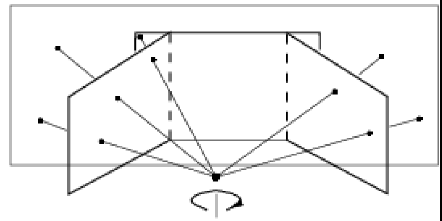


Image warping and stitching

Thurs Oct 15

Last time

- Feature-based alignment
  - 2D transformations
  - Affine fit
  - RANSAC

Robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)
  - *Verify* transformation (search for other matches consistent with *T*)

Source: L. Lazebnik

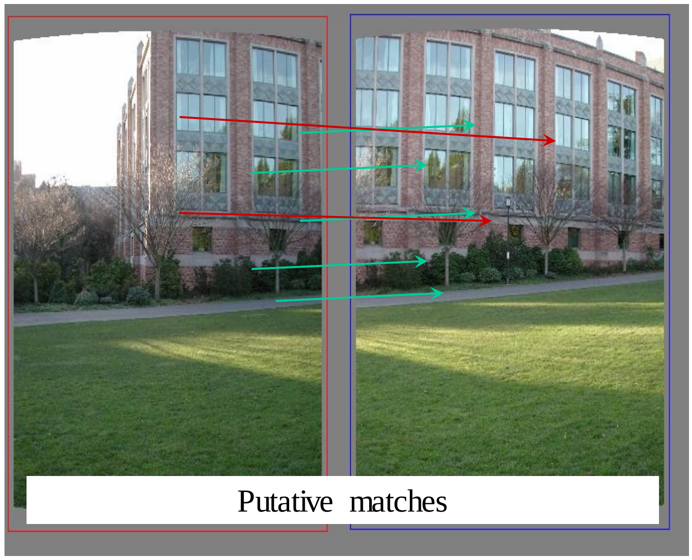
RANSAC: General form

RANSAC loop:

1. Randomly select a *seed group* of points on which to base transformation estimate (e.g., a 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 estimate of transformation on all of the inliers

Keep the transformation with the largest number of inliers

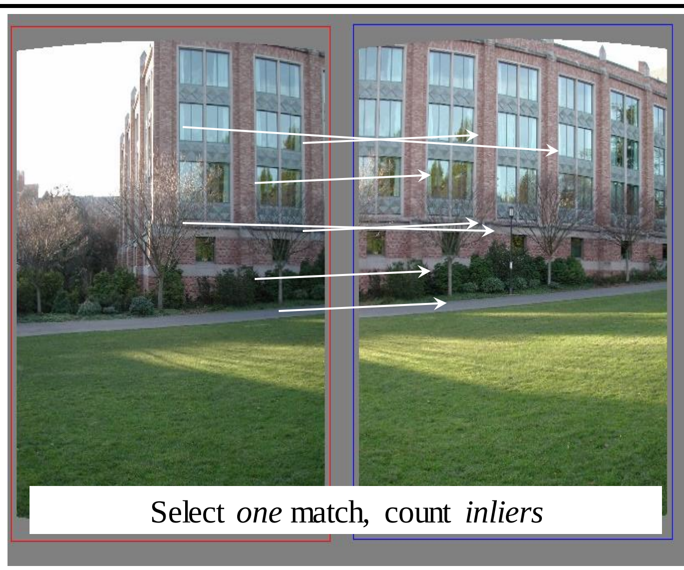
RANSAC example: Translation



Putative matches

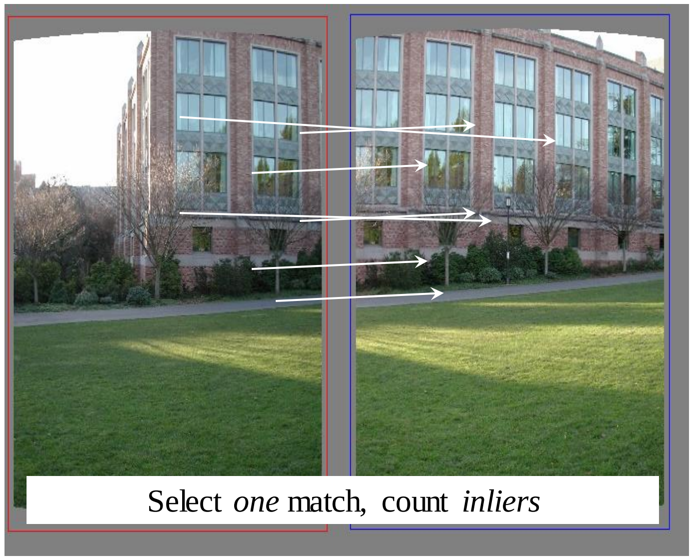
Source: Rick Szeliski

RANSAC example: Translation

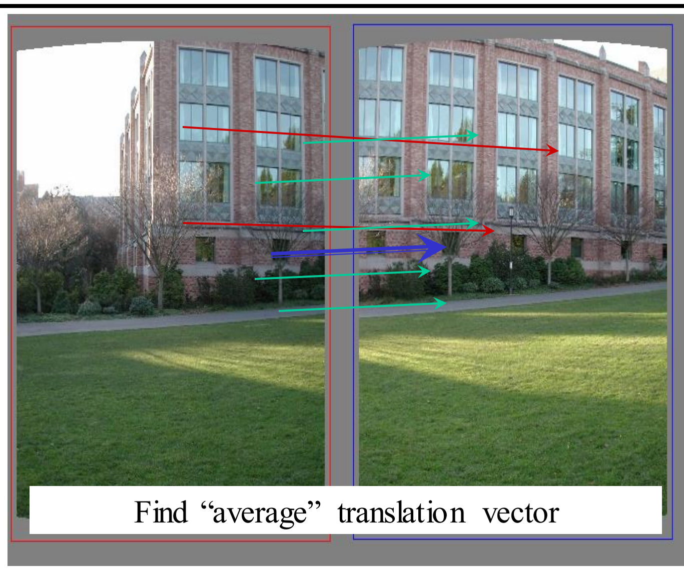


Select *one* match, count *inliers*

RANSAC example: Translation

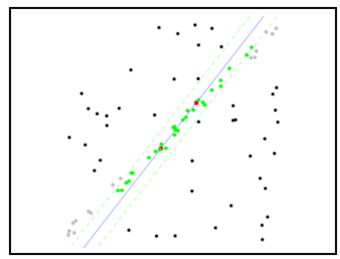


RANSAC example: Translation



RANSAC pros and cons

- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Lots of parameters to tune
  - Doesn't work well for low inlier ratios (too many iterations, or can fail completely)
  - Can't always get a good initialization of the model based on the minimum number of samples



Lana Lazebnik

Another example

Automatic scanned document rotater using Hough lines and RANSAC

Gen Hough vs RANSAC

GHT
- Single correspondence -> vote for all consistent parameters
- Represents uncertainty in the model parameter space
- Linear complexity in number of correspondences and number of voting cells; beyond 4D vote space impractical
- Can handle high outlier ratio

RANSAC
- Minimal subset of correspondences to estimate model -> count inliers
- Represents uncertainty in image space
- Must search all data points to check for inliers each iteration
- Scales better to high-d parameter spaces

Kristen Grauman

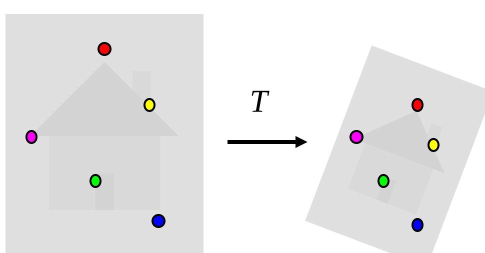
Today

- Image mosaics
  - Fitting a 2D transformation
    - Affine, Homography
  - 2D image warping
  - Computing an image mosaic

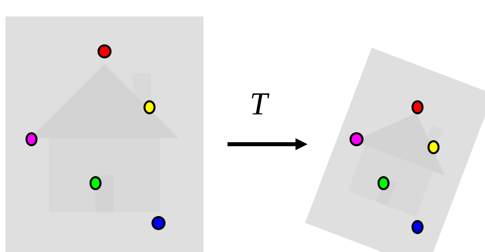
Mosaics

Obtain a wider angle view by combining multiple images.

Main questions



Alignment: Given two images, what is the transformation between them?



Warping: Given a source image and a transformation, what does the transformed output look like?

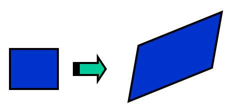
2D Affine Transformations

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

Affine transformations are combinations of …

- Linear transformations, and
- Translations

Parallel lines remain parallel



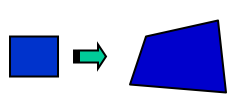
Projective Transformations

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

Projective transformations:

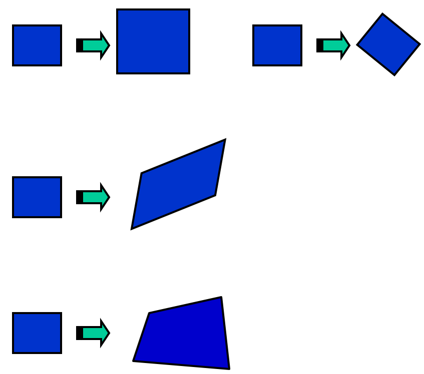
- Affine transformations, and
- Projective warps

Parallel lines do not necessarily remain parallel



2D transformation models

- Similarity (translation, scale, rotation)
- Affine
- Projective (homography)



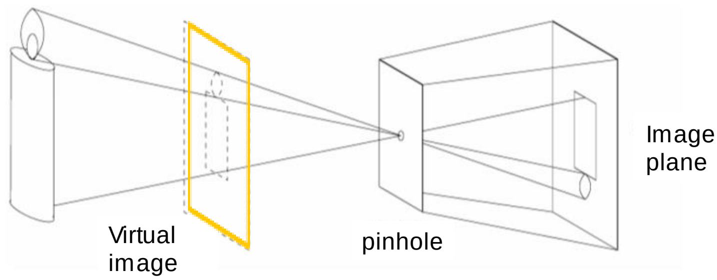
How to stitch together a panorama (a.k.a. mosaic)?

- Basic Procedure
  - Take a sequence of images from the same position
    - Rotate the camera about its optical center
  - Compute transformation between second image and first
  - Transform the second image to overlap with the first
  - Blend the two together to create a mosaic
  - (If there are more images, repeat)
- …but **wait**, why should this work at all?
  - What about the 3D geometry of the scene?
  - Why aren't we using it?

Source: Steve Seitz

Pinhole camera

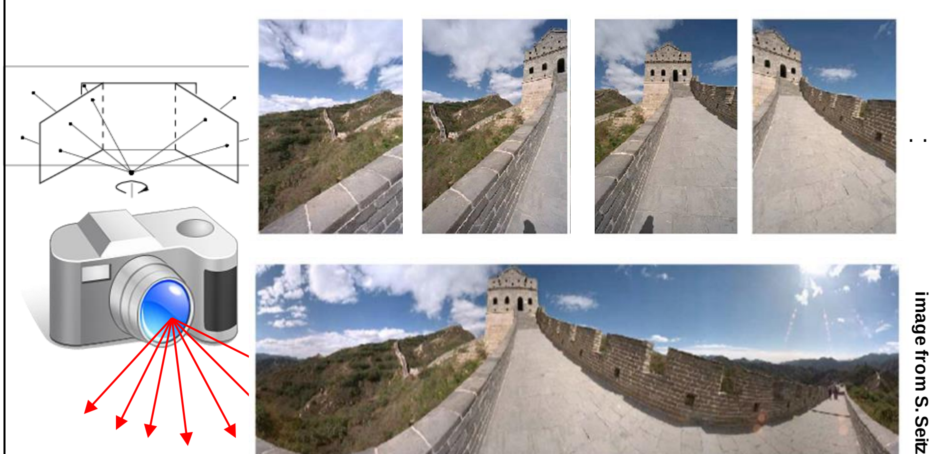
- Pinhole camera is a simple model to approximate imaging process, perspective **projection**.



If we treat pinhole as a point, only one ray from any given point can enter the camera.

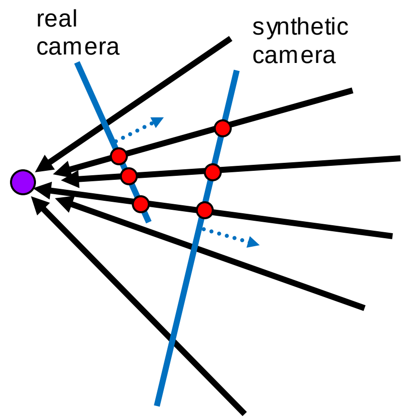
Fig from Forsyth and Ponce

Mosaics



Obtain a wider angle view by combining multiple images.

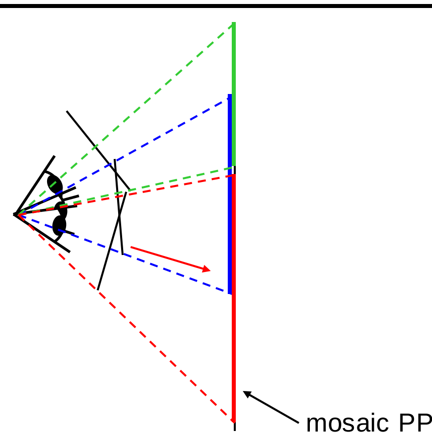
Mosaics: generating synthetic views



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

Source: Alyosha Efros

Image reprojection



The mosaic has a natural interpretation in 3D

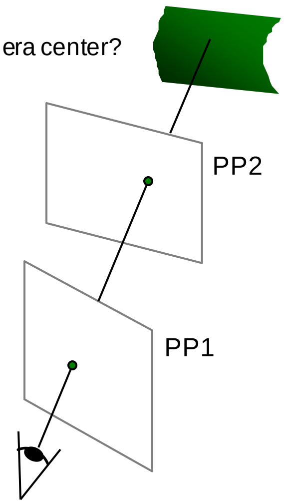
- The images are reprojected onto a common plane
- The mosaic is formed on this plane
- Mosaic is a *synthetic wide-angle camera*

Source: Steve Seitz

Image reprojection

Basic question

- How to relate two images from the same camera center?
  - how to map a pixel from PP1 to PP2



Answer

- Cast a ray through each pixel in PP1
- Draw the pixel where that ray intersects PP2

Observation:
Rather than thinking of this as a 3D reprojection, think of it as a 2D **image warp** from one image to another.

Source: Alyosha Efros

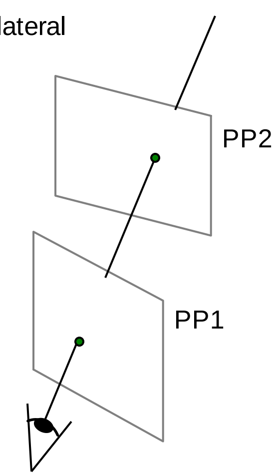
Image reprojection: Homography

A projective transform is a mapping between any two PPs with the same center of projection

- rectangle should map to arbitrary quadrilateral
- parallel lines aren't
- but must preserve straight lines

called **Homography**

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



Source: Alyosha Efros

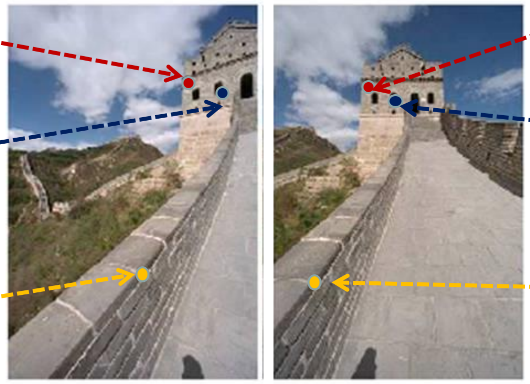
Homography

(x_1, y_1) (x'_1, y'_1)

(x_2, y_2) (x'_2, y'_2)

$\vdots$

(x_n, y_n) (x'_n, y'_n)



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…

Solving for homographies

p' = Hp

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

Can set scale factor i=1. So, there are 8 unknowns.

Set up a system of linear equations:

Ah = b

where vector of unknowns h = [a,b,c,d,e,f,g,h]^T

Need at least 8 eqs, but the more the better…

Solve for h. If overconstrained, solve using least-squares:

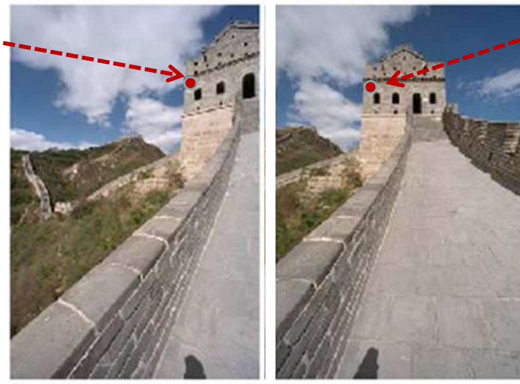
$$\min \|Ah - b\|^2$$

```
>> help lmdivide
```

BOARD

Homography

(x, y)



$\left(\frac{wx'}{w}, \frac{wy'}{w} \right) = (x', y')$

To **apply** a given homography **H**

- Compute **p'** = **Hp** (regular matrix multiply)
- Convert **p'** from homogeneous to image coordinates

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

RANSAC for estimating homography

RANSAC loop:

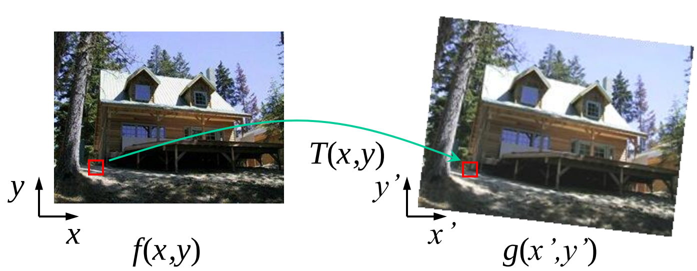
1. Select four feature pairs (at random)
2. Compute homography H
3. Compute *inliers* where $SSD(p_i', \boldsymbol{H}p_i) < \varepsilon$
4. Keep largest set of inliers
5. Re-compute least-squares H estimate on all of the inliers

Slide credit: Steve Seitz

Today

- Image mosaics
  - Fitting a 2D transformation
    - Affine, Homography
  - 2D image warping
  - Computing an image mosaic

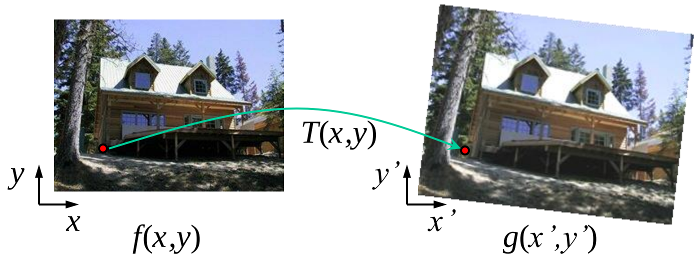
Image warping



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 from Alyosha Efros, CMU

Forward warping



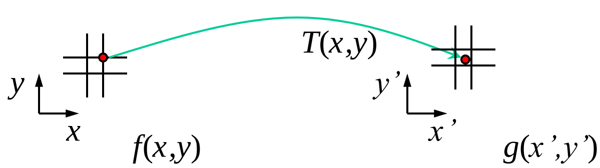
Send each pixel $f(x,y)$ to its corresponding location

$(x',y') = T(x,y)$ in the second image

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

Slide from Alyosha Efros, CMU

Forward warping



Send each pixel $f(x,y)$ to its corresponding location

$(x',y') = T(x,y)$ in the second image

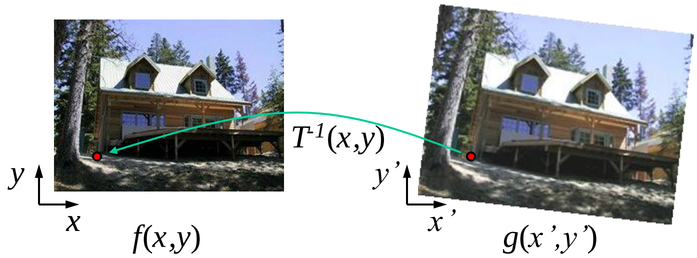
Q: what if pixel lands "between" two pixels?

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

– Known as "splatting"

Slide from Alyosha Efros, CMU

Inverse warping



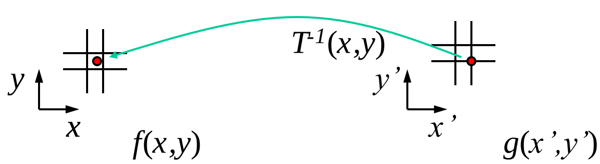
Get each pixel $g(x',y')$ from its corresponding location

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

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

Slide from Alyosha Efros, CMU

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?

A: *Interpolate* color value from neighbors

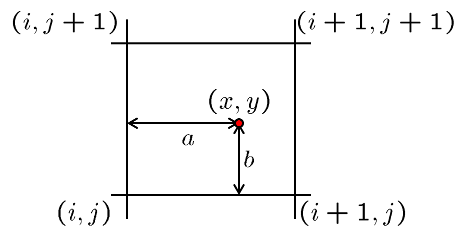
– nearest neighbor, bilinear…

`>> help interp2`

Slide from Alyosha Efros, CMU

Bilinear interpolation

Sampling at *f(x,y):*



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

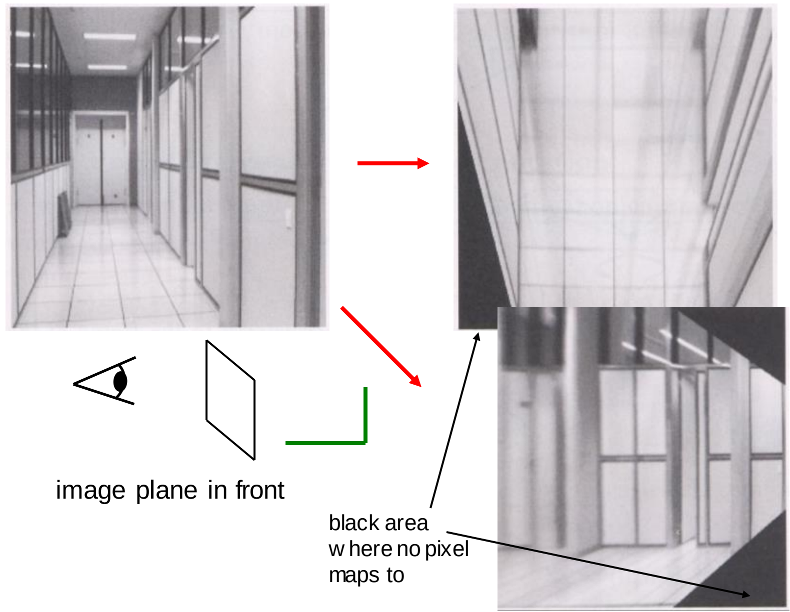
Slide from Alyosha Efros, CMU

Recap: How to stitch together a panorama (a.k.a. mosaic)?

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

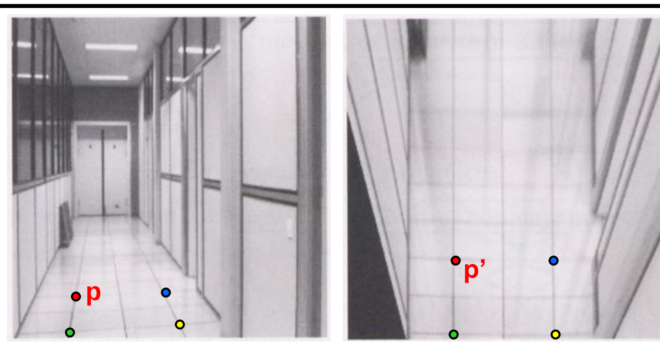
Source: Steve Seitz

Image warping with homographies



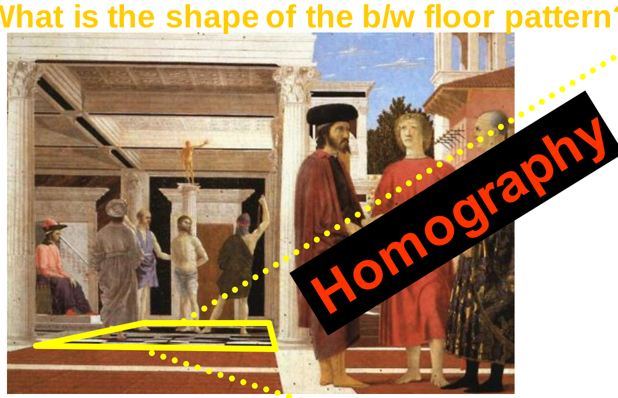
Source: Steve Seitz

Image rectification



Analysing patterns and shapes

What is the shape of the b/w floor pattern?



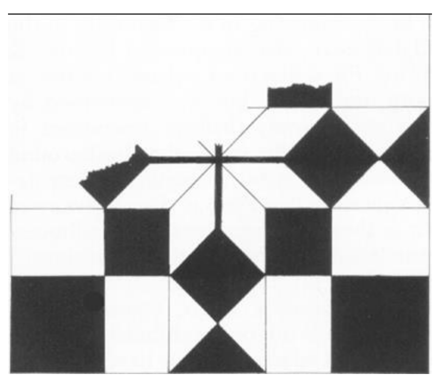
The floor (enlarged)

Automatically rectified floor

Slide from Antonio Criminisi

Analysing patterns and shapes

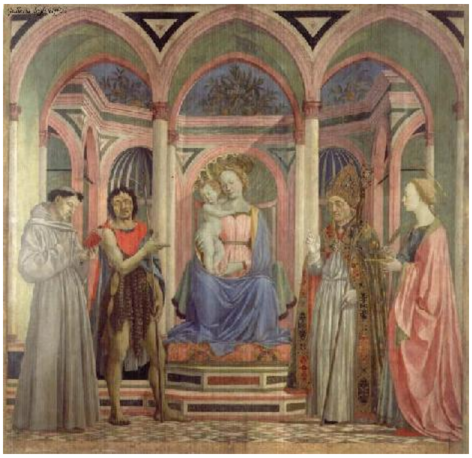
Automatic rectification



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

Slide from Antonio Criminisi

Analysing patterns and shapes



What is the (complicated) shape of the floor pattern?

Automatically rectified floor

***St. Lucy Altarpiece,* D. Veneziano**

Slide from Criminisi

Analysing patterns and shapes

Automatic rectification

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

Slide from Criminisi

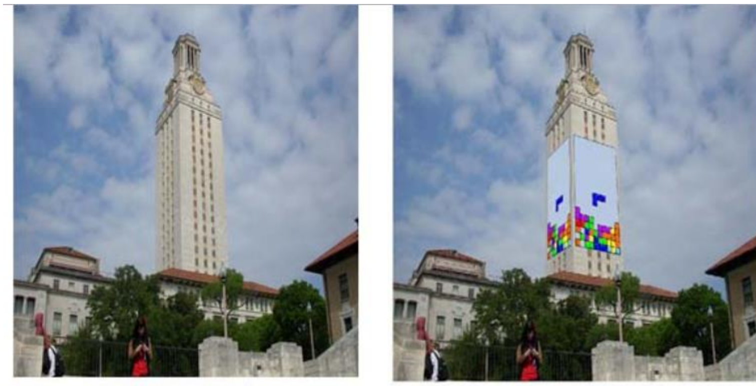
Andrew Harp

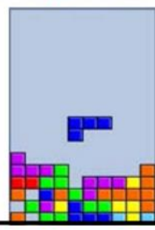
Andy Luong

Sung Ju Hwang

Ekapol Chuangsuwanich, CMU

Jesse Vera





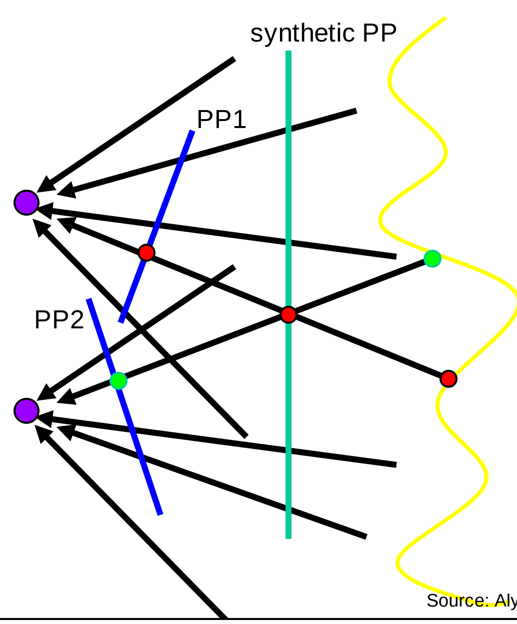
Kevin Gladstone

HP frames commercials

- http://www.youtube.com/watch?v=2RPl5vPEoQk

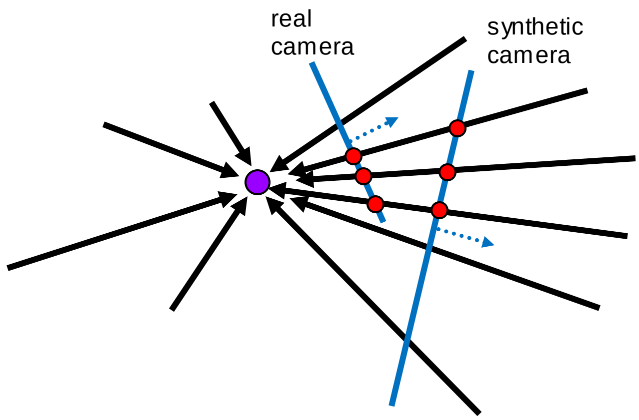
Changing camera center

Does it still work?



Source: Alyosha Efros

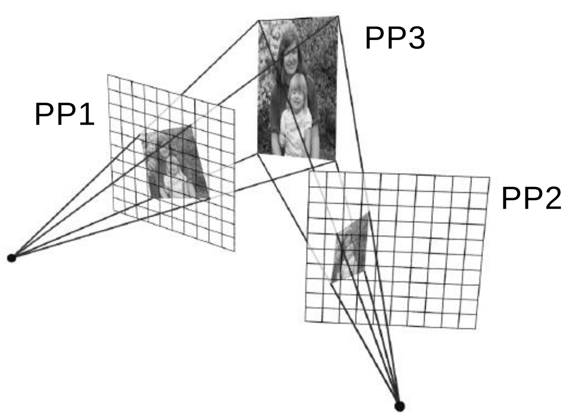
Recall: same camera center



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

Source: 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

Source: Alyosha Efros

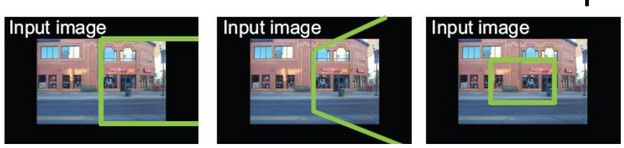
Boundary extension

- Wide-Angle Memories of Close-Up Scenes, Helene Intraub and Michael Richardson, Journal of Experimental Psychology: Learning, Memory, and Cognition, 1989, Vol. 15, No. 2, 179-187

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

Creating and Exploring a Large Photorealistic Virtual Space



Current view, and desired view in green

Synthesized view from new camera

Induced camera motion

Summary: alignment & warping

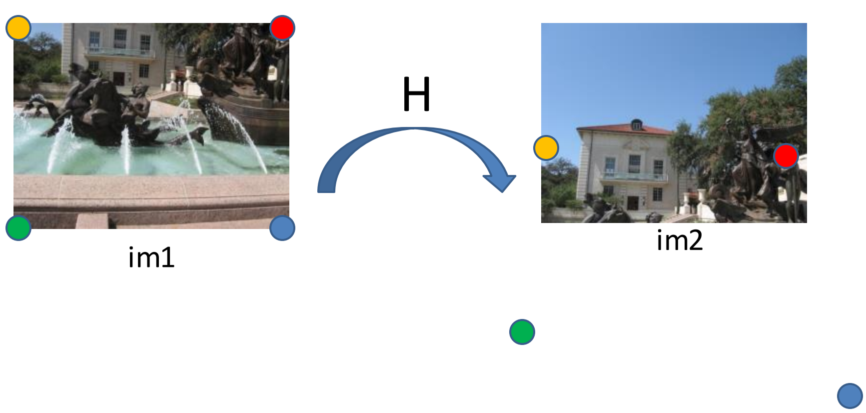
- Write **2d transformations** as matrix-vector multiplication (including translation when we use homogeneous coordinates)
- Perform **image warping** (forward, inverse)
- **Fitting transformations**: solve for unknown parameters given corresponding points from two views (affine, projective (homography)).
- **Mosaics**: uses homography and image warping to merge views taken from same center of projection.

Panoramas: main steps

- **1. Collect correspondences (manually for now)**
- **2. Solve for homography matrix H**
  - Least squares solution
- **3. Warp content from one image frame to the other to combine: say im1 into im2 reference frame**
- **4. Overlay im2 content onto the warped im1 content.**

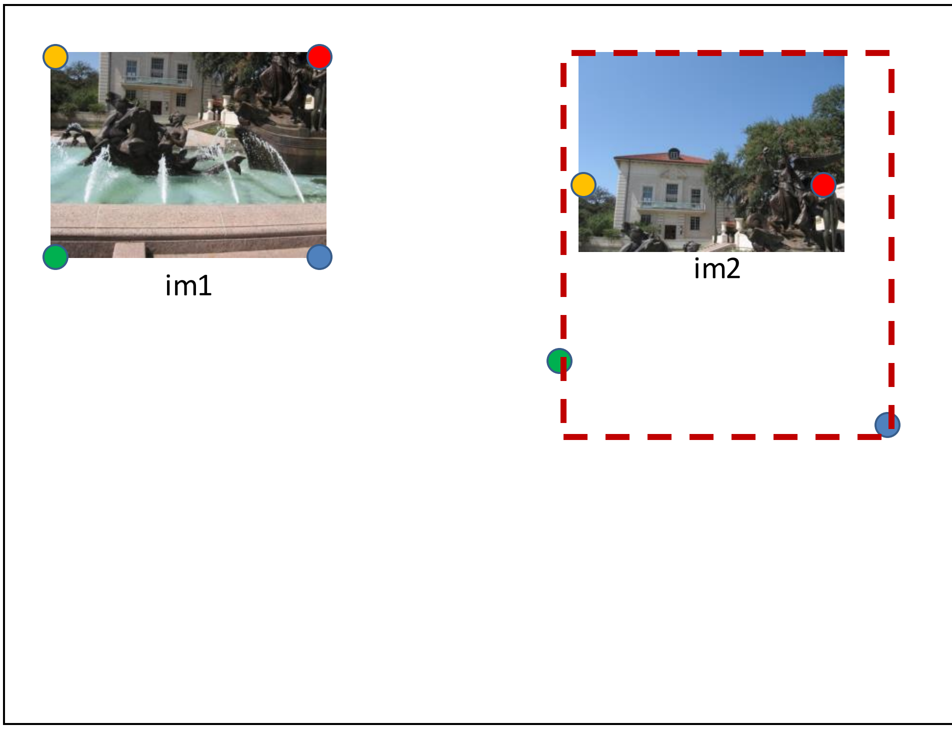
Panoramas: main steps

- **1. Collect correspondences (manually for now)**
- **2. Solve for homography matrix H**
  - Least squares solution
- **3. Warp content from one image frame to the other to combine: say im1 into im2 reference frame**
  - Determine bounds of the new combined image:
    - Where will the corners of im1 fall in im2's coordinate frame?
    - We will attempt to lookup colors for any of these positions we can get from im1.
- **4. Overlay im2 content onto the warped im1 content.**



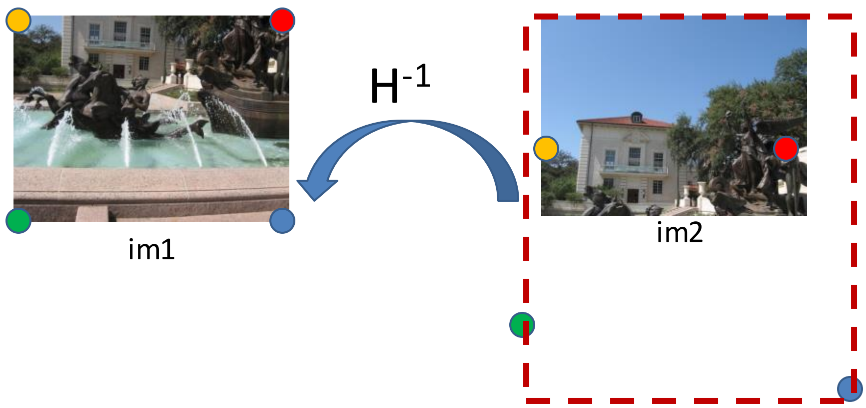
(Assuming we have solved for the H that maps points from im1 to im2.)

$$\begin{bmatrix} wx_2 \\ wy_2 \\ w \end{bmatrix} = \mathbf{H} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

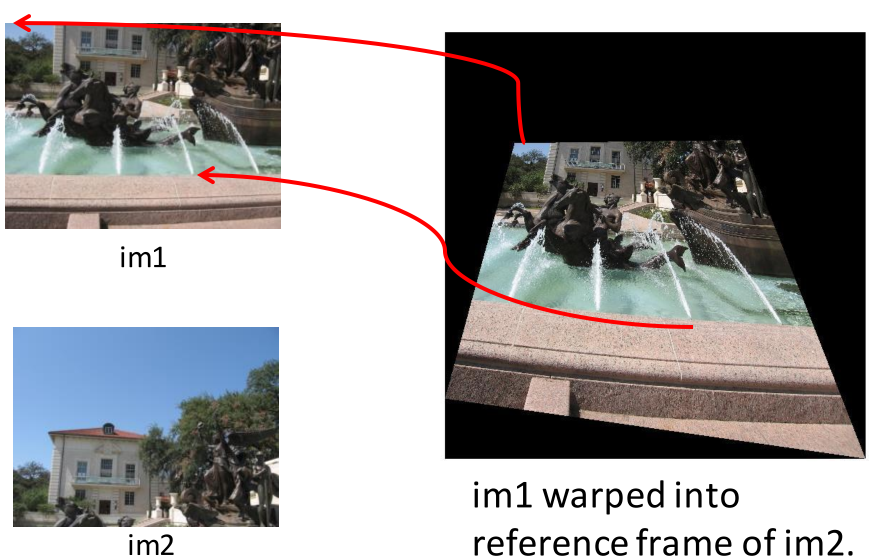


Panoramas: main steps

- **1. Collect correspondences (manually for now)**
- **2. Solve for homography matrix H**
  - Least squares solution
- **3. Warp content from one image frame to the other to combine: say im1 into im2 reference frame**
  - Determine bounds of the new combined image:
    - Where will the corners of im1 fall in im2's coordinate frame?
    - We will attempt to lookup colors for any of these positions we can get from im1.
  - Inverse warp:
    - Compute coordinates in im1's reference frame (via homography) for all points in that range.
    - Lookup all colors for all these positions from im1 (interp2)
- **4. Overlay im2 content onto the warped im1 content.**



(Assuming we have solved for the H that maps points from im1 to im2.)

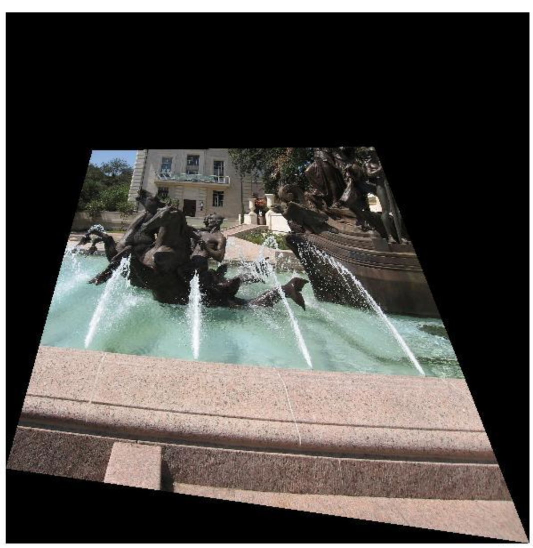


im1 warped into reference frame of im2.

Use interp2 to ask for the colors (possibly interpolated) from im1 at all the positions needed in im2's reference frame.

Panoramas: main steps

- **1. Collect correspondences (manually for now)**
- **2. Solve for homography matrix H**
  - Least squares solution
- **3. Warp content from one image frame to the other to combine: say im1 into im2 reference frame**
  - Determine bounds of the new combined image:
    - Where will the corners of im1 fall in im2's coordinate frame?
    - We will attempt to lookup colors for any of these positions we can get from im1.
  - Inverse warp:
    - Compute coordinates in im1's reference frame (via homography) for all points in that range.
    - Lookup all colors for all these positions from im1 (interp2)
- **4. Overlay im2 content onto the warped im1 content.**
  - Careful about new bounds of the output image



Summary: alignment & warping

- Write **2d transformations** as matrix-vector multiplication (including translation when we use homogeneous coordinates)
- Perform **image warping** (forward, inverse)
- **Fitting transformations**: solve for unknown parameters given corresponding points from two views (affine, projective (homography)).
- **Mosaics**: uses homography and image warping to merge views taken from same center of projection.