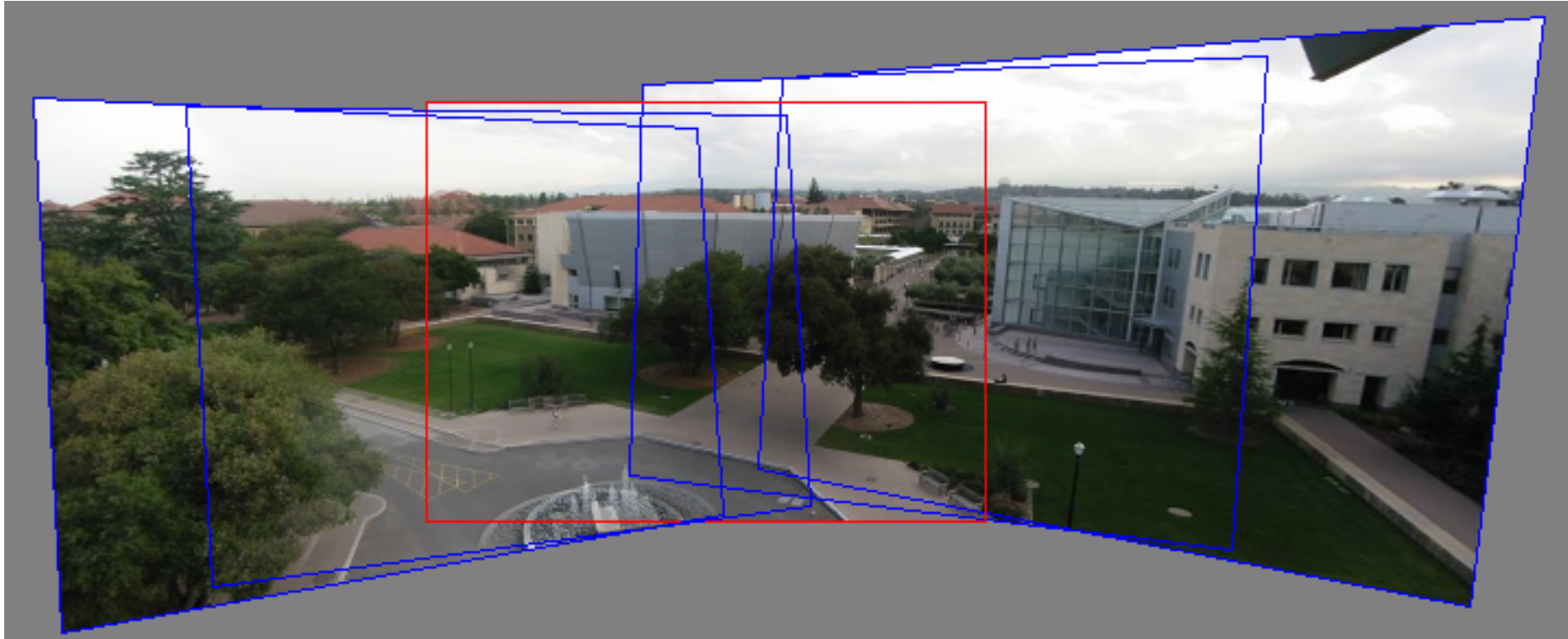
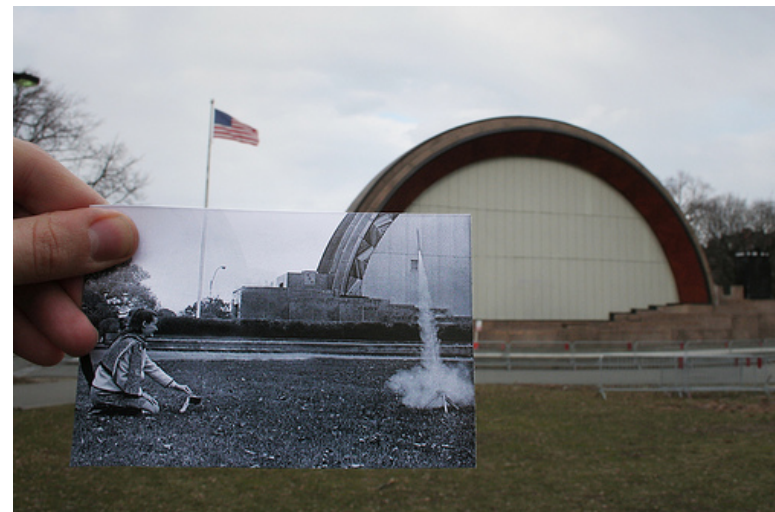
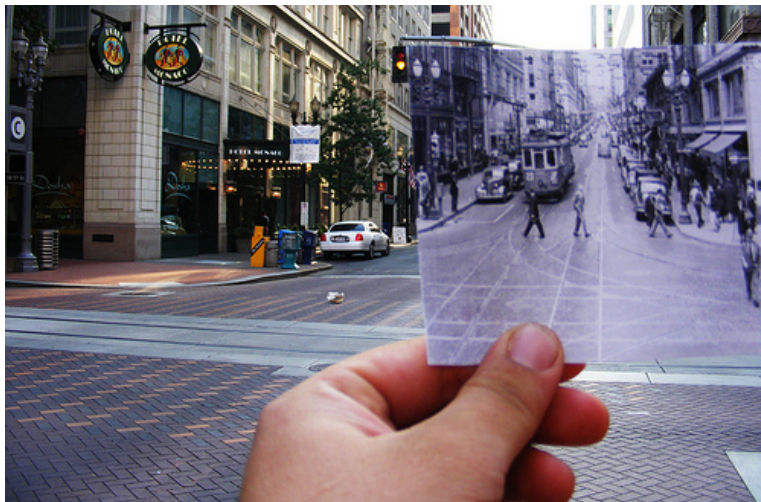
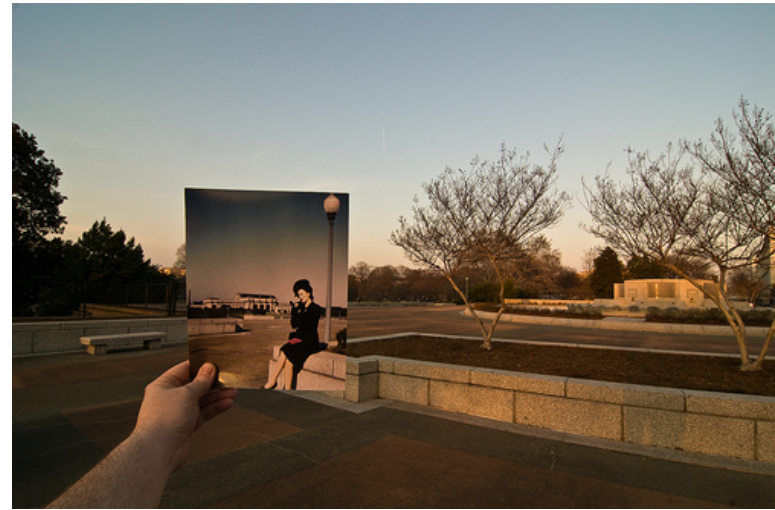


Image alignment



Alignment applications

- [A look into the past](#)



Alignment applications

- [A look into the past](#)



Alignment applications

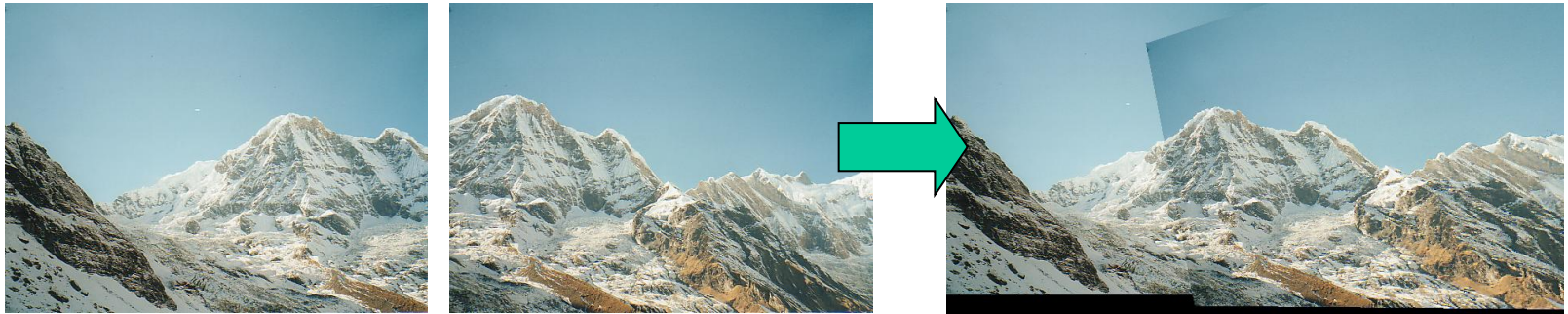
- [Cool video](#)

Alignment applications

Instance recognition



Alignment challenges



Small degree of overlap
Intensity changes



Occlusion,
clutter,
viewpoint
change

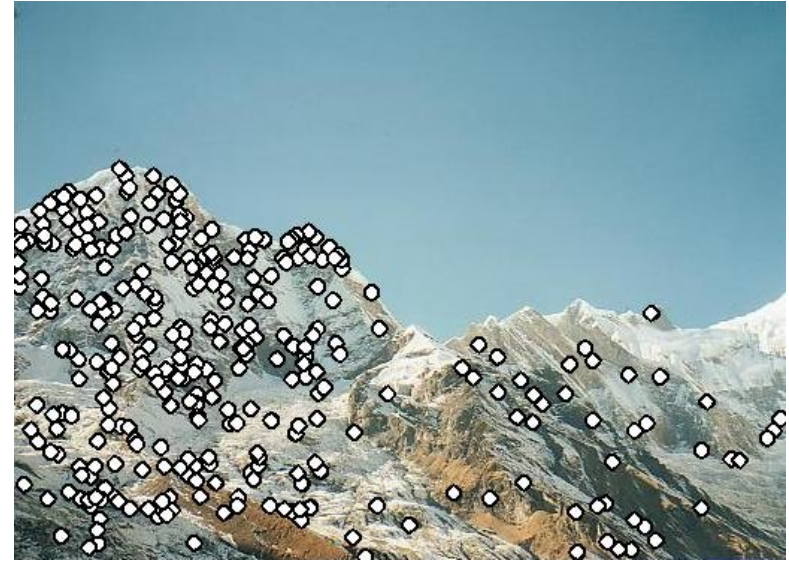
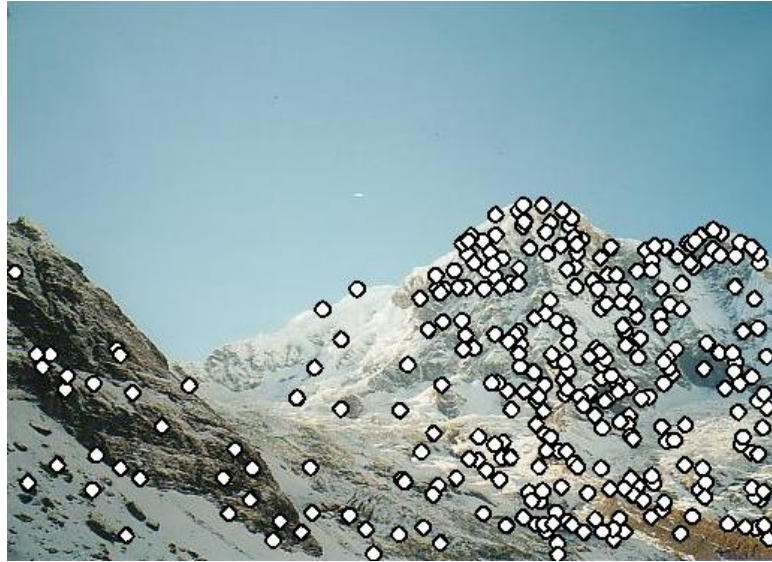
Image Alignment



Robust feature-based alignment



Robust feature-based alignment



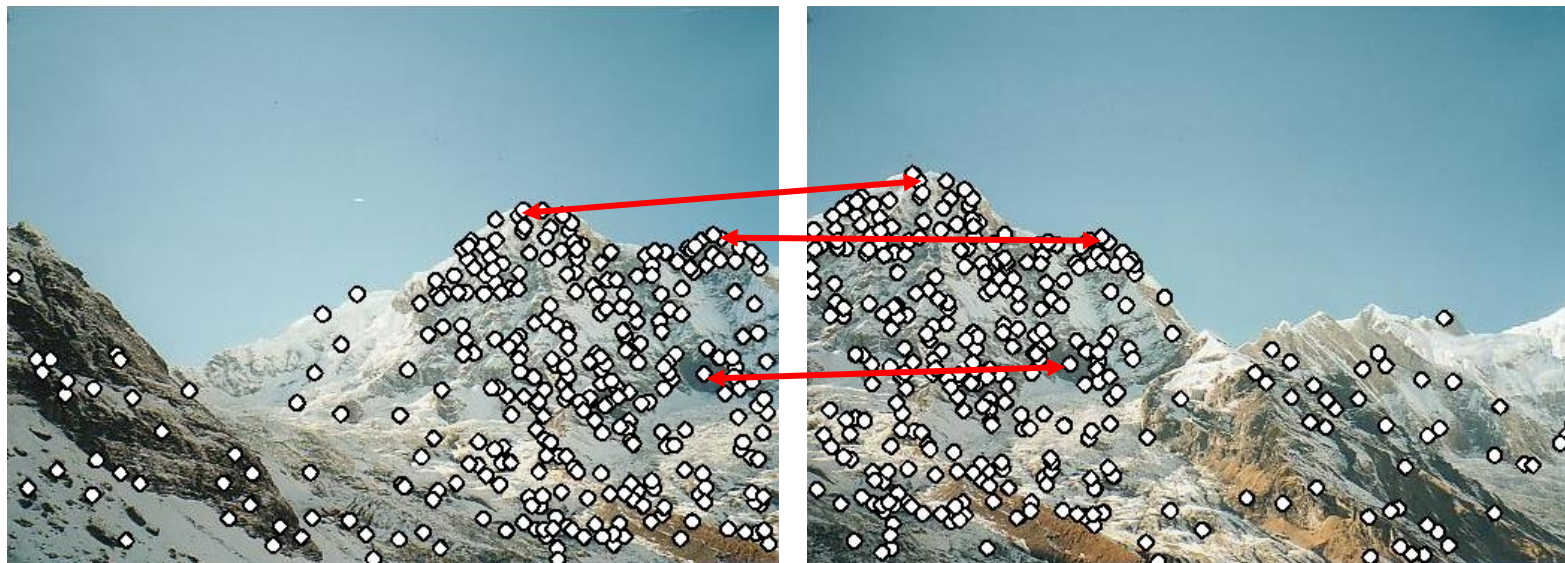
- Extract features

Robust feature-based alignment



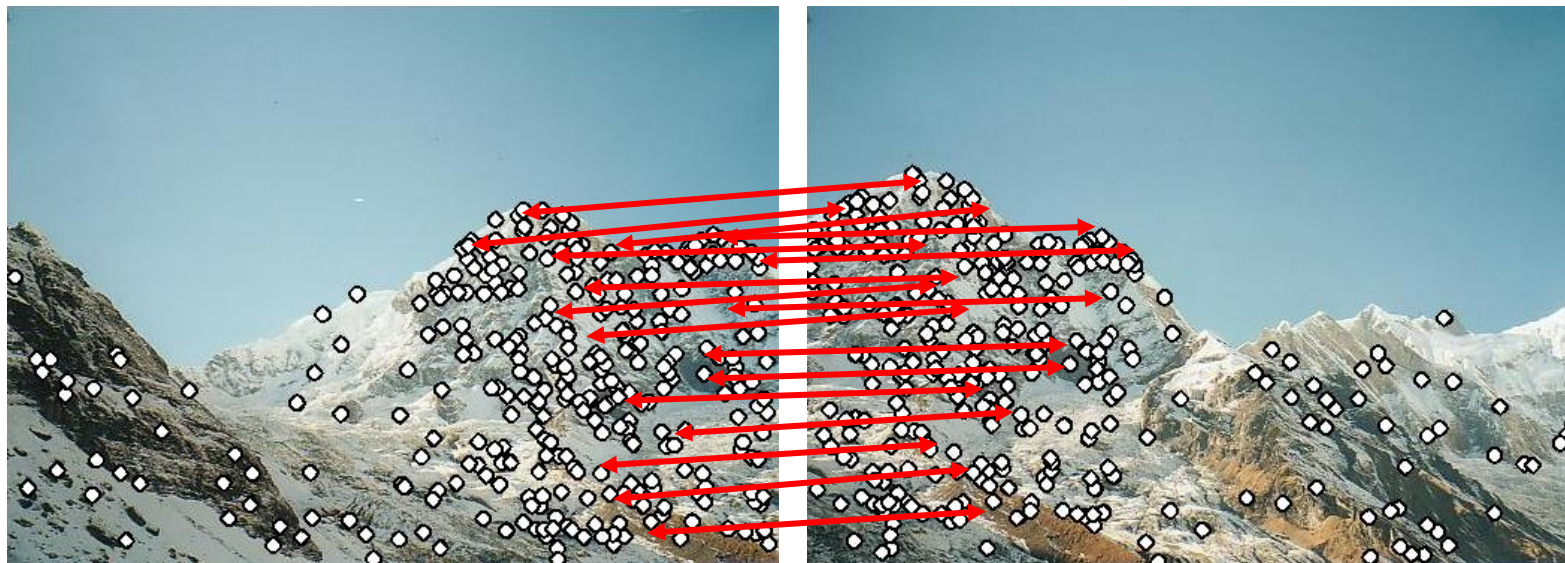
- Extract features
- Compute *putative matches*

Robust feature-based alignment



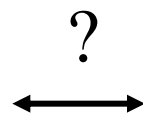
- Extract features
- Compute *putative matches*
- Loop:
 - *Hypothesize transformation T*

Robust feature-based alignment

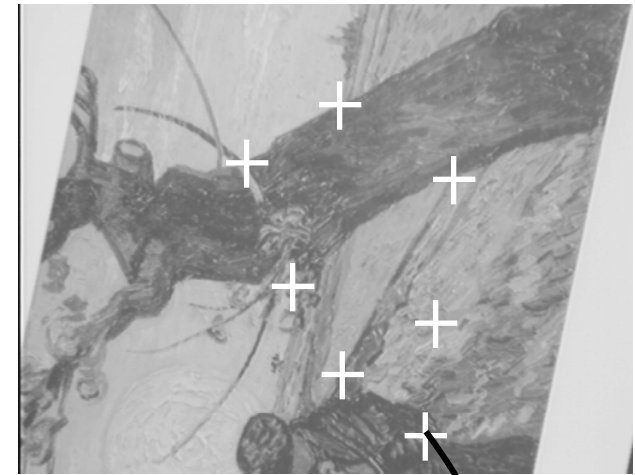
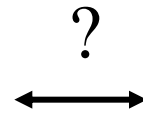
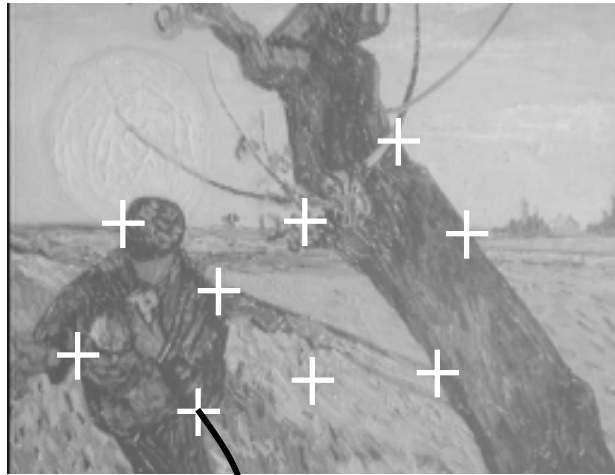


- Extract features
- Compute *putative matches*
- Loop:
 - *Hypothesize* transformation T
 - *Verify* transformation (search for other matches consistent with T)

Generating putative correspondences



Generating putative correspondences



feature
descriptor

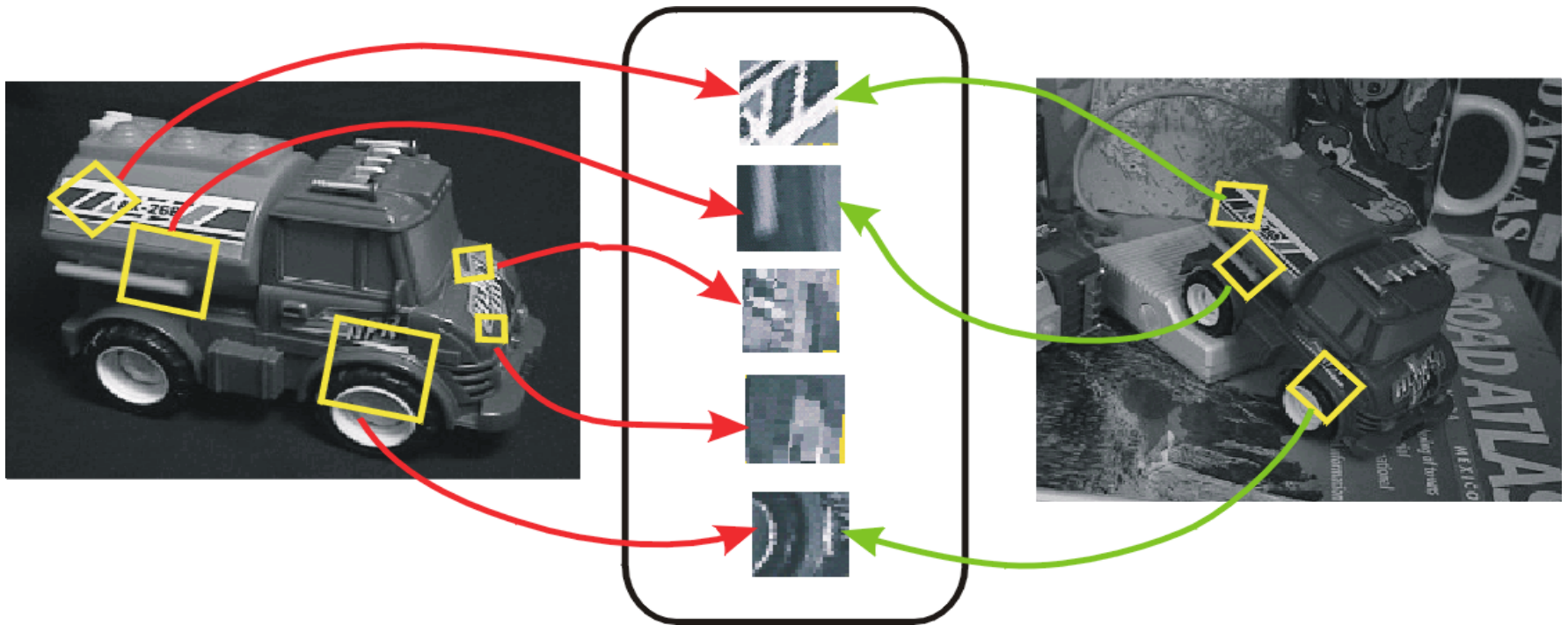


feature
descriptor

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

Feature descriptors

- Recall: feature detection and description



Feature descriptors

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

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

– Not invariant to intensity change

- Normalized correlation

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

– Invariant to affine intensity change

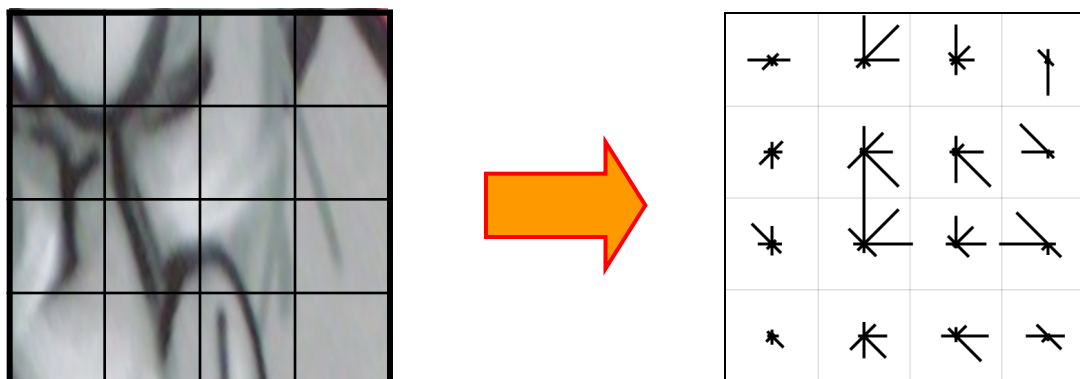
Disadvantage of intensity vectors as descriptors

- Small deformations can affect the matching score a lot



Feature descriptors: SIFT

- Descriptor computation:
 - Divide patch into 4x4 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.

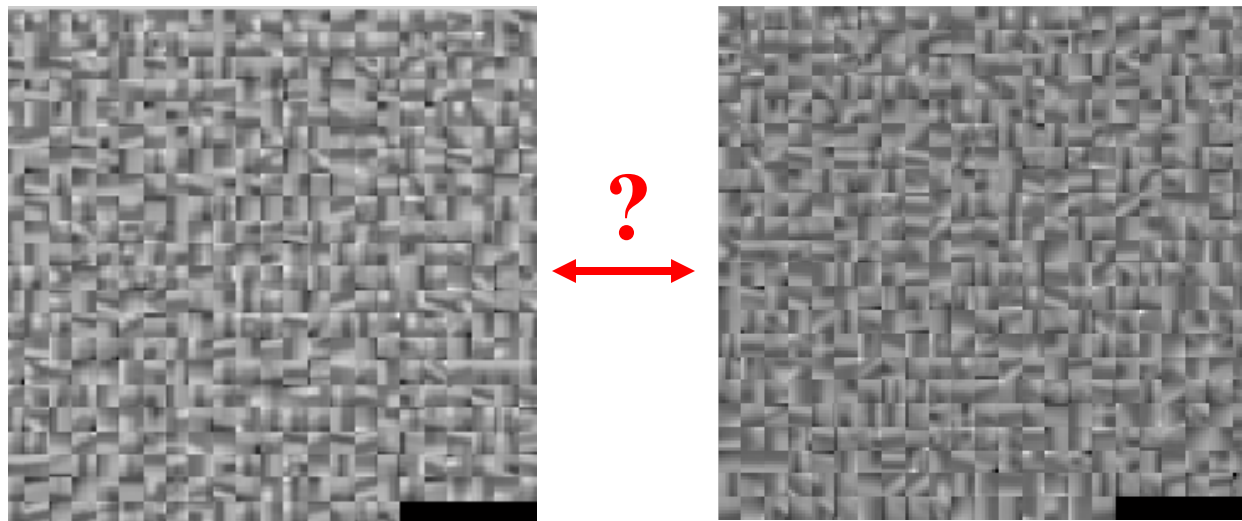
Feature descriptors: SIFT

- Descriptor computation:
 - Divide patch into 4x4 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

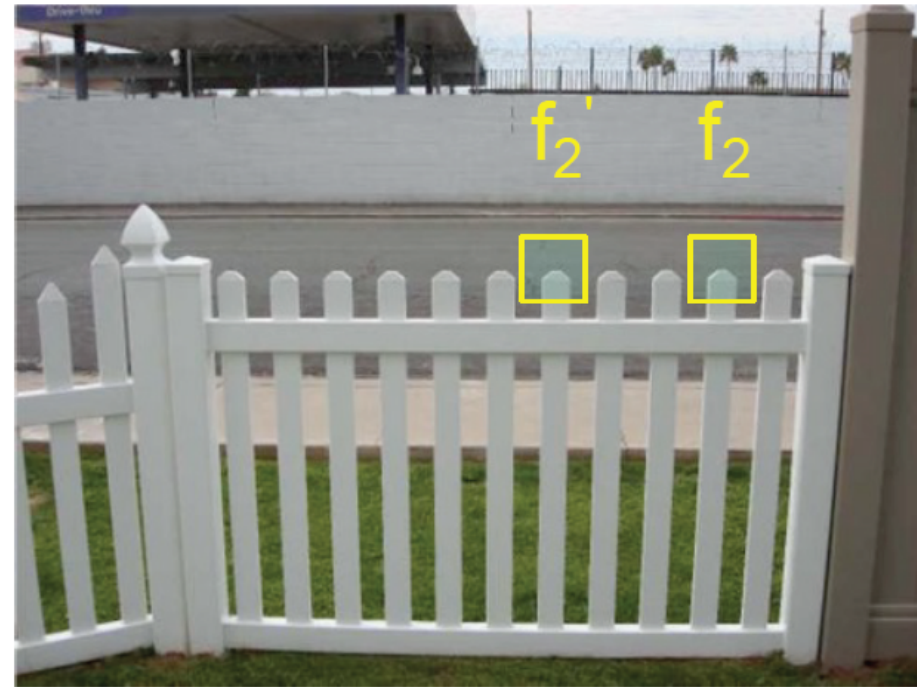
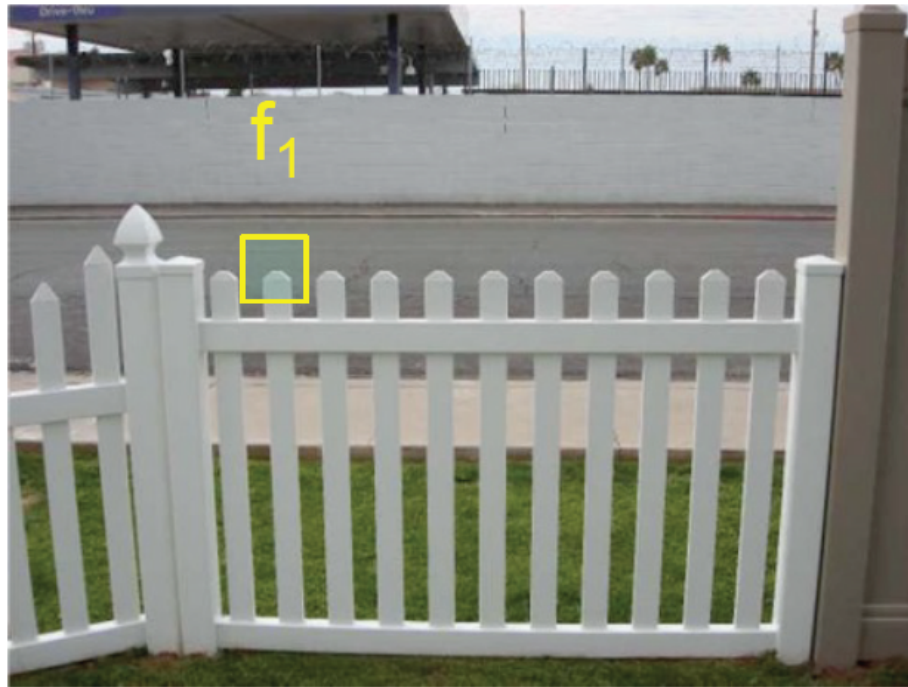
David G. Lowe. ["Distinctive image features from scale-invariant keypoints."](#) *IJCV* 60 (2), pp. 91-110, 2004.

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

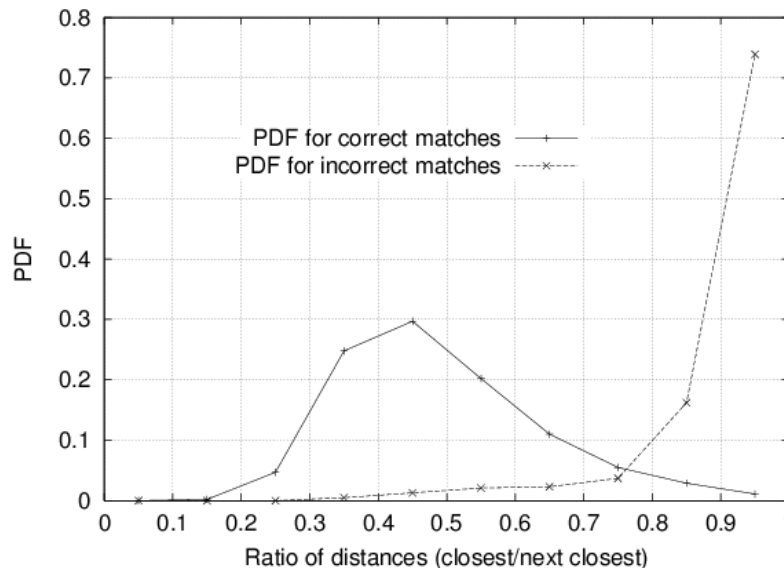


Problem: Ambiguous putative matches



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**

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

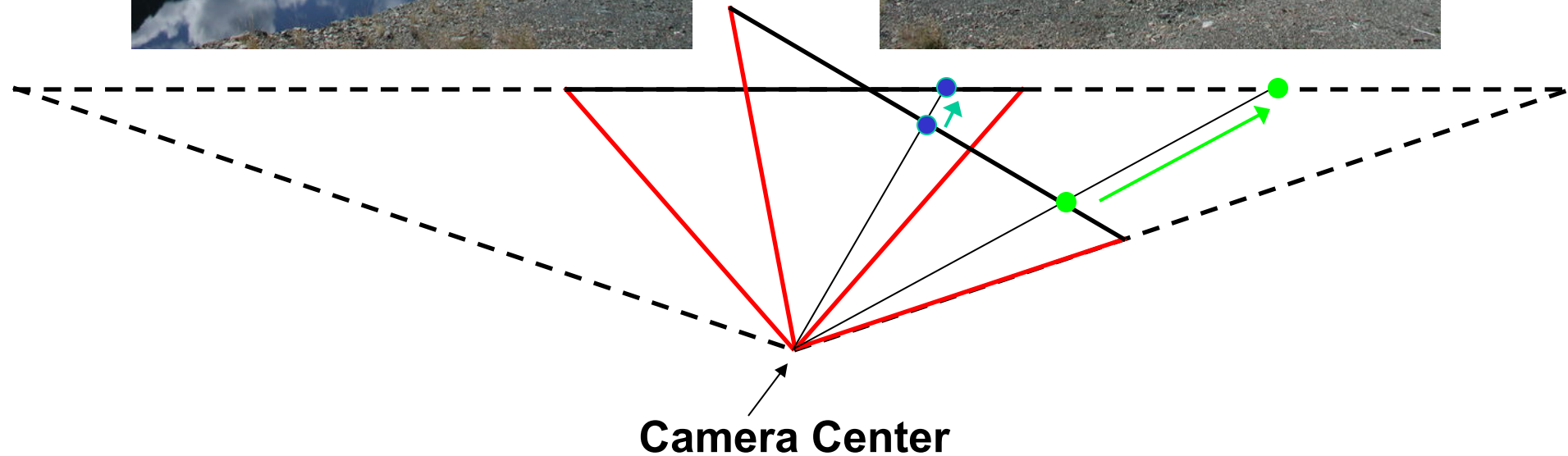
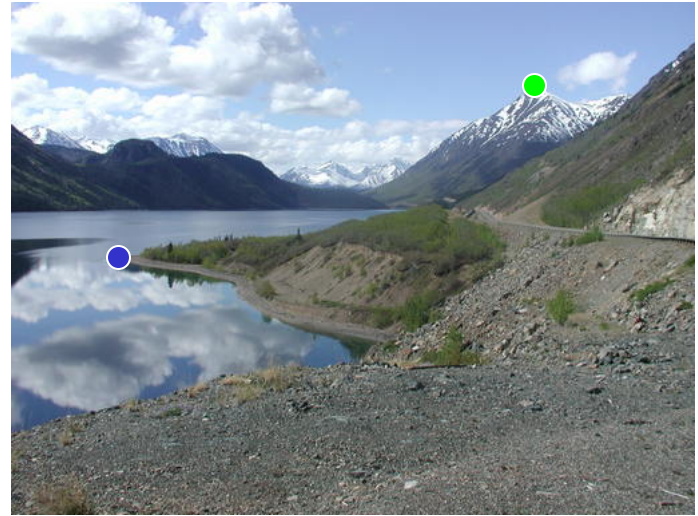
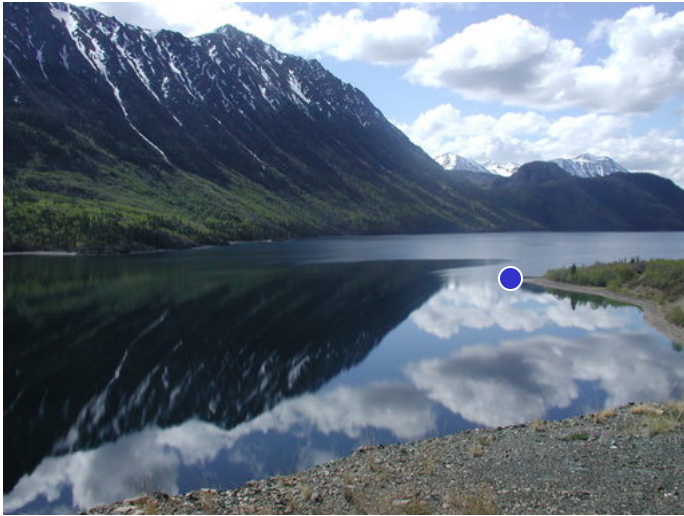
Keep the transformation with the largest number of inliers

Robust feature-based alignment



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

Homography Example



Common transformations



original

Transformed



translation



rotation



aspect



affine

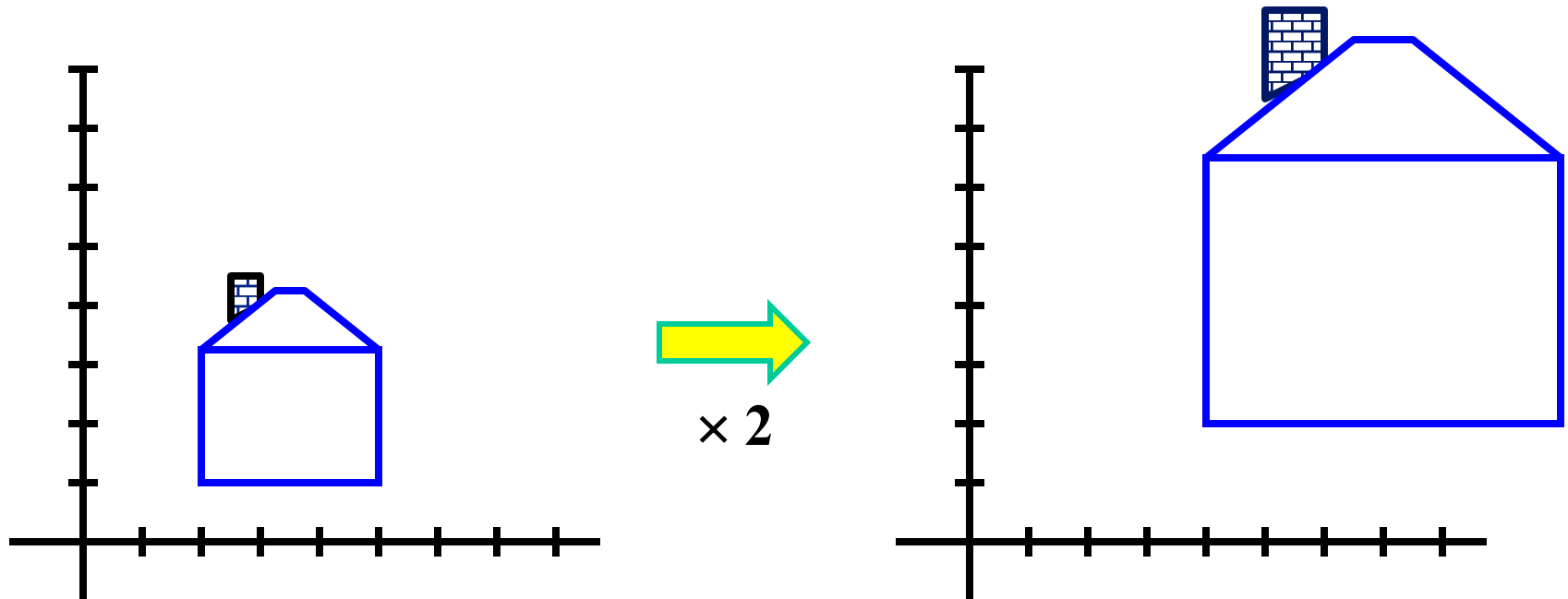


perspective

Scaling

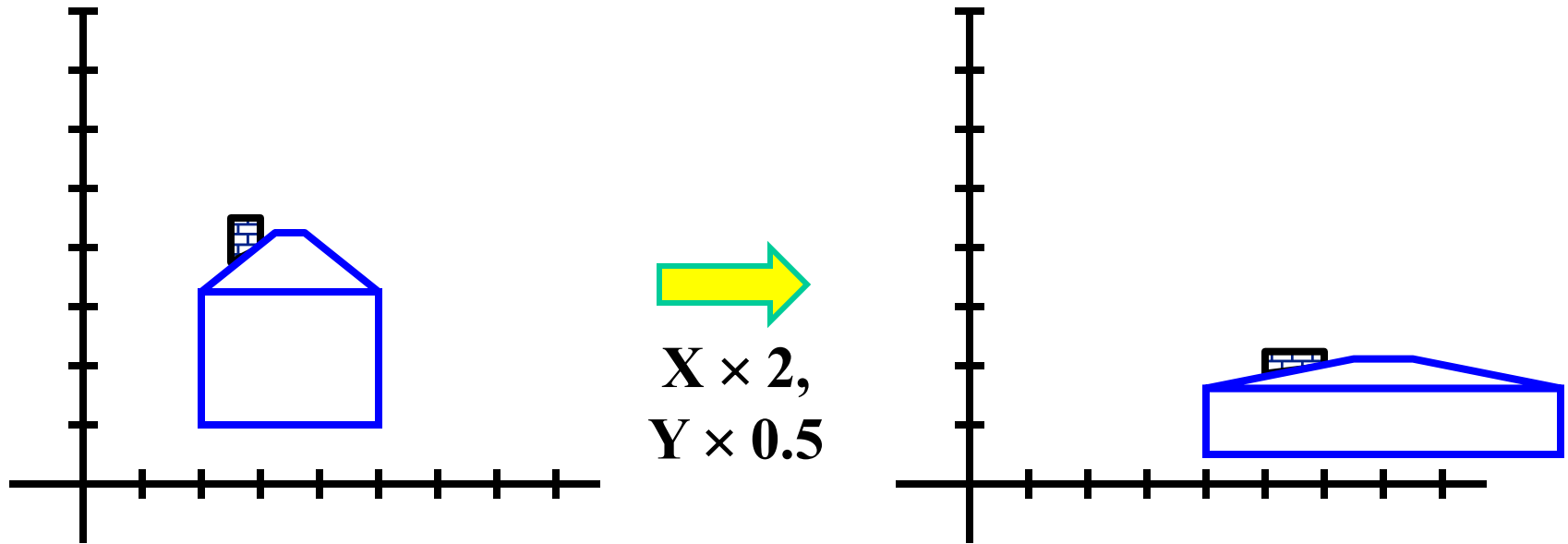
Scaling a coordinate means multiplying each of its components by a scalar

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



Scaling

Non-uniform scaling: different scalars per component:



Scaling

Scaling operation:

$$x' = ax$$

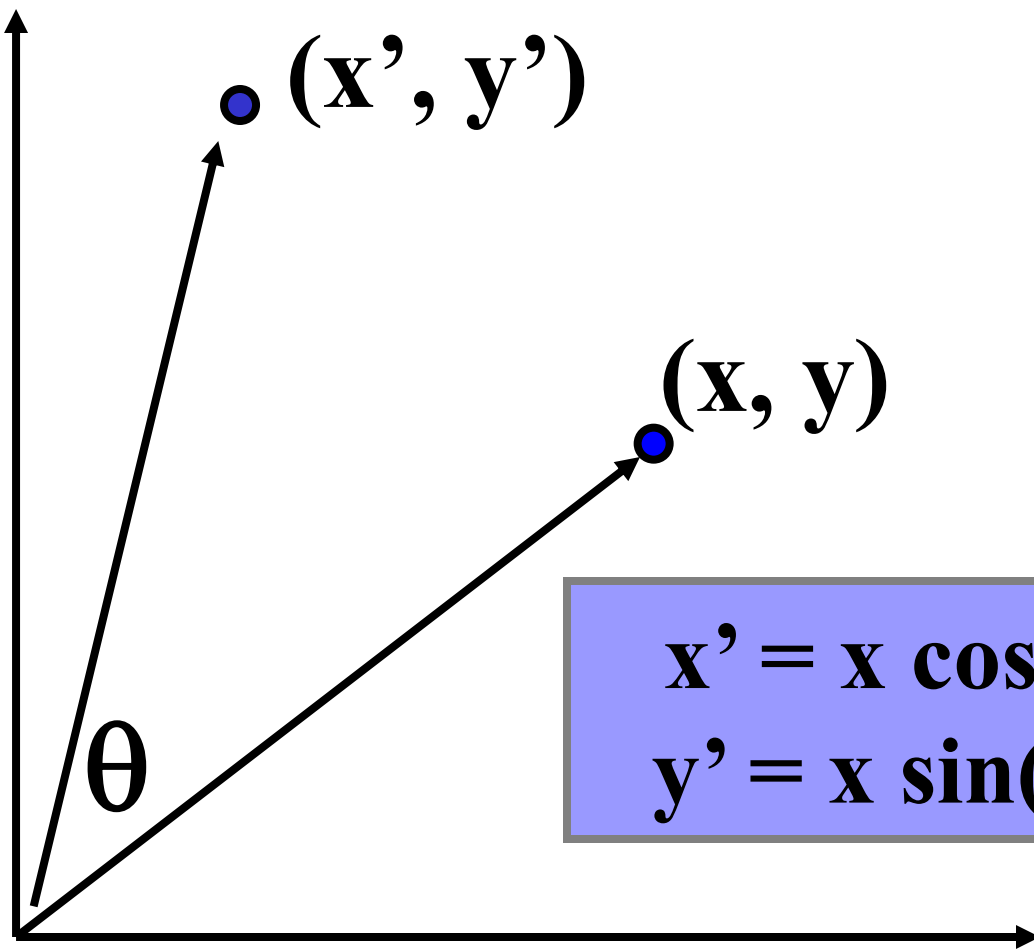
$$y' = by$$

Or, in matrix form:

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

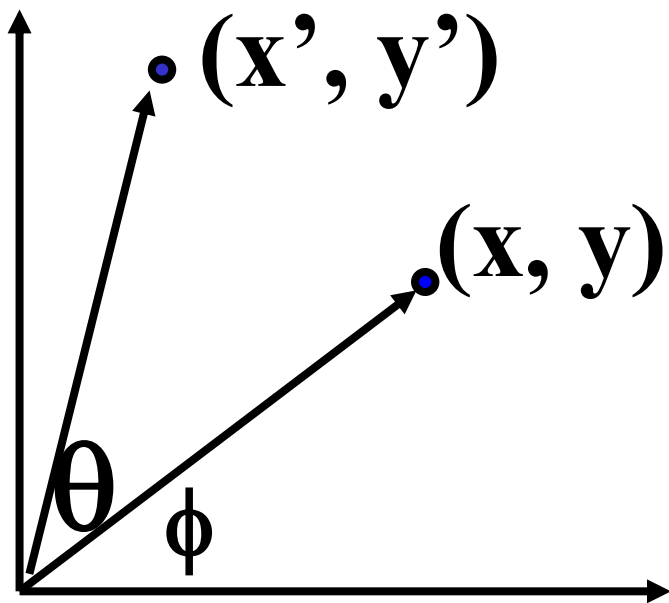
scaling matrix S

2-D Rotation



$$\begin{aligned}x' &= x \cos(\theta) - y \sin(\theta) \\y' &= x \sin(\theta) + y \cos(\theta)\end{aligned}$$

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...

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

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

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 $\sin(\theta)$ and $\cos(\theta)$ 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 $-\theta$
- For rotation matrices **$\mathbf{R}^{-1} = \mathbf{R}^T$**

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

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

Translate

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

Affine

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

Affine Transformations

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

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

or

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

Projective Transformations

Projective transformations are combos of

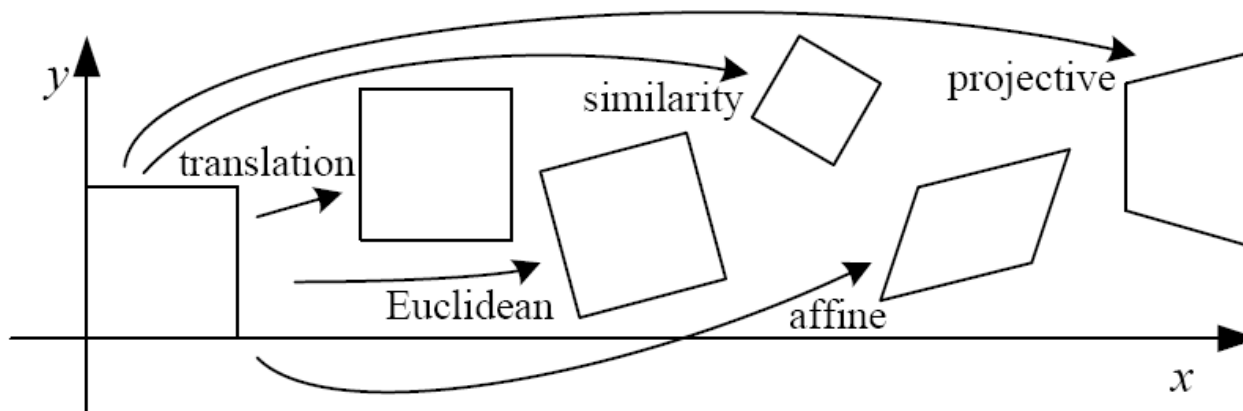
- Affine transformations, and
- Projective warps

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

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)**

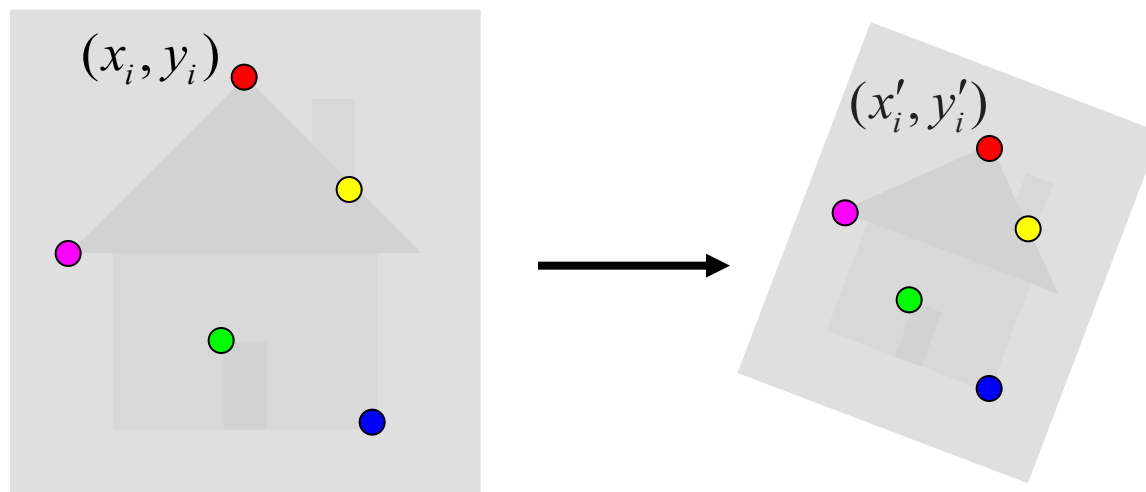
2D image transformations



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

Fitting an affine transformation

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



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

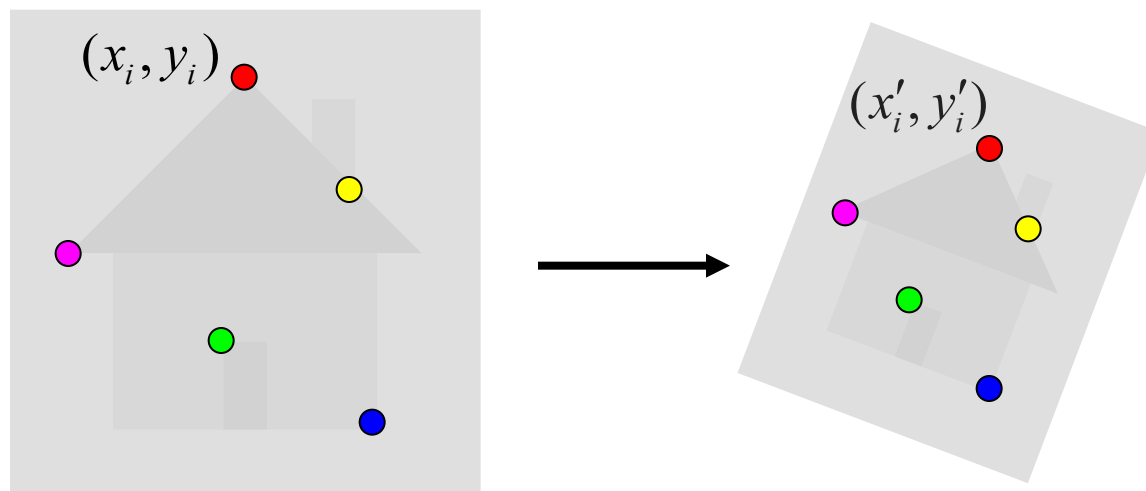
Want to find \mathbf{M} , \mathbf{t} to minimize

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

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

Fitting an affine transformation

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

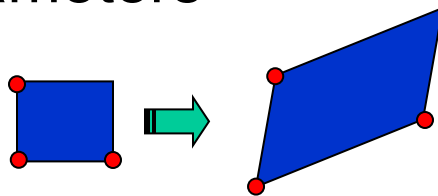


$$\begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$
$$\begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \dots \\ x'_i \\ y'_i \\ \dots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

Fitting an affine transformation

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

- 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 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:

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

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

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

3 equations,
only 2 linearly
independent

Fitting a homography

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

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

Feature-based alignment: Overview

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