CS5760: Computer Vision

RANSAC

http://www.wired.com/gadgetlab/2010/07/camera-software-lets-you-see-into-the-past/

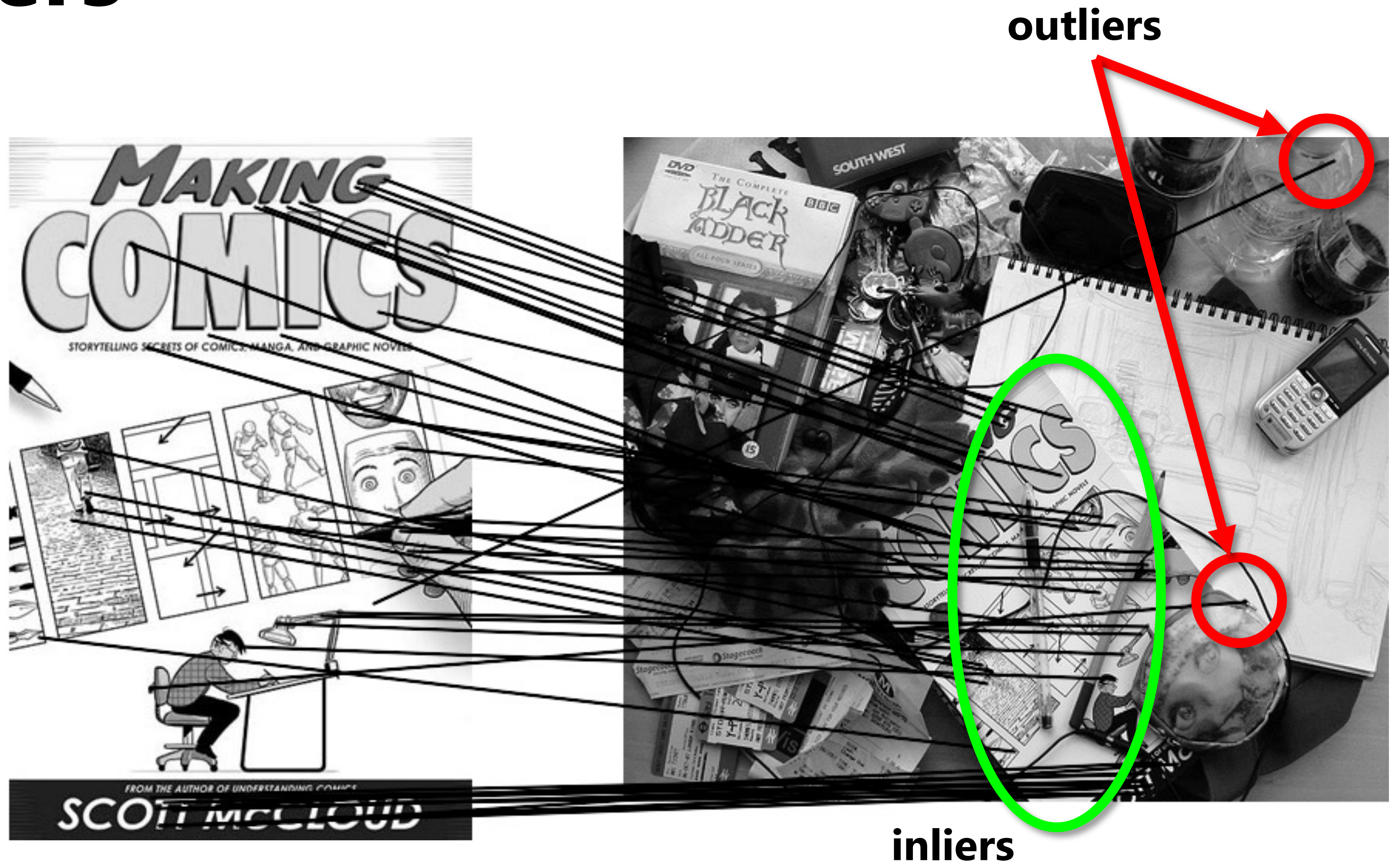
Reading

- Szeliski (2^{nd} edition): Chapter 8.1

Announcements

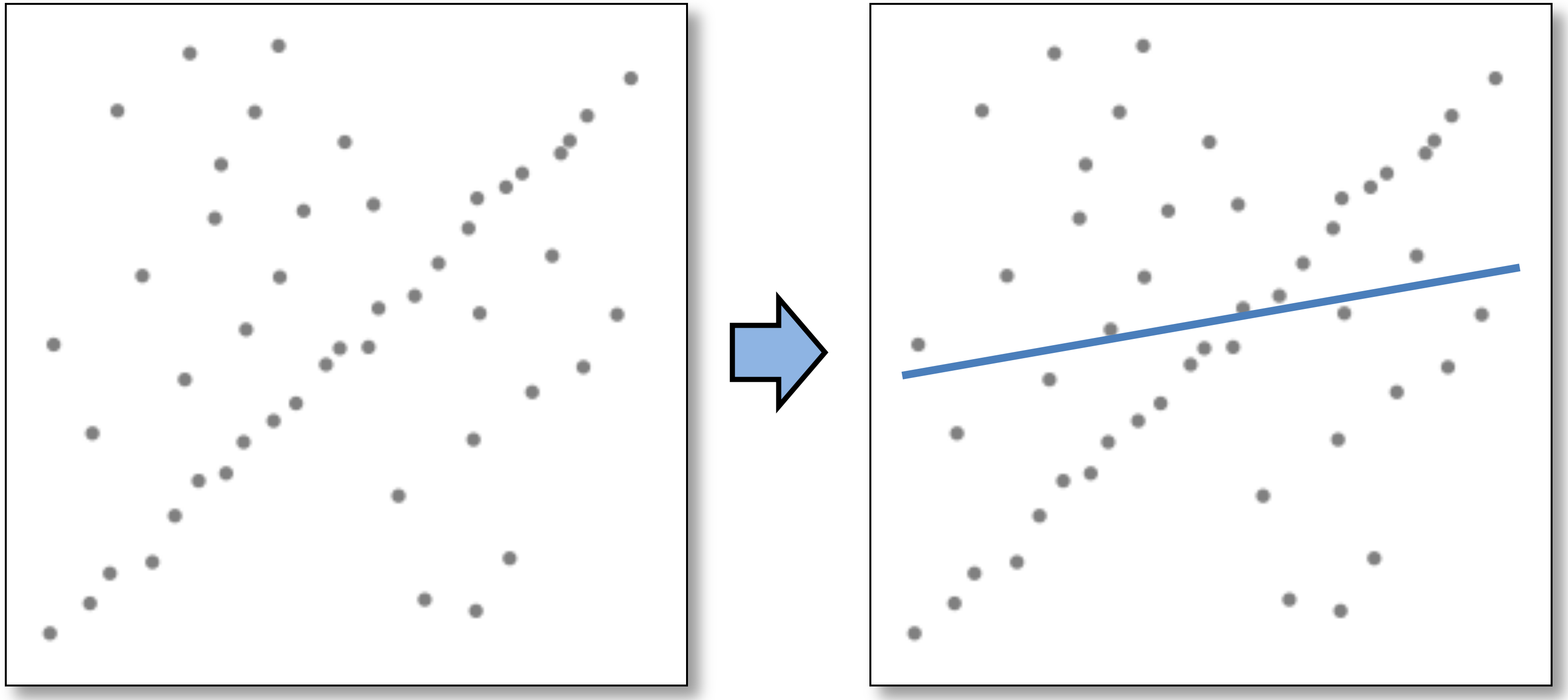
- Project 2 due tomorrow, February 24, by 8pm
  - Report due Thursday, March 2 by 8pm on CMSX
- Take-home midterm to be released after February Break
  - To be distributed in class at 2:15pm Thursday, March 2
  - Due Tuesday, March 7 by 1pm (beginning of class)
  - Open book, open note (but no Google)
  - To be done on your own
- No class on Tuesday (February Break)

Outliers



Robustness

- Let's consider the problem of linear regression



Problem: Fit a line to these datapoints

Least squares fit

- How can we fix this?

We need a better cost function…

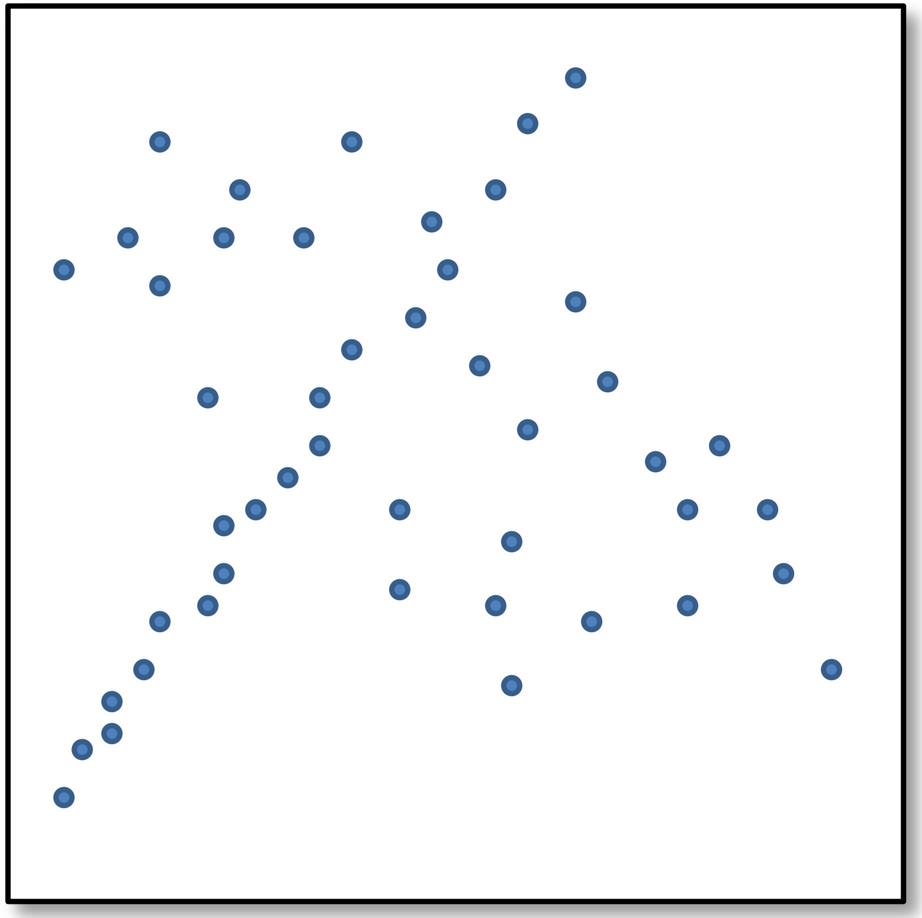
- Suggestions?

Idea

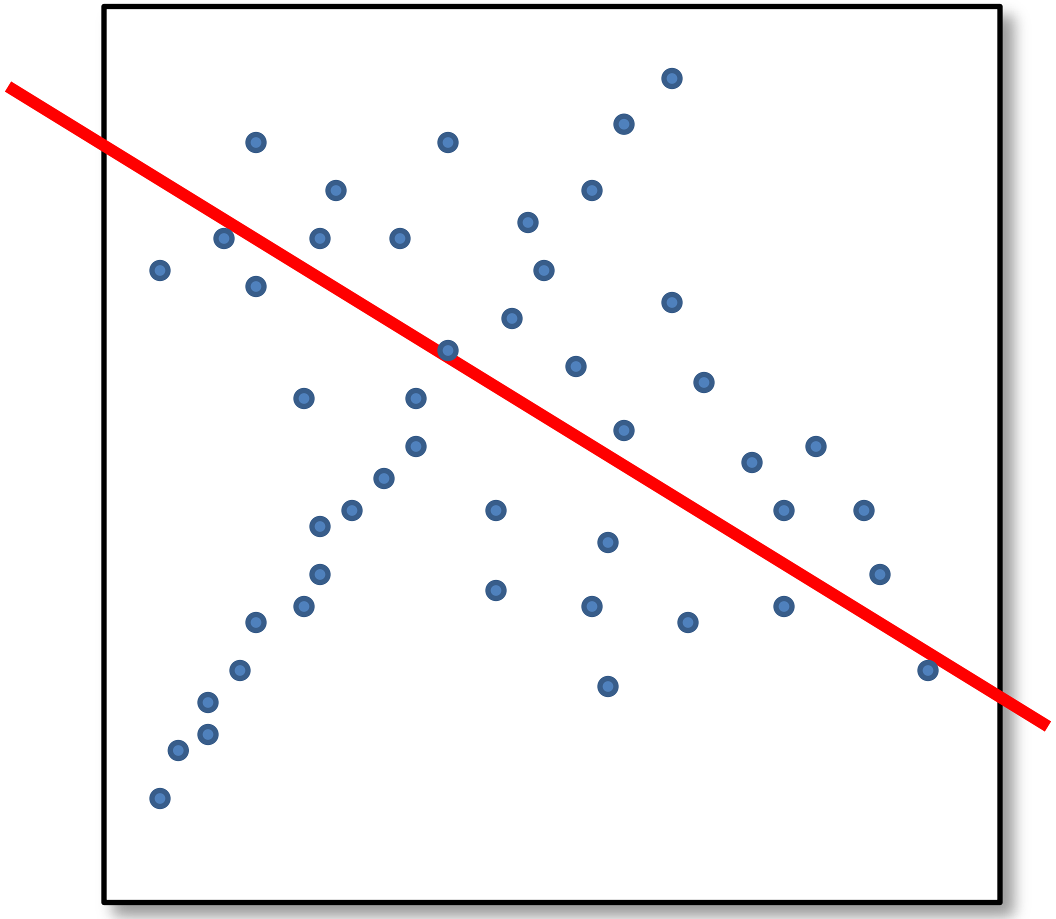
- Given a hypothesized line
- Count the number of points that "agree" with the line
  - "Agree" = within a small distance of the line
  - I.e., the **inliers** to that line

- For all possible lines, select the one with the largest number of inliers

Counting inliers

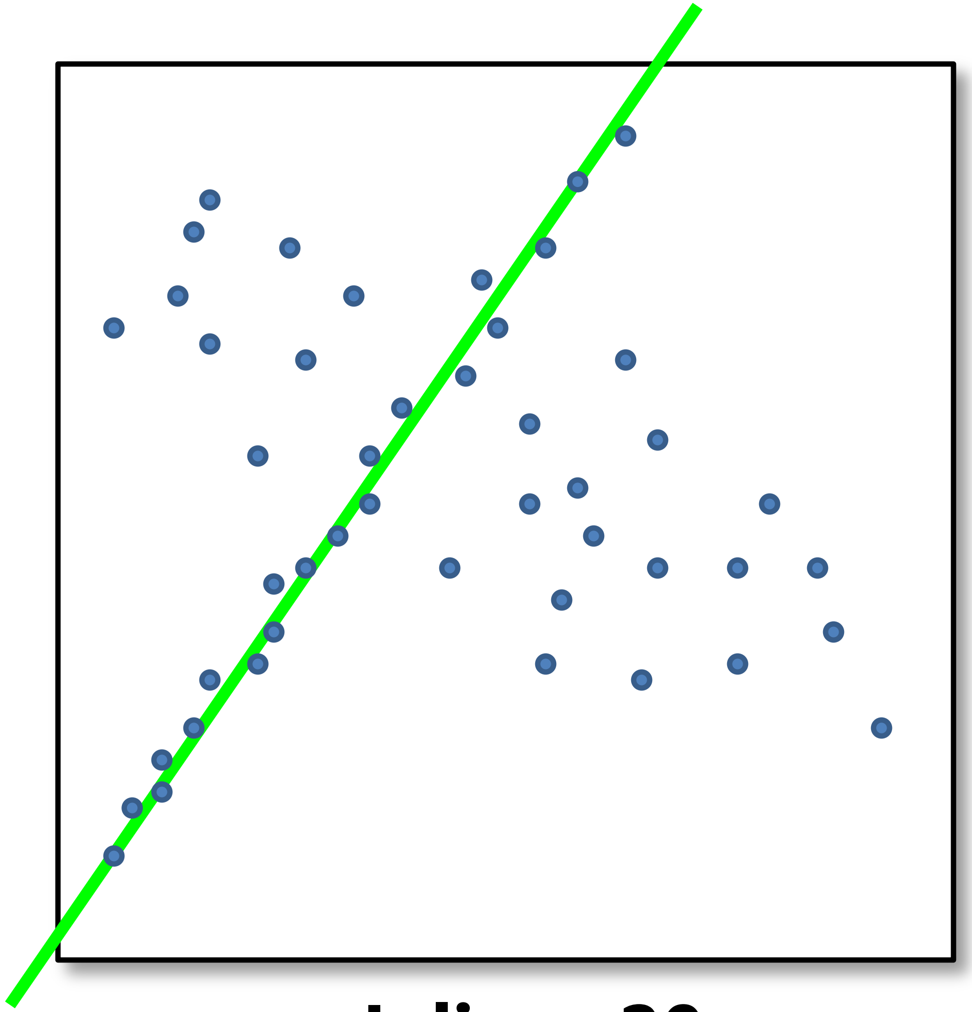


Counting inliers



Inliers: 3

Counting inliers

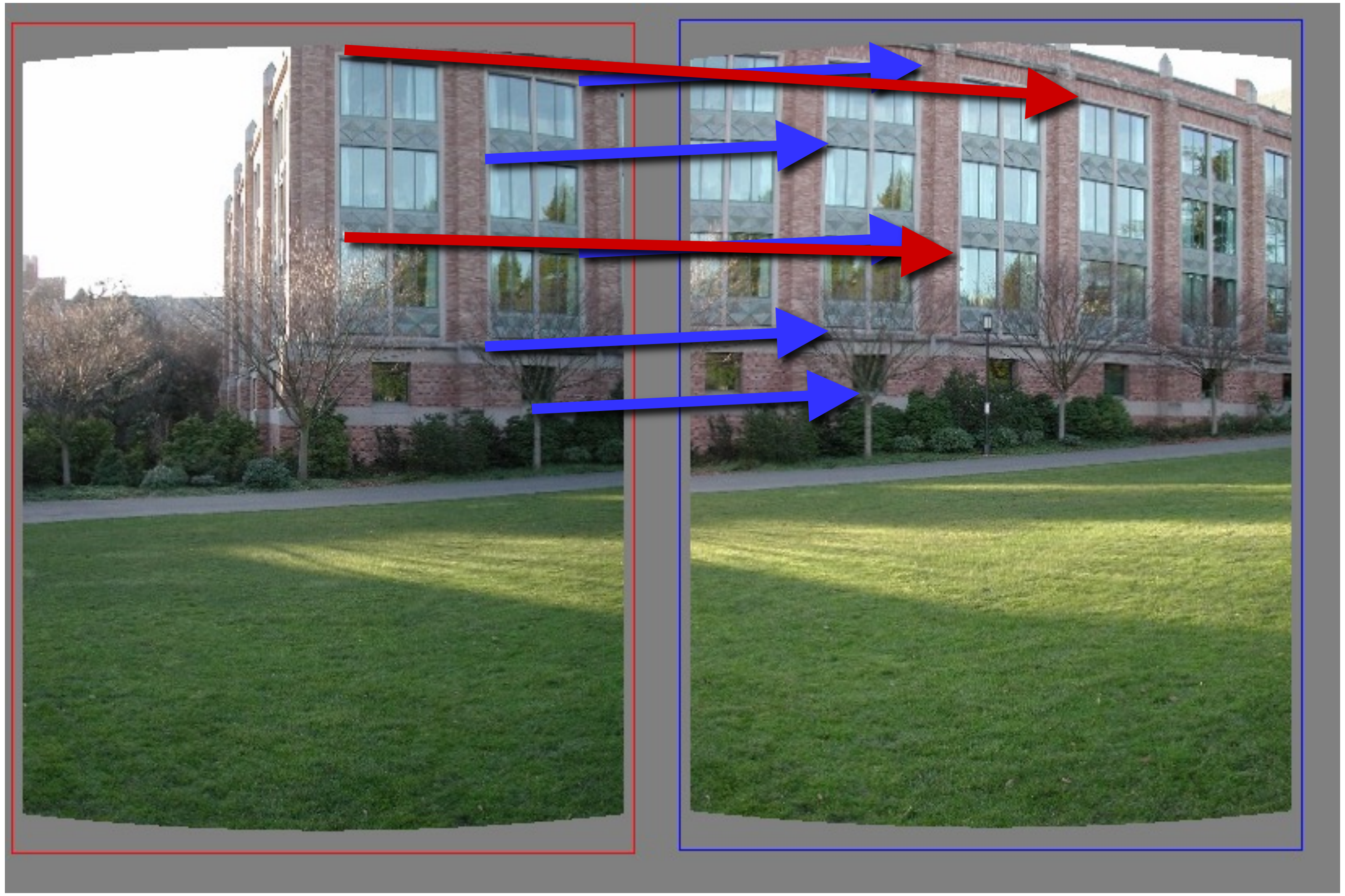


Inliers: 20

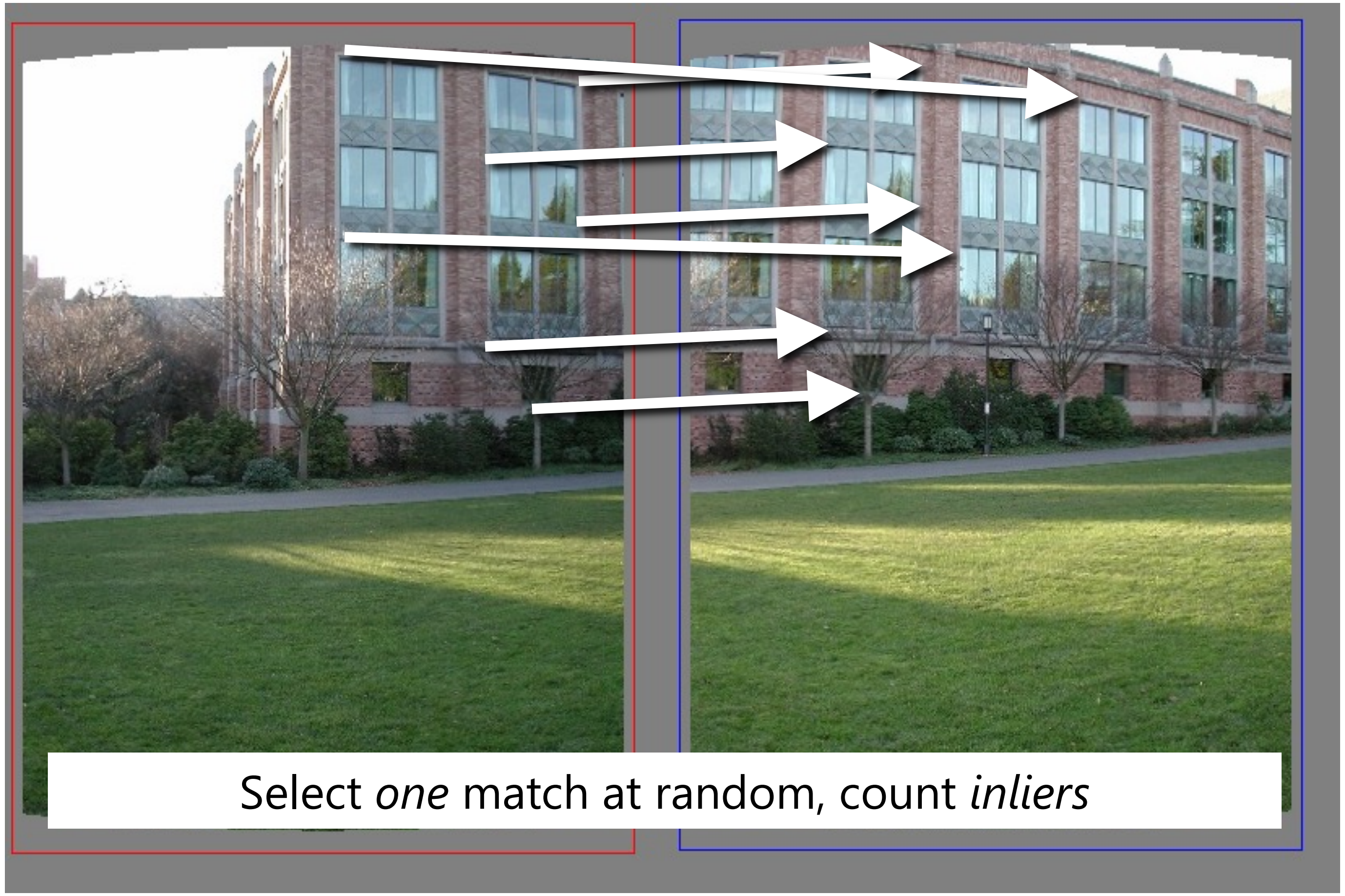
How do we find the best line?

- Unlike least-squares, no simple closed-form solution
- Hypothesize-and-test
  - Try out many lines, keep the best one
  - Which lines?

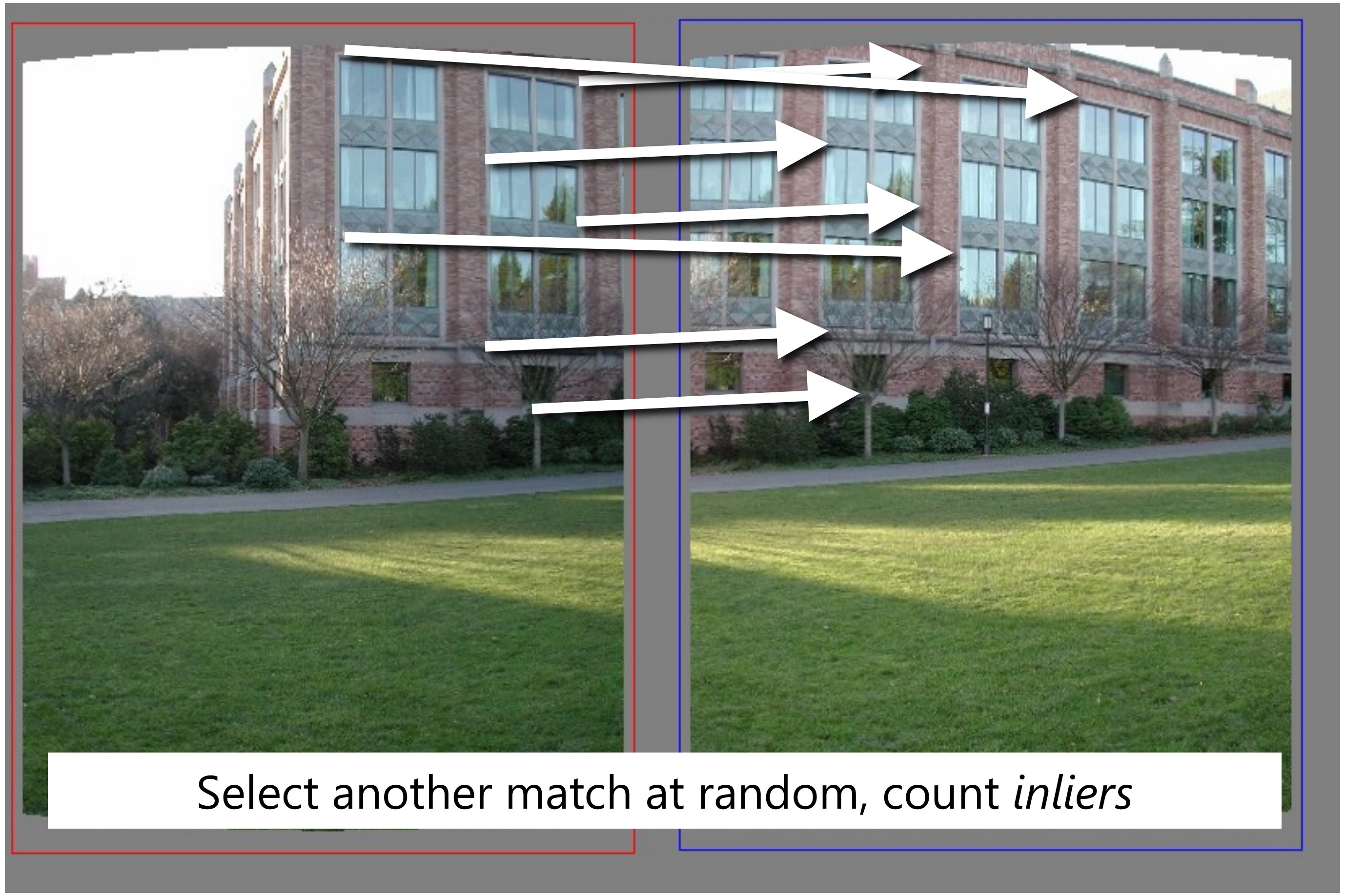
Translations



<u>RA</u>ndom <u>SA</u>mple <u>C</u>onsensus

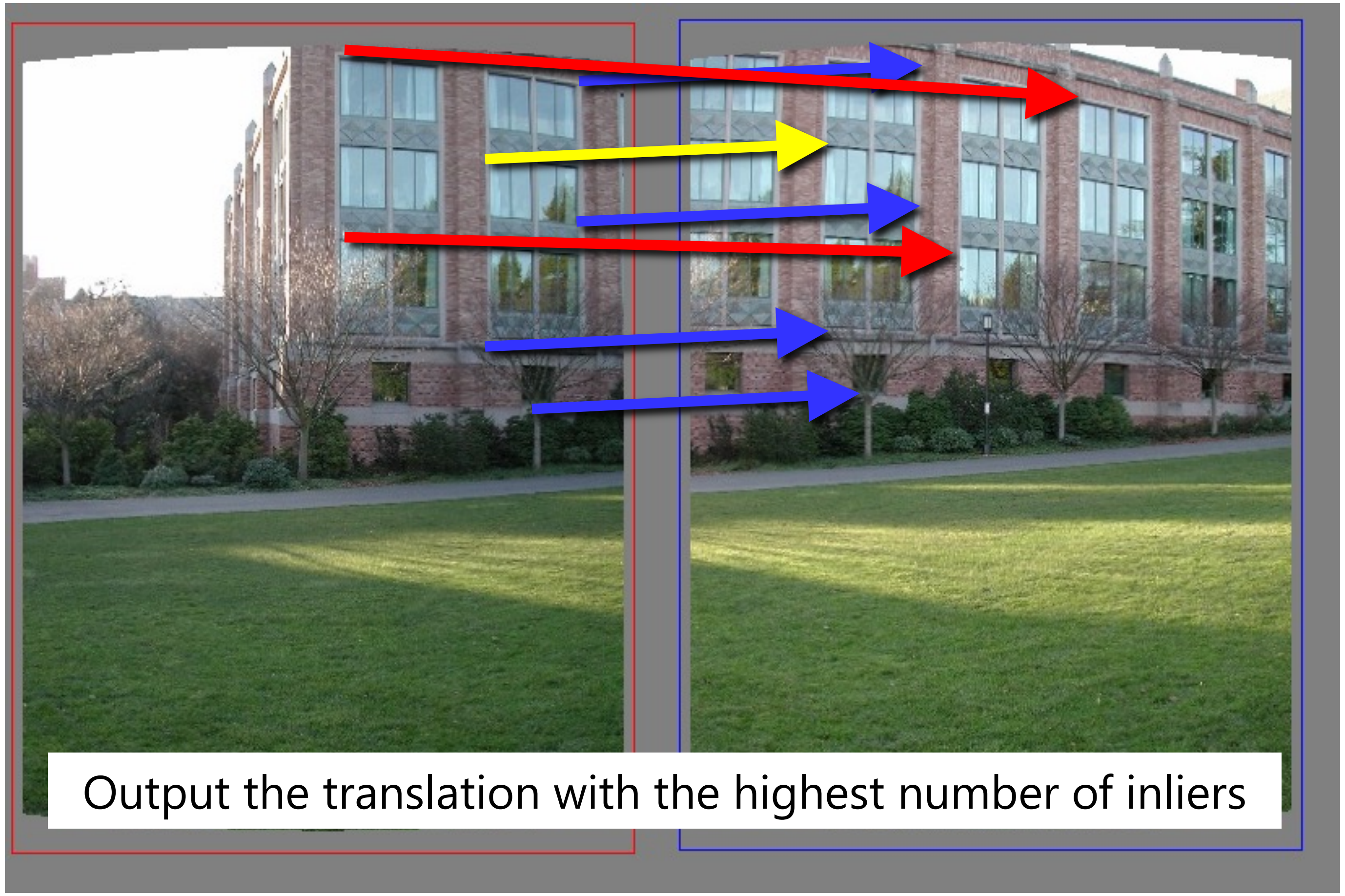


RAndom SAmple Consensus



Select another match at random, count *inliers*

RAndom SAmple Consensus



Output the translation with the highest number of inliers

RANSAC

- Idea:
  - All the inliers will agree with each other on the translation vector; the (hopefully small) number of outliers will (hopefully) disagree with each other
    - RANSAC only has guarantees if there are < 50% outliers

  - "All good matches are alike; every bad match is bad in its own way."

– Tolstoy via Alyosha Efros

RANSAC

- **Inlier threshold** related to the amount of noise we expect in inliers
  - Often model noise as Gaussian w/ some standard deviation (e.g. 3 pixels)
- **Number of rounds** related to the percentage of outliers we expect, and the probability of success we'd like to guarantee
  - Suppose there are 20% outliers, and we want to find the correct answer with at least 99% probability
  - How many rounds do we need?

Scratch space

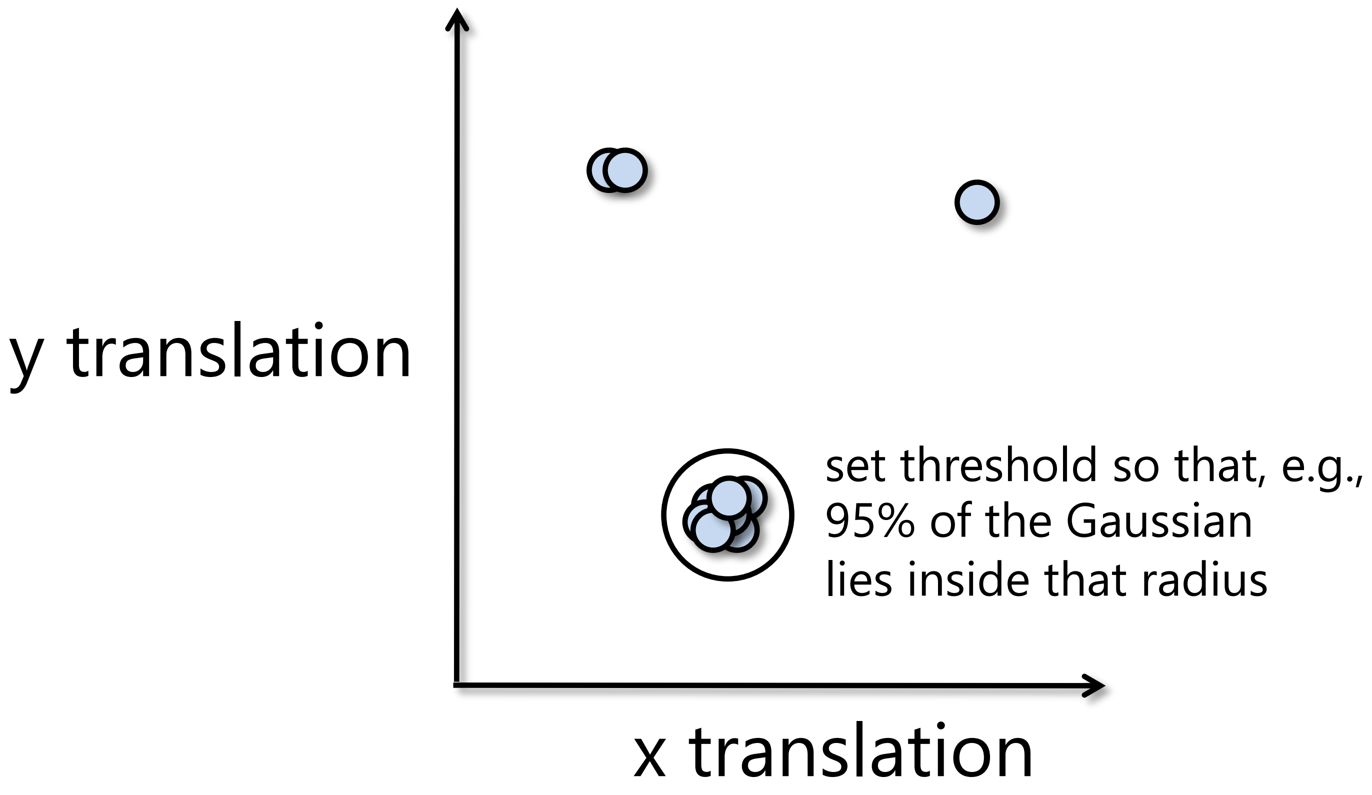
$$0.2^N \le 0.01$$

$$N \log 0.2 \le \cancel{2}\ - \log(0.01)$$

$$N \ge \frac{\log 0.01}{\log 0.2}$$

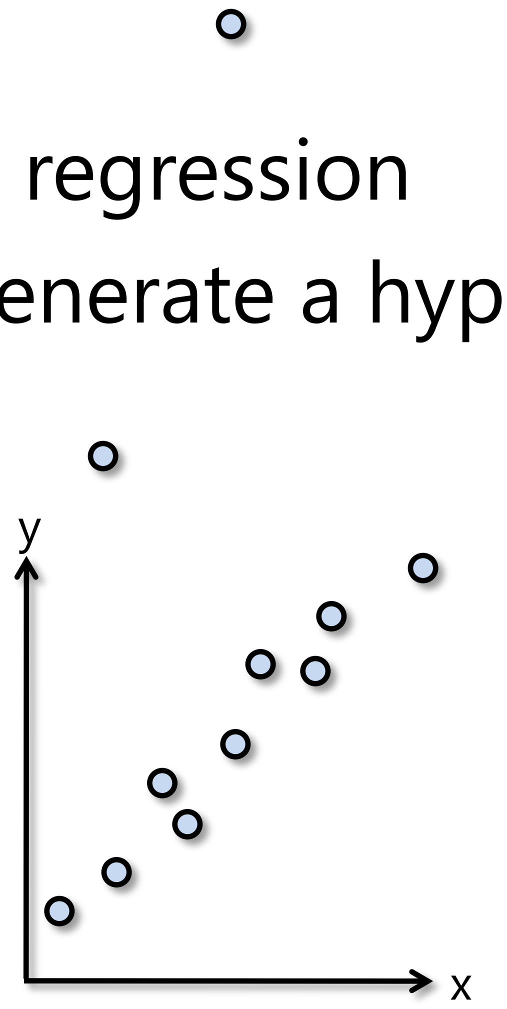
$$N \ge 3$$

RANSAC: Another view



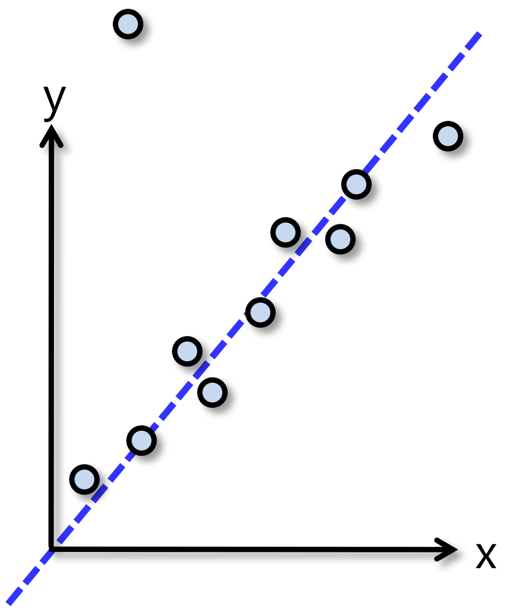
RANSAC

- Back to linear regression
- How do we generate a hypothesis?



RANSAC

- Back to linear regression
- How do we generate a hypothesis?



RANSAC

- General version:
  1. Randomly choose *s* samples
     - Typically *s* = minimum sample size that lets you fit a model
  2. Fit a model (e.g., line) to those samples
  3. Count the number of inliers that approximately fit the model
  4. Repeat *N* times
  5. Choose the model that has the largest set of inliers

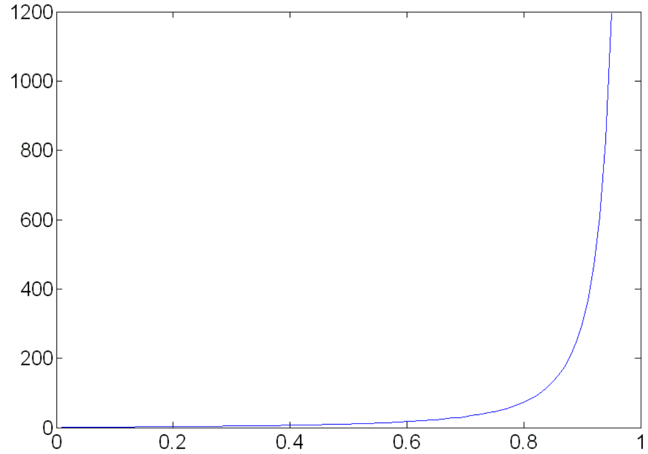
How many rounds?

- If we have to choose *s* samples each time
  - with an outlier ratio *e*
  - and we want the right answer with probability *p*

$$N \geq \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

	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

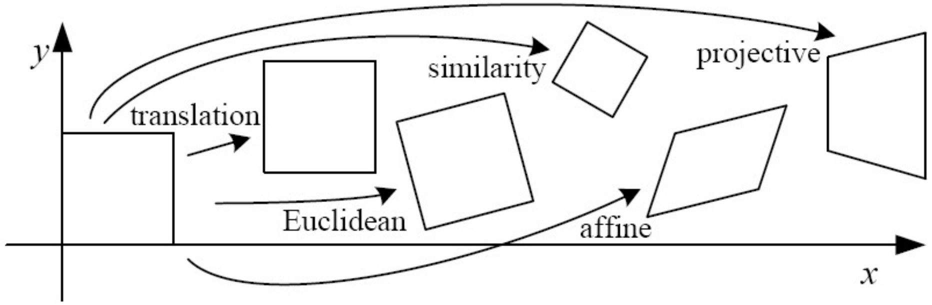
p = 0.99



Source: M. Pollefeys

How big is *s*?

- For alignment, depends on the motion model
  - Here, each sample is a correspondence (pair of matching points)

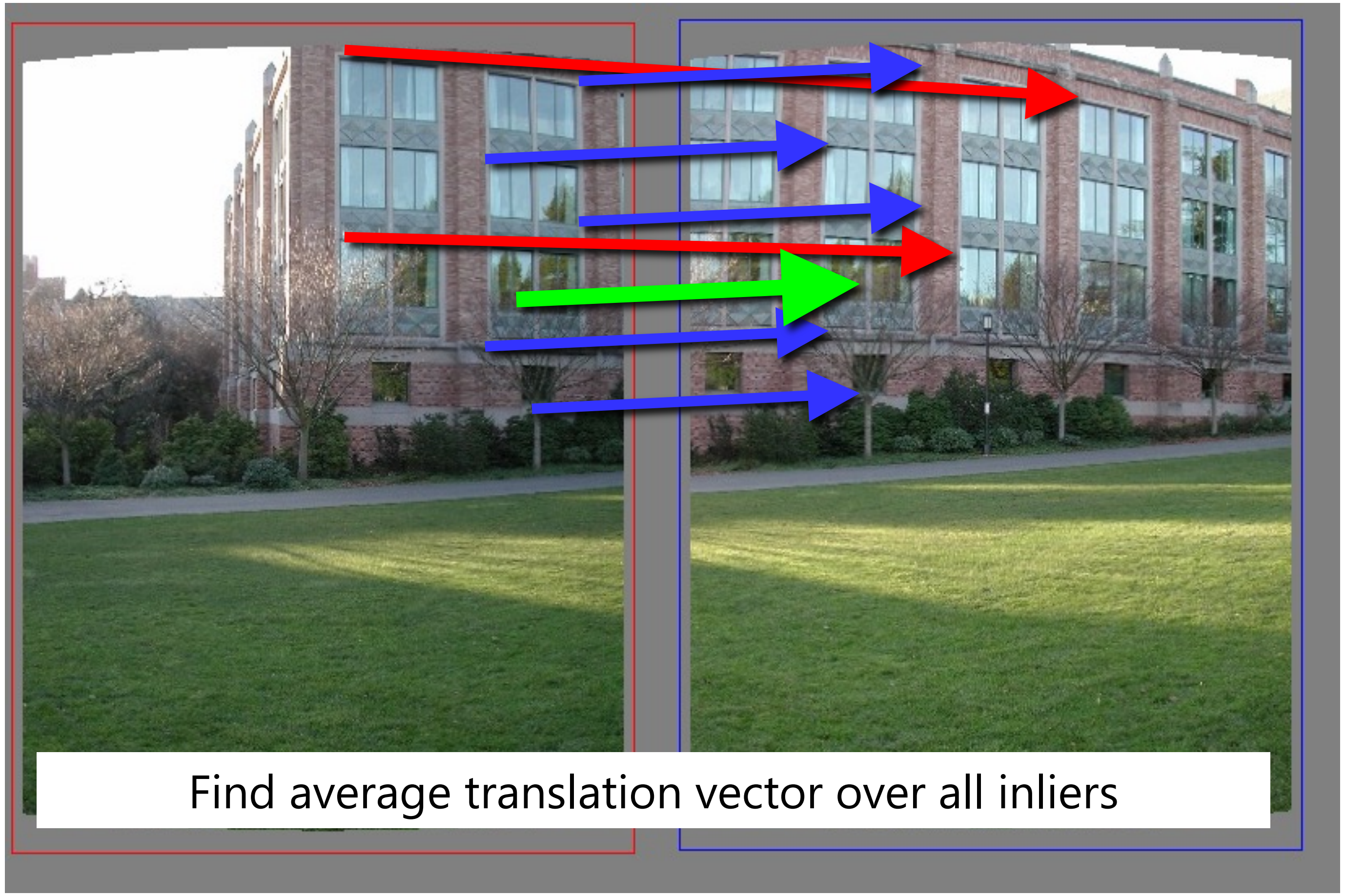


Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\left[\begin{array}{c\|c} \boldsymbol{I} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	2	orientation $+\cdots$	
rigid (Euclidean)	$\left[\begin{array}{c\|c} \boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	3	lengths $+\cdots$	
similarity	$\left[\begin{array}{c\|c} s\boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	4	angles $+\cdots$	
affine	$\left[\begin{array}{c} \boldsymbol{A} \end{array}\right]_{2\times 3}$	6	parallelism $+\cdots$	
projective	$\left[\begin{array}{c} \tilde{\boldsymbol{H}} \end{array}\right]_{3\times 3}$	8	straight lines	

RANSAC pros and cons

- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Parameters to tune
  - Sometimes too many iterations are required
  - Can fail for extremely low inlier ratios
  - We can often do better than brute-force sampling

Final step: least squares fit



RANSAC

- An example of a "voting"-based fitting scheme
- Each hypothesis gets voted on by each data point, best hypothesis wins

- There are many other types of voting schemes
  - E.g., Hough transforms…

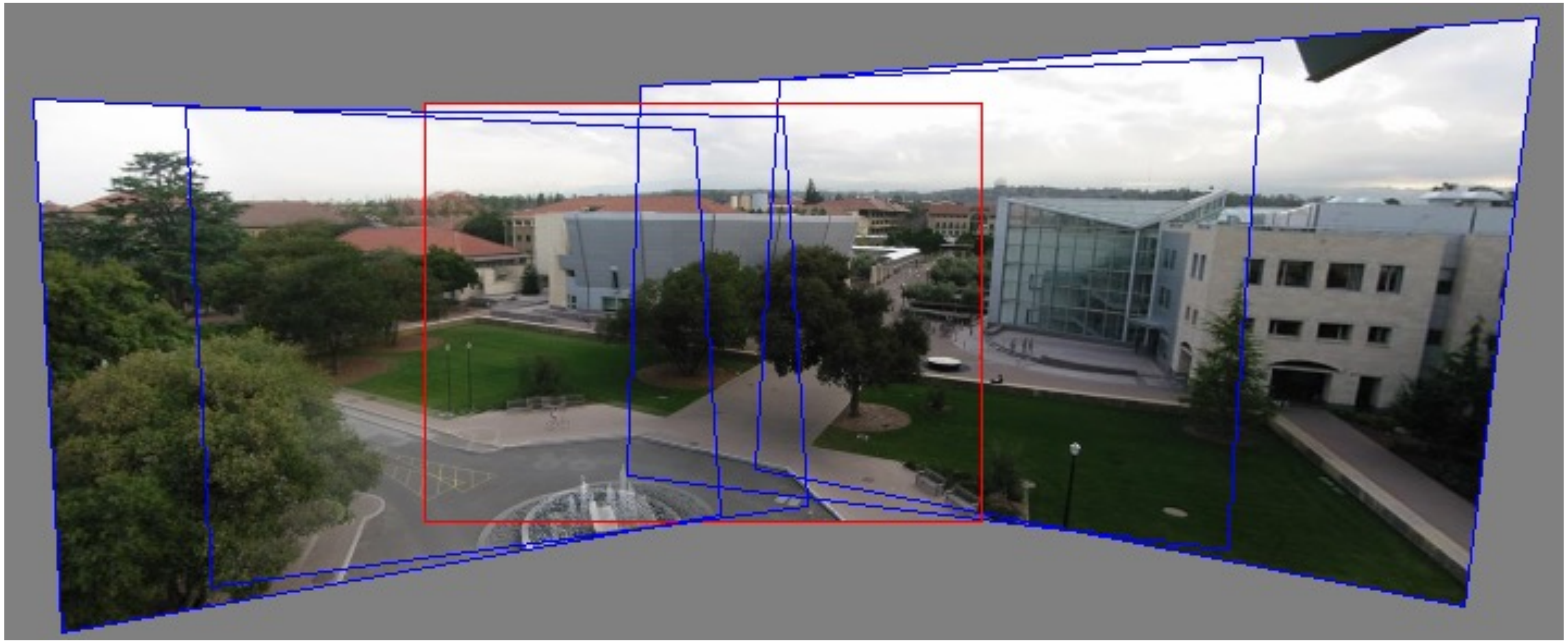
Panoramas

- Now we know how to create panoramas!

- Given two images:
  - Step 1: Detect features
  - Step 2: Match features
  - Step 3: Compute a homography using RANSAC
  - Step 4: Combine the images together (somehow)

- What if we have more than two images?

Can we use homographies to create a 360 panorama?



- To figure this out, we need to know what a **camera** is

360 panorama

Questions?