SAIR
Spatial AI & Robotics Lab

CSE 473/573-A

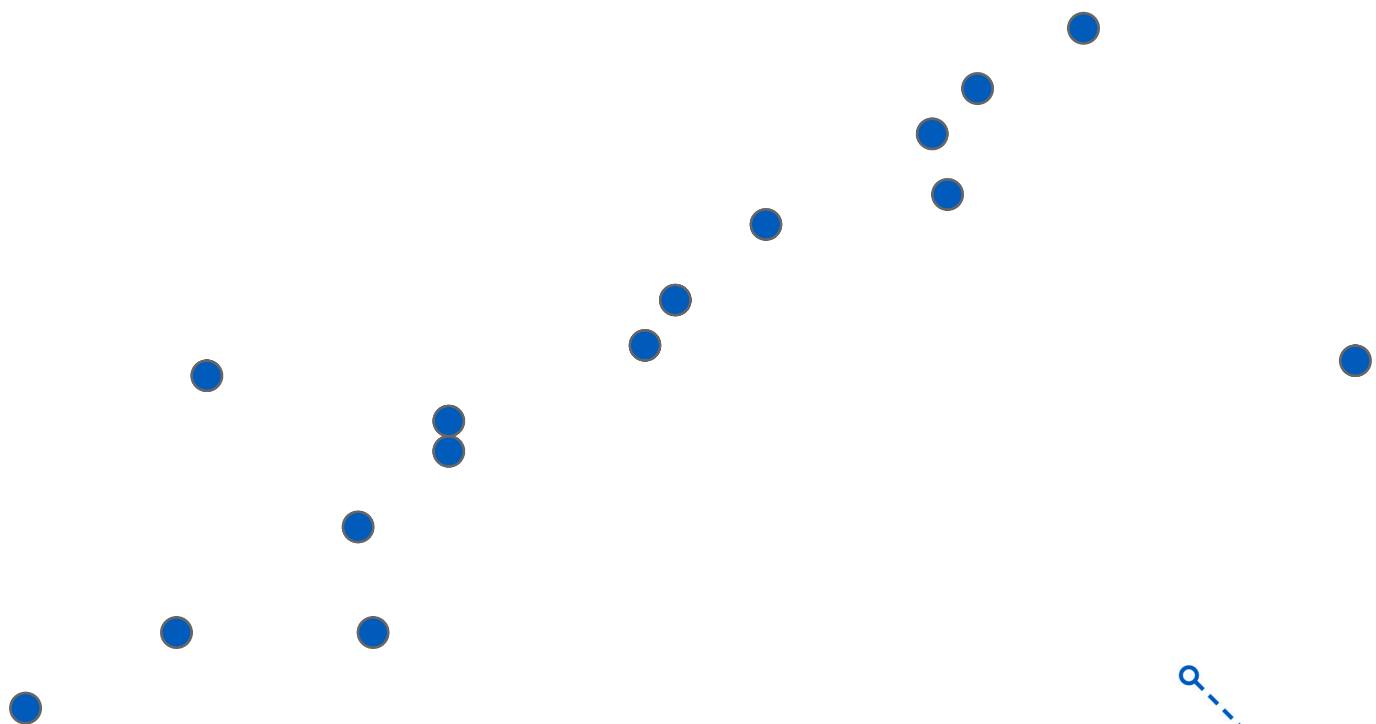
L12: RANSAC

Chen Wang
Spatial AI & Robotics Lab
Department of Computer Science and Engineering

University at Buffalo The State University of New York

Problem: Line Fitting

- Not all data may be representative
- There may be "outliers"
- If you use them, your result may not be accurate



Solutions?

- Least Squares Fit?
  - Closed for solution…
  - Sensitive to outliers

- Hypothesize and Test
  - Try out as many lines as we want
  - Keep the best lines
  - But which are the best?

RANSAC

- **RAN**dom **S**ample **C**onsensus
  - An iterative method for estimating a mathematical model from a data set that contains outliers.

- Motivation: we want to avoid the impact of outliers, so let's look for “inliers”, and use those only.

- Idea: if an outlier is chosen to compute the current model, then the model won't have much support from rest of the points.

RANSAC

RANSAC loop:

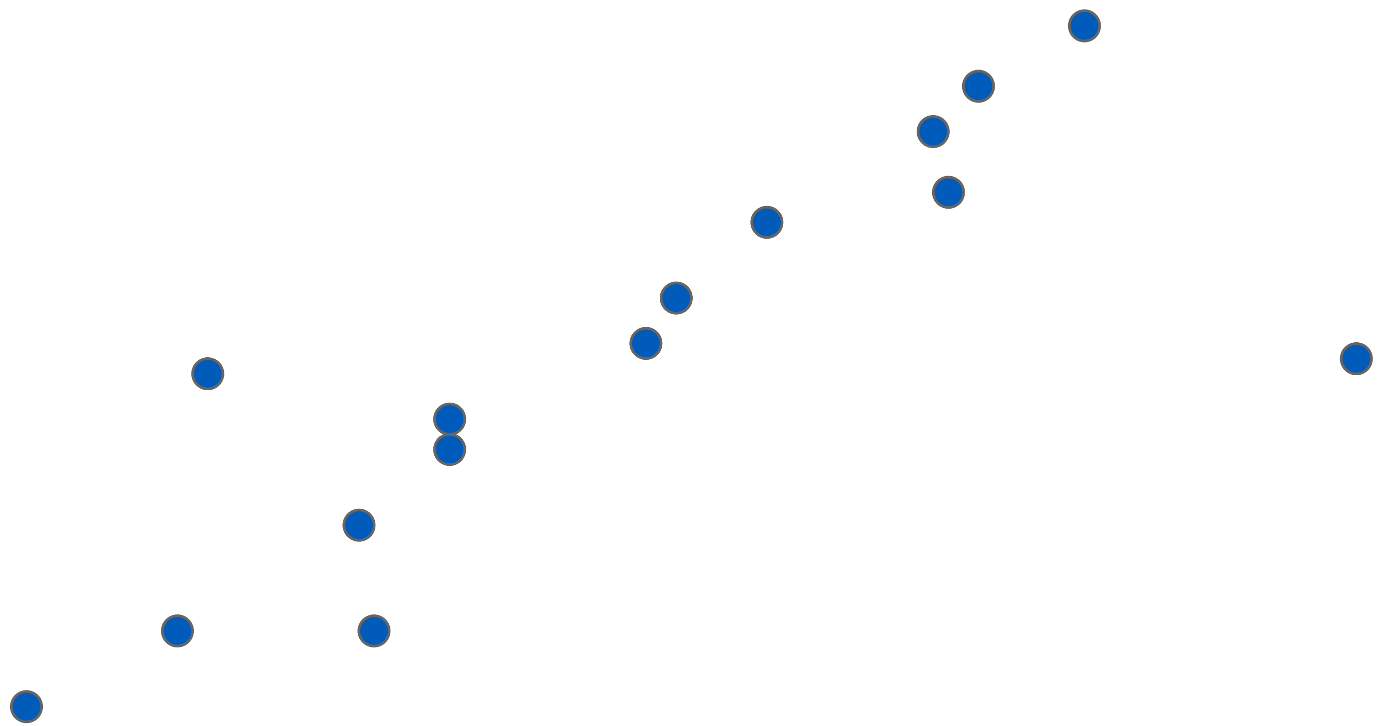
1. Randomly select a **seed group** of points on which to base transformation estimate (e.g., a group of matches)
2. Compute transformation (**model**) from seed group
3. Find **inliers** to this transformation
4. If the number of inliers is sufficiently large, re-compute least-squares estimate on all inliers.

- Keep the model with the largest number of inliers.

RANSAC Line Fitting Example

Task:

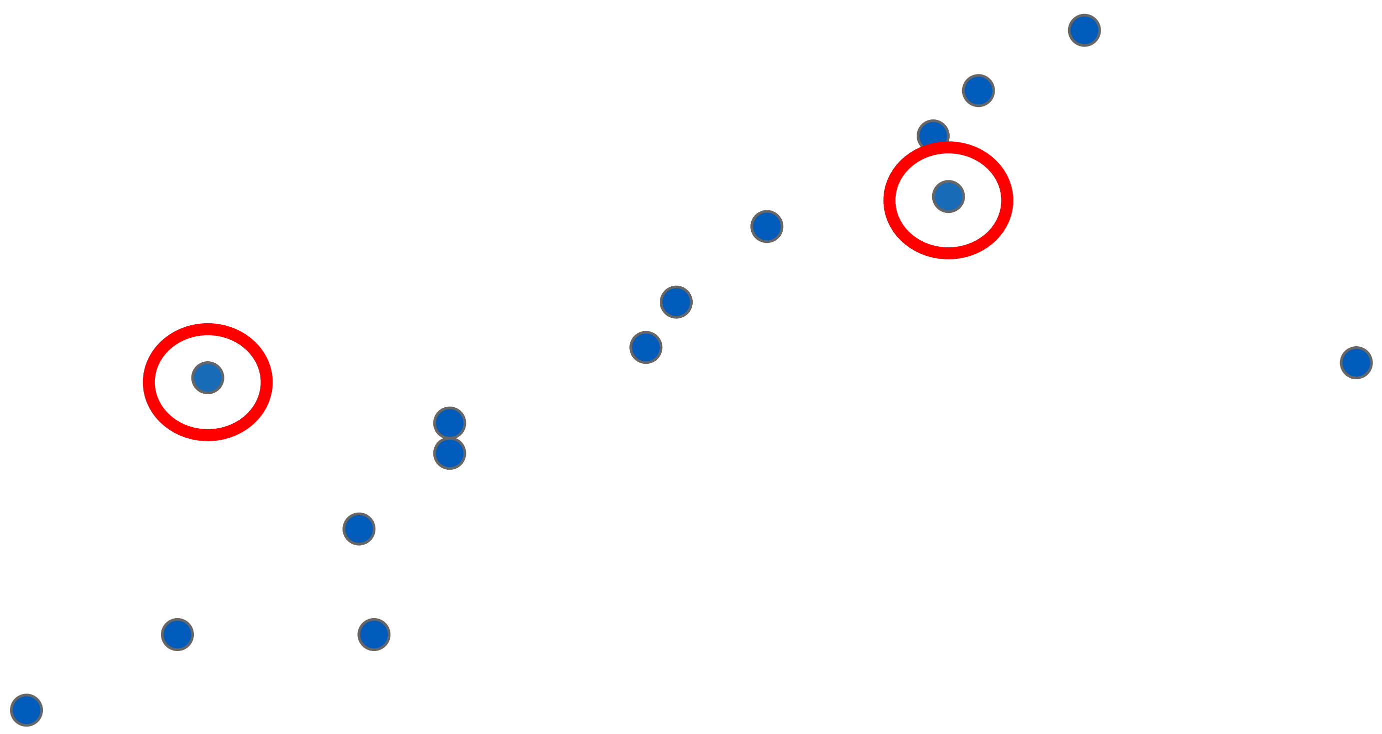
Estimate best line



Slide credit: Jinxiang Chai, CMU

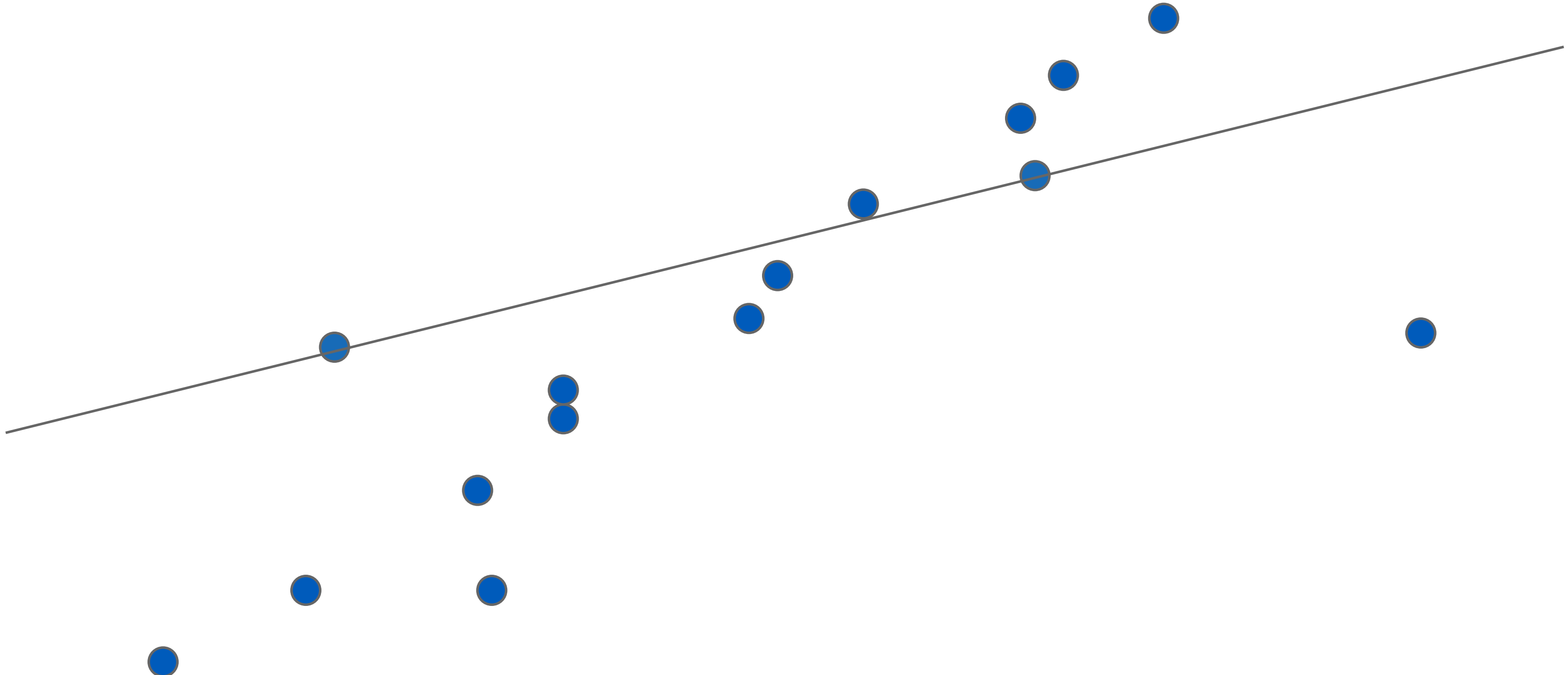
RANSAC Line Fitting Example

Sample two points



RANSAC Line Fitting Example

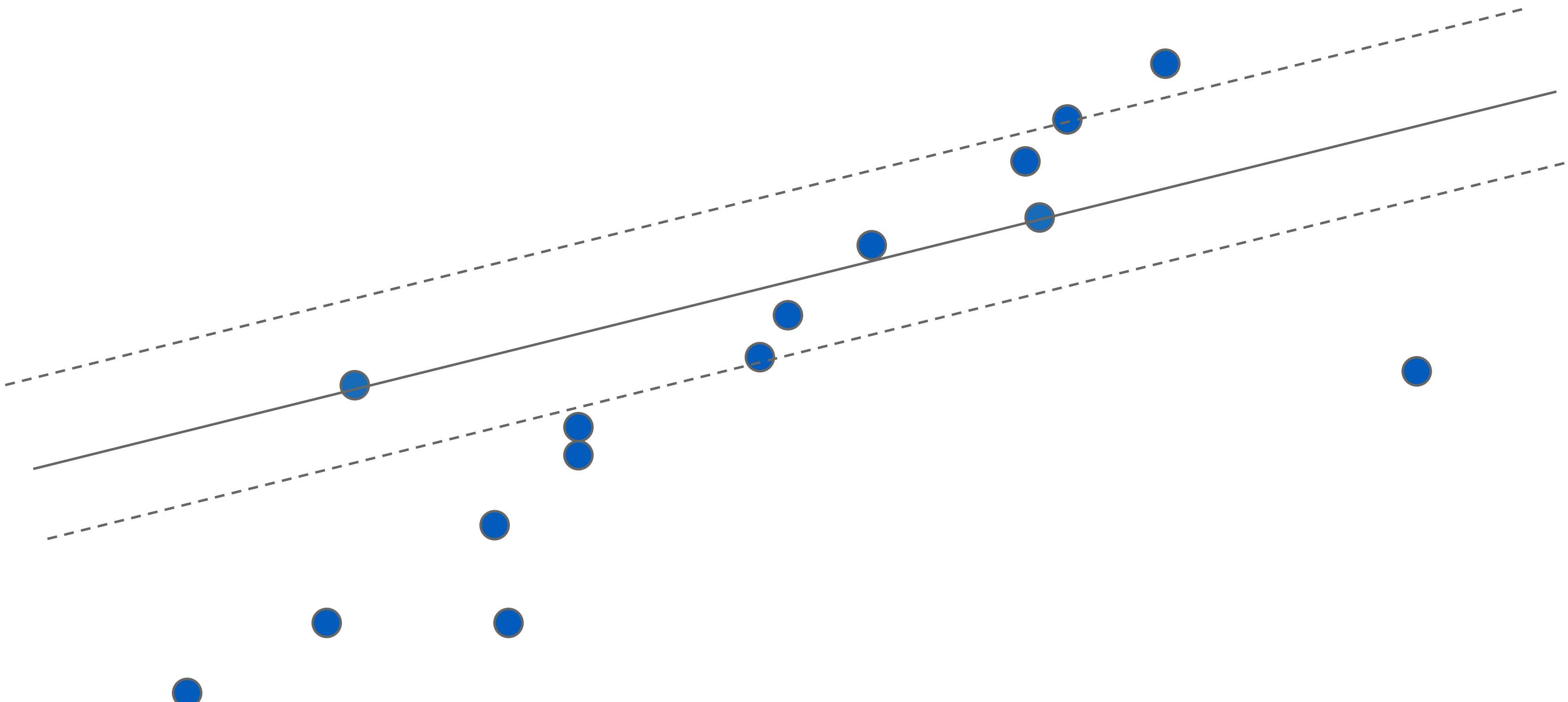
Fit Line



SAIR
Spatial AI & Robotics Lab

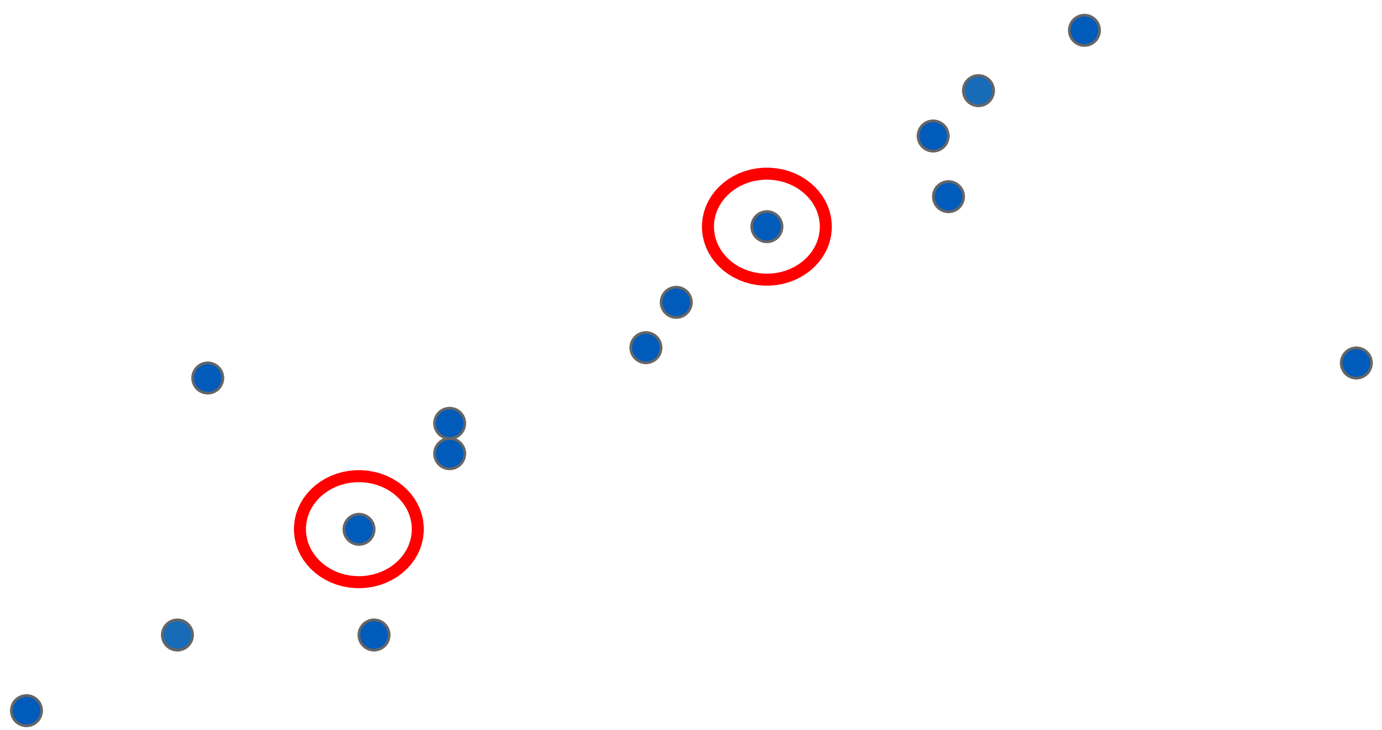
RANSAC Line Fitting Example

Total number of points within a threshold of line.



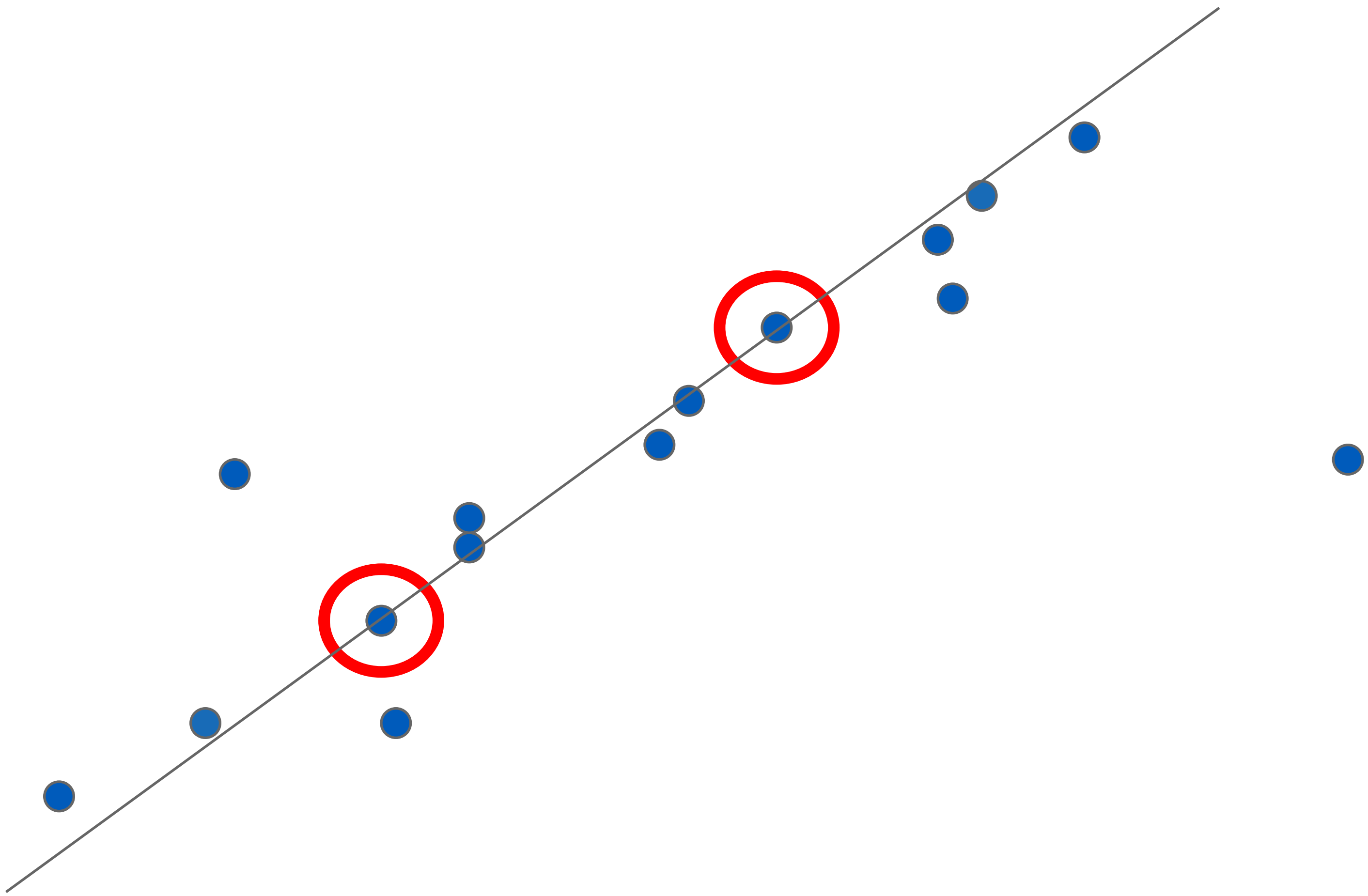
RANSAC Line Fitting Example

Repeat, until get a good result



RANSAC Line Fitting Example

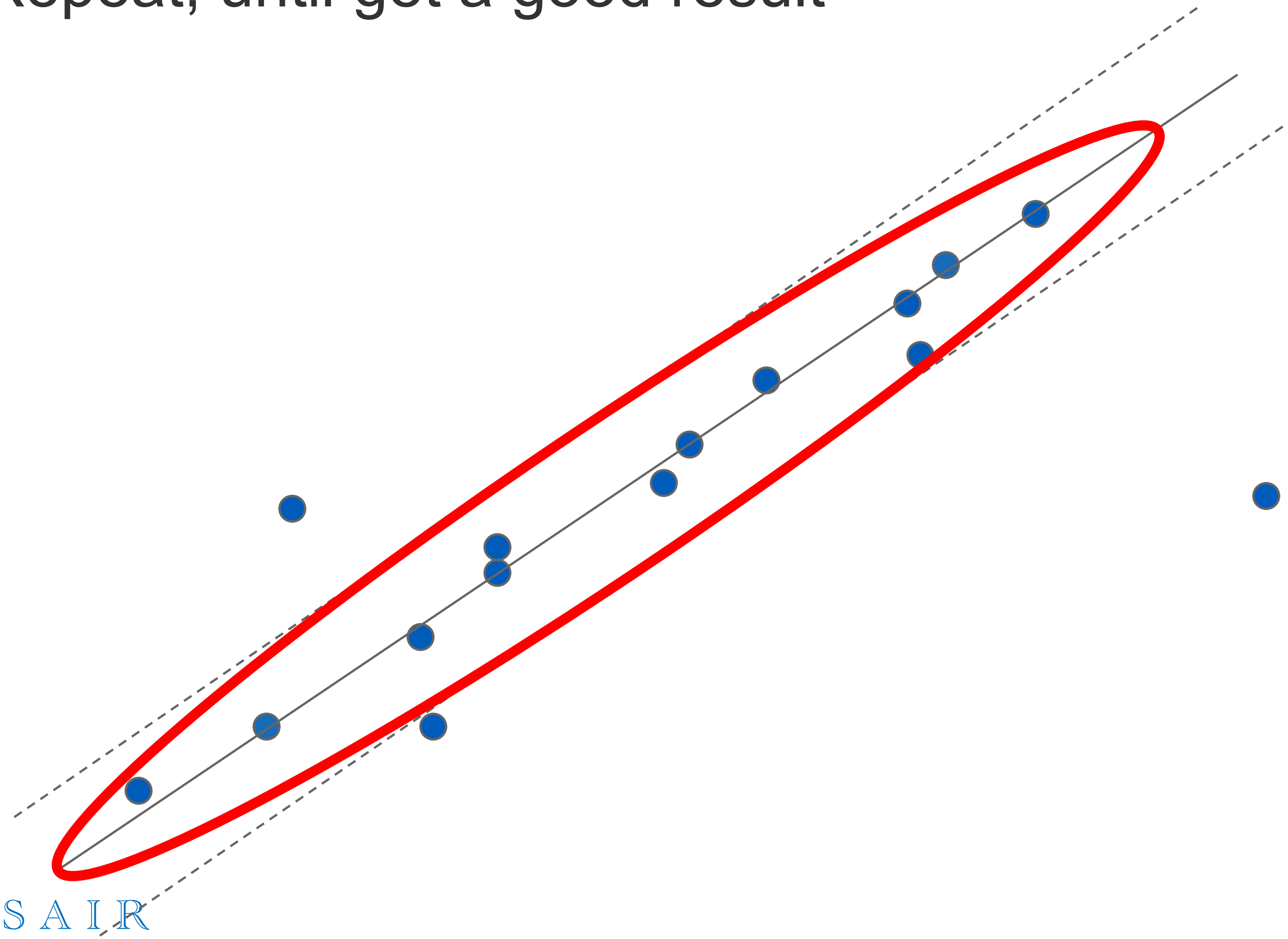
Repeat, until get a good result



SAIR
Spatial AI & Robotics Lab

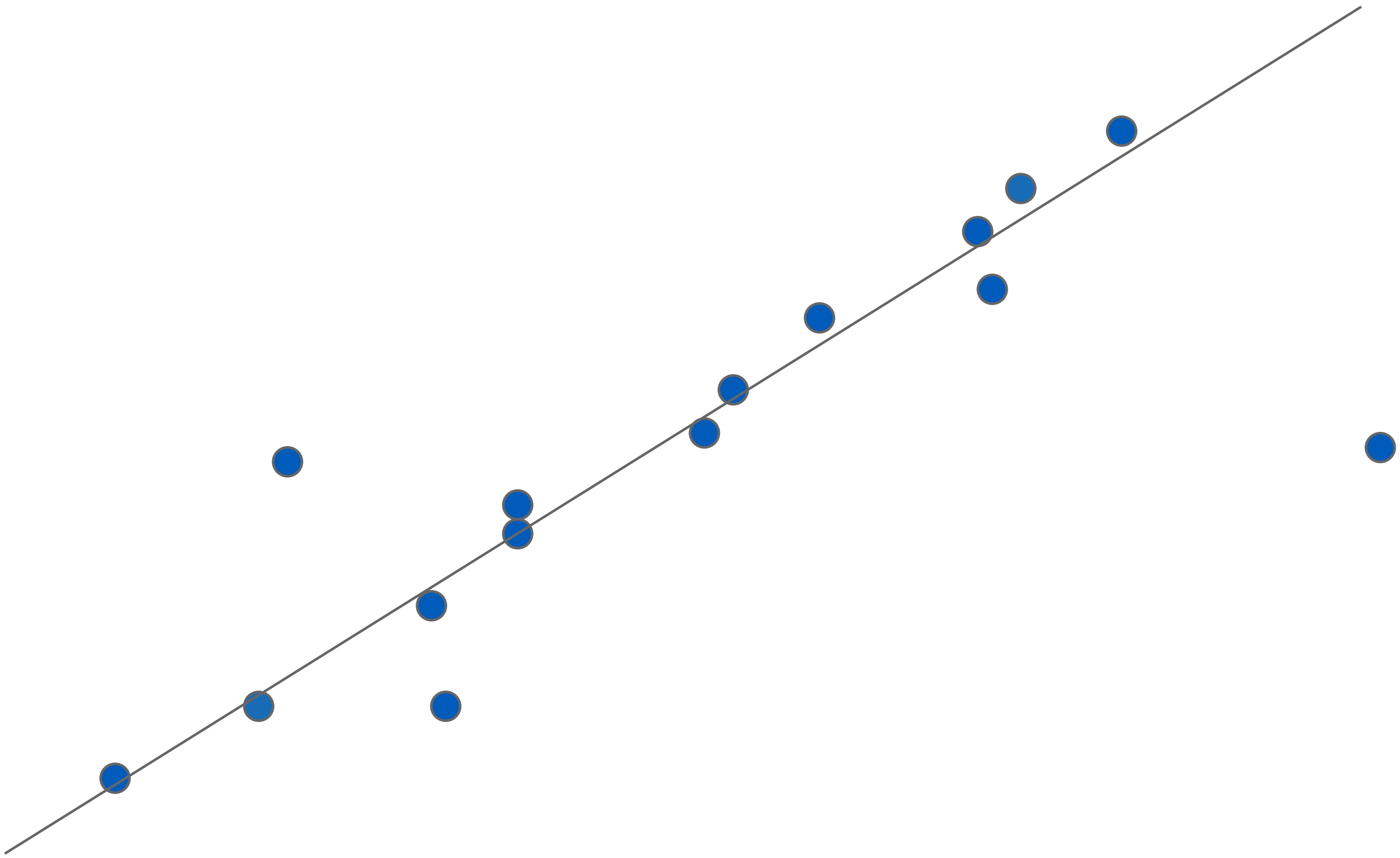
RANSAC Line Fitting Example

Repeat, until get a good result



RANSAC Line Fitting Example

Repeat, until get a good result



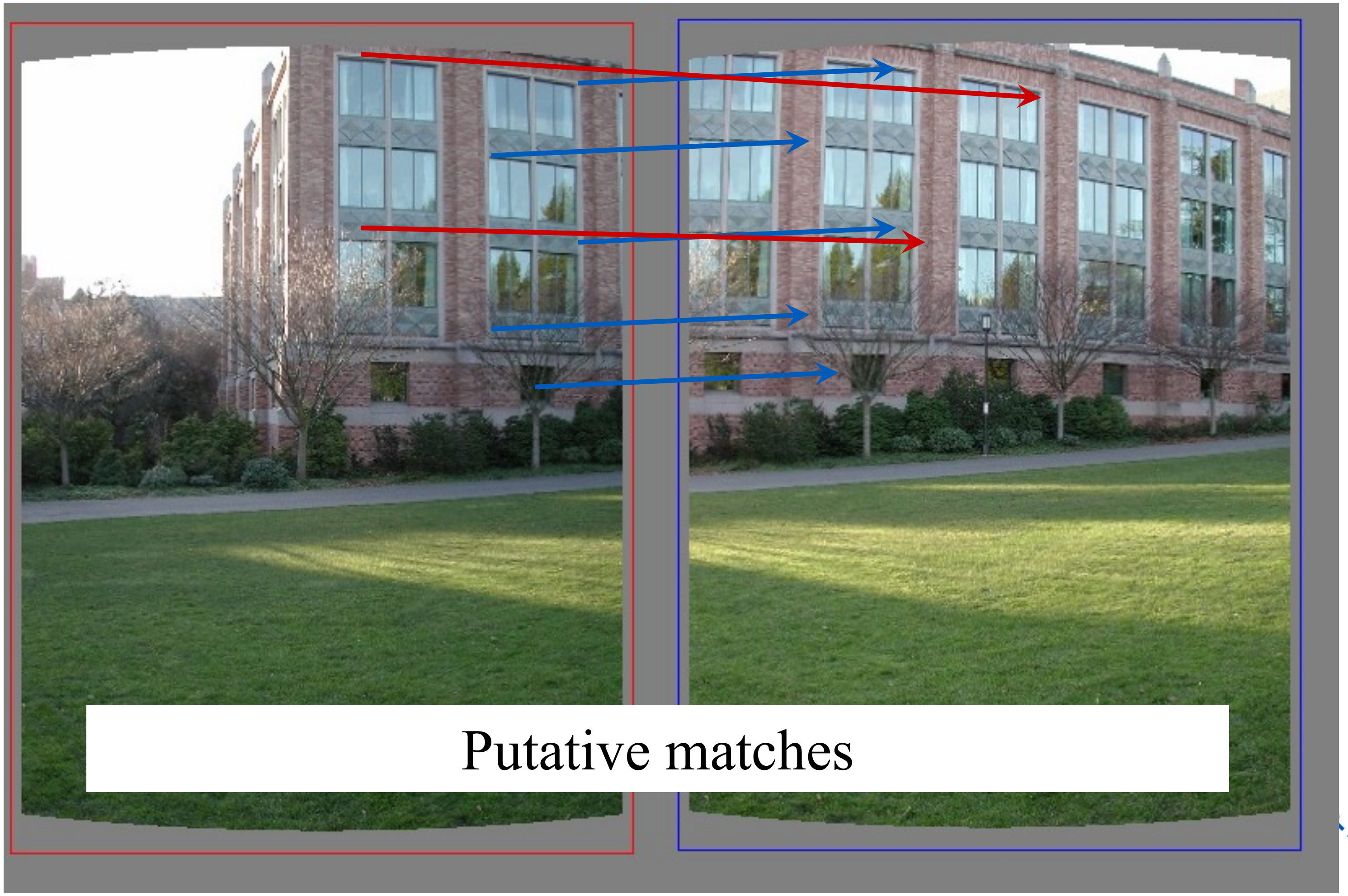
How to choose parameters?

- Number of sampled points n: minimum points to fit a model.
- Inlier threshold δ.
  - Choose δ so that a good point with noise is likely within threshold.
- To determine the number of iterations K.
  - Desired probability of success (p): at least one useful result.
  - Let w be the probability of choosing an inlier when selecting a point.
    - w = number of inliers in data / number of points in data
  - n points selected independently for estimating a model.
  - w^n: the probability that all n points are inliers.
  - $1 - w^n$: probability of at least one of the n points is an outlier.
  - $(1 - w^n)^k$: after k iterations, never select a set of n inlier points.
  - $1 - p = (1 - w^n)^K$
  - $K = \frac{\log(1-p)}{\log(1-w^n)}$

	$p = 0.99$, proportion of outliers $(1 - w)$						
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

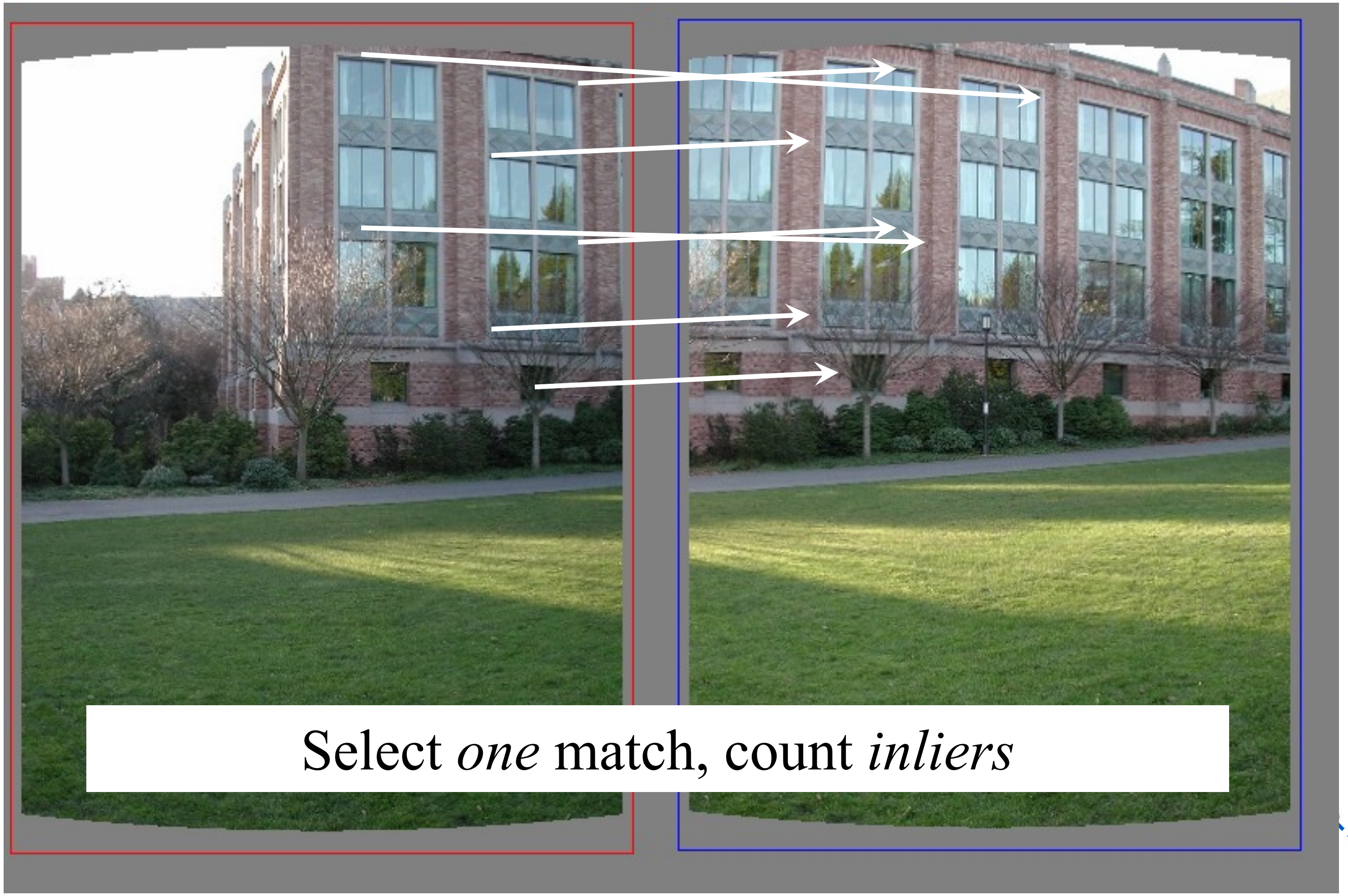
SAIR
Spatial AI & Robotics Lab

RANSAC example: Translation

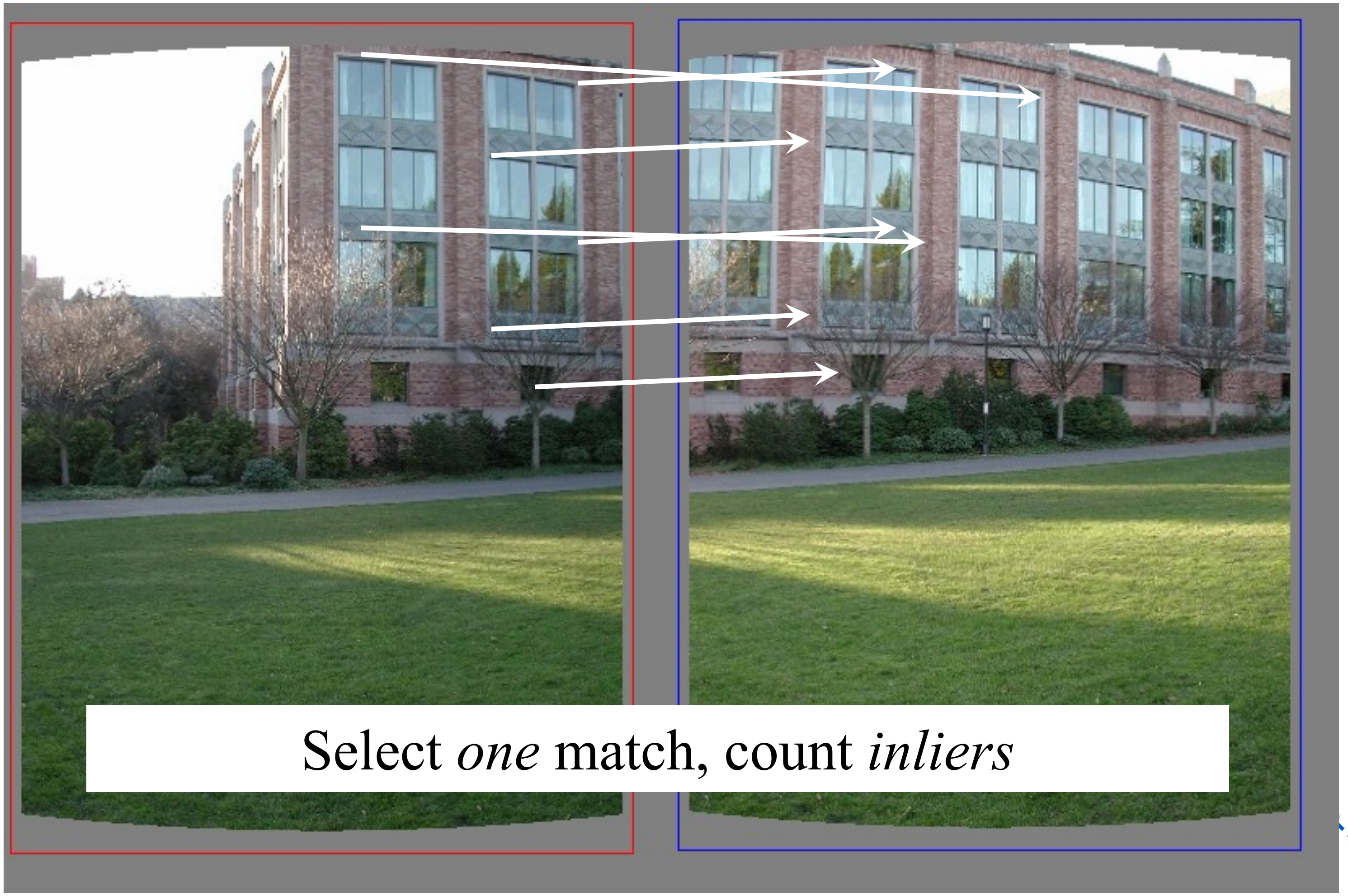


Source: Rick Szeliski

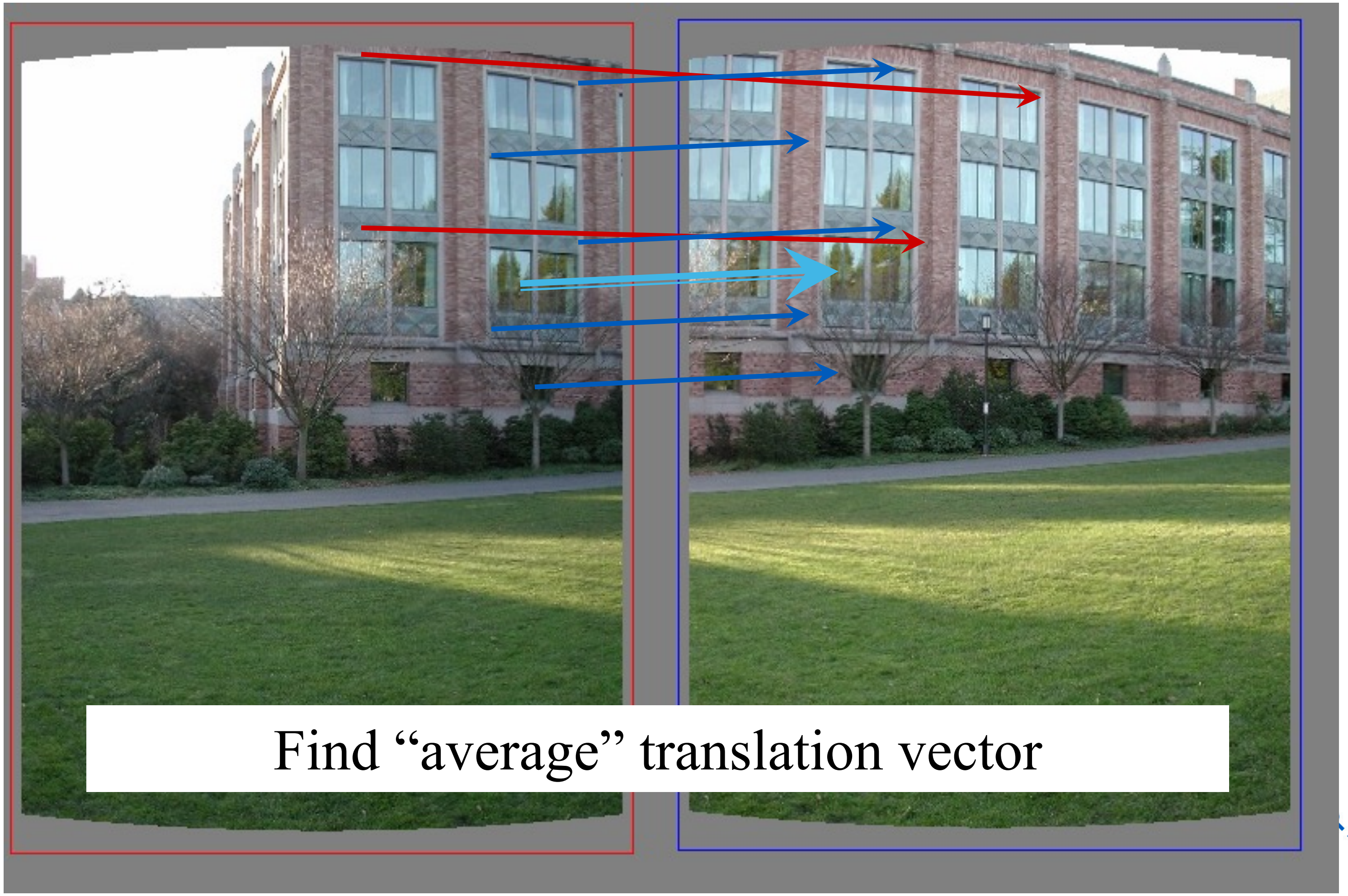
RANSAC example: Translation



RANSAC example: Translation



RANSAC example: Translation



Summary

- Choose:
  - Inlier threshold
    - Related to the amount of noise we expect
  - Number of rounds
    - Related to the percentage of outliners we expect
    - Related to the probability of success we are hoping for.

RANSAC ALGORITHM

- Input:
  - Data: a set of observed data points
  - Model: a model to fit the data
  - Threshold: a threshold to determine inliers
- Output: BestModel: the model with the most inliers
- Repeat for a fixed number of iterations:
  - 1. Select a random subset of data
  - 2. Fit the model to the data points in the subset
  - 3. Determine the inliers by comparing the fitted model to data
  - 4. If the number of inliers exceeds the threshold
    - re-estimate the model using all the inliers
  - 5. Store the model if it has the most inliers seen so far
- Return BestModel

RANSAC Algorithm

```
Given:
    data – A set of observations.
    model – A model to explain the observed data points.
    n – The minimum number of data points required to estimate the model parameters.
    k – The maximum number of iterations allowed in the algorithm.
    t – A threshold value to determine data points that are fit well by the model (inlier).
    d – The number of close data points (inliers) required to assert that the model fits well to the data.

Return:
    bestFit – The model parameters which may best fit the data (or null if no good model is found).


iterations = 0
bestFit = null
bestErr = something really large // This parameter is used to sharpen the model parameters to the best data
fitting as iterations goes on.

while iterations < k do
    maybeInliers := n randomly selected values from data
    maybeModel := model parameters fitted to maybeInliers
    confirmedInliers := empty set
    for every point in data do
        if point fits maybeModel with an error smaller than t then
            add point to confirmedInliers
        end if
    end for
    if the number of elements in confirmedInliers is > d then
        // This implies that we may have found a good model.
        // Now test how good it is.
        betterModel := model parameters fitted to all the points in confirmedInliers
        thisErr := a measure of how well betterModel fits these points
        if thisErr < bestErr then
            bestFit := betterModel
            bestErr := thisErr
        end if
    end if
    increment iterations
end while

return bestFit
```

RANSAC Properties

Good

- Robust to outliers
- Applicable for larger number of model parameters than Hough transform.
- Optimization parameters are easier to choose than Hough transform.
  - Lines with normal form works for Hough, but slope-intercept form not.

Bad

- Computational time grows quickly with outliers and parameters
  - While Hough transform grows quickly with number of parameters.
- Not good for getting multiple fits
  - Hough transform can fit multiple lines simultaneously.

More applications

- Computing a homography (e.g., image stitching)
- Estimating fundamental matrix (relating two views)