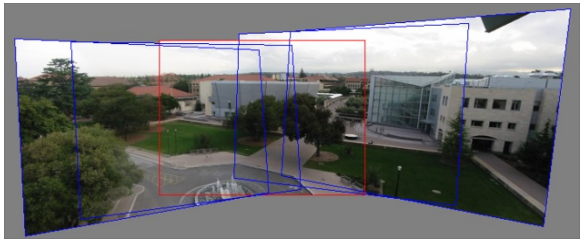
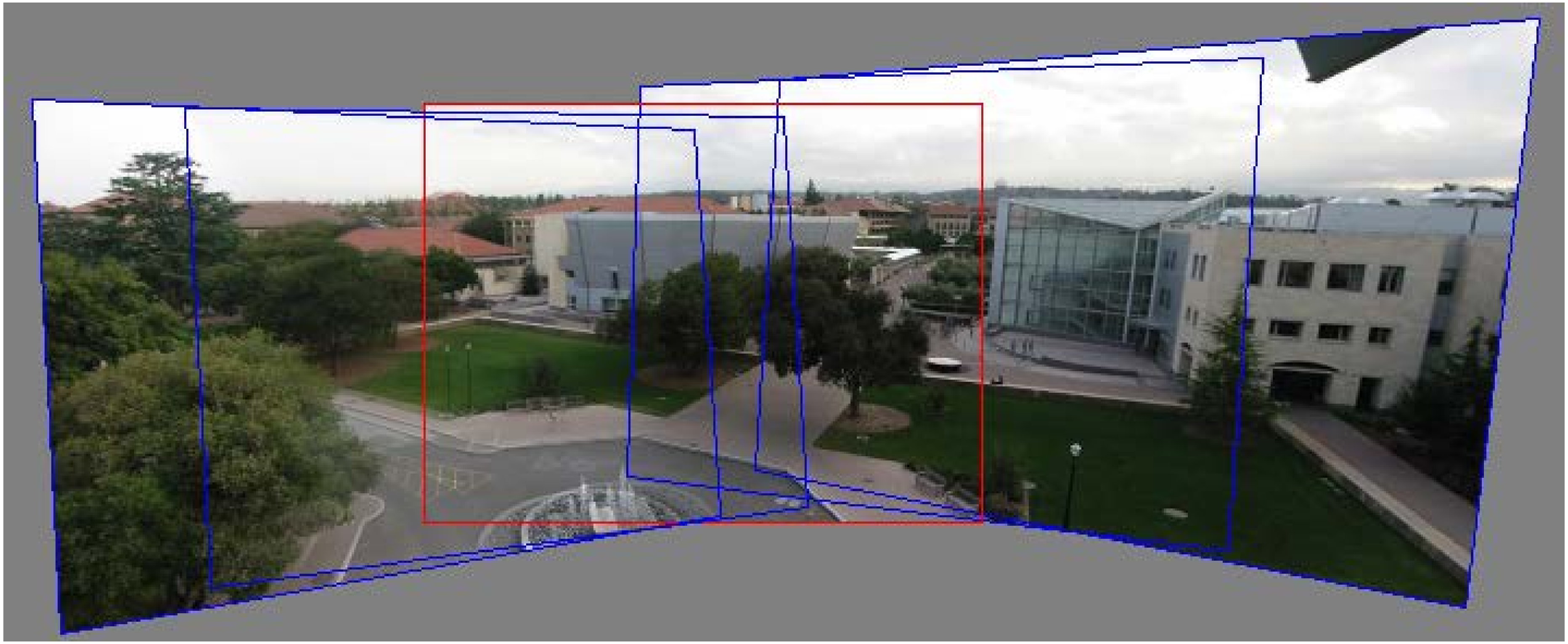
Introduction to Computer Vision



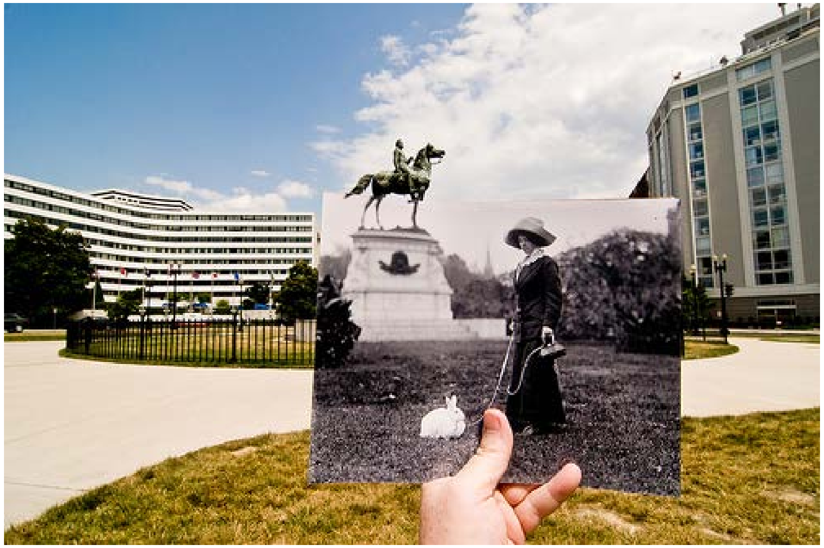
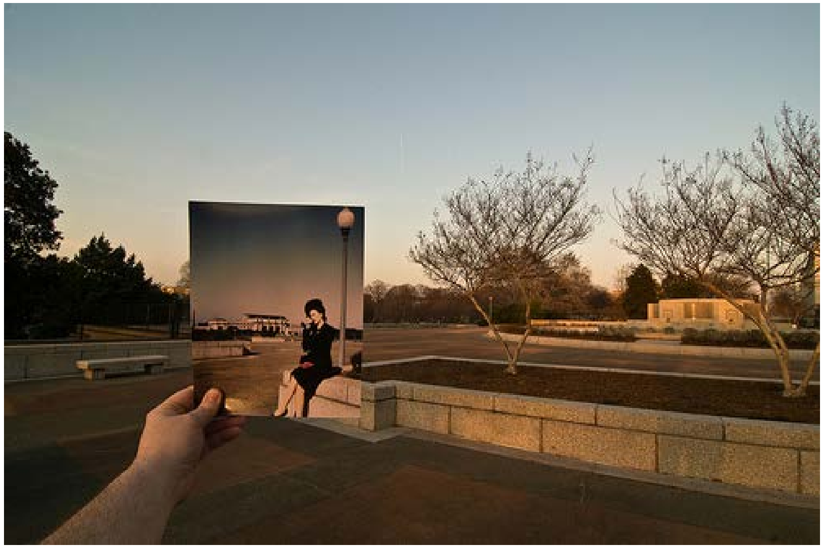
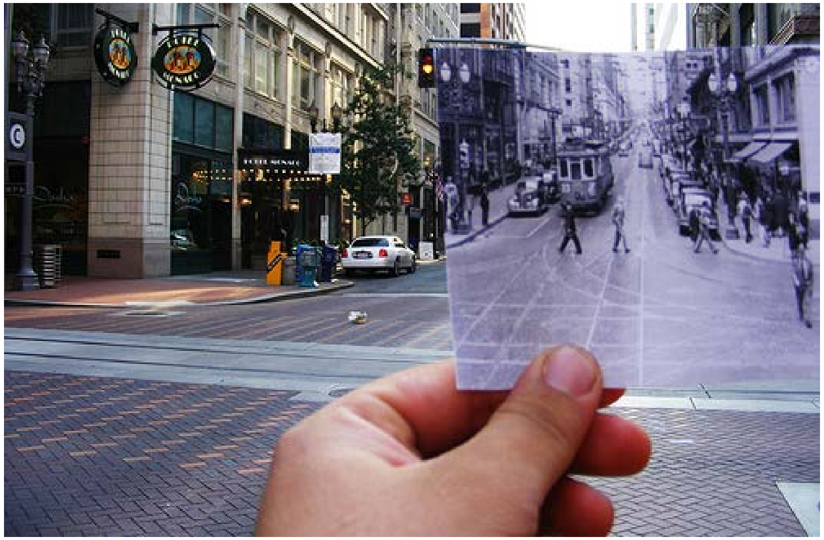
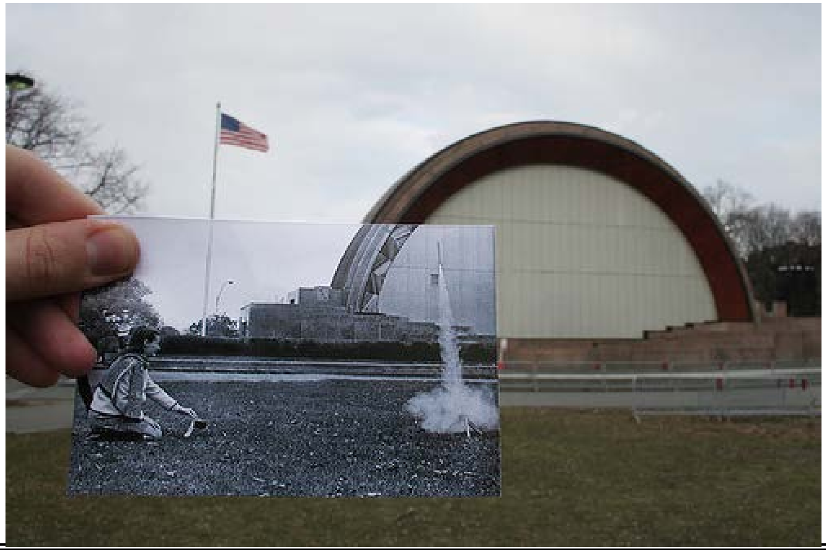
Homography and Alignment

Prof. Kyoung Mu Lee
Dept. of ECE, Seoul National University

Robust and large-scale alignment



A look into the past

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

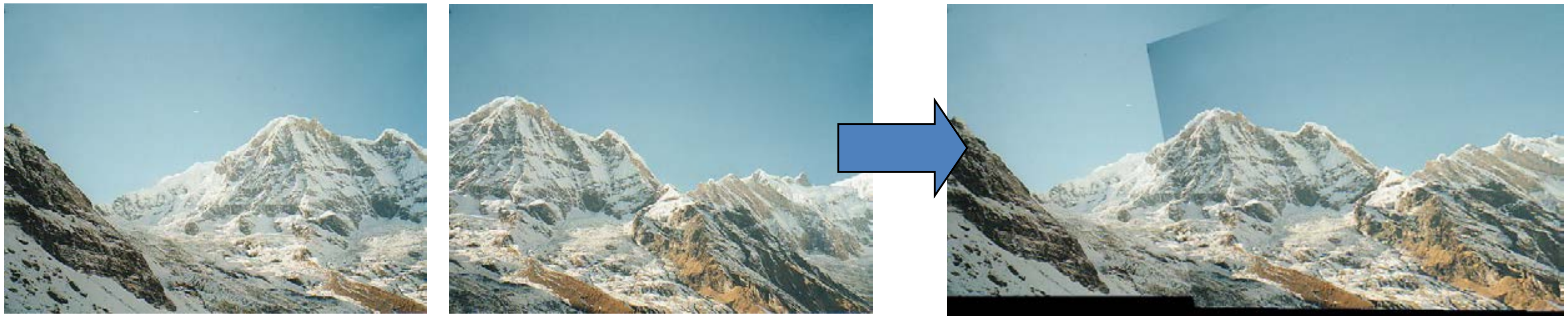
A look into the past

- Leningrad during the blockade

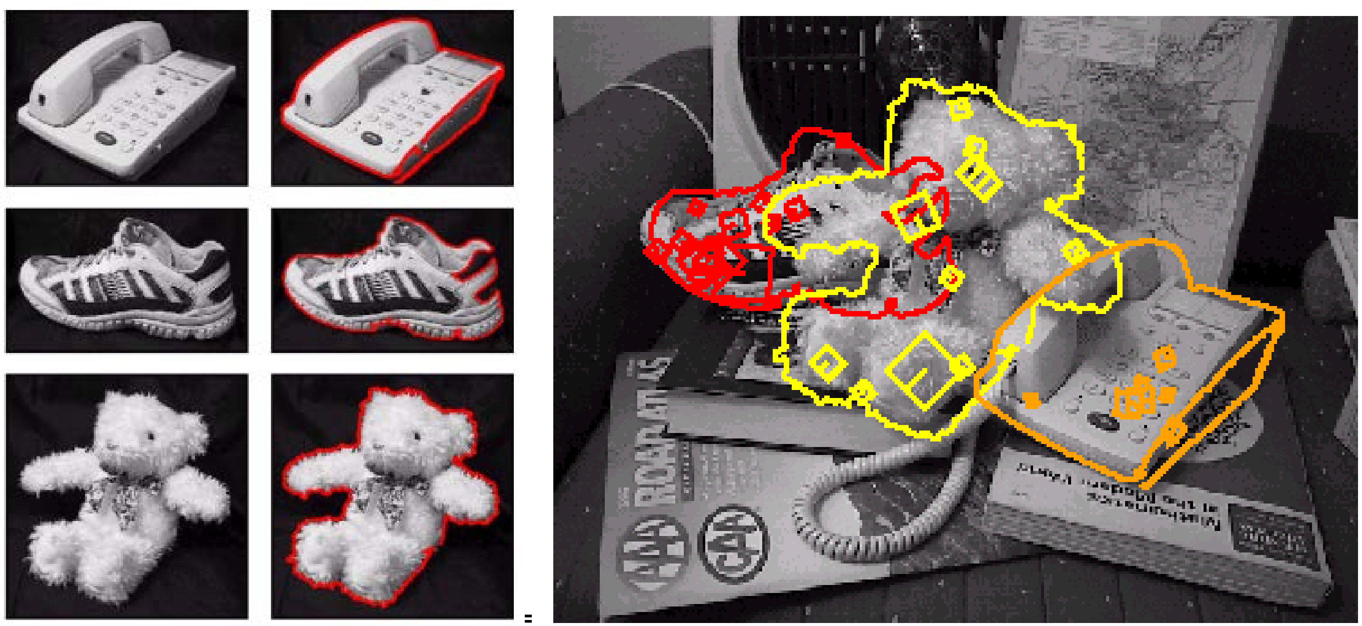
http://komen-dant.livejournal.com/345684.html

Bing streetside images

Image alignment: Applications

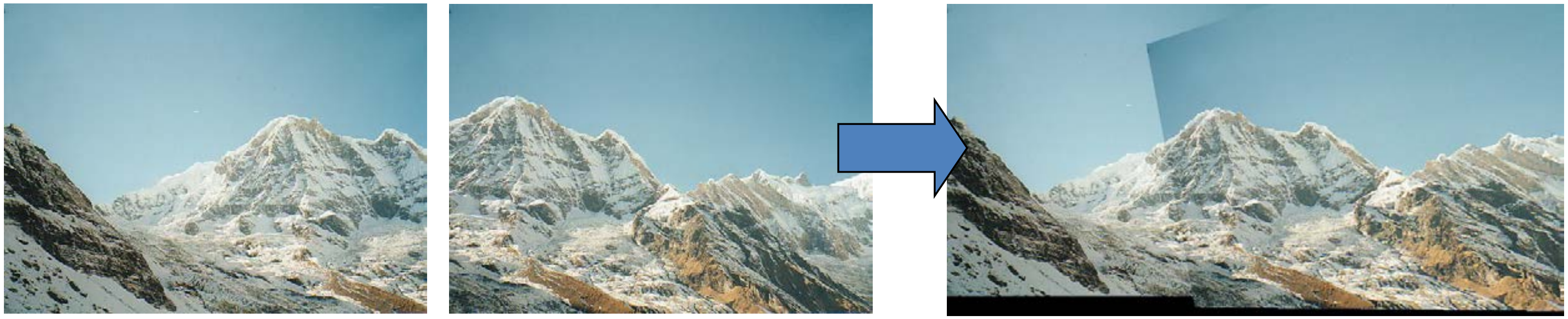


Panorama stitching



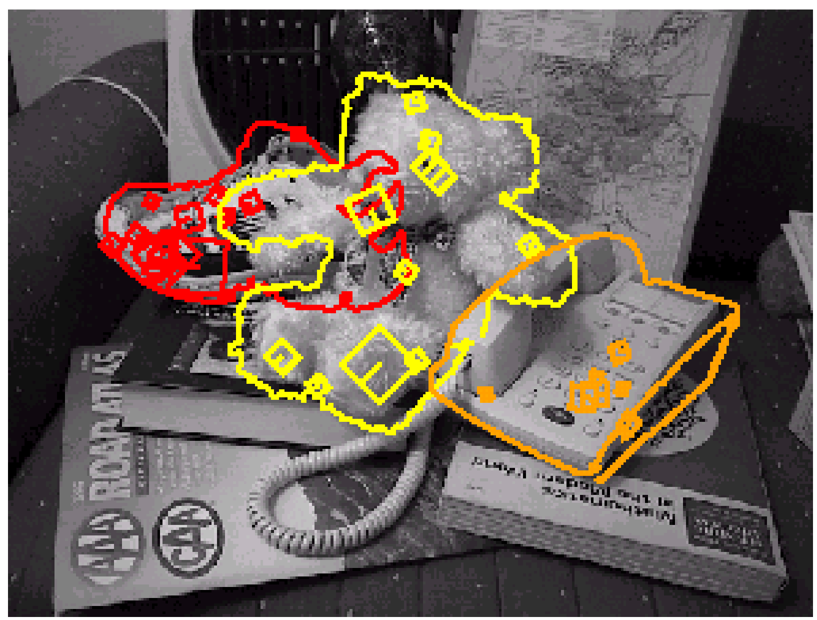
Recognition of object instances

Image alignment: Challenges



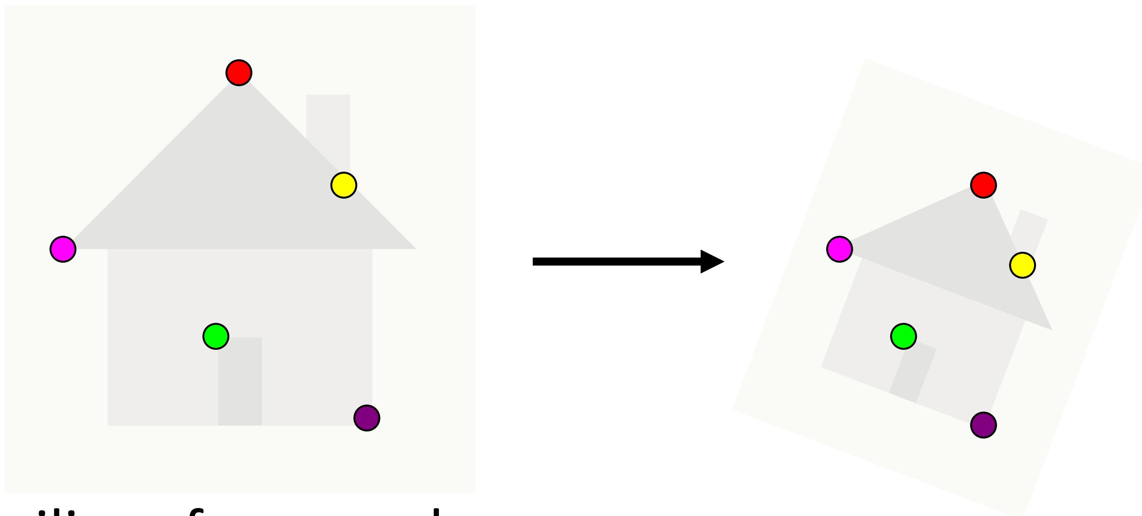
Small degree of overlap

Intensity changes



Occlusion, clutter

Image alignment

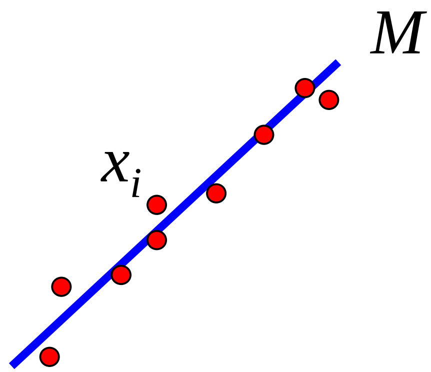


- Two families of approaches:
  - ✓ **Direct (pixel-based) alignment**
    - Search for alignment where most pixels agree
  - ✓ **Feature-based alignment**
    - Search for alignment where *extracted features* agree
    - Can be verified using pixel-based alignment

Slide credit: Kristen Grauman

Alignment as fitting

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

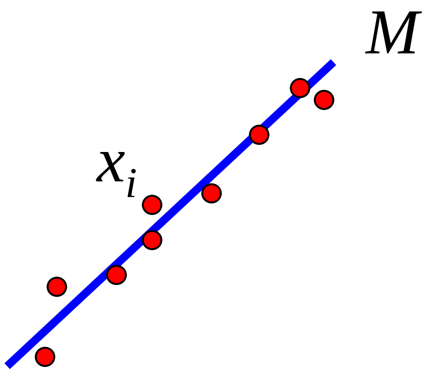


Find model *M* that minimizes

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

Alignment as fitting

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

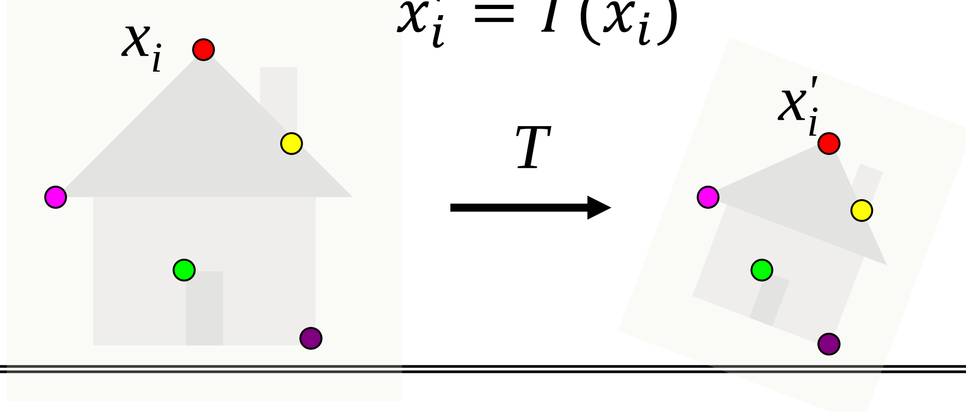


Find model M that minimizes

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

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

$$x_i' = T(x_i)$$



Find transformation T that minimizes

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

2D transformation models

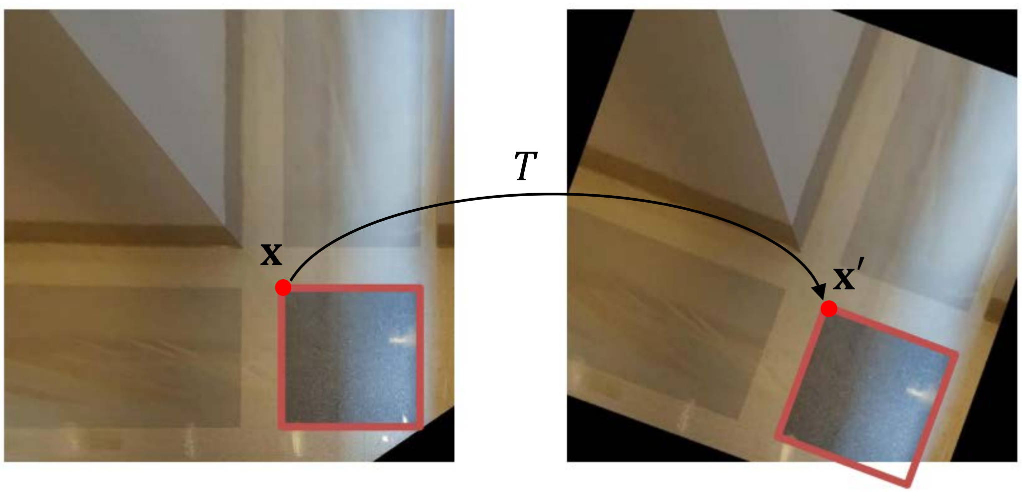
- Euclidean (translation, rotation)

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

Translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotation



Invariant Properties

- Length
- Area
- Angle

DOF (Degree of Freedom)

- 3 (2 translation + 1 rotation)

$$\mathbf{x}' = T(\mathbf{x})$$

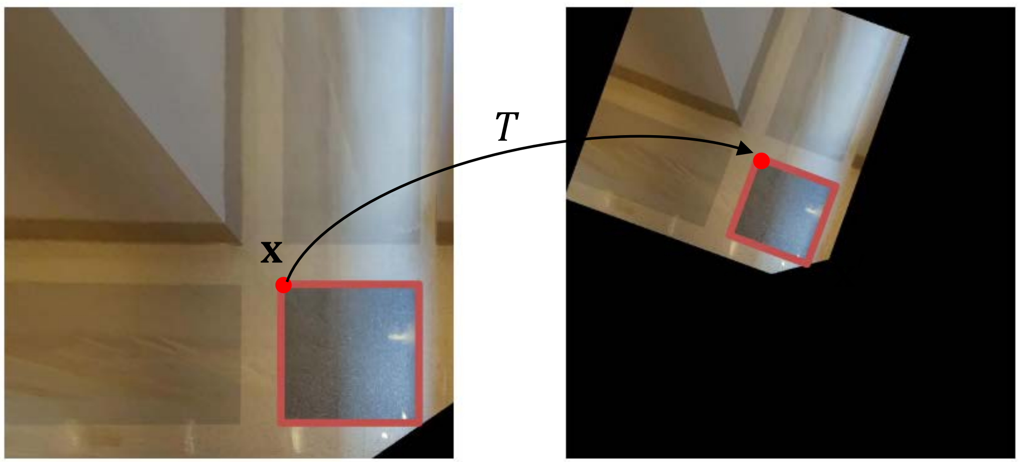
$$\boxed{\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}$$

Euclidean

- Similarity (translation, scale, rotation)

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

Scaling



Invariant Properties

- Length ratio
- Angle

DOF (Degree of Freedom)

- 4 (2 translation + 1 rotation+ 1 scale)

$$\mathbf{x}' = T(\mathbf{x})$$

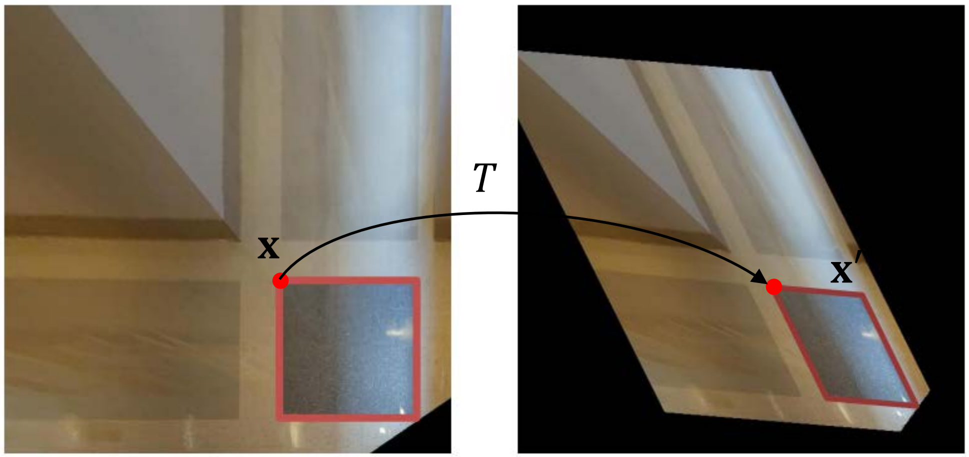
$$\boxed{\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}$$

Similarity

- Affine

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Shearing



Invariant Properties
- Parallelism
- Area ratio
- Length ratio

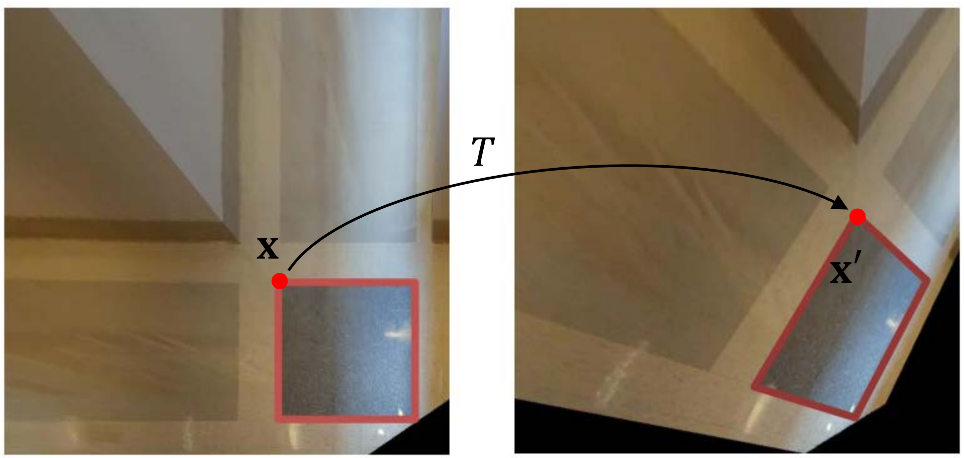
DOF (Degree of Freedom)
- 6

$$\mathbf{x}' = T(\mathbf{x})$$

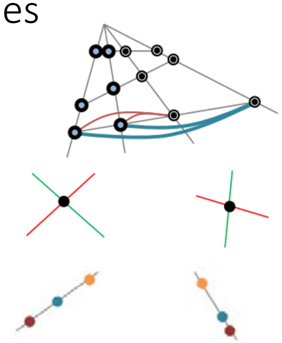
$$\boxed{\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}$$

Affine

- Projective (homography)
  - ✓ Linear mapping from plane to plane



Invariant Properties
- Cross ratio
- Concurrency
- Colinearity



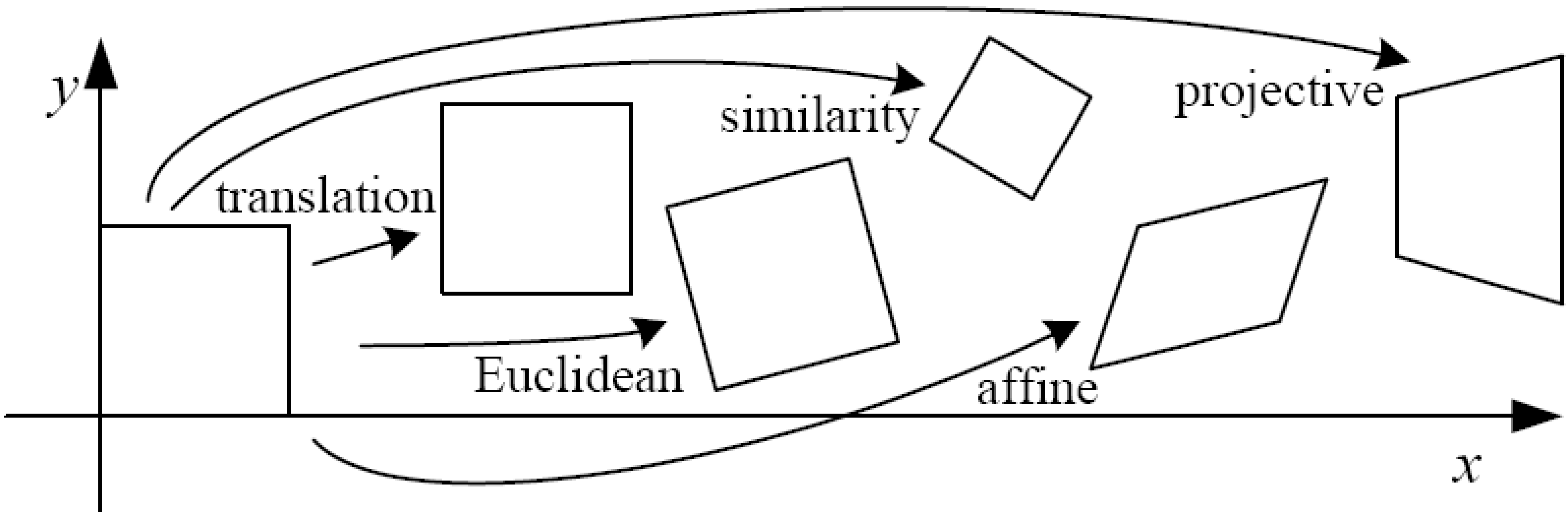
DOF (Degree of Freedom)
- 8 (9 – 1 scale)

$$\mathbf{x}' = T(\mathbf{x})$$

$$\boxed{\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}$$

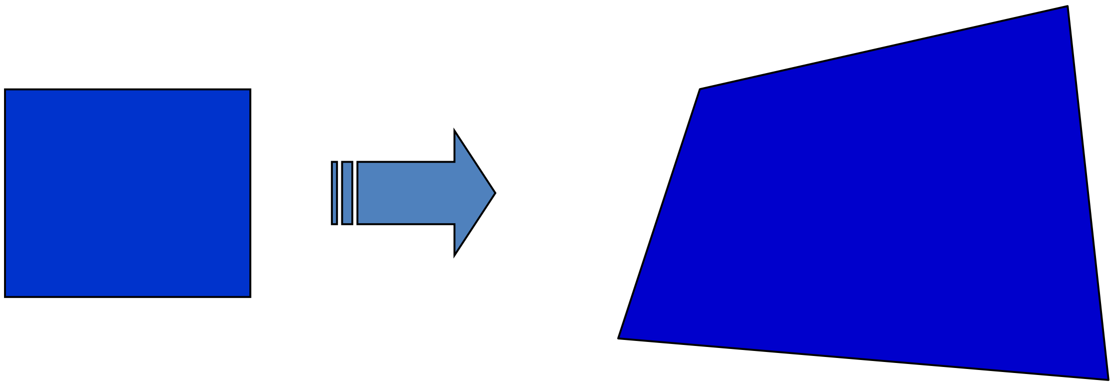
Projective (Homography)

2D transformation models



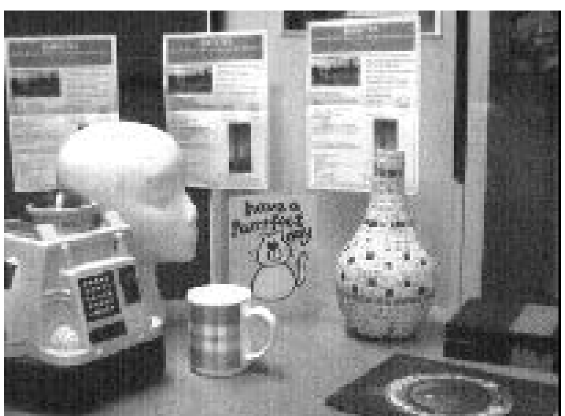
Fitting a plane by projective transformation

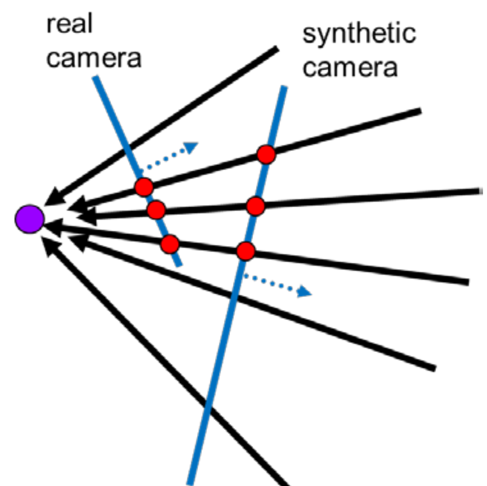
- **Homography:** plane projective transformation (transformation taking a quad to another arbitrary quad)



- The transformation between two views of a *planar surface*

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





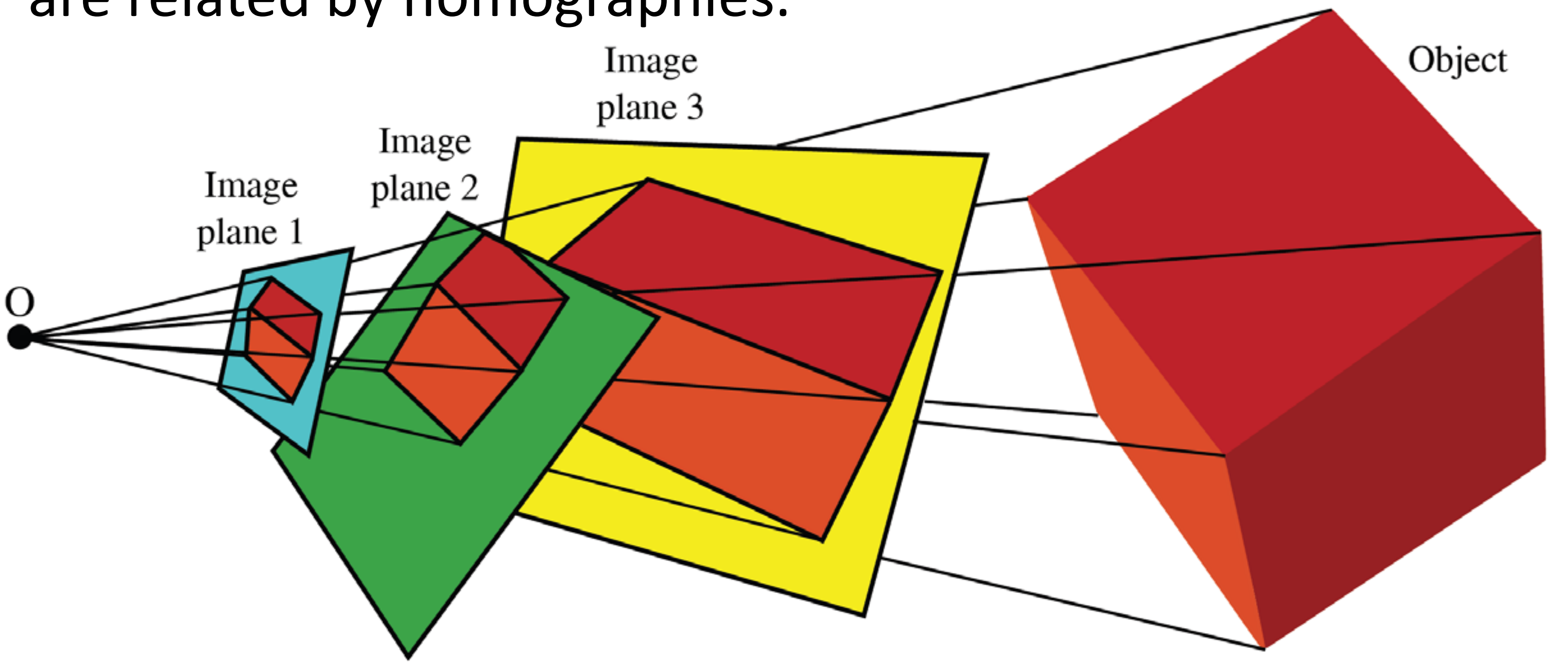
Slide credit: Svetlana Lazebnik

Properties of the homography

Homography is a linear transformation of a ray

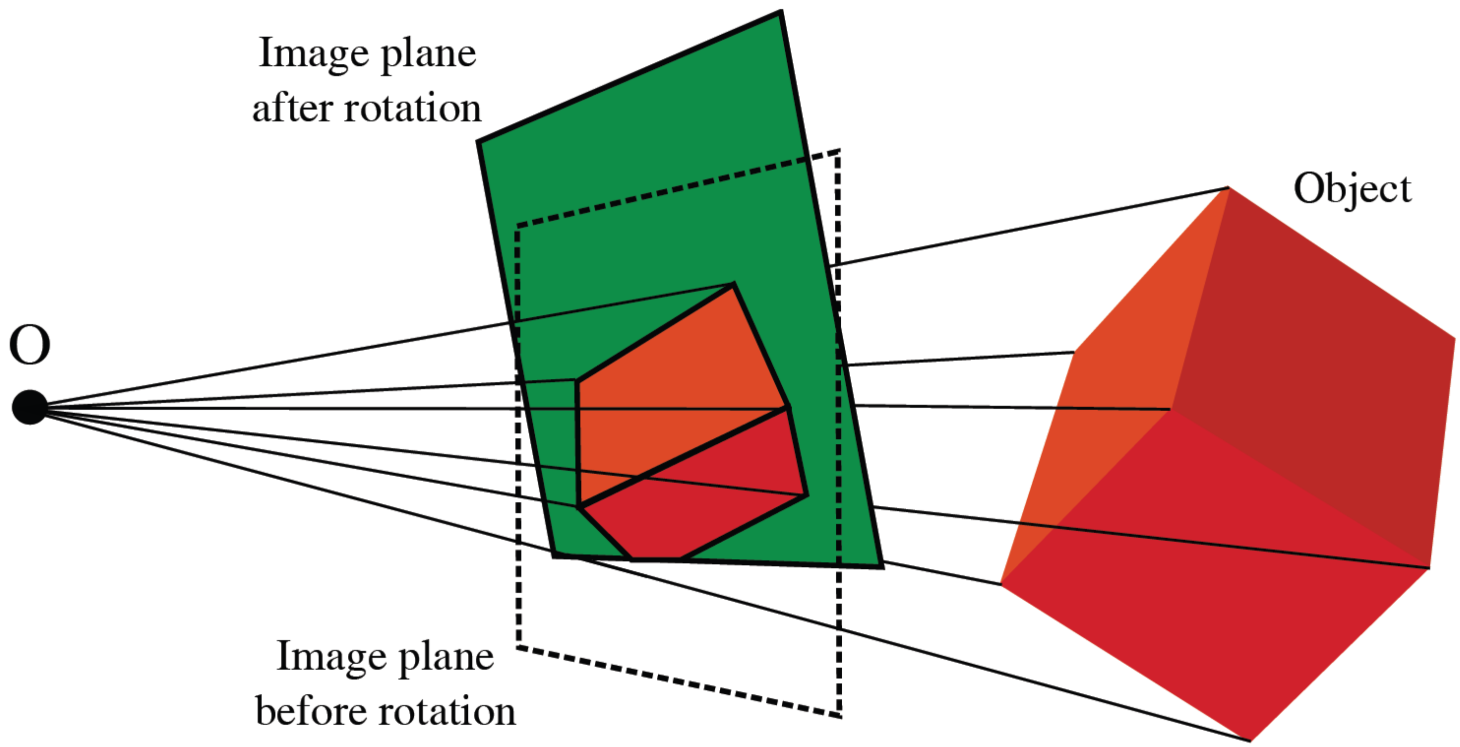
$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Equivalently, leave rays and linearly transform image plane – all images formed by all planes that cut the same ray bundle are related by homographies.



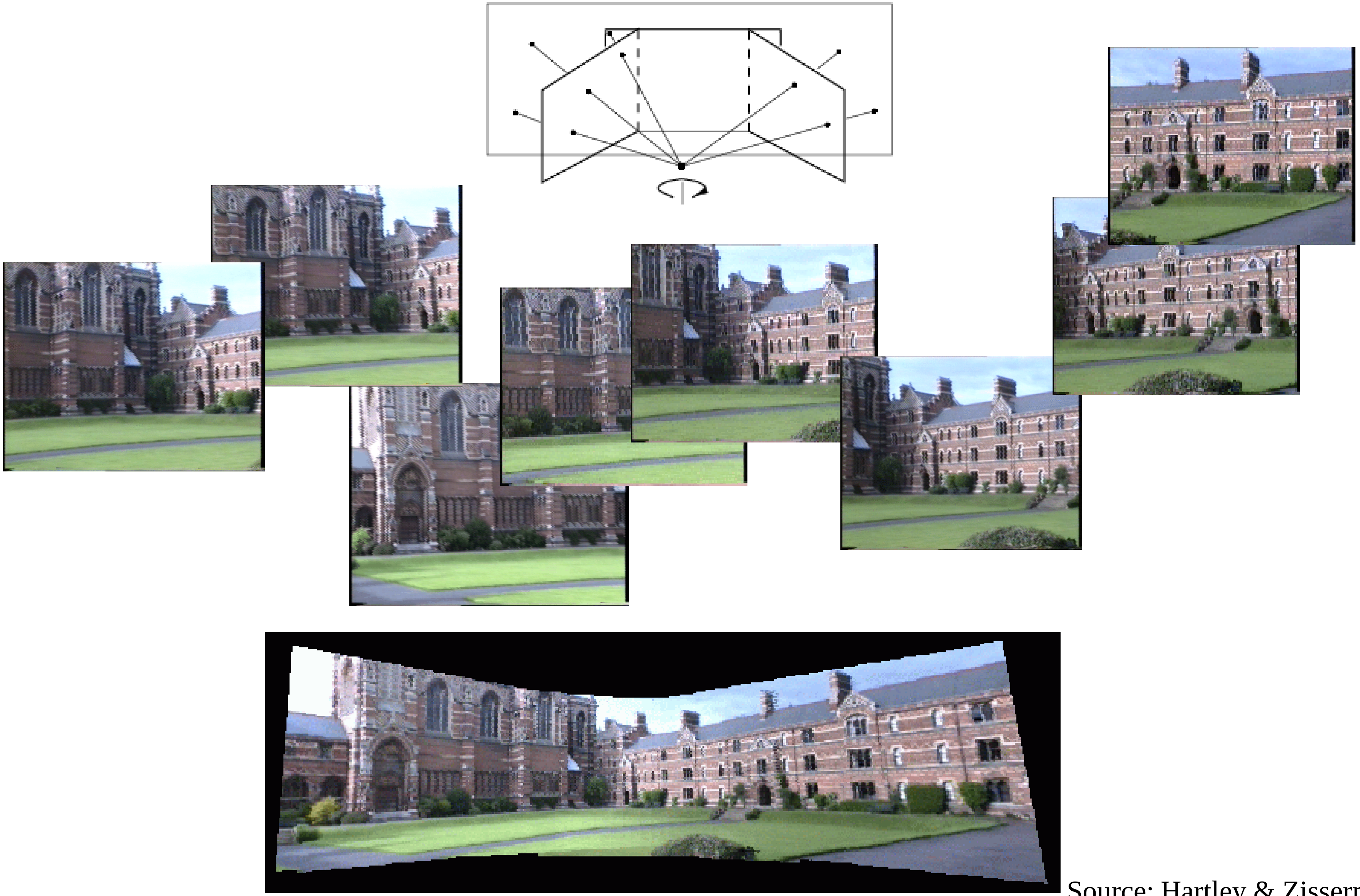
Source: Simon Prince

Special case is camera under pure rotation.



Source: Simon Prince

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 credit: Svetlana Lazebnik

Fitting a homography

- Recall: homogeneous coordinates

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

Converting *to* homogeneous image coordinates

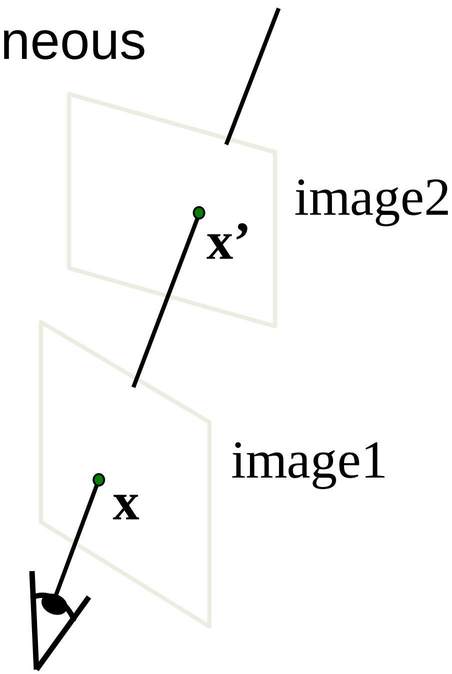
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

Converting *from* homogeneous image coordinates

- Equation for homography:

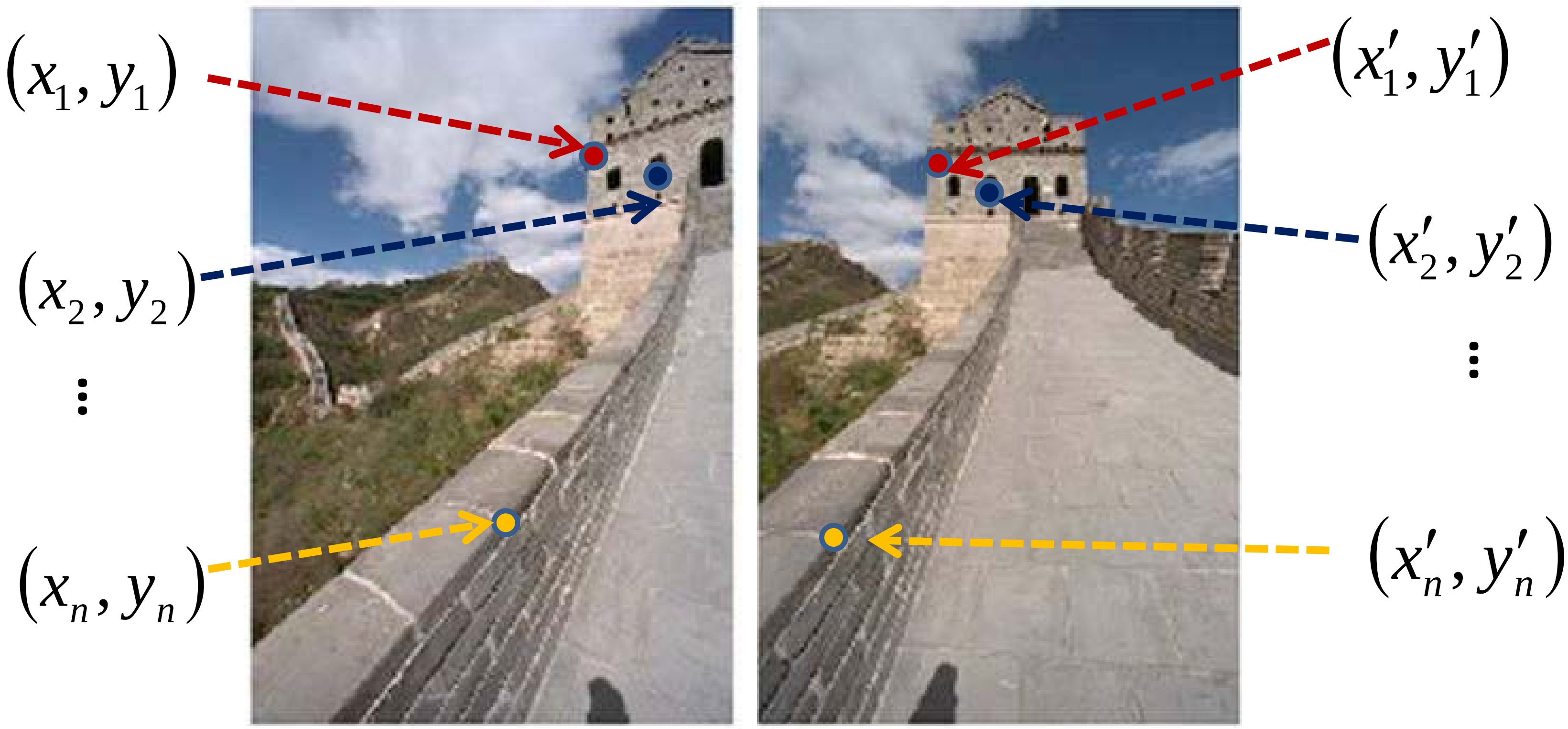
$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

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



Slide credit: Svetlana Lazebnik

Fitting a homography



To **compute** the homography given pairs of corresponding points in the images, we need to set up an equation where the parameters of **H** are the unknowns…

Slide credit: Kristen Grauman

Fitting a homography

- Equation for homography:

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

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

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

$$\Rightarrow \begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{h}_1^T \mathbf{x}_i \\ \mathbf{h}_2^T \mathbf{x}_i \\ \mathbf{h}_3^T \mathbf{x}_i \end{bmatrix} = \begin{bmatrix} y_i' \mathbf{h}_3^T \mathbf{x}_i - \mathbf{h}_2^T \mathbf{x}_i \\ \mathbf{h}_1^T \mathbf{x}_i - x_i' \mathbf{h}_3^T \mathbf{x}_i \\ x_i' \mathbf{h}_2^T \mathbf{x}_i - y_i' \mathbf{h}_1^T \mathbf{x}_i \end{bmatrix} = 0$$

single correspondence

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

3 equations, only 2 linearly independent

Slide credit: Svetlana Lazebnik

Direct linear transform (DLT)

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

- $\mathbf{H}$ has 8 degrees of freedom (9 parameters, but scale is arbitrary)
- One match gives us two linearly independent equations
- Four matches needed for a minimal solution (null space of 8x9 matrix)
- More than four: homogeneous least squares

Slide credit: Svetlana Lazebnik

- 4 point case:
  - ✓ Solve the exact solution **H** satisfying

$$\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ \mathbf{A}_4 \end{bmatrix} \mathbf{h} = \mathbf{0} \quad \Rightarrow \quad \mathbf{A}\mathbf{h} = \mathbf{0}$$

  - ✓ The size of **A** is *8*x*9* or *12*x*9* (rank is *8)*
  - ✓ 1-D null-space **h** is the nontrivial solutions
  - ✓ Choose the one with $\|\mathbf{h}\| = 1$
  - ✓ If we take SVD of $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathrm{T}}$, then the solution **h** is the last column of **V**, which is the eigenvector corresponding to the smallest eigenvalue

The DLT algorithm (cont.)

- More points case:
  - ✓ Over-determined solution satisfying

$$\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_n \end{bmatrix} \mathbf{h} = \mathbf{0} \quad \Rightarrow \quad \mathbf{Ah} = \mathbf{0}$$

  - ✓ No exact solution due to the “noise”
  - ✓ So, find the approximate solution

$$\mathbf{h}^* = \min_{\mathbf{h}} \|\mathbf{Ah}\|$$

  - ✓ Additional constraint needed for nontrivial solution e.g. $\|\mathbf{h}\| = 1$
  - ✓ The solution **h** is the last column of **V**,

$$\mathbf{A} = \mathbf{UDV}^{\mathrm{T}}$$

The DLT algorithm (cont.)

<u>Objective</u>

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix **H** such that $\mathbf{x}_i' = \mathbf{H}\mathbf{x}_i$

<u>Algorithm</u>

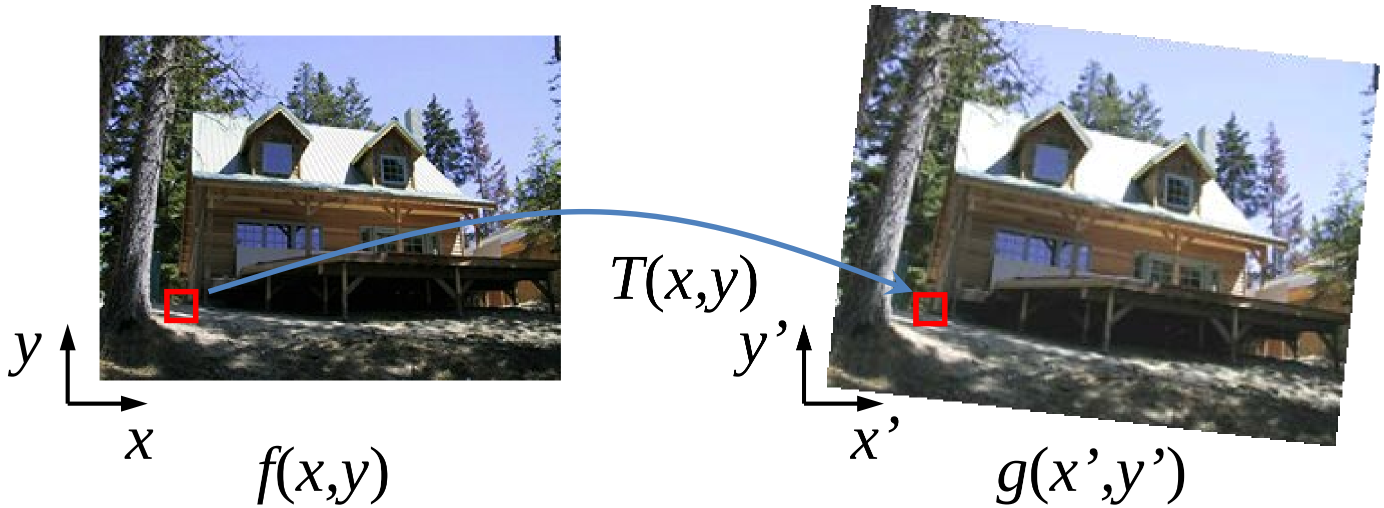
(i) For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$ compute $\mathbf{A}_i$. Usually only two first rows needed.

(ii) Assemble n 2x9 matrices $\mathbf{A}_i$ into a single 2nx9 matrix **A**

(iii) Obtain SVD of **A**. Solution for **h** is last column of **V**

(iv) Determine **H** from **h**

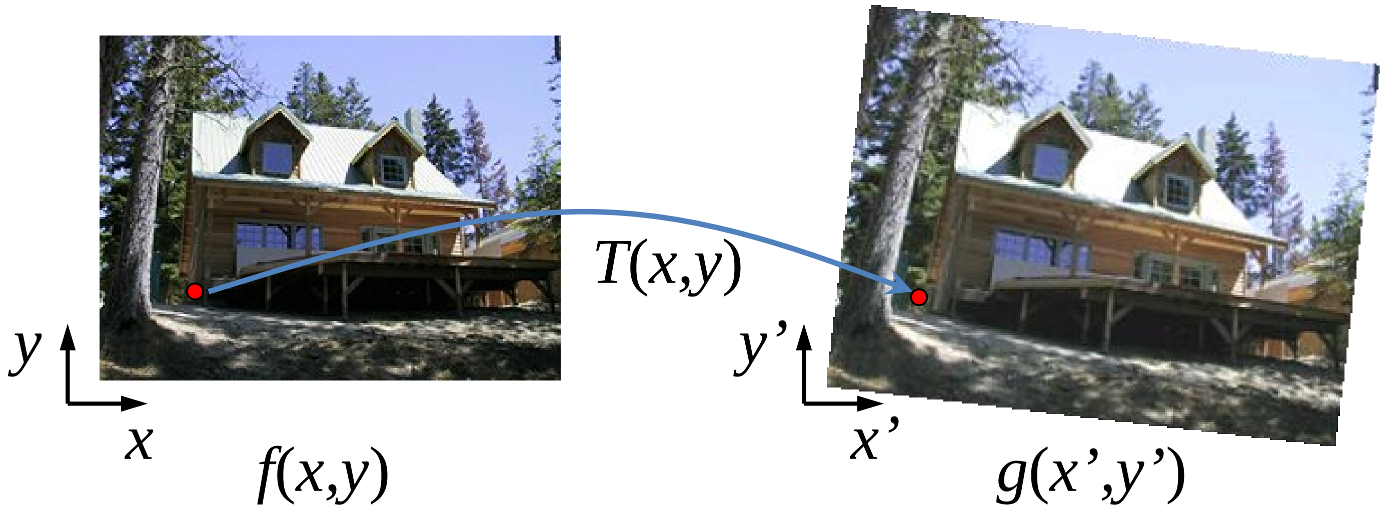
Image warping



Given a coordinate transform and a source image $f(x,y)$, how do we compute a transformed image $g(x',y') = f(T(x,y))$?

Slide credit: Alyosha Efros

Image warping

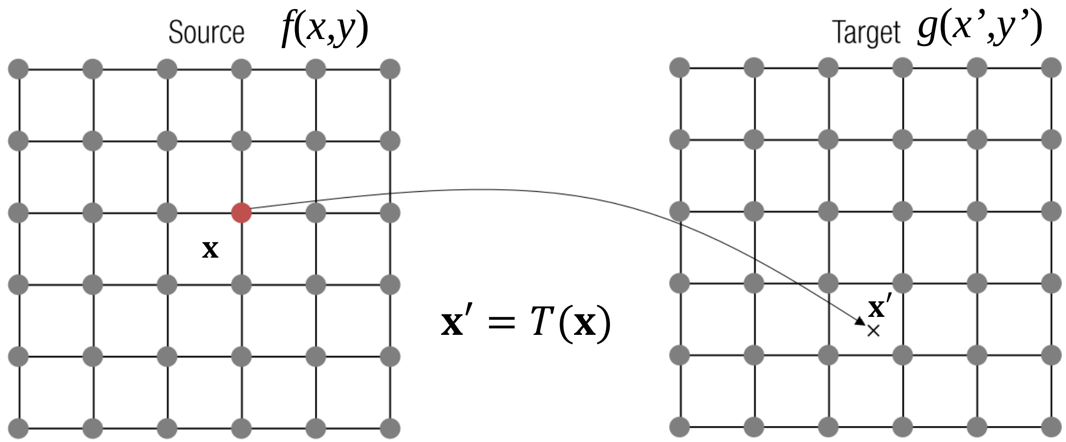


- Send each pixel $f(x,y)$ to its corresponding location $(x',y') = T(x,y)$ in the second image

Q: what if pixel lands "between" two pixels?

Slide credit: Alyosha Efros

Forward warping



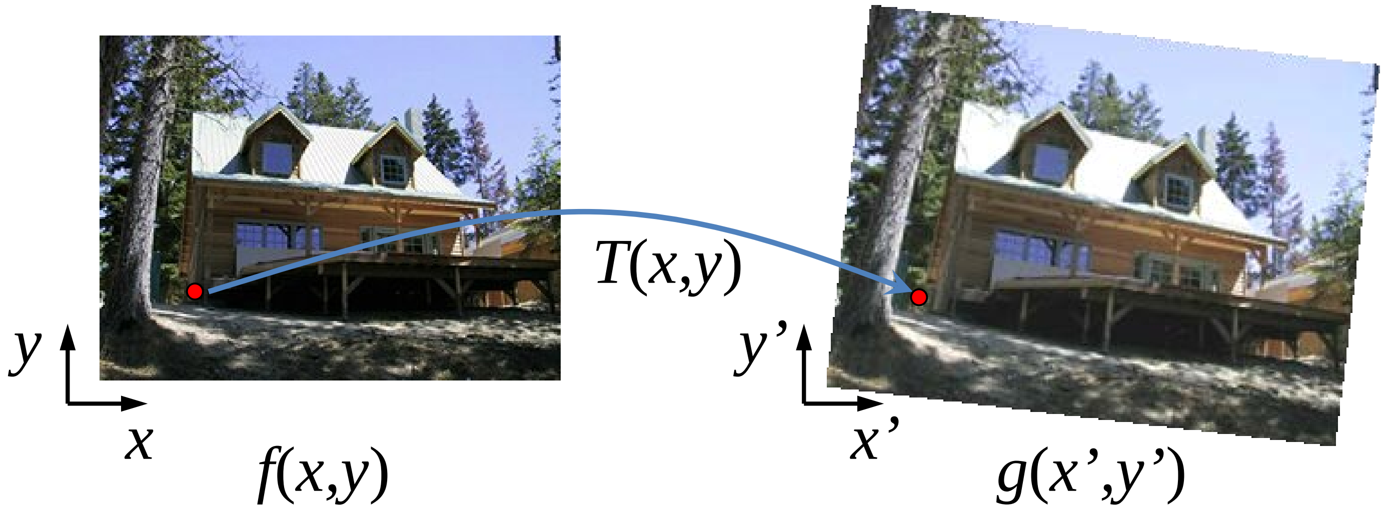
- Send each pixel $f(x,y)$ to its corresponding location $(x',y') = T(x,y)$ in the second image

Q: what if pixel lands "between" two pixels?

A: distribute color among neighboring pixels (x',y')

- – Known as "splatting"

Inverse warping



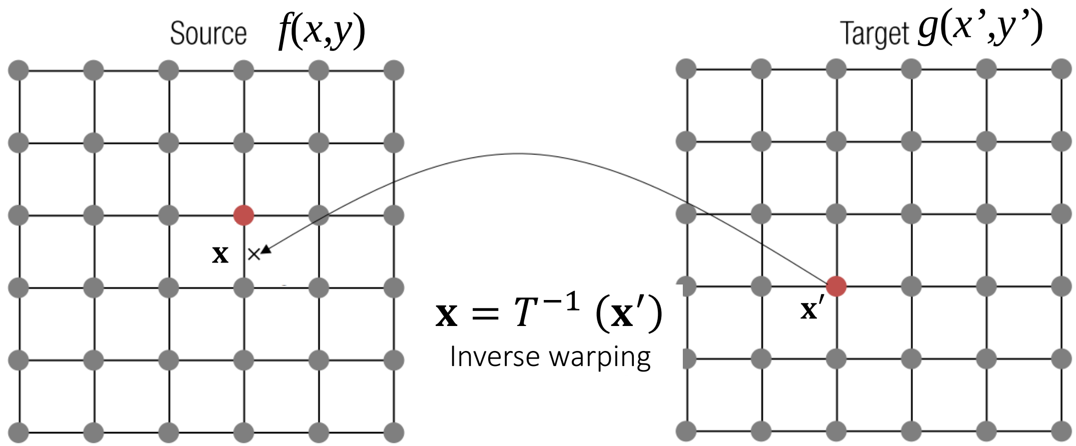
Get each pixel $g(x',y')$ from its corresponding location

$(x,y) = T^{-1}(x',y')$ in the first image

Q: what if pixel comes from "between" two pixels?

Slide credit: Alyosha Efros

Inverse (backward) warping



Get each pixel $g(x',y')$ from its corresponding location

$(x,y) = T^{-1}(x',y')$ in the first image

Q: what if pixel comes from "between" two pixels?

A: *Interpolate* color value from neighbors

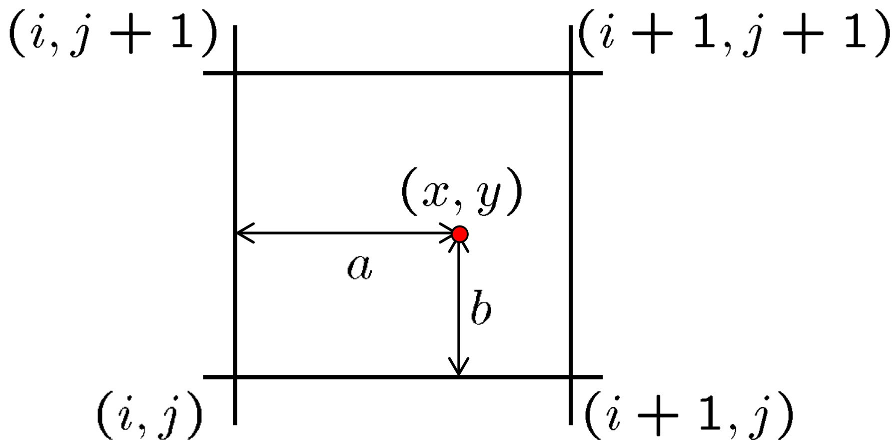
– nearest neighbor, bilinear…

```
>> help interp2
```

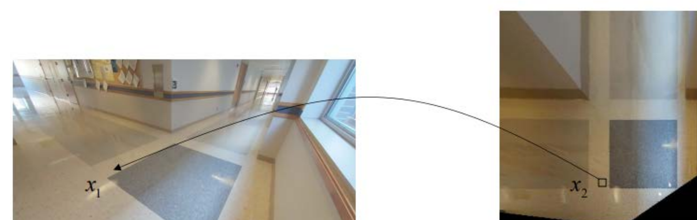
Slide credit: Alyosha Efros

Bilinear interpolation

Sampling at *f(x,y)*:



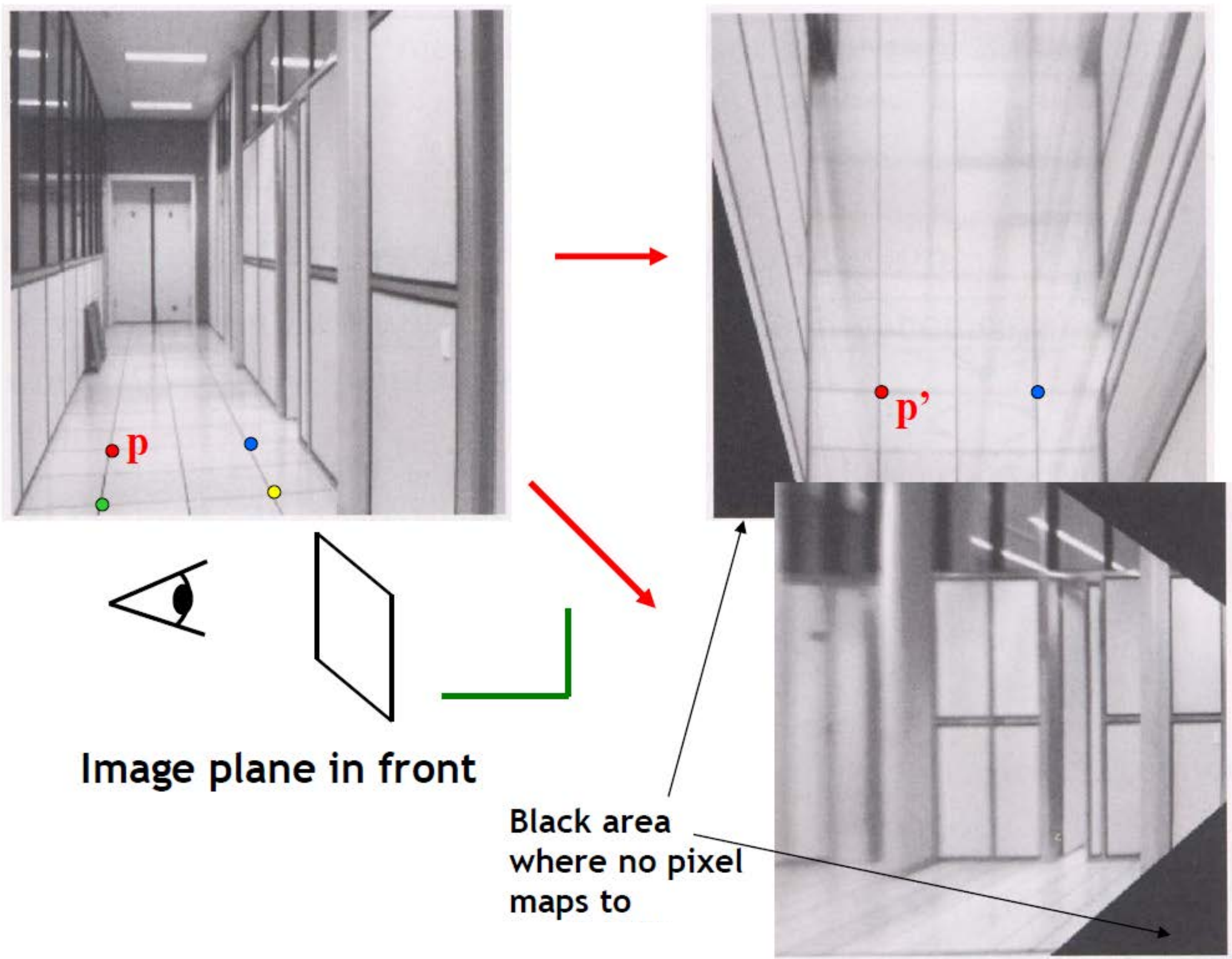
$$
\begin{aligned}
f(x,y) = \quad & (1-a)(1-b) && f[i,j] \\
& +a(1-b) && f[i+1,j] \\
& +ab && f[i+1,j+1] \\
& +(1-a)b && f[i,j+1]
\end{aligned}
$$



Backward warping

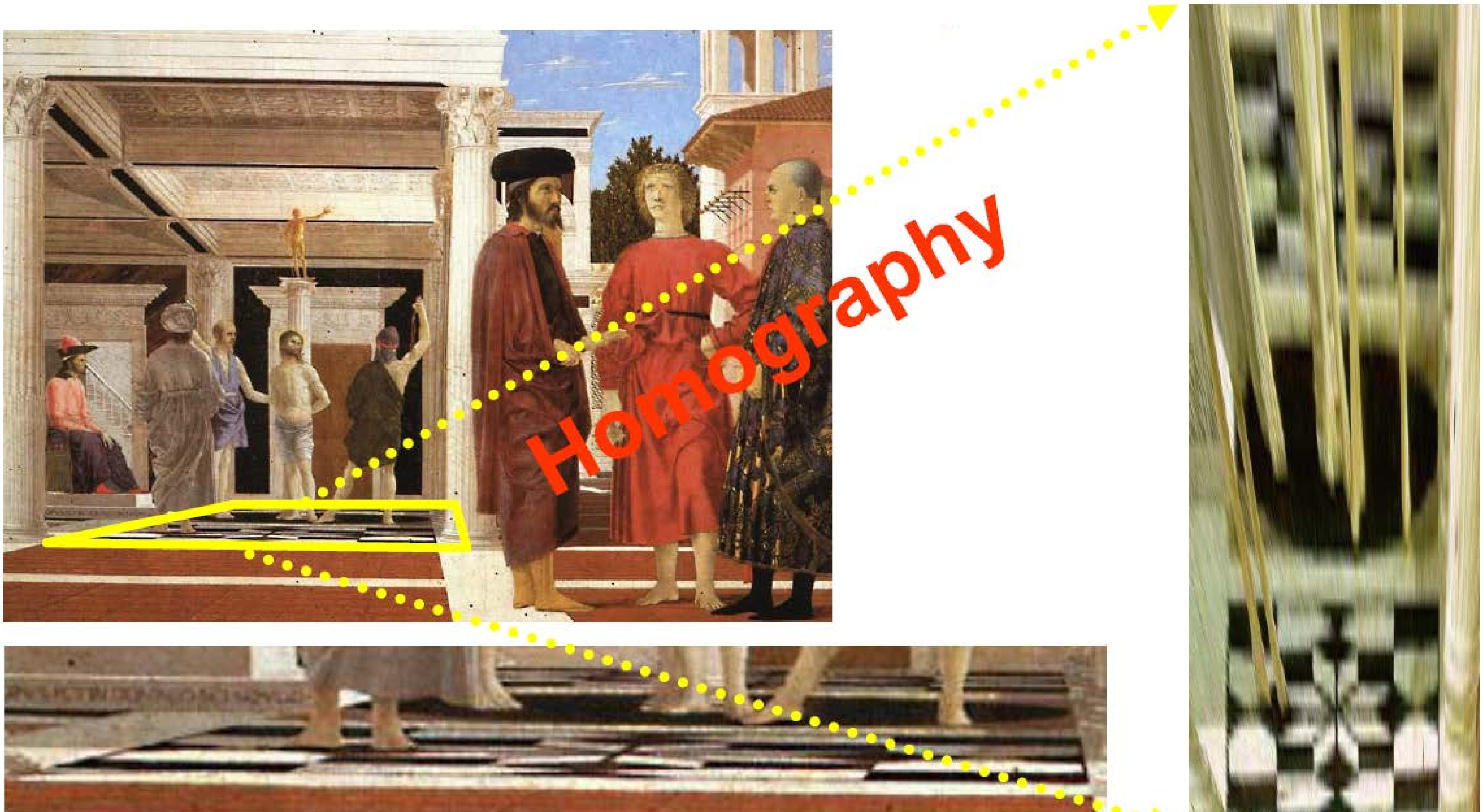
Slide credit: Alyosha Efros

Applications: Synthesizing new view images



Slide credit:Antonio Criminisi:

Applications: Synthesizing new view images

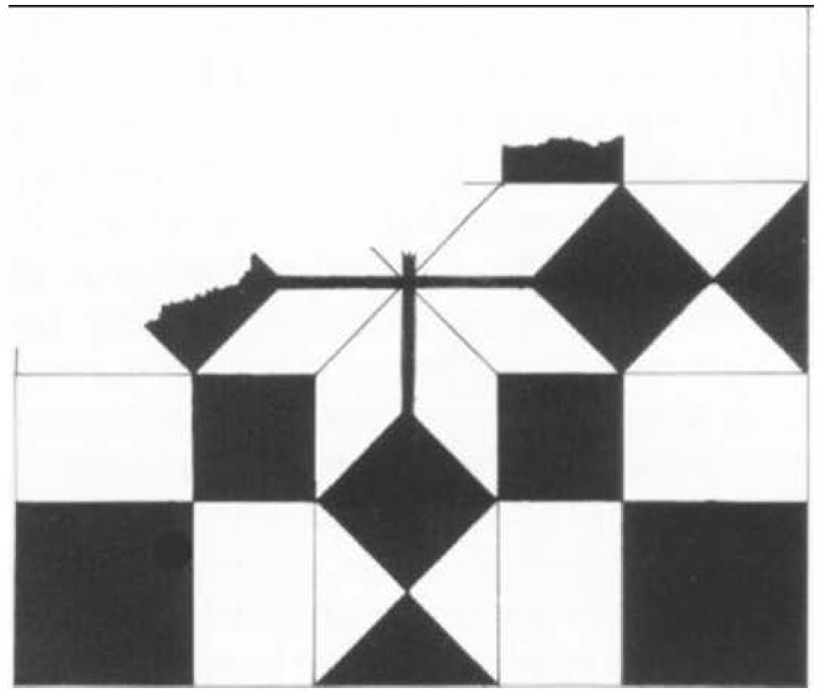


The floor (enlarged)

Slide credit:Antonio Criminisi:

Applications: Synthesizing new view images

Automatic rectification

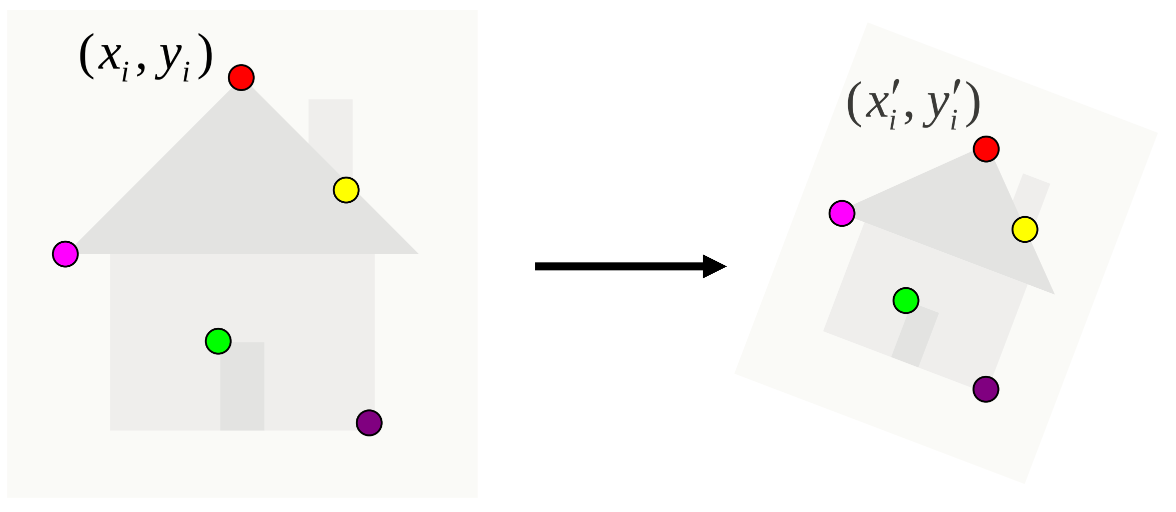


From Martin Kemp *The Science of Art*
(manual reconstruction)

Slide credit:Antonio Criminisi:

Robust feature-based alignment

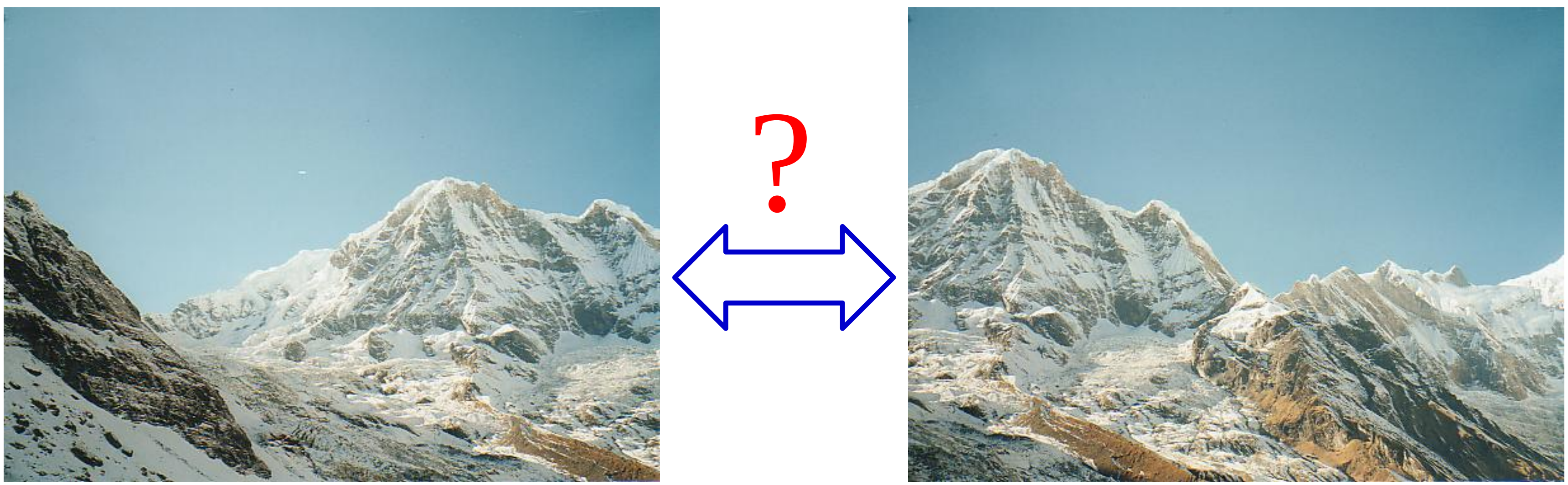
- So far, we've assumed that we are given a set of "ground-truth" correspondences between the two images we want to align
- What if we don't know the correspondences?



Slide credit: Svetlana 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?

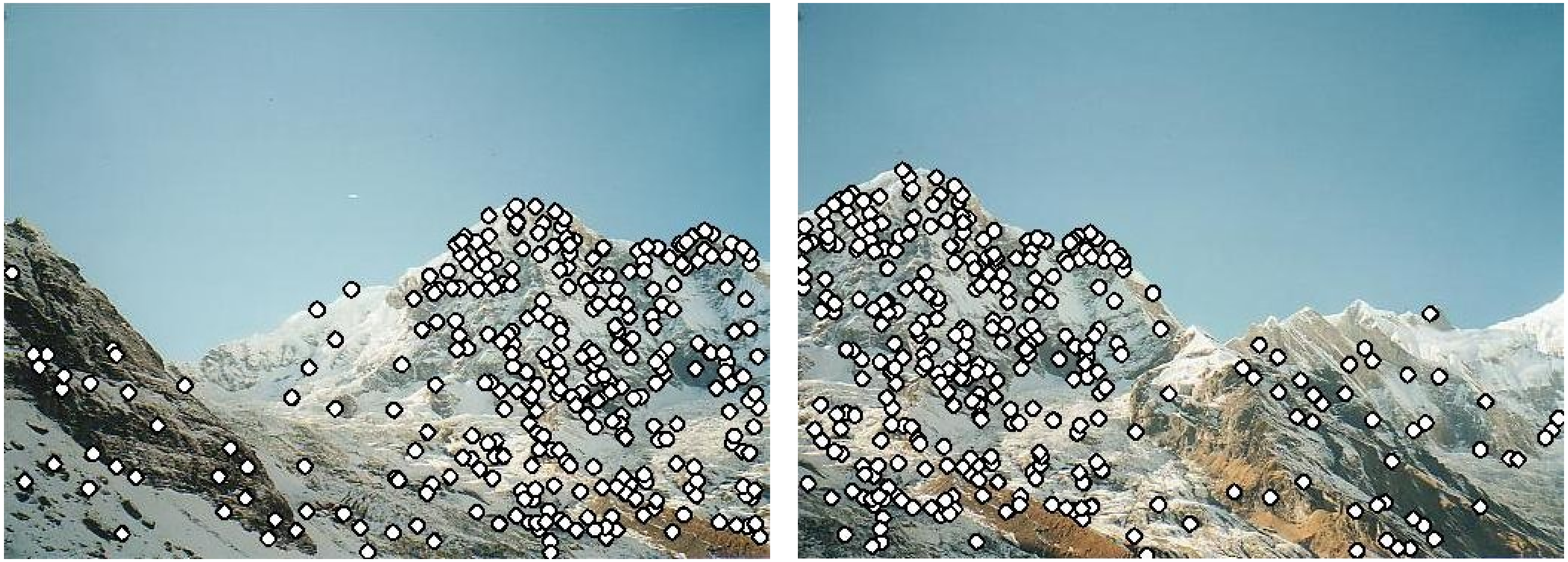


Slide credit: Svetlana Lazebnik

Robust feature-based alignment

Slide credit: Svetlana Lazebnik

Robust feature-based alignment



- Extract features

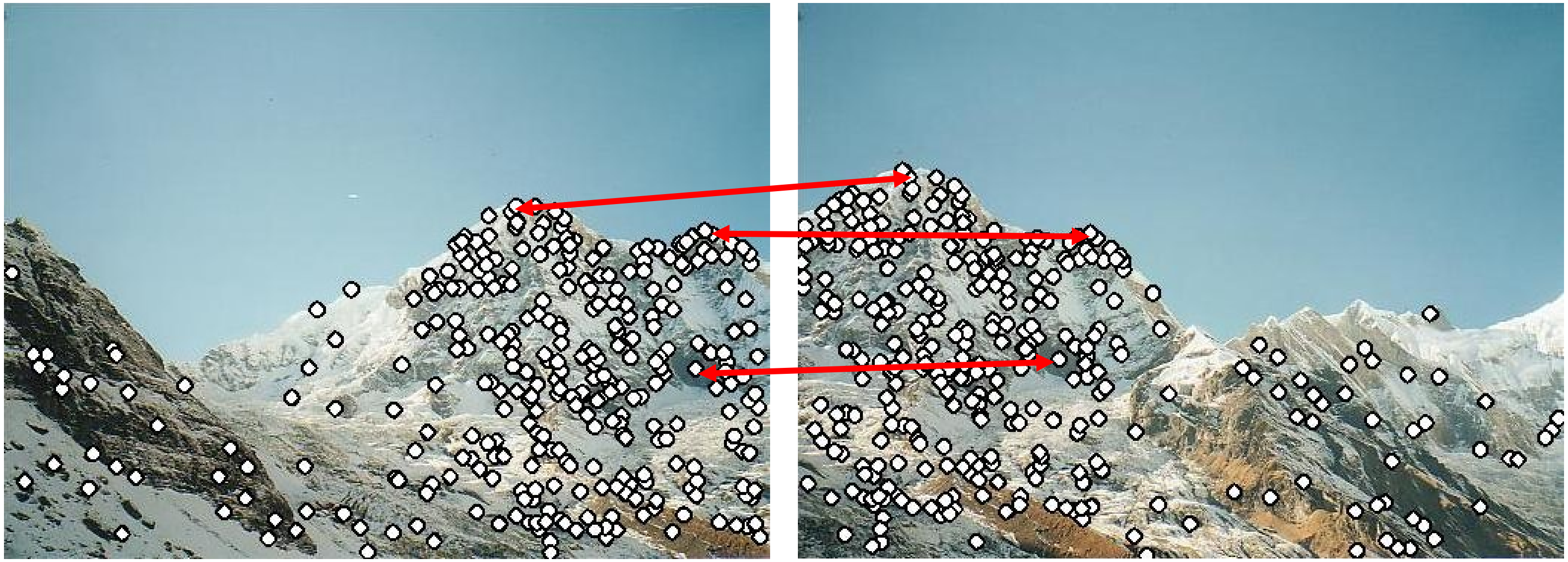
Slide credit: Svetlana Lazebnik

Robust feature-based alignment

- Extract features
- Compute *putative matches*

Slide credit: Svetlana Lazebnik

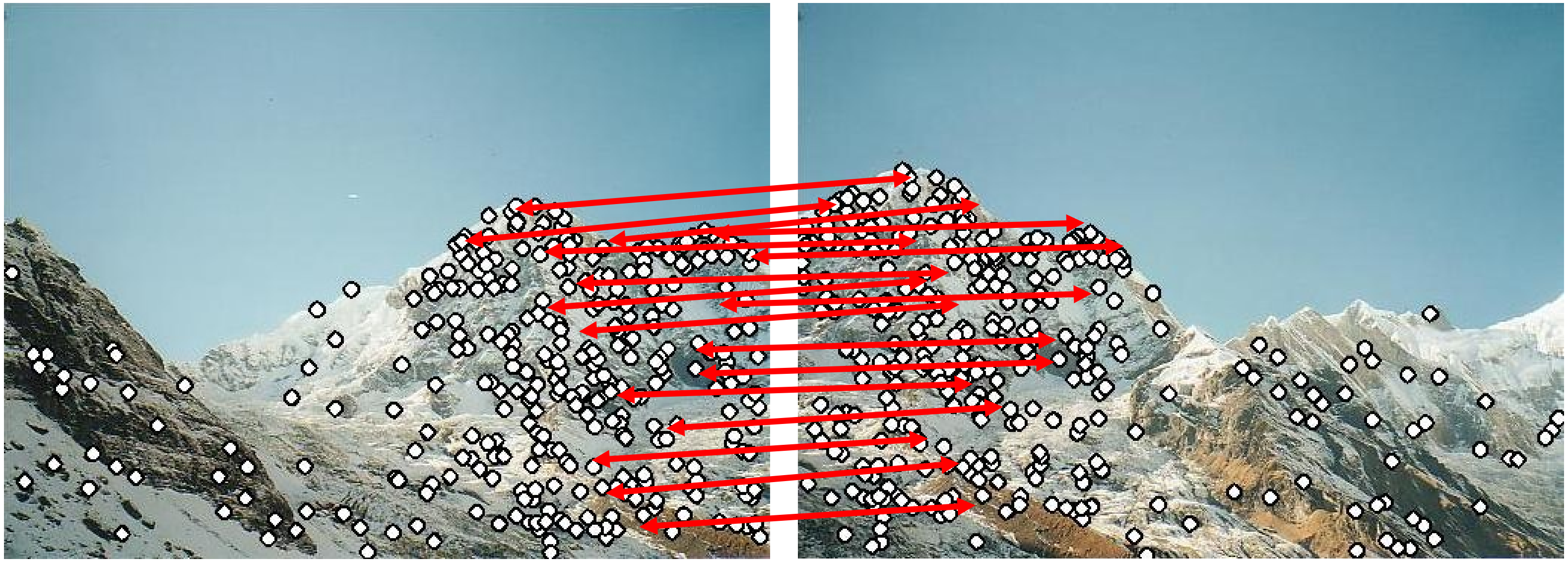
Robust feature-based alignment



- Extract features
- Compute *putative matches*
- Loop:
  - ✓ *Hypothesize* homography ***H***

Slide credit: Svetlana Lazebnik

Robust feature-based alignment



- Extract features
- Compute *putative matches*
- Loop:
  - ✓ *Hypothesize* homography ***H***
  - ✓ *Verify* homography (search for other matches consistent with ***H***)

Slide credit: Svetlana Lazebnik

Robust feature-based alignment

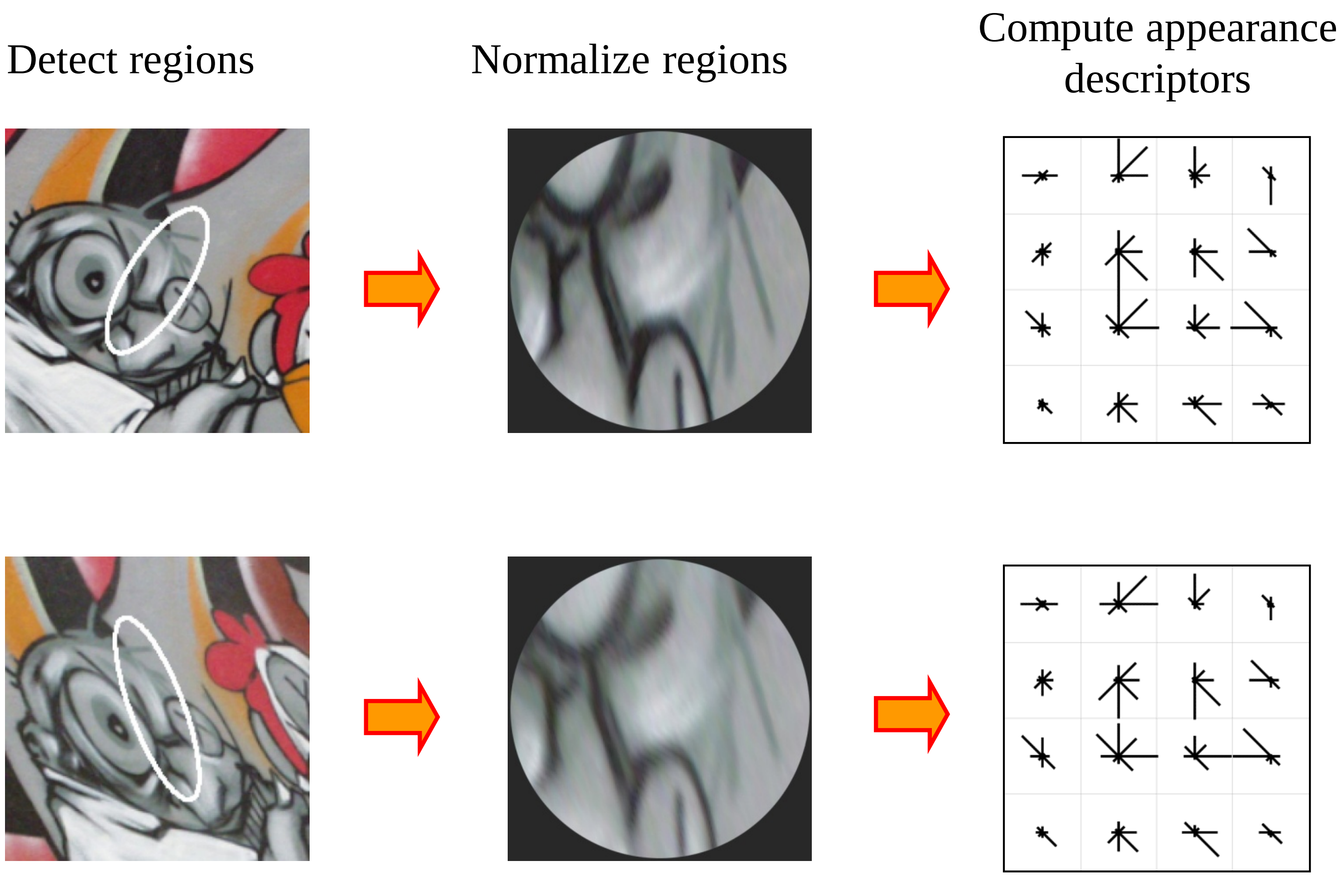
- Extract features
- Compute *putative matches*
- Loop:
  - ✓ *Hypothesize* homography ***H***
  - ✓ *Verify* homography (search for other matches consistent with ***H***)

→ RANSAC

Slide credit: Svetlana Lazebnik

Feature descriptors

- Recall: covariant detectors => invariant descriptors



Slide credit: Svetlana Lazebnik

Feature descriptors

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

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

  - – Not invariant to intensity change

  - ✓ Normalized correlation

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

  - – Invariant to affine intensity change

Slide credit: Svetlana Lazebnik

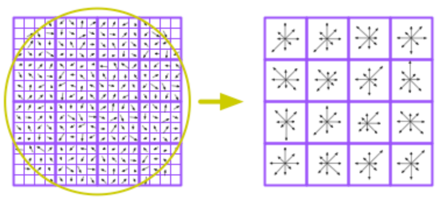
Disadvantage of intensity vectors as descriptors

- Small deformations can affect the matching score a lot

Slide credit: Svetlana 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



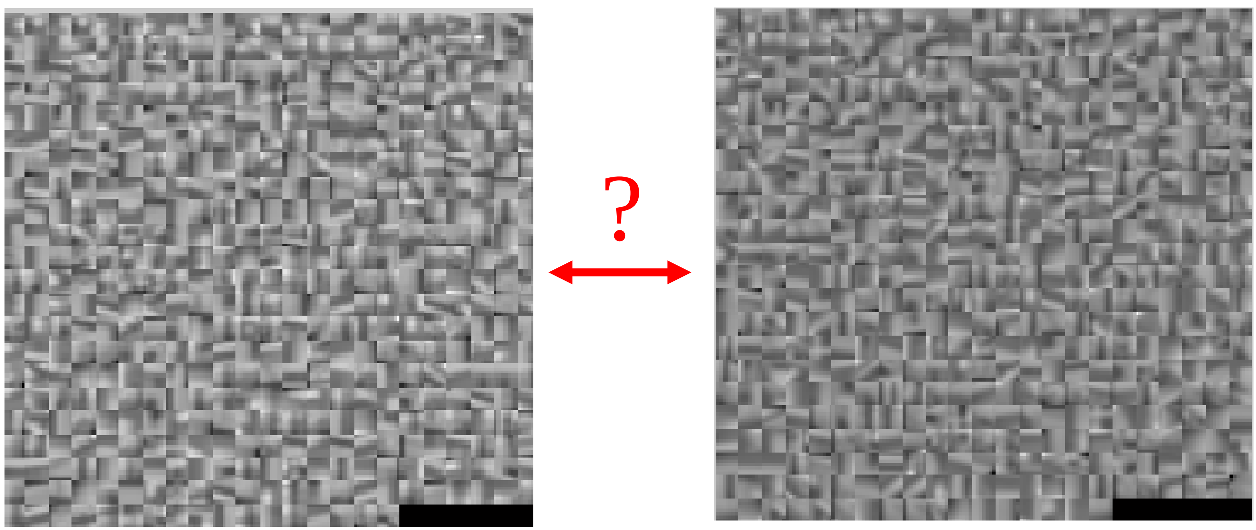
- Advantage over raw vectors of pixel values
  - ✓ Gradients less sensitive to illumination change
  - ✓ Pooling of gradients over the sub-patches achieves robustness to small shifts, but still preserves some spatial information

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

Slide credit: Svetlana Lazebnik

Feature matching

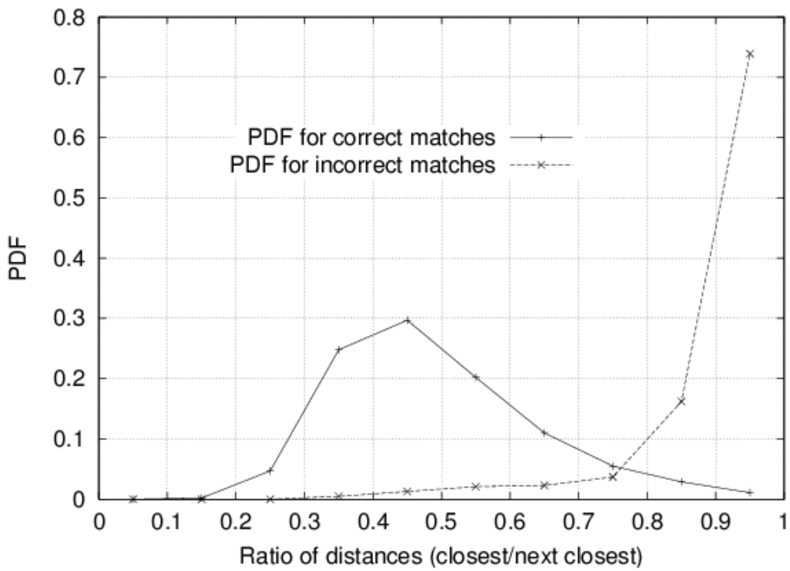
- Generating *putative matches*: for each patch in one image, find a short list of patches in the other image that could match it based solely on appearance



Slide credit: Svetlana Lazebnik

Feature space outlier rejection

- 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 credit: Svetlana 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 credit: Svetlana Lazebnik

Automatic computation of a H using RANSAC

Objective

Compute homography between two images

Algorithm

(i) **Interest points:** Compute interest points in each image

(ii) **Putative correspondences:** Compute a set of interest point matches based on some similarity measure

(iii) **RANSAC robust estimation:** Repeat for *N* samples

(a) Select 4 correspondences and compute **H**

(b) Calculate the distance $d_\perp$ for each putative match

(c) Compute the number of inliers consistent with **H** ($d_\perp < t$)

Choose **H** with most inliers

(iv) **Optimal estimation:** re-estimate **H** from all inliers by minimizing ML cost function with Levenberg-Marquardt

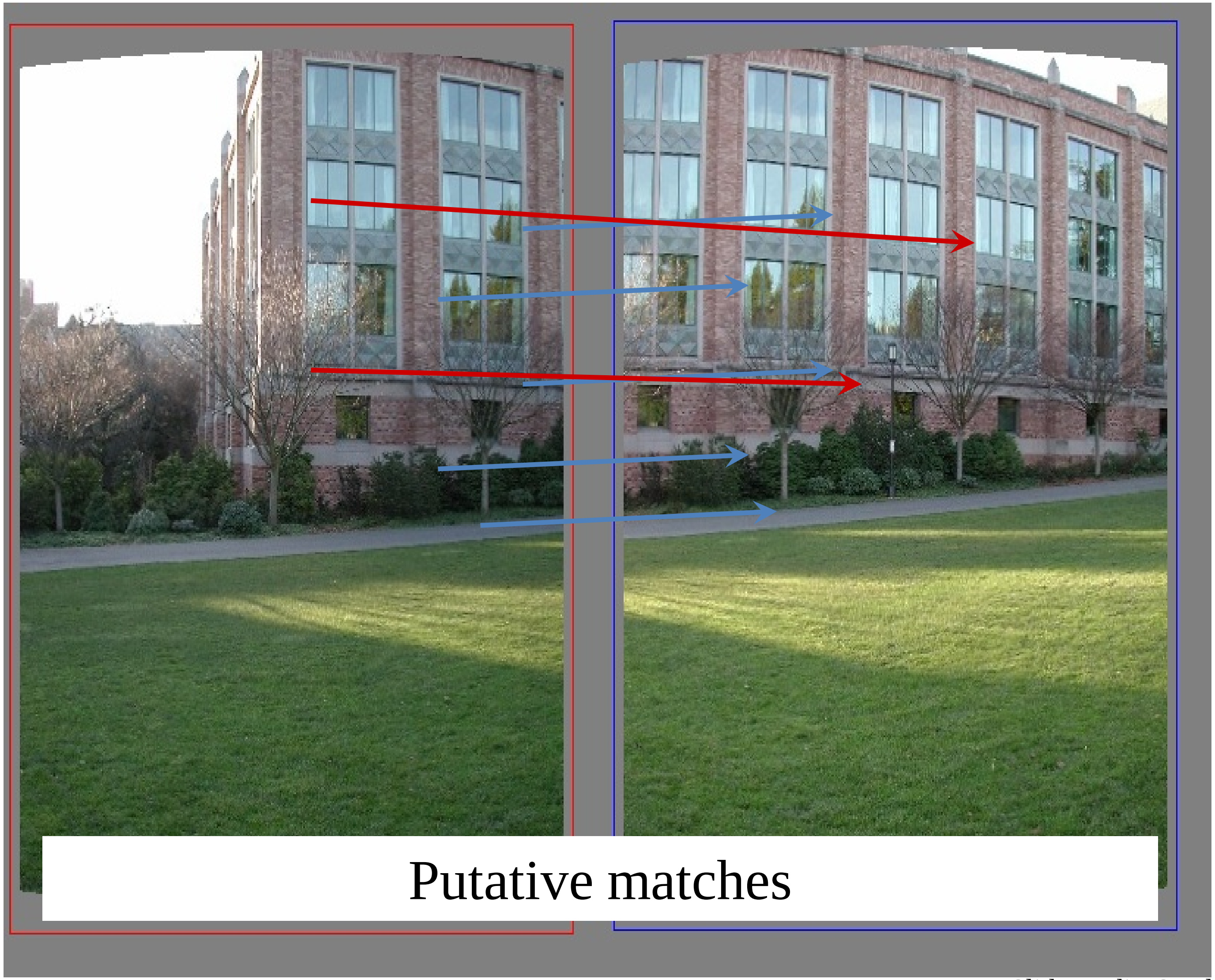
(v) Guided matching: Determine more matches using prediction by computed **H**

Optionally iterate last two steps until convergence

Automatic computation of a H using RANSAC

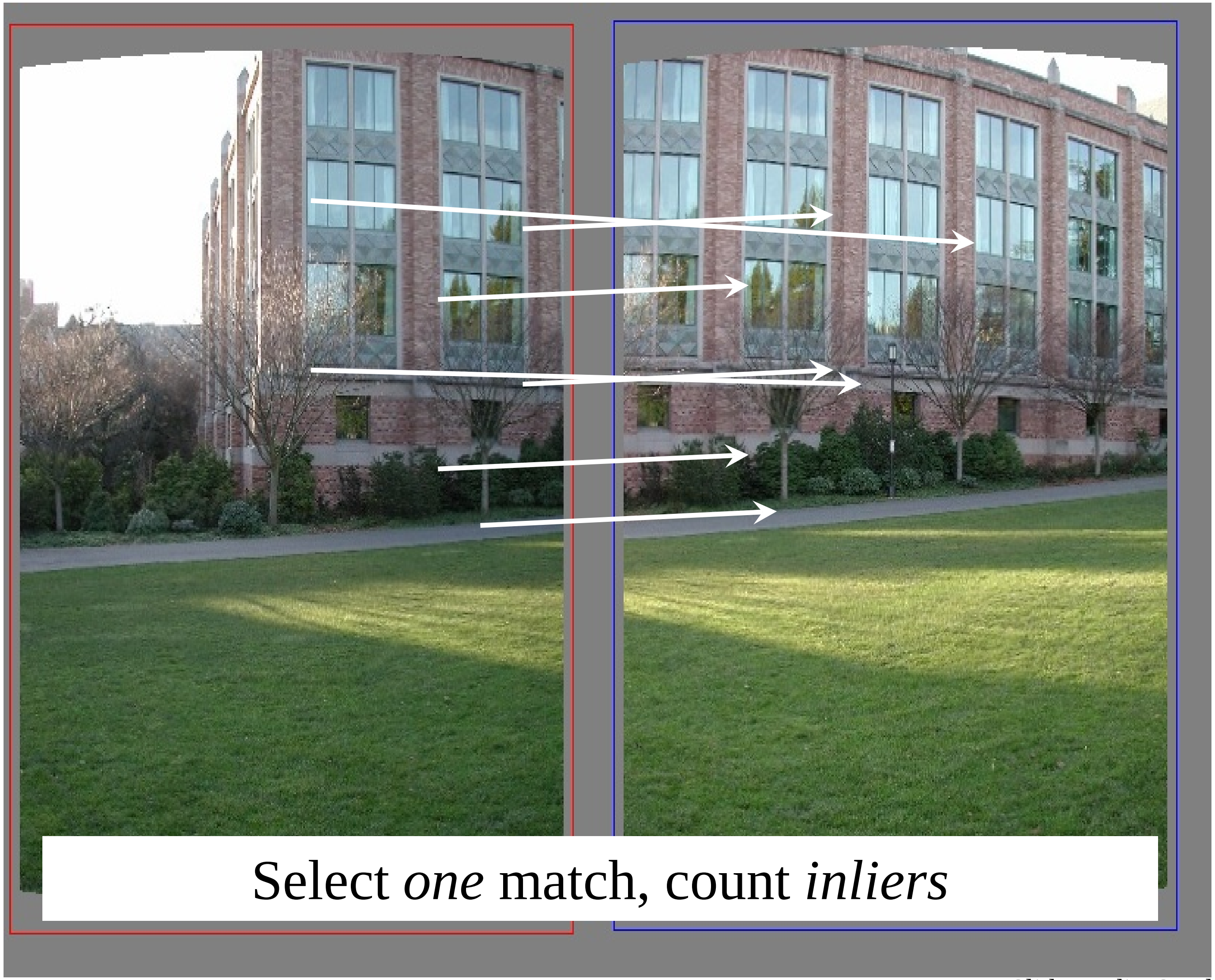
- Determining putative correspondences:
  - ✓ Select corner points in each images
  - ✓ Match points using similarity measure (SIFT, SAD, SSD, NCC, etc) symmetrically within a search region/ using nearest neighbor search
- Distance measure:
  - ✓ Symmetric transfer error $d_{\perp} = d\left(\mathbf{x}_i, \mathbf{H}^{-1}\mathbf{x}'_i\right)^2 + d\left(\mathbf{x}'_i, \mathbf{H}\mathbf{x}_i\right)^2$
  - ✓ Reprojection error $d_{\perp} = d\left(\mathbf{x}_i, \hat{\mathbf{x}}_i\right)^2 + d\left(\mathbf{x}'_i, \hat{\mathbf{x}}'_i\right)^2$, subject to $\hat{\mathbf{x}}'_i = \hat{\mathbf{H}}\hat{\mathbf{x}}_i$
- Sample selection:
  - ✓ Avoid degenerate samples (collinear ones)
  - ✓ Samples with good spatial distribution
- Robust ML estimation (cycle approach)
  - ✓ Carry out ML estimation on inliers using Levenberg-Marquardt algorithm
  - ✓ Recompute the inliers using new **H**
  - ✓ Iterate cycle until converges

RANSAC example: Translation



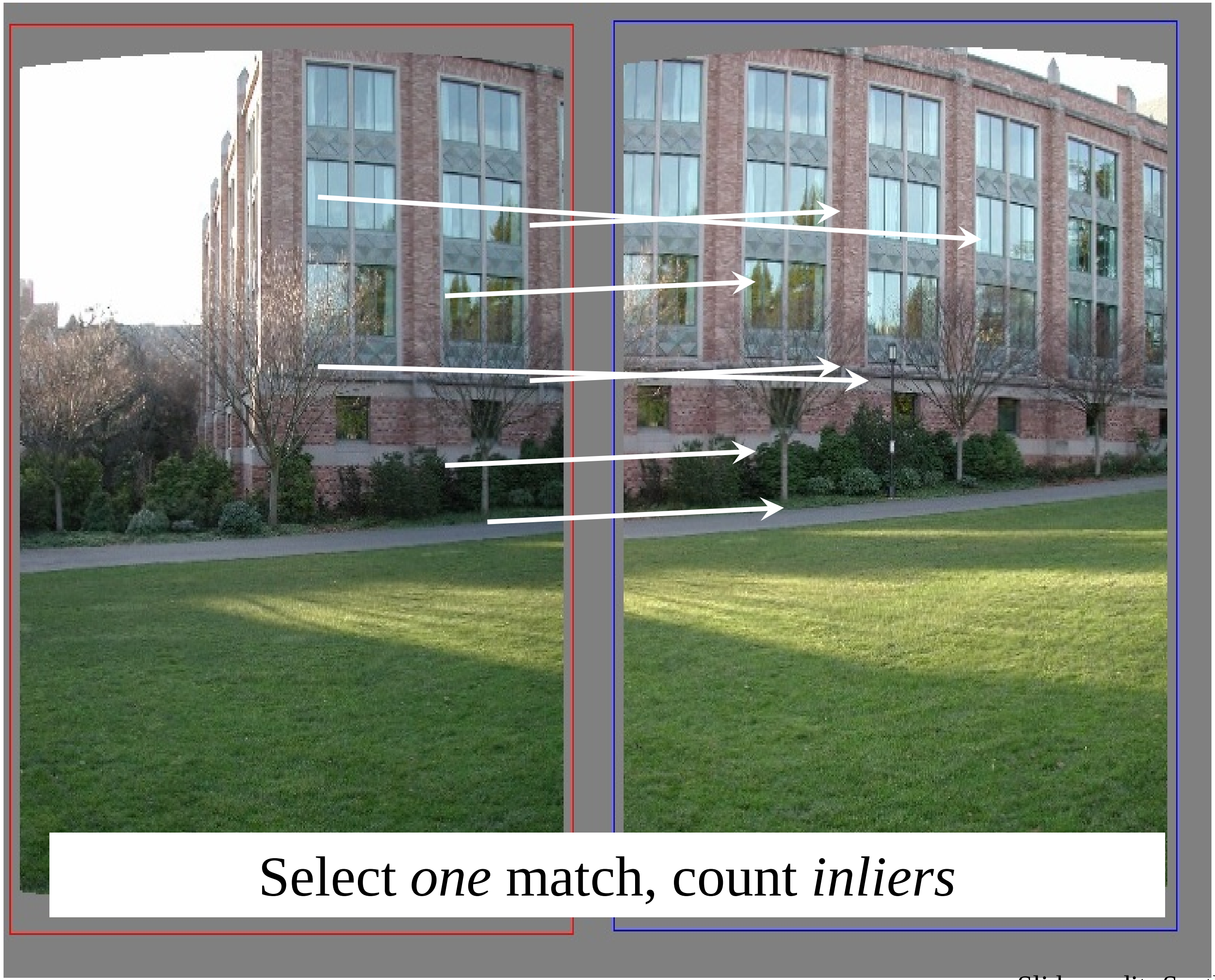
Slide credit: Svetlana Lazebnik

RANSAC example: Translation



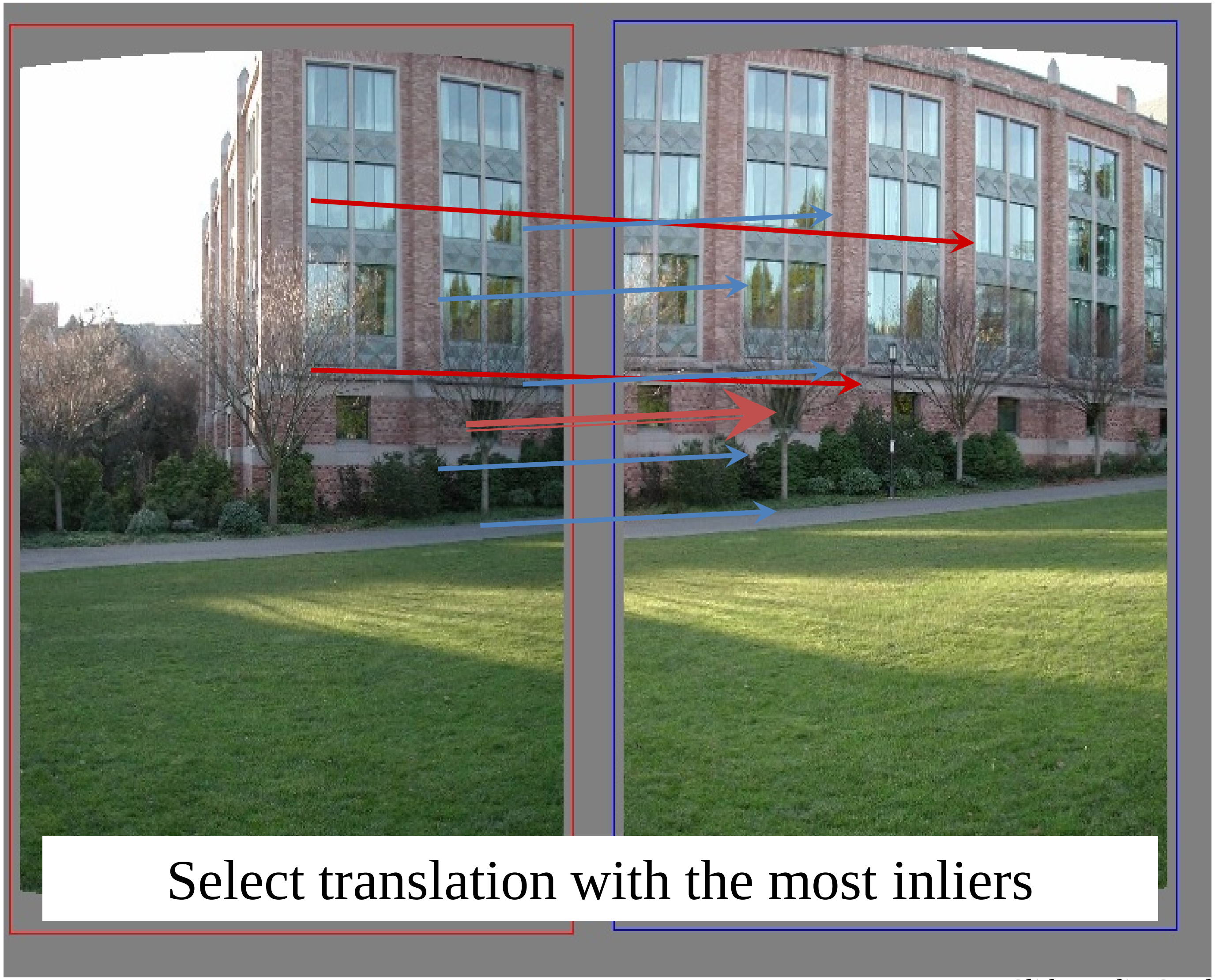
Slide credit: Svetlana Lazebnik

RANSAC example: Translation



Slide credit: Svetlana Lazebnik

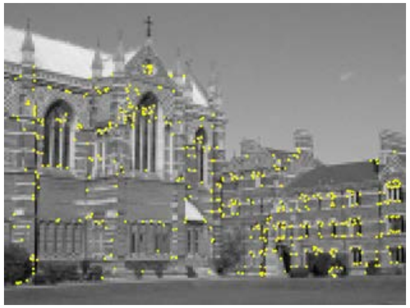
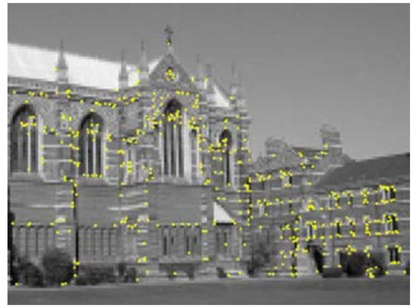
RANSAC example: Translation



Select translation with the most inliers

Slide credit: Svetlana Lazebnik

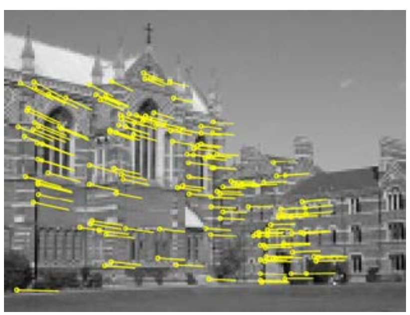
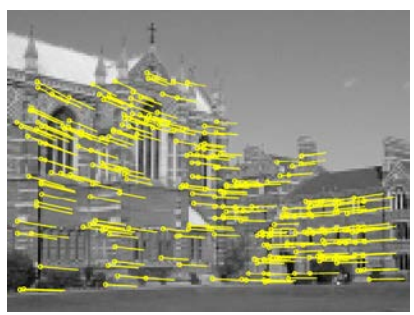
Example: robust computation

Interest points
(500/image)


Putative correspondences
(268)

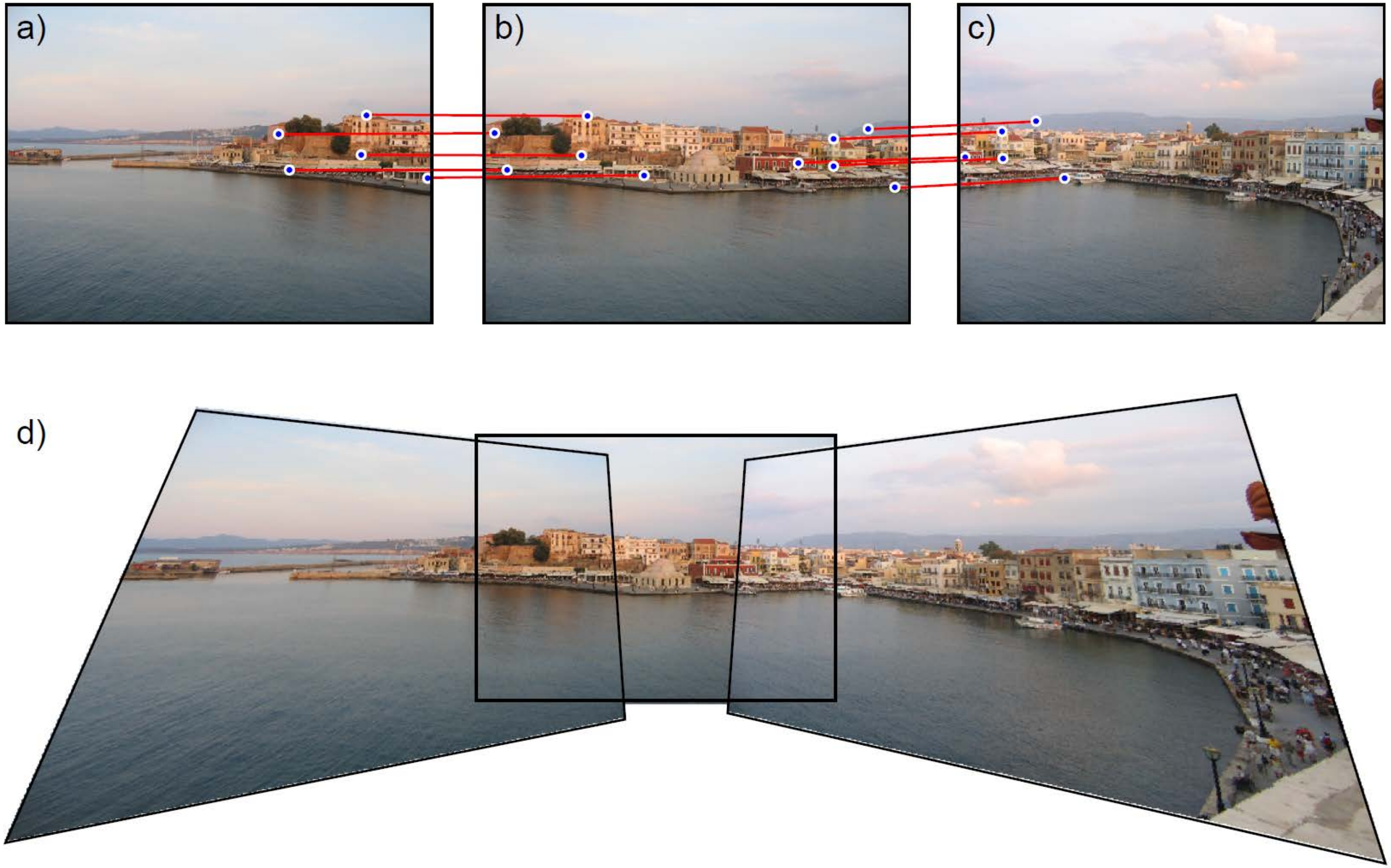
Outliers (117)

Inliers (151)

Final inliers (262)

Visual panoramas



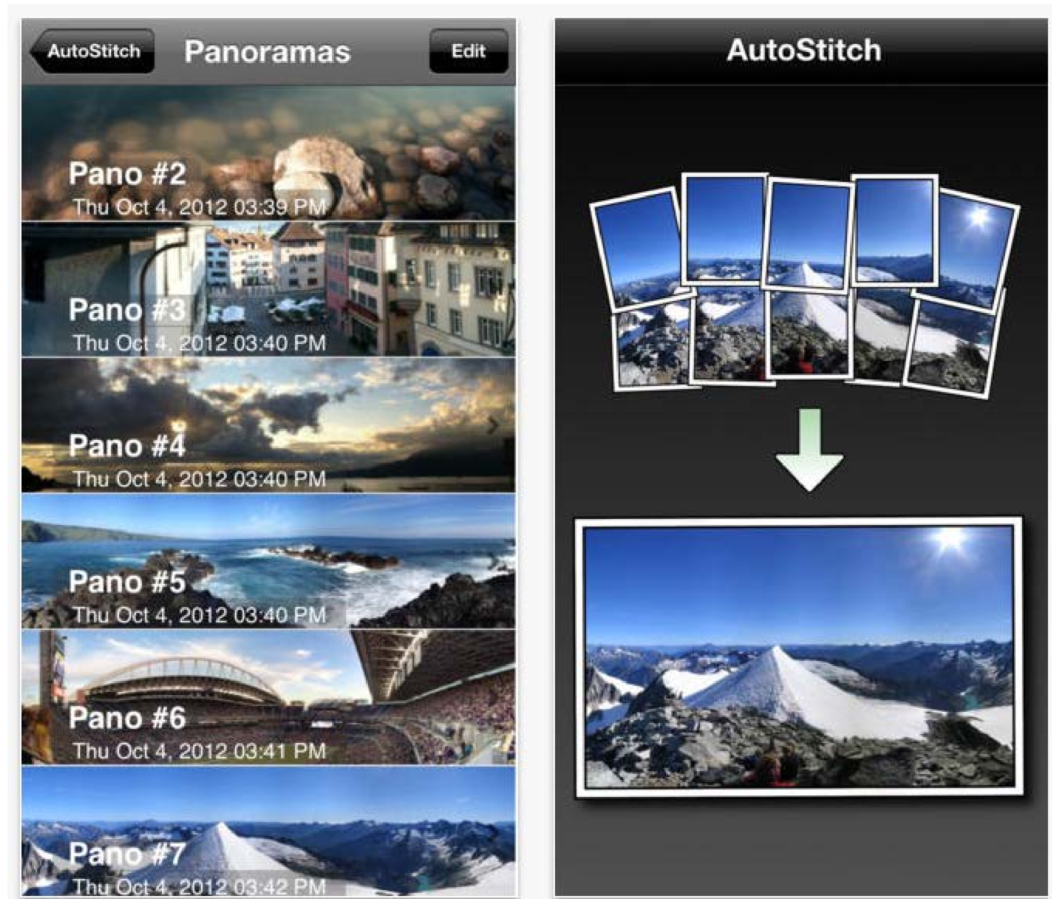
Panorama Stitching

AutoStitch Panorama

By Cloudburst Research Inc.

Open iTunes to buy and download apps.

View In iTunes

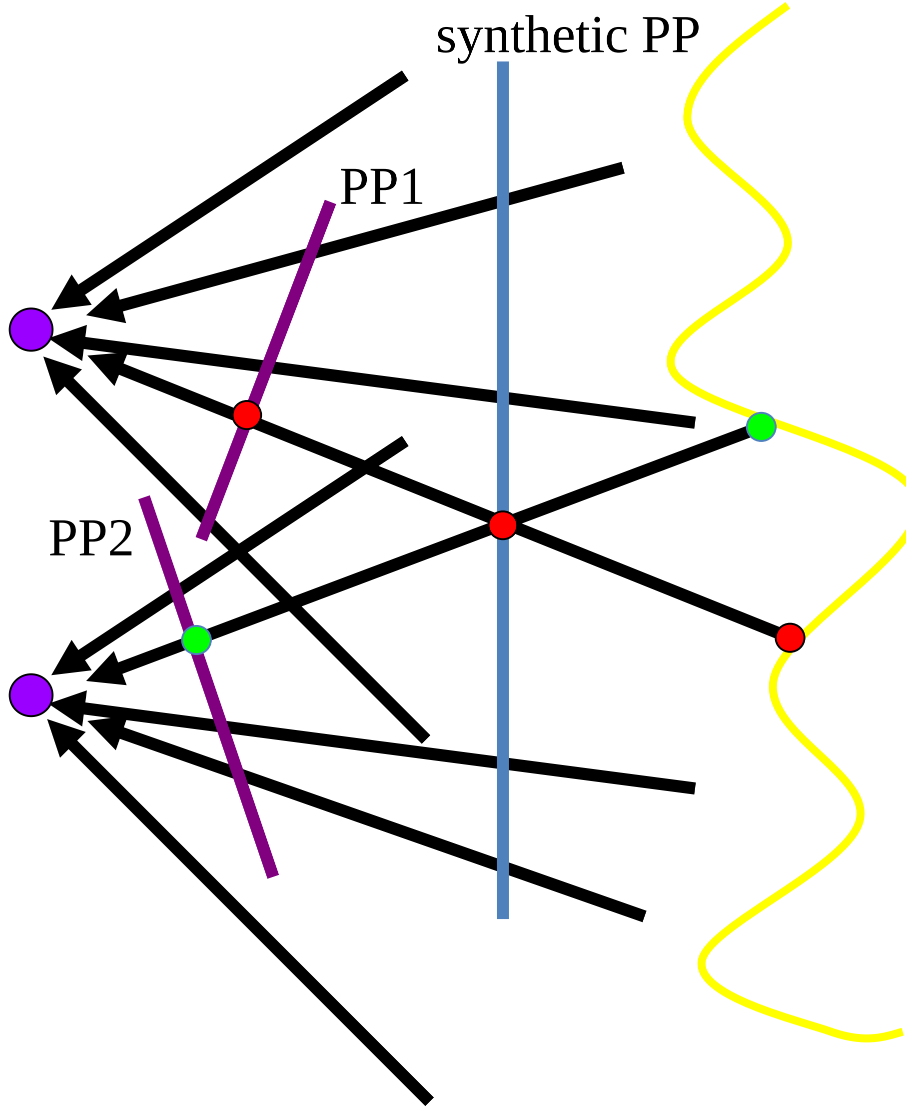


Problems with RANSAC

- In many practical situations, the percentage of outliers (incorrect putative matches) is often very high (90% or above)
- Alternative strategy: Generalized Hough Transform

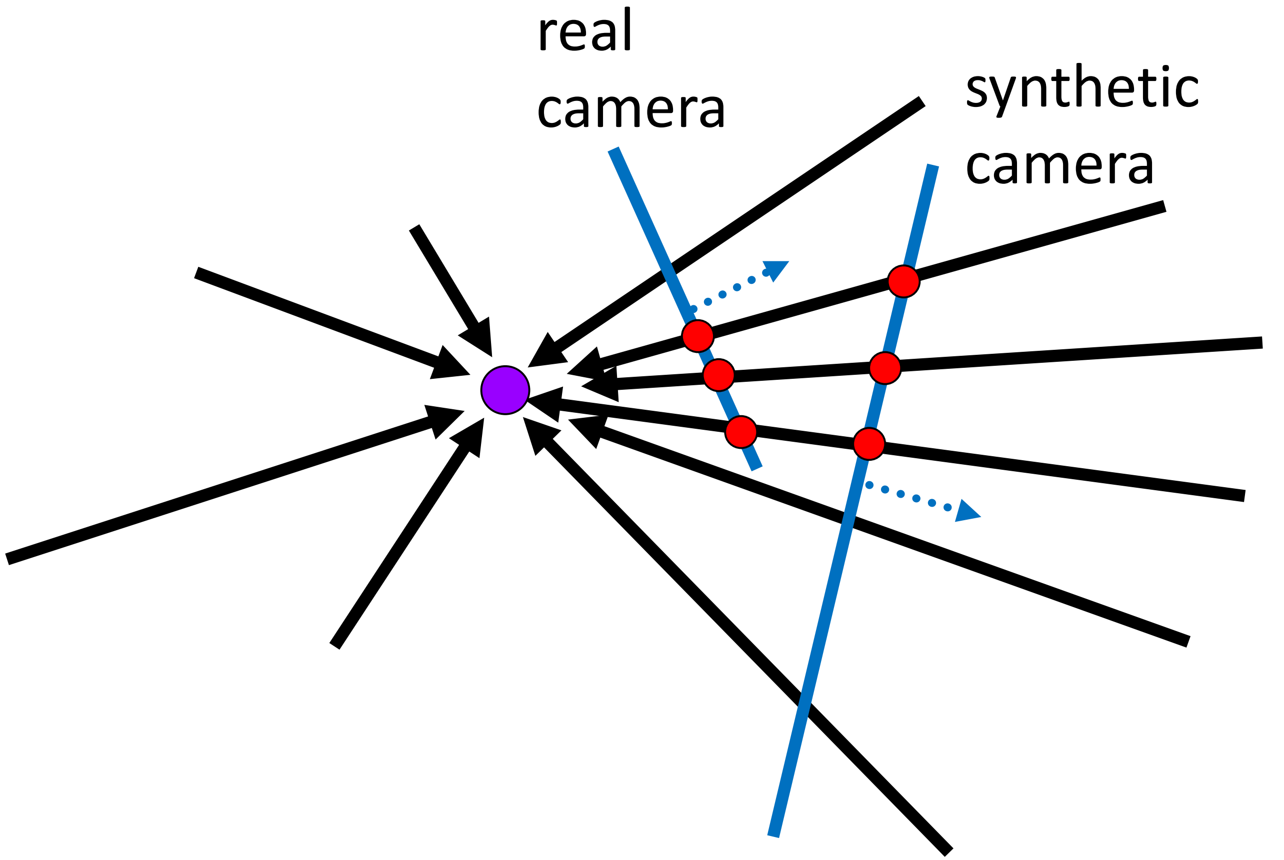
Changing camera center

- Does it still work?



Slide credit: Alyosha Efros

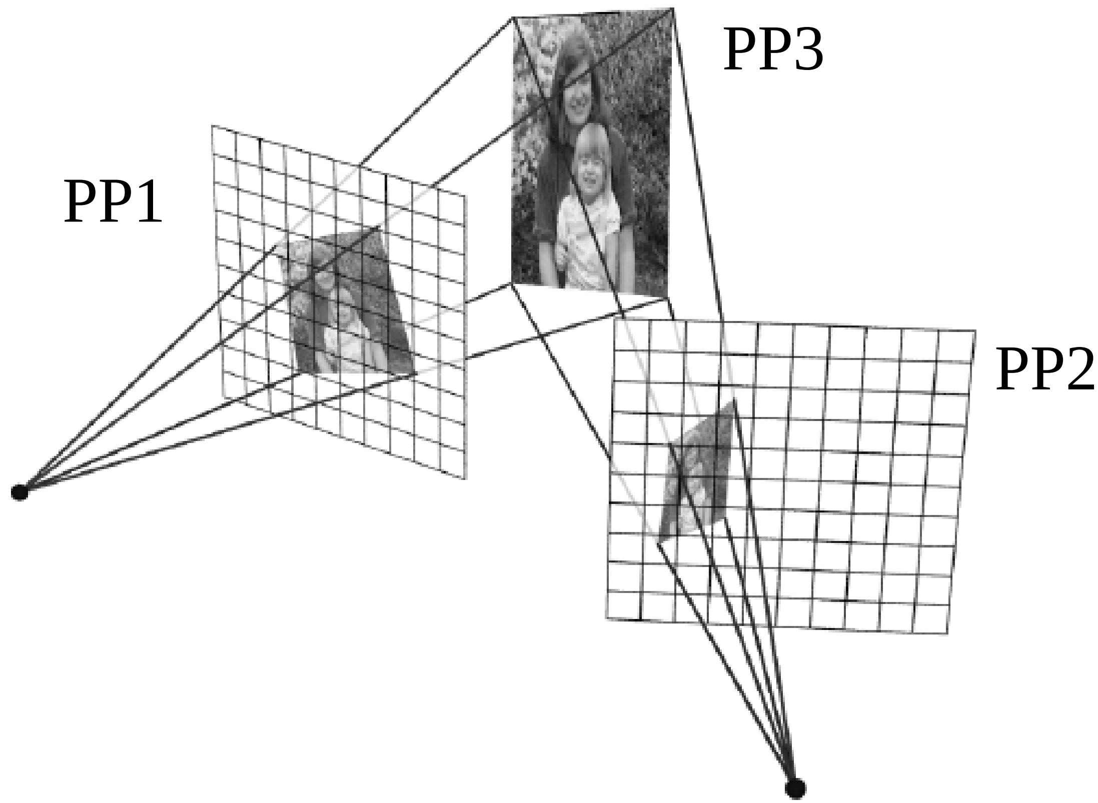
Recall: same camera center



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

Slide credit: Alyosha Efros

…Or: Planar scene (or far away)

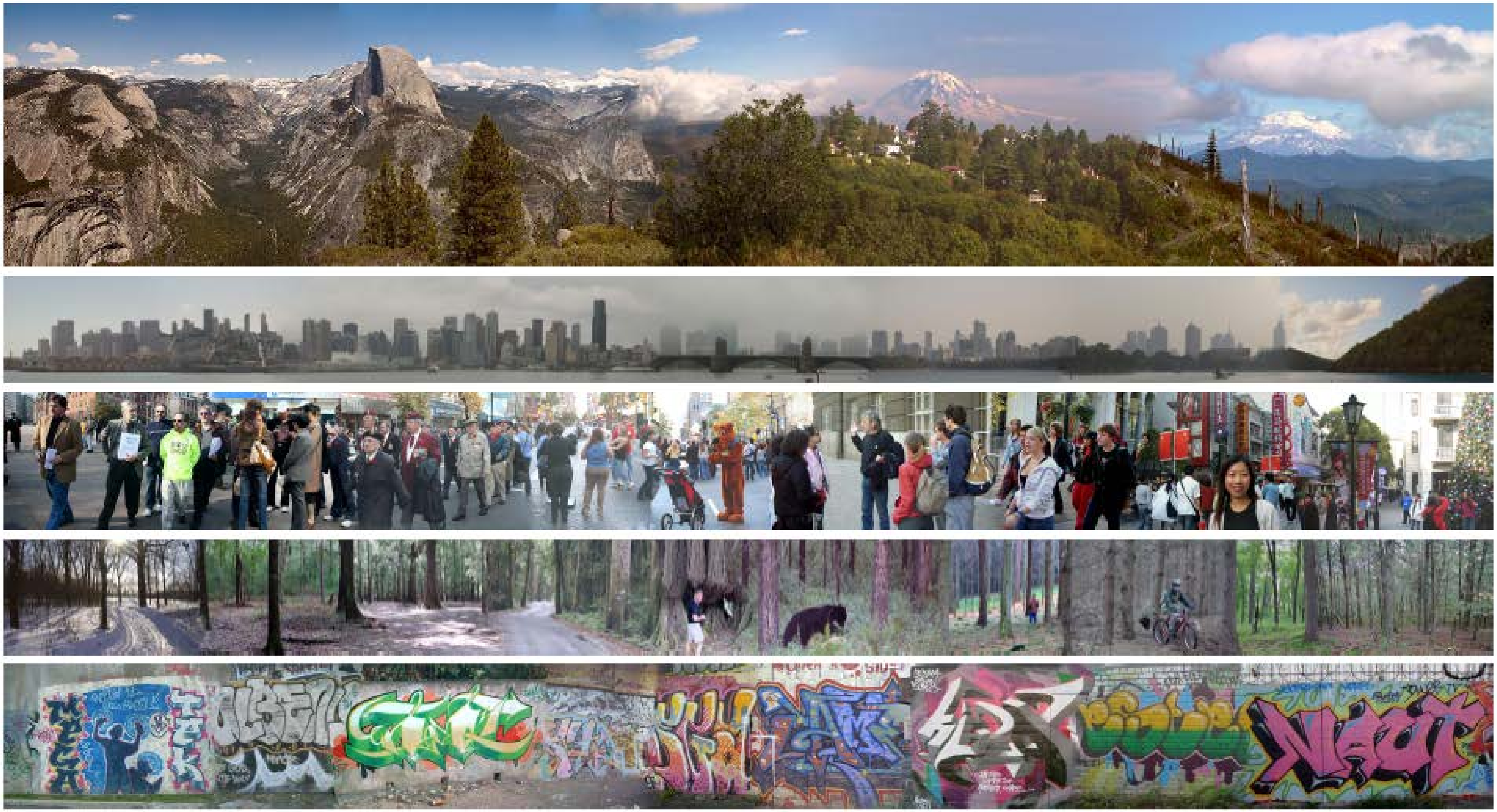


- PP3 is a projection plane of both centers of projection, so we are OK!
- This is how big aerial photographs are made

Slide credit: Alyosha Efros

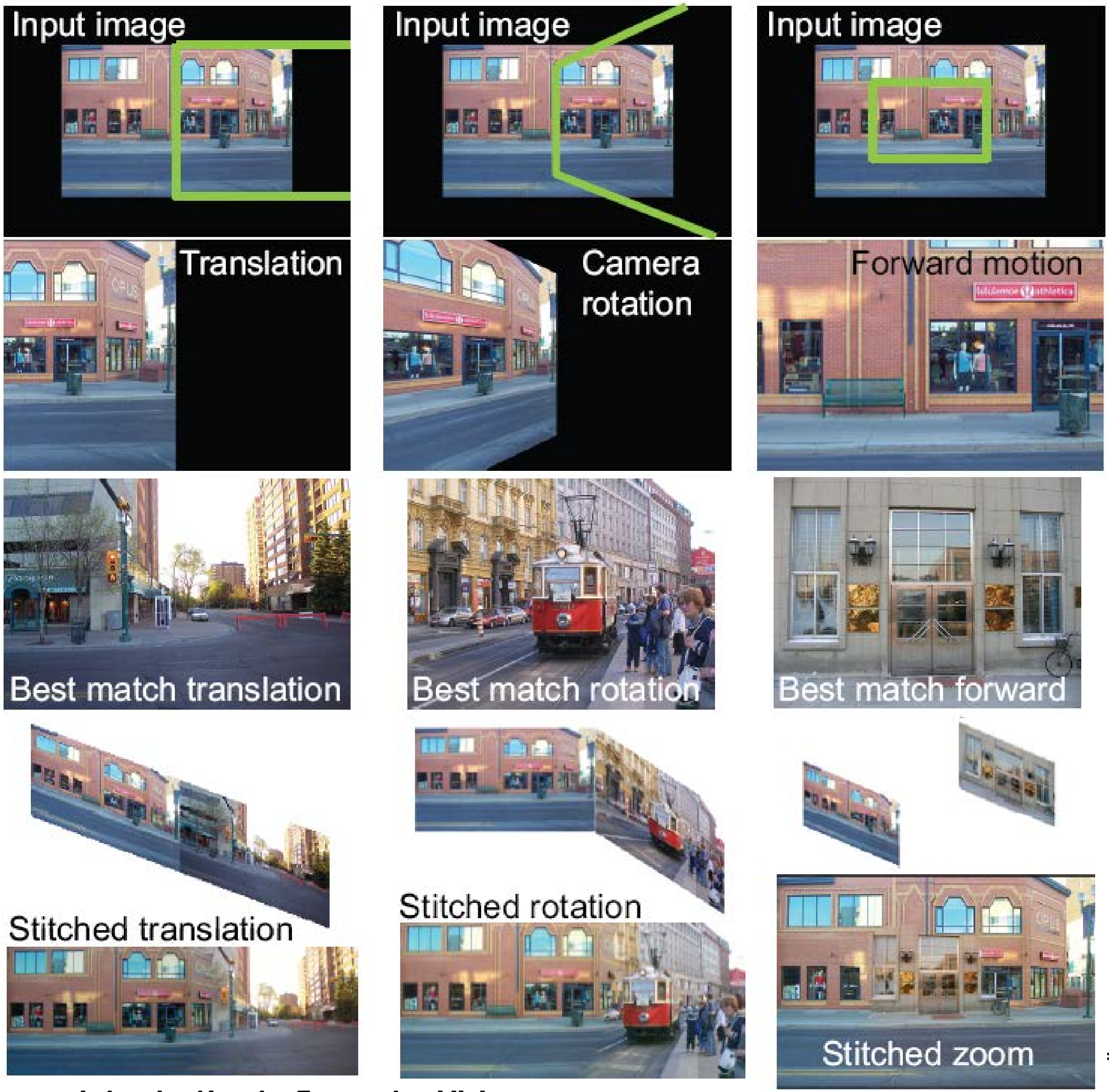

Map
Satellite
Hybrid
200 ft
100 m
©2007 Google - Terms of Use

Creating and Exploring a Large Photorealistic Virtual Space



Josef Sivic, Biliana Kaneva, Antonio Torralba, Shai Avidan and William T. Freeman, Internet Vision Workshop, CVPR 2008.
http://www.youtube.com/watch?v=E0rboU10rPo

Slide credit: Kristen Granman



Current view, and desired view in green

Synthesized view from new camera

Induced camera motion

Slide credit: Kristen Granman

Next

- Stereo
- Szeliski, Ch. 11 and F&P, Ch. 7