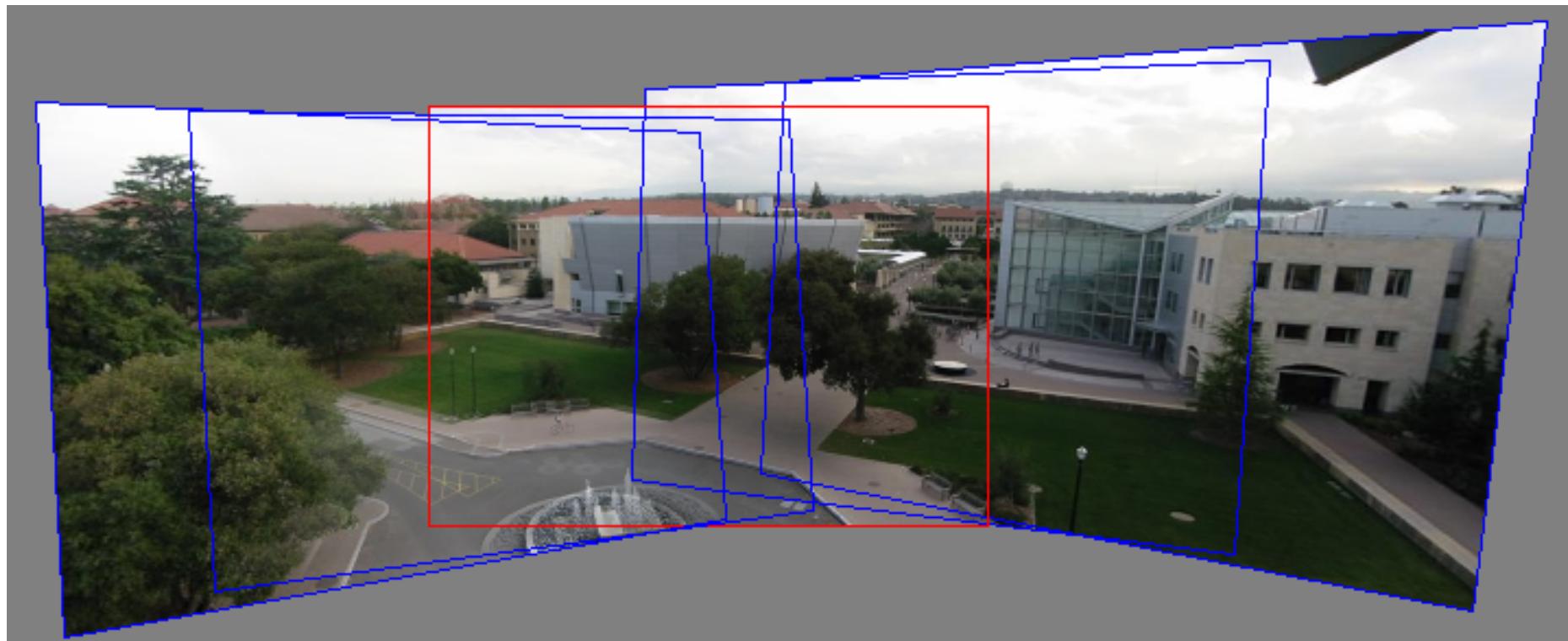
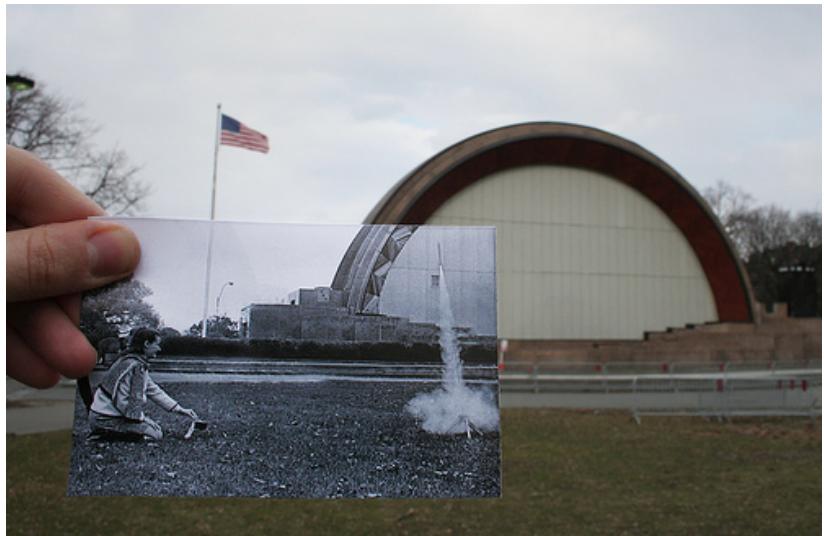
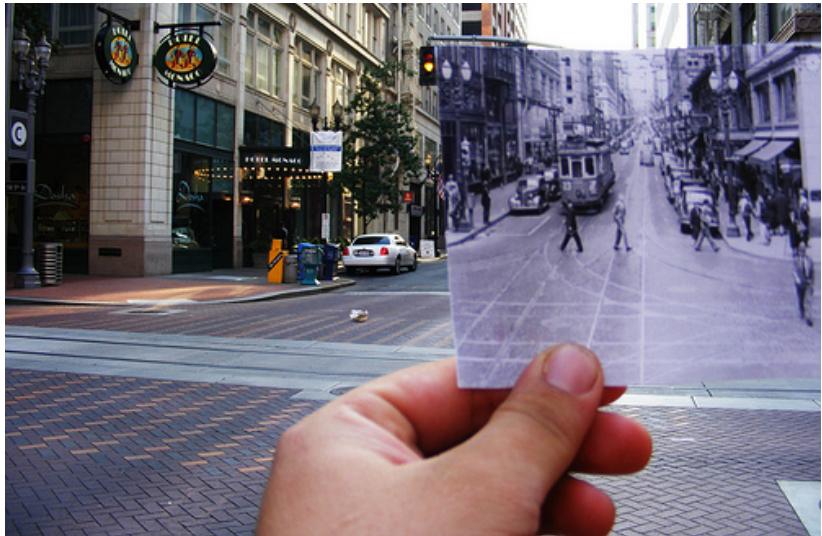


Image alignment



A look into the past



<http://blog.flickr.net/en/2010/01/27/a-look-into-the-past/>

A look into the past

Leningrad during the blockade



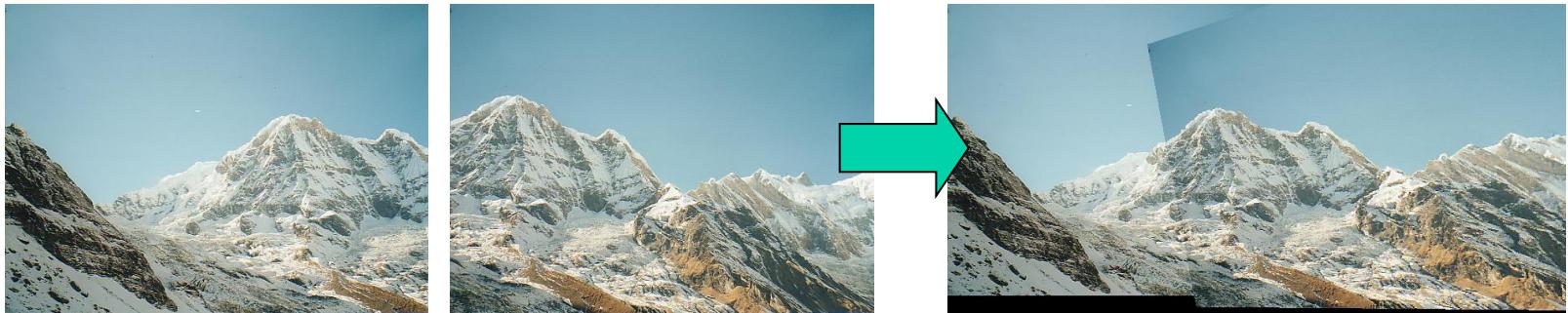
<http://komen-dant.livejournal.com/345684.html>

Bing streetside images

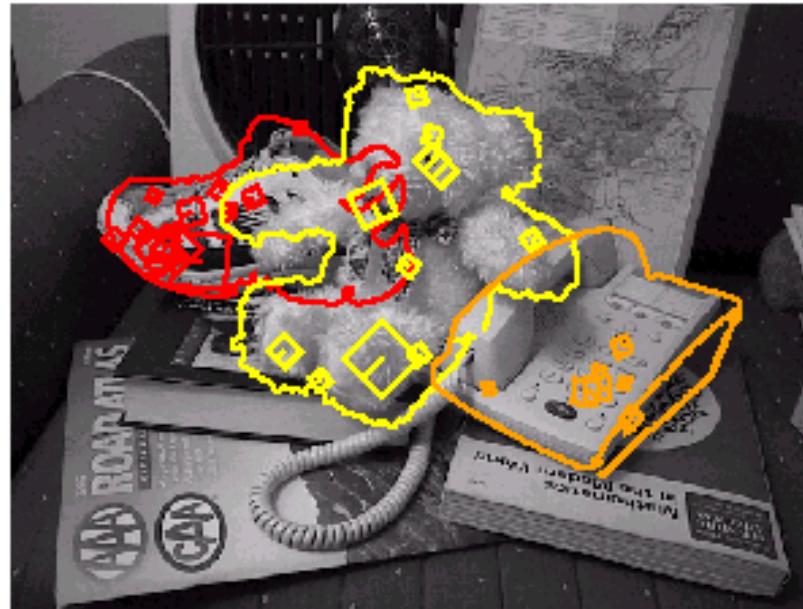


<http://www.bing.com/community/blogs/maps/archive/2010/01/12/new-bing-maps-application-streetside-photos.aspx>

Image alignment: Applications

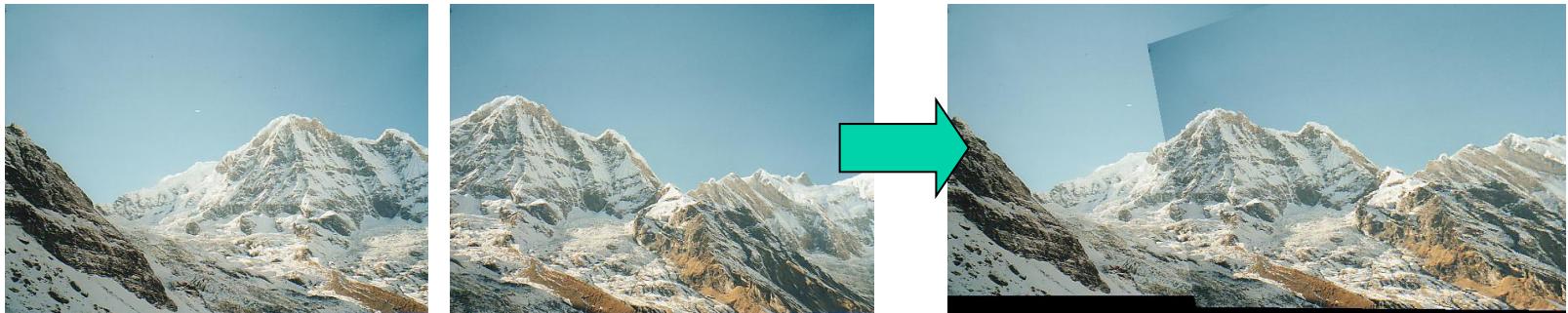


Panorama stitching

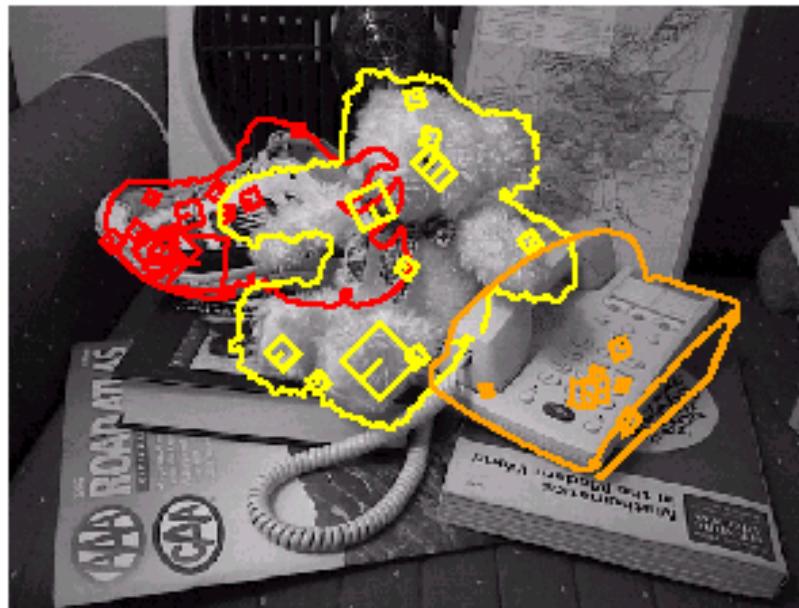
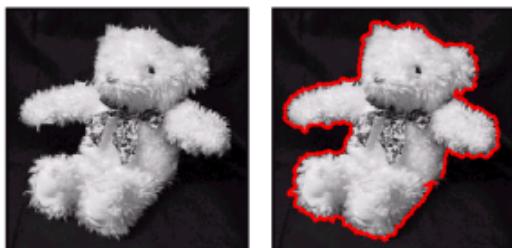
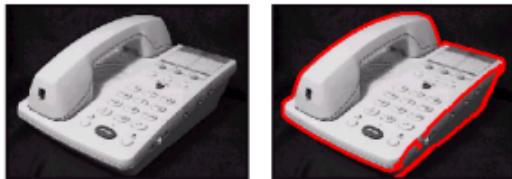


Recognition
of object
instances

Image alignment: Challenges

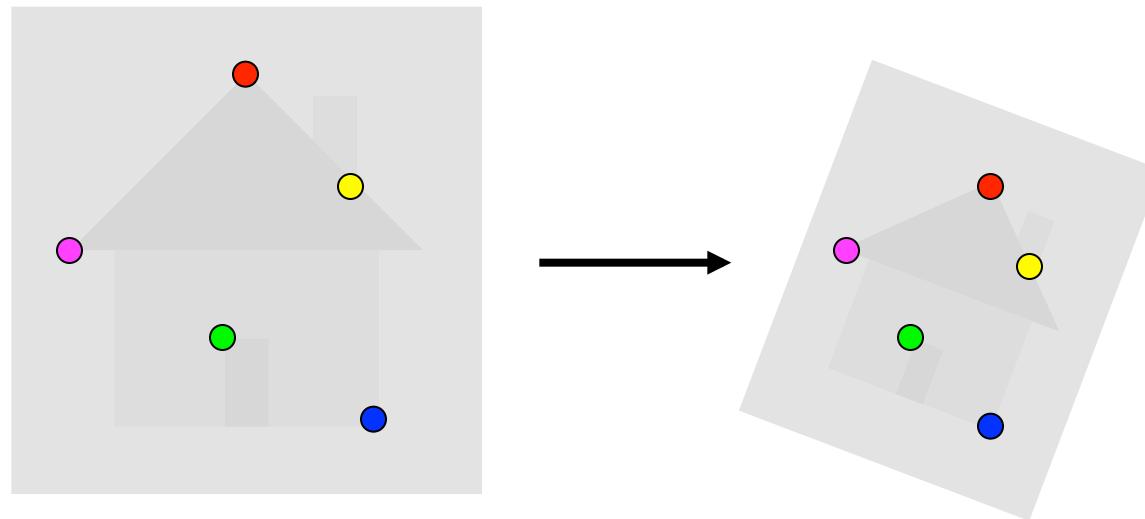


Small degree of overlap
Intensity changes



Occlusion,
clutter

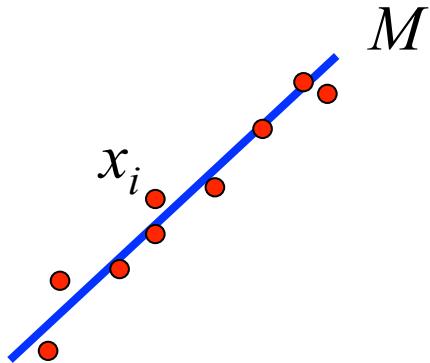
Image alignment



- Two families of approaches:
 - **Direct (pixel-based) alignment**
 - Search for alignment where most pixels agree
 - **Feature-based alignment**
 - Search for alignment where *extracted features* agree
 - Can be verified using pixel-based alignment

Alignment as fitting

- Previous lectures: fitting a model to features in one image

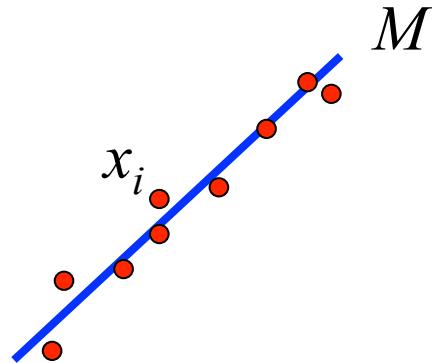


Find model M that minimizes

$$\sum_i \text{residual}(x_i, M)$$

Alignment as fitting

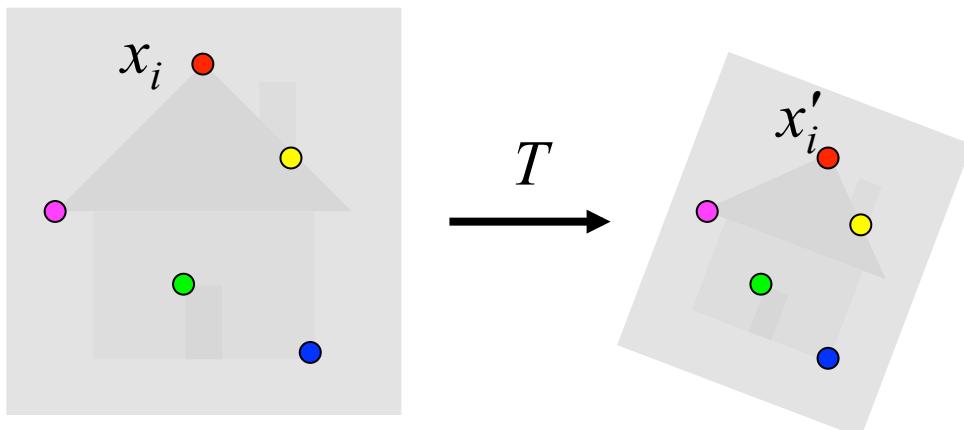
- Previous lectures: fitting a model to features in one image



Find model M that minimizes

$$\sum_i \text{residual}(x_i, M)$$

- Alignment: fitting a model to a transformation between pairs of features (*matches*) in two images

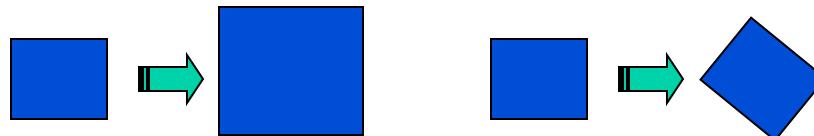


Find transformation T that minimizes

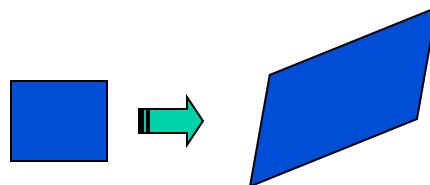
$$\sum_i \text{residual}(T(x_i), x'_i)$$

2D transformation models

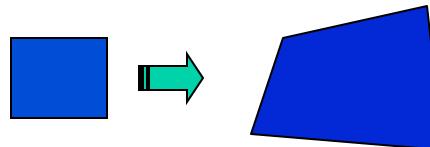
- Similarity
(translation,
scale, rotation)



- Affine

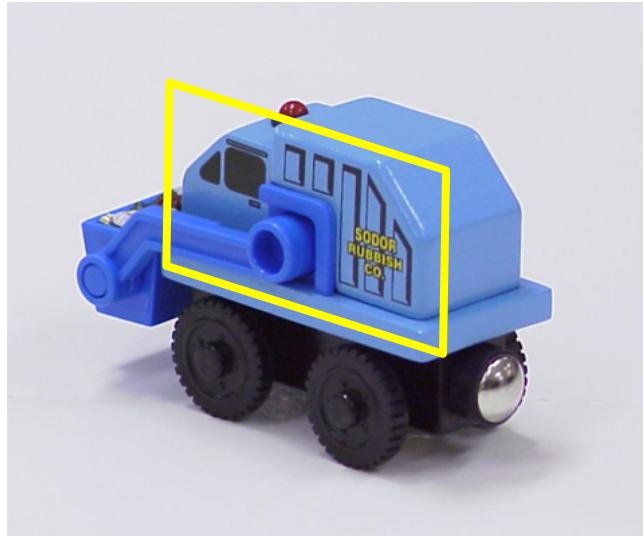


- Projective
(homography)



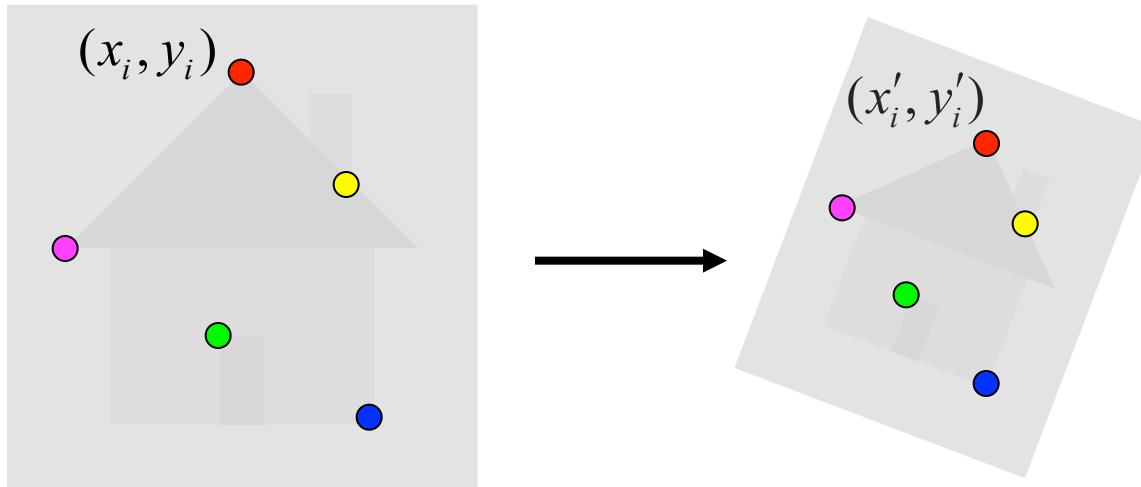
Let's start with affine transformations

- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras
- Can be used to initialize fitting for more complex models



Fitting an affine transformation

- Assume we know the correspondences, how do we get the transformation?



$$\begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$
$$\begin{bmatrix} x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ \dots & & & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} x'_i \\ y'_i \\ \dots \end{bmatrix}$$

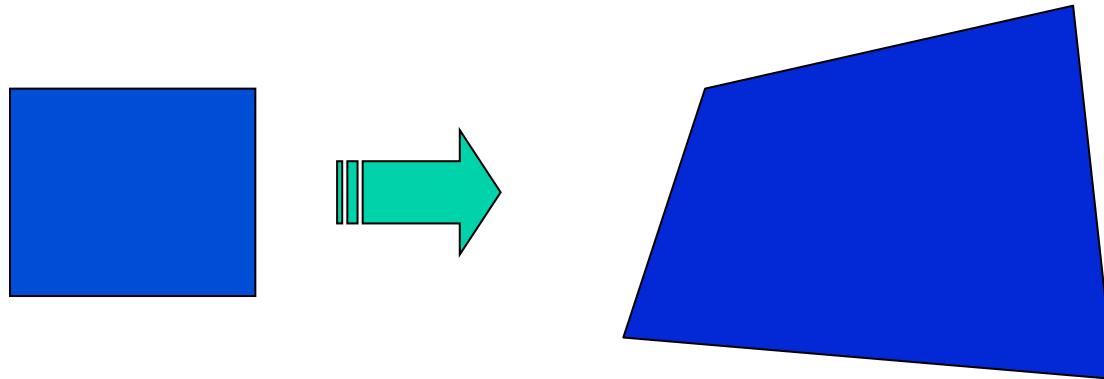
Fitting an affine transformation

$$\begin{bmatrix} & & \cdots \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x'_i \\ y'_i \\ \cdots \end{bmatrix}$$

- Linear system with six unknowns
- Each match gives us two linearly independent equations: need at least three to solve for the transformation parameters

Fitting a plane projective transformation

- **Homography:** plane projective transformation
(transformation taking a quad to another arbitrary quad)



Homography

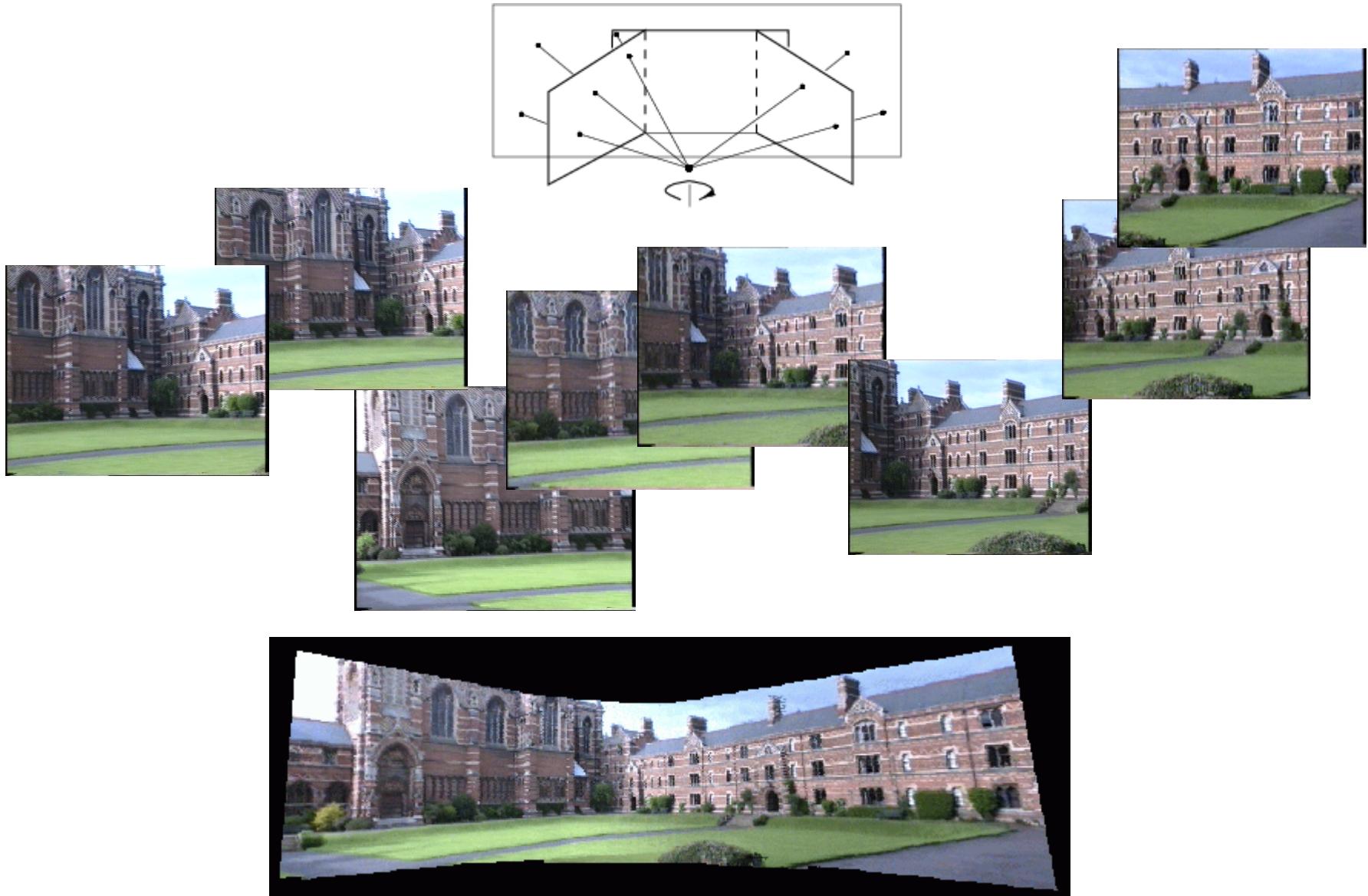
- The transformation between two views of a planar surface



- The transformation between images from two cameras that share the same center



Application: Panorama stitching



Source: Hartley & Zisserman

Fitting a homography

- Recall: homogeneous coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Converting *to* homogeneous
image coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

Converting *from* homogeneous
image coordinates

Fitting a homography

- Recall: homogeneous coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Converting *to* homogeneous
image coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

Converting *from* homogeneous
image coordinates

- Equation for homography:

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Fitting a homography

- Equation for homography:

$$\lambda \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \quad \lambda \mathbf{x}'_i = \mathbf{H} \mathbf{x}_i$$
$$\mathbf{x}'_i \times \mathbf{H} \mathbf{x}_i = 0$$

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{h}_1^T \mathbf{x}_i \\ \mathbf{h}_2^T \mathbf{x}_i \\ \mathbf{h}_3^T \mathbf{x}_i \end{bmatrix} = \begin{bmatrix} y'_i \mathbf{h}_3^T \mathbf{x}_i - \mathbf{h}_2^T \mathbf{x}_i \\ \mathbf{h}_1^T \mathbf{x}_i - x'_i \mathbf{h}_3^T \mathbf{x}_i \\ x'_i \mathbf{h}_2^T \mathbf{x}_i - y'_i \mathbf{h}_1^T \mathbf{x}_i \end{bmatrix}$$

$$\begin{bmatrix} 0^T & -\mathbf{x}_i^T & y'_i \mathbf{x}_i^T \\ \mathbf{x}_i^T & 0^T & -x'_i \mathbf{x}_i^T \\ -y'_i \mathbf{x}_i^T & x'_i \mathbf{x}_i^T & 0^T \end{bmatrix} \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix} = 0$$

3 equations,
only 2 linearly
independent

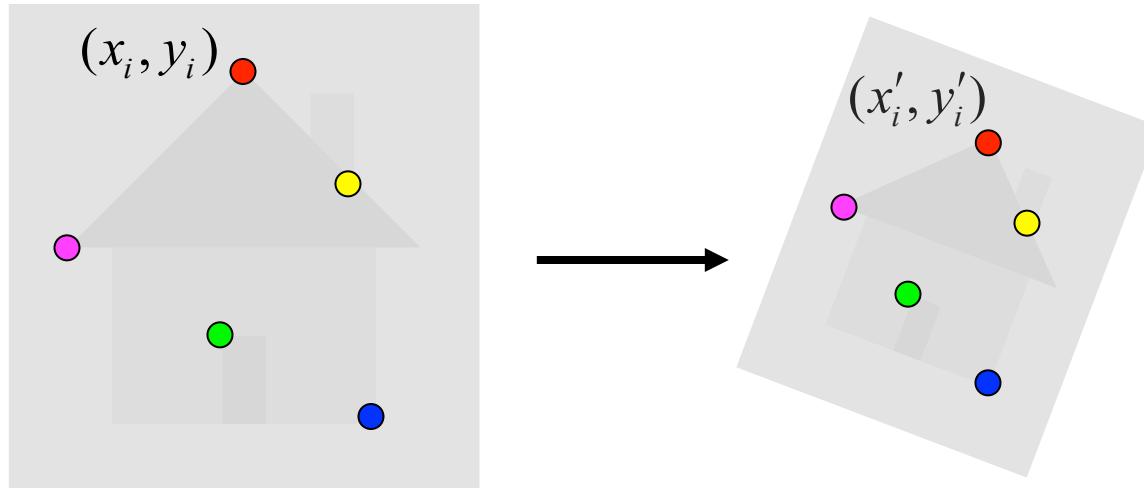
Direct linear transform

$$\begin{bmatrix} 0^T & \mathbf{x}_1^T & -y'_1 \mathbf{x}_1^T \\ \mathbf{x}_1^T & 0^T & -x'_1 \mathbf{x}_1^T \\ \dots & \dots & \dots \\ 0^T & \mathbf{x}_n^T & -y'_n \mathbf{x}_n^T \\ \mathbf{x}_n^T & 0^T & -x'_n \mathbf{x}_n^T \end{bmatrix} \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix} = 0 \quad \mathbf{A h} = 0$$

- \mathbf{H} has 8 degrees of freedom (9 parameters, but scale is arbitrary)
- One match gives us two linearly independent equations
- Homogeneous least squares: find \mathbf{h} minimizing $\|\mathbf{Ah}\|^2$
 - Eigenvector of $\mathbf{A}^T \mathbf{A}$ corresponding to smallest eigenvalue
 - Four matches needed for a minimal solution

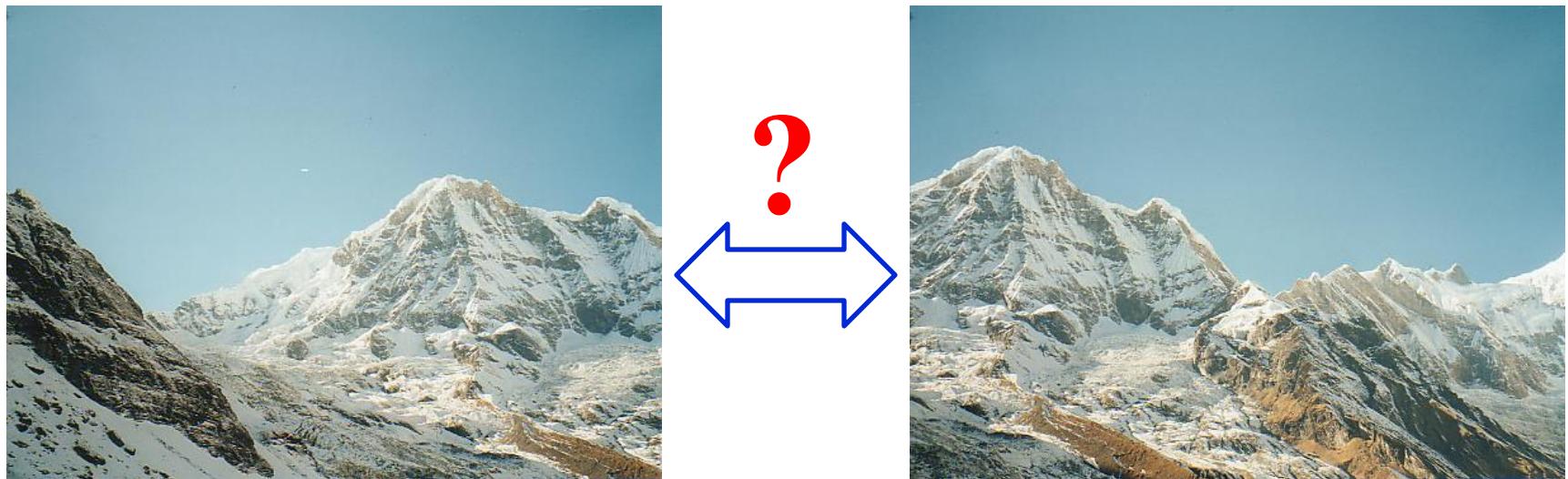
Robust feature-based alignment

- So far, we've assumed that we are given a set of “ground-truth” correspondences between the two images we want to align
- What if we don't know the correspondences?



Robust feature-based alignment

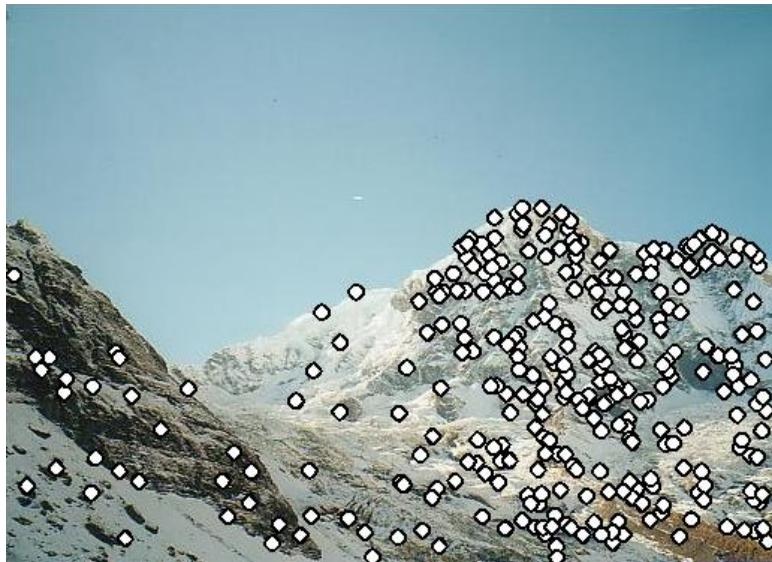
- So far, we've assumed that we are given a set of “ground-truth” correspondences between the two images we want to align
- What if we don't know the correspondences?



Robust feature-based alignment

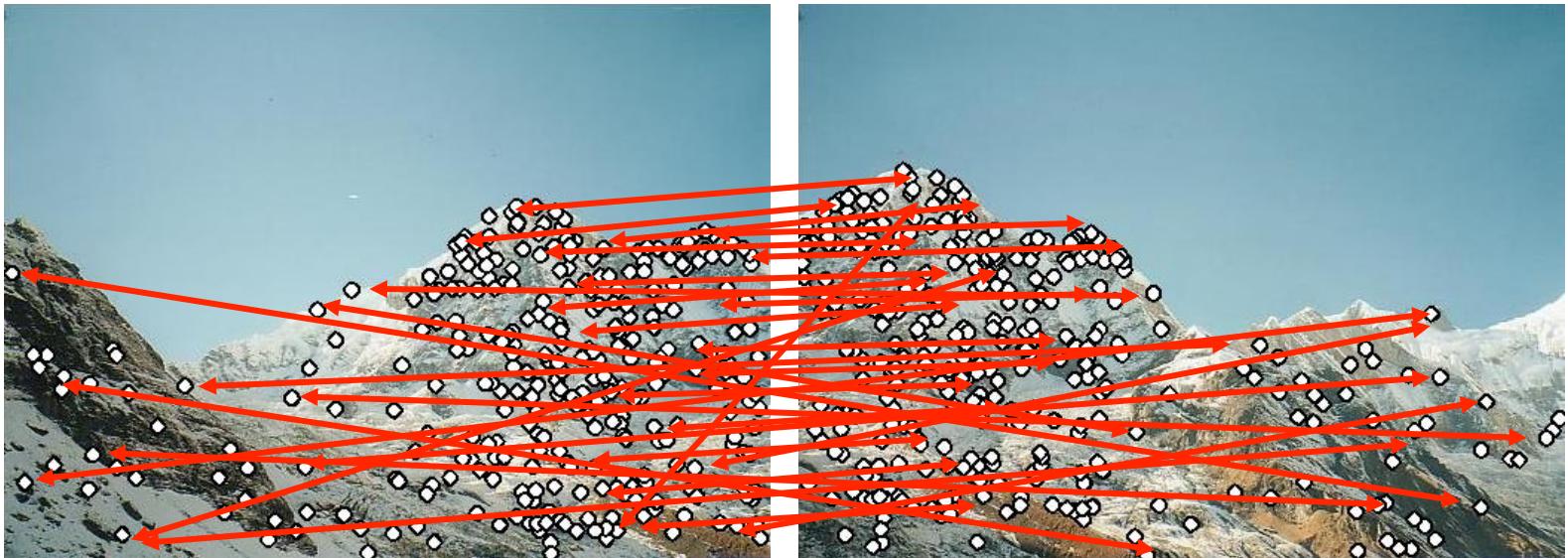


Robust feature-based alignment



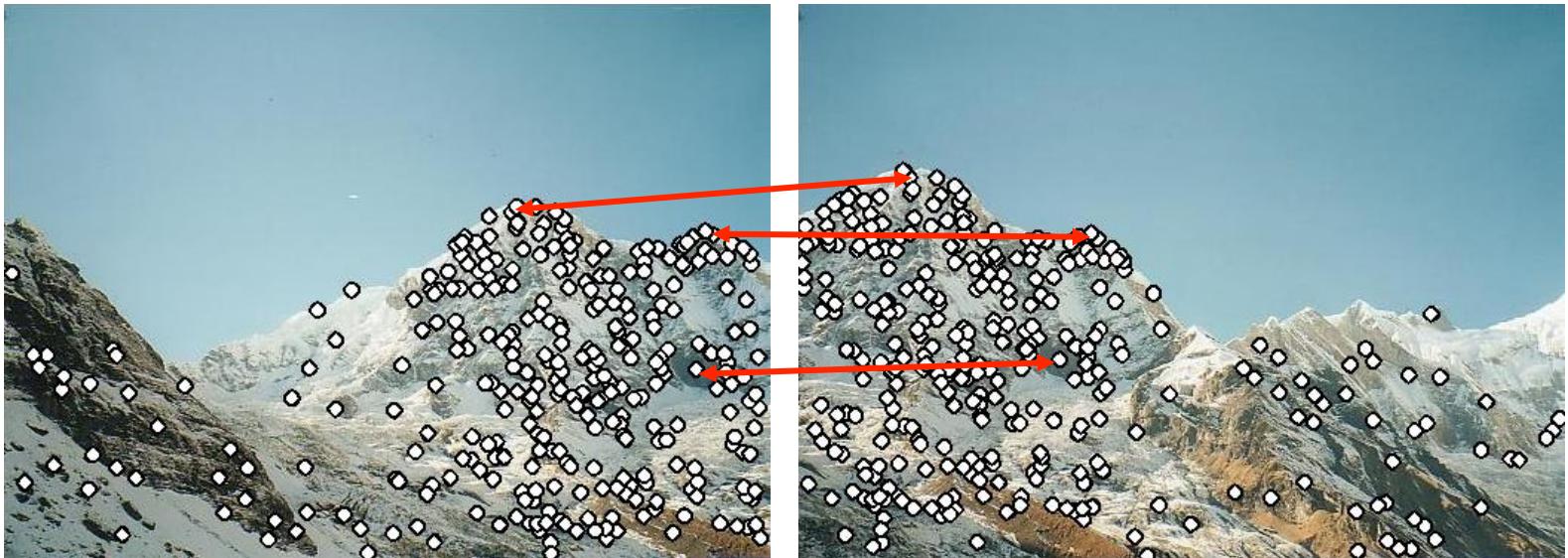
- Extract features

Robust feature-based alignment



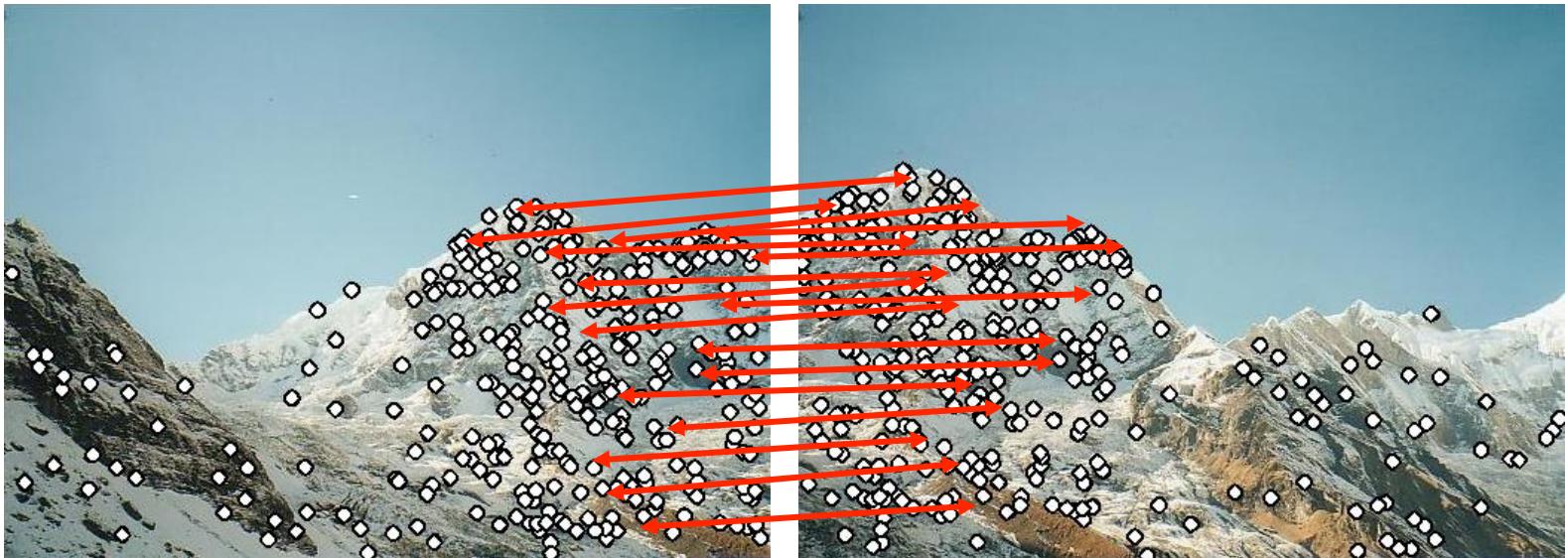
- Extract features
- Compute *putative matches*

Robust feature-based alignment



- Extract features
- Compute *putative matches*
- Loop:
 - *Hypothesize* transformation T

Robust feature-based alignment



- Extract features
- Compute *putative matches*
- Loop:
 - *Hypothesize* transformation T
 - *Verify* transformation (search for other matches consistent with T)

Robust feature-based alignment



- Extract features
- Compute *putative matches*
- Loop:
 - *Hypothesize* transformation T
 - *Verify* transformation (search for other matches consistent with T)

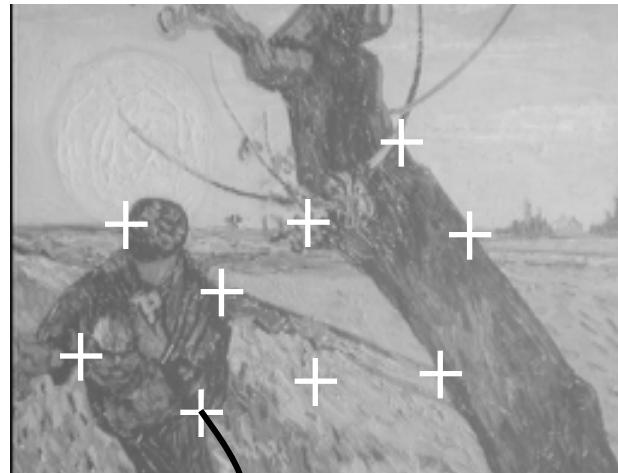
Generating putative correspondences



?
↔

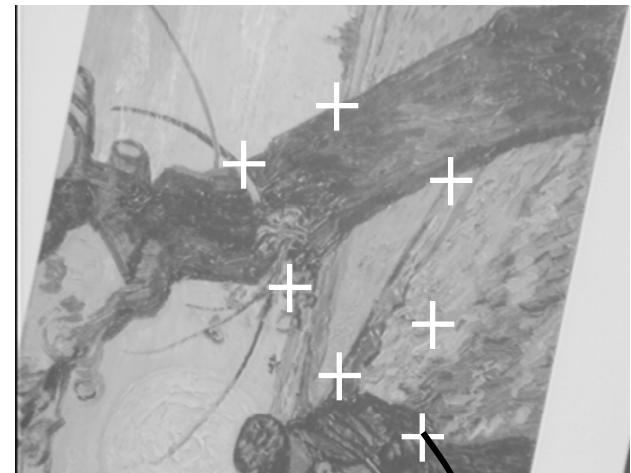


Generating putative correspondences



?

↔



feature
descriptor

?

=

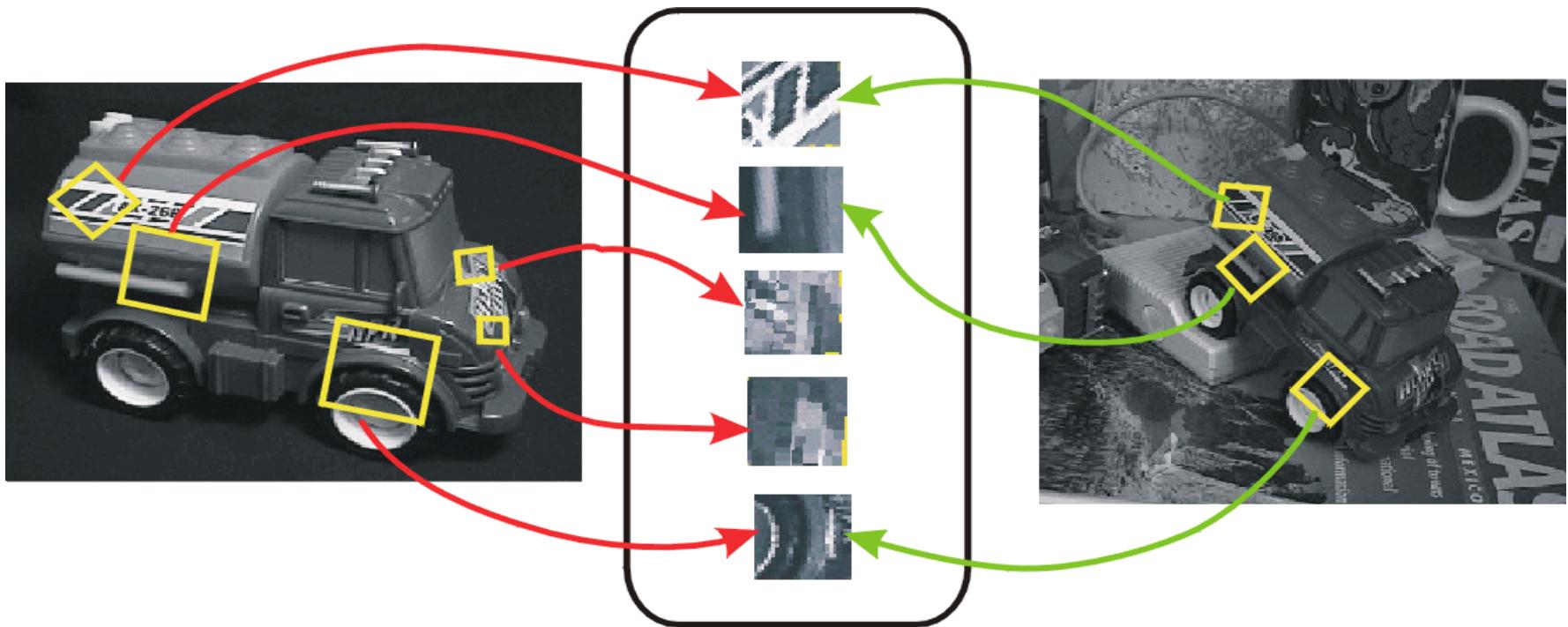


feature
descriptor

- Need to compare *feature descriptors* of local patches surrounding interest points

Feature descriptors

- Covariant detectors => invariant descriptors



Feature descriptors

- Simplest descriptor: vector of raw intensity values
- How to compare two such vectors?
 - Sum of squared differences (SSD)

$$\text{SSD}(u, v) = \sum_i (u_i - v_i)^2$$

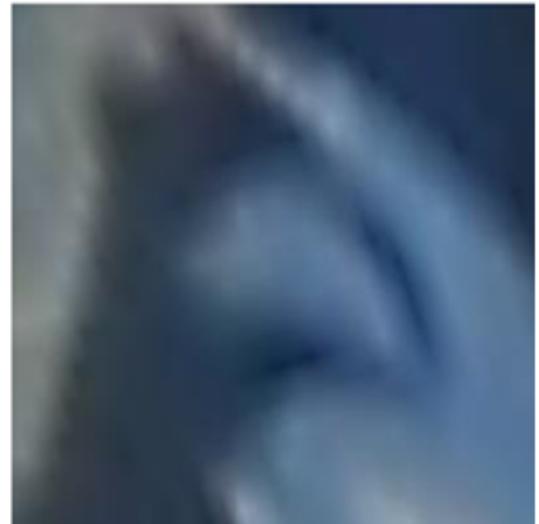
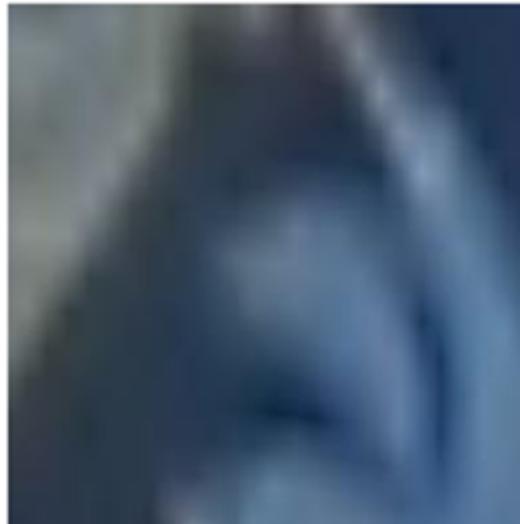
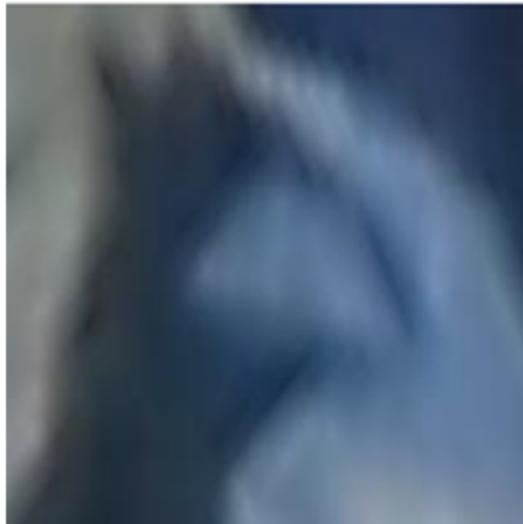
- Not invariant to intensity change
- Normalized correlation

$$\rho(u, v) = \frac{\sum_i (u_i - \bar{u})(v_i - \bar{v})}{\sqrt{\left(\sum_j (u_j - \bar{u})^2 \right) \left(\sum_j (v_j - \bar{v})^2 \right)}}$$

- Invariant to affine intensity change

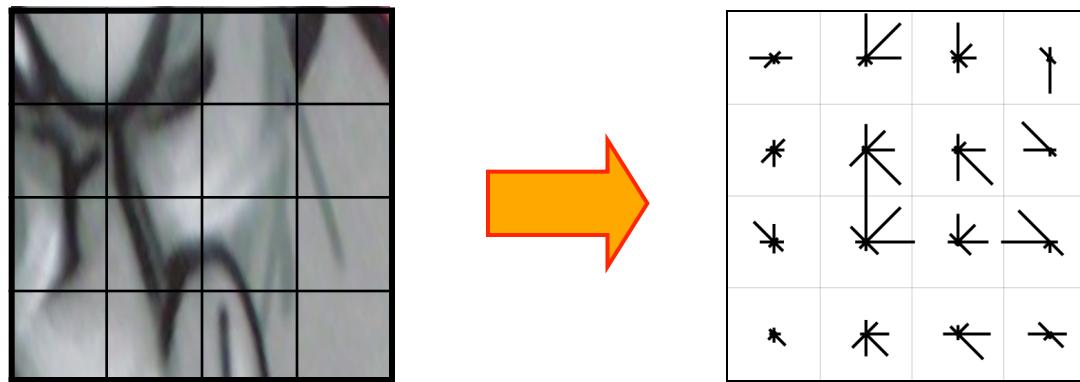
Disadvantage of intensity vectors as descriptors

- Small deformations can affect the matching score a lot



Feature descriptors: SIFT

- Descriptor computation:
 - Divide patch into 4×4 sub-patches
 - Compute histogram of gradient orientations (8 reference angles) inside each sub-patch
 - Resulting descriptor: $4 \times 4 \times 8 = 128$ dimensions



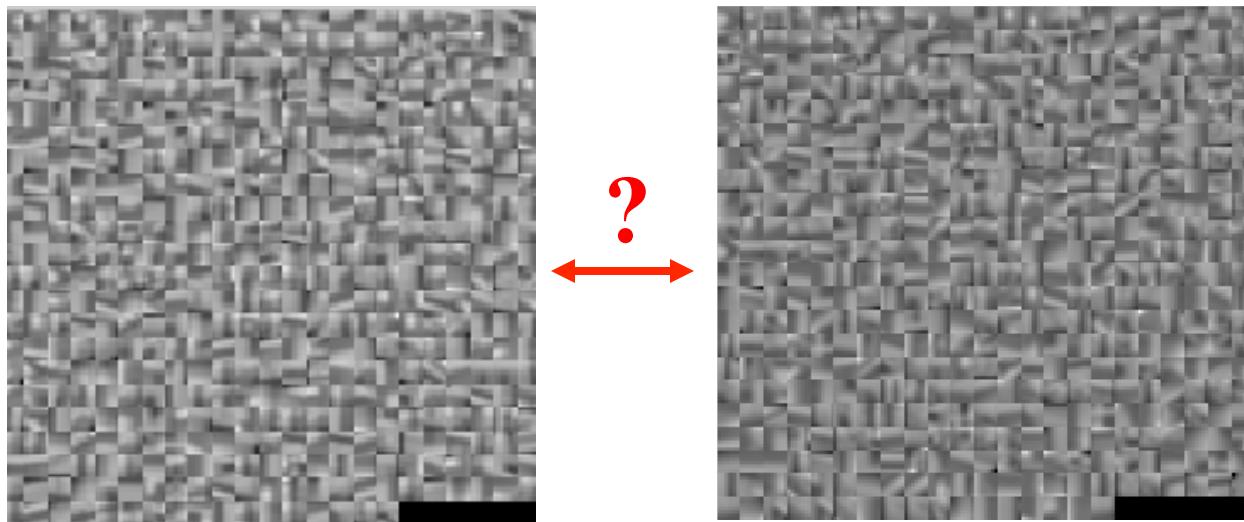
David G. Lowe. ["Distinctive image features from scale-invariant keypoints."](#) IJCV 60 (2), pp. 91-110, 2004.

Feature descriptors: SIFT

- Descriptor computation:
 - Divide patch into 4×4 sub-patches
 - Compute histogram of gradient orientations (8 reference angles) inside each sub-patch
 - Resulting descriptor: $4 \times 4 \times 8 = 128$ dimensions
- Advantage over raw vectors of pixel values
 - Gradients less sensitive to illumination change
 - Pooling of gradients over the sub-patches achieves robustness to small shifts, but still preserves some spatial information

Feature matching

- Generating *putative matches*: for each patch in one image, find a short list of patches in the other image that could match it based solely on appearance



RANSAC

- The set of putative matches contains a very high percentage of outliers

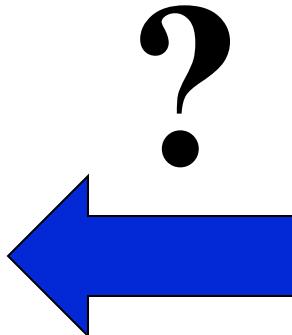
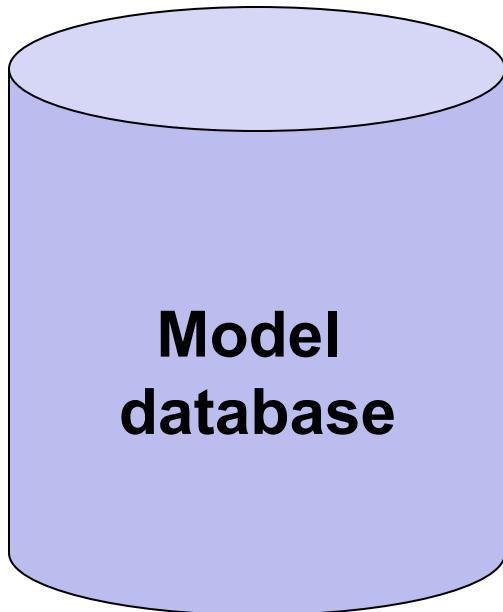
RANSAC loop:

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

Keep the transformation with the largest number of inliers

Scalability: Alignment to large databases

- What if we need to align a test image with thousands or millions of images in a model database?
 - Efficient putative match generation
 - Approximate descriptor similarity search, inverted indices



Large-scale visual search

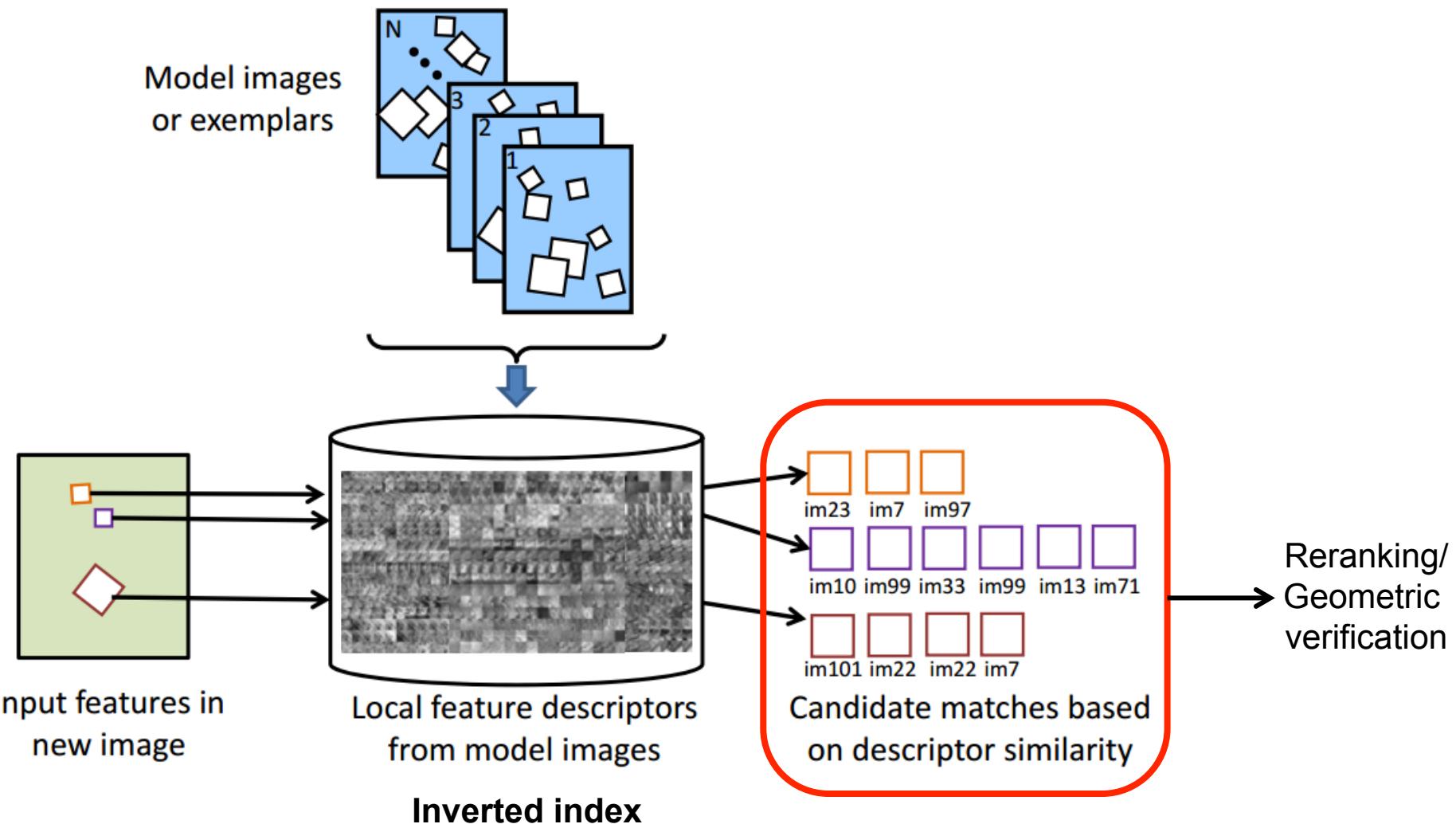
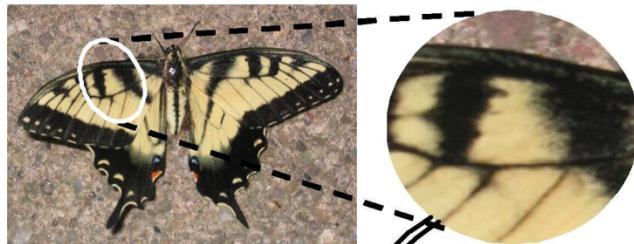


Figure from: Kristen Grauman and Bastian Leibe, [Visual Object Recognition](#), Synthesis Lectures on Artificial Intelligence and Machine Learning, April 2011, Vol. 5, No. 2, Pages 1-181

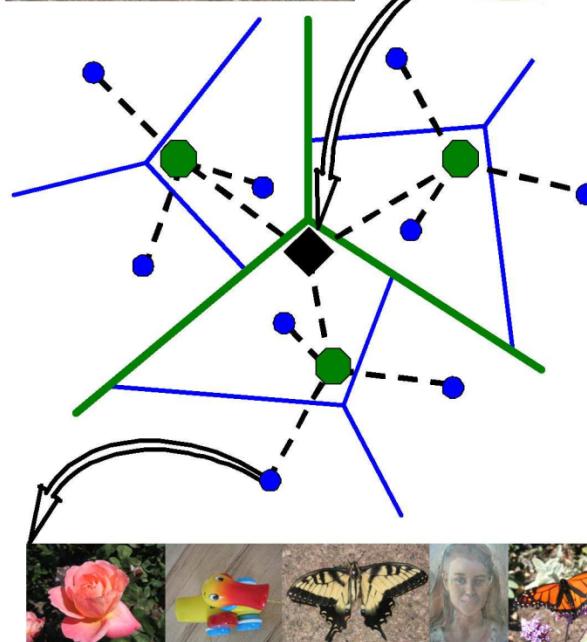
Large-scale visual search

- What if we need to align a test image with thousands or millions of images in a model database?
 - Efficient putative match generation
 - Fast nearest neighbor search, inverted indexes

Test image



Vocabulary tree with inverted index



Database



D. Nistér and H. Stewénius,
[Scalable Recognition with a Vocabulary Tree](#),
CVPR 2006

Review: Alignment

- Estimating transformation parameters
 - Affine
 - Projective (homography)
- Robust alignment
- Large-scale visual search