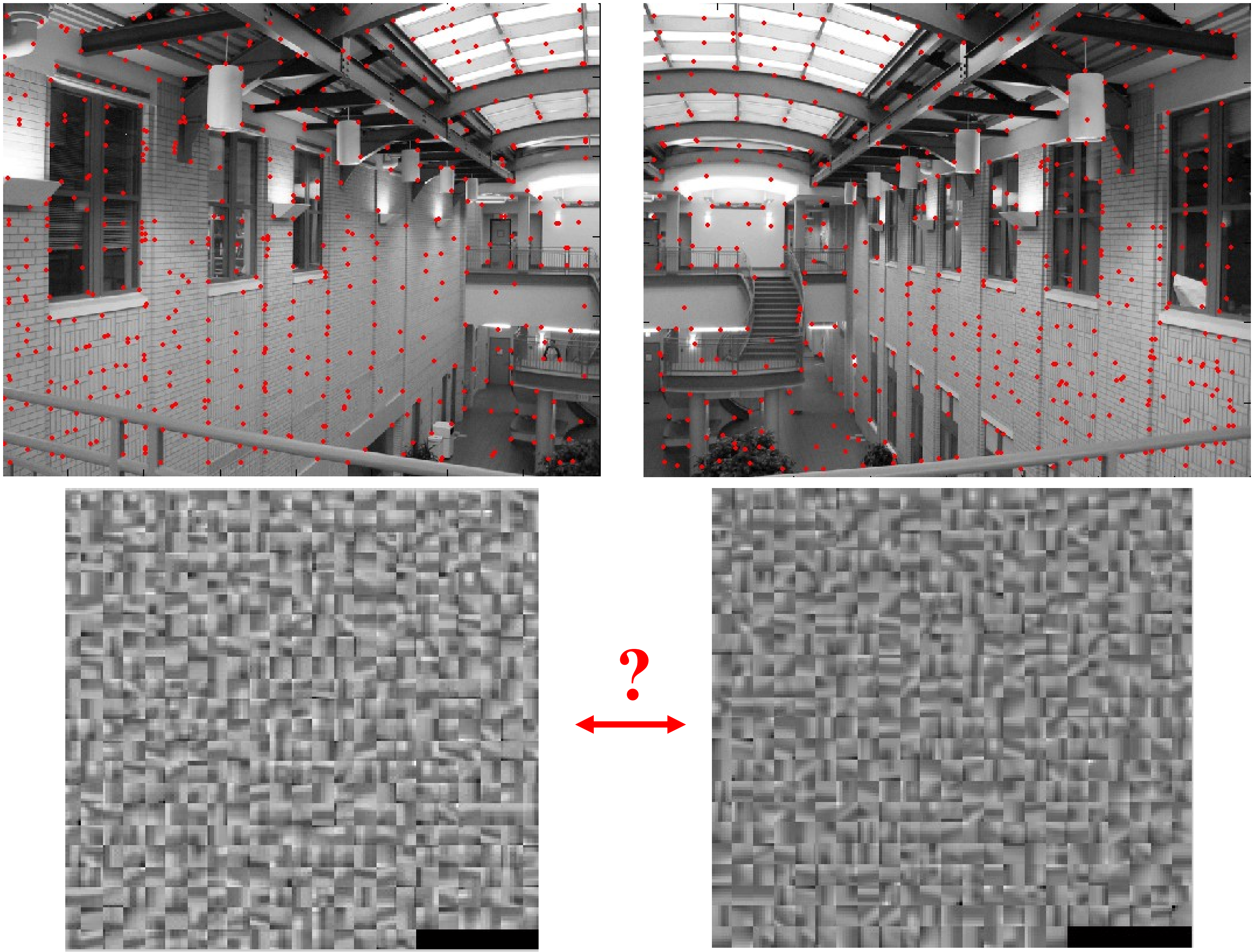
Feature Point Matching

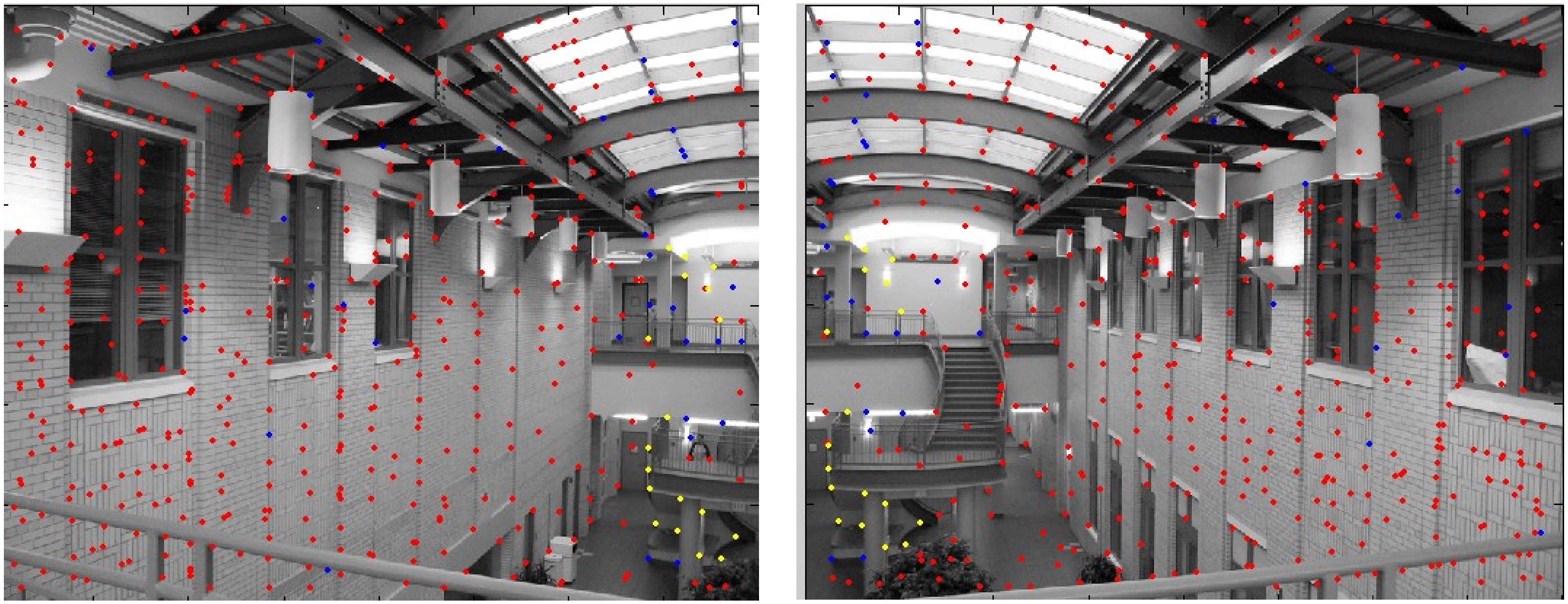
Assumptions:

- Given two images, we computed feature points & their descriptors in both images
- For each feature point in each image we computed the ***k*** best <u>good</u> matches in the other image (***k***=1,2..)   (e.g. exhaustive search)
- We selected candidate matching pairs. E.g. two points are a matched pair only if each is in the top ***k*** matches of the other point.

Computing Feature Points & Descriptors



Matching Pairs of Feature Points



Yellow – Correct matches

Blue – Wrong matches

Red – No match

Goal: Compute Homography using Matches

- How to overcome wrong matches?
- Robust methods needed
- Why is left image is so big?

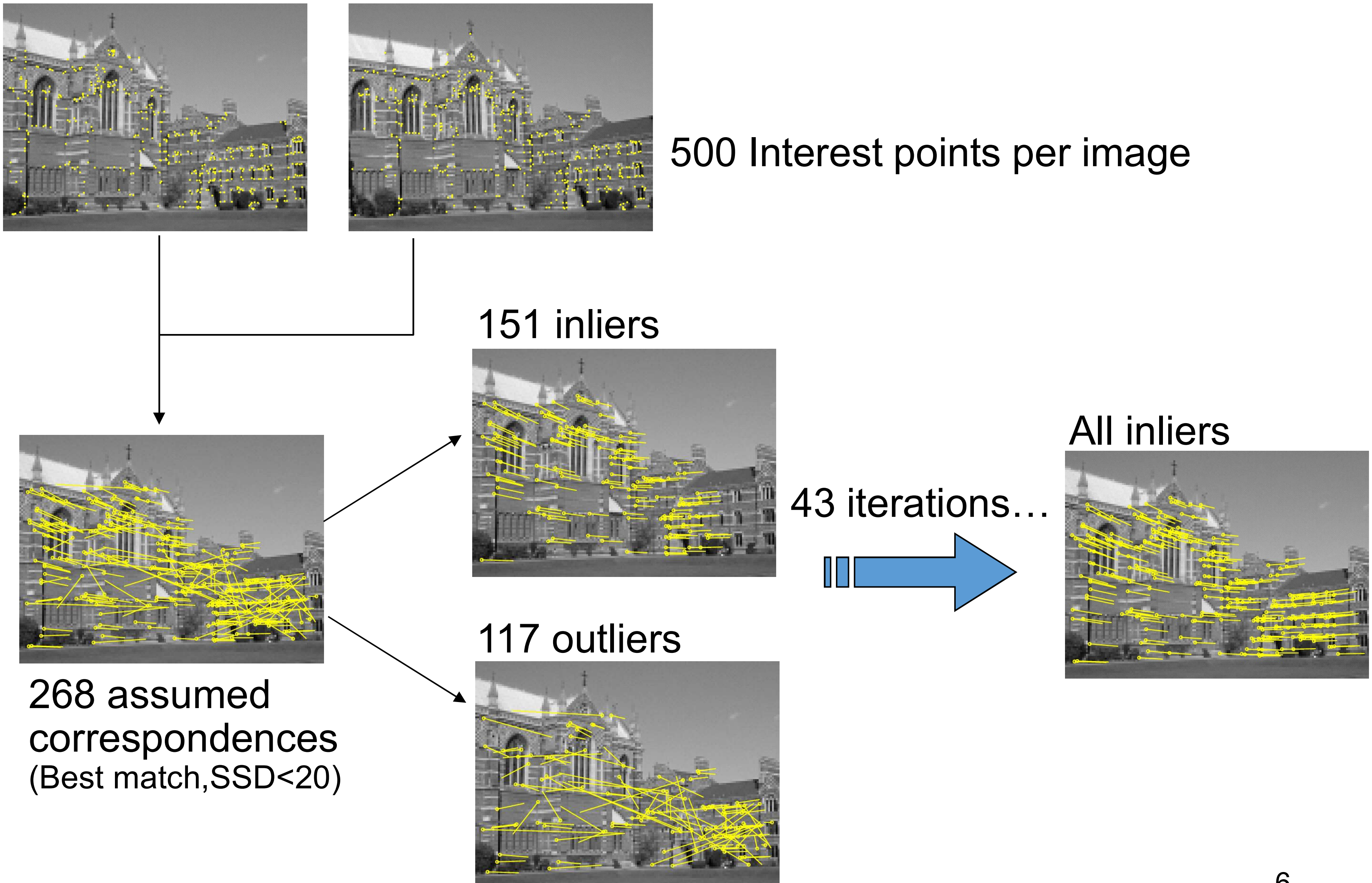
What are Robust Methods?

- The goal of many algorithms is **parametric model fitting**
- Data measured in real images is **inaccurate**
  - Occlusions
  - Noise
  - Ambiguity
  - Wrong feature extraction

outliers

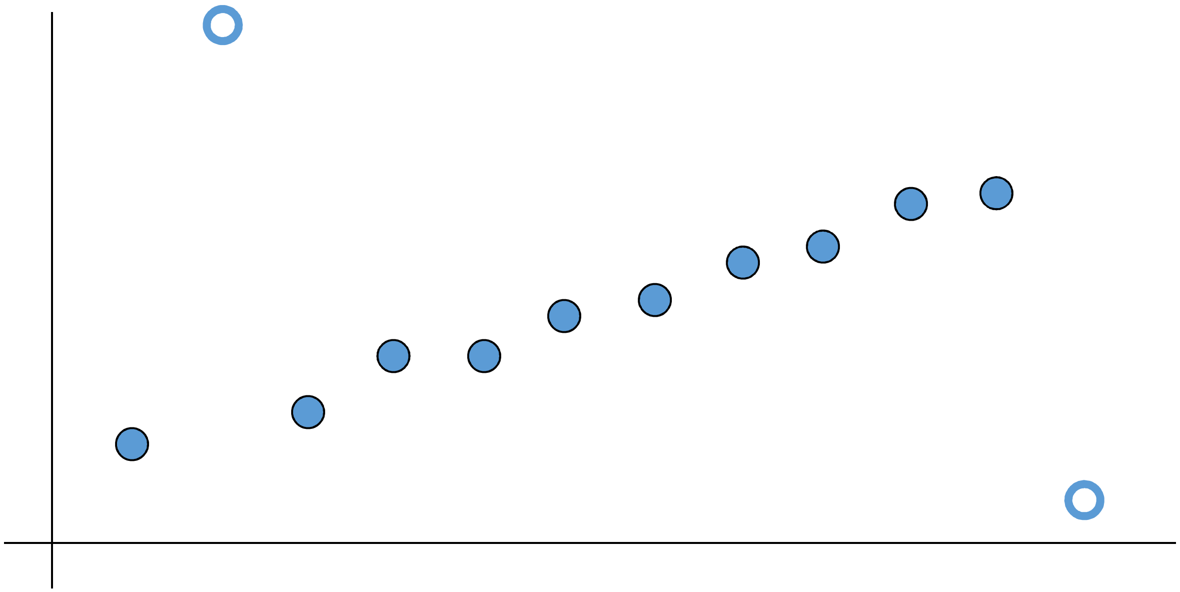
- Goal of robust methods: tolerate outliers

Example: Robust Computation of 2D Homography



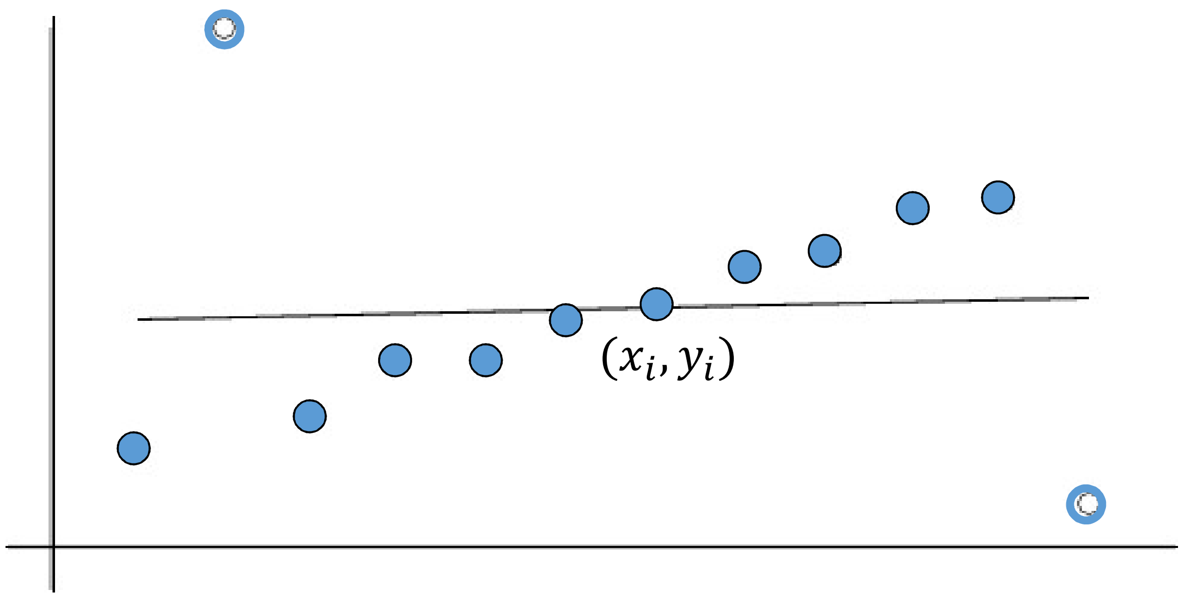
Line Fit Problem

- Straight line fitting to a set of 2D points



Line Fit using Least Squares

- Fit a straight line to a set of 2D points



Least Square error minimization is not robust.

One outlier may cause a large mistake

Least Square Line Fitting

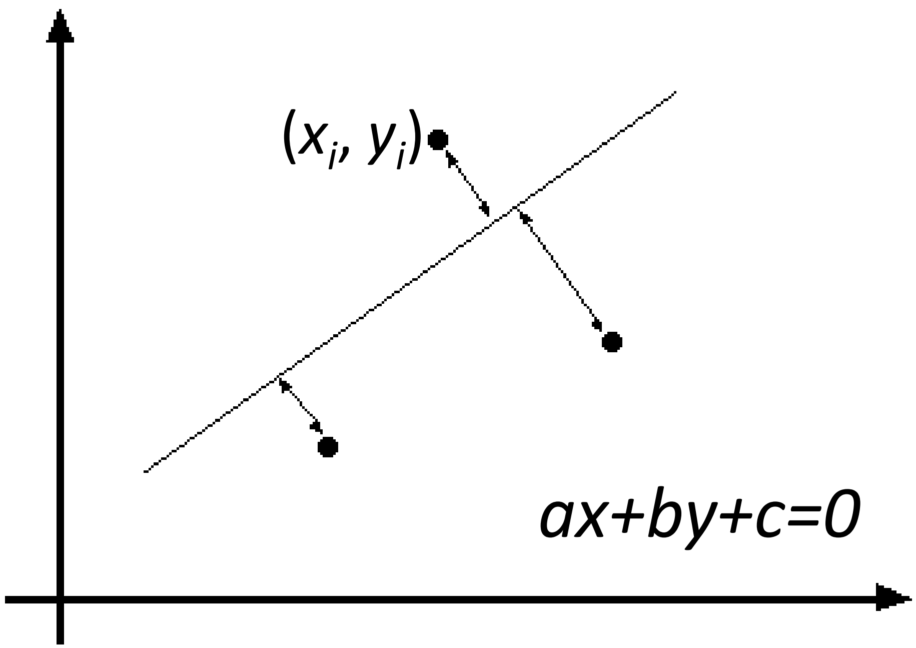
distance from a point (x_i, y_i) to a line $(ax + by + c = 0)$:

$$d = \frac{|ax_i + by_i + c|}{\sqrt{a^2 + b^2}}$$

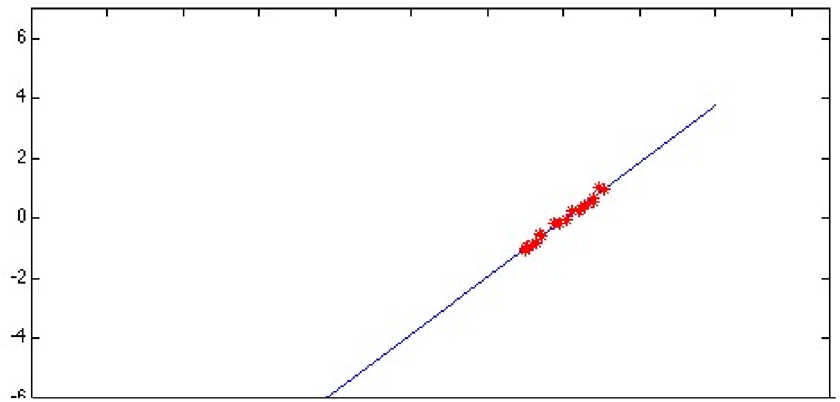
Find (a, b, c) to minimize the sum of squared perpendicular distances

$$E = \sum_{i=1}^{n} (ax_i + by_i + c)^2$$

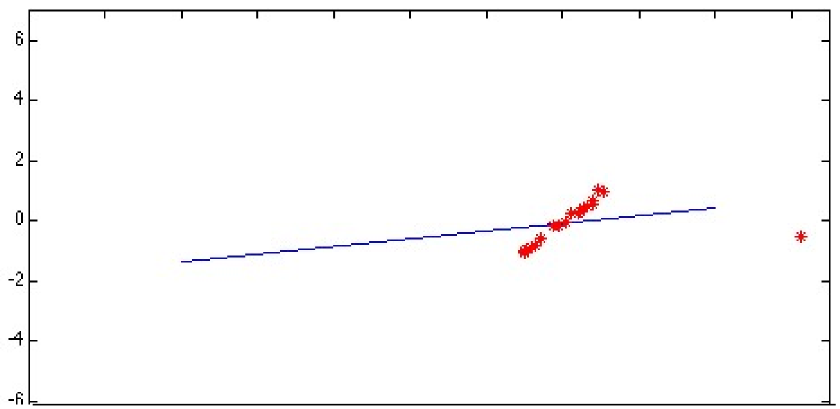
$$s.t. \qquad \|n\| = \sqrt{a^2 + b^2} = 1$$



Least Square Line Fitting Example



LS fit to a clean set of points



One outlier leads to disaster

Least Squares Conclusions

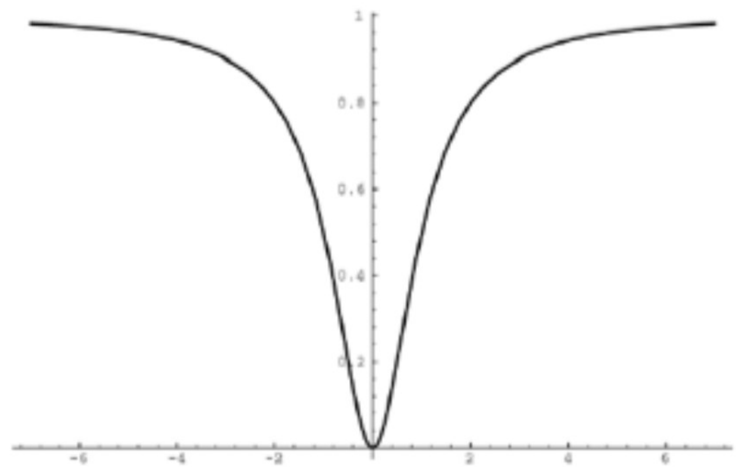
- Good
  - Very clear objective function
  - Easy optimization
- Bad
  - Sensitive to Outliers
  - Can not find multiple matches

Robust Norms
(From Distance to Cost)

- Squared Error $c(d) = d^2$

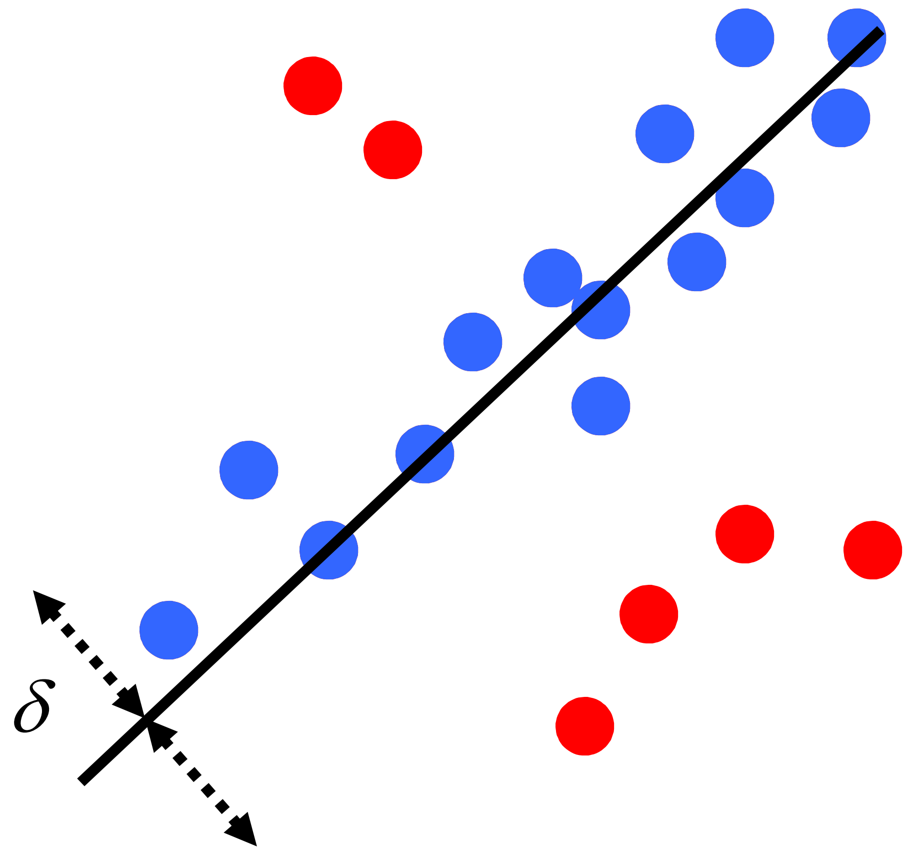
- Absolute Value $c(d) = |d|$
  - Hard to minimize

- Robust $c(d,e) = d^2/(e^2+d^2)$
  - Small d: square error
  - Large d: maximum 1



RANSAC (1981)
(RANdom SAmple Consensus)

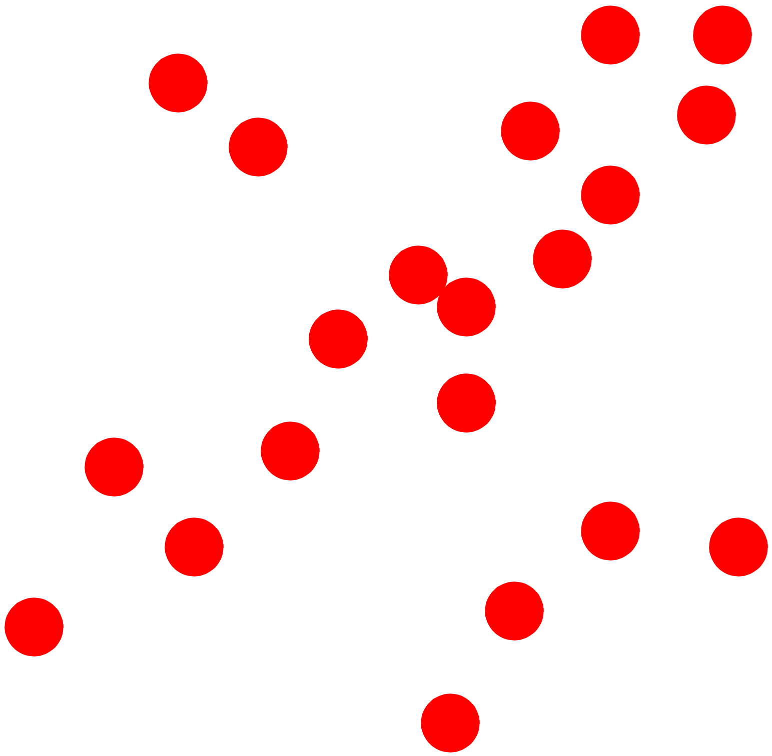
Estimate parameters of a model by random sampling of observed data



RANSAC Algorithm

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

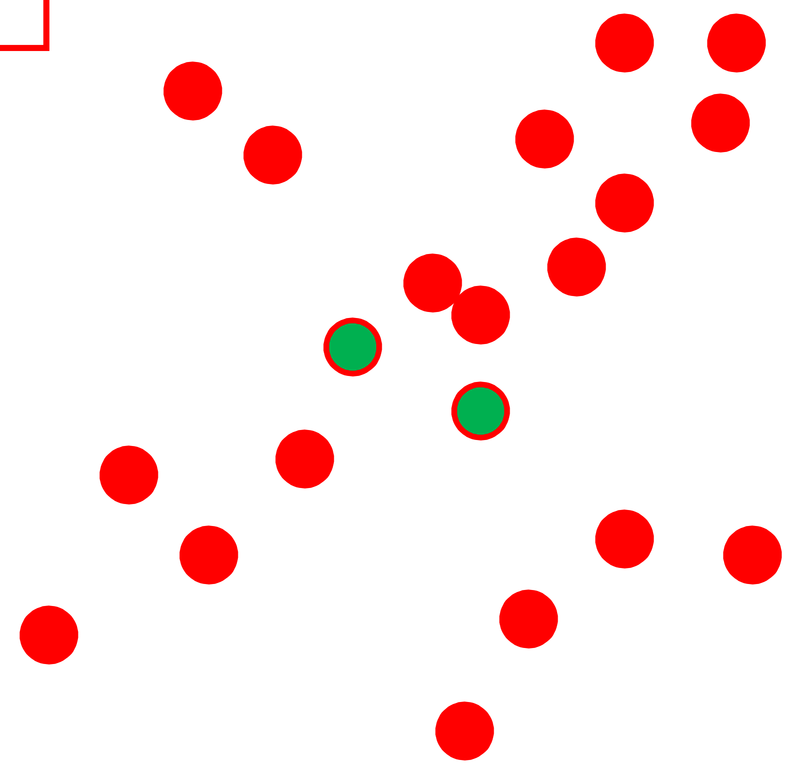
Repeat 1-3 until the best model is found with high confidence



RANSAC Line fitting example

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

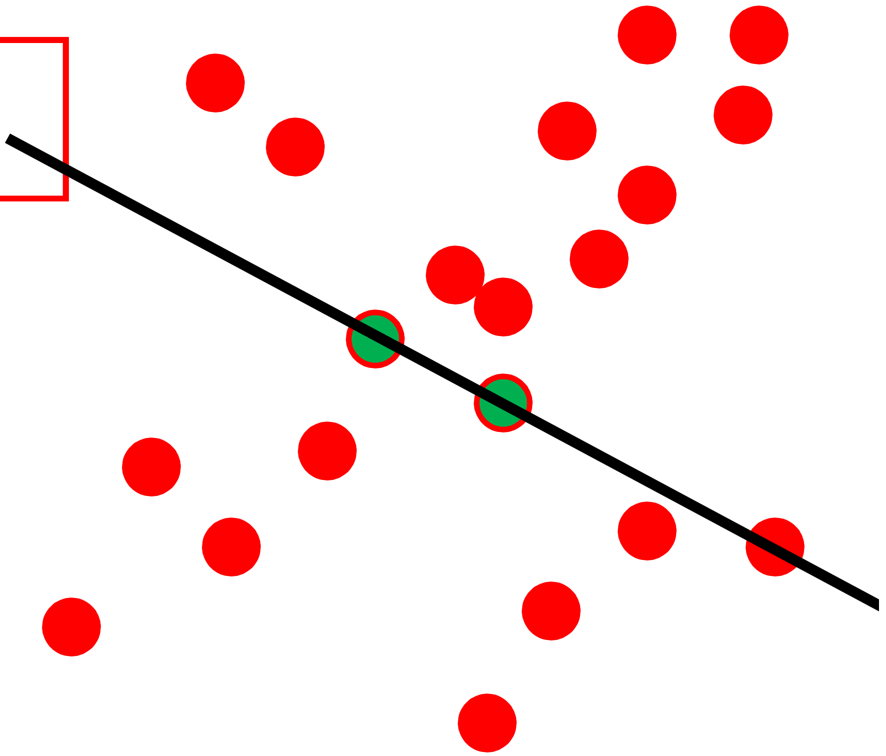
Repeat 1-3 until the best model is found with high confidence



RANSAC:: line fitting example

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

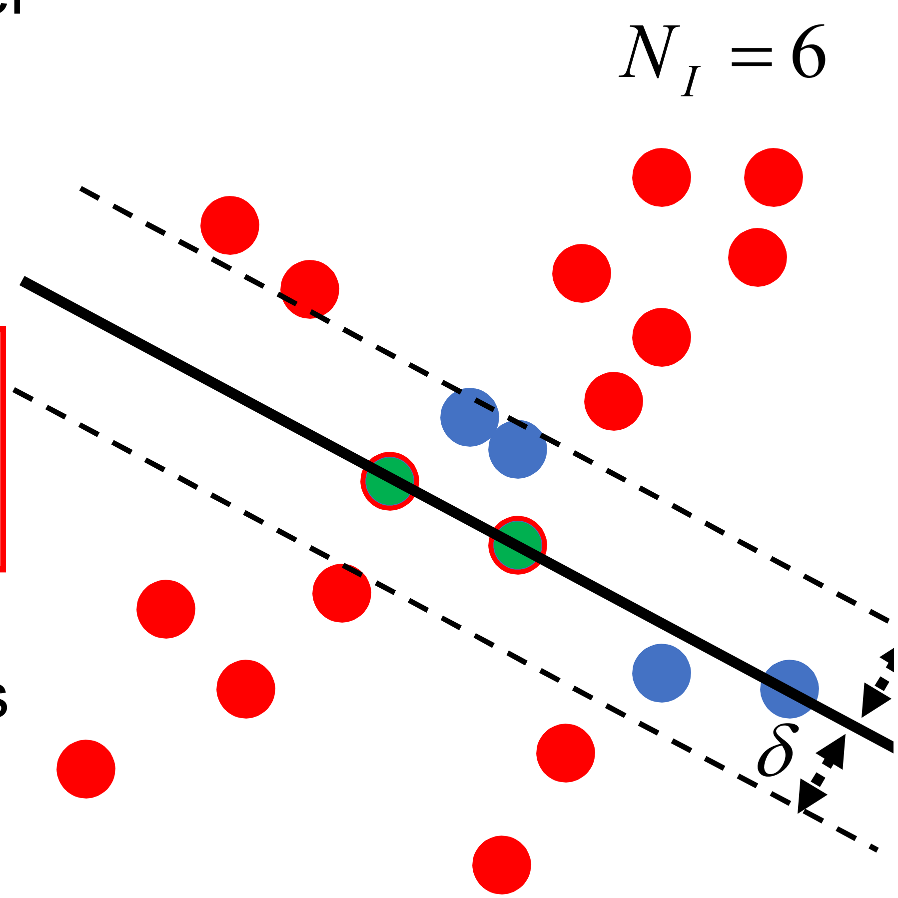
Repeat 1-3 until the best model is found with high confidence



RANSAC   Line fitting example

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

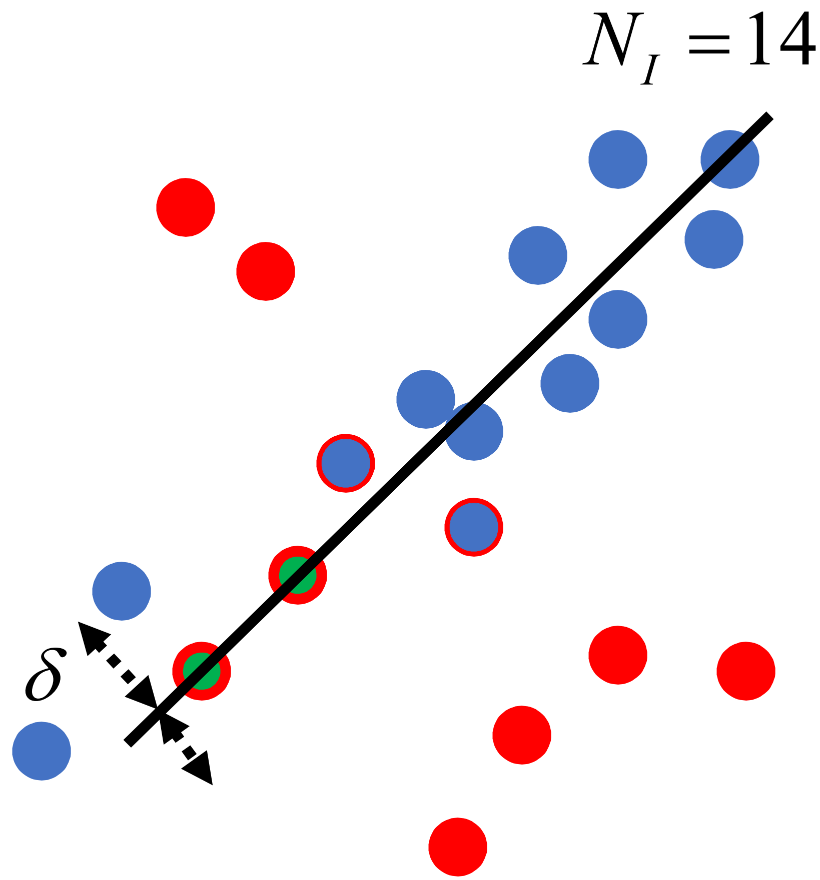
Repeat 1-3 until the best model is found with high confidence



RANSAC Line fitting example

1. **Sample** (randomly) the number of points required to fit the model
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 Parameter Selection

$\boldsymbol{s}$: Number of points needed to fit the model

$\boldsymbol{\omega}$: Probability of choosing an inlier

$\boldsymbol{\omega^s}$: Probability that all $\boldsymbol{s}$ points are inliers

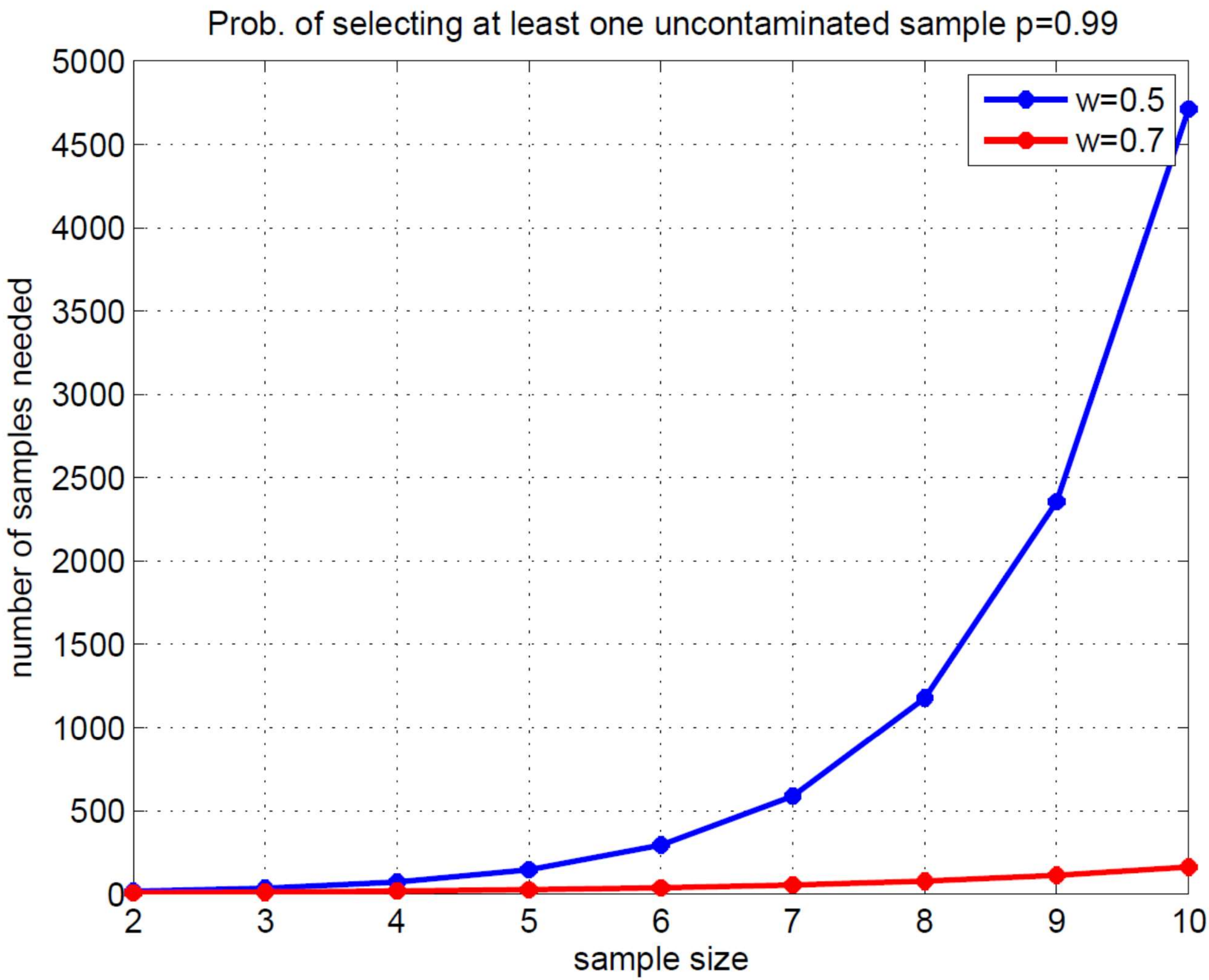
$\mathbf{1}-\boldsymbol{\omega^s}$: Prob. that at least one point is an outlier

$\boldsymbol{p}$: Prob. of choosing $\boldsymbol{s}$ inliers in some iterations

$\mathbf{1}-\boldsymbol{p}=(\mathbf{1}-\boldsymbol{\omega^s})^{\boldsymbol{N}}$: Prob. to never select $\boldsymbol{s}$ inliers

$\boldsymbol{N}$: Number of RANSAC iterations

$$\boxed{N=\log(1-p)/\log(1-\omega^s)}$$


Prob. of selecting at least one uncontaminated sample p=0.99
w=0.5
w=0.7
number of samples needed
sample size
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
2
3
4
5
6
7
8
9
10

Sufficient Number of Iterations (N)

- *p = 0.99*
- w : inliers probability,
  1- w : outlier probability

Sample Size	proportion of outliers $(1\text{-}w)$						
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

RANSAC Conclusions

Good

- Robust to outliers
- Suitable for many model fitting cases
- Easy to implement

Bad

- Computational time grows quickly with fraction of outliers and number of parameters

Back to Homography Estimation

RANSAC loop:

1. Randomly select 4 feature pairs
2. Compute homography ***H*** (exact)
3. Compute *inliers* where $D(p_i', \mathbf{H}p_i) < \varepsilon$
4. Keep largest set of inliers
5. Re-compute ***H*** using least-squares on all inliers in largest set

Adaptively Determining Number of Iterations

w is often unknown a priori. Assume the worst case (e.g. 50%) and adapt during iterations

- *Compute N* using w=50%; *loop_count* =0
- While *N* > *loop_count* repeat:
  - Randomly Sample data, compute model, and count the number of inliers. Update Max[#inliers] Set *w* = (Max[#inliers] / #points)
  - Recompute *N* from w
  - Increment the *loop_count* by 1
- Terminate