COSC579: Image Align, Mosaic, Stitch

Jeremy Bolton, PhD
Assistant Teaching Professor

Outline

GEORGETOWN UNIVERSITY

Image alignment

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

Why Mosaic?

- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°

Slide from Brown & Lowe

Why Mosaic?

- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°
  - Human FOV = 200 x 135°

Slide from Brown & Lowe

Why Mosaic?

- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°
  - Human FOV = 200 x 135°
  - Panoramic Mosaic = 360 x 180°

Slide from Brown & Lowe

Mosaics: stitching images together

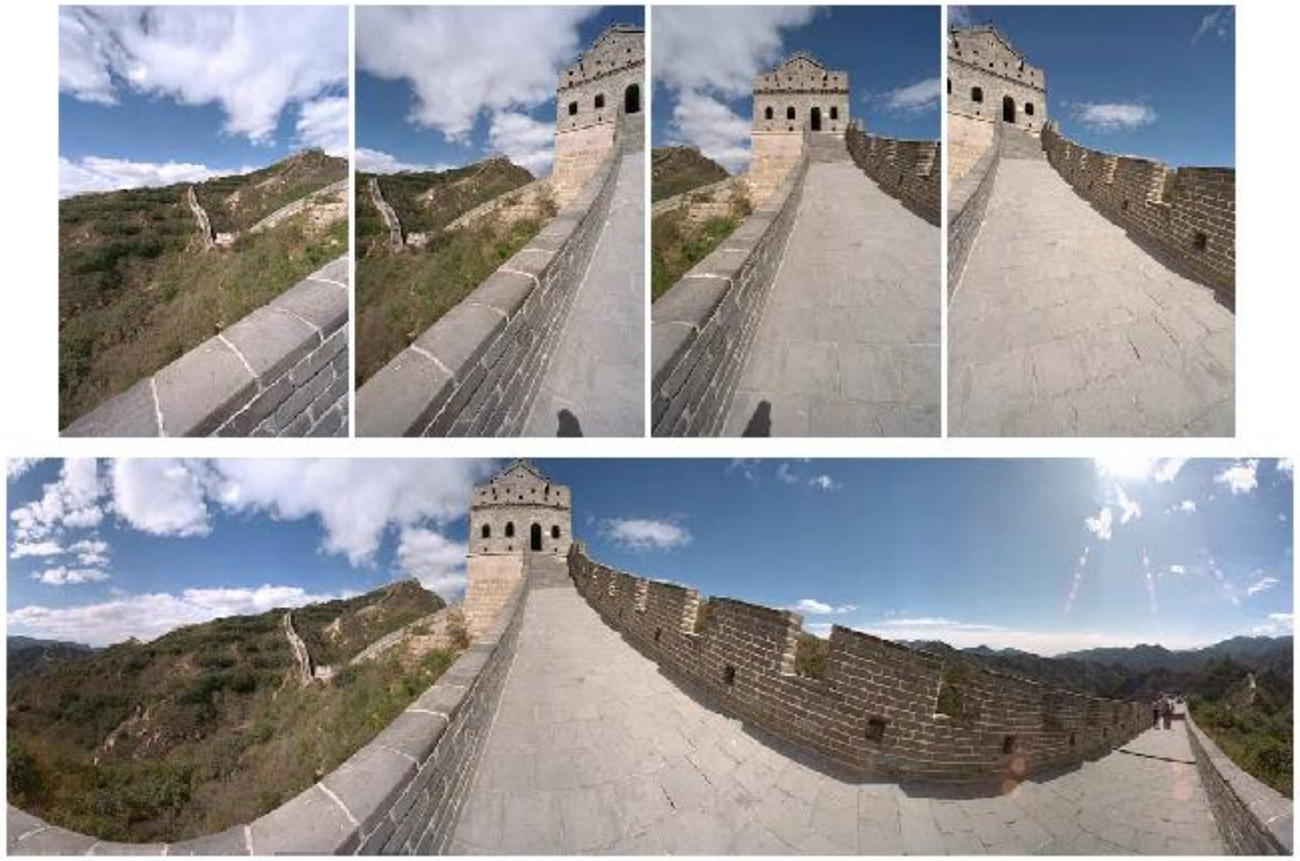


Image alignment

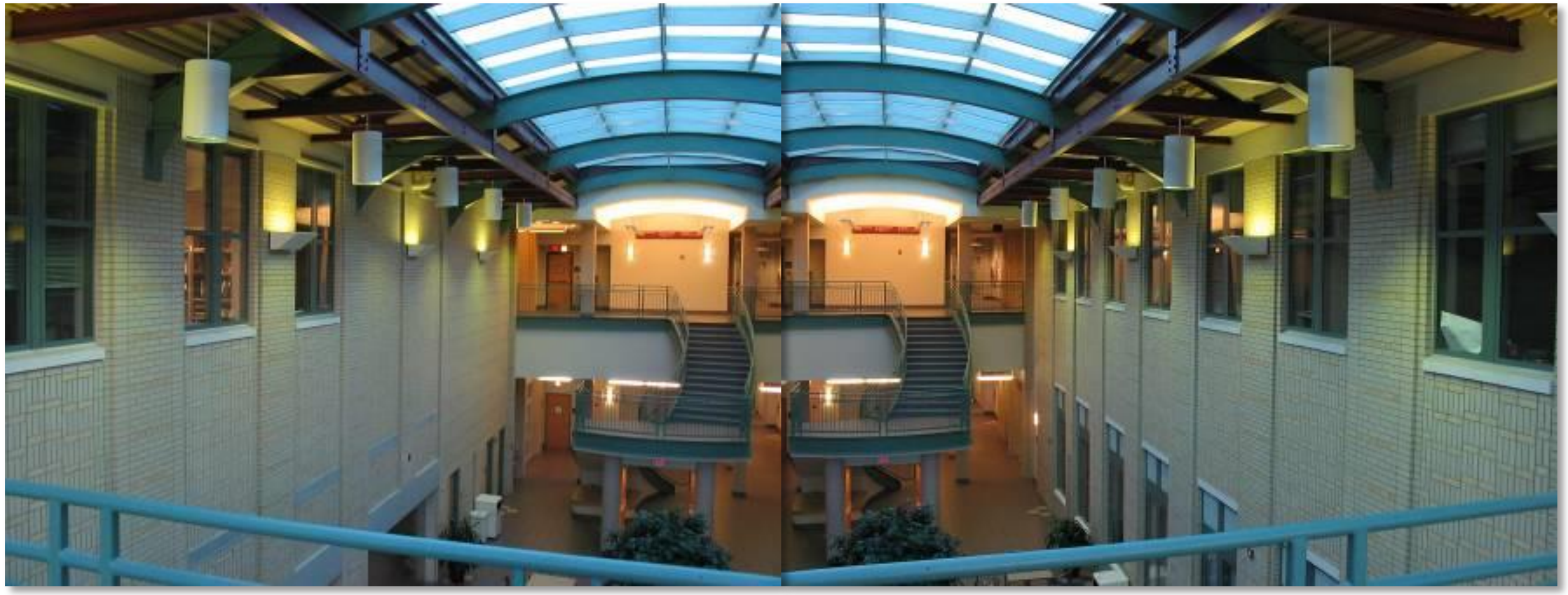
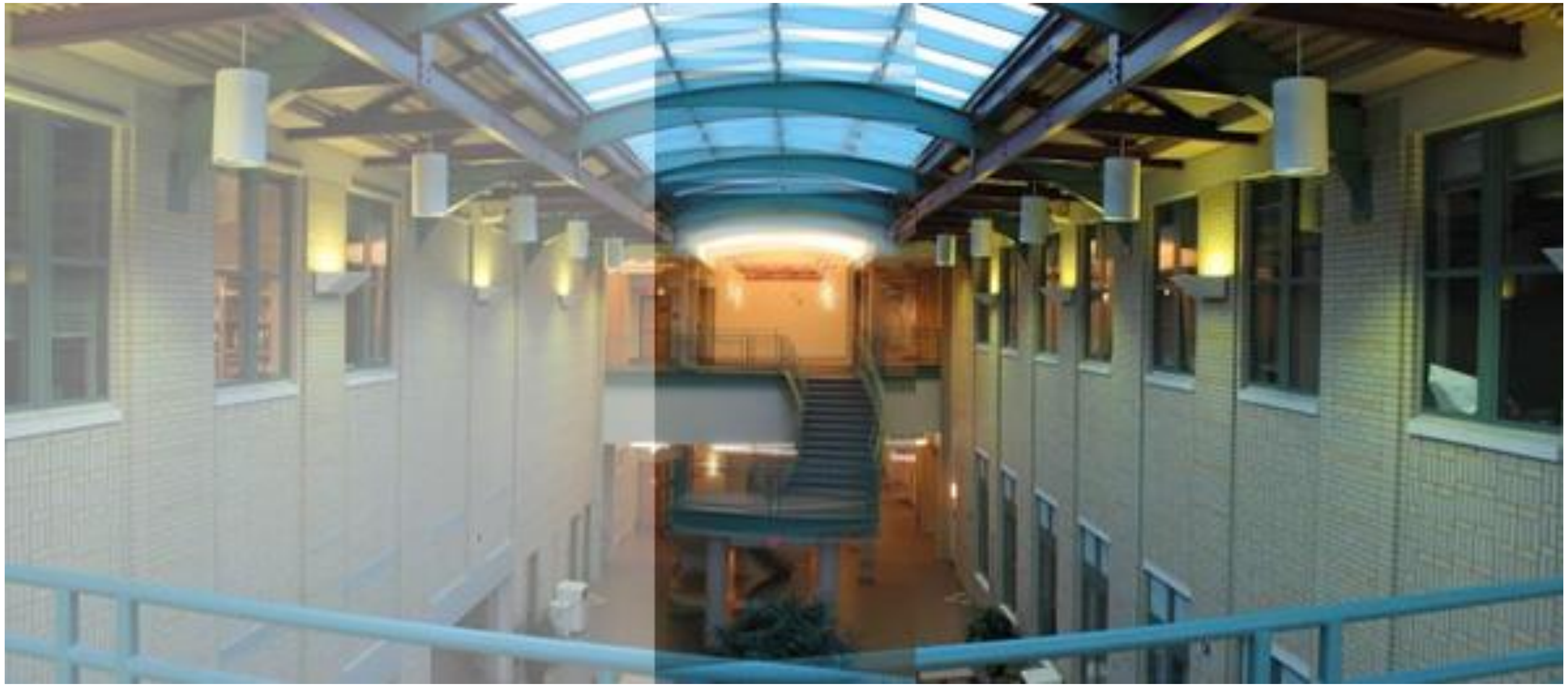


Image taken from same viewpoint, just rotated.

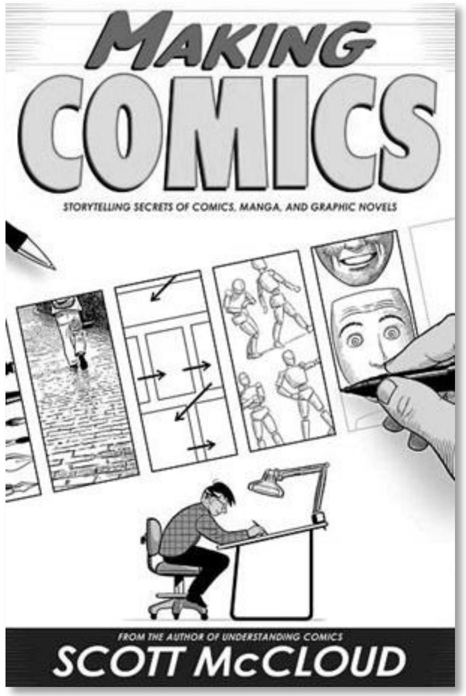
Can we line them up?

Image alignment

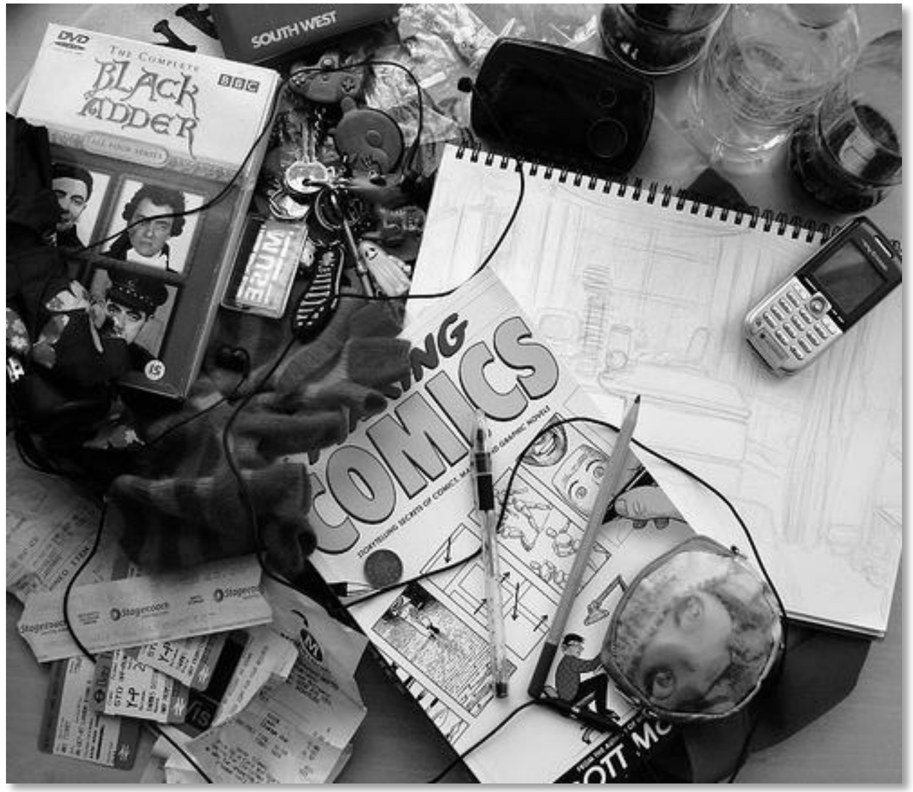


Why don't these image line up exactly?

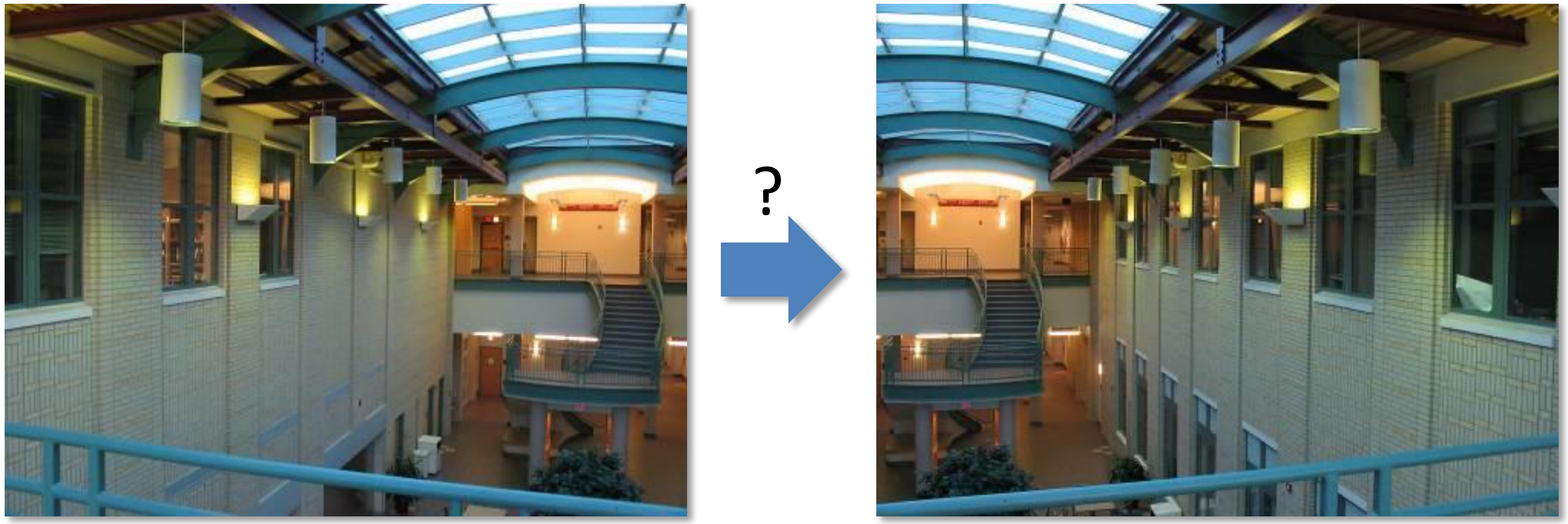
What is the geometric relationship between these two images?


MAKING COMICS
STORYTELLING SECRETS OF COMICS, MANGA, AND GRAPHIC NOVELS
FROM THE AUTHOR OF UNDERSTANDING COMICS
SCOTT McCLOUD

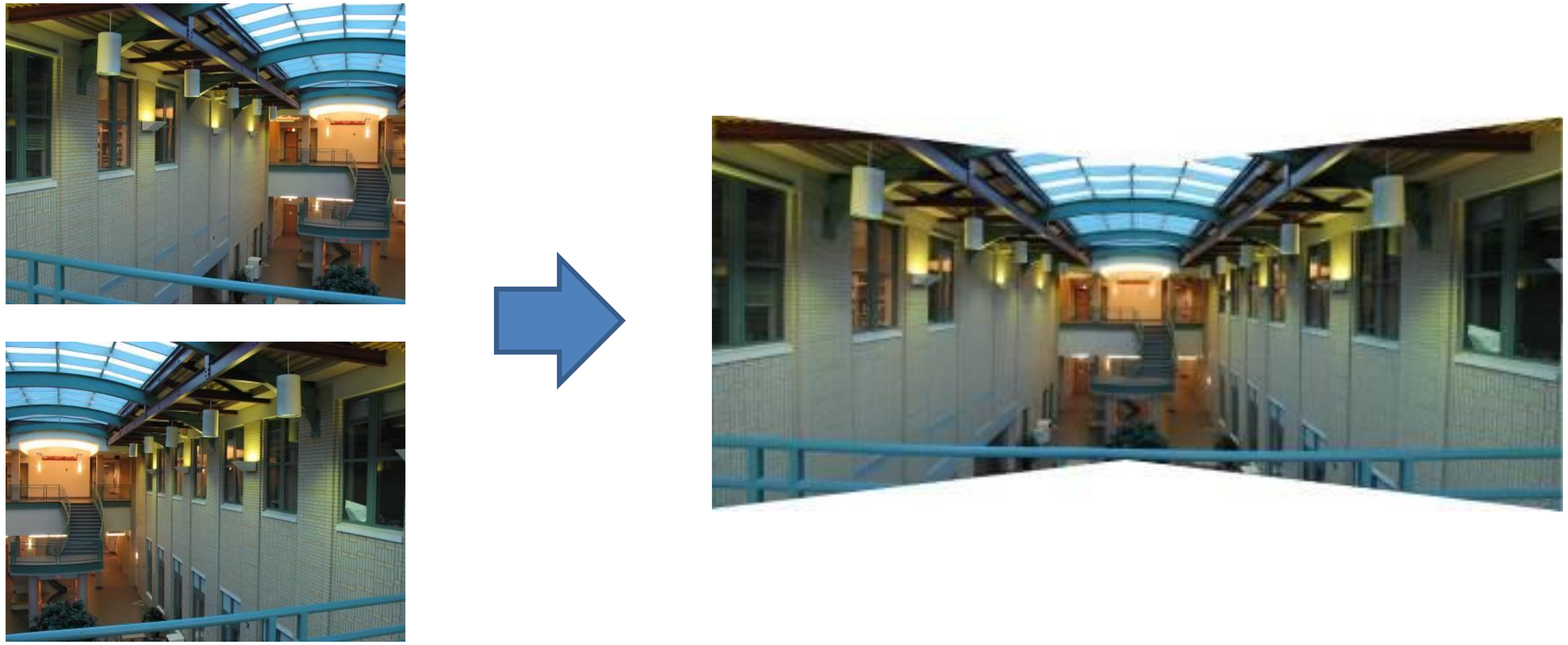
$\xrightarrow{?}$



What is the geometric relationship between these two images?



Is this an affine transformation?



GEORGETOWN UNIVERSITY

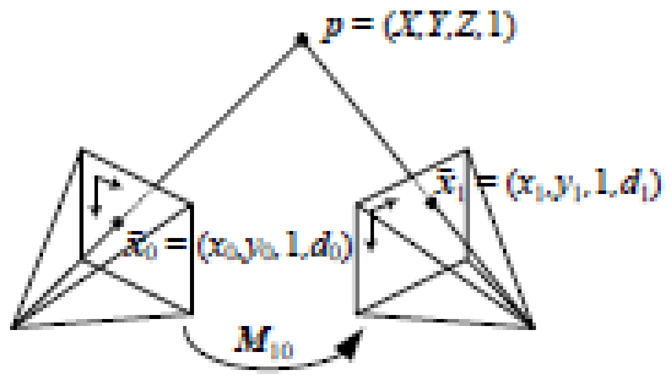
Problem: Transforming from one image to another

- Conceptually, given a camera matrix P, we can map a point to the image plane. The reverse transformation will map the image coordinate to a line (or point if we know the disparity or depth)

$$\bar{x}_0 \sim \tilde{K}_0 E_0 p = \tilde{P}_0 p.$$

- If the Camera matrix for the second image is known, we can then map to the second image

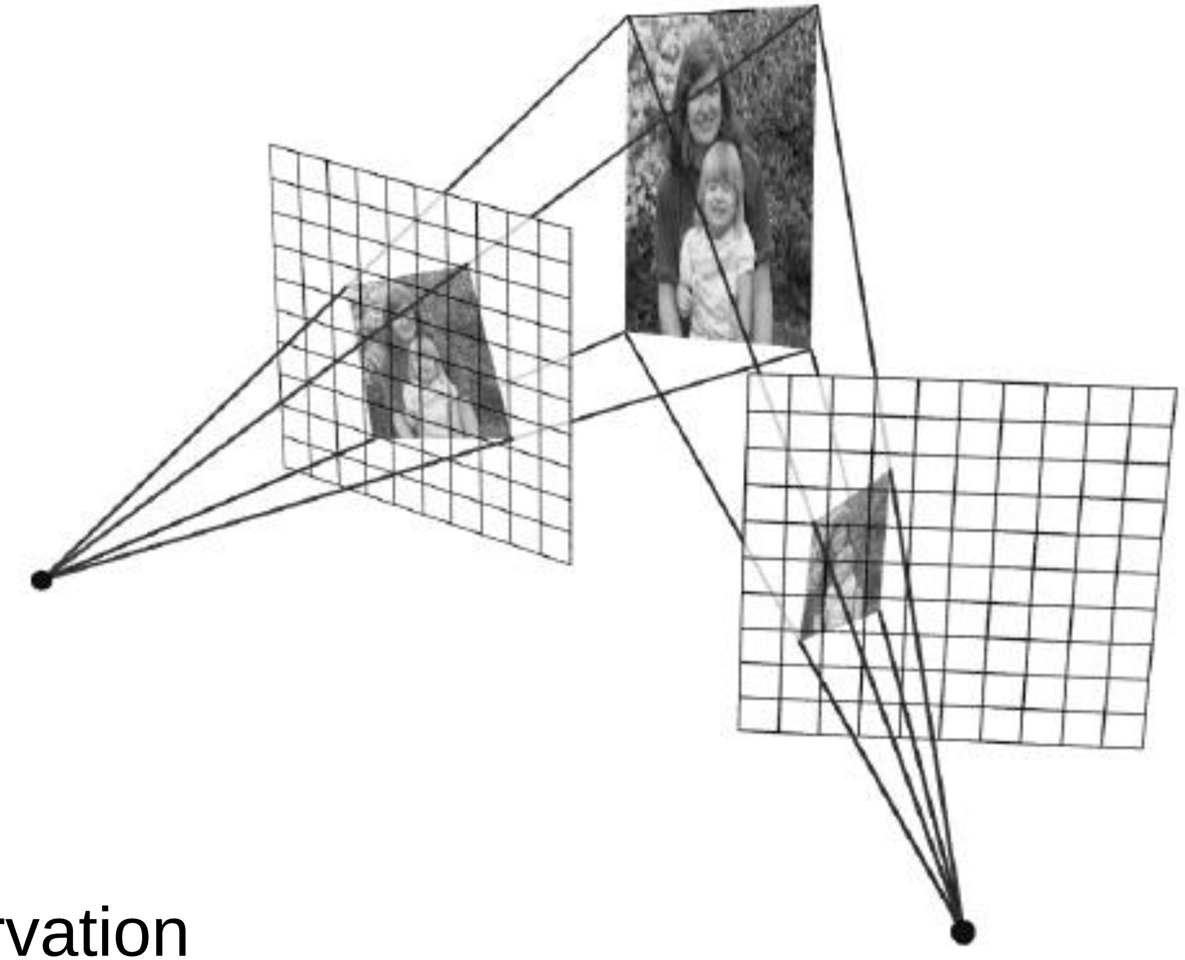
$$\bar{x}_1 \sim \tilde{K}_1 E_1 p = \tilde{K}_1 E_1 E_0^{-1} \tilde{K}_0^{-1} \bar{x}_0 = \tilde{P}_1 \tilde{P}_0^{-1} \bar{x}_0 = M_{10} \bar{x}_0.$$


$p = (X,Y,Z,1)$
$\bar{x}_1 = (x_1, y_1, 1, d_1)$
$\bar{x}_0 = (x_0, y_0, 1, d_0)$
M_{10}

(a)

GEORGETOWN UNIVERSITY

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

Transforming from one image to another Simplifying Assumption: Planar Scene

- Simplifying Assumption:
  - Choose a plane to project back onto (planar scene)

$$\tilde{x}_1 \sim \tilde{K}_1 E_1 p = \tilde{K}_1 E_1 E_0^{-1} \tilde{K}_0^{-1} \tilde{x}_0 = \tilde{P}_1 \tilde{P}_0^{-1} \tilde{x}_0 = M_{10} \tilde{x}_0.$$

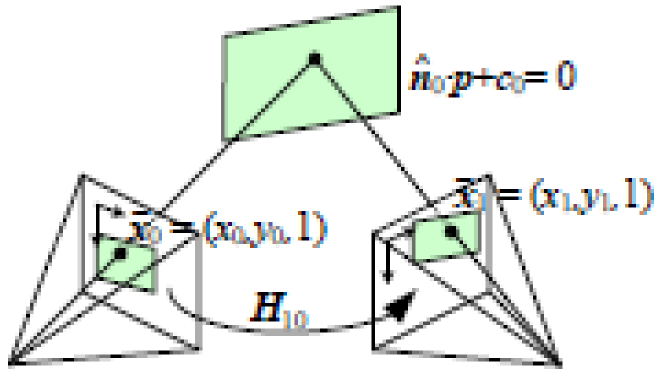
  - Analogous to replacing last row of P0 with homogeneous plane equation

$$\hat{n}_0 \cdot p + c_0$$

  - Observe the points map onto a plane, so we are not concerned about the depth (d=0)
  - Thus we can ignore the last column of M_{10}

Observe: This sequence of matrix operations can be collapsed into one 3x3 homography.

$$\tilde{x}_1 \sim \tilde{H}_{10} \tilde{x}_0,$$


$\hat{n}_0 \cdot p + c_0 = 0$
$\tilde{x}_0 = (x_0, y_0, 1)$
$\tilde{x}_1 = (x_1, y_1, 1)$
H_{10}

(b)

GEORGETOWN UNIVERSITY

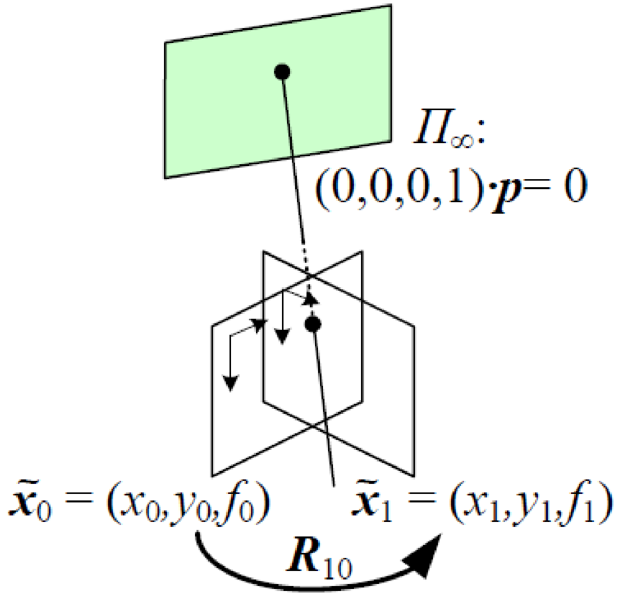
Transforming from one image to another Simplifying Assumption: Pure Rotation

- Assume no translation and only pure rotation (eg panorama)
  - Assume the scene is "large", points are at infinity.
  - Assume no translation
  - Assume simple Intrinsics Matrix

$$\tilde{x}_1 \sim \tilde{\boldsymbol{H}}_{10}\tilde{x}_0, \qquad \tilde{\boldsymbol{H}}_{10} = \boldsymbol{K}_1\boldsymbol{R}_1\boldsymbol{R}_0^{-1}\boldsymbol{K}_0^{-1} = \boldsymbol{K}_1\boldsymbol{R}_{10}\boldsymbol{K}_0^{-1}$$

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_1 & & \\ & f_1 & \\ & & 1 \end{bmatrix} \boldsymbol{R}_{10} \begin{bmatrix} f_0^{-1} & & \\ & f_0^{-1} & \\ & & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ y_1 \\ f_1 \end{bmatrix} \sim \boldsymbol{R}_{10} \begin{bmatrix} x_0 \\ y_0 \\ f_0 \end{bmatrix}$$


Π_∞:
$(0,0,0,1)\cdot\boldsymbol{p} = 0$
$\tilde{\boldsymbol{x}}_0 = (x_0, y_0, f_0)$
$\tilde{\boldsymbol{x}}_1 = (x_1, y_1, f_1)$
$\boldsymbol{R}_{10}$

This further simplification results in less parameters, since the rotation matrix entries can be calculated using 3 angles.

GEORGETOWN UNIVERSITY

Homographies

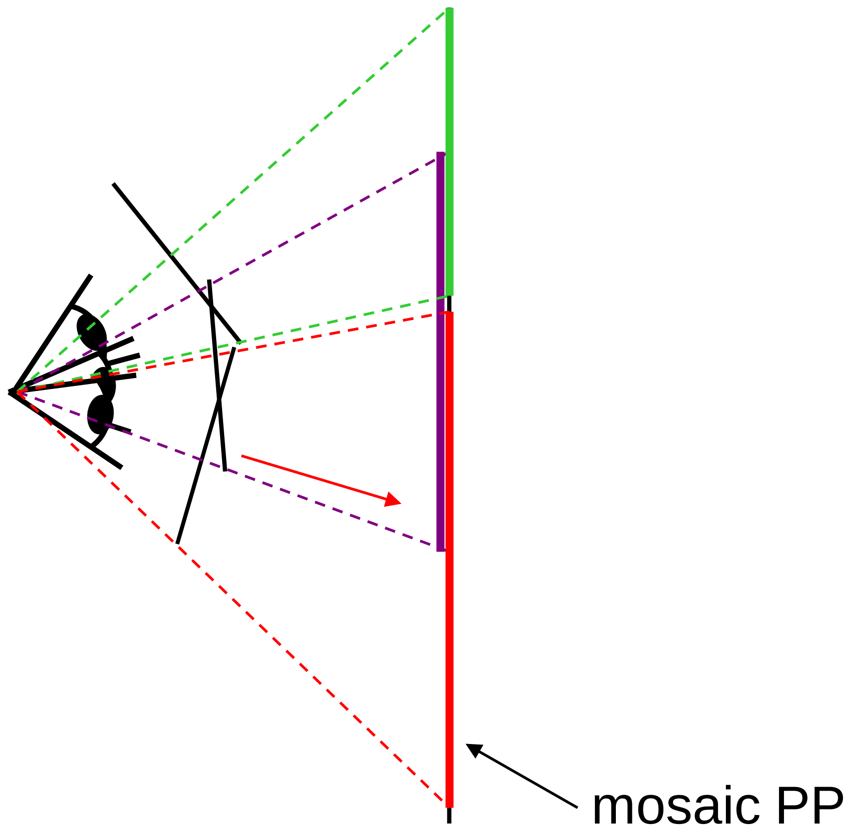
- Perspective projection of a plane
  - Lots of names for this:
  - Modeled as a 2D warp using homogeneous coordinates

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

To apply a homography **H**

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

Image reprojection for mosaicing



- The mosaic has a natural interpretation in 3D
  - The images are reprojected onto a common plane
    - idea: replace camera with slide projector, project onto new PP
  - The mosaic is formed on this plane

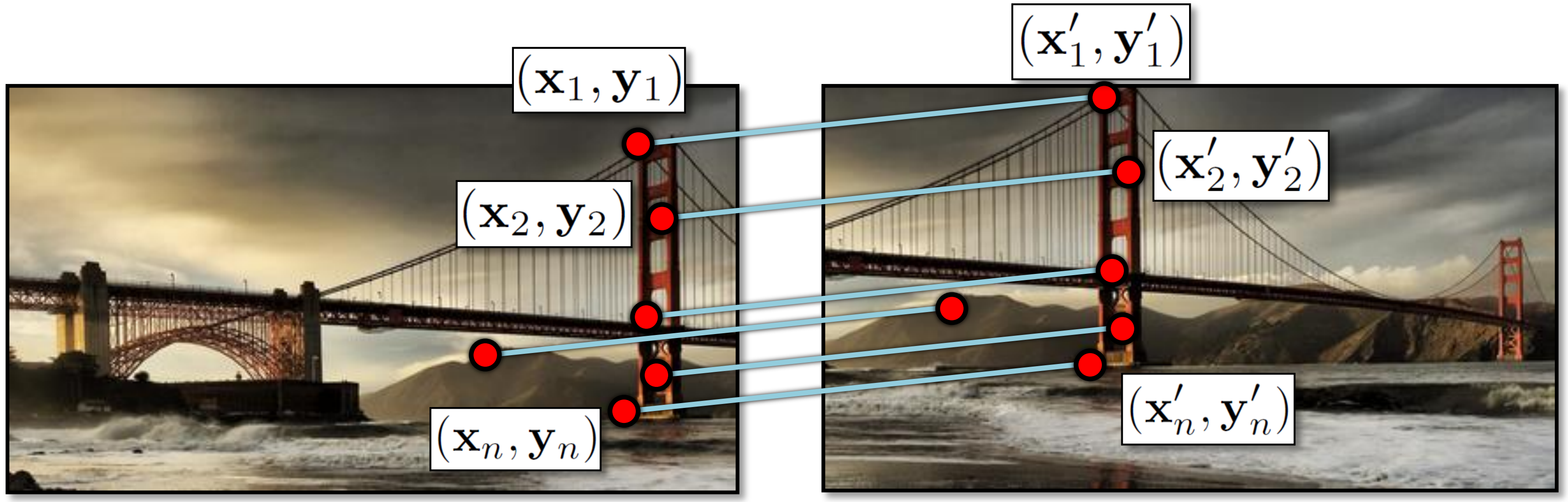
An Image Alignment Scheme (mosaic)

Given images A and B

1. Find corresponding points between images (You will manually do this for your assignment)
   1. Compute image features for A and B (Later!)
   2. Match features between A and B (Later!)
2. Solve for homography between A and B using least squares on set of matches.
3. Blend images (composite)

What could go wrong?

Estimating Translations



$(\mathbf{x}_1, \mathbf{y}_1)$
$(\mathbf{x}_2, \mathbf{y}_2)$
$(\mathbf{x}_n, \mathbf{y}_n)$
$(\mathbf{x}'_1, \mathbf{y}'_1)$
$(\mathbf{x}'_2, \mathbf{y}'_2)$
$(\mathbf{x}'_n, \mathbf{y}'_n)$

$$\begin{aligned} \mathbf{x}_i + \mathbf{x_t} &= \mathbf{x}'_i \\ \mathbf{y}_i + \mathbf{y_t} &= \mathbf{y}'_i \end{aligned}$$

- Problem: more equations than unknowns
  - "Overdetermined" system of equations
  - We will find the *least squares* solution

GEORGETOWN UNIVERSITY

Simplified Least squares formulation

- For each point $(\mathbf{x}_i, \mathbf{y}_i)$

$$\begin{aligned} \mathbf{x}_i + \mathbf{x_t} &= \mathbf{x}_i' \\ \mathbf{y}_i + \mathbf{y_t} &= \mathbf{y}_i' \end{aligned}$$

- we define the *residuals* as

$$\begin{aligned} r_{\mathbf{x}_i}(\mathbf{x}_t) &= (\mathbf{x}_i + \mathbf{x}_t) - \mathbf{x}_i' \\ r_{\mathbf{y}_i}(\mathbf{y}_t) &= (\mathbf{y}_i + \mathbf{y}_t) - \mathbf{y}_i' \end{aligned}$$

GEORGETOWN UNIVERSITY

Least squares formulation

- Goal: minimize **sum of squared residuals**

$$C(\mathbf{x}_t, \mathbf{y}_t) = \sum_{i=1}^{n} \left(r_{\mathbf{x}_i}(\mathbf{x}_t)^2 + r_{\mathbf{y}_i}(\mathbf{y}_t)^2 \right)$$

- "Least squares" solution
- For translations, is equal to mean displacement

GEORGETOWN UNIVERSITY

Simplified Least squares formulation

- Can also write as a matrix equation

$$\underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ & \vdots & \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\substack{\mathbf{A} \\ 2n \times 2}} \underbrace{\begin{bmatrix} x_t \\ y_t \end{bmatrix}}_{\substack{\mathbf{t} \\ 2 \times 1}} = \underbrace{\begin{bmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ x'_2 - x_2 \\ y'_2 - y_2 \\ \vdots \\ x'_n - x_n \\ y'_n - y_n \end{bmatrix}}_{\substack{\mathbf{b} \\ 2n \times 1}}$$

GEORGETOWN UNIVERSITY

Least squares

$$\mathbf{A}\mathbf{t} = \mathbf{b}$$

- Find **t** that minimizes

$$||\mathbf{A}\mathbf{t} - \mathbf{b}||^2$$

- To solve, form the *normal equations*

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{t} = \mathbf{A}^{\mathrm{T}}\mathbf{b}$$

$$\mathbf{t} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}$$

GEORGETOWN UNIVERSITY

Affine Assumption: Oversimplification

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- How many unknowns?
- How many equations per match?
- How many matches do we need?

Affine transformations

- Residuals:

$$\begin{aligned} r_{x_i}(a,b,c,d,e,f) &= (ax_i + by_i + c) - x'_i \\ r_{y_i}(a,b,c,d,e,f) &= (dx_i + ey_i + f) - y'_i \end{aligned}$$

- Cost function:

$$C(a,b,c,d,e,f) = \sum_{i=1}^{n} \left(r_{x_i}(a,b,c,d,e,f)^2 + r_{y_i}(a,b,c,d,e,f)^2 \right)$$

Affine transformations

- Matrix form

$$\underbrace{\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ & & & \vdots & & \\ x_n & y_n & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_n & y_n & 1 \end{bmatrix}}_{\substack{\mathbf{A} \\ 2n \times 6}} \underbrace{\begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix}}_{\substack{\mathbf{t} \\ 6 \times 1}} = \underbrace{\begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ \vdots \\ x'_n \\ y'_n \end{bmatrix}}_{\substack{\mathbf{b} \\ 2n \times 1}}$$

GEORGETOWN UNIVERSITY

Solving for homographies

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{aligned} x'_i &= \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}} \\ y'_i &= \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}} \end{aligned}$$

$$\begin{aligned} x'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\ y'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12} \end{aligned}$$

Solve for transformation parameters

$$x'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{00}x_i + h_{01}y_i + h_{02}$$
$$y'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{10}x_i + h_{11}y_i + h_{12}$$

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -y'_i x_i & -y'_i y_i & -y'_i \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Observe: here we construct the A matrix differently (such that the linear combination with the transformation parameters yields 0). This is a different way to formulate the system of equations.

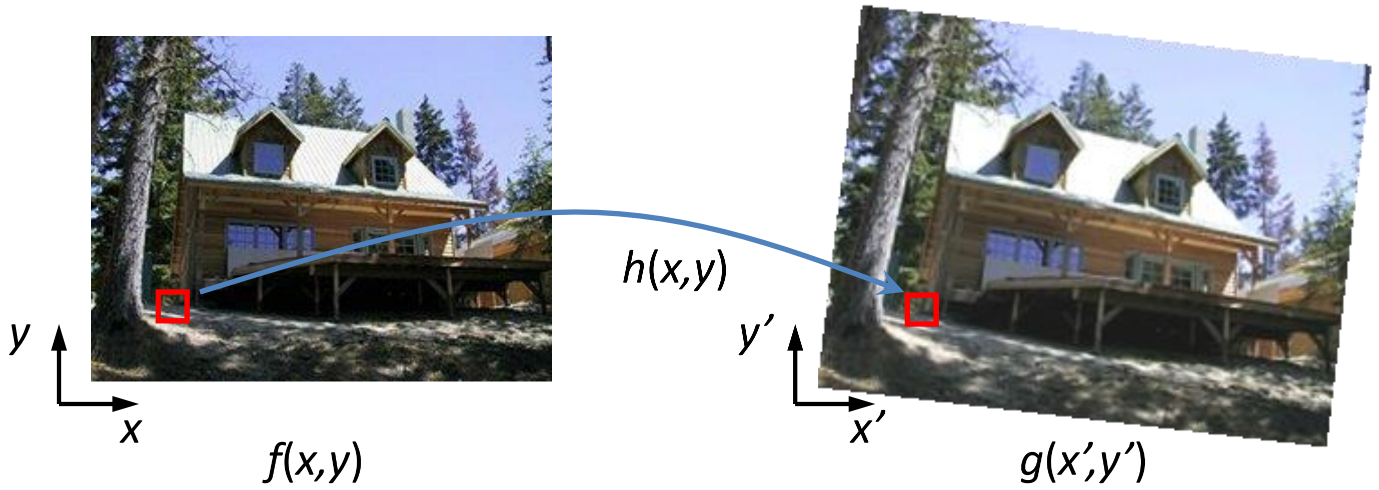
Solving for homographies

$$\underbrace{\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix}}_{\substack{\mathbf{A} \\ 2n \times 9}} \underbrace{\begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix}}_{\substack{\mathbf{h} \\ 9}} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}}_{\substack{\mathbf{0} \\ 2n}}$$

Defines a least squares problem: minimize $\|\mathbf{A}\mathbf{h} - \mathbf{0}\|^2$

- Since $\mathbf{h}$ is only defined up to scale, solve for unit vector $\hat{\mathbf{h}}$
- Solution: $\hat{\mathbf{h}}$ = eigenvector of $\mathbf{A}^T\mathbf{A}$ with smallest eigenvalue
- Works with 4 or more points

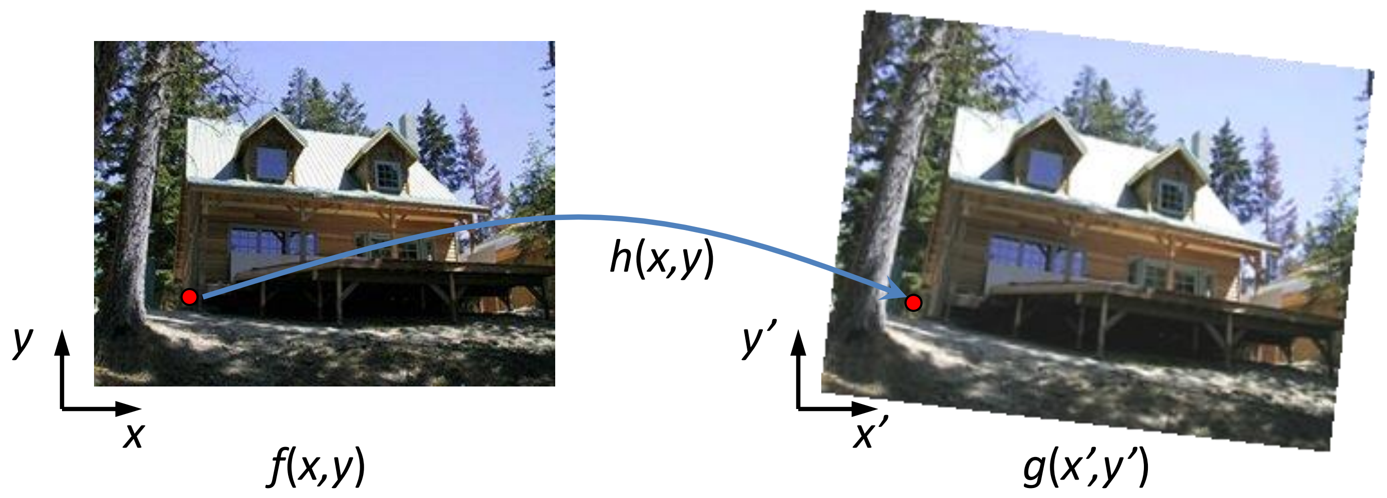
We have now learned how to solve for warping transformation … Done?



- 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*))?

GEORGETOWN UNIVERSITY

Forward warping

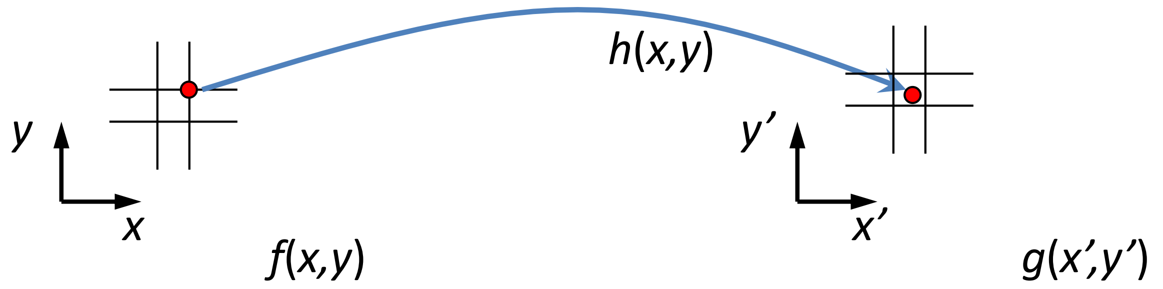


- Send each pixel *f*(*x*,*y*) to its corresponding location
- $(x',y') = h(x,y)$ in the second image

Q: what if pixel lands “between” two pixels?

GEORGETOWN UNIVERSITY

Forward warping



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

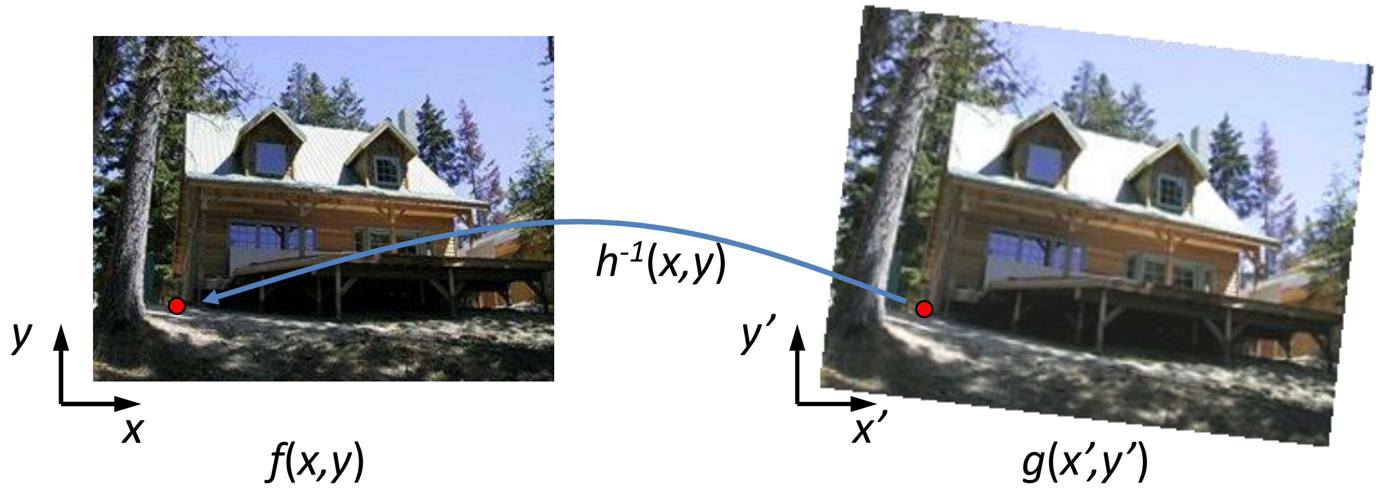
Q: what if pixel lands "between" two pixels?

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

- – Known as "splatting"

GEORGETOWN UNIVERSITY

Inverse warping

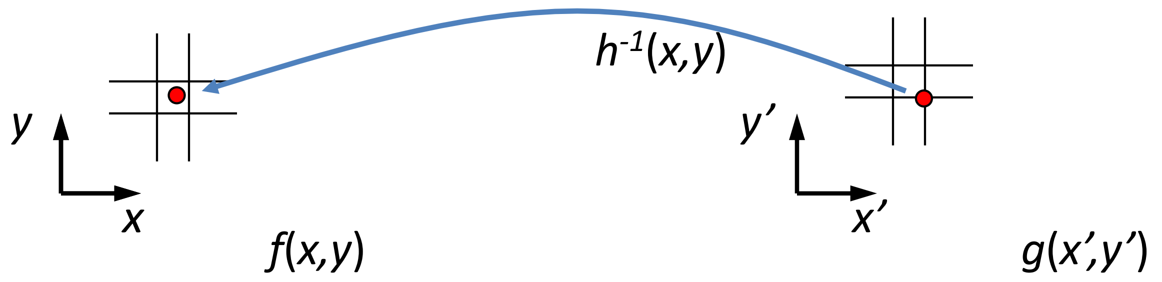


- Get each pixel $g(x',y')$ from its corresponding location
- $(x,y) = h^{-1}(x',y')$ in the first image

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

GEORGETOWN UNIVERSITY

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

- Resampling techniques before
  - nearest neighbor, interpolation, …

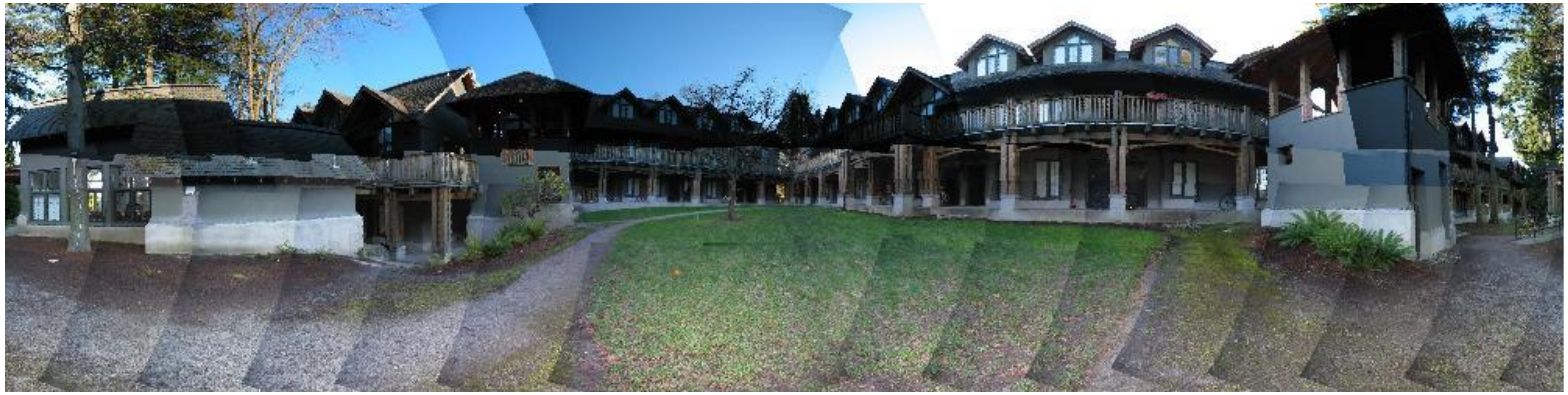
Forward vs. inverse warping

- Q: which is better?

- A: usually inverse—eliminates holes
  - however, it requires an invertible warp function

Blending

- We’ve aligned the images – now what?



Blending

- Want to seamlessly blend them together

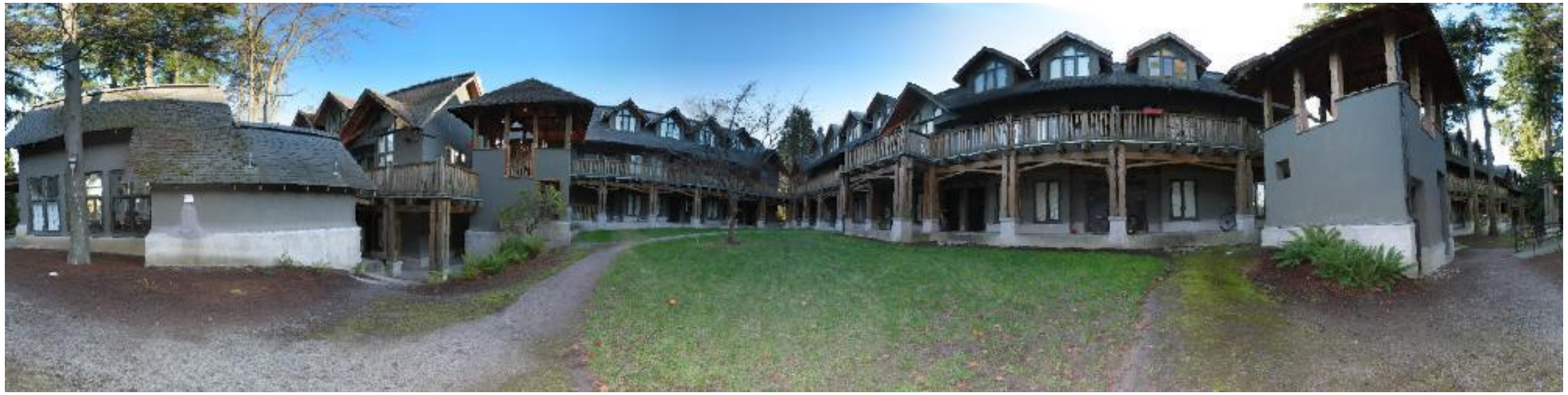
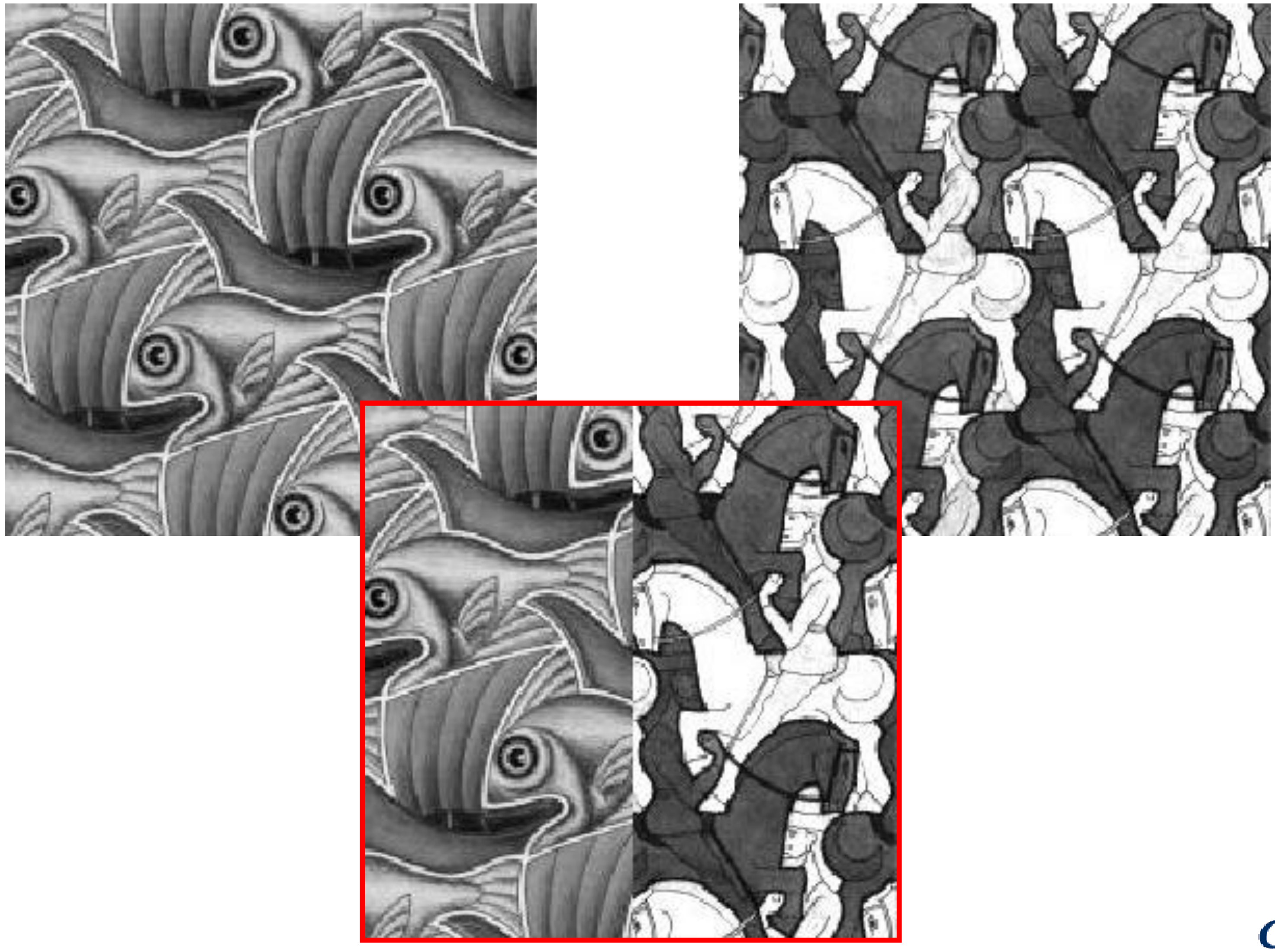
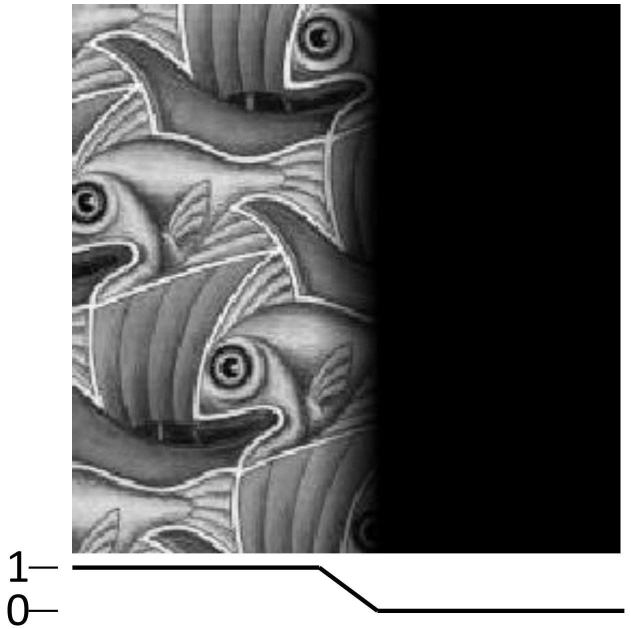


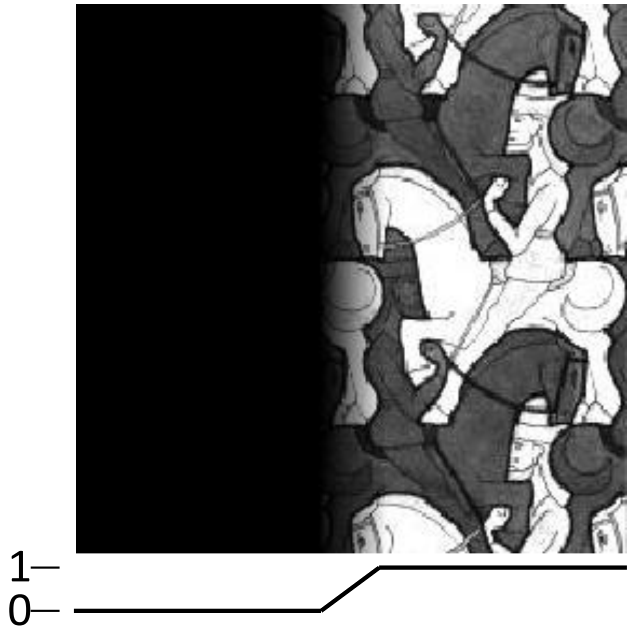
Image Blending



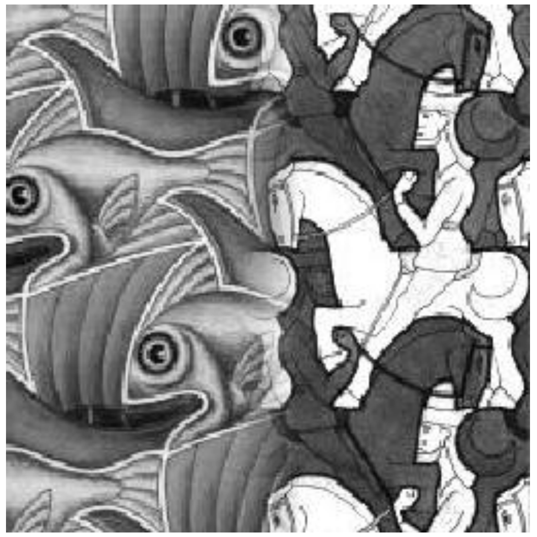
Feathering



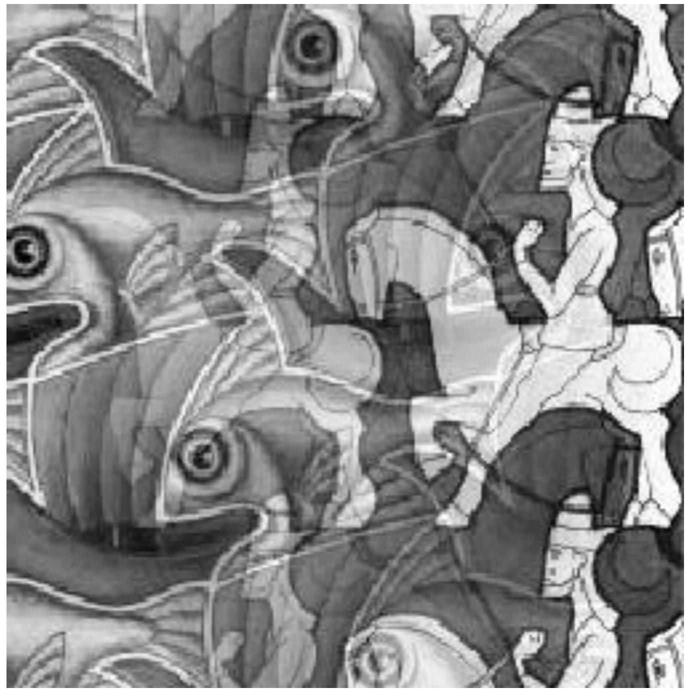
+

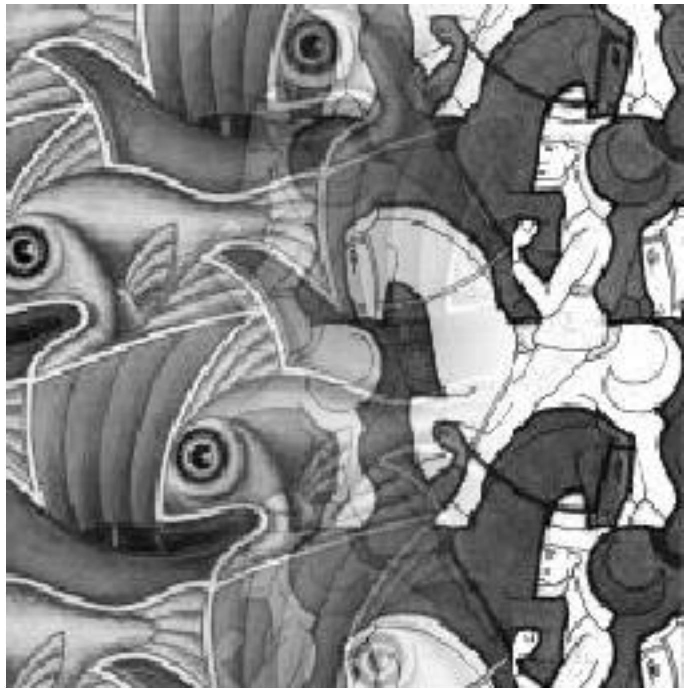


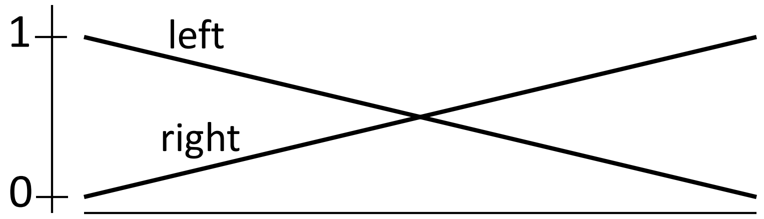
=

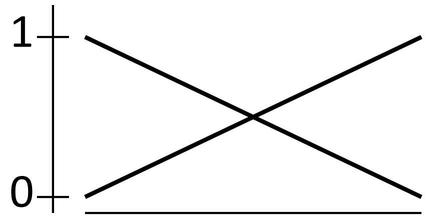


Effect of window size

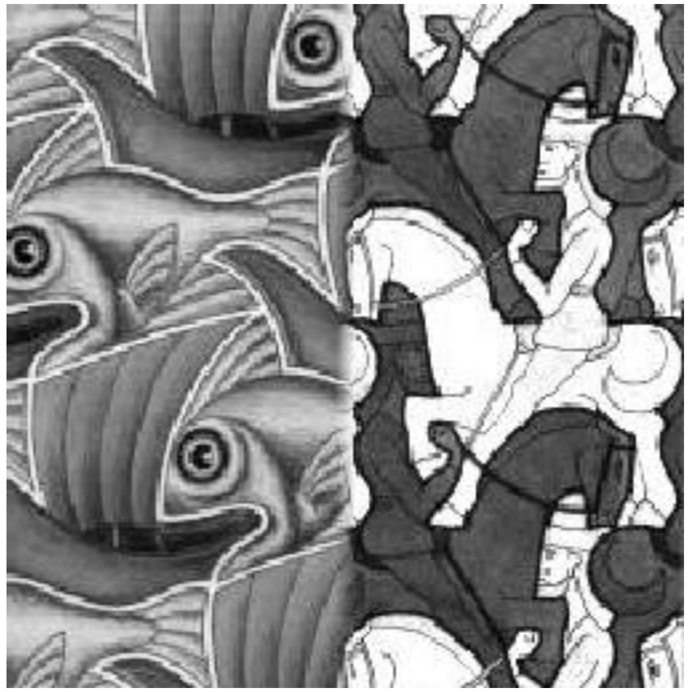


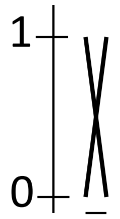


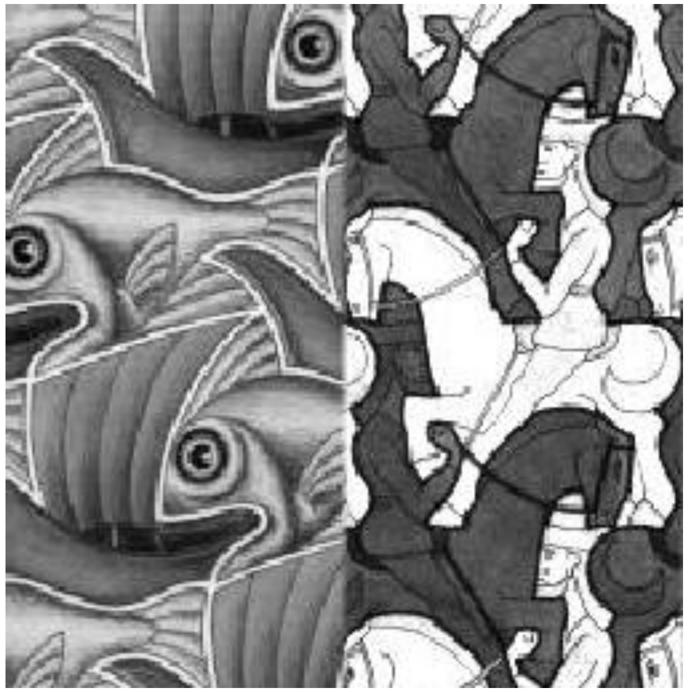


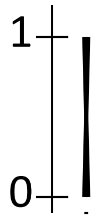


Effect of window size

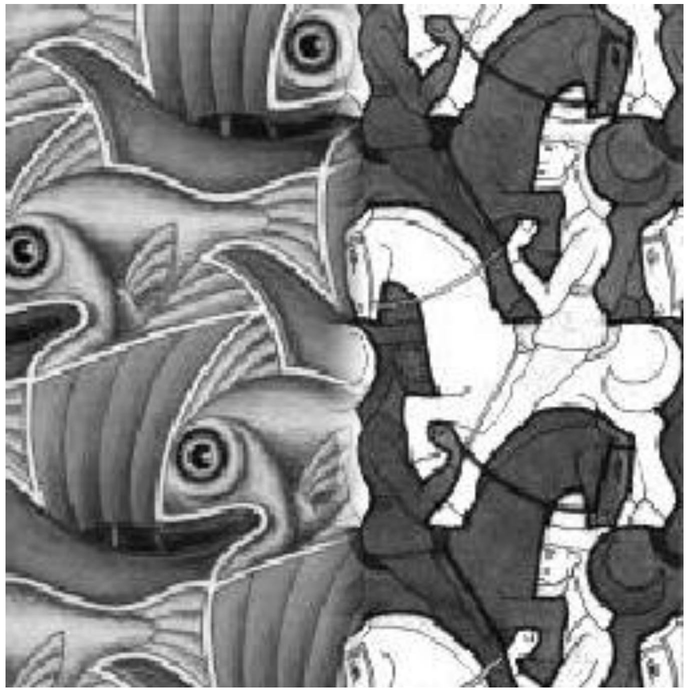


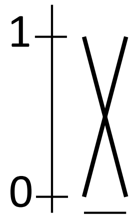






Good window size



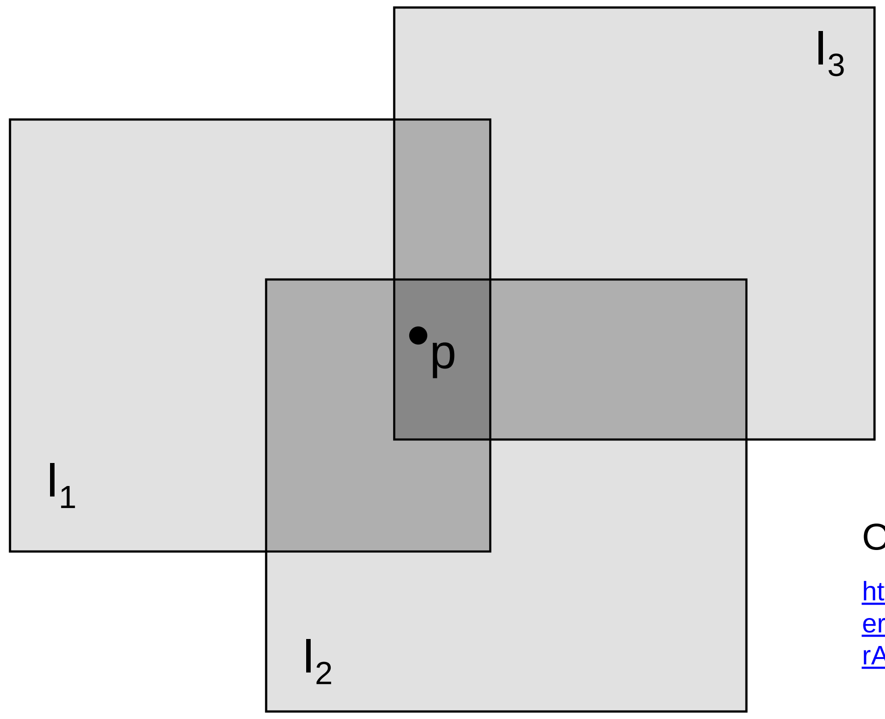


"Optimal" window: smooth but not ghosted

- Doesn't always work...

GEORGETOWN UNIVERSITY

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

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

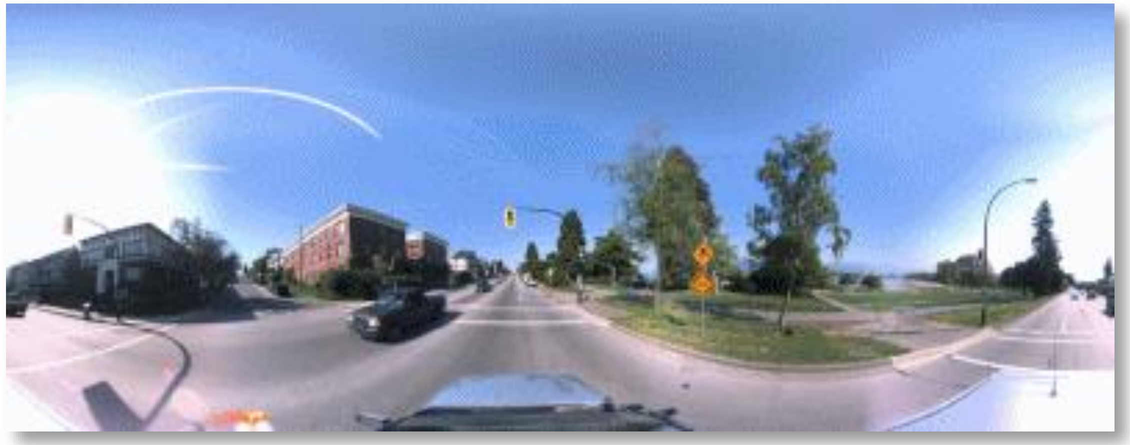
   Q: what if $\alpha = 0$?

GEORGETOWN UNIVERSITY

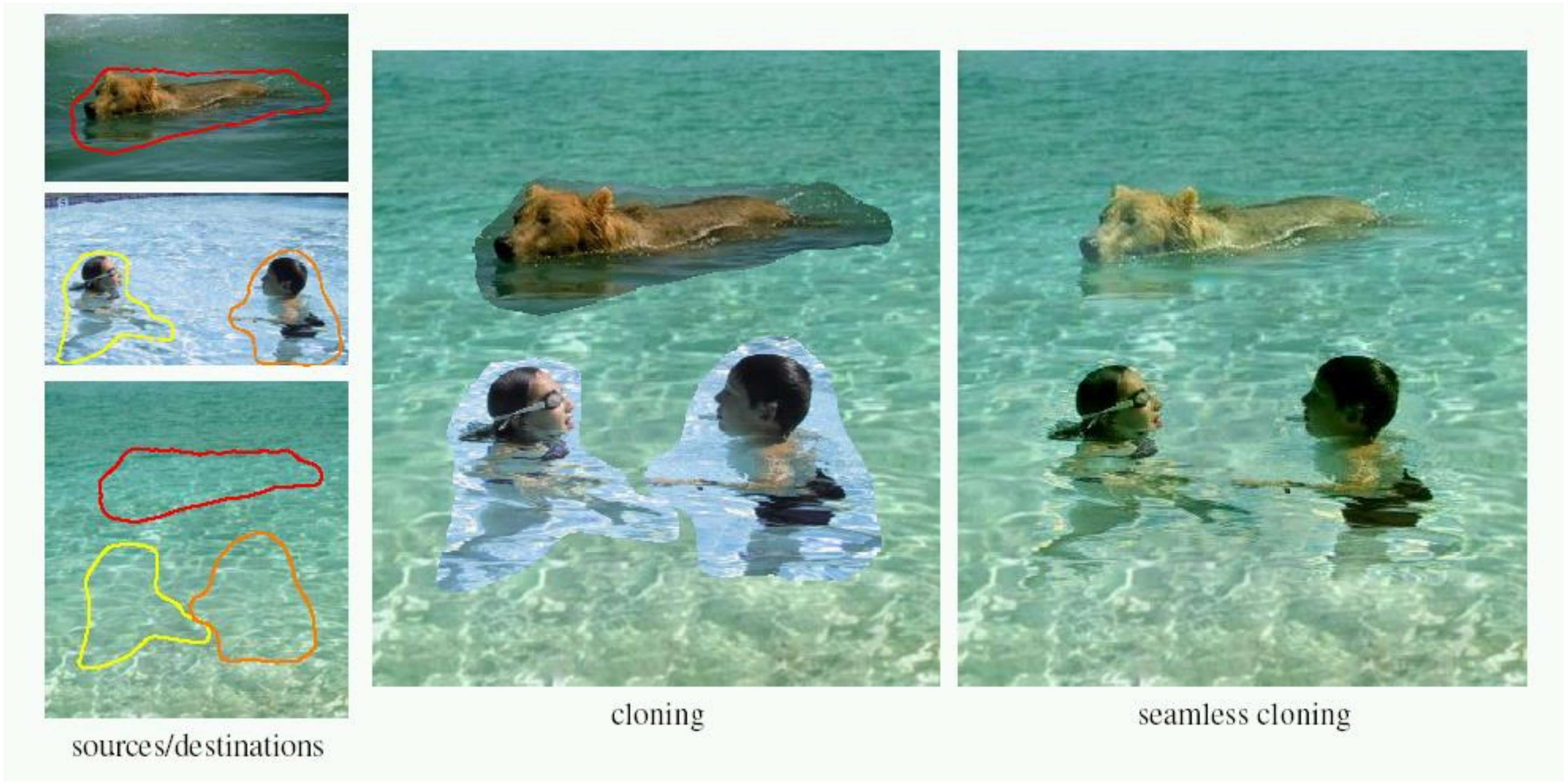
Other Fun Applications of Compositing and Blending

Some panorama examples

- Every image on Google Streetview



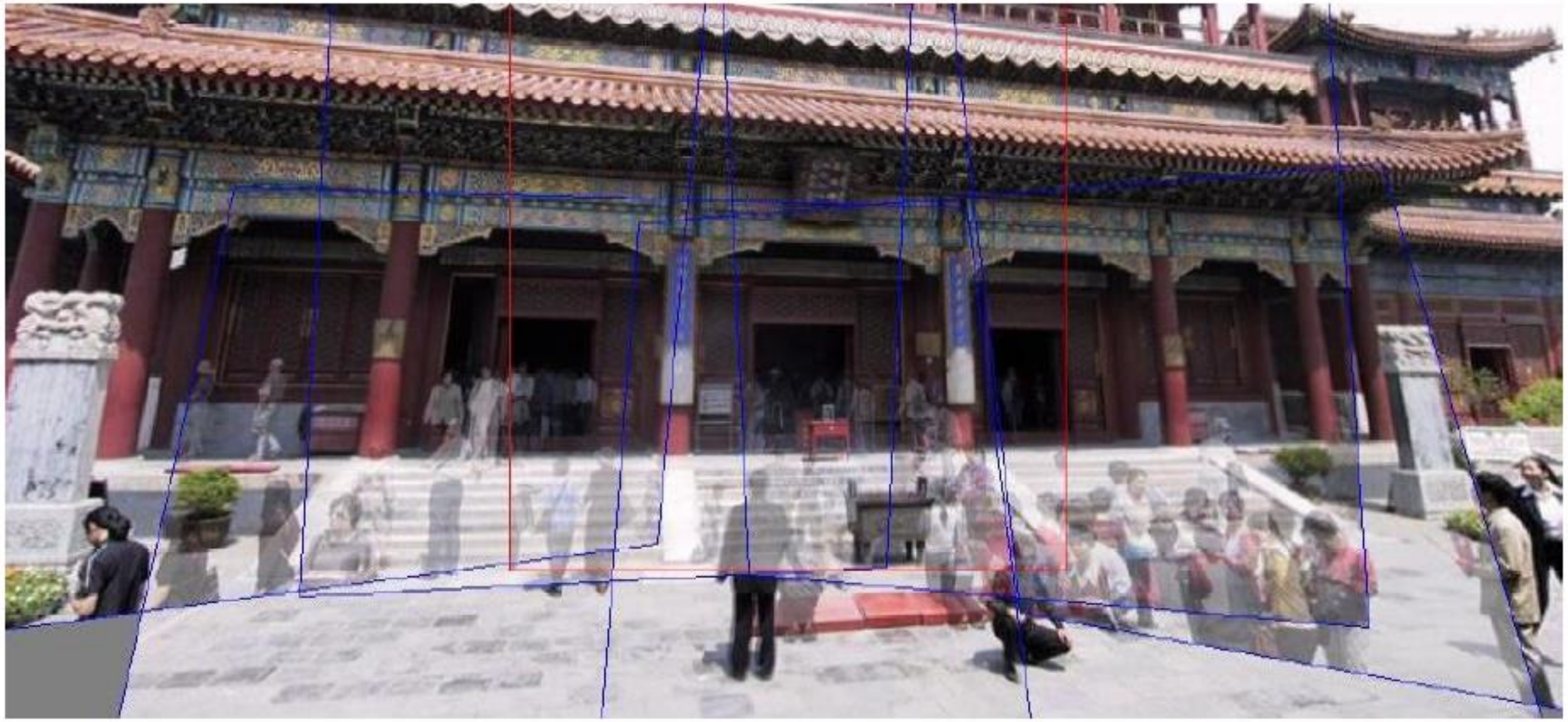
Poisson Image Editing



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

GEORGETOWN UNIVERSITY

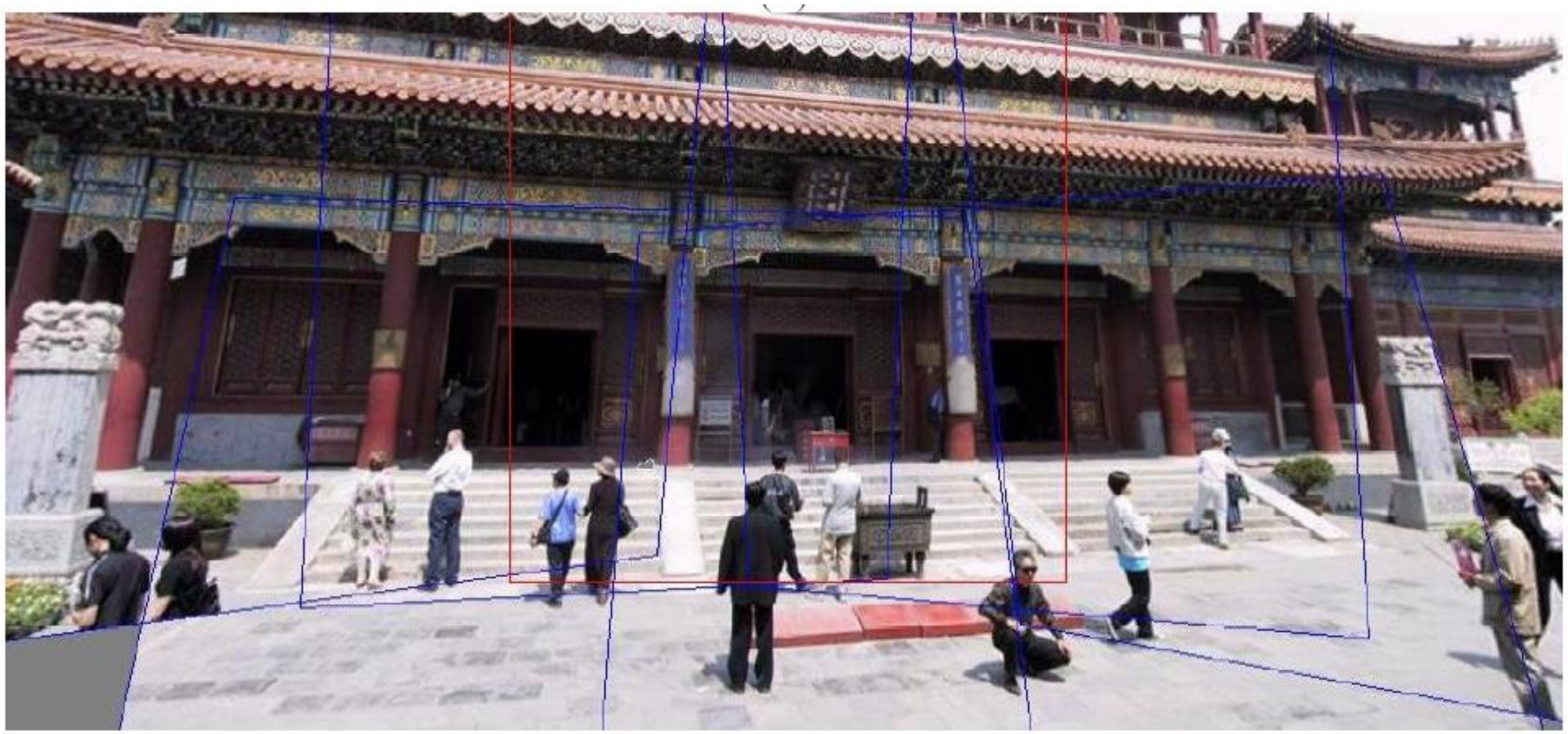
Magic: ghost removal



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.

GEORGETOWN UNIVERSITY

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.

GEORGETOWN UNIVERSITY