Announcements

Project 2 out today
- panorama signup
- help session at end of class

Today
- mosaic recap
- blending

Mosaics

Full screen panoramas (cubic): http://www.panoramas.dk/
Mars: http://www.panoramas.dk/fullscreen3/f2_mars97.html
2003 New Years Eve: http://www.panoramas.dk/fullscreen3/f1.html

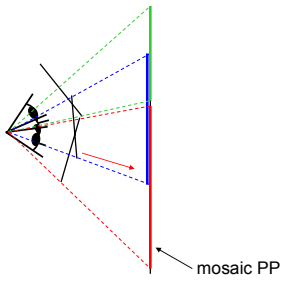
Image Mosaics

Goal
- Stitch together several images into a seamless composite

How to do it?

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
- Shift the second image to overlap with the first
- Blend the two together to create a mosaic
- If there are more images, repeat

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

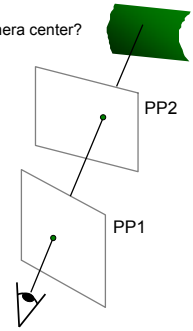
Image reprojection

Basic question

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

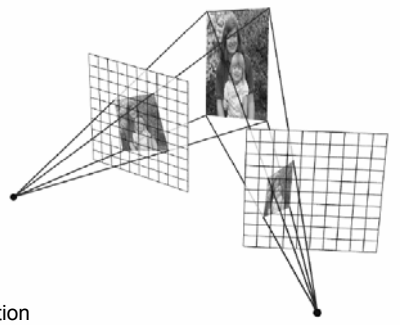
Answer

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



Don't need to know what's in the scene!

Image reprojection



Observation

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

Homographies

Perspective projection of a plane

- Lots of names for this:
  - **homography**, texture-map, colineation, planar projective map
- Modeled as a 2D warp using homogeneous coordinates

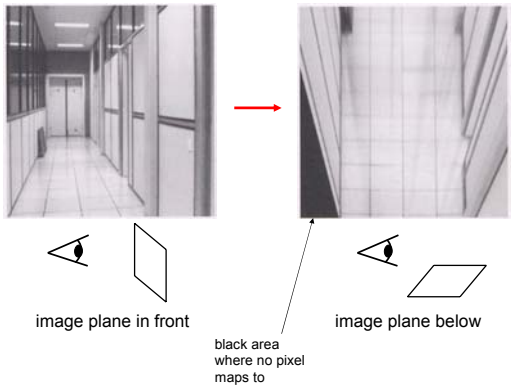
$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\mathbf{p'} \qquad \mathbf{H} \qquad \mathbf{p}$$

To apply a homography **H**

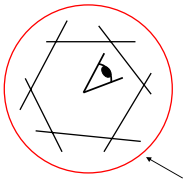
- Compute **p' = Hp** (regular matrix multiply)
- Convert **p'** from homogeneous to image coordinates
  - divide by w (third) coordinate

Image warping with homographies



Panoramas

What if you want a 360° field of view?



mosaic Projection Cylinder/Sphere

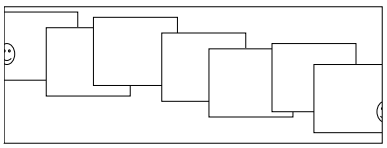
Idea is the same as before, but the warp is a little more complicated (see Rick's slides)

- key property: once you warp two images onto a cylinder or sphere, you can align them with a simple translation
  - assuming camera pan

Assembling the panorama

Stitch pairs together, blend, then crop

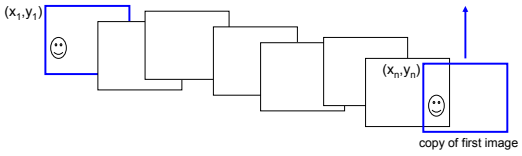
Problem: Drift



Error accumulation

- small errors accumulate over time

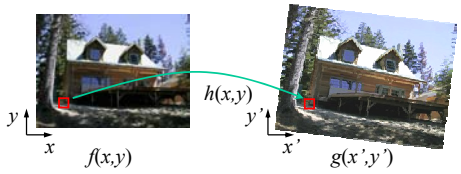
Problem: Drift



Solution

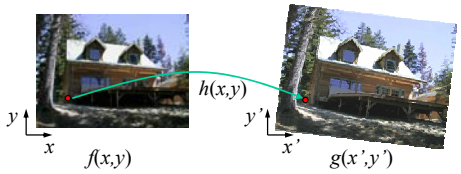
- add another copy of first image at the end
- this gives a constraint: $y_n = y_1$
- there are a bunch of ways to solve this problem
  - add displacement of $(y_1 - y_n)/(n-1)$ to each image after the first
  - compute a global warp: $y' = y + ax$
  - run a big optimization problem, incorporating this constraint
    - » best solution, but more complicated
    - » known as "bundle adjustment" (we'll cover next week, proj 3)

Image warping



Given a coordinate transform $(x',y') = h(x,y)$ and a source image $f(x,y)$, how do we compute a transformed image $g(x',y') = f(h(x,y))$?

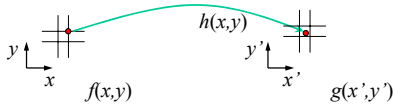
Forward warping



Send each pixel $f(x,y)$ to its corresponding location
$(x',y') = h(x,y)$ in the second image

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

Forward warping



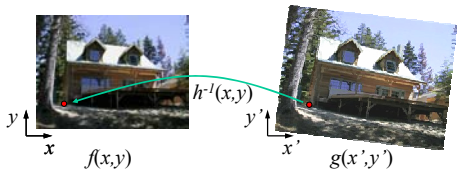
Send each pixel $f(x,y)$ to its corresponding location
$(x',y') = h(x,y)$ in the second image

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

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

- Known as "splatting"

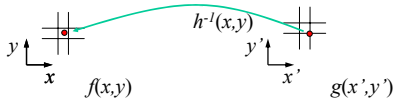
Inverse warping



Get each pixel $g(x',y')$ from its corresponding location
$(x,y) = h^{-1}(x',y')$ in the first image

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

Inverse warping



Get each pixel $g(x',y')$ from its corresponding location
$(x,y) = h^{-1}(x',y')$ in the first image

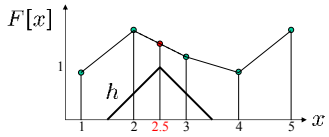
Q: what if pixel comes from "between" two pixels?

A: interpolate color value from neighbors
- known as image *resampling*

Image resampling

How can we estimate image colors off the grid?

Two Steps:
- *reconstruct* a continuous image $\tilde{f}$
- *resample* that image at desired locations

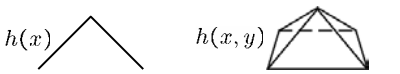


d = 1 in this example

Image reconstruction
- Convert F to a continuous function
  $f_F(x) = F(\frac{x}{d})$ when $\frac{x}{d}$ is an integer, 0 otherwise
- Reconstruct by cross-correlation:
  $\tilde{f} = h \otimes f_F$

Resampling filters

What does the 2D version of this hat function look like?



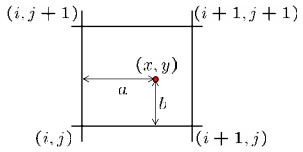
performs linear interpolation

(tent function) performs **bilinear interpolation**

Better filters give better resampled images
- Bicubic is common choice
  - fit 3rd degree polynomial surface to pixels in neighborhood
  - can also be implemented by a cross-correlation

Bilinear interpolation

A simple method for resampling images



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

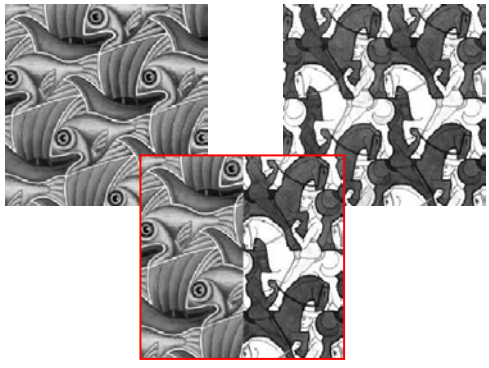
Forward vs. inverse warping

Q: which is better?

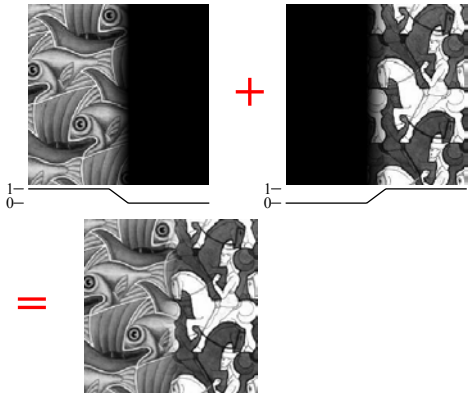
A: usually inverse—eliminates holes

- however, it requires an invertible warp function—not always possible...

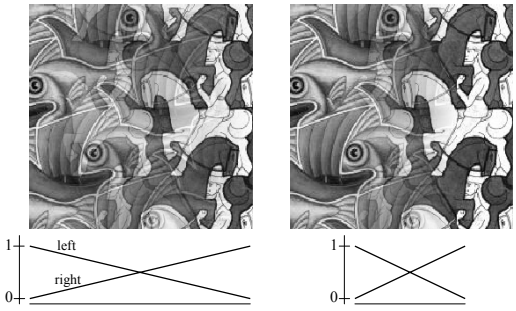
Image Blending



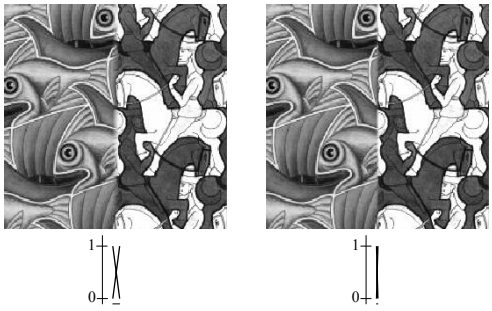
Feathering



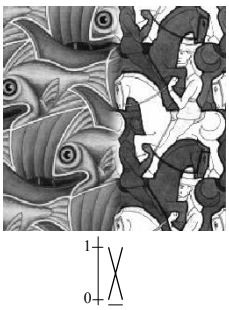
Effect of window size



Effect of window size



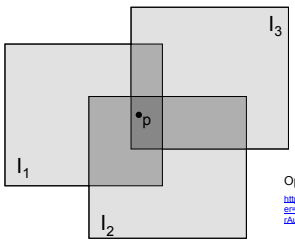
Good window size



"Optimal" window: smooth but not ghosted

- Doesn't always work...

Alpha Blending



Optional: see Blinn (CGA, 1994) for details:

http://ieeexplore.ieee.org/iel1/38/7531/00310740.pdf?isNumber=7531&prod=JNL&arnumber=310740&arSt=83&ared=87&arAuthor=Blinn%2C+J.F.

Encoding blend weights: I(x,y) = (αR, αG, αB, α)

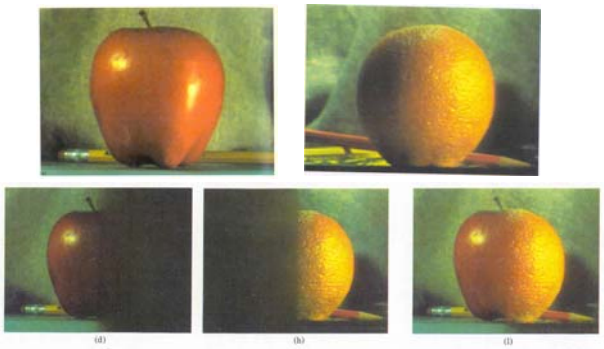
color at p = $\frac{(\alpha_1 R_1,\ \alpha_1 G_1,\ \alpha_1 B_1) + (\alpha_2 R_2,\ \alpha_2 G_2,\ \alpha_2 B_2) + (\alpha_3 R_3,\ \alpha_3 G_3,\ \alpha_3 B_3)}{\alpha_1 + \alpha_2 + \alpha_3}$

Implement this in two phases:

1. accumulate all images: I(p) = I(p) + I_i(p)
2. at the end, normalize: I(p) = I(p) / I(p)[4]

   Q: what if α = 0?

Pyramid blending



Create a Laplacian pyramid, blend each level

- Burt, P. J. and Adelson, E. H., A multiresolution spline with applications to image mosaics, ACM Transactions on Graphics, 42(4), October 1983, 217-236.

Poisson Image Editing

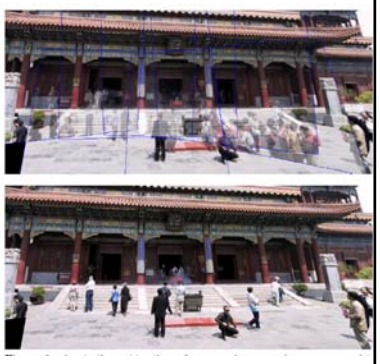
Blend the gradients of the two images, then integrate
For more info: Perez et al, SIGGRAPH 2003

- http://research.microsoft.com/vision/cambridge/papers/perez_siggraph03.pdf

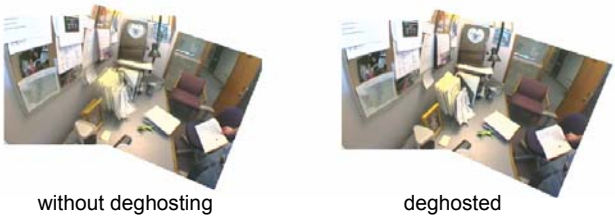
Graph cut blending

- Select only one image per output pixel, using spatial continuity

Kwatra et al., SIGGRAPH 03
Agarwala et al., SIGGRAPH 04



Deghosting using optical flow



without deghosting deghosted

Compensate for misalignments, parallax, lens distortion

- compute residual pixel-wise warp needed for perfect alignment
- (optical flow—will cover this later in the course)
- apply that warp to register the images (deghosting)
- covered in Szesliski & Shum reading

Project 2 (out today)

1. Take pictures on a tripod (or handheld)
2. Warp to spherical coordinates
3. Automatically compute pair-wise alignments
4. Correct for drift
5. Blend the images together
6. Crop the result and import into a viewer

AutoStitch

Method so far is not completely automatic

- need to know which pairs fit together

Newer methods are fully automatic

- AutoStitch, by Matthew Brown and David Lowe:
  - http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html
- Microsoft Picture It! Digital Image Pro 10.0

Other types of mosaics

Can mosaic onto *any* surface if you know the geometry

- See NASA's Visible Earth project for some stunning earth mosaics
  - http://earthobservatory.nasa.gov/Newsroom/BlueMarble/

Slit images

y-t slices of the video volume are known as *slit images*

- take a single column of pixels from each input image

Slit images: cyclographs

Slit images: photofinish