CS5670: Computer Vision

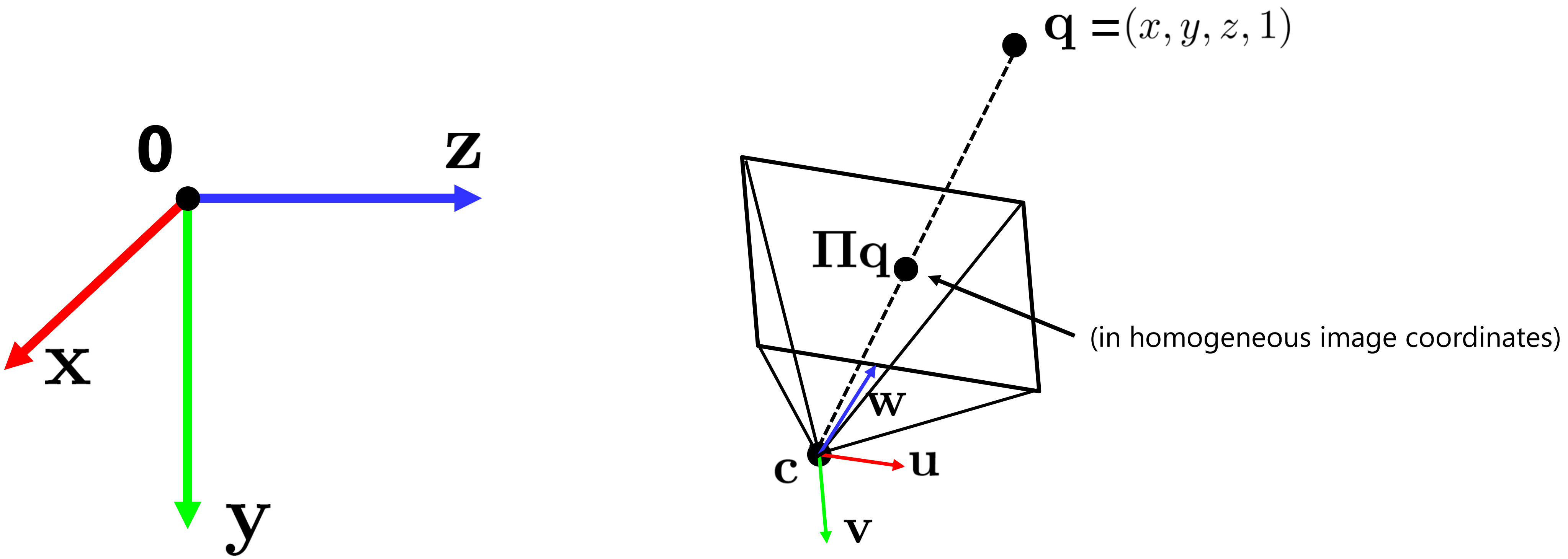
Panoramas

[Space Shuttle Discovery Flight Deck, Gigapan](#)

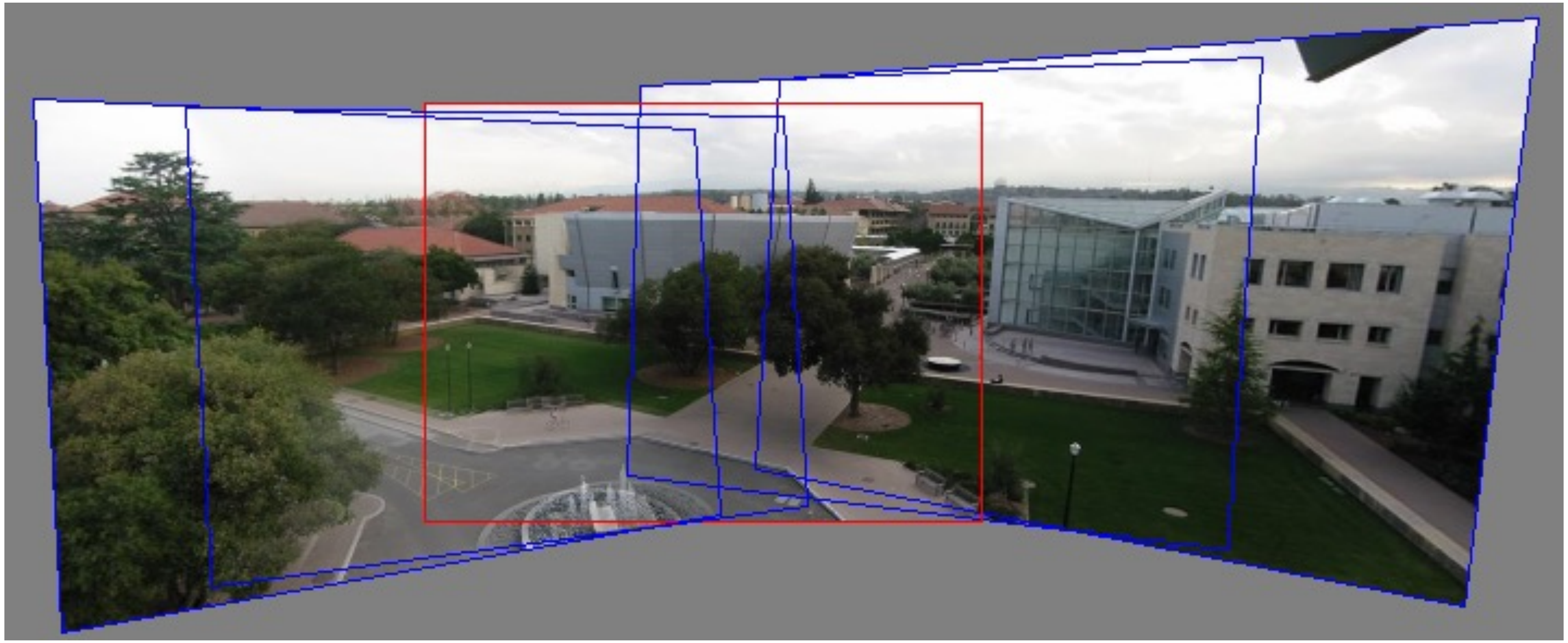
Announcements

- Midterm exam due by the start of class today
- Project 3: Autostitch (Panorama Stitching)
  - Released today, March 7
  - Due on Friday, March 17, by 8pm
  - Artifact due on Monday March 20 by 8pm
  - To be done in groups of 2
  - If you need help finding a team member, let us know
- Please fill out Midterm Course Evaluations
- No quiz on Thursday

From last time: Projection matrix

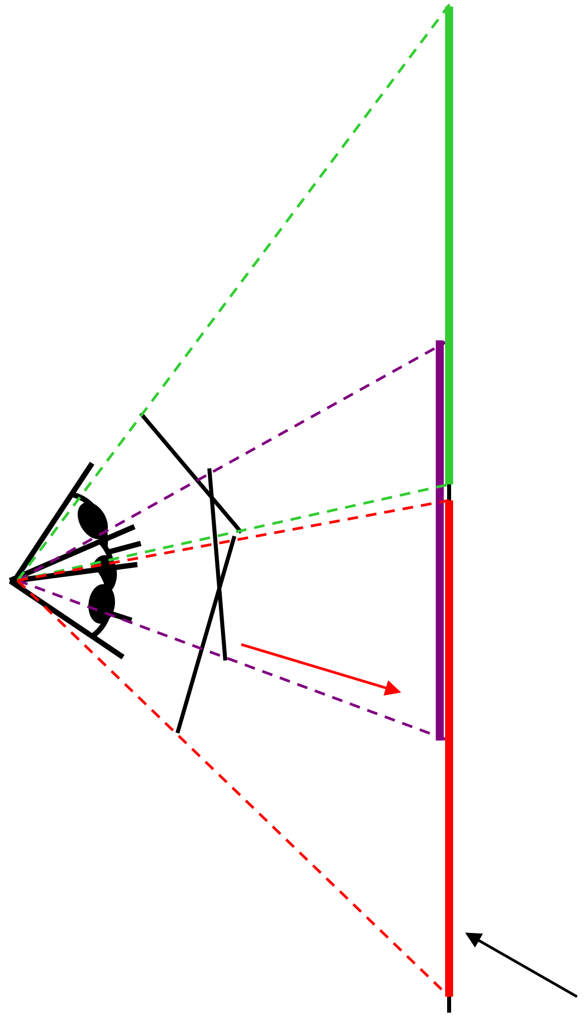


Back to panoramas



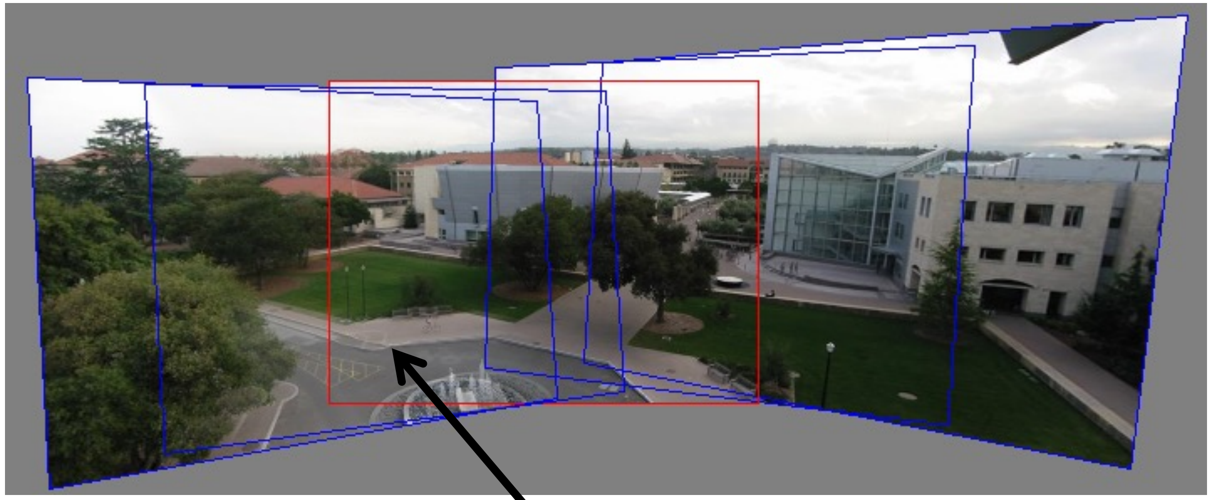
Can we use homographies to create a 360 degree panorama?

Idea: project images onto a common plane



mosaic projection plane

"Mosaic"



each image is warped with a homography $\mathbf{H}$

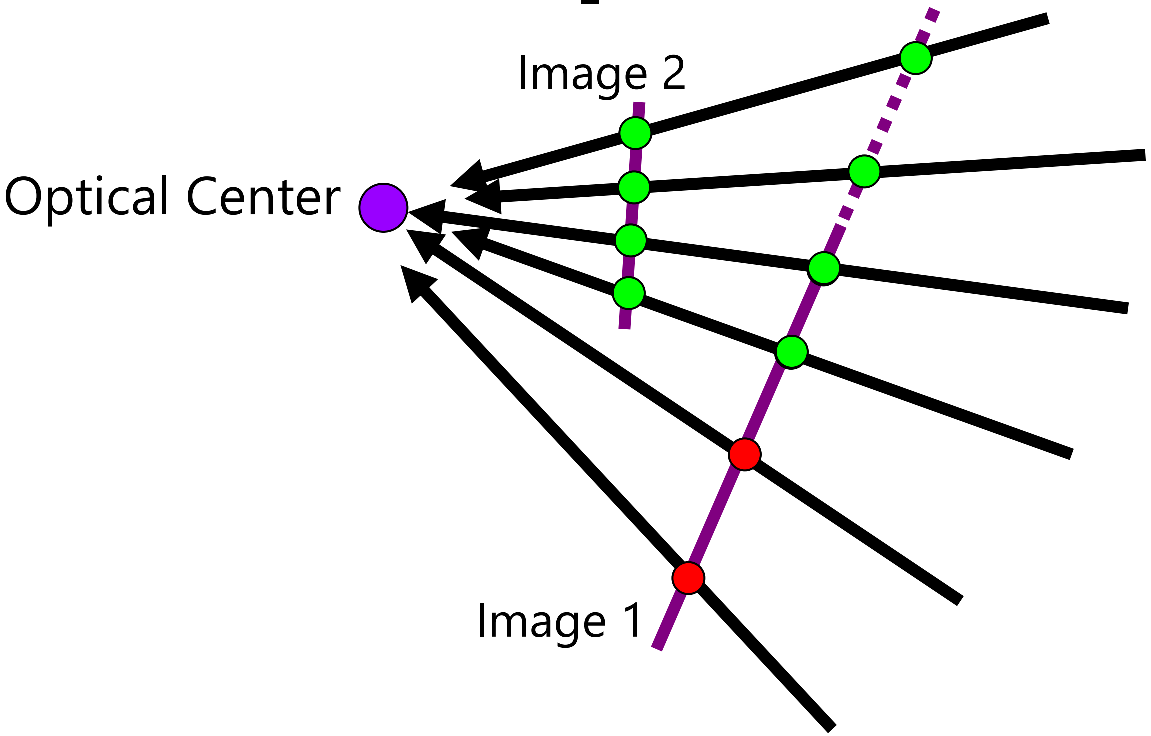
We'll see what this homography means next

Can't create a 360 panorama this way… we'll fix this shortly

Creating a panorama

- 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

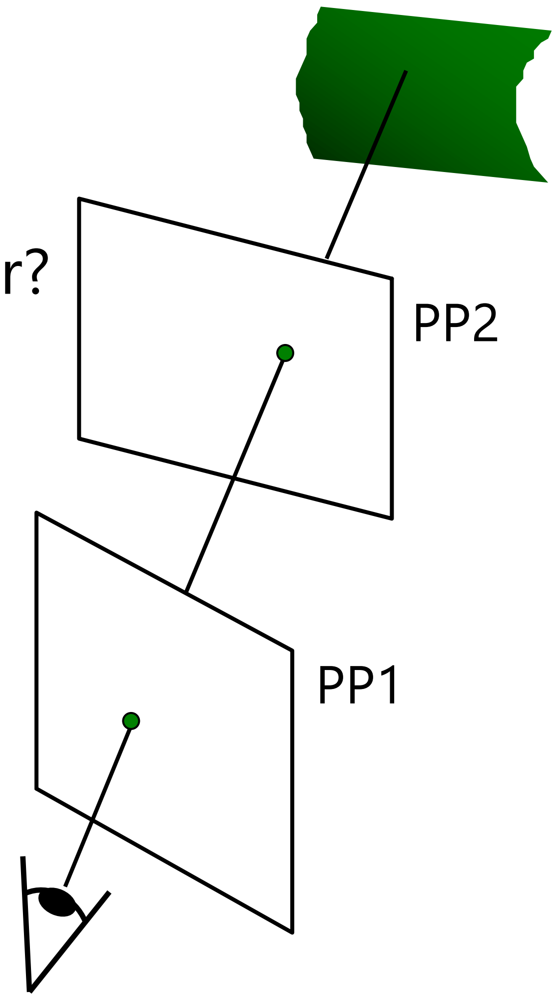
Geometric interpretation of mosaics



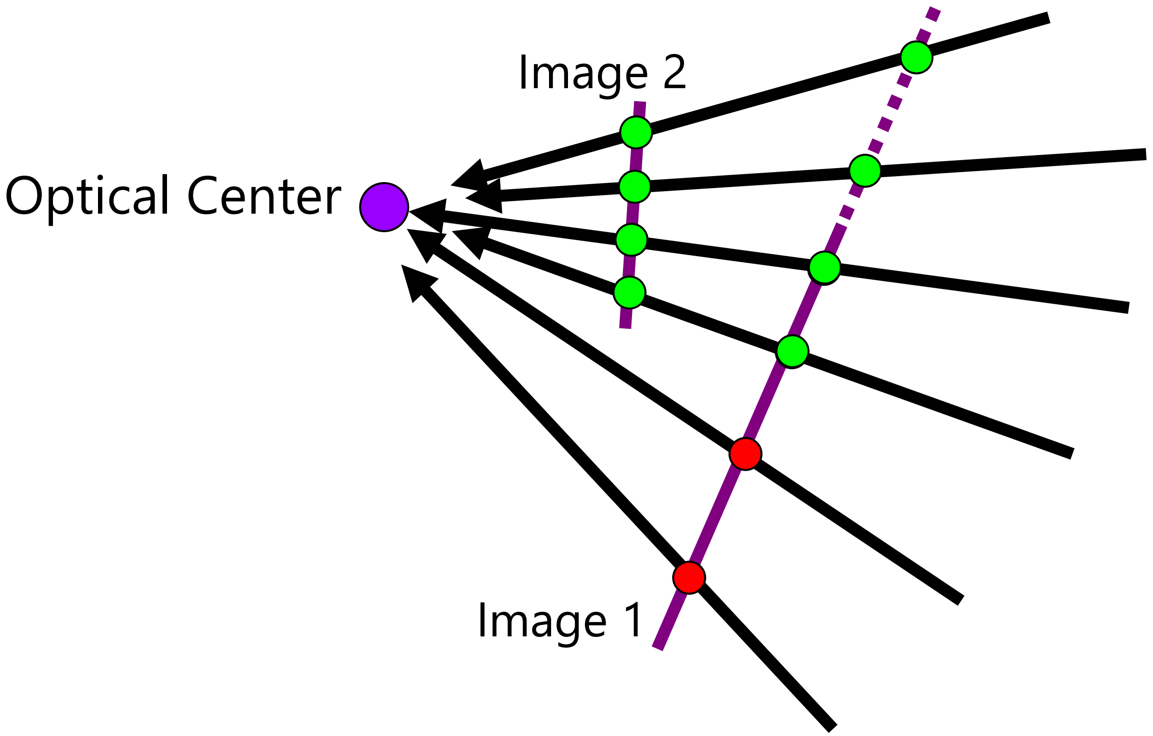
- If we capture all 360° of rays, we can create a 360° panorama
- The basic operation is *projecting* an image from one plane to another
- The projective transformation is scene-INDEPENDENT

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



What is the transformation?



Step 1: Convert pixels in image 2 to rays in camera 2's coordinate system.

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \mathbf{K}_2^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

Step 2: Convert rays in camera 2's coordinates to rays in camera 1's coordinates.

$$\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} = \mathbf{R}_2^T \mathbf{K}_2^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

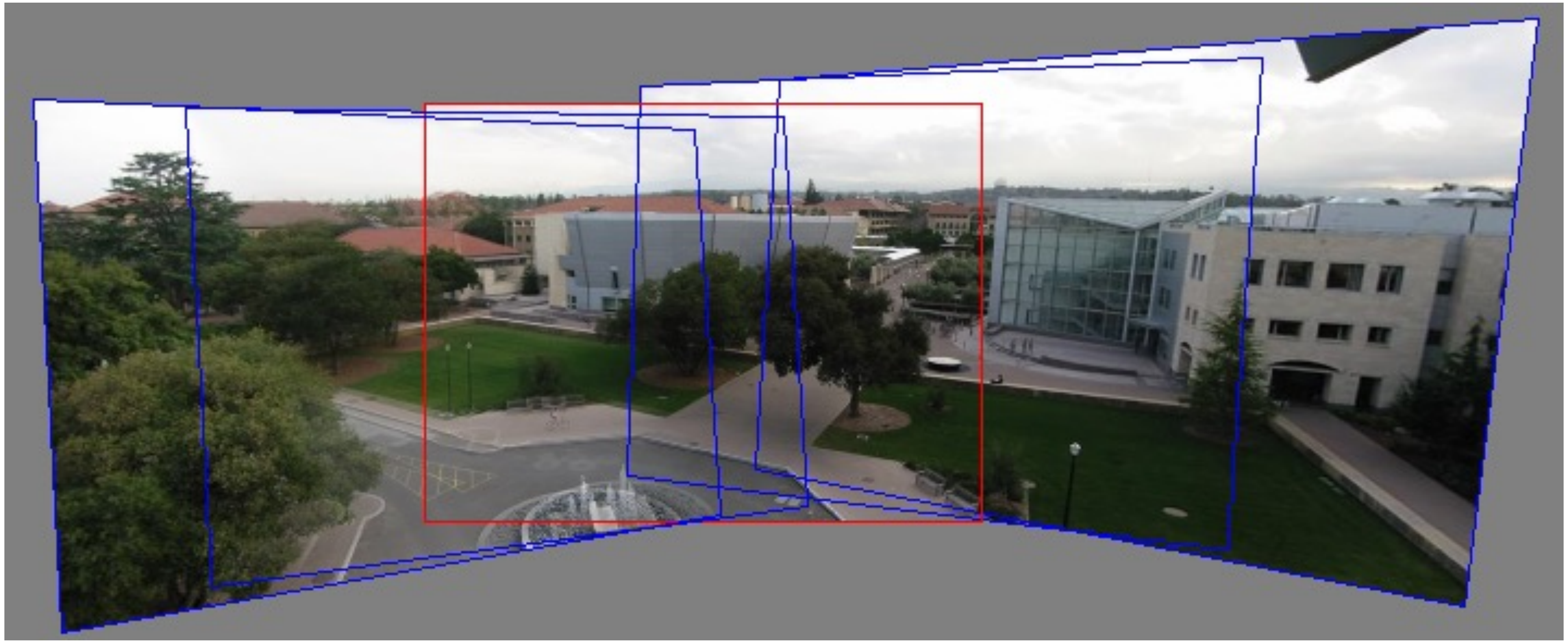
Step 3: Convert rays in camera 1's coordinates to pixels in image 1's coordinates.

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \sim \underbrace{\mathbf{K}_1 \mathbf{R}_2^T \mathbf{K}_2^{-1}}_{\text{3x3 homography}} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

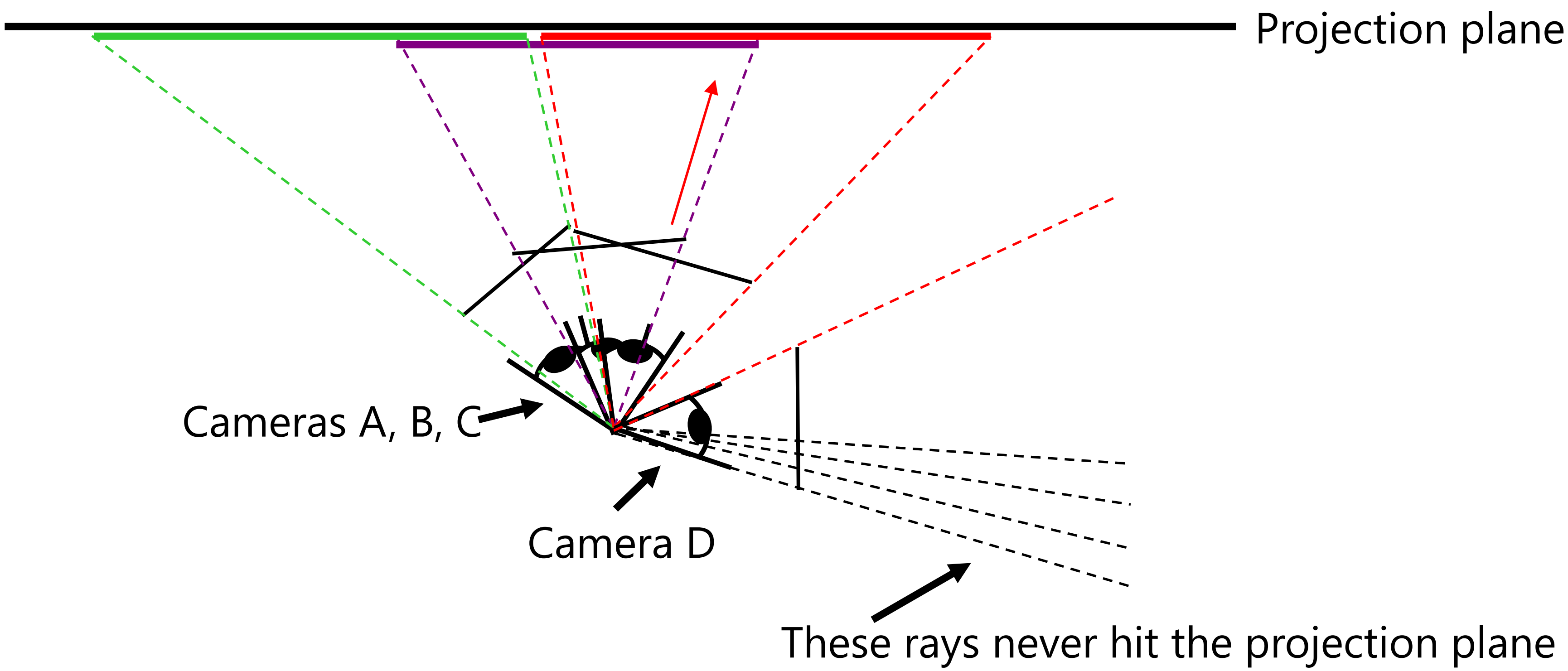
How do we map points in image 2 into image 1?

	image 1	image 2
intrinsics	$\mathbf{K}_1$	$\mathbf{K}_2$
extrinsics (rotation only)	$\mathbf{R}_1 = \mathbf{I}_{3\times 3}$	$\mathbf{R}_2$

Can we use homography to create a 360 panorama?

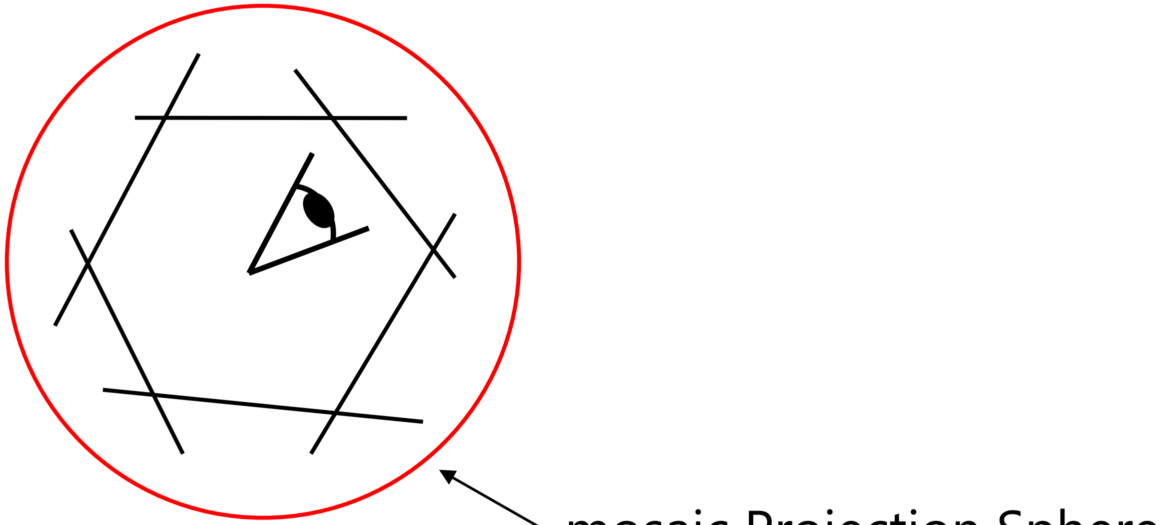


Answer: No

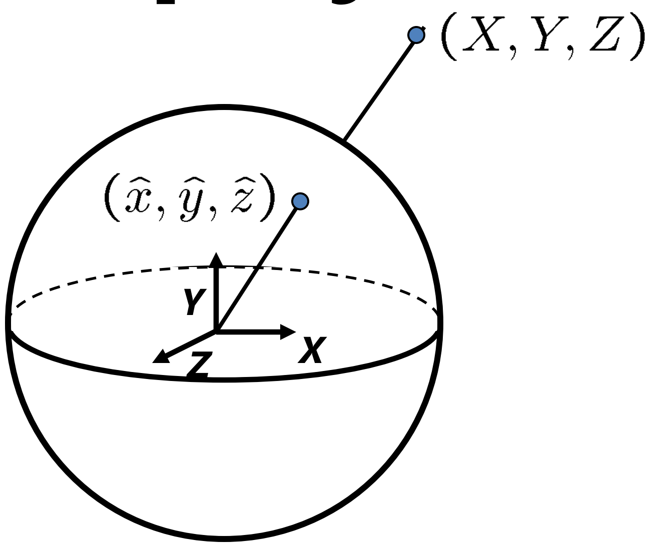

Projection plane
Cameras A, B, C
Camera D
These rays never hit the projection plane

Panoramas

- What if you want a 360° field of view?



Spherical projection



unit sphere

– Map 3D point (X,Y,Z) onto sphere

$$(\hat{x}, \hat{y}, \hat{z}) = \frac{1}{\sqrt{X^2+Y^2+Z^2}}(X, Y, Z)$$

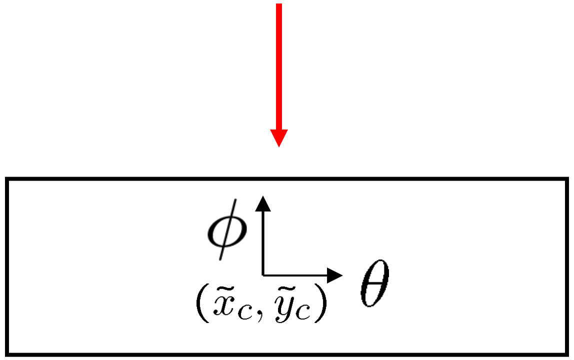
- Convert to spherical coordinates

$$(sin\theta cos\phi, sin\phi, cos\theta cos\phi) = (\hat{x}, \hat{y}, \hat{z})$$

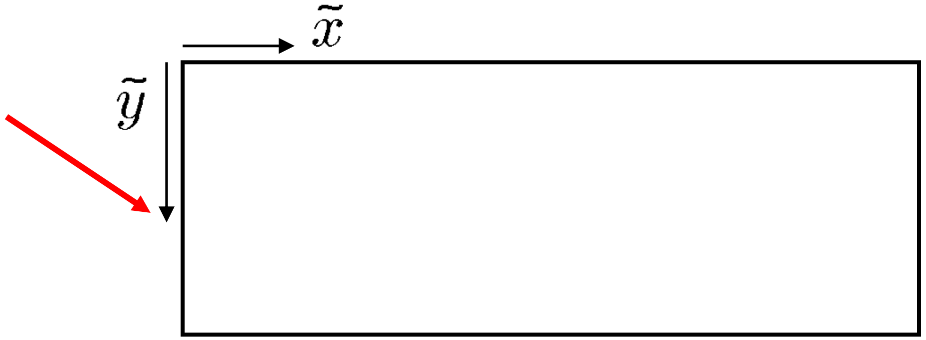
- Convert to spherical image coordinates

$$(\tilde{x}, \tilde{y}) = (s\theta, s\phi) + (\tilde{x}_c, \tilde{y}_c)$$

  – s defines size of the final image
    » often convenient to set s = camera focal length



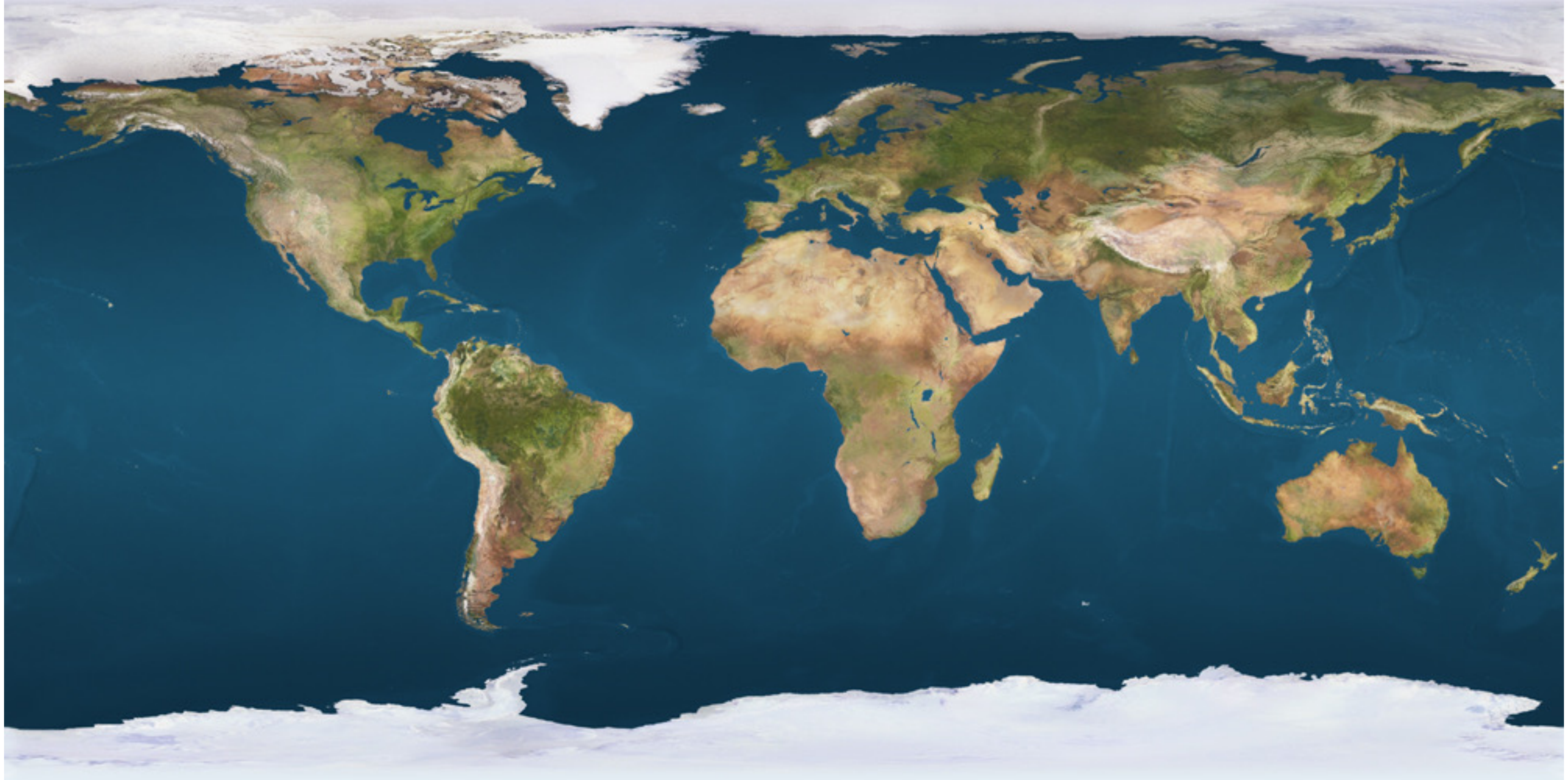
unwrapped sphere



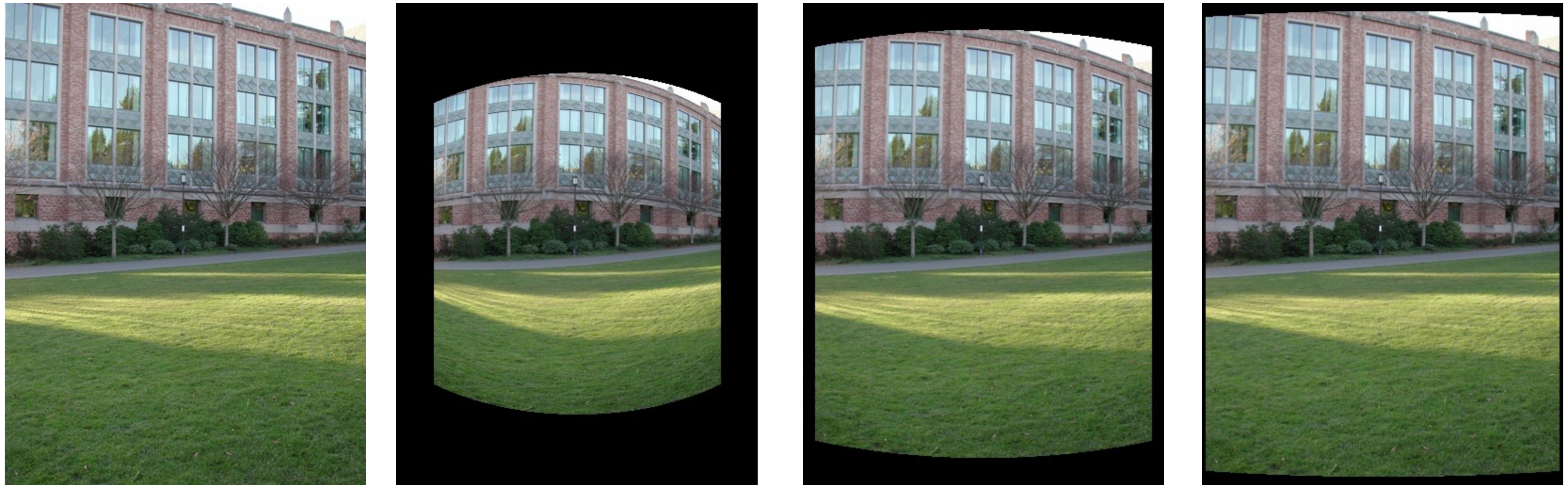
Spherical image

Unwrapping a sphere

Credit: JHT's Planetary Pixel Emporium



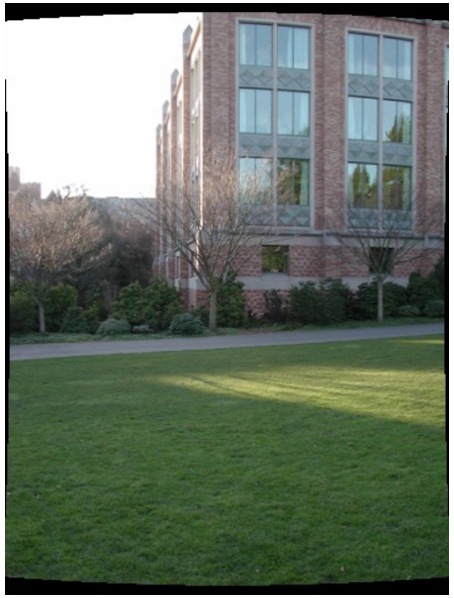
Spherical reprojection



input f = 200 (pixels) f = 400 f = 800

- Map image to spherical coordinates
  – need to know the focal length

Aligning spherical images

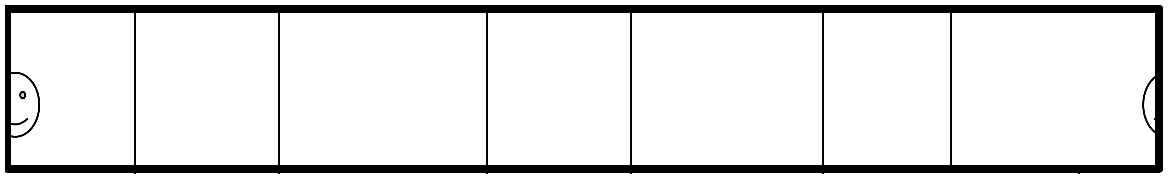


- Suppose we rotate the camera by θ about the vertical axis
  - How does this change the spherical image?

Aligning spherical images

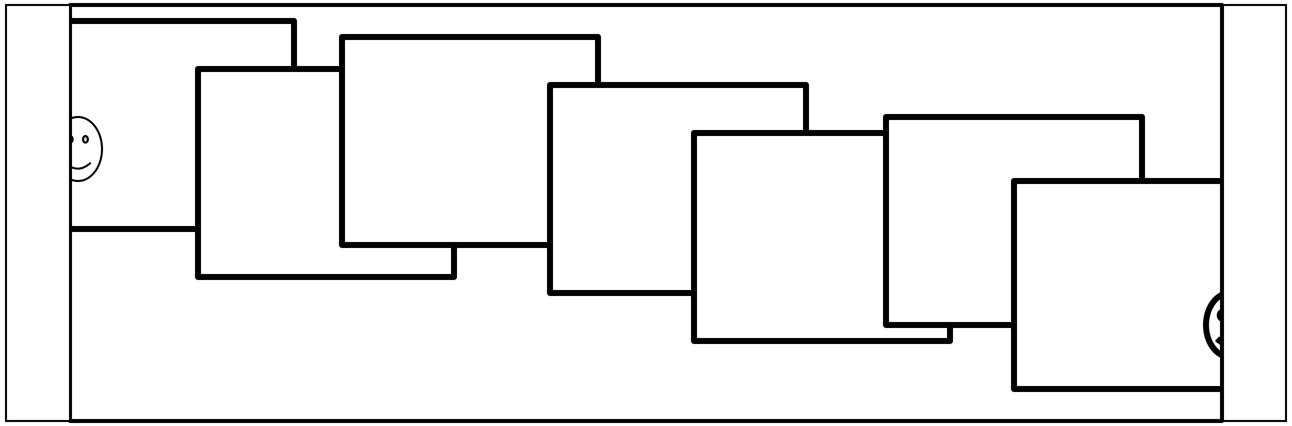
- Suppose we rotate the camera by θ about the vertical axis
  - How does this change the spherical image?
    - Translation by θ
  - This means that we can align spherical images by translation

Assembling the panorama



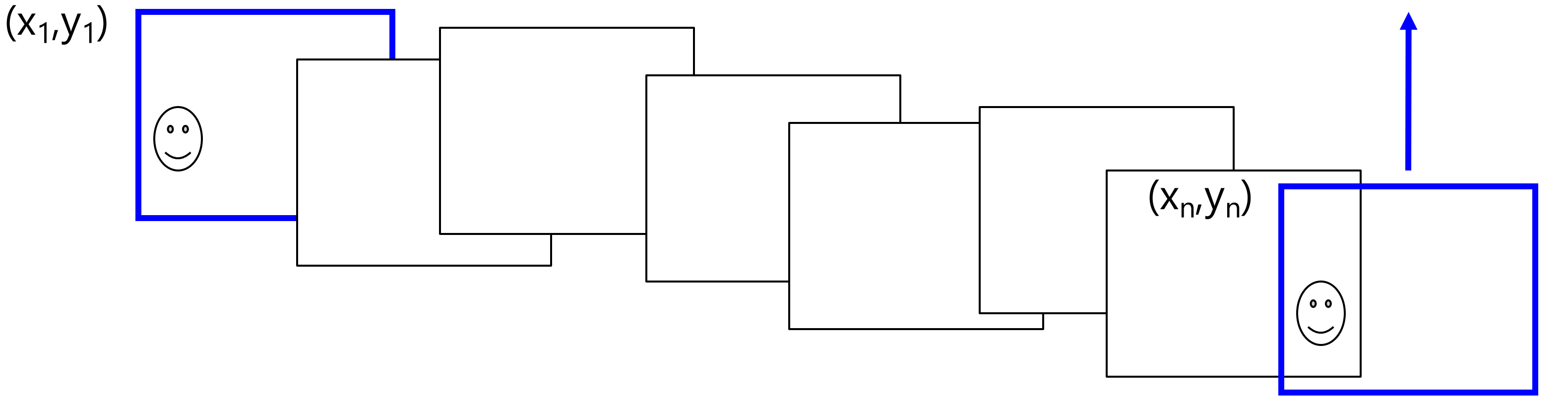
- Stitch pairs together, blend, then crop

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
    - **apply an affine warp: $y' = y + ax$ [you will implement this for P3]**
    - run a big optimization problem, incorporating this constraint
      - best solution, but more complicated
      - known as "bundle adjustment"

Project 3

1. Take pictures on a tripod (or handheld)
2. Warp to spherical coordinates (not needed if using homographies to align images)
3. Extract features
4. Align neighboring pairs using feature matching + RANSAC
5. Write out list of neighboring translations
6. Correct for drift
7. Read in warped images and blend them
8. Crop the result and import into a viewer

- Roughly based on Autostitch
  - By Matthew Brown and David Lowe
  - http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

Spherical panoramas

Microsoft Lobby: http://www.acm.org/pubs/citations/proceedings/graph/258734/p251-szeliski

Different projections are possible

Cube-map

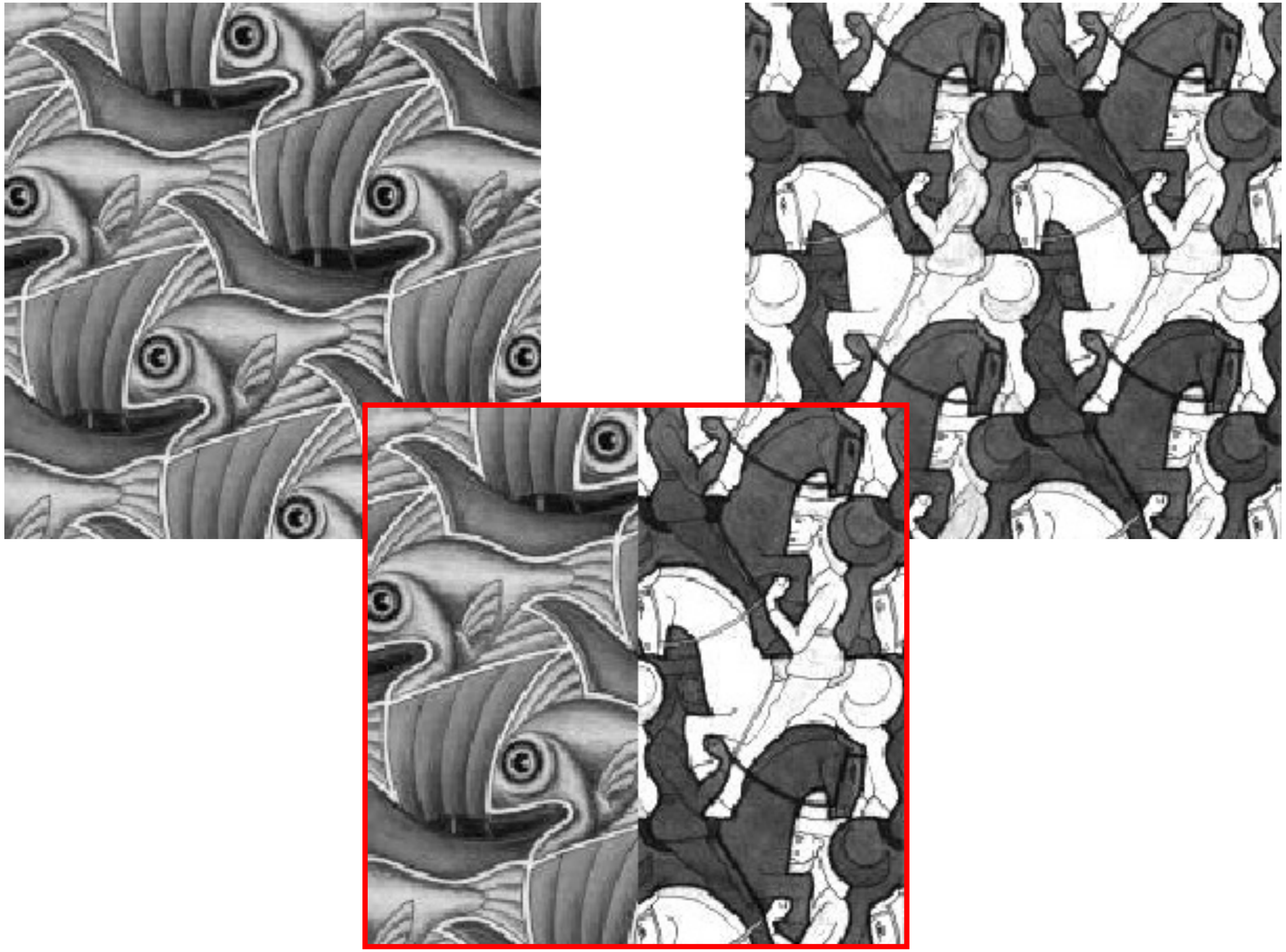
Blending

- We've aligned the images – now what?

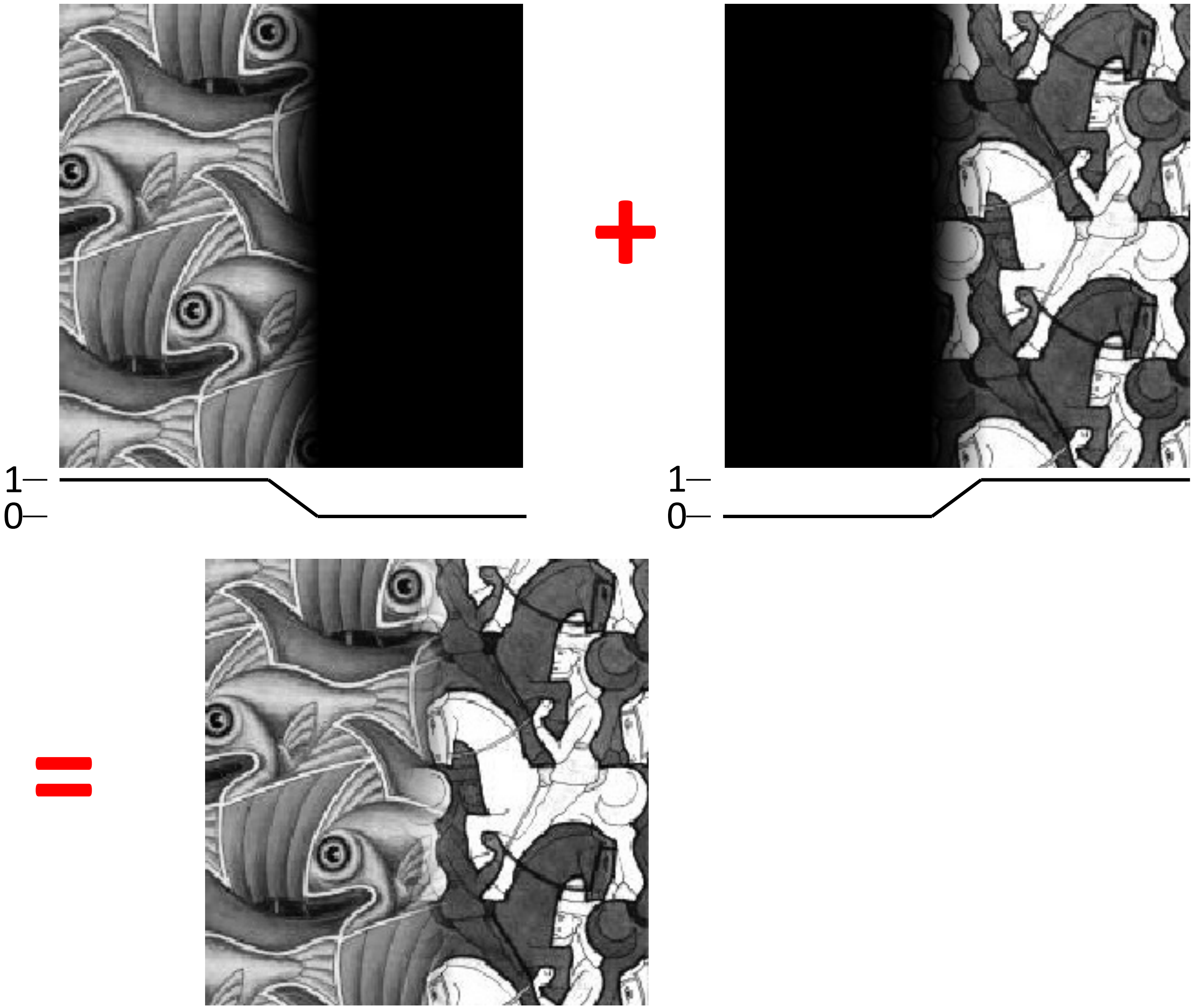
Blending

- Want to seamlessly blend them together

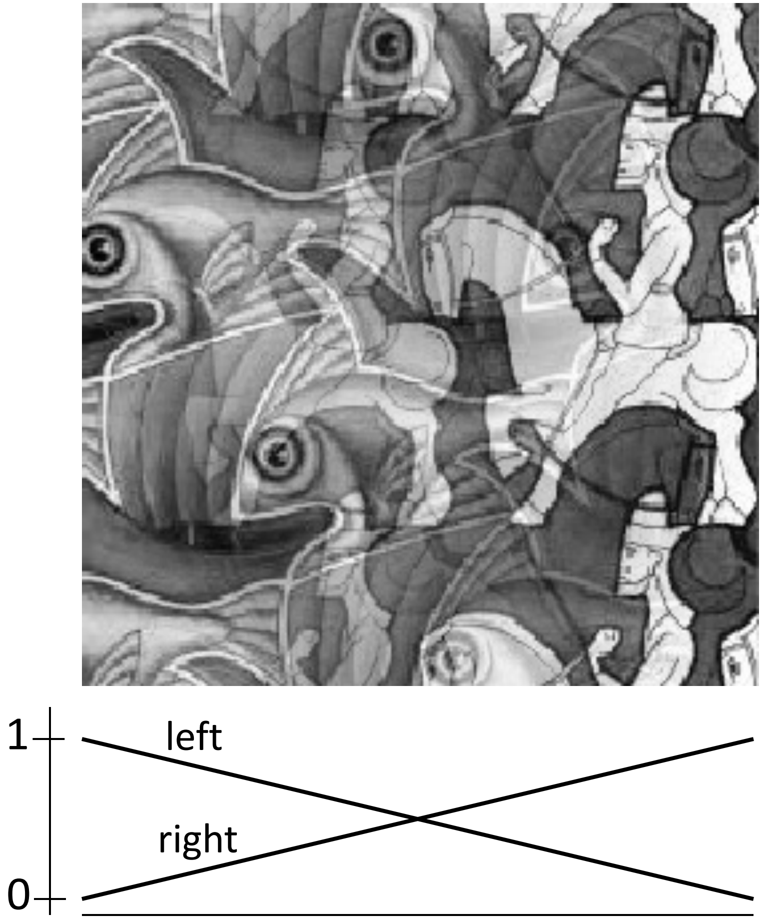
Image Blending

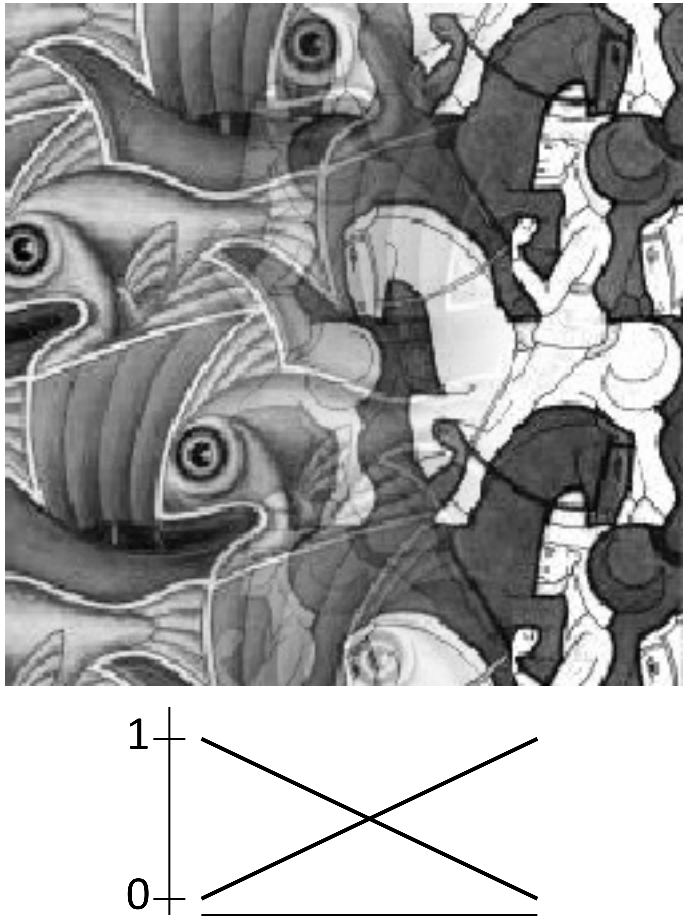


Feathering

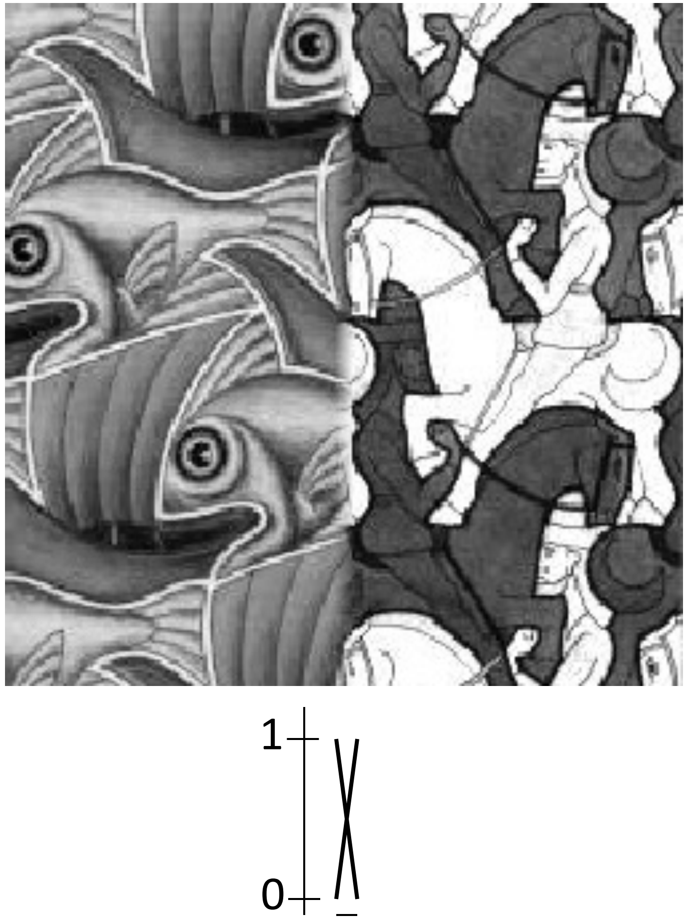


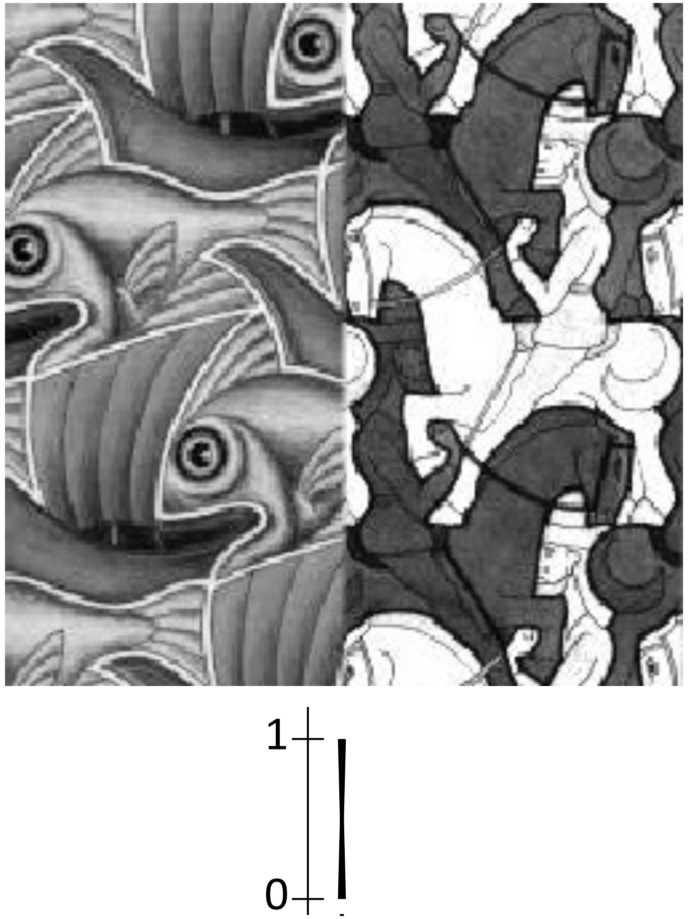
Effect of window size



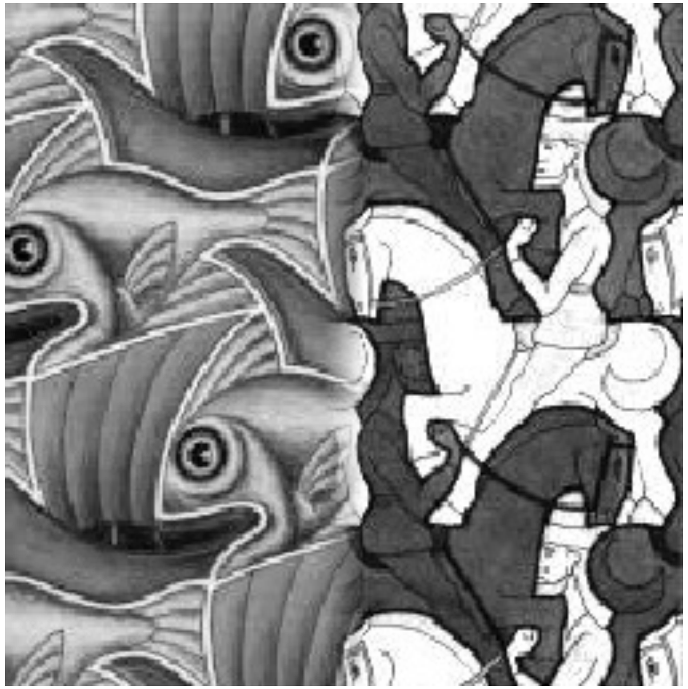


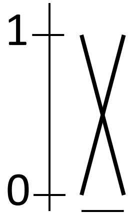
Effect of window size





Good window size

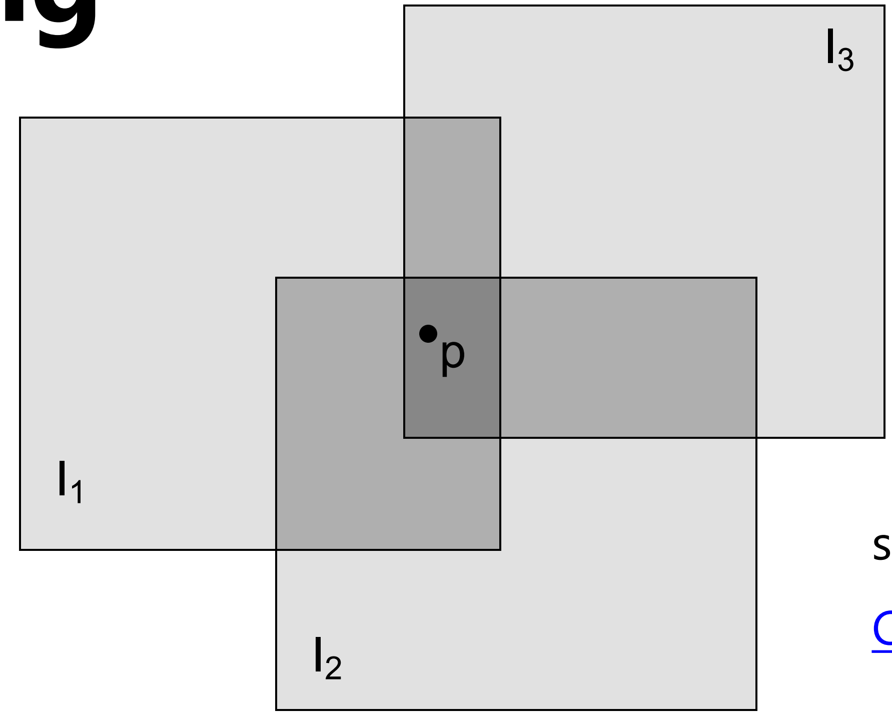




"Optimal" window: smooth but not ghosted

- Doesn't always work...

Alpha Blending



see Blinn (CGA, 1994) for details:

[Compositing, Part 1: Theory]

Encoding blend weights: $I(x,y) = (\alpha R, \alpha G, \alpha B, \alpha)$

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 steps:

1. accumulate: add up the (α premultiplied) RGBα values at each pixel
2. normalize: divide each pixel's accumulated RGB by its α value

   Q: what if $\alpha = 0$?

Poisson Image Editing

For more info: [Perez et al, SIGGRAPH 2003]

Some panorama examples

“Before SIGGRAPH Deadline” Photo credit: Doug Zongker

Some panorama examples

- Every image on Google Streetview

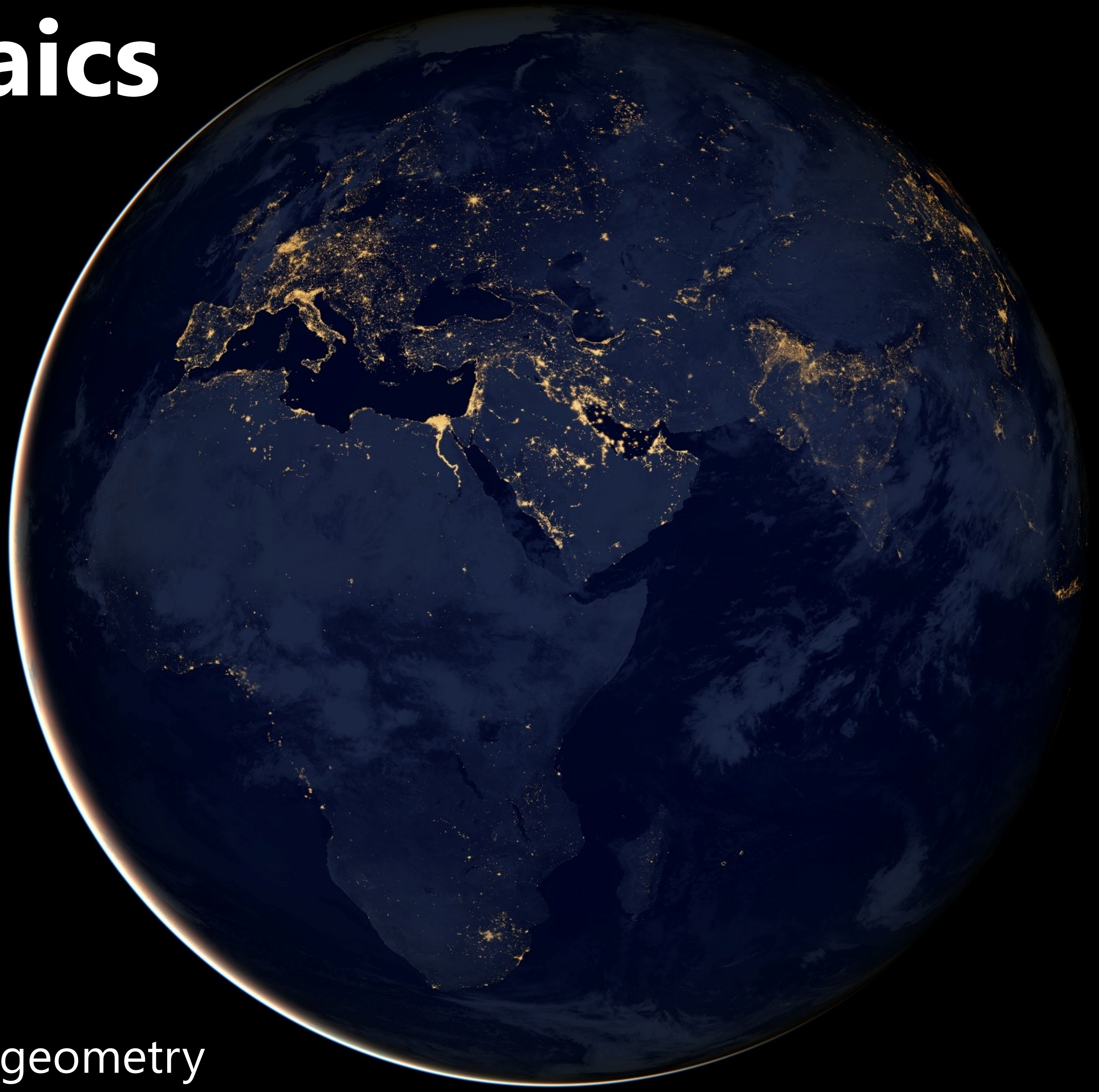
Magic: ghost removal

M. Uyttendaele, A. Eden, and R. Szeliski.
Eliminating ghosting and exposure artifacts in image mosaics.
ICCV 2001

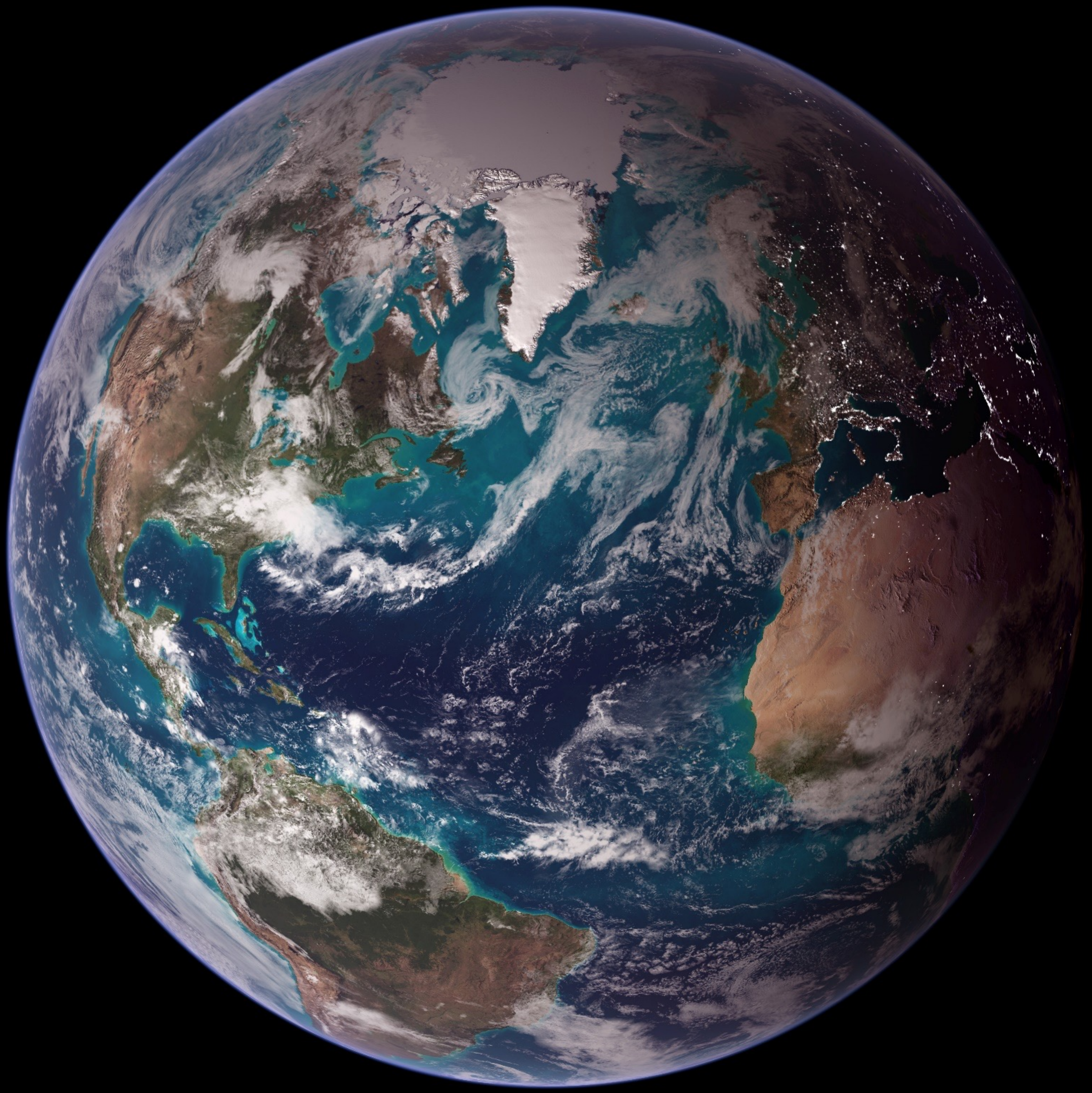
Magic: ghost removal

M. Uyttendaele, A. Eden, and R. Szeliski.
Eliminating ghosting and exposure artifacts in image mosaics.
ICCV 2001

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



https://www.nasa.gov/centers/wallops/news/frozen_sos.ht
ml

Science on a Sphere

https://news.vcu.edu/article/Science_On_a_Sphere_now_at_VCU_offers_a_world_of_possibilities

Questions?

Project 3 Demo