CS5670: Computer Vision

Noah Snavely

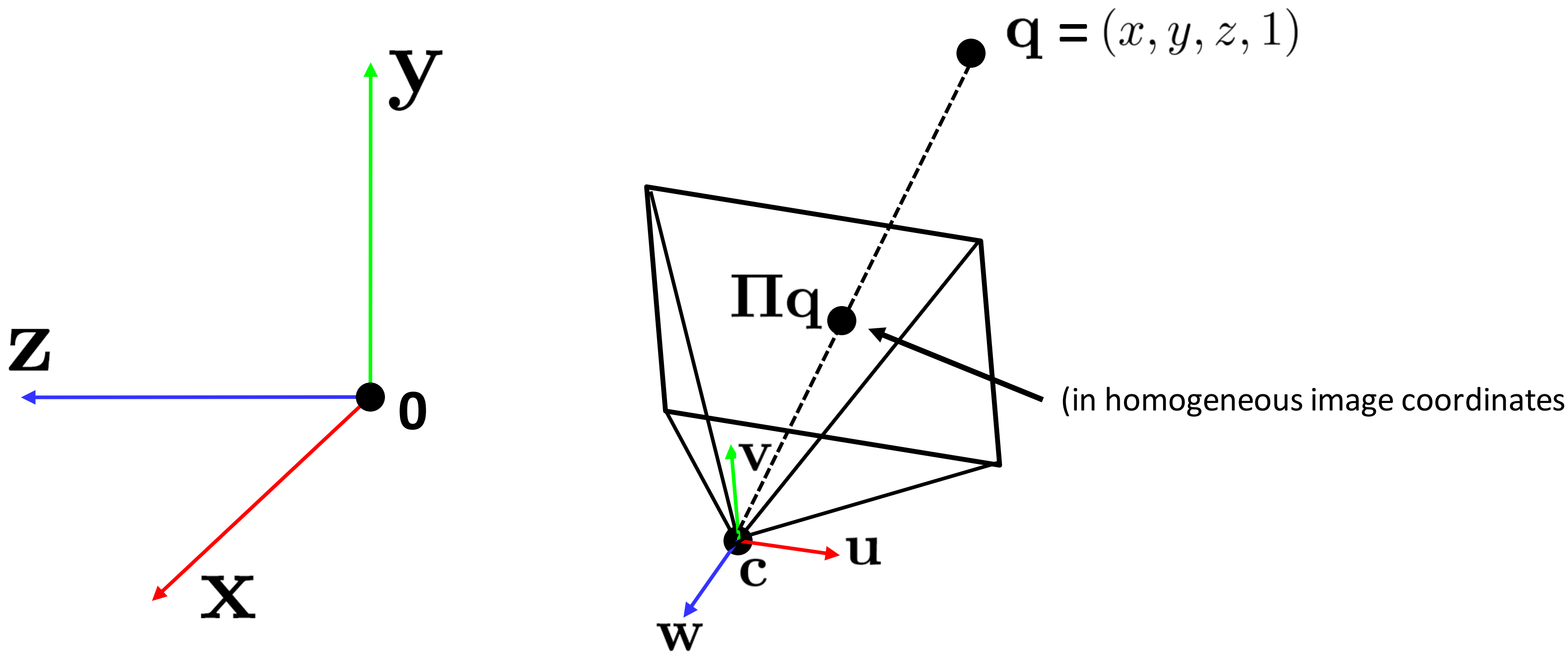
Panoramas

What's inside your fridge?

Announcements

- Project 3 (Autostitch) is out, due next Thursday, March 28 by 11:59pm
  - Artifact due Friday, March 29 by 11:59pm
  - Project to be done in pairs
- Take-home midterm
  - To be distributed at the end of class
  - Due at the beginning of class in one week, Wednesday, March 20

Camera projection matrix: recap



Projection matrix

$$\mathbf{\Pi} = \underbrace{\mathbf{K}}_{\text{intrinsics}} \underbrace{\underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\begin{bmatrix} & \mathbf{R} & & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} & \mathbf{I}_{3\times 3} & & -\mathbf{c} \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{translation}}}$$

The **K** matrix converts 3D rays in the camera's coordinate system to 2D image points in image (pixel) coordinates.

This part converts 3D points in world coordinates to 3D rays in the camera's coordinate system. There are 6 parameters represented (3 for position/translation, 3 for rotation).

Projection matrix

$$\boldsymbol{\Pi} = \underset{\text{intrinsics}}{\mathbf{K}} \underbrace{\underset{\text{projection}}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}} \underset{\text{rotation}}{\begin{bmatrix} \mathbf{R} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix}} \underset{\text{translation}}{\begin{bmatrix} \mathbf{I}_{3\times 3} & -\mathbf{c} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix}}}_{\left[\ \mathbf{R} \ \middle| \ \underbrace{-\mathbf{R}\mathbf{c}}_{(\mathbf{t} \text{ in book's notation})}\ \right]}$$

$$\boldsymbol{\Pi} = \mathbf{K} \left[\ \mathbf{R} \ \middle| \ -\mathbf{R}\mathbf{c}\ \right]$$

Typical intrinsics matrix

$$\mathbf{K} = \begin{bmatrix} -f & 0 & x_c \\ 0 & -f & y_c \\ 0 & 0 & 1 \end{bmatrix}$$

- **2D affine transform** corresponding to a scale by *f* (focal length) and a translation by (x_c, y_c) (principal point)

Questions?

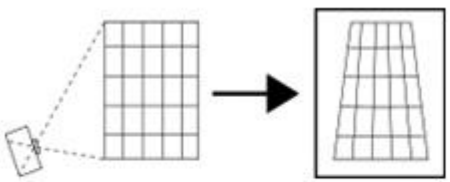
Perspective distortion

- Problem for architectural photography: converging verticals

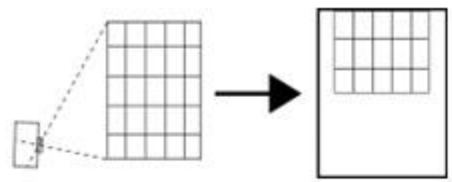
Source: F. Durand

Perspective distortion

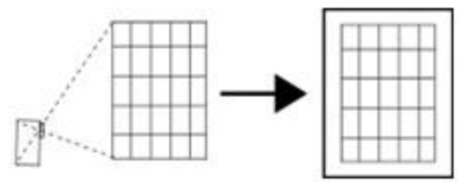
- Problem for architectural photography: converging verticals



Tilting the camera upwards results in converging verticals

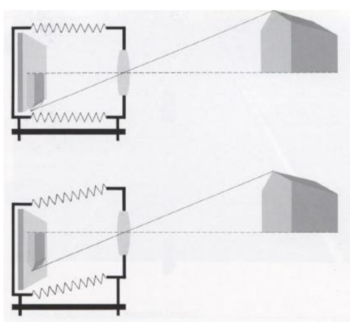


Keeping the camera level, with an ordinary lens, captures only the bottom portion of the building



Shifting the lens upwards results in a picture of the entire subject

- Solution: view camera (lens shifted w.r.t. film)



(Corresponds to shifting the *principal point*)

http://en.wikipedia.org/wiki/Perspective_correction_lens

Source: F. Durand

Perspective distortion

- Problem for architectural photography: converging verticals
- Result:

Source: F. Durand

Perspective distortion

- What does a sphere project to?

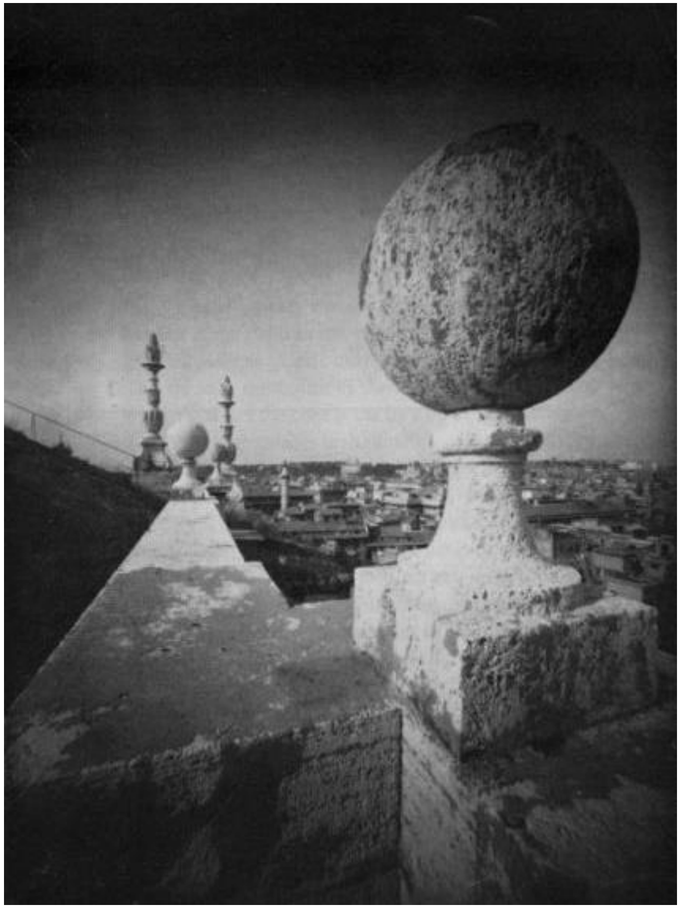
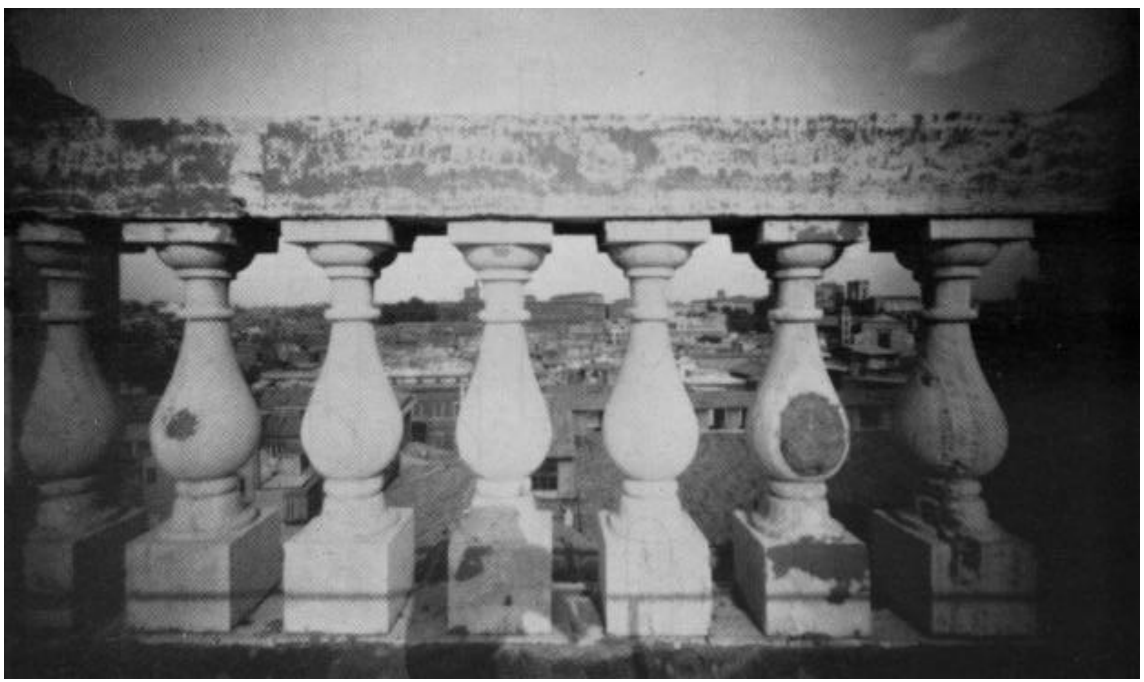
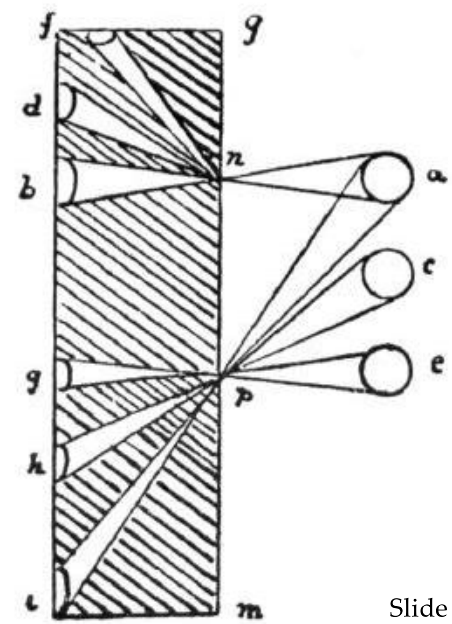


Image source: F. Durand

Perspective distortion

- The exterior columns appear bigger
- The distortion is not due to lens flaws
- Problem pointed out by Da Vinci





Slide by F. Durand

Perspective distortion: People

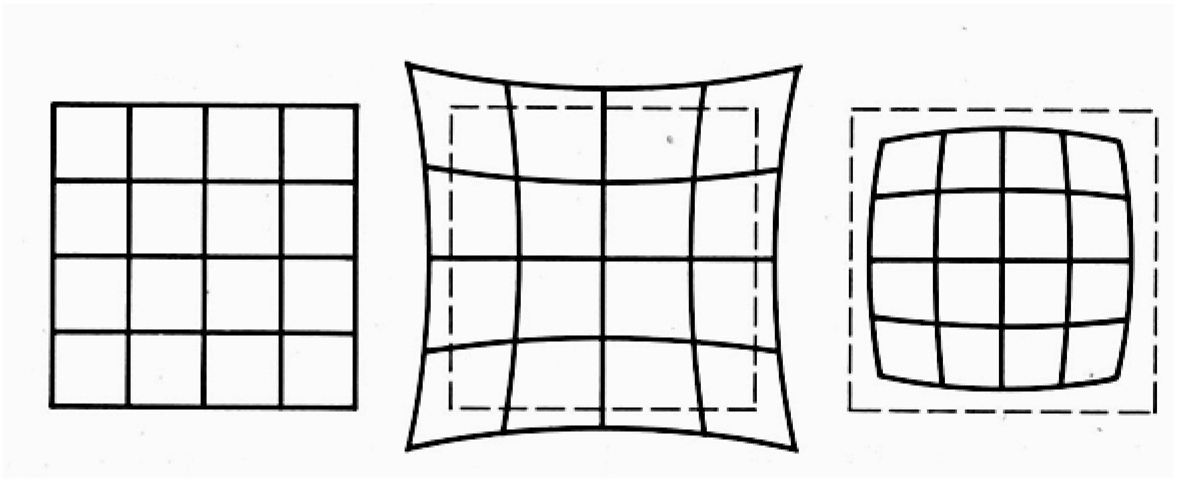
Wide angle

Standard

Telephoto

http://petapixel.com/2013/01/11/how-focal-length-affects-your-subjects-apparent-weight-as-seen-with-a-cat/

Lens distortion



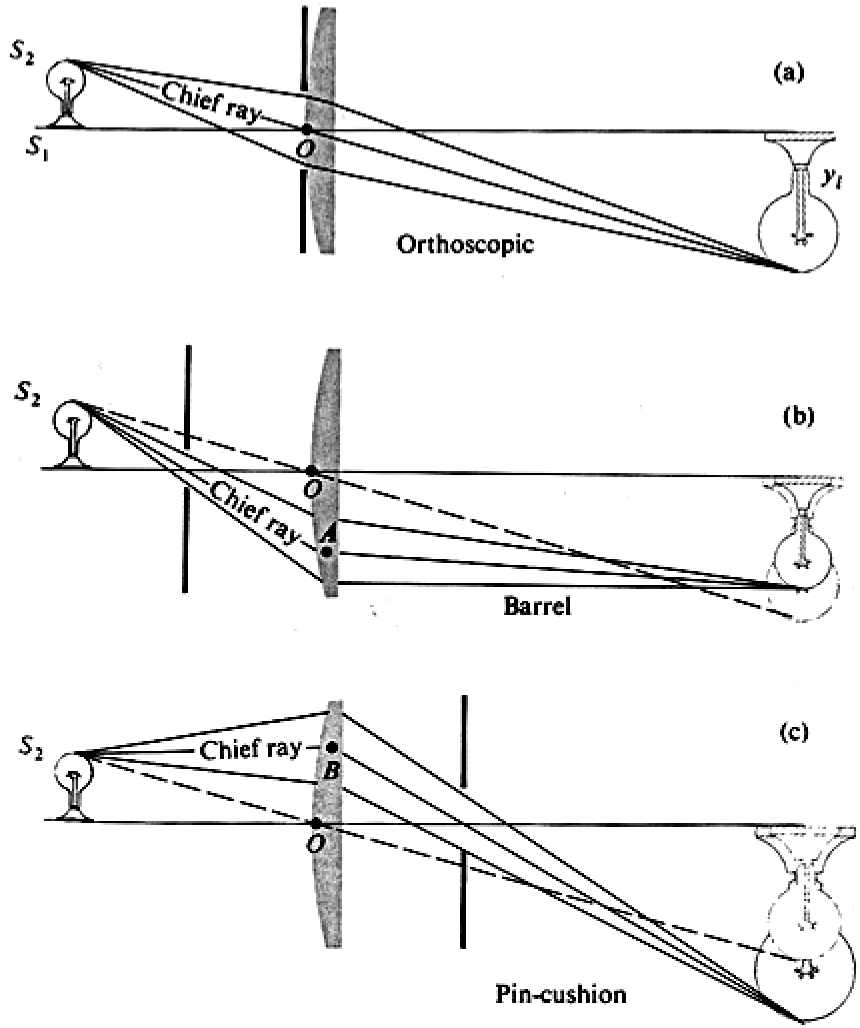
No distortion Pin cushion Barrel

- **Radial distortion** of the image
  - Caused by imperfect lenses
  - Points are distorted along radial lines
  - Deviations are most noticeable for rays that pass through the edge of the lens

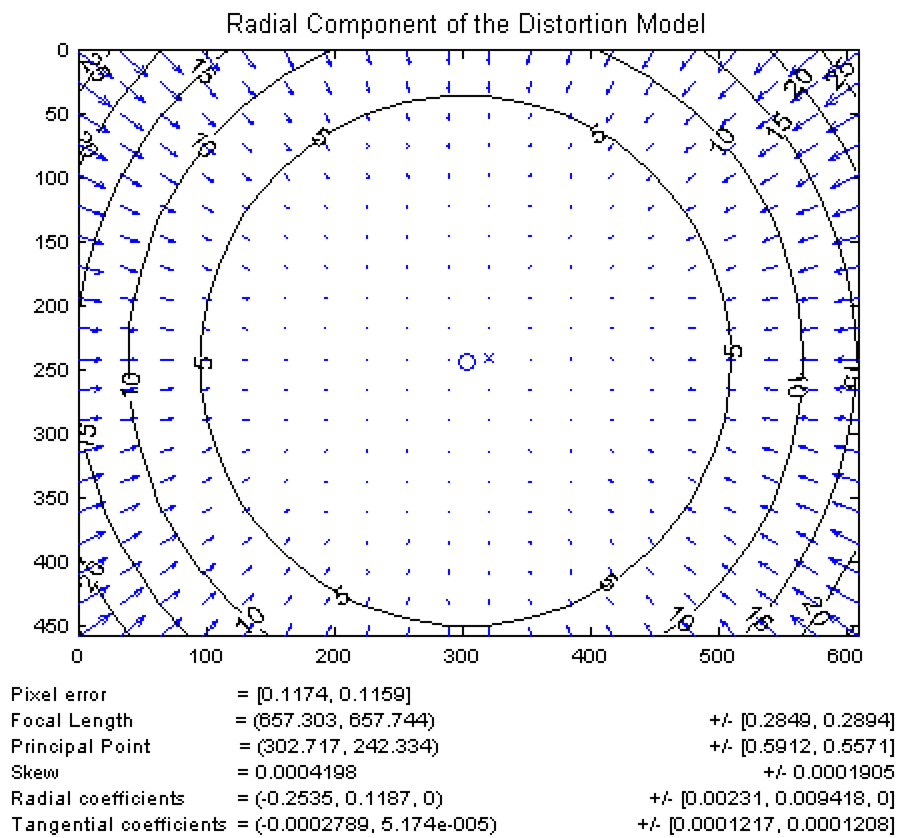

©2004 Vincent Bockaert 123di.com


©2004 Vincent Bockaert 123di.com

Radial distortion



Radial distortion



- Arrows show motion of projected points relative to an ideal (distortion-free lens)

[Image credit: J. Bouguet http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example.html]

Correcting radial distortion

from [Helmut Dersch](#)

Modeling distortion

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

$$
\begin{aligned}
x'_n &= \widehat{x}/\widehat{z} \\
y'_n &= \widehat{y}/\widehat{z}
\end{aligned}
$$

Apply radial distortion by approximating with a (low-degree) polynomial

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

Apply focal length & translate image center

$$
\begin{aligned}
x' &= f x'_d + x_c \\
y' &= f y'_d + y_c
\end{aligned}
$$

- To model lens distortion
  - Use the above conversion of rays to pixels, rather than simply multiplying by the intrinsics matrix

Other types of projection

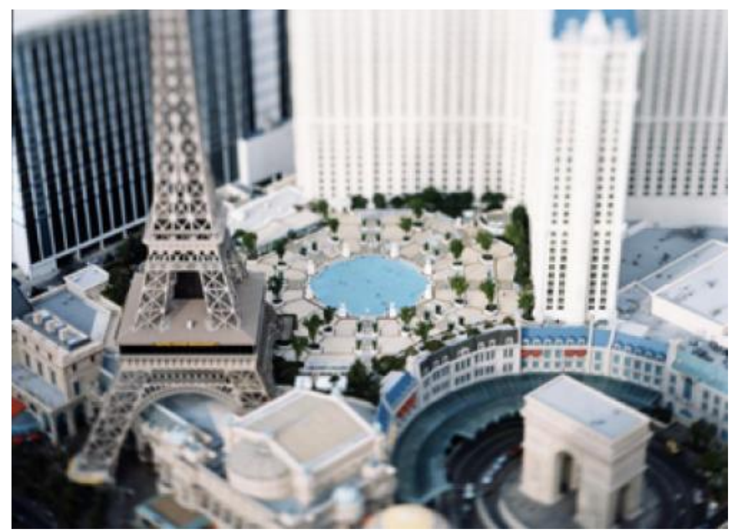
- Lots of intriguing variants…
- (I’ll just mention a few fun ones)

360 degree field of view…

- Basic approach
  - Take a photo of a parabolic mirror with an orthographic lens (Nayar)
  - Or buy one a lens from a variety of omnicam manufacturers…
    - See http://www.cis.upenn.edu/~kostas/omni.html

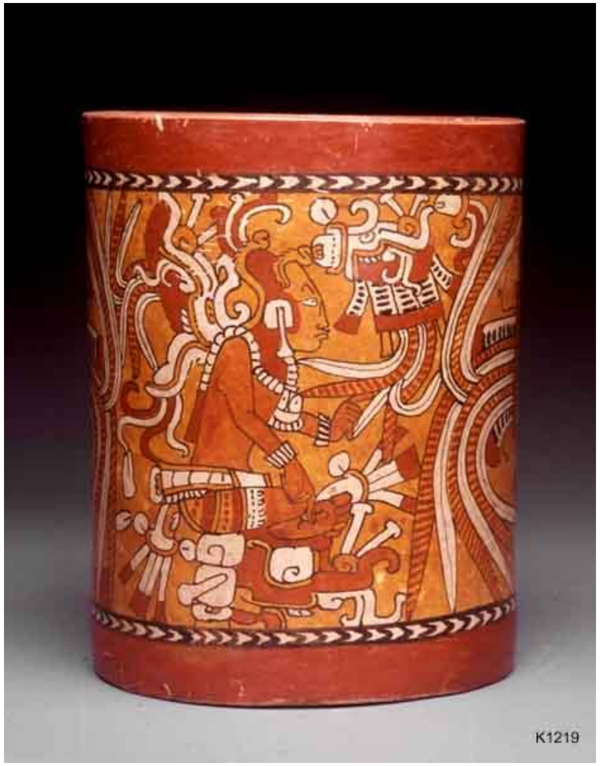
Tilt-shift

http://www.northlight-images.co.uk/article_pages/tilt_and_shift_ts-e.html



TitIt-shift images from Olivo Barbieri
and Photoshop imitations

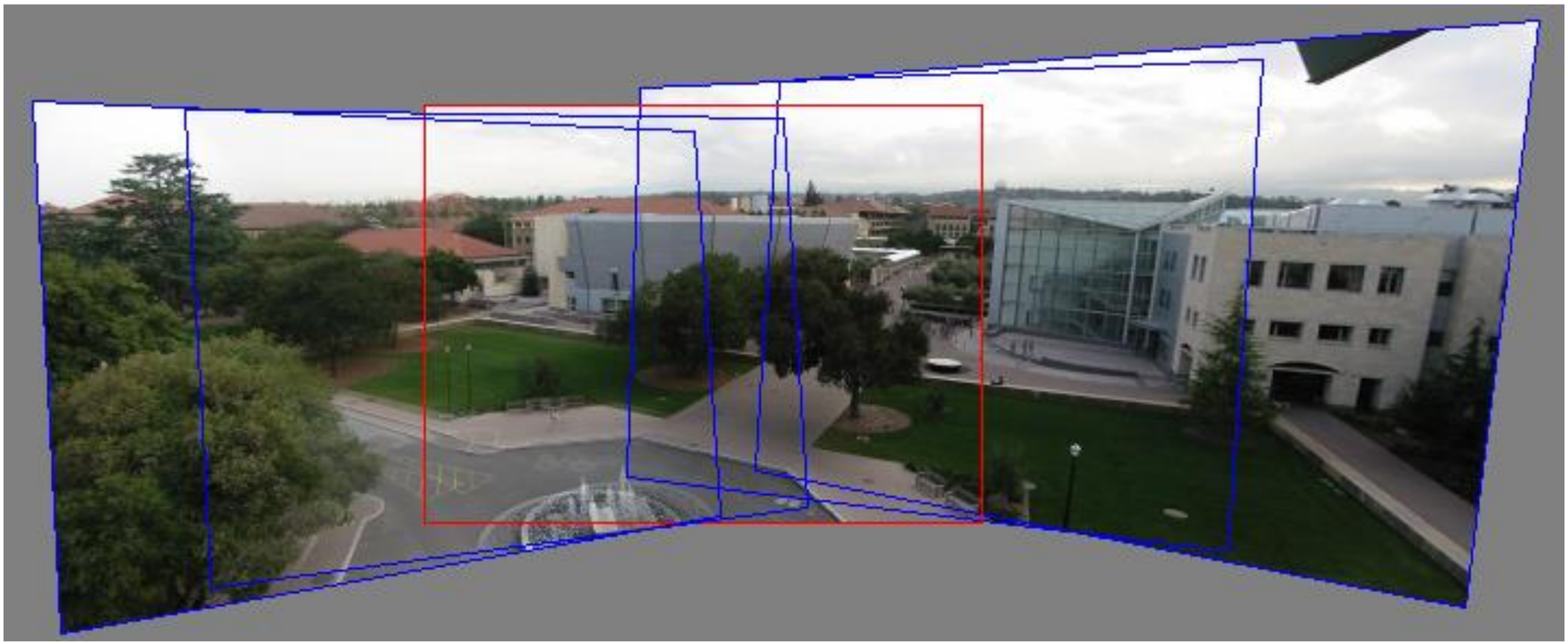
Rotating sensor (or object)



Rollout Photographs © Justin Kerr
http://research.famsi.org/kerrmaya.html

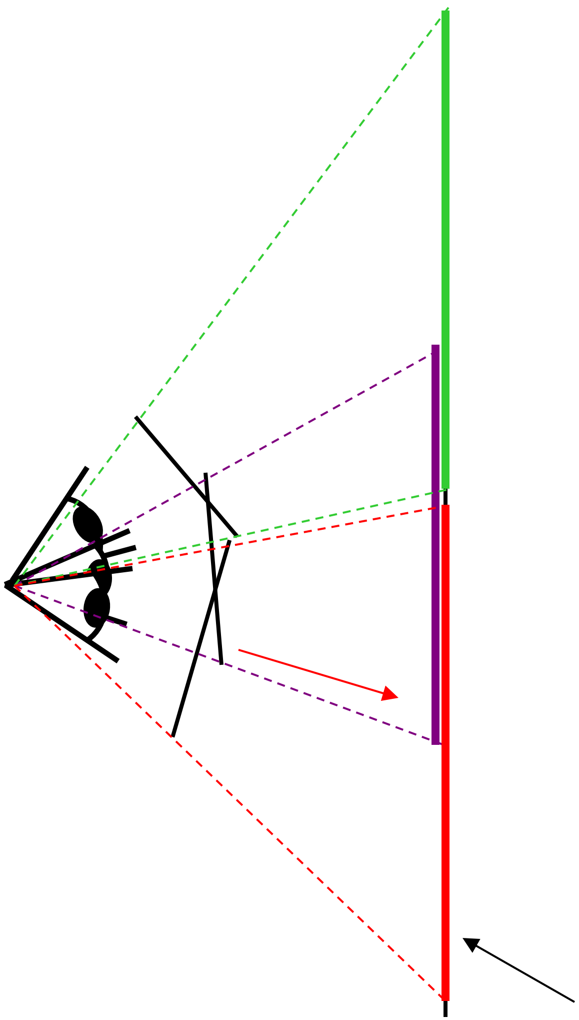
Also known as “cyclographs”, “peripheral images”

Back to panoramas

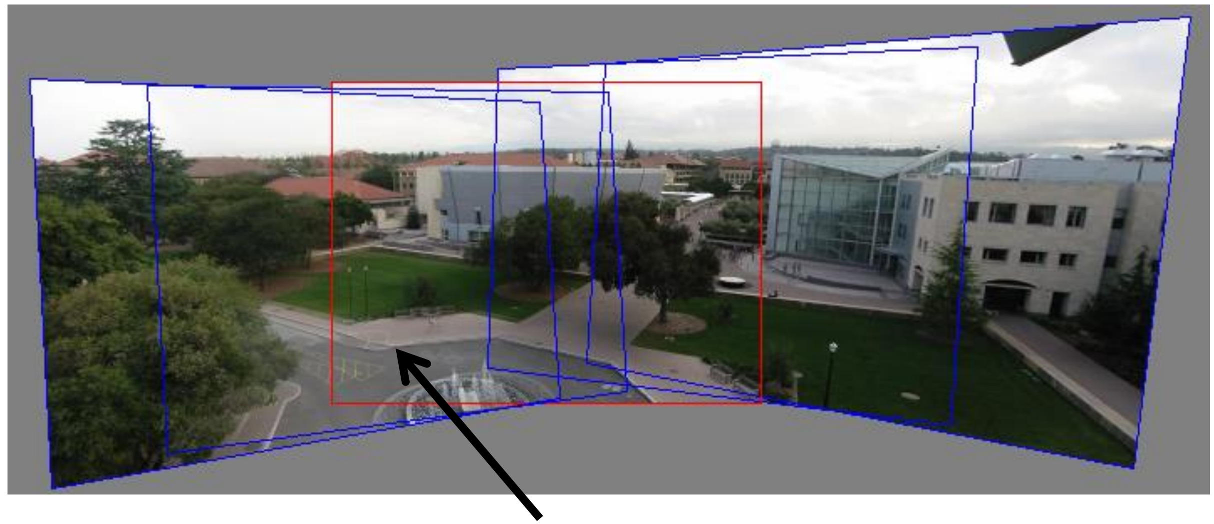


Can we use homographies to create a 360 panorama?

Idea: projecting images onto a common plane



mosaic projection plane



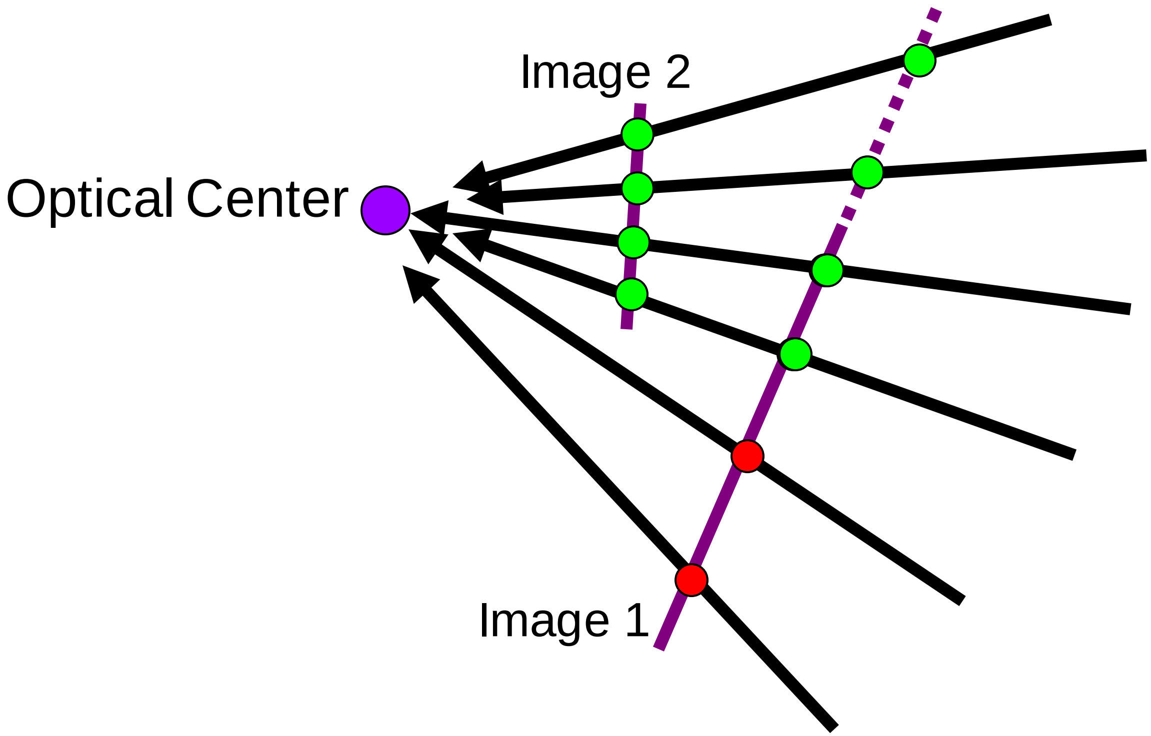
each image is warped
with a homography $\mathbf{H}$

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



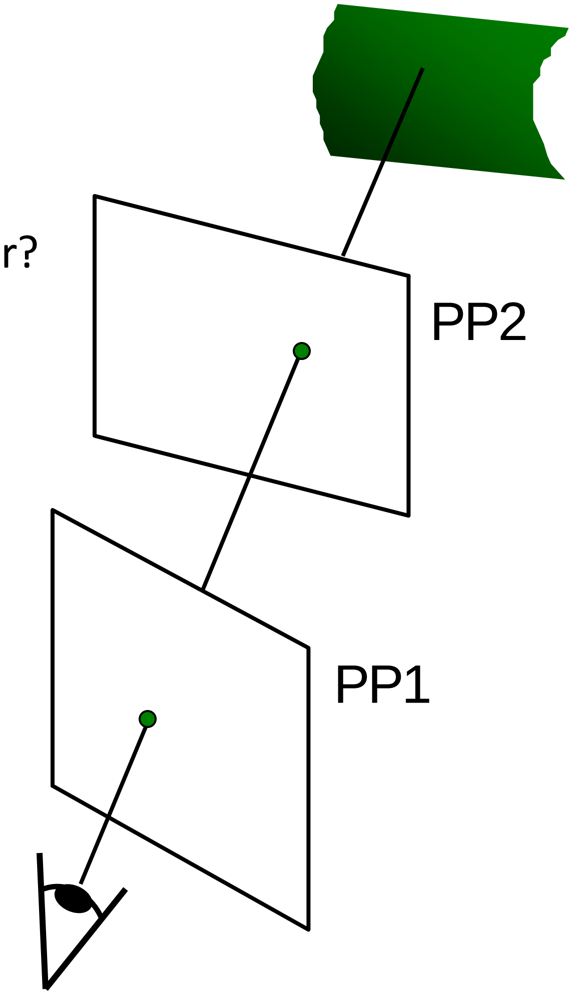
- If we capture all 360º of rays, we can create a 360º panorama
- The basic operation is *projecting* an image from one plane to another
- The projective transformation is scene-INDEPENDENT
  - This depends on all the images having the same optical center

Image reprojection

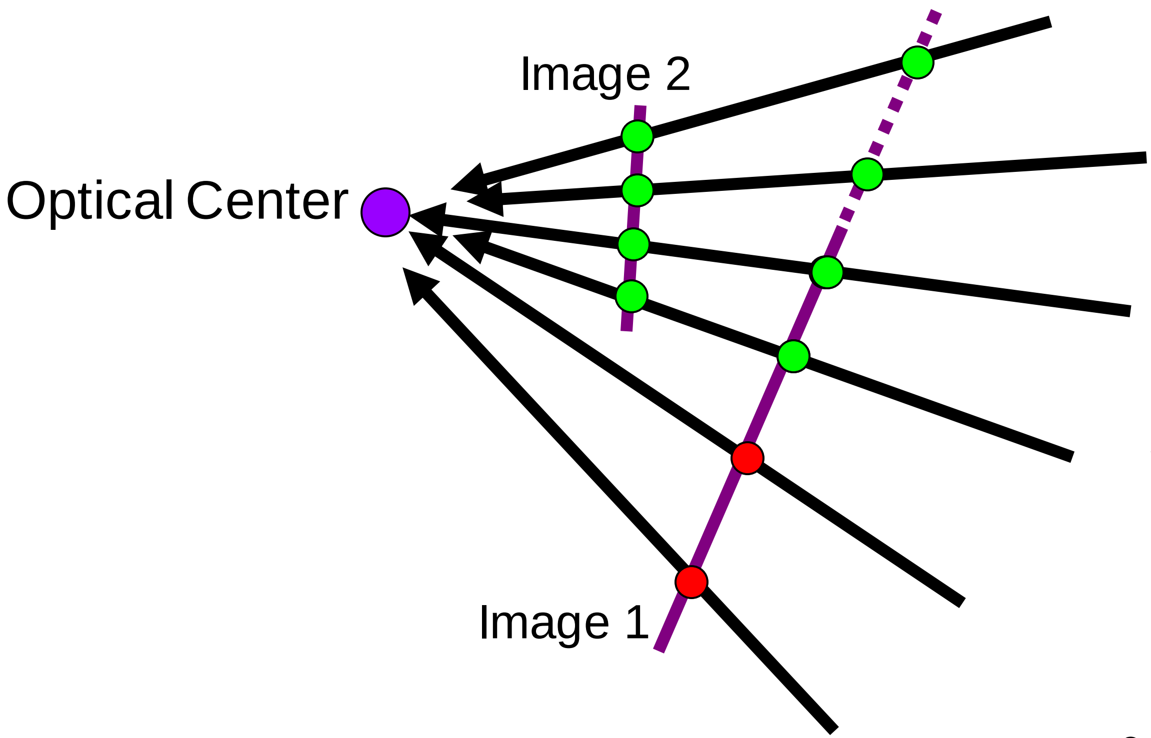
- Basic question
  - How to relate two images from the same camera center?
    - how to map a pixel from PP1 to PP2

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.

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

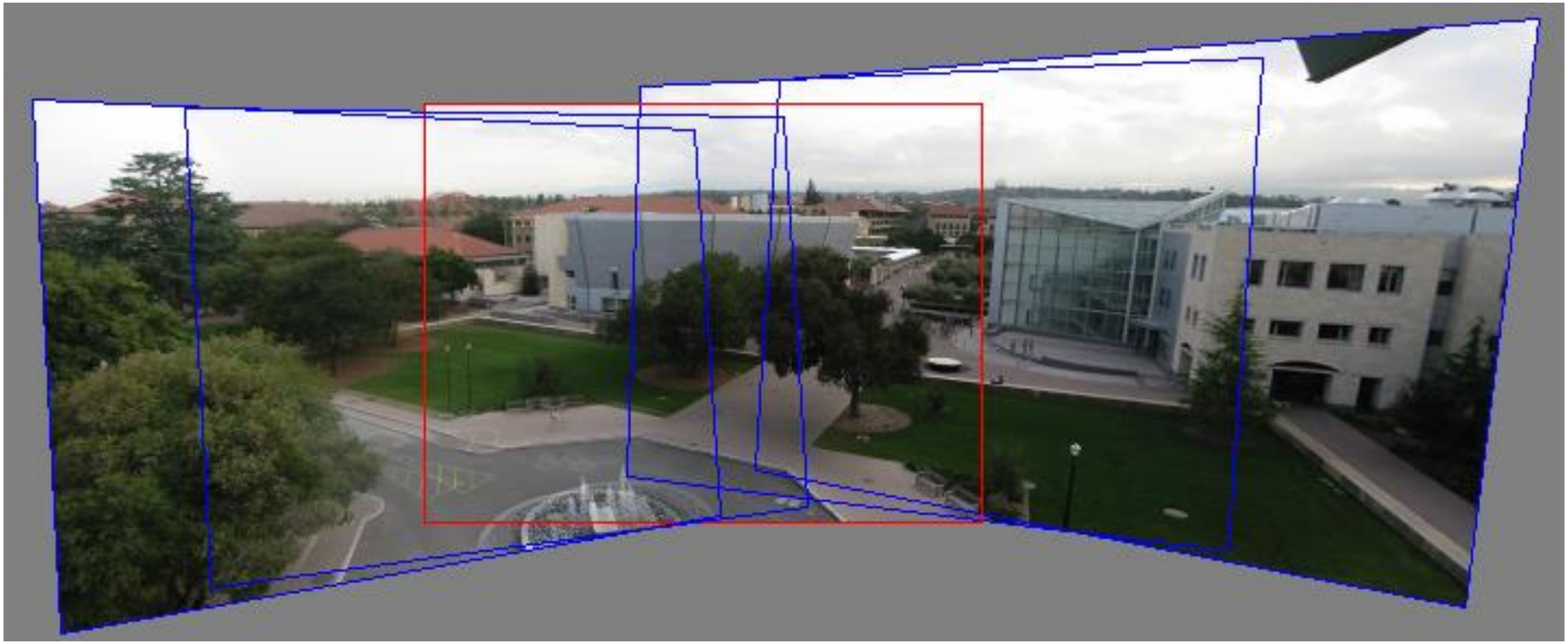
Step 3: Convert rays in camera 1's coordinates to pixels in image 1's coordinates.

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

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

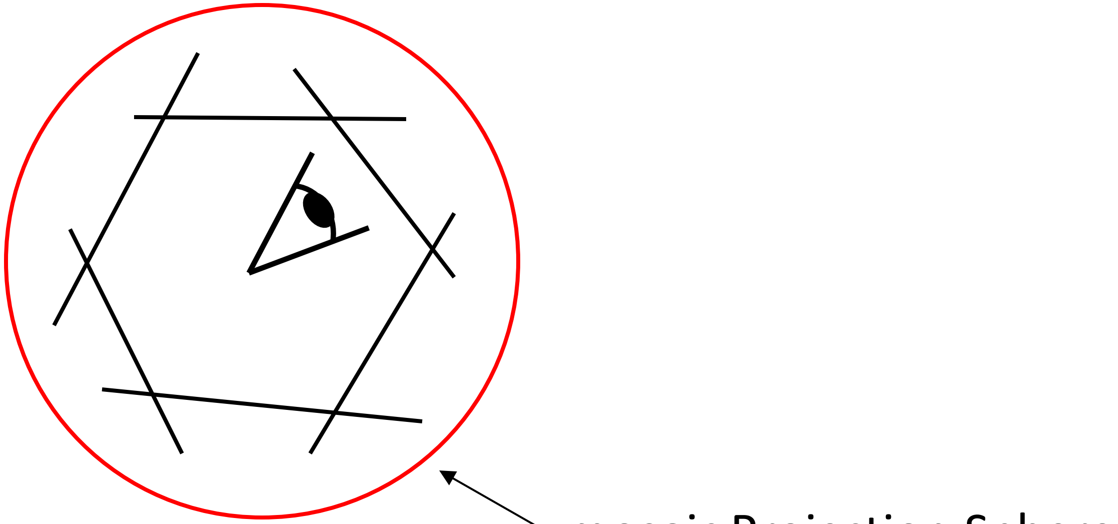
	image 1	image 2
intrinsics	$\mathbf{K}_1$	$\mathbf{K}_2$
extrinsics (rotation only)	$\mathbf{R}_1 = \mathbf{I}_{3\times 3}$	$\mathbf{R}_2$

Can we use homography to create a 360 panorama?

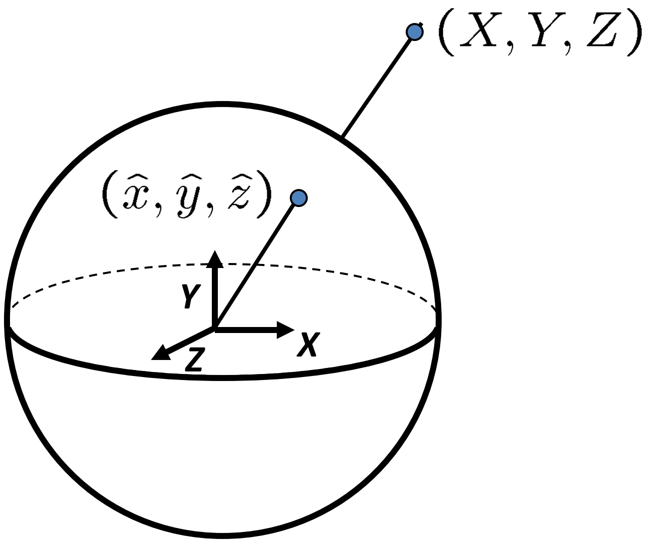


Panoramas

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



Spherical projection



unit sphere

- Map 3D point (X,Y,Z) onto sphere

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

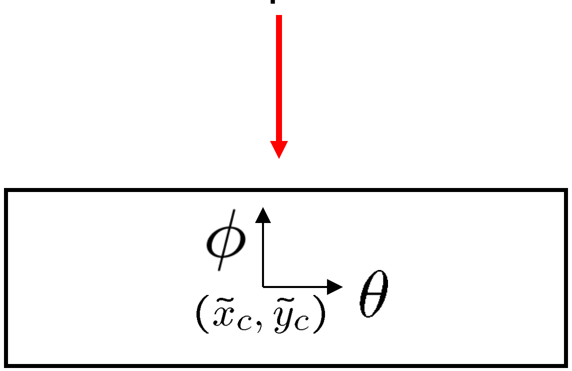
- Convert to spherical coordinates

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

- 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

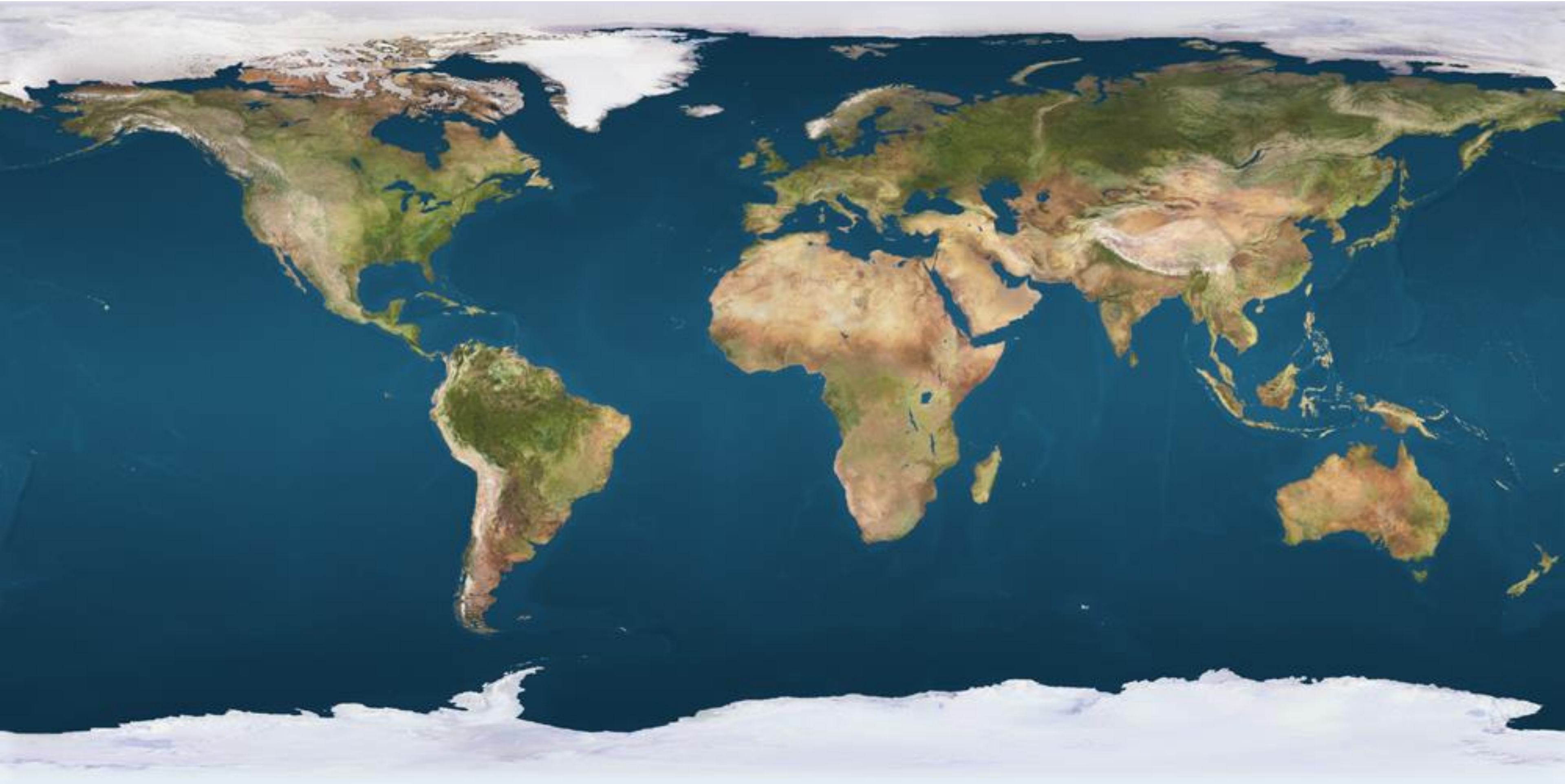


unwrapped sphere

Spherical image

Unwrapping a sphere

Credit: JHT's Planetary Pixel Emporium

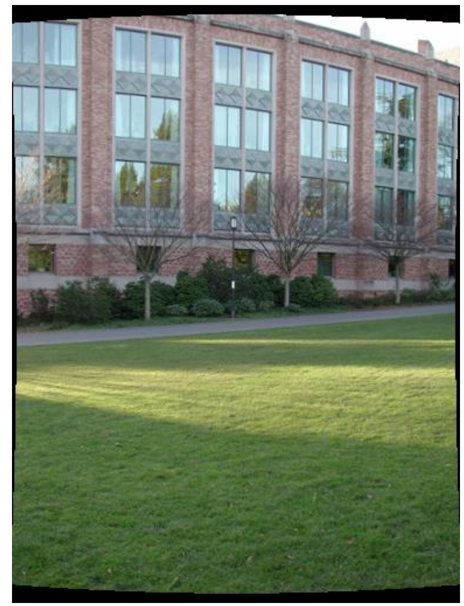


Spherical reprojection

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

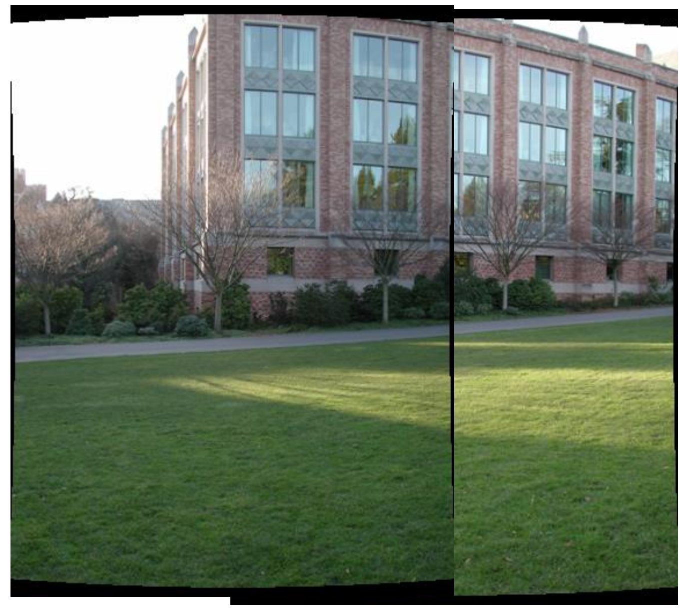
- Map image to spherical coordinates
  - need to know the focal length

Aligning spherical images



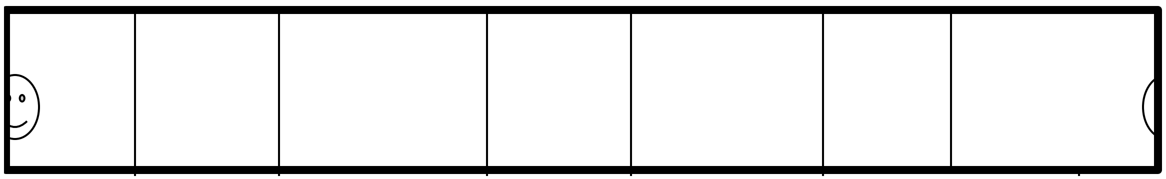
- Suppose we rotate the camera by θ about the vertical axis
  - How does this change the spherical image?

Aligning spherical images



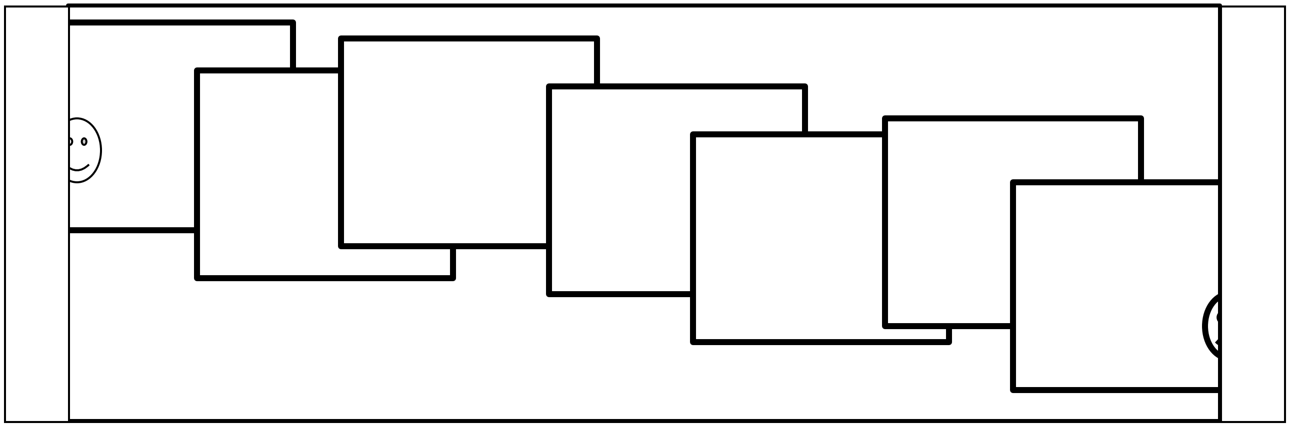
- Suppose we rotate the camera by θ about the vertical axis
  - How does this change the spherical image?
    - Translation by θ
  - This means that we can align spherical images by translation

Assembling the panorama



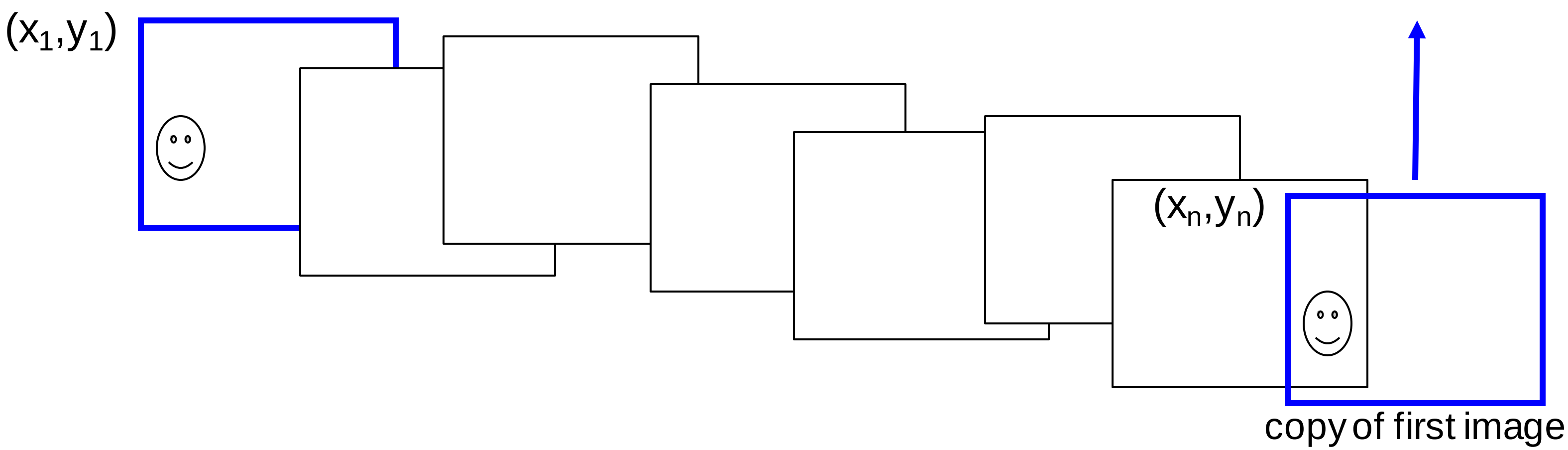
- Stitch pairs together, blend, then crop

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

- Take pictures on a tripod (or handheld)
- Warp to spherical coordinates (optional if using homographies to align images)
- Extract features
- Align neighboring pairs using RANSAC
- Write out list of neighboring translations
- Correct for drift
- Read in warped images and blend them
- 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

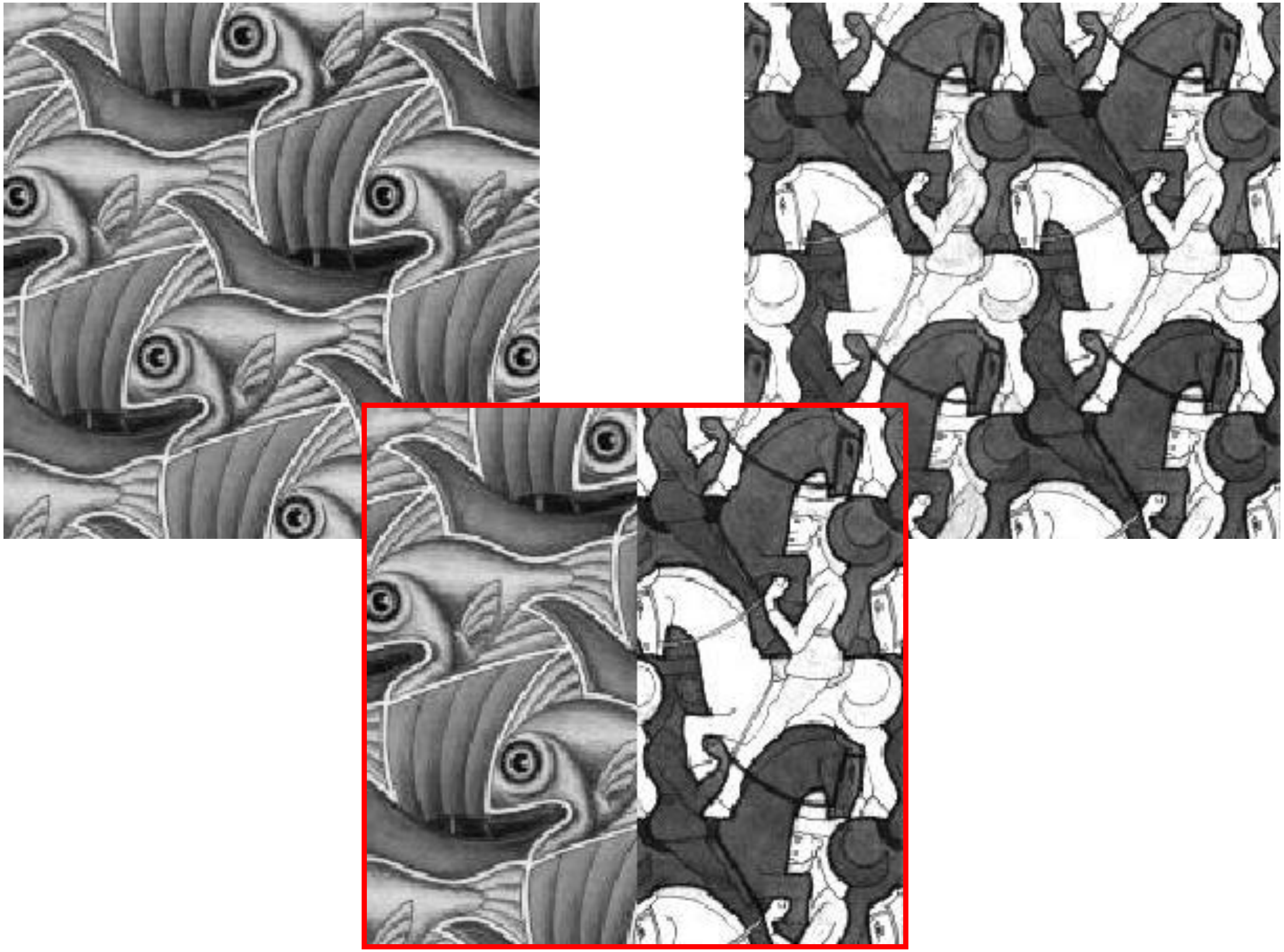
Blending

- We've aligned the images – now what?

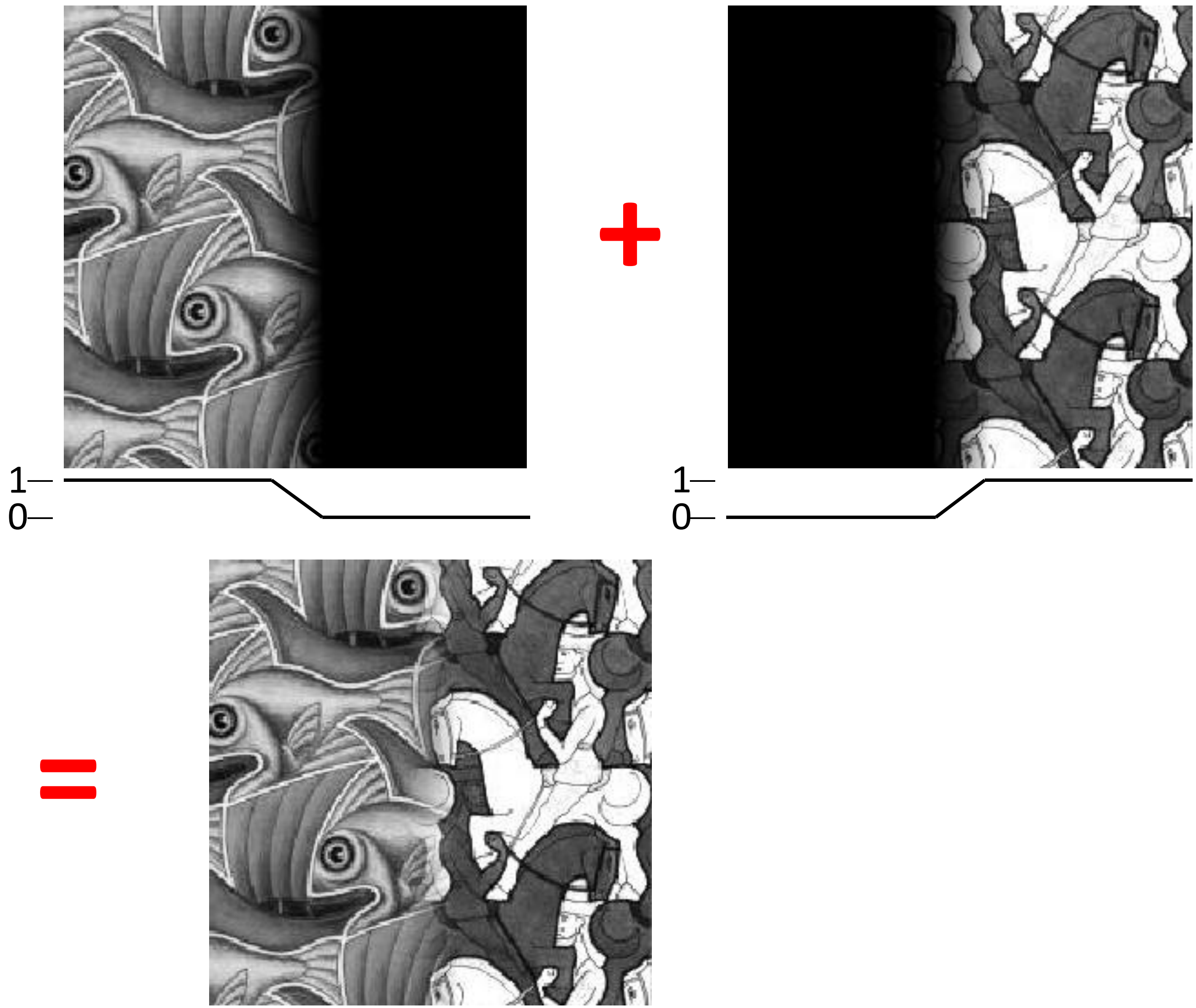
Blending

- Want to seamlessly blend them together

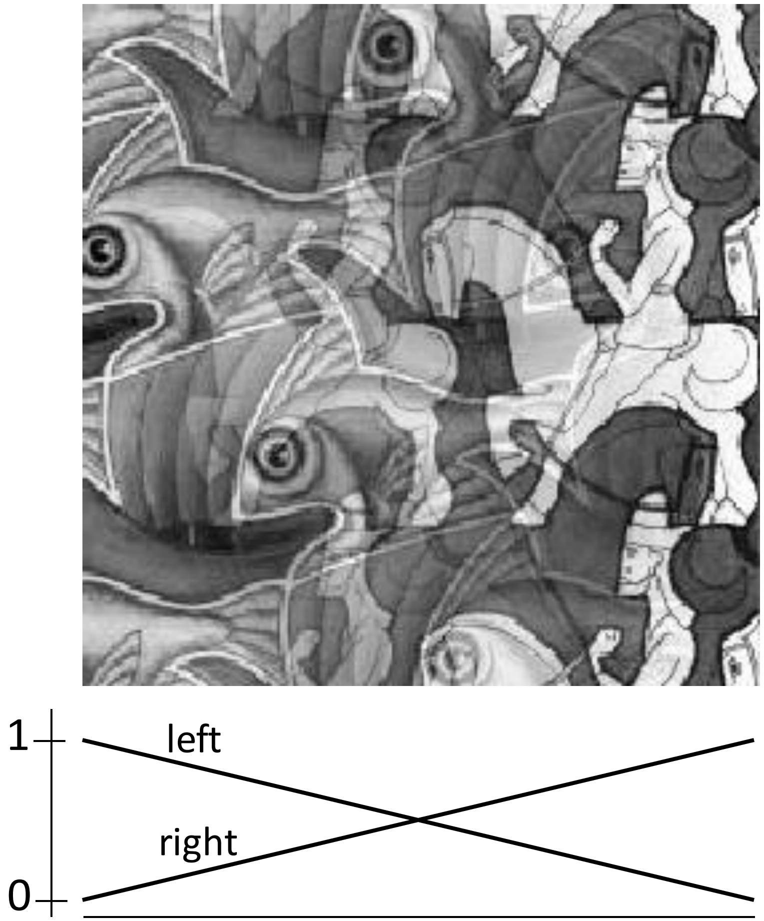
Image Blending

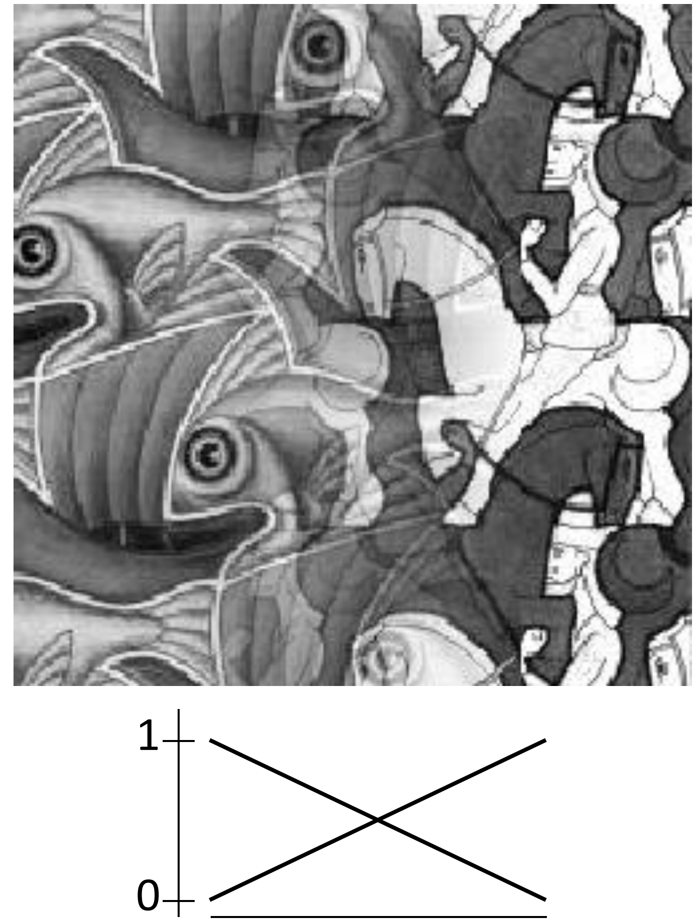


Feathering

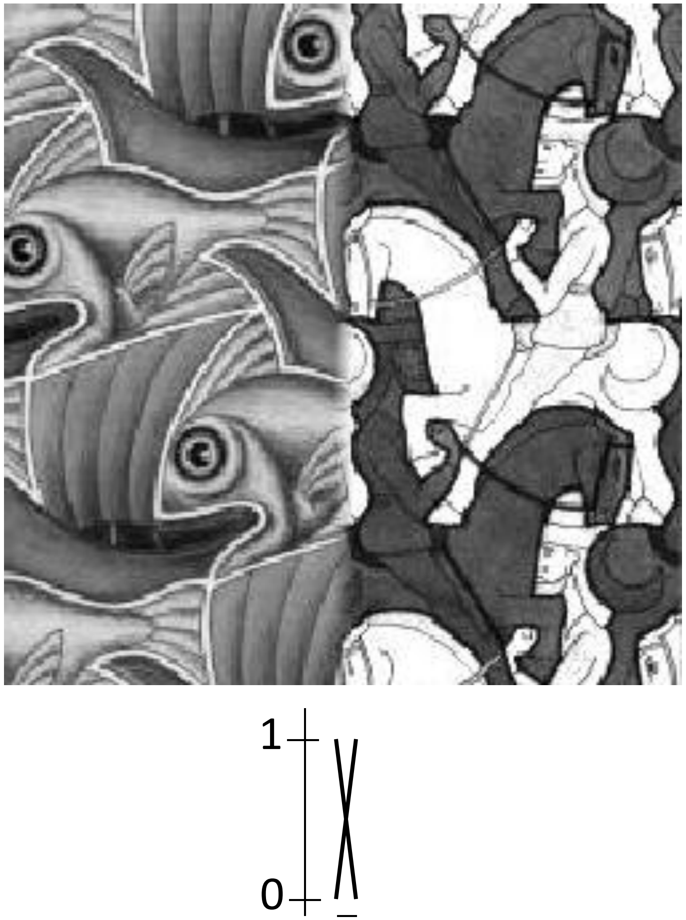


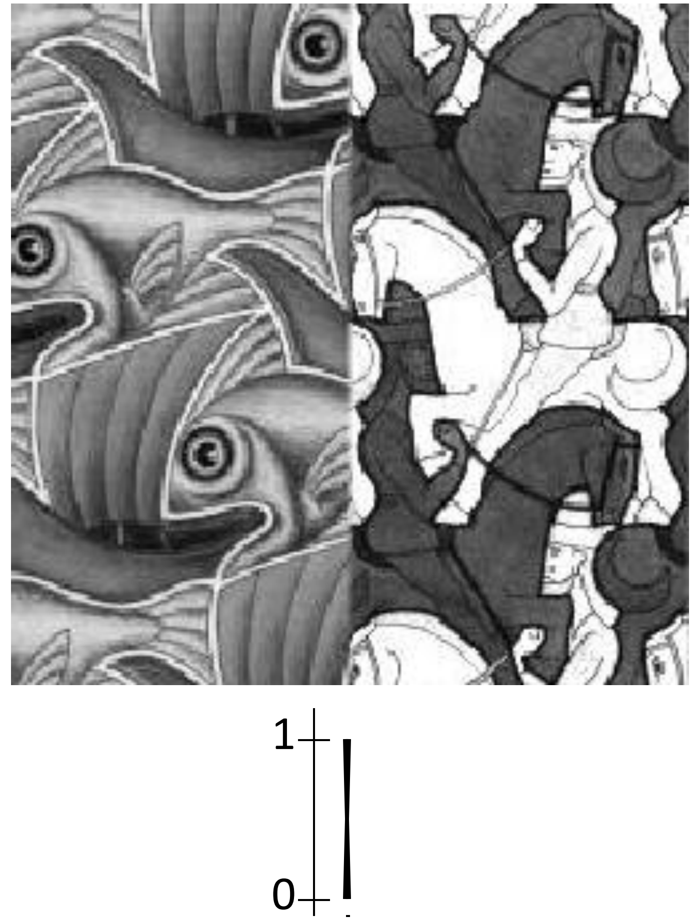
Effect of window size



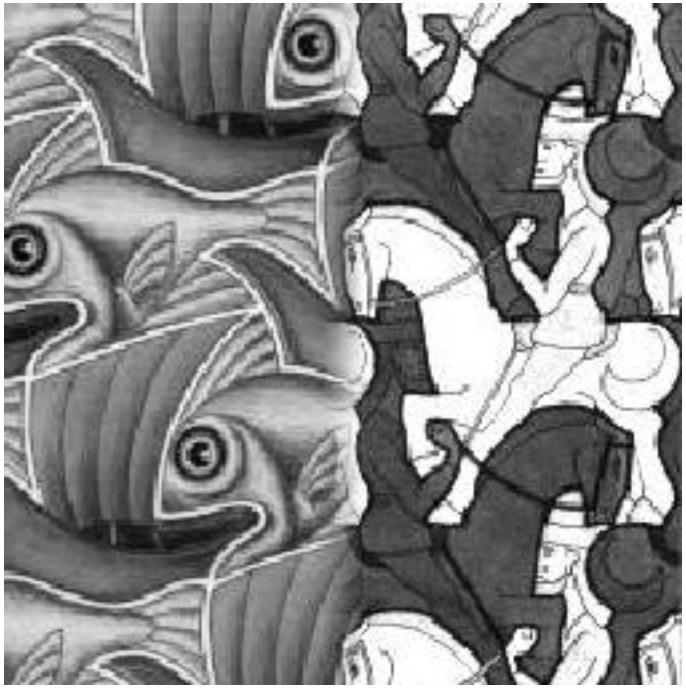


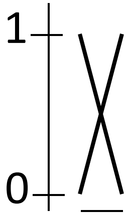
Effect of window size





Good window size

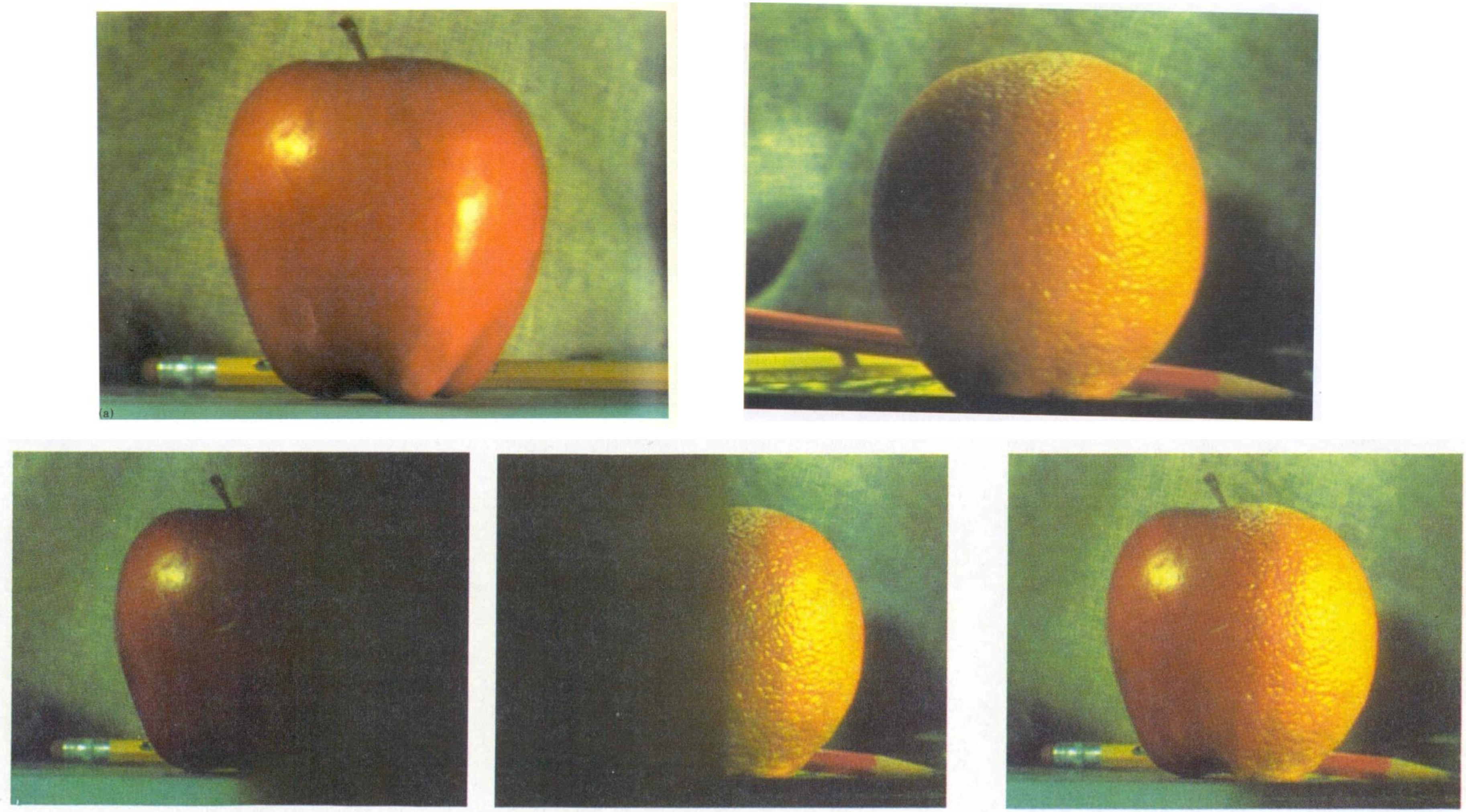




"Optimal" window: smooth but not ghosted

- Doesn't always work...

Pyramid blending



(d) (h) (l)

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.

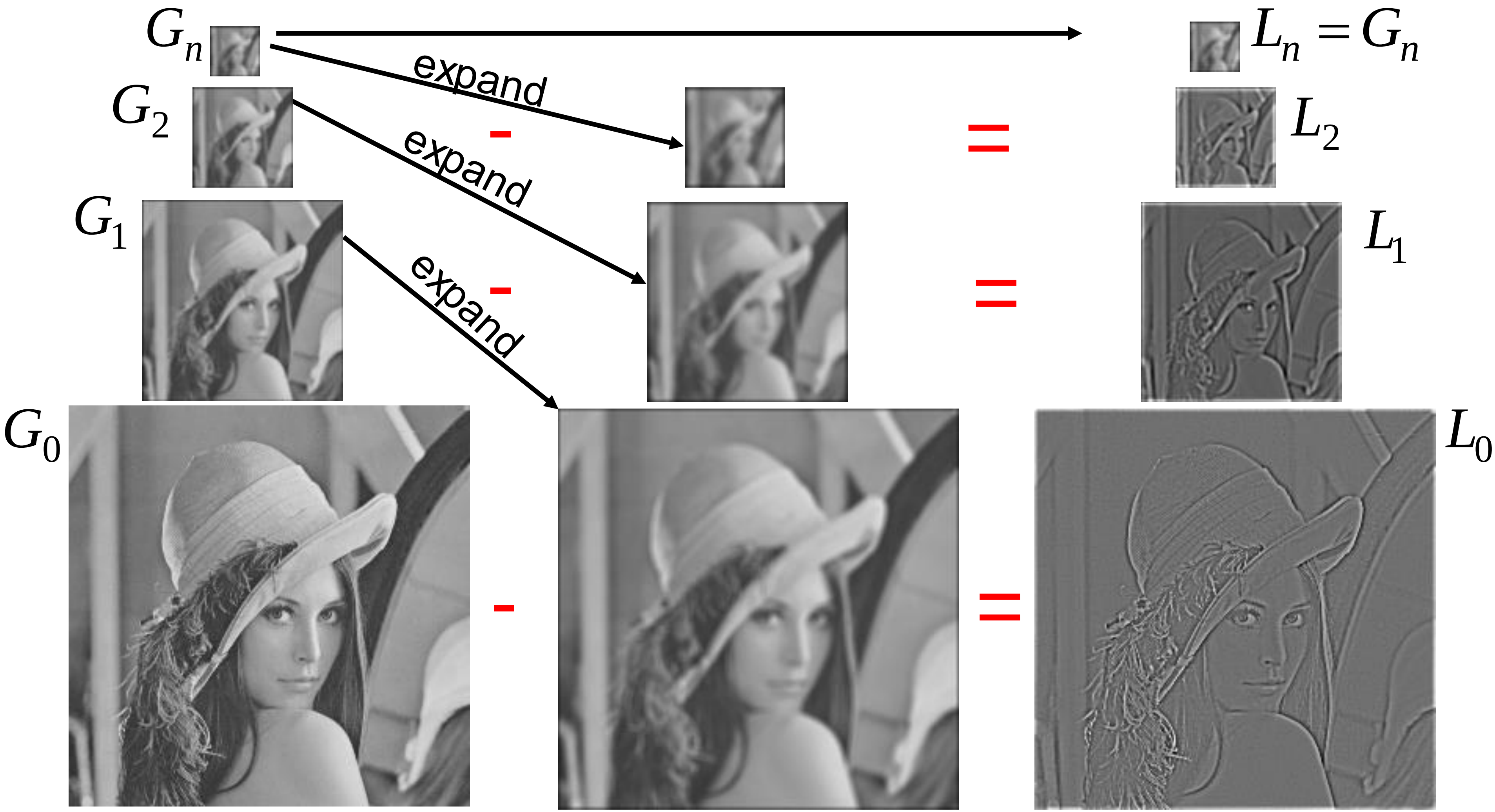
The Laplacian Pyramid

$$L_i = G_i - \text{expand}(G_{i+1})$$

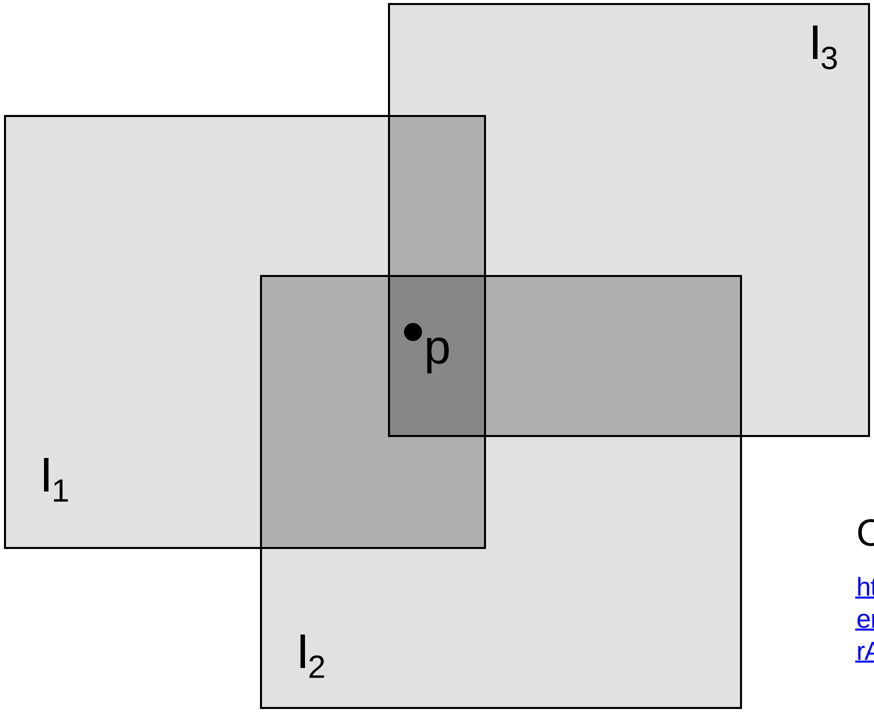
Gaussian Pyramid

$$G_i = L_i + \text{expand}(G_{i+1})$$

Laplacian Pyramid



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

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

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

sources/destinations

cloning

seamless cloning

- For more info: Perez et al, SIGGRAPH 2003
  - http://research.microsoft.com/vision/cambridge/papers/perez_siggraph03.pdf

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*.**
In Proceedings of the Interational Conference on Computer Vision and Pattern Recognition, volume 2, pages 509--516, Kauai, Hawaii, December 2001.

Magic: ghost removal

M. Uyttendaele, A. Eden, and R. Szeliski.
***Eliminating ghosting and exposure artifacts in image mosaics*.**
In Proceedings of the Interational Conference on Computer Vision and Pattern Recognition, volume 2, pages 509--516, Kauai, Hawaii, December 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
    - http://earthobservatory.nasa.gov/Newsroom/BlueMarble/
    - Click for images…

Questions?

Alternative to feathering

- **Cut** and **fuse**

Interactive Digital Photomontage

Aseem Agarwala, Mira Dontcheva
Maneesh Agrawala, Steven Drucker, Alex Colburn
Brian Curless, David Salesin, Michael Cohen