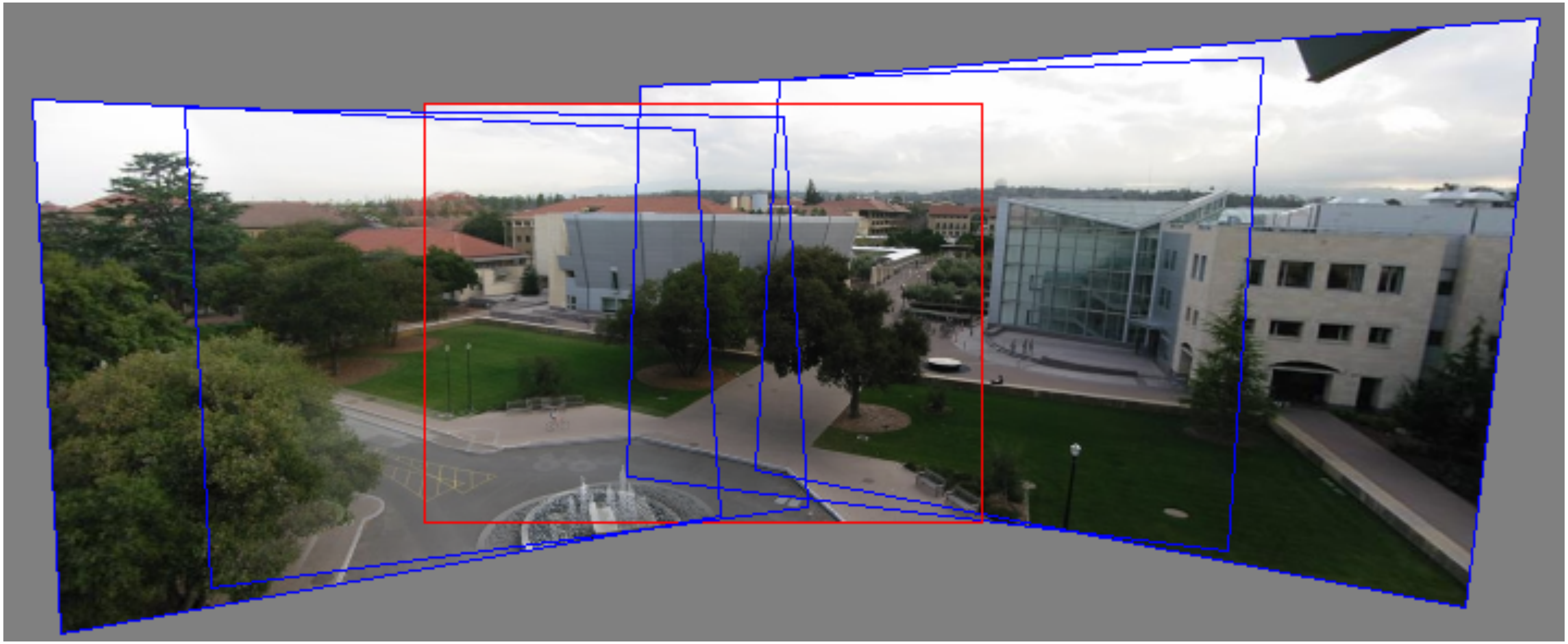
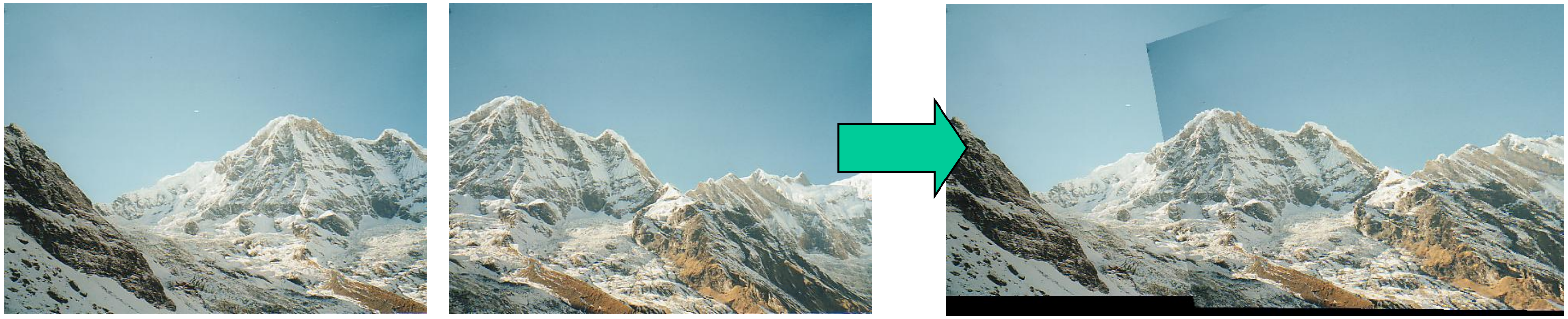
Image alignment



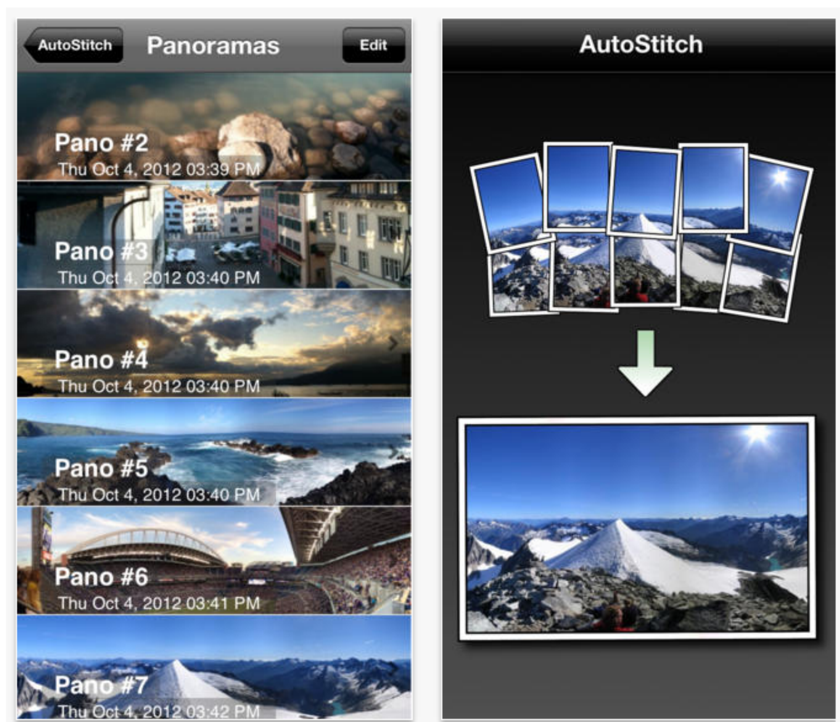
Slides from Derek Hoiem, Svetlana Lazebnik

Image source

Alignment applications



Panorama stitching

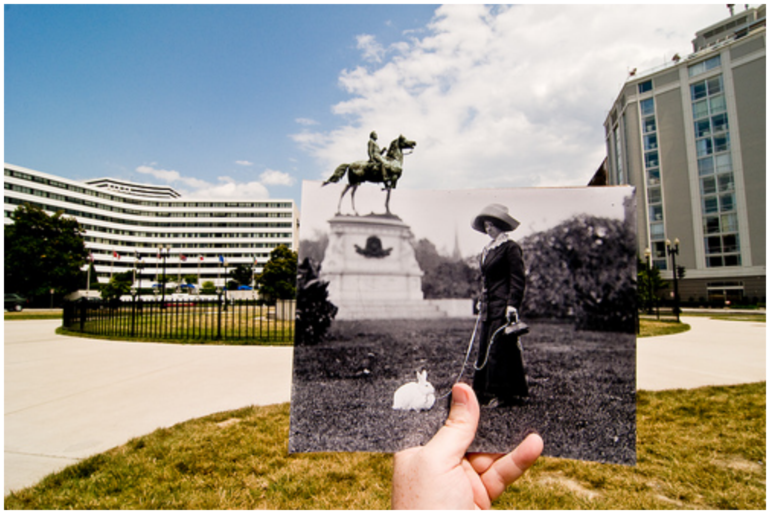


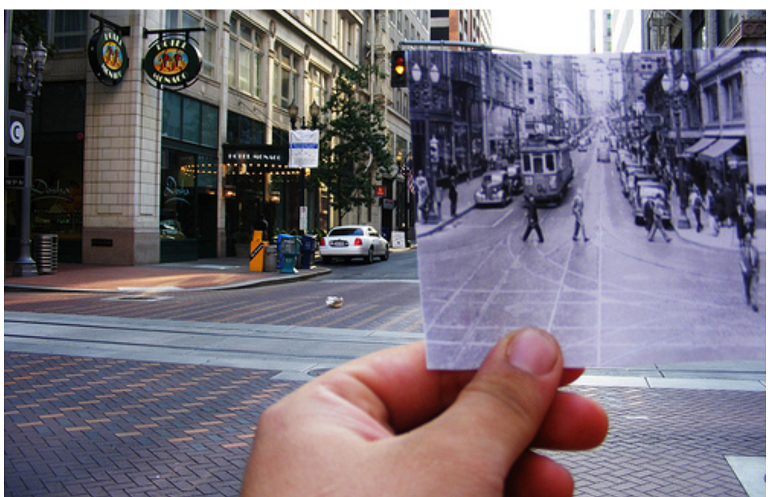
Slide from L. Lazebnik

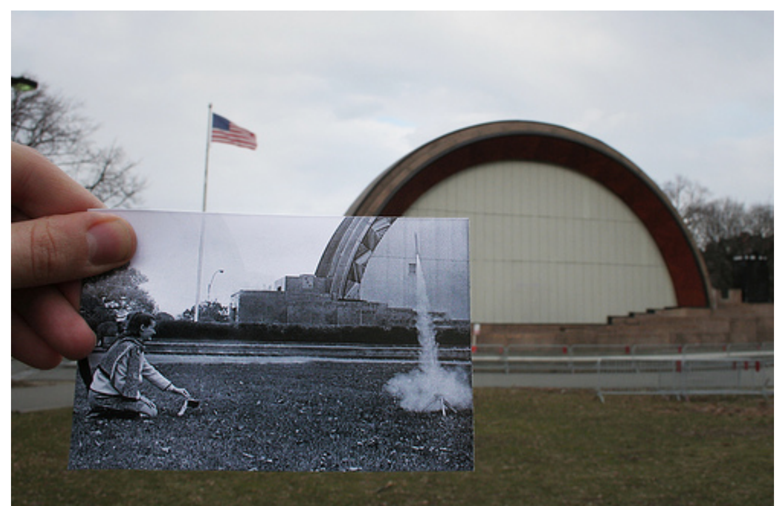
http://matthewalunbrown.com/autostitch/autostitch.html

Alignment applications

- [A look into the past]()



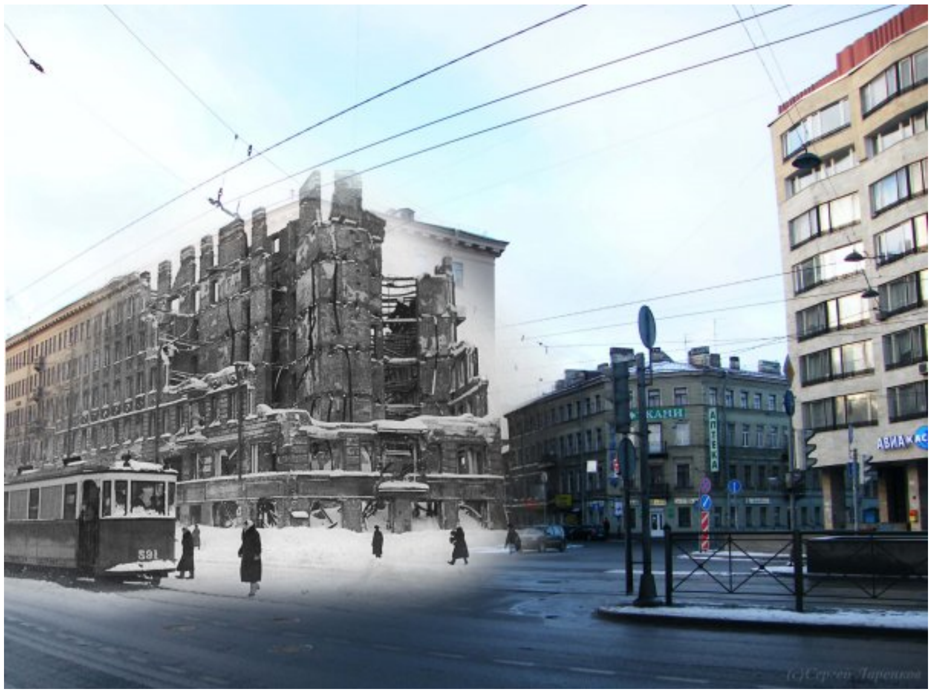




Slide from L. Lazebnik

Alignment applications

- A look into the past



Slide from L. Lazebnik

Alignment applications

- Cool video

Slide from L. Lazebnik

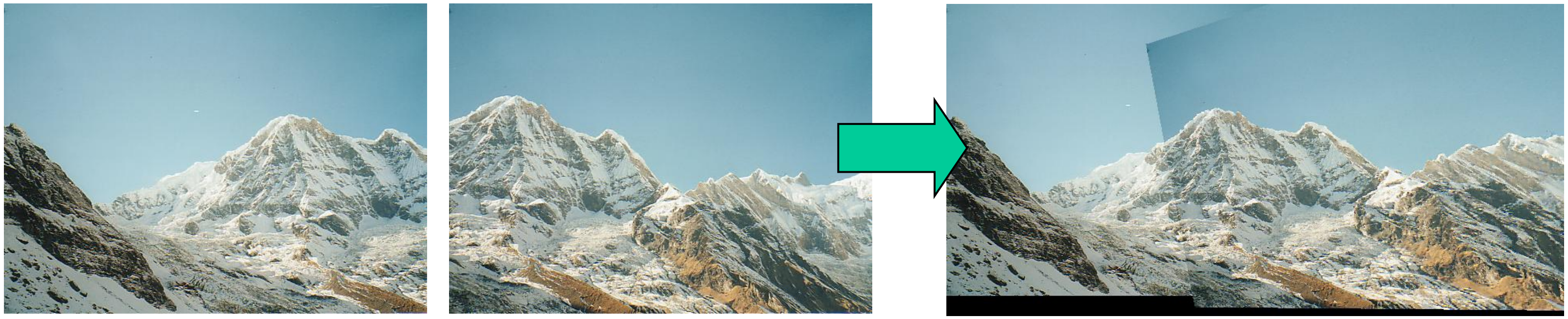
Alignment applications

Instance recognition

T. Weyand and B. Leibe, [Visual landmark recognition from Internet photo collections: A large-scale evaluation](), CVIU 2015

Slide from L. Lazebnik

Alignment challenges



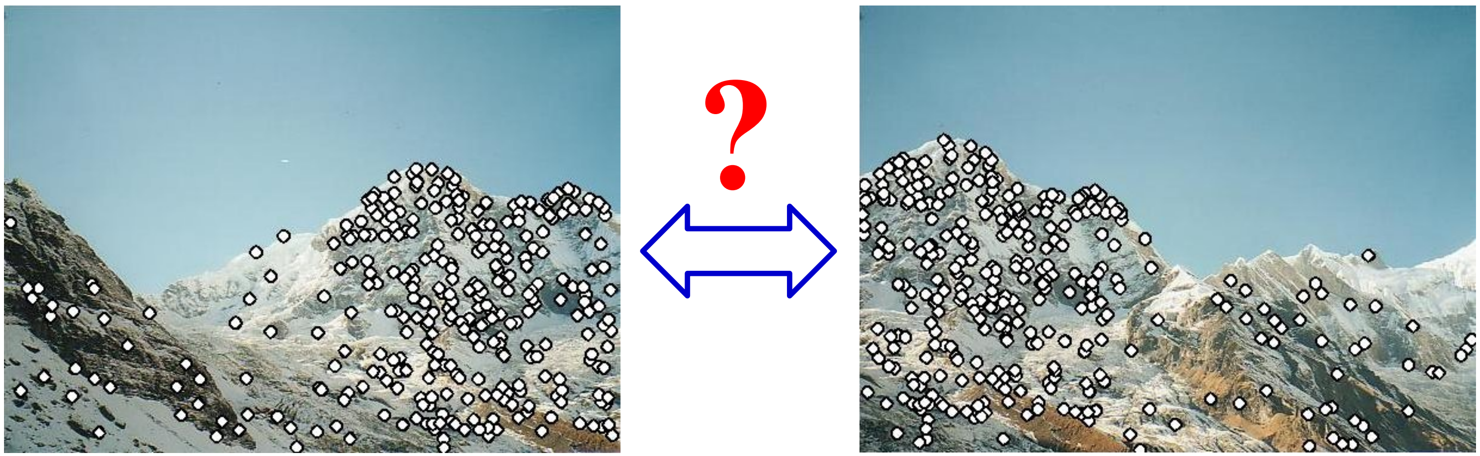
Small degree of overlap
Intensity changes

Occlusion, clutter, viewpoint change

Slide from L. Lazebnik

Feature-based alignment

- Search sets of feature matches that agree in terms of:
  a) Local appearance
  b) Geometric configuration



Slide from L. Lazebnik

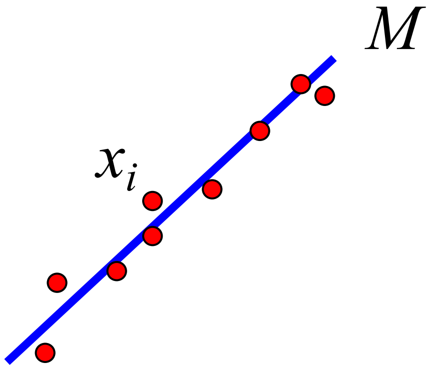
Feature-based alignment: Overview

- Alignment as fitting
  - Affine transformations
  - Homographies
- Robust alignment
  - Descriptor-based feature matching
  - RANSAC
- Applications

Slide from L. Lazebnik

Alignment as fitting

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

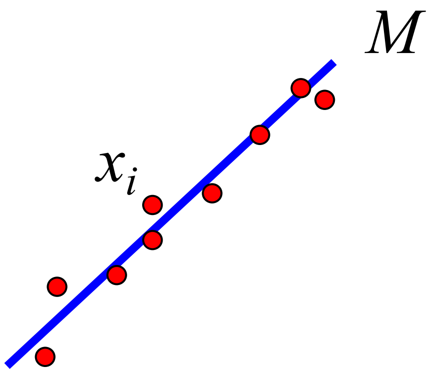


Find model M that minimizes

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

Alignment as fitting

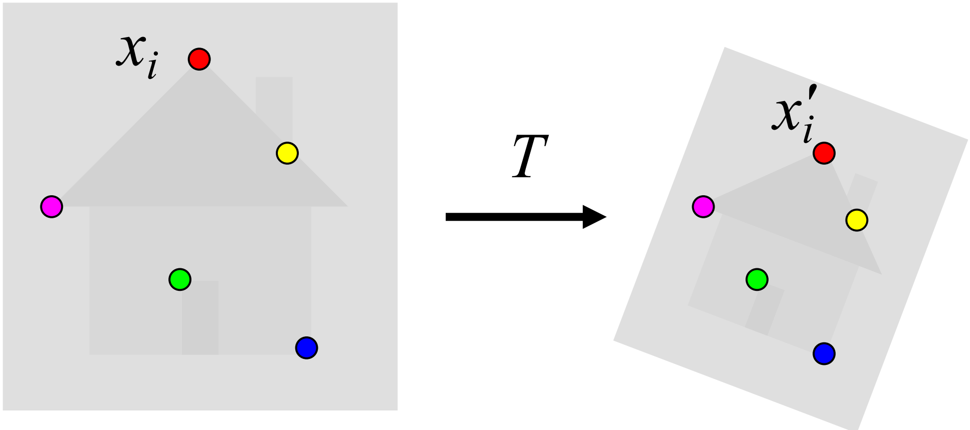
- Previous lectures: fitting a model to features in one image



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

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

Slide from L. Lazebnik

Common transformations

original

Transformed

translation

rotation

aspect

affine

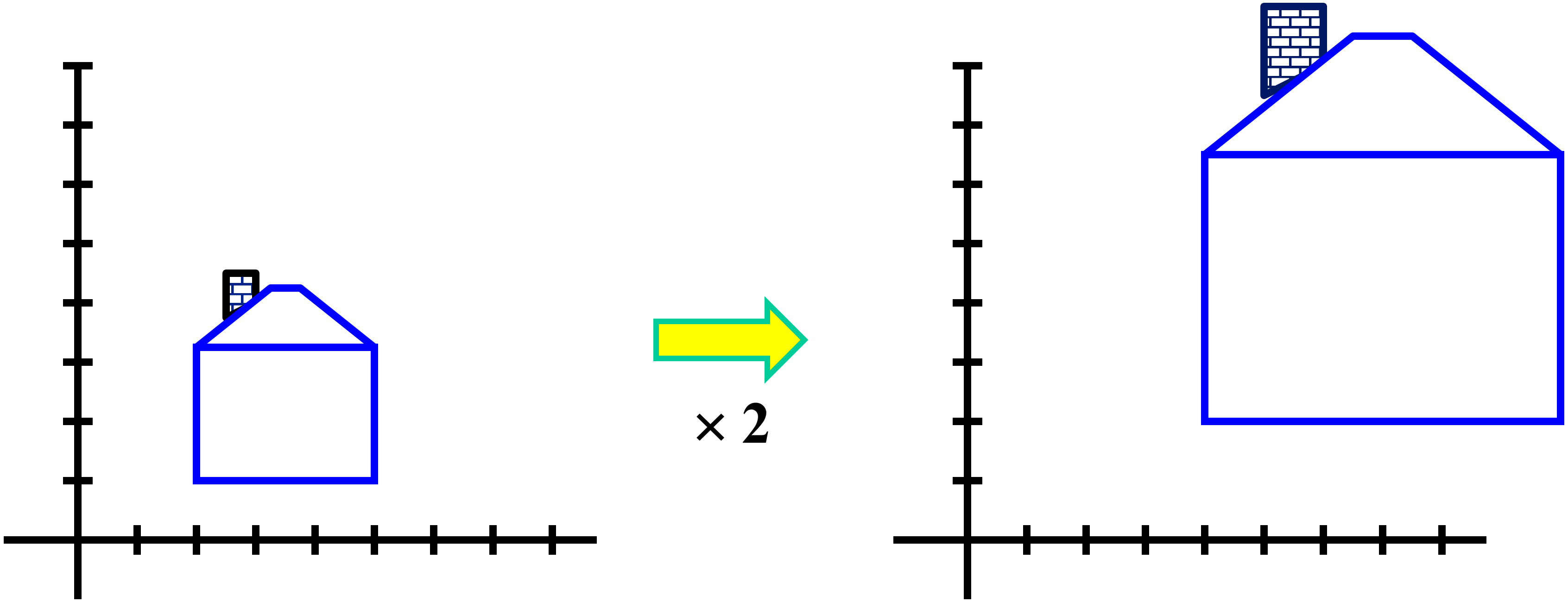
perspective

Slide from A. Efros, S. Seitz, D. Hoiem

Scaling

Scaling a coordinate means multiplying each of its components by a scalar

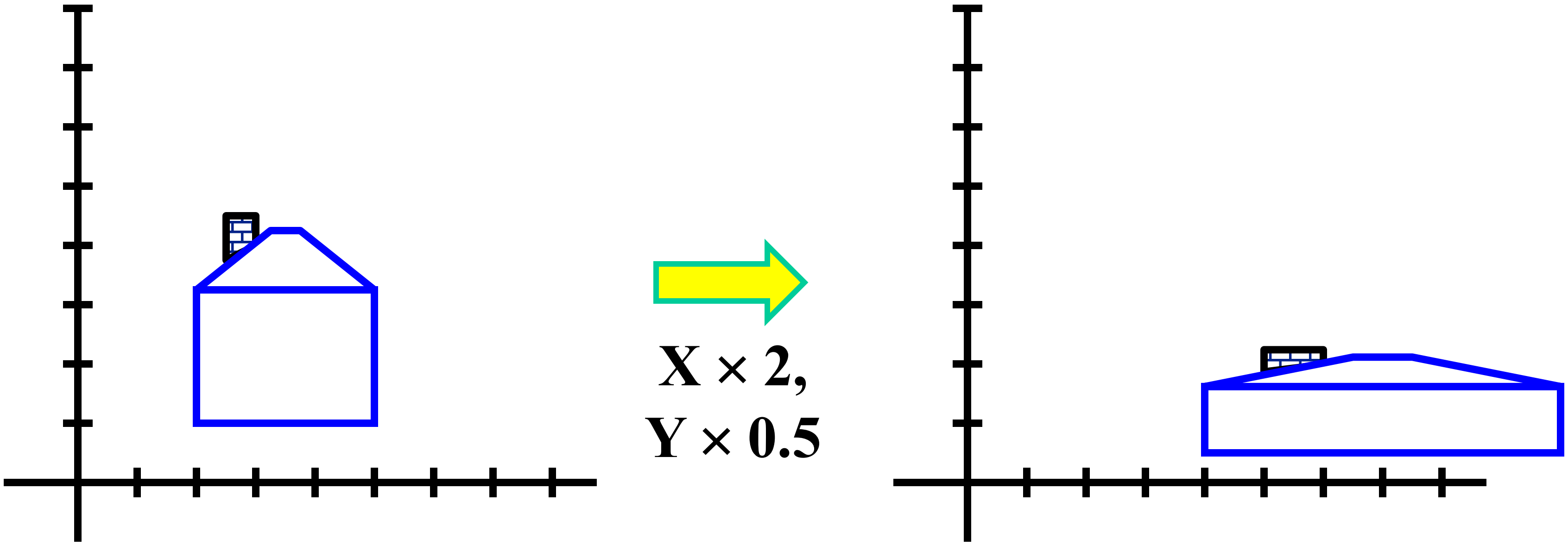
Uniform scaling means this scalar is the same for all components:



Slide from A. Efros, S. Seitz, D. Hoiem

Scaling

Non-uniform scaling: different scalars per component:



Slide from A. Efros, S. Seitz, D. Hoiem

Scaling

Scaling operation:

$$x' = ax$$

$$y' = by$$

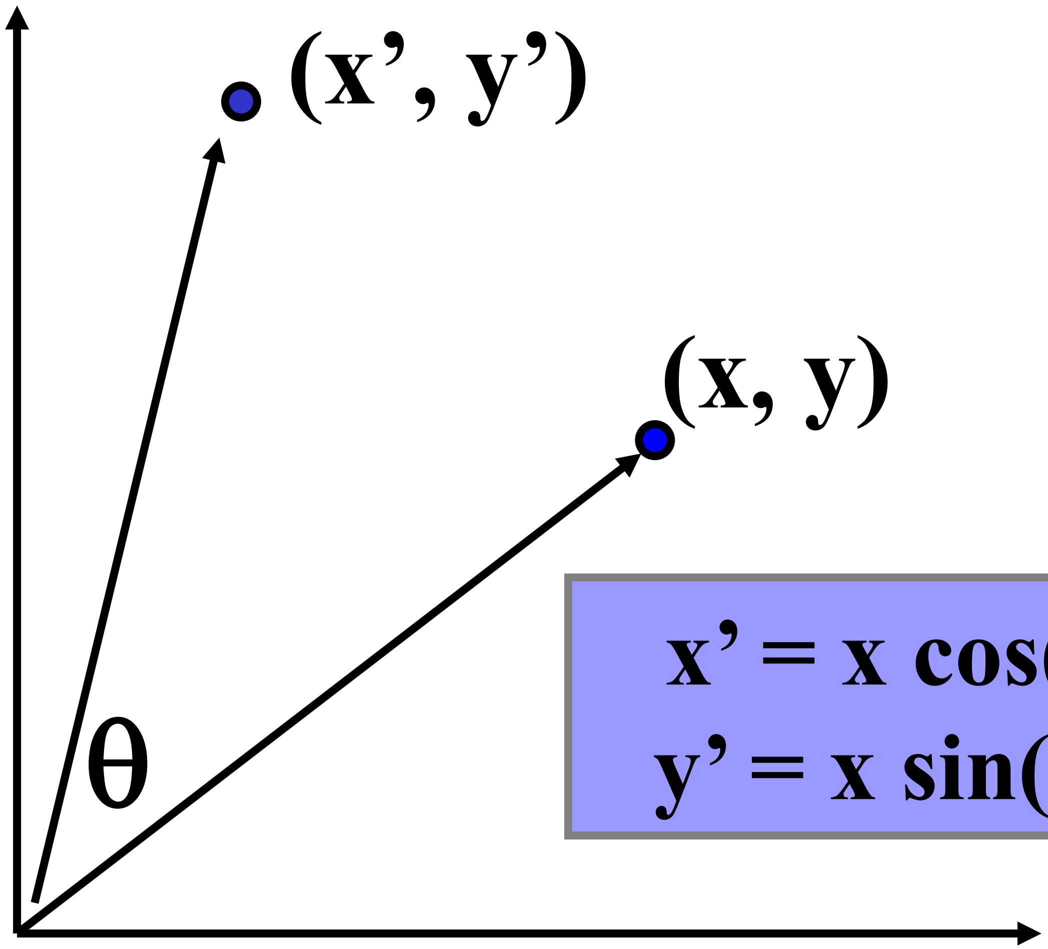
Or, in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\textit{scaling matrix S}} \begin{bmatrix} x \\ y \end{bmatrix}$$

scaling matrix S

Slide from A. Efros, S. Seitz, D. Hoiem

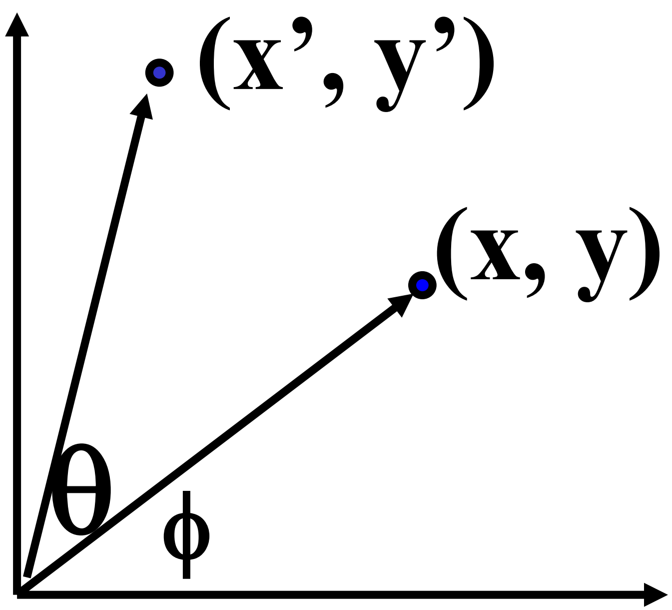
2-D Rotation



$$x' = x\cos(\theta) - y\sin(\theta)$$
$$y' = x\sin(\theta) + y\cos(\theta)$$

Slide from A. Efros, S. Seitz, D. Hoiem

2-D Rotation



Polar coordinates…

$x = r \cos(\phi)$
$y = r \sin(\phi)$
$x' = r \cos(\phi + \theta)$
$y' = r \sin(\phi + \theta)$

Trig Identity…

$x' = r \cos(\phi) \cos(\theta) - r \sin(\phi) \sin(\theta)$
$y' = r \sin(\phi) \cos(\theta) + r \cos(\phi) \sin(\theta)$

Substitute…

$x' = x \cos(\theta) - y \sin(\theta)$
$y' = x \sin(\theta) + y \cos(\theta)$

Slide from A. Efros, S. Seitz, D. Hoiem

2-D Rotation

This is easy to capture in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}}_{\mathbf{R}} \begin{bmatrix} x \\ y \end{bmatrix}$$

Even though sin(θ) and cos(θ) are nonlinear functions of θ,

- ***x' is a linear combination of x and y***
- ***y' is a linear combination of x and y***

What is the inverse transformation?

- Rotation by –θ
- For rotation matrices $\mathbf{R}^{-1} = \mathbf{R}^{T}$

Slide from A. Efros, S. Seitz, D. Hoiem

Basic 2D transformations

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Scale

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & \alpha_x \\ \alpha_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Shear

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotate

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Translate

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Affine

Affine is any combination of translation, scale, rotation, shear

Slide from A. Efros, S. Seitz, D. Hoiem

Affine Transformations

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

or

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Affine transformations are combinations of

- Linear transformations, and
- Translations

Properties of affine transformations:

- Lines map to lines
- Parallel lines remain parallel
- Ratios are preserved
- Closed under composition

Slide from L. Lazebnik.

Projective Transformations

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

Projective transformations are combos of

- Affine transformations, and
- Projective warps

Properties of projective transformations:

- Lines map to lines
- Parallel lines do not necessarily remain parallel
- Ratios are not preserved
- Closed under composition
- Models change of basis
- **Projective matrix is defined up to a scale (8 DOF)**

Slide from L. Lazebnik.

Homography

Definition

- General mathematics:

  homography = projective linear transformation

- Vision (most common usage):

  homography = linear transformation between two image planes

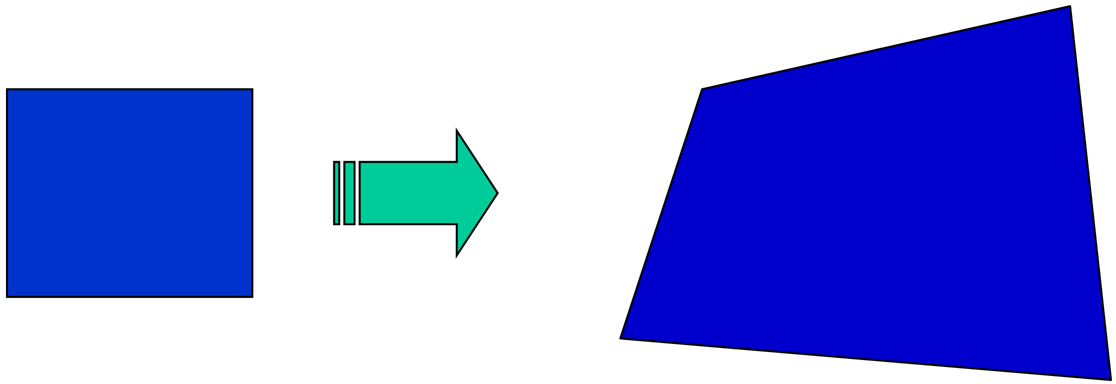
Examples

- Project 3D surface into frontal view
- Relate two views that differ only by rotation

Slide from A. Efros, S. Seitz, D. Hoiem

Fitting a plane projective transformation

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



Slide from L. Lazebnik.

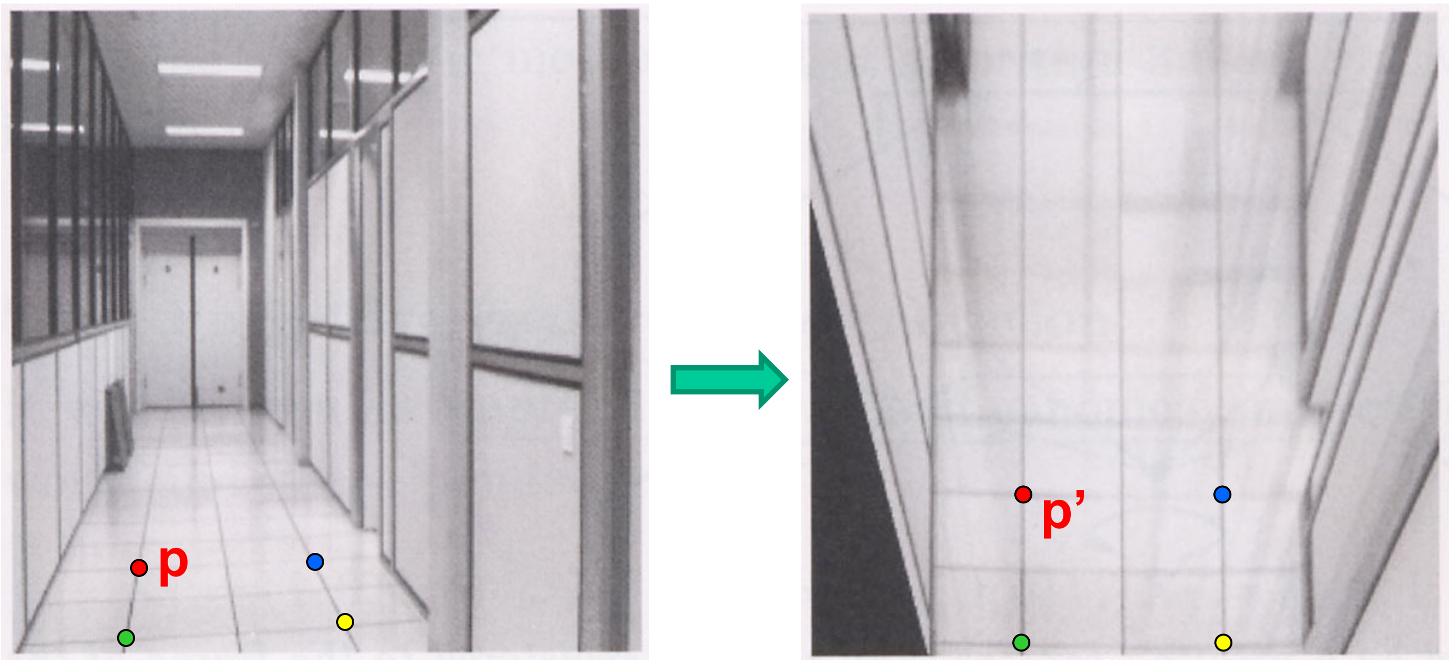
Homography

- The transformation between two views of a planar surface


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

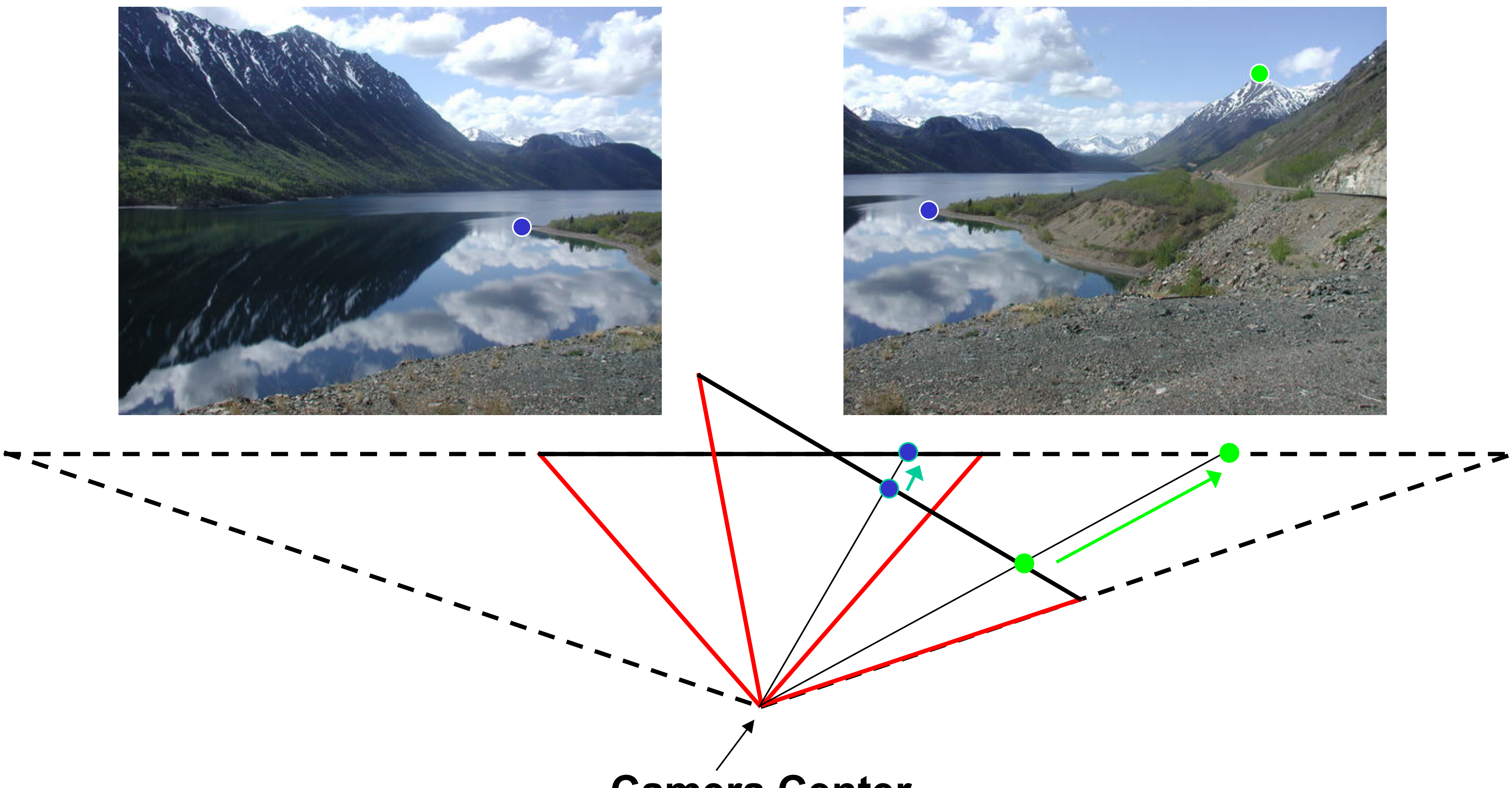

Slide from L. Lazebnik.

Homography example: Image rectification



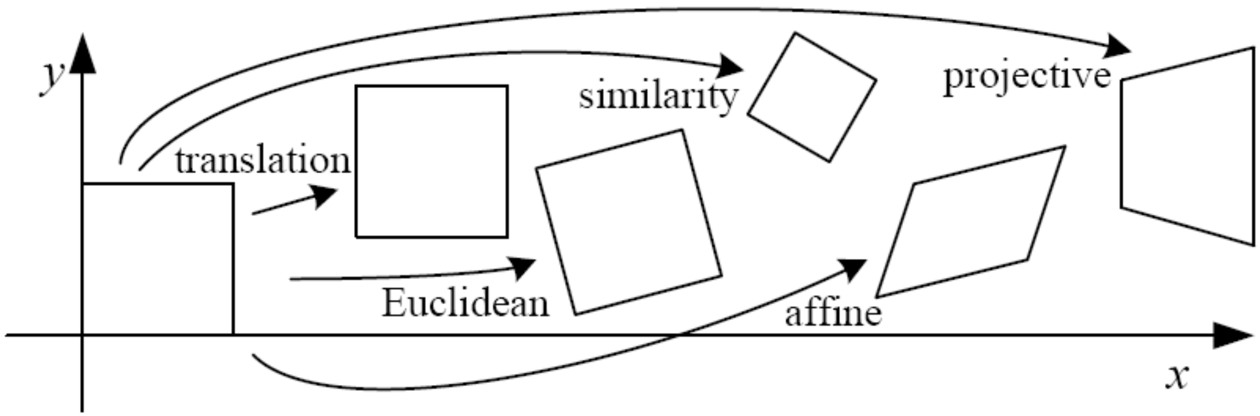
Slide from A. Efros, S. Seitz, D. Hoiem

Homography Example



Slide from A. Efros, S. Seitz, D. Hoiem

2D image transformations

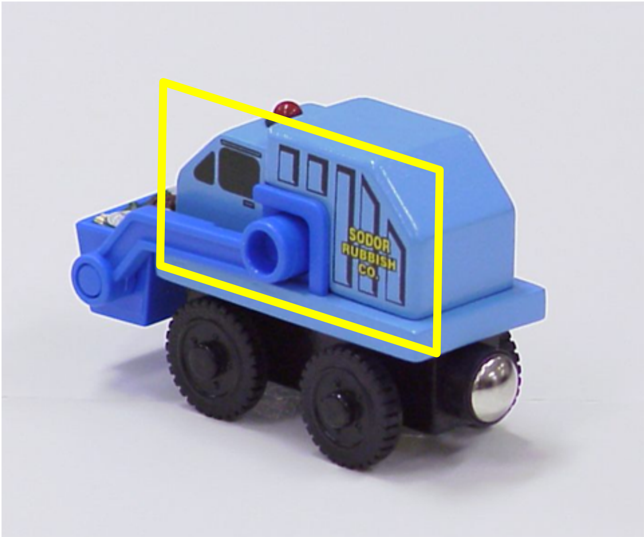


Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2	orientation $+ \cdots$	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3	lengths $+ \cdots$	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4	angles $+ \cdots$	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6	parallelism $+ \cdots$	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times 3}$	8	straight lines	

Slide from R. Szeliski

Let’s start with affine transformations

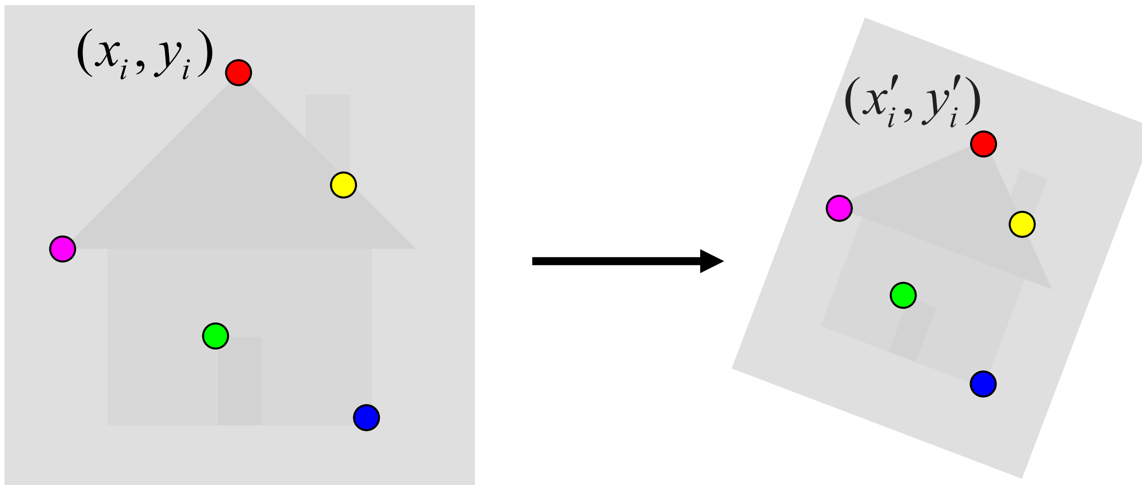
- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras
- Can be used to initialize fitting for more complex models



Slide from L. Lazebnik.

Fitting an affine transformation

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



$$\mathbf{x}_i' = \mathbf{M}\mathbf{x}_i + \mathbf{t}$$

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

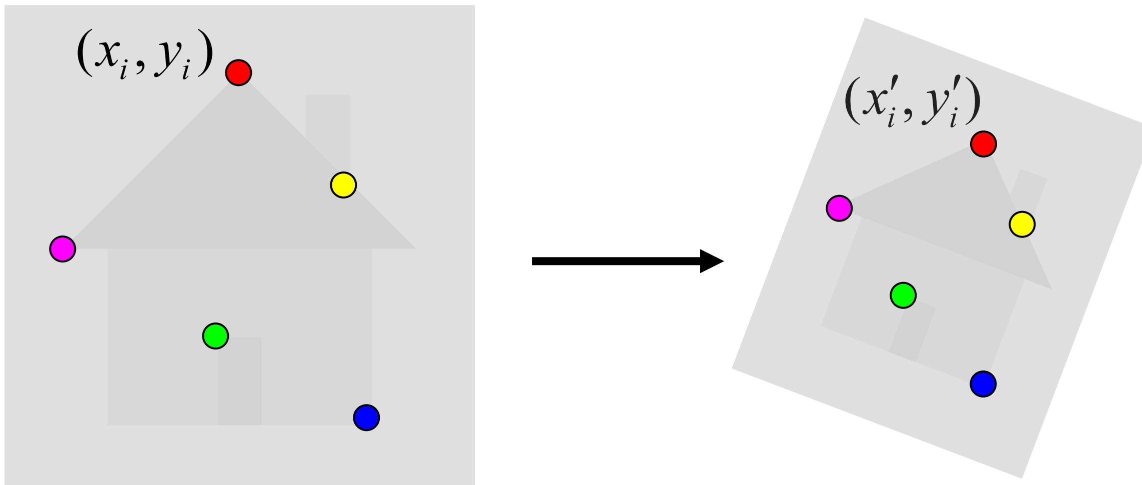
Want to find **M**, **t** to minimize

$$\sum_{i=1}^{n} \| \mathbf{x}_i' - \mathbf{M}\mathbf{x}_i - \mathbf{t} \|^2$$

Slide from L. Lazebnik.

Fitting an affine transformation

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



$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

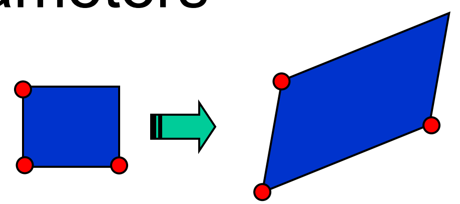
$$\begin{bmatrix} & \\ & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

Slide from L. Lazebnik.

Fitting an affine transformation

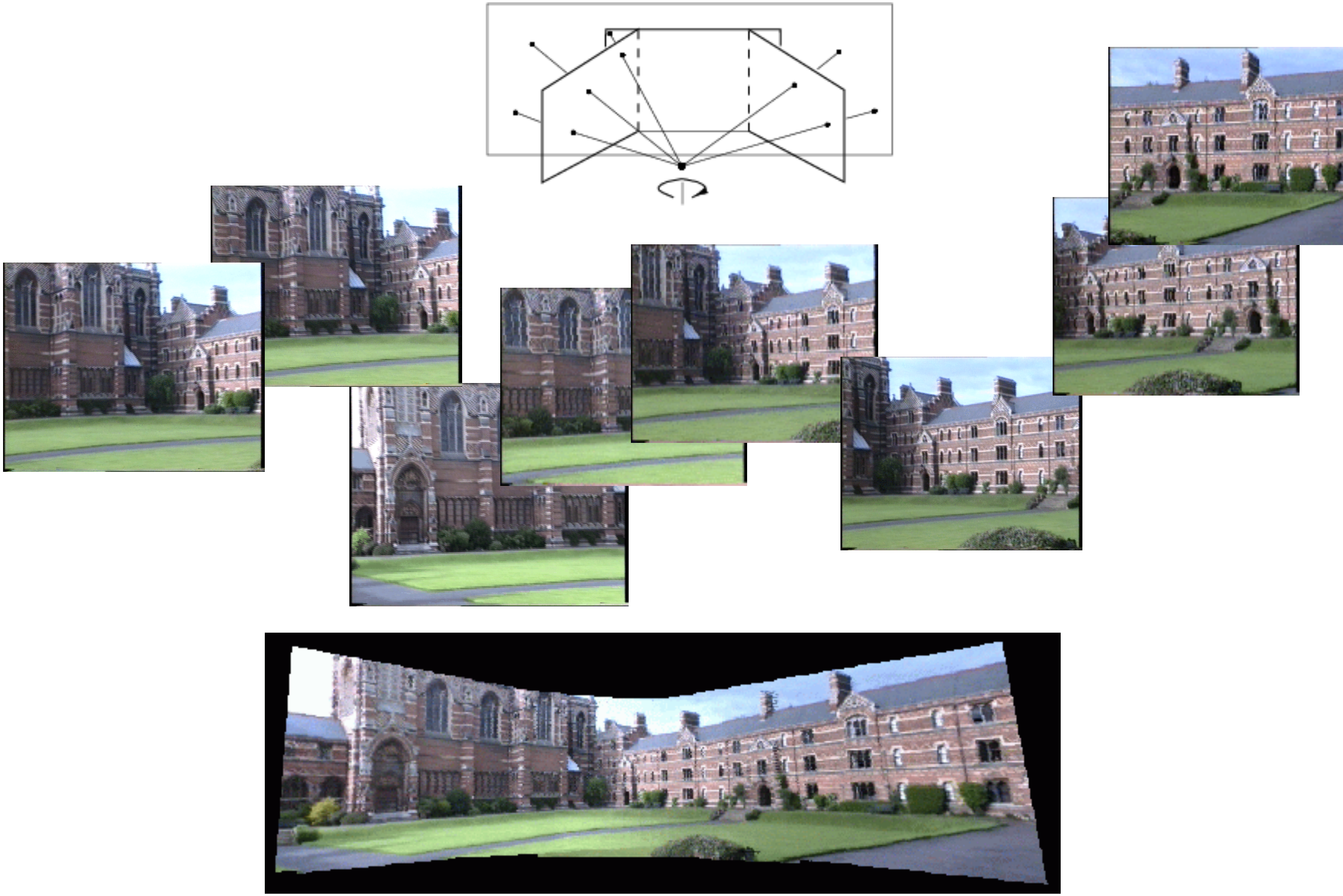
$$\begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

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



Slide from L. Lazebnik.

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

Slide from L. Lazebnik.

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:

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

Slide from L. Lazebnik.

Fitting a homography

- Equation for homography:

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

$$\lambda \mathbf{x}_i' = \mathbf{H}\,\mathbf{x}_i$$

$$\mathbf{x}_i' \times \mathbf{H}\,\mathbf{x}_i = 0$$

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

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

3 equations, only 2 linearly independent

Slide from L. Lazebnik.

Fitting a homography

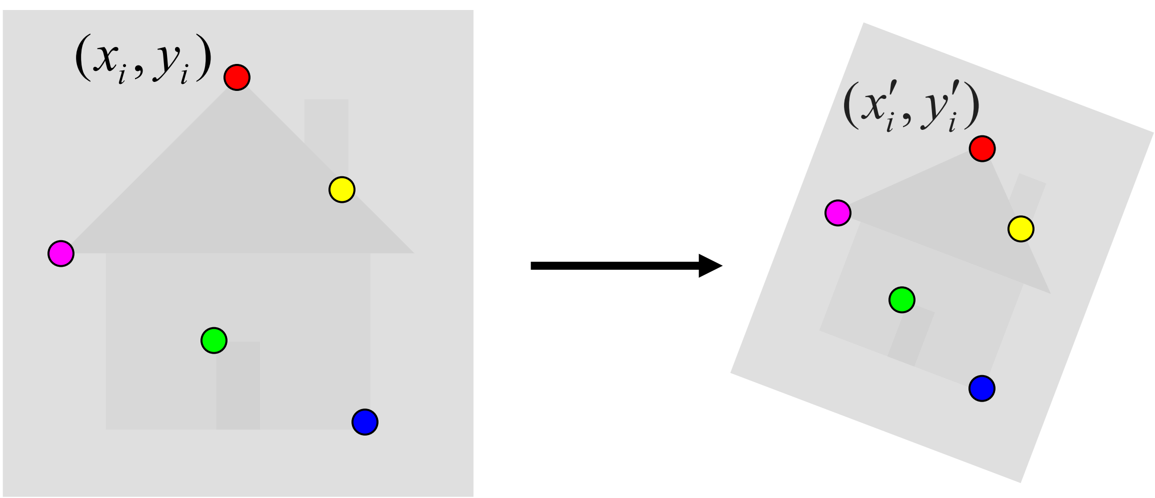
$$\begin{bmatrix} 0^T & \mathbf{x}_1^T & -y_1' \, \mathbf{x}_1^T \\ \mathbf{x}_1^T & 0^T & -x_1' \, \mathbf{x}_1^T \\ \cdots & \cdots & \cdots \\ 0^T & \mathbf{x}_n^T & -y_n' \, \mathbf{x}_n^T \\ \mathbf{x}_n^T & 0^T & -x_n' \, \mathbf{x}_n^T \end{bmatrix} \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix} = 0 \qquad \mathbf{A}\,\mathbf{h} = 0$$

- 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 $\|\mathbf{A}\mathbf{h}\|^2$
  - Eigenvector of $\mathbf{A}^\mathbf{T}\mathbf{A}$ corresponding to smallest eigenvalue
  - Four matches needed for a minimal solution

Slide from L. Lazebnik.

Robust feature-based alignment

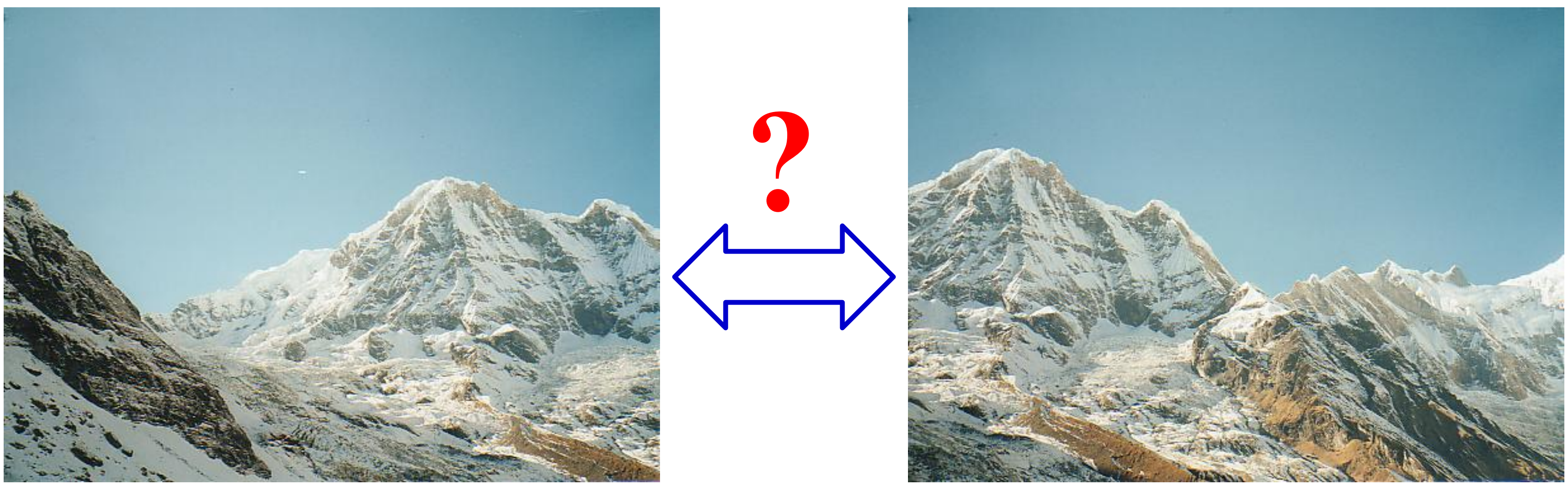
- So far, we've assumed that we are given a set of "ground-truth" correspondences between the two images we want to align
- What if we don't know the correspondences?



Slide from L. Lazebnik.

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?



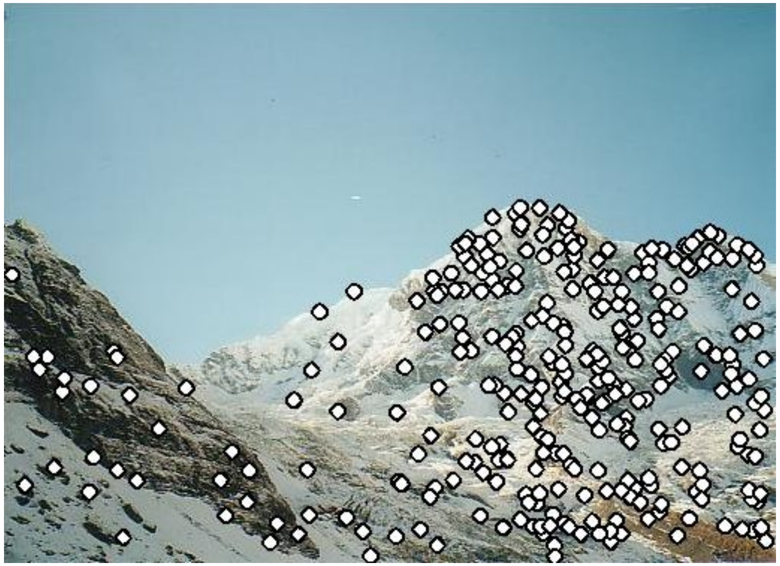
Slide from L. Lazebnik.

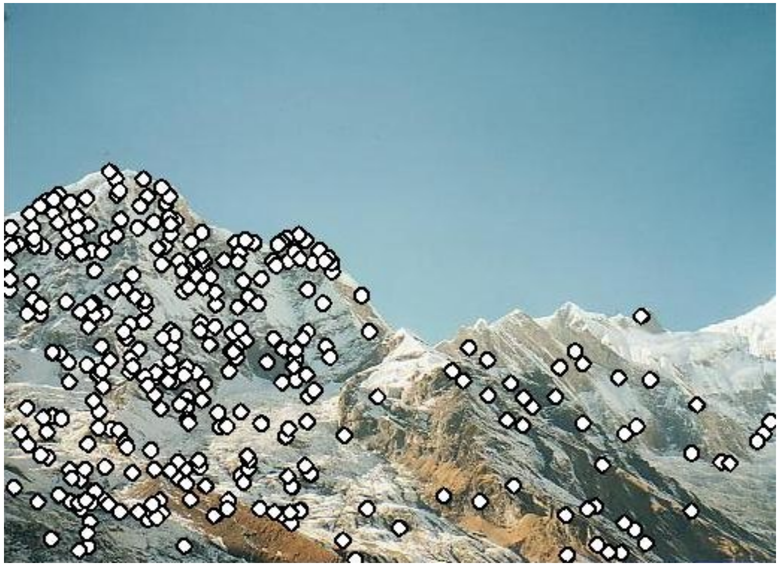
Robust feature-based alignment

Slide from L. Lazebnik.

Robust feature-based alignment





- Extract features

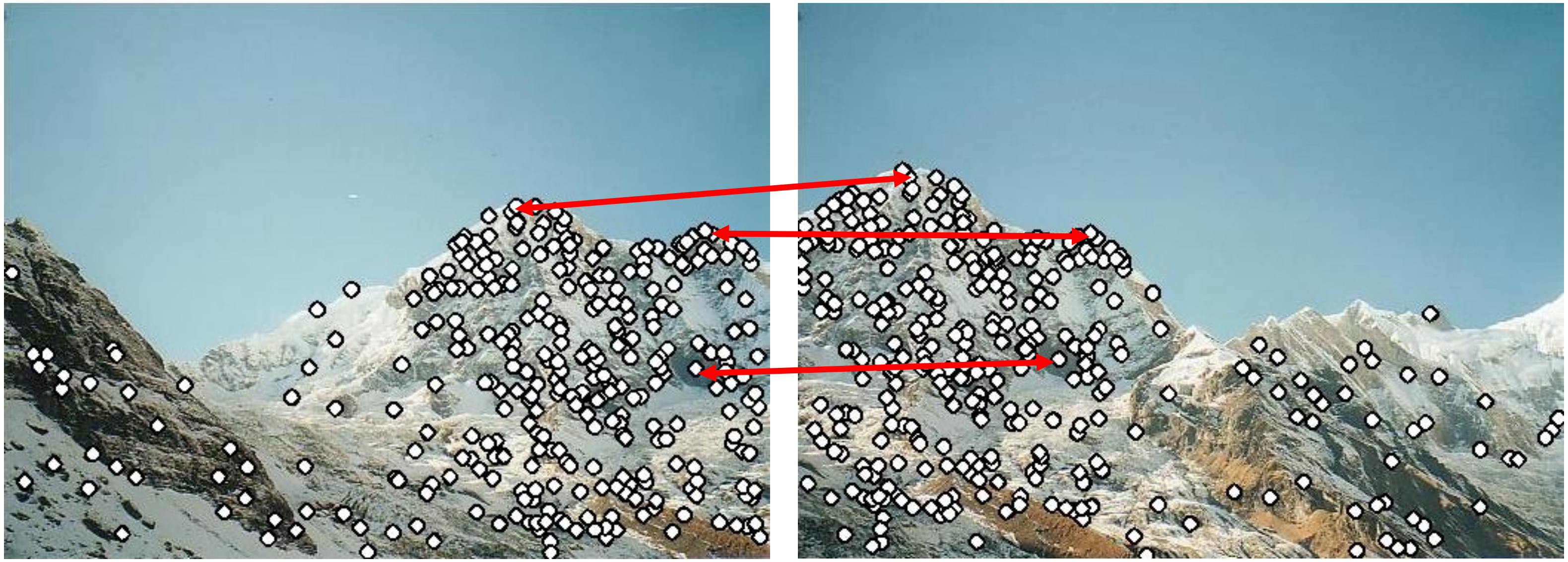
Slide from L. Lazebnik.

Robust feature-based alignment

- Extract features
- Compute *putative matches*

Slide from L. Lazebnik.

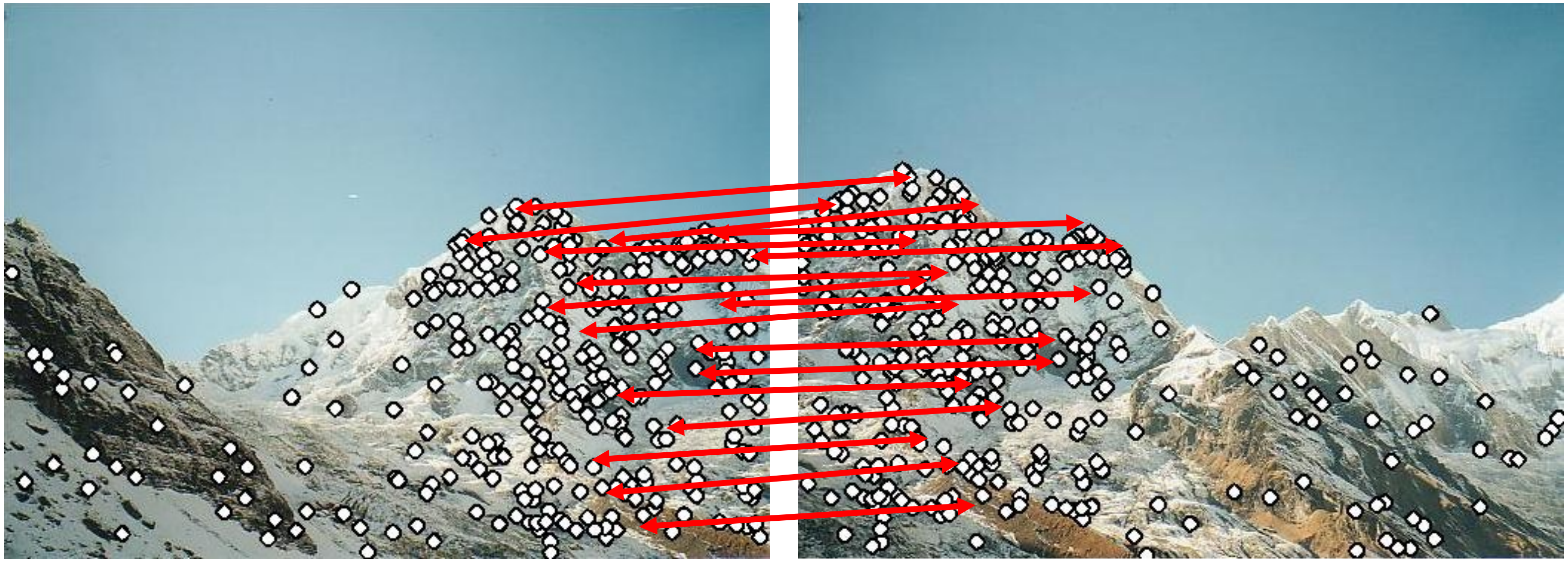
Robust feature-based alignment



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

Slide from L. Lazebnik.

Robust feature-based alignment



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

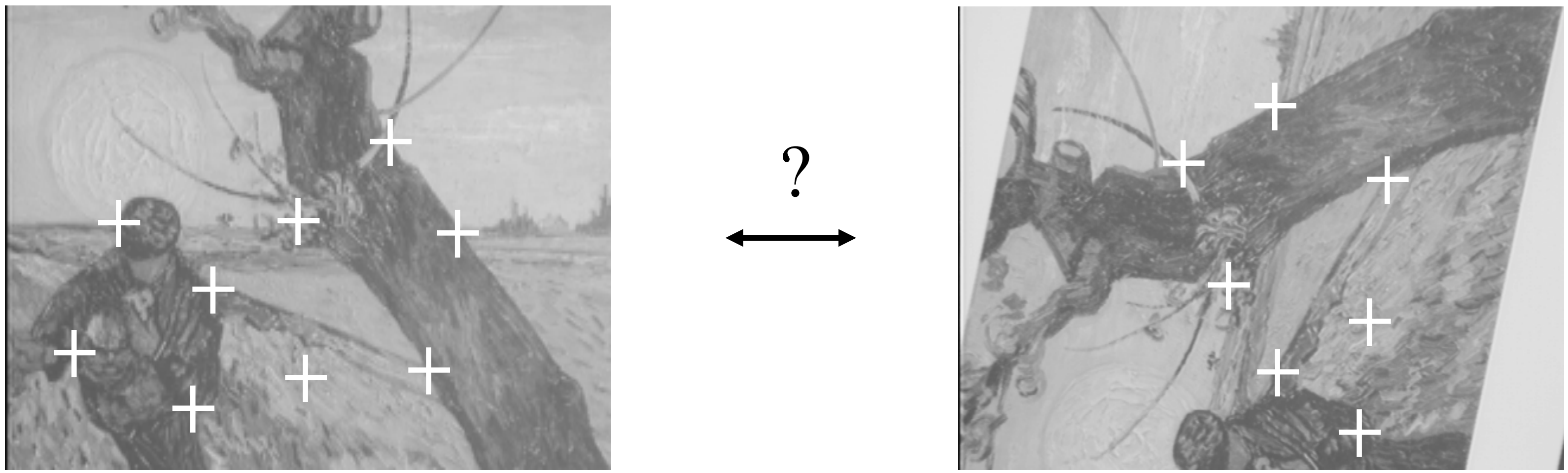
Slide from L. Lazebnik.

Robust feature-based alignment

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

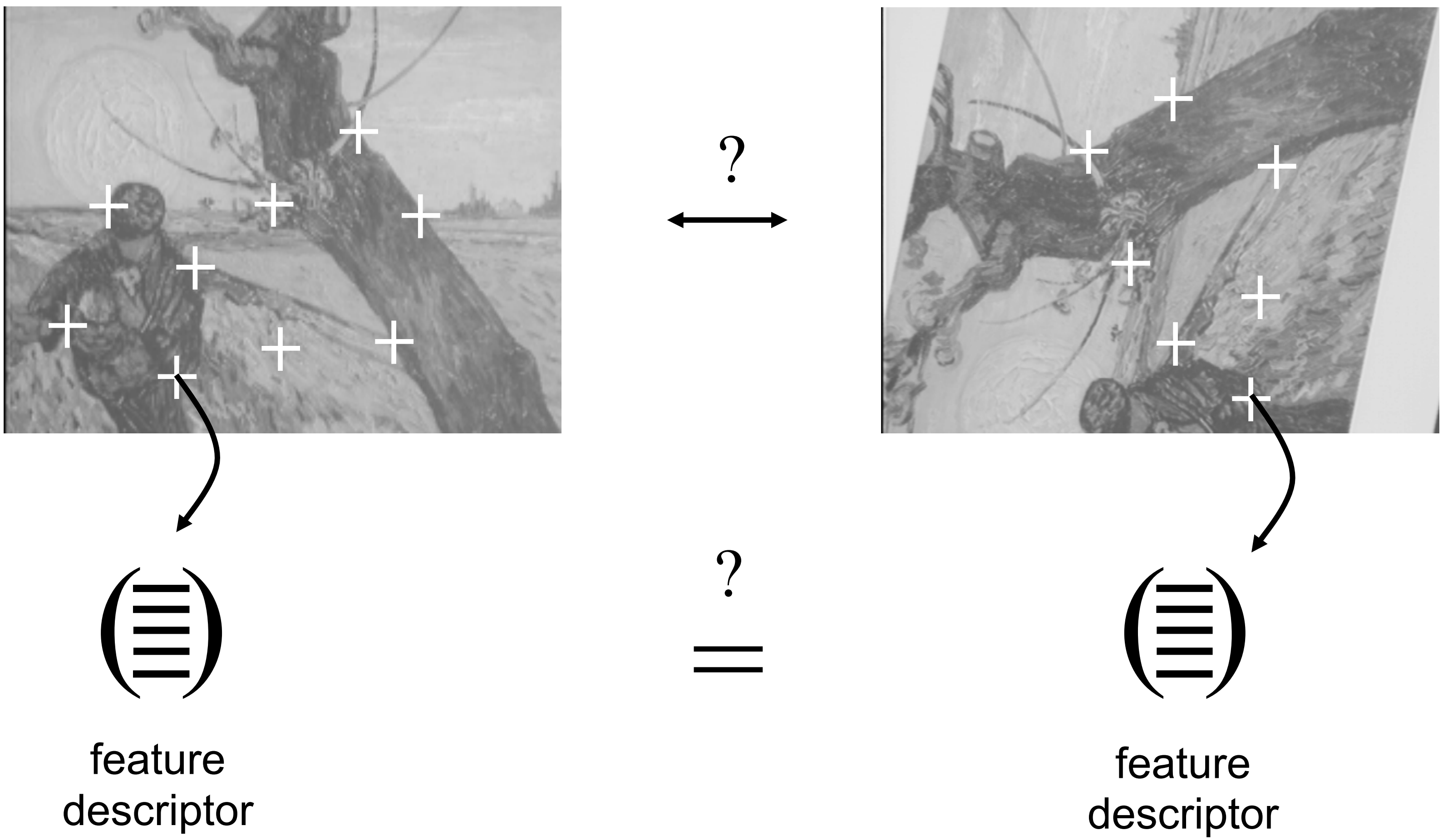
Slide from L. Lazebnik.

Generating putative correspondences



Slide from L. Lazebnik.

Generating putative correspondences

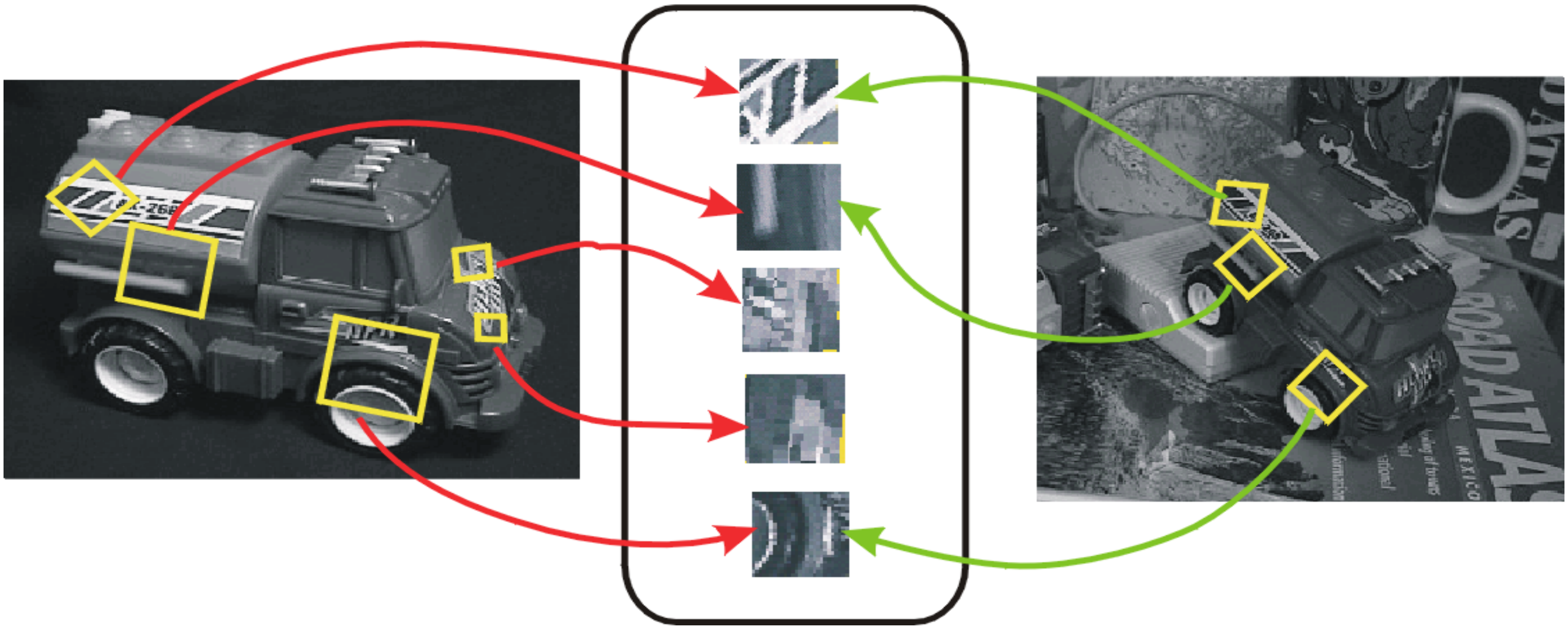


- Need to compare *feature descriptors* of local patches surrounding interest points

Slide from L. Lazebnik.

Feature descriptors

- Recall: feature detection and description



Slide from L. Lazebnik.

Feature descriptors

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

$$\mathrm{SSD}(\mathbf{u},\mathbf{v}) = \sum_i \left(u_i - v_i\right)^2$$

    – Not invariant to intensity change

  - Normalized correlation

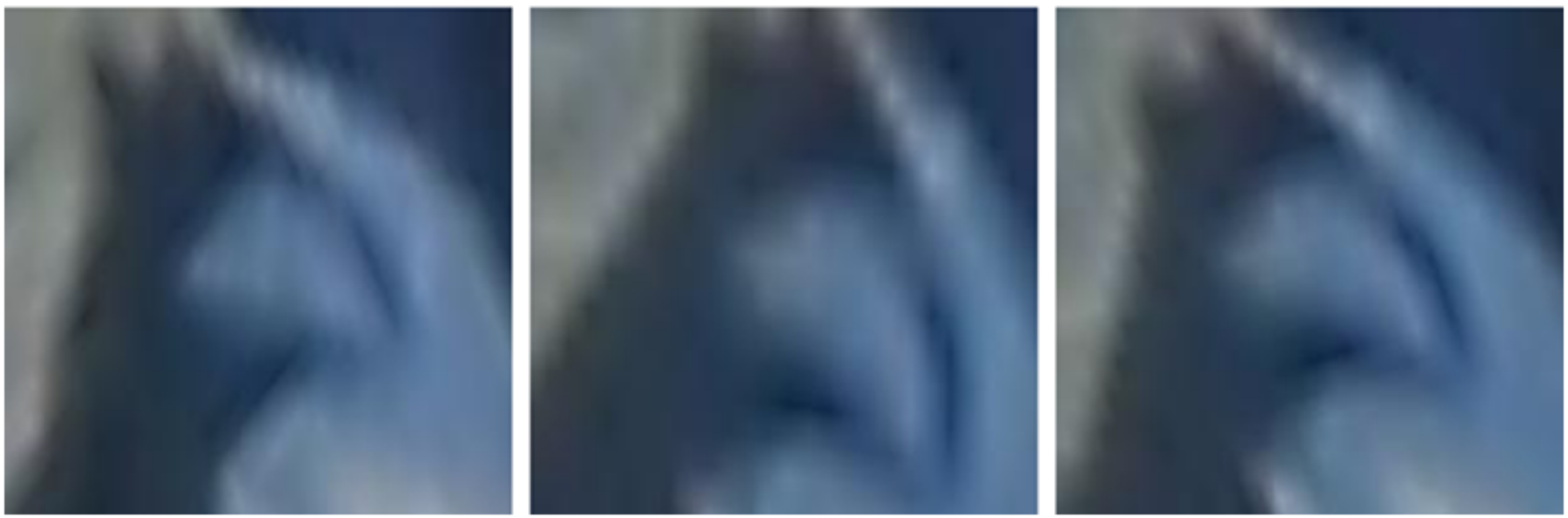
$$\rho(\mathbf{u},\mathbf{v}) = \frac{(\mathbf{u}-\overline{\mathbf{u}})}{\|\mathbf{u}-\overline{\mathbf{u}}\|} \cdot \frac{(\mathbf{v}-\overline{\mathbf{v}})}{\|\mathbf{v}-\overline{\mathbf{v}}\|} = \frac{\sum_i (u_i - \overline{\mathbf{u}})(v_i - \overline{\mathbf{v}})}{\sqrt{\left(\sum_j (u_j - \overline{\mathbf{u}})^2\right)\left(\sum_j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

    – Invariant to affine intensity change

Slide from L. Lazebnik.

Disadvantage of intensity vectors as descriptors

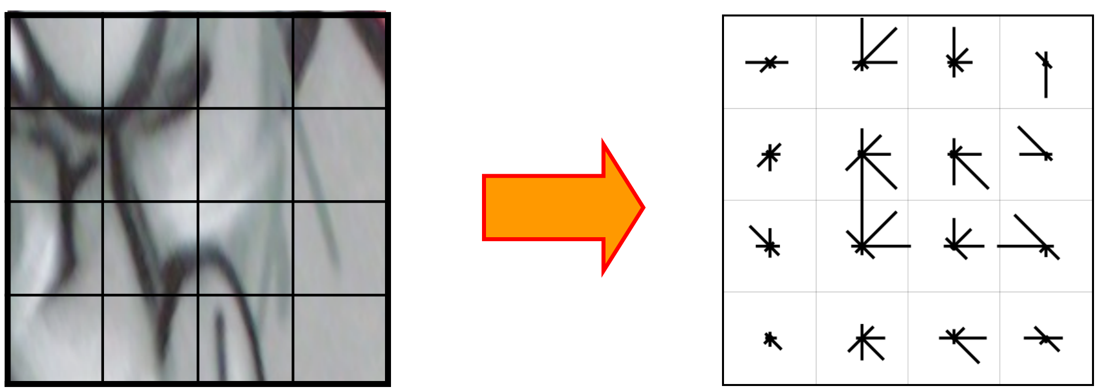
- Small deformations can affect the matching score a lot



Slide from L. Lazebnik.

Feature descriptors: SIFT

- Descriptor computation:
  - Divide patch into 4x4 sub-patches
  - Compute histogram of gradient orientations (8 reference angles) inside each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions



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

Slide from L. Lazebnik.

Feature descriptors: SIFT

- Descriptor computation:
  - Divide patch into 4x4 sub-patches
  - Compute histogram of gradient orientations (8 reference angles) inside each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions

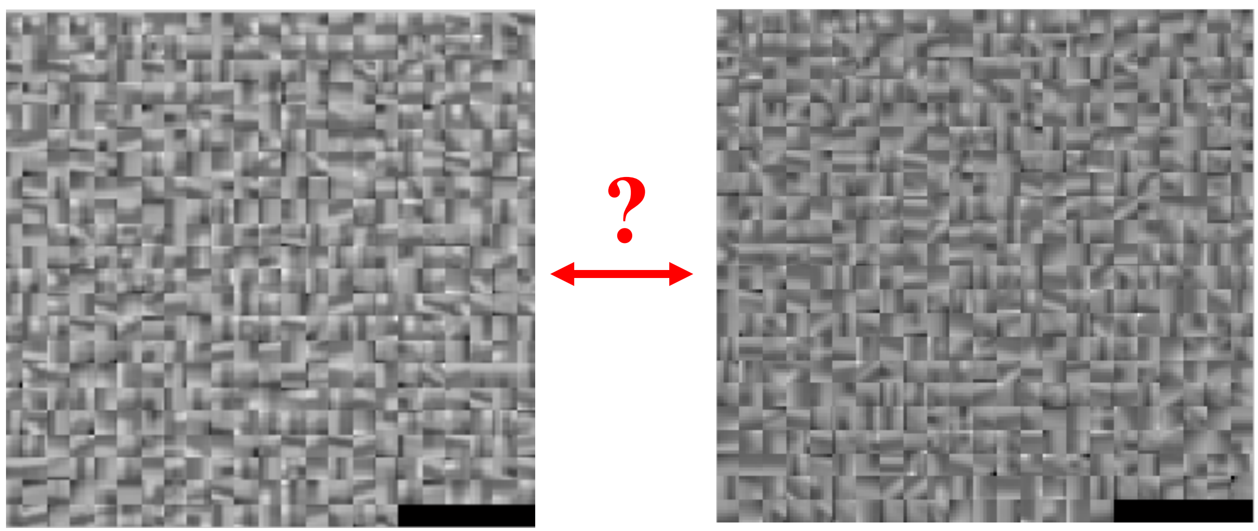
- Advantage over raw vectors of pixel values
  - Gradients less sensitive to illumination change
  - Pooling of gradients over the sub-patches achieves robustness to small shifts, but still preserves some spatial information

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

Slide from L. Lazebnik.

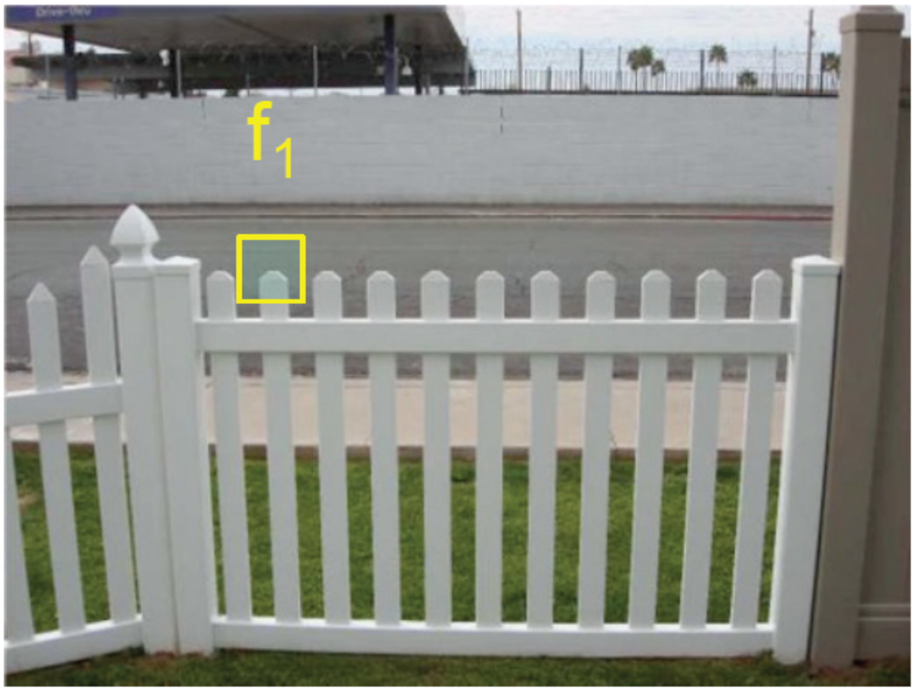
Feature matching

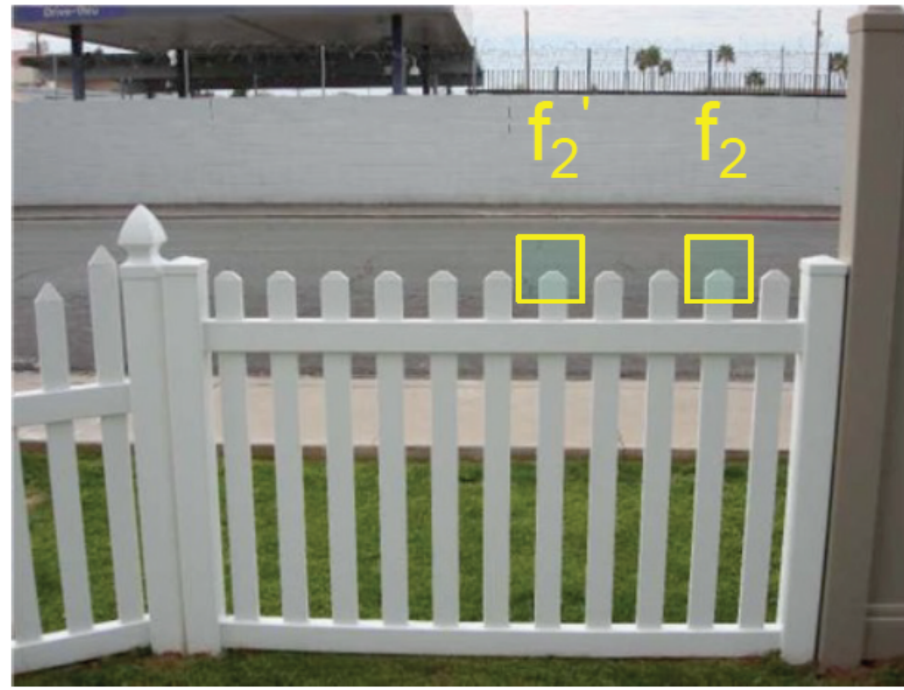
- Generating *putative matches*: for each patch in one image, find a short list of patches in the other image that could match it based solely on appearance



Slide from L. Lazebnik.

Problem: Ambiguous putative matches

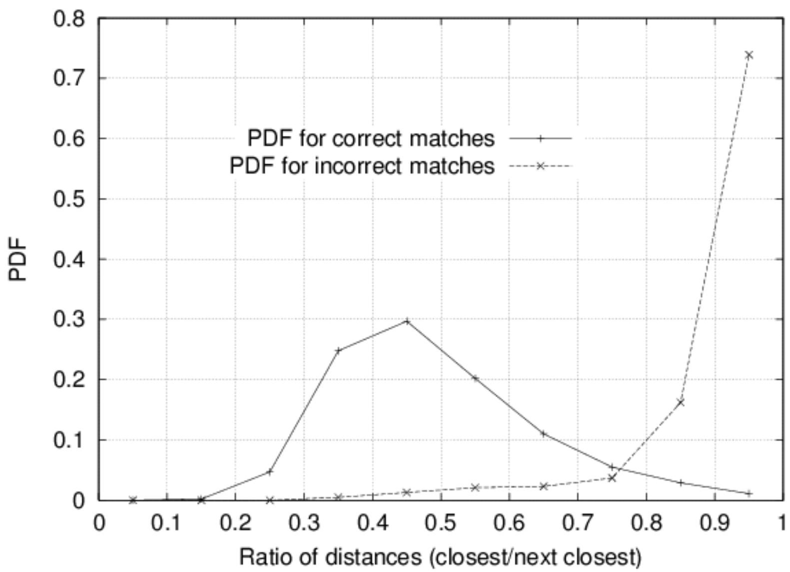




Source: Y. Furukawa

Rejection of unreliable matches

- How can we tell which putative matches are more reliable?
- Heuristic: compare distance of **nearest** neighbor to that of **second** nearest neighbor
  - Ratio of closest distance to second-closest distance will be *high* for features that are *not* distinctive



Threshold of 0.8 provides good separation

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

Slide from L. Lazebnik.

RANSAC

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

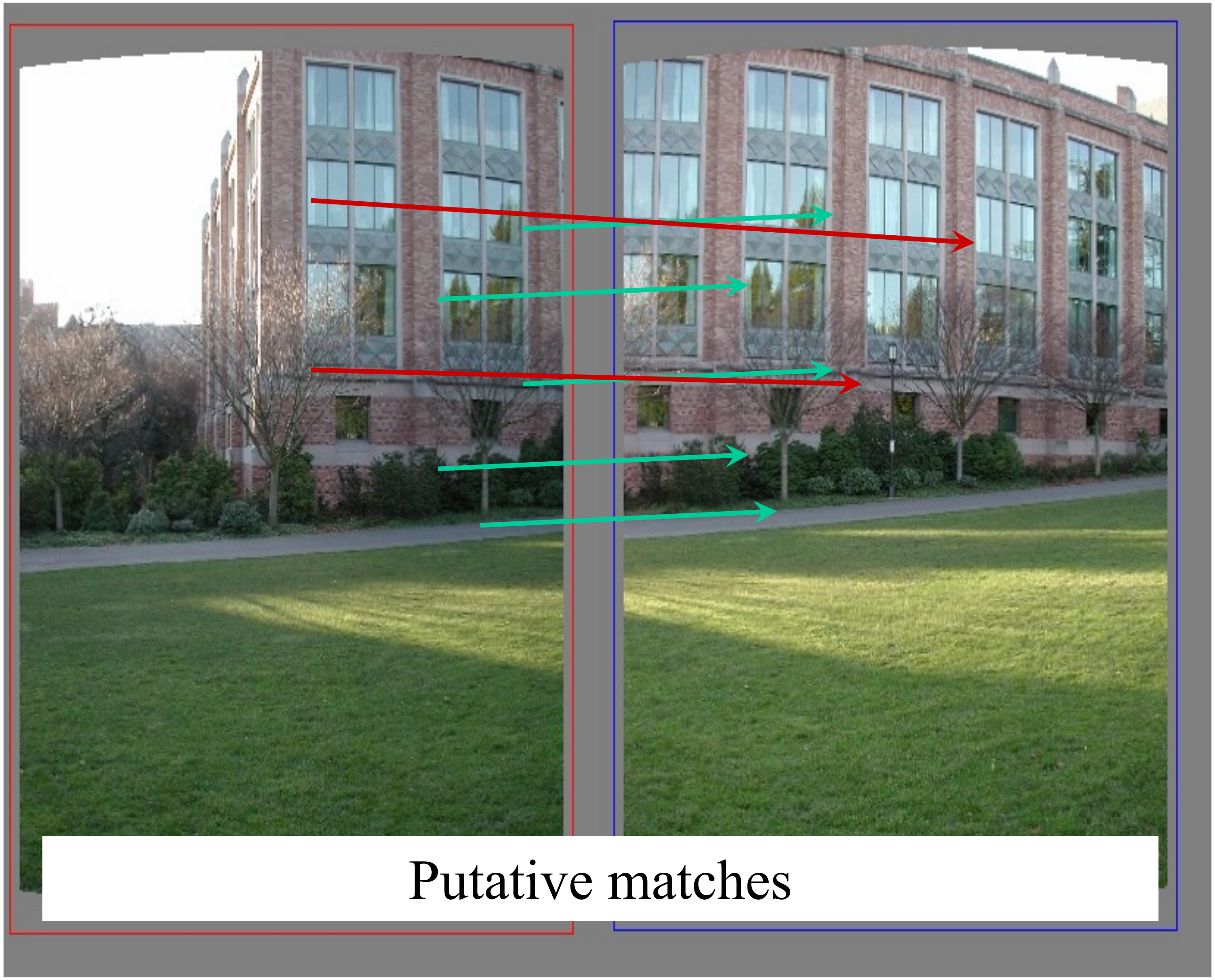
RANSAC loop:

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

Slide from L. Lazebnik.

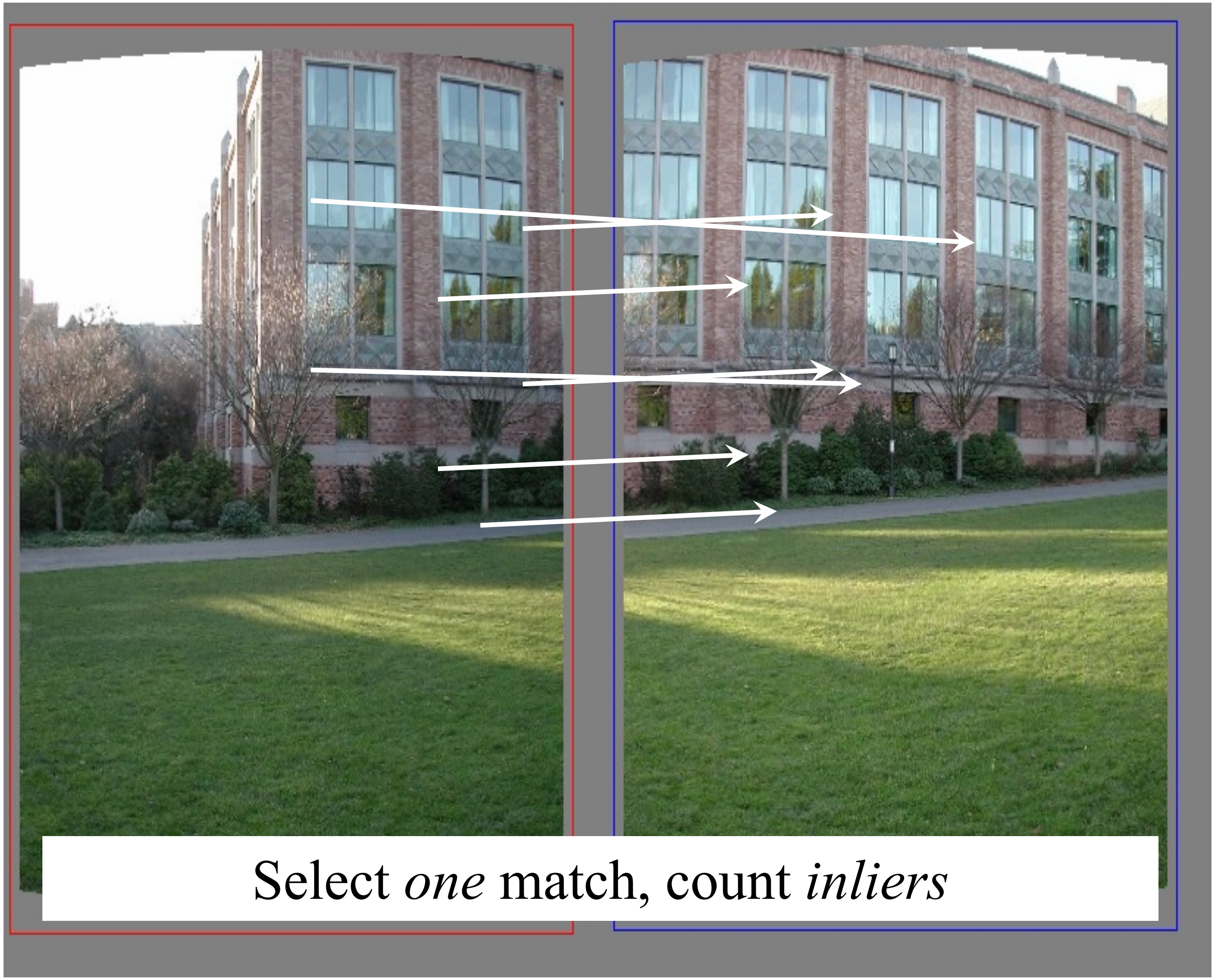
RANSAC example: Translation



Putative matches

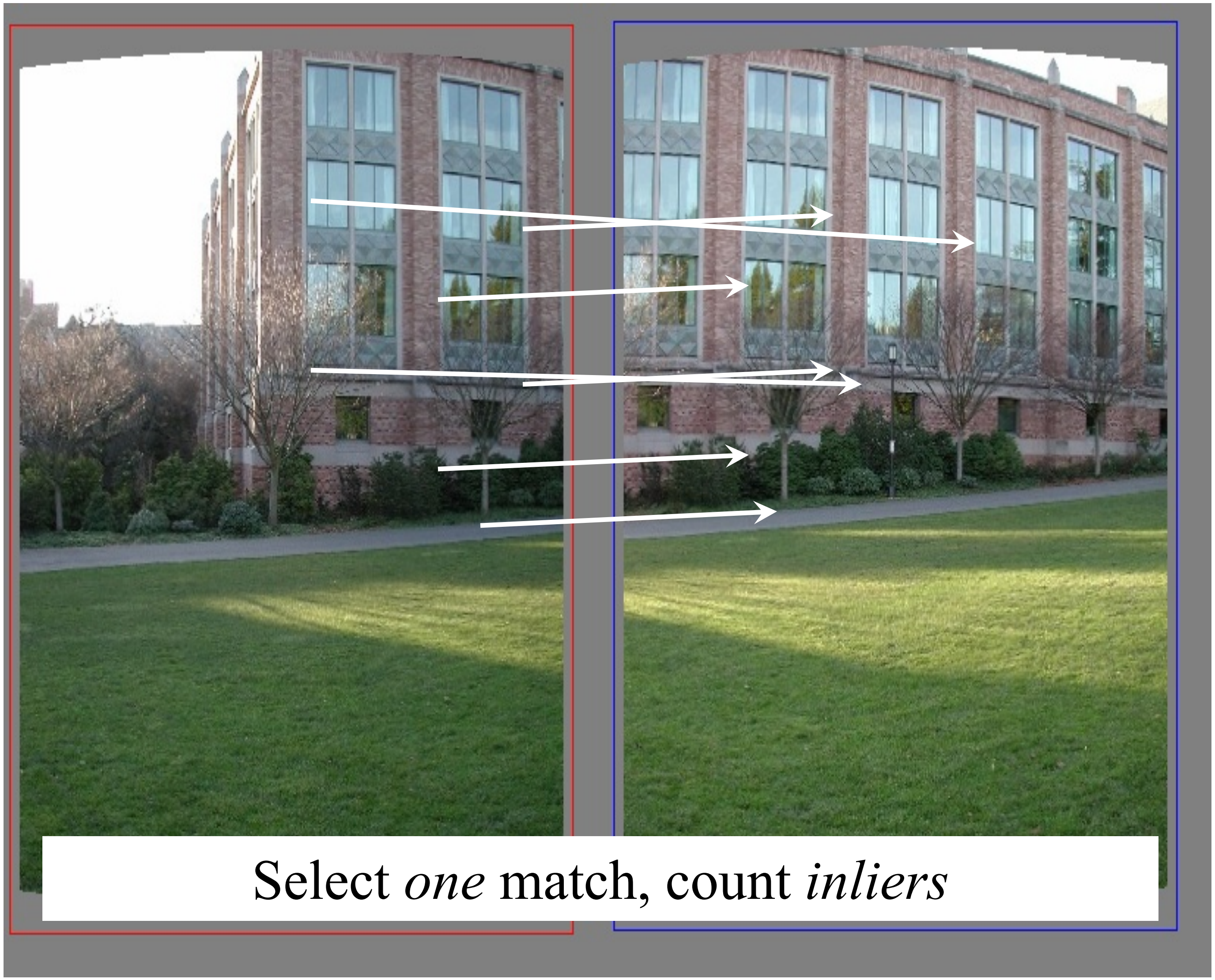
Slide from L. Lazebnik.

RANSAC example: Translation



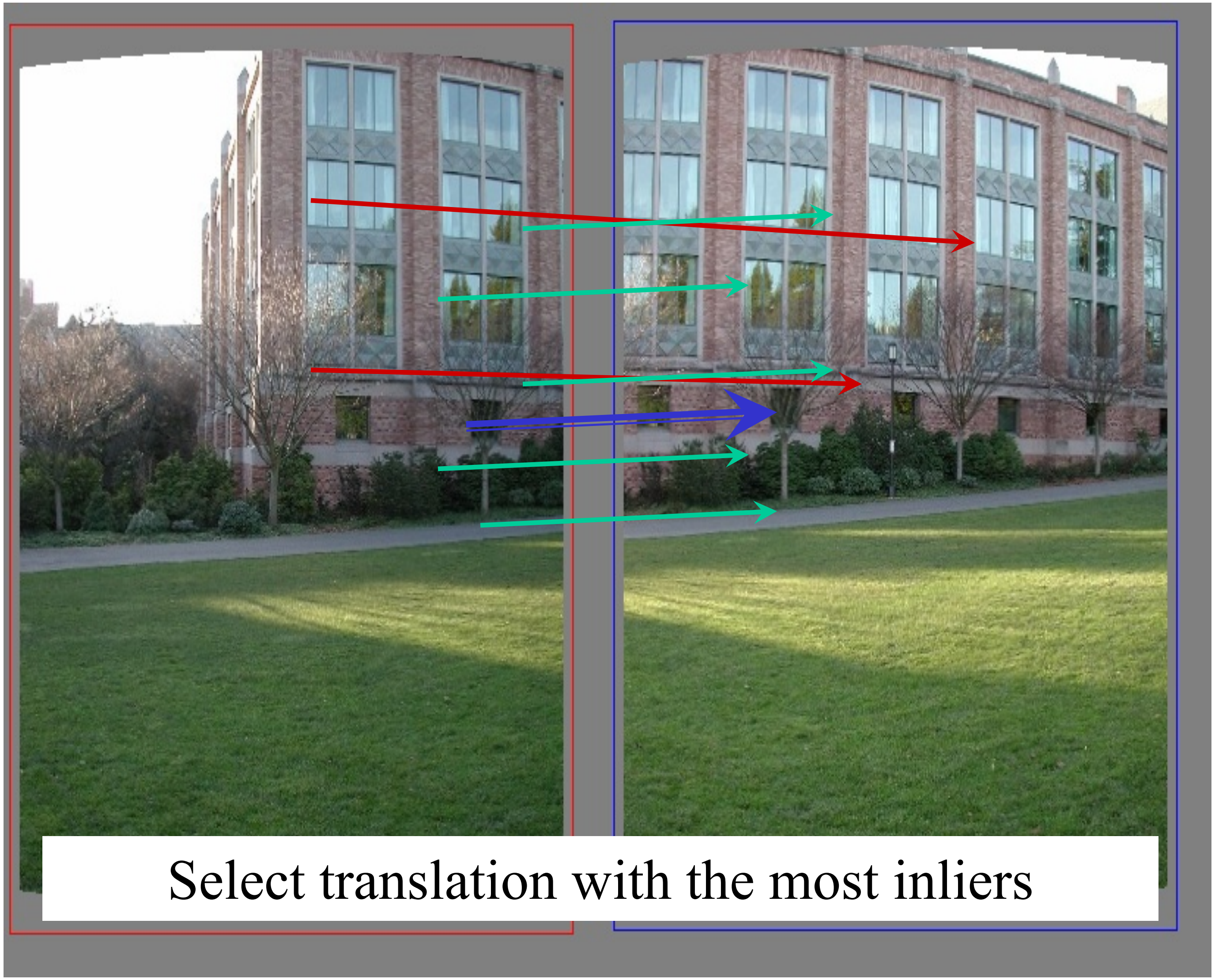
Slide from L. Lazebnik.

RANSAC example: Translation



Slide from L. Lazebnik.

RANSAC example: Translation



Slide from L. Lazebnik.

Feature-based alignment: Overview

- Alignment as fitting
  - Affine transformations
  - Homographies
- Robust alignment
  - Descriptor-based feature matching
  - RANSAC
- Applications

Slide from L. Lazebnik