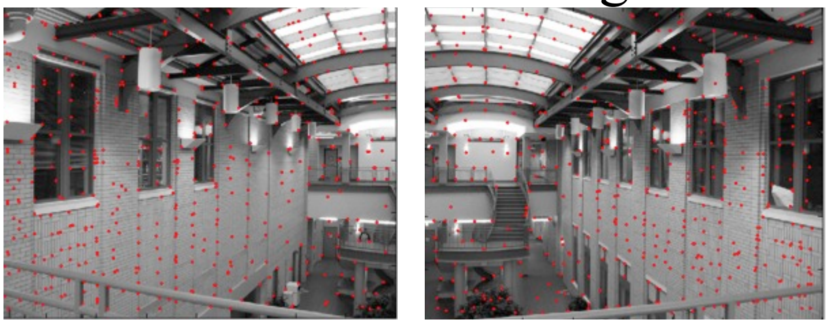
Robust Feature Matching (RANSAC)

adapted from CSE 576
by Richard Szeliski

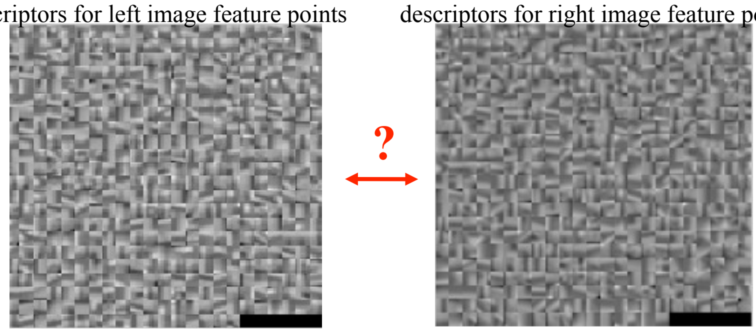
Outline

- Feature matching
  - exhaustive search
  - hashing
  - nearest neighbor techniques
- RANSAC

Feature matching



descriptors for left image feature points descriptors for right image feature points

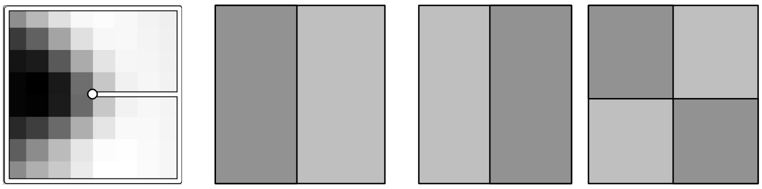


Feature matching

- Exhaustive search
  - for each feature in one image, look at *all* the other features in the other image(s)
- Hashing
  - compute a short descriptor from each feature vector, or hash longer descriptors (randomly)
- Nearest neighbor techniques
  - *k*-trees and their variants (Best Bin First)

Wavelet-based hashing

Compute a short (3-vector) descriptor from an 8x8 patch using a Haar "wavelet"



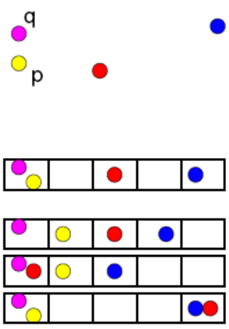
Quantize each value into 10 (overlapping) bins (10^3 total entries)

[Brown, Szeliski, Winder, CVPR'2005]

Locality sensitive hashing

[Indyk-Motwani'98]

- Idea: construct hash functions $g: R^d \rightarrow U$ such that for any points p,q:
  - If $D(p,q) \leq r$, then $Pr[g(p)=g(q)]$ is ~~"high"~~ "not-so-small"
  - If $D(p,q) > cr$, then $Pr[g(p)=g(q)]$ is "small"
- Then we can solve the problem by hashing



Nearest neighbor techniques

k-D tree
and

Best Bin
First
(BBF)

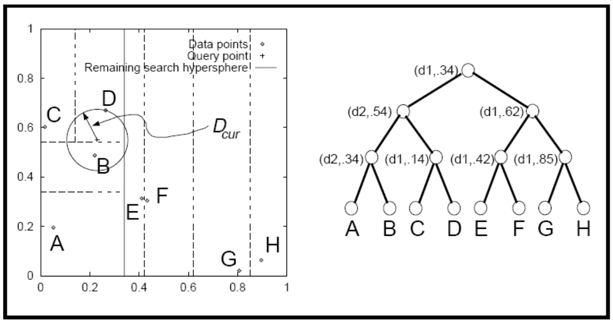


Figure 6: kd-tree with 8 data points labelled A-H, dimension of space k=2. On the right is the full tree, the leaf nodes containing the data points. Internal node information consists of the dimension of the cut plane and the value of the cut in that dimension. On the left is the 2D feature space carved into various sizes and shapes of bin, according to the distribution of the data points. The two representations are isomorphic. The situation shown on the left is after initial tree traversal to locate the bin for query point "+" (contains point D). In standard search, the closest nodes in the tree are examined first (starting at C). In BBF search, the closest bins to query point q are examined first (starting at B). The latter is more likely to maximize the overlap of (i) the hypersphere centered on q with radius D_{cur}, and (ii) the hyperrectangle of the bin to be searched. In this case, BBF search reduces the number of leaves to examine, since once point B is discovered, all other branches can be pruned.

Indexing Without Invariants in 3D Object Recognition, Beis and Lowe, PAMI'99

Robust feature matching through RANSAC

Nikon D70. Stitched Panorama. The sky has been retouched. No other image manipulation.

with a lot of slides stolen from
Steve Seitz and Rick Szeliski

15-463: Computational Photography
Alexei Efros, CMU, Fall 2005

Strategies to match images robustly

(a) Working with individual features: For each feature point, find most similar point in other image (SIFT distance)

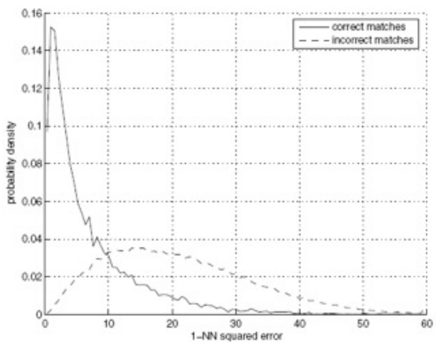
Reject ambiguous matches where there are too many similar points

(b) Working with all the features: Given some good feature matches, look for possible homographies relating the two images

Reject homographies that don't have many feature matches.

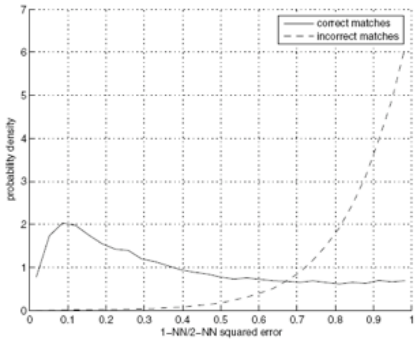
(a) Feature-space outlier rejection

- Let's not match all features, but only these that have "similar enough" matches?
- How can we do it?
  - SSD(patch1,patch2) < threshold
  - How to set threshold? Not so easy.

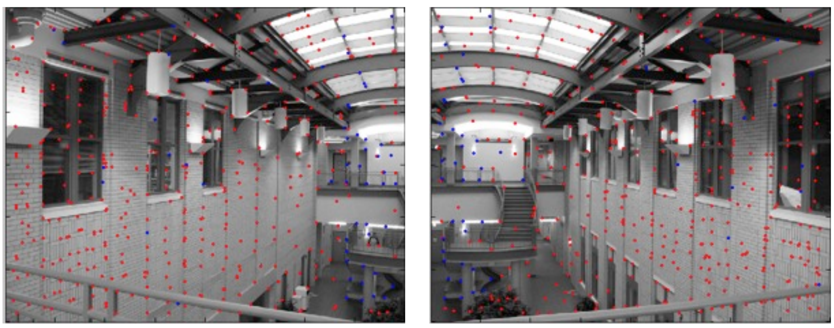


Feature-space outlier rejection

- A better way [Lowe, 1999]:
  - 1-NN: SSD of the closest match
  - 2-NN: SSD of the second-closest match
  - Look at how much better 1-NN is than 2-NN, e.g. 1-NN/2-NN
  - That is, is our best match so much better than the rest?



Feature-space outlier rejection



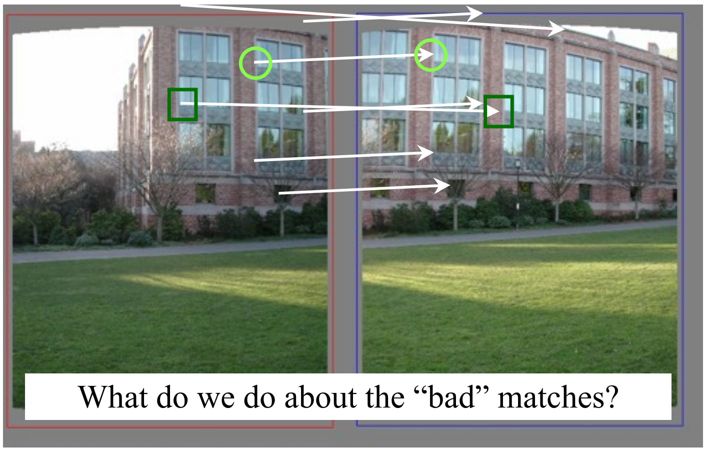
inlier

outlier

- Can we now compute H from the blue points?
  - No! Still too many outliers…
  - What can we do?

(b) Matching many features--looking for a good homography

Simplified illustration with translation instead of homography

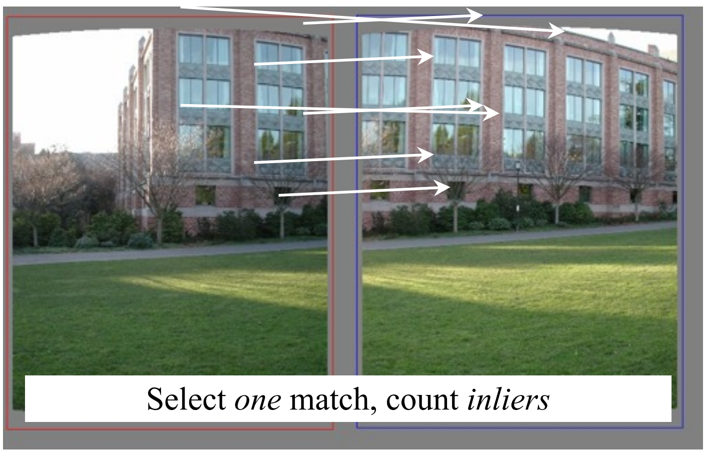


Good

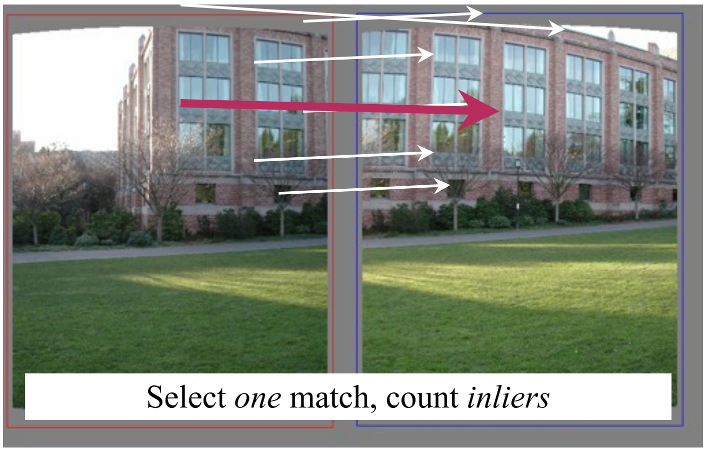
Bad

Note: at this point we don’t know which ones are good/bad

RAndom SAmple Consensus

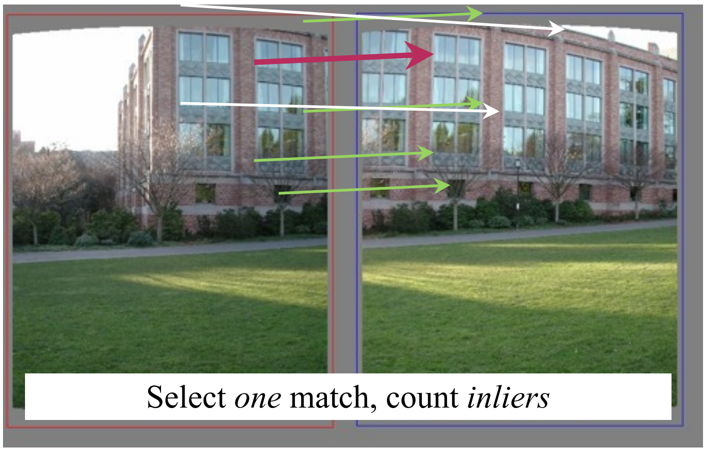


RAndom SAmple Consensus



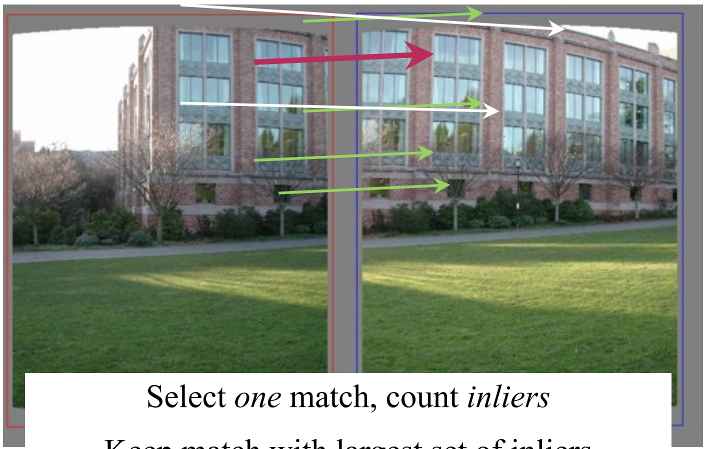
0 inliers

RAndom SAmple Consensus



4 inliers

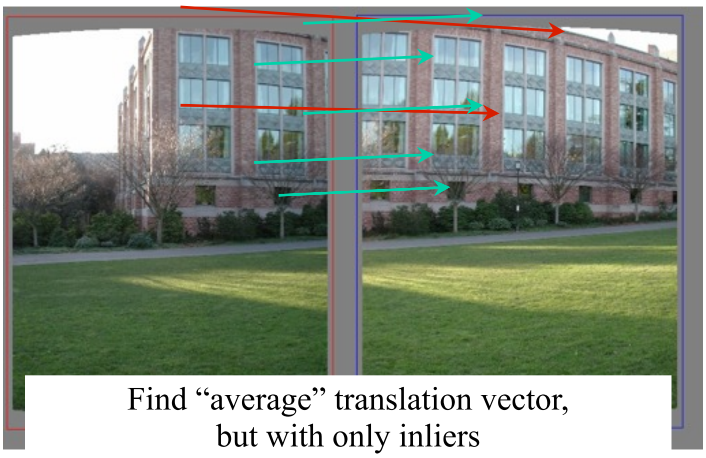
<u>RA</u>ndom <u>SA</u>mple <u>C</u>onsensus



Select *one* match, count *inliers*

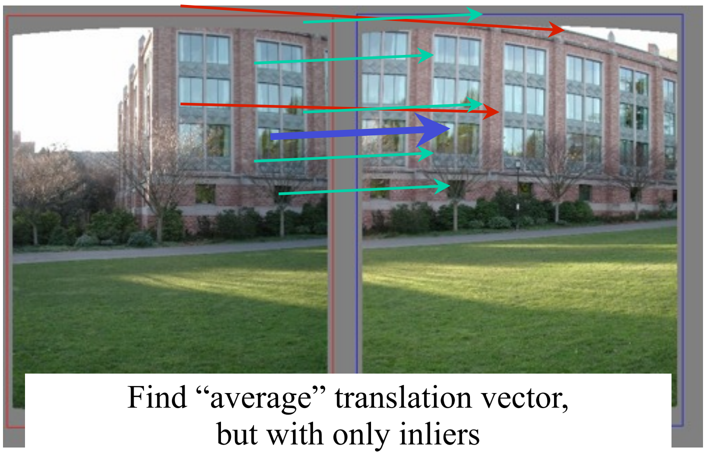
Keep match with largest set of inliers

At the end: Least squares fit



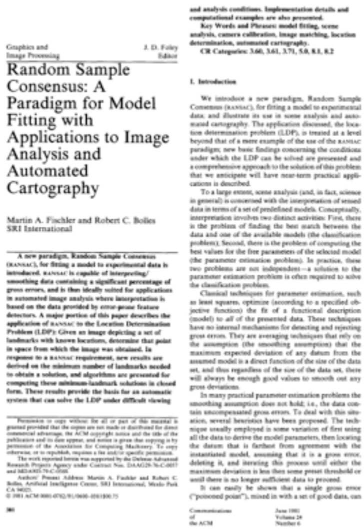
Find “average” translation vector,
but with only inliers

At the end: Least squares fit



Reference

- M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.
- http://portal.acm.org/citation.cfm?id=358692



Graphics and Image Processing — J. D. Foley, Editor

Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography

Martin A. Fischler and Robert C. Bolles
SRI International

A new paradigm, Random Sample Consensus (RANSAC), for fitting a model to experimental data is introduced. RANSAC is capable of interpreting/smoothing data containing a significant percentage of gross errors, and is thus ideally suited for applications in automated image analysis where interpretation is based on the data provided by error-prone feature detectors. A major portion of this paper describes the application of RANSAC to the Location Determination Problem (LDP): Given an image depicting a set of landmarks with known locations, determine that point in space from which the image was obtained. In response to a RANSAC requirement, new results are derived on the minimum number of landmarks needed to obtain a solution, and algorithms are presented for computing these minimum-landmark solutions in closed form. These results provide the basis for an automatic system that can solve the LDP under difficult viewing and analysis conditions. Implementation details and computational examples are also presented.

Key Words and Phrases: model fitting, scene analysis, camera calibration, image matching, location determination, automated cartography.

CR Categories: 3.60, 3.61, 3.71, 5.0, 8.1, 8.2

1. Introduction

We introduce a new paradigm, Random Sample Consensus (RANSAC), for fitting a model to experimental data; and illustrate its use in scene analysis and automated cartography. The application discussed, the location determination problem (LDP), is treated at a level beyond that of a mere example of the use of the RANSAC paradigm; new basic findings concerning the conditions under which the LDP can be solved are presented and a comprehensive approach to the solution of this problem that we anticipate will have near-term practical applications is described.

To a large extent, scene analysis (and, in fact, science in general) is concerned with the interpretation of sensed data in terms of a set of predefined models. Conceptually, interpretation involves two distinct activities: First, there is the problem of finding the best match between the data and one of the available models (the classification problem); Second, there is the problem of computing the best values for the free parameters of the selected model (the parameter estimation problem). In practice, these two problems are not independent—a solution to the parameter estimation problem is often required to solve the classification problem.

Classical techniques for parameter estimation, such as least squares, optimize (according to a specified objective function) the fit of a functional description (model) to *all* of the presented data. These techniques have no internal mechanisms for detecting and rejecting gross errors. They are averaging techniques that rely on the assumption (the smoothing assumption) that the maximum expected deviation of any datum from the assumed model is a direct function of the size of the data set, and thus regardless of the size of the data set, there will always be enough good values to smooth out any gross deviations.

In many practical parameter estimation problems the smoothing assumption does not hold; i.e., the data contain uncompensated gross errors. To deal with this situation, several heuristics have been proposed. The technique usually employed is some variation of first using all the data to derive the model parameters, then locating the datum that is farthest from agreement with the instantiated model, assuming that it is a gross error, deleting it, and iterating this process until either the maximum deviation is less then some preset threshold or until there is no longer sufficient data to proceed.

It can easily be shown that a single gross error ("poisoned point"), mixed in with a set of good data, can

RANSAC for estimating homography

RANSAC loop:

Select four feature pairs (at random)

Compute homography H (exact)

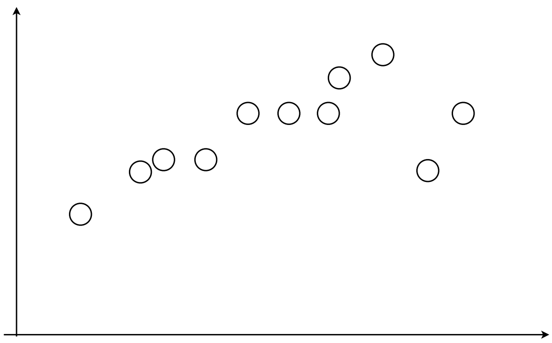
Compute *inliers* where $\|p_i', \boldsymbol{H}\, p_i\| < \varepsilon$

Keep largest set of inliers

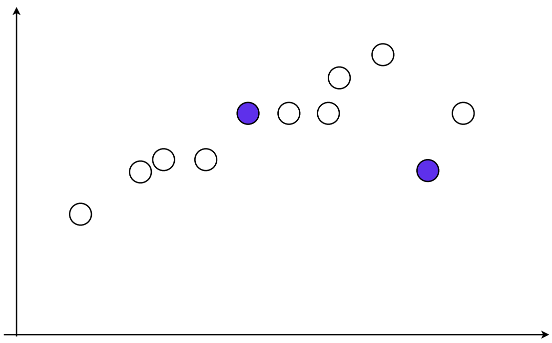
Re-compute least-squares H estimate using all of the inliers

Simple example: fit a line

- Rather than homography H (8 numbers) fit y=ax+b (2 numbers a, b) to 2D pairs

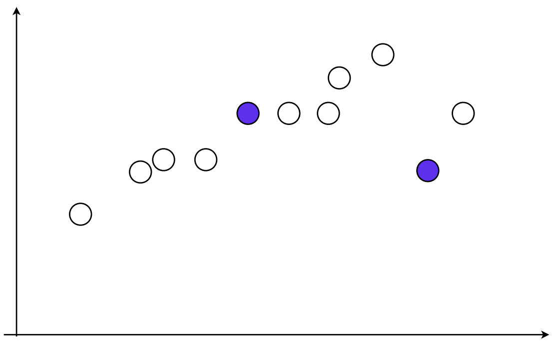


Simple example: fit a line



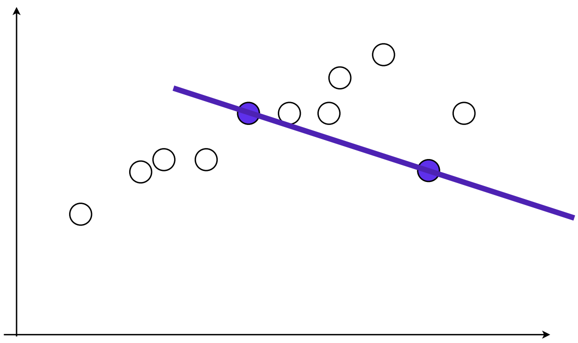
Simple example: fit a line

- Pick 2 points



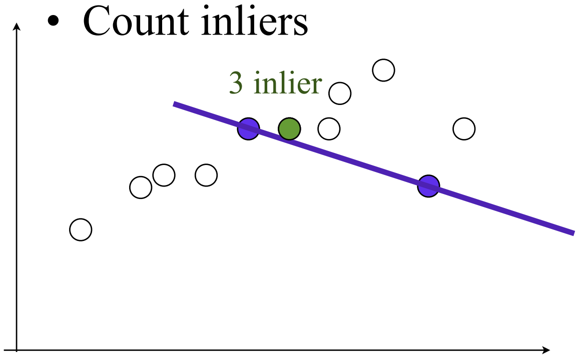
Simple example: fit a line

- Pick 2 points
- Fit line

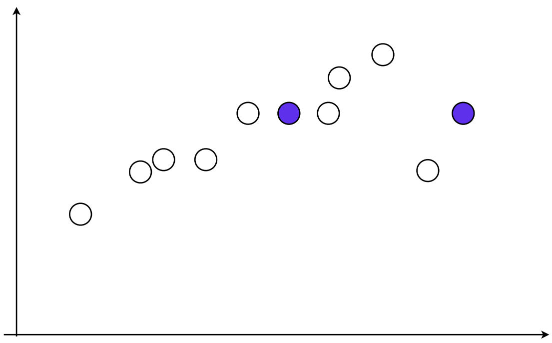


Simple example: fit a line

- Pick 2 points
- Fit line
- Count inliers

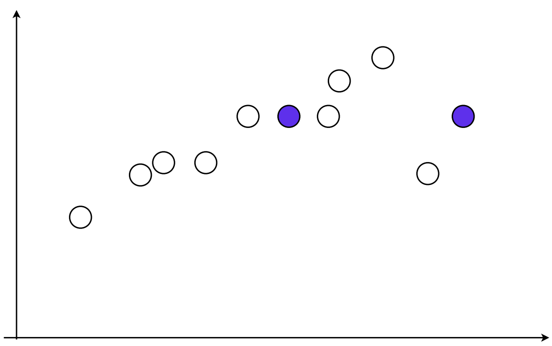


Simple example: fit a line



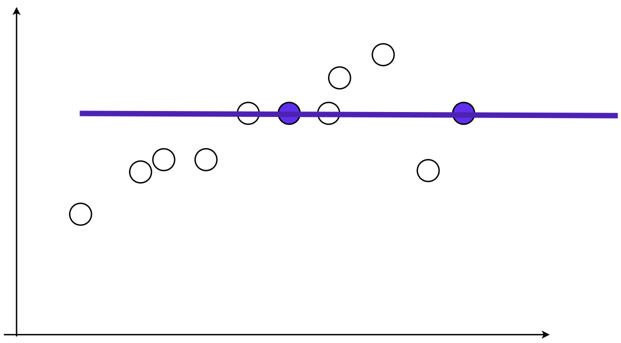
Simple example: fit a line

- Pick 2 points



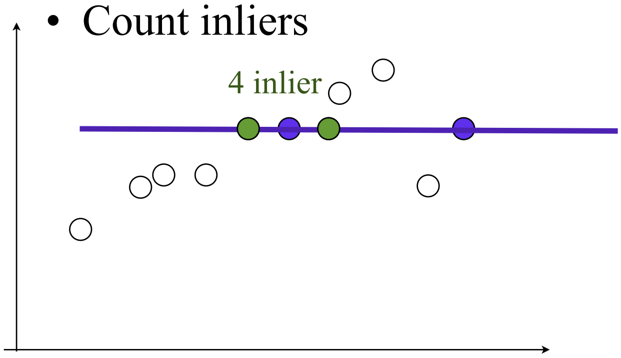
Simple example: fit a line

- Pick 2 points
- Fit line

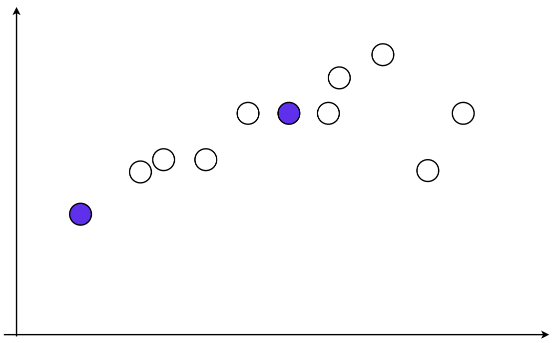


Simple example: fit a line

- Pick 2 points
- Fit line
- Count inliers

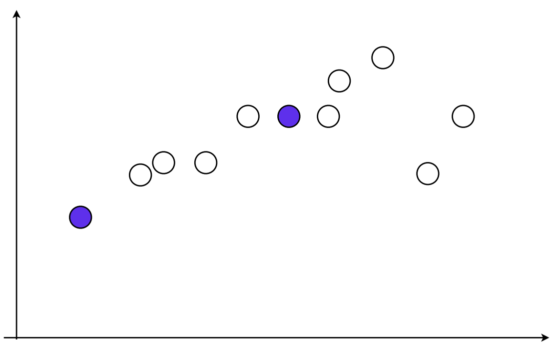


Simple example: fit a line



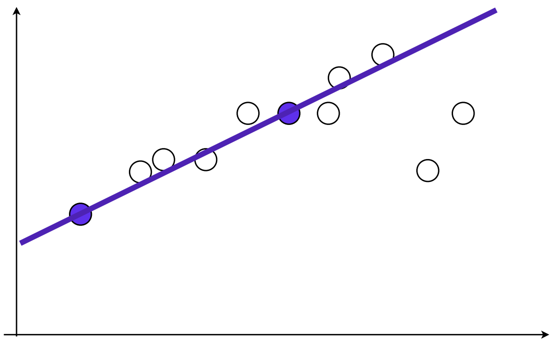
Simple example: fit a line

- Pick 2 points



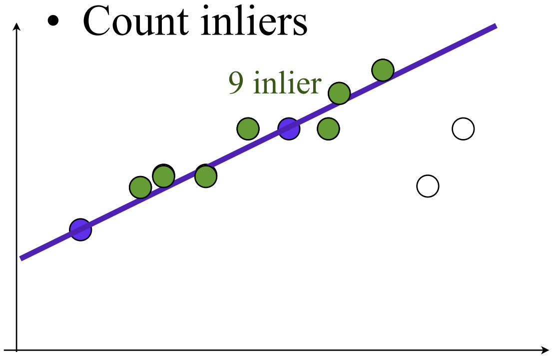
Simple example: fit a line

- Pick 2 points
- Fit line

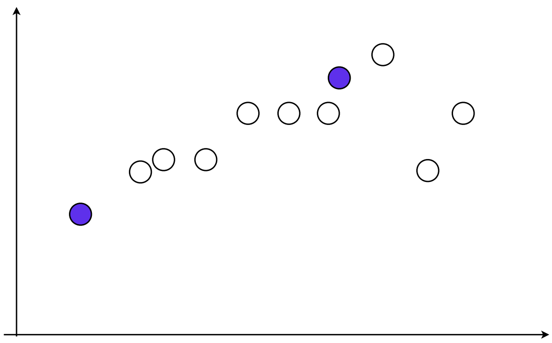


Simple example: fit a line

- Pick 2 points
- Fit line
- Count inliers

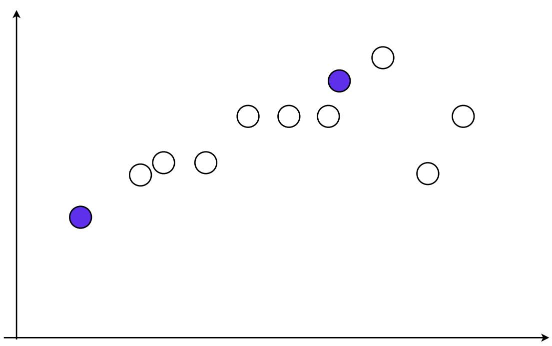


Simple example: fit a line



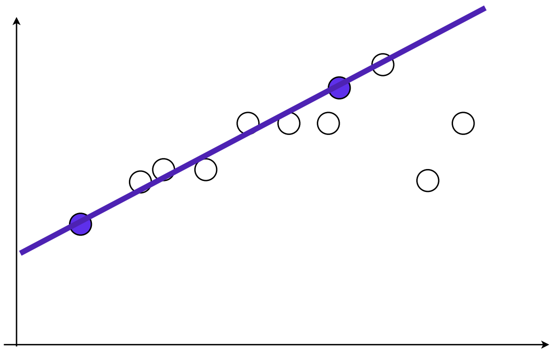
Simple example: fit a line

- Pick 2 points



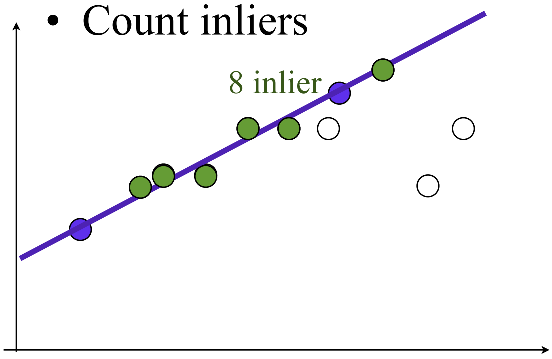
Simple example: fit a line

- Pick 2 points
- Fit line

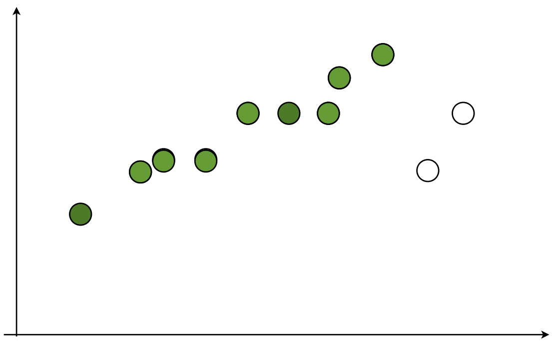


Simple example: fit a line

- Pick 2 points
- Fit line
- Count inliers

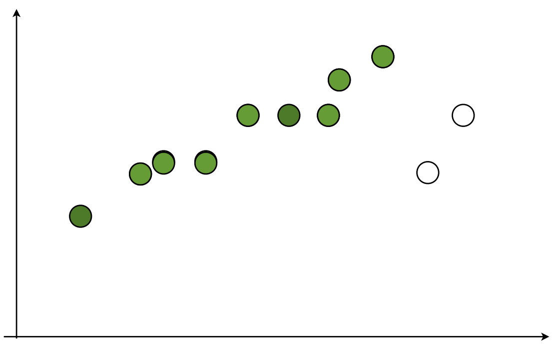


Simple example: fit a line



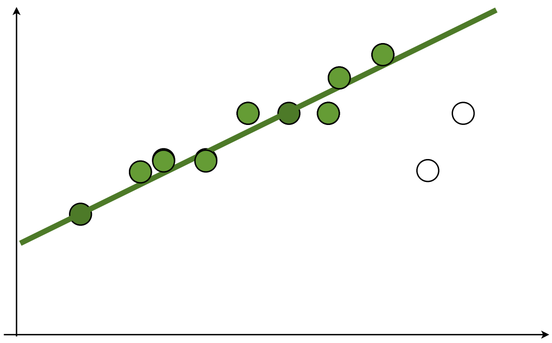
Simple example: fit a line

- Use biggest set of inliers

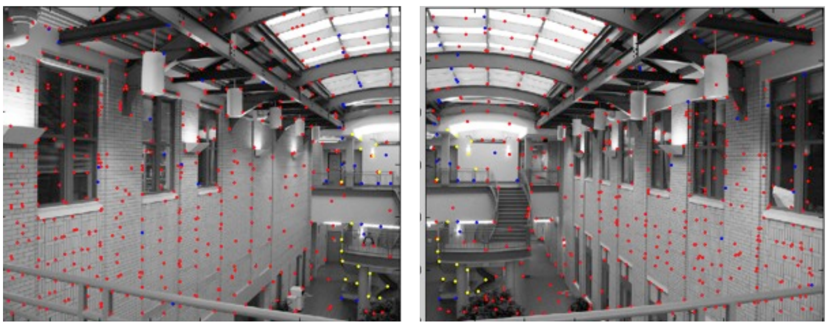


Simple example: fit a line

- Use biggest set of inliers
- Do least-square fit



RANSAC



red:
  rejected by 2nd nearest
  neighbor criterion
blue:
  Ransac outliers
yellow:
  inliers

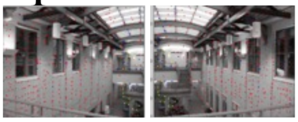
Robustness

- Proportion of inliers in our pairs is G (for "good")
- Our model needs P pairs
  - P=4 for homography
- Probability that we pick P inliers?
  - G^P
- Probability that after N RANSAC iterations we have **not** picked a set of inliers?
  - $(1-G^P)^N$

Robustness: example

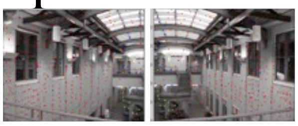
- `Matlab: p=4; x=0.5; n=1000; (1-x^p)^n`
- Proportion of inliers **G=0.5**
- Probability that we pick P=4 inliers?
  - $0.5^4=0.0625$ (6% chance)
- Probability that we have **not** picked a set of inliers?
  - N=100 iterations:
    $(1-0.5^4)^{100}=0.00157$ (1 chance in 600)
  - N=1000 iterations:
    1 chance in 1e28

Robustness: example



- Proportion of inliers **G=0.3**
- Probability that we pick P=4 inliers?
  - 0.3^4=0.0081 (0.8% chance)
- Probability that we have **not** picked a set of inliers?
  - N=100 iterations: $(1-0.3^4)^{100}$=0.44 (1 chance in 2)
  - N=1000 iterations: 1 chance in 3400

Robustness: example



- Proportion of inliers **G=0.1**
- Probability that we pick P=4 inliers?
  - 0.1^4=0.0001 (0.01% chances, 1 in 10,000)
- Probability that we have **not** picked a set of inliers?
  - N=100 iterations: $(1-0.1^4)^{100}$=0.99
  - N=1000 iterations: 90%
  - N=10,000: 36%
  - N=100,000: 1 in 22,000

Robustness: conclusions

- Effect of number of parameters of model/ number of necessary pairs
  - Bad exponential
- Effect of percentage of inliers
  - Base of the exponential
- Effect of number of iterations
  - Good exponential

RANSAC recap

- For fitting a model with low number P of parameters (8 for homographies)
- Loop
  - Select P random data points
  - Fit model
  - Count inliers
    (other data points well fit by this model)
- Keep model with largest number of inliers