CPSC 4040/6040 Computer Graphics Images

Joshua Levine
levinej@clemson.edu

Lecture 21
Panoramas

Nov. 10, 2015

Slide Credits:
Alexei Efros
Frédo Durand

Agenda

- Q04 Due
- PA06 Questions?
- PA07 posted tonight (hopefully)

Continuing from Lec21

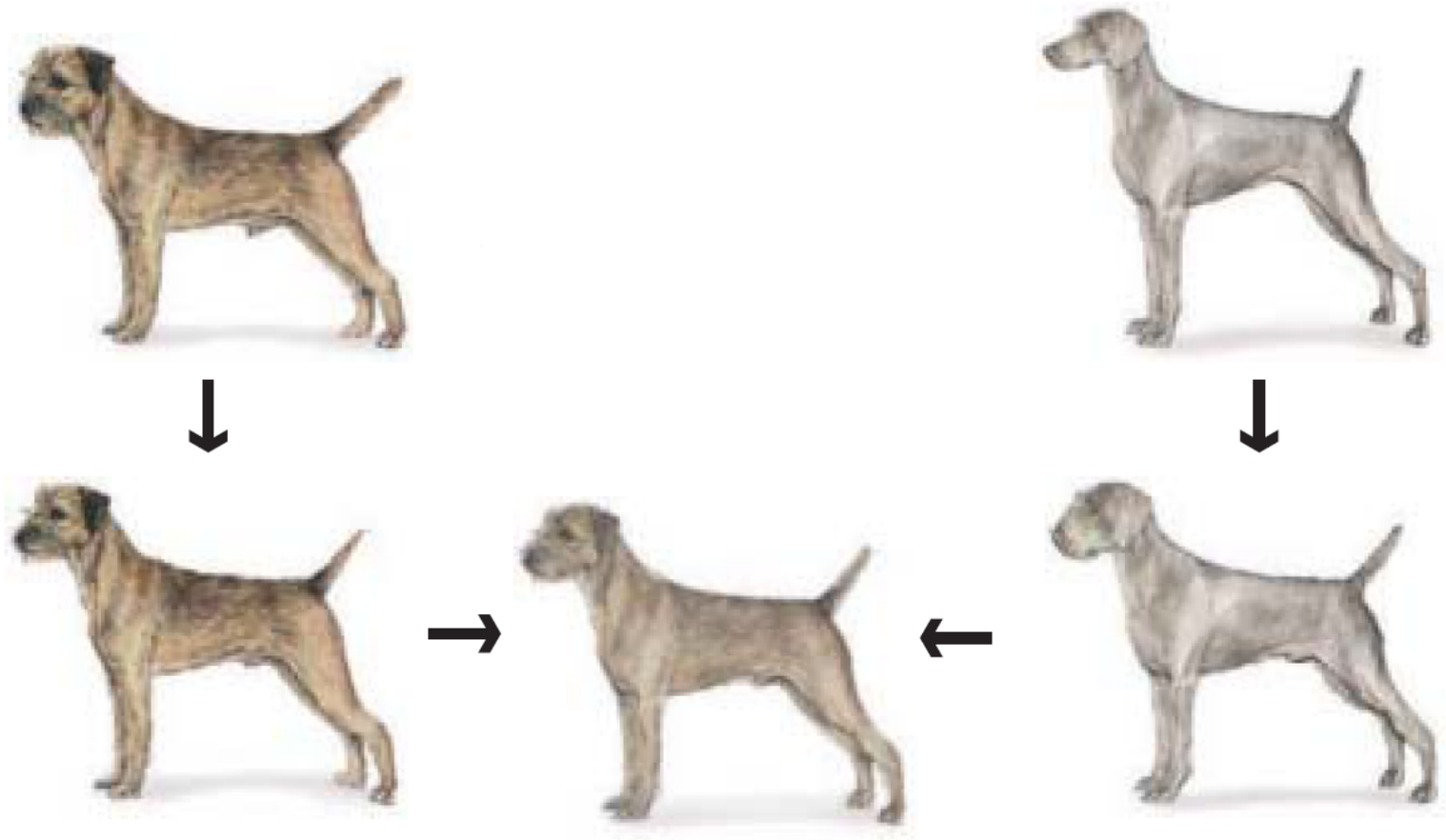
Morphing: Object Averaging

Image Average

- The aim is to find “an average” between two objects
  - Not an average of two images of objects…
  - …but an image of the average object!
- How do we know what the average object looks like?
  - We haven’t a clue!
  - But we can often fake something reasonable, usually with required user/artist input

Object Average

Idea: Warp First, Then Cross-Dissolve

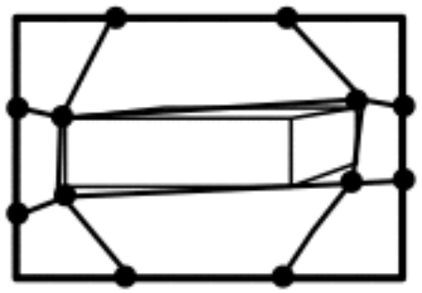


Morphing procedure:

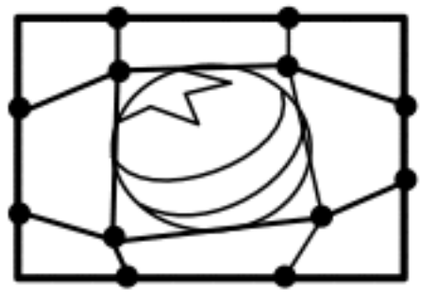
For every intermediate step t,

1. Find the average shape (the “mean dog”)
    - local warping
2. Find the average color
    - Cross-dissolve the warped images

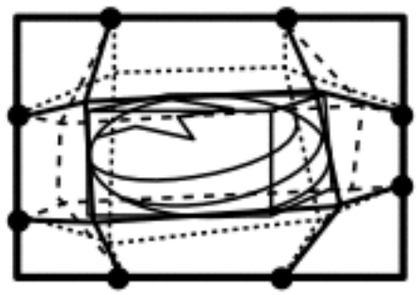
Mesh-Based Warping



starting mesh



ending mesh



intermediate mesh for frame in middle of sequence

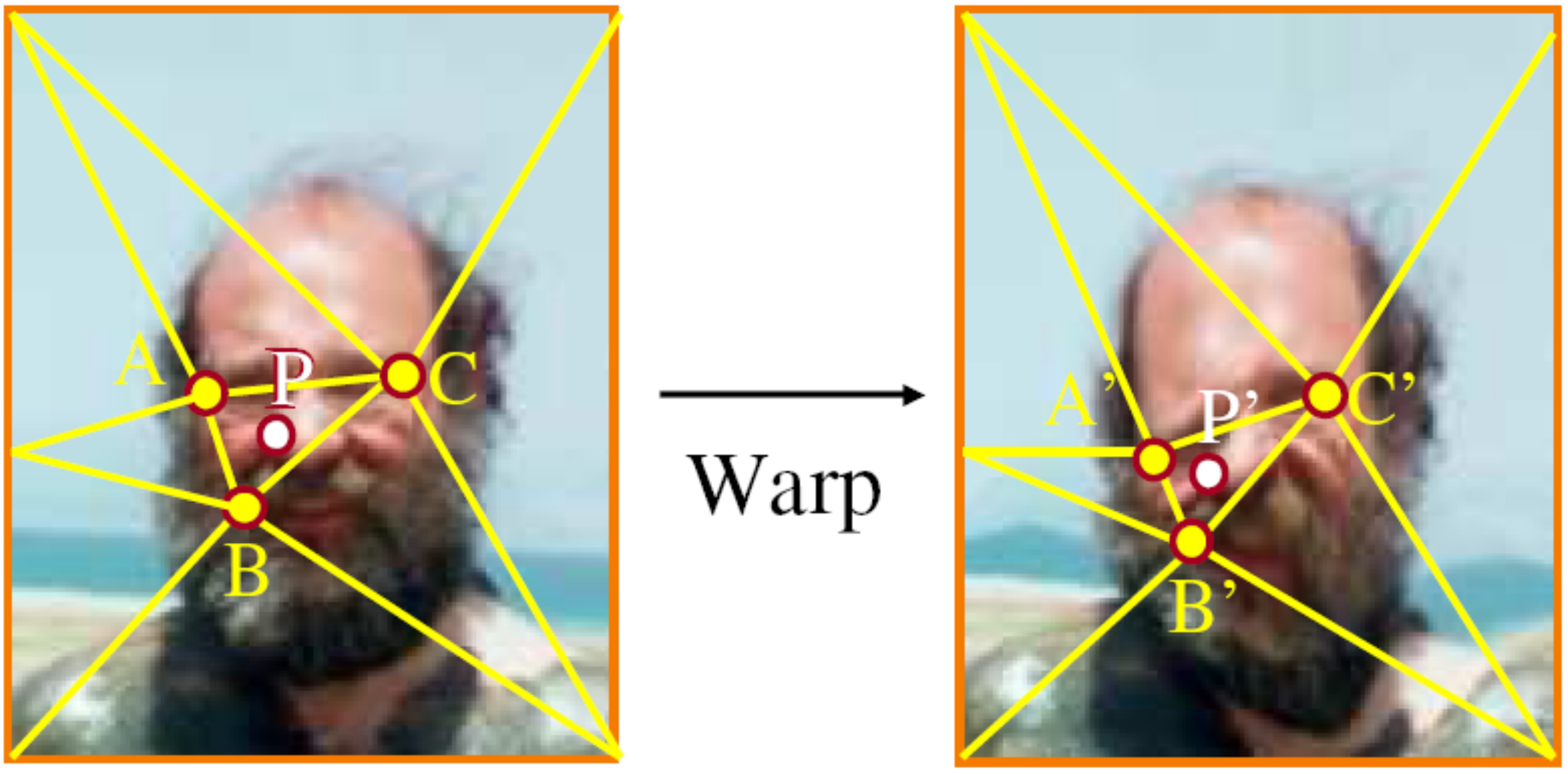
- How big of a grid is necessary?
- Really, we'd rather specify just a few points, not a grid

Example: Triangle Warping

$$P = w_A A + w_B B + w_C C$$

$$P' = w_A A' + w_B B' + w_C C'$$

Barycentric coordinate

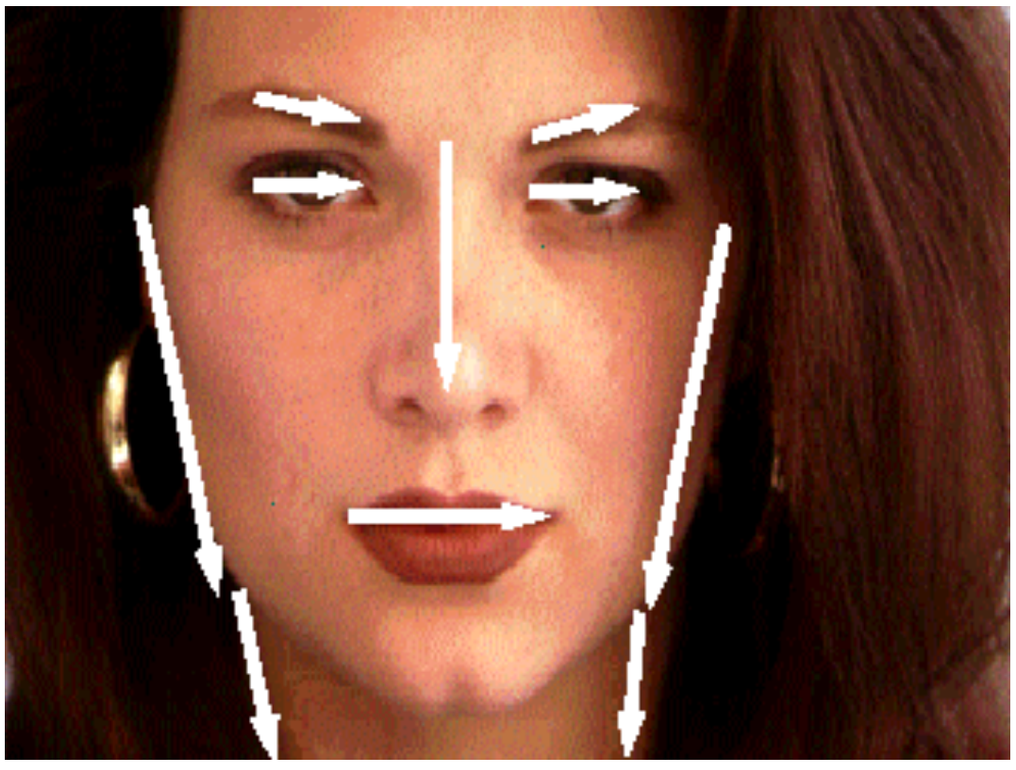


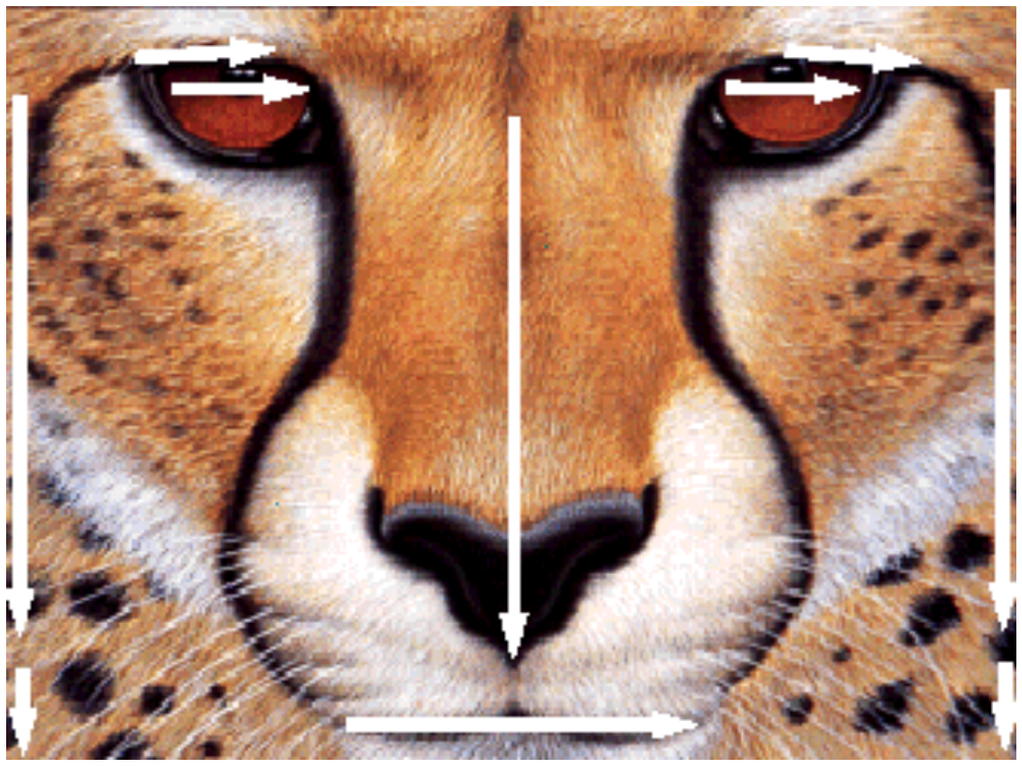
Michael Jackson's Black or White

http://youtu.be/F2AitTPI5U0?t=5m15s

The Beier & Neely Algorithm

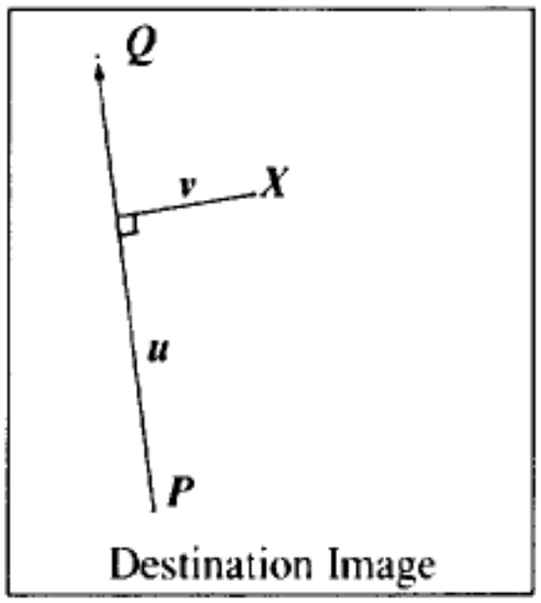
- Specify the warp by specifying corresponding vectors
- Interpolate to a complete warping function



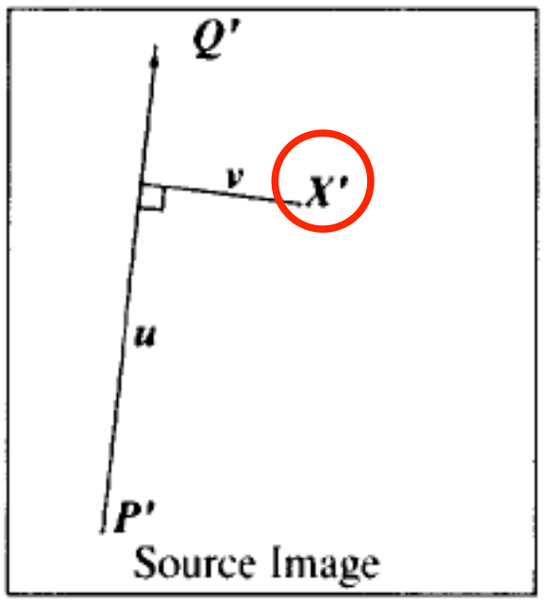


How Line Segments Specify Points

- Given PQ and P'Q', they define a warp for a point X to a point X'
- Measure distance u along the segment
- The direction of PQ defines a side, measure distance v from the right side



Destination Image



Source Image

$$u = \frac{(X - P) \cdot (Q - P)}{\|Q - P\|^2} \quad (1)$$

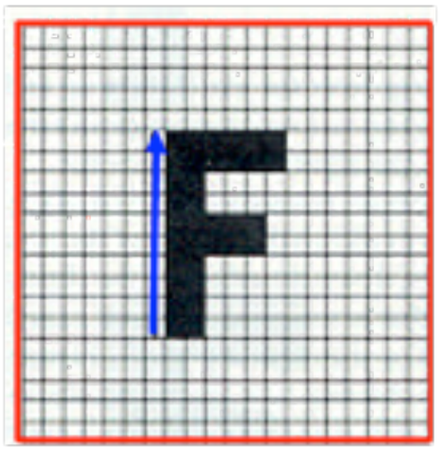
$$v = \frac{(X - P) \cdot Perpendicular(Q - P)}{\|Q - P\|} \quad (2)$$

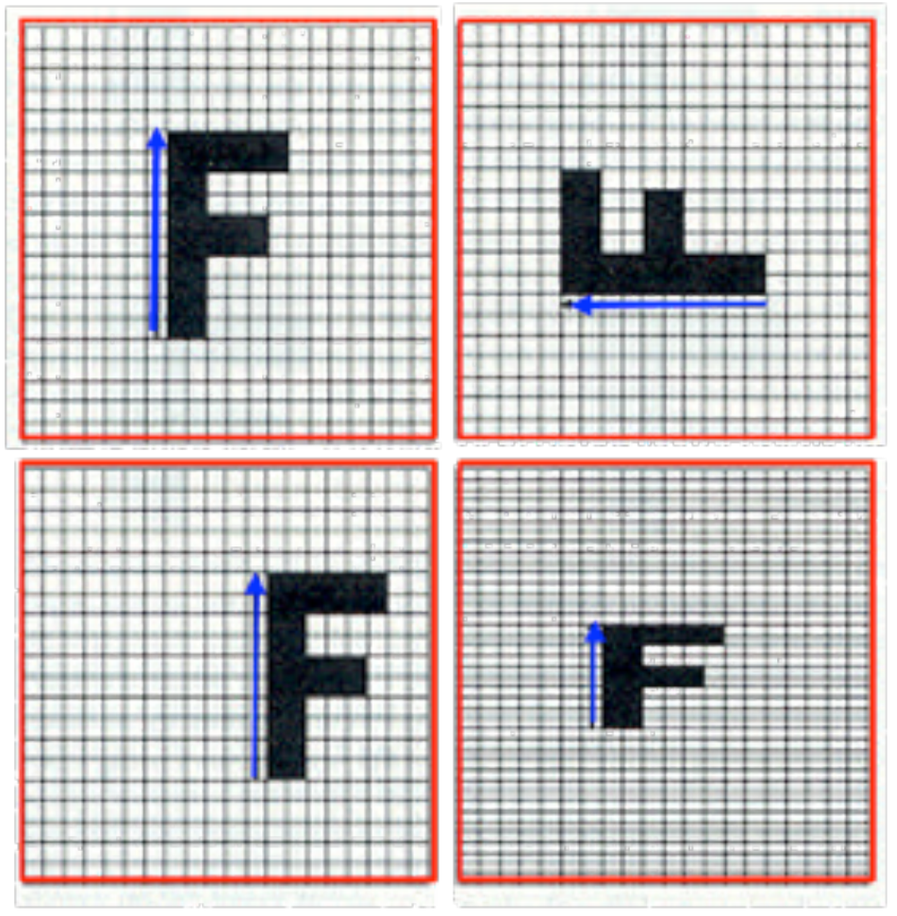
$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot Perpendicular(Q' - P')}{\|Q' - P'\|} \quad (3)$$

One Line Segment (Globally) Warping an Image

- For each X in the destination image:
  1. Find the corresponding u,v
  2. Find X’ in the source image for that u,v
  3. OUT(X) = IN(X’)

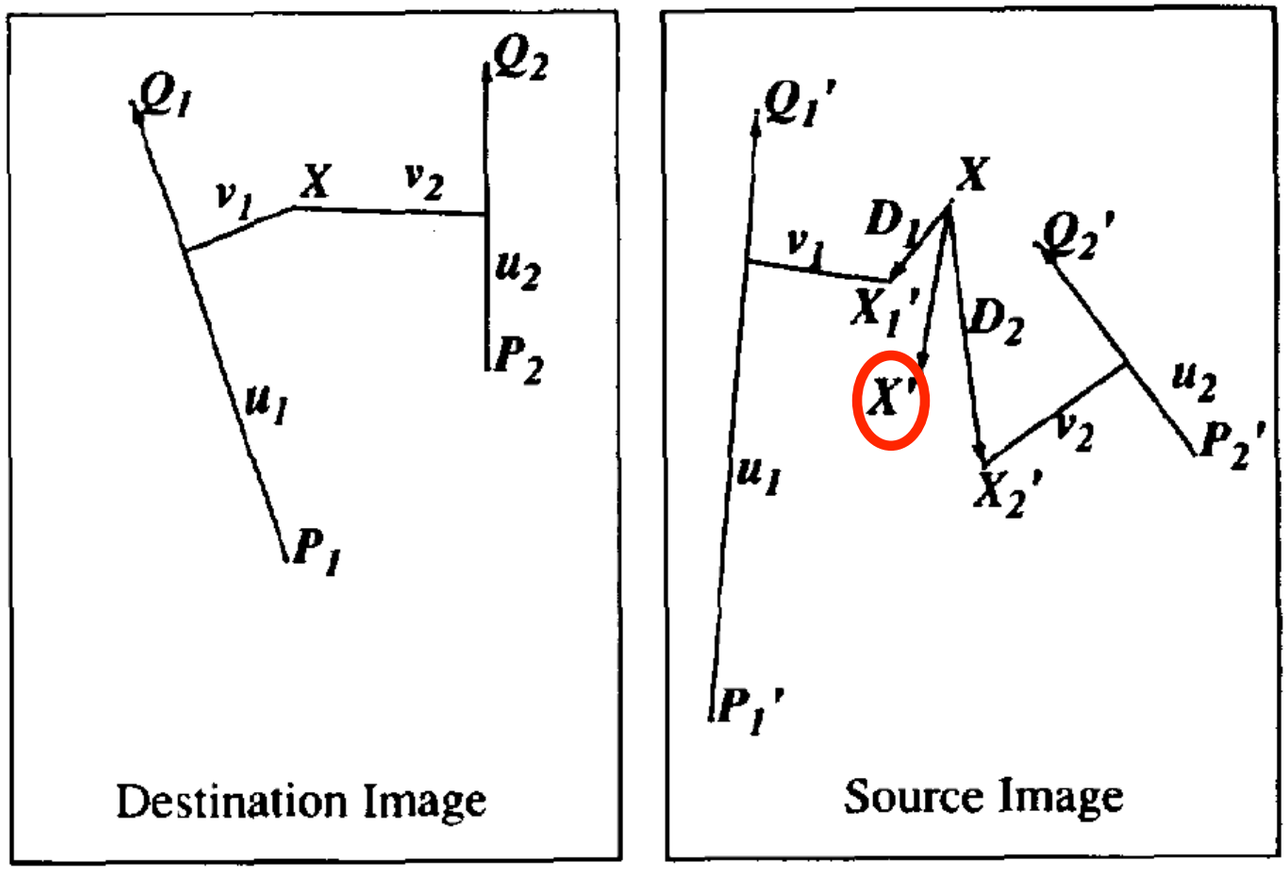
Examples





Each of these is an Euclidean warp, why?

Working with Multiple Segments

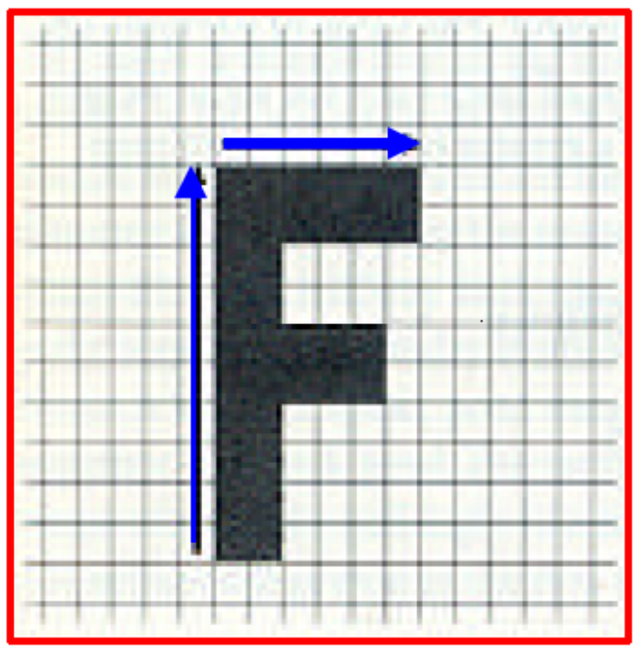


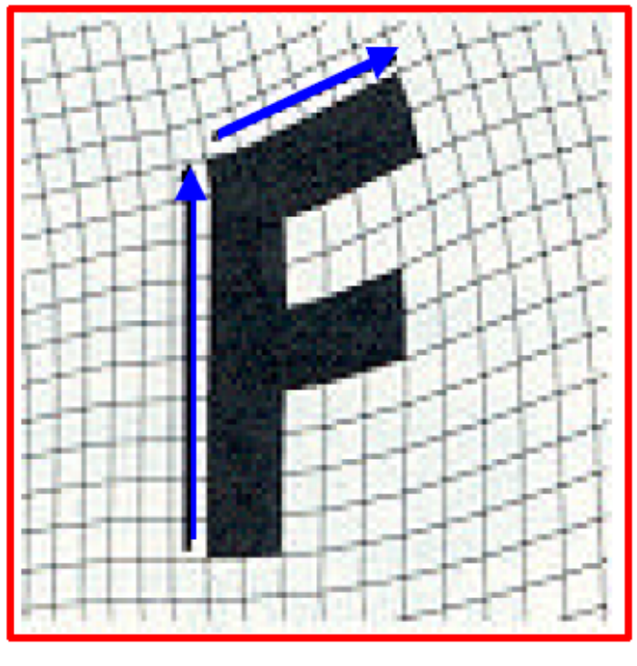
$$D_i = X_i^{'} - X_i$$

$$weight[i] = \left(\frac{length[i]^p}{a + dist[i]} \right)^b$$

- For each segment pair, perform transformation from X to Xi'
- Compute a weighted average of each Xi' where weights are based on:
  - length = length of the line segment,
  - dist = distance to line segment, and
  - Influence controlled by parameters a, p, b.

Algorithm

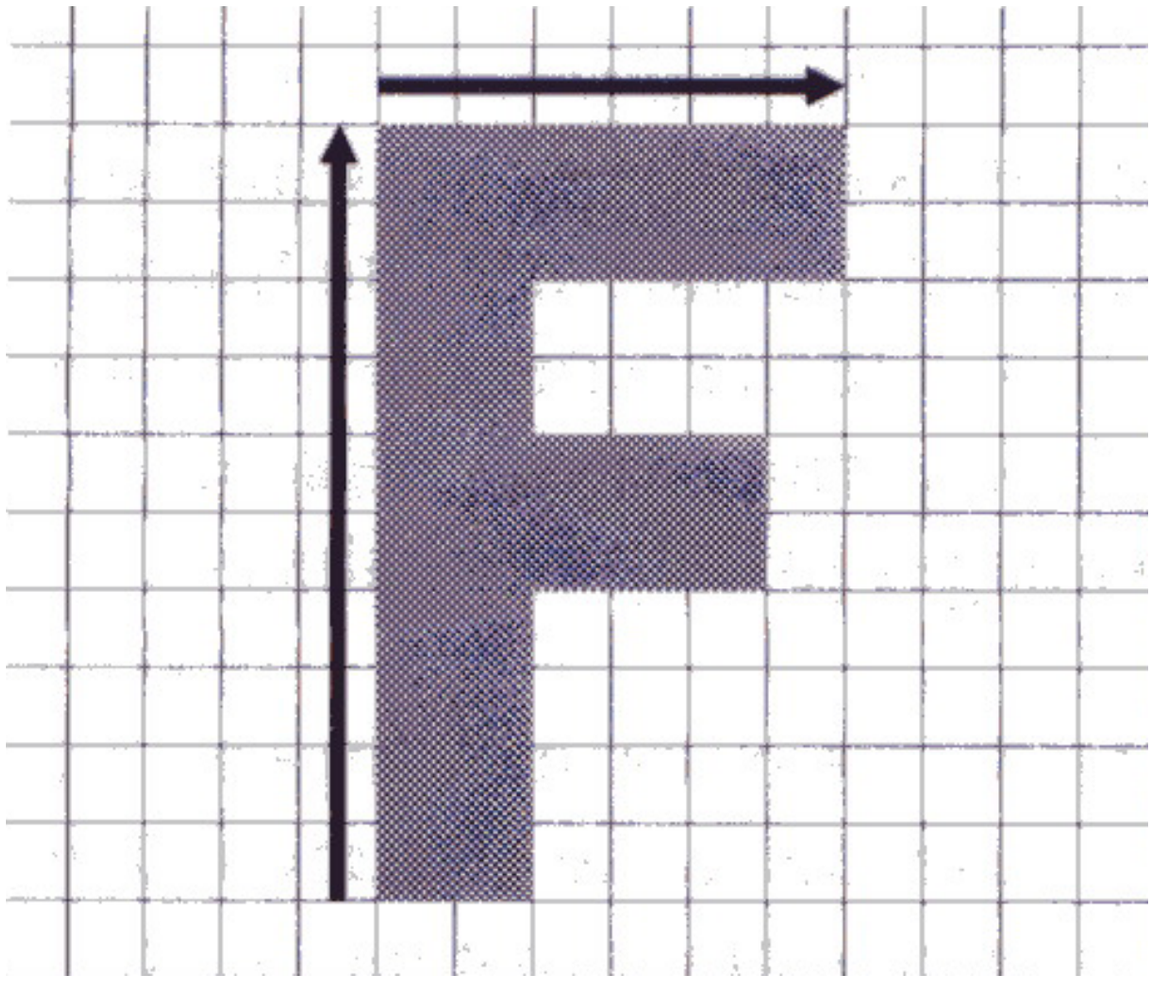


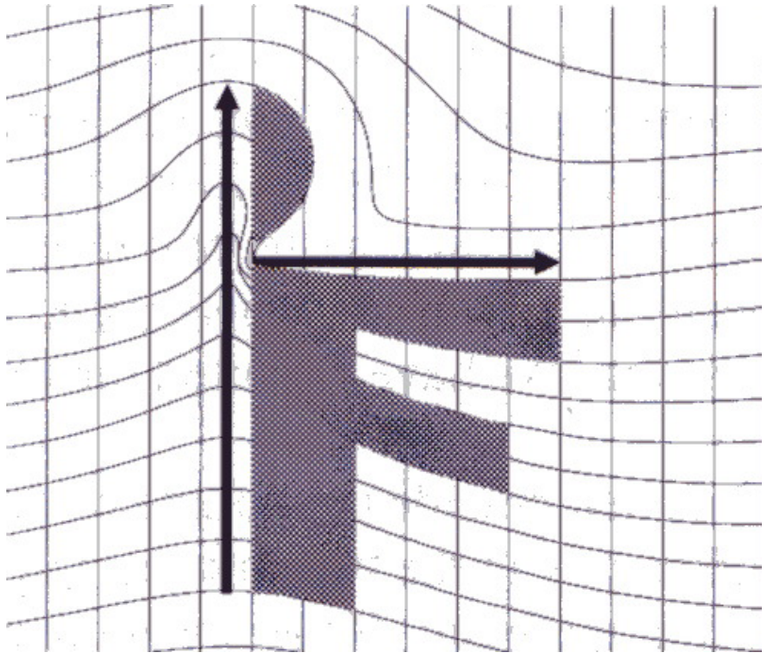


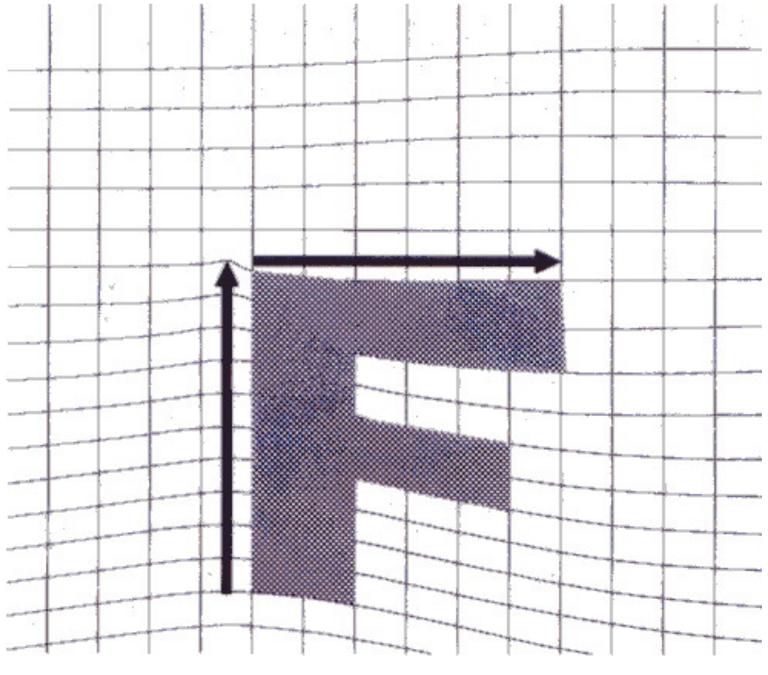
```
WarpImage(SourceImage,  L'[...], L[...])
begin
        foreach destination pixel X do
                XSum = (0,0)
                WeightSum = 0
                foreach line L[i] in destination do
                        X'[i]= X transformed by (L[i],L'[i])
                        weight[i] = weight assigned to X'[i]
                        XSum = Xsum + X'[i] * weight[i]
                        WeightSum += weight[i]
                end
                X' = XSum/WeightSum
                DestinationImage(X) = SourceImage(X')
        end
        return Destination
end
```

Comparison to Mesh Morphing

- Pros: More Expressive
- Cons: Speed and Control







Summary: Full Segment-Based Morph Pipeline

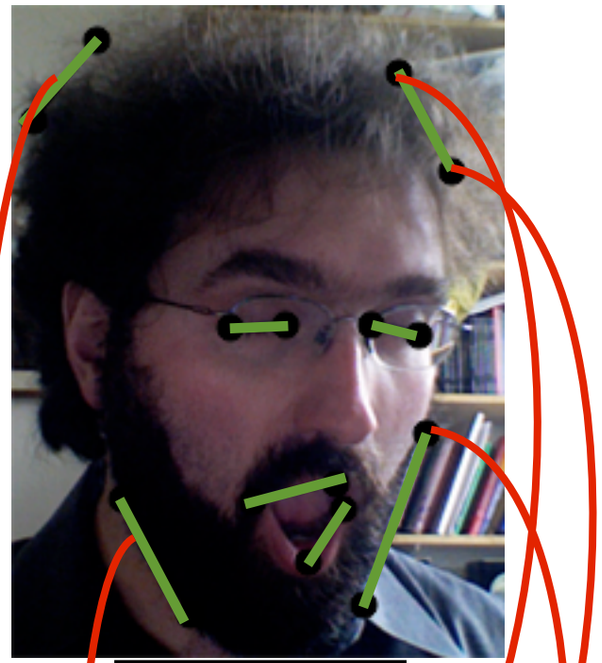
Input Images

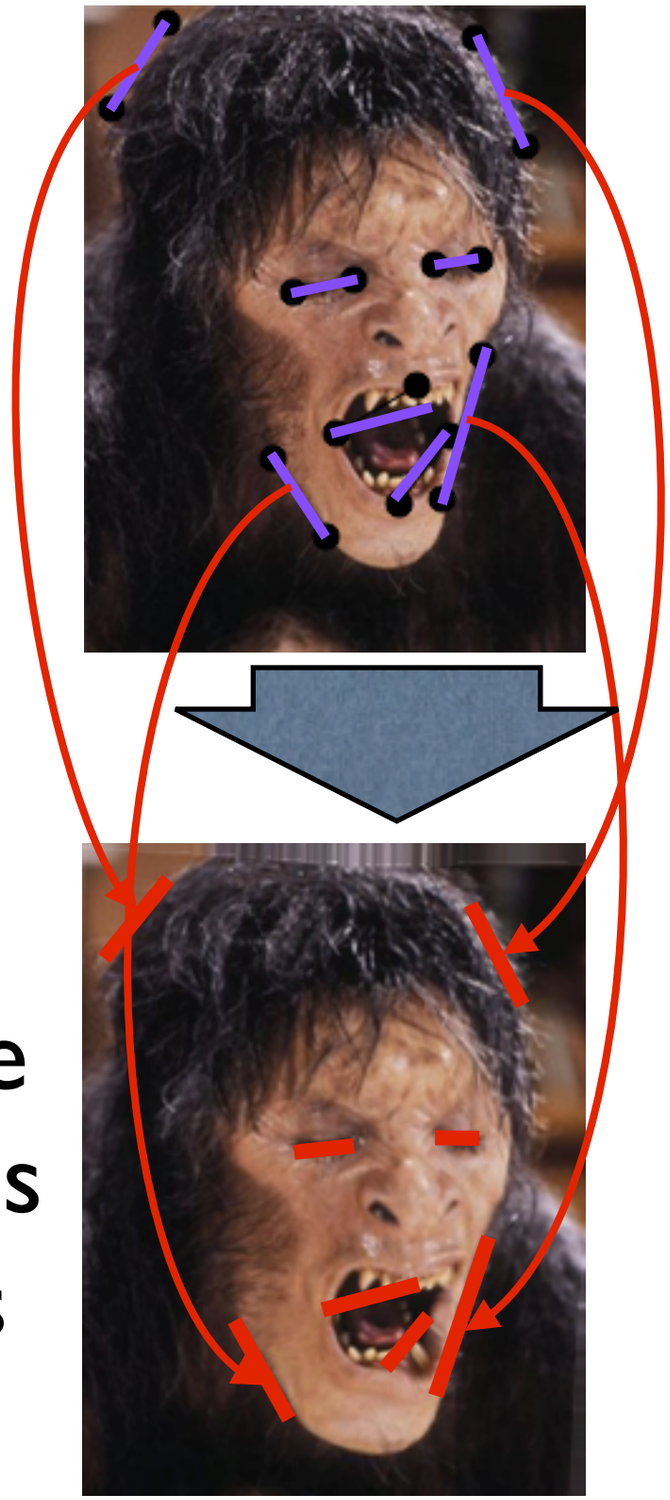
Segments

Interpolate Segments

t=0.5

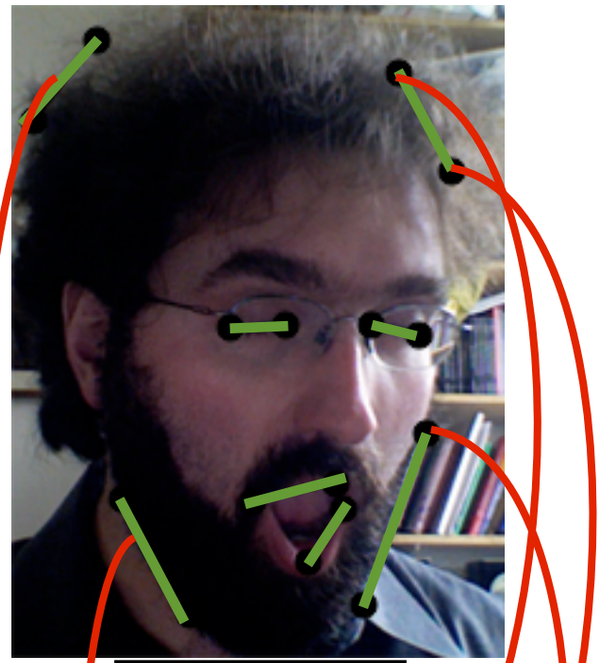
Warp Images to Segments

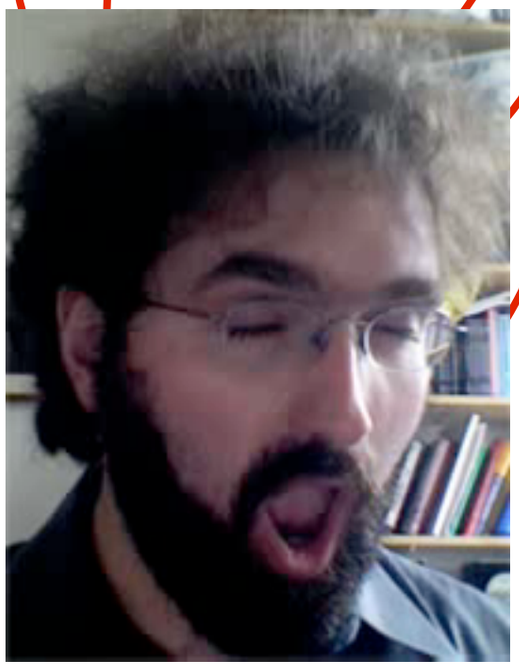




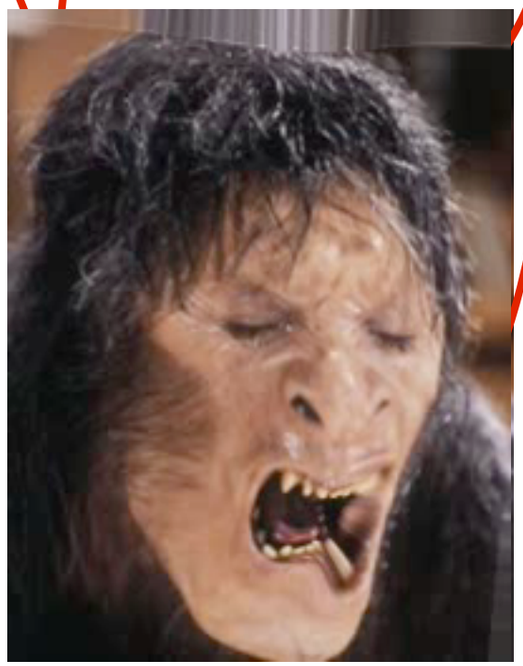
The red segments are at the same location in both images
Image features such as eyes are aligned

Warp Images to Segments





The red segments are at the same location in both images
Image features such as eyes are aligned



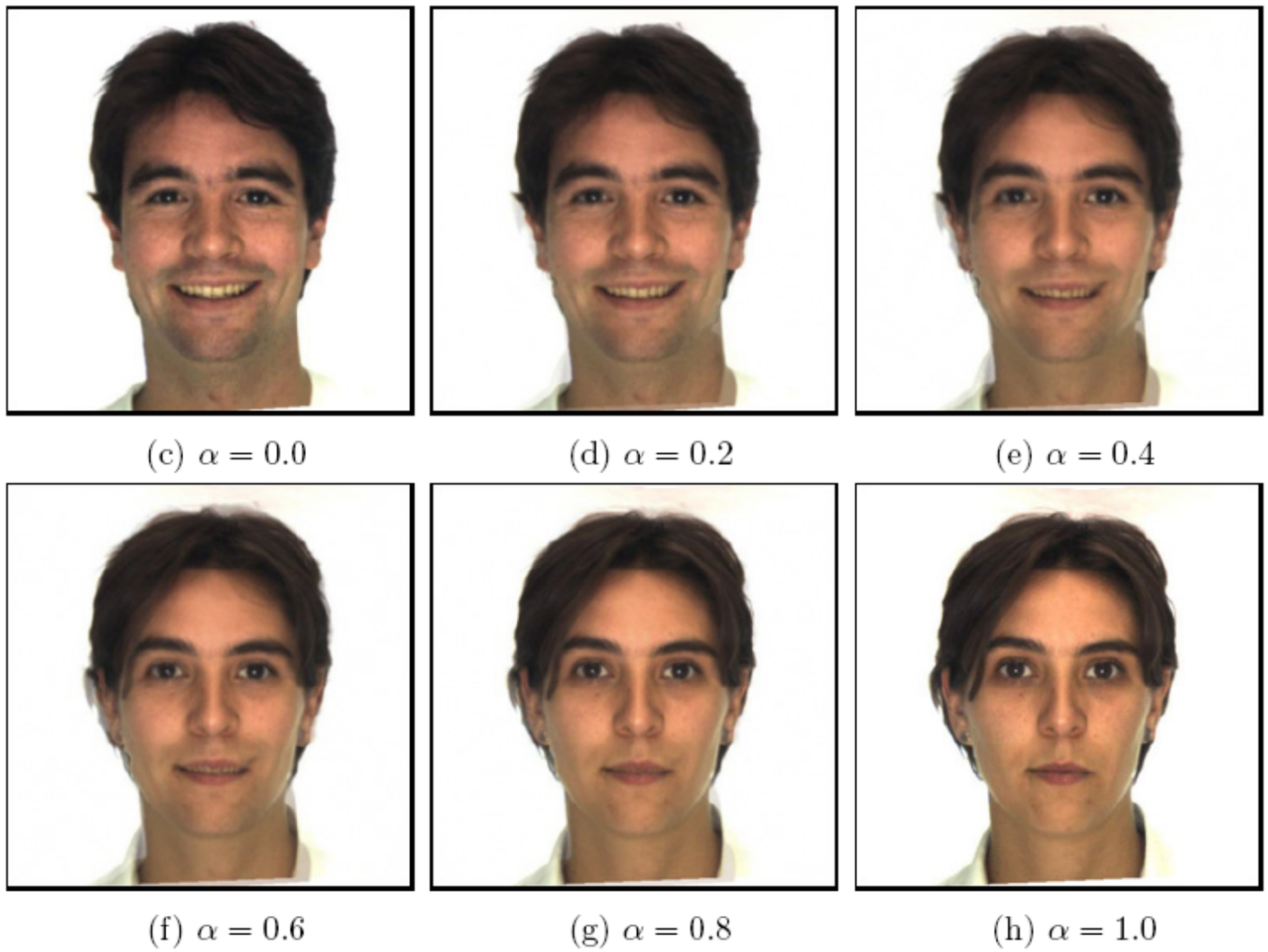
Interpolate Color

Interpolate Color

Other Thoughts

Morphing and Matting

- Extract foreground first to avoid artifacts in the background



(c) $\alpha = 0.0$ (d) $\alpha = 0.2$ (e) $\alpha = 0.4$

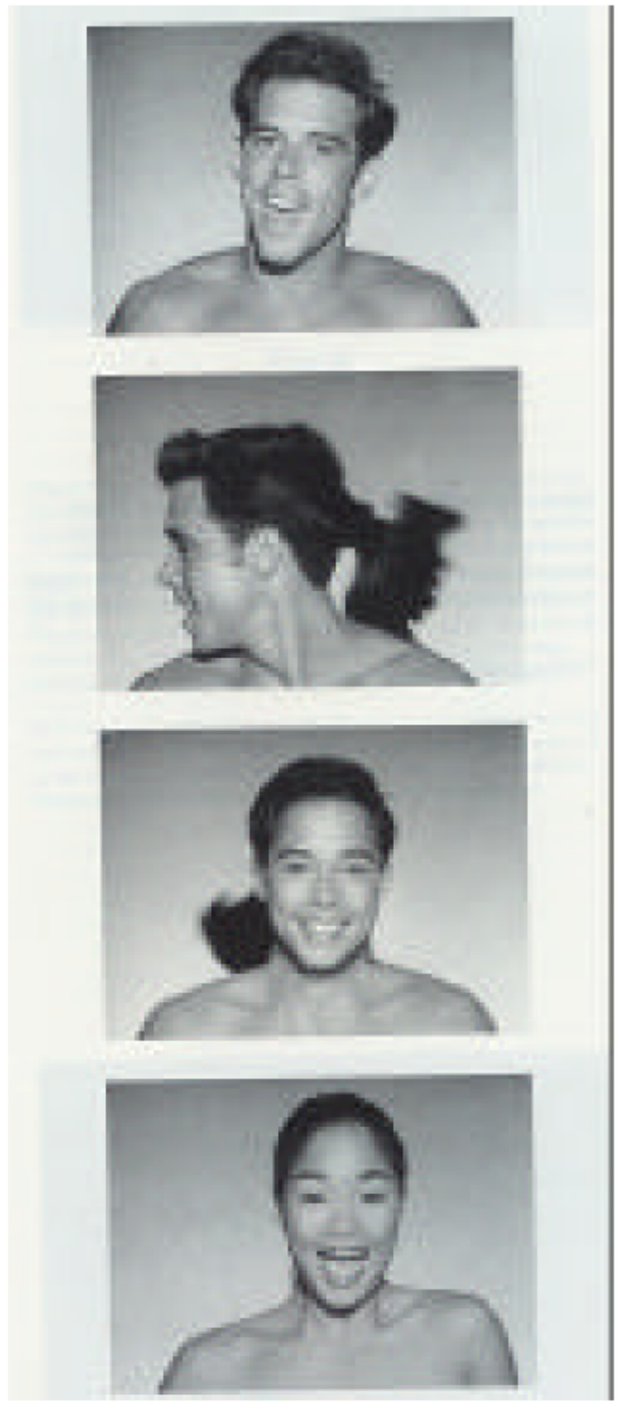
(f) $\alpha = 0.6$ (g) $\alpha = 0.8$ (h) $\alpha = 1.0$

Animated Sequences

- Specify keyframes and interpolate the lines for the in between frames

- Require a lot of tweaking!

- Very time consuming, but can be done!



Panoramas

(Switching Topics)

Why Panoramas?

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

Slide from Brown & Lowe

Why Panoramas?

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

Slide from Brown & Lowe

Why Panoramas?

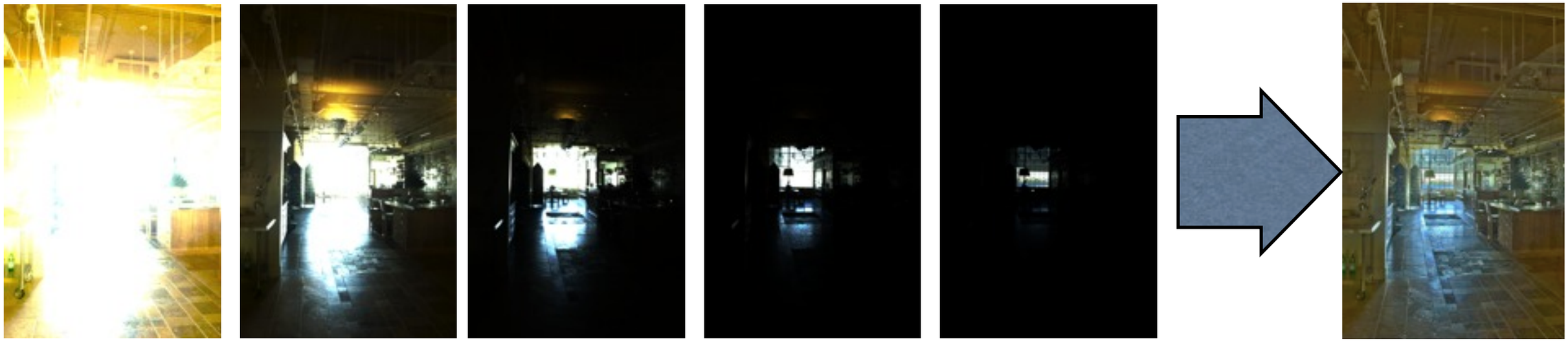
- Are you getting the whole picture?
- Compact Camera FOV = 50 x 35°
- Human FOV = 200 x 135°
- Panoramic Mosaic = 360 x 180°

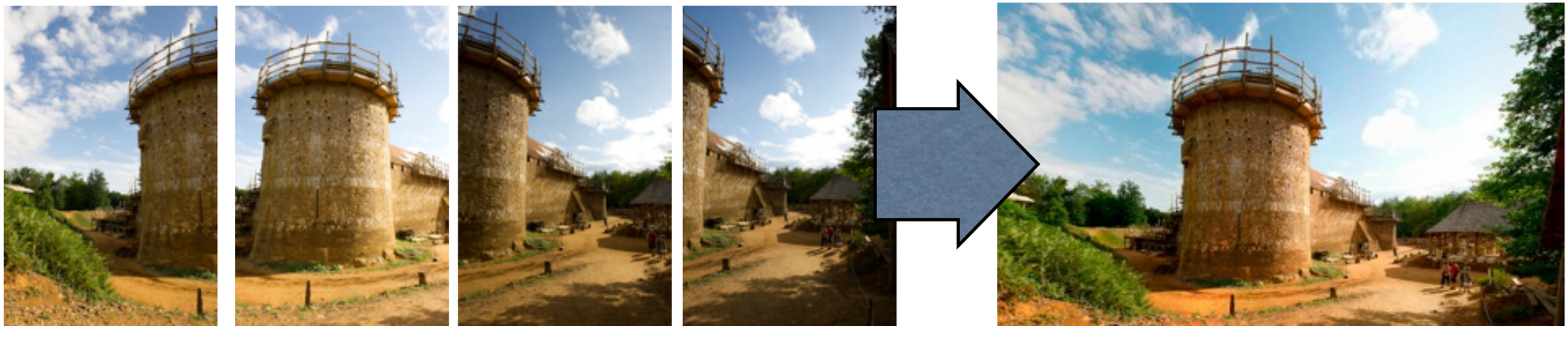
Slide from Brown & Lowe

Panoramas: Stitching Images Together

We Will Employ a Similar Idea to How HDR Images are Made

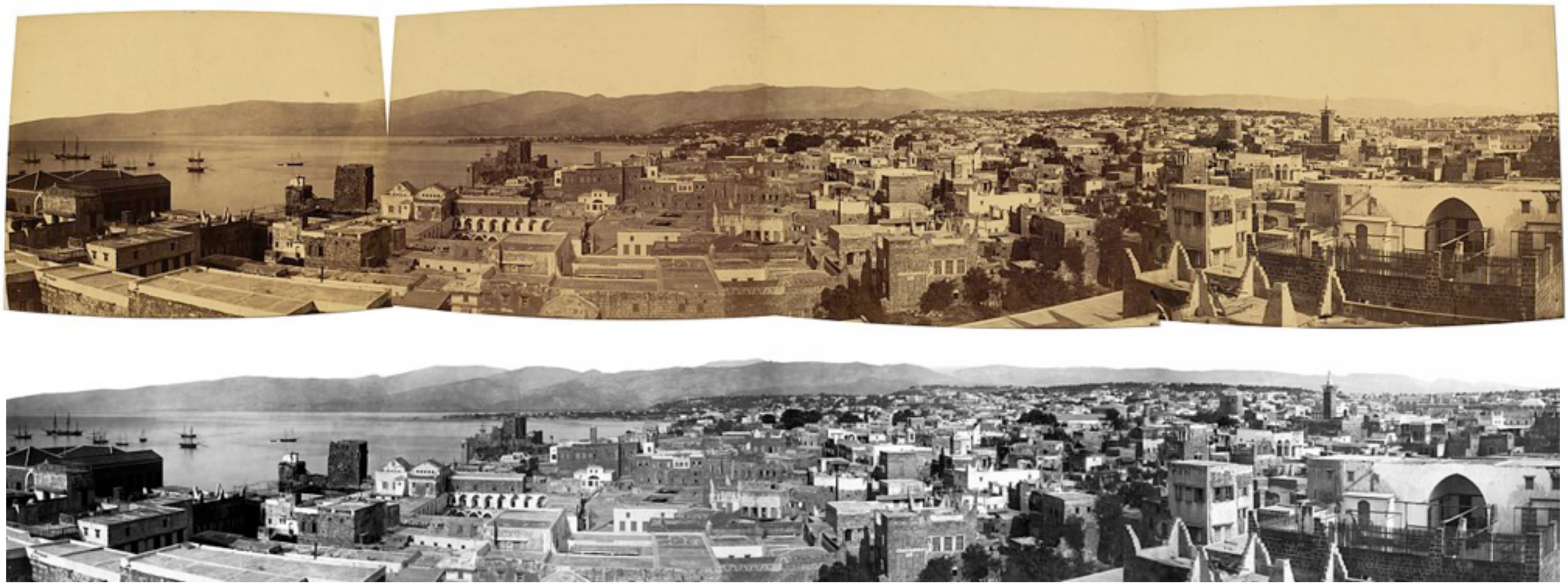
- Medium is limited (range, field of view)
- Take multiple images
- Merge





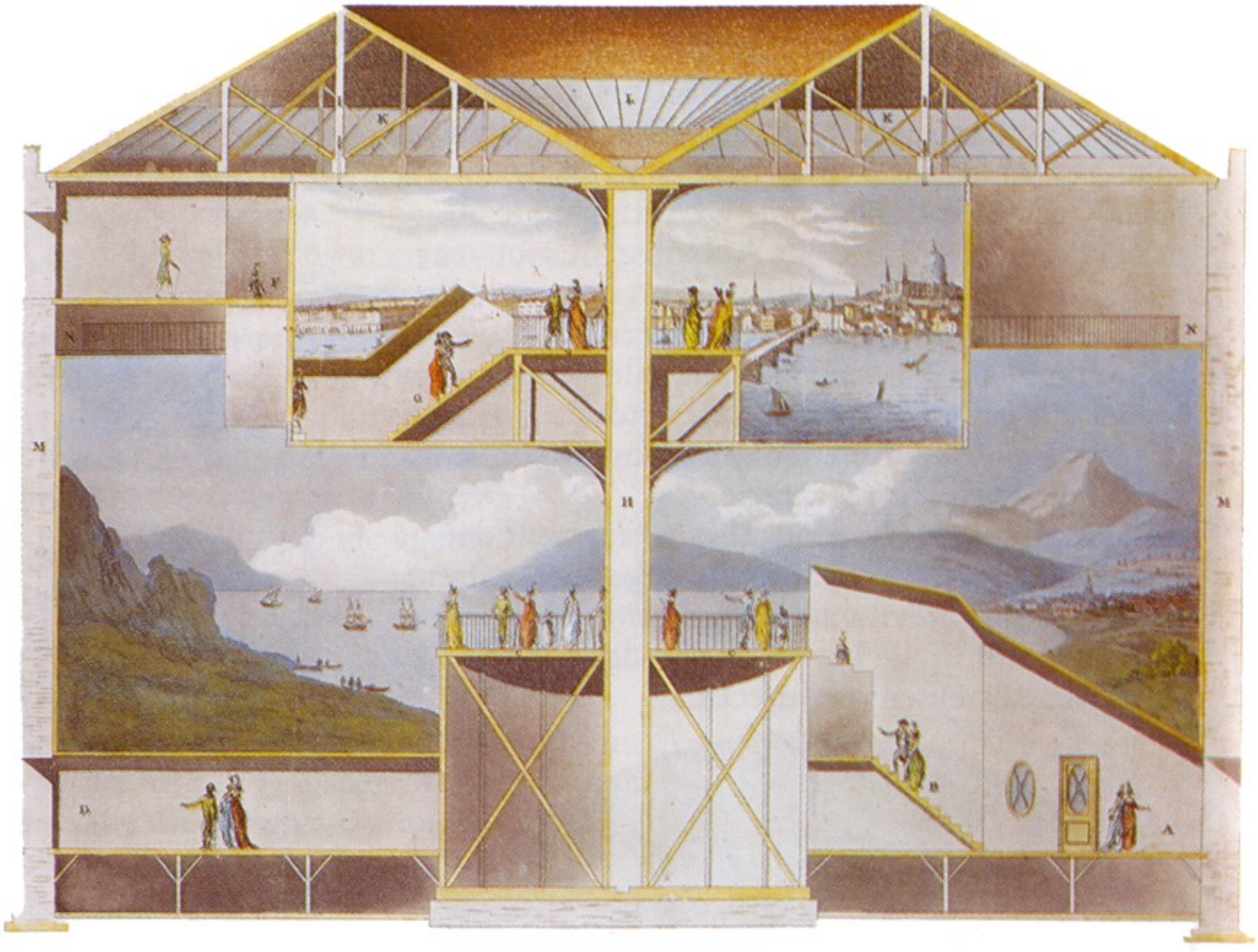
An Old Idea

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



This image by the firm of Maison Bonfils depicts the city of Beirut, Lebanon, sometime in the last third of the 19th century. Maison Bonfils was the extraordinarily prolific venture of the French photographer Félix Bonfils (1831–85), his wife Marie-Lydie Cabanis Bonfils (1837–1918), and their son, Adrien Bonfils (1861–1928). The Bonfils moved to Beirut in 1867 and, over the next five decades, their firm produced one of the world's most important bodies of photographic work about the Middle East. Maison Bonfils was known for landscape photographs, panoramas, biblical scenes, and posed “ethnographic” portraits. The family’s marketing acumen and commercial sense helped make their photographs known around the world. “Panoramic” photographs employ a variety of techniques to create a wide angle of view. This “panoramic view” is comprised of four aerial photographs set together to give the viewer a broader image than would have been practical with a single photograph.

An Old Idea (Robert Barker (1739-1806)



http://www.janeausten.co.uk/the-barker-family-panorama-painters/

A Really Old Idea (1085-1145)

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

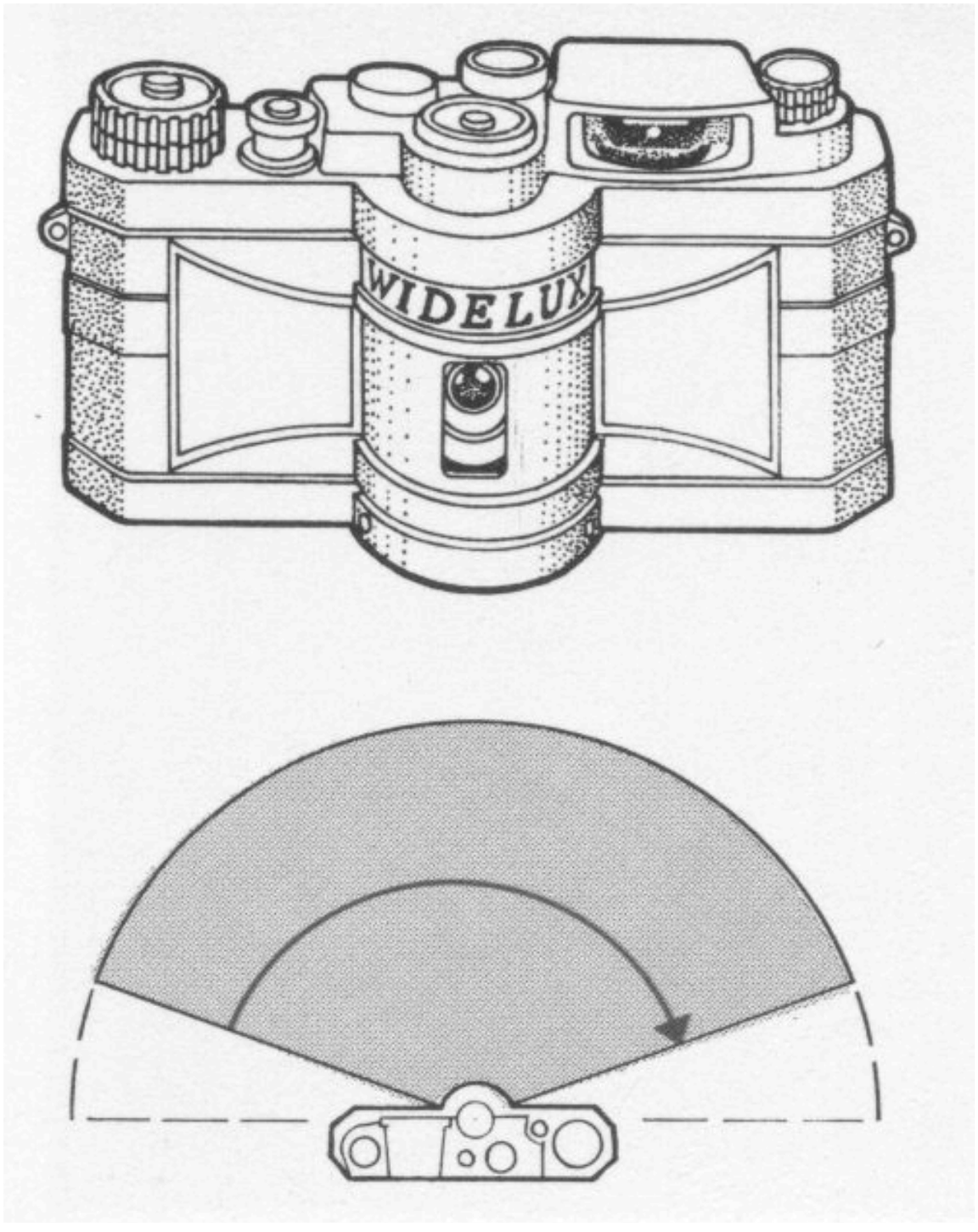
A Really Old Idea (1085-1145)

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

A Really Old Idea (1085-1145)

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

Traditional Panoramas



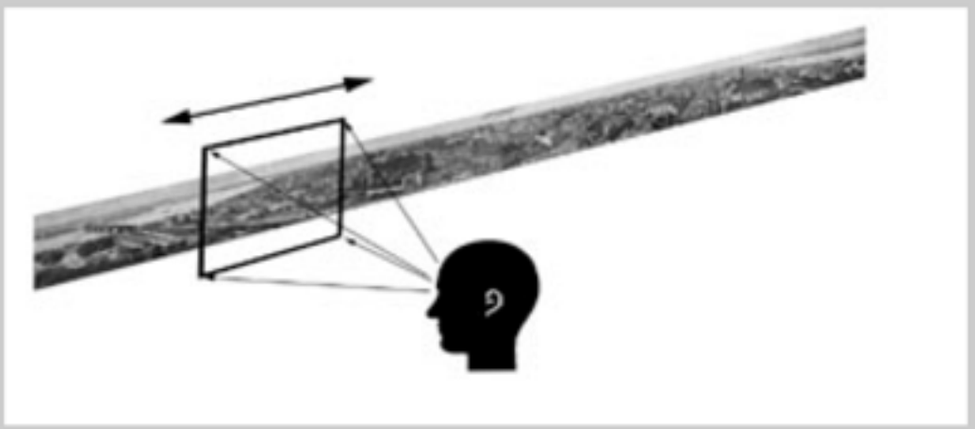


Figure 1. Linear presentation of a circular panorama (old panoramic photographs on paper or a large canvas, rectangular images on a computer screen).

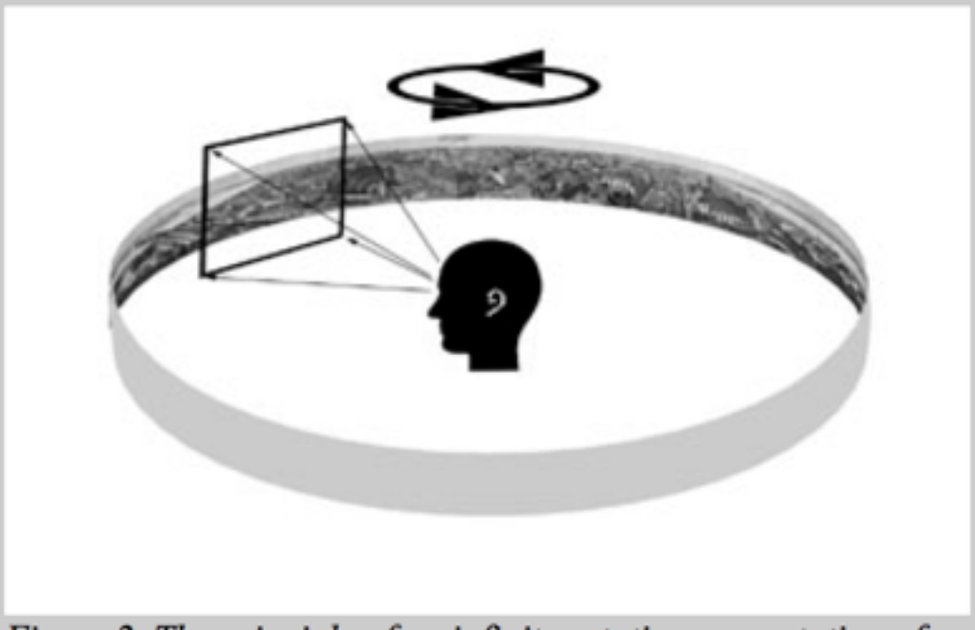
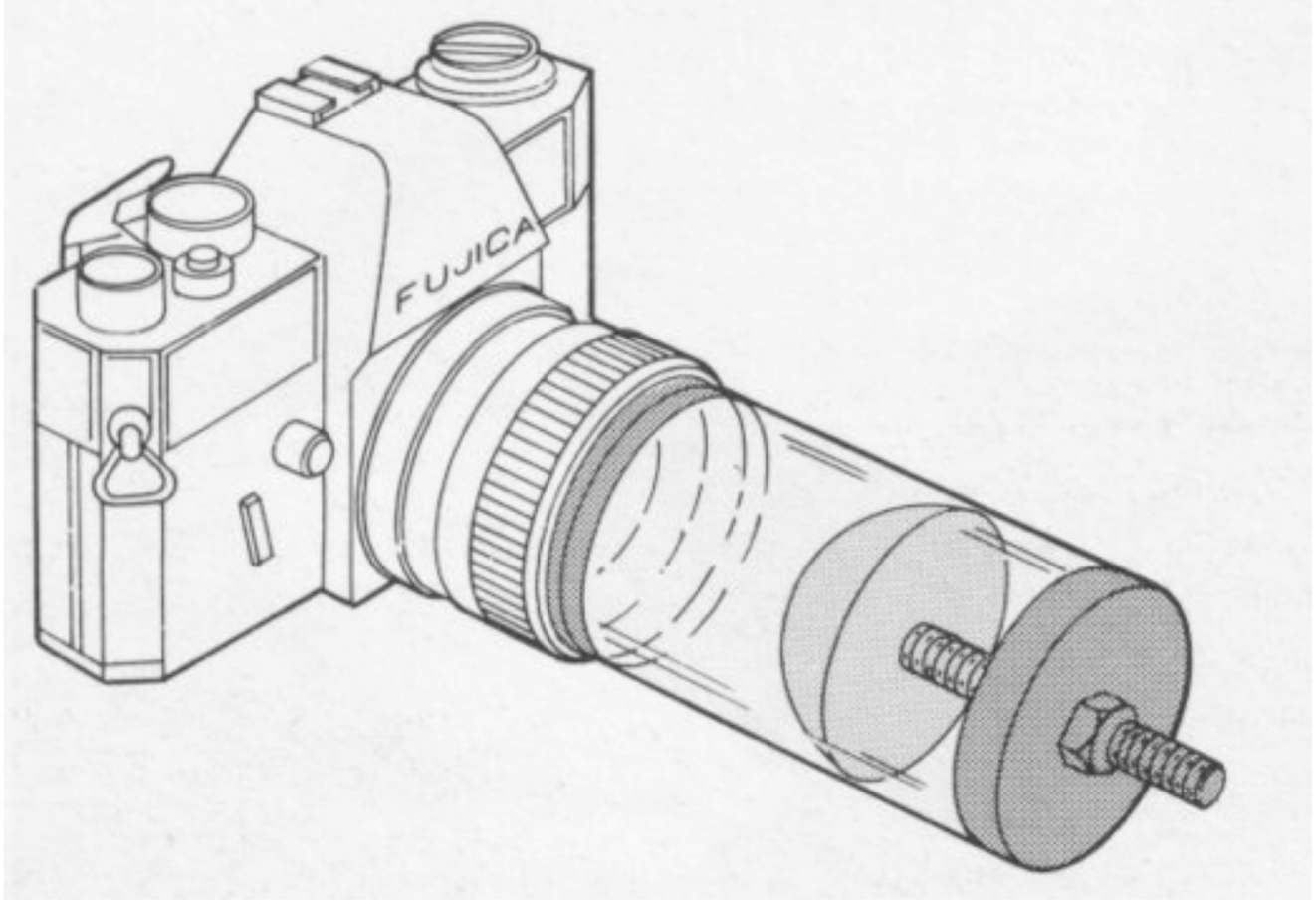


Figure 2. The principle of an infinite rotation presentation of a circular panorama (circular room or a computer screen and panorama viewer).

http://www.eurofresh.se/history/

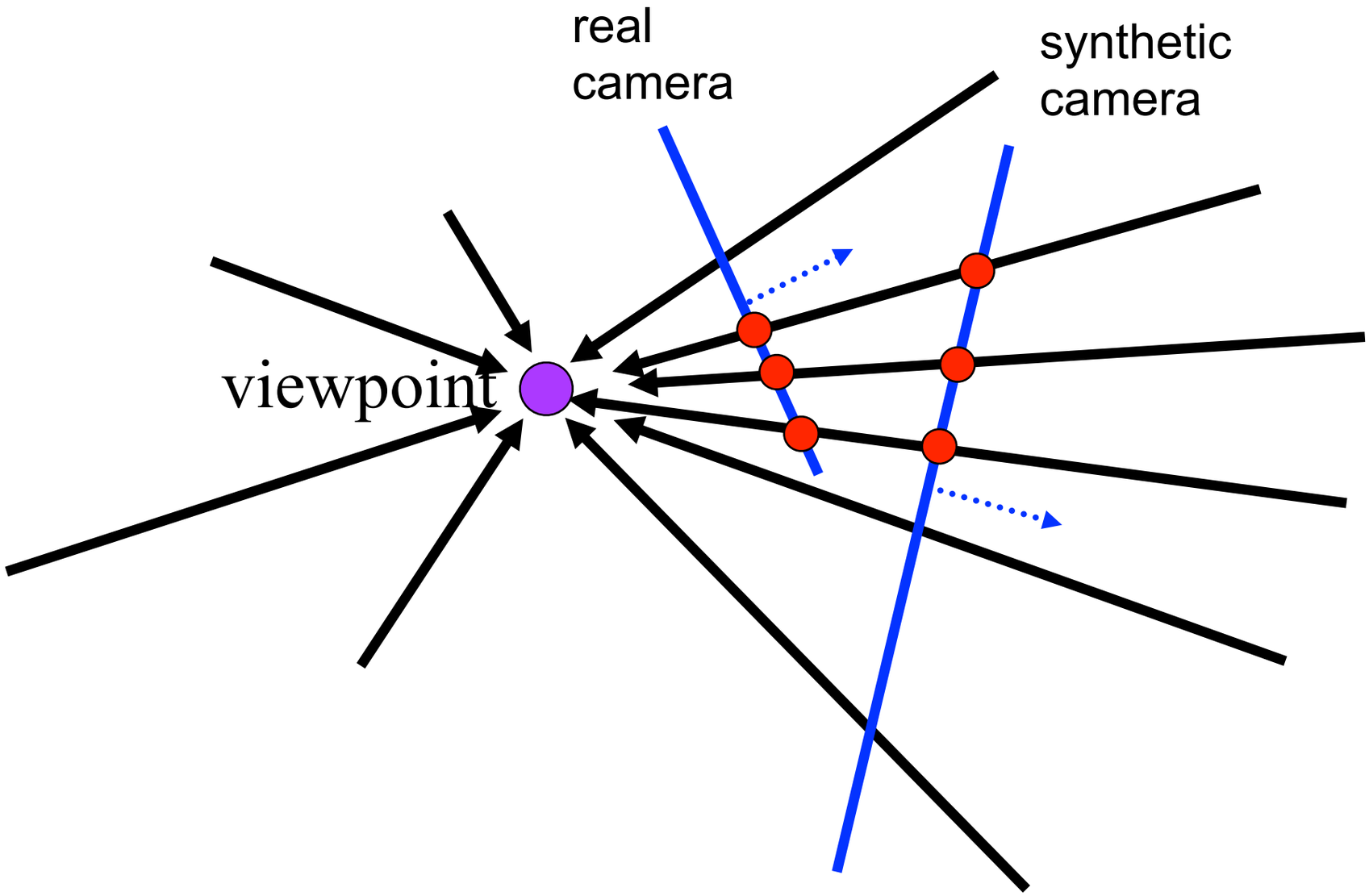
Modern Panoramas

- http://www.0-360.com/
- http://www.bugeyedigital.com/product_main/036-0360d.html
- http://www.panoramicphoto.com/timeline.htm



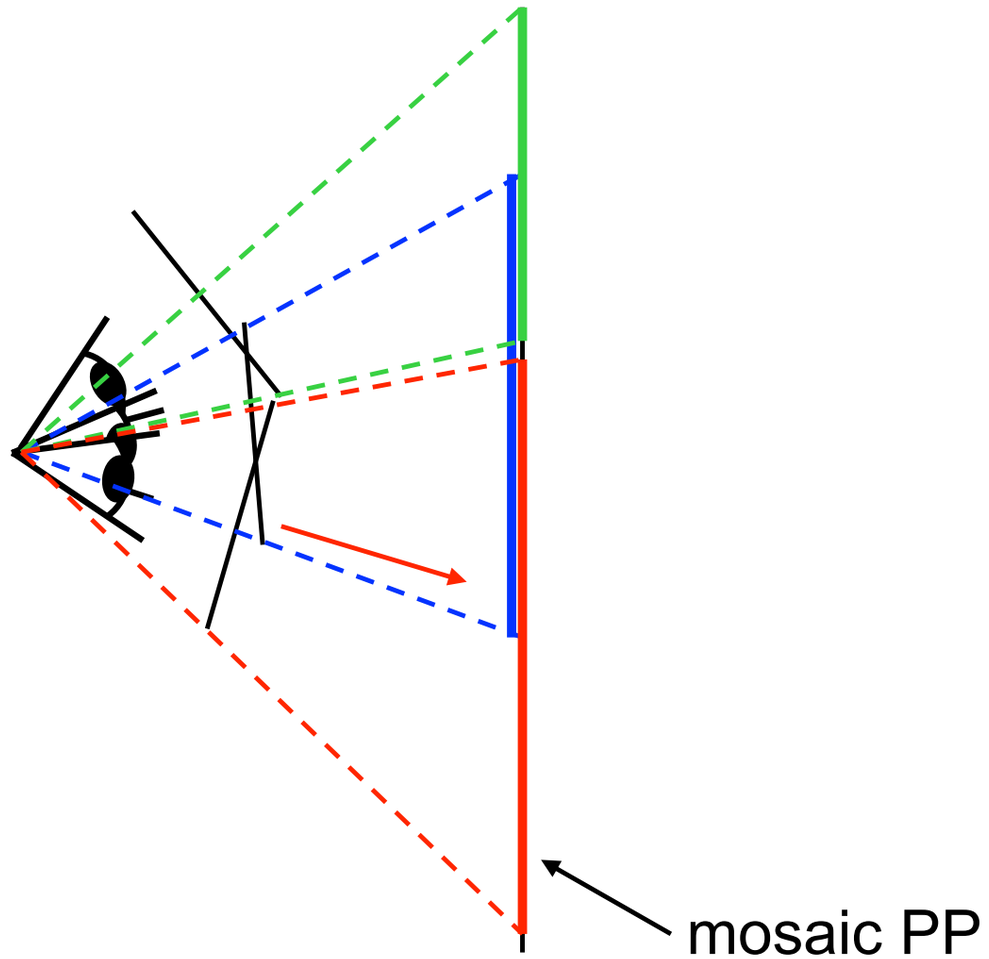
Reprojection

A Pencil of Rays Contains All Views



Can generate any synthetic camera view
as long as it has **the same center of projection**!

Image Reprojection

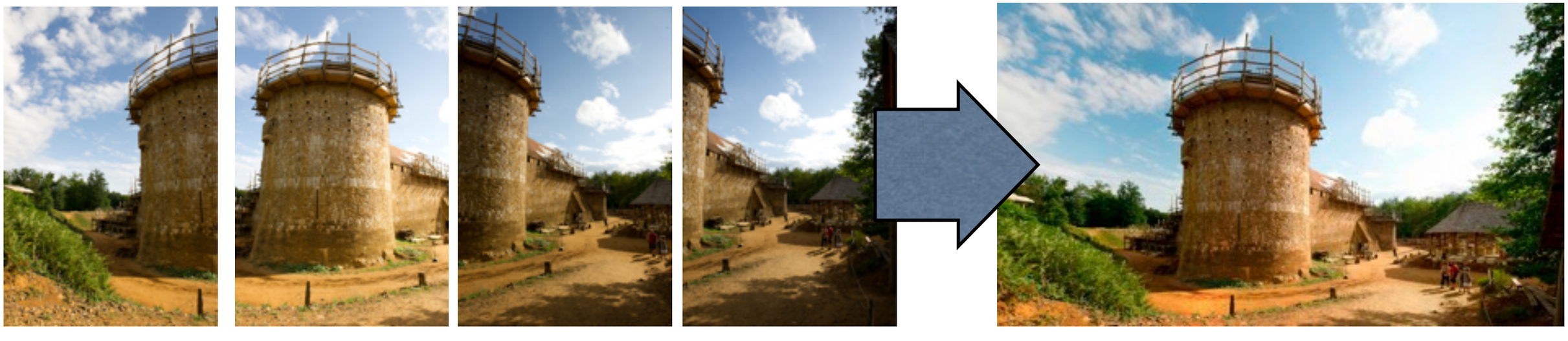


The mosaic has a natural interpretation in 3D

- The images are reprojected onto a common plane
- The mosaic is formed on this plane
- Mosaic is a *synthetic wide-angle camera*

Virtual Wide Angle

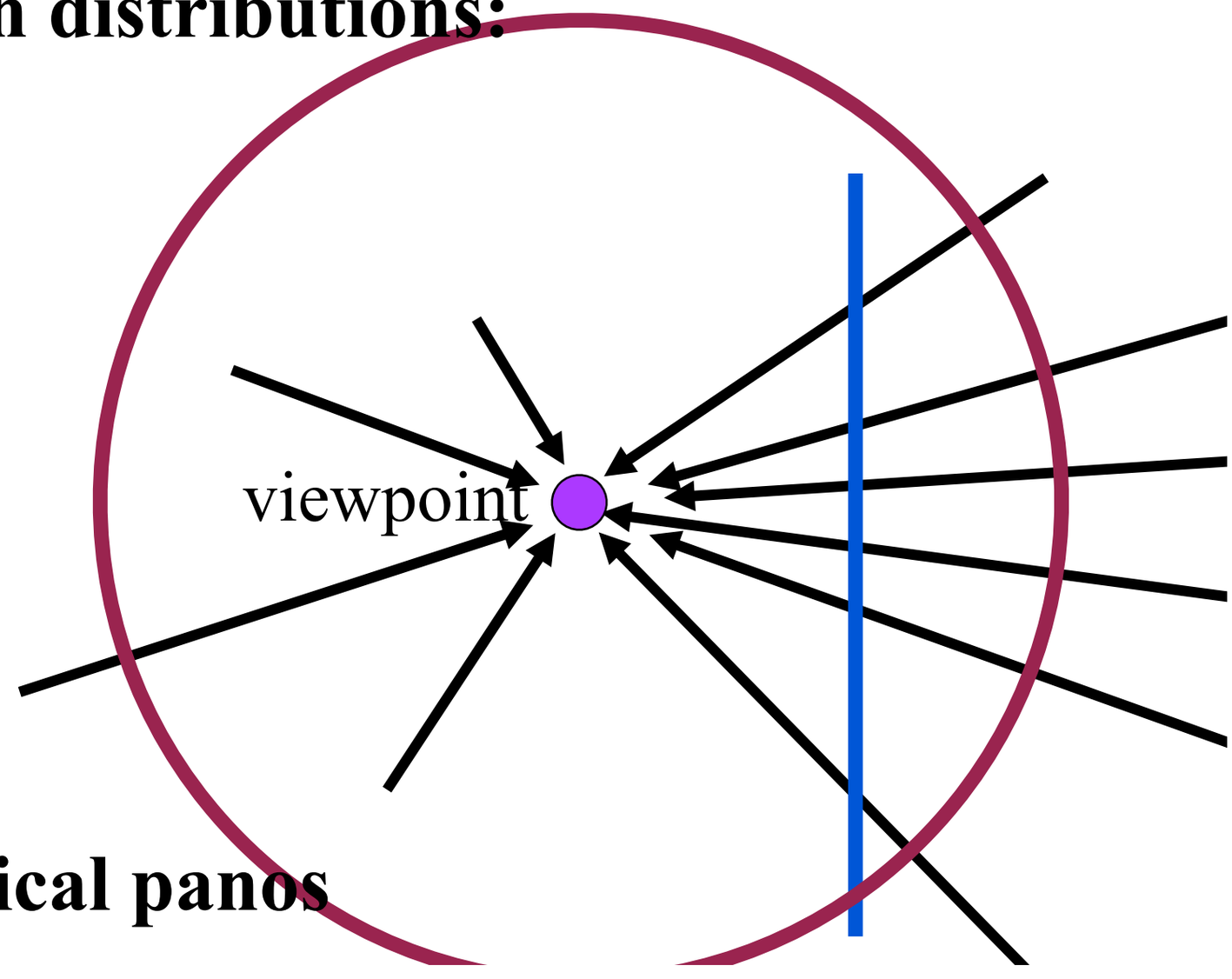
- Take N images in different directions
- Deduce the image that would have been taken by a wider angle lens
- ...but wait, why should this work at all?
  - What about the 3D geometry of the scene?
  - Why aren’t we using it?



Another Interpretation

- **Depth does not matter**
- **We can pretend that each pixel is at a convenient depth**

- **Three convenient depth distributions:**
  - spherical
  - planar
  - cylindrical

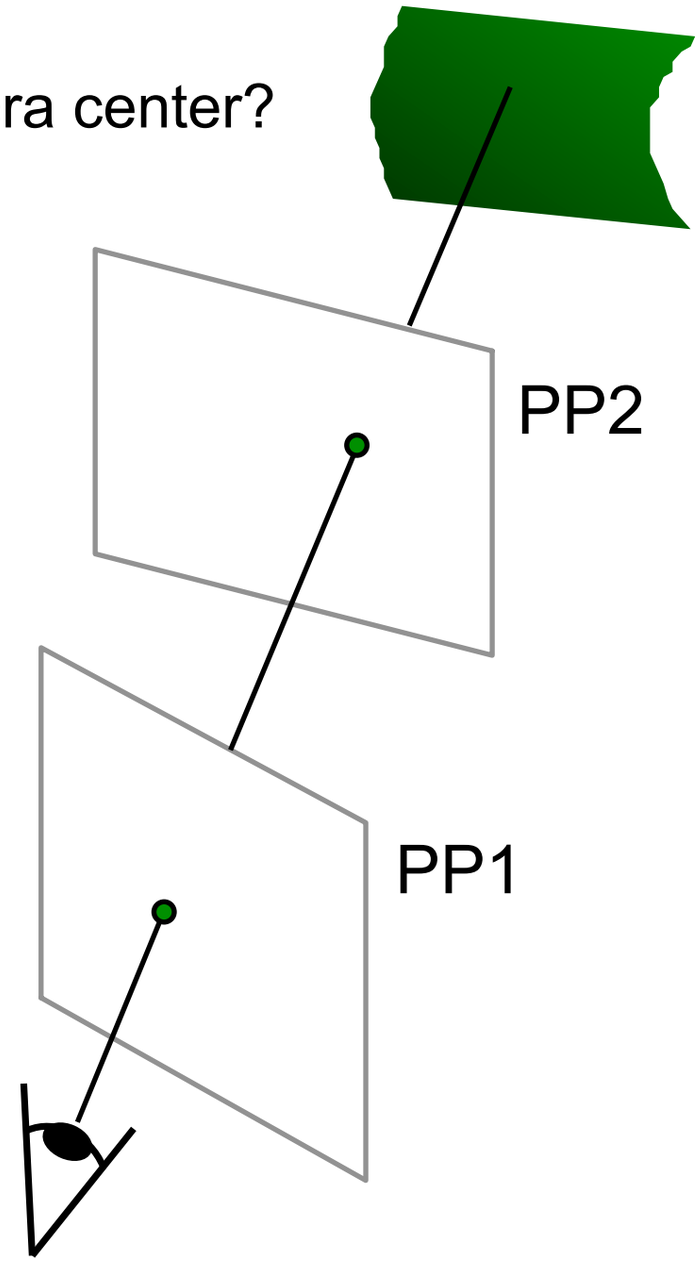


- **We focus on planar**
  - **it makes life more linear**
  - **Still useful for spherical panos**

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

But don't we need to know the geometry of the two planes in respect to the eye?

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

Projective Geometry

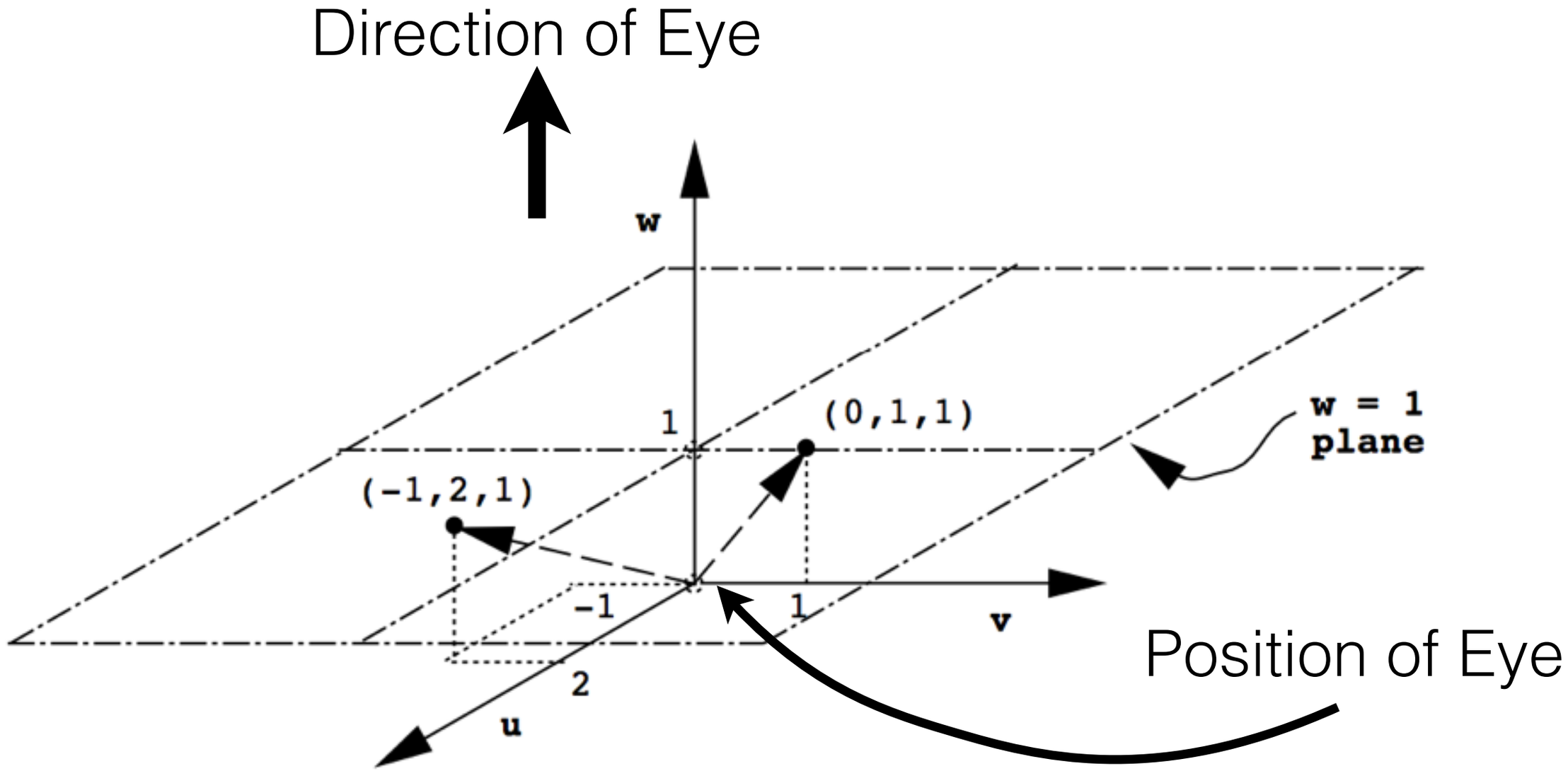
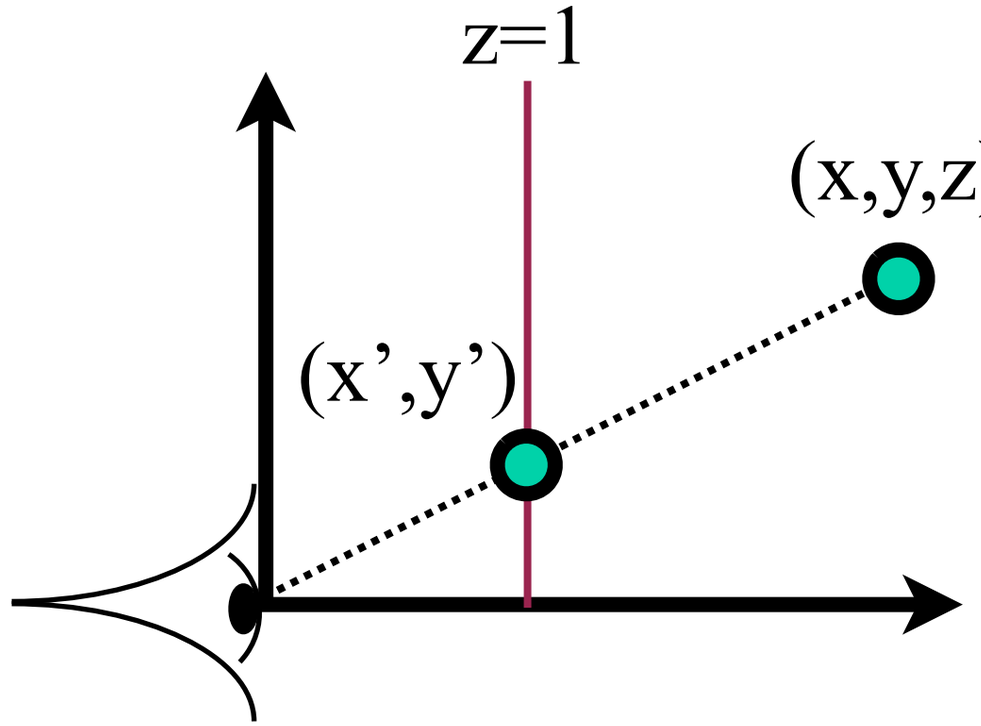


Figure 9.10: Homogeneous Coordinates Lie on the Plane $w = 1$

- When a_{31}, a_{32} do not equal zero, we tilt this plane
- But we are still looking at it along the w-axis, eye is located at (0,0,0)
- Dividing by w shortens vector to lie in w = 1 plane
- Given (x,y,w) and (x/w,y/w,1), the point still lies on the same ray exiting from the eye passing through (0,0,0) and (x,y,w)

Simple Perspective Projection

- Project all points to the $z = 1$ plane, viewpoint at the origin:
  - $x' = x/z$
  - $y' = y/z$

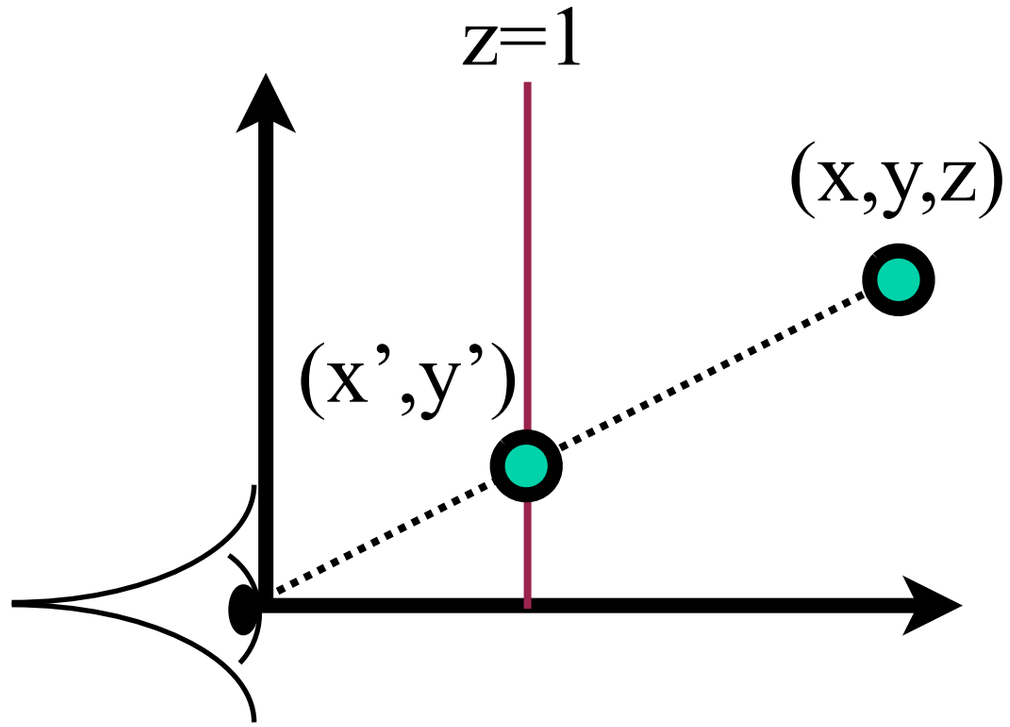


Simple Perspective Projection

- Project all points to the $z = 1$ plane, viewpoint at the origin:

  –x’=x/z

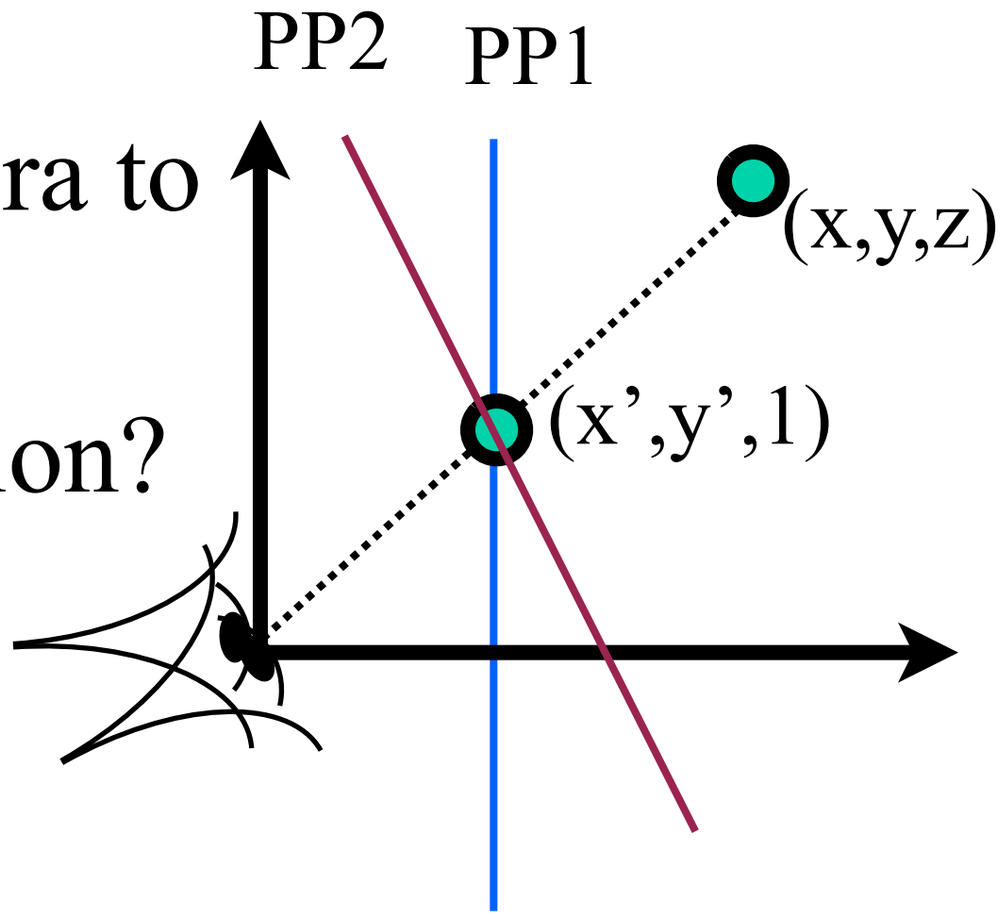
  –y’=y/z



- Can we represent this with a matrix?
  - –not directly (division)
  - –but we can cheat...
  - –add a third coordinate to the result
    - –interpret as : we always divide by 3rd coordinate
    - –see next slide...

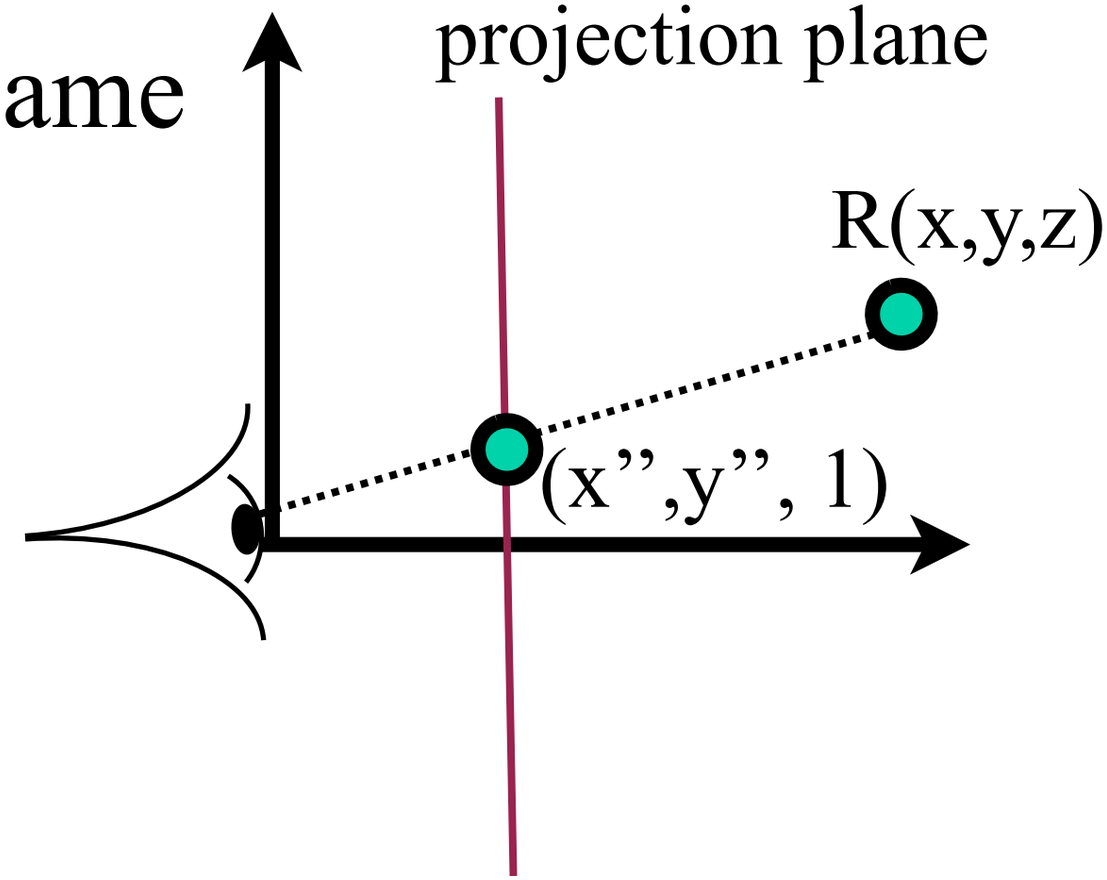
3D rotation

- We observe point x, y, z with camera PP1
- Now we rotate the 3D camera to PP2
- What is the new 2D projection?



3D rotation

- Now we rotate the 3D camera
- What is the new projection?
- Rotating the camera is the same as rotating the world in the opposite direction
- To project a (x,y,z) wrt rotated camera:
  - Apply rotation R to (x,y, z)
  - Apply projection division

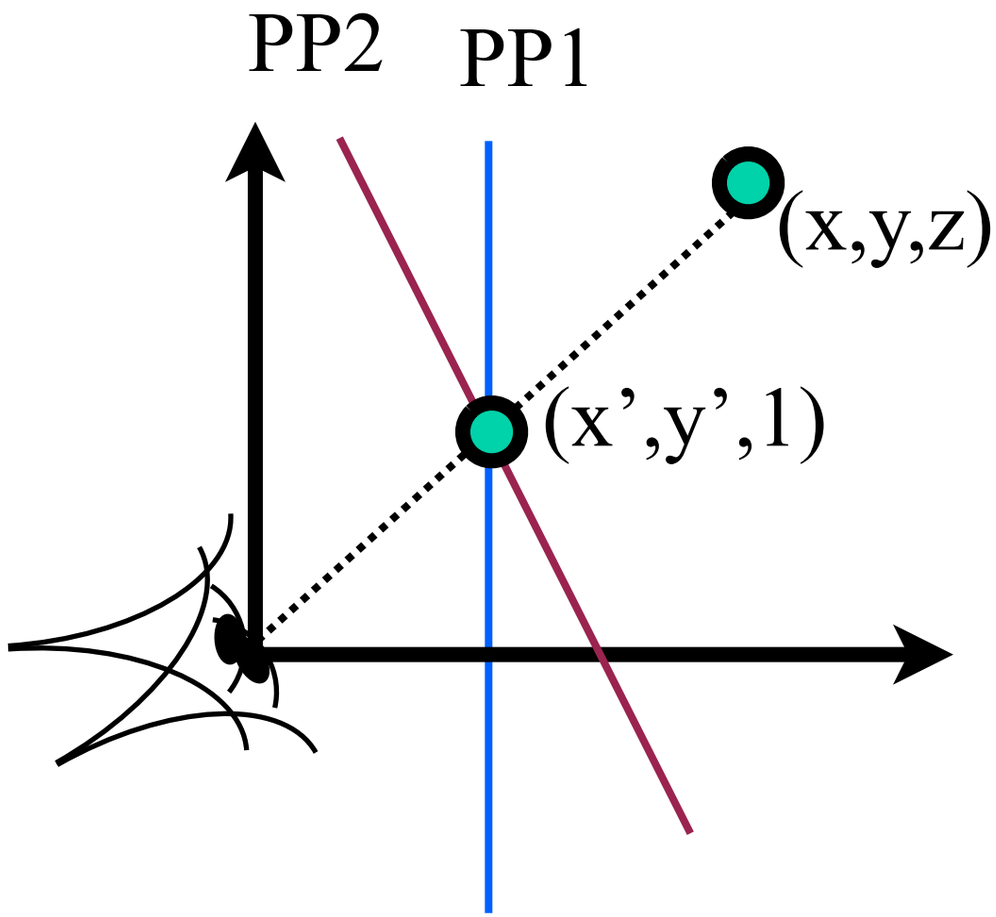


Summary

- Canonical projection = division by z
- Homogeneous coordinates are a notation trick to encapsulate this
- For other directions, just apply 3D rotation first
- But... this applies only when we know z
  - And for panorama stitching, we don’t
  - What are we going to do? Are we in big trouble? Should we give up? Buy a 3D scanner?

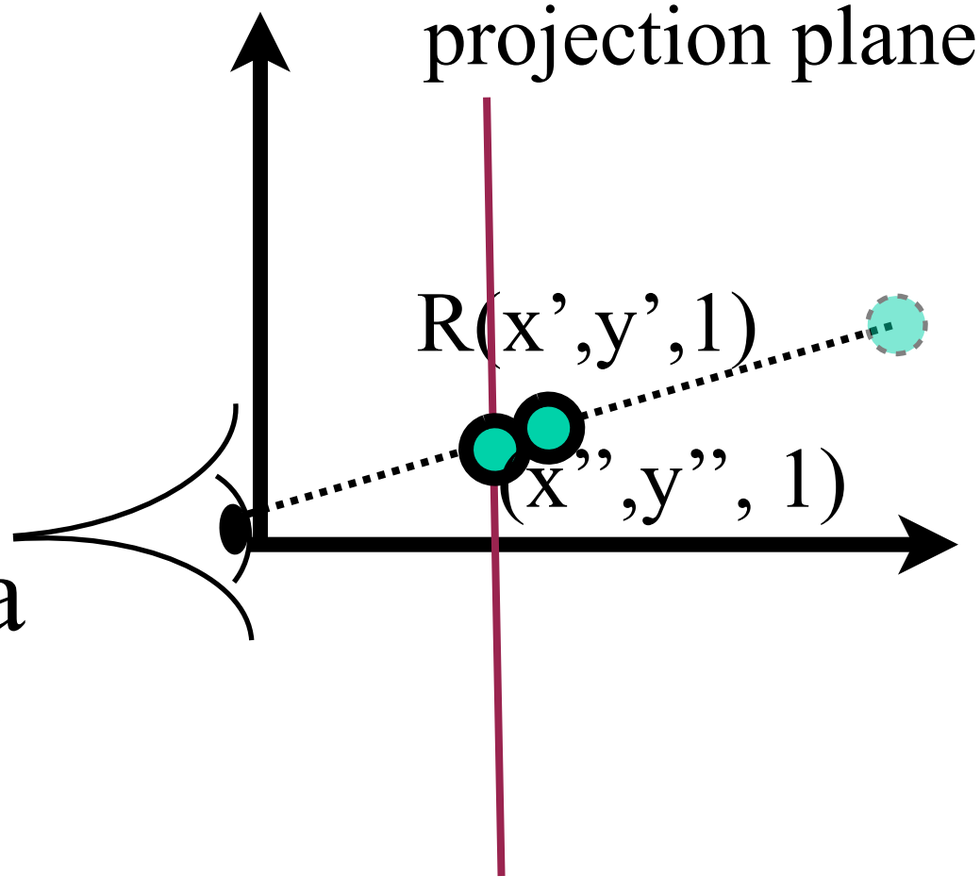
Camera rotation: unknown z

- Same thing but use (x’, y’, 1) instead of (x, y, z)
  - Rotate (x’, y’, 1) by R
  - Perform division



3D rotation

- Same thing but use (x’, y’, 1) instead of (x, y, z)
  - Rotate (x’, y’, 1) by R
  - Perform division

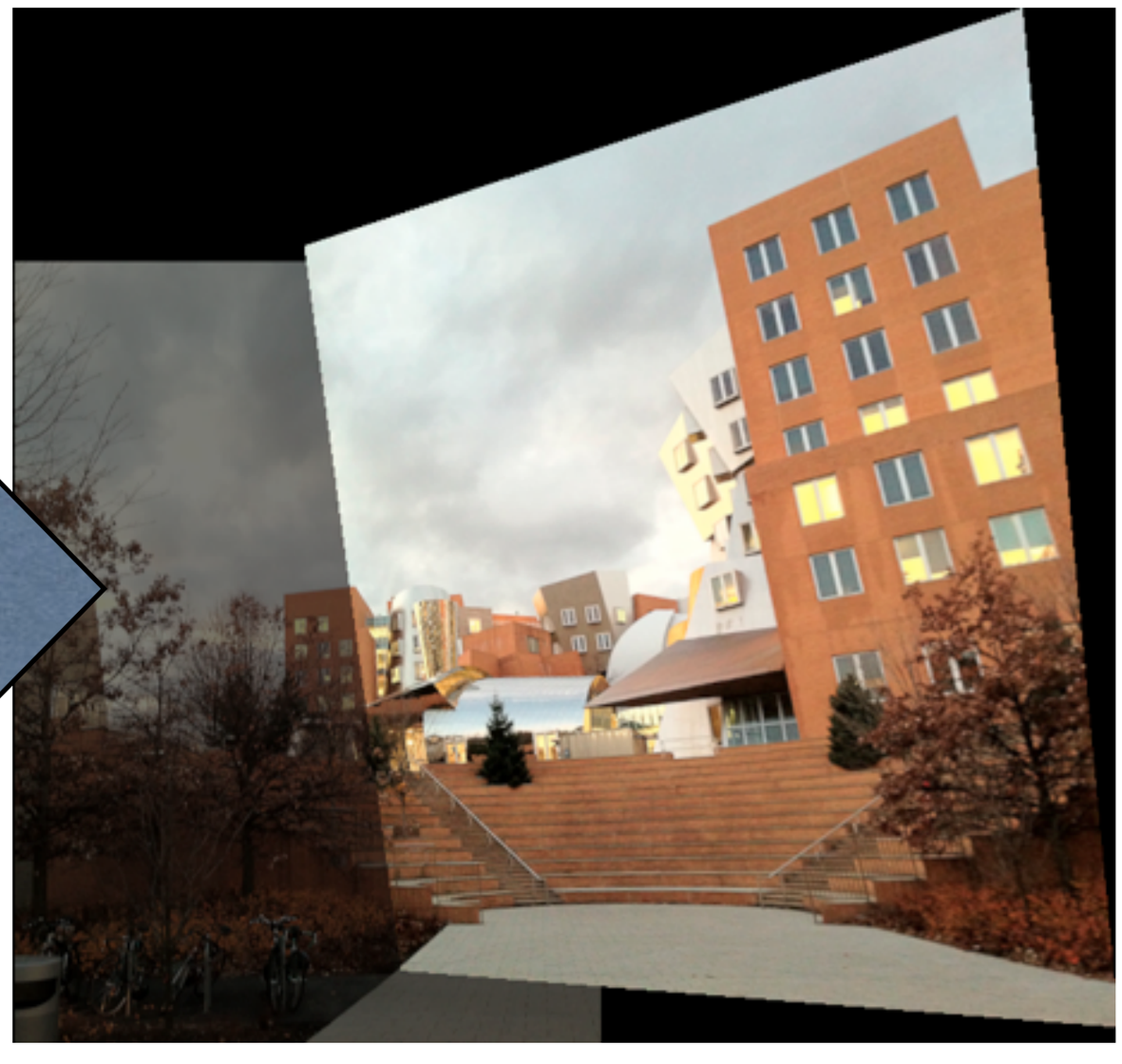


- Makes sense: depth does not matter, all points along a light ray project to the same point.
  - We arbitrarily (but conveniently) choose the point at depth 1

Homographies

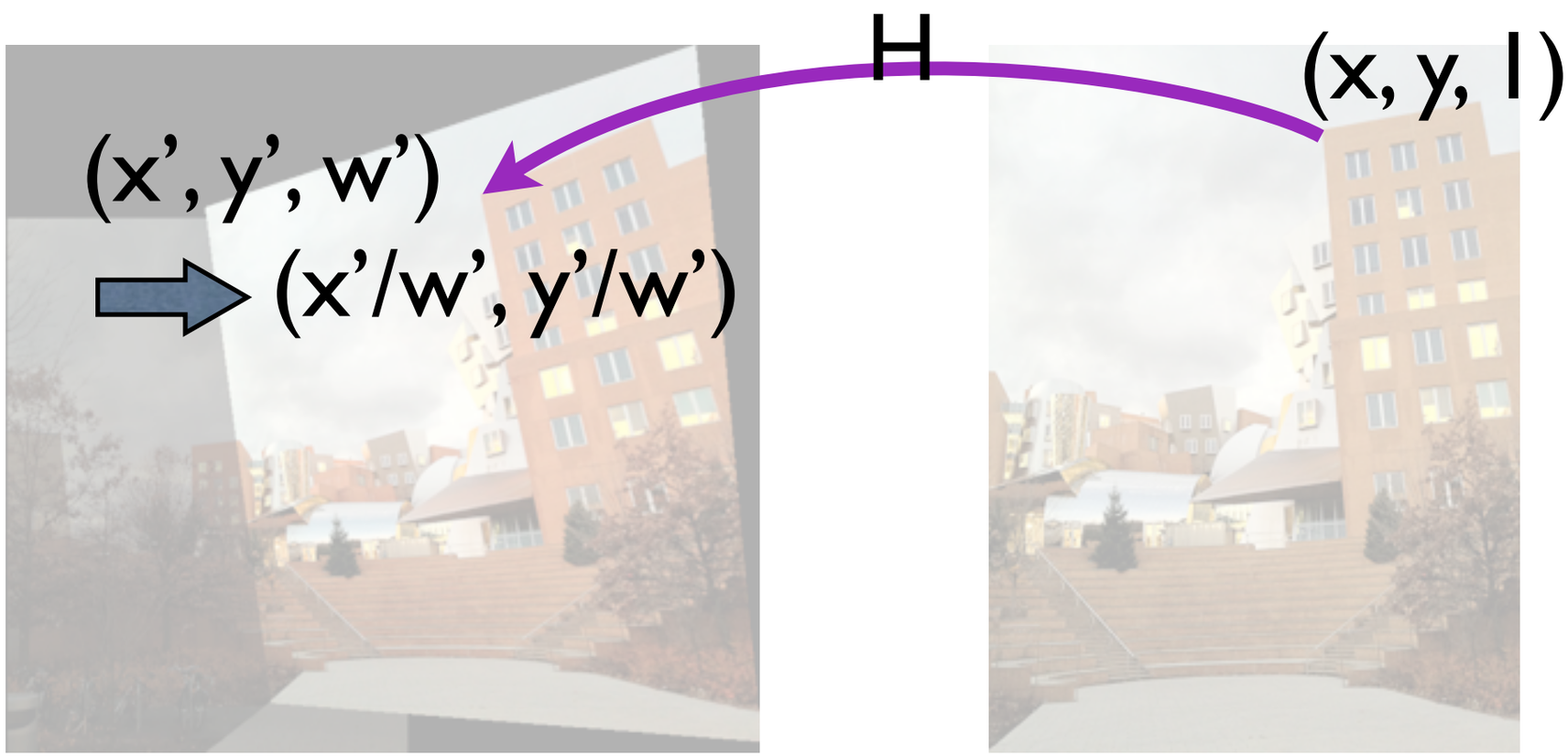
Image Stitching Pipeline

- The user gives us 4 correspondences
- We reproject one image to match the other one
- Creates a wider angle view



2D homogenous version

- 2D points represented as homogenous coordinates (x, y, 1)
- Mapped into the other image by a 3x3 matrix H, into homogenous coordinates (x’, y’, w’)
- get Euclidean coordinates by dividing by w’

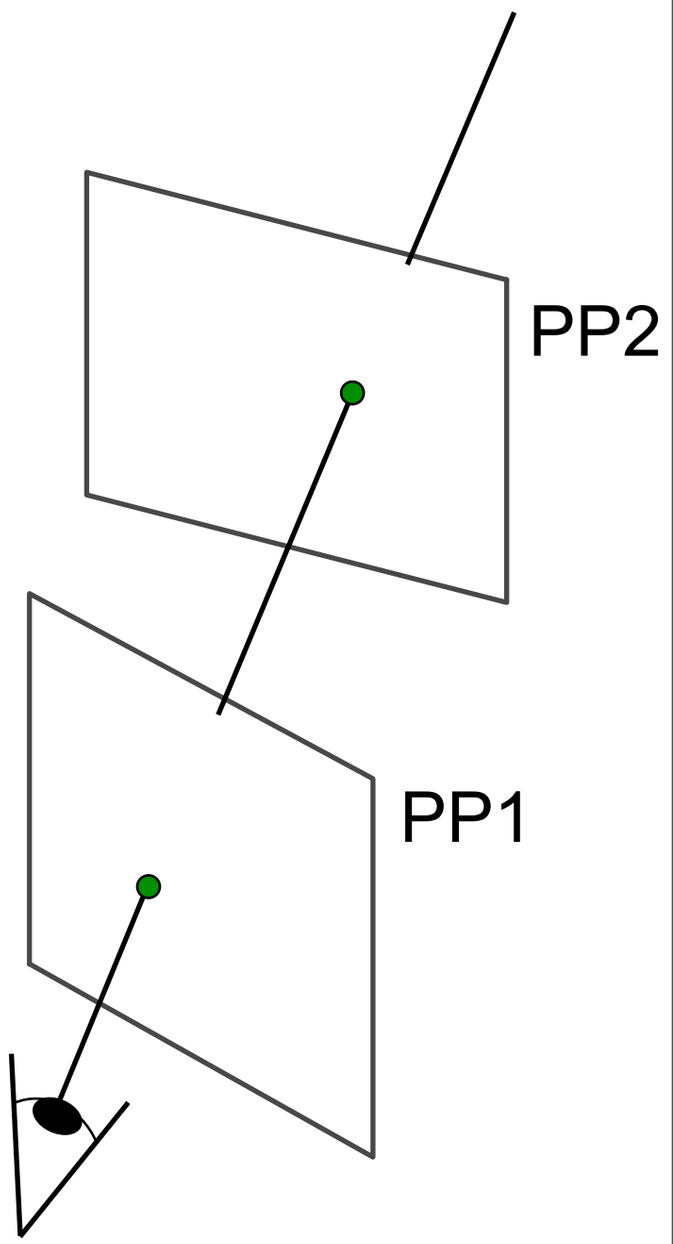


Wrapping it up: Homography

- **Projective mapping between any two planes**
- **represented as 3x3 matrix in homogenous coordinates**
  - corresponds to the 3D rotation

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

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



To apply a homography **H**

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

Scale Invariance

- All points (wx, wy, w) for any non-zero w represent the same (2d) Euclidean point (x,y)

- Essentially, these points all lie on the same ray emanating from the eye

General Pair of Planes

- Homographies can project from any plane to any other plane

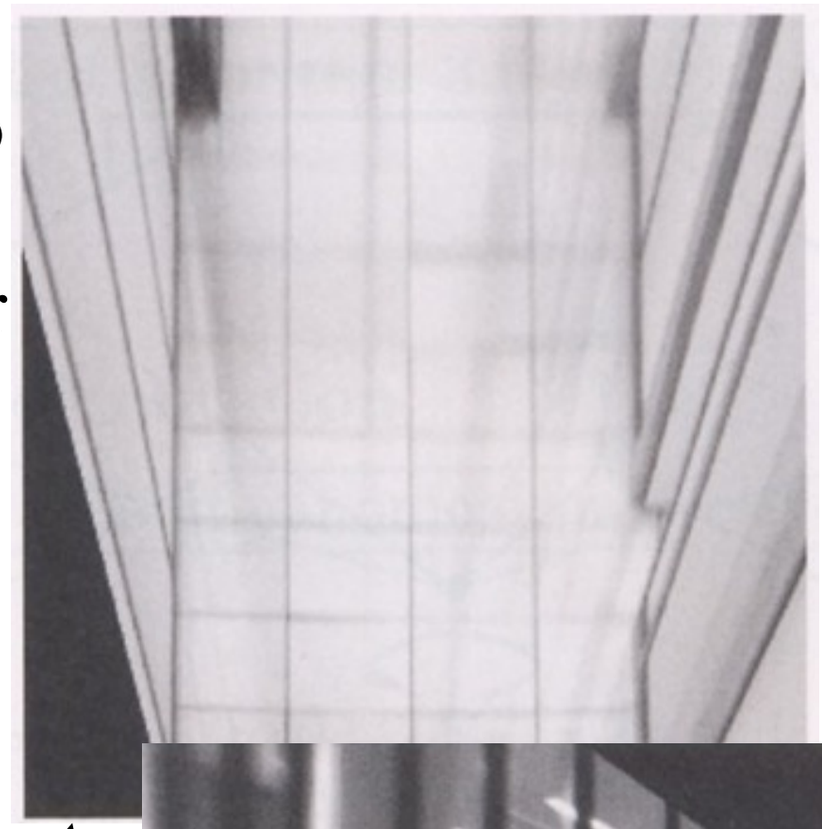
+

Homework
due 9pm

→

Image Warping with Homographies

homography so that image is parallel to floor



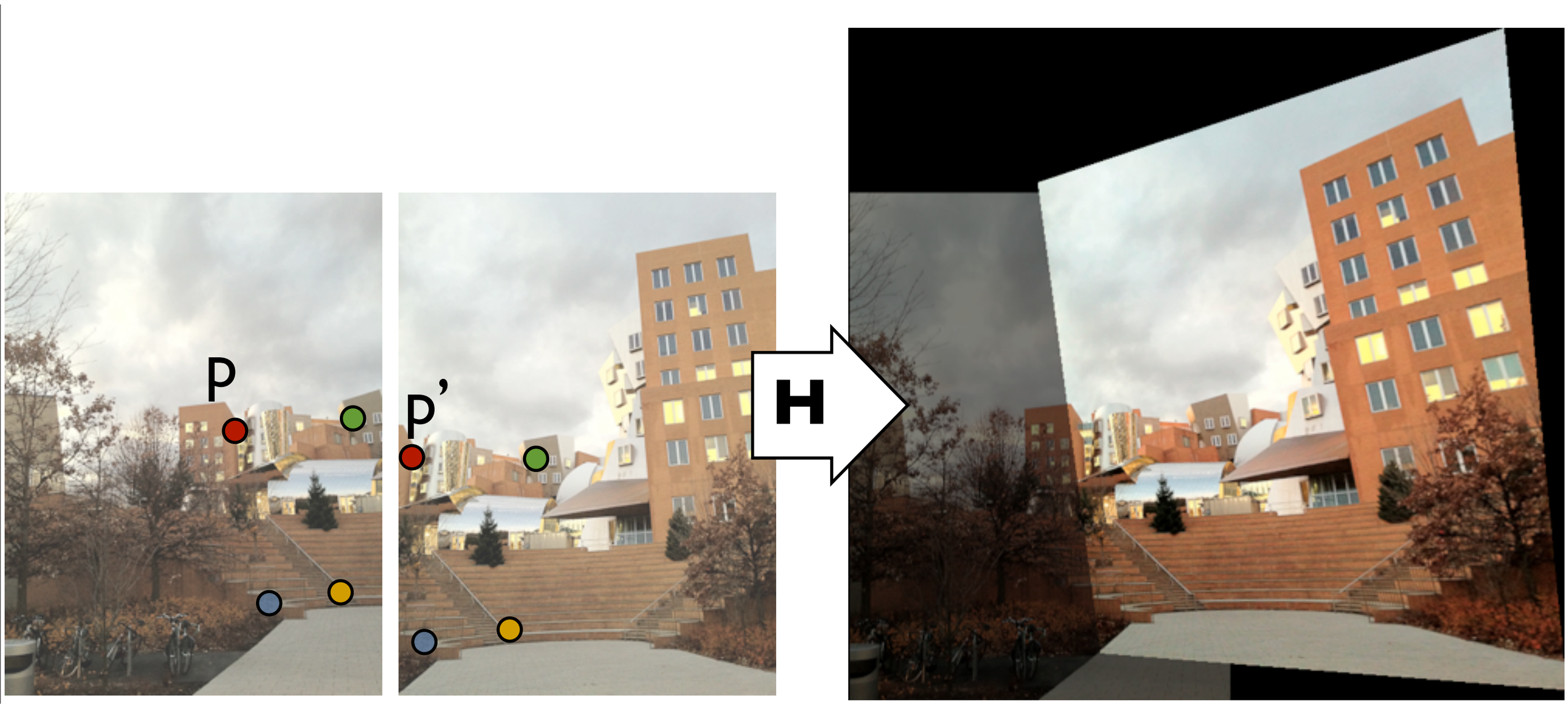
homography so that image is parallel to right wall

black area where no pixel maps to

Solving for Homographies

Goal

- Given correspondences
- Find homography matrix H that maps each p_i to the p_i'



Homography equation

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} y \\ x \\ 1 \end{pmatrix} = \begin{pmatrix} y'w' \\ x'w' \\ w' \end{pmatrix}$$

- We are given pairs of corresponding points
  - x, y, x', y' are known
- Unknowns: matrix coefficients and w'

Homography equation

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} y \\ x \\ 1 \end{pmatrix} = \begin{pmatrix} y'w' \\ x'w' \\ w' \end{pmatrix}$$

- **We are given pairs of corresponding points**
  - x, y, x', y' are known
- **Unknowns: matrix coefficients and w'**
  - but w' is easy to get:

$$w' = gy + hx + i$$

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} y \\ x \\ 1 \end{pmatrix} = \begin{pmatrix} y'w' \\ x'w' \\ w' \end{pmatrix}$$

$$w' = gy + hx + i$$

- **For a pair of points (x, y)->(x',y') we have**

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} y \\ x \\ 1 \end{pmatrix} = \begin{pmatrix} y'w' \\ x'w' \\ w' \end{pmatrix}$$

$$w' = gy + hx + i$$

- **For a pair of points (x, y)->(x',y') we have**

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

- **Unknowns: a, b, c, d, e, f, g, h, i**
  - Linear!

How many pairs?

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} y \\ x \\ 1 \end{pmatrix} = \begin{pmatrix} y'w' \\ x'w' \\ w' \end{pmatrix}$$

- **Each correspondence pair gives us two equations**

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

We can assume i = 1, and just solve for the 8 unknowns!

- **How many unknowns?**
  - 9
  - but H is defined up to scale. Four pairs are enough!

Forming the linear system

- We have 4x2 linear equations in our 8 unknowns
- Represent as a matrix systemAx=B:

$$\begin{pmatrix} & & \\ & A & \\ & & \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ f \\ g \\ h \\ i \end{pmatrix} = B$$

- Now we need to fill matrix A and vector B

Forming the matrix

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

$$\overbrace{\begin{pmatrix} & & & & & & & & \end{pmatrix}}^{a \quad b \quad c \quad d \quad f \quad g \quad h \quad i} \begin{pmatrix} a \\ b \\ c \\ d \\ f \\ g \\ h \\ i \end{pmatrix} = \begin{matrix} \cdot \\ \cdot \\ \\ \\ \cdot \\ \cdot \\ \cdot \end{matrix}$$

Forming the matrix

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

$a\ \ b\ \ c\ \ d\ \ f\ \ g\ \ h\ \ i$

$$\begin{pmatrix} \cdots\cdots \\ y\ \ x\ \ 1\ \ 0\ \ 0\ \ 0\ \text{-}yy'\text{-}xy'\text{-}y' \\ \cdots\cdots \\ \cdots\cdots \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ f \\ g \\ h \\ i \end{pmatrix} = \begin{matrix} \cdot \\ \cdot \\ \cdot \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \end{matrix}$$

Solve for scale invariance

- We know that there exists a full family of solutions kH, for any non-zero k
- i.e., if (a, b, c, d, e, f, g, h, i) is a solution, so is (ka, kb, kc, kd, ke, kf, kg, kh, ki)

- Adding more correspondences won’t help

How to Compute?

- Use Matrix class (for n x n matrices)
- Call Matrix.inv()
    - Computes with an SVD
- Multiply by the B vector, produces an 8x1 vector.

Extensions

Julian Beever: Manual Homographies

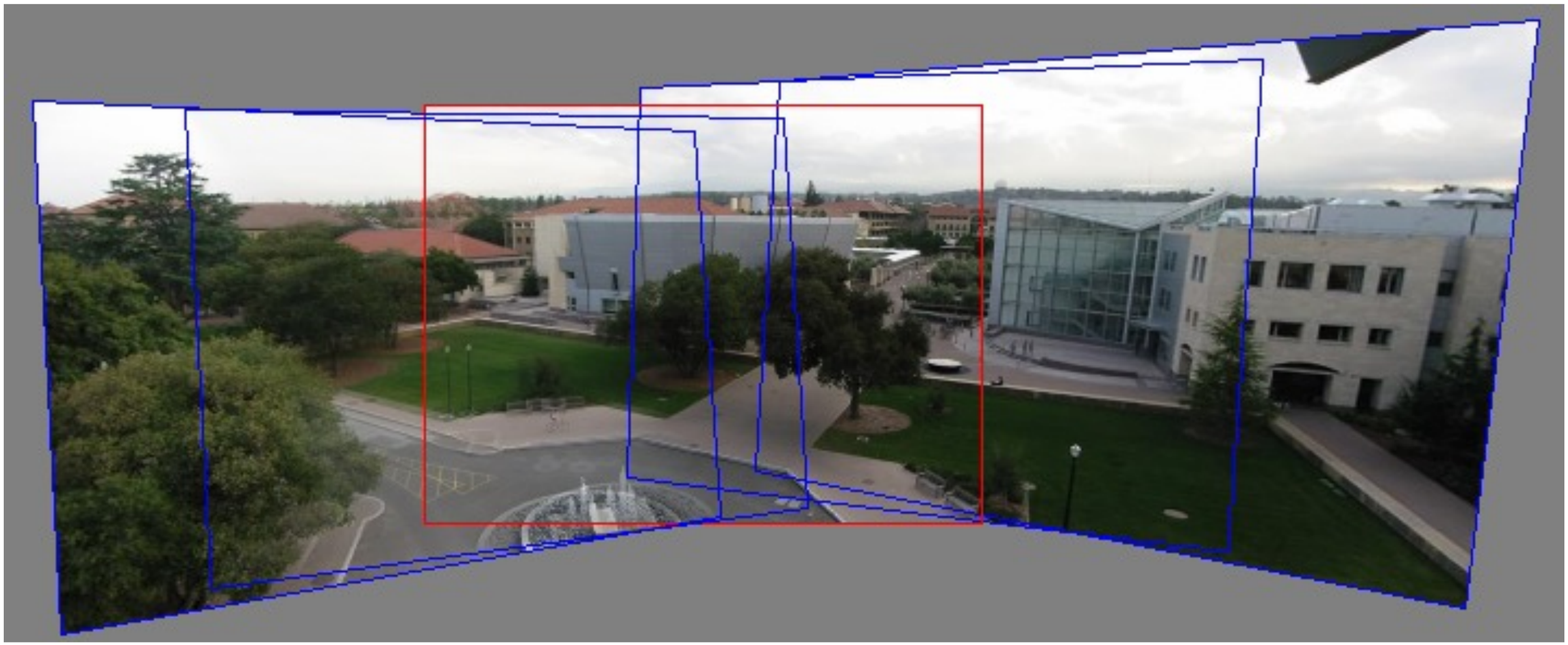
http://www.julianbeever.net/

Other Examples in Art

- The Ambassadors, Hans Holbein the Younger, 1533

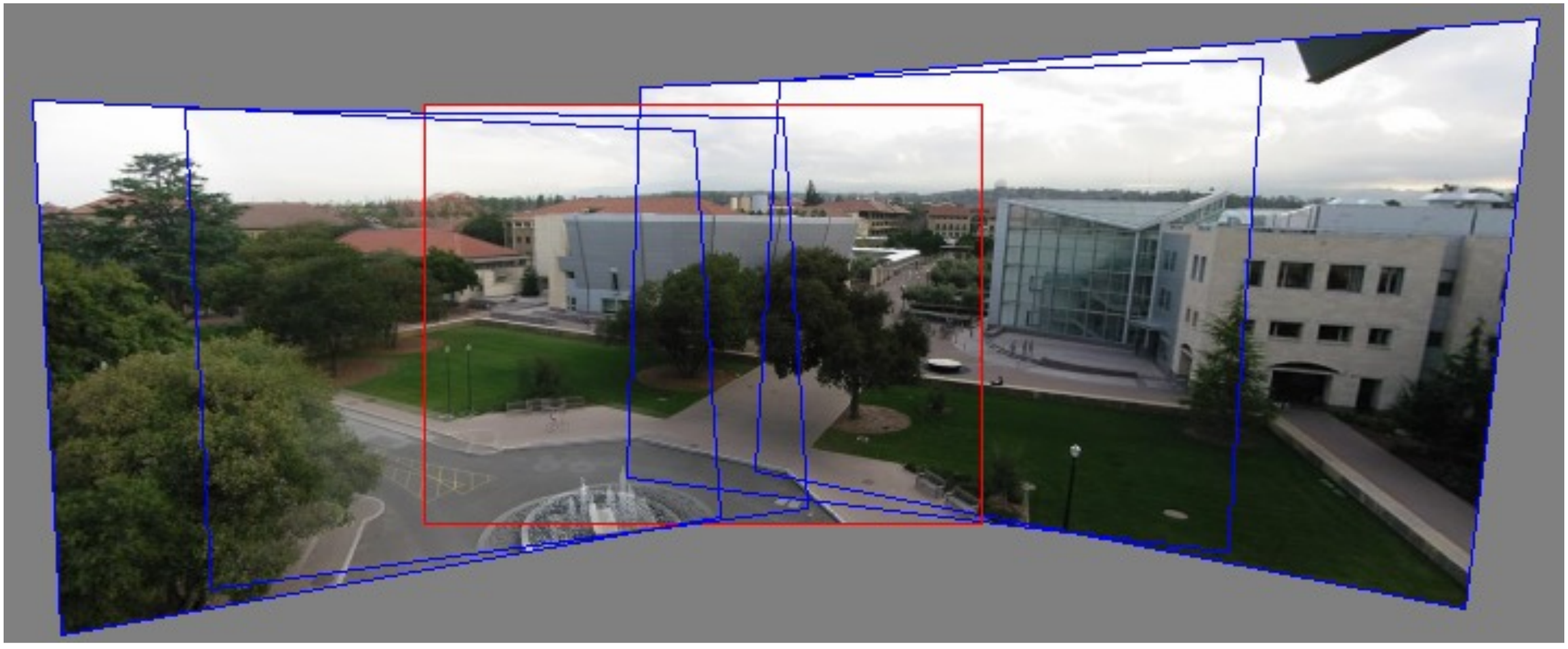
http://en.wikipedia.org/wiki/The_Ambassadors_%28Holbein%29

Multiple Images



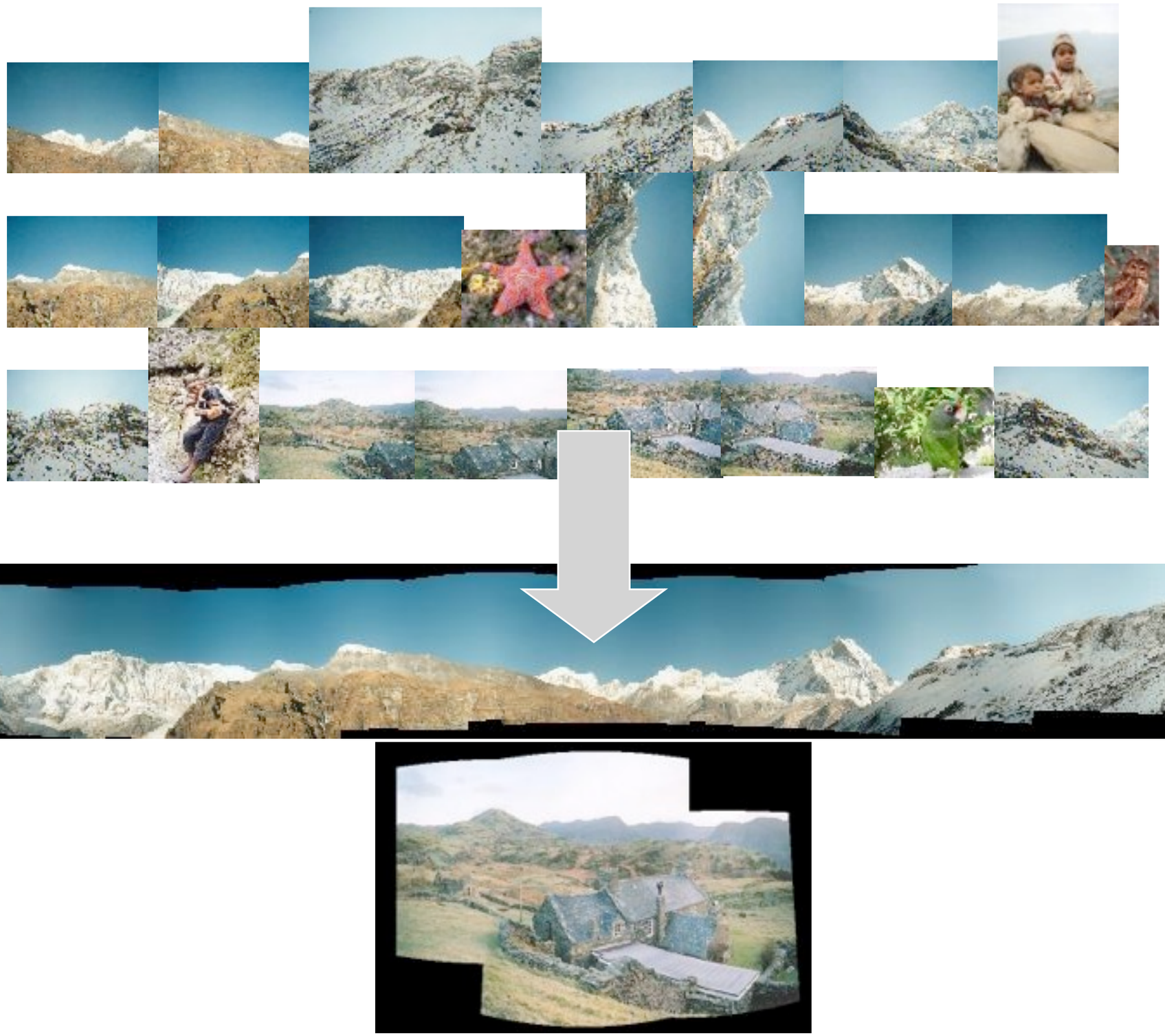
1. **Pick one image (red)**
2. **Warp the other images towards it (usually, one by one)**
3. **blend**

Computing the Warp



- **For each output pixel**
- **compute input location with homography matrix**
  - p’=Hp, followed by division
- **copy pixel color (with appropriate antialiasing)**

We Won’t Get to in this Class: Automatic Panoramas



http://research.microsoft.com/~brown/papers/iccv2003.pdf

We Won’t Get to in this Class: Panorama Weaving / Seam Editing

Figure 1: *Panorama Weaving on a challenging data-set (Nation, 12848 x 3821, 9 images) with moving objects during acquisition, registration issues and varying exposure. Our initial automatic solution (bottom, left) was computed in 4.6 seconds at full resolution for a result with lower seam energy than Graph Cuts. Additionally, we present a system for the interactive user exploration of the seam solution space (bottom, right), easily enabling:* (a) *the resolution of moving objects,* (b) *the hiding of registration artifacts (split pole) in low contrast areas (scooter) or* (c) *the fix of semantic notions for which automatic decisions can be unsatisfactory (stoplight colors are inconsistent after the automatic solve). The user editing session took only a few minutes. (top) the final, color-corrected panorama.*

http://www.sci.utah.edu/~bsumma/projects/weaving/index.html

We Won’t Get to in this Class:
Poisson Image Editing / Gradient Blending / Color Correction

(a) (b) (c)

Figure 9.18 Poisson image editing (Pérez, Gangnet, and Blake 2003) © 2003 ACM: (a) The dog and the two children are chosen as source images to be pasted into the destination swimming pool. (b) Simple pasting fails to match the colors at the boundaries, whereas (c) Poisson image blending masks these differences.

See Szeliski, Ch.9

We Won't Get to in this Class: View Morphing

- **Seitz & Dyer**

  http://www.cs.washington.edu/homes/seitz/vmorph/vmorph.htm

- **Interpolation consistent with 3D view interpolation**

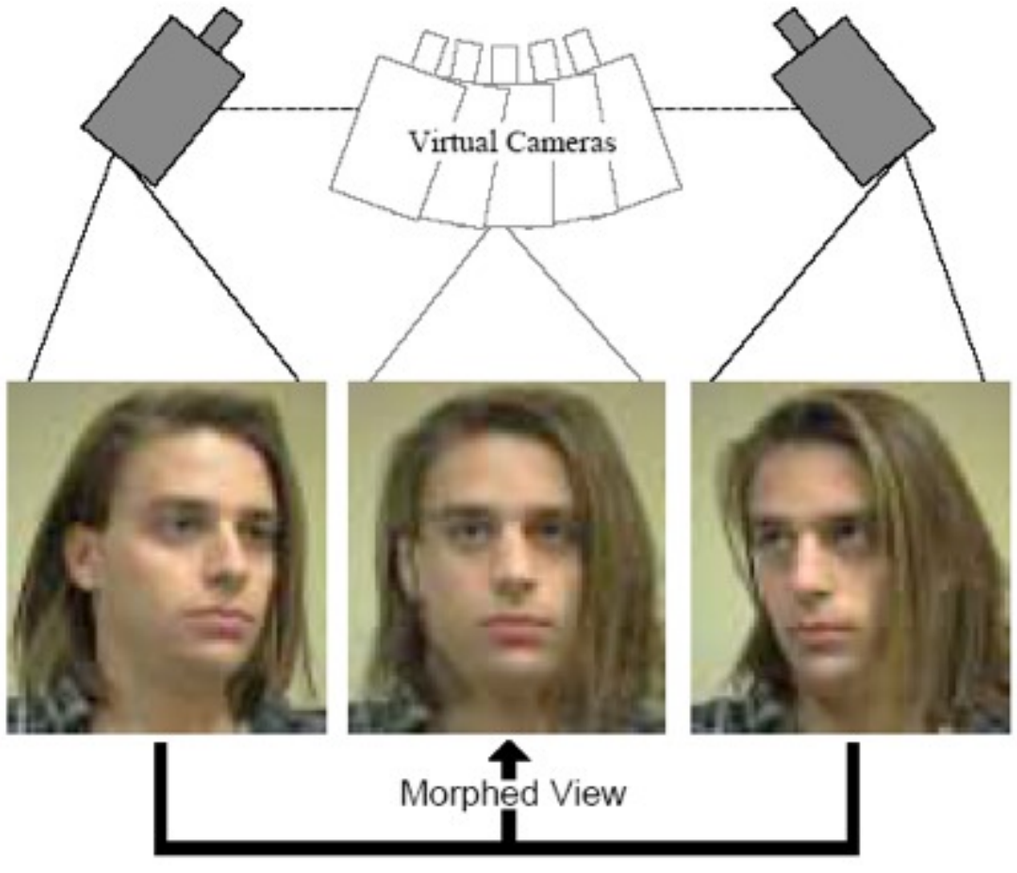


Figure 1: View morphing between two images of an object taken from two different viewpoints produces the illusion of physically moving a virtual camera.

We Won’t Get to in this Class: View Morphing

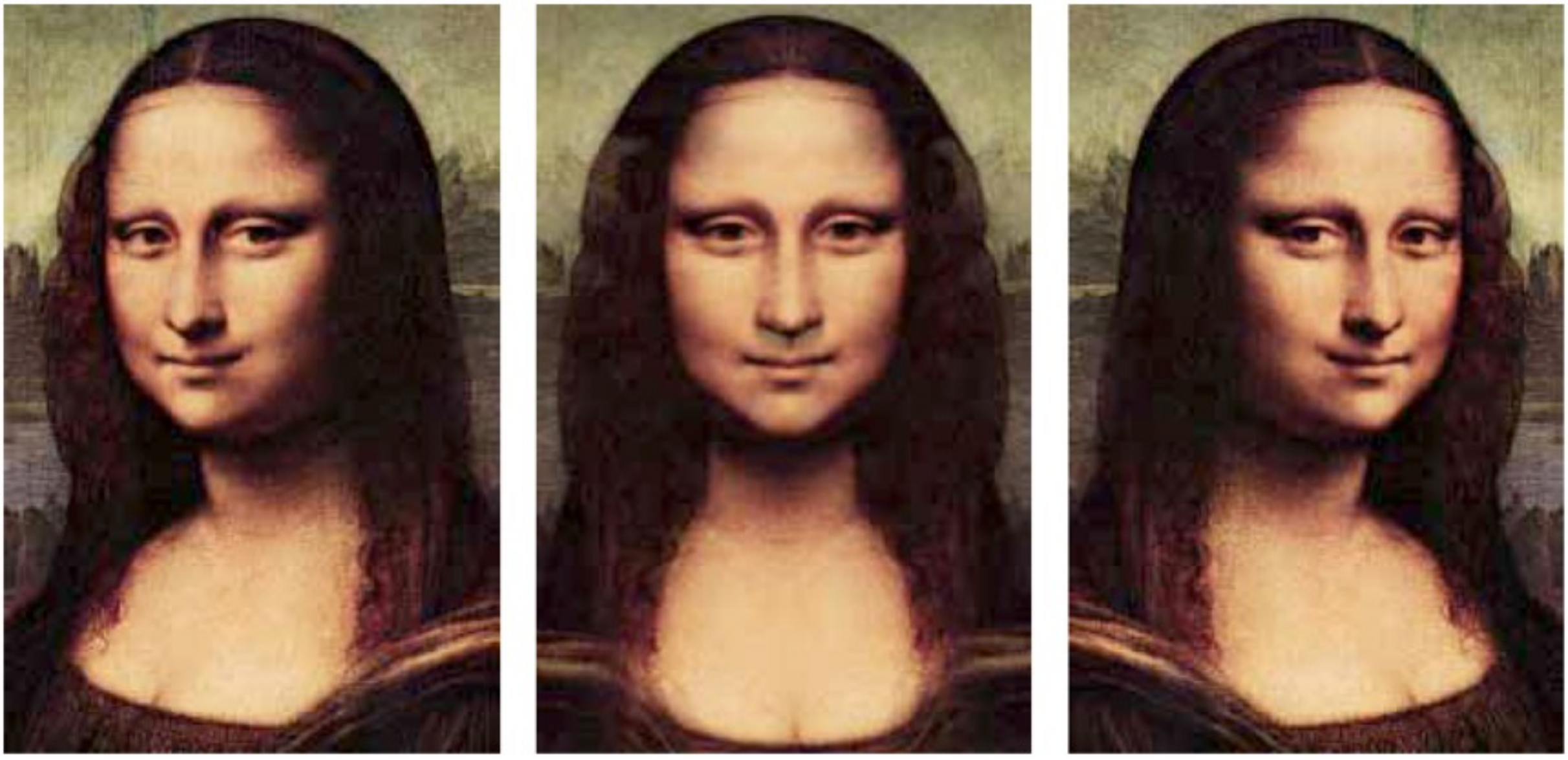


Figure 9: Mona Lisa View Morph. Morphed view (center) is halfway between original image (left) and it’s reflection (right).

http://homes.cs.washington.edu/~seitz/vmorph/vmorph.htm

Lec23 Required Reading

- Hunt, Ch. 9
- House, Ch. 14