Lecture 18: Multi-view reconstruction

Announcements

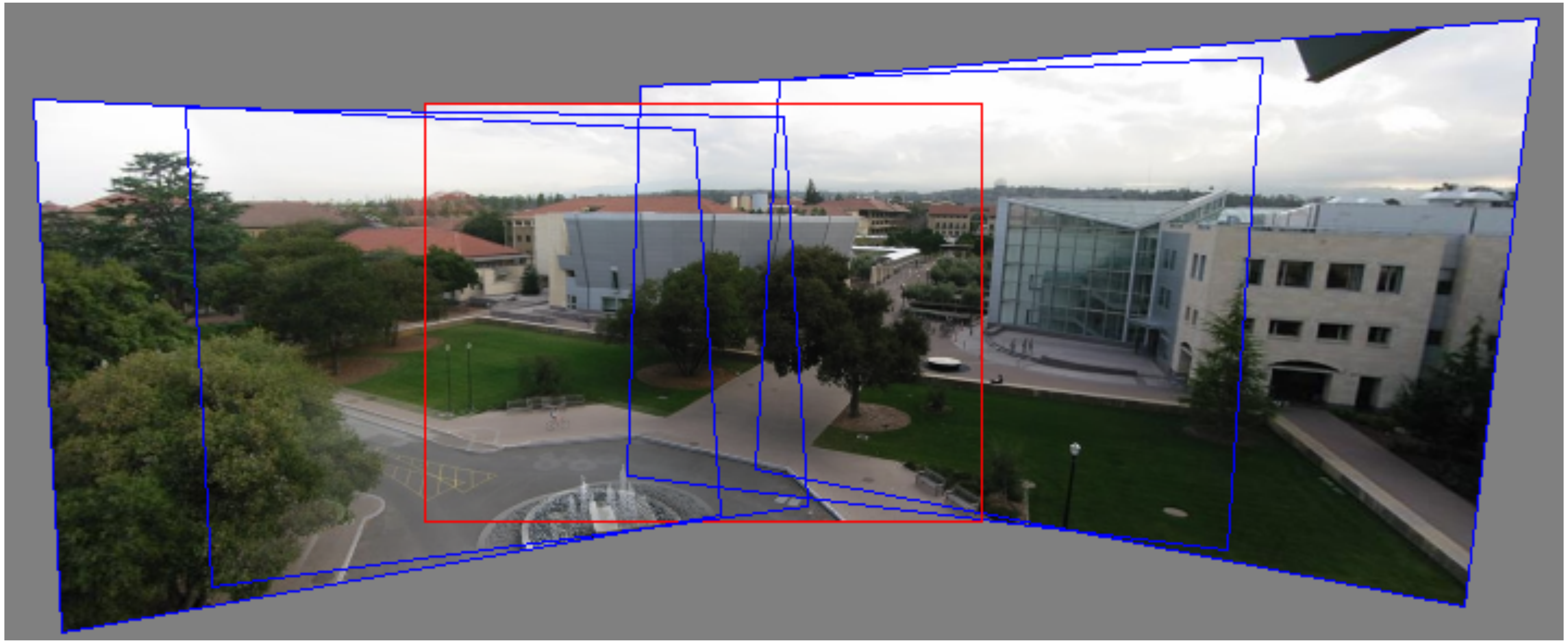
- If you have a question: just ask it (or send message saying that you have a question)
- Please send us feedback!

- PS8 out: representation learning
- Final presentation will take place over video chat.
  - We’ll send a sign-up sheet next week

Today

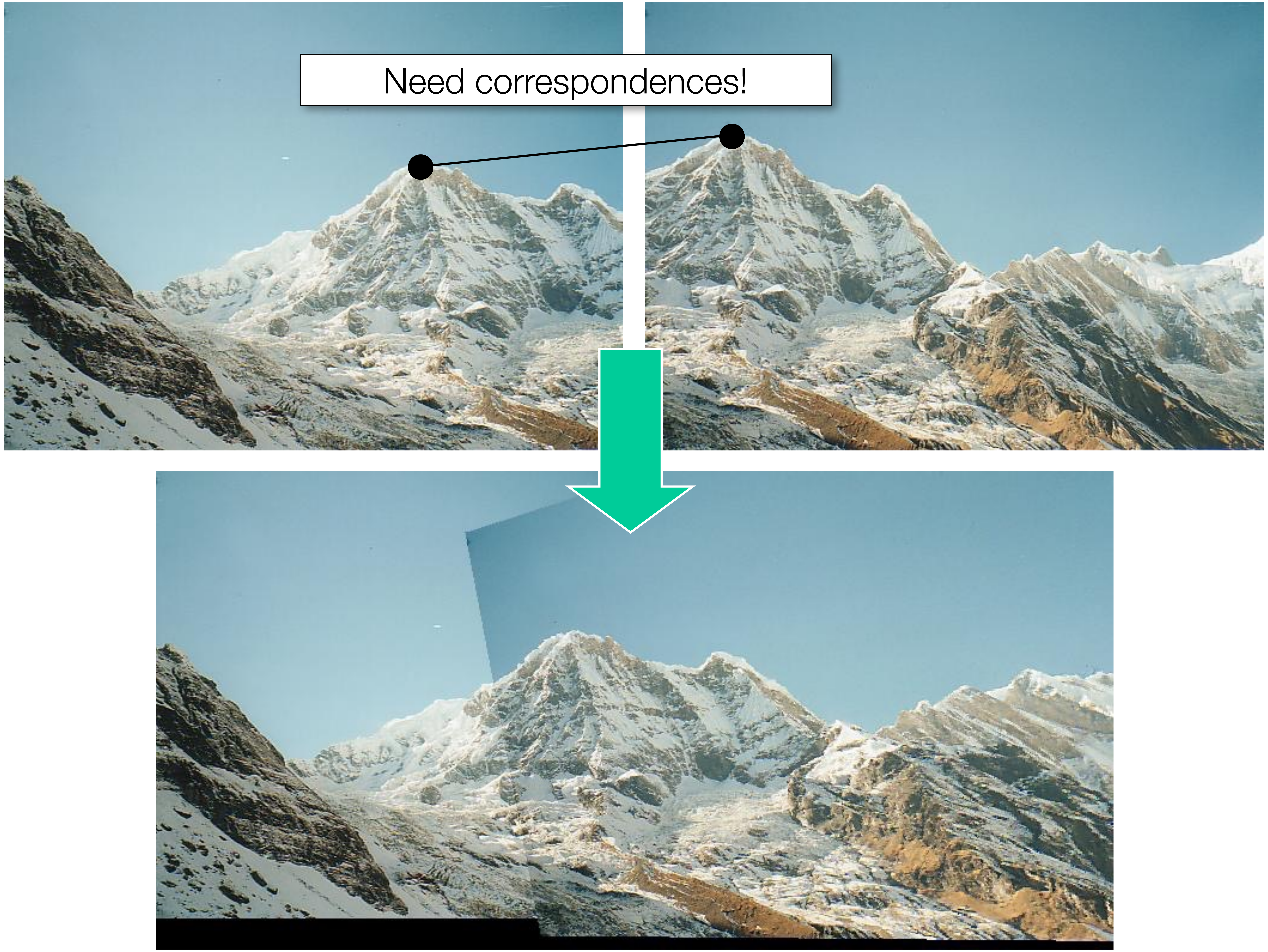
- Finding correspondences
- RANSAC
- Structure from motion

Motivating example: panoramas



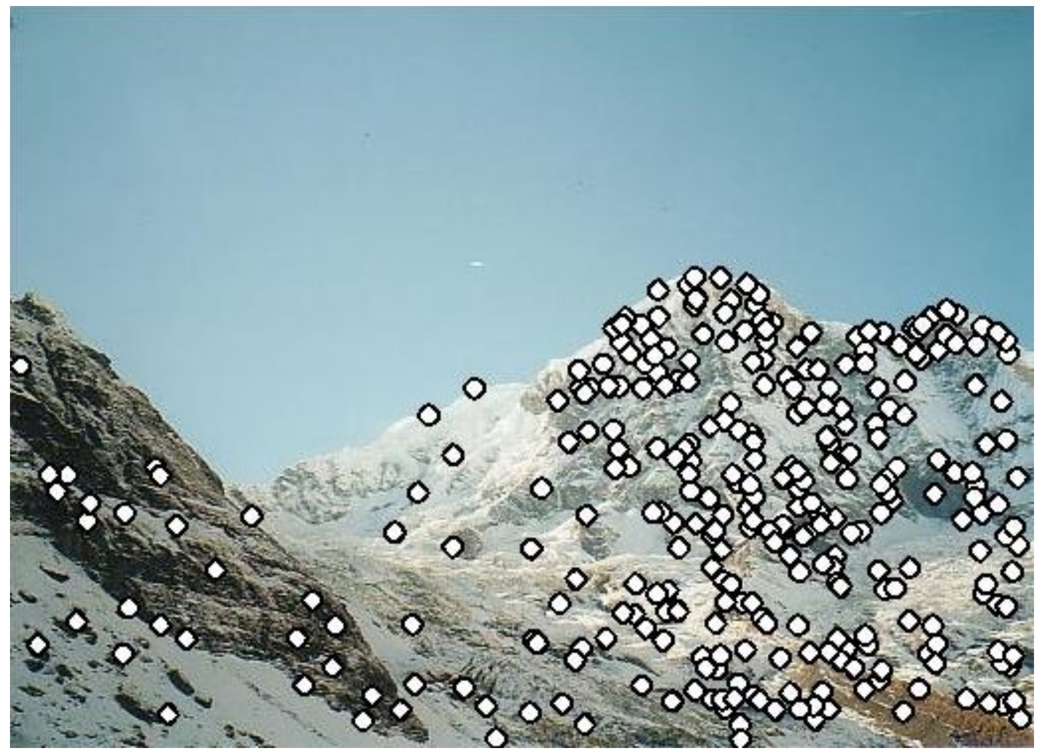
Source: N. Snavely

Warping with a homography



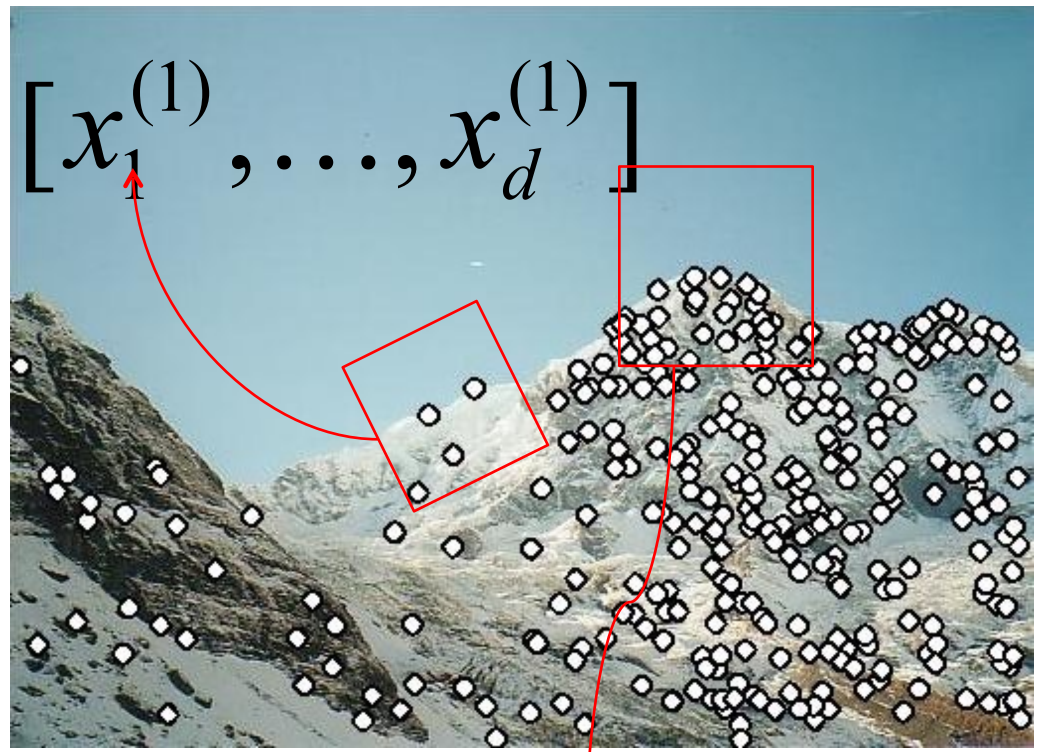
Local features: main components

1) **Detection:** Identify the interest points



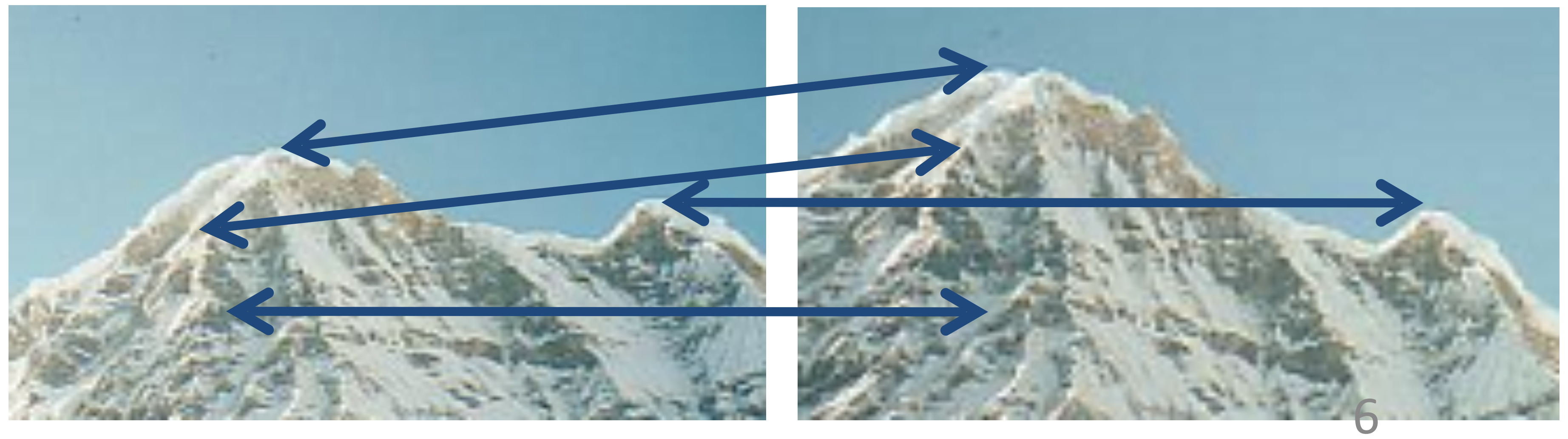
2) **Description:** Extract vector feature descriptor surrounding each interest point.

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

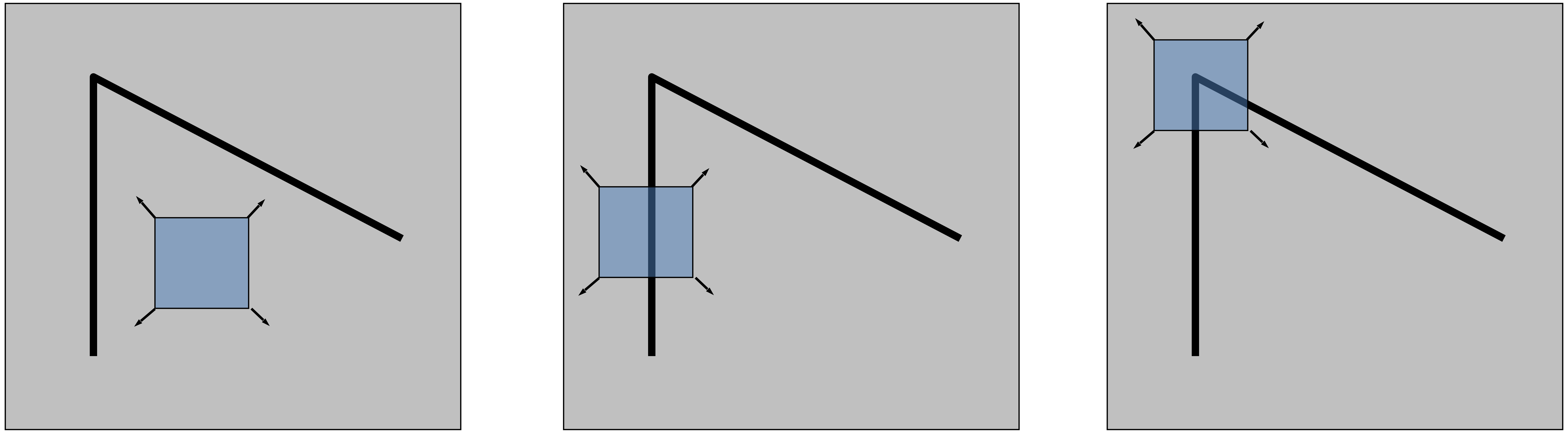
3) **Matching:** Determine correspondence between descriptors in two views



Source: K. Grauman

Which features should we match?

- How does the window change when you shift it?
- Shifting the window in any direction causes a big change



"flat" region:
no change in all directions

"edge":
no change along the edge direction

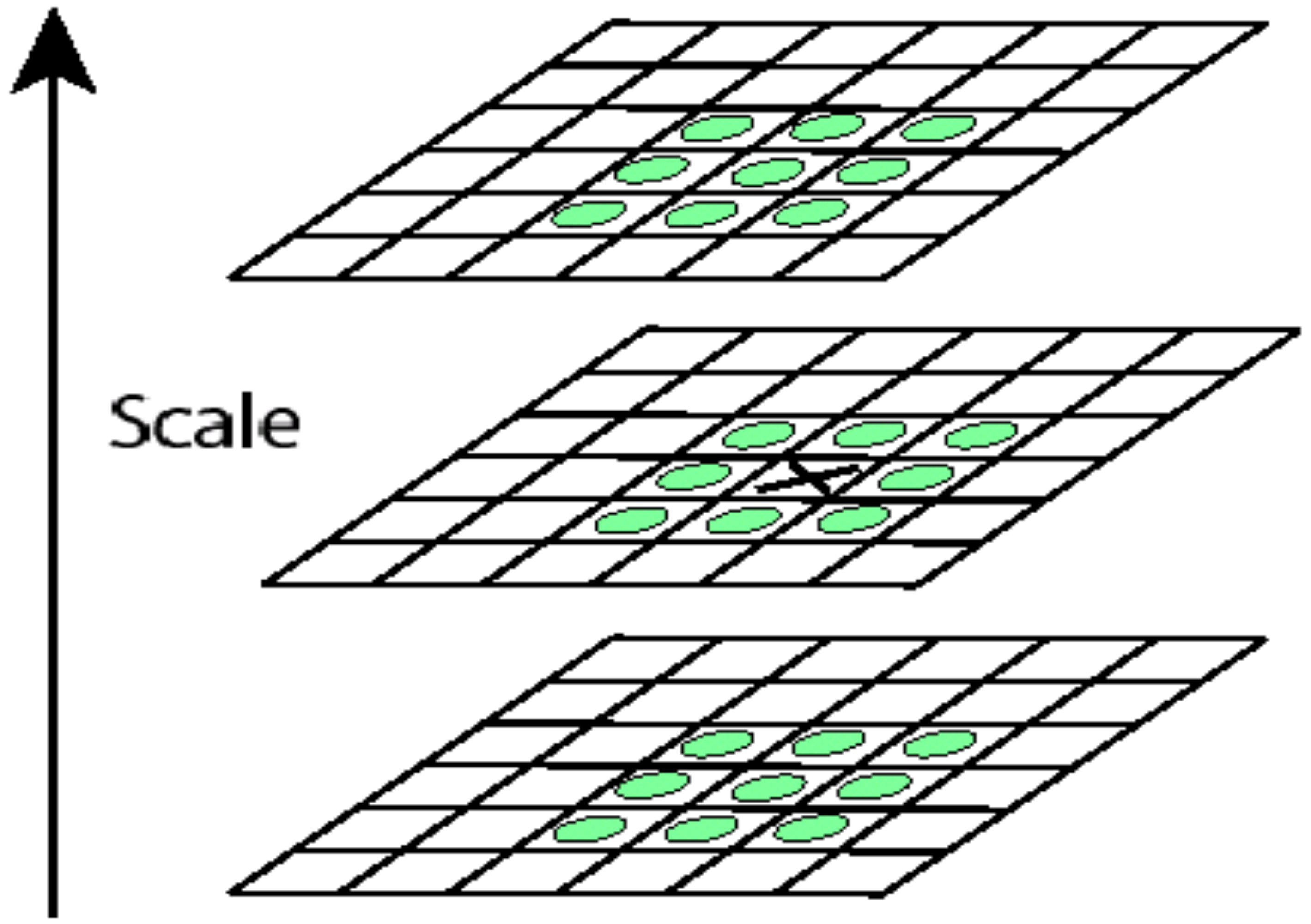
"corner":
significant change in all directions

Source: S. Seitz, D. Frolova, D. Simakov, N. Snavely

Finding keypoints

$$\circ \begin{bmatrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{bmatrix} =$$

Compute difference-of-Gaussians filter (approx. to Laplacian)

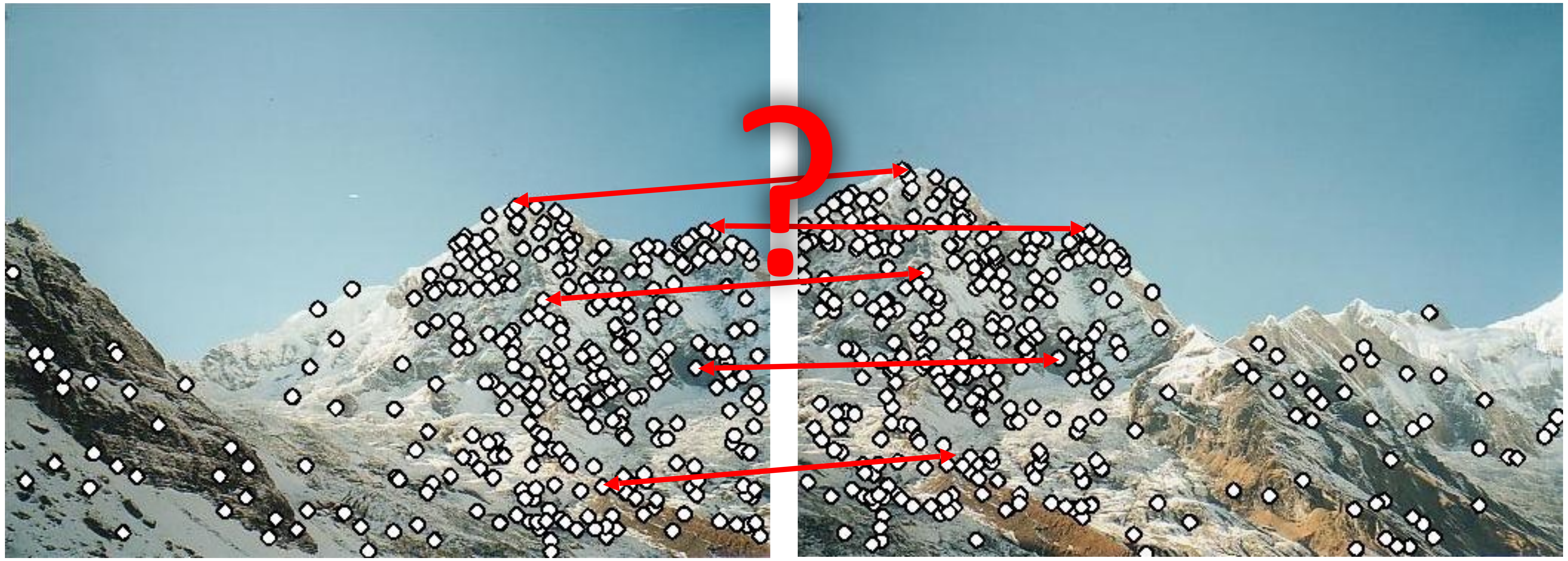


Find local optima in space/ scale using pyramid

Feature descriptors

We know how to detect good points

Next question: **How to match them?**



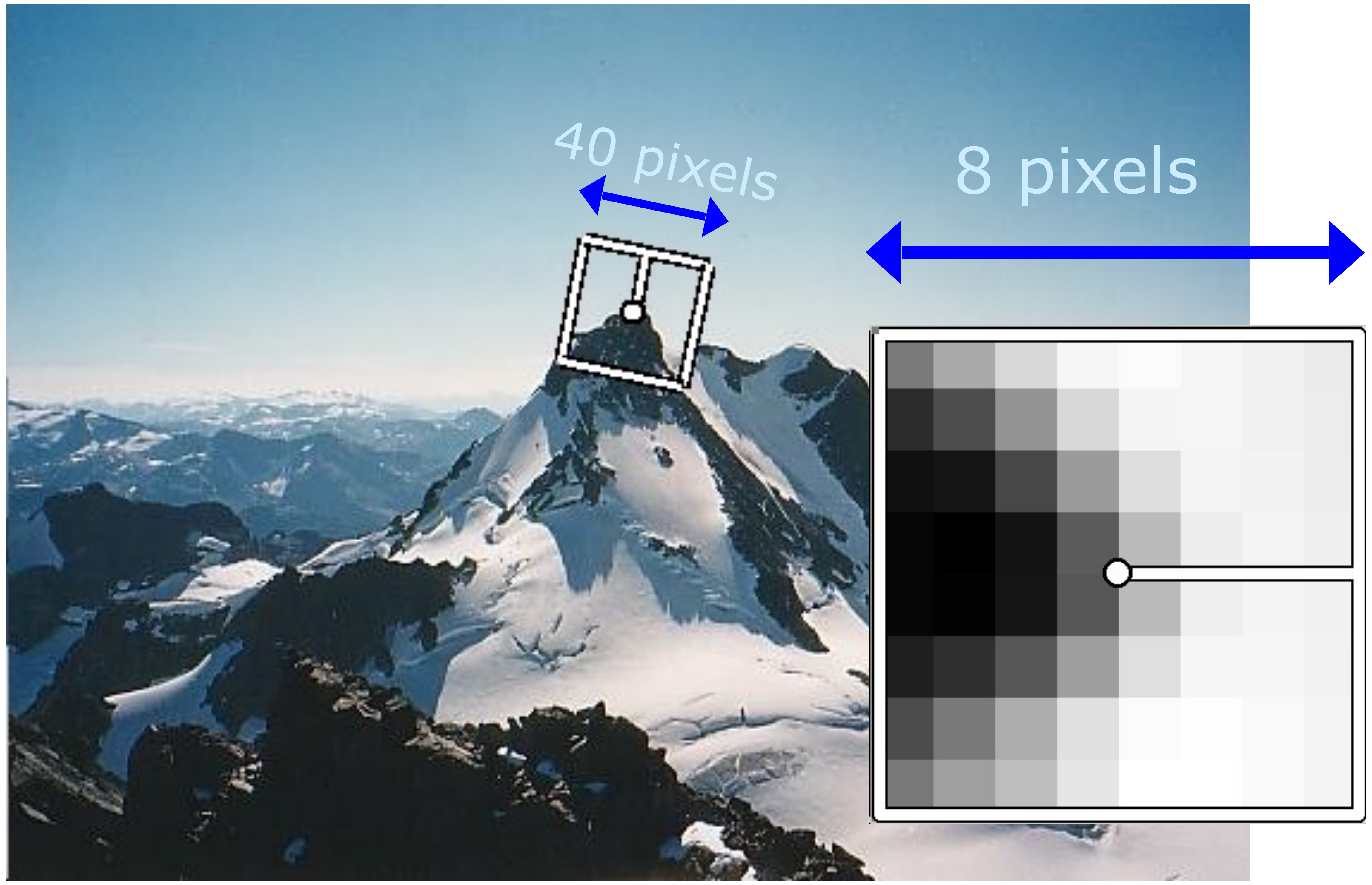
Answer: Come up with a *descriptor* for each point, find similar descriptors between the two images

Source: N. Snavely

Simple idea: normalized image patch

Take 40x40 window around feature

- Find dominant orientation
- Rotate to horizontal
- Sample 8x8 square window centered at feature
- Intensity normalize the window by subtracting the mean, dividing by the standard deviation in the window

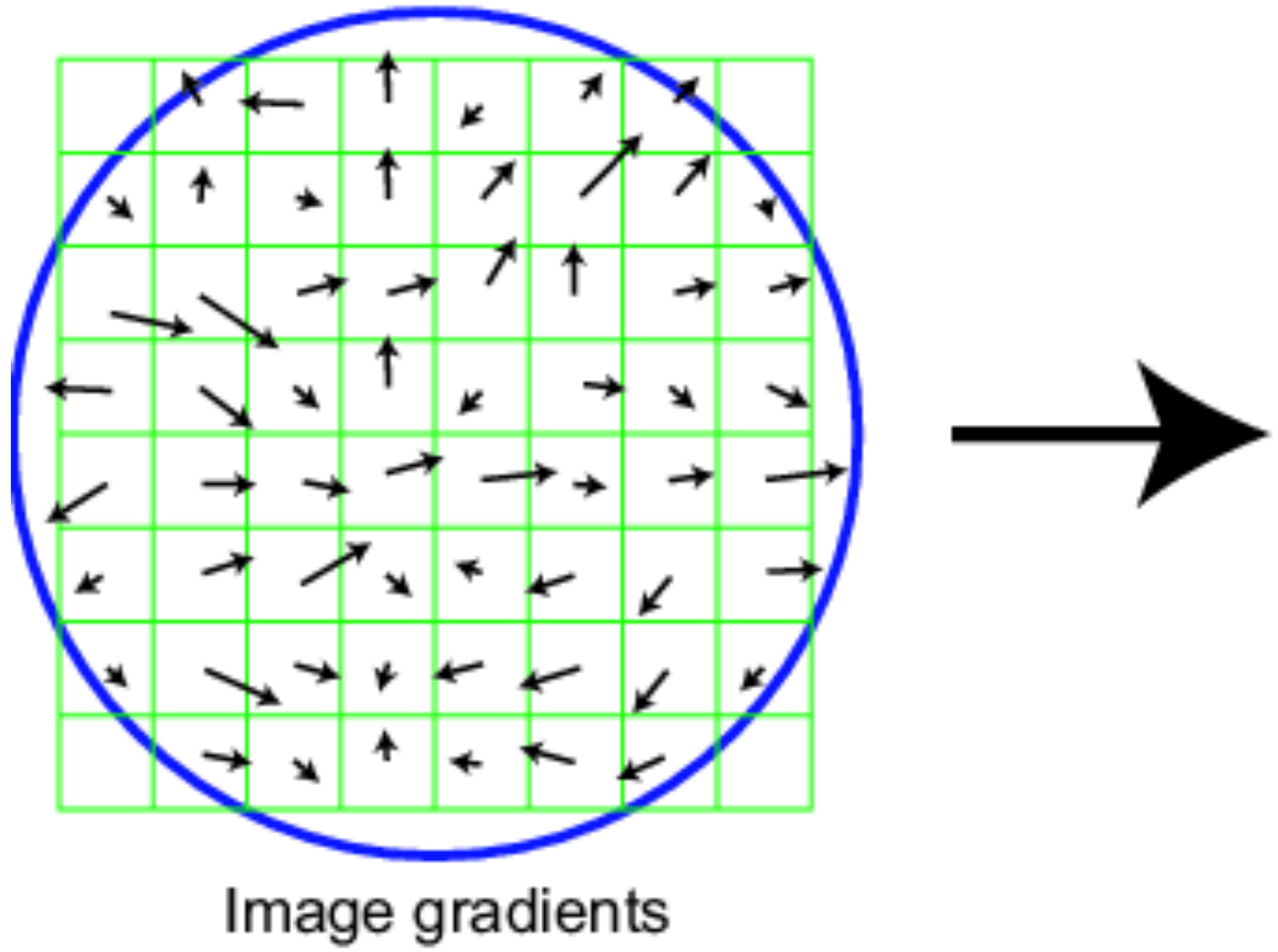


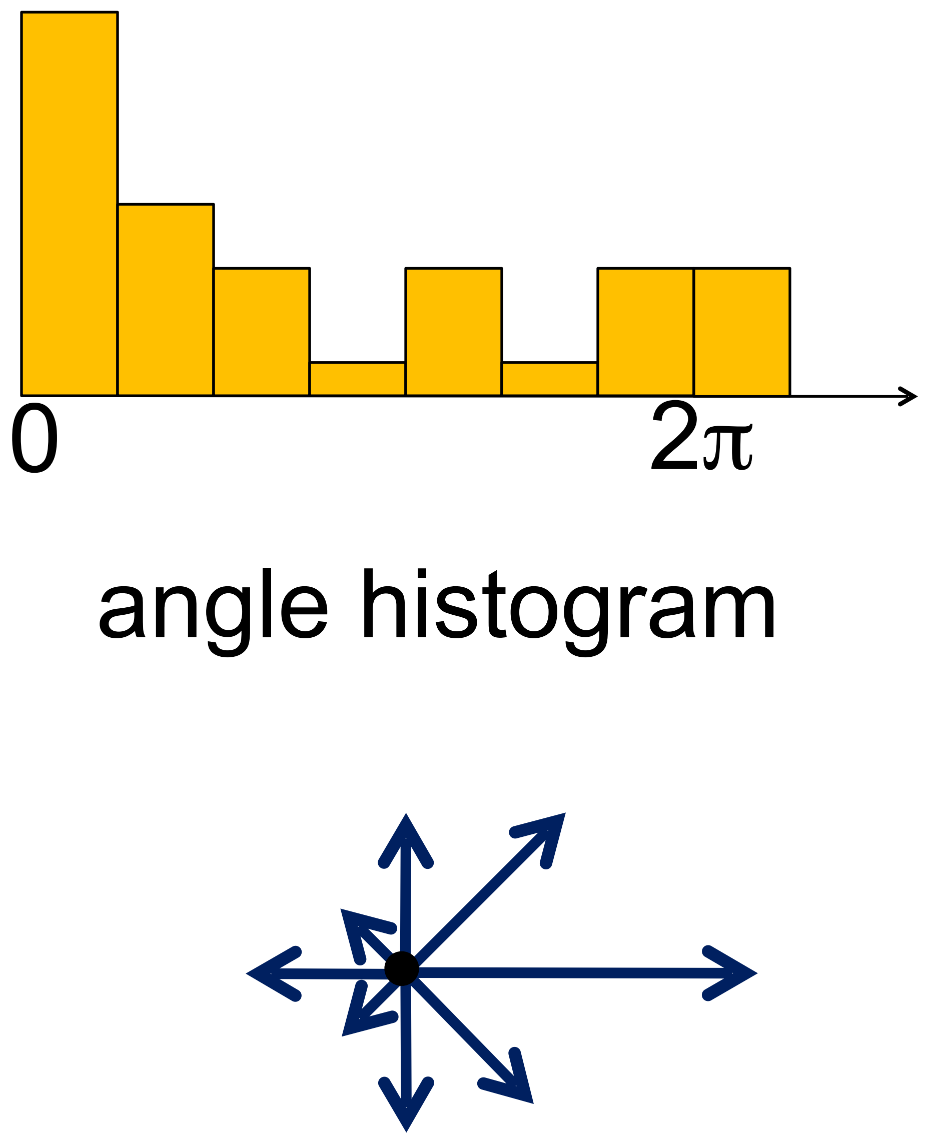
Source: N. Snavely, M. Brown

Scale Invariant Feature Transform

Basic idea: hand-crafted CNN

- Take 16x16 square window around detected feature
- Compute edge orientation for each pixel
- Create histogram of edge orientations



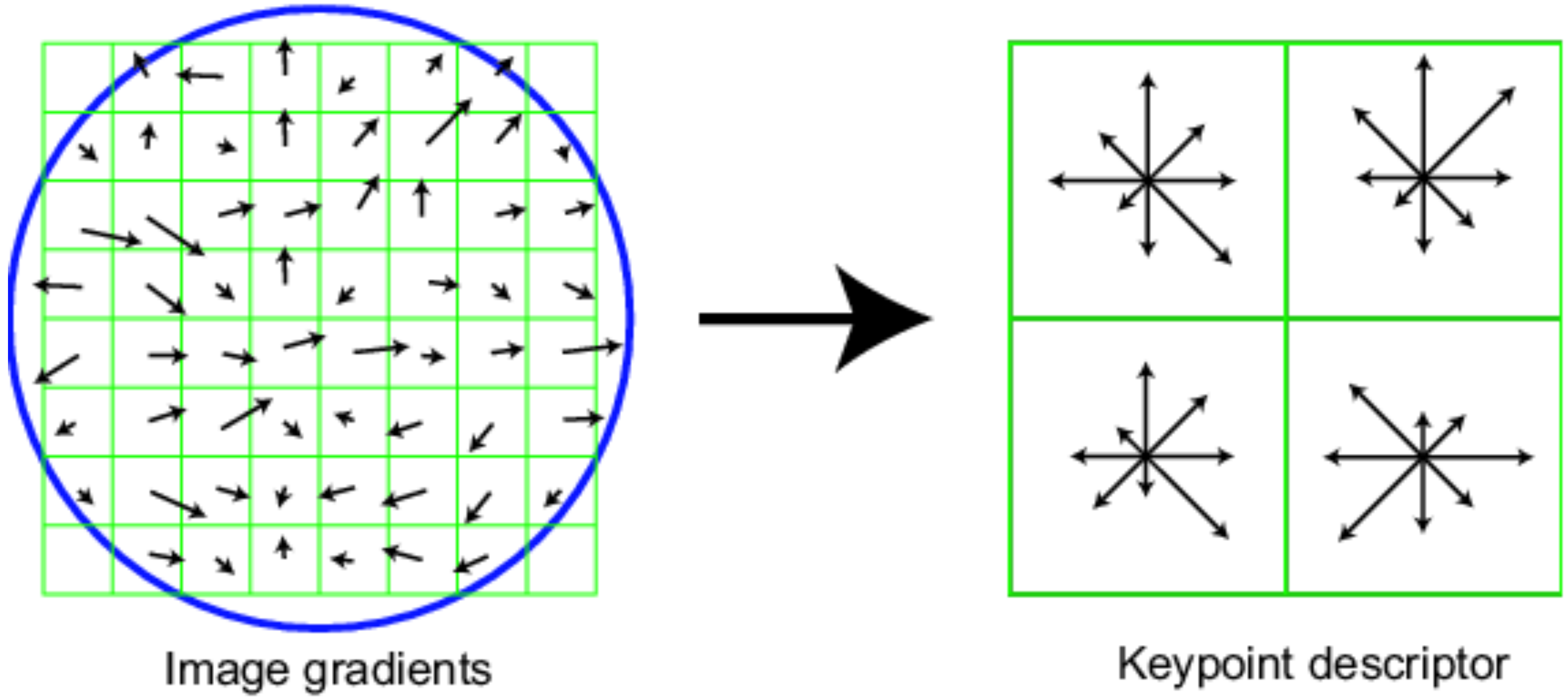


Source: N. Snavely, D. Lowe

Scale Invariant Feature Transform

Create the descriptor:

- Rotation invariance: rotate by “dominant” orientation
- Spatial invariance: spatial pool to 2x2
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor

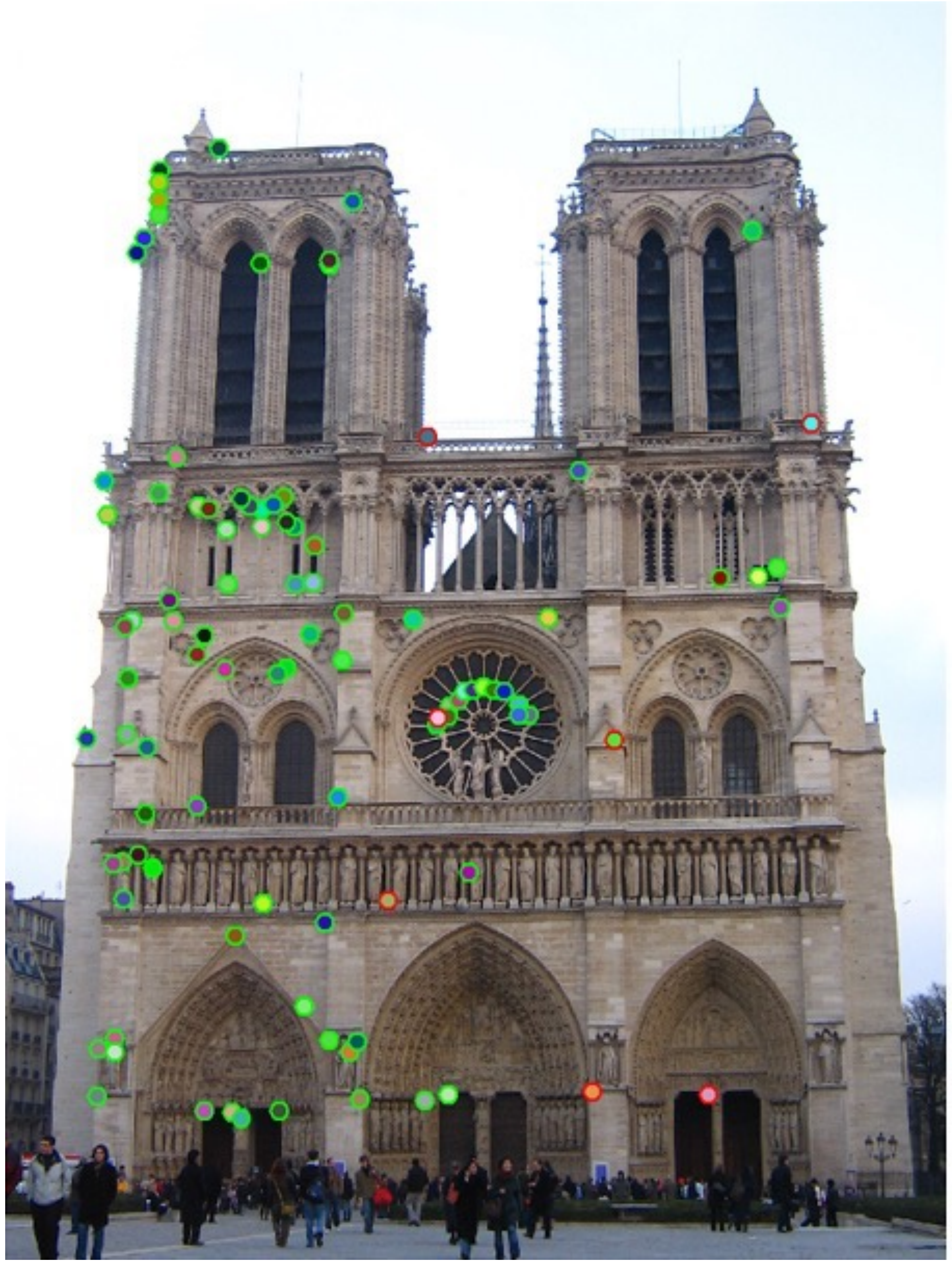


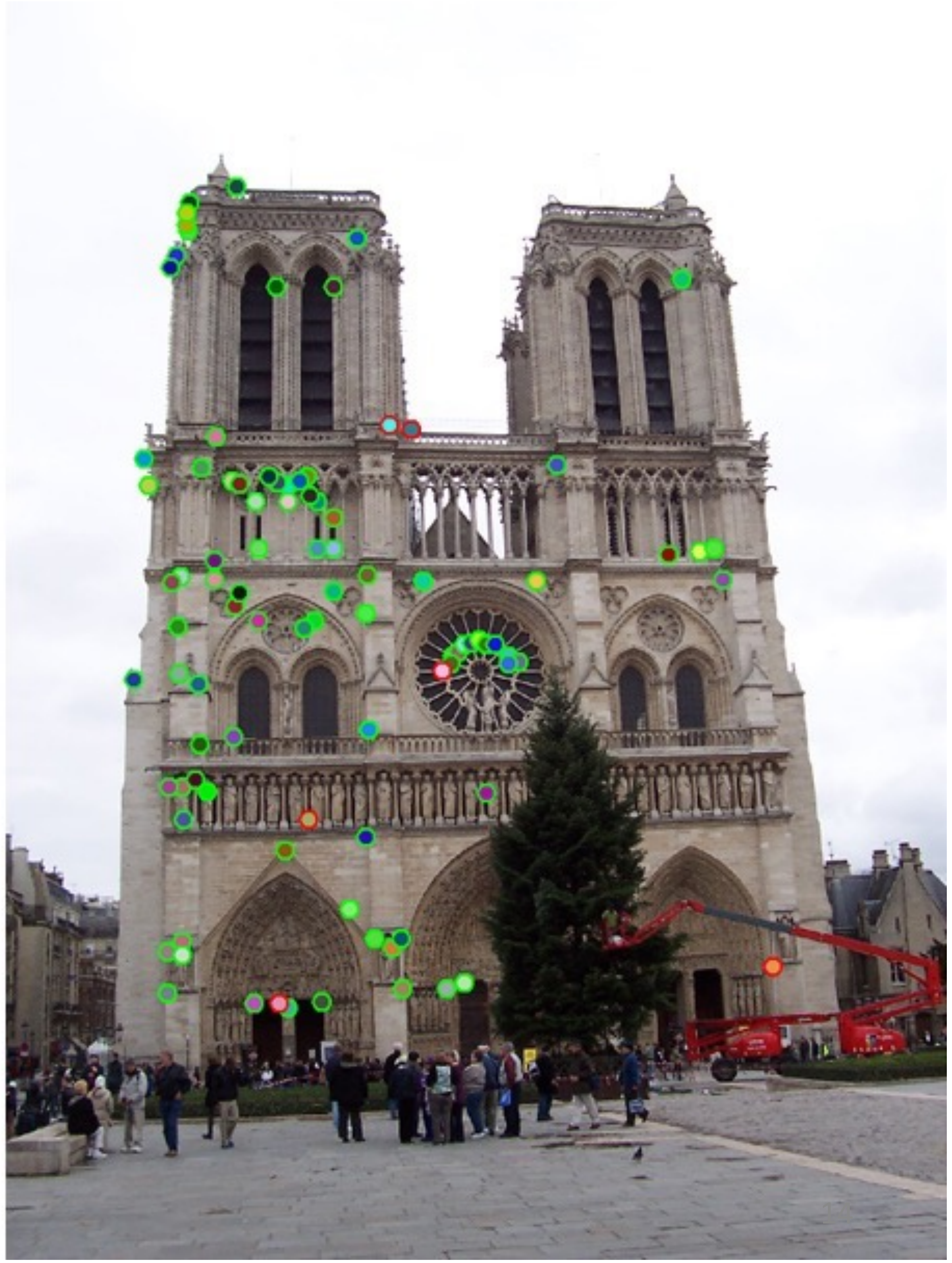
Source: N. Snavely, D. Lowe

SIFT invariances

Source: N. Snavely

Which features match?



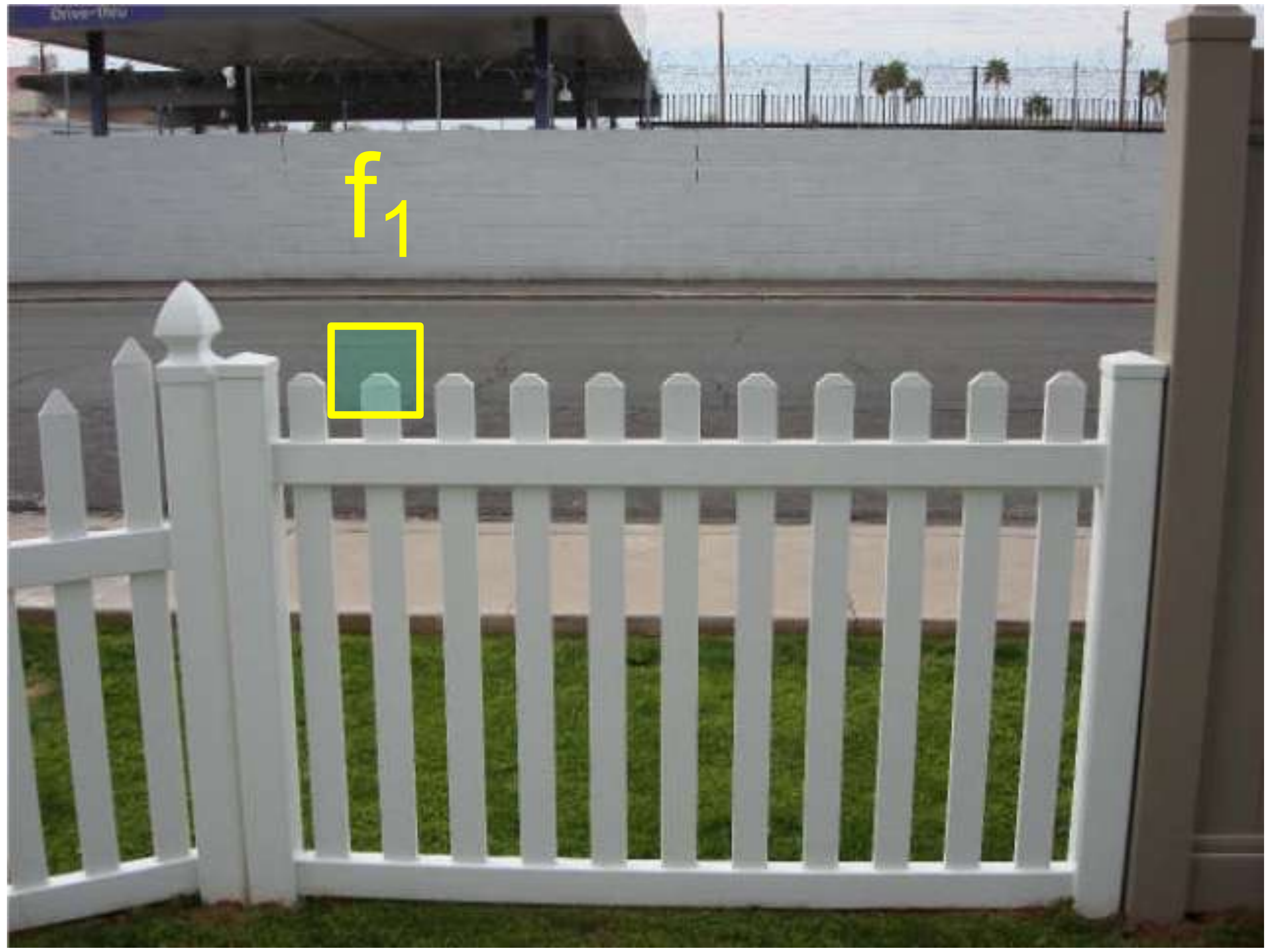


Source: N. Snavely

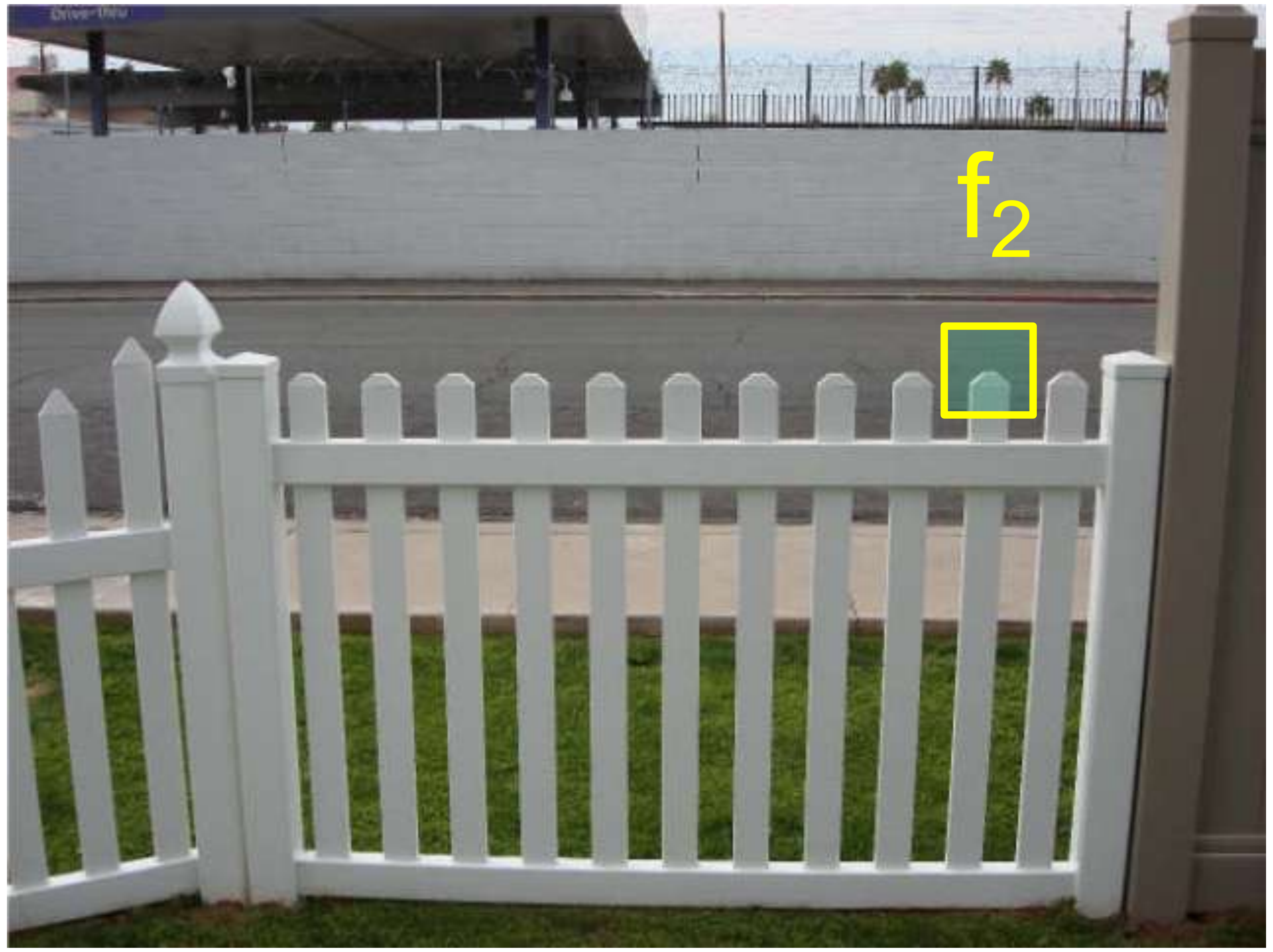
Finding matches

How do we know if two features match?

- Simple approach: are they the nearest neighbor in L_2 distance, $||f_1 - f_2||$
- Can give good scores to ambiguous (incorrect) matches



I_1



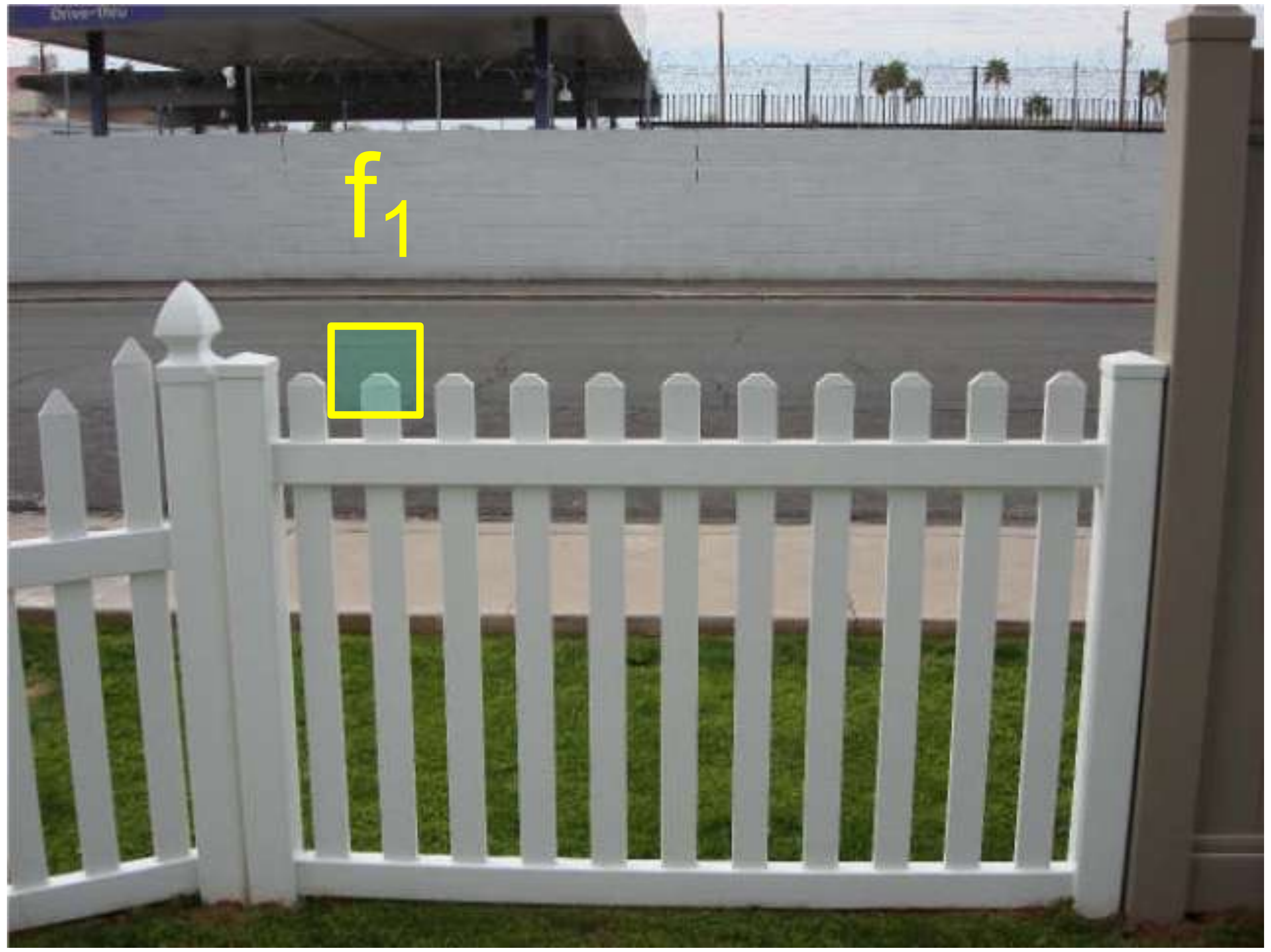
I_2

Source: N. Snavely

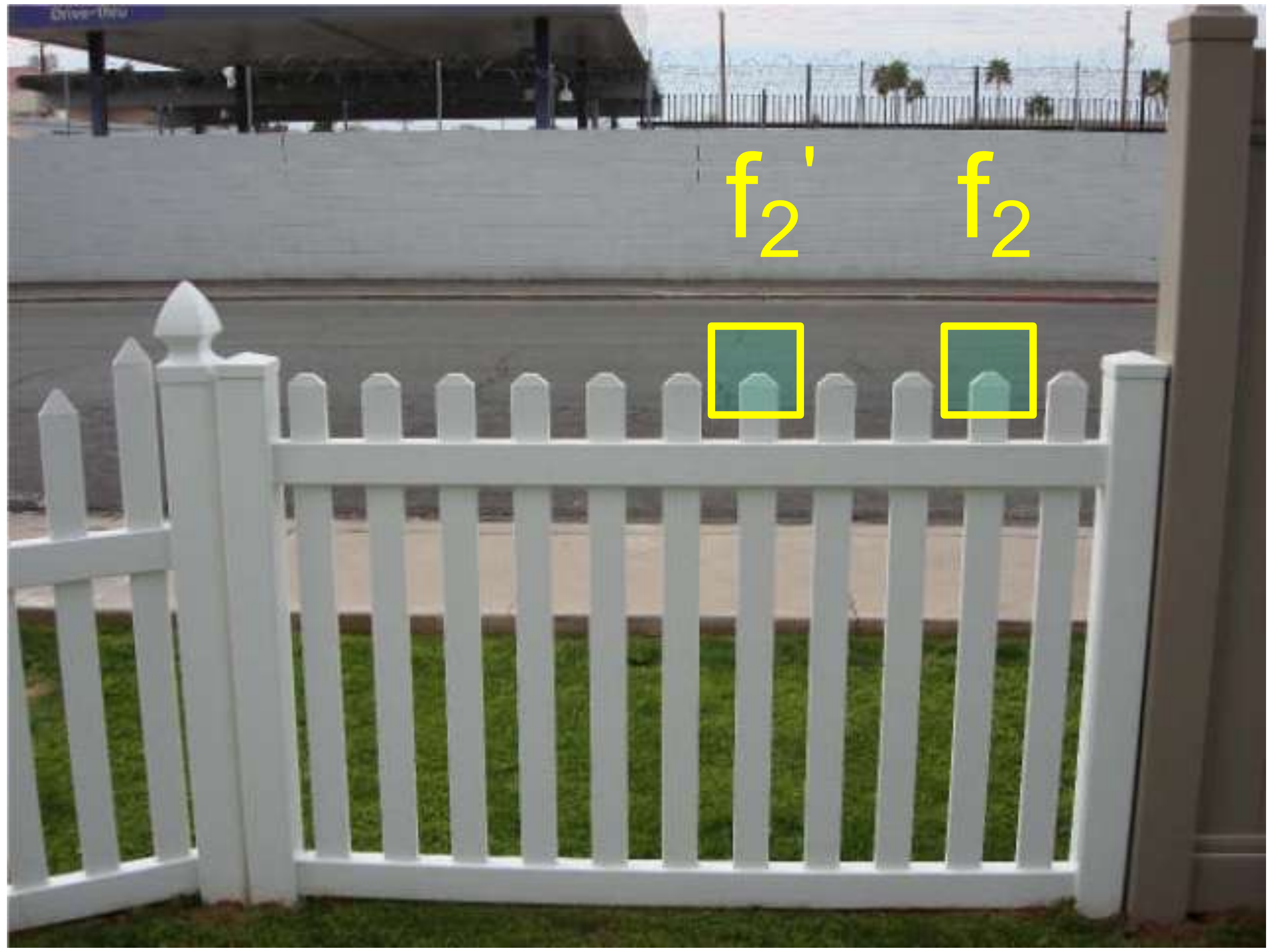
Finding matches

Add extra tests:

- Ratio distance = $||f_1 - f_2|| \,/\, ||f_1 - f_2'||$
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2^{nd} best SSD match to f_1 in I_2
- Forward-backward consistency: f_1 should also be nearest neighbor of f_2



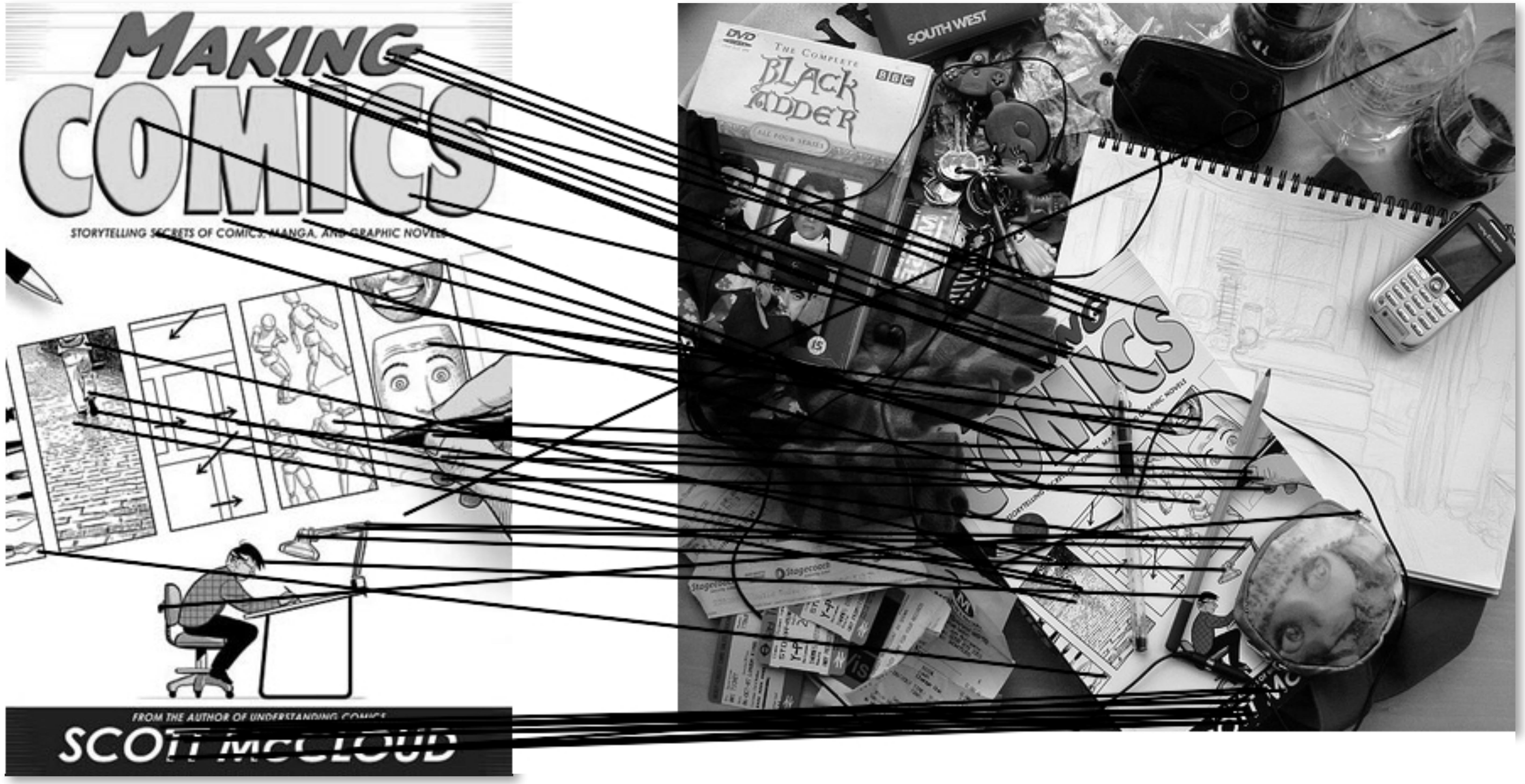
I_1



I_2

Source: N. Snavely

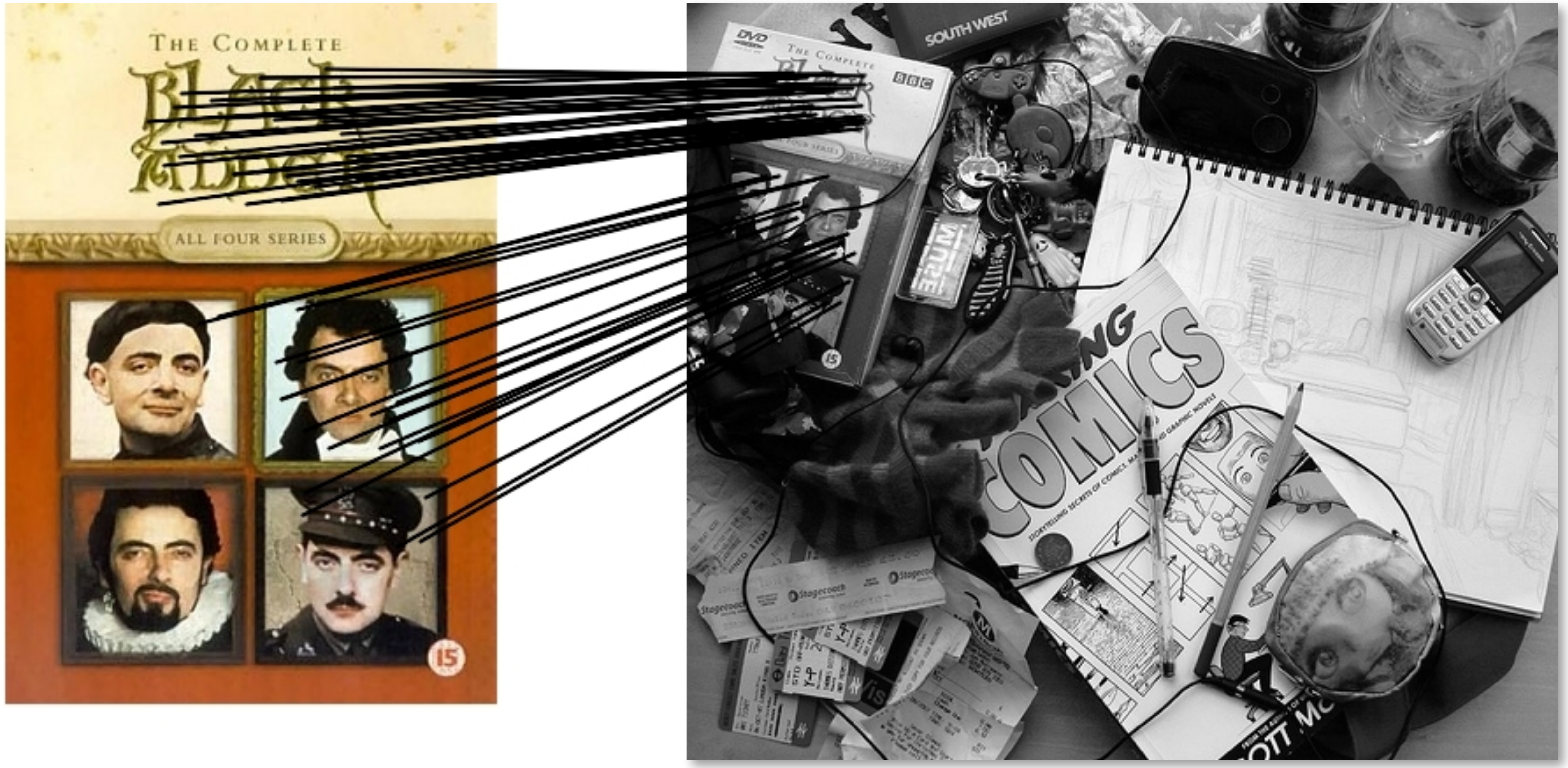
Feature matching example



51 feature matches after ratio test

Source: N. Snavely

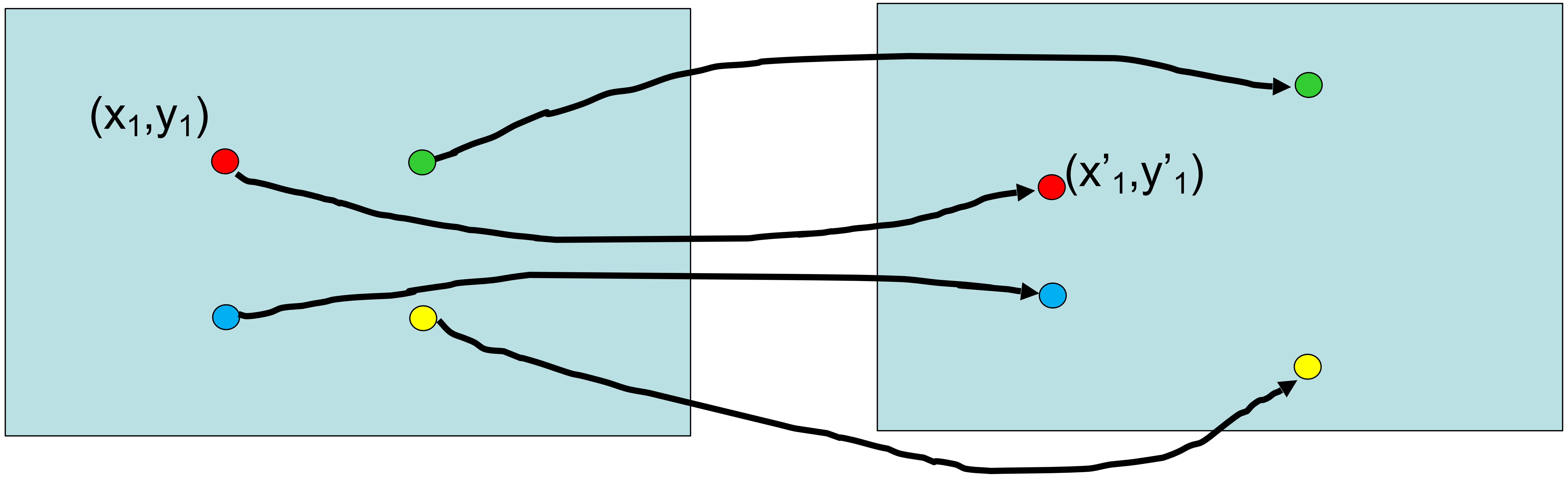
Feature matching example



58 feature matches after ratio test

Source: N. Snavely

From matches to homography



$$\begin{bmatrix} x_1' \\ y_1' \\ w_1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Source: Torralba, Isola, Freeman

From matches to homography

Point in 1st image

Matched point in 2nd

minimize $$J(H) = \sum_i ||f_H(p_i) - p_i'||^2$$

where $f_H(p_i) = Hp_i/(H_3^T p_i)$ applies homography

- Plug into nonlinear least squares solver and solve!
- Can also use robust loss (e.g. L_1)
- Can be slow

Direct linear transform

$$\begin{bmatrix} x_1' \\ y_1' \\ w_1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Going to heterogeneous coordinates:

$$x_1' = \frac{ax_1 + by_1 + c}{gx_1 + hy_1 + i}$$

$$y_1' = \frac{dx_1 + ey_1 + f}{gx_1 + hy_1 + i}$$

Re-arranging the terms:

$$gx_1x'_1 + hy_1x'_1 + ix_1 = ax_1 + by_1 + c$$

$$gx_1y'_1 + hy_1y'_1 + ix_1 = dx_1 + ey_1 + f$$

Source: Torralba, Freeman, Isola

Direct linear transform

$$g x_1 x'_1 + h y_1 x'_1 + i x_1 = a x_1 + b y_1 + c$$

$$g x_1 y'_1 + h y_1 y'_1 + i x_1 = d x_1 + e y_1 + f$$

Re-arranging the terms:

$$g x_1 x'_1 + h y_1 x'_1 + i x'_1 - a x_1 - b y_1 - c = 0$$

$$g x_1 y'_1 + h y_1 y'_1 + i y'_1 - d x_1 - e y_1 - f = 0$$

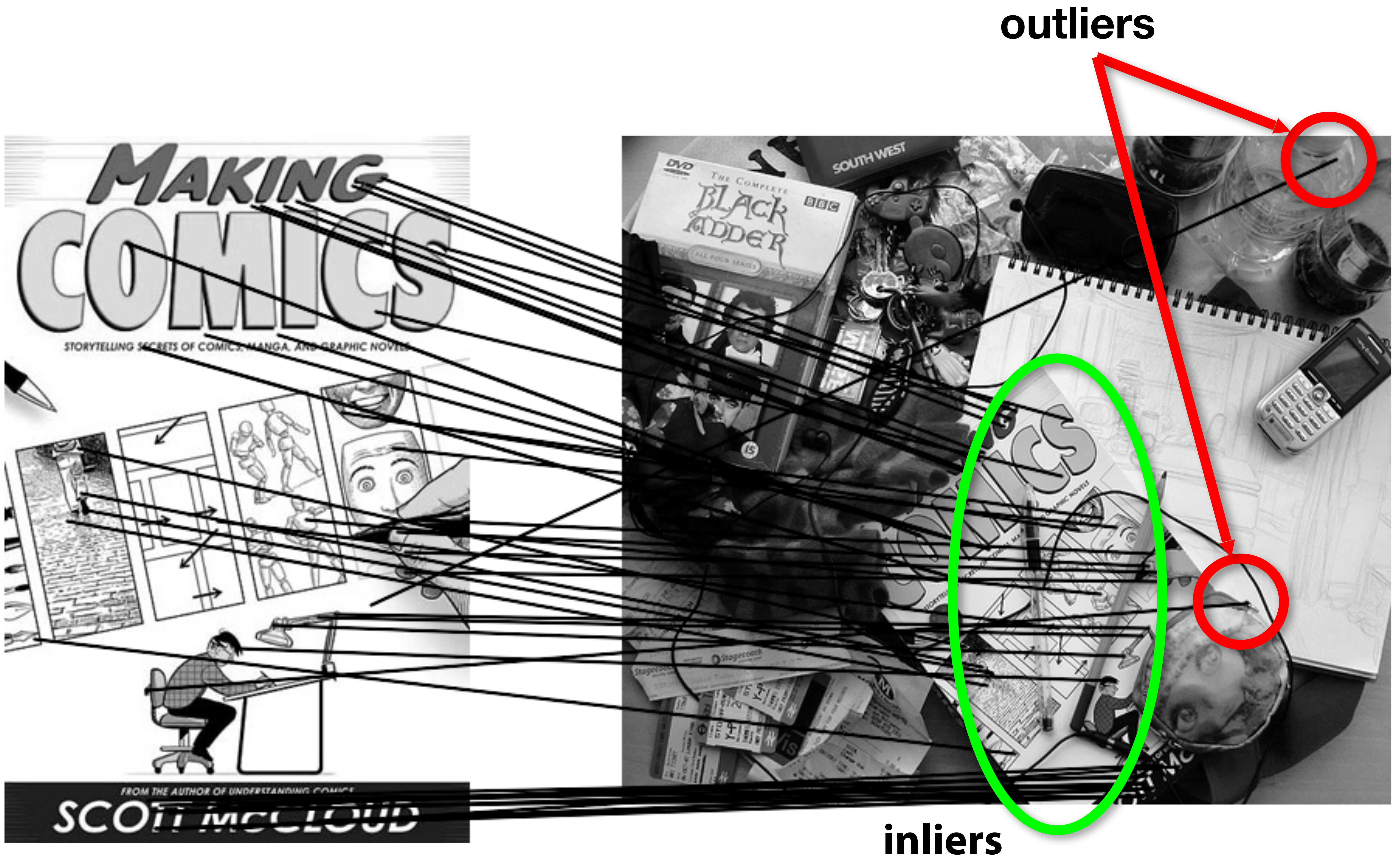
In matrix form. Can solve using Singular Value Decomposition (SVD).

$$\begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1 x'_1 & y_1 x'_1 & x'_1 \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1 y'_1 & y_1 y'_1 & y'_1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Fast to solve (but not using "right" loss function). Uses an algebraic trick.
Often used in practice for initial solutions!

Source: Torralba, Freeman, Isola

Outliers

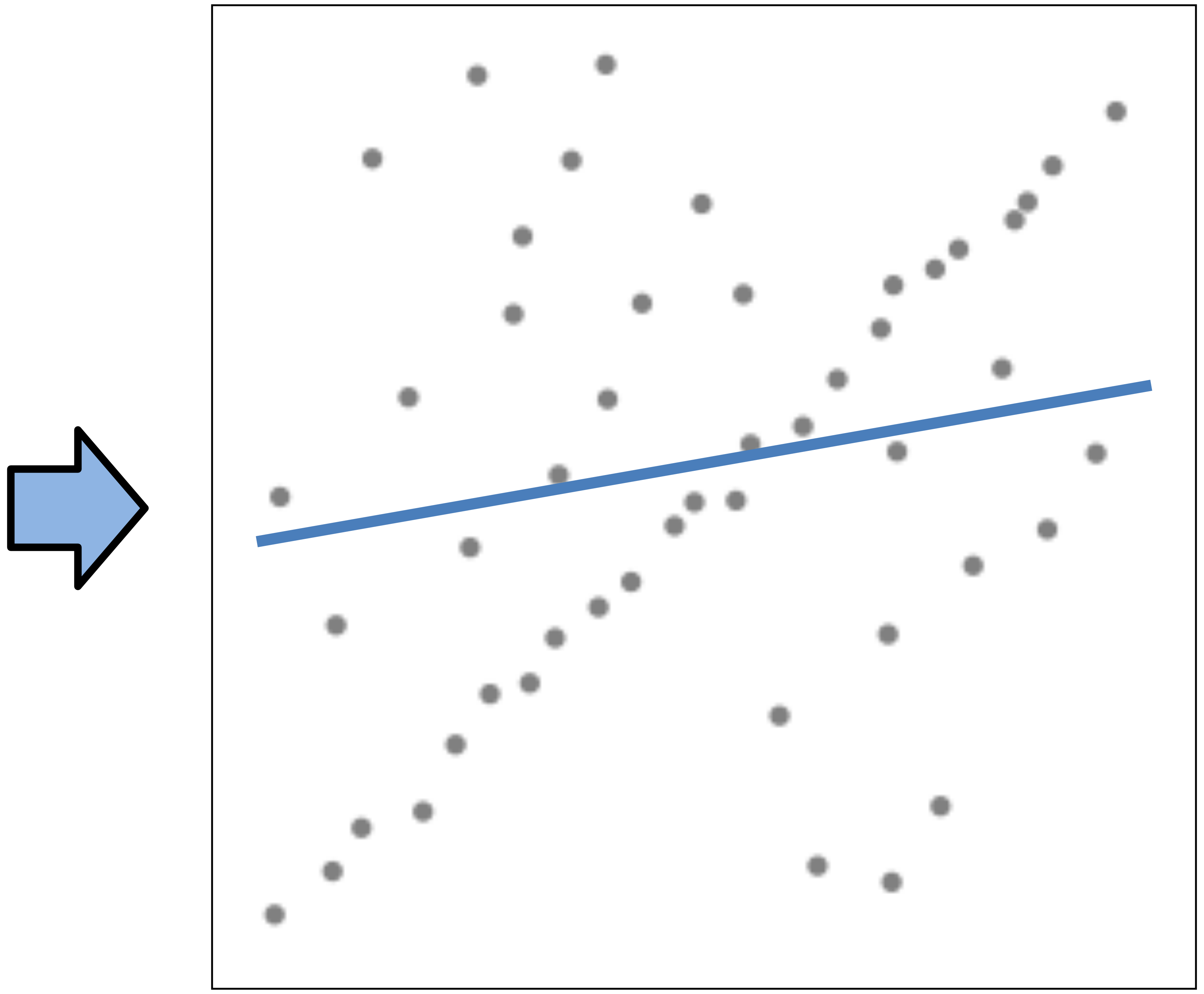


Source: N. Snavely

Robustness

- Let's consider the problem of linear regression

Problem: Fit a line to these data points

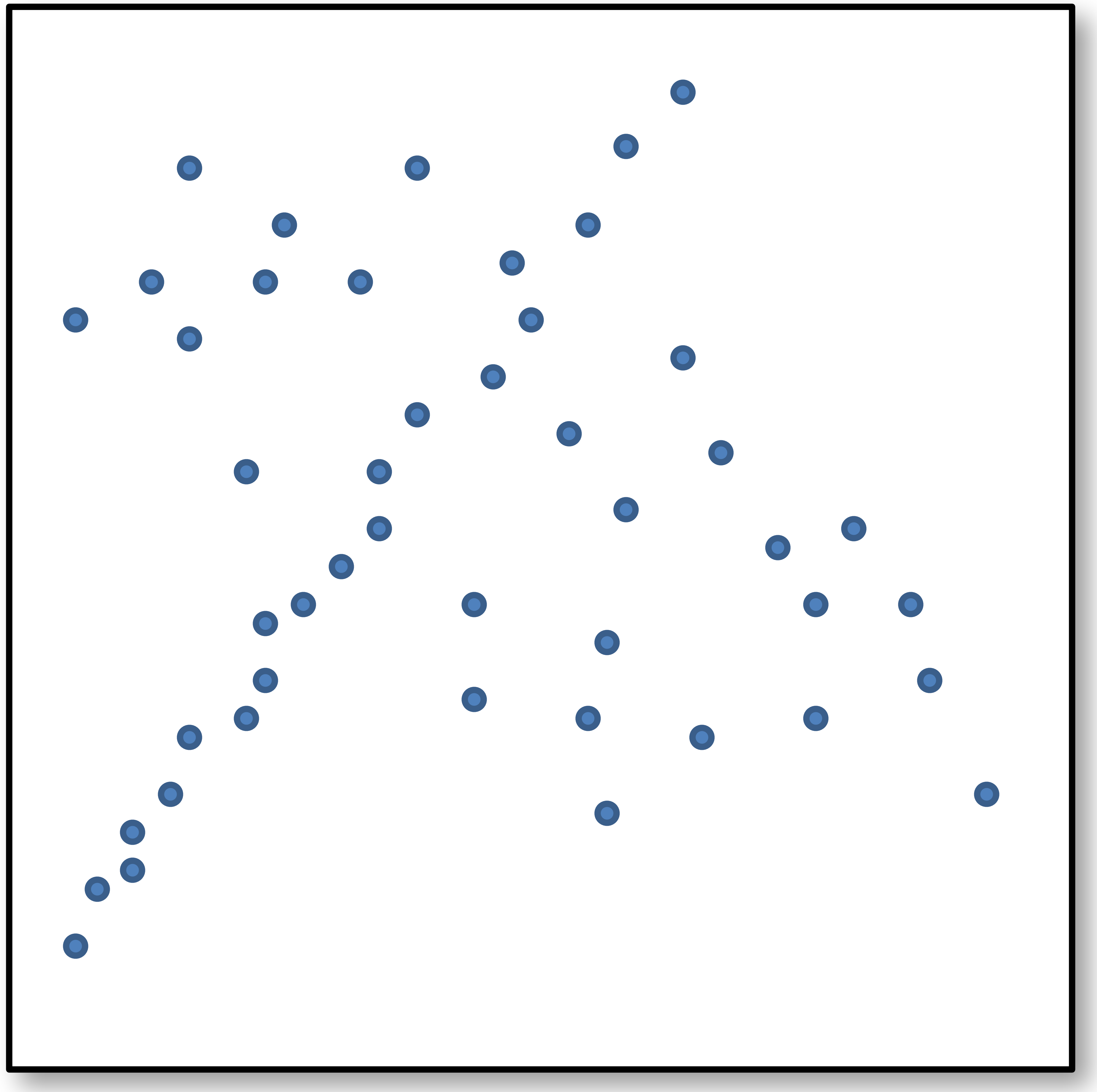


Least squares fit

- How can we fix this?

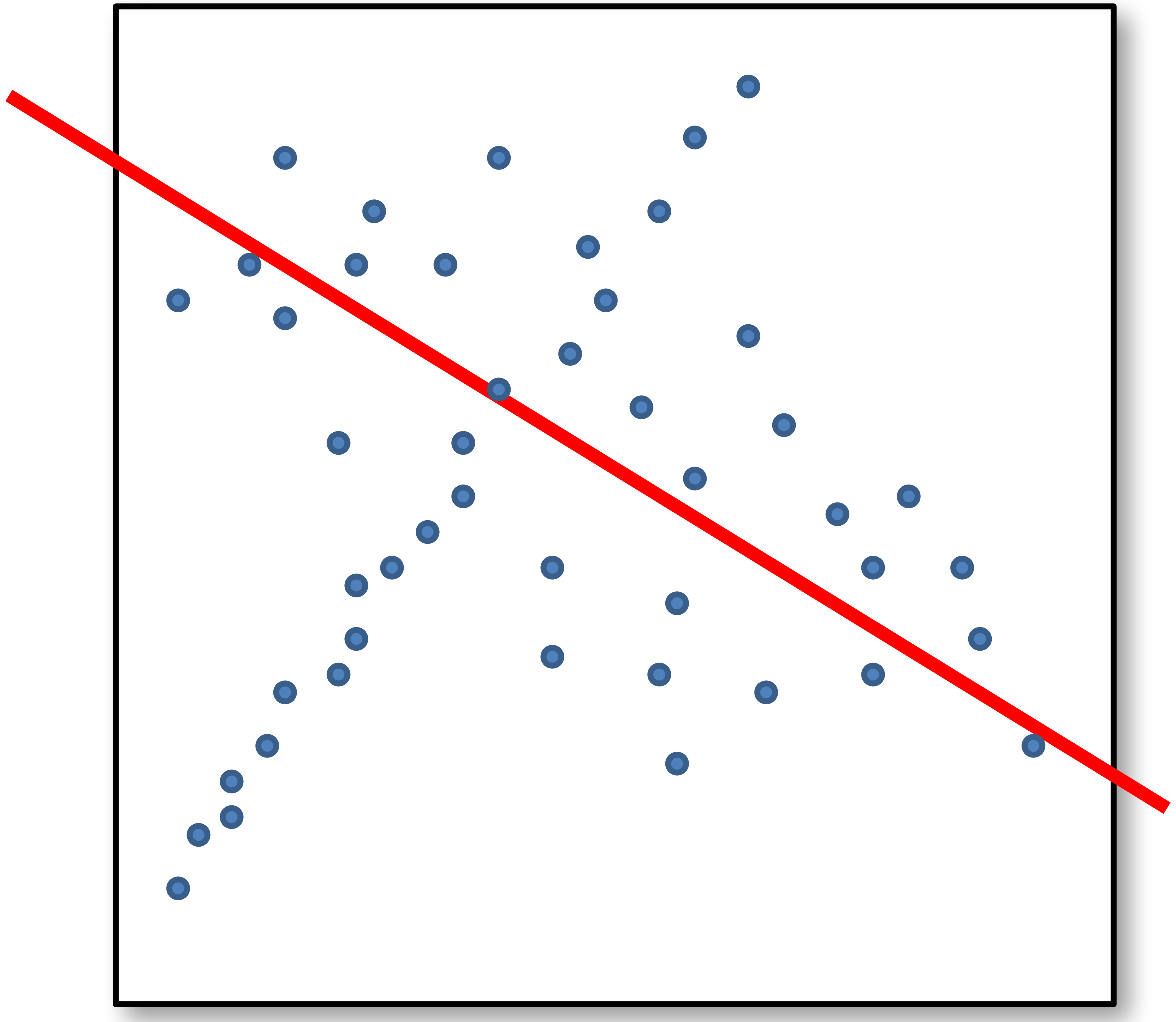
Source: N. Snavely

Counting inliers



Source: N. Snavely

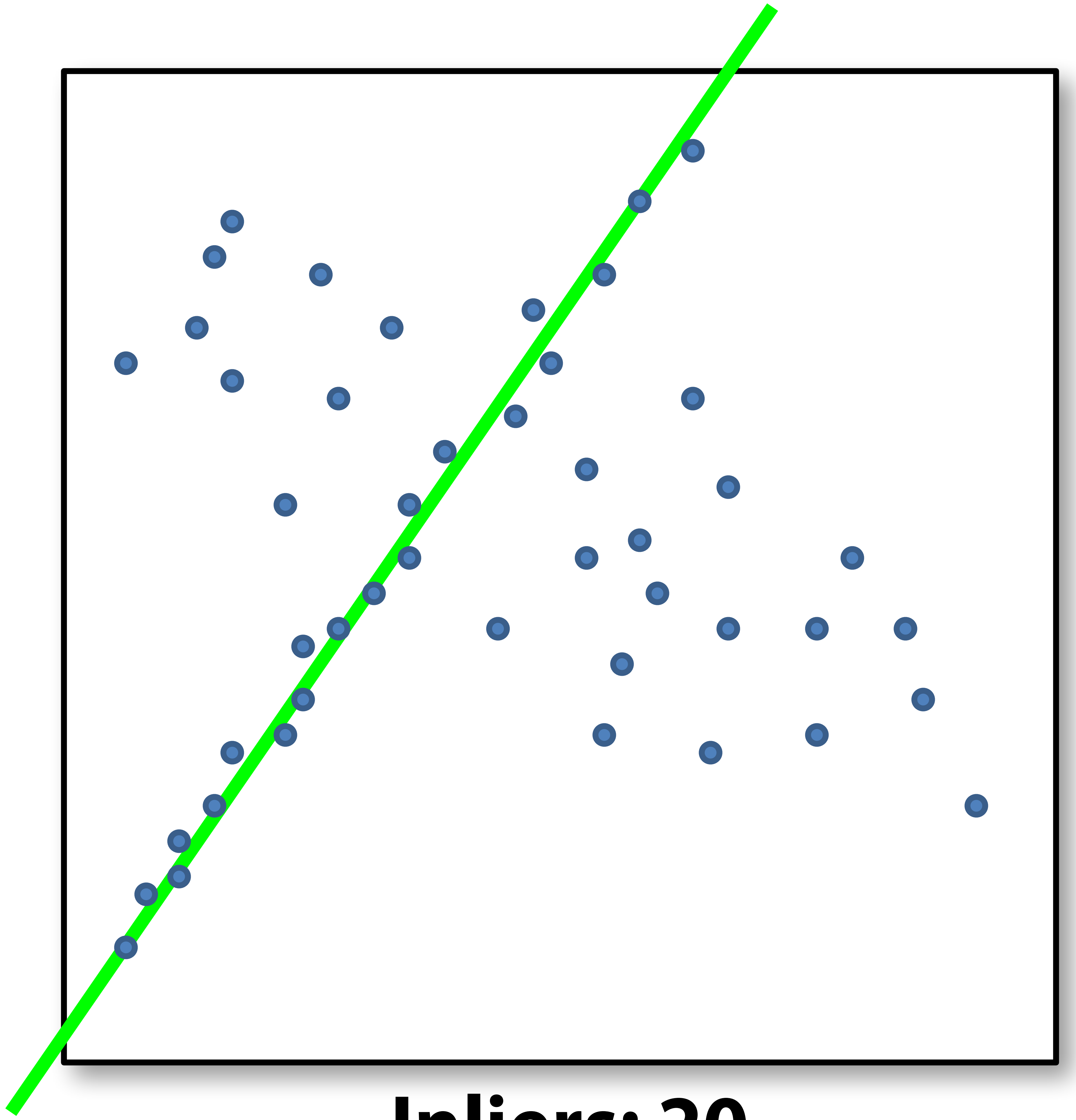
Counting inliers



Inliers: 3

Source: N. Snavely

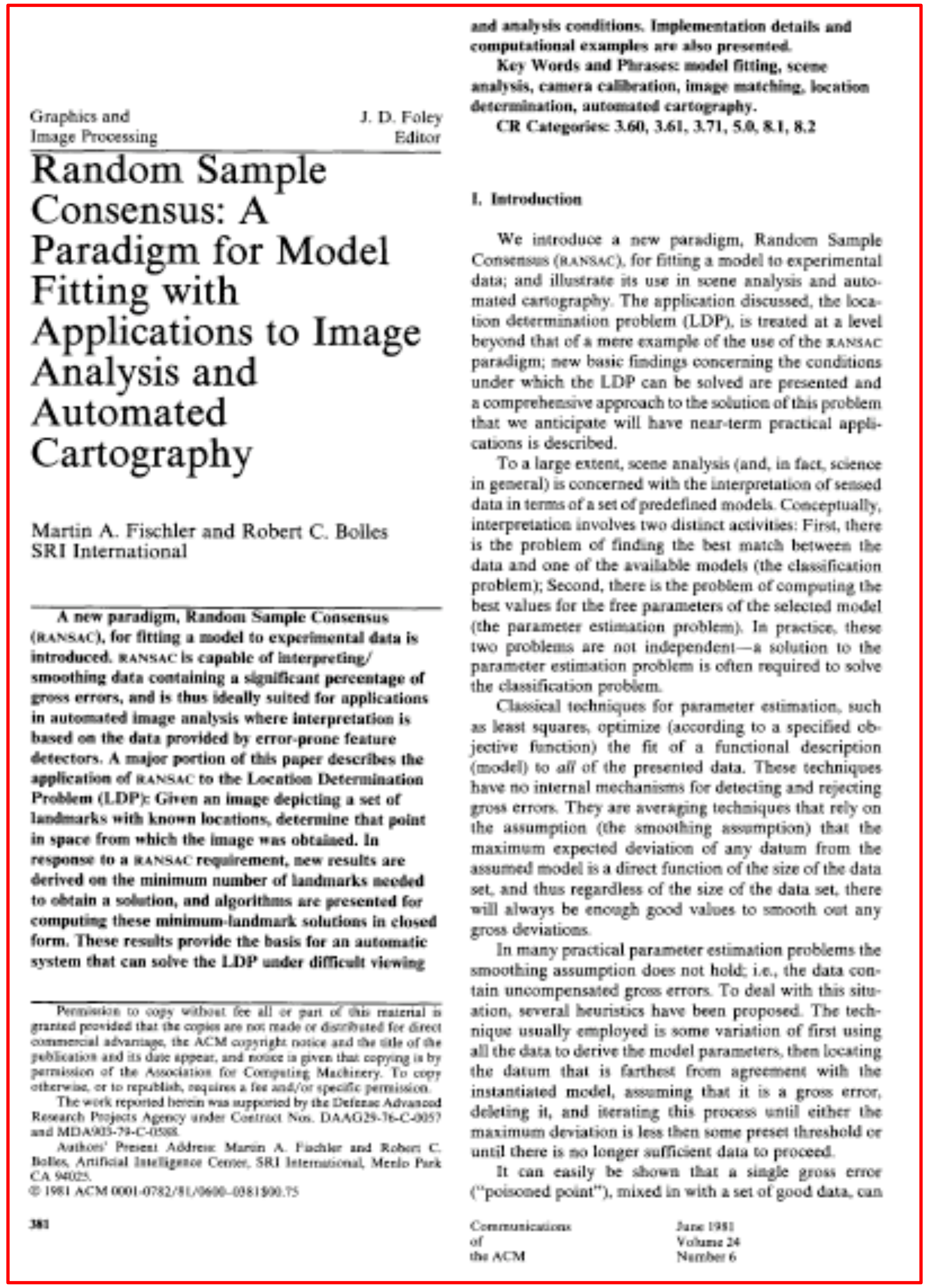
Counting inliers



Inliers: 20

Source: N. Snavely

- 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.



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

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

The work reported herein was supported by the Defense Advanced Research Projects Agency under Contract Nos. DAAG29-76-C-0057 and MDA903-79-C-0588.

Authors' Present Address: Martin A. Fischler and Robert C. Bolles, Artificial Intelligence Center, SRI International, Menlo Park CA 94025.

© 1981 ACM 0001-0782/81/0600-0381$00.75

I. 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

381

Communications of the ACM — June 1981, Volume 24, Number 6

RANSAC: random sample consensus

RANSAC loop (for N iterations):

- Select four feature pairs (at random)
- Compute homography H
- Count **inliers** where $||p_i' - \boldsymbol{H}\, p_i|| < \varepsilon$

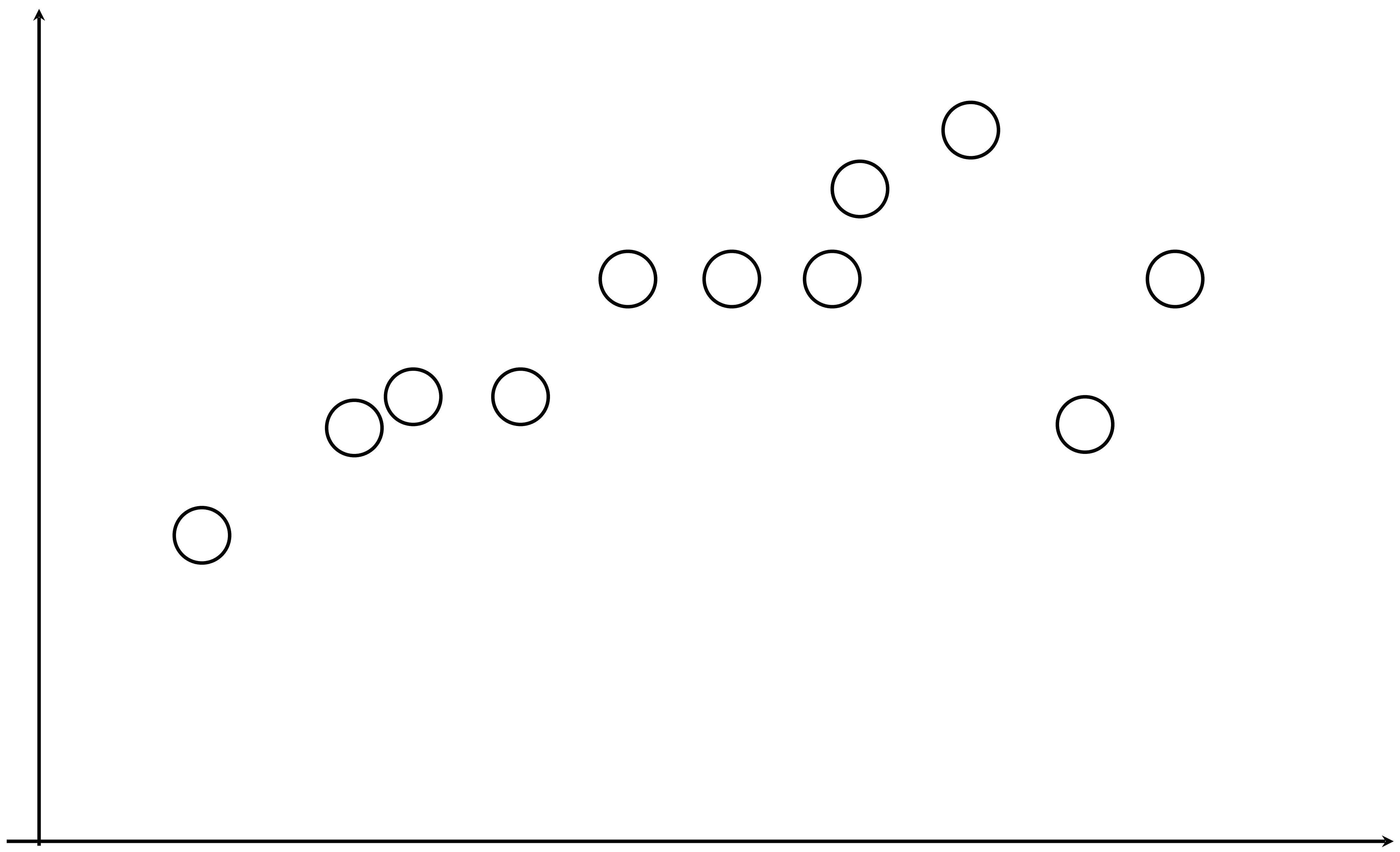
Afterwards:

- Choose largest set of inliers
- Recompute $\boldsymbol{H}$ using only those inliers (often using high-quality nonlinear least squares)

Source: Torralba, Freeman, Isola

Simple example: fit a line

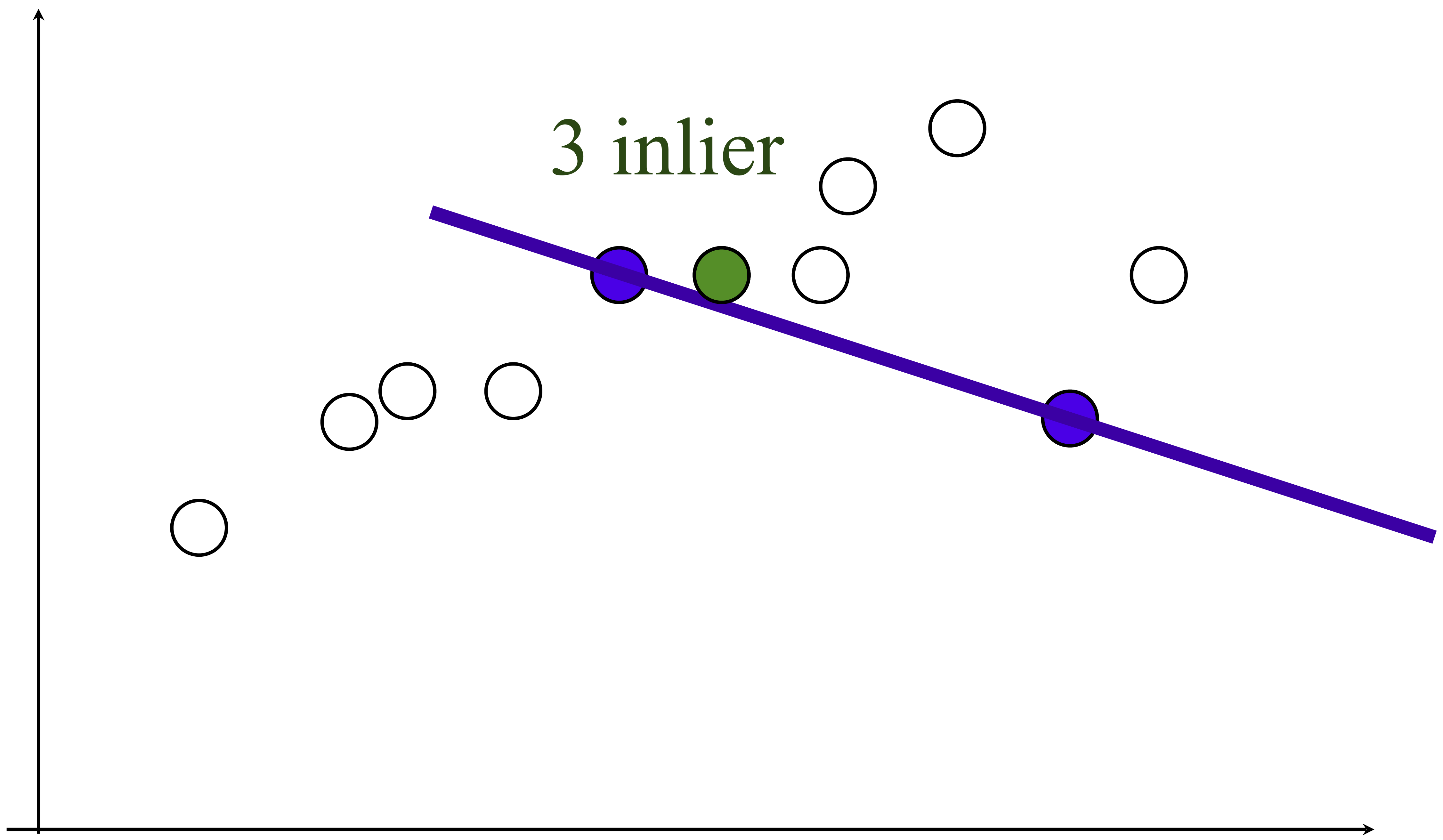
- Rather than homography H (8 numbers) fit y=ax+b (2 numbers a, b) to 2D pairs



Source: Torralba, Freeman, Isola

Simple example: fit a line

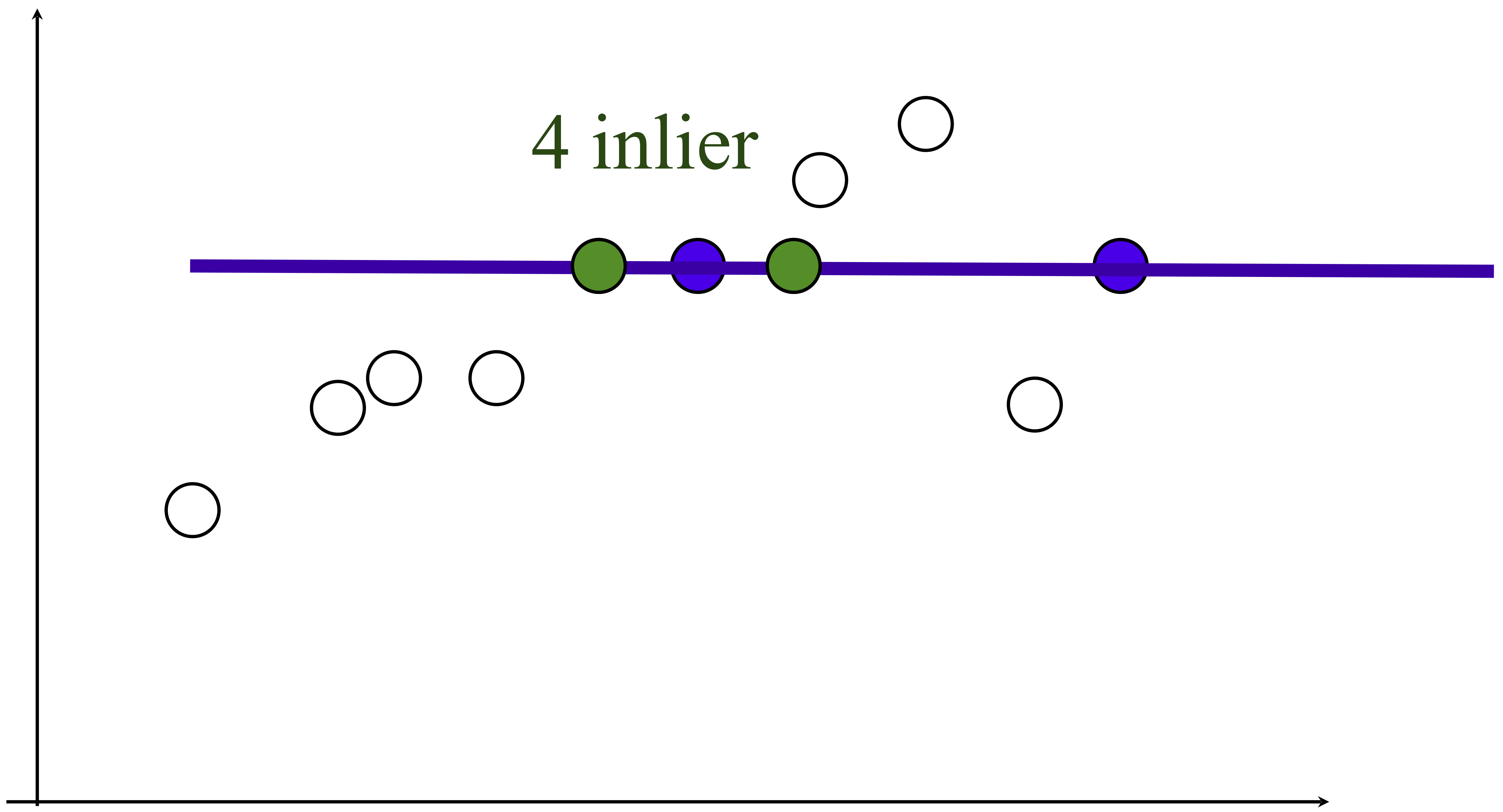
- Pick 2 points
- Fit line
- Count inliers



Source: Torralba, Freeman, Isola

Simple example: fit a line

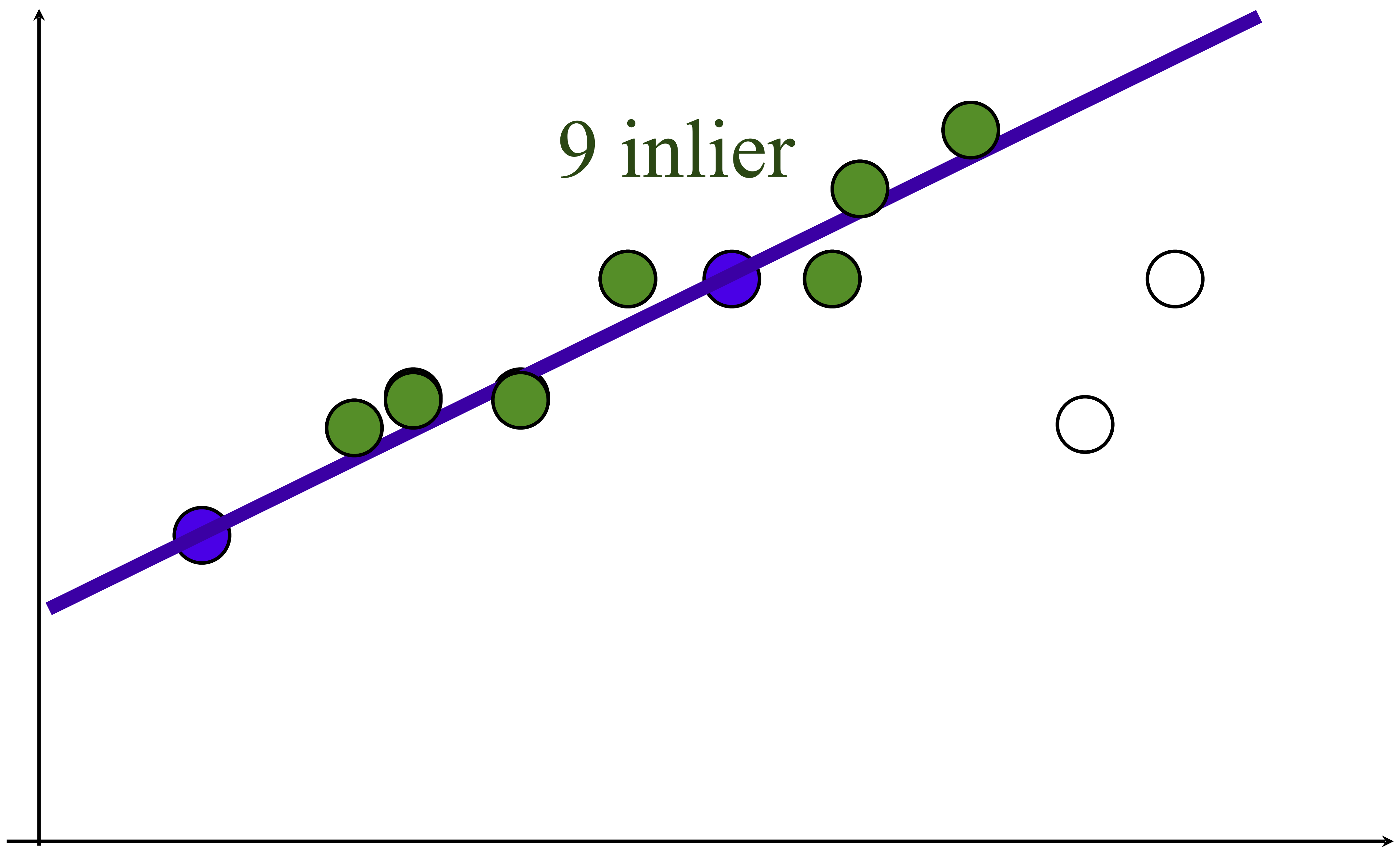
- Pick 2 points
- Fit line
- Count inliers



Source: Torralba, Freeman, Isola

Simple example: fit a line

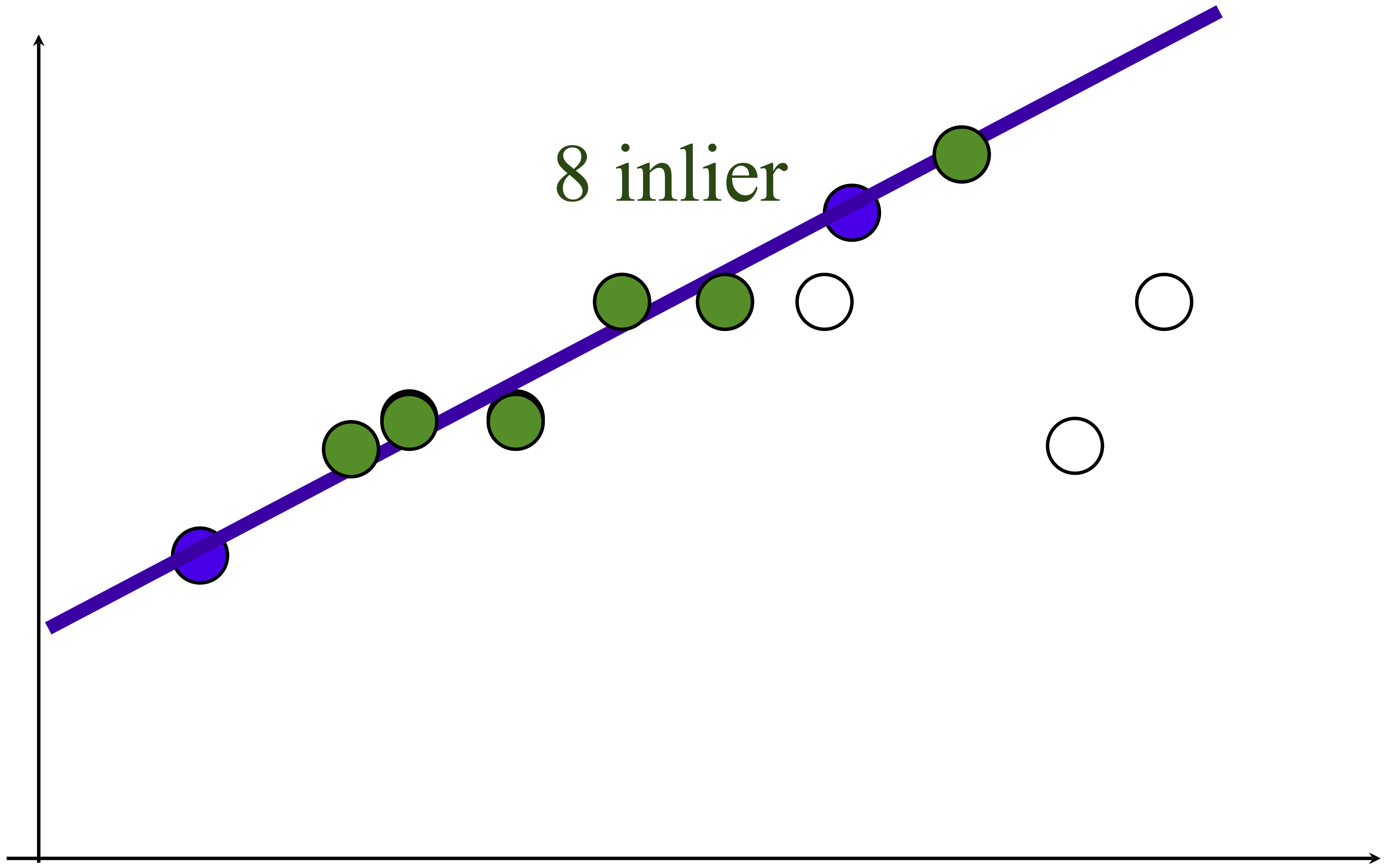
- Pick 2 points
- Fit line
- Count inliers



Source: Torralba, Freeman, Isola

Simple example: fit a line

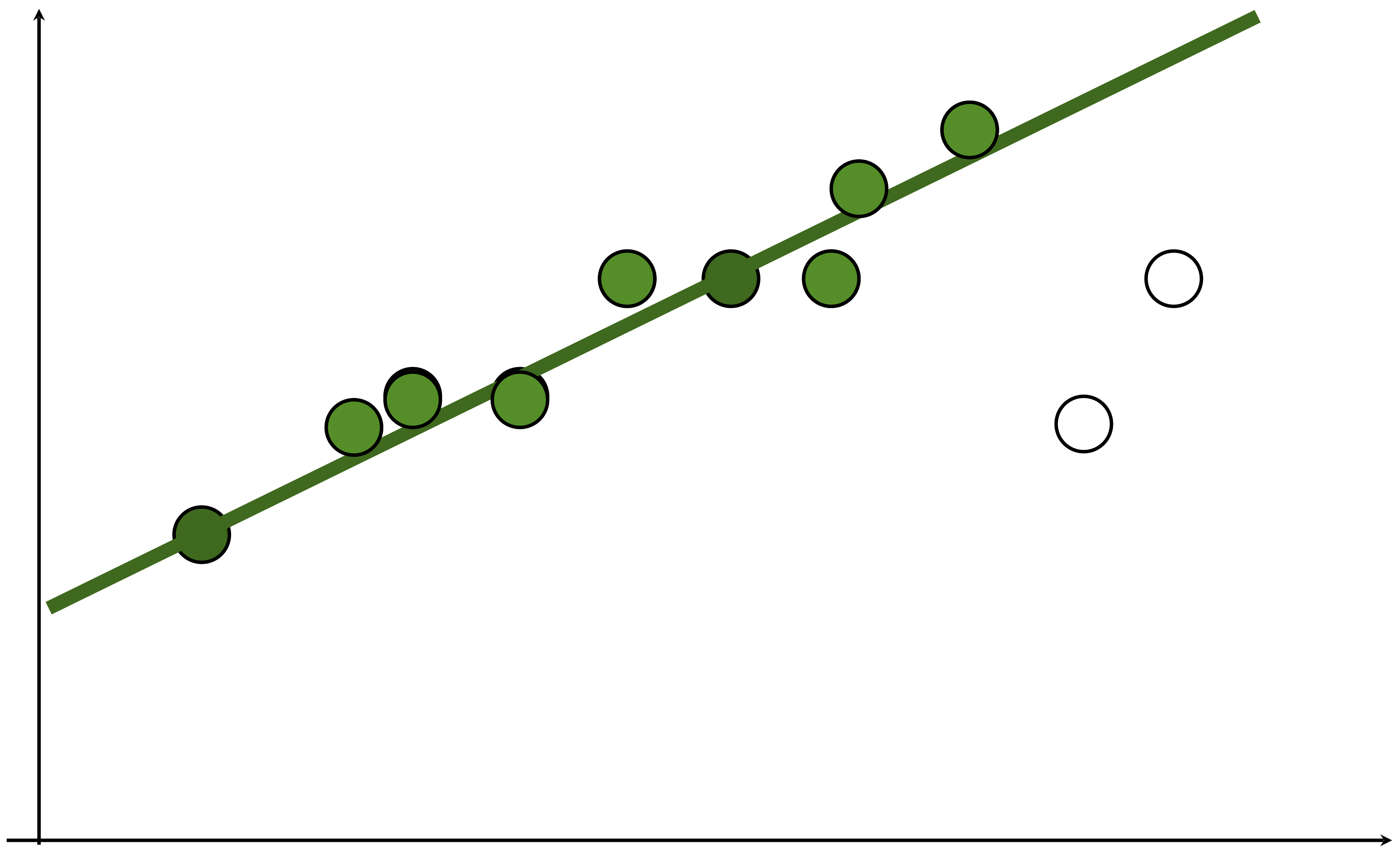
- Pick 2 points
- Fit line
- Count inliers



Source: Torralba, Freeman, Isola

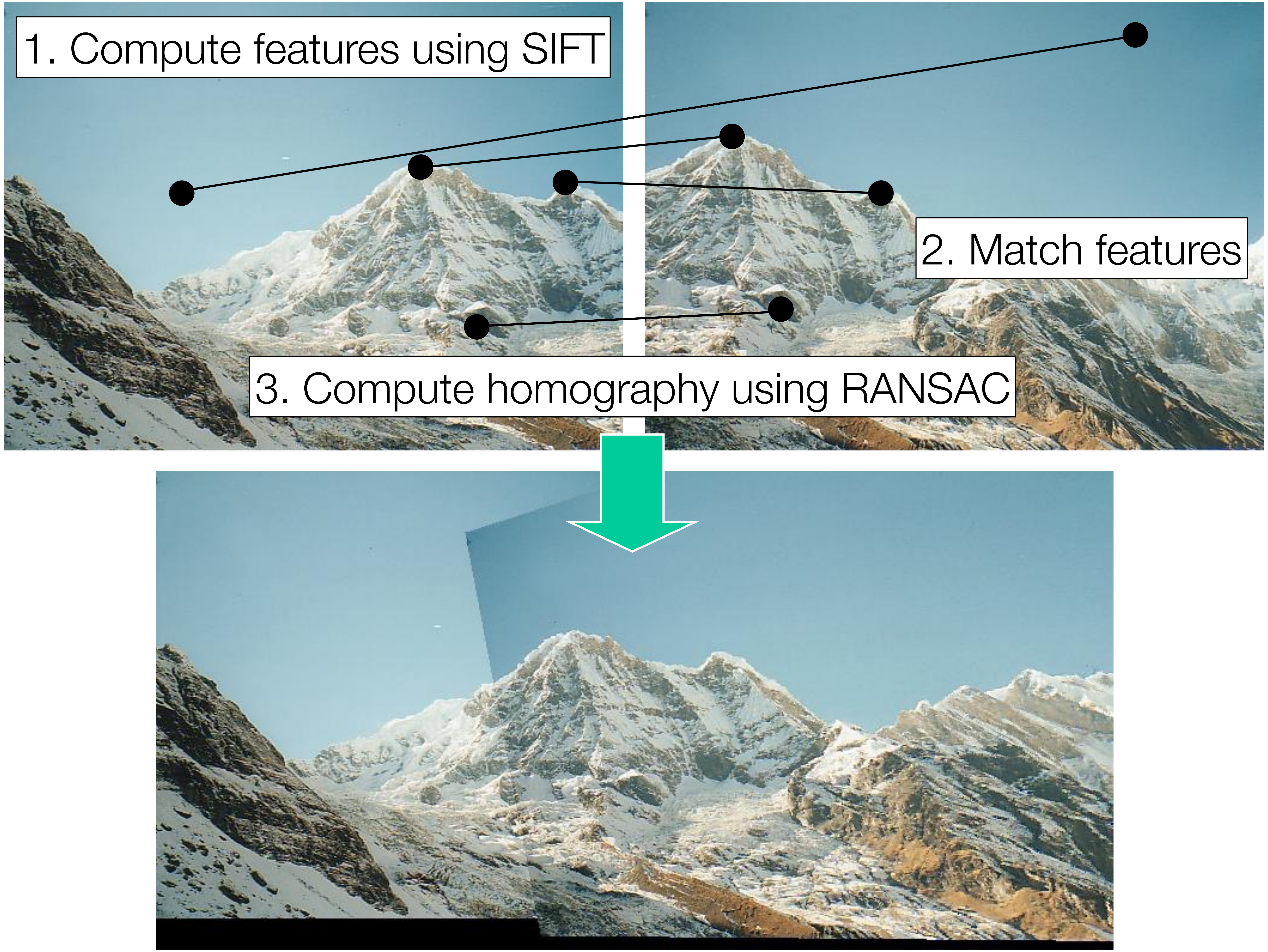
Simple example: fit a line

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



Source: Torralba, Freeman, Isola

Warping with a homography



Source: N. Snavely

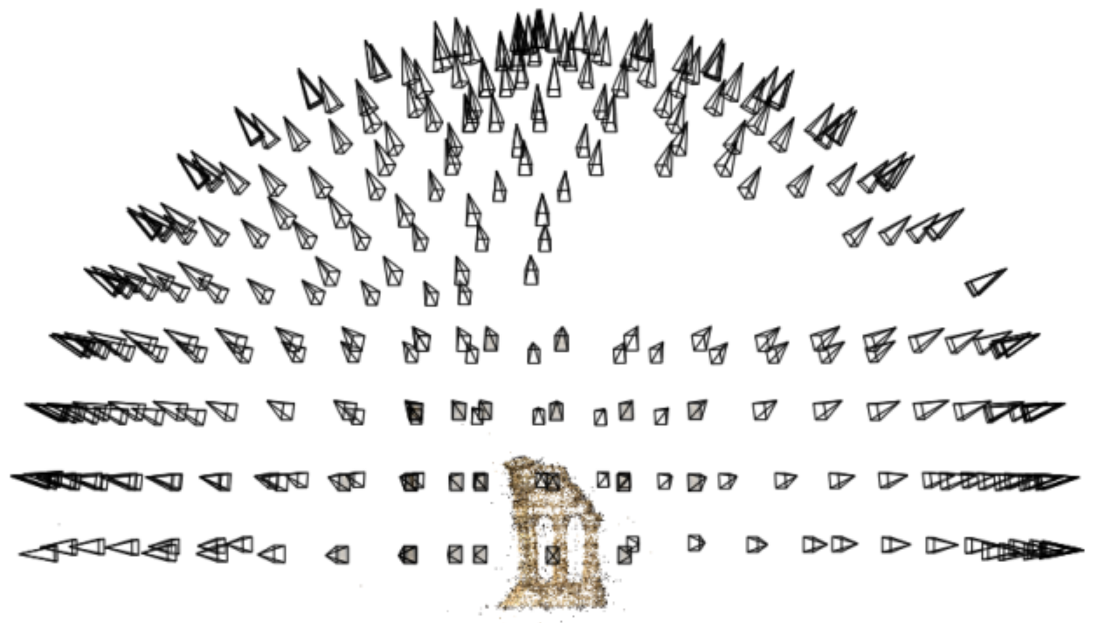
Estimating 3D structure

- Given many images, how can we
  a) figure out where they were all taken from?
  b) build a 3D model of the scene?

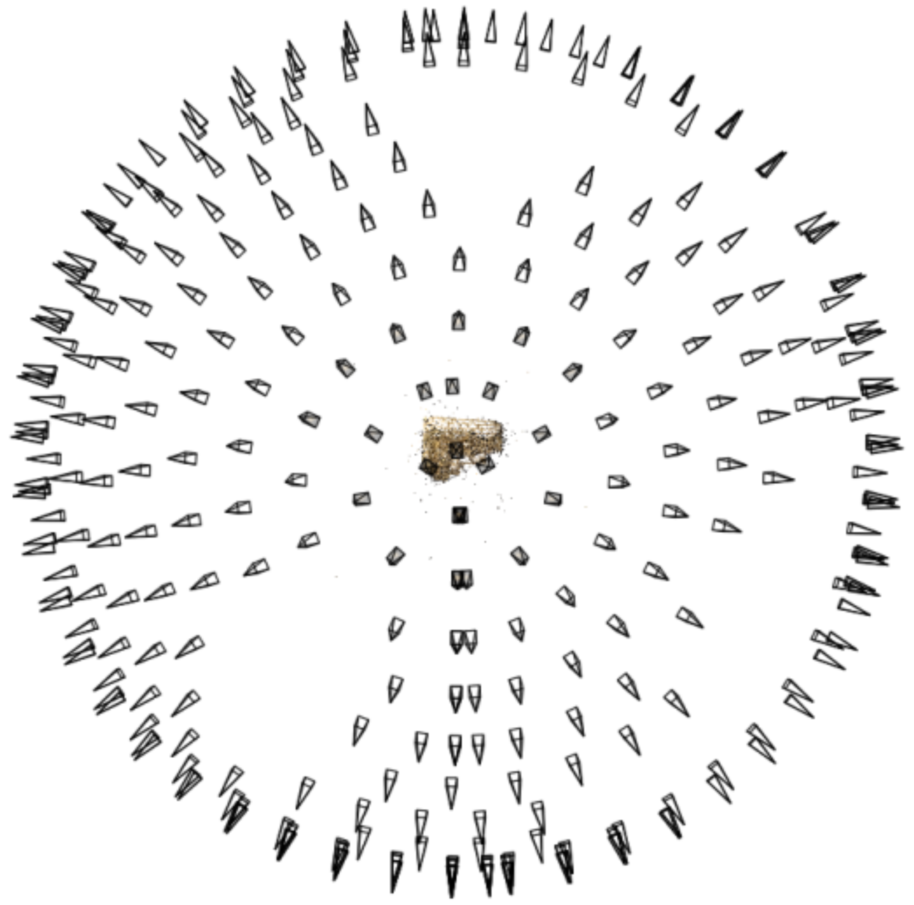
This is the **structure from motion** problem

Source: N. Snavely

Structure from motion



Reconstruction (side)



(top)

- Input: images with points in correspondence $p_{i,j} = (u_{i,j}, v_{i,j})$

- Output
  - structure: 3D location $\mathbf{x}_i$ for each point p_i
  - motion: camera parameters $\mathbf{R}_j$, $\mathbf{t}_j$ possibly $\mathbf{K}_j$

- Objective function: minimize reprojection error

Source: N. Snavely

Camera calibration & triangulation

- Suppose we know 3D points
  – And have matches between these points and an image
  – Computing camera parameters similar to homography estimation

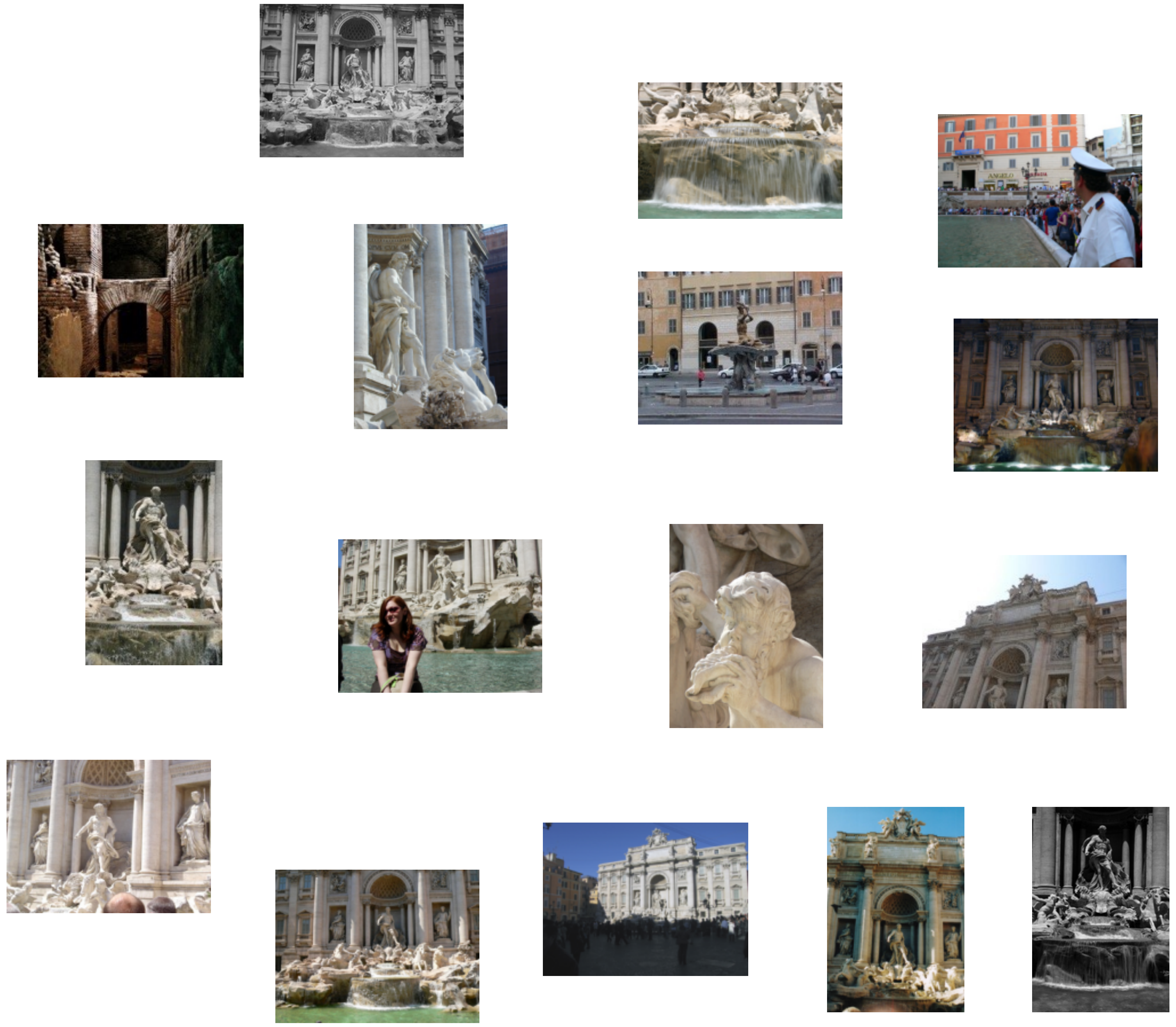
- Suppose we have know camera parameters, each of which observes a point
  – How can we compute the 3D location of that point?

- Seems like a chicken-and-egg problem, but in SfM we can solve both at once

Source: N. Snavely

Feature detection

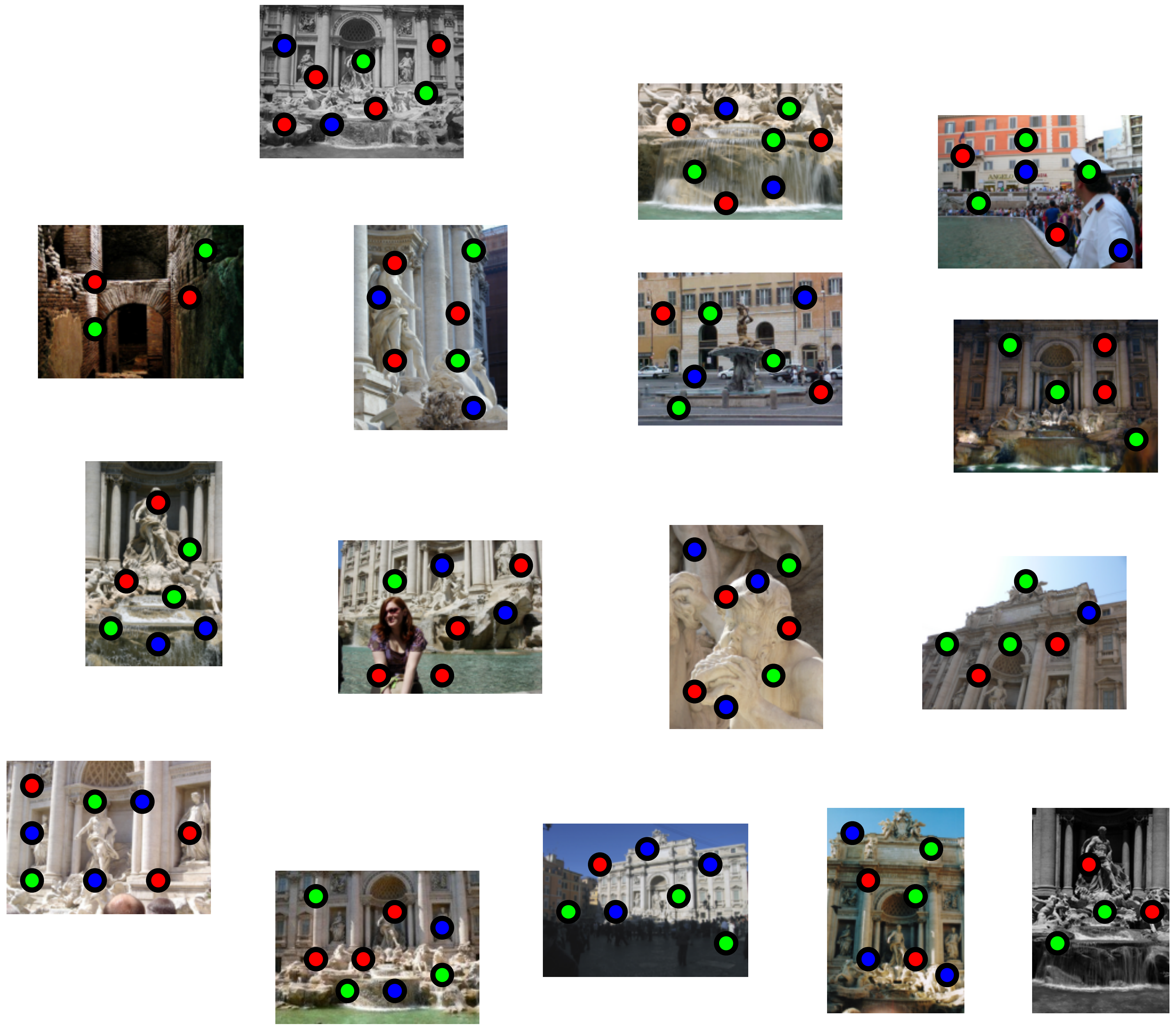
Detect features using SIFT [Lowe, IJCV 2004]



Source: N. Snavely

Feature detection

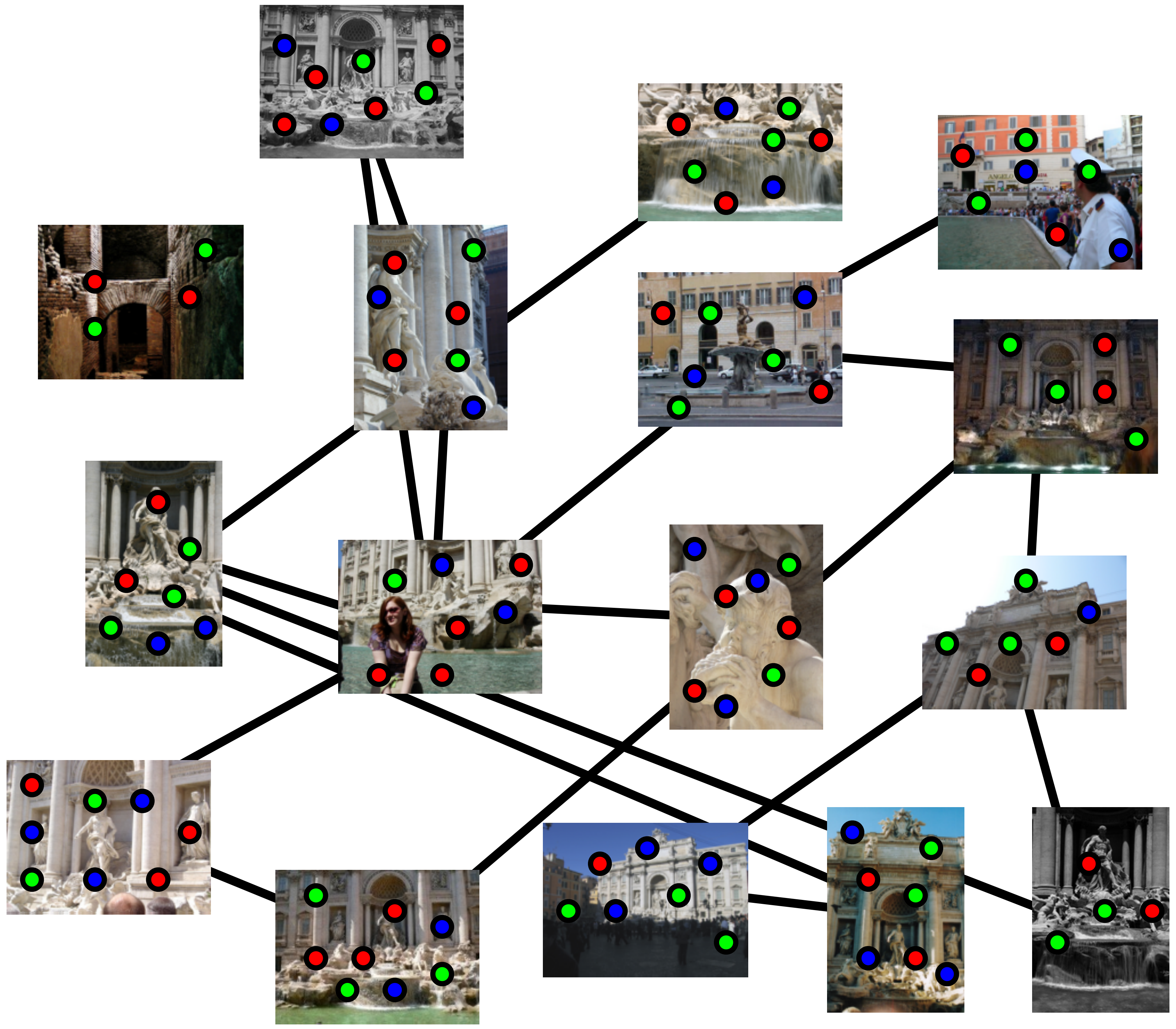
Detect features using SIFT [Lowe, IJCV 2004]



Source: N. Snavely

Feature matching

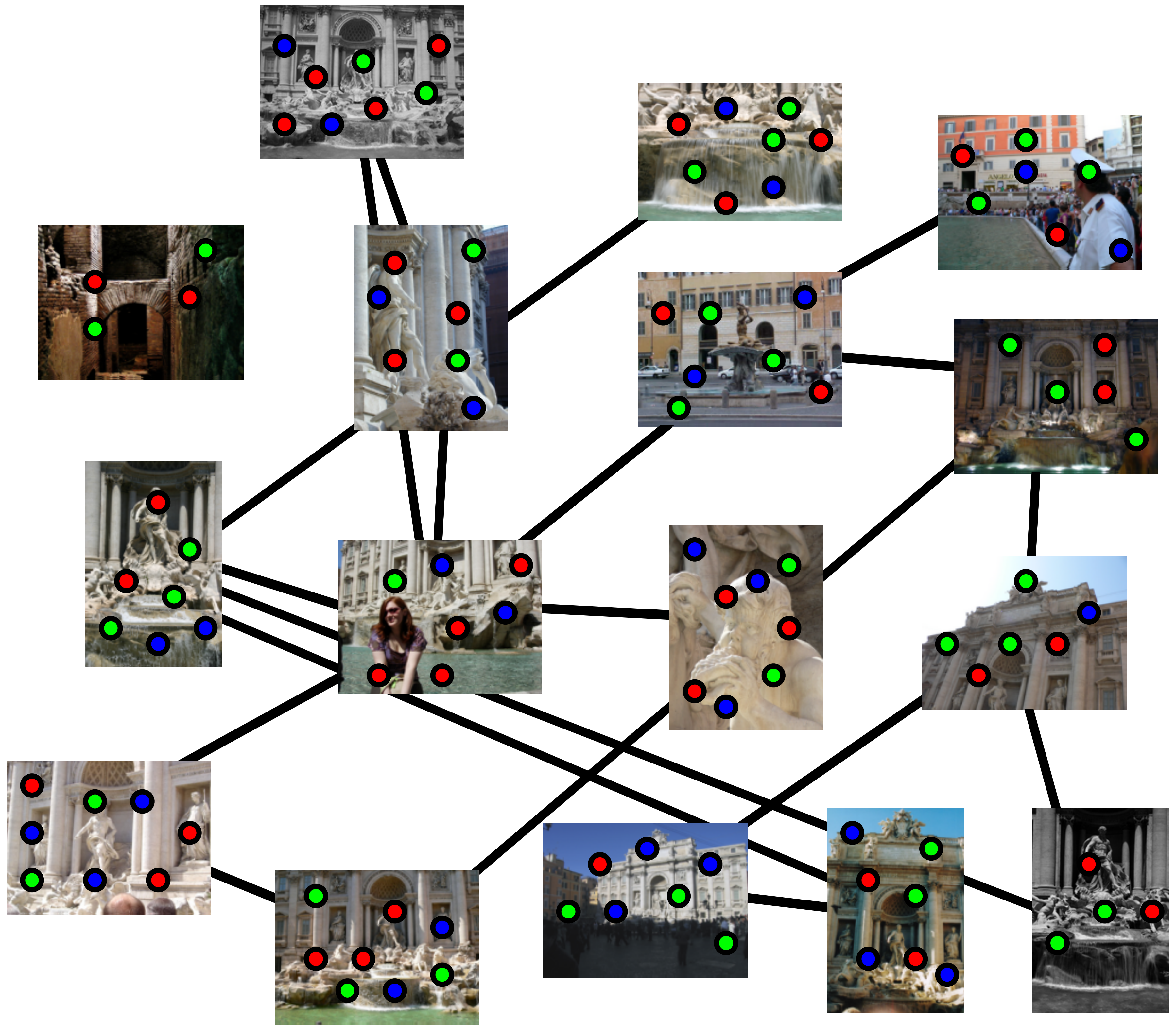
Match features between each pair of images



Source: N. Snavely

Feature matching

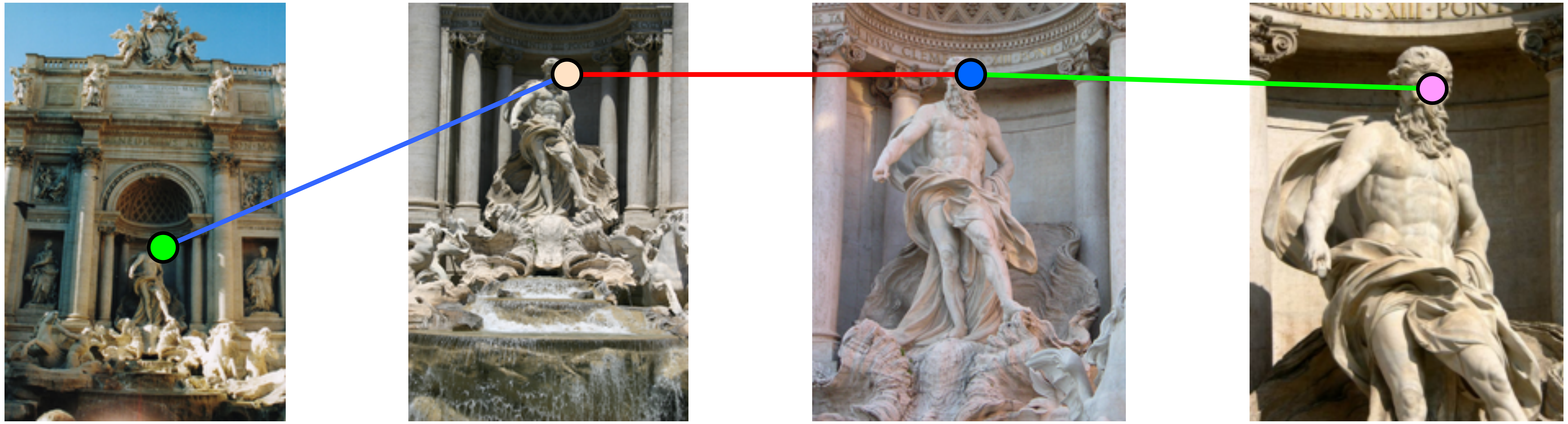
Refine matching using RANSAC to estimate fundamental matrix between each pair



Source: N. Snavely

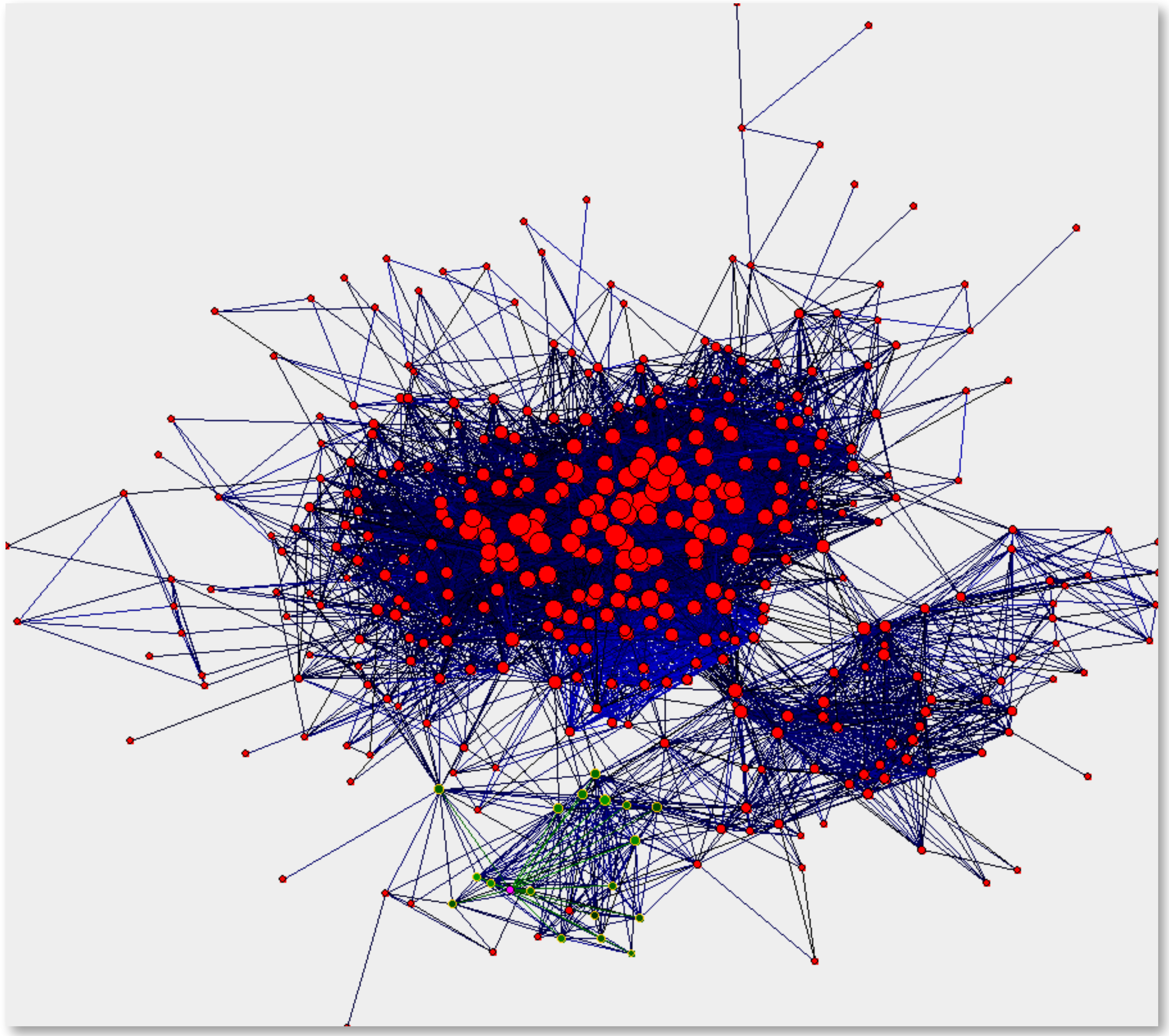
Correspondence estimation

- Link up pairwise matches to form connected components of matches across several images



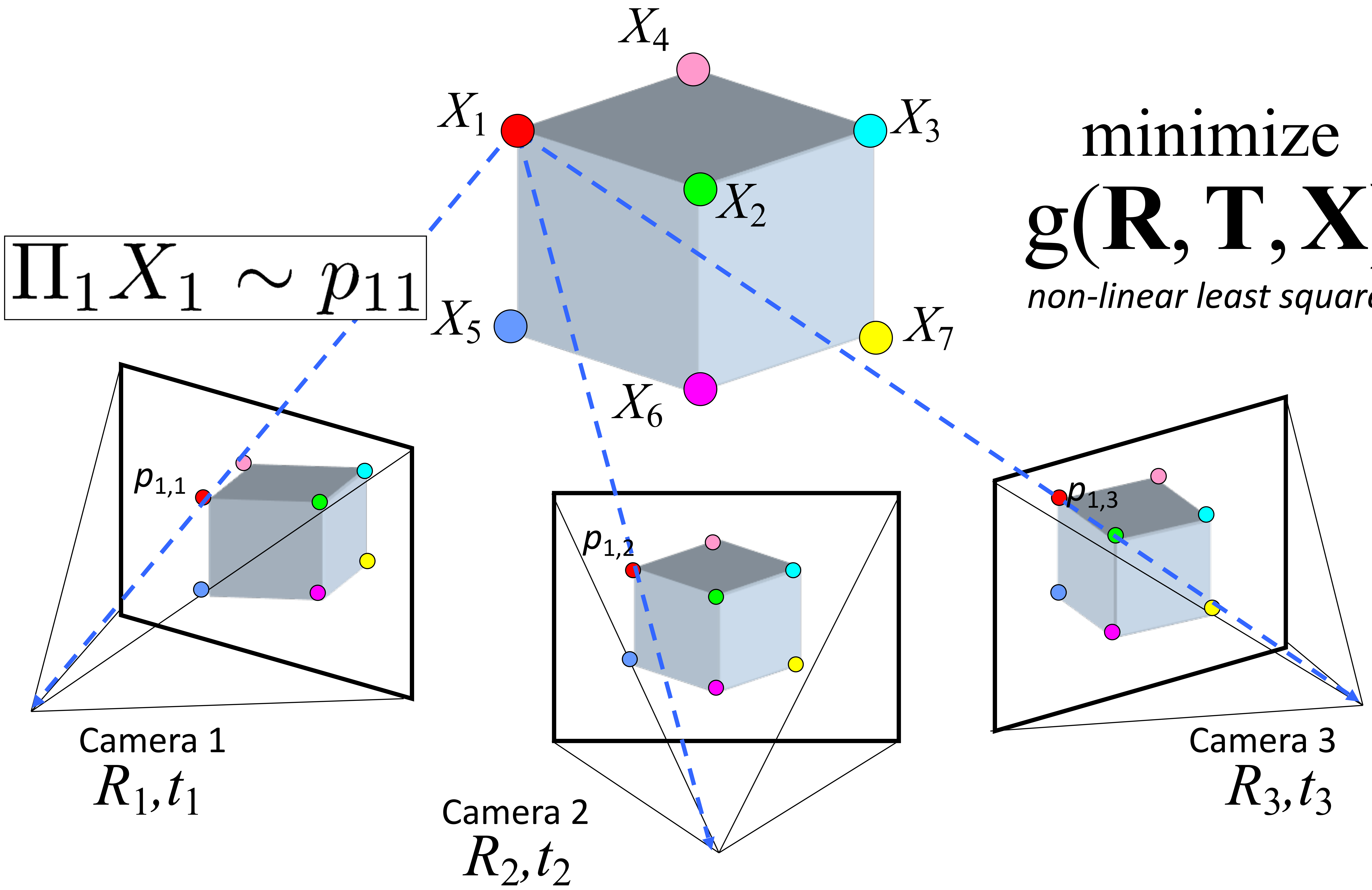
Source: N. Snavely

Image connectivity graph



Source: N. Snavely

Structure from motion



minimize

$$g(\mathbf{R}, \mathbf{T}, \mathbf{X})$$

non-linear least squares

Source: N. Snavely

Structure from motion

- Minimize sum of squared reprojection errors:

$$g(\mathbf{X},\mathbf{R},\mathbf{T}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \underbrace{w_{ij}}_{\substack{\textit{indicator variable}:\\ \text{is point } i \text{ visible in image } j\text{ ?}}} \cdot \left\| \underbrace{\mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j)}_{\substack{\textit{predicted}\\ \text{image location}}} - \underbrace{\begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}}_{\substack{\textit{observed}\\ \text{image location}}} \right\|^2$$

- Minimizing this function is called *bundle adjustment*
  – Optimized using non-linear least squares, e.g. Levenberg-Marquardt

Source: N. Snavely

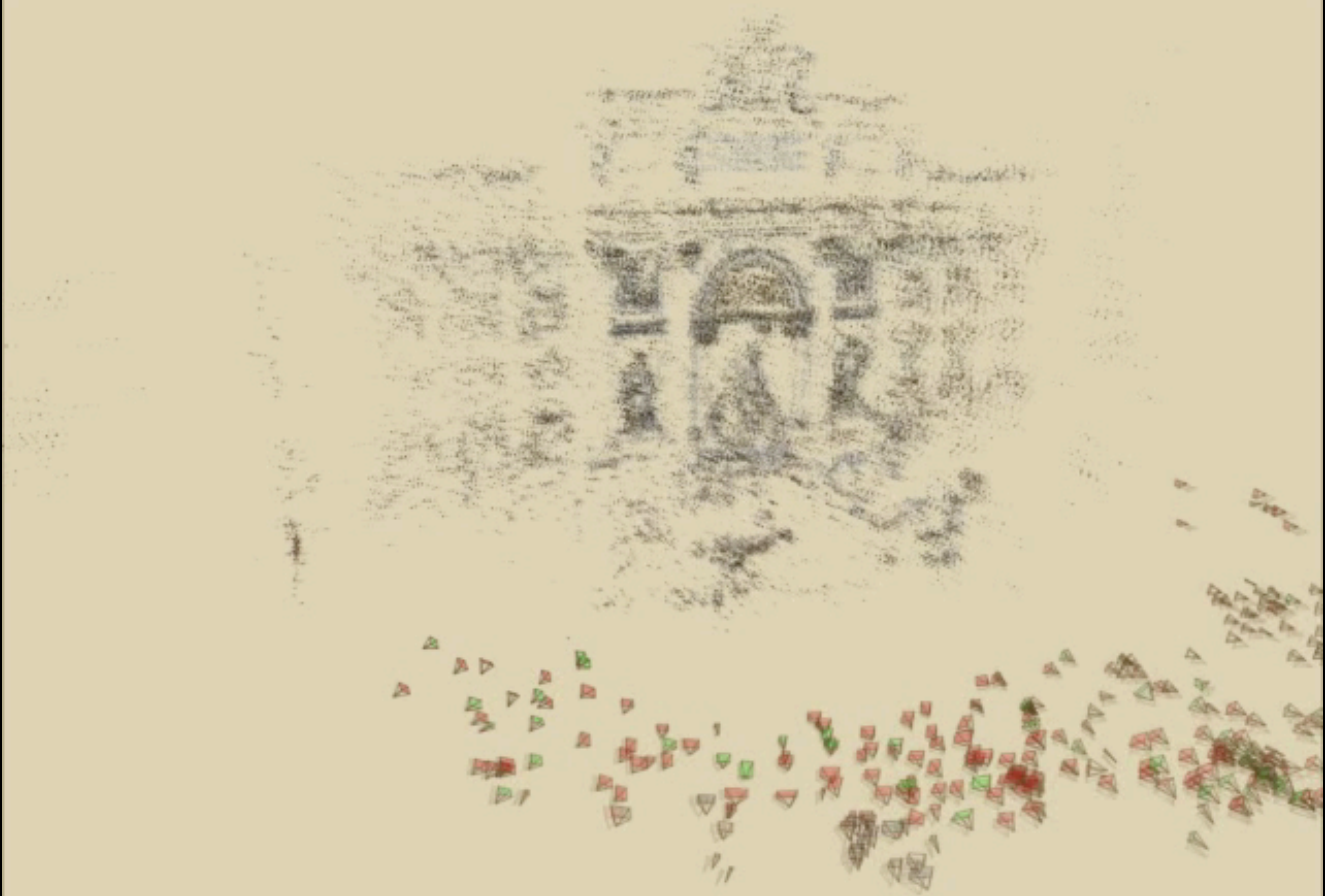
Photo Tourism

Exploring photo collections in 3D

Noah Snavely Steven M. Seitz Richard Szeliski

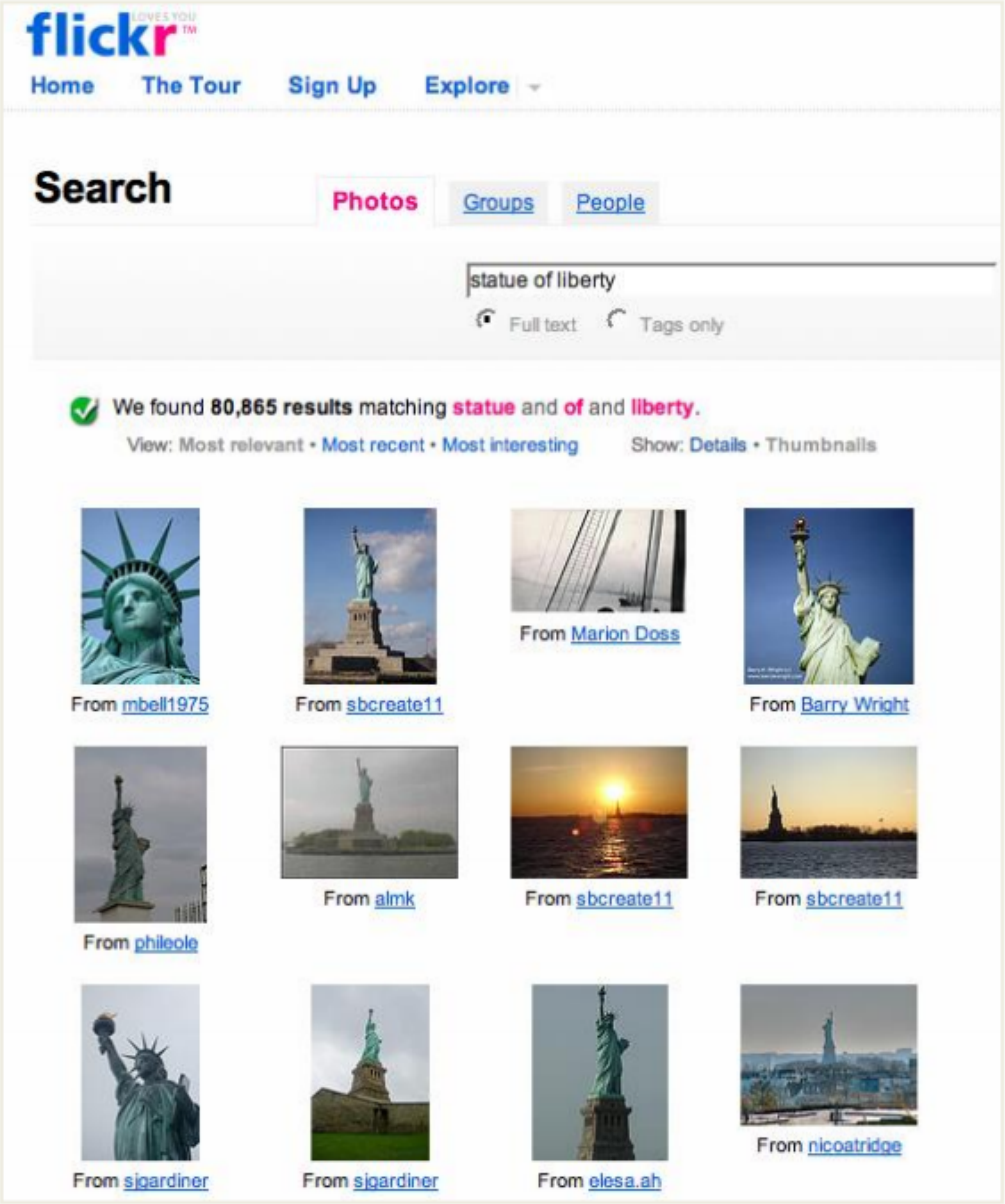
University of Washington *Microsoft Research*

SIGGRAPH 2006




Name: brodo_1608736
Added by: brodo
Date: November 21, 20...

Multi-view stereo

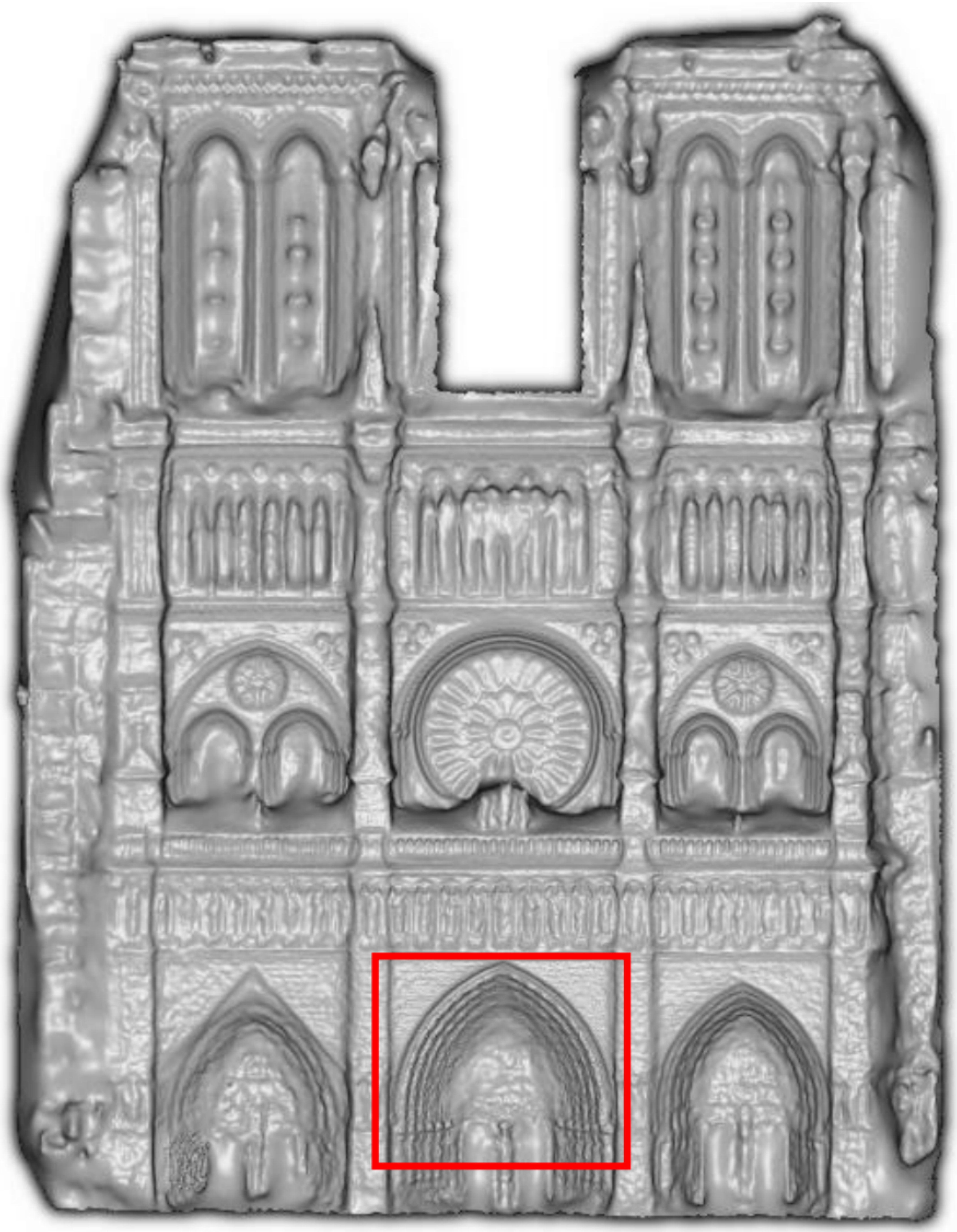


We have the camera pose. Estimate depth using stereo!

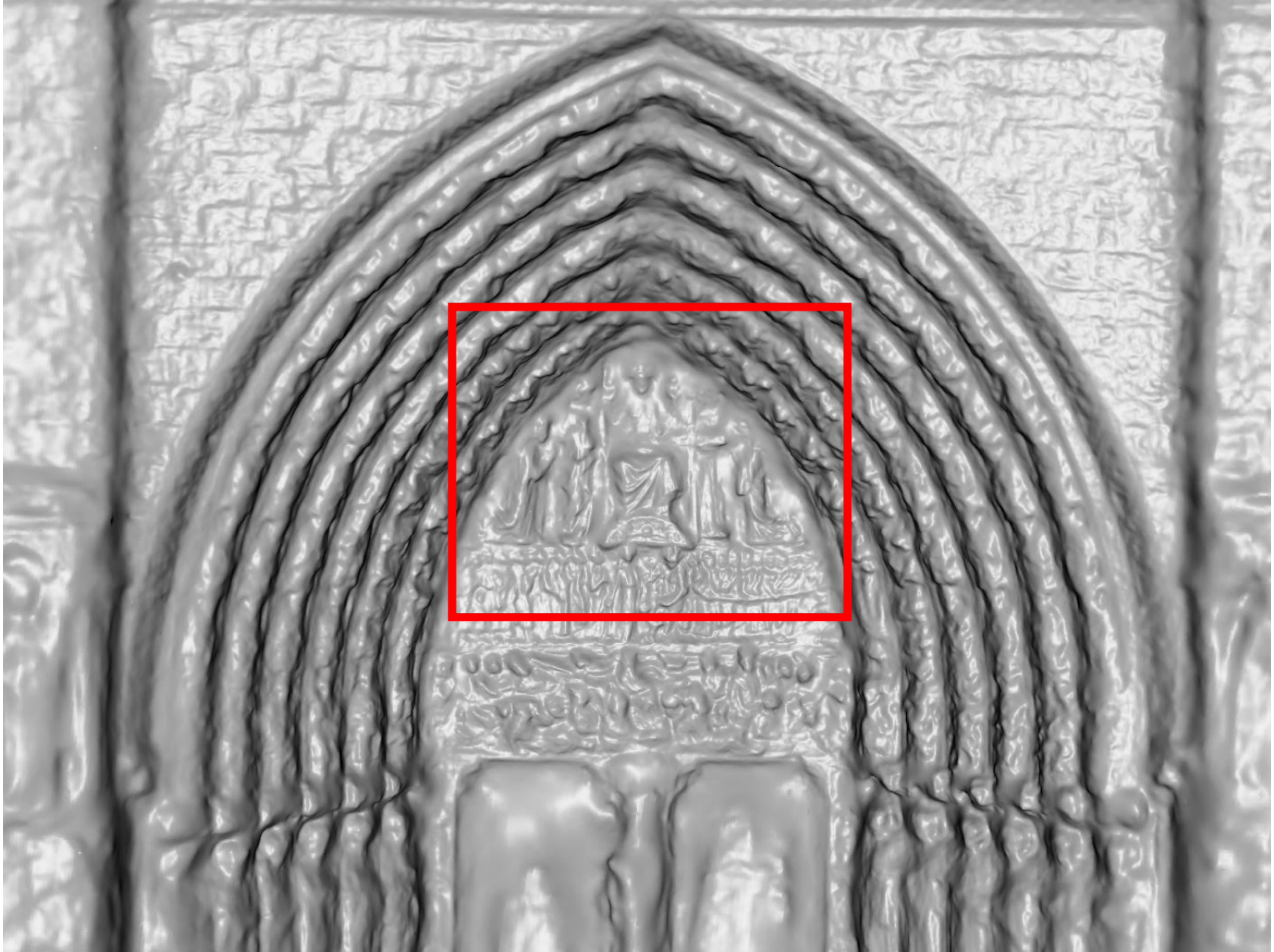
Source: N. Snavely

Notre Dame de Paris

653 images
313 photographers



Source: N. Snavely



Source: N. Snavely