Announcements

- Midterm due Friday, beginning of lecture
- Guest lecture on Friday: Antonio Criminisi, Microsoft Research

Mosaics

http://www.destination360.com/start1.htm
http://www.vrseattle.com/html/vrview.php?cat_id=11&vrs_id=vrs380

Today's Readings

- Szeliski and Shum paper (sections 1 and 2, skim the rest)
  - http://www.acm.org/pubs/citations/proceedings/graph/258734/p251-szeliski/

Image Mosaics

Goal

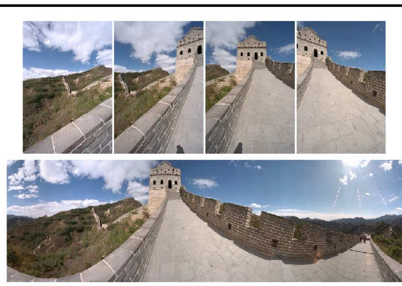
- Stitch together several images into a seamless composite

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

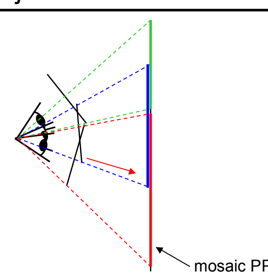
Aligning images



How to account for warping?

- Translations are not enough to align the images
- Photoshop demo

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

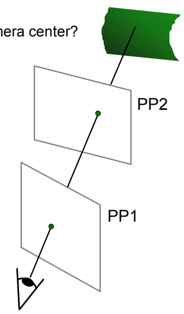
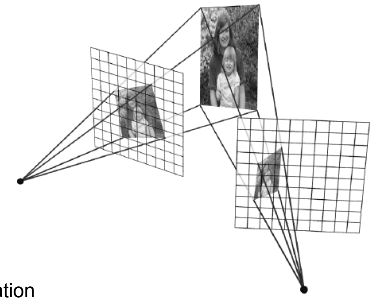


Image reprojection



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

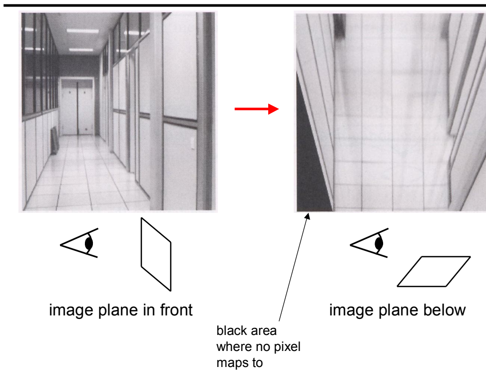
Homographies

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

$$\underbrace{\begin{bmatrix} sx' \\ sy' \\ s \end{bmatrix}}_{\mathbf{p'}} = \underbrace{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\mathbf{p}}$$

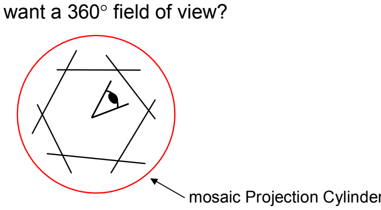
To apply a homography **H**
- Compute **p' = Hp** (regular matrix multiply)
- Convert **p'** from homogeneous to image coordinates
  - divide by w (third) coordinate

Image warping with homographies

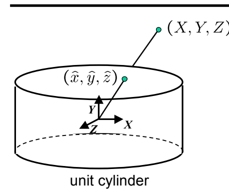


Panoramas

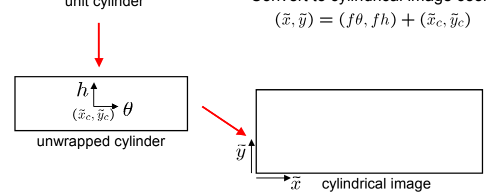
What if you want a 360° field of view?



Cylindrical projection

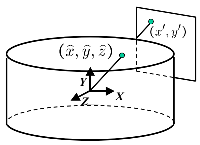


- Map 3D point (X,Y,Z) onto cylinder
  $$(\hat{x},\hat{y},\hat{z}) = \frac{1}{\sqrt{X^2+Z^2}}(X,Y,Z)$$
- Convert to cylindrical coordinates
  $$(sin\theta, h, cos\theta) = (\hat{x},\hat{y},\hat{z})$$
- Convert to cylindrical image coordinates
  $$(\tilde{x},\tilde{y}) = (f\theta, fh) + (\tilde{x}_c,\tilde{y}_c)$$

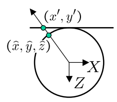


Cylindrical reprojection

How to map from a cylinder to a planar image?



side view



top-down view

- Apply camera projection matrix
  - for project 1, account for focal length and assume principle point is at center of image
    - » x'_c = ½ image width, y'_c = ½ image height

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} -f & 0 & w/2 & 0 \\ 0 & -f & h/2 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ 1 \end{bmatrix}$$

- Convert to image coordinates
  - divide by third coordinate (w)

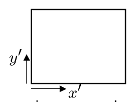
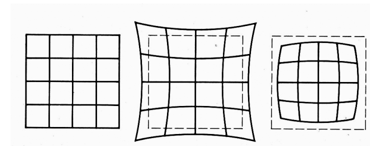


image coords

Distortion

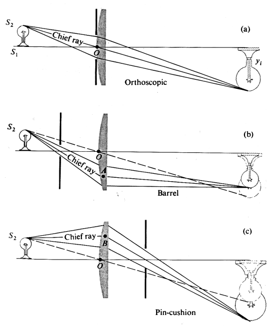


No distortion   Pin cushion   Barrel

Radial distortion of the image

- Caused by imperfect lenses
- Deviations are most noticeable for rays that pass through the edge of the lens

Distortion



Modeling distortion

Project $(\hat{x}, \hat{y}, \hat{z})$ to "normalized" image coordinates

$$x'_n = \hat{x}/\hat{z}$$
$$y'_n = \hat{y}/\hat{z}$$

Apply radial distortion

$$r^2 = {x'_n}^2 + {y'_n}^2$$
$$x'_d = x'_n(1 + \kappa_1 r^2 + \kappa_2 r^4)$$
$$y'_d = y'_n(1 + \kappa_1 r^2 + \kappa_2 r^4)$$

Apply focal length translate image center

$$x' = f x'_d + x_c$$
$$y' = f y'_d + y_c$$

To model lens distortion

- Use above projection operation instead of standard projection matrix multiplication

Cylindrical reprojection

Image 384x300   **f = 180 (pixels)**   **f = 280**   **f = 380**

Map image to cylindrical coordinates

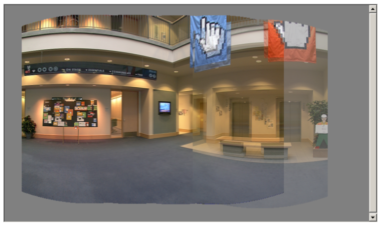
- need to know the focal length

Cylindrical panoramas

Steps

- Reproject each image onto a cylinder
- Blend
- Output the resulting mosaic

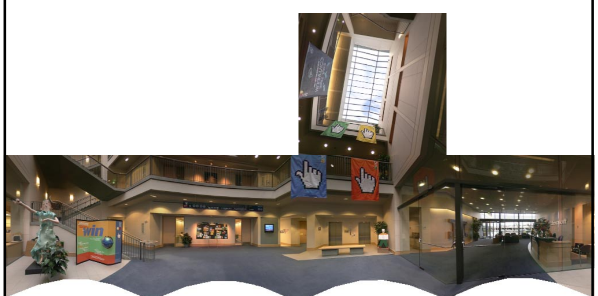
Cylindrical image stitching



What if you don't know the camera rotation?
- Solve for the camera rotations
  - Note that a rotation of the camera is a **translation** of the cylinder!
  - Use Lukas-Kanade to solve for translations of cylindrically-warped images

Full-view Panorama

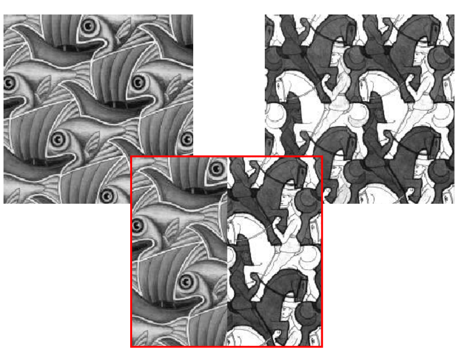
Different projections are possible



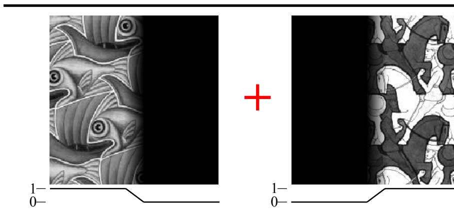
Project 2 (out on Friday)

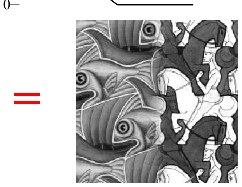
1. Take pictures on a tripod (or handheld)
2. Warp to cylindrical coordinates
3. Automatically compute pair-wise alignments
4. Correct for drift
5. Blend the images together
6. Crop the result and import into a viewer

Image Blending



Feathering





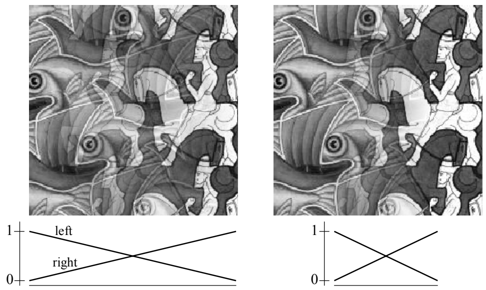
Encoding transparency

$I(x,y) = (\alpha R, \alpha G, \alpha B, \alpha)$

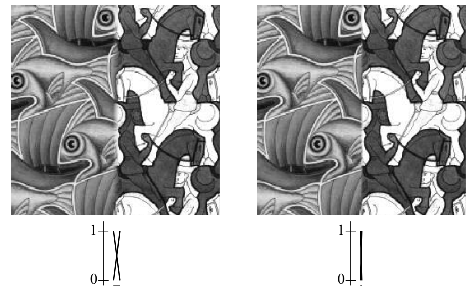
$I_{blend} = I_{left} + I_{right}$

See Blinn reading (CGA, 1994) for details

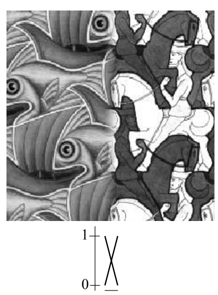
Effect of window size



Effect of window size



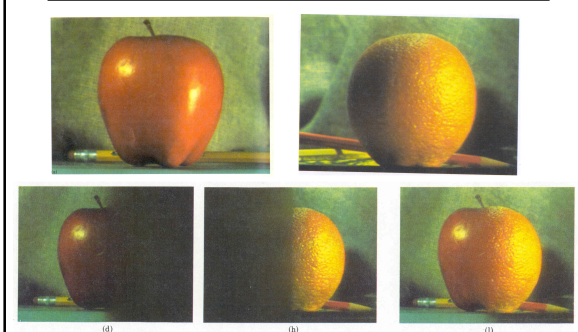
Good window size



"Optimal" window: smooth but not ghosted

- Doesn't always work...

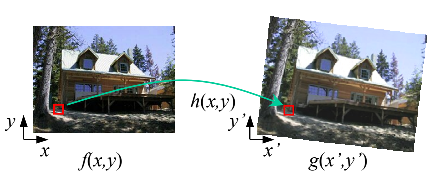
Pyramid blending



Create a Laplacian pyramid, blend each level

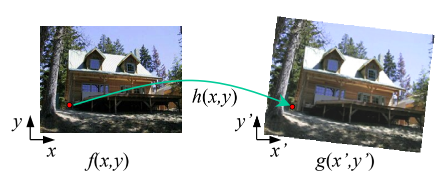
- Burt, P. J. and Adelson, E. H., A multiresolution spline with applications to image mosaics, ACM Transactions on Graphics, 42(4), October 1983, 217-236.

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 $g(x',y') = f(h(x,y))$?

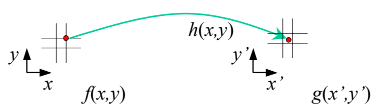
Forward warping



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



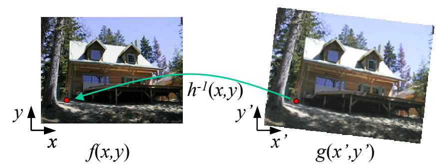
Send each pixel $f(x,y)$ to its corresponding location
$(x',y') = h(x,y)$ in the second image

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

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

- Known as "splatting"

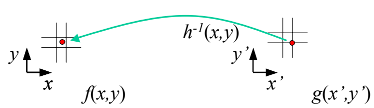
Inverse warping



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

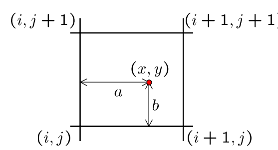
Q: what if pixel comes from "between" two pixels?

A: *resample* color value

- We discussed resampling techniques before
  - nearest neighbor, bilinear, Gaussian, bicubic

Bilinear interpolation

A common method for resampling images



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

Forward vs. inverse warping

Q: which is better?

A: usually inverse—eliminates holes

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

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/

Summary

Things to take home from this lecture

- Image alignment
- Image reprojection
  - homographies
  - cylindrical projection
- Radial distortion
- Creating cylindrical panoramas
- Image blending
- Image warping
  - forward warping
  - inverse warping
  - bilinear interpolation