Homographies and RANSAC

Computer vision 6.869

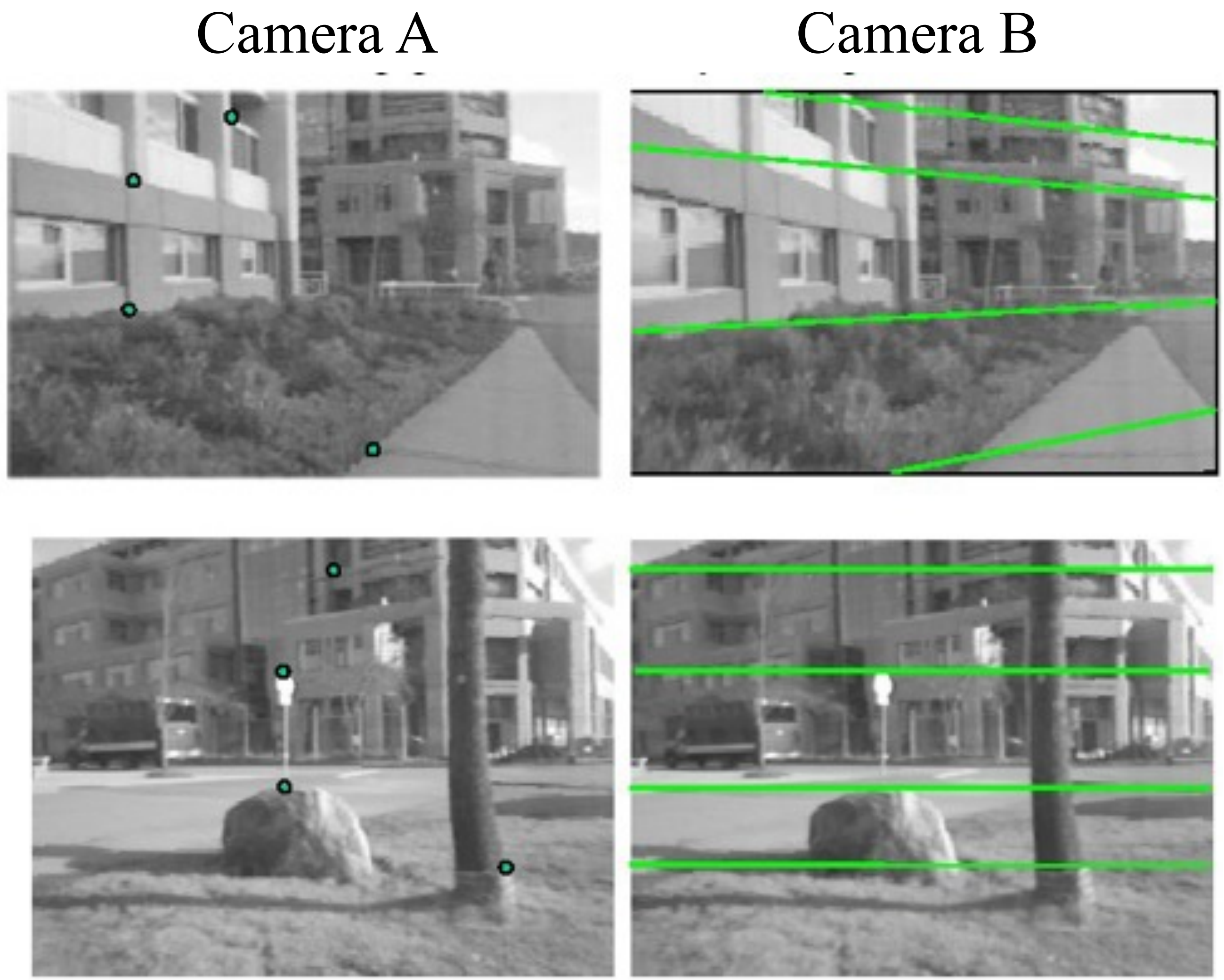
Bill Freeman and Antonio Torralba

March 30, 2011

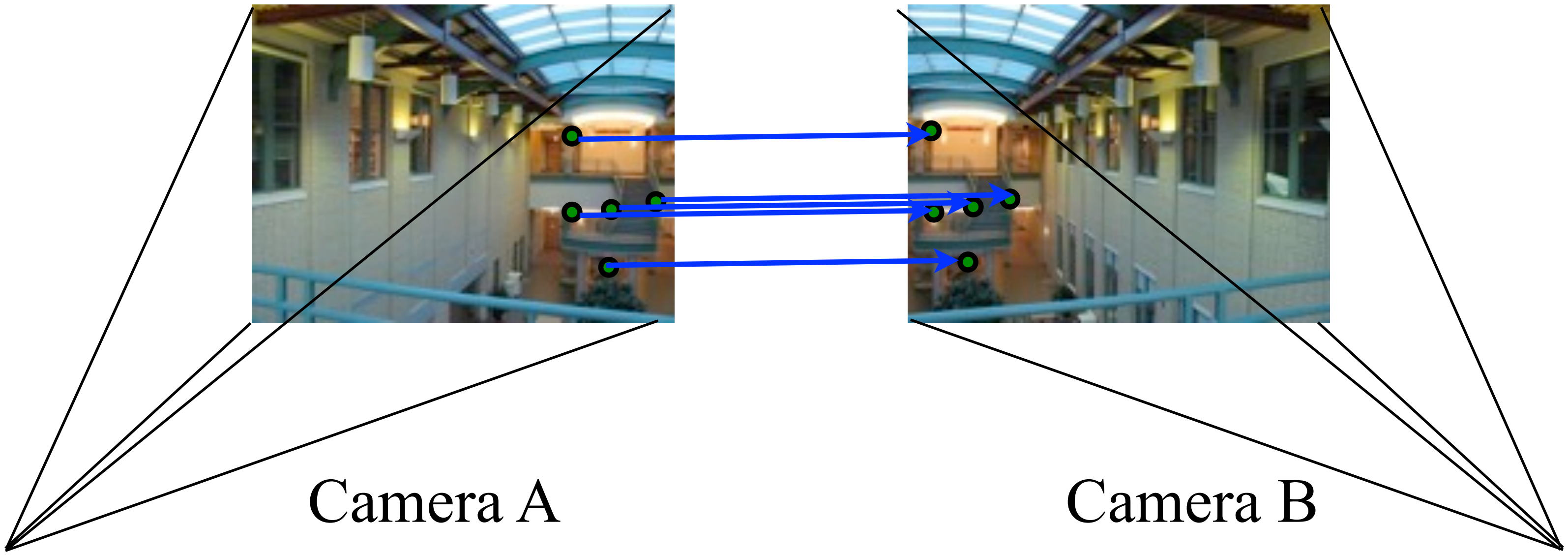
Homographies and RANSAC

- Homographies
- RANSAC
- Building panoramas
- Phototourism

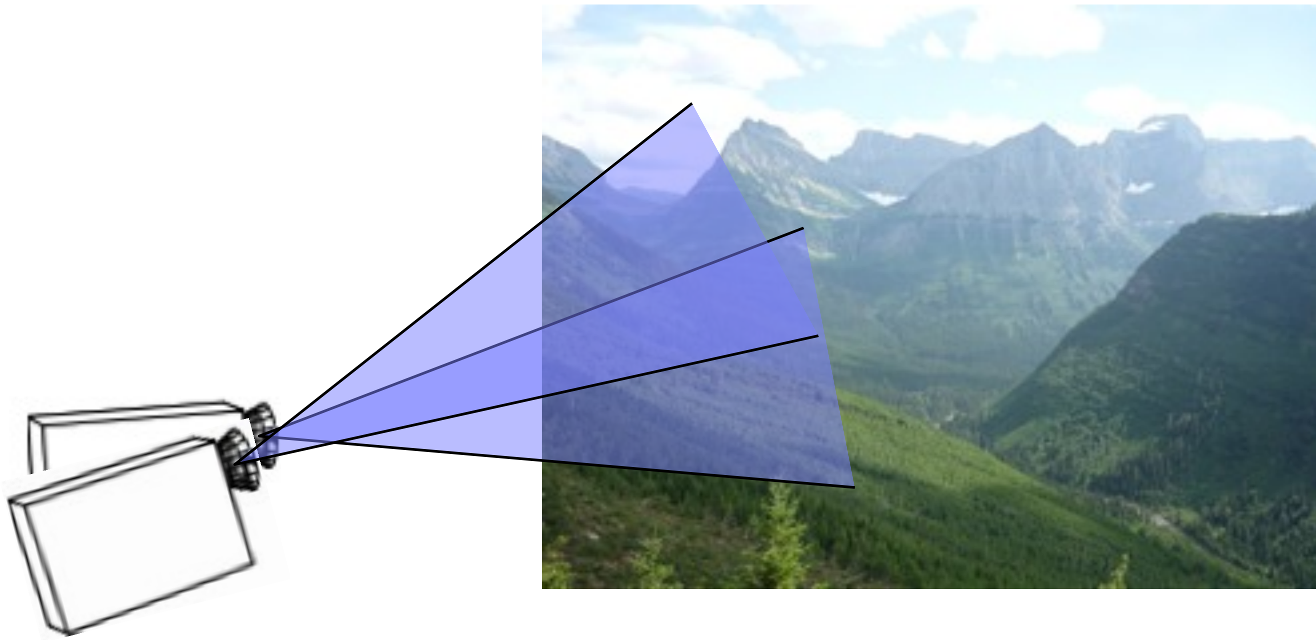
Depth-based ambiguity of position



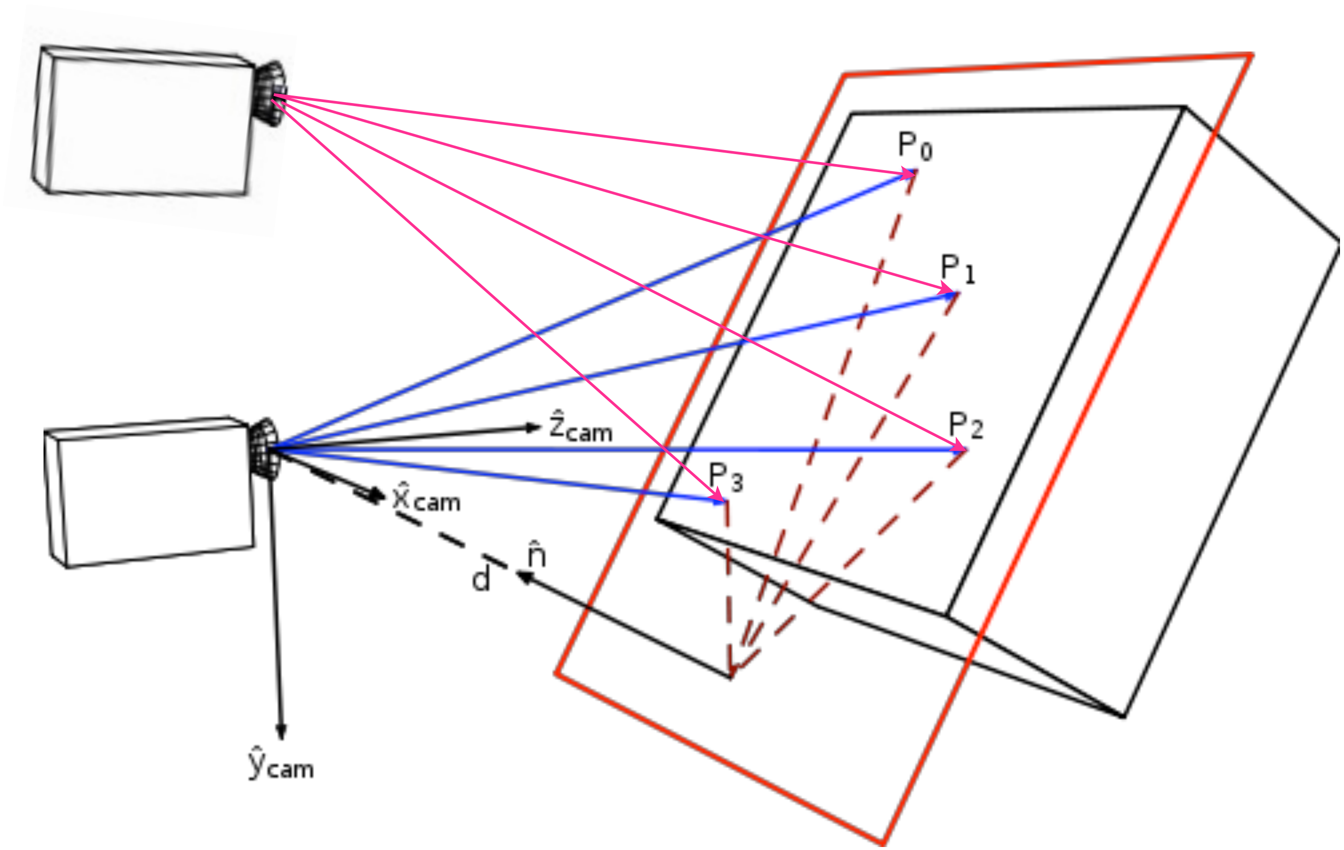
Under what conditions can you know where to translate each point of image A to where it would appear in camera B (with calibrated cameras), knowing nothing about image depths?



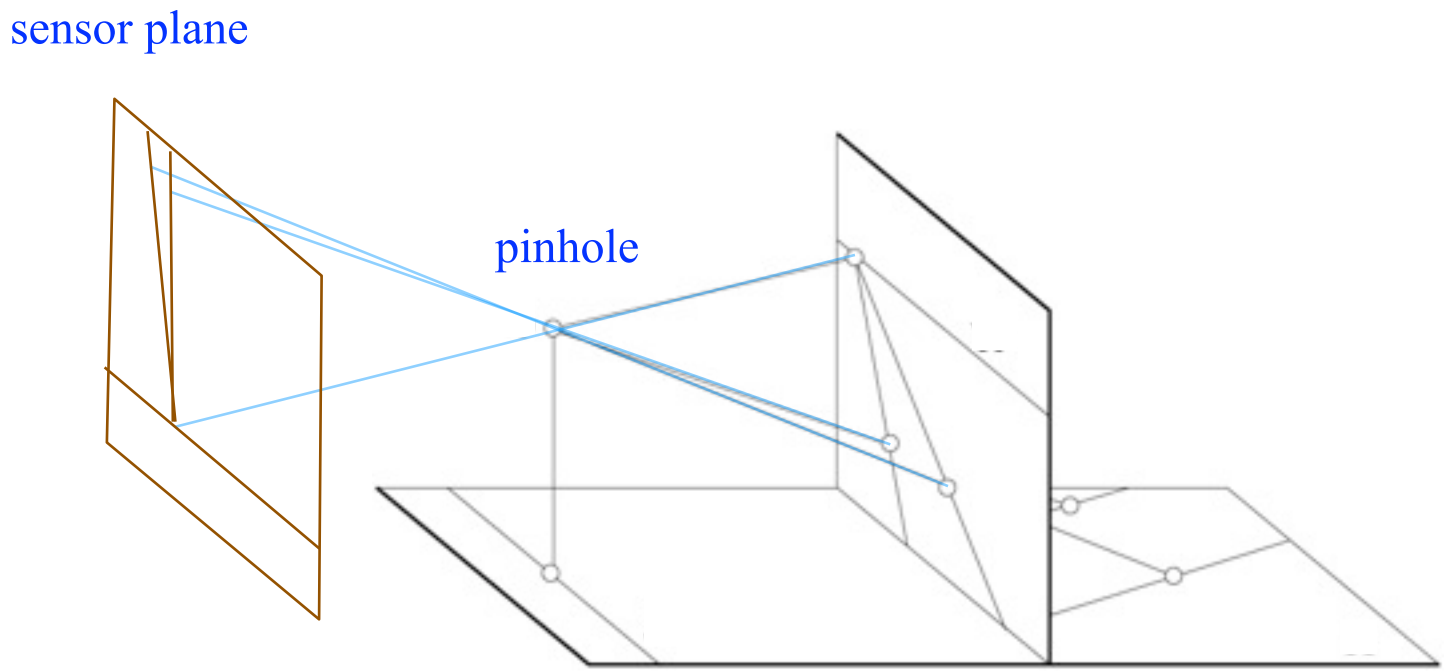
(a) camera rotation



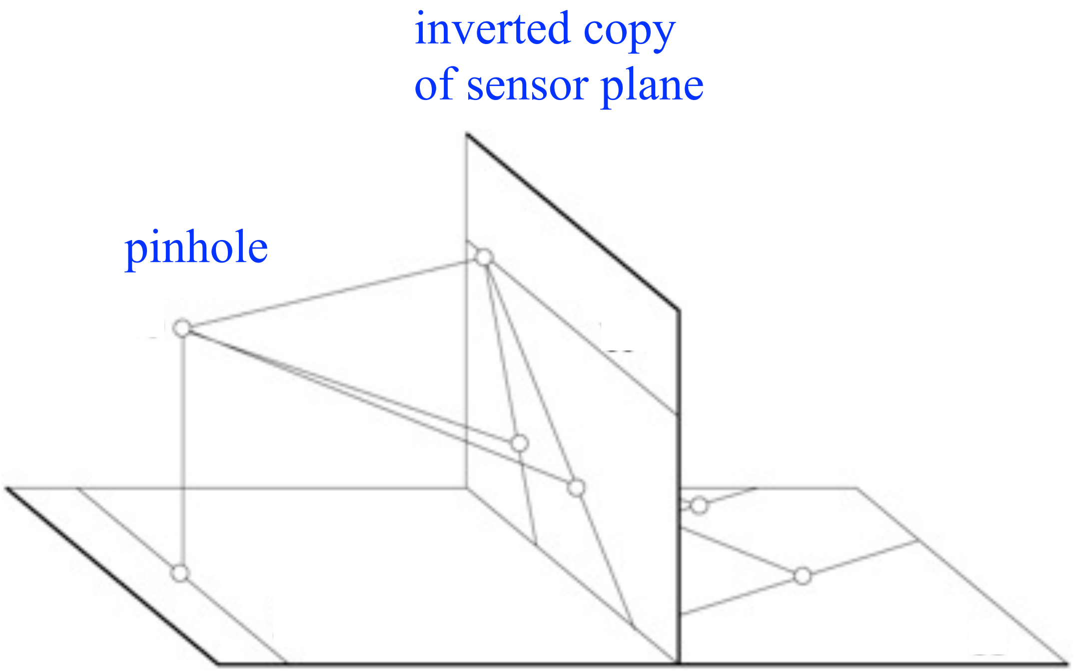
and (b) imaging a planar surface



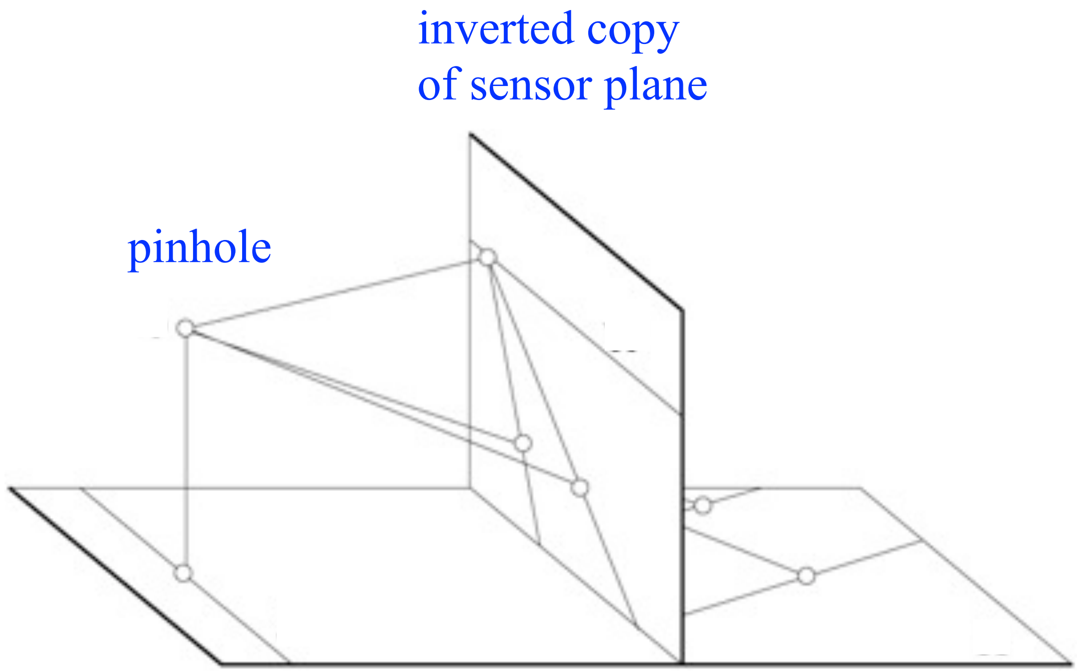
Geometry of perspective projection



Geometry of perspective projection

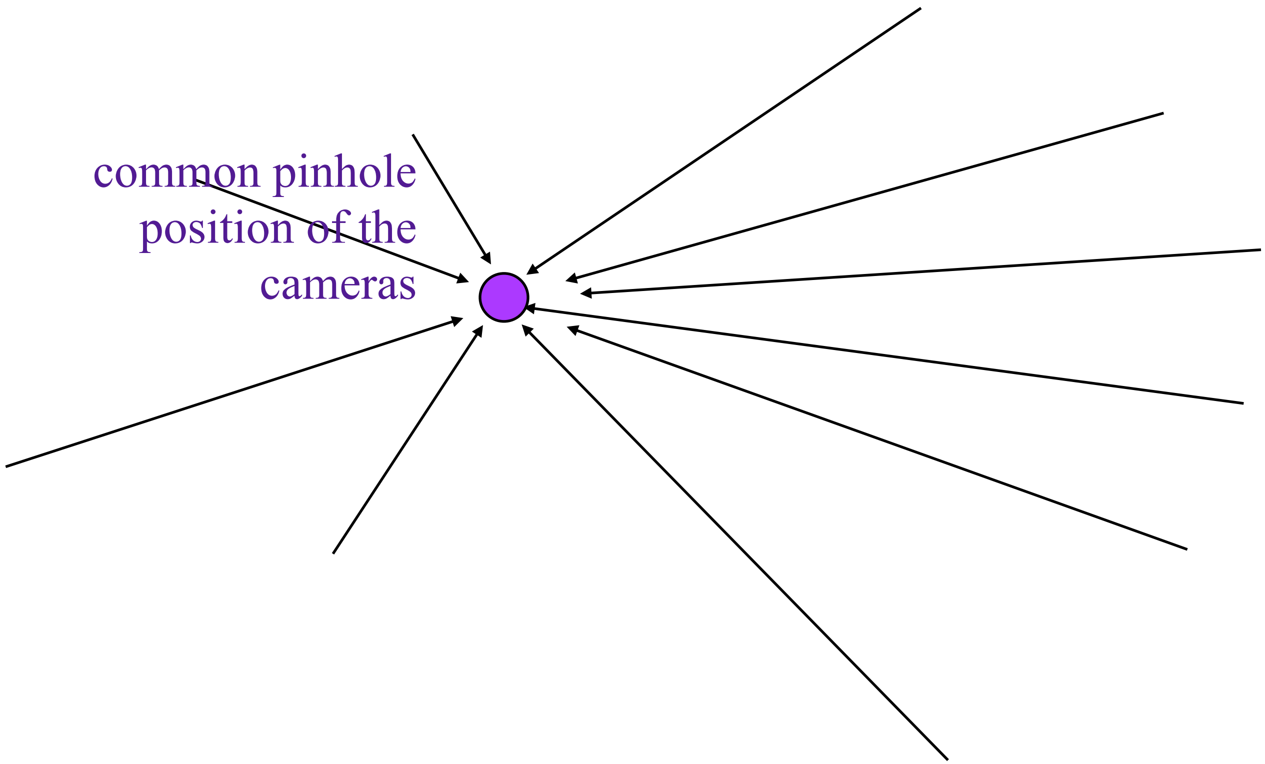


Geometry of perspective projection

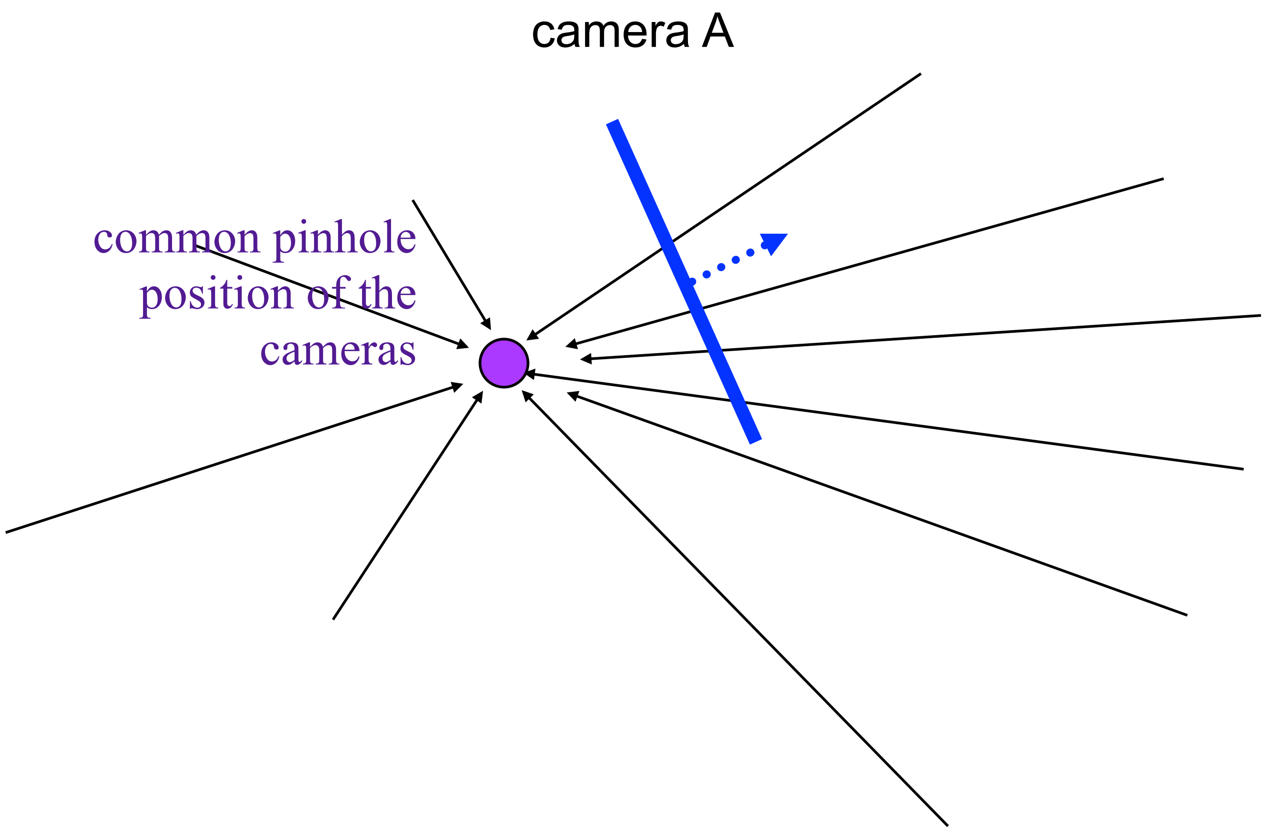


Let's look at this scene from above...

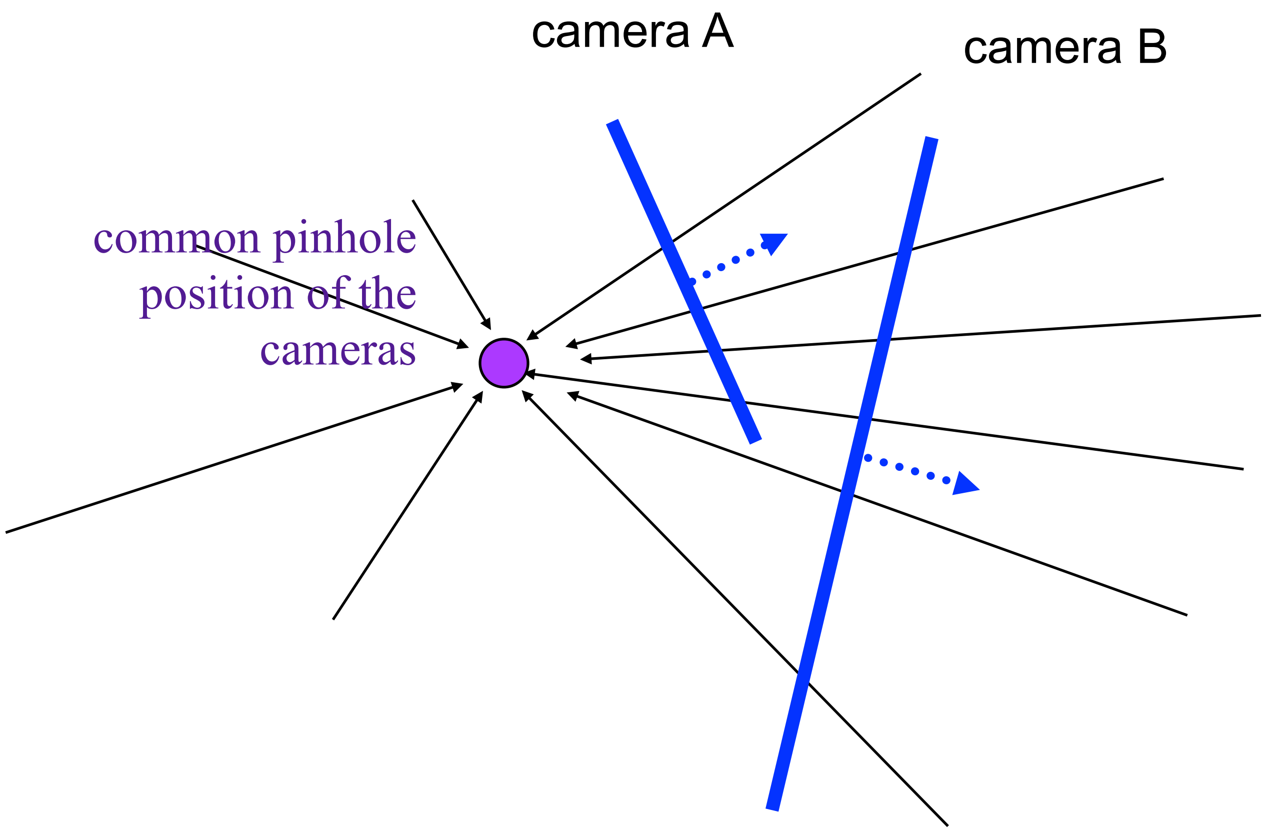
Two cameras with same center of projection



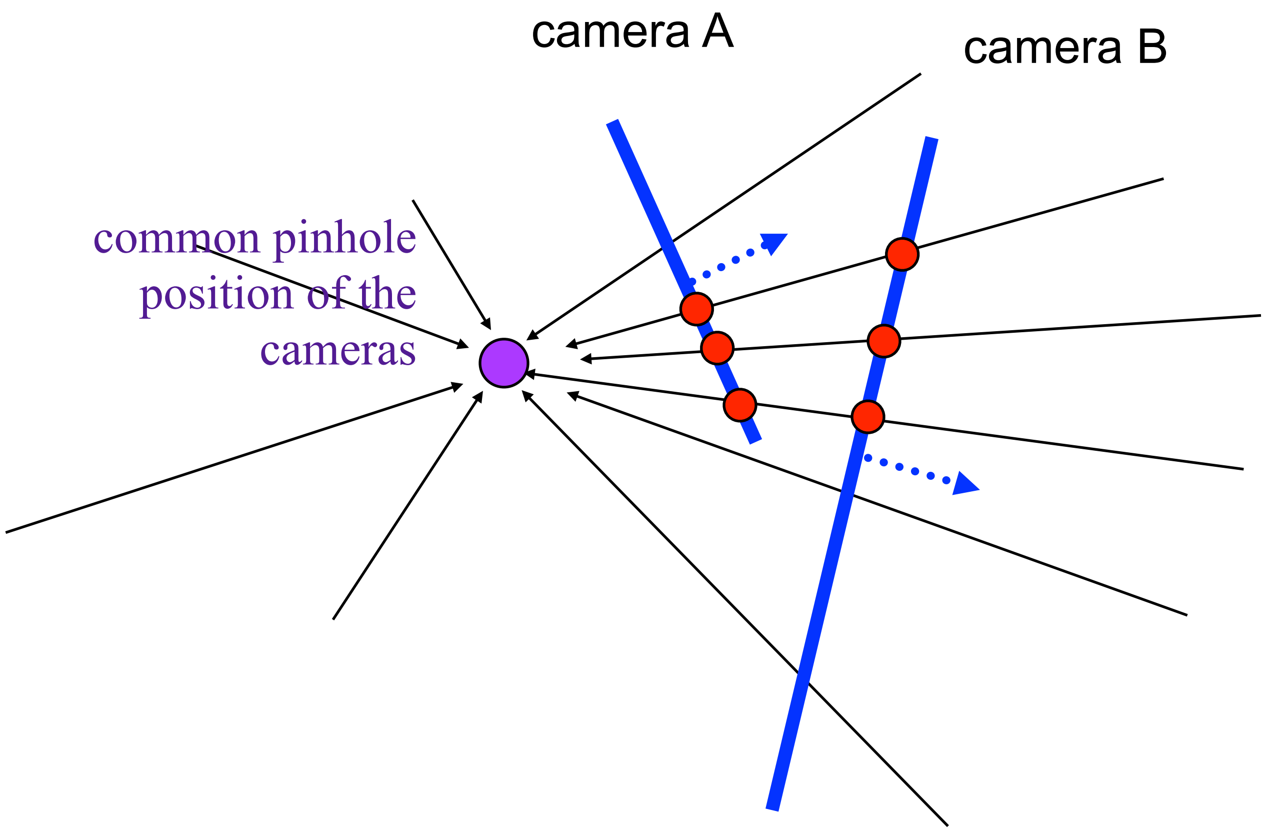
Two cameras with same center of projection



Two cameras with same center of projection

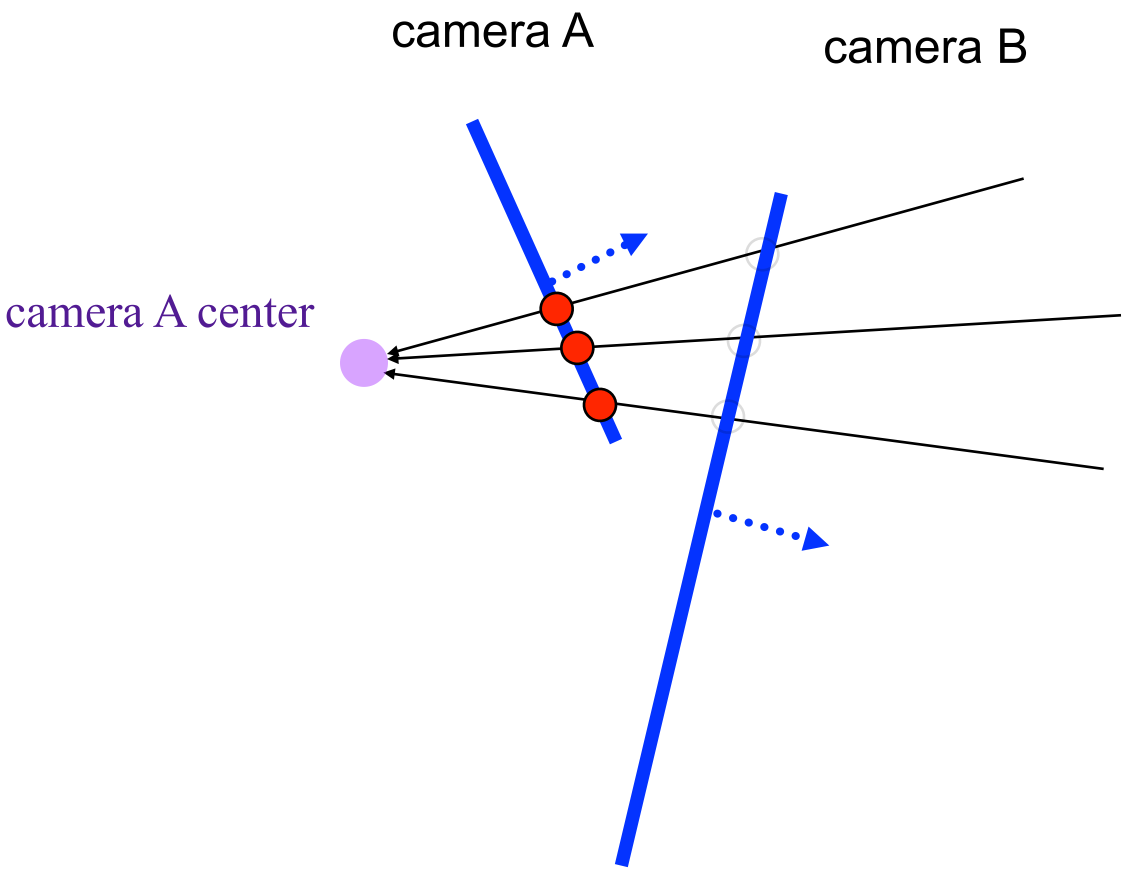


Two cameras with same center of projection

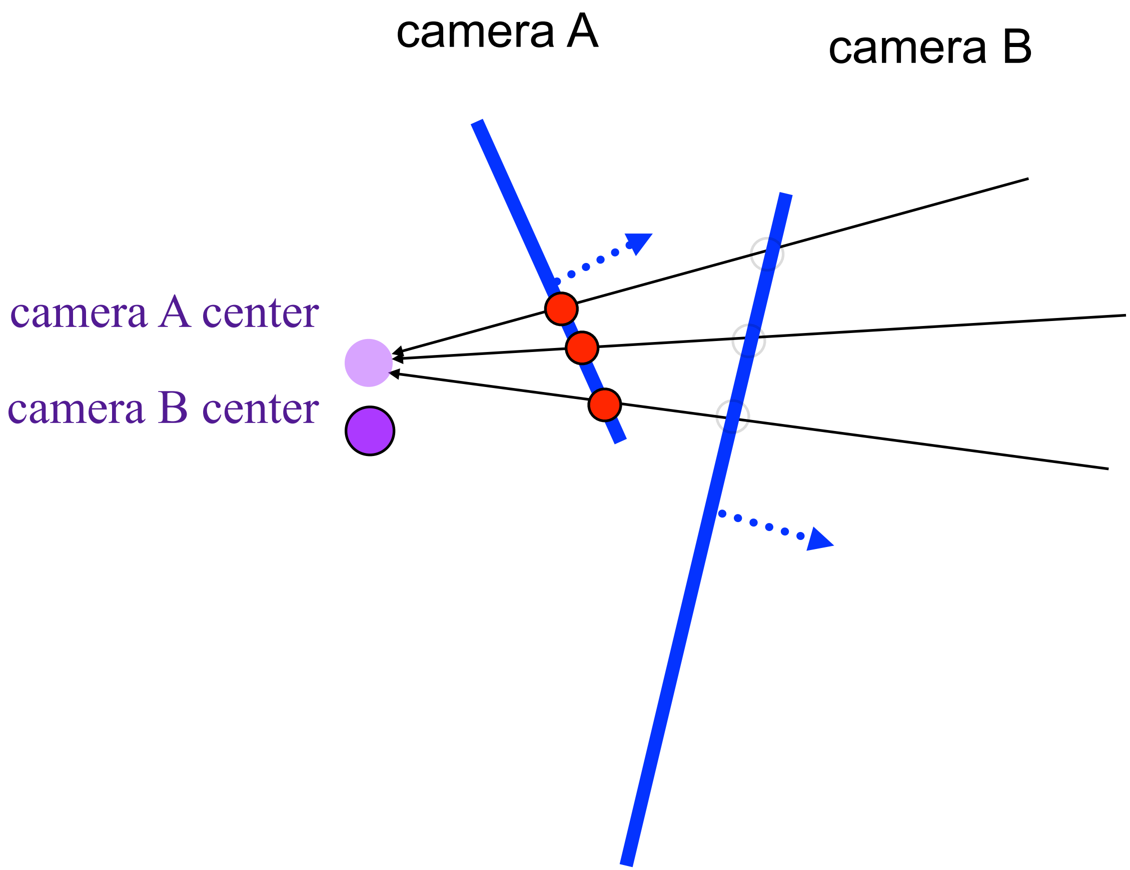


Can generate any synthetic camera view
as long as it has **the same center of projection**!

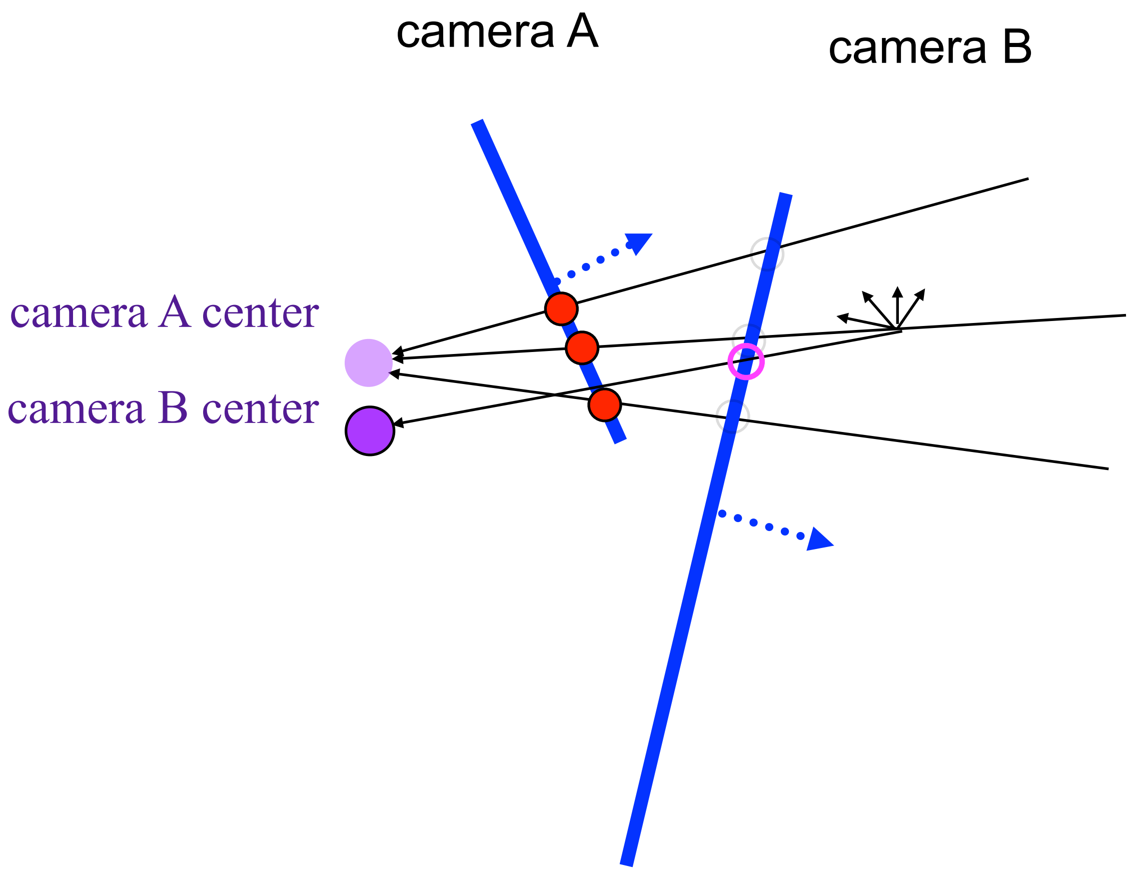
Two cameras with offset centers of projection



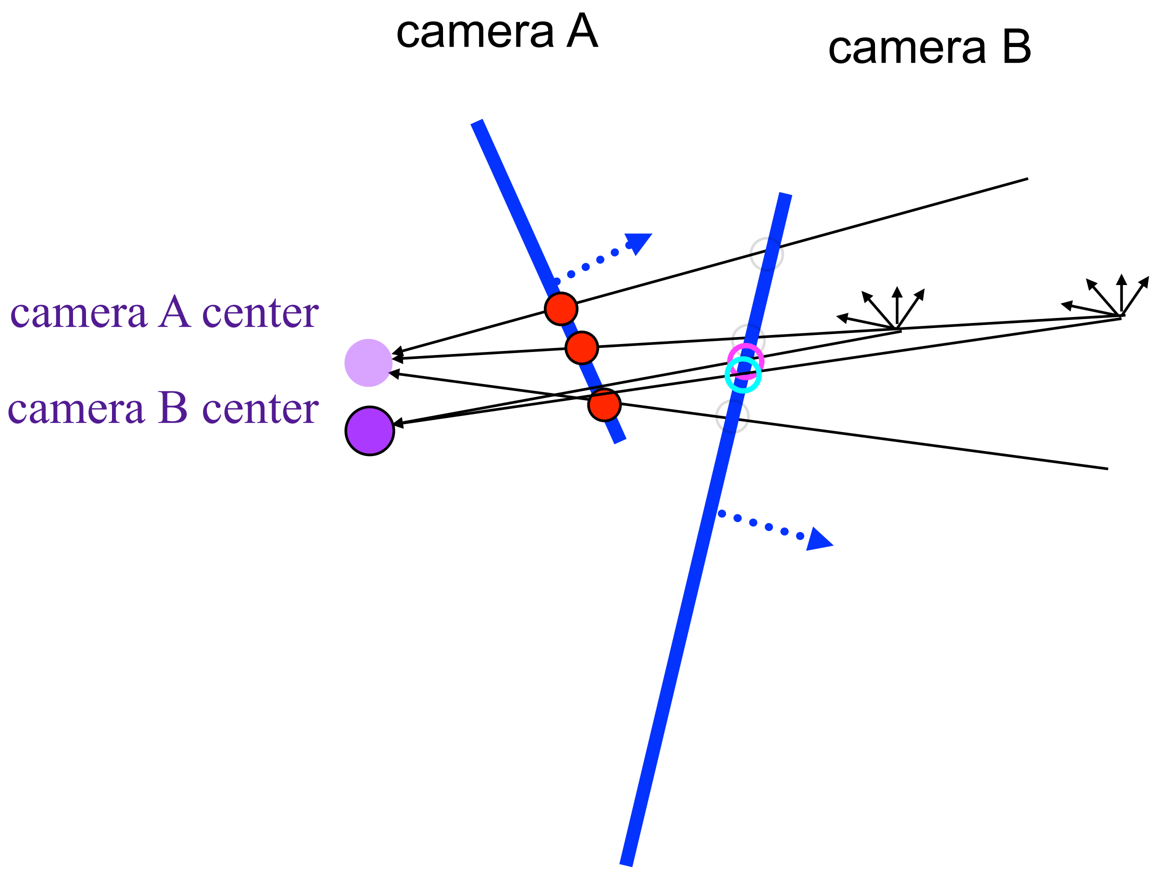
Two cameras with offset centers of projection



Two cameras with offset centers of projection

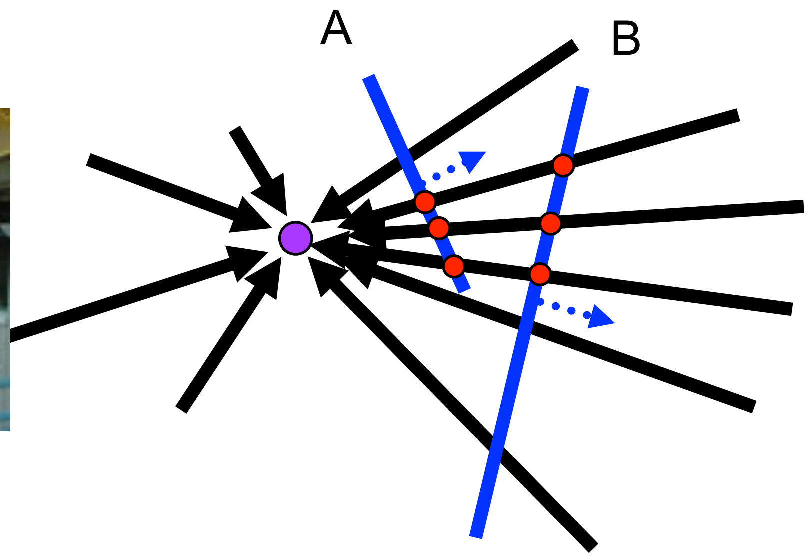


Two cameras with offset centers of projection



Recap

- **When we only rotate the camera (around nodal point) depth does not matter**
- **It only performs a 2D warp**
  - one-to-one mapping of the 2D plane
  - plus of course reveals stuff that was outside the field of view



- **Now we just need to figure out this mapping**

Aligning images: translation?

Aligning images: translation?

left on top

right on top

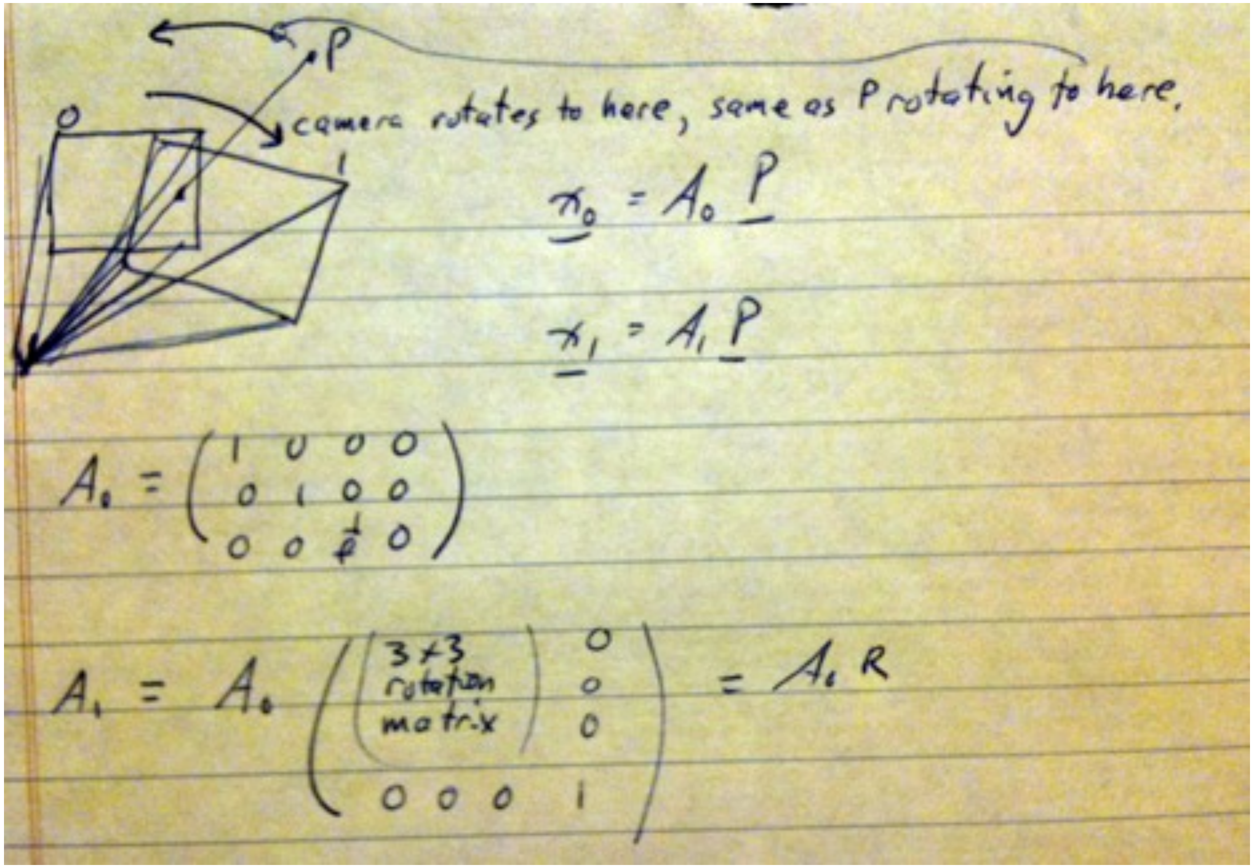
Aligning images: translation?

left on top

right on top

Translations are not enough to align the images

homography


P
camera rotates to here, same as P rotating to here.
0
1
$\underline{x}_0 = A_0 \underline{P}$
$\underline{x}_1 = A_1 \underline{P}$
$A_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{pmatrix}$
$A_1 = A_0 \begin{pmatrix} \begin{pmatrix} \text{3x3} \\ \text{rotation} \\ \text{matrix} \end{pmatrix} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix} = A_0 R$

homography

we seek M_{10} such that

$$\underline{x}_0 = M_{10}\,\underline{x}_1 \qquad \text{for all } \underline{x}_0, \underline{x}_1$$

$$\underbrace{A_0\,\underline{P}}_{x_0} = M_{10}\,\underbrace{A_0\,R\,\underline{P}}_{x_1} \qquad \text{for all } \underline{P}\text{, so}$$

$$A_0 = M_{10}\,A_0\,R \qquad \text{mult by } R^{-1} = \begin{pmatrix} \begin{pmatrix}\text{inverse}\\ \text{rotation}\end{pmatrix} & \begin{matrix}0\\0\\0\end{matrix} \\ 0\ 0\ 0 & 1 \end{pmatrix}$$

$$A_0 R^{-1} = M_{10}\,A_0$$

$$A_0 R^{-1} B = M_{10} \qquad \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & f & 0 \end{pmatrix}}_{A_0} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \\ 0 & 0 & 0 \end{pmatrix}}_{B} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

homography

$$M_{10} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{pmatrix} R^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \\ 0 & 0 & 0 \end{pmatrix}$$

$$\underline{x}_0 = M_{10}\,\underline{x}_1 \qquad \text{for all } \underline{x}_0, \underline{x}_1$$

homography

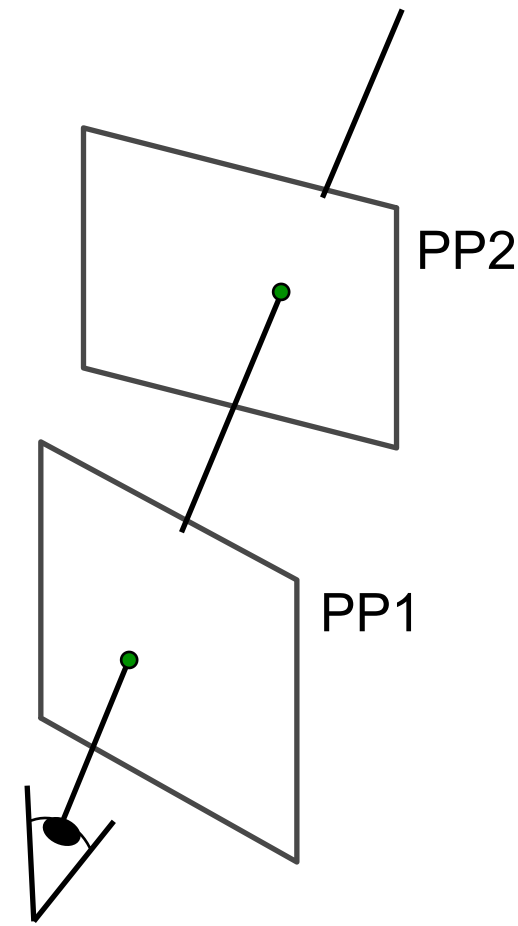
$$M_{10} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{pmatrix} R^{-1} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \\ 0 & 0 & 0 \end{pmatrix}$$

$$\underline{x}_0 = M_{10}\, \underline{x}_1 \qquad \text{for all } \underline{x}_0, \underline{x}_1$$

How many pairs of points does it take to specify M_10?

Homography

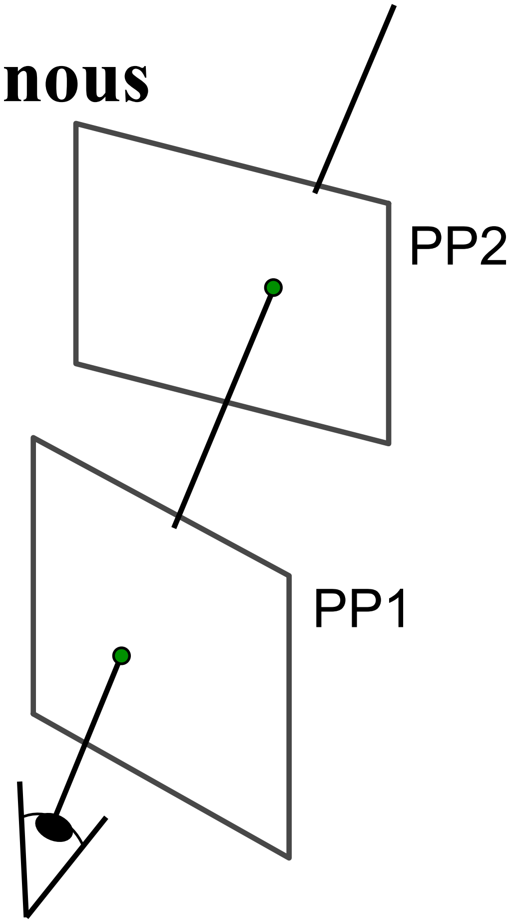
See Szeliski Sect 2.1.5, Mapping from one camera to another.



Homography

- **Projective – mapping between any two projection planes with the same center of projection**
- **called Homography**
- **represented as 3x3 matrix in homogenous coordinates**

$$\underset{\mathbf{p'}}{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}} = \underset{\mathbf{H}}{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}} \underset{\mathbf{p}}{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}$$



See Szeliski Sect 2.1.5, Mapping from one camera to another.

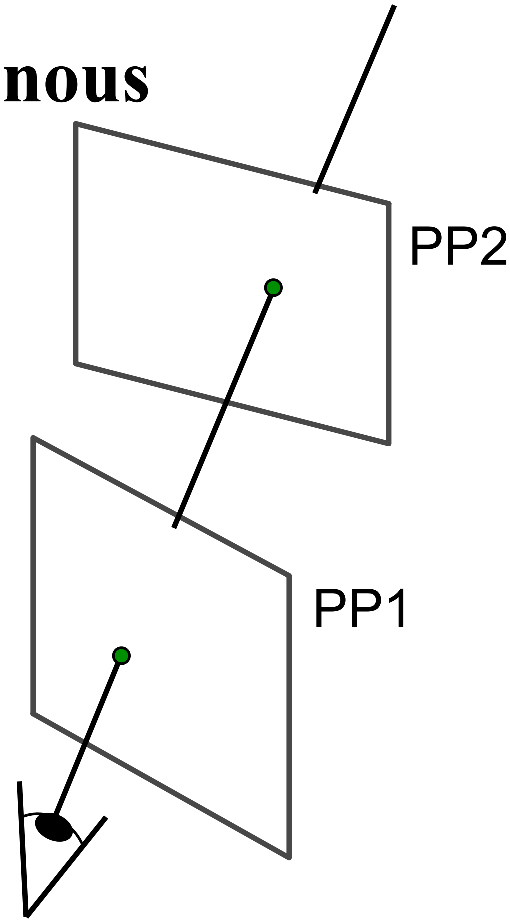
Homography

- **Projective – mapping between any two projection planes with the same center of projection**
- **called Homography**
- **represented as 3x3 matrix in homogenous coordinates**

$$\underbrace{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}}_{\mathbf{p'}} = \underbrace{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\mathbf{p}}$$

To apply a homography **H**

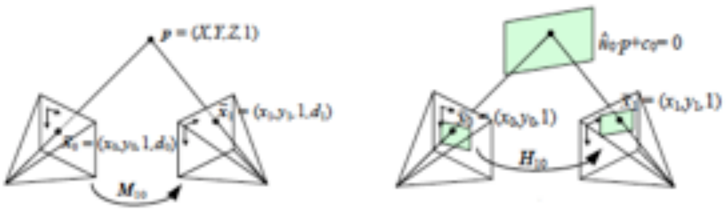
- Compute **p’** = **Hp** (regular matrix multiply)
- Convert **p’** from homogeneous to image coordinates (divide by w)



See Szeliski Sect 2.1.5, Mapping from one camera to another.

Images of planar objects, taken by generically offset cameras, are also related by a homography.

From Szeliski book



(a) (b)

Figure 2.12 A point is projected into two images: (a) relationship between the 3D point coordinate $(X, Y, Z, 1)$ and the 2D projected point $(x, y, 1, d)$; (b) planar homography induced by points all lying on a common plane $\hat{n}_0 \cdot p + c_0 = 0$.

Mapping from one camera to another

What happens when we take two images of a 3D scene from different camera positions or orientations (Figure 2.12a)? Using the full rank 4×4 camera matrix $\tilde{P} = \tilde{K}E$ from (2.64), we can write the projection from world to screen coordinates as

$$\tilde{x}_0 \sim \tilde{K}_0 E_0 p = \tilde{P}_0 p. \tag{2.68}$$

Assuming that we know the z-buffer or disparity value d_0 for a pixel in one image, we can compute the 3D point location p using

$$p \sim E_0^{-1} \tilde{K}_0^{-1} \tilde{x}_0 \tag{2.69}$$

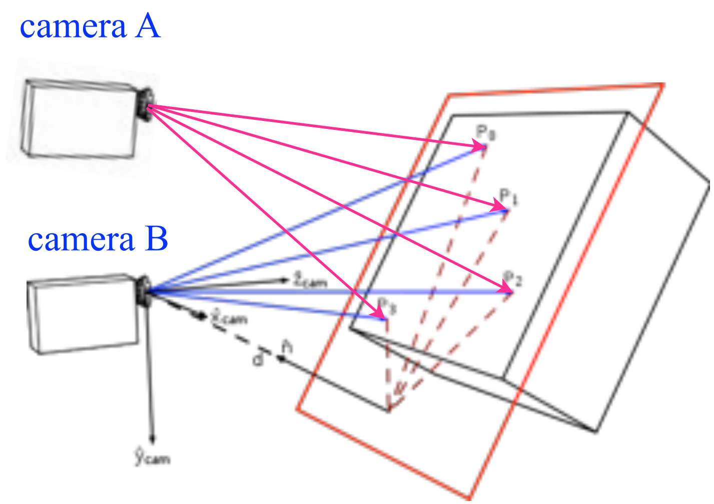
and then project it into another image yielding

$$\tilde{x}_1 \sim \tilde{K}_1 E_1 p = \tilde{K}_1 E_1 E_0^{-1} \tilde{K}_0^{-1} \tilde{x}_0 = \tilde{P}_1 \tilde{P}_0^{-1} \tilde{x}_0 = M_{10} \tilde{x}_0. \tag{2.70}$$

Unfortunately, we do not usually have access to the depth coordinates of pixels in a regular photographic image. However, for a *planar scene*, as discussed above in (2.66), we can replace the last row of P_0 in (2.64) with a general *plane equation*, $\hat{n}_0 \cdot p + c_0$ that maps points on the plane to $d_0 = 0$ values (Figure 2.12b). Thus, if we set $d_0 = 0$, we can ignore the last column of M_{10} in (2.70) and also its last row, since we do not care about the final z-buffer depth. The mapping equation (2.70) thus reduces to

$$\tilde{x}_1 \sim \tilde{H}_{10} \tilde{x}_0, \tag{2.71}$$

where $\tilde{H}_{10}$ is a general 3×3 homography matrix and $\tilde{x}_1$ and $\tilde{x}_0$ are now 2D homogeneous coordinates (i.e., 3-vectors) (Szeliski 1996).This justifies the use of the 8-parameter homography as a general alignment model for mosaics of planar scenes (Mann and Picard 1994; Szeliski 1996).



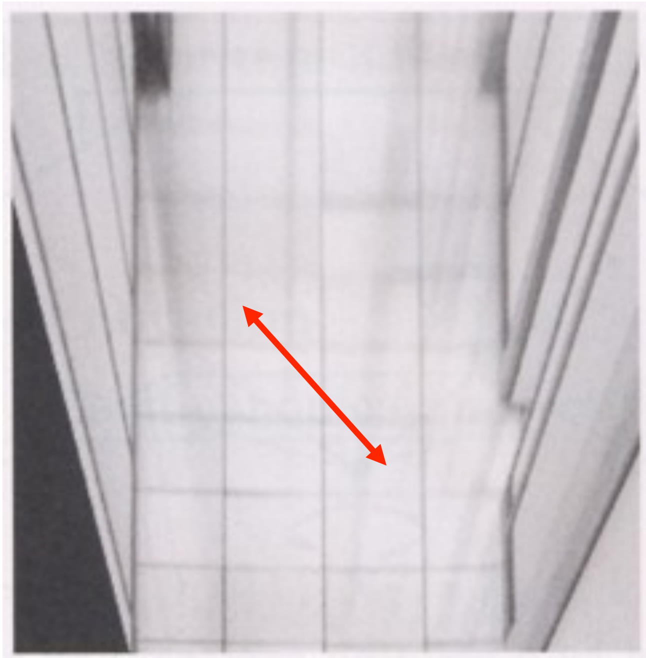
Measurements on planes

Measurements on planes

Measurements on planes

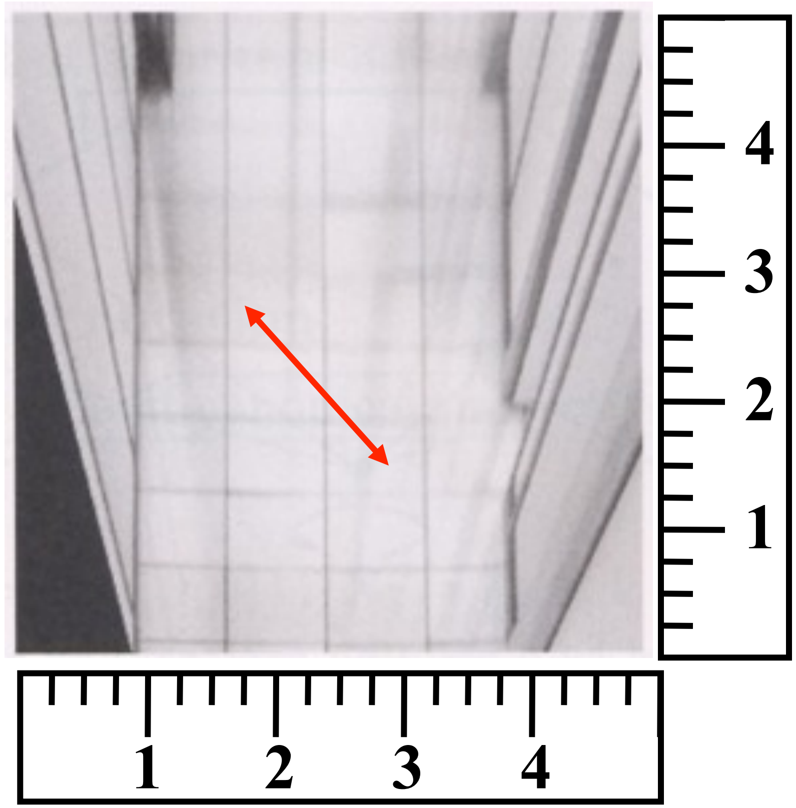
Approach: unwarp then measure

Measurements on planes



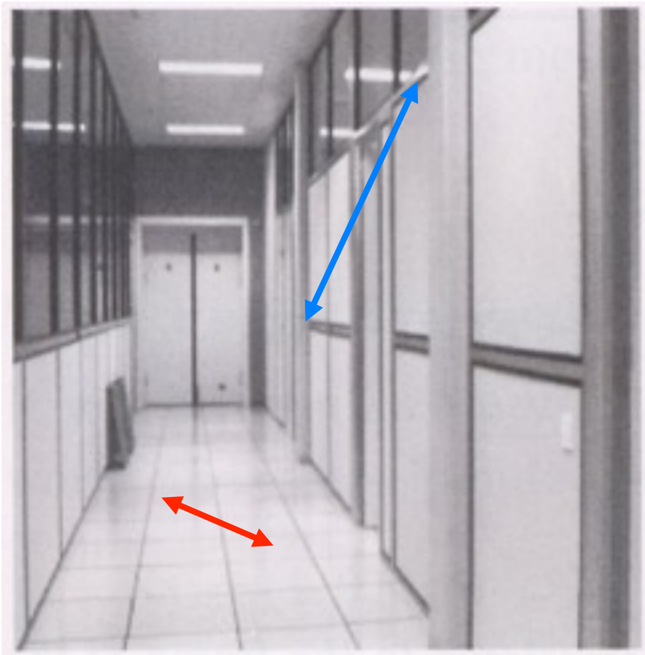
Approach: unwarp then measure

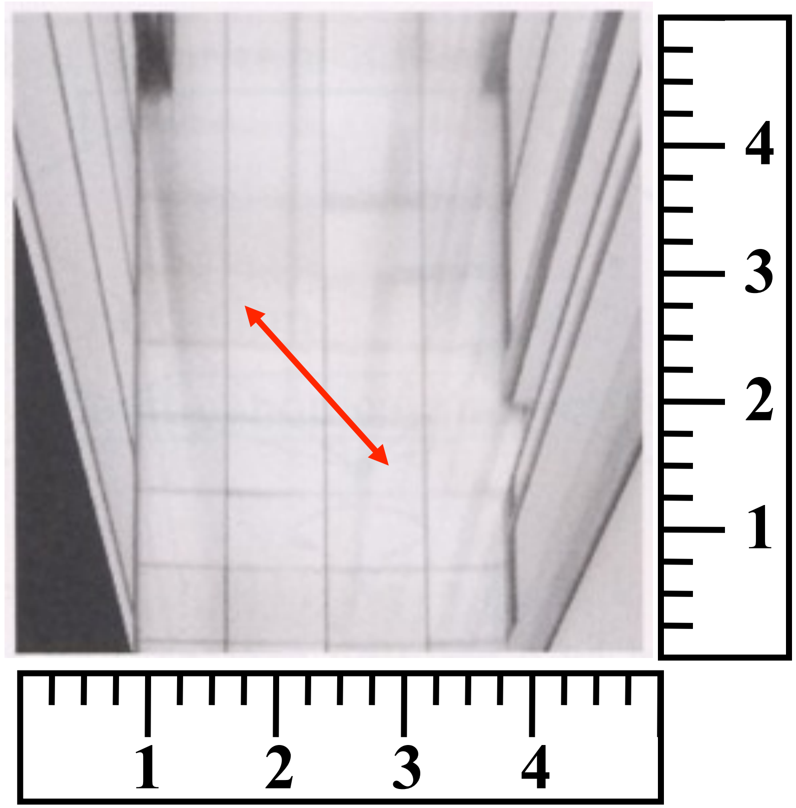
Measurements on planes



Approach: unwarp then measure

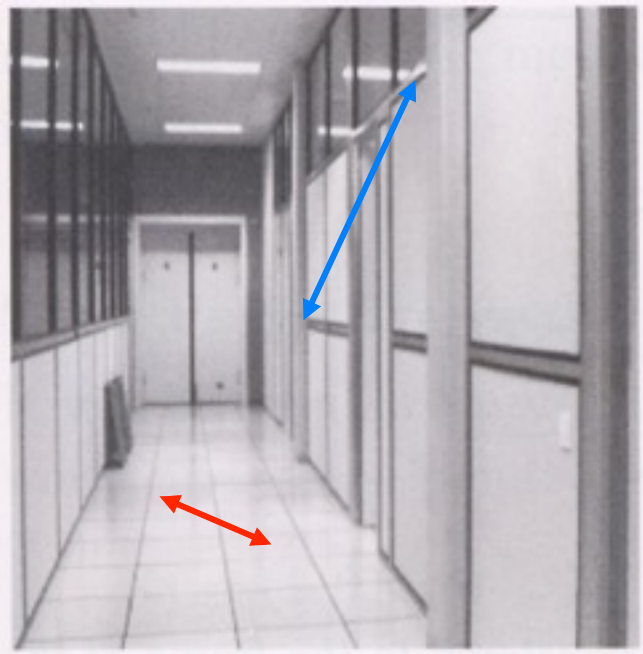
Measurements on planes

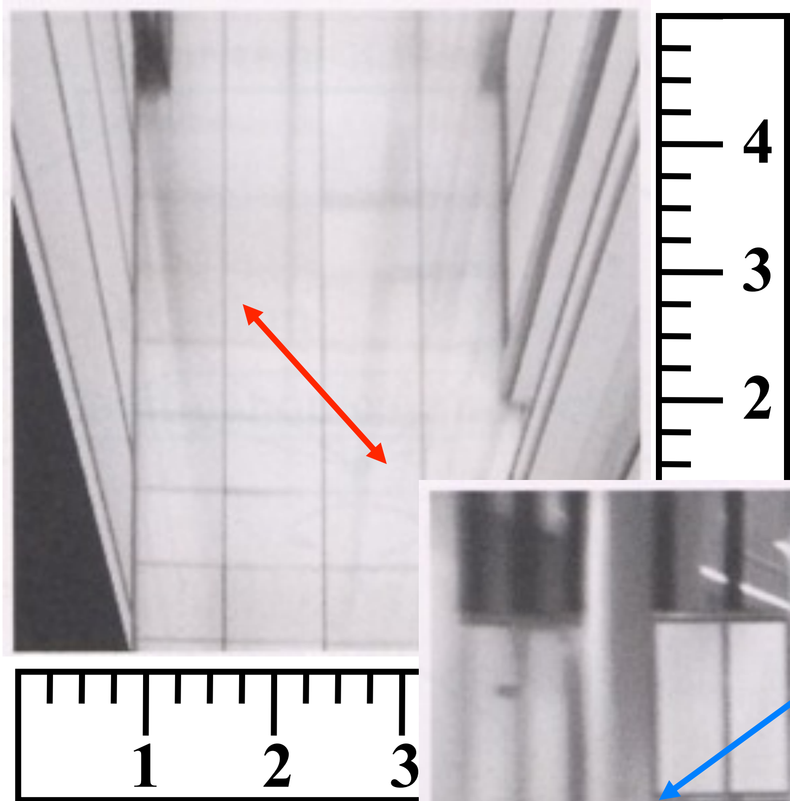


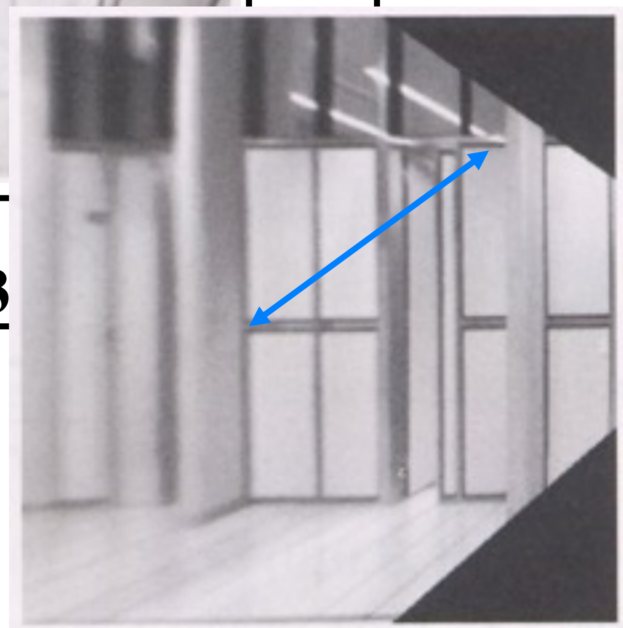


Approach: unwarp then measure

Measurements on planes

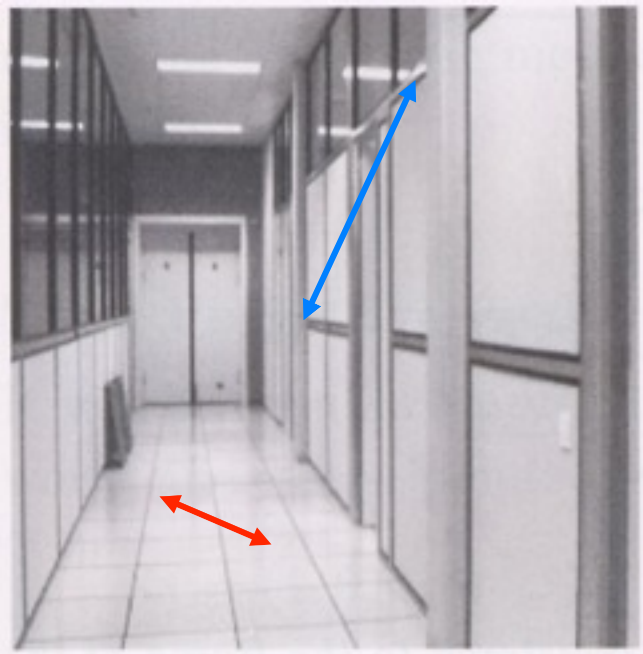


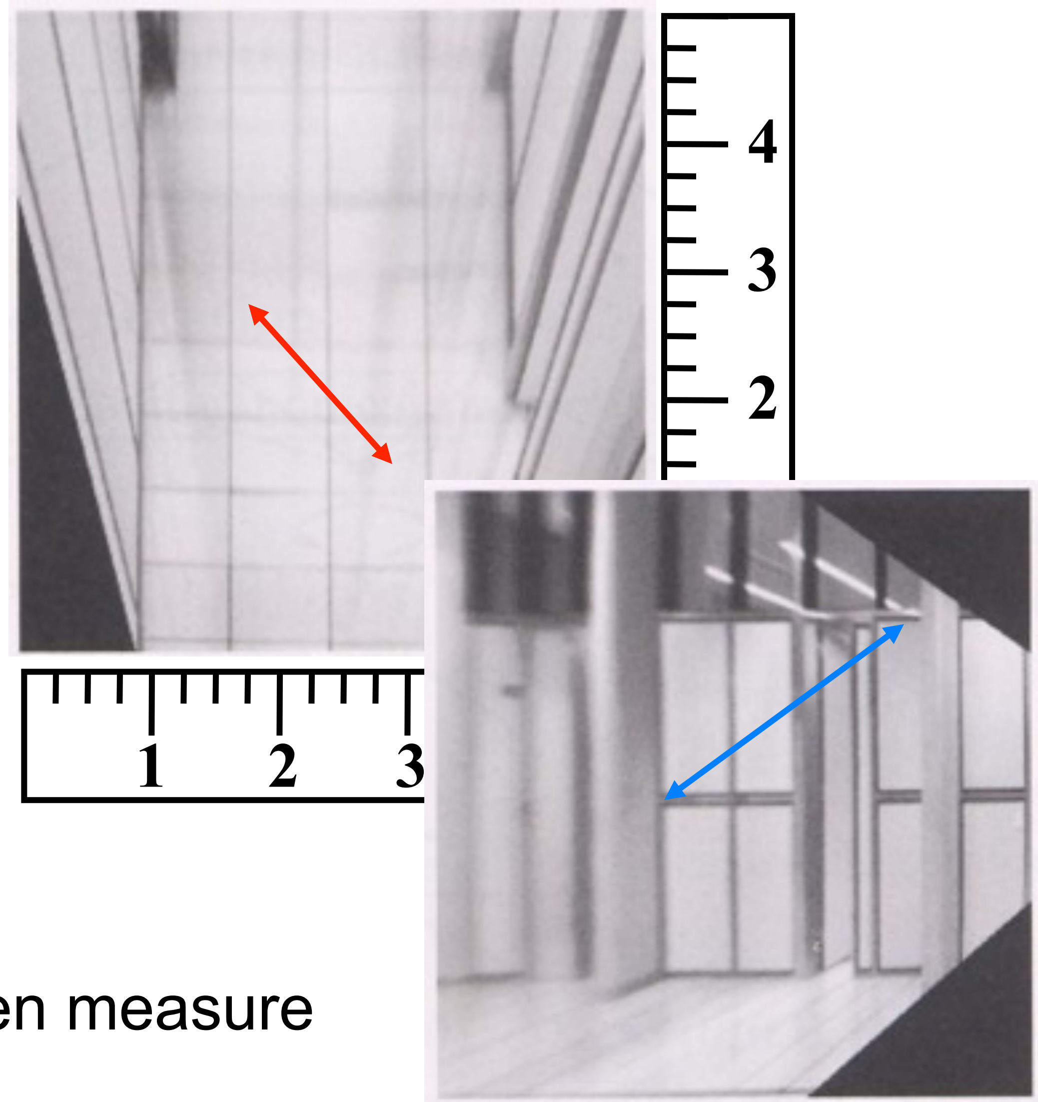




Approach: unwarp then measure

Measurements on planes

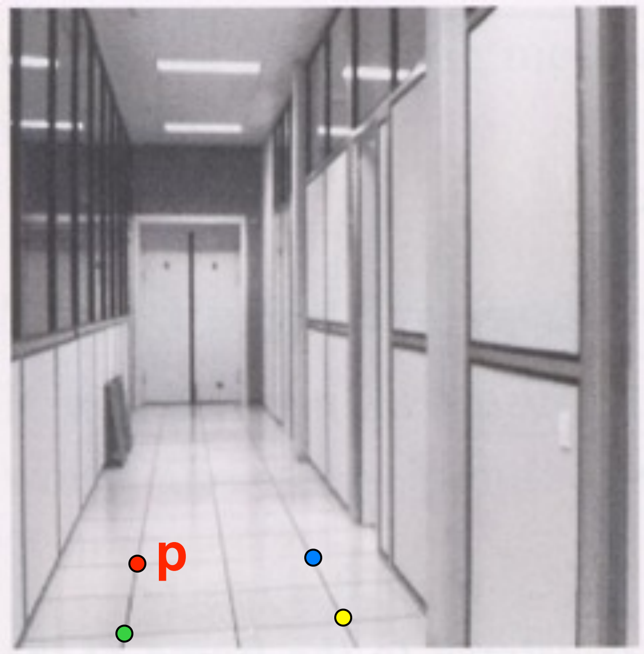


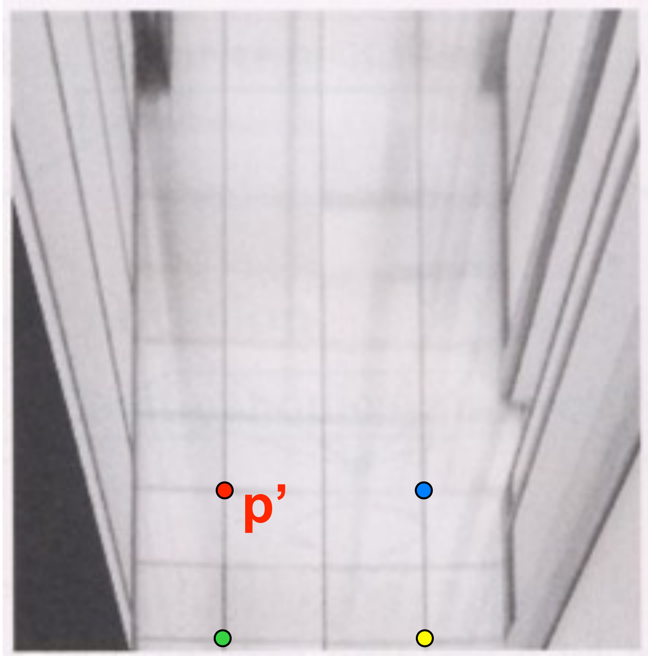


Approach: unwarp then measure

How to unwarp?

Image rectification





To unwarp (rectify) an image

- solve for homography **H** given **p** and **p’**
- solve equations of the form: w**p’** = **Hp**
  - linear in unknowns: w and coefficients of **H**
  - H is defined up to an arbitrary scale factor
  - how many points are necessary to solve for **H**?

Solving for homographies

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Solving for homographies

$$\begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$x_i' = \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

$$y_i' = \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

Solving for homographies

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{aligned} x'_i &= \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}} \\ y'_i &= \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}} \end{aligned}$$

$$\begin{aligned} x'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\ y'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12} \end{aligned}$$

Solving for homographies

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{aligned} x'_i &= \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}} \\ y'_i &= \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}} \end{aligned}$$

$$\begin{aligned} x'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\ y'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12} \end{aligned}$$

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -y'_i x_i & -y'_i y_i & -y'_i \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Solving for homographies

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & & \vdots & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

A **h** **0**

Solving for homographies

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

A 2n × 9 **h** 9 **0** 2n

Solving for homographies

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$\underset{2n \times 9}{\mathbf{A}} \qquad \underset{9}{\mathbf{h}} \qquad \underset{2n}{\mathbf{0}}$$

Defines a least squares problem: $\text{minimize } \|\mathbf{A}\mathbf{h} - \mathbf{0}\|^2$

Solving for homographies

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & & \vdots & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$\underset{2n \times 9}{\mathbf{A}} \qquad \underset{9}{\mathbf{h}} \qquad \underset{2n}{\mathbf{0}}$$

Defines a least squares problem: $\text{minimize } \|\mathbf{A}\mathbf{h} - \mathbf{0}\|^2$

- Since **h** is only defined up to scale, solve for unit vector $\hat{\mathbf{h}}$

Solving for homographies

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

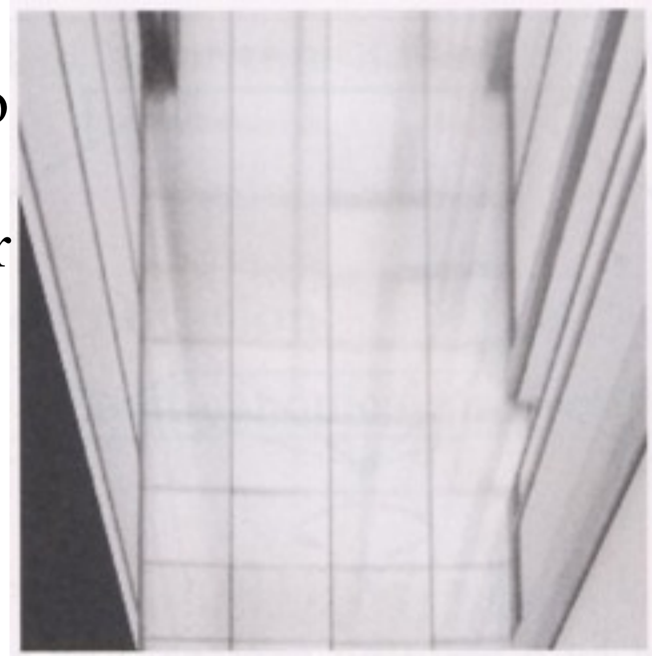
A $2n \times 9$ **h** 9 **0** $2n$

Defines a least squares problem: $\text{minimize } \|\mathbf{A}\mathbf{h} - \mathbf{0}\|^2$

- Since **h** is only defined up to scale, solve for unit vector $\hat{\mathbf{h}}$
- Solution: $\hat{\mathbf{h}}$ = eigenvector of $\mathbf{A}^T\mathbf{A}$ with smallest eigenvalue
- Works with 4 or more points

Image warping with homographies

homography so that image is parallel to floor



homography so that image is parallel to right wall

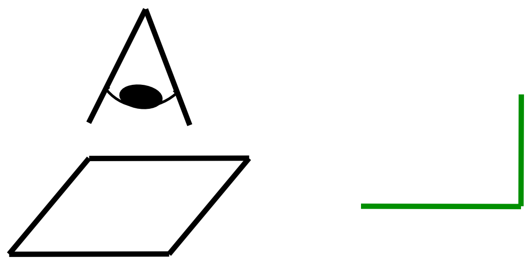
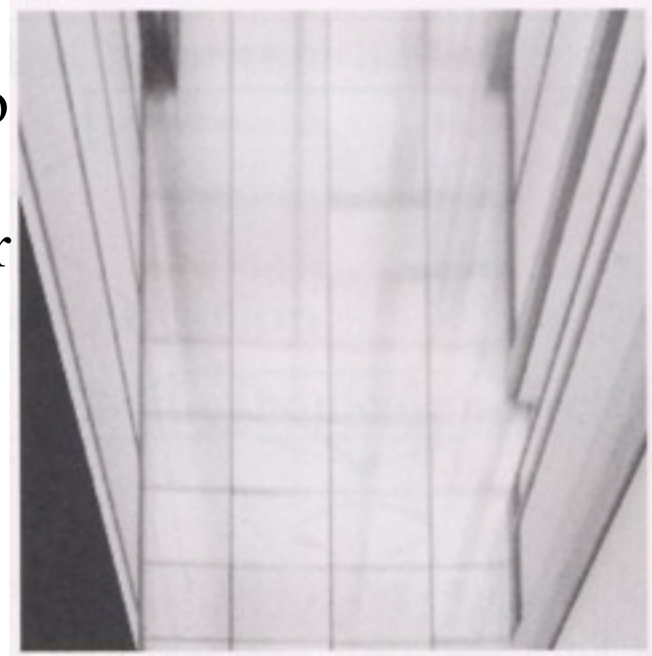


image plane below

Image warping with homographies

homography so that image is parallel to floor



homography so that image is parallel to right wall

black area where no pixel maps to

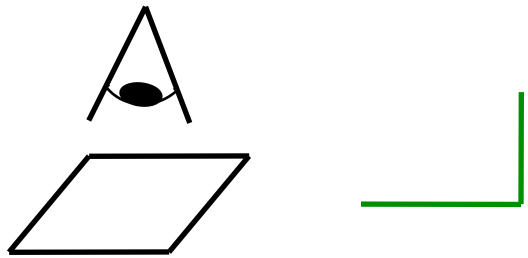


image plane below

Image warping with homographies

homography so that image is parallel to floor

homography so that image is parallel to right wall

black area where no pixel maps to

automatic image mosaicing

automatic image mosaicing

- **Basic Procedure**
  - Take a sequence of images **from the same position.**
    - Rotate the camera about its optical center (entrance pupil).
  - Robustly compute the homography transformation between second image and first.
  - Transform (warp) the second image to overlap with first.
  - Blend the two together to create a mosaic.
  - If there are more images, repeat.

Robust feature matching through RANSAC

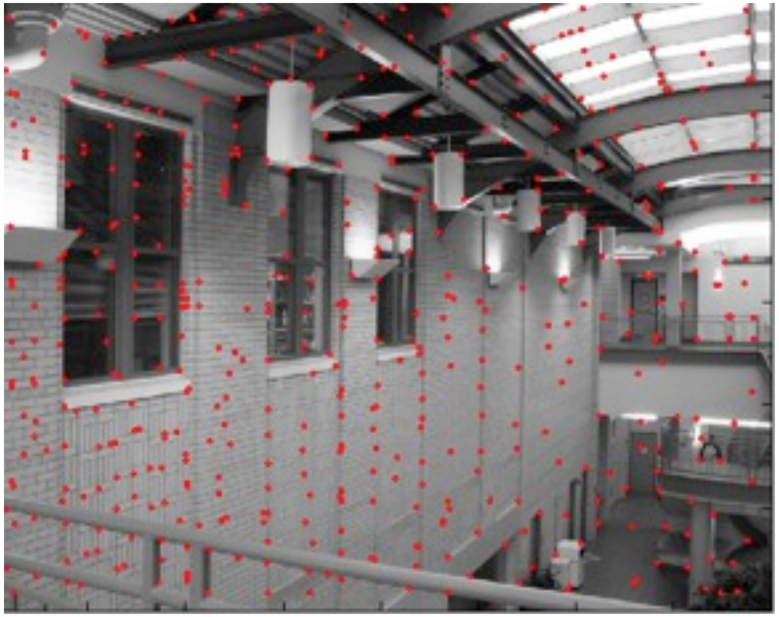
© **Krister Parmstrand**

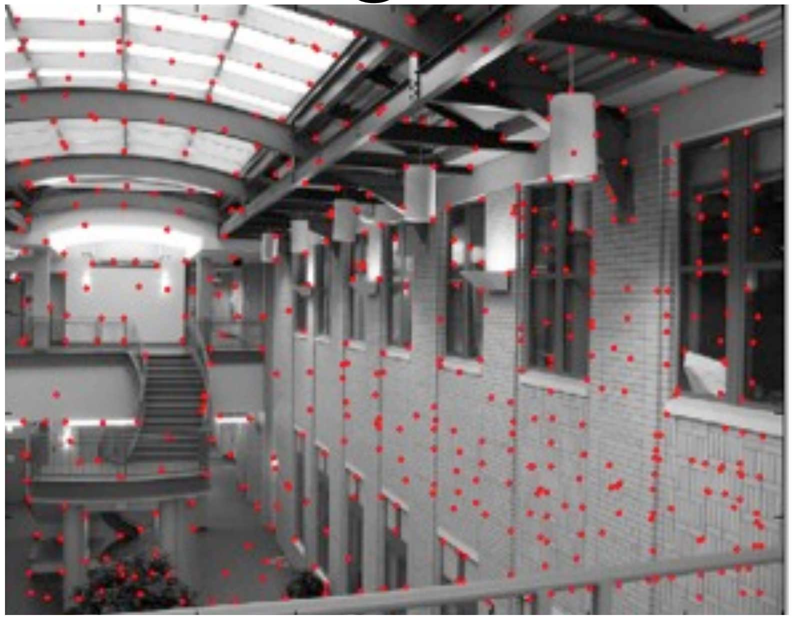
Nikon D70. Stitched Panorama. The sky has been retouched. No other image manipulation.

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

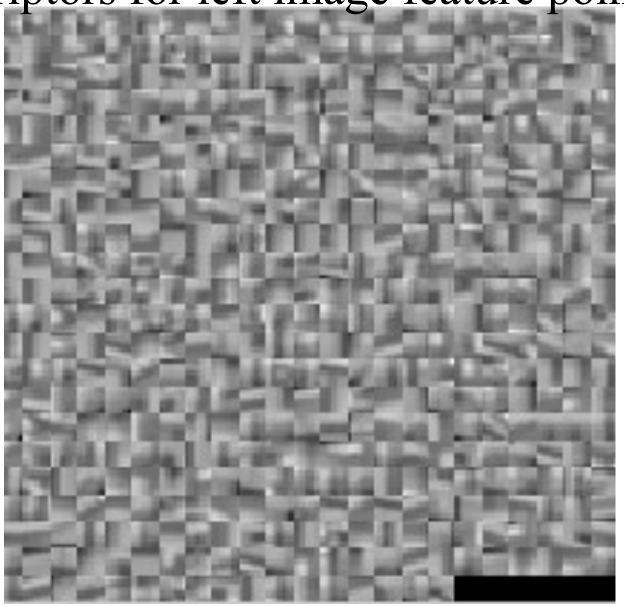
15-463: Computational Photography
Alexei Efros, CMU, Fall 2005

Feature matching



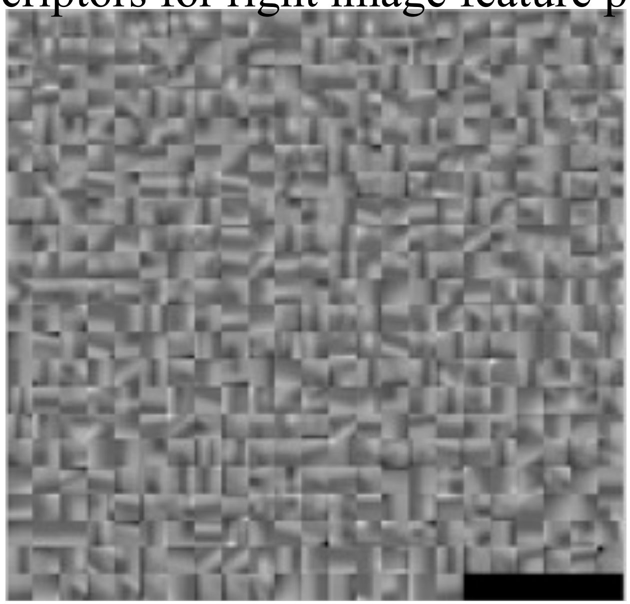


descriptors for left image feature points



?

descriptors for right image feature points



Strategies to match images robustly

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

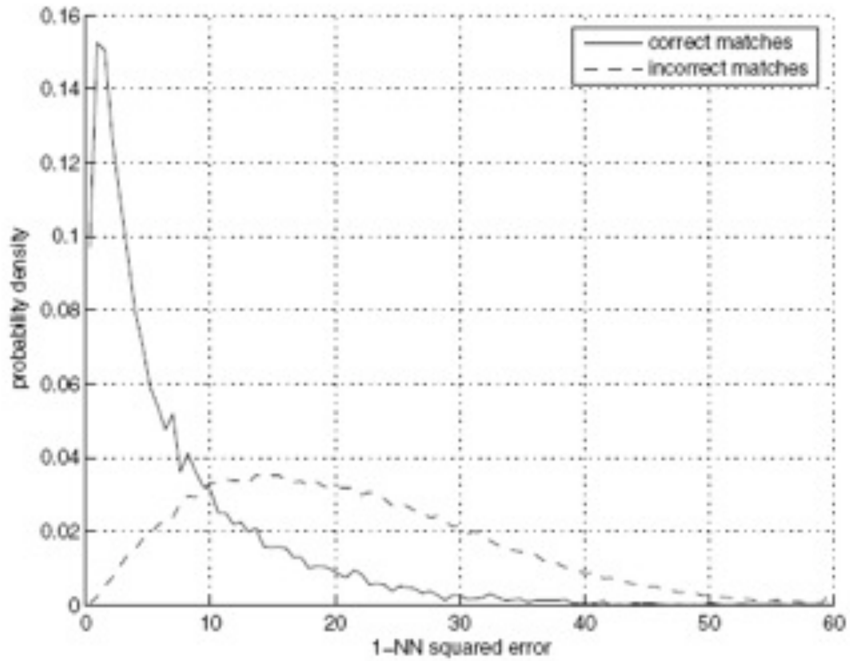
Reject ambiguous matches where there are too many similar points

(b) <u>Working with all the features:</u> 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.

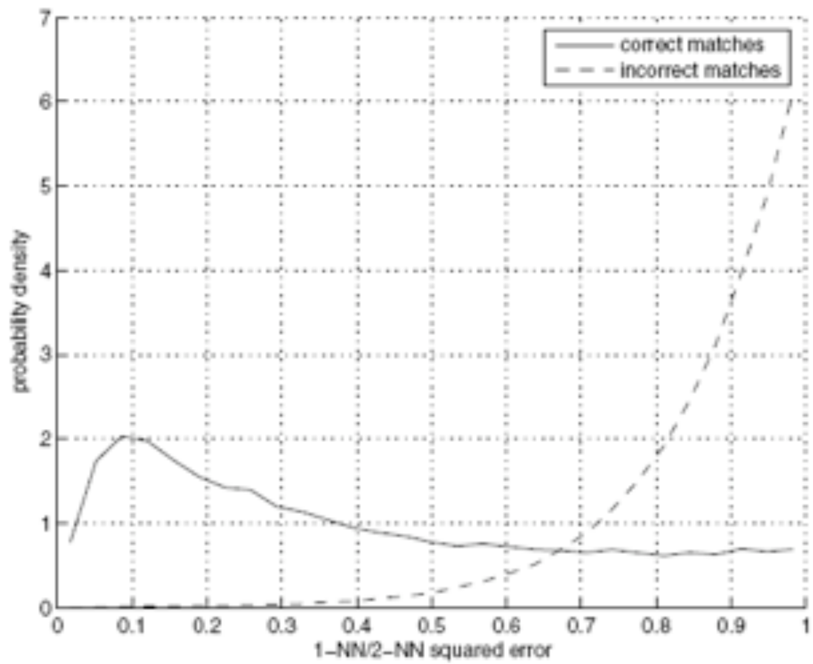


Feature-space outlier rejection

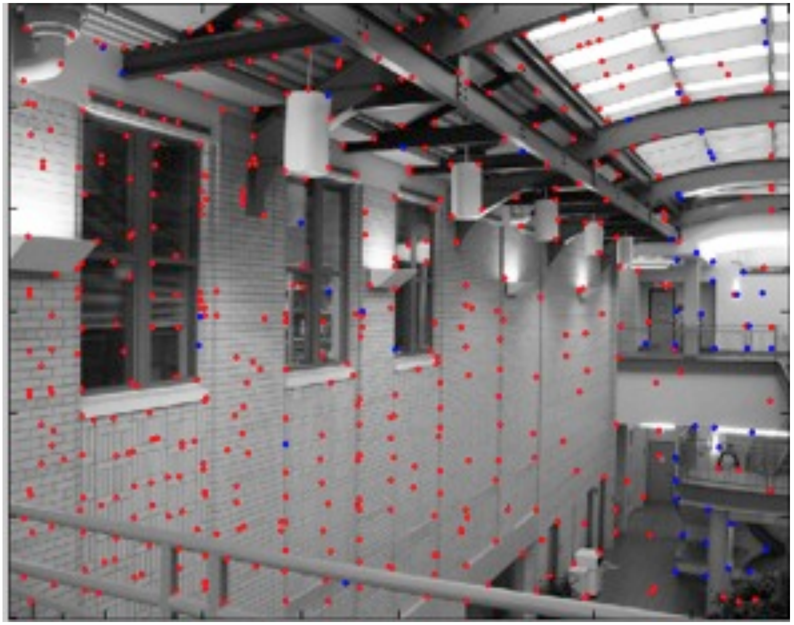
- A better way [Lowe, 1999]:
  - 1-NN: SSD of the closest match
  - 2-NN: SSD of the <u>second-closest</u> match
  - Look at how much better 1-NN is than 2-NN, e.g. 1-NN/2-NN
  - That is, is our best match so much better than the rest?

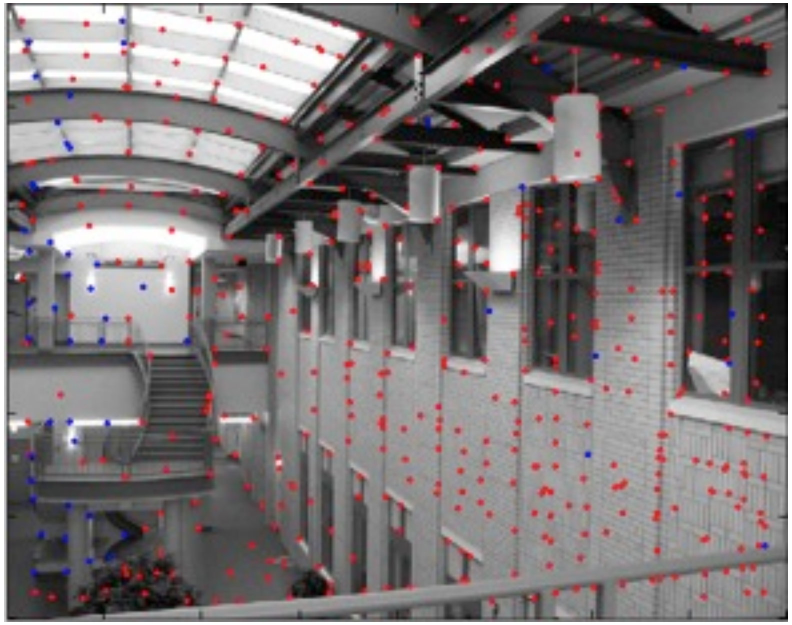
Feature-space outlier rejection

- A better way [Lowe, 1999]:
  - 1-NN: SSD of the closest match
  - 2-NN: SSD of the <u>second-closest</u> 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?

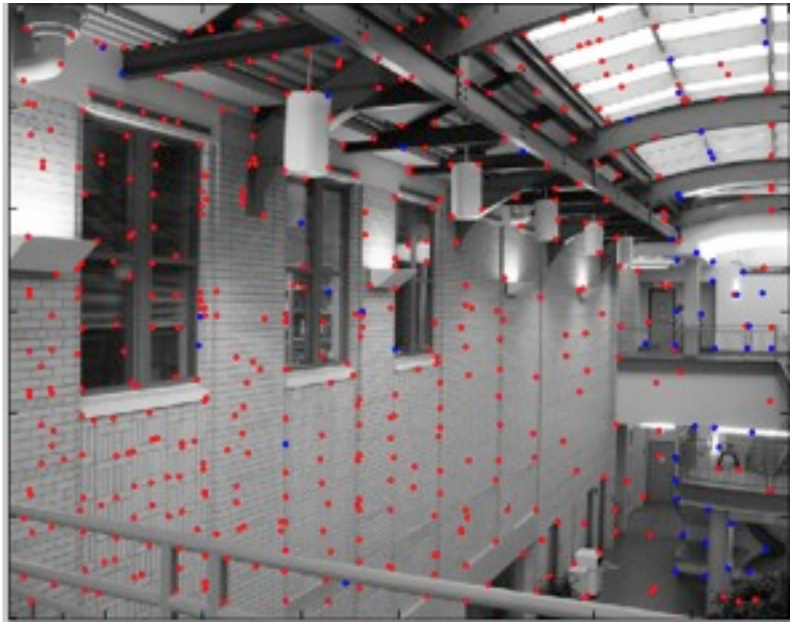
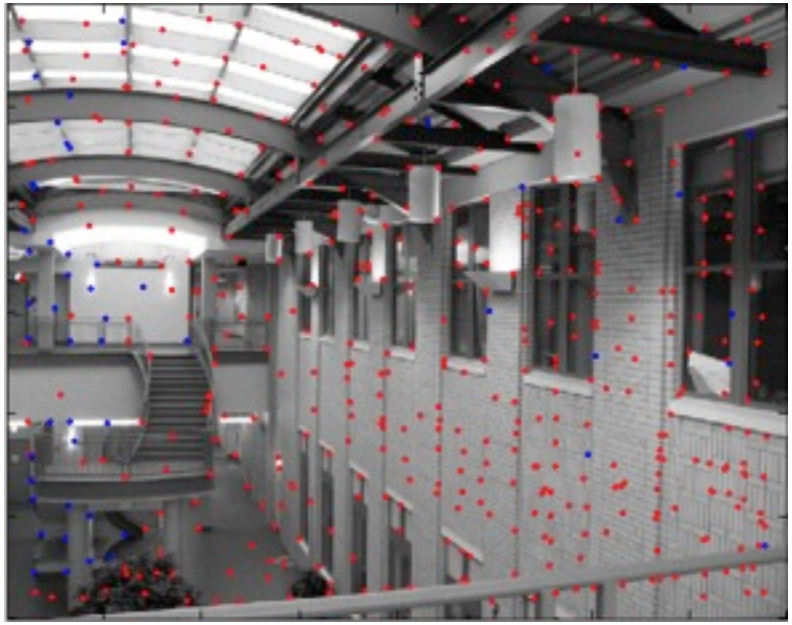


Feature-space outlier rejection



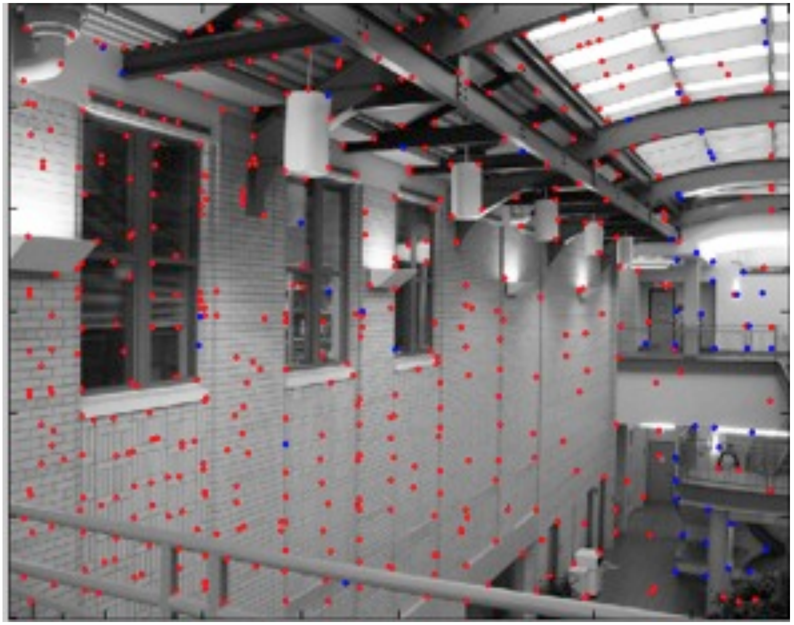


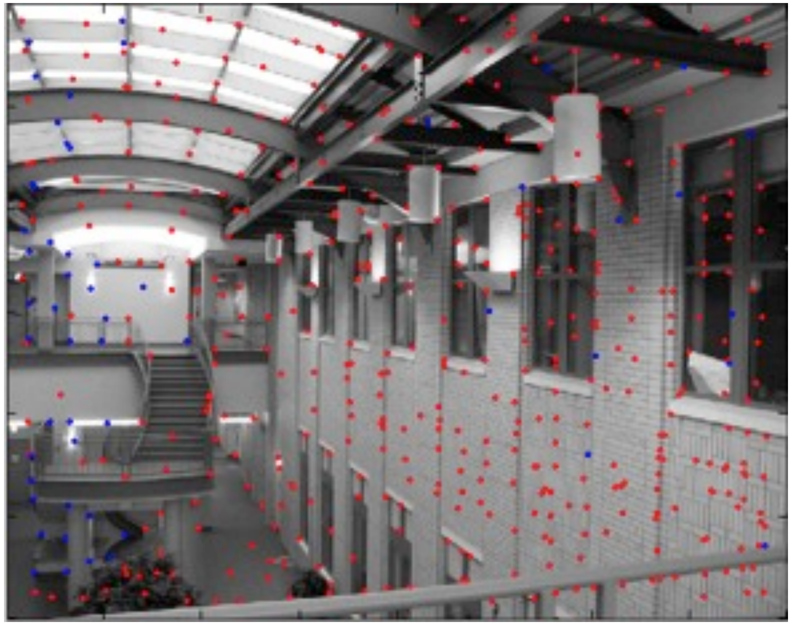
Feature-space outlier rejection

- Can we now compute H from the blue points?

Feature-space outlier rejection

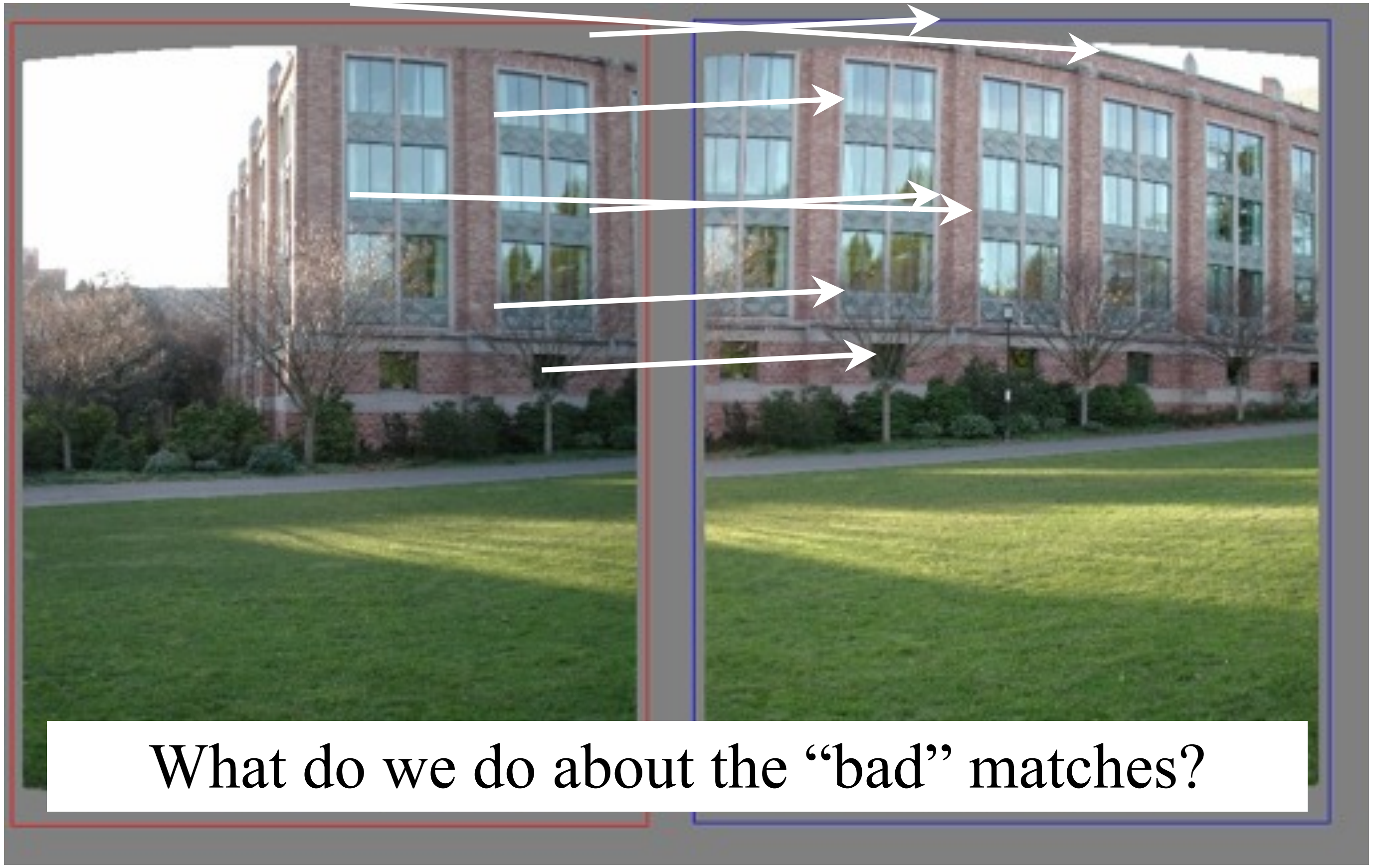




- Can we now compute H from the blue points?
  - No! Still too many outliers…
  - What can we do?

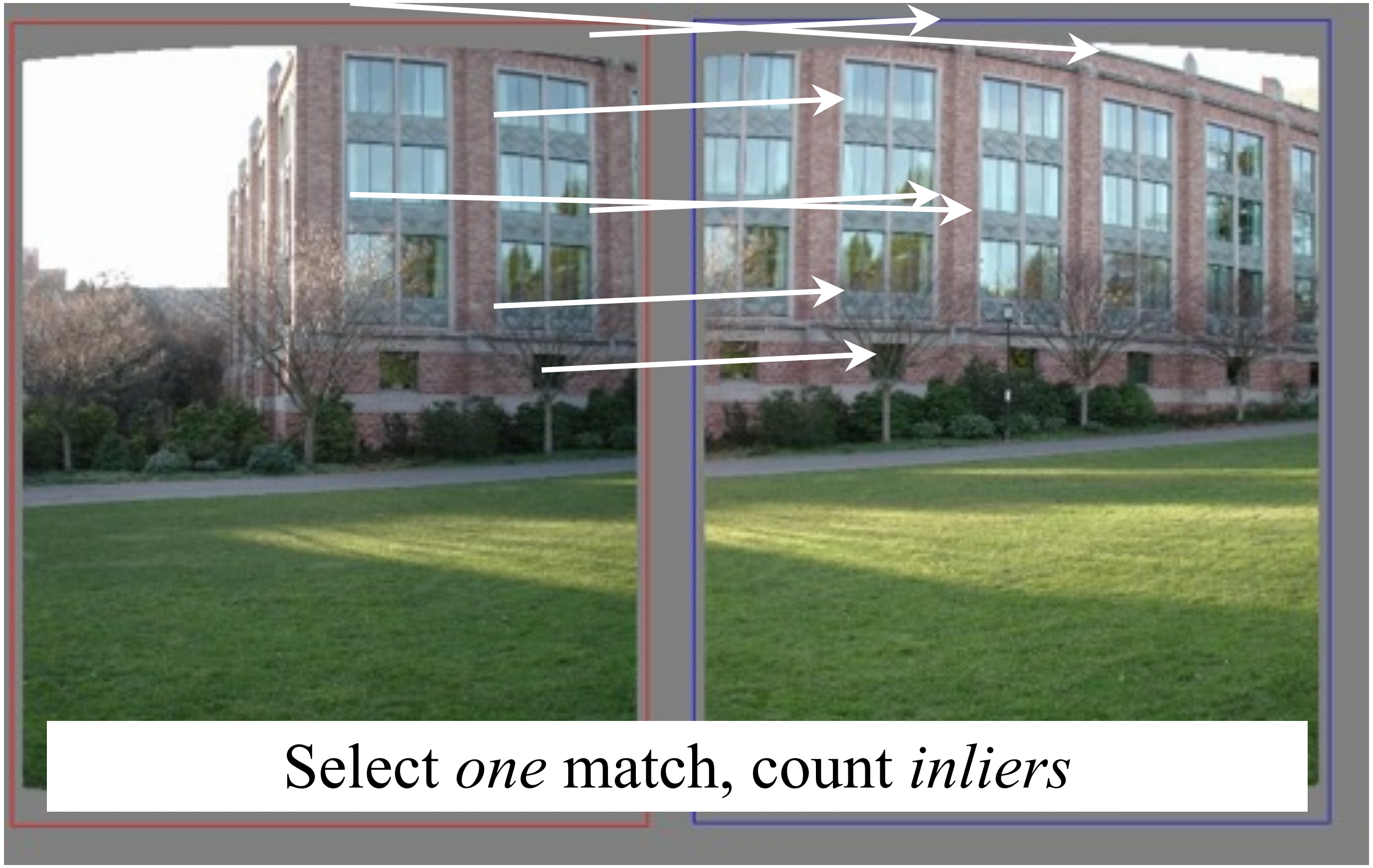
(b) Matching many features--looking for a good homography

Simplified illustration with translation instead of homography



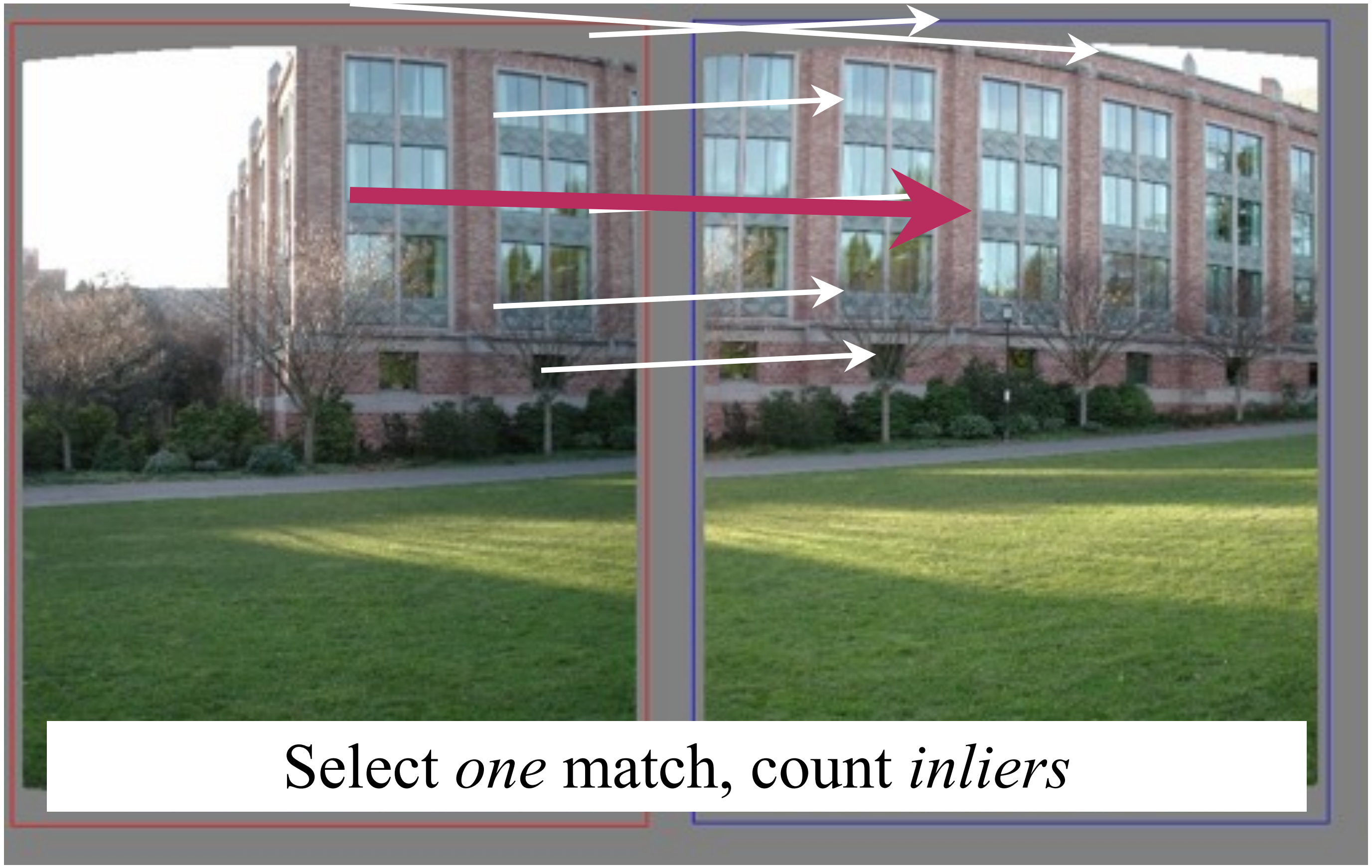
Note: at this point we don’t know which ones are good/bad

RAndom SAmple Consensus



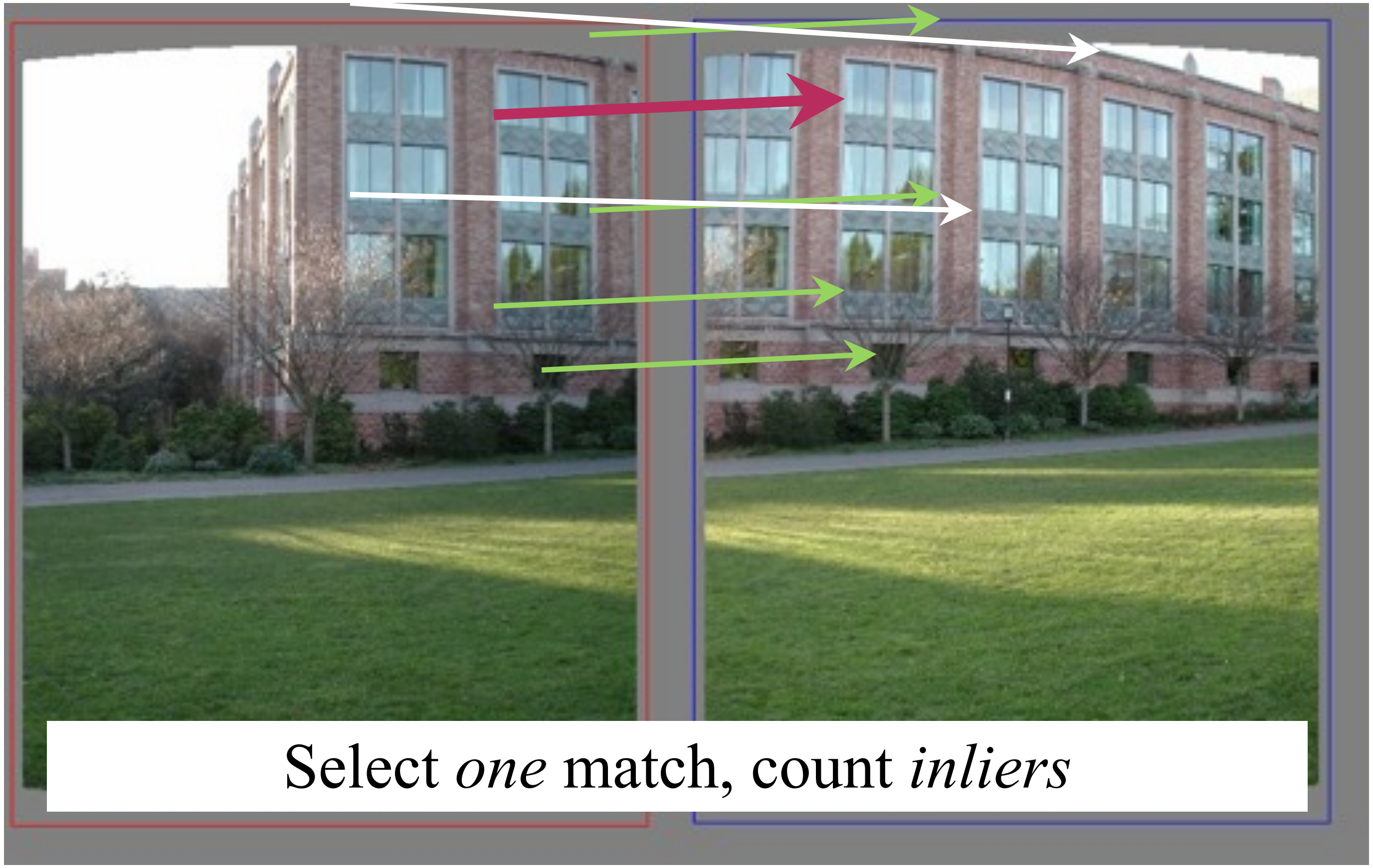
Select *one* match, count *inliers*

RAndom SAmple Consensus



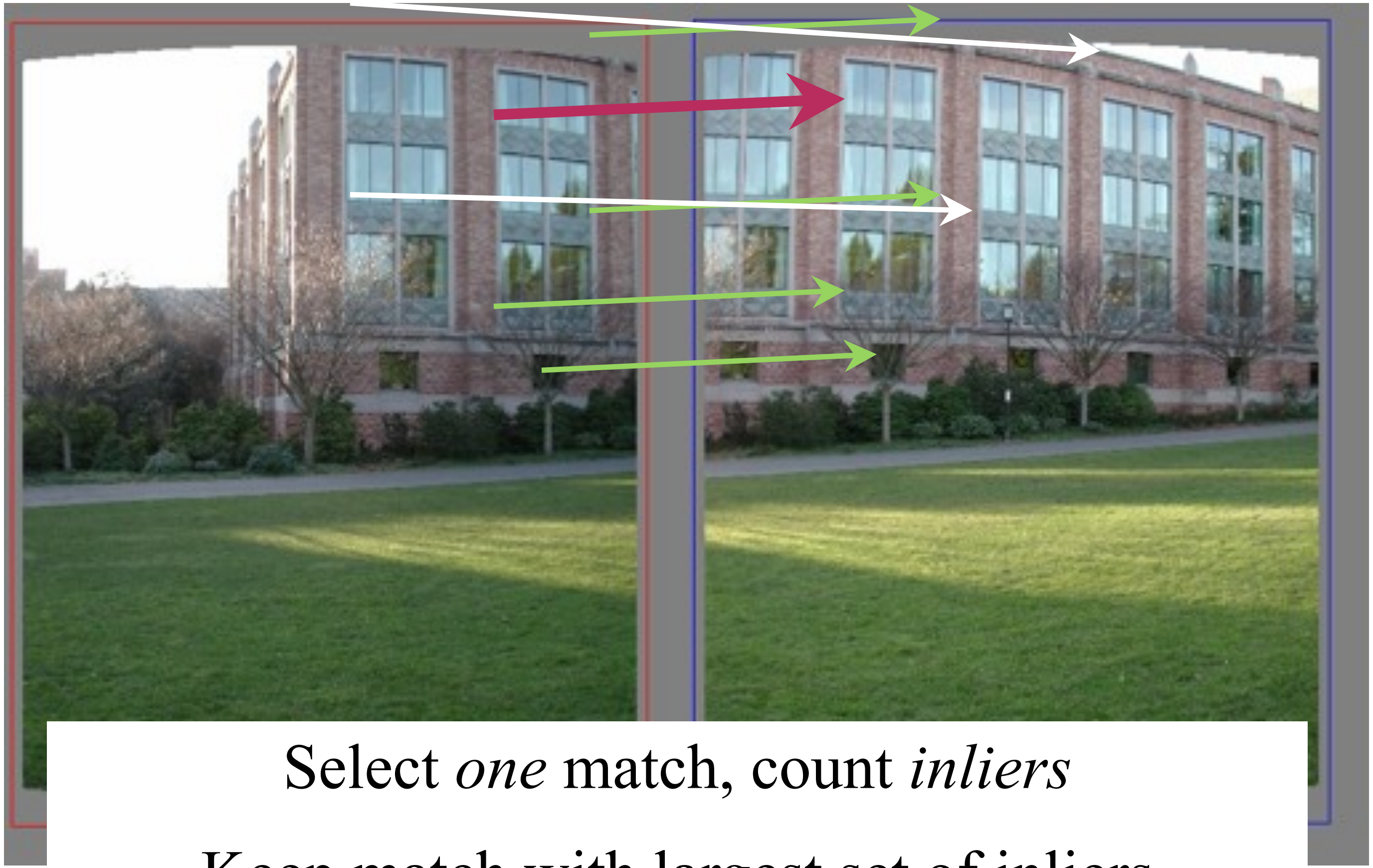
0 inliers

RAndom SAmple Consensus



4 inliers

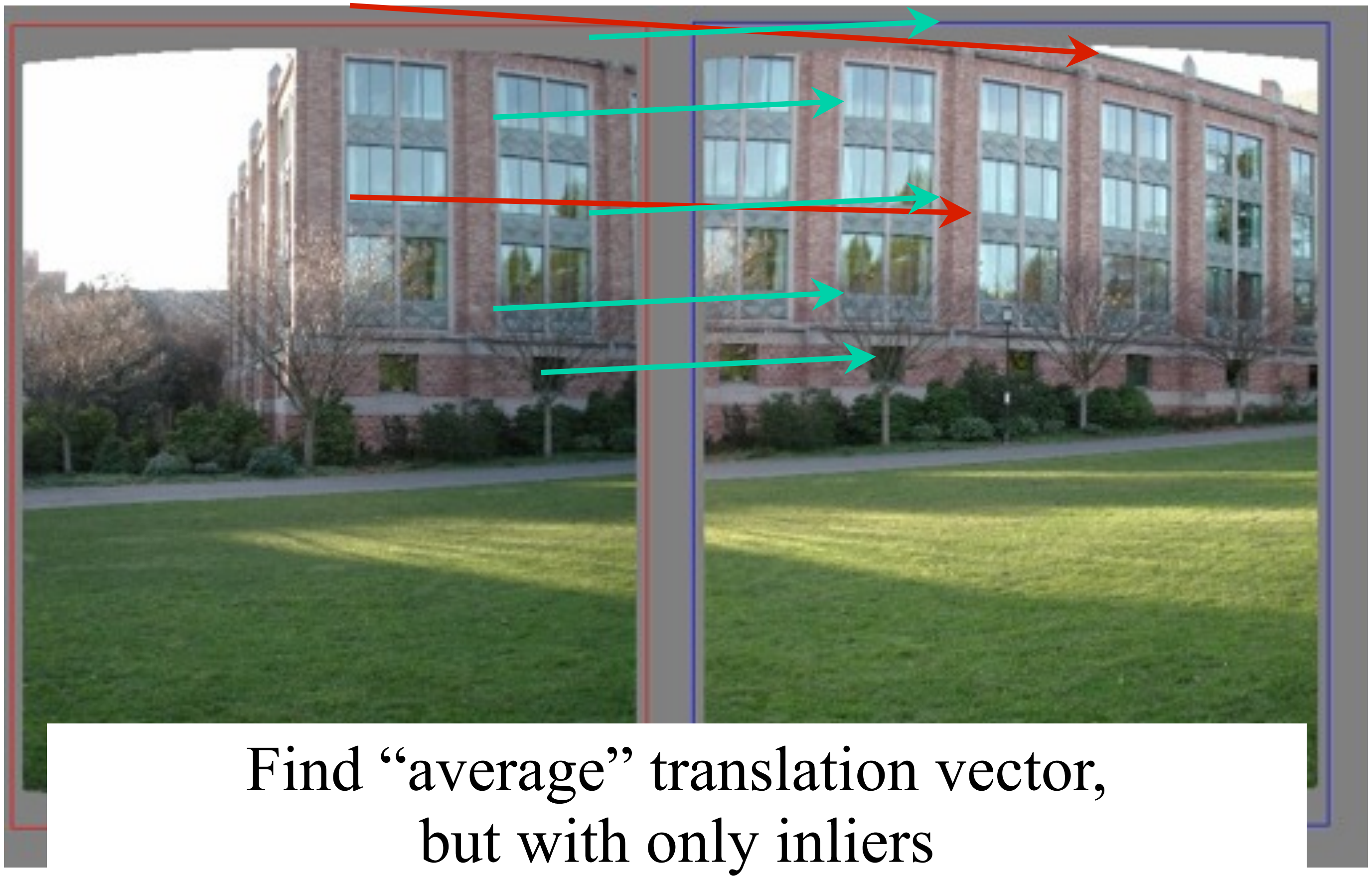
RAndom SAmple Consensus



Select *one* match, count *inliers*

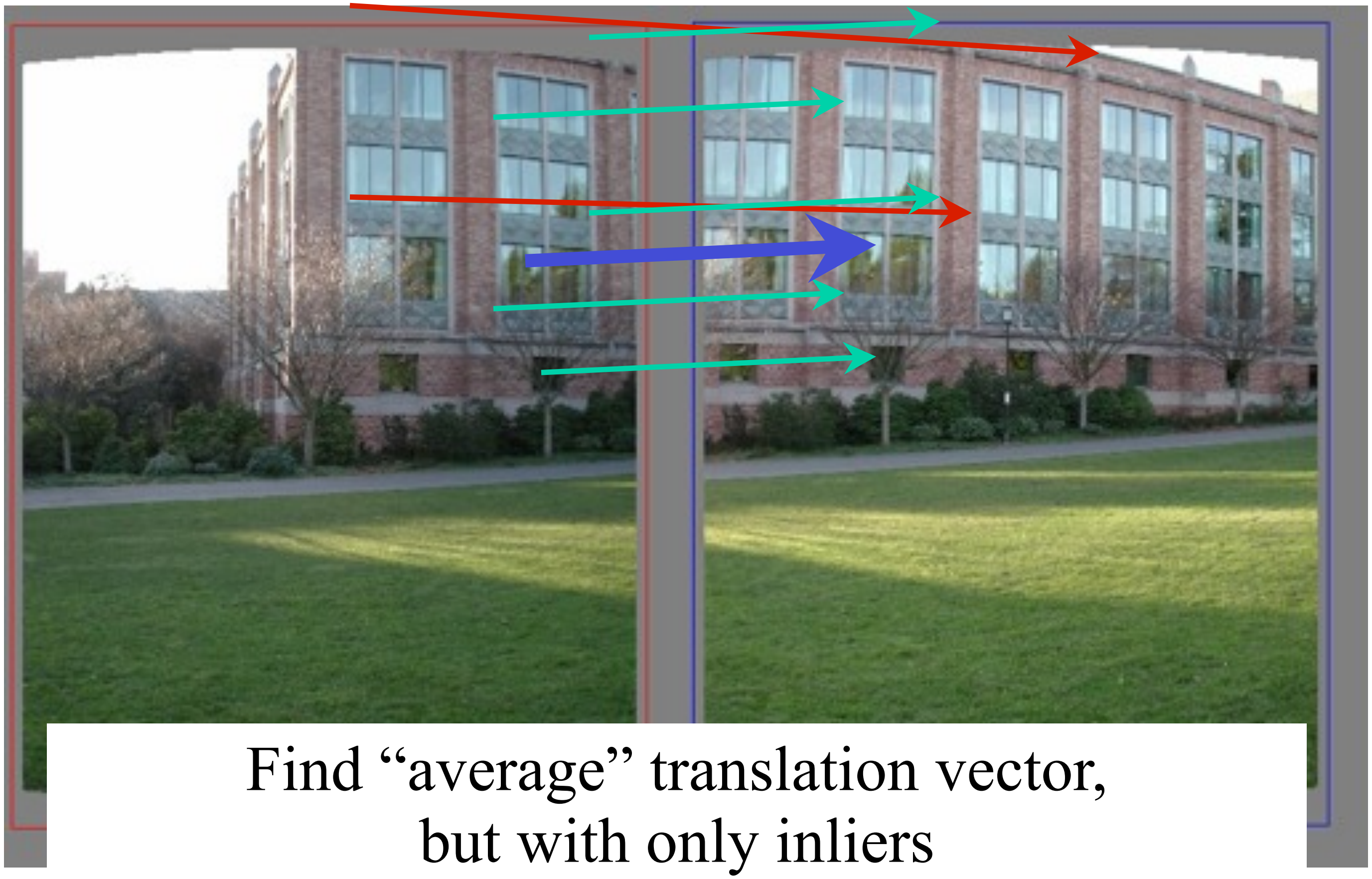
Keep match with largest set of inliers

At the end: Least squares fit



Find “average” translation vector,
but with only inliers

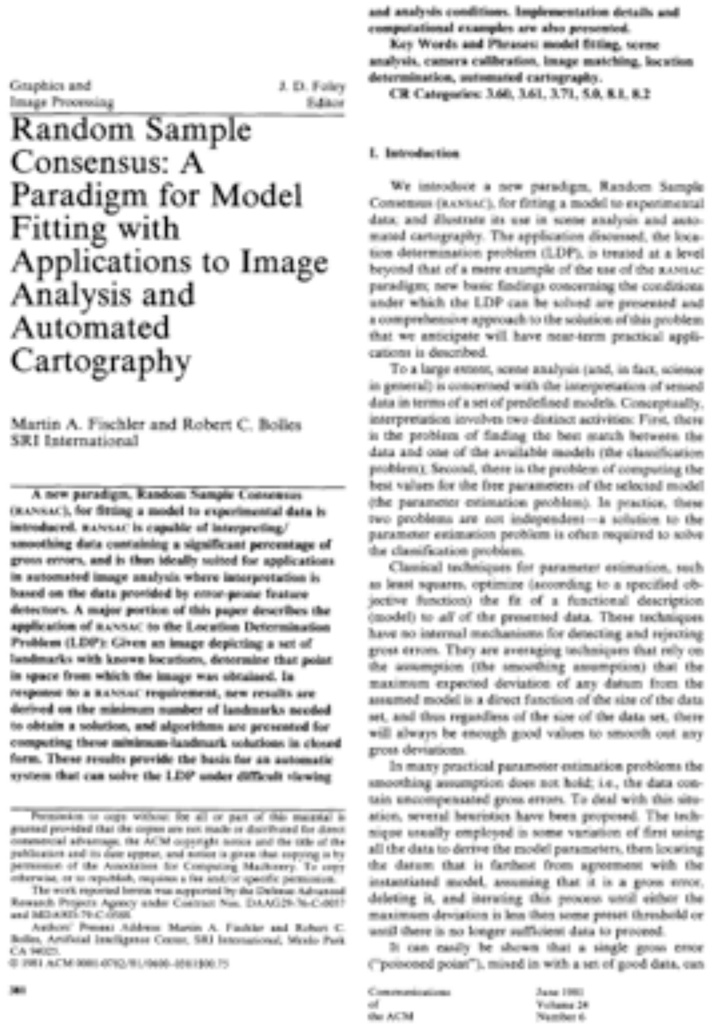
At the end: Least squares fit



Find “average” translation vector,
but with only inliers

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



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

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.

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

RANSAC for estimating homography

RANSAC for estimating homography

RANSAC loop:

RANSAC for estimating homography

RANSAC loop:

- Select four feature pairs (at random)

RANSAC for estimating homography

RANSAC loop:

- Select four feature pairs (at random)
- Compute homography H (exact)

RANSAC for estimating homography

RANSAC loop:

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

RANSAC for estimating homography

RANSAC loop:

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

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

RANSAC for estimating homography

RANSAC loop:

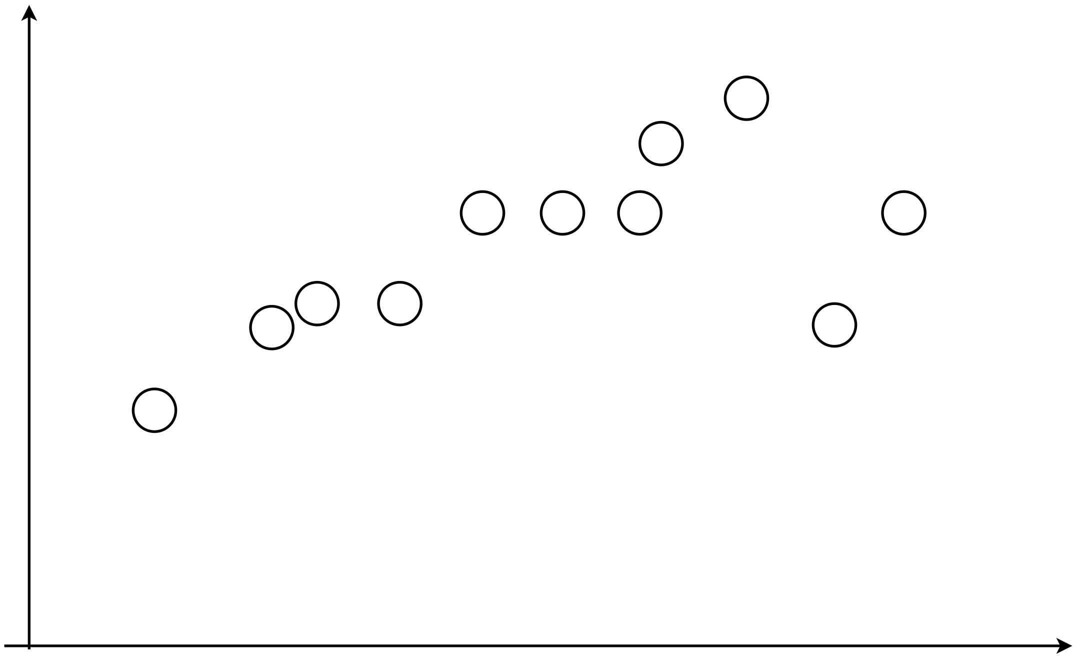
- Select four feature pairs (at random)
- Compute homography H (exact)
- Compute *inliers* where $\|p_i', \boldsymbol{H}\, p_i\| < \varepsilon$

Keep largest set of inliers

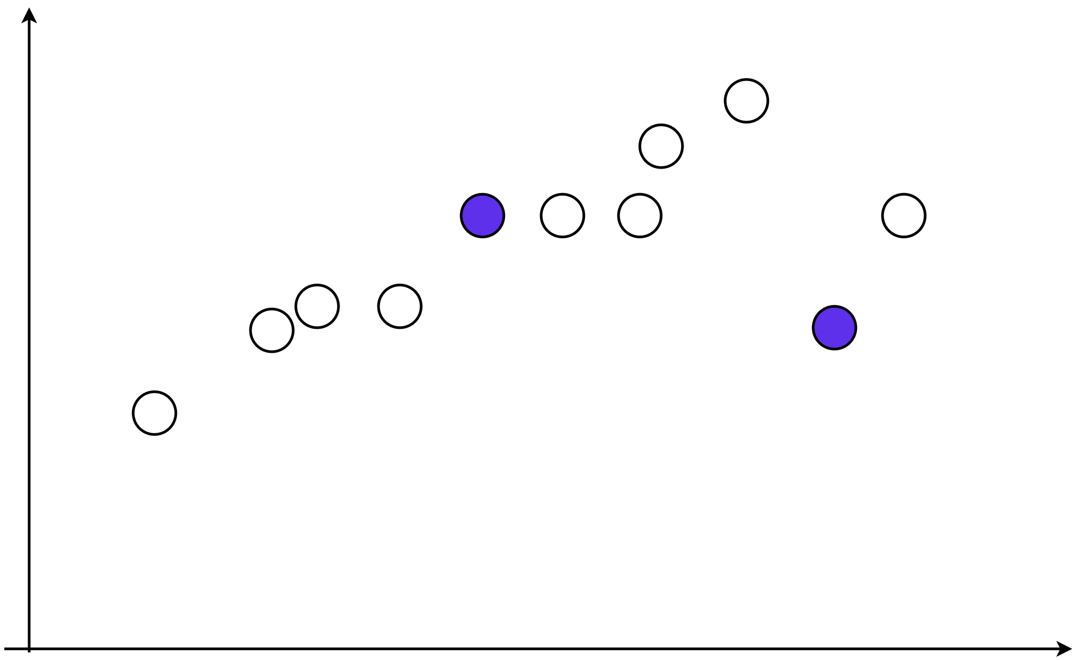
Re-compute least-squares H estimate using all of the inliers

Simple example: fit a line

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

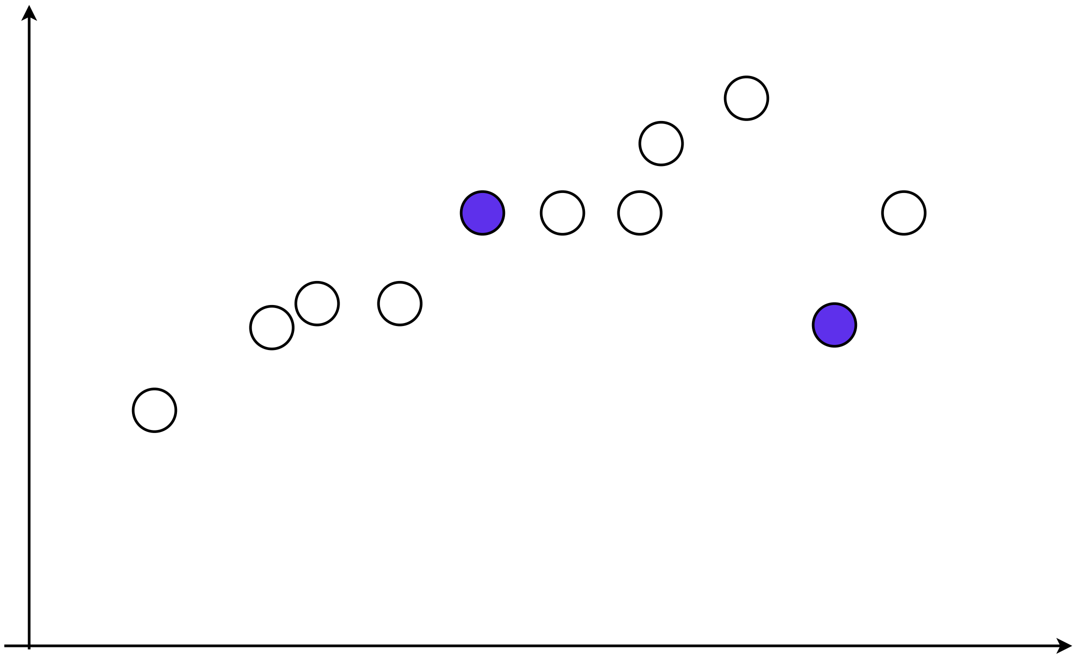


Simple example: fit a line



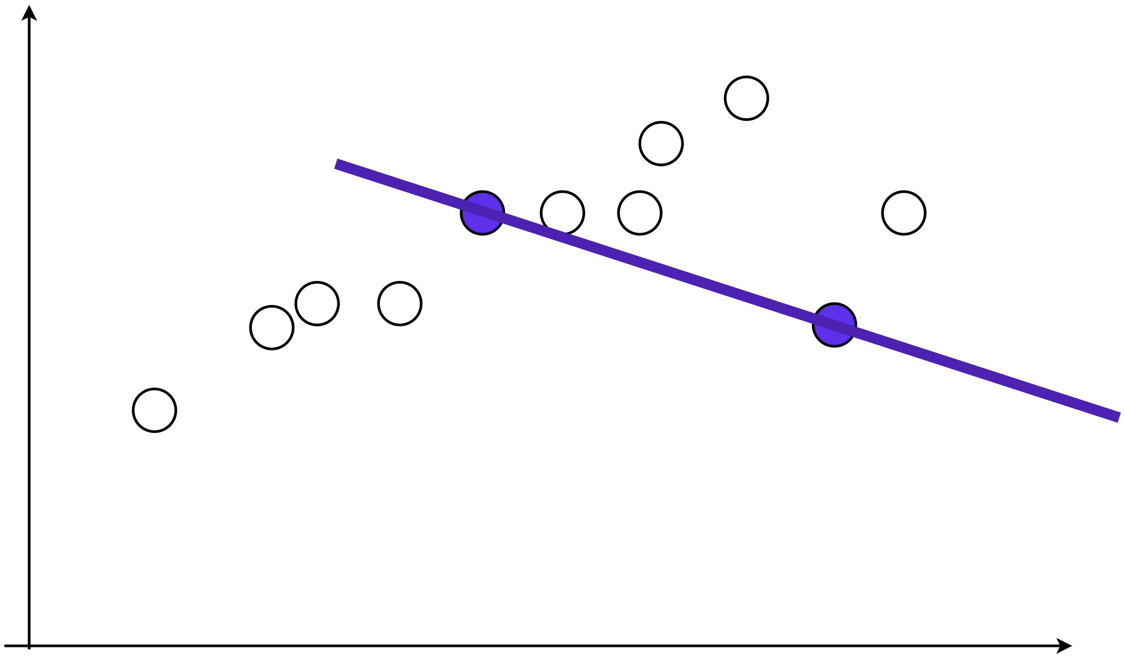
Simple example: fit a line

- Pick 2 points



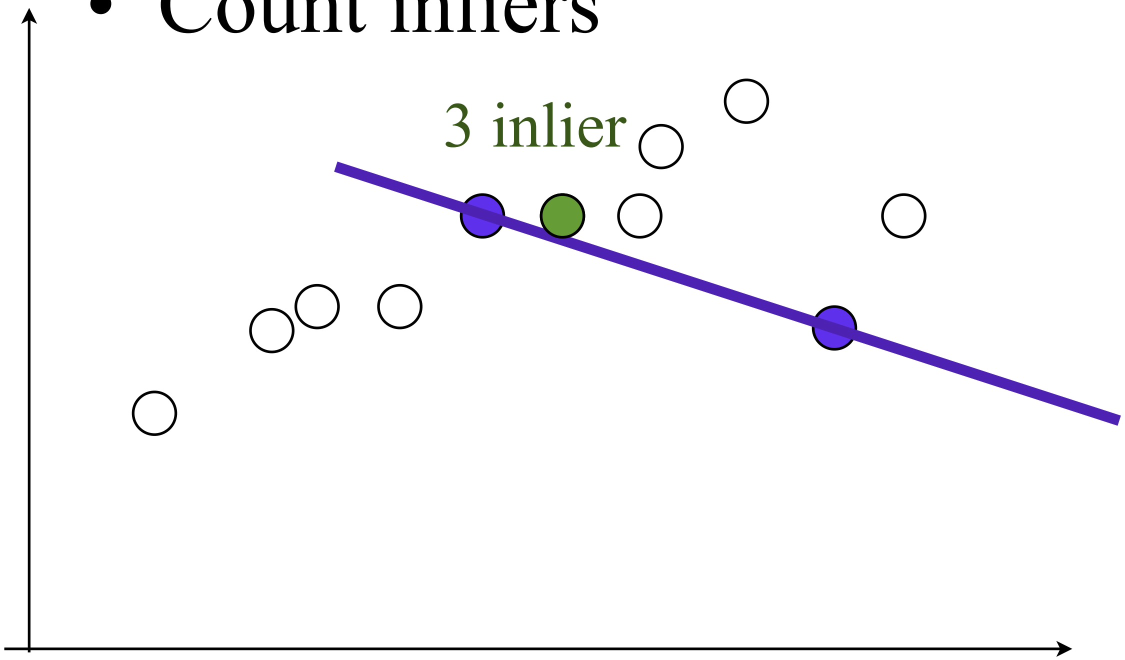
Simple example: fit a line

- Pick 2 points
- Fit line

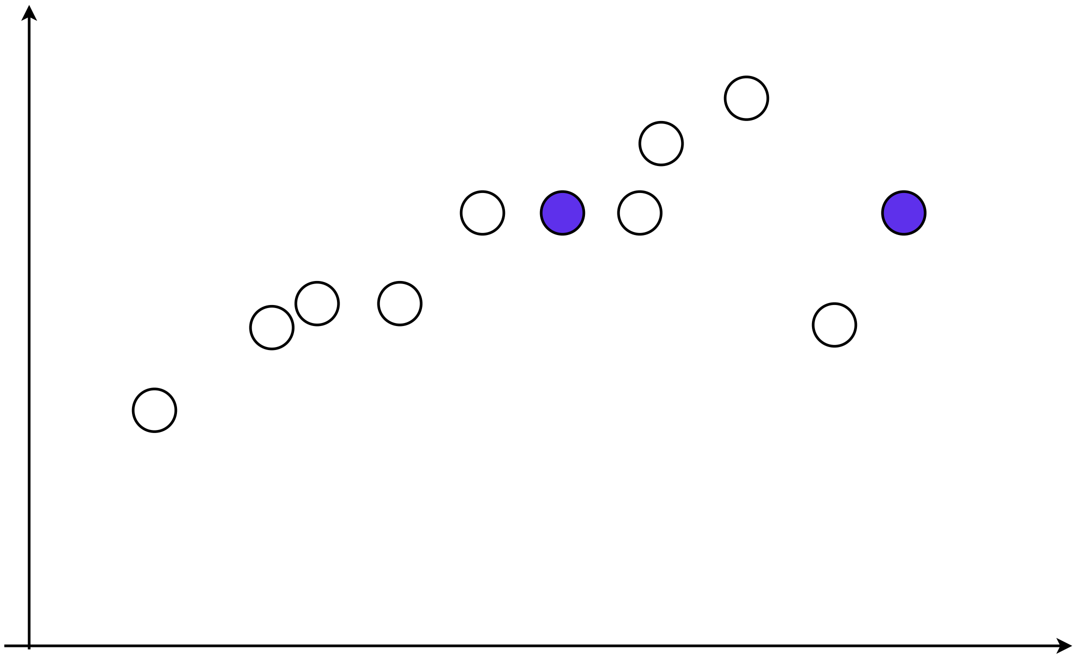


Simple example: fit a line

- Pick 2 points
- Fit line
- Count inliers

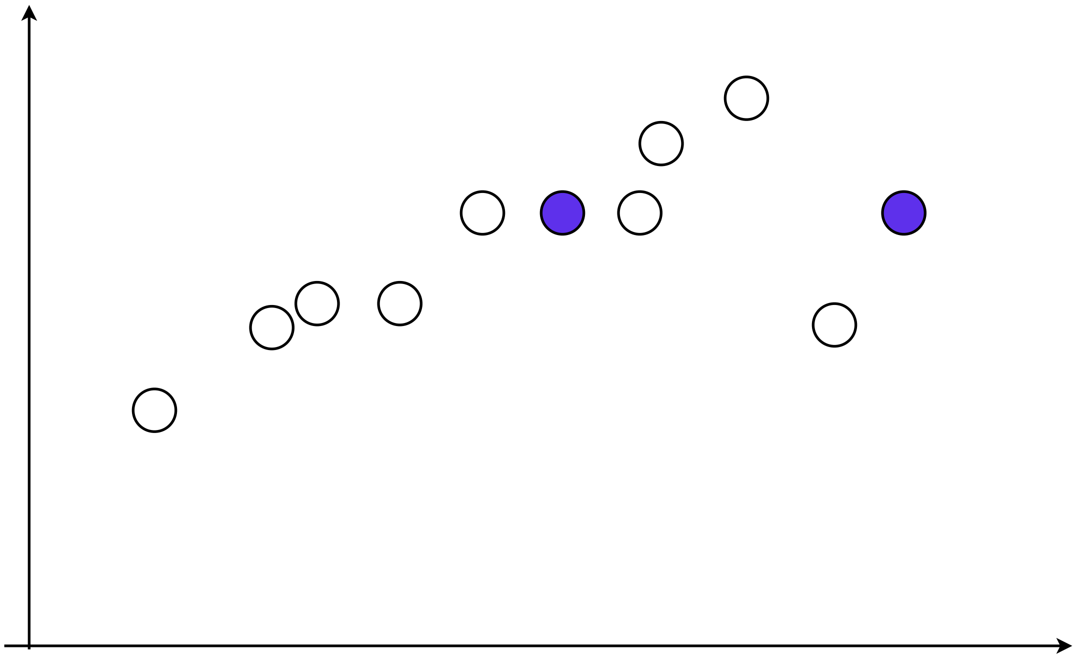


Simple example: fit a line



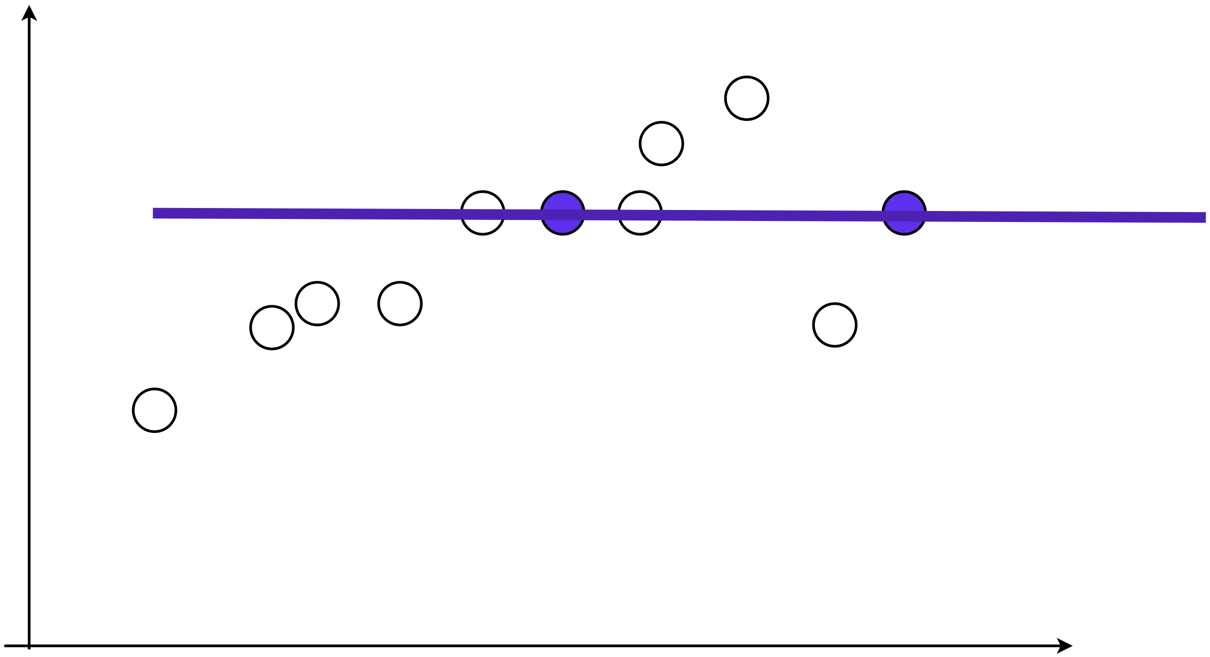
Simple example: fit a line

- Pick 2 points



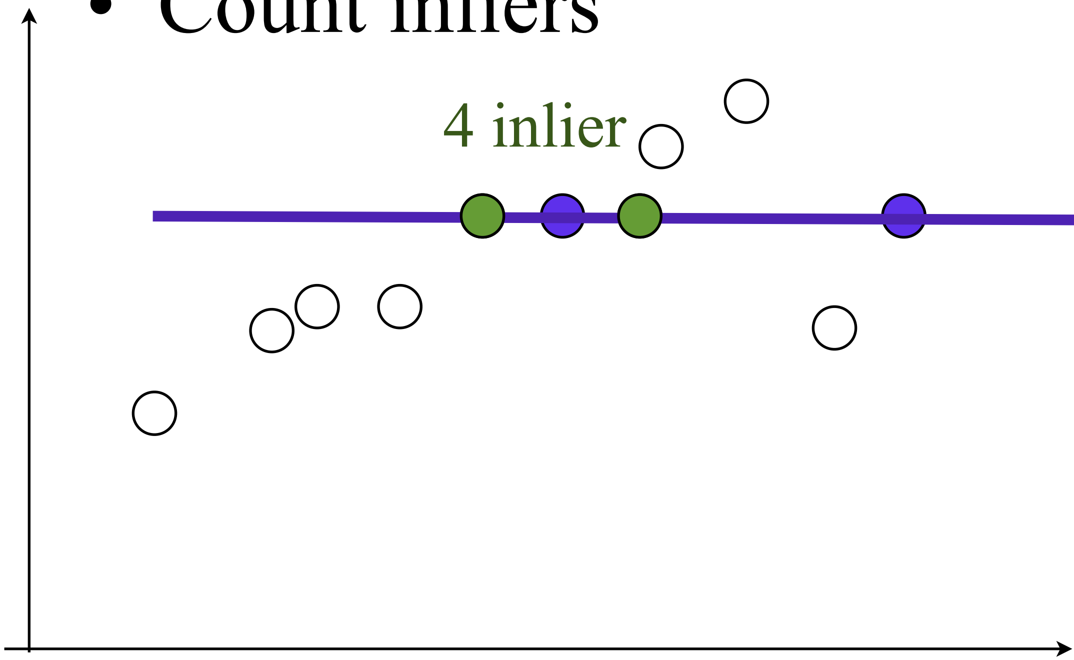
Simple example: fit a line

- Pick 2 points
- Fit line

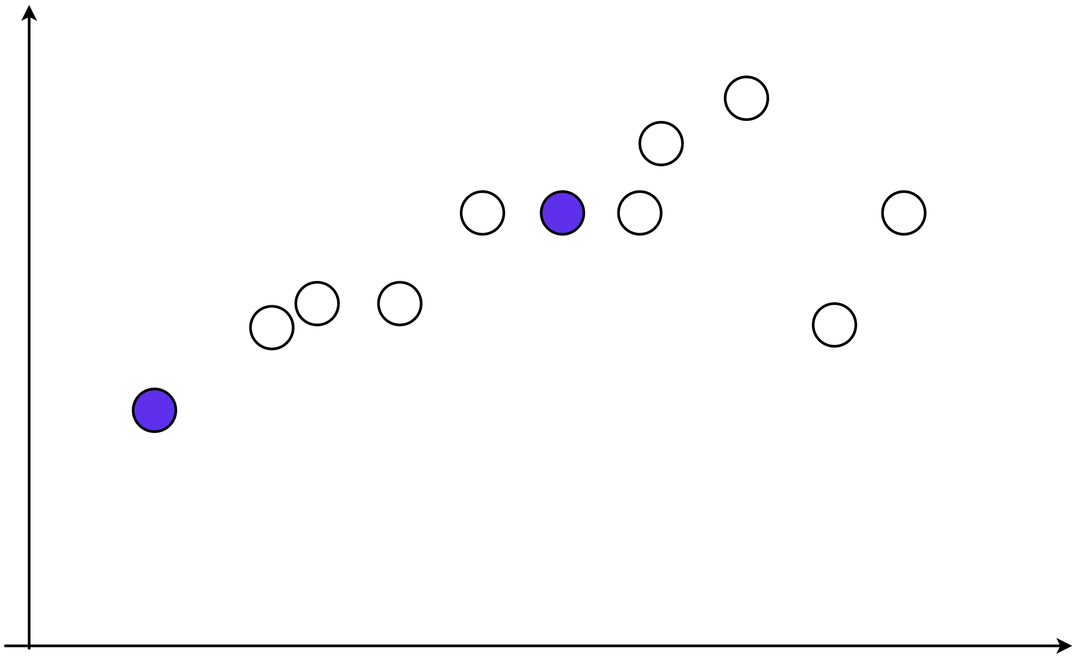


Simple example: fit a line

- Pick 2 points
- Fit line
- Count inliers

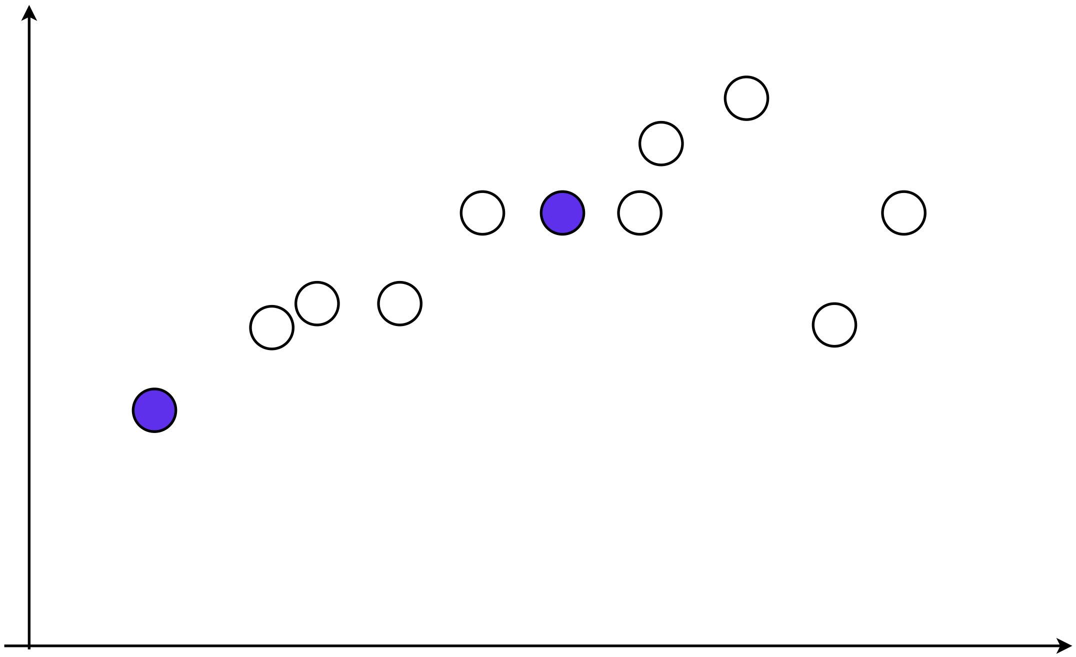


Simple example: fit a line



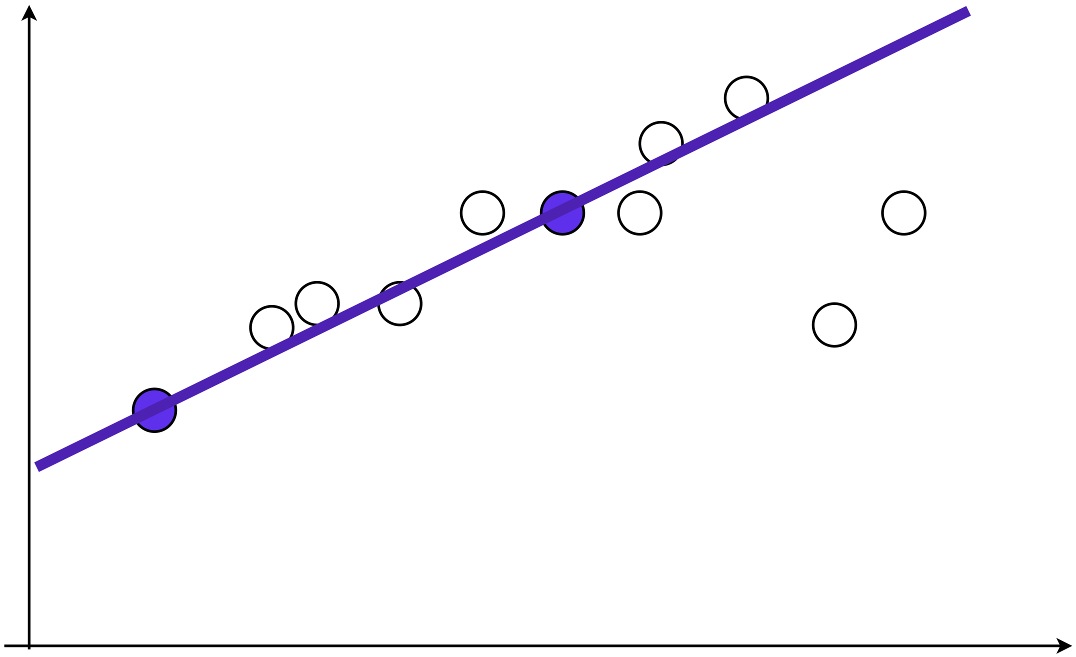
Simple example: fit a line

- Pick 2 points



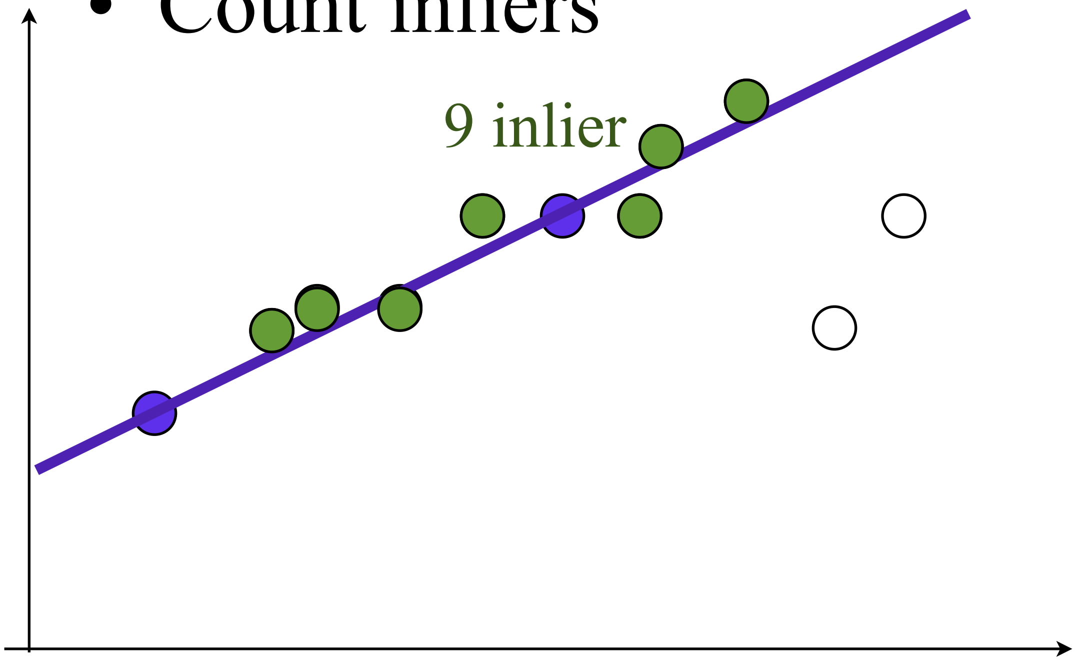
Simple example: fit a line

- Pick 2 points
- Fit line

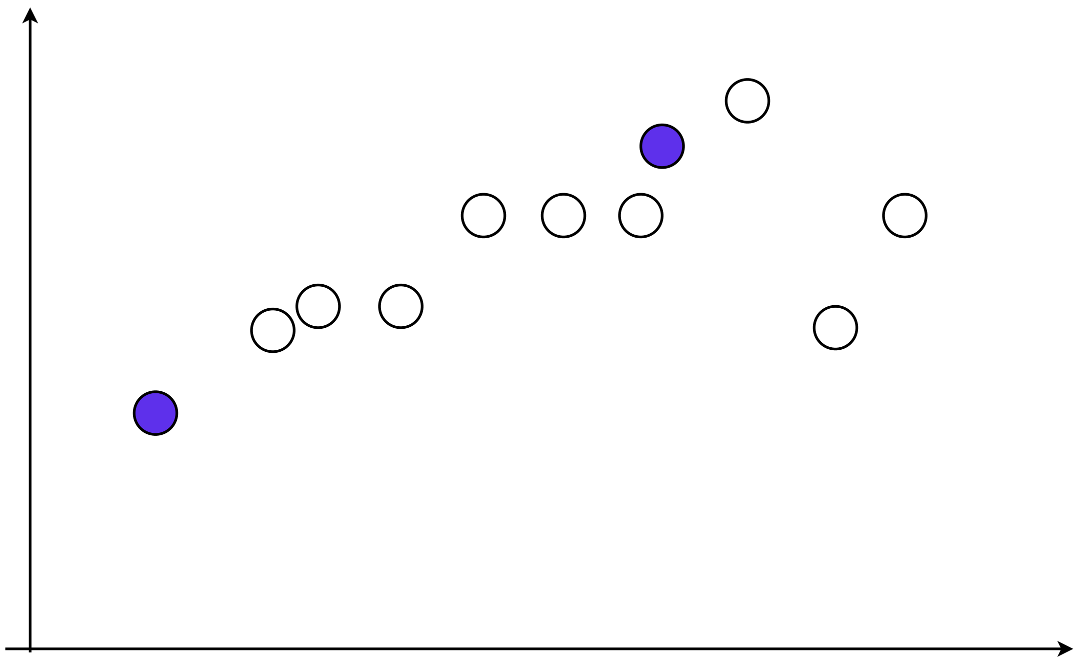


Simple example: fit a line

- Pick 2 points
- Fit line
- Count inliers

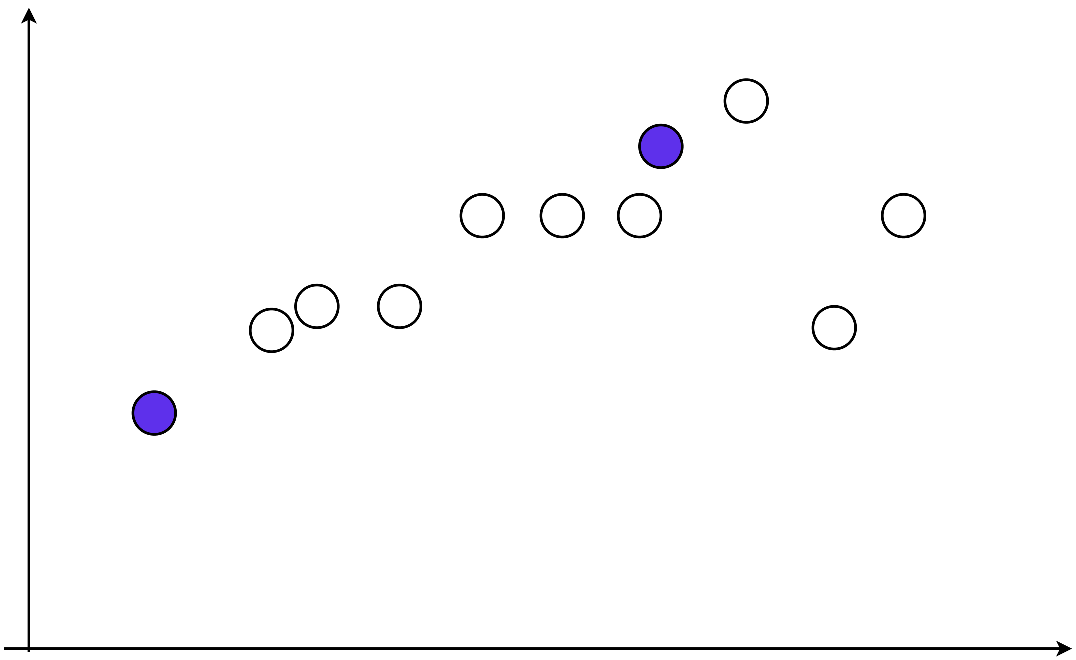


Simple example: fit a line



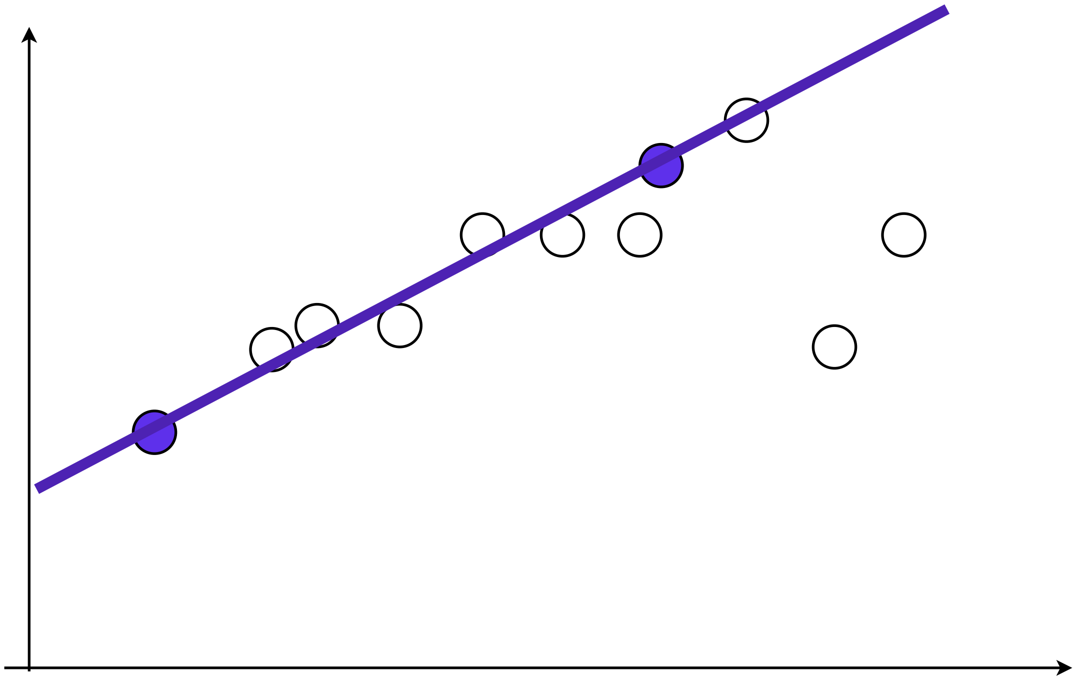
Simple example: fit a line

- Pick 2 points



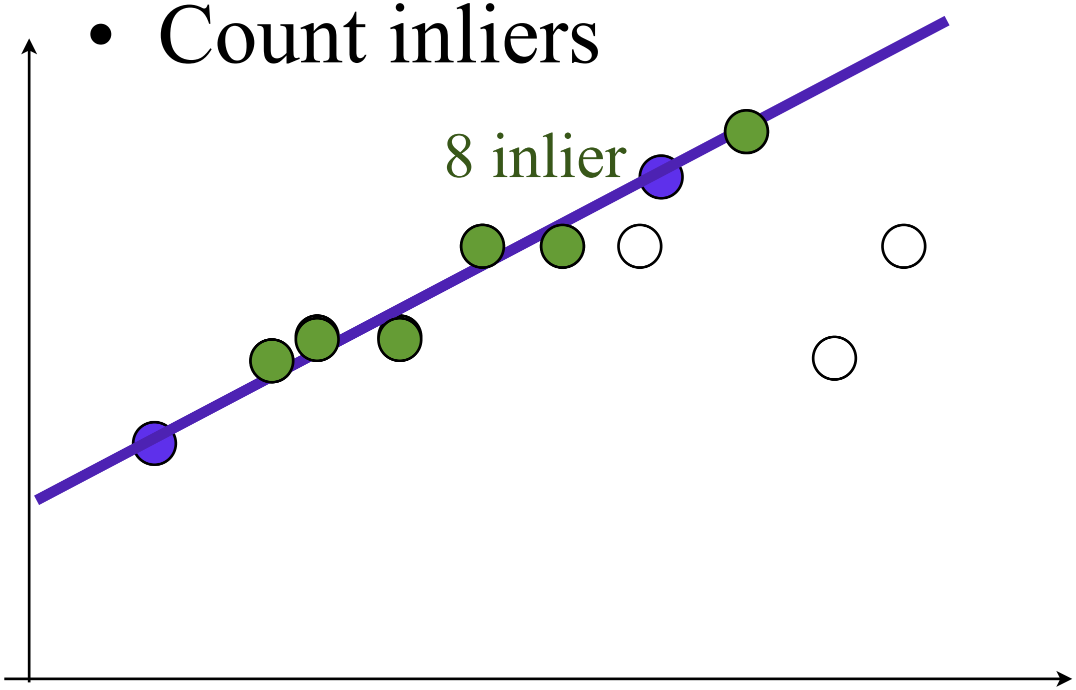
Simple example: fit a line

- Pick 2 points
- Fit line

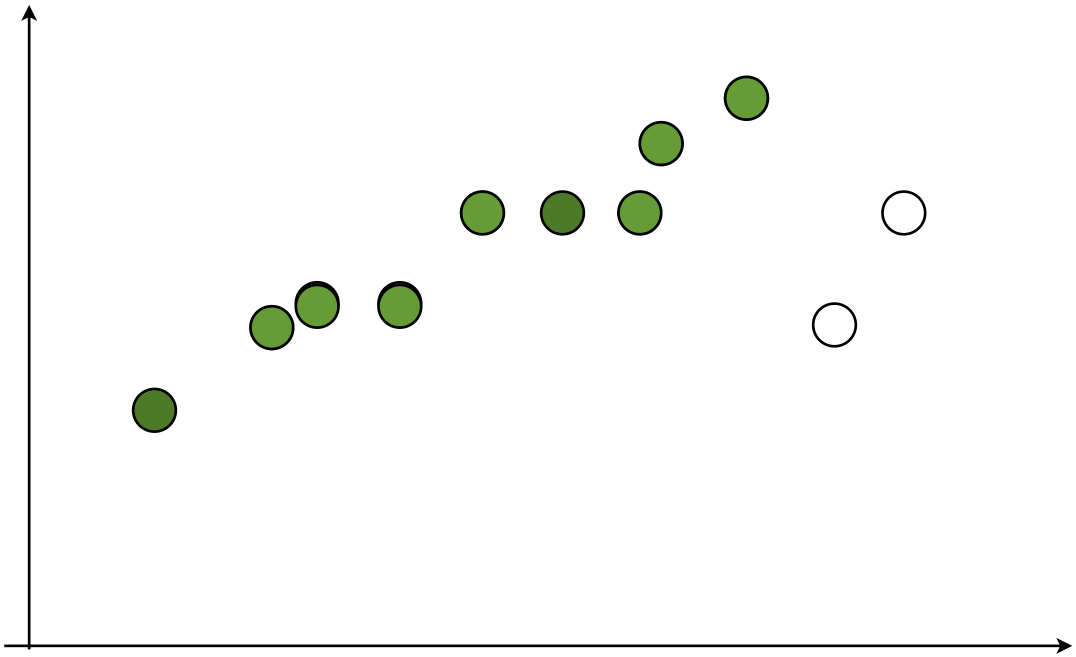


Simple example: fit a line

- Pick 2 points
- Fit line
- Count inliers

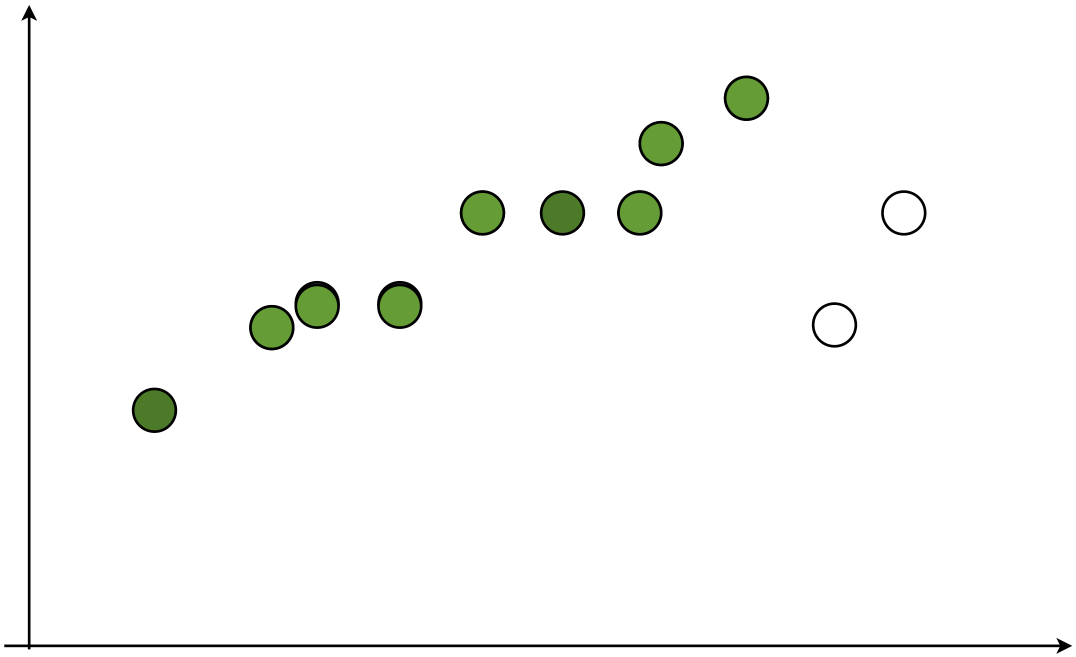


Simple example: fit a line



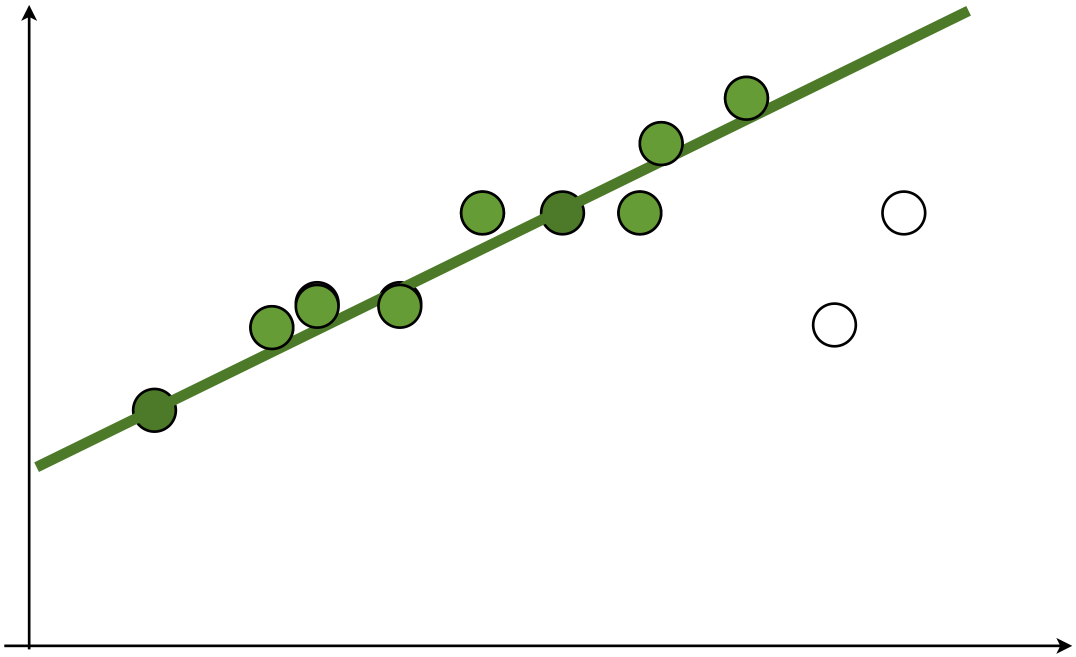
Simple example: fit a line

- Use biggest set of inliers

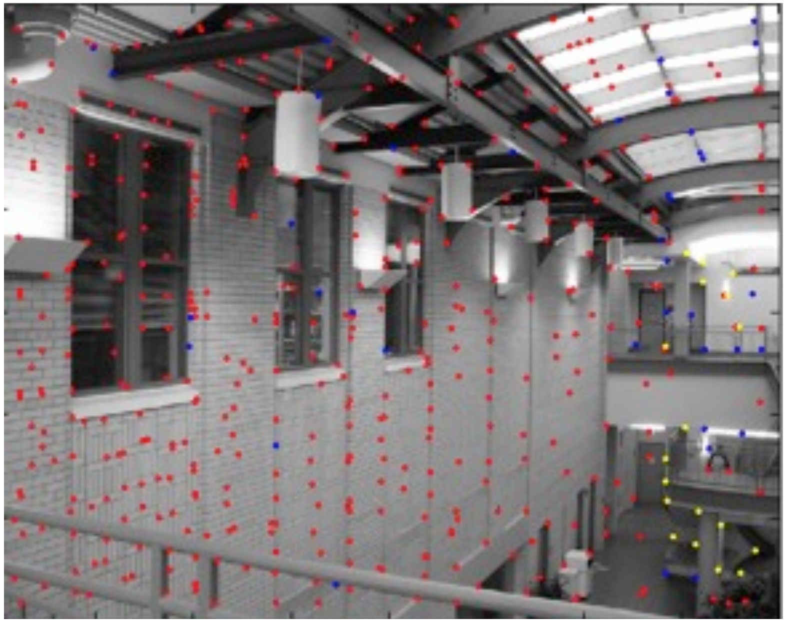


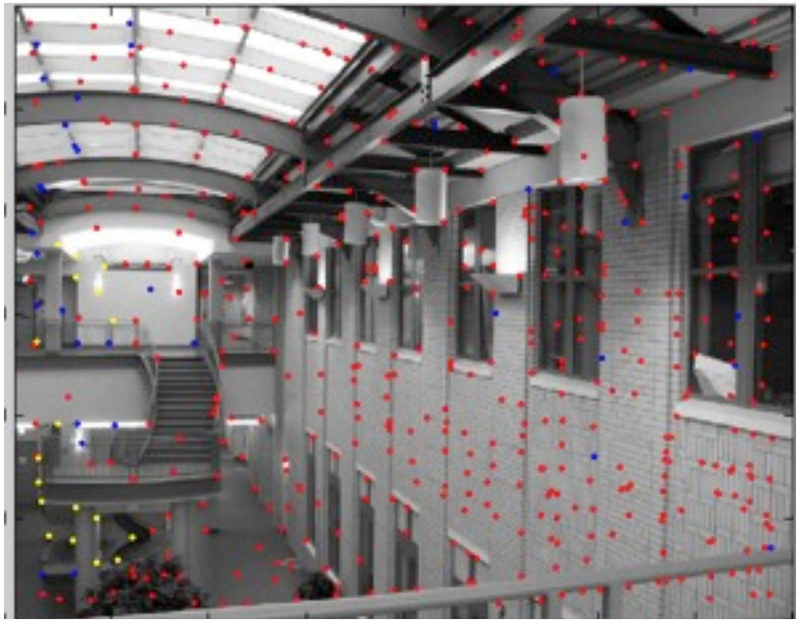
Simple example: fit a line

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



RANSAC





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

Robustness

Robustness

- Proportion of inliers in our pairs is G (for “good”)
- Our model needs P pairs
  - P=4 for homography

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?

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

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?

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:
    $(1\text{-}0.5^4)^{100}$=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:
    $(1\text{-}0.3^4)^{100}$=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

Robustness: conclusions

- Effect of number of parameters of model/ number of necessary pairs

Robustness: conclusions

- Effect of number of parameters of model/ number of necessary pairs
  - Bad exponential

Robustness: conclusions

- Effect of number of parameters of model/ number of necessary pairs
  - Bad exponential
- Effect of percentage of inliers

Robustness: conclusions

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

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

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

Example: Recognising Panoramas

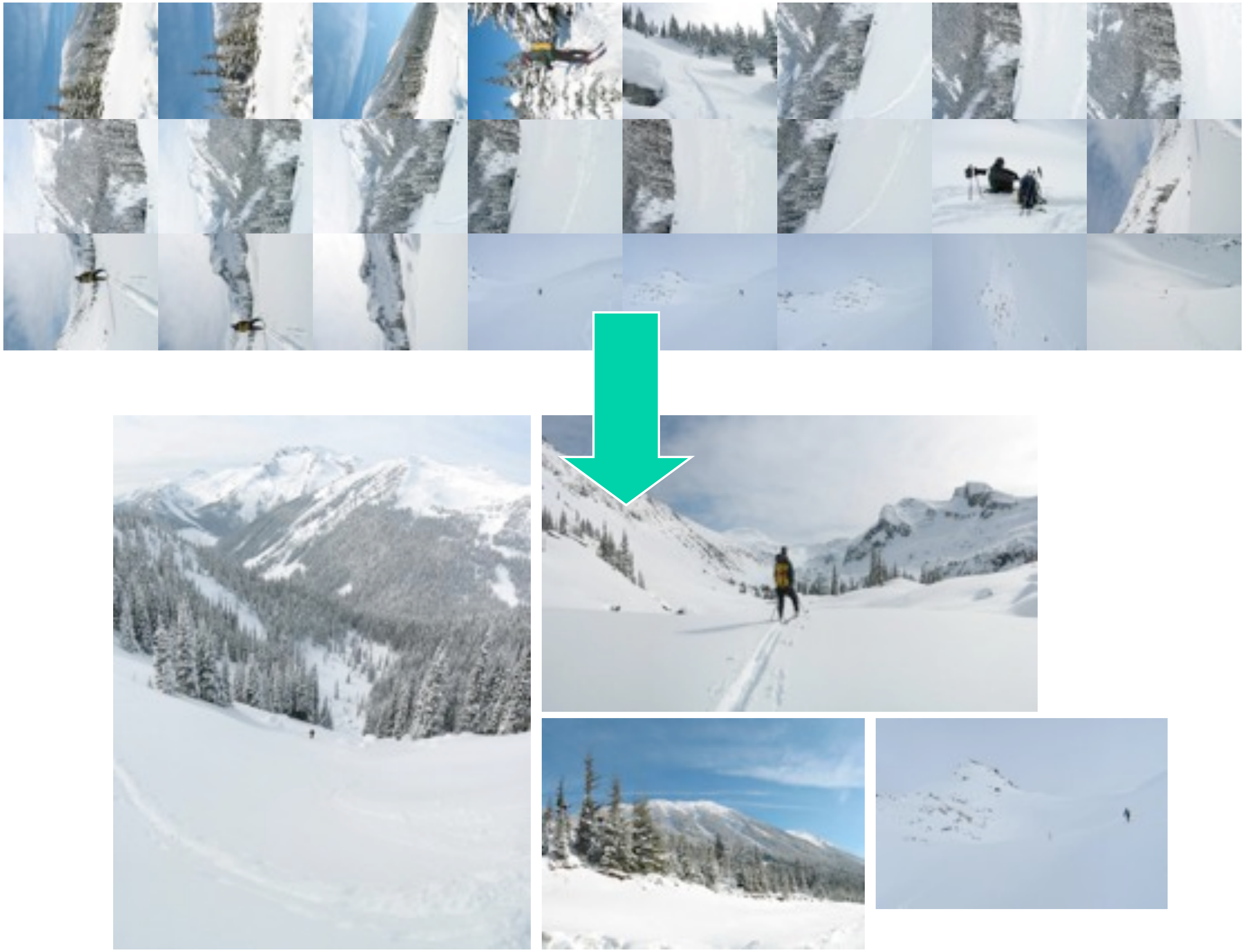
M. Brown and D. Lowe,

University of British Columbia

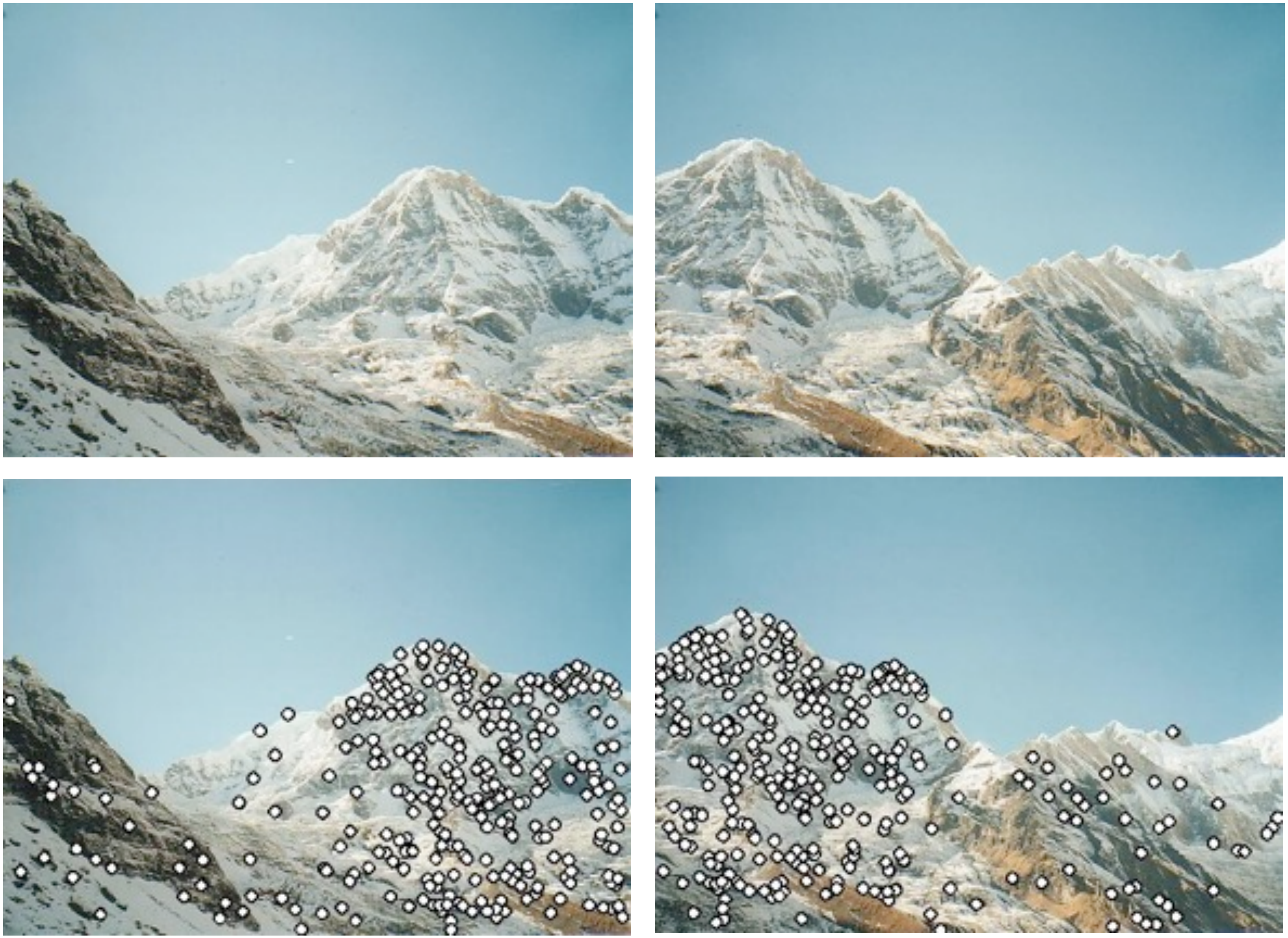
* M. Brown and D. Lowe. Automatic Panoramic Image Stitching using Invariant Features. International Journal of Computer Vision, 74(1), pages 59-73, 2007 (pdf 3.5Mb | bib)

* M. Brown and D. G. Lowe. Recognising Panoramas. In Proceedings of the 9th International Conference on Computer Vision (ICCV2003), pages 1218-1225, Nice, France, 2003 (pdf 820kb | ppt | bib)

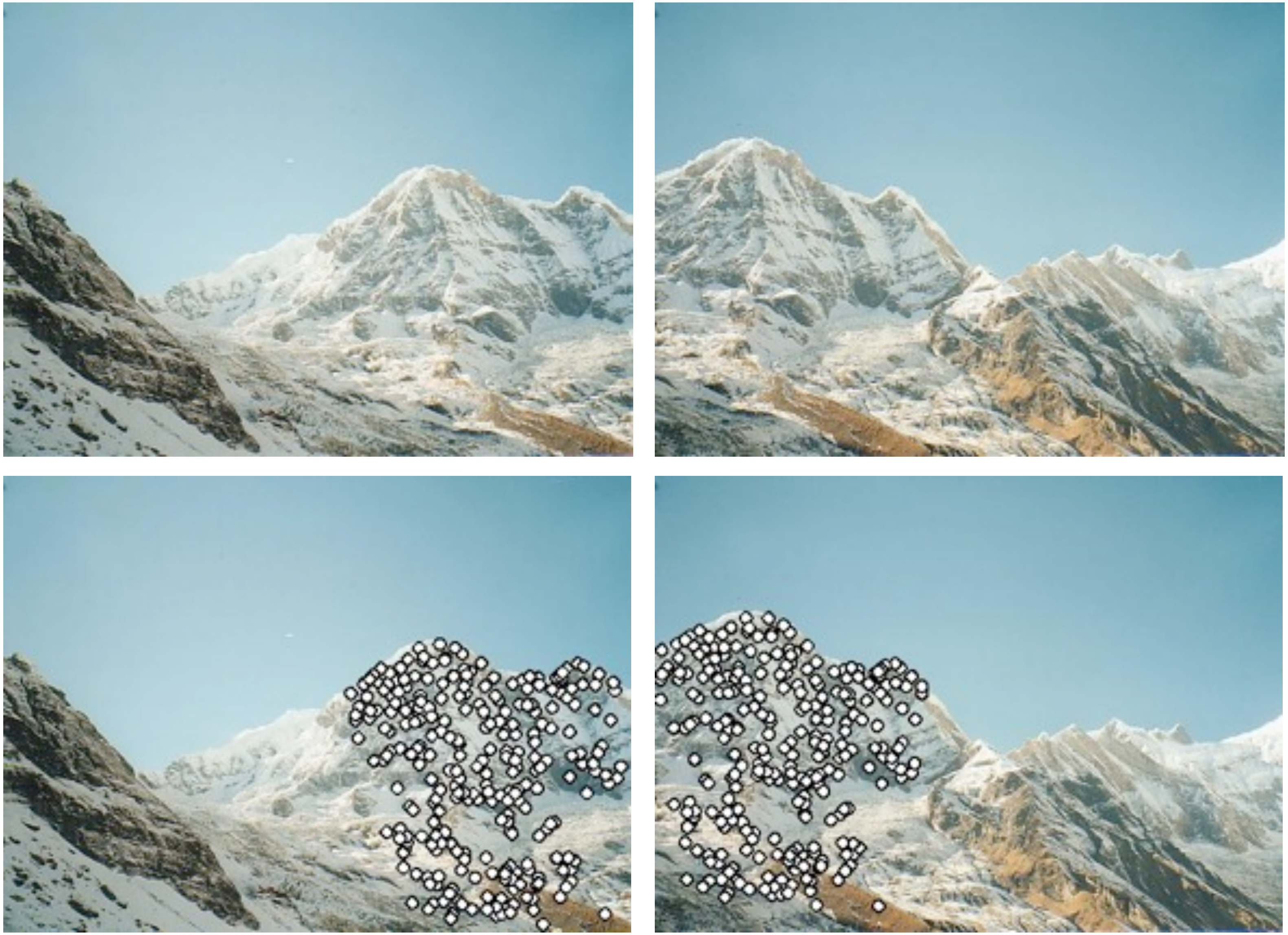
“Recognising Panoramas”?



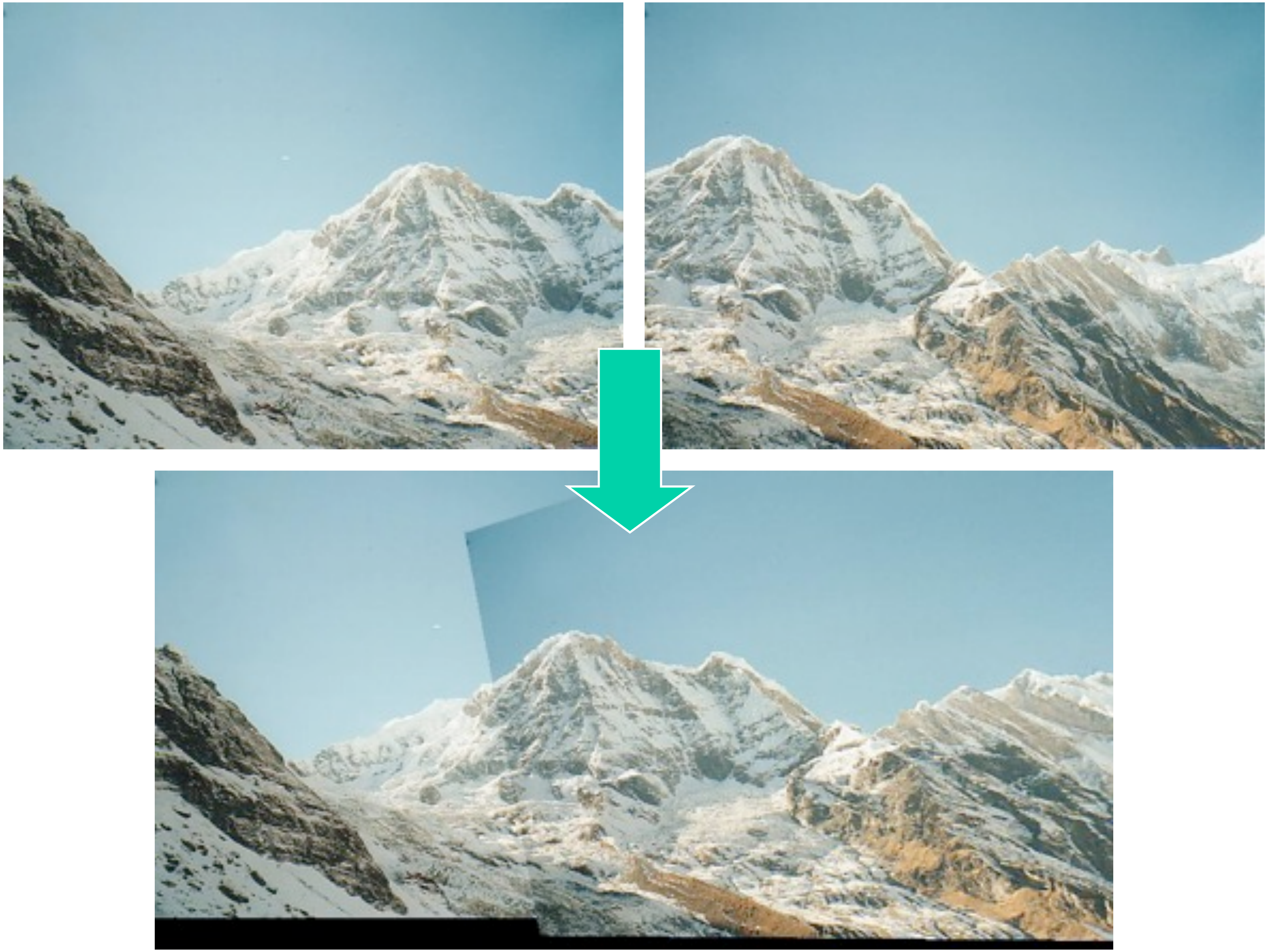
RANSAC for Homography



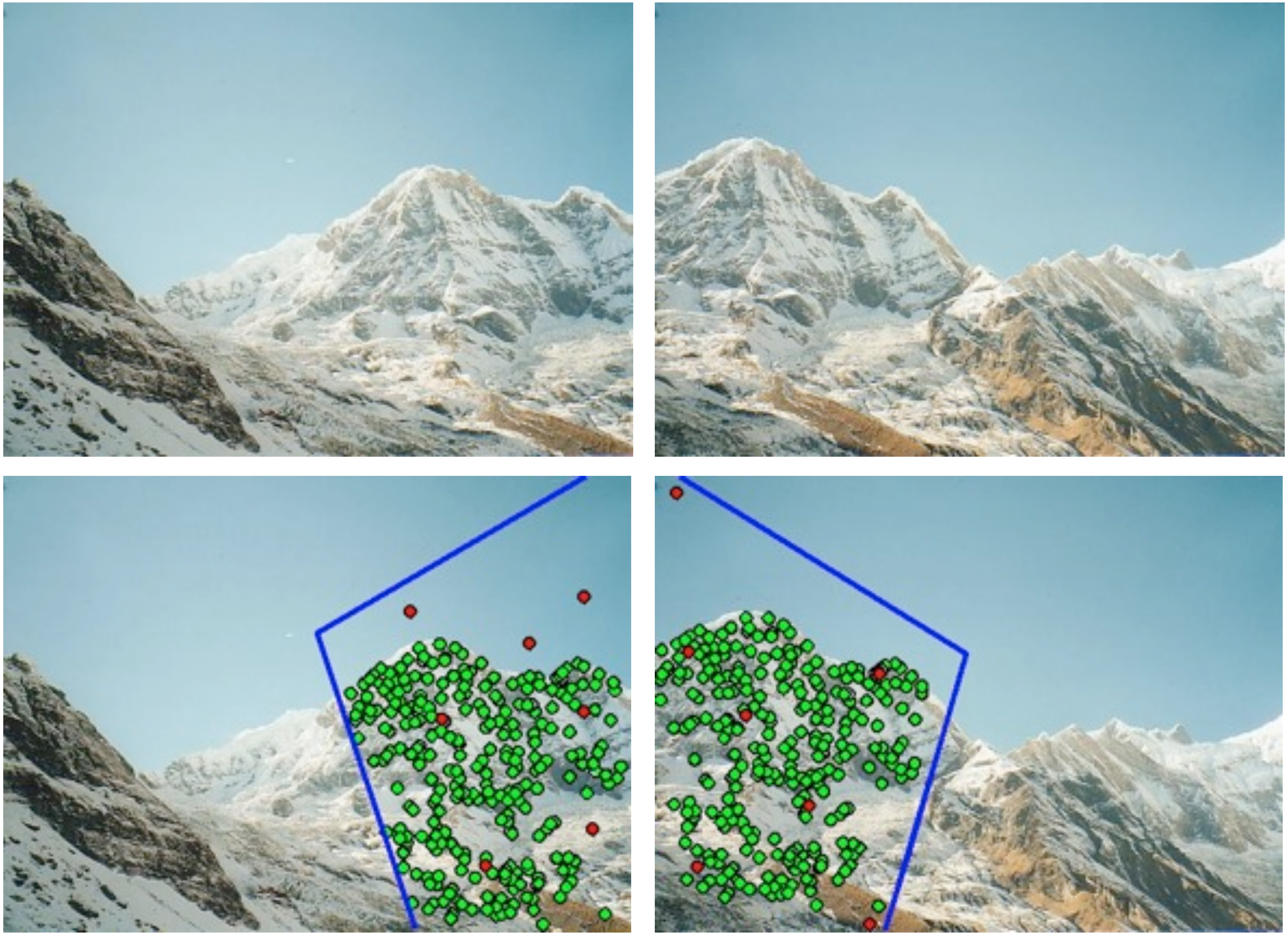
RANSAC for Homography



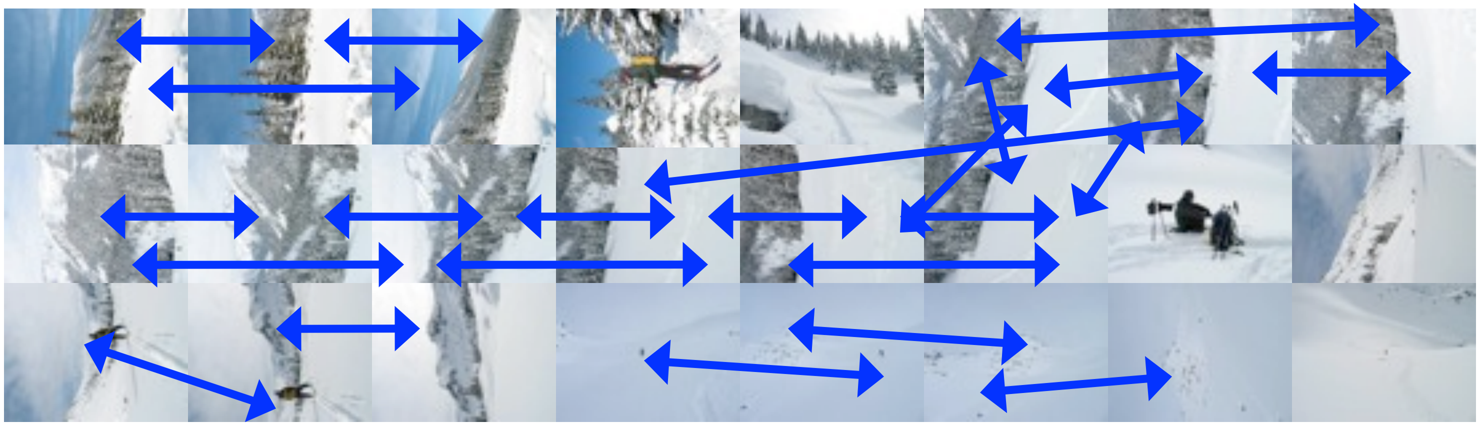
RANSAC for Homography



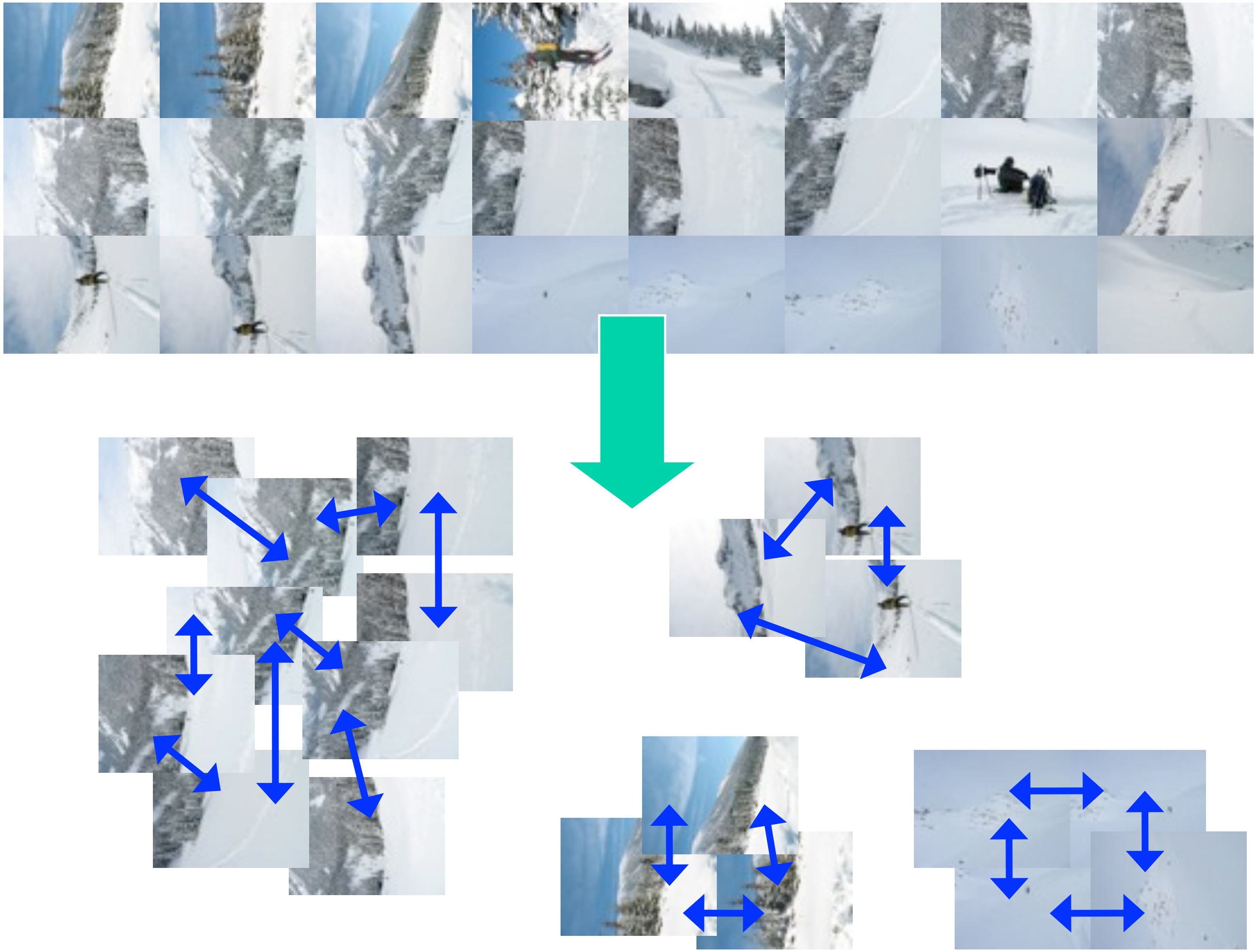
Probabilistic model for verification



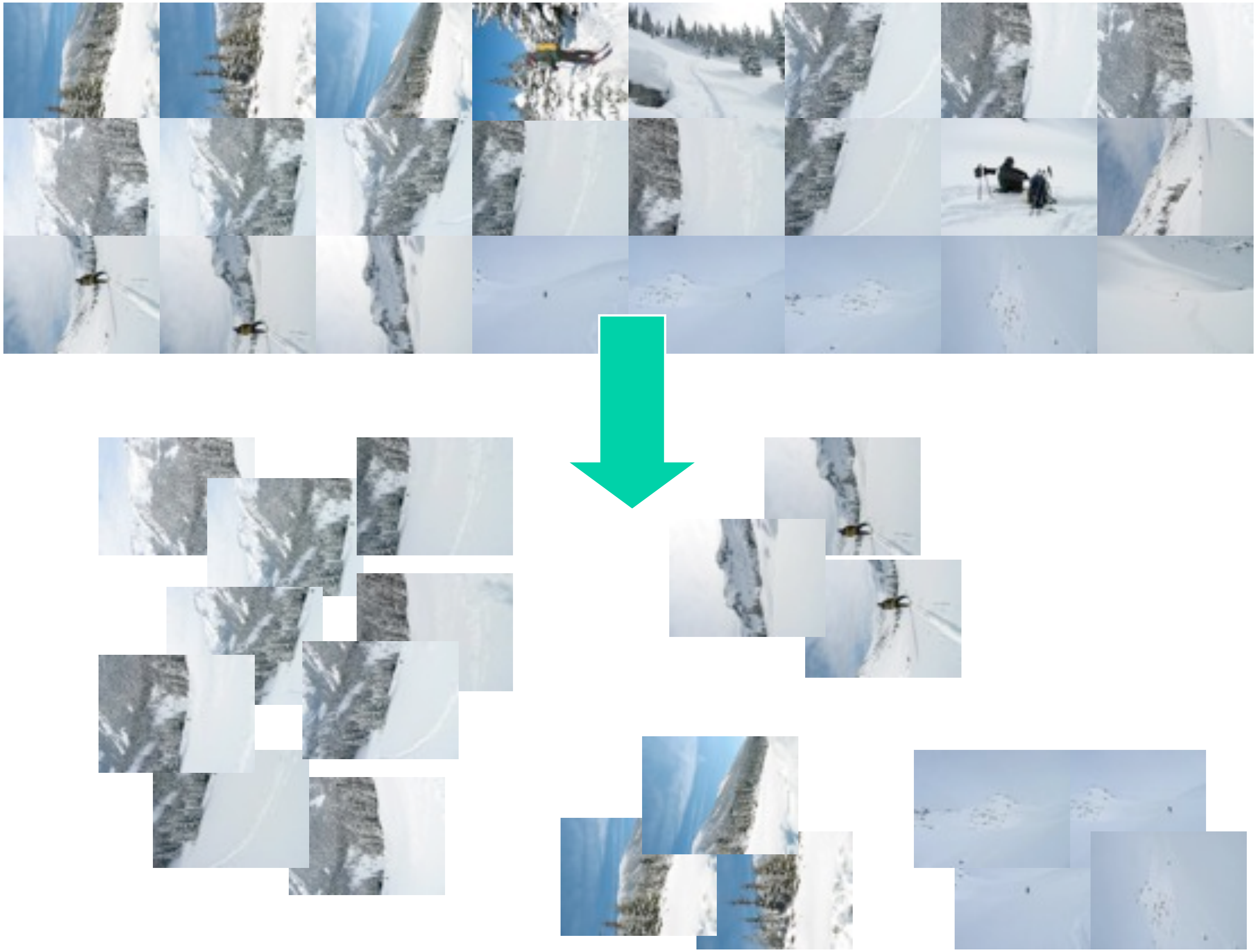
Finding the panoramas



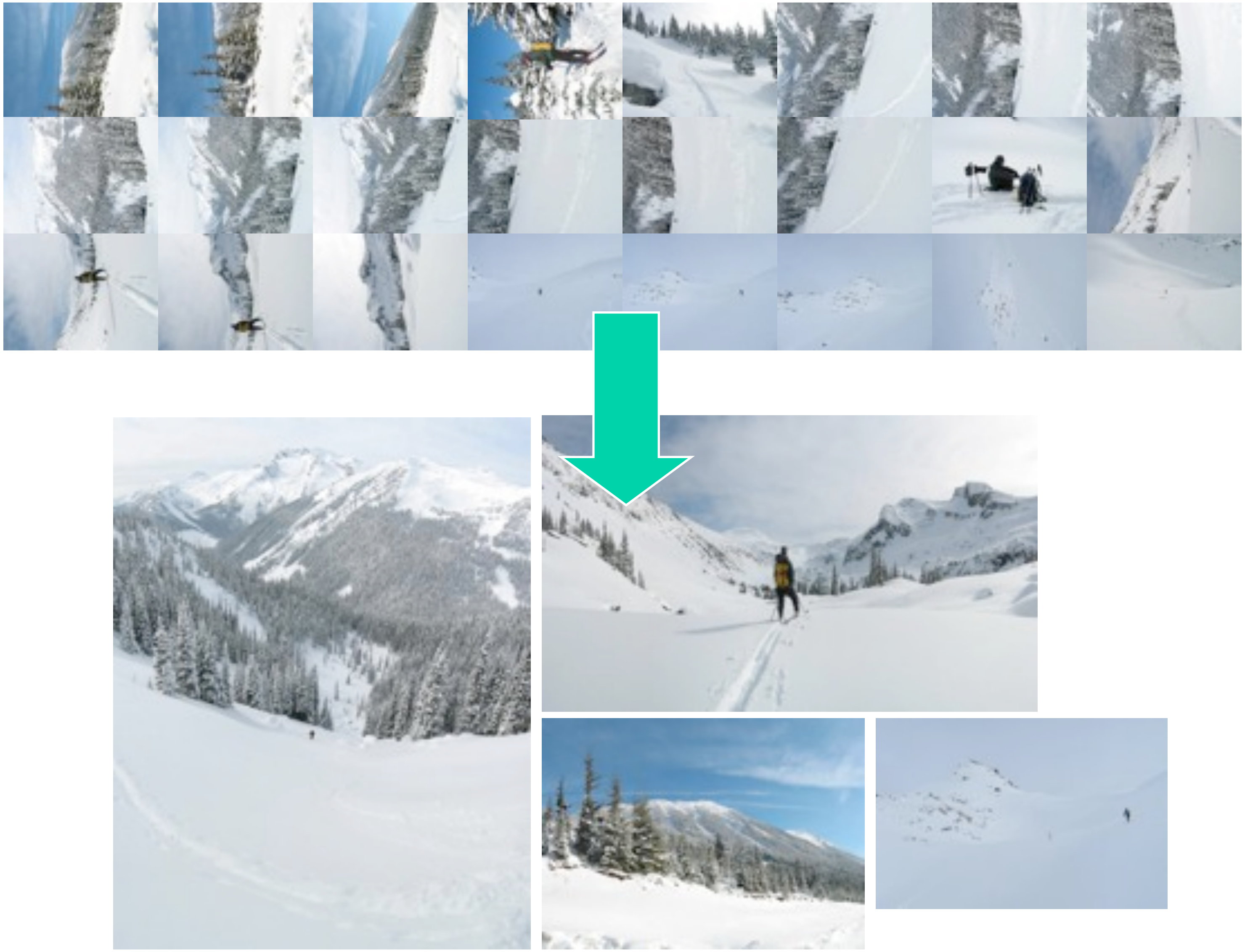
Finding the panoramas



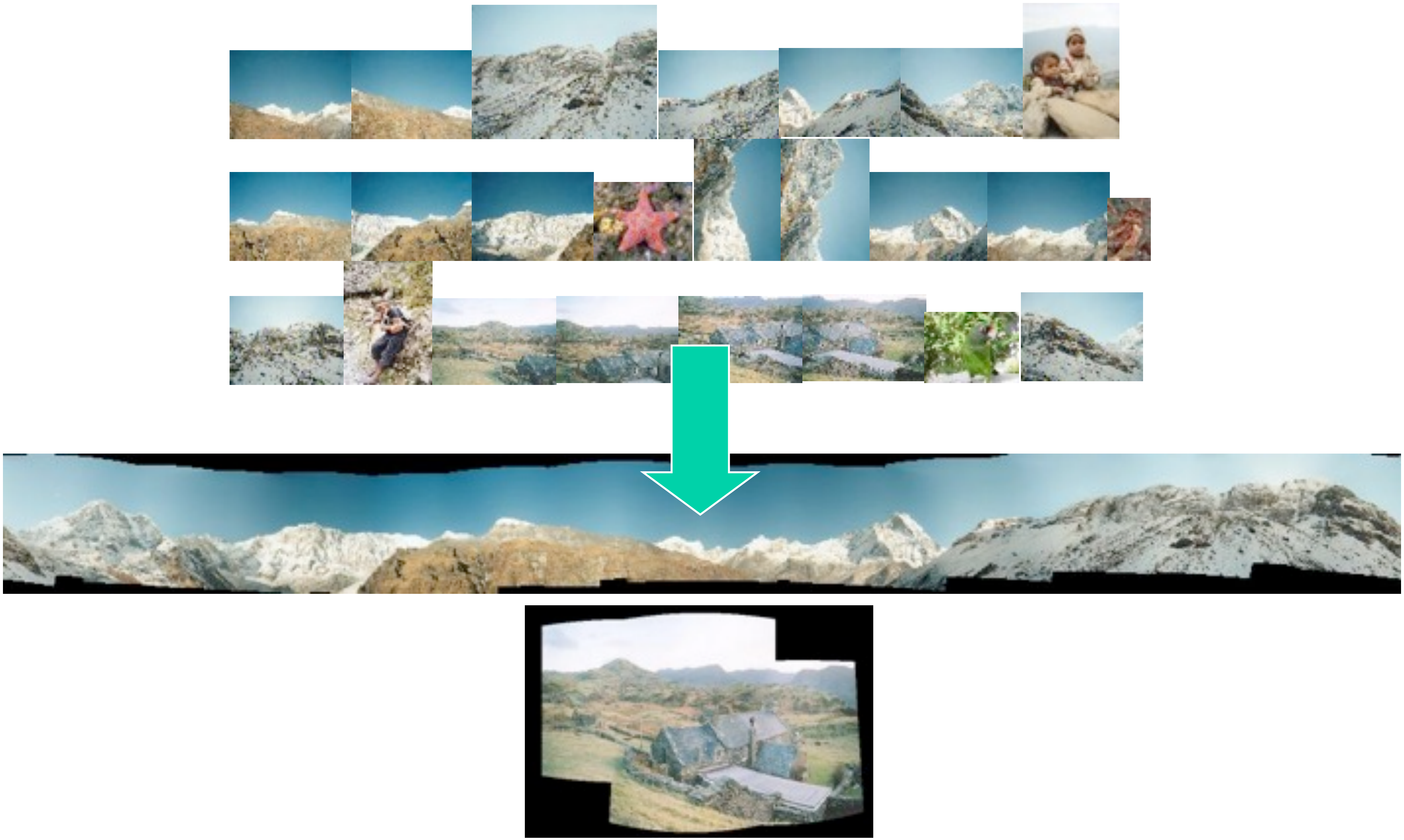
Finding the panoramas



Finding the panoramas



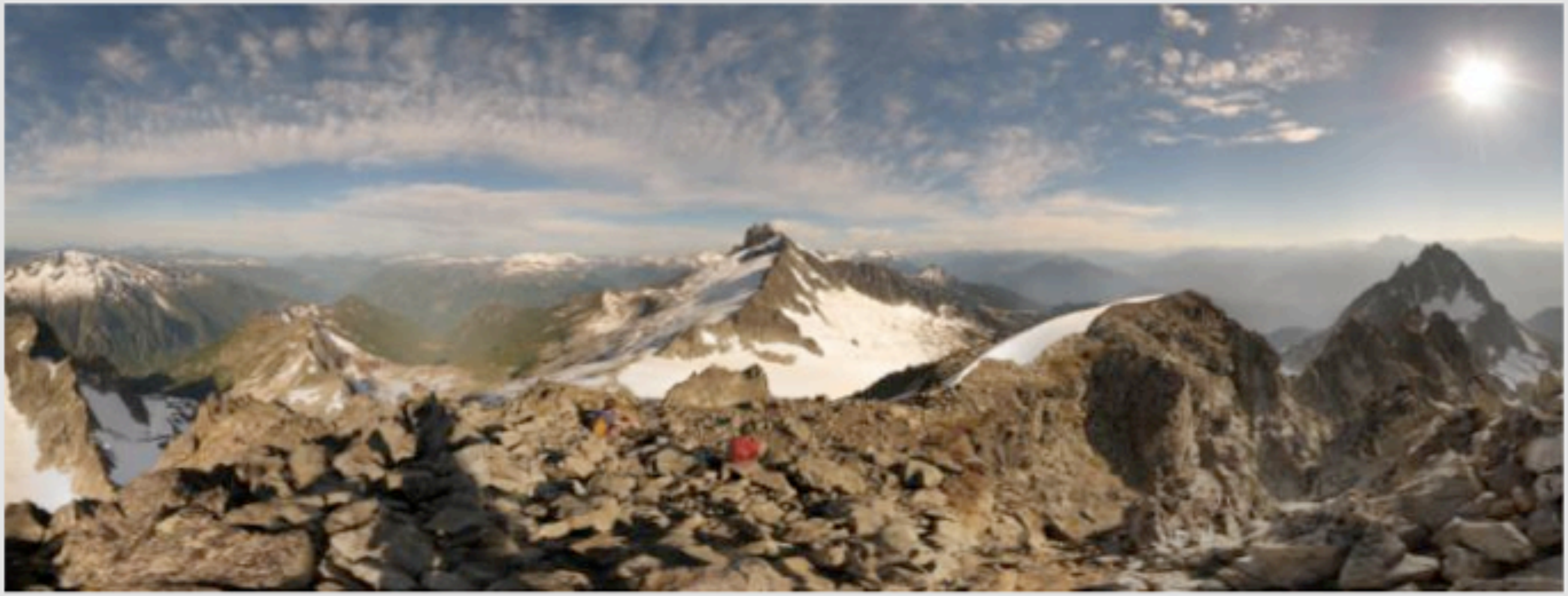
Results



AUTOSTITCH

AutoStitch | Gallery | **Download (Windows demo)** | Buy Autopano | Licensing | Press | FAQ | Publications

AutoStitch :: a new dimension in automatic image stitching



Serratus

Welcome to AutoStitch. If you have an iPhone, please check out our new iPhone version of AutoStitch below! If you're looking for the Windows demo version, you can download it using the link above, or read on to find out more about AutoStitch. Thanks for visiting!

Benefits of Laplacian image compositing

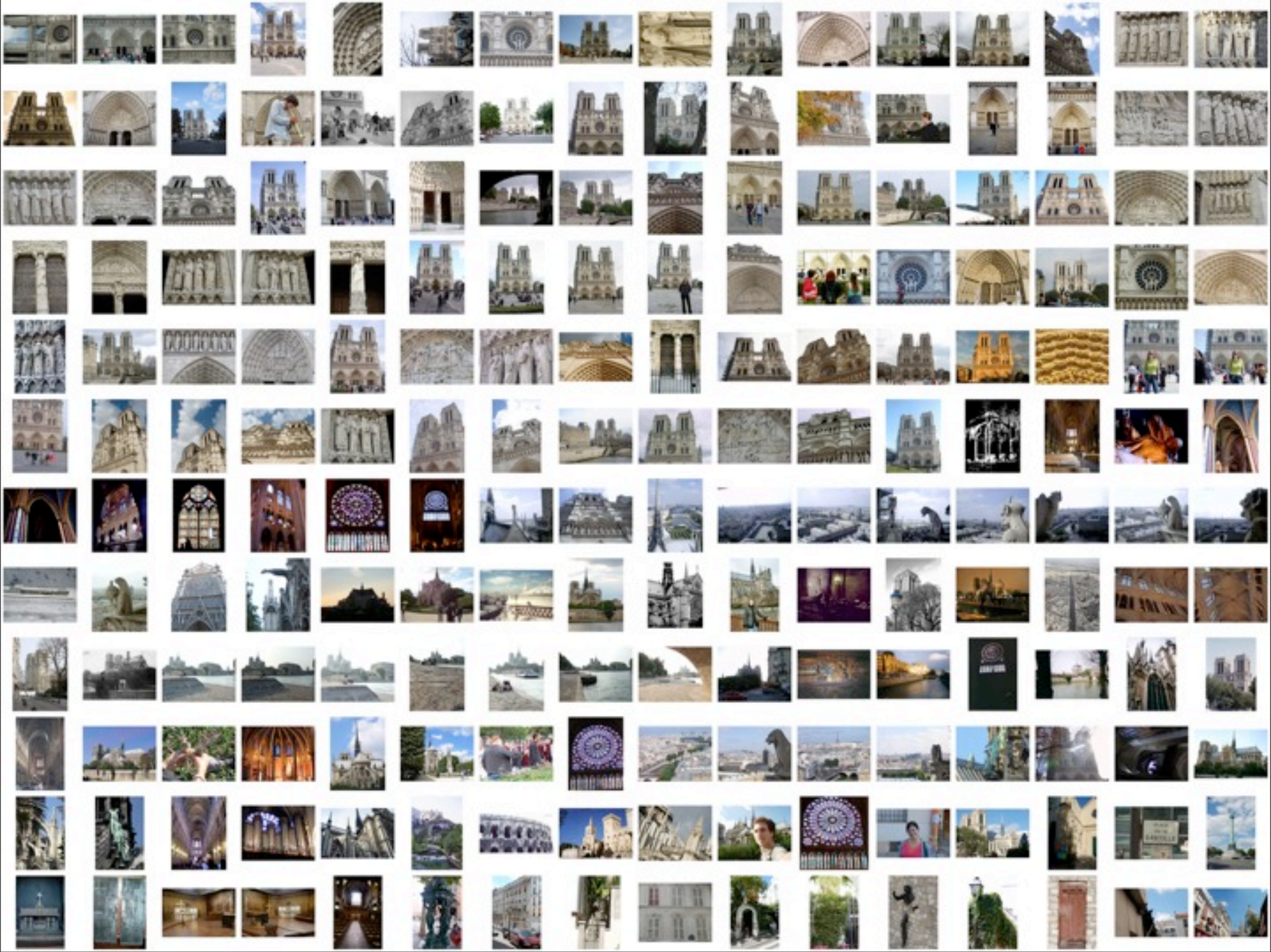
(a) Linear blending (b) Multi-band blending

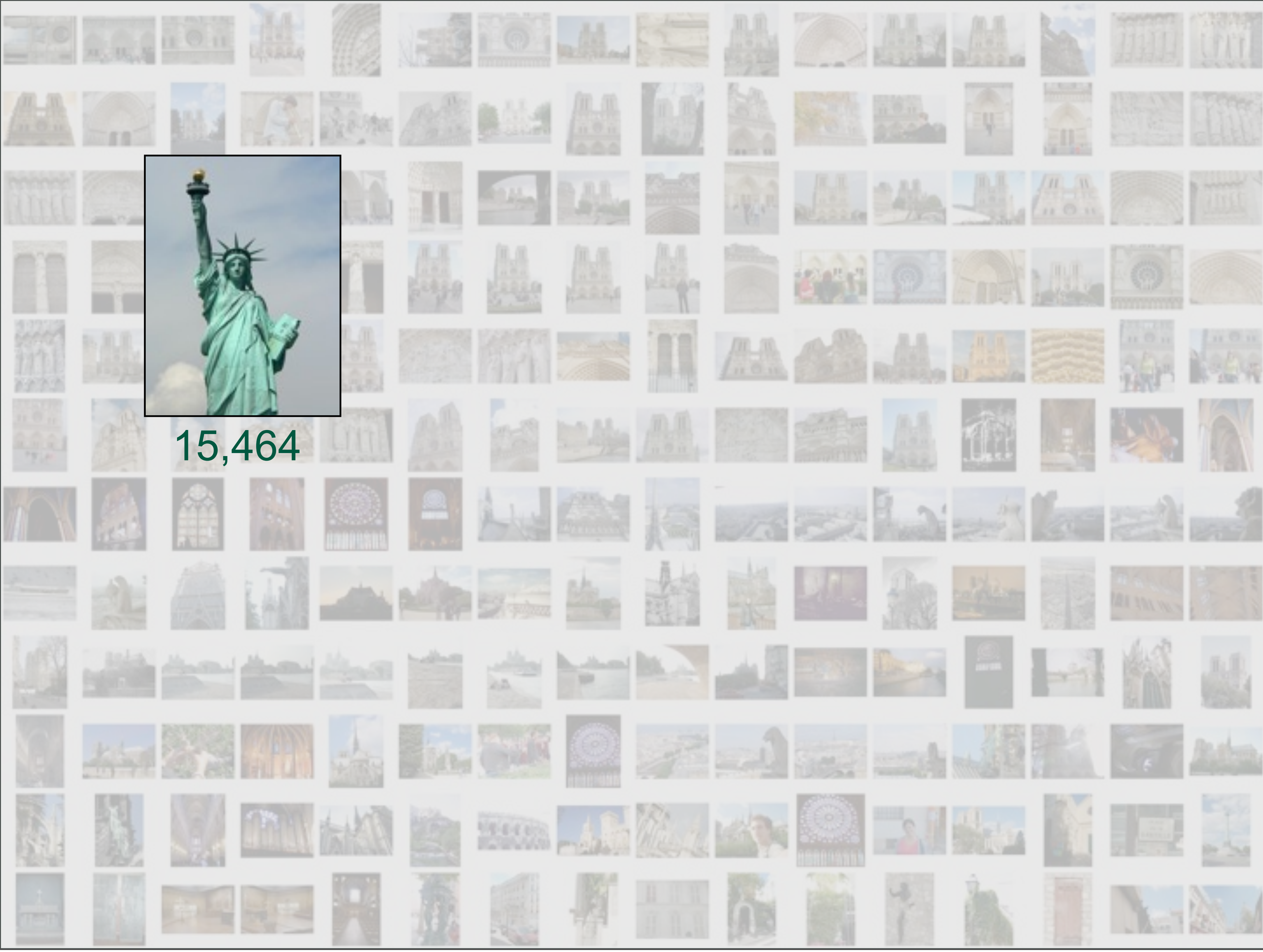
7. Comparison of linear and multi-band blending. The image on the right was blended using multi-band ble
bands and $\sigma = 5$ pixels. The image on the left was linearly blended. In this case matches on the m
have caused small misregistrations between the images, which cause blurring in the linearly blended resu
lti-band blended image is clear.

M. Brown and D. Lowe. Automatic Panoramic Image Stitching using Invariant Features. International Journal of Computer Vision, 74(1), pages 59-73, 2007

Photo Tourism:
Exploring Photo Collections in 3D

Noah Snavely
Steven M. Seitz
University of Washington
Richard Szeliski
Microsoft Research




15,464

15,464

37,383

15,464

37,383

76,389

Photo Tourism
Exploring photo collections in 3D

Noah Snavely   Steven M. Seitz   Richard Szeliski
University of Washington   *Microsoft Research*

SIGGRAPH 2006

Photo Tourism overview

Photo Tourism overview

Input photographs

Photo Tourism overview

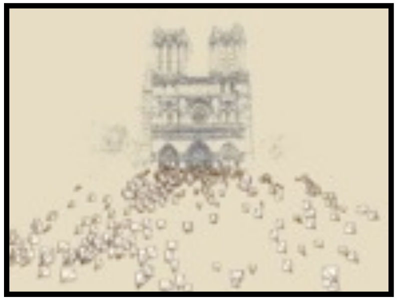
Input photographs

Scene reconstruction

Photo Tourism overview

Input photographs

Scene reconstruction



Relative camera positions and orientations

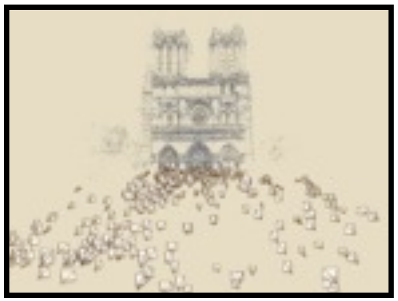
Point cloud

Sparse correspondence

Photo Tourism overview

Input photographs

Scene reconstruction



Relative camera positions and orientations

Point cloud

Sparse correspondence

Photo Explorer

Photo Tourism overview

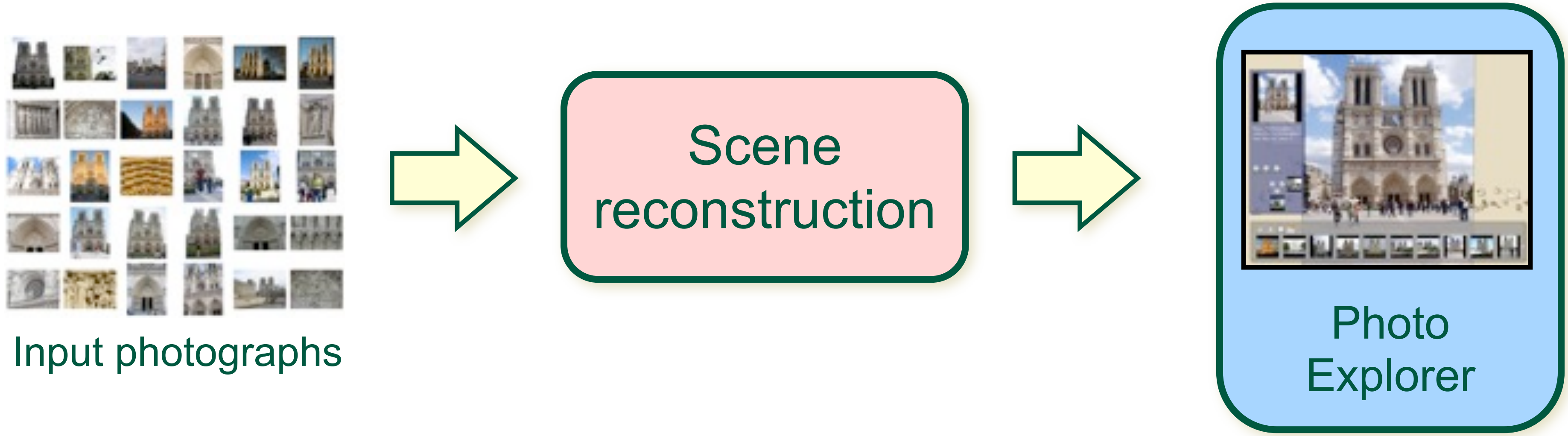


Photo Tourism overview

Input photographs

Scene reconstruction

Photo Explorer

Scene reconstruction

Scene reconstruction

- Automatically estimate
  - position, orientation, and focal length of cameras
  - 3D positions of feature points

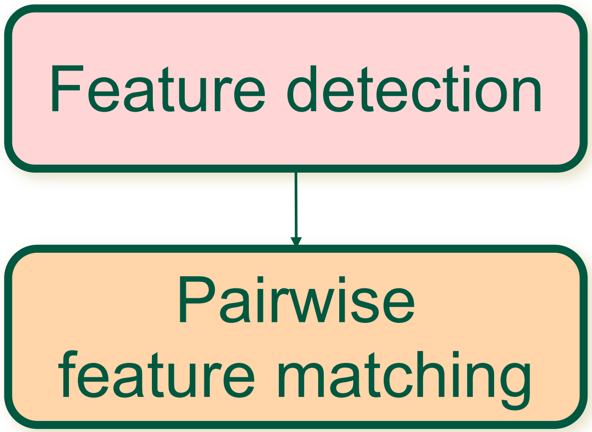
Scene reconstruction

- Automatically estimate
  - position, orientation, and focal length of cameras
  - 3D positions of feature points

Feature detection

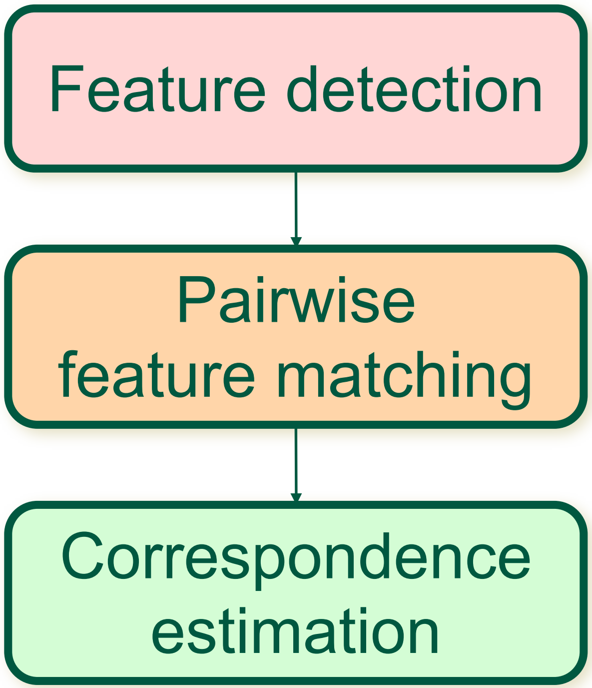
Scene reconstruction

- Automatically estimate
  - position, orientation, and focal length of cameras
  - 3D positions of feature points



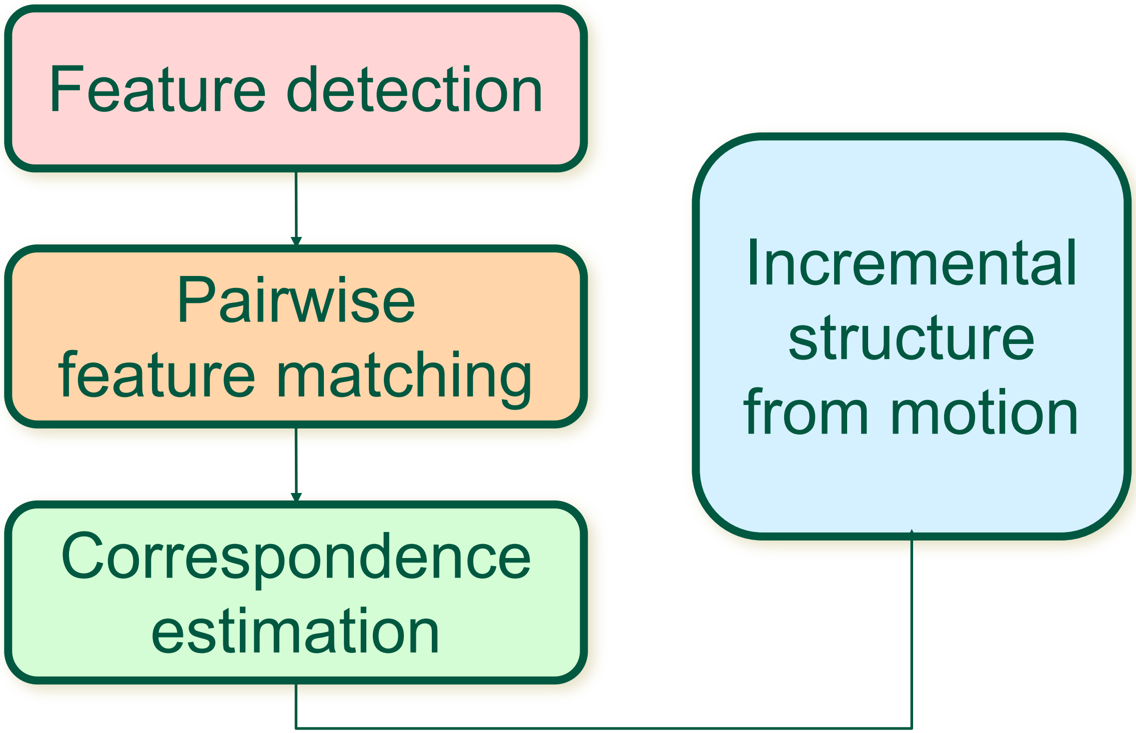
Scene reconstruction

- Automatically estimate
  - position, orientation, and focal length of cameras
  - 3D positions of feature points



Scene reconstruction

- Automatically estimate
  - position, orientation, and focal length of cameras
  - 3D positions of feature points



Feature detection

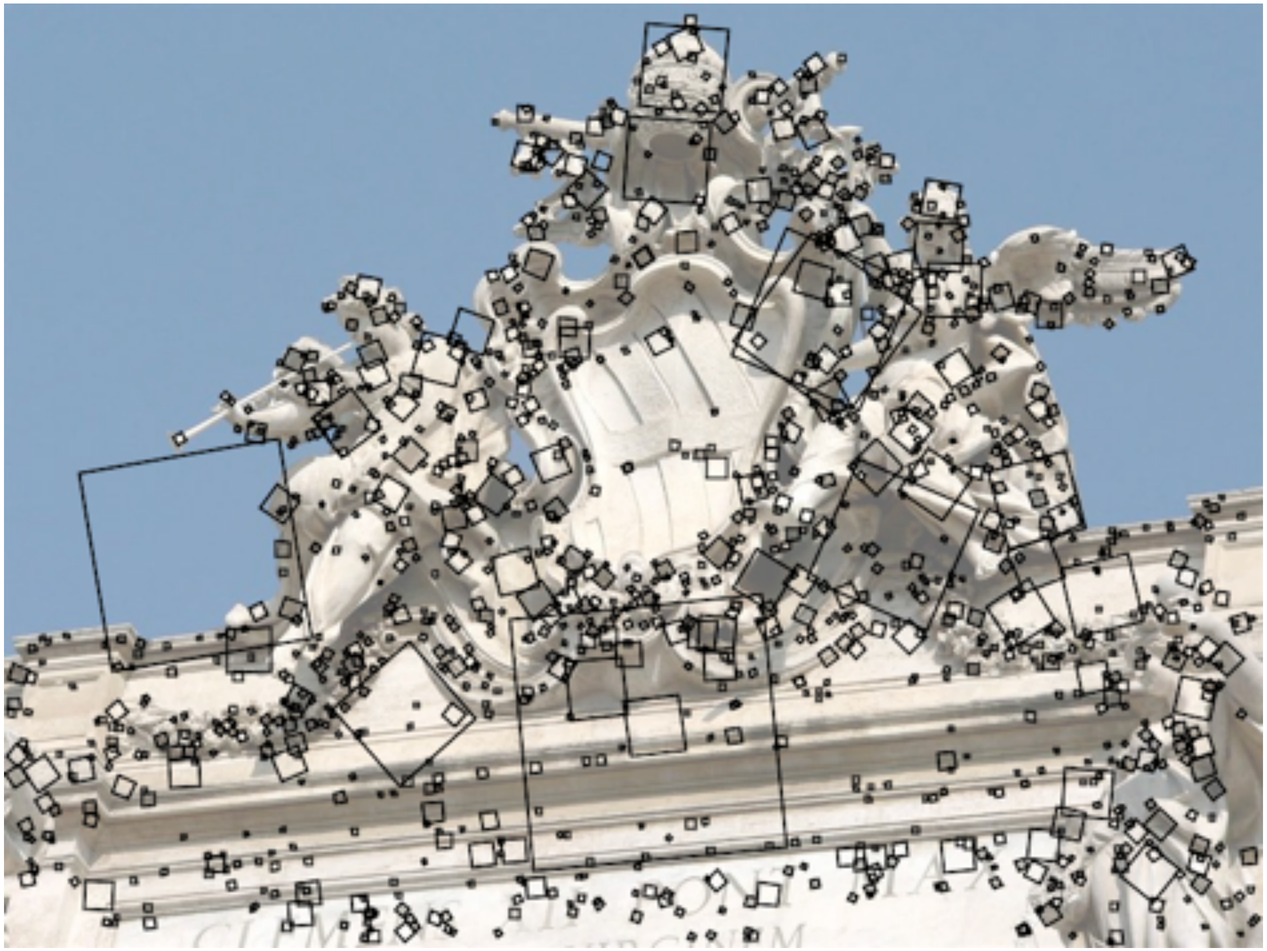
Detect features using SIFT [Lowe, IJCV 2004]

Feature detection

Detect features using SIFT [Lowe, IJCV 2004]

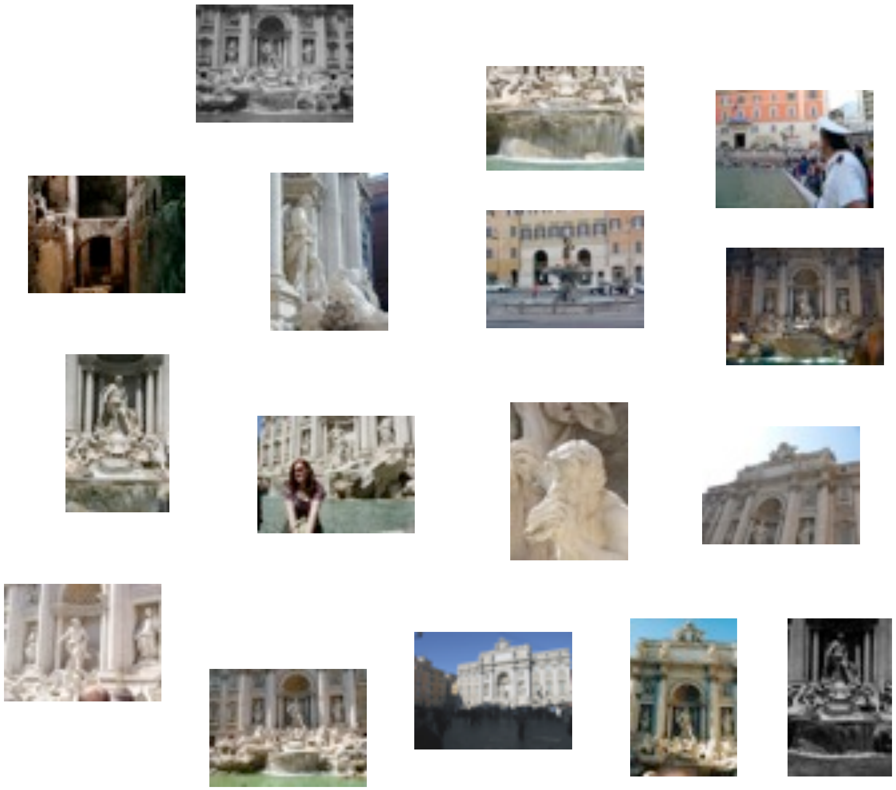
Feature detection

Detect features using SIFT [Lowe, IJCV 2004]



Feature detection

Detect features using SIFT [Lowe, IJCV 2004]



Feature detection

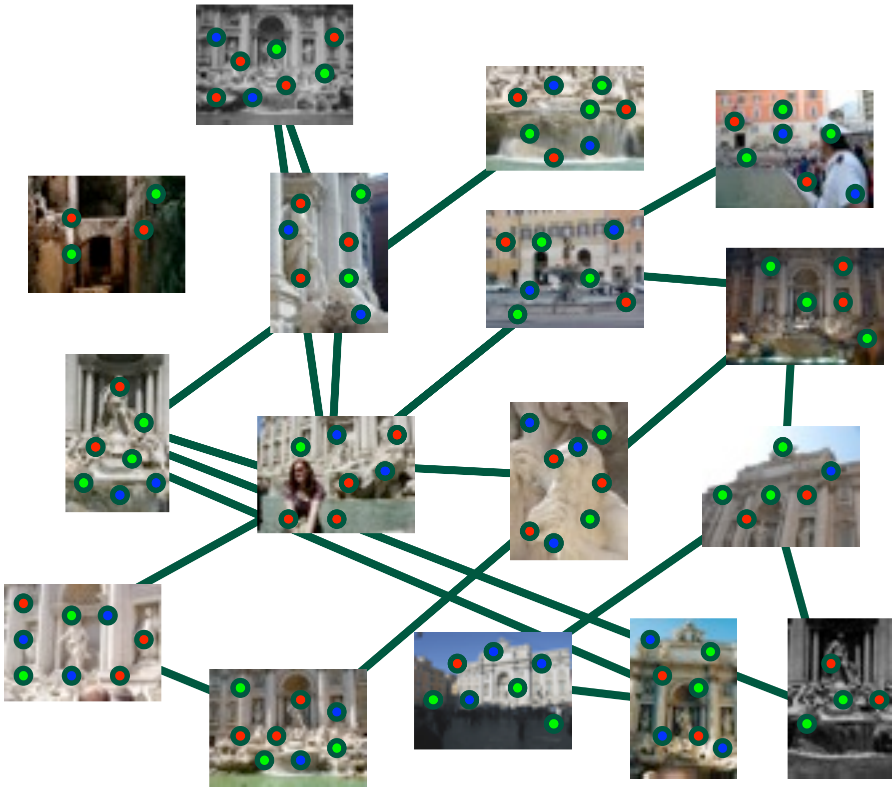
Detect features using SIFT [Lowe, IJCV 2004]

Feature matching

Match features between each pair of images

Feature matching

Match features between each pair of images

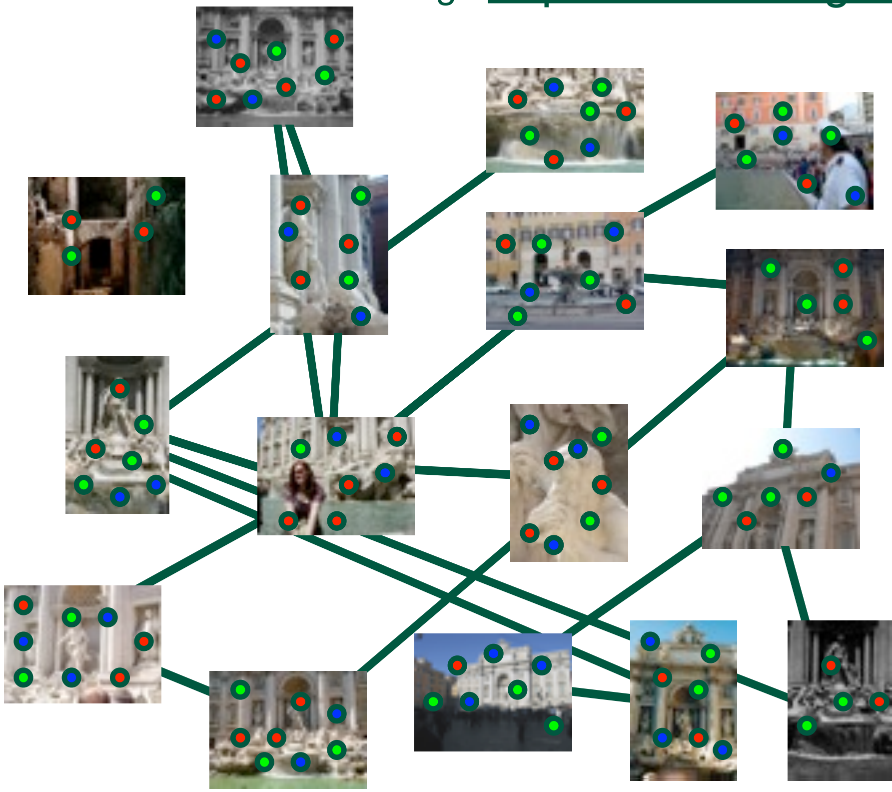


Feature matching

Refine matching using RANSAC [Fischler & Bolles 1987] to estimate fundamental matrices between pairs

(See 6.801/6.866 for fundamental matrix, or Hartley and Zisserman, Multi-View Geometry.

See also the fundamental matrix song: http://danielwedge.com/fmatrix/)

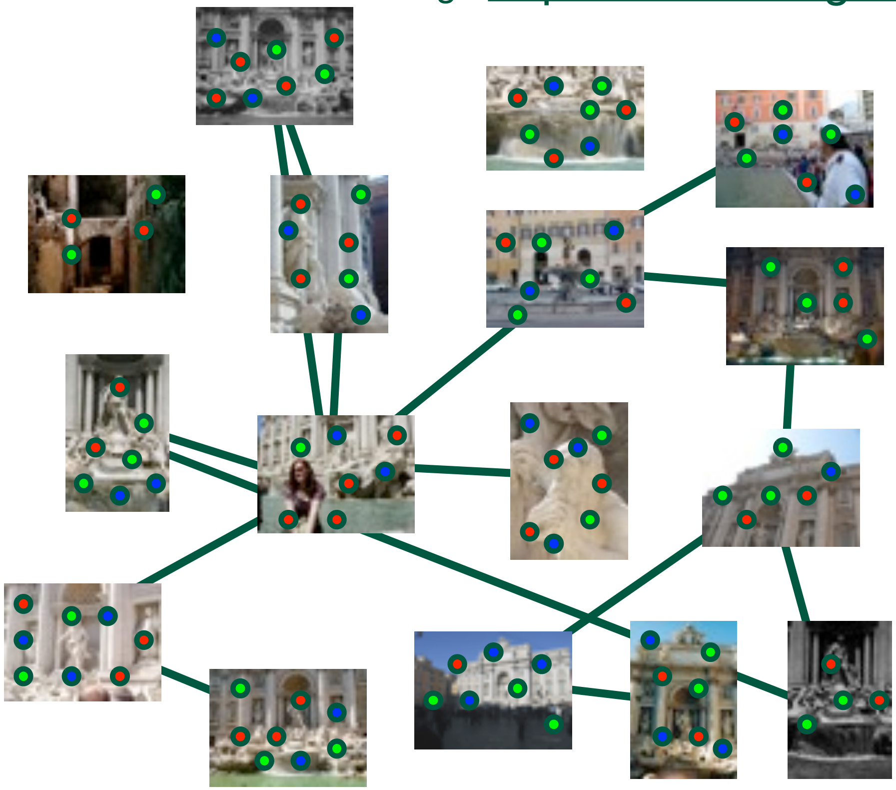


Feature matching

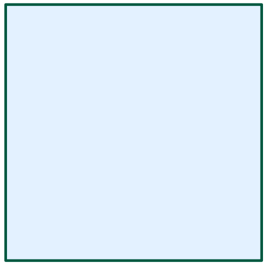
Refine matching using RANSAC [Fischler & Bolles 1987] to estimate fundamental matrices between pairs

(See 6.801/6.866 for fundamental matrix, or Hartley and Zisserman, Multi-View Geometry.

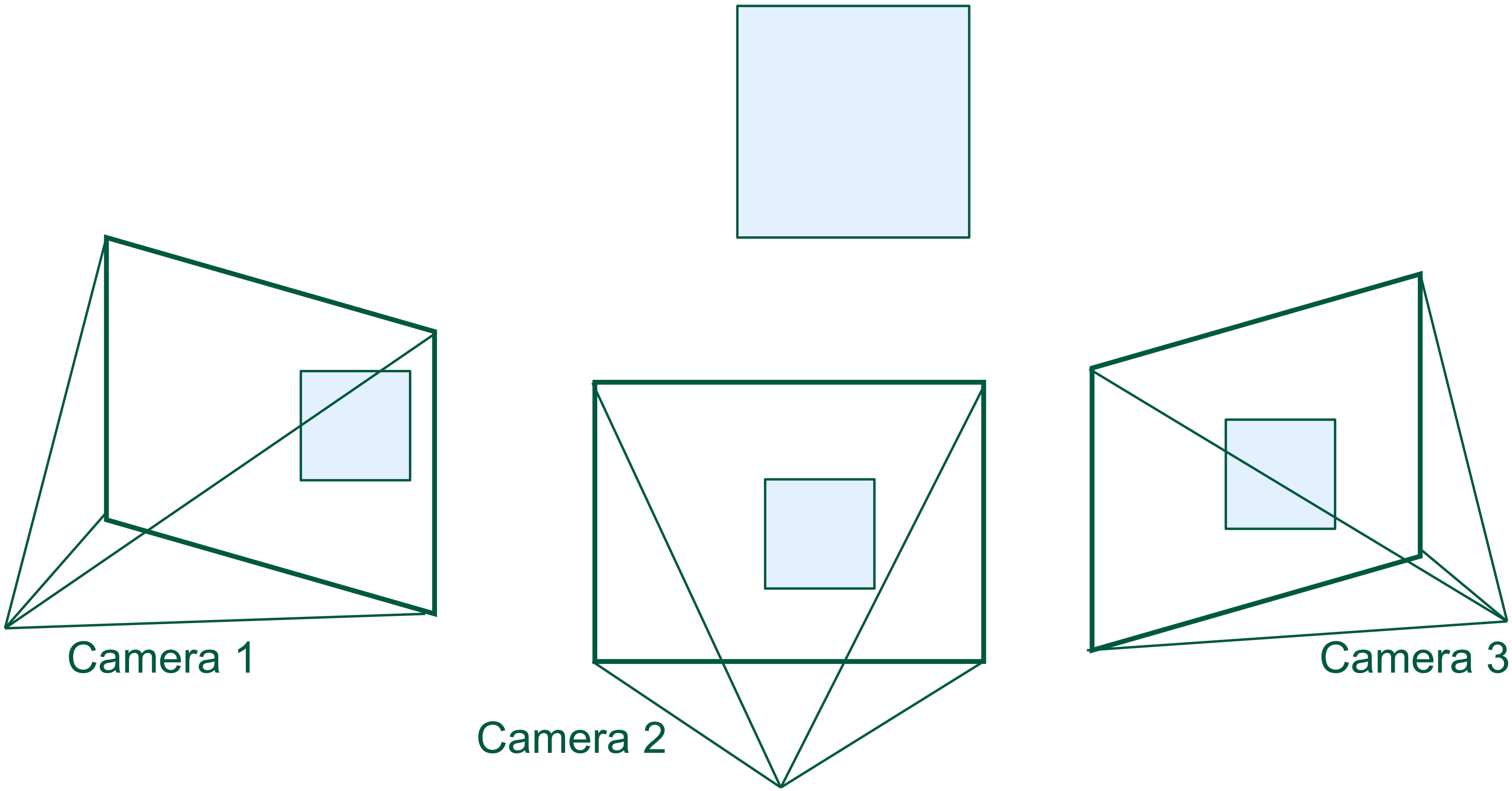
See also the fundamental matrix song: http://danielwedge.com/fmatrix/)



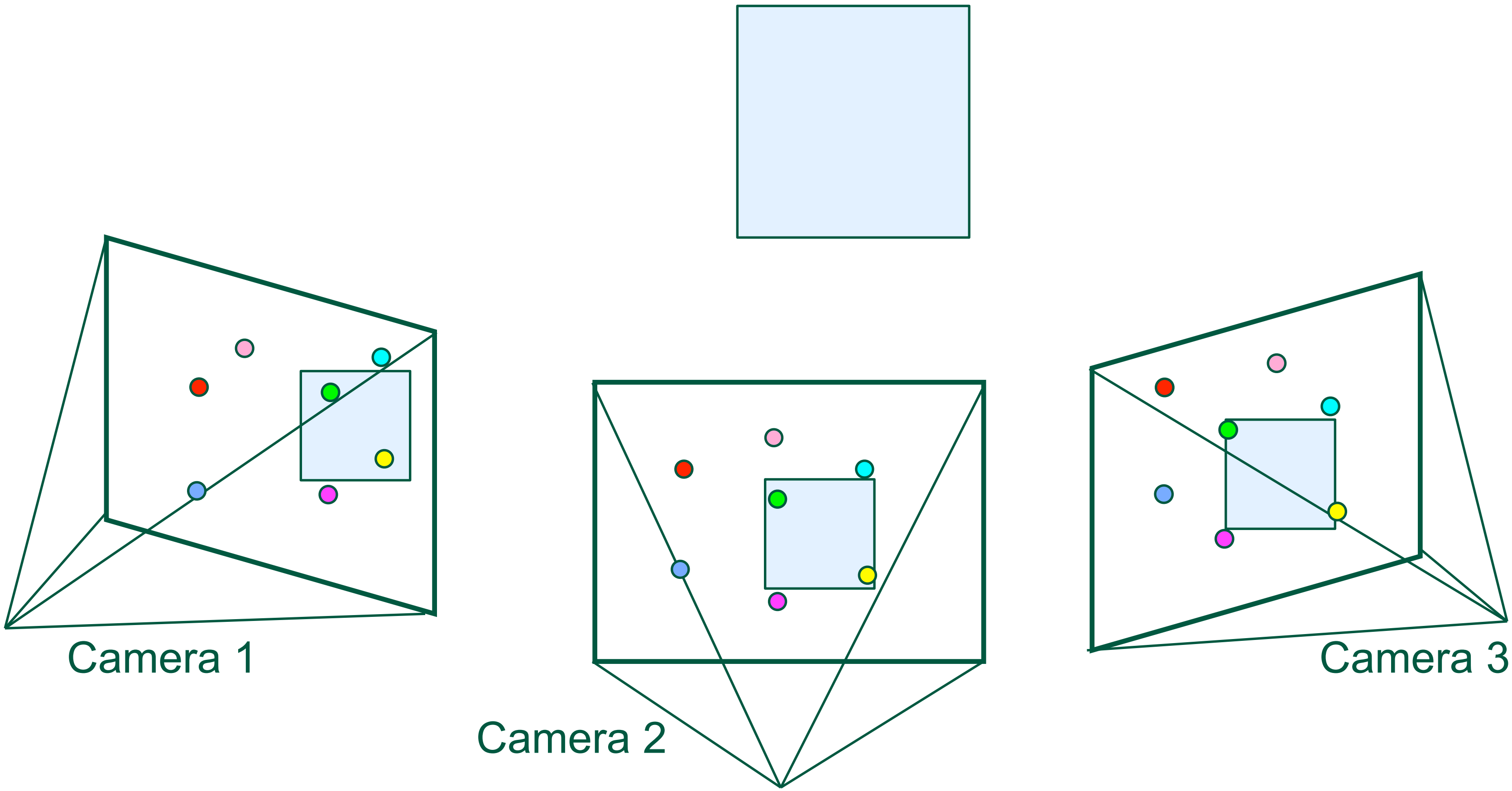
Structure from motion



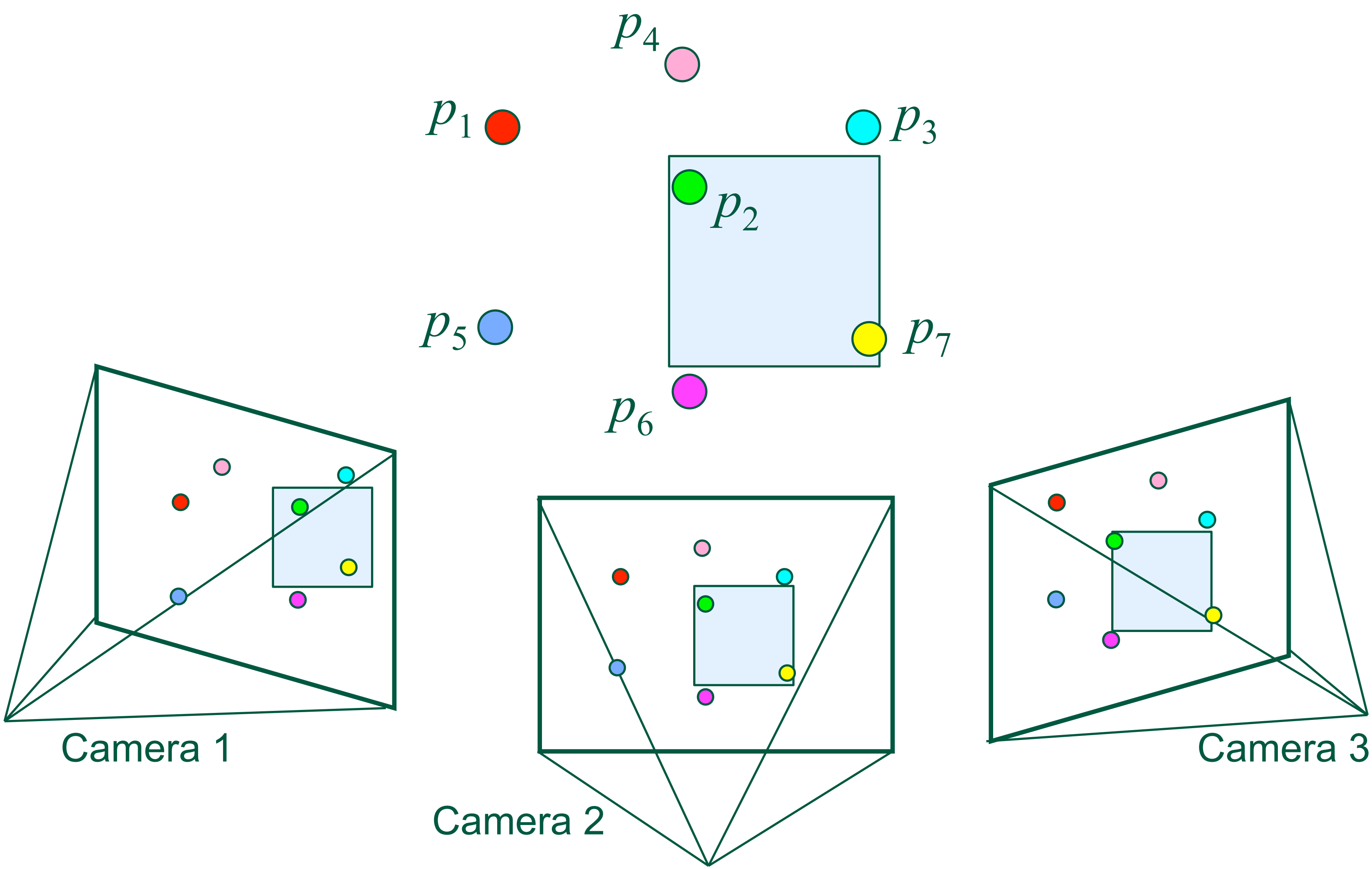
Structure from motion



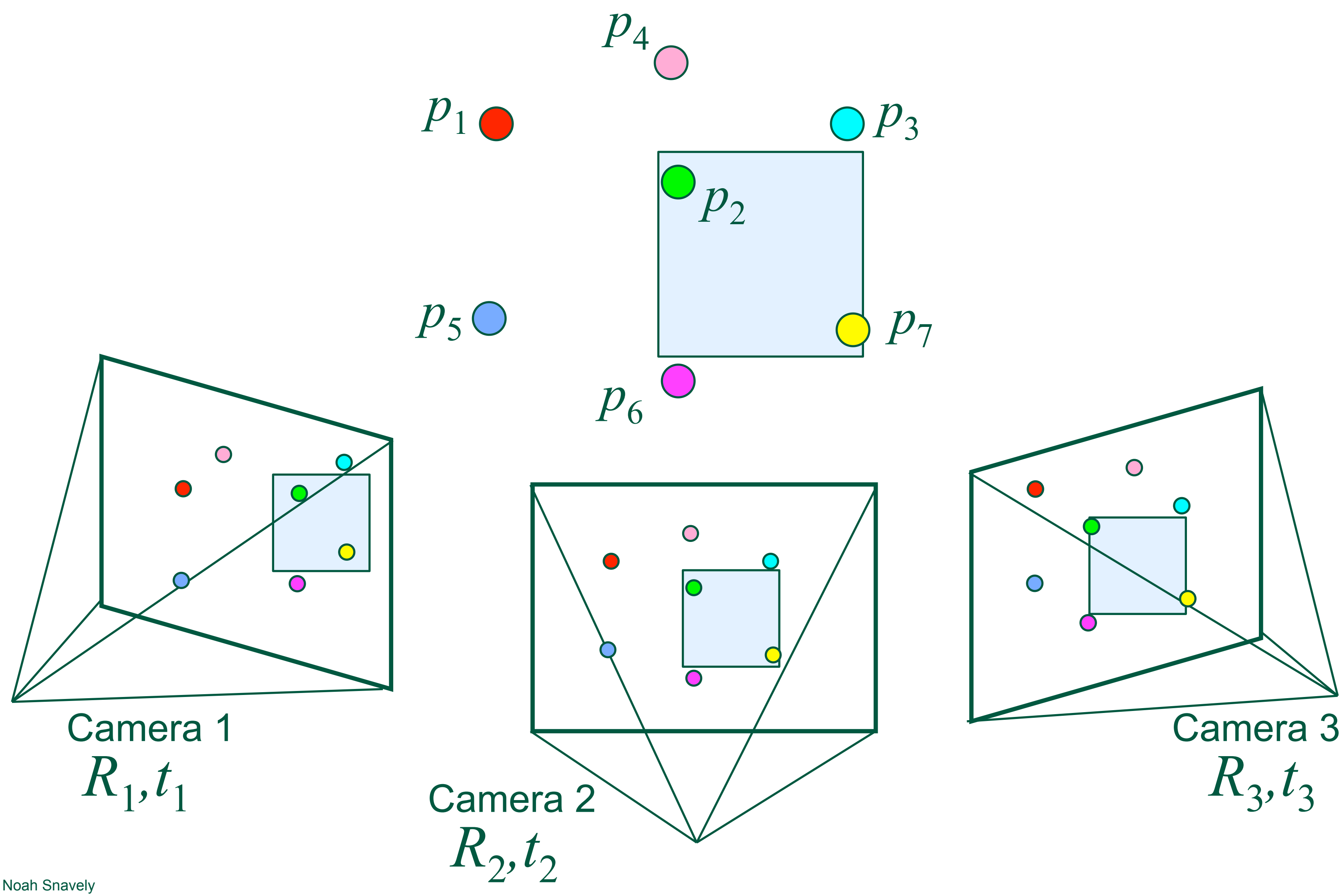
Structure from motion



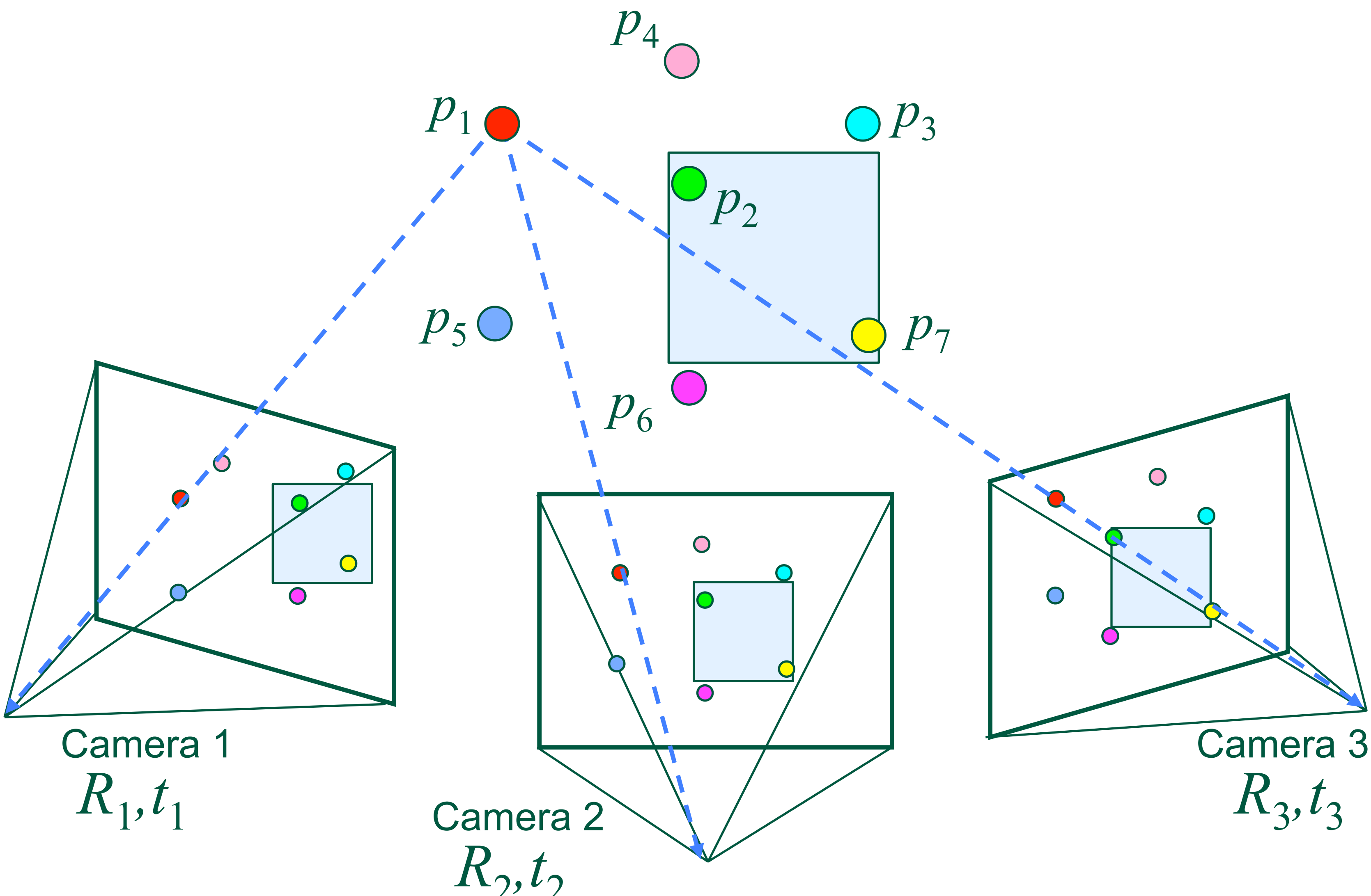
Structure from motion



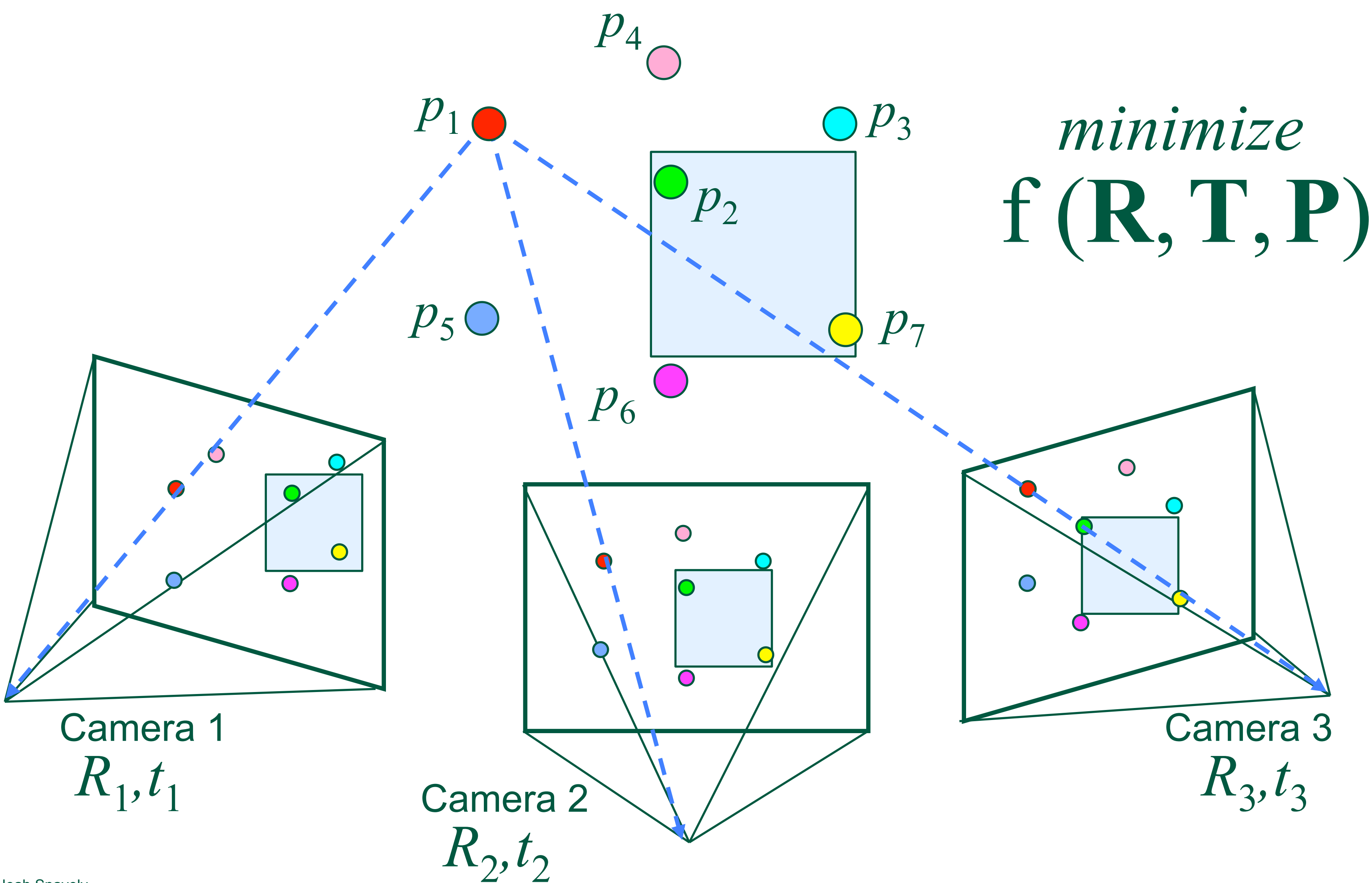
Structure from motion



Structure from motion



Structure from motion



Links

- Code available: http://phototour.cs.washington.edu/bundler/
- http://phototour.cs.washington.edu/
- http://livelabs.com/photosynth/
- http://www.cs.cornell.edu/~snavely/