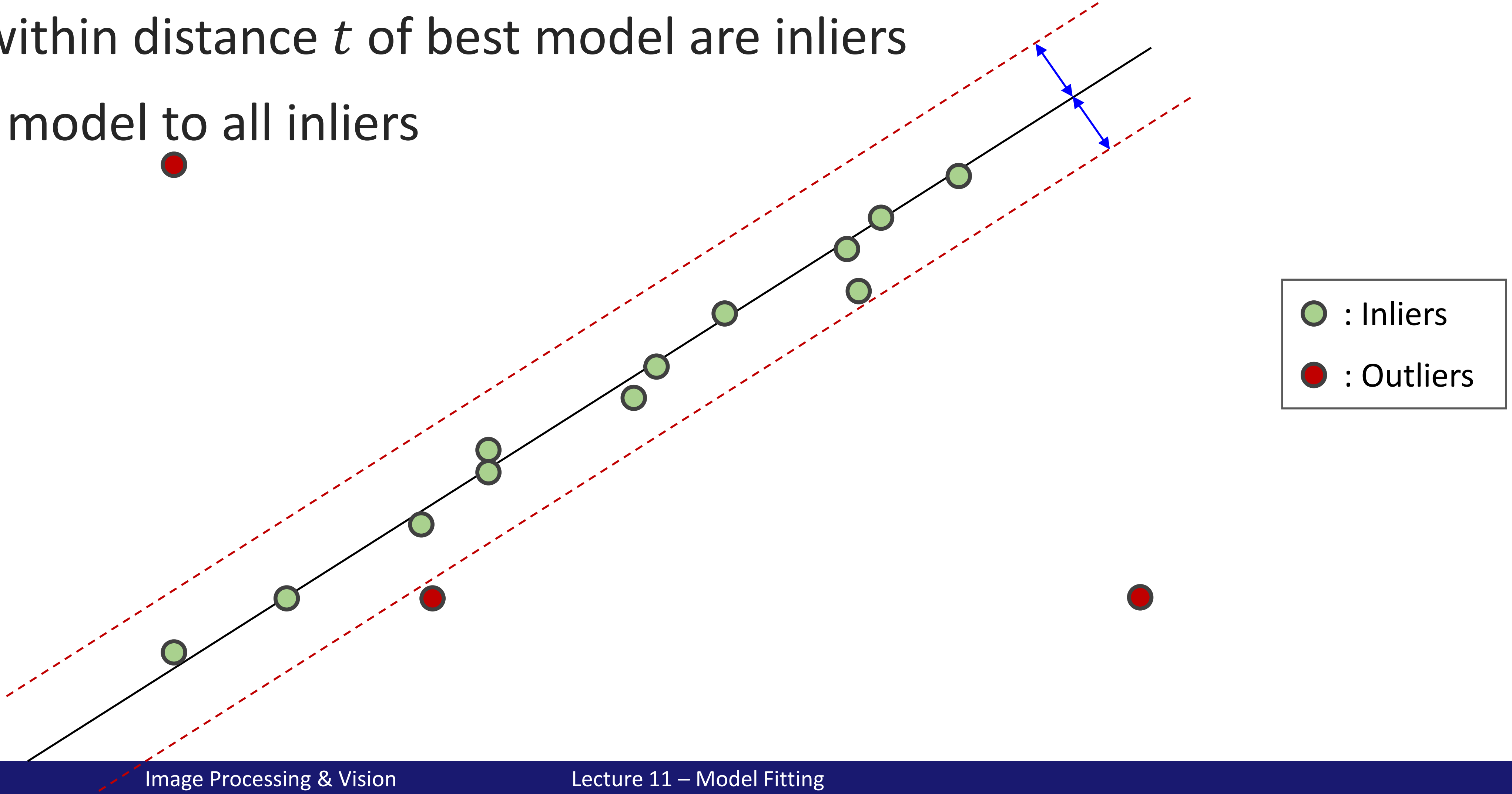


RANSAC (RANdom SAmple Consensus)

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



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 ω^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: $\boldsymbol{k}$ 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

Hartley and Zisserman, Multiple View Geometry in Computer Vision, 2000

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

Application of RANSAC: **Automatic Matching**

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

Left image

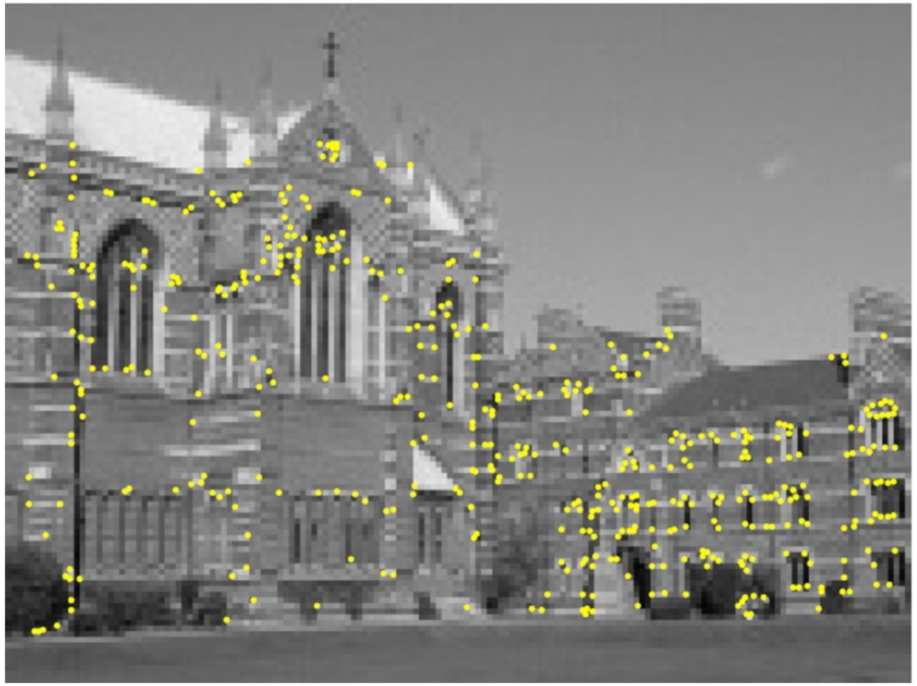
Right image

Application of RANSAC: **Automatic Matching**

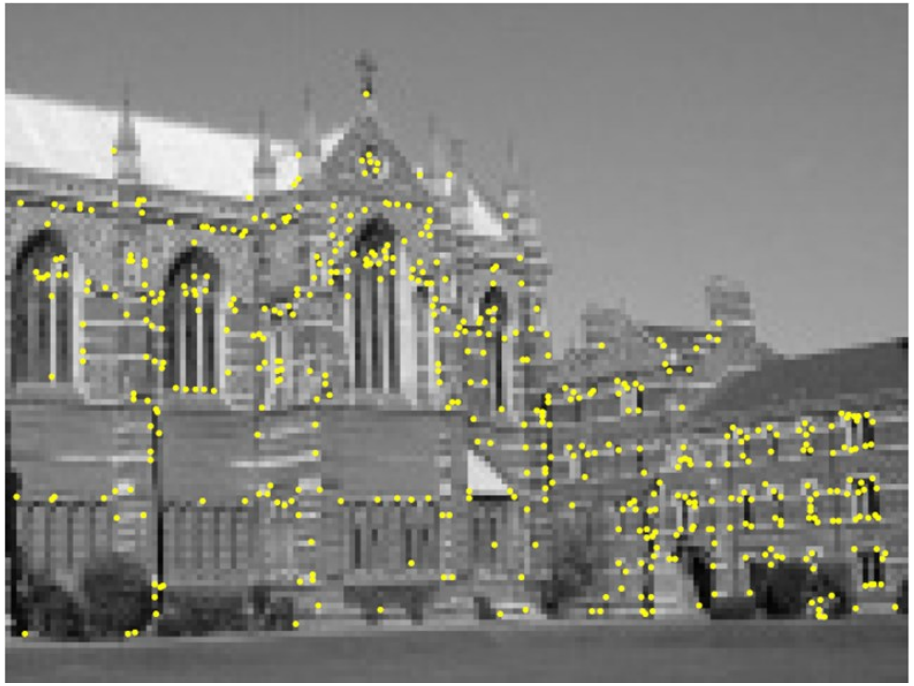
① Feature Extraction

— Find features in pair of images using Harris corner detector

— Assumes images are roughly the same scale



Left image



Right image

Application of RANSAC: **Automatic Matching**

② Finding Feature Matches – Initial Match Hypothesis

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

268 matched features on left image

Application of RANSAC: **Automatic Matching**

③ Outliers & Inliers after RANSAC

— n is 4 for this problem (a homography relating 2 images)

— 43 samples used with t = 1.25 pixels (Assume up to 45% outliers)

151 **inliers**

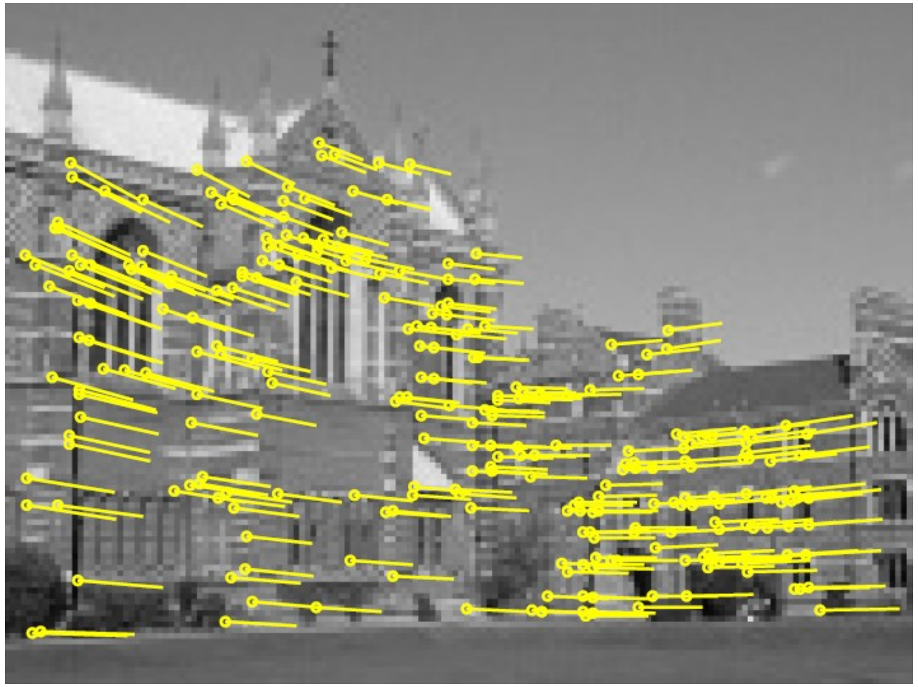
117 **outliers**

Application of RANSAC: **Automatic Matching**

④ Final Matches

— Maximum Likelihood Estimation (MLE)

— Guided Matching



Final set of 262 matches

Summary: 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)