
Computer vision

Computer Vision

Lecture 4: Features III

Dr. Dina Khattab

dina.khattab@cis.asu.edu.eg

Scientific Computing Department

Instructor:	Dr. Dina Khattab
Email:	dina.khattab@cis.asu.edu.eg
Office:	Main Building – 4^{th} floor – Room 302
Office Hours:	Monday 12:00 - 2:00 PM Thursday 11:00 AM to 12:00 PM

Agenda

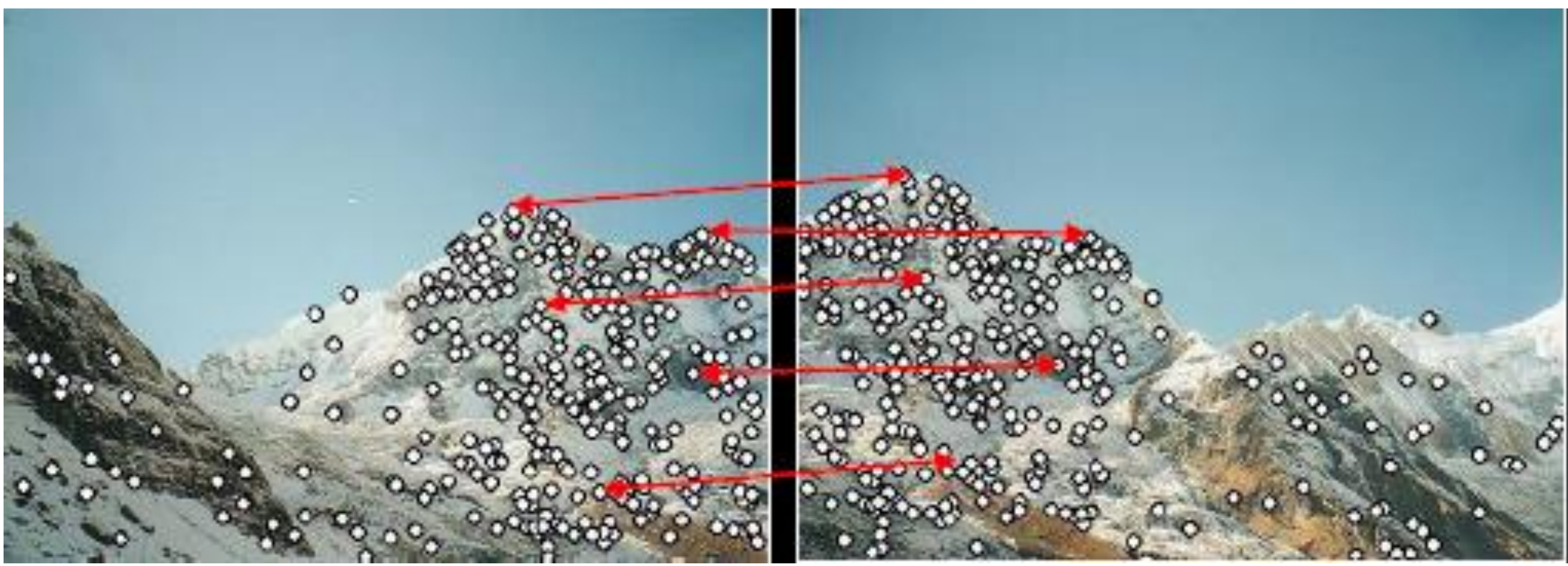
– Feature Alignment (RANSAC)

Building Panorama

- Detect and describe in both images (Harris – SIFT- HOG - .. etc.).

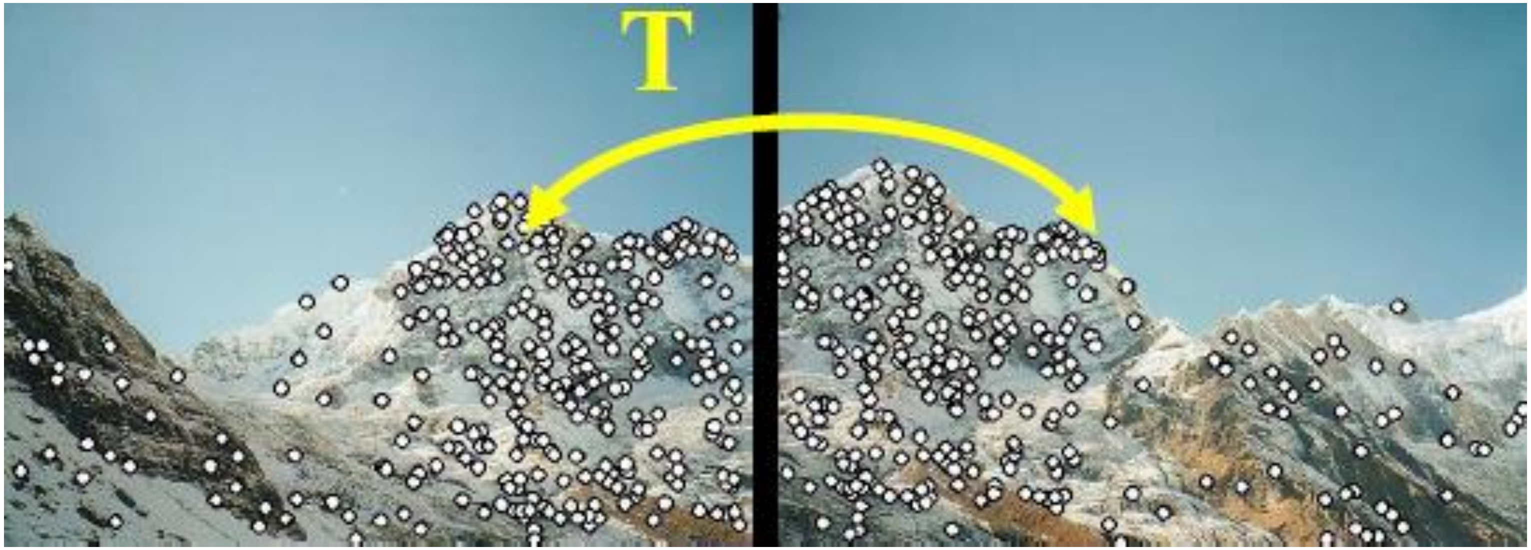

Building Panorama

- Match features - find corresponding pairs



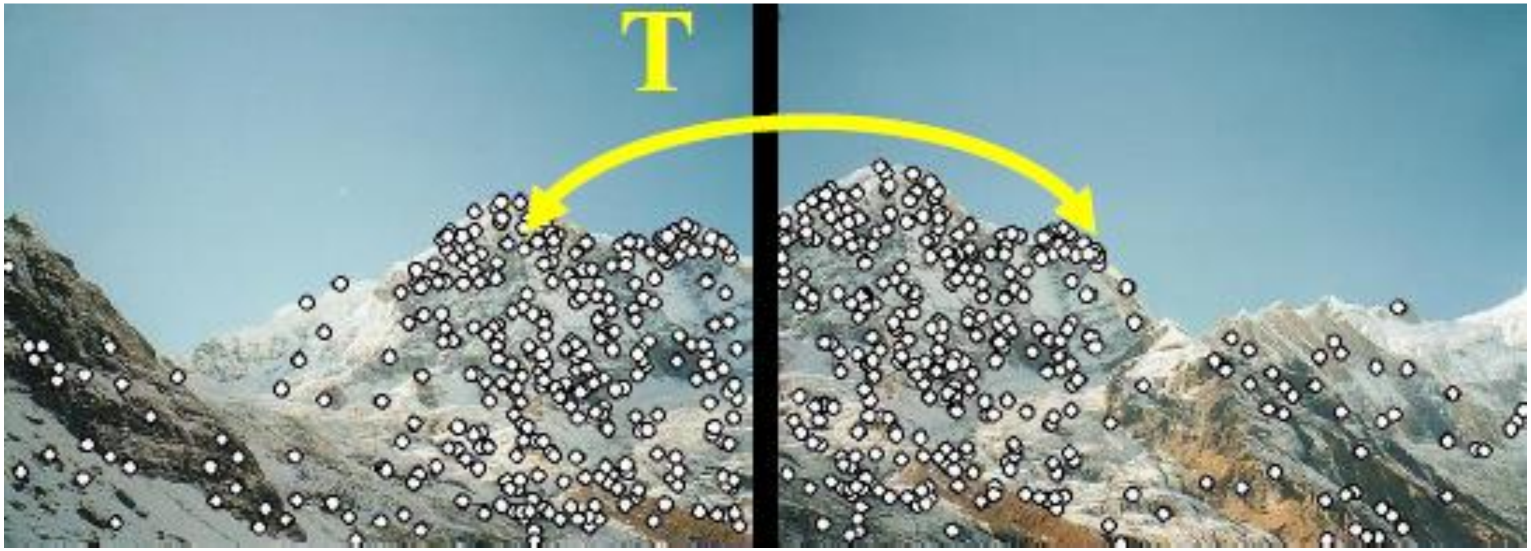
Building Panorama

- Use these pairs to align images (Compute and apply the best transformation: *e.g. affine, translation, or rotation)*



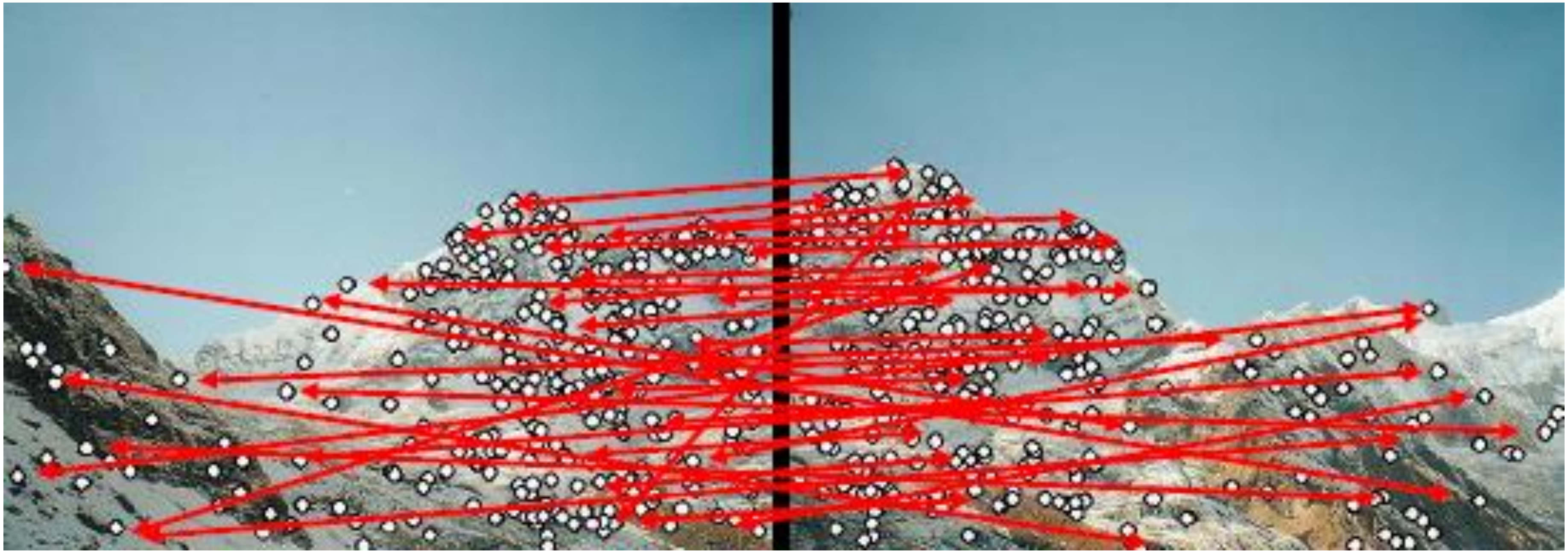
Building Panorama

- Use these pairs to align images (Compute and apply the best transformation: *e.g. affine, translation, or rotation)*



FEATURE-BASED ALIGNMENT

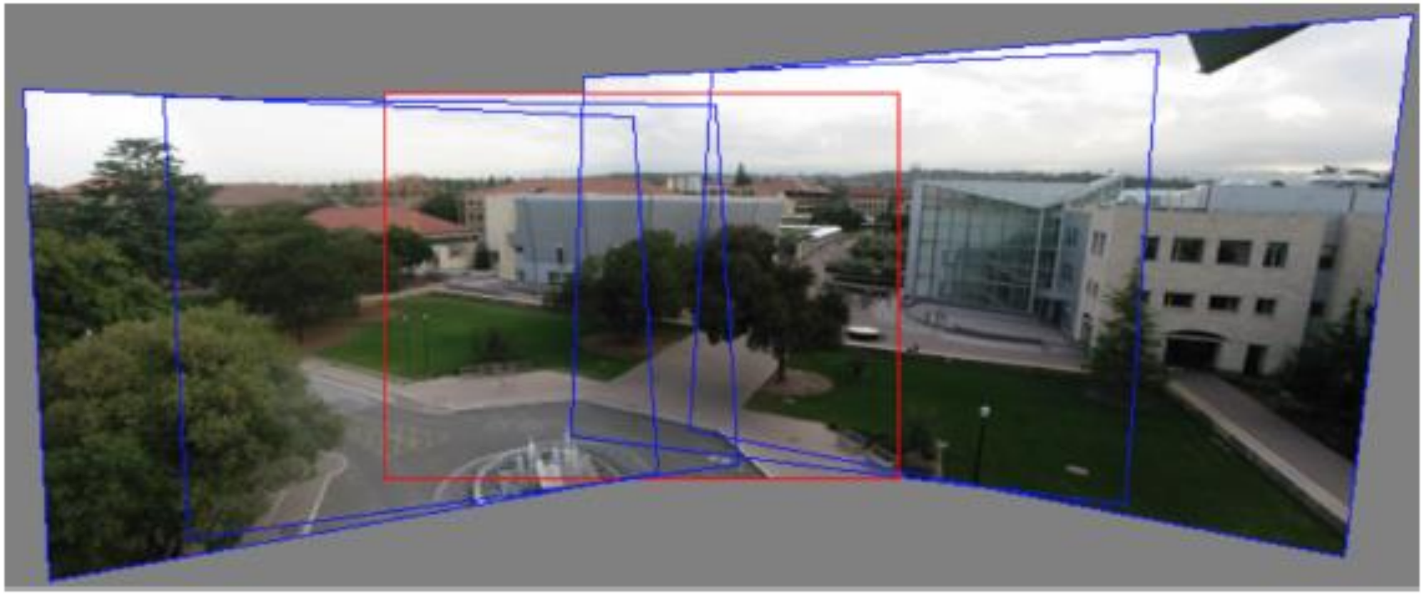
First: Finding good matches



- Still we would get set of incorrect matches (outliers) , so what we do?? [Compute ***closest*** *matches]*

Second: Compute Transformations

- Images from different perspectives need to be transformed to a common perspective before they can be stitched together.



Second: Compute Transformations

- The transformation of an image from one projective plane to another may involve translation, scaling (up or down), rotation, shear, and changes in aspect ratio

translation

rotation

aspect

affine

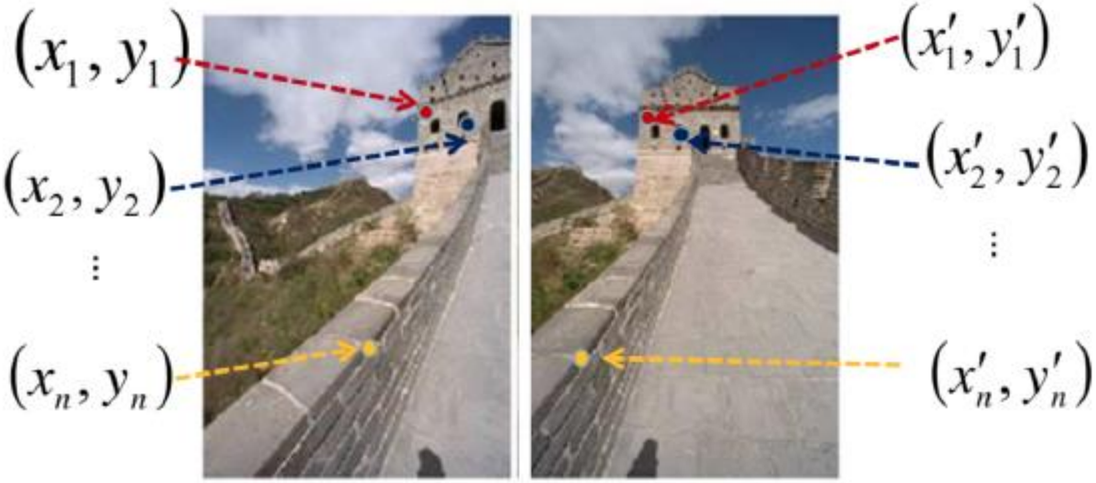
perspective

Second: Compute Transformations

- Stitching together multiple images requires an understanding of projective geometry and homography.
- In addition to 3D Euclidean space, an additional dimension W (distance between camera and an image) is considered.
- The 4D space is called "projective space" and the coordinates in projective space are called "homogeneous coordinates".

Homography

- Homography/transformation matrix H describes the transformation between points in one picture and a related neighboring picture (different perspective).



- Homography matrix H can be estimated given N matched keypoint pairs using least squares.

$$\begin{bmatrix} wx' \\ wy' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homography matrix for basic transformations

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Translate

$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{s}_x & 0 & 0 \\ 0 & \boldsymbol{s}_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ 1 \end{bmatrix}$$

Scale

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotate

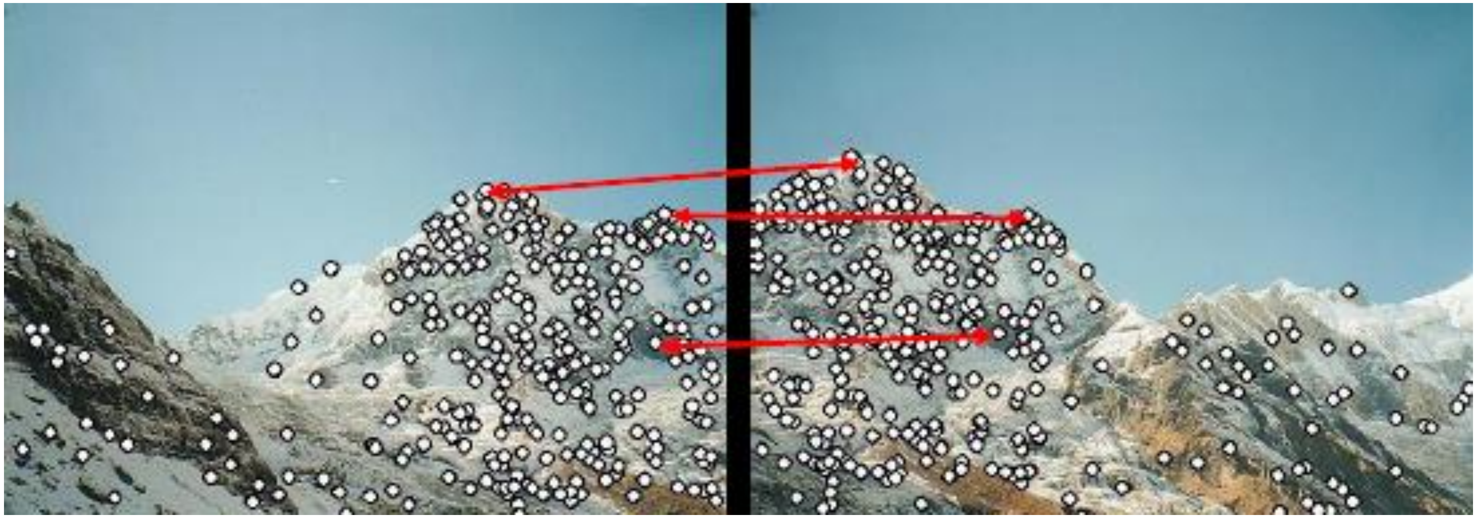
$$\begin{bmatrix} \boldsymbol{x}' \\ \boldsymbol{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \boldsymbol{sh}_x & 0 \\ \boldsymbol{sh}_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ 1 \end{bmatrix}$$

Shear

Finding good matches

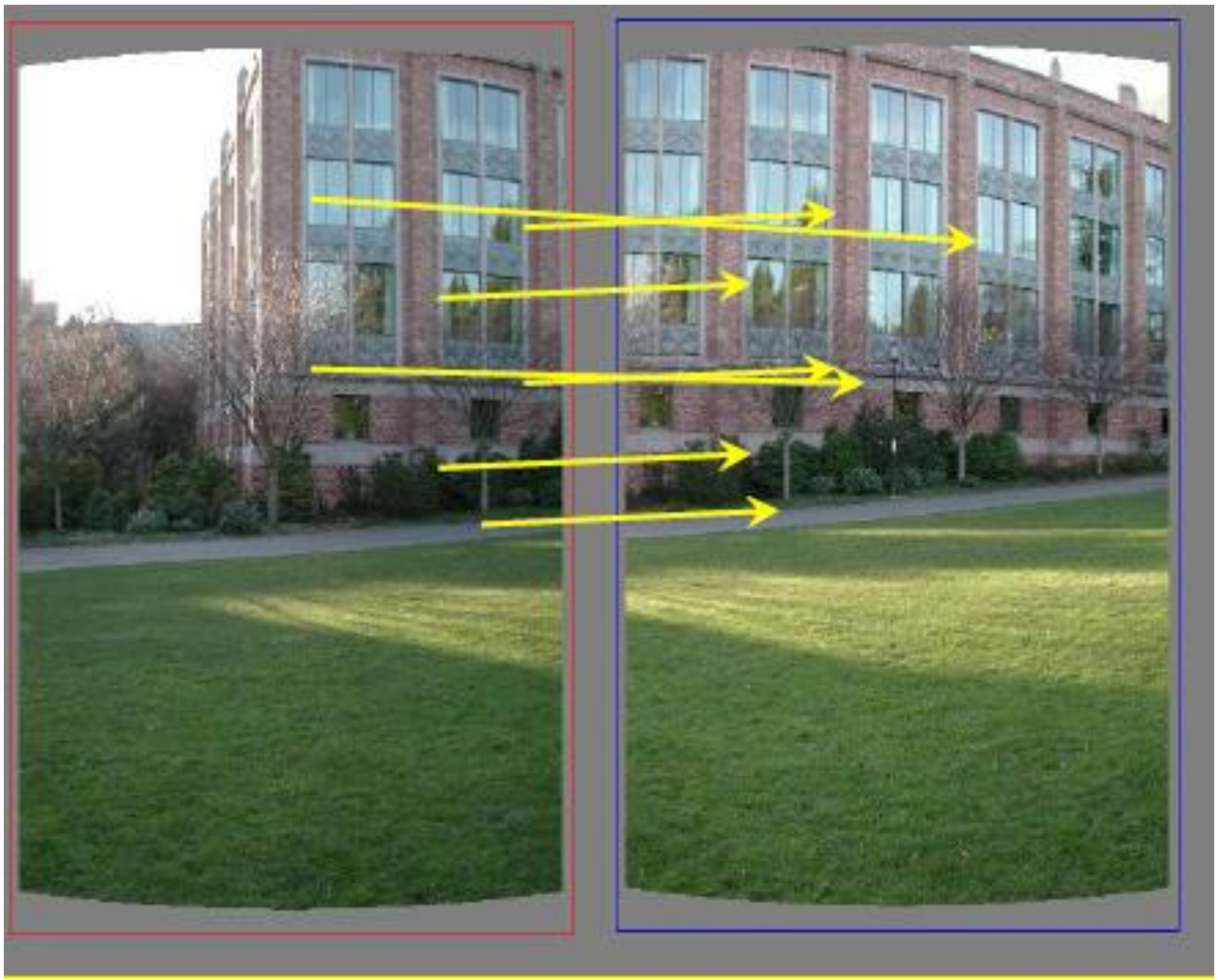
RANDOM SAMPLE CONSENSUS (RANSAC)

Feature-based alignment to find transforms

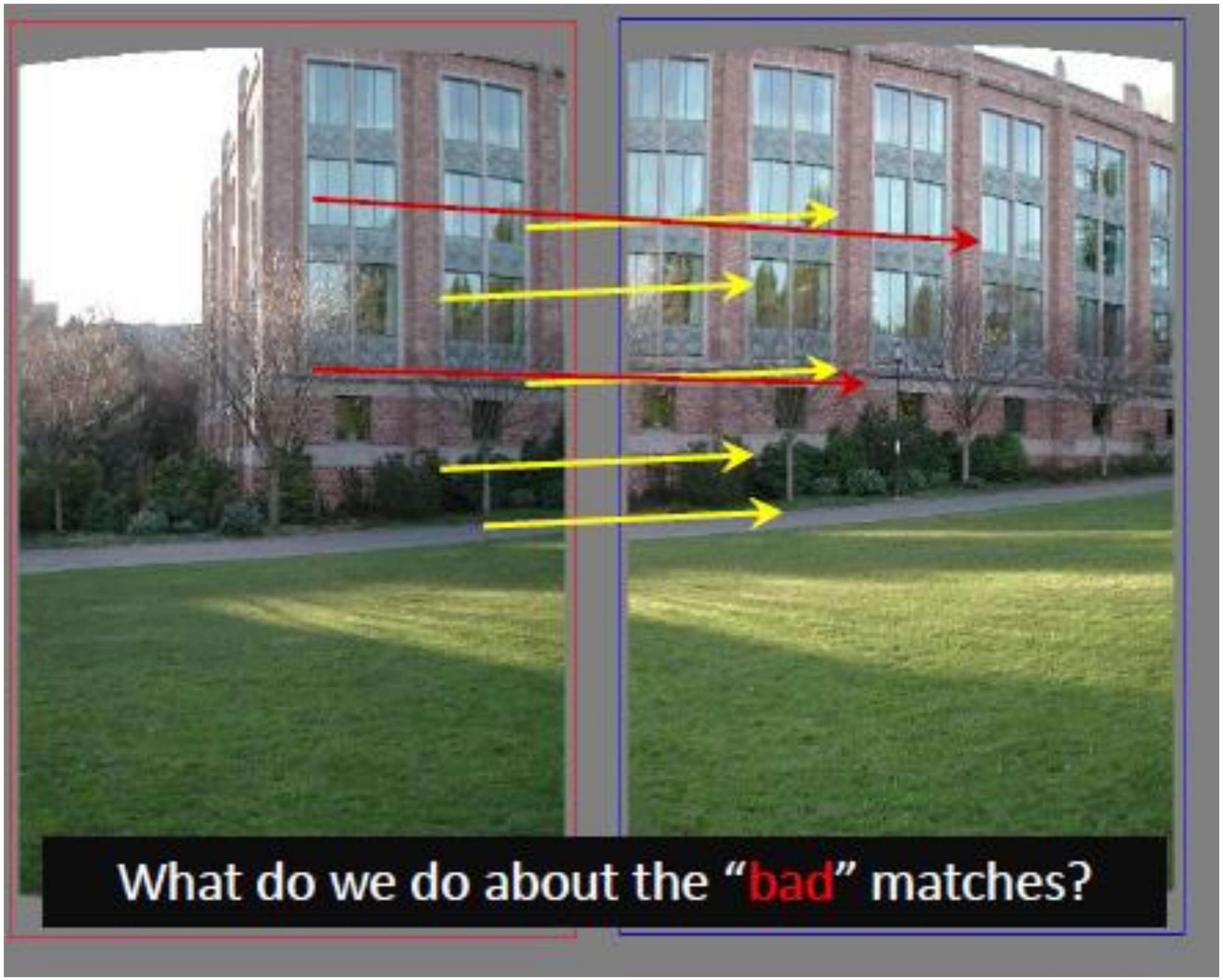


- Loop until happy:
  - *Hypothesize* transformation T from some matches.
  - *Verify* transformation (search for other matches consistent with T) – mark best

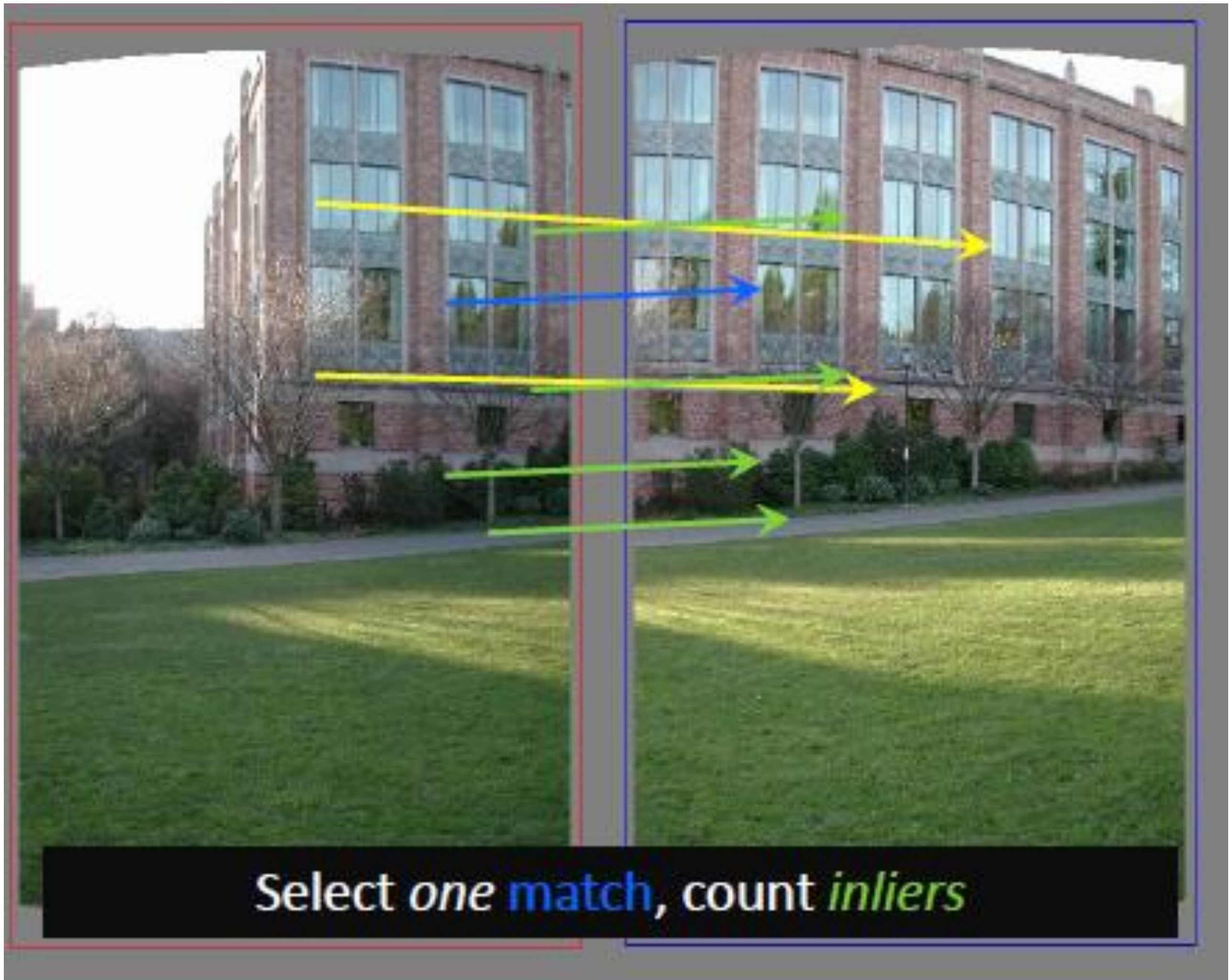
Matching features



Matching features

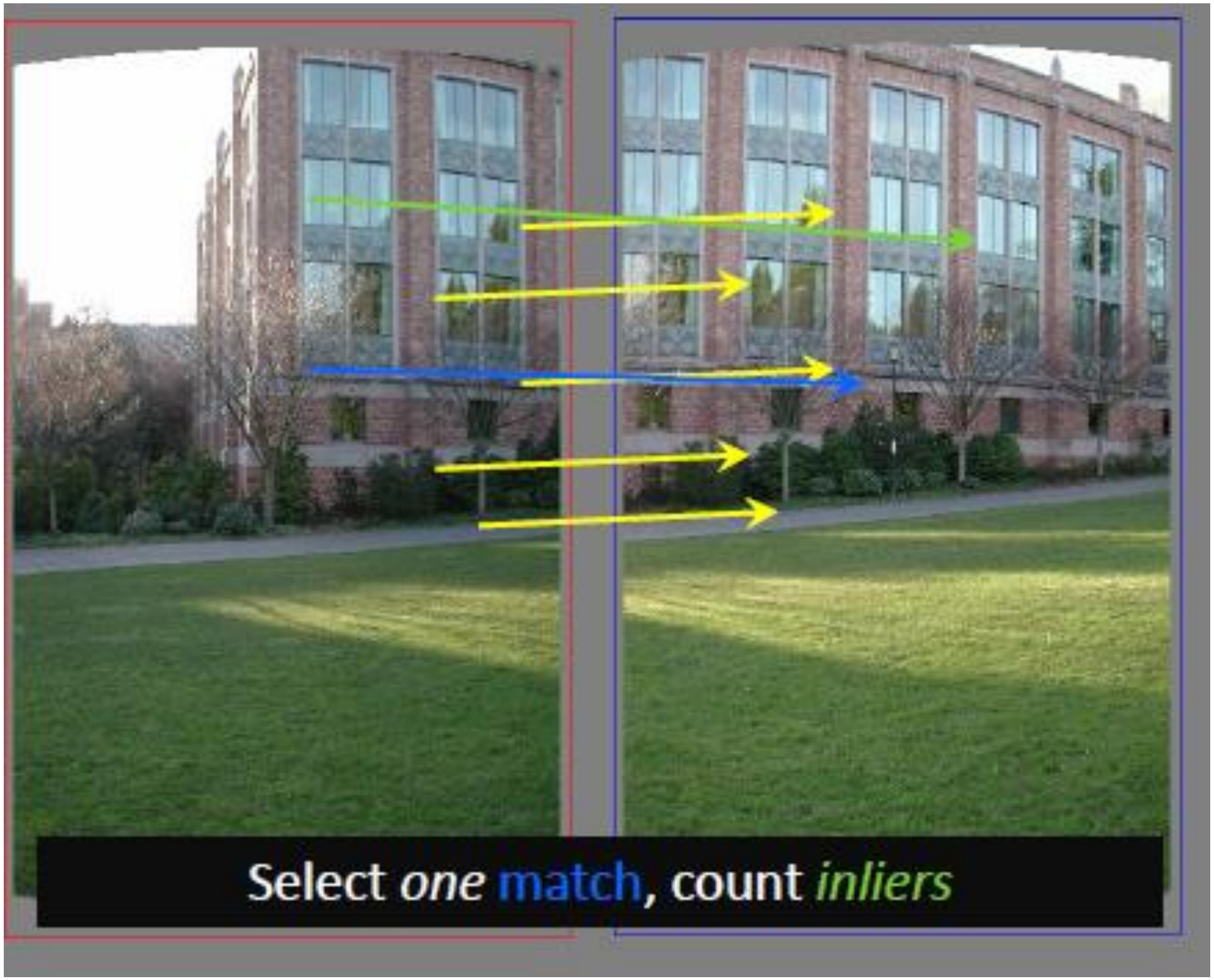


RAndom SAmple Consensus (1)
Select one



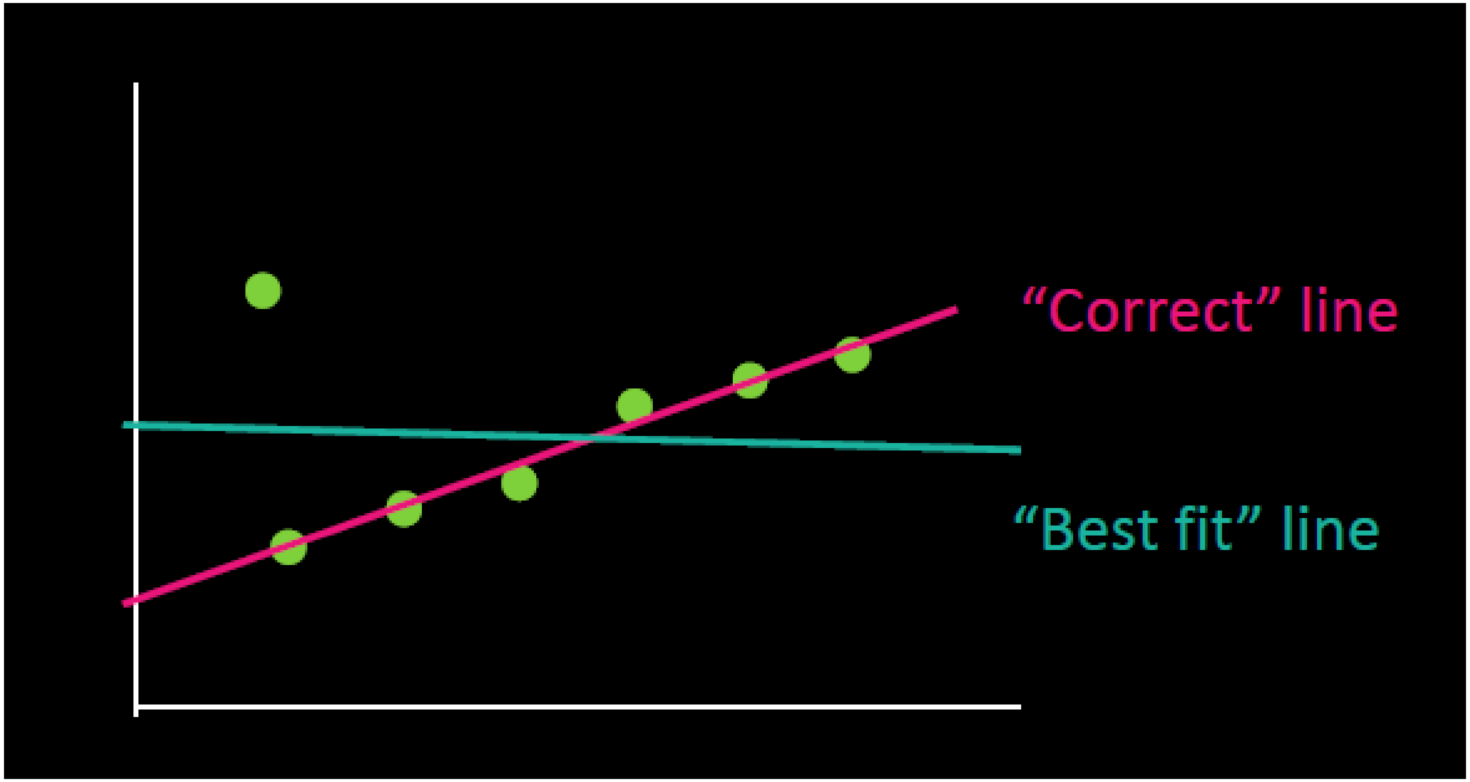
5 total inliers (including selected)

RAndom SAmple Consensus (2)
Select one



2 total inliers

Simple Example: Fitting a line



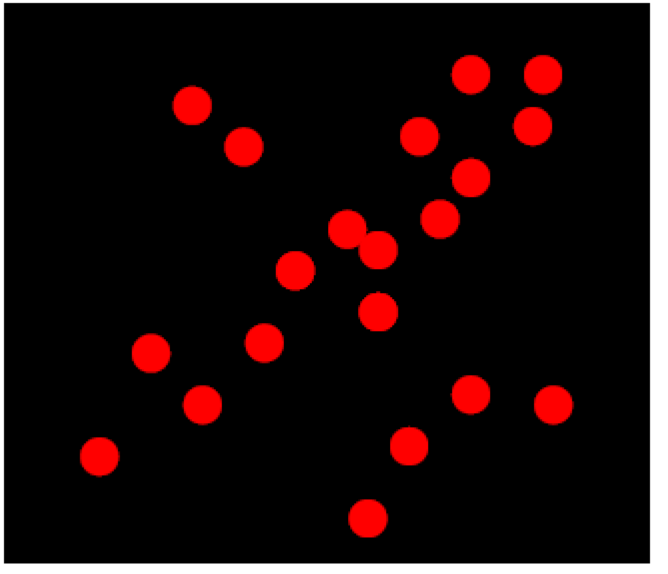
So question is how to find those outliers? Points don’t fit to our model !

RANSAC: Main idea

- RANdom SAmple Consensus:
  - Randomly pick some points to define your line (model).
  - Repeat enough times until you find a good line (model) – on with many inliers.

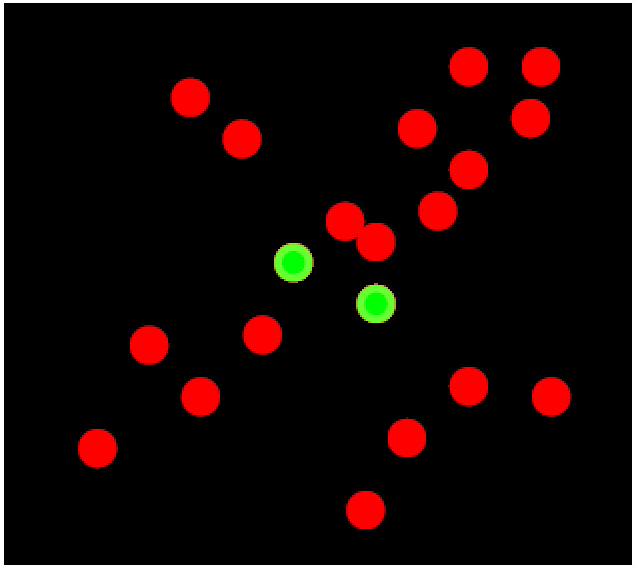
RANSAC

1. **Sample** (randomly) the number of points required to fit the model.
2. **Solve** for model parameters using sample.
3. **Score** by the fraction of ***inliers*** within a preset threshold of the model.
4. **Repeat** 1-3 until the best model is found with high confidence.



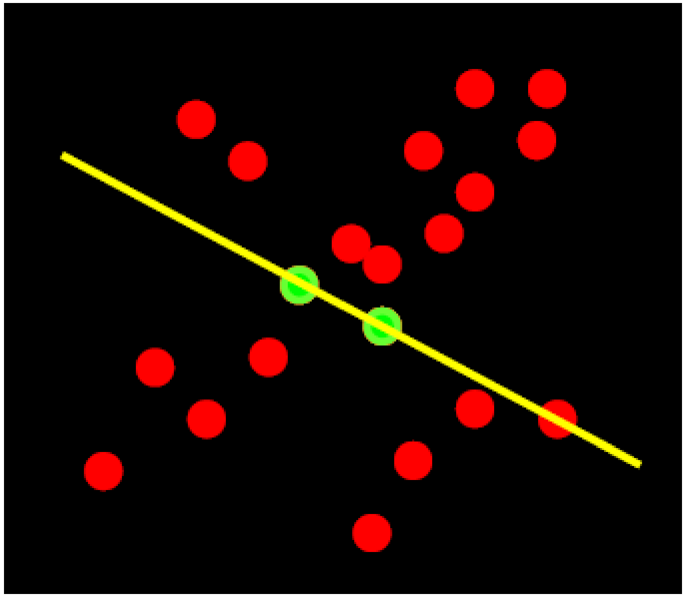
RANSAC

1. **Sample** (randomly) the number of points required to fit the model.
2. **Solve** for model parameters using sample.
3. **Score** by the fraction of ***inliers*** within a preset threshold of the model.
4. **Repeat** 1-3 until the best model is found with high confidence.



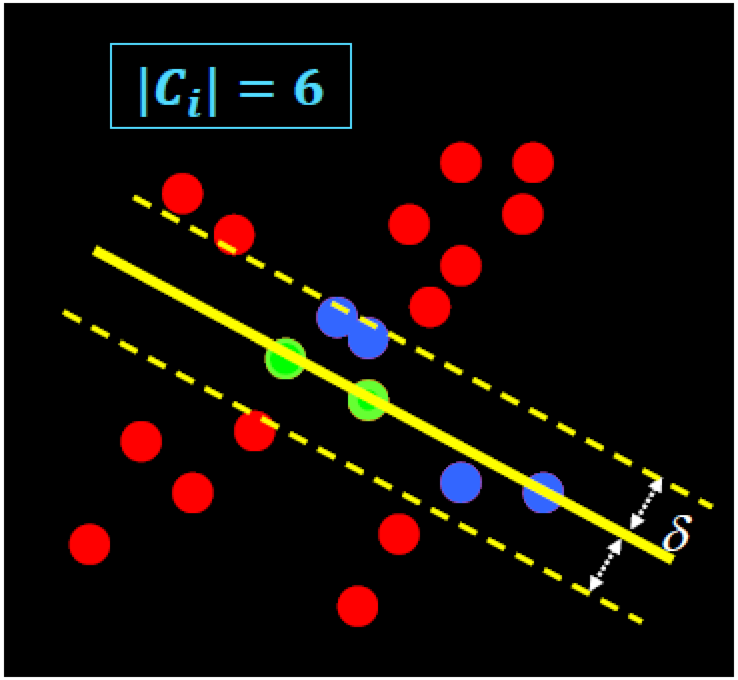
RANSAC

1. **Sample** (randomly) the number of points required to fit the model.
2. **Solve** for model parameters using sample.
3. **Score** by the fraction of ***inliers*** within a preset threshold of the model.
4. **Repeat** 1-3 until the best model is found with high confidence.



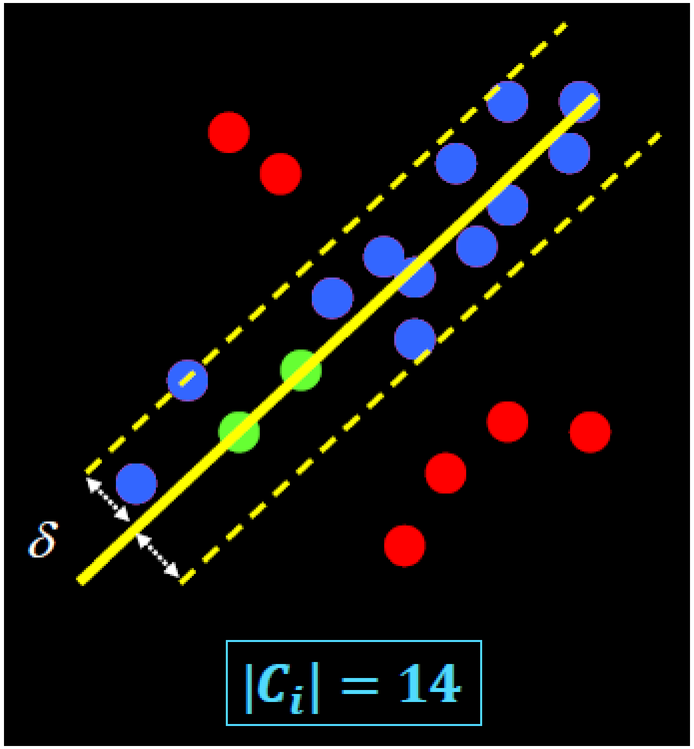
RANSAC

1. **Sample** (randomly) the number of points required to fit the model.
2. **Solve** for model parameters using sample.
3. **Score** by the fraction of ***inliers*** within a preset threshold of the model.
4. **Repeat** 1-3 until the best model is found with high confidence.



RANSAC

1. **Sample** (randomly) the number of points required to fit the model.
2. **Solve** for model parameters using sample.
3. **Score** by the fraction of ***inliers*** within a preset threshold of the model.
4. **Repeat** 1-3 until the best model is found with high confidence.



RANSAC [Fischler & Bolles 1981]

1. Randomly select s points (or point pairs) to form a *sample.*
2. Instantiate the model.
3. Get consensus set C_i: The points within error bounds (distance threshold) of the model
4. If $|C_i|>T$ (inliers), terminate and return model
5. Repeat for N trials, return model with max $|C_i|$

After RANSAC

- re-compute the model using a least-squares estimate using all of the inliers that were within the threshold distance.

- Keep this transformation as the model that best approximates the data.

Choosing the parameters

- Initial number of points s
  - Typically minimum number needed to fit the model
- Distance threshold T
  - Choose T so probability for inlier is high (e.g. 0.95)
- Number of iterations N
  - Choose N so that, with probability p, at least one random sample set is free from outliers (e.g. $p = 0.99$)

$N > \log(1-p) / \log(1-(1-e)^s)$

- Set p=0.99 – chance of getting good sample

$s = 2, e = 5\%$ => N=2

$s = 2, e = 50\%$ => N=17

$s = 4, e = 5\%$ => N=3

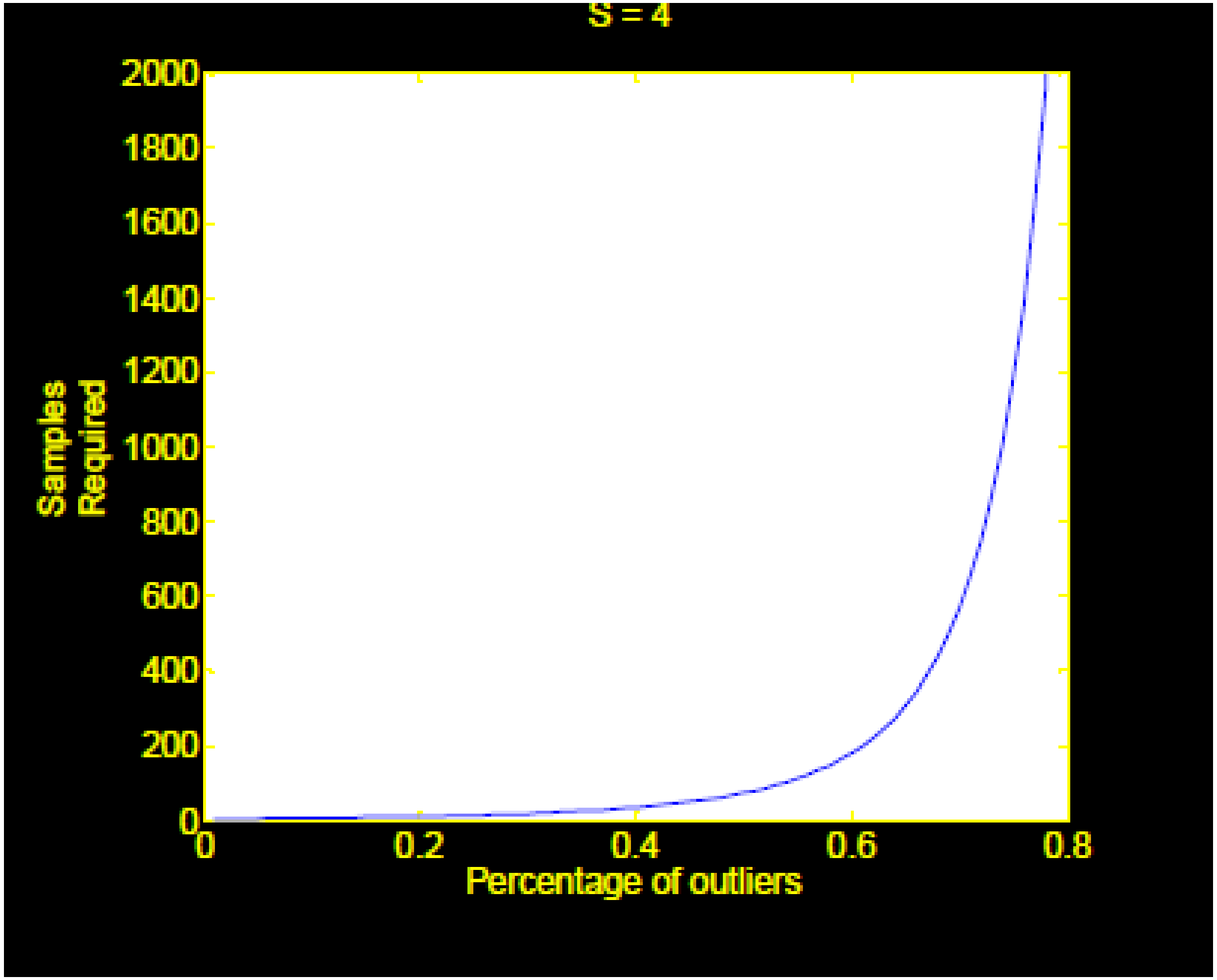
$s = 4, e = 50\%$ => N=72

$s = 8, e = 5\%$ => N=5

$s = 8, e = 50\%$ => N=1177

	proportion of outliers *e*						
s	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

- N increases steeply with s

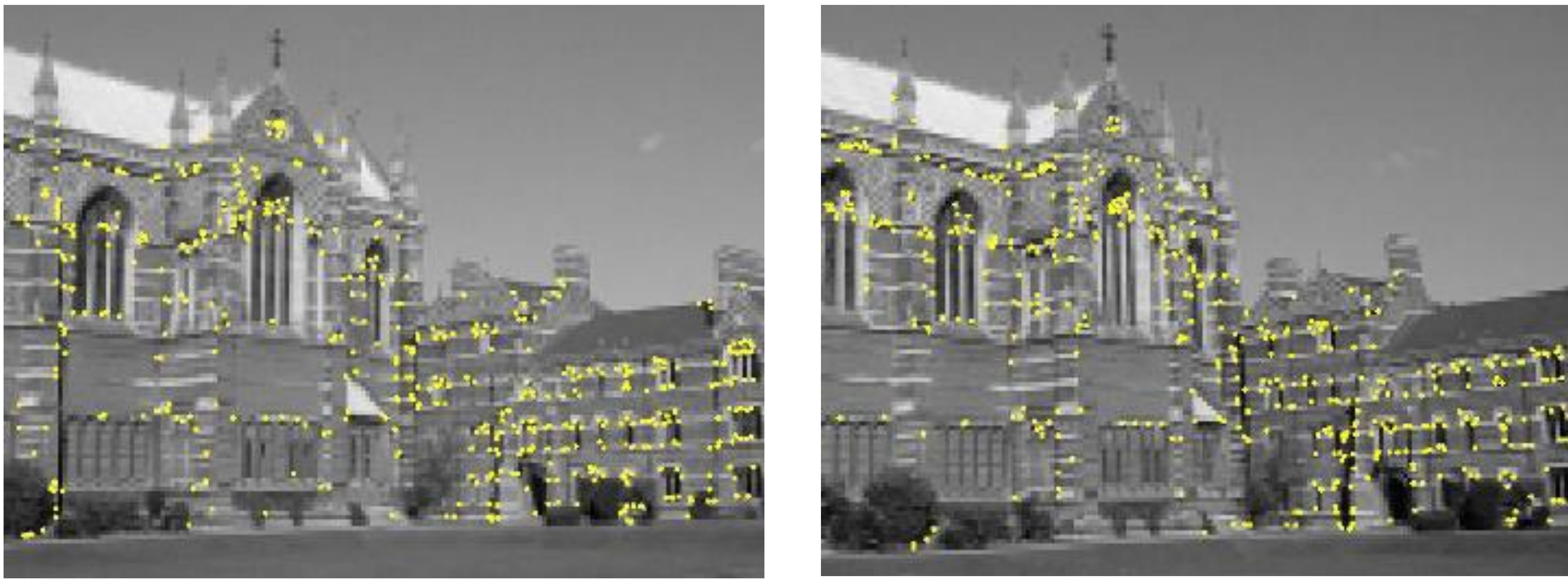


S = 4
Samples Required
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
0.2
0.4
0.6
0.8
Percentage of outliers

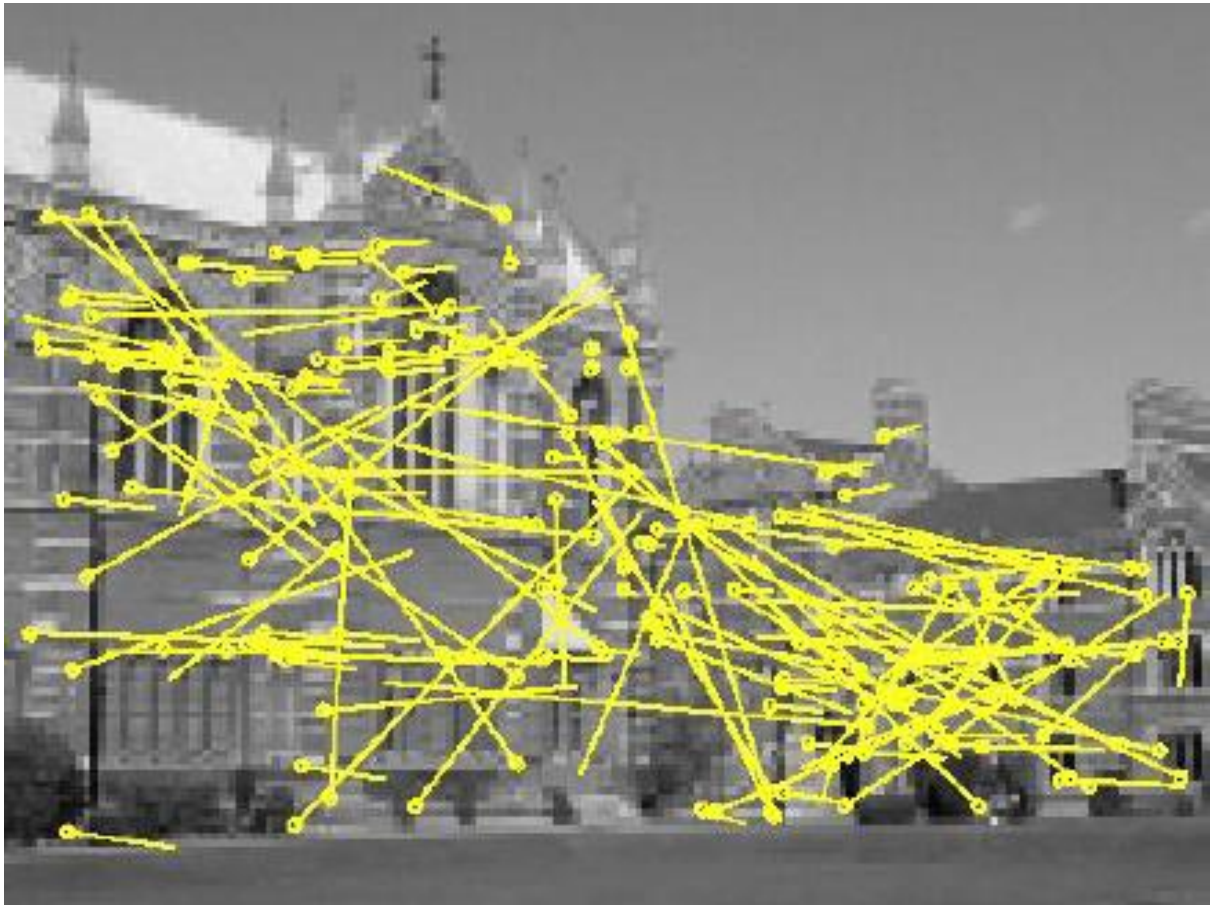
Adaptive Procedure

- $N=\infty$, count =0, $e = 1.0$
- While N > count
  - Choose a sample and count the number of inliers
  - Set $e_0 = 1 - \dfrac{number\ of\ inliers}{total\ number\ of\ points}$
  - If $e_0 < e$ Set $e = e_0$ and recompute N from e:

    $$N > \log(1-p) / \log(1-(1-e)^s)$$
  - Increment the count by 1

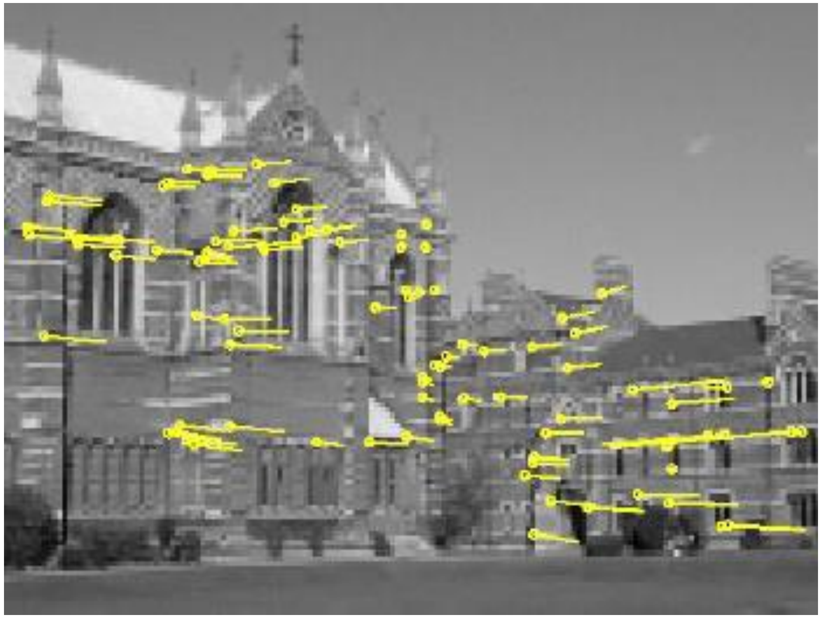


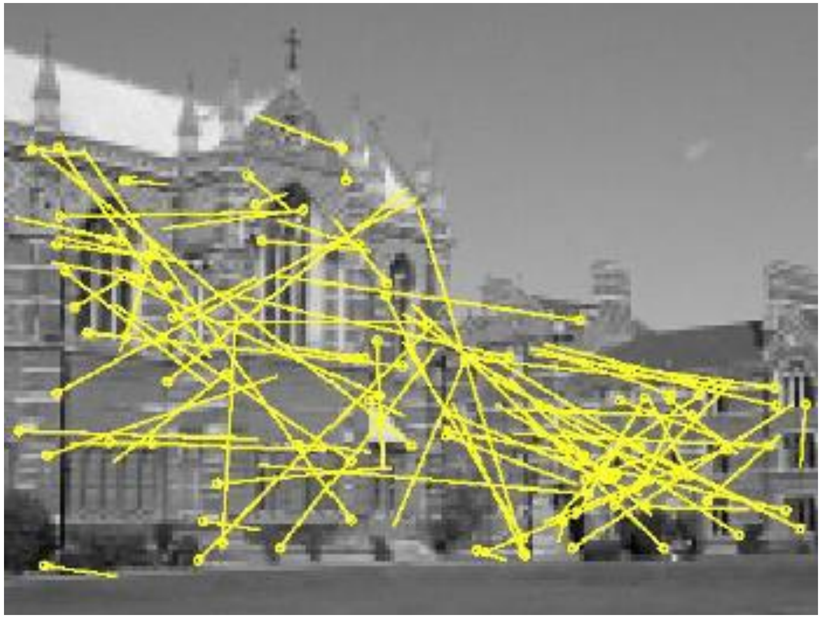
- 188 matches



inliers (99)

outliers(89)





RANSAC: Conclusions

The good…

- Simple and general
- Applicable to many different problems, often works well in practice for model fitting in images (e.g., line/edge detection)
- Robust to large numbers of outliers
- Parameters are easy to choose

RANSAC: Conclusions

The not-so-good…

- Computational time grows quickly with the number of model parameters
- Not as good for getting multiple fits

RANSAC: Conclusions

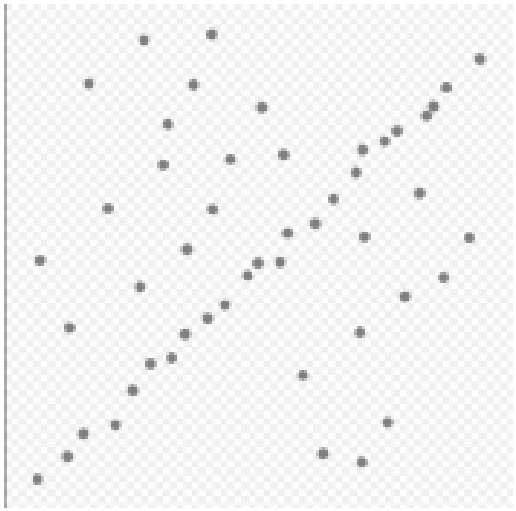
Common applications

- Computing a homography (e.g., image stitching) or other image transform.
- Estimating fundamental matrix (relating two views).
- *Pretty much every problem in robot vision.*

PRACTICE

Questions

- Given the following set of 2-D point cloud, assume that there are 60% inliers; compute the minimal number of RANSAC iterations needed to get, with probability 95%, at least one random sample that is free from outliers.



solution

$N > \log(1-p) / \log(1-(1-e)^s)$

$P = 0.95$ $e = 0.4$, $s= 2$

$N > \log(1-0.95)/ \log(1-(1-0.4)^2)$

$N> \log(0.05)/\log(1-0.36)$

$N> \log(0.05)/\log(0.64)$

$N> 6.712$

$N=7$

Credit for

CS 4495 Computer Vision (Spring 2015)

A. Bob - College of Computing, Georgia Tech.

CS131 "Computer Vision: Foundations and Applications" by University of Stanford (Fall 2019)

CAP5415 " Computer Vision " University of Central Florida, Center of Research in Computer Vision (UCF CRCV), Fall 2020