Local features and image matching

Wed March 2
Prof. Kristen Grauman
UT-Austin

Announcements

- Reminder: Pset 2 due tomorrow
- Reminder: Midterm exam is Wed March 9
  - See practice exam handout from last time
- My office hours today: 12:15-1:15

Last time

- RANSAC for robust fitting
  - Lines, translation
- Image mosaics
  - Fitting a 2D transformation
    - Affine, Homography

Today

Mosaics wrap-up:
How to warp one image to the other, given H?

How to detect *which features* to match?

Motivation for feature-based alignment: Image mosaics

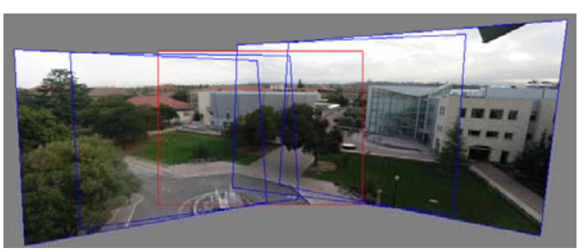


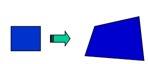
Image from http://graphics.cs.cmu.edu/courses/15-463/2010_fa

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



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

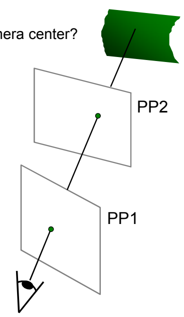
Mosaics

Obtain a wider angle view by combining multiple images.

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.

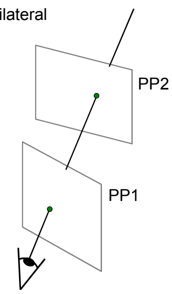
Source: Alyosha Efros

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

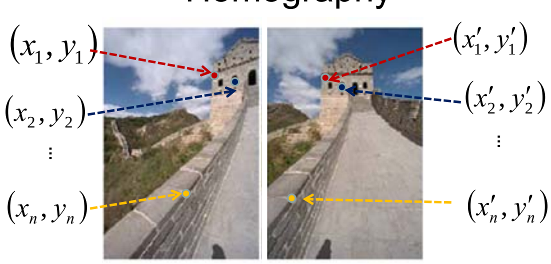
called **Homography**

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



Source: Alyosha Efros

Homography



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:

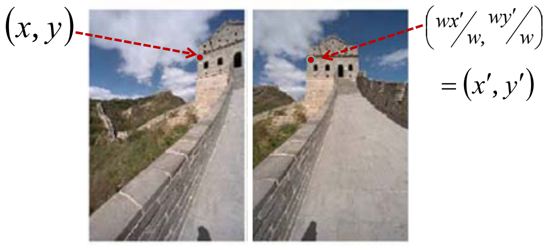
Ah = b

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
```

Homography

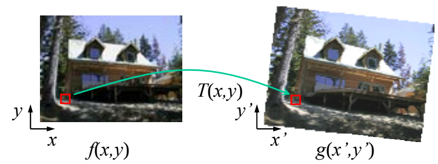


To **apply** a given homography **H**

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

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

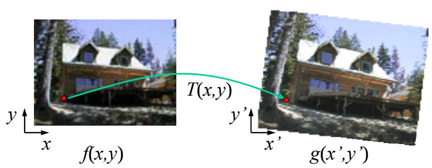
Image warping



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

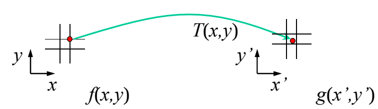


Send each pixel *f*(*x,y*) to its corresponding location
(*x',y'*) = *T*(*x,y*) in the second image

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

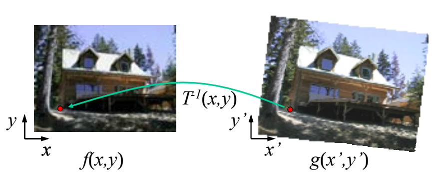
Q: what if pixel lands "between" two pixels?

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

– Known as "splatting"

Slide from Alyosha Efros, CMU

Inverse warping

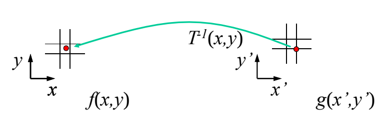


Get each pixel *g*(*x',y'*) from its corresponding location
(*x,y*) = T^{-1}(*x',y'*) in the first image

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

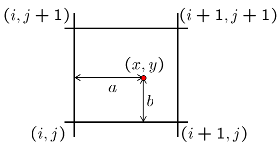
– nearest neighbor, bilinear…

Slide from Alyosha Efros, CMU

`>> help interp2`

Bilinear interpolation

Sampling at *f(x,y)*:



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

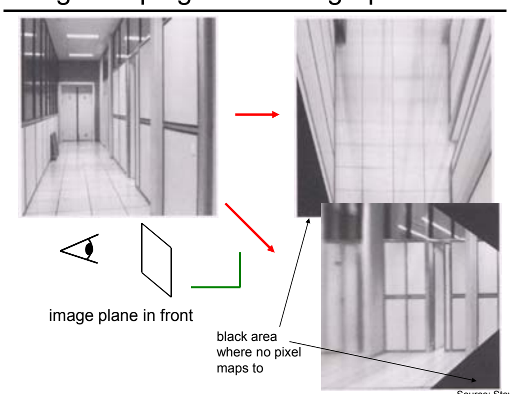
Slide from Alyosha Efros, CMU

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)

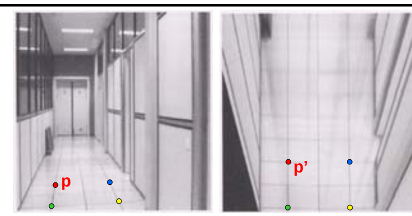
Source: Steve Seitz

Image warping with homographies

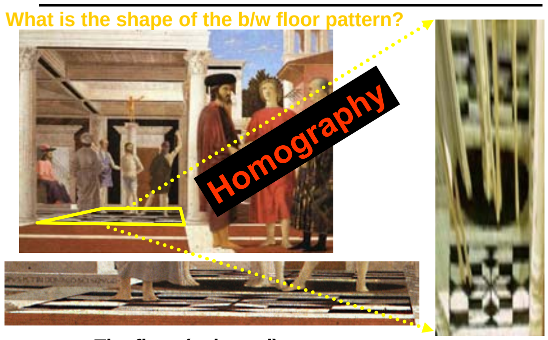


Source: Steve Seitz

Image rectification



Analysing patterns and shapes



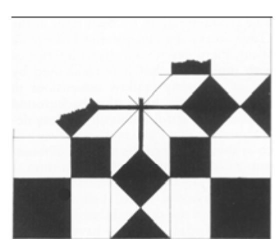
The floor (enlarged)

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

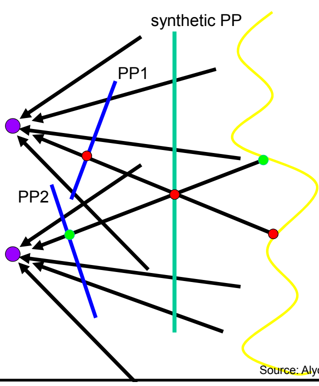
Automatic rectification

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

Slide from Criminisi

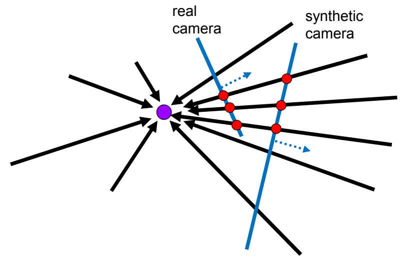
Changing camera center

Does it still work?



Source: Alyosha Efros

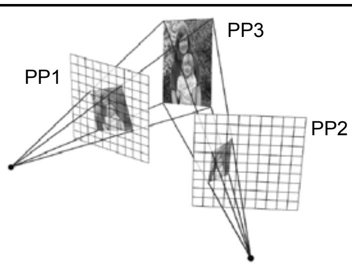
Recall: same camera center



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

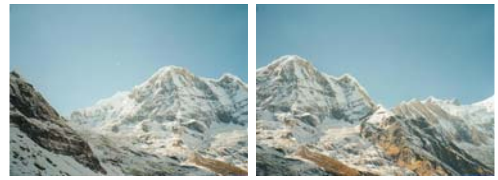
RANSAC for estimating homography

RANSAC loop:

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

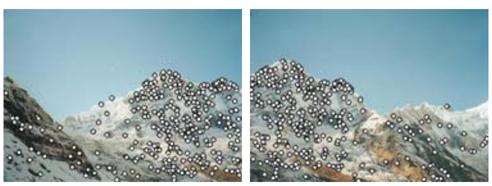
Slide credit: Steve Seitz

Robust feature-based alignment



Source: L. Lazebnik

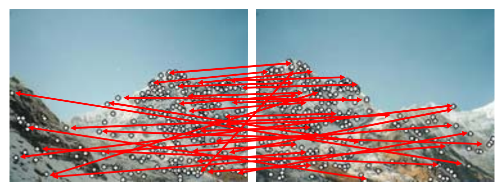
Robust feature-based alignment



- Extract features

Source: L. Lazebnik

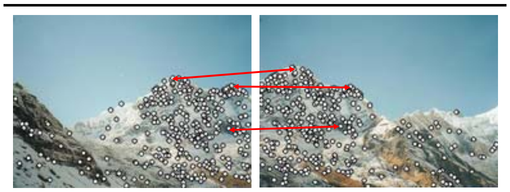
Robust feature-based alignment



- Extract features
- Compute *putative matches*

Source: L. Lazebnik

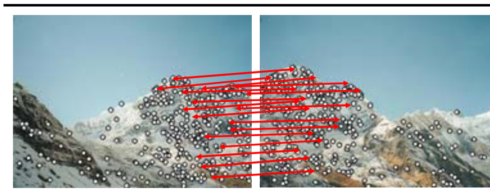
Robust feature-based alignment



- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)

Source: L. Lazebnik

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

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

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.

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

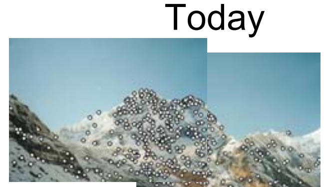
Creating and Exploring a Large Photorealistic Virtual Space

Current view, and desired view in green

Synthesized view from new camera

Induced camera motion

Today



Mosaics wrap-up:
How to warp one image to the other, given H?

How to detect *which features* to match?

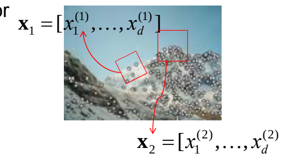
Detecting local invariant features

- Detection of interest points
  - Harris corner detection
  - Scale invariant blob detection: LoG
- (Next time: description of local patches)

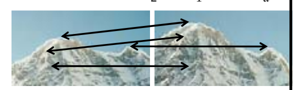
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.



3) Matching: Determine correspondence between descriptors in two views



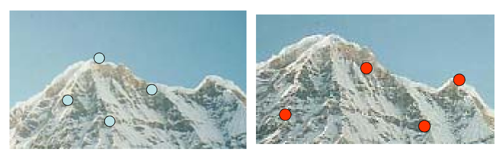
Kristen Grauman

Local features: desired properties

- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature has a distinctive description
- Compactness and efficiency
  - Many fewer features than image pixels
- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion

Goal: interest operator repeatability

- We want to detect (at least some of) the same points in both images.

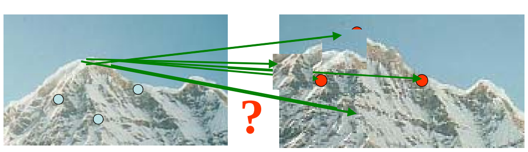


No chance to find true matches!

- Yet we have to be able to run the detection procedure *independently* per image.

Goal: descriptor distinctiveness

- We want to be able to reliably determine which point goes with which.



- Must provide some invariance to geometric and photometric differences between the two views.

Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

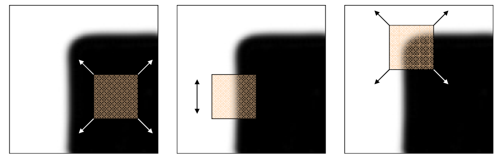
3) Matching: Determine correspondence between descriptors in two views

- What points would you choose?

Corners as distinctive interest points

We should easily recognize the point by looking through a small window

Shifting a window in *any direction* should give *a large change* in intensity



"flat" region: no change in all directions

"edge": no change along the edge direction

"corner": significant change in all directions

Slide credit: Alyosha Efros, Darya Frolova, Denis Simakov

Corners as distinctive interest points

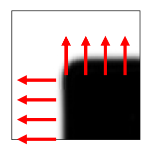
$$M = \sum w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).

Notation: $I_x \Leftrightarrow \frac{\partial I}{\partial x}$ $\quad I_y \Leftrightarrow \frac{\partial I}{\partial y}$ $\quad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$

What does this matrix reveal?

First, consider an axis-aligned corner:



What does this matrix reveal?

First, consider an axis-aligned corner:

$$M = \sum \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

This means dominant gradient directions align with x or y axis

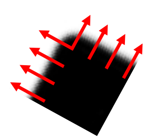
Look for locations where **both** λ's are large.

If either λ is close to 0, then this is **not** corner-like.

What if we have a corner that is not aligned with the image axes?

What does this matrix reveal?

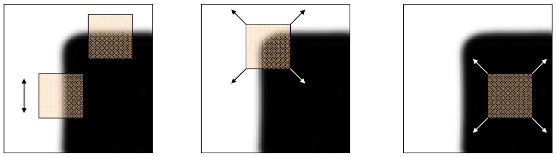
Since M is symmetric, we have $M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$



$$M x_i = \lambda_i x_i$$

The *eigenvalues* of *M* reveal the amount of intensity change in the two principal orthogonal gradient directions in the window.

Corner response function



"edge": $\lambda_1 >> \lambda_2$, $\lambda_2 >> \lambda_1$

"corner": λ_1 and λ_2 are large, $\lambda_1 \sim \lambda_2$;

"flat" region: λ_1 and λ_2 are small;

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

Harris corner detector

1) Compute *M* matrix for each image window to get their *cornerness* scores.
2) Find points whose surrounding window gave large corner response (*f*> threshold)
3) Take the points of local maxima, i.e., perform non-maximum suppression

Example of Harris application

Kristen Grauman

Example of Harris application

Compute corner response at every pixel.

Kristen Grauman

Example of Harris application

Kristen Grauman

Properties of the Harris corner detector

Rotation invariant? Yes

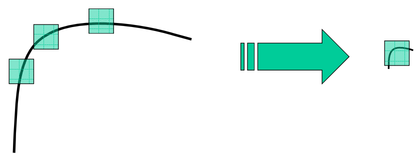
$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$

Scale invariant?

Properties of the Harris corner detector

Rotation invariant? Yes

Scale invariant? No



All points will be classified as edges

Corner !

Summary

- Image warping to create mosaic, given homography
- Interest point detection
  - Harris corner detector
  - Next time:
    - Laplacian of Gaussian, automatic scale selection