Robust fitting

Camera calibration

- Given a set of correspondences between 3D world points and image points:
  - $(\overrightarrow{Q_1}, \overrightarrow{q_1})$, … , $(\overrightarrow{Q_n}, \overrightarrow{q_n})$
- Set up an optimization problem:

$$\min_{\mathbf{p}} \|A\mathbf{p}\|_2$$

$$\text{such that}$$

$$\|\mathbf{p}\|_2 = 1$$

- Solve for P

Homography estimation

- Given a set of correspondences between 3D world points on a plane and image points:
  - $(\overrightarrow{Q_1}, \overrightarrow{q_1}), \ldots, (\overrightarrow{Q_n}, \overrightarrow{q_n})$
- Set up an optimization problem:

$$\min_{\mathbf{h}} \|A\mathbf{h}\|_2$$

$$\text{such that}$$

$$\|\mathbf{h}\|_2 = 1$$

- Solve for H

Fundamental matrix/ essential matrix estimation

- Given a set of correspondences between image points in two images:
  - $\left(\overrightarrow{q_1^1}, \overrightarrow{q_1^2}\right), \ \ldots \ , \left(\overrightarrow{q_n^1}, \overrightarrow{q_n^2}\right)$
- Set up an optimization problem:

$$\min_{\mathbf{f}} \|A\mathbf{f}\|_2$$
$$\text{such that}$$
$$\|\mathbf{f}\|_2 = 1$$
$$\text{rank}(F) = 2$$

- Solve for F

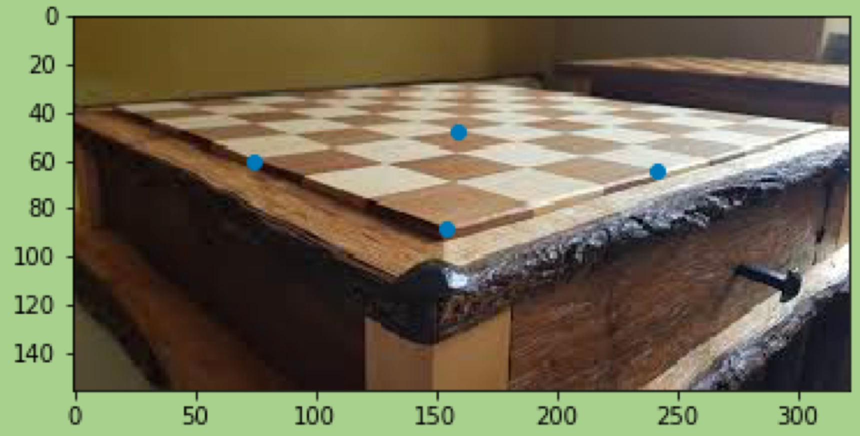
What happens when correspondences are wrong?

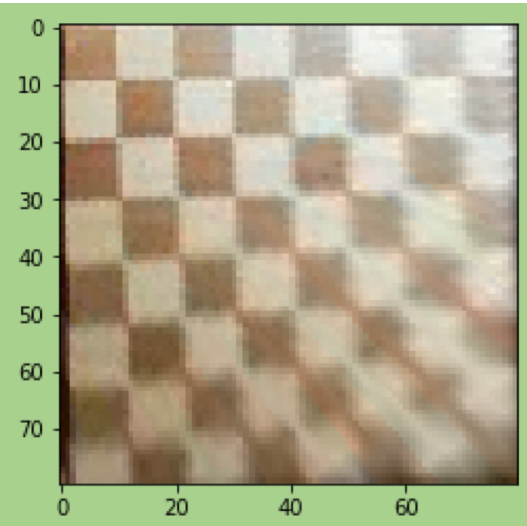
- Not just noisy
  - Noise (e.g., ~1 pixel) can be handled because we are solving a minimization problem, rather than exactly satisfy equations

- Wrong correspondences can be way off.

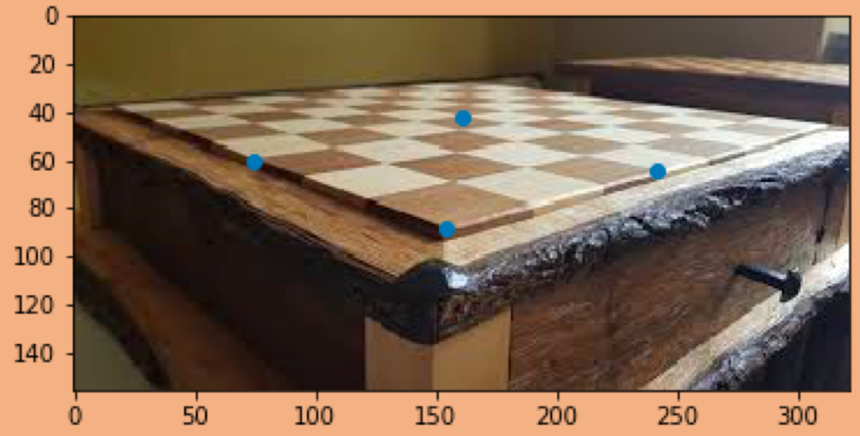
Impact of incorrect correspondences

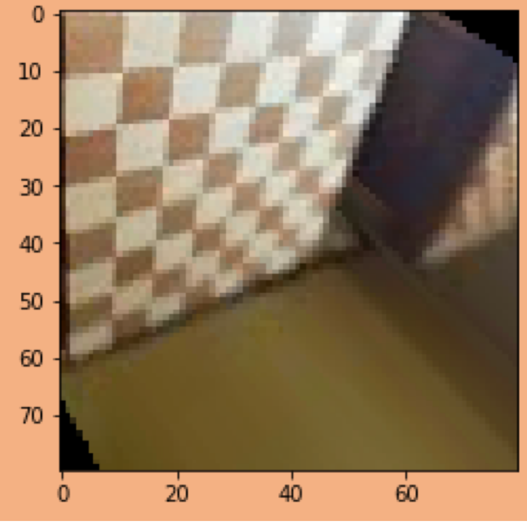
Correct:



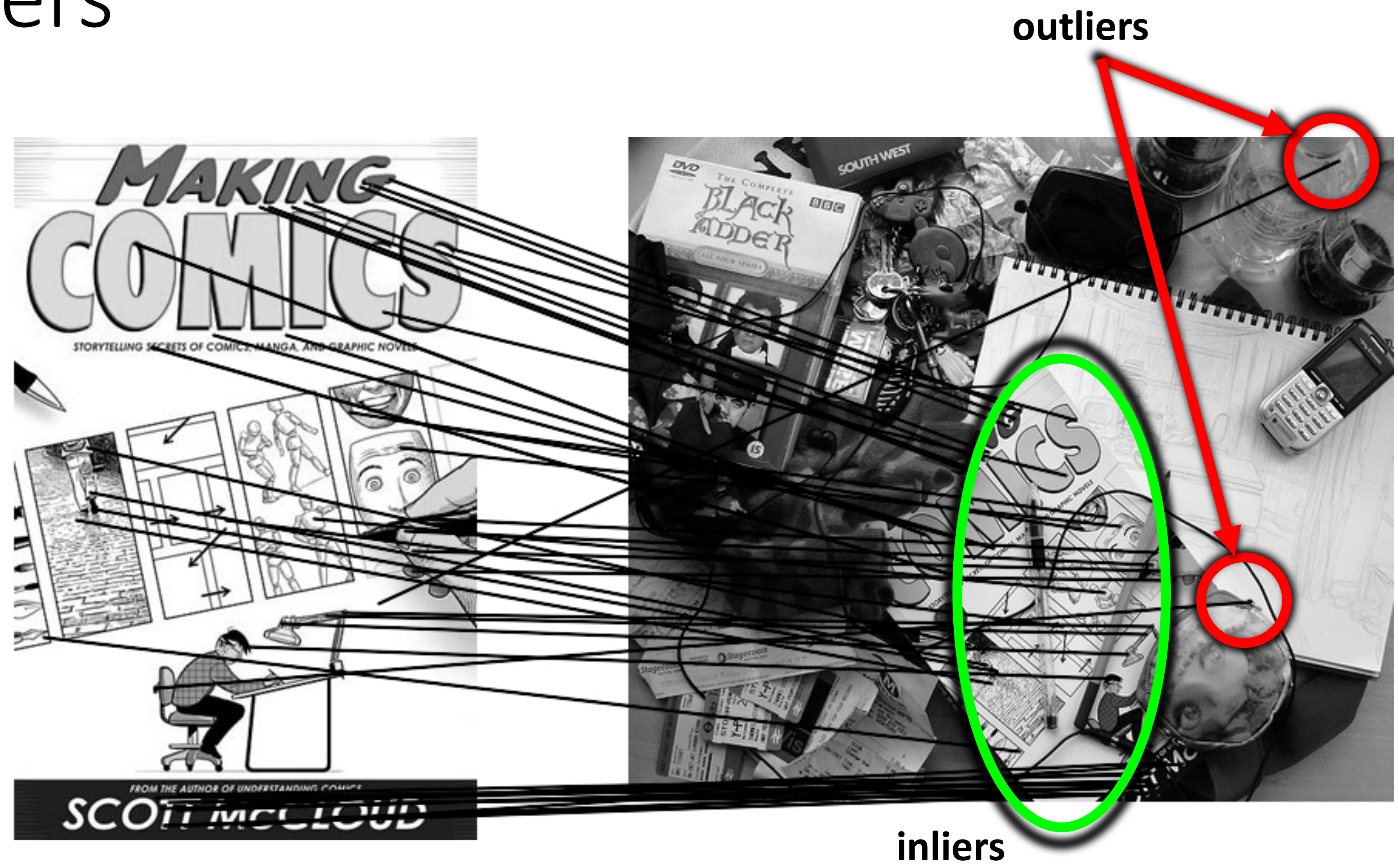


Incorrect:





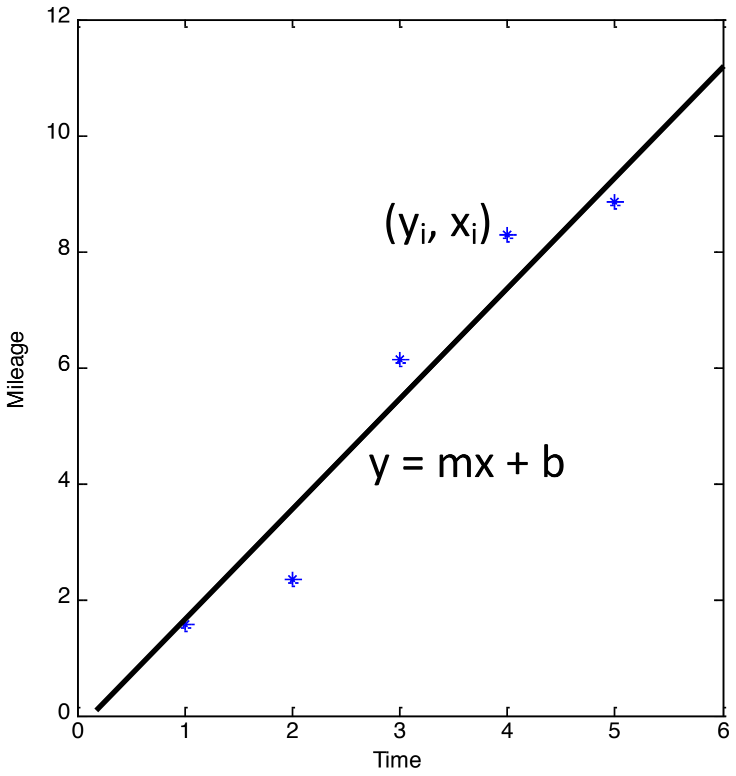
Outliers


outliers
inliers
MAKING COMICS
STORYTELLING SECRETS OF COMICS, MANGA, AND GRAPHIC NOVELS
FROM THE AUTHOR OF UNDERSTANDING COMICS
SCOTT McCLOUD
DVD
THE COMPLETE BLACK ADDER
BBC
SOUTH WEST

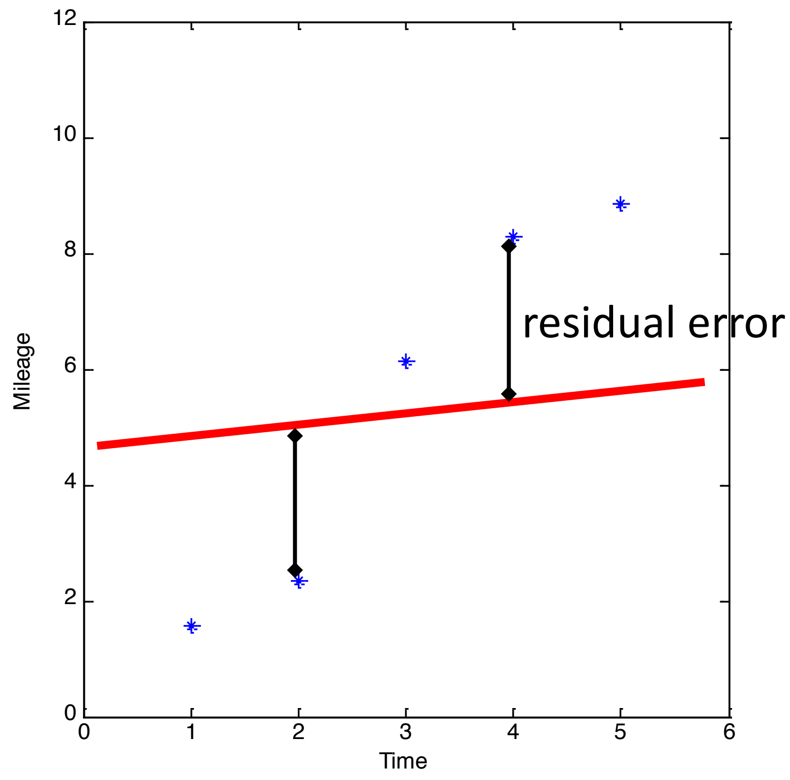
Fitting in general

- Fitting: find the parameters of a model that best fit the data
- Other examples:
  - least squares linear regression

Least squares: linear regression

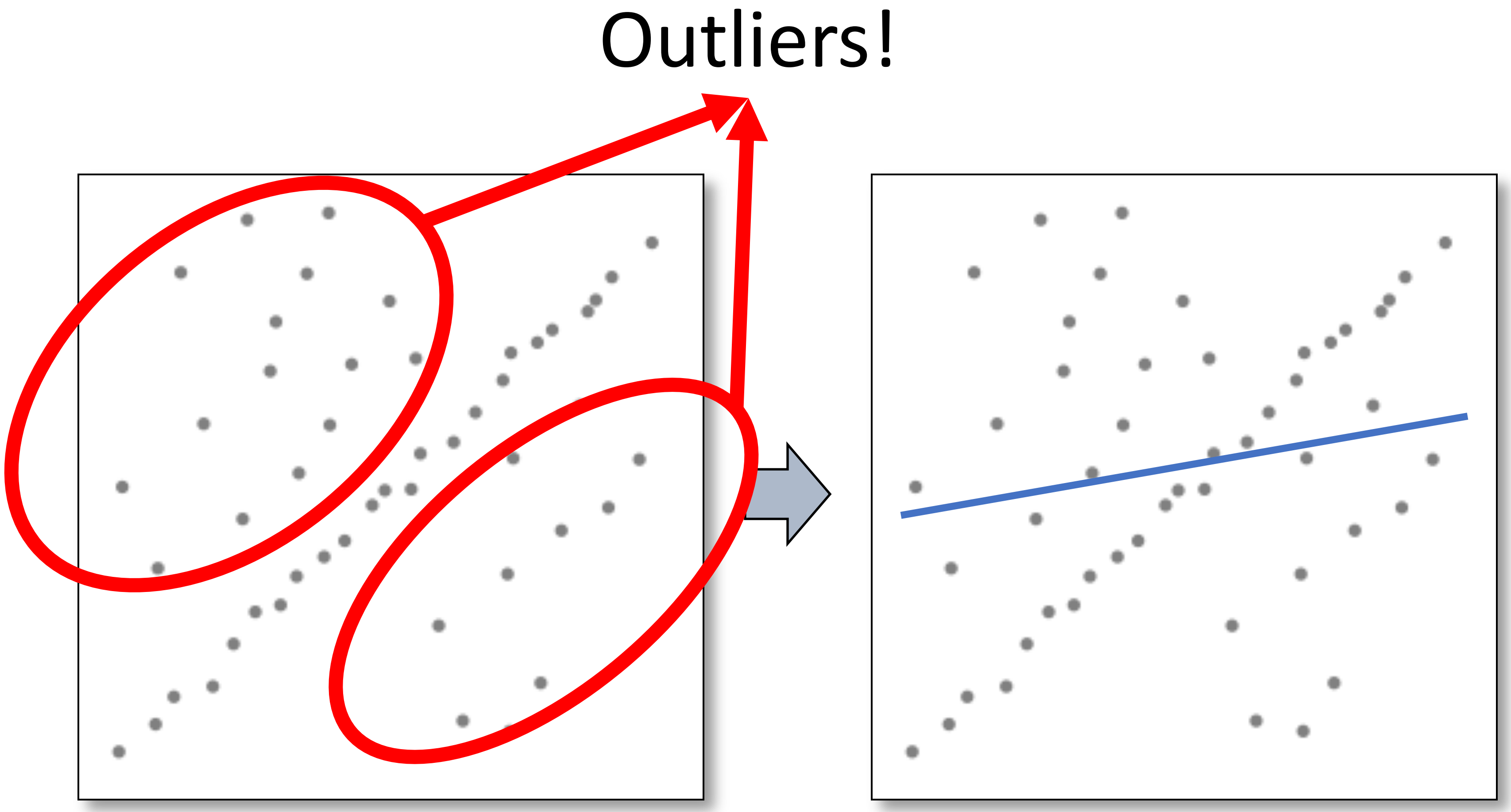


Linear regression



$$\min_{m,b} \sum_i (y_i - (mx_i + b))^2$$

Robustness



Problem: Fit a line to these datapoints

Least squares fit

How do we find the best line in the presence of outliers?

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

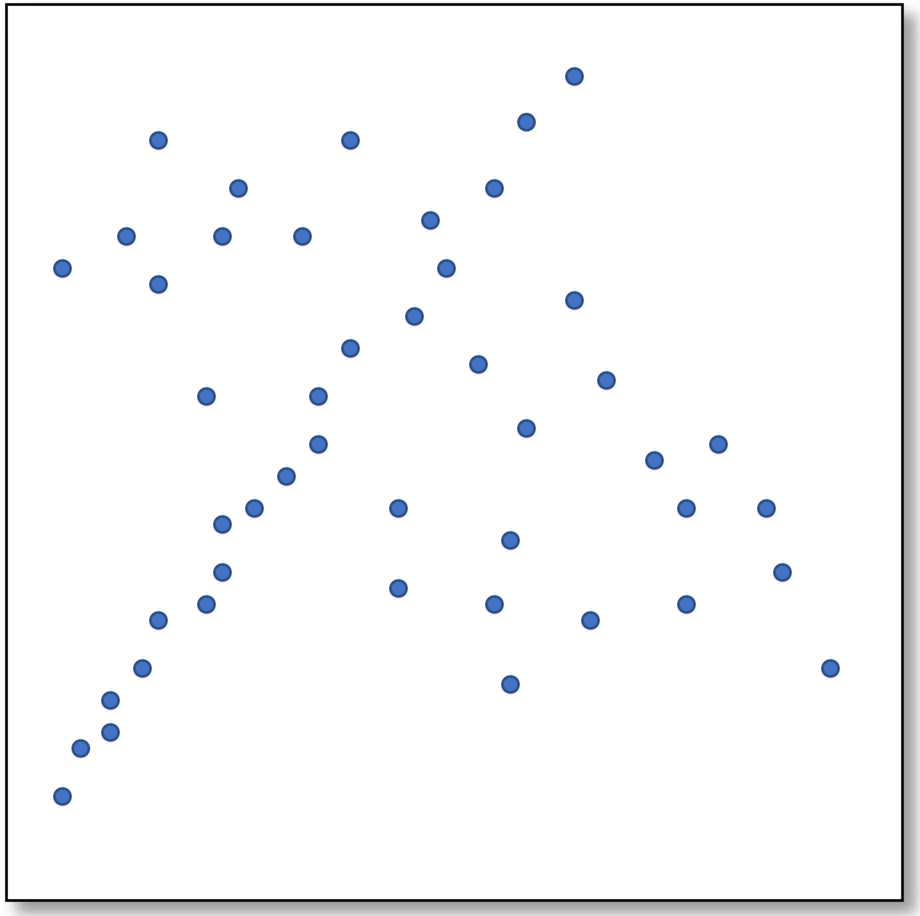
Choosing lines to hypothesize

- Randomly choose lines?
  - Not optimal: highly unlikely to run into correct line by chance

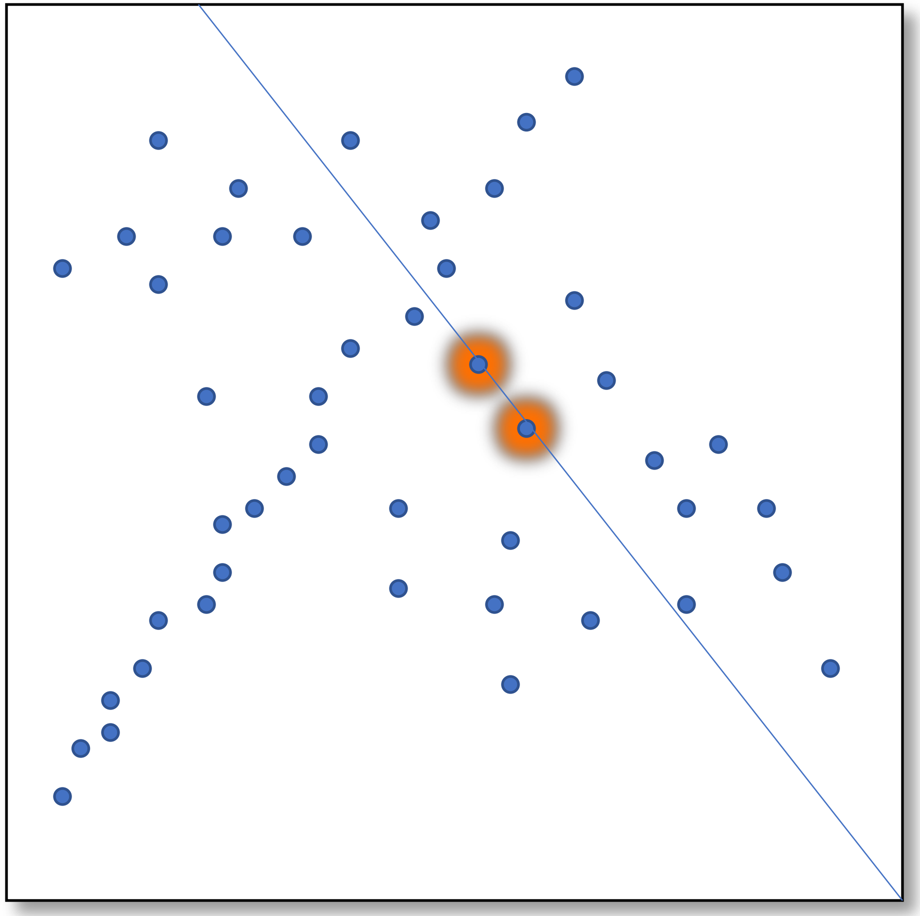
- Idea: randomly sample data points from the dataset and fit line to them

- How many data points should we sample?
  - Any point we pick might be an outlier ☹
  - Want to maximize the chance that all sampled points are inliers → get the true line
  - Idea: sample the *minimum number of points necessary to get a line*

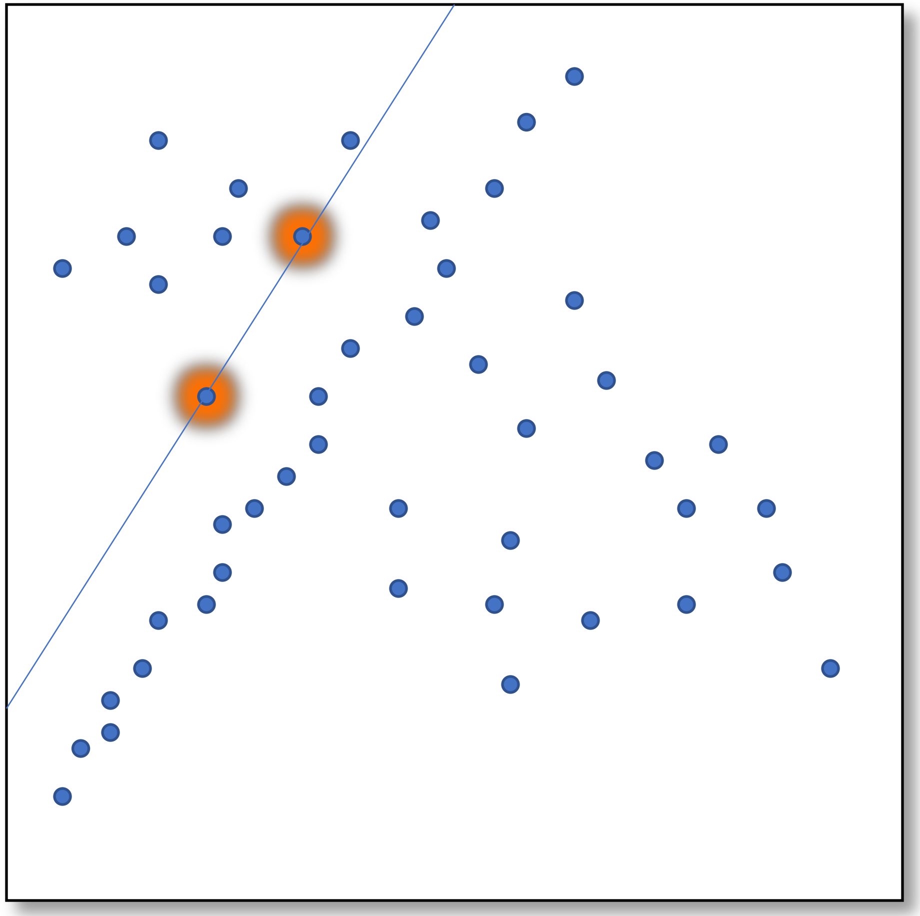
Choosing lines to hypothesize



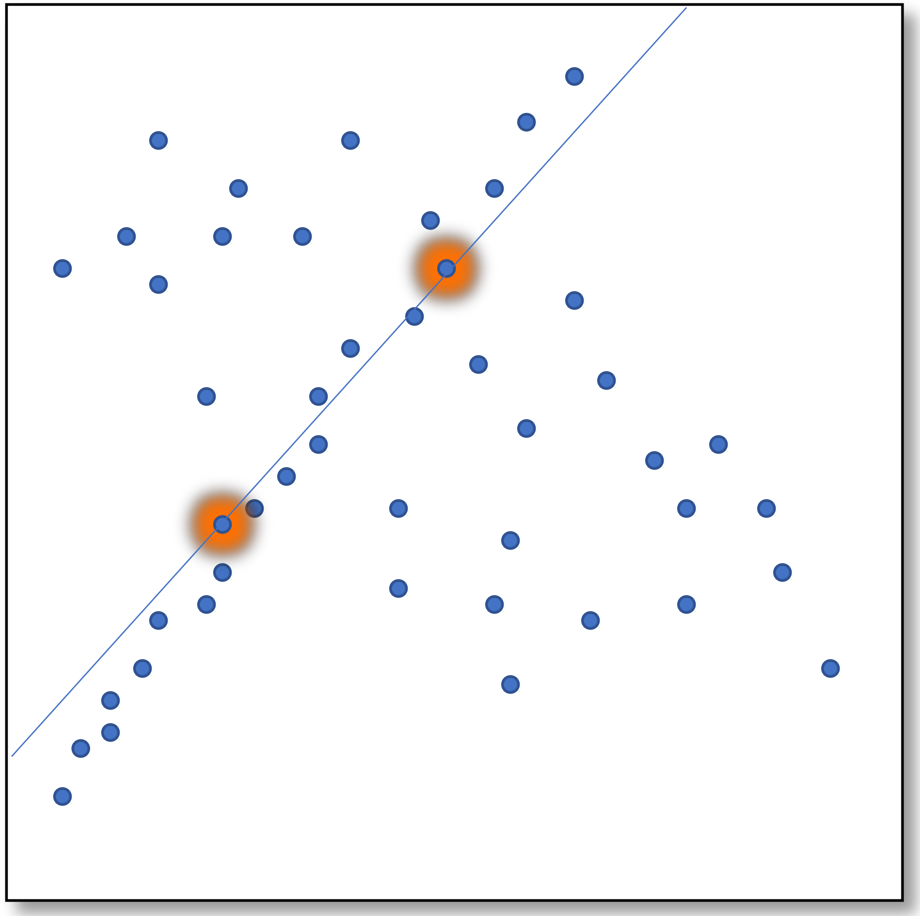
Choosing lines to hypothesize



Choosing lines to hypothesize



Choosing lines to hypothesize

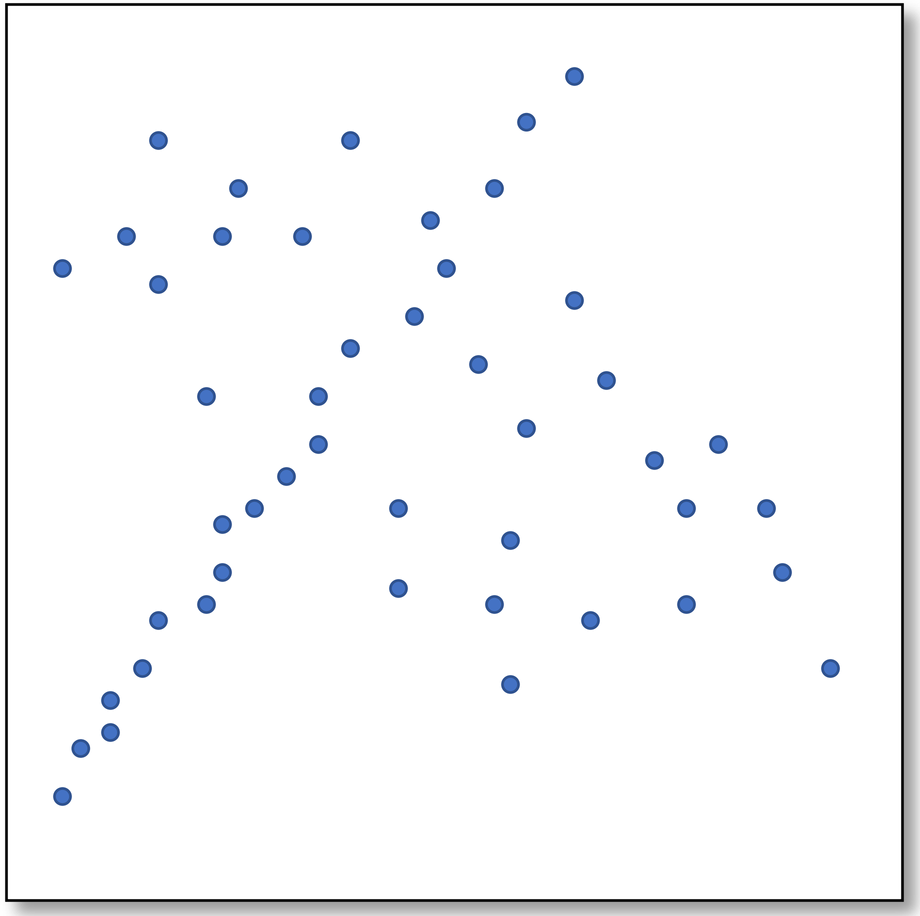


Measuring goodness of a line

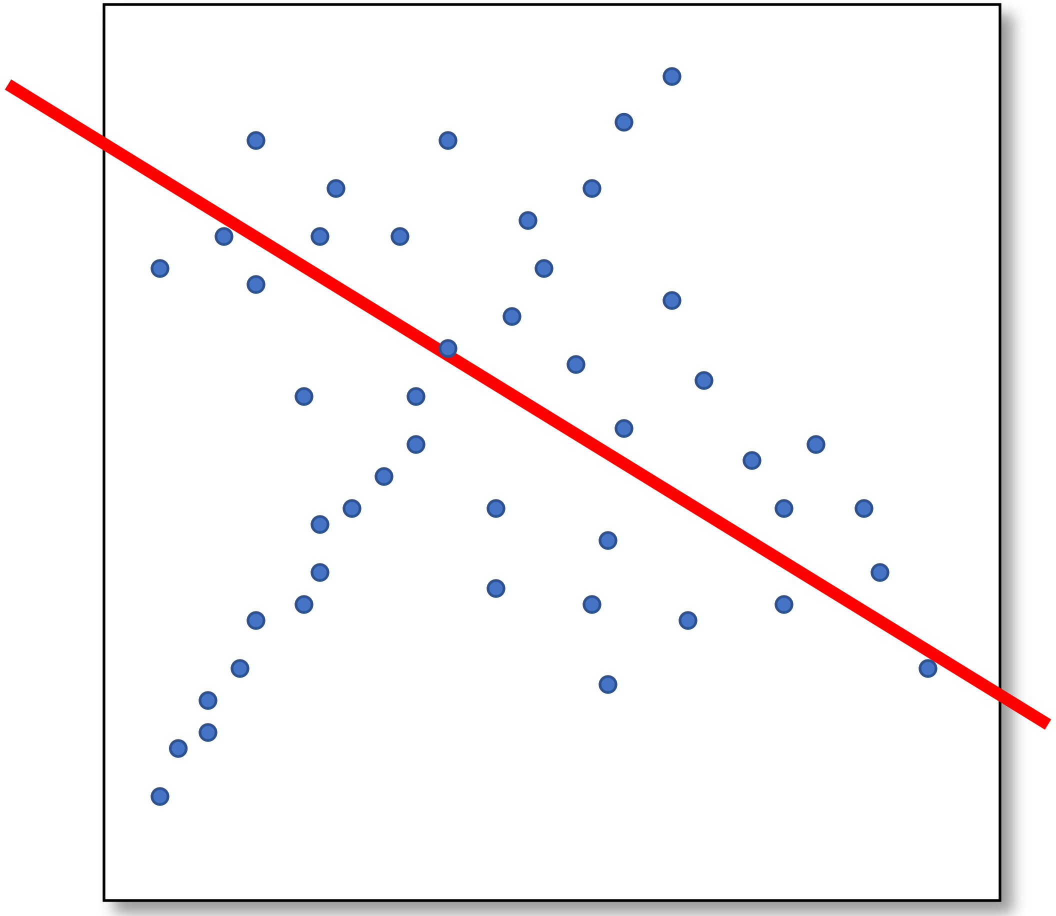
- Given a hypothesized line
- Count the number of points that “agree” with the line
  - “Agree” = within a small distance of the line
  - I.e., the **inliers** to that line

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

Counting inliers

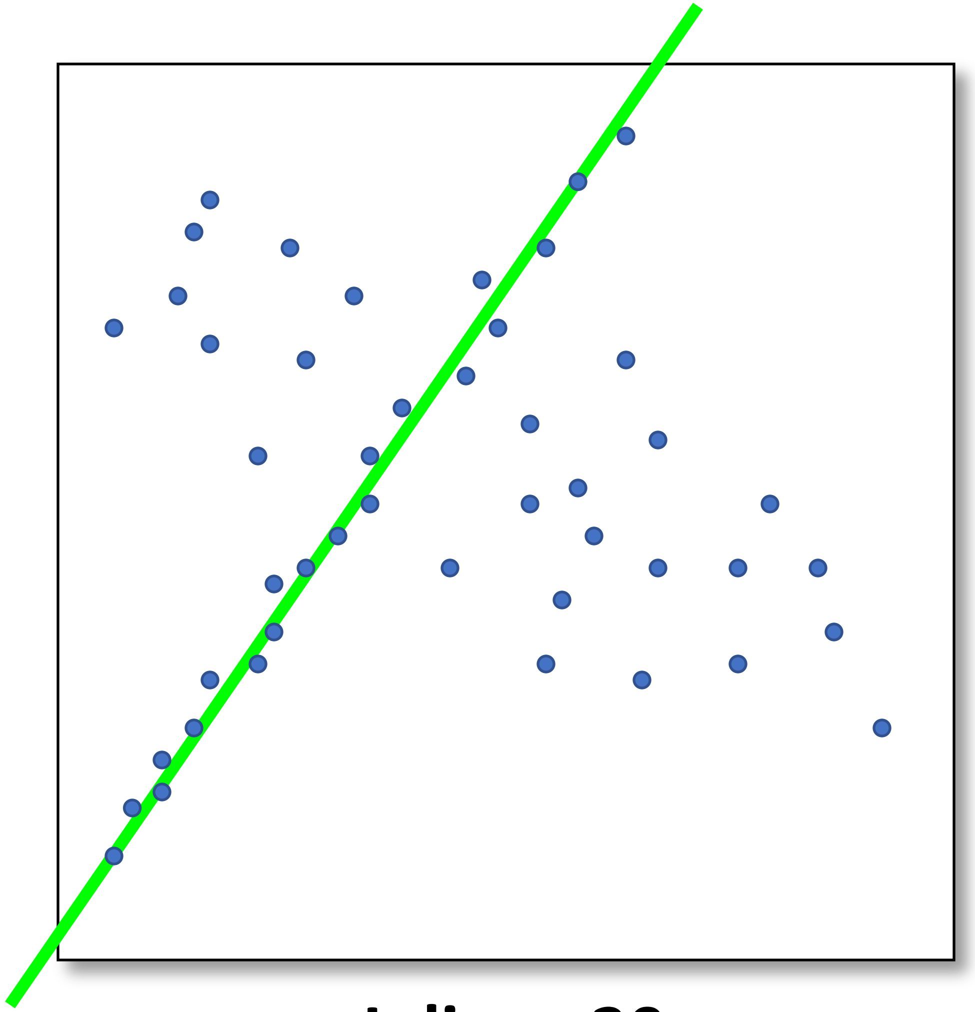


Counting inliers



Inliers: 3

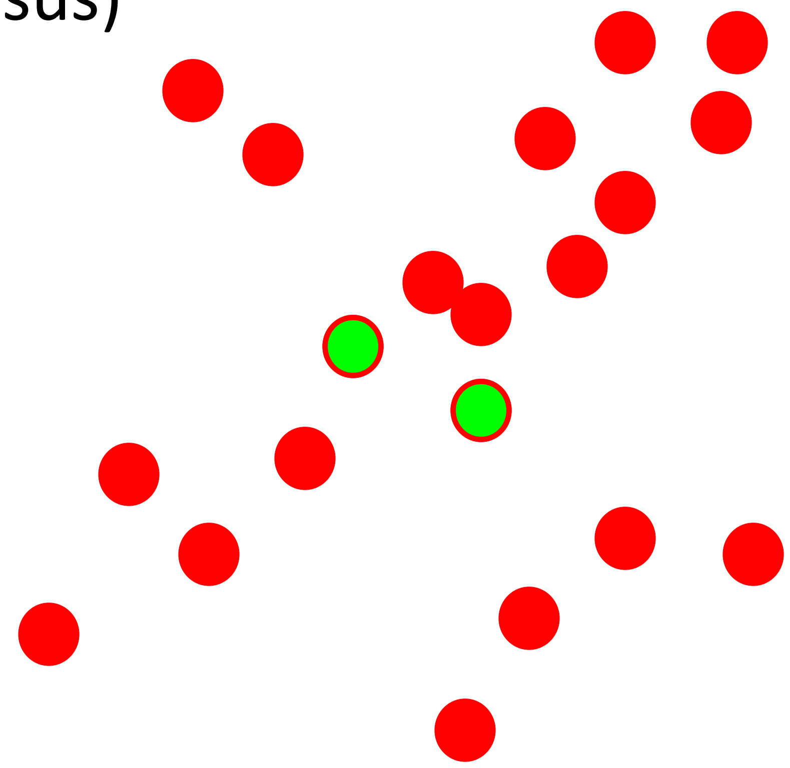
Counting inliers



Inliers: 20

RANSAC (Random Sample Consensus)

Line fitting example



Algorithm:

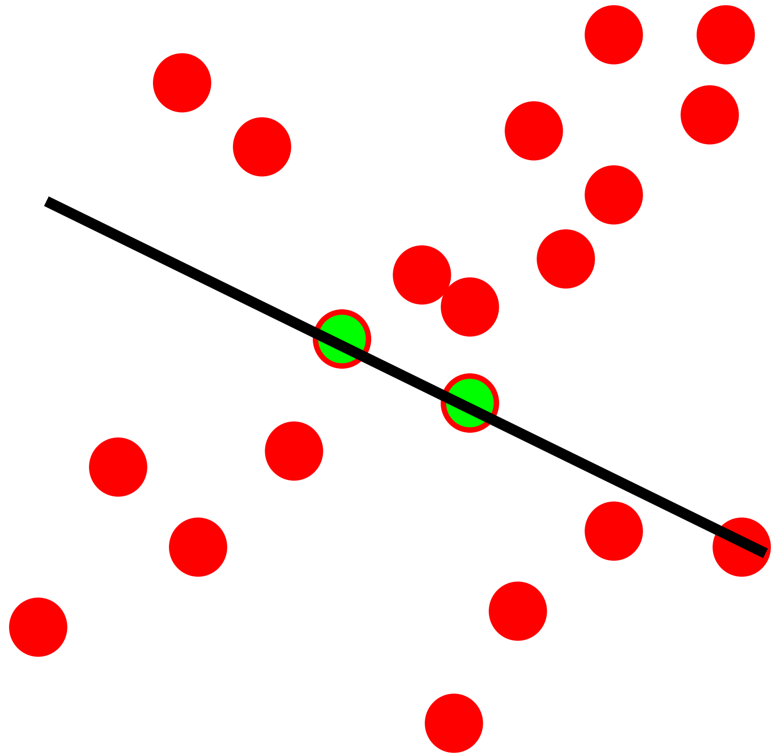
1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Illustration by Savarese

RANSAC

Line fitting example



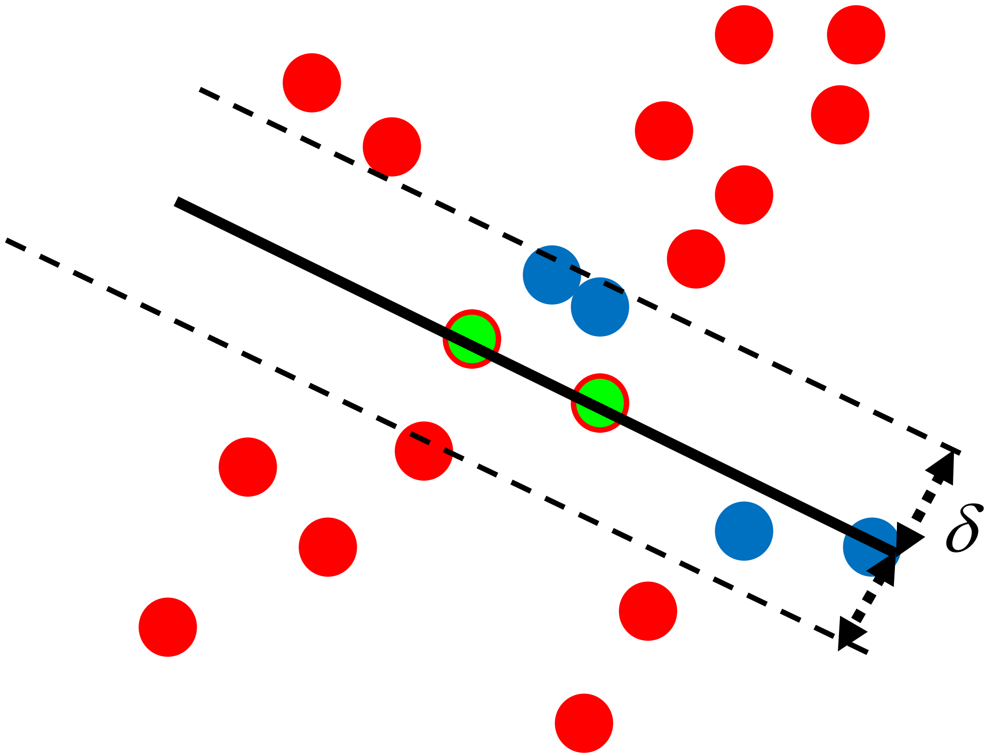
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC

Line fitting example

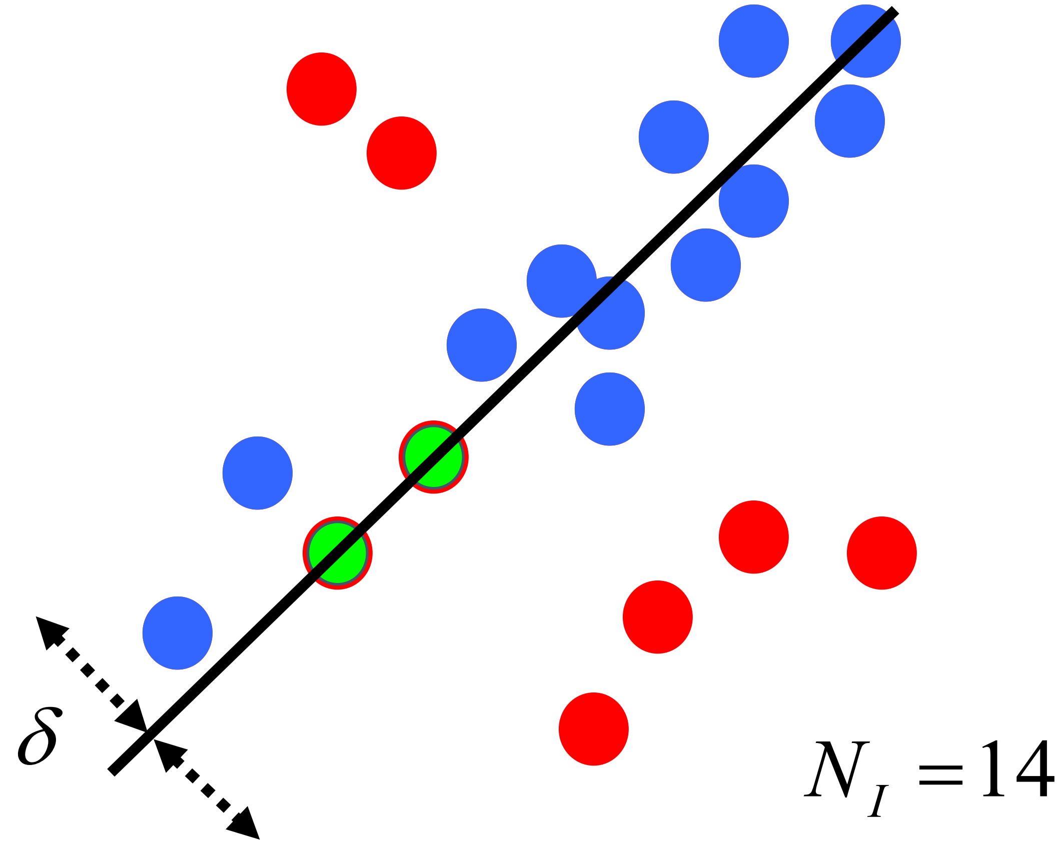


Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC



Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

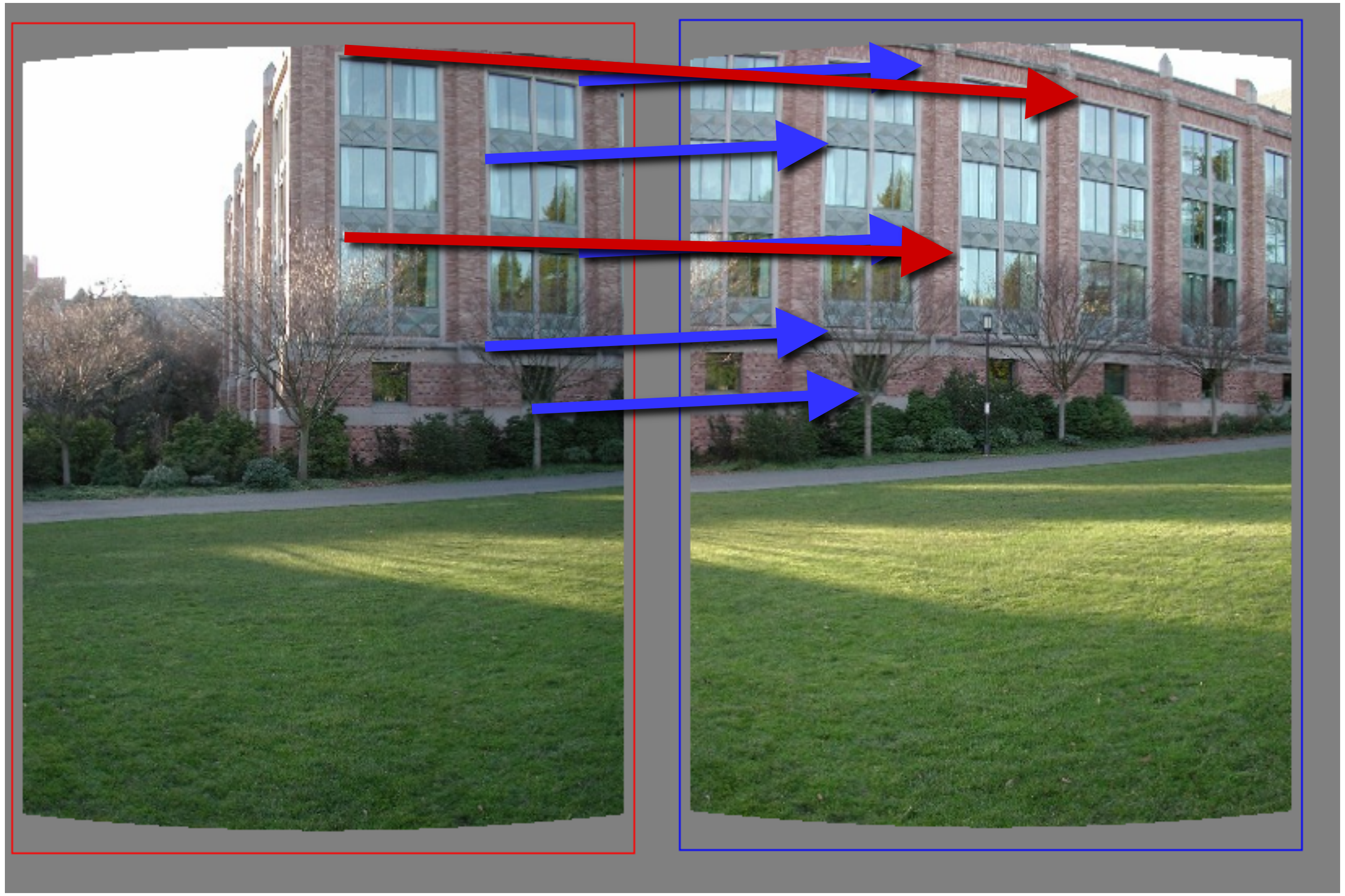
Repeat 1-3 until the best model is found with high confidence

RANSAC

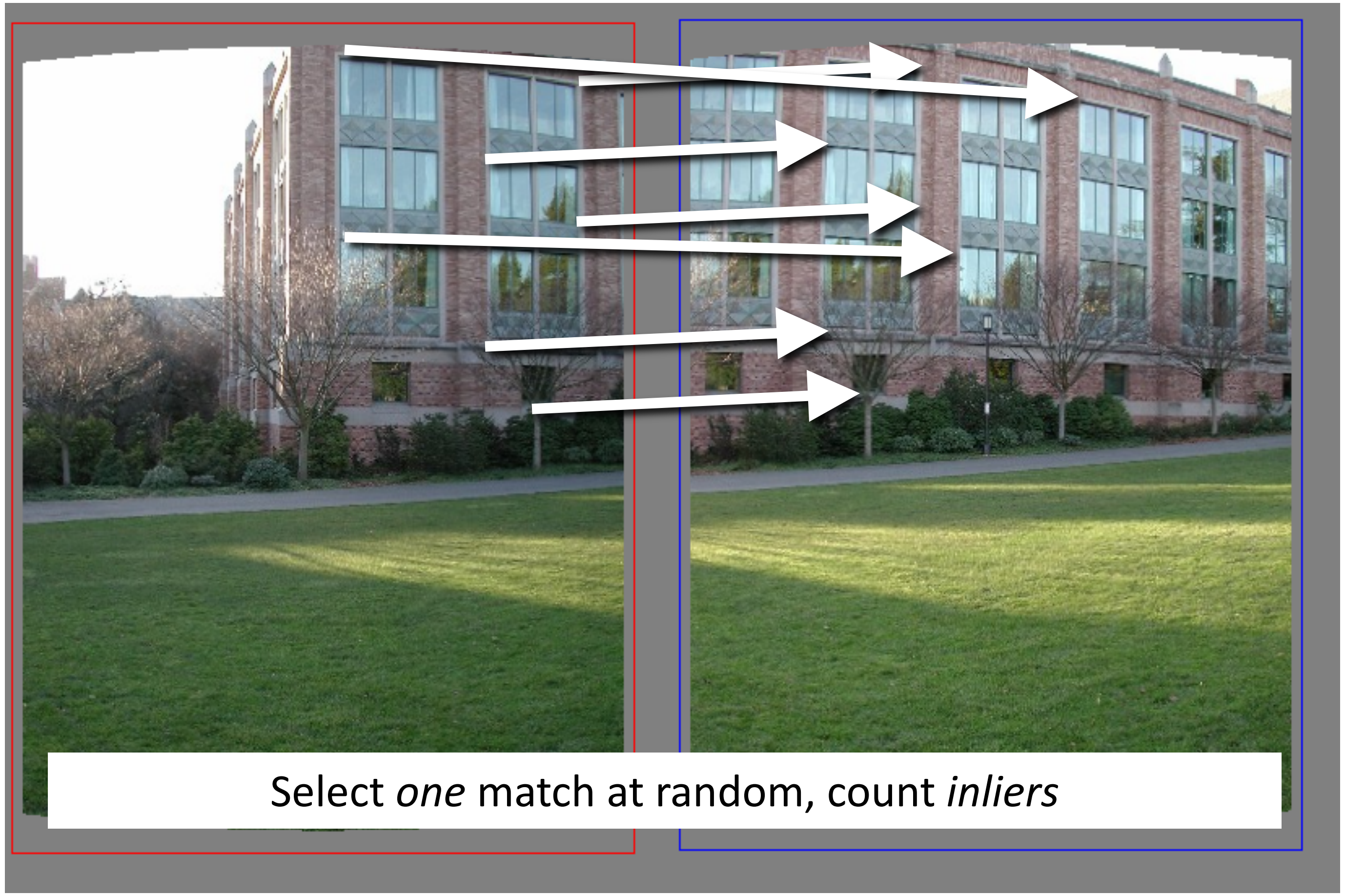
- Idea:
  - All the inliers will agree with each other on the translation vector; the (hopefully small) number of outliers will (hopefully) disagree with each other
    - RANSAC only has guarantees if there are < 50% outliers
  - "All good marriages are alike; every bad marriage is bad in its own way."

– Tolstoy via Alyosha Efros

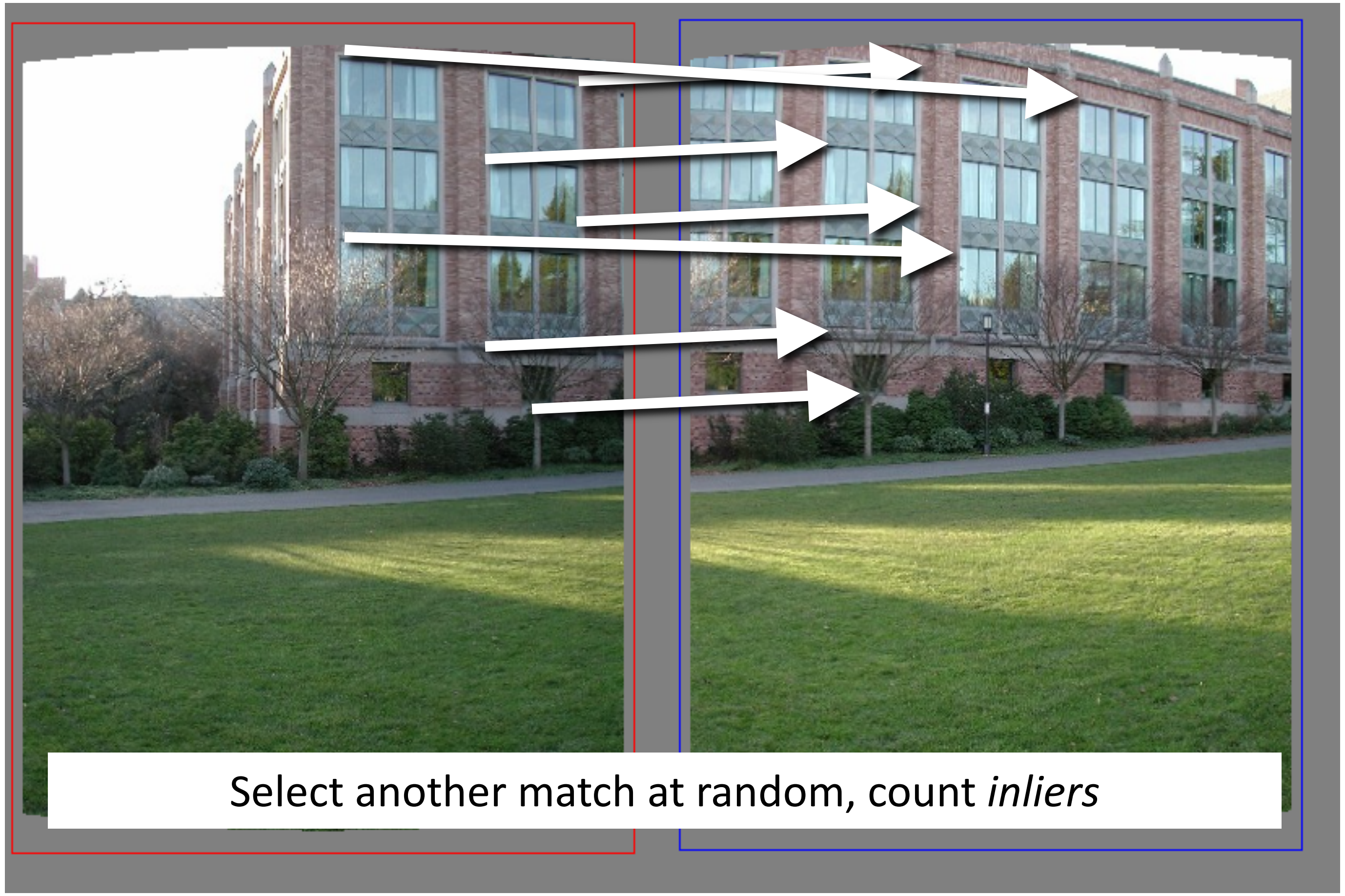
Translations



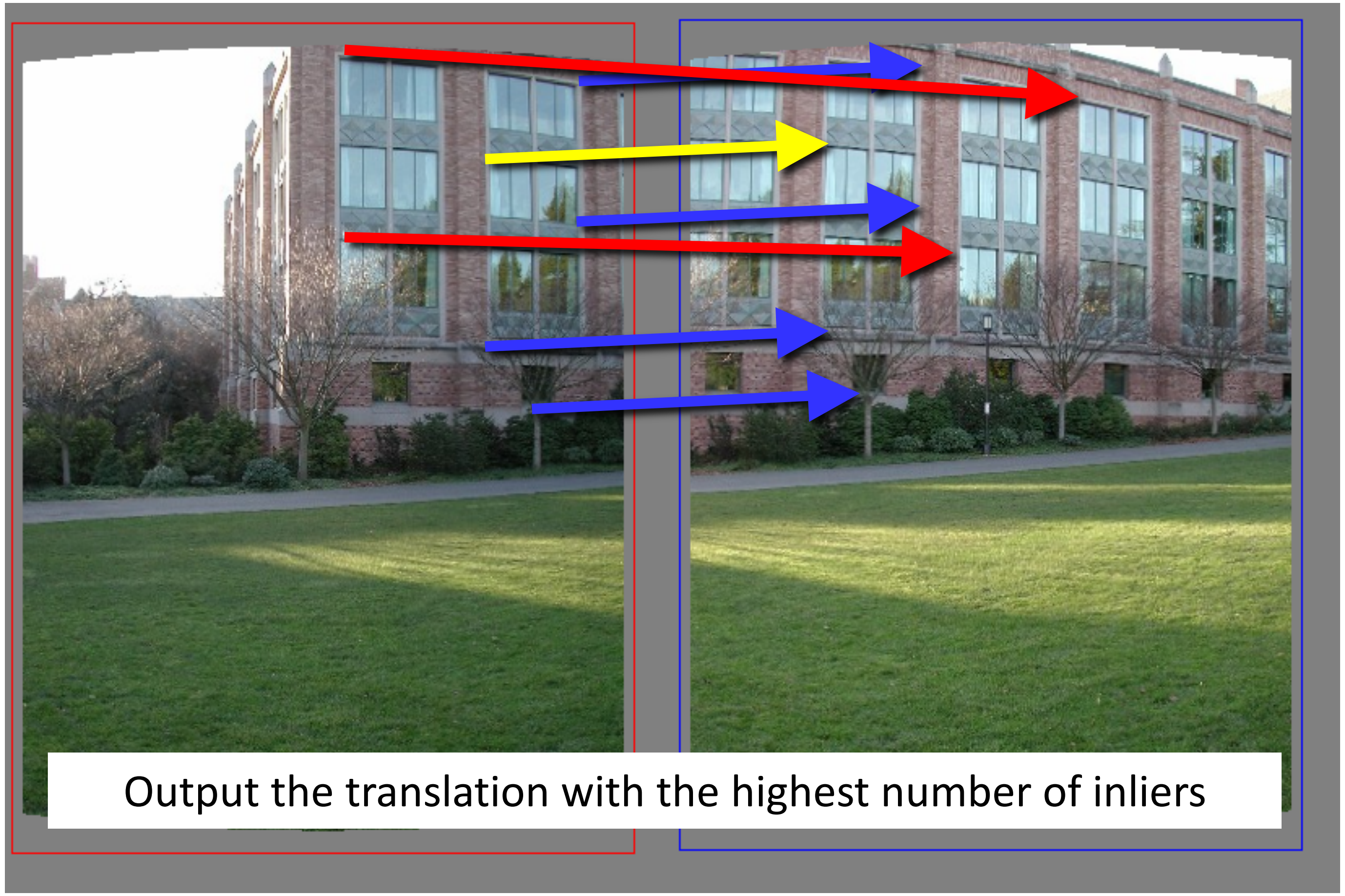
<u>RA</u>ndom <u>SA</u>mple <u>C</u>onsensus



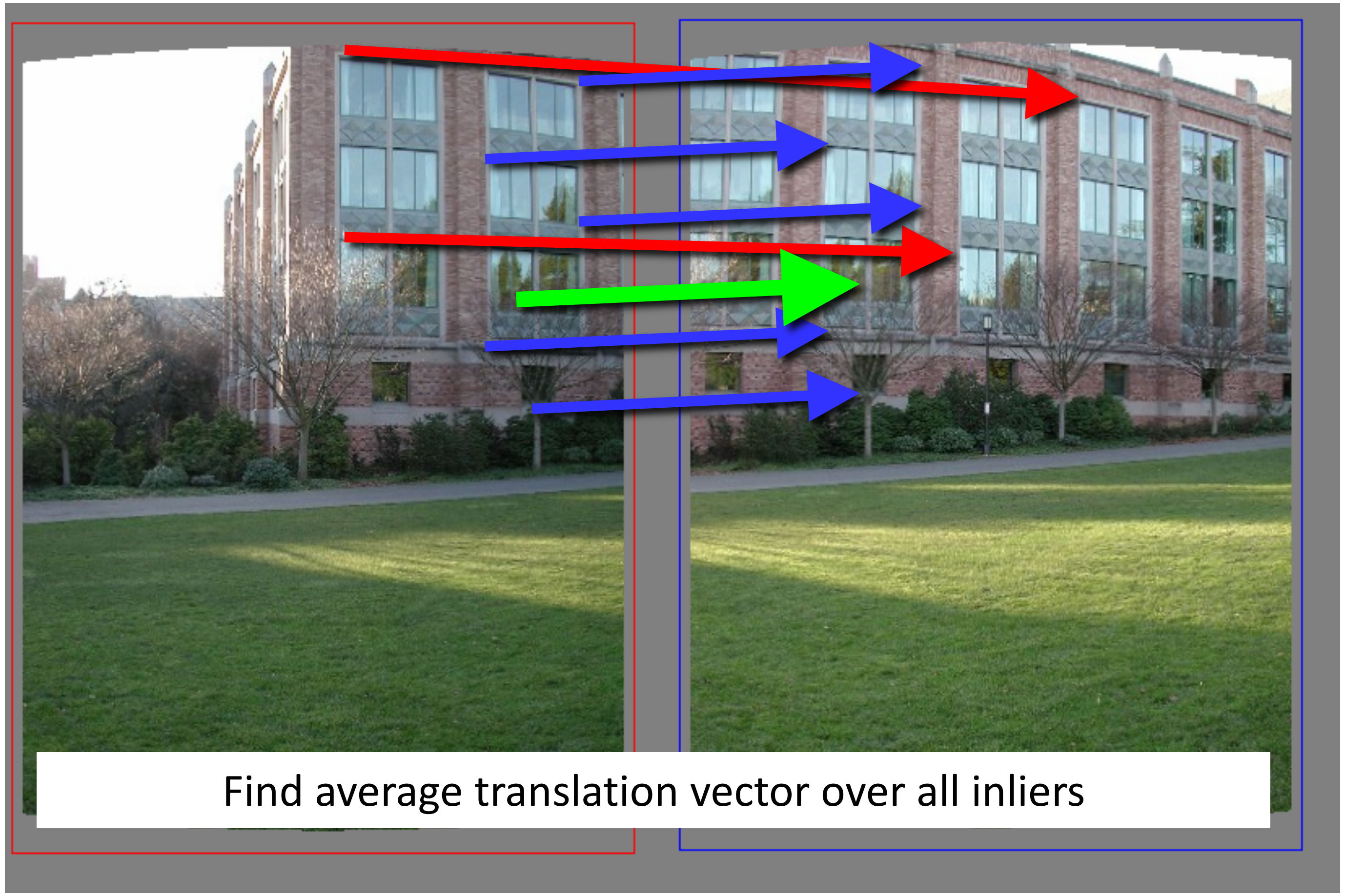
RAndom SAmple Consensus



RAndom SAmple Consensus



Final step: least squares fit



RANSAC - hyperparameters

- **Inlier threshold** related to the amount of noise we expect in inliers
  - Often model noise as Gaussian with 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 99% probability
  - How many rounds do we need?

How many rounds?

- If we have to choose *k* samples each time
  - with an inlier ratio p
  - and we want the right answer with probability *P*

	proportion of inliers *p*						
k	95%	90%	80%	75%	70%	60%	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

To ensure that the random sampling has a good chance of finding a true set of inliers, a sufficient number of trials S must be tried. Let p be the probability that any given correspondence is valid and P be the total probability of success after S trials. The likelihood in one trial that all k random samples are inliers is p^k. Therefore, the likelihood that S such trials will all fail is

$$1 - P = (1 - p^k)^S \tag{6.29}$$

and the required minimum number of trials is

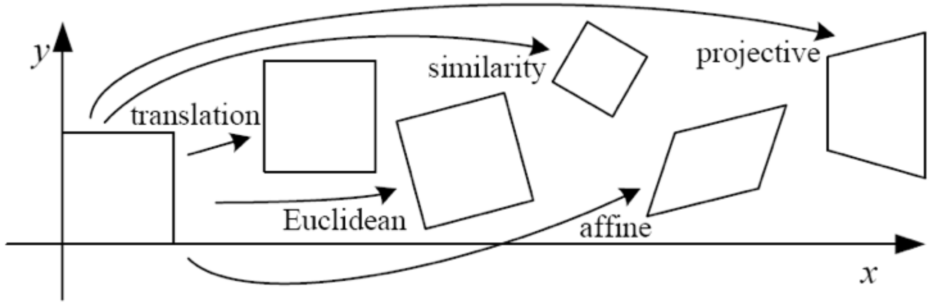
$$S = \frac{\log(1 - P)}{\log(1 - p^k)}. \tag{6.30}$$

	proportion of inliers *p*						
k	95%	90%	80%	75%	70%	60%	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

$P = 0.99$

How big is *k*?

- 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	$\left[\, \boldsymbol{I} \mid \boldsymbol{t} \,\right]_{2\times 3}$	2	orientation $+\cdots$	▭
rigid (Euclidean)	$\left[\, \boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	3	lengths $+\cdots$	◇
similarity	$\left[\, s\boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	4	angles $+\cdots$	◇
affine	$\left[\, \boldsymbol{A} \,\right]_{2\times 3}$	6	parallelism $+\cdots$	▱
projective	$\left[\, \tilde{\boldsymbol{H}} \,\right]_{3\times 3}$	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

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…

RANSAC - Setup

- Given
  - A dataset $D = \{ p_1, p_2, \ldots, p_N\}$
    - Example 1: Line fitting: $\{ (x_1, y_1), \ldots, (x_n, y_n)\}$
    - Example 2: Homography fitting: $\{ (\vec{Q}_1, \overrightarrow{q_1}), (\overrightarrow{Q_2}, \overrightarrow{q_2}), \ldots, (\overrightarrow{Q_N}, \overrightarrow{q_N})\}$
  - A set of parameters θ that need to be fitted
    - Line fitting: $\theta = (m, b)$
    - Homography estimation $\theta = H, ||h|| = 1$
  - A cost function $C(p, \theta)$
    - Line fitting: $C\big((x, y), (m, b)\big) = ||y - (mx + b)||^2$
    - Homography estimation $C\left((\vec{Q}, \vec{q}), H\right) = E(H)$ (Reprojection error)
  - A minimum number needed k
    - Line fitting: 2
    - Homography estimation: 4

RANSAC - Setup

- Given
  - A dataset $D = \{ p_1, p_2, \ldots, p_N\}$
  - A set of parameters θ that need to be fitted
  - A cost function $C(p, \theta)$
  - k
  - $\theta^* = \min_{\theta} \sum_i C(p_i, \theta)$?
  - Problem: outliers

RANSAC - Algorithm

- Given: $D = \{ p_1, p_2, \dots, p_N\}, C(\theta, p)$,k
- $\theta_{best} \leftarrow \phi, D_{inlier} \leftarrow \phi$
- For i = 1, …, S
  - Sample k points
  - Minimize C for these k points to get θ_{hyp}
  - Compute the set of inliers: $D_{hyp} = \{p \in D: C\left(\theta_{hyp}, p\right) < \delta\}$
  - If size of $D_{hyp}$is more than size of D_{inlier}
    - $\theta_{best} \leftarrow \theta_{hyp}$
    - $D_{inlier} \leftarrow D_{hyp}$
- Minimize θ over D_{inlier}

RANSAC: how many iterations do we need?

- p = inlier fraction
- k = minimum number of data points
- S = iter
- $P = \left(1 - \left(1 - p^k\right)^S\right)$