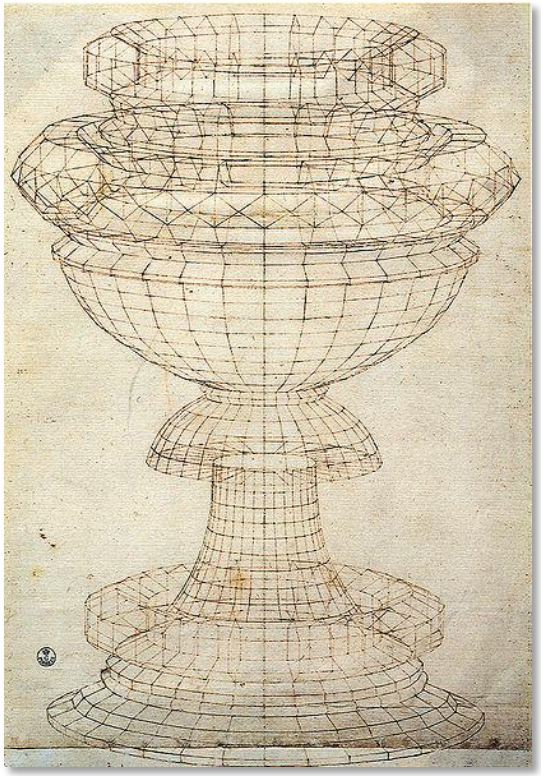
CS4670: Computer Vision

Noah Snavely

Lecture 13: Projection, Part 2



Perspective study of a vase by Paolo Uccello

Reading

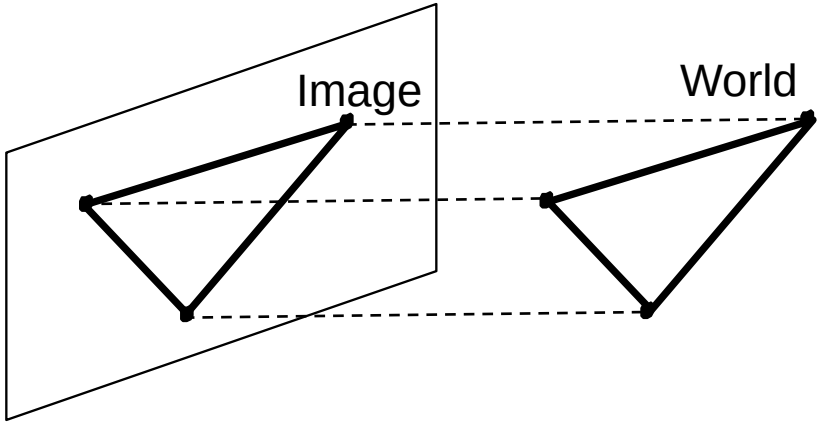
- Szeliski 2.1.3-2.1.6

Announcements

- Project 2a due Friday, 8:59pm
- Project 2b out Friday
- Take-home prelim after Fall break

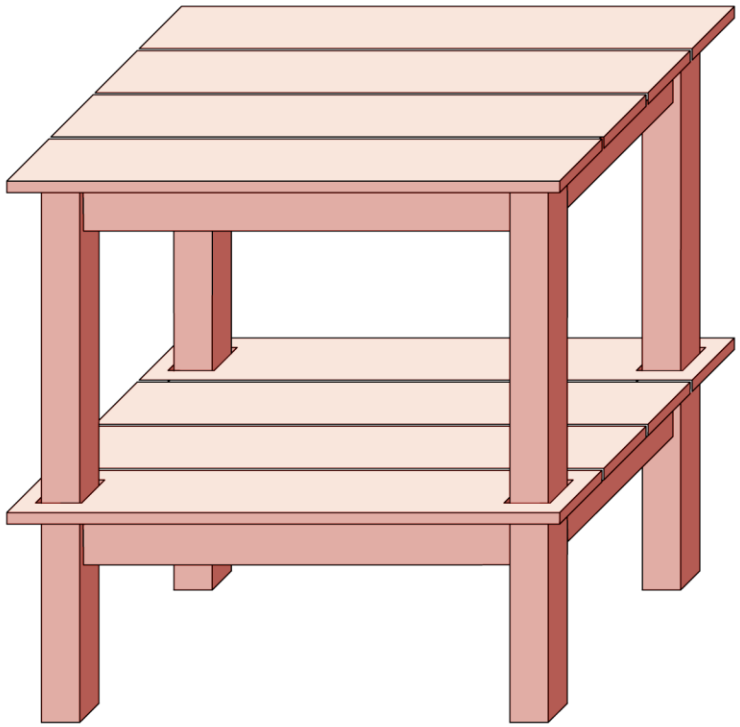
Orthographic projection

- Special case of perspective projection
  - Distance from the COP to the PP is infinite

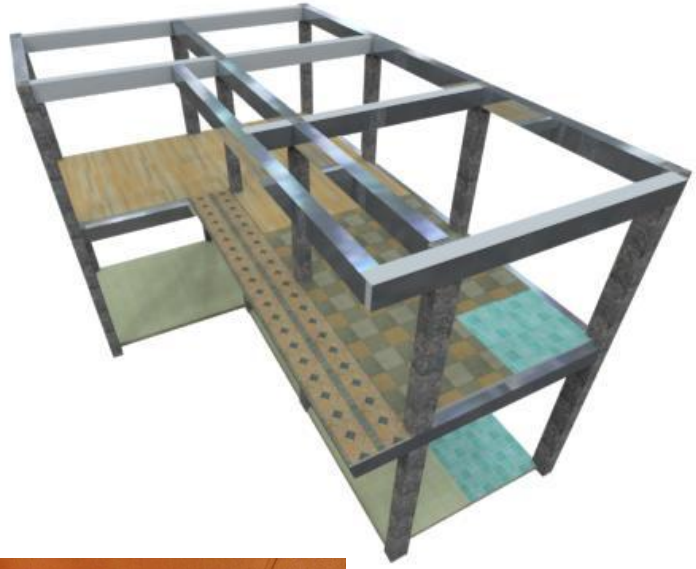


$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow (x, y)$$

Orthographic projection

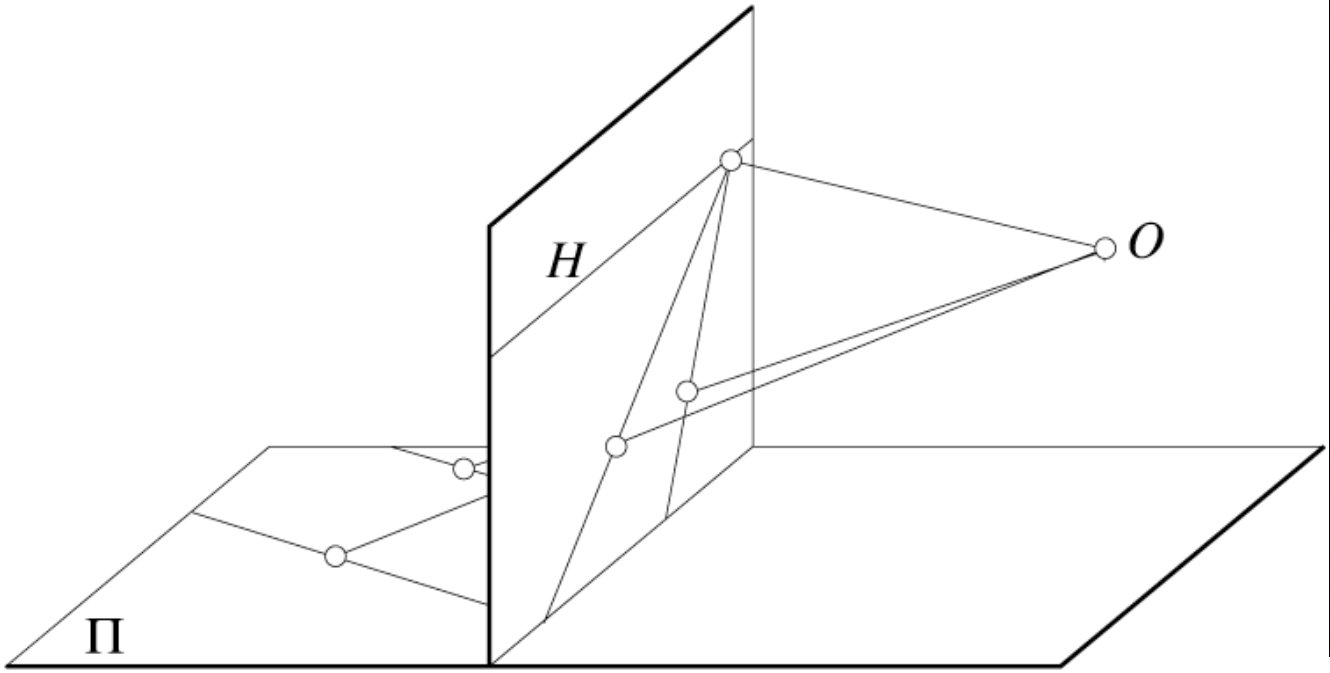


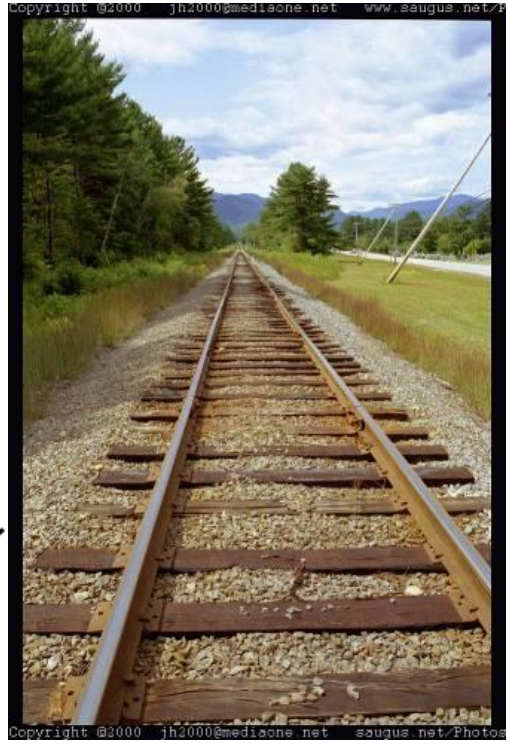
Perspective projection



Projection properties

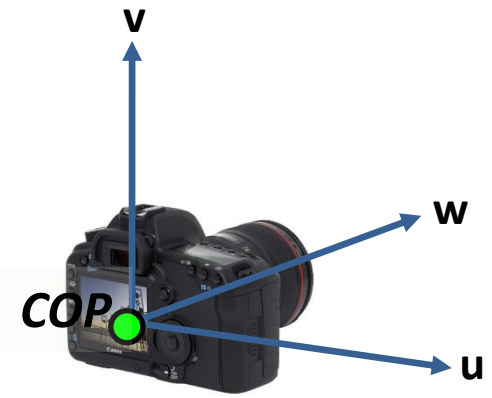
- Parallel lines converge at a vanishing point
  - Each direction in space has its own vanishing point
  - But parallels parallel to the image plane remain parallel





Camera parameters

- How can we model the geometry of a camera?



Camera

Three important coordinate systems:

1. *World* coordinate system (3D)
2. *Camera* coordinate system (3D)
3. *Image* coordinate system (2D)

"The World"

We will define a projection matrix that maps from 1 → 3 (via 2)

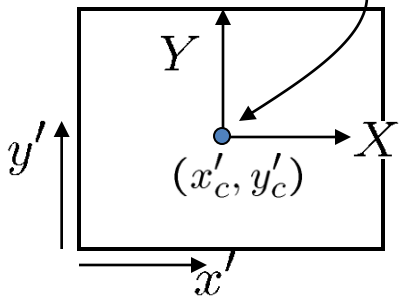
Camera parameters

- To project a point (*x*,*y*,*z*) in *world* coordinates into a camera
- First transform (*x*,*y*,*z*) into *camera* coordinates (*u*,*v*,*w*)
  - Need to know
    - Camera position (in world coordinates)
    - Camera orientation (in world coordinates)
- Then project into the image plane
  - Need to know camera *intrinsics*

Camera parameters

A camera is described by several parameters

- Translation T of the optical center from the origin of world coords
- Rotation R of the image plane
- focal length f, principle point (x'_c, y'_c), pixel size (s_x, s_y)
- blue parameters are called "extrinsics," red are "intrinsics"



Projection equation

$$\mathbf{x} = \begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{\Pi X}$$

- The projection matrix models the cumulative effect of all parameters
- Useful to decompose into a series of operations

identity matrix

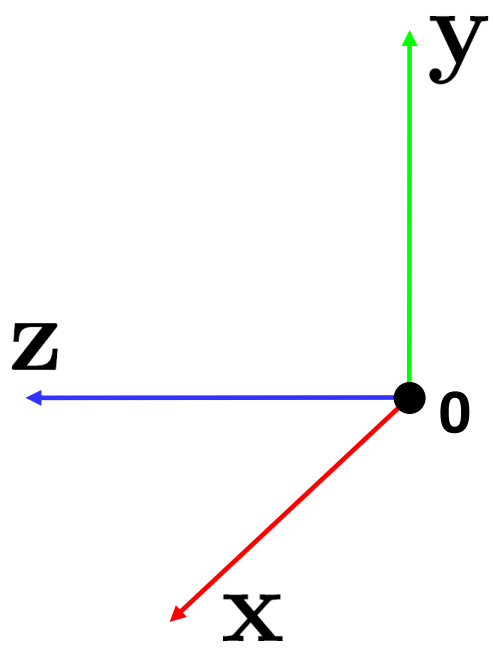
$$\mathbf{\Pi} = \begin{bmatrix} -fs_x & 0 & x'_c \\ 0 & -fs_y & y'_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{0}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{T}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}$$

intrinsics projection rotation translation

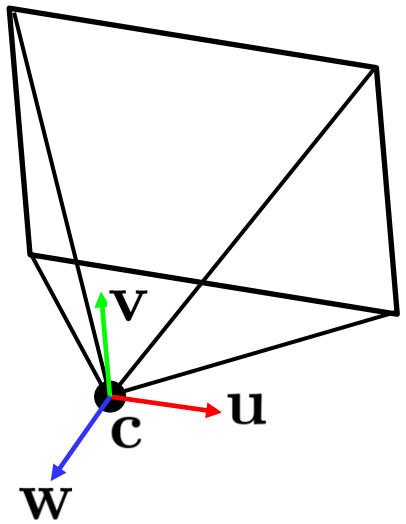
- The definitions of these parameters are **not** completely standardized
  - especially intrinsics—varies from one book to another

Extrinsics

- How do we get the camera to “canonical form”?
  - (Center of projection at the origin, x-axis points right, y-axis points up, z-axis points backwards)

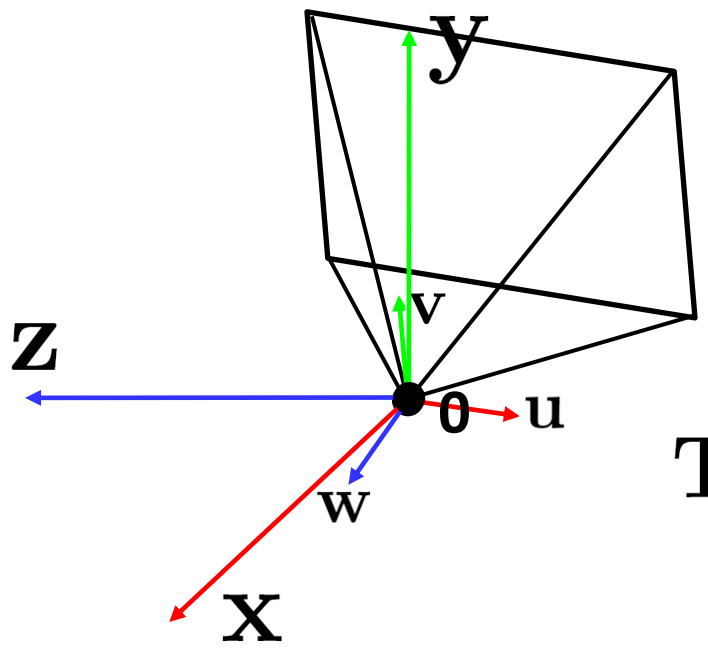


Step 1: Translate by -**c**



Extrinsics

- How do we get the camera to “canonical form”?
  - (Center of projection at the origin, x-axis points right, y-axis points up, z-axis points backwards)



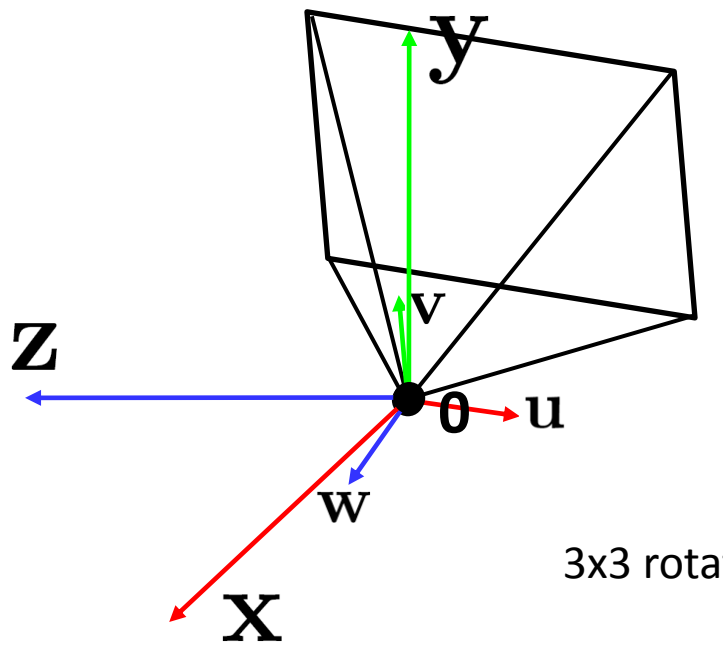
Step 1: Translate by -**c**

How do we represent translation as a matrix multiplication?

$$\mathbf{T} = \begin{bmatrix} \mathbf{I}_{3\times 3} & -\mathbf{c} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}$$

Extrinsics

- How do we get the camera to “canonical form”?
  - (Center of projection at the origin, x-axis points right, y-axis points up, z-axis points backwards)



Step 1: Translate by -**c**

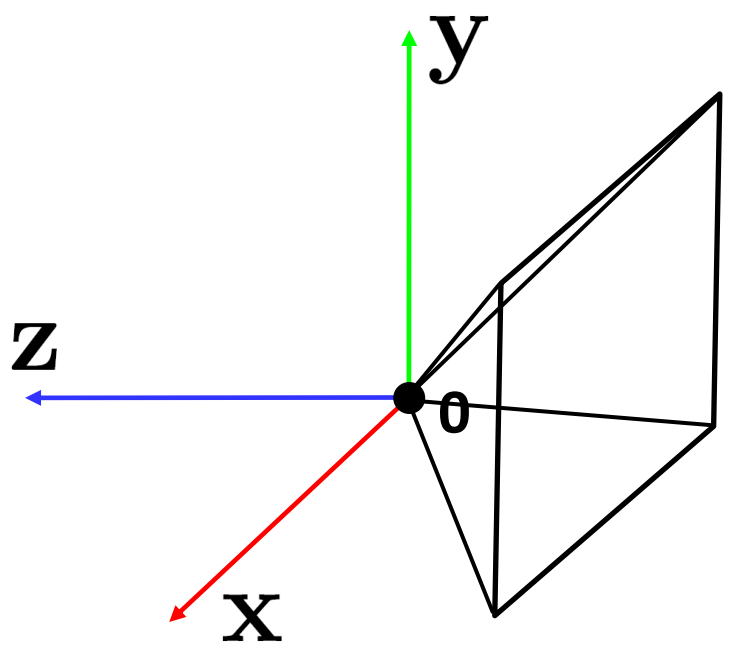
Step 2: Rotate by **R**

$$\mathbf{R} = \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \\ \mathbf{w}^T \end{bmatrix}$$

3x3 rotation matrix

Extrinsics

- How do we get the camera to “canonical form”?
  - (Center of projection at the origin, x-axis points right, y-axis points up, z-axis points backwards)



Step 1: Translate by -**c**

Step 2: Rotate by **R**

$$\mathbf{R} = \begin{bmatrix} \mathbf{u}^T \\ \mathbf{v}^T \\ \mathbf{w}^T \end{bmatrix}$$

Perspective projection

$$\underbrace{\begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

(intrinsics)

(converts from 3D rays in camera coordinate system to pixel coordinates)

in general, $\mathbf{K} = \begin{bmatrix} -f & s & c_x \\ 0 & -\alpha f & c_y \\ 0 & 0 & 1 \end{bmatrix}$ (upper triangular matrix)

α : **aspect ratio** (1 unless pixels are not square)

s : **skew** (0 unless pixels are shaped like rhombi/parallelograms)

(c_x, c_y) : **principal point** ((0,0) unless optical axis doesn't intersect projection plane at origin)

Focal length

- Can think of as “zoom”

24mm

50mm

200mm

800mm

- Also related to *field of view*

Projection matrix

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

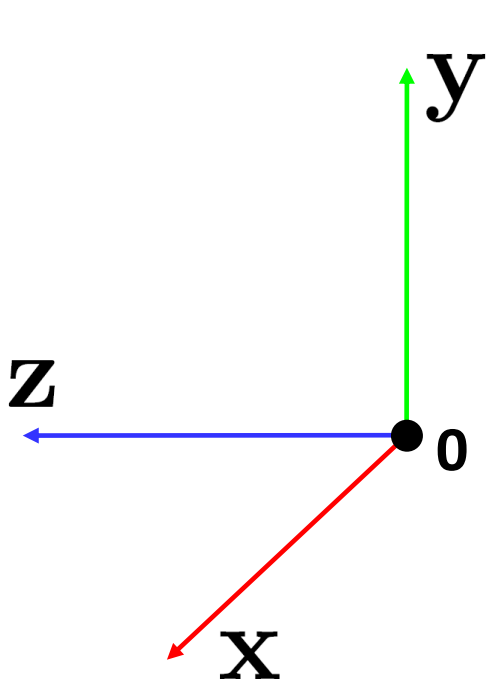
intrinsics

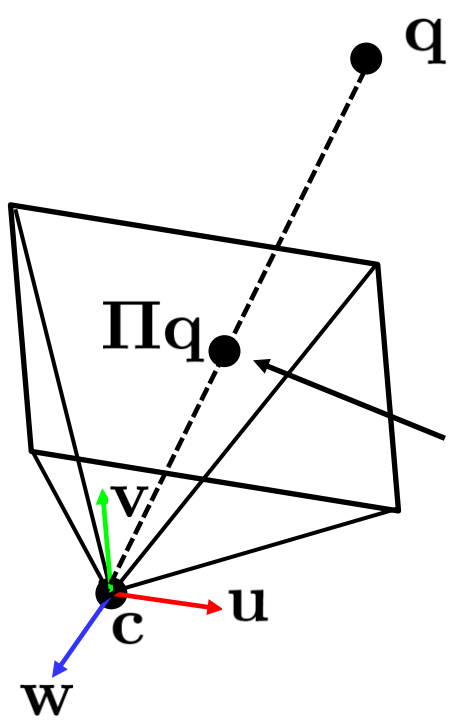
$$\begin{bmatrix} \mathbf{R} & \underbrace{-\mathbf{Rc}}_{\mathbf{t}} \end{bmatrix}$$

(**t** in book's notation)

$$\Pi = \mathbf{K} \begin{bmatrix} \mathbf{R} & -\mathbf{Rc} \end{bmatrix}$$

Projection matrix





(in homogeneous image coordinates)

Questions?

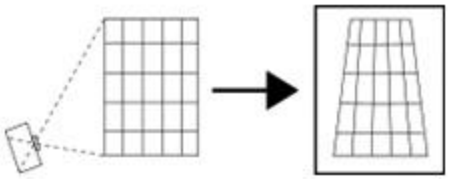
Perspective distortion

- Problem for architectural photography: converging verticals

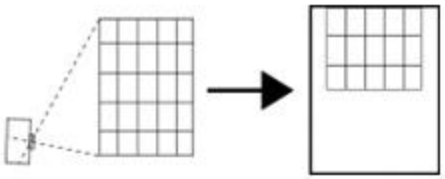
Source: F. Durand

Perspective distortion

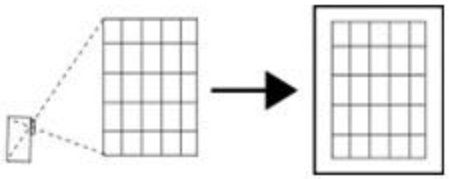
- Problem for architectural photography: converging verticals



Tilting the camera upwards results in converging verticals

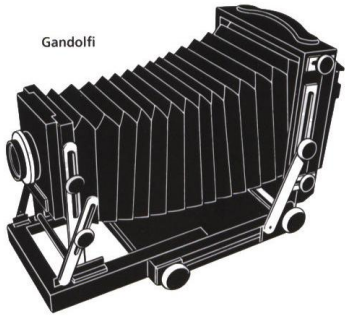


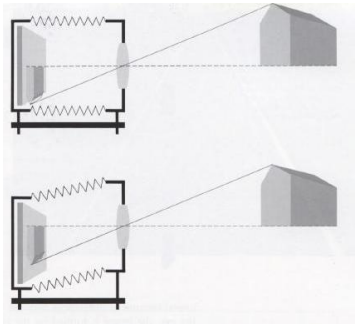
Keeping the camera level, with an ordinary lens, captures only the bottom portion of the building



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

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



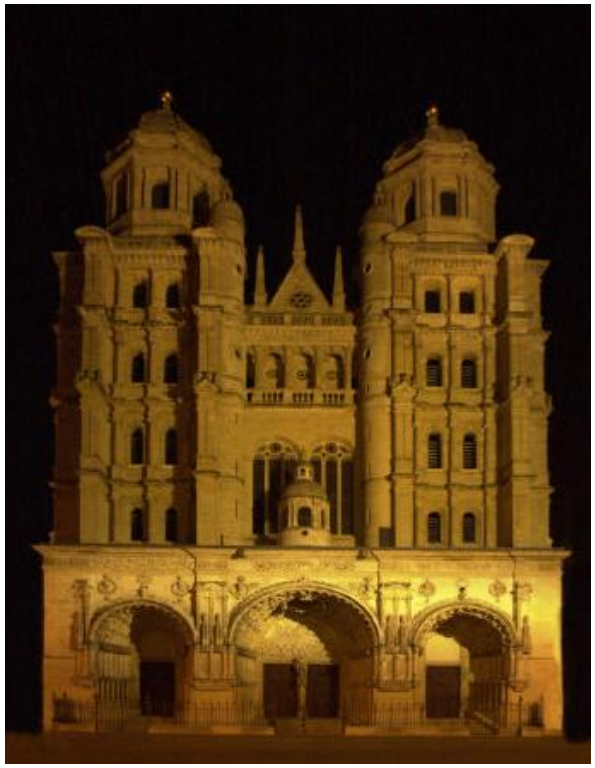


http://en.wikipedia.org/wiki/Perspective_correction_lens

Source: F. Durand

Perspective distortion

- Problem for architectural photography: converging verticals
- Result:



Source: F. Durand

Perspective distortion

- What does a sphere project to?

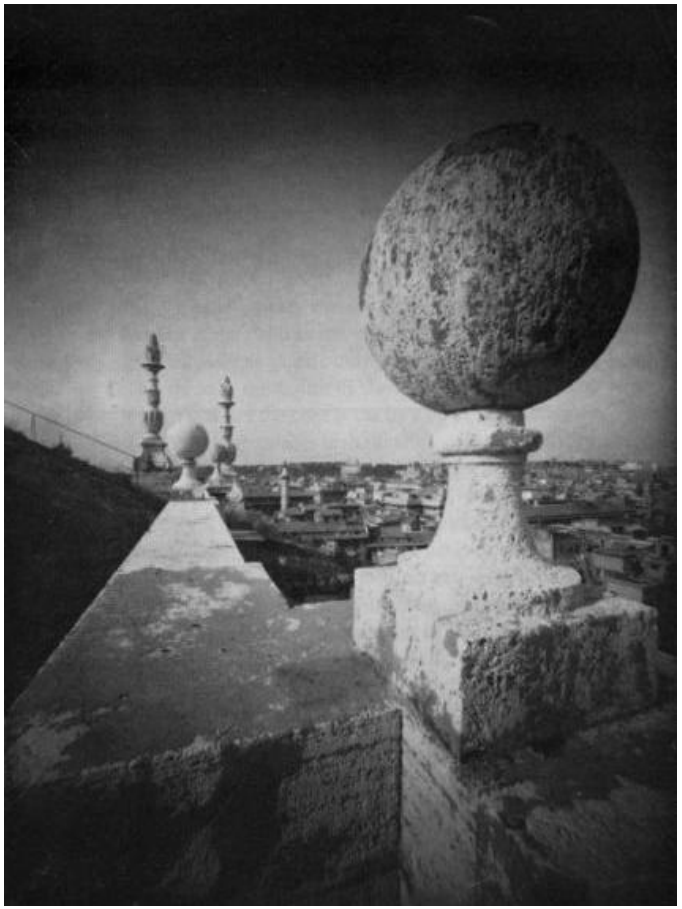
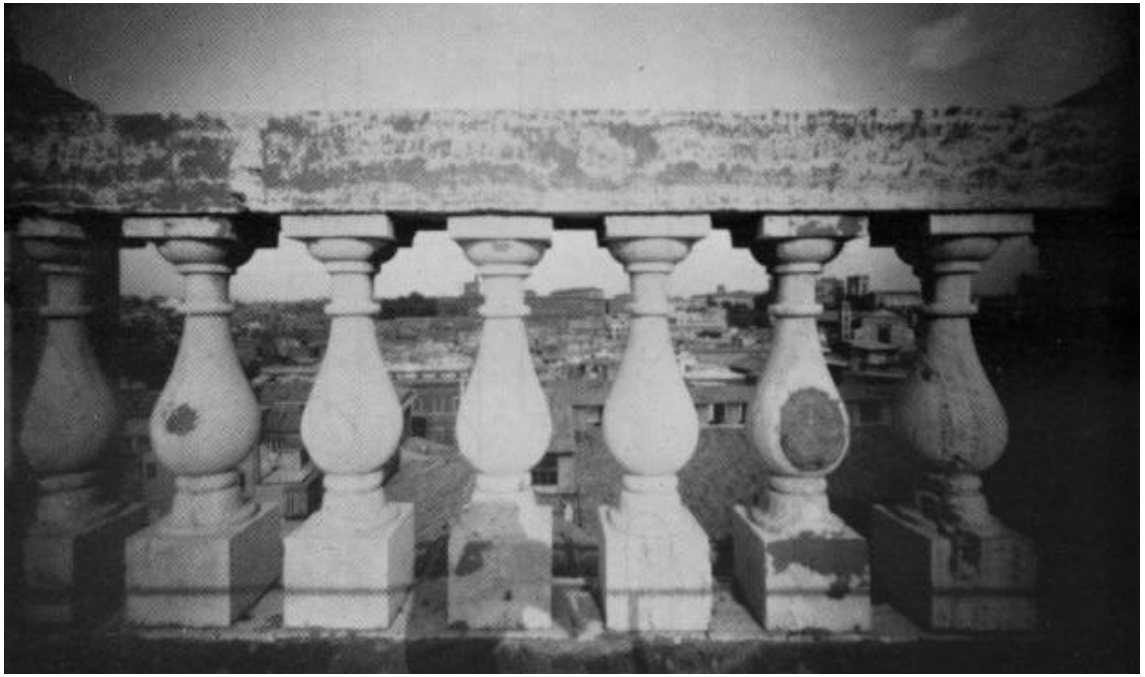
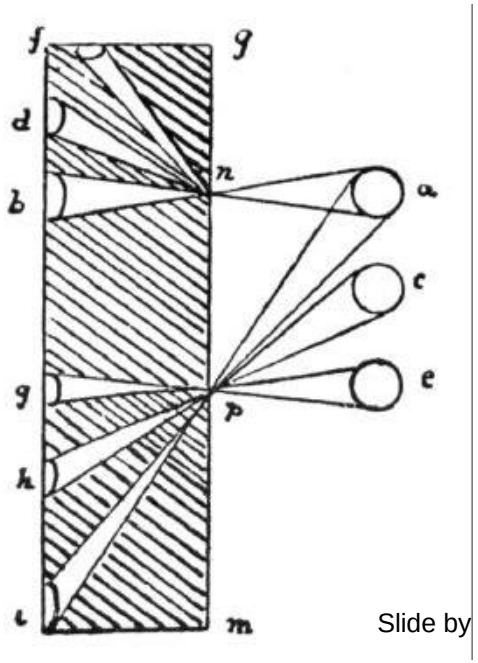


Image source: F. Durand

Perspective distortion

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

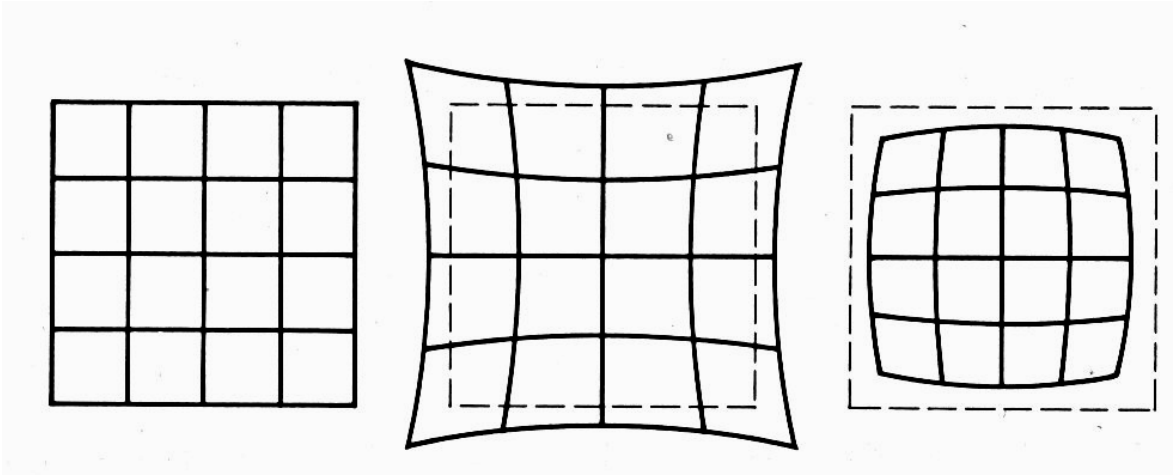




Slide by F. Durand

Perspective distortion: People

Distortion



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

Correcting radial distortion

from Helmut Dersch

Other types of projection

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

360 degree field of view…

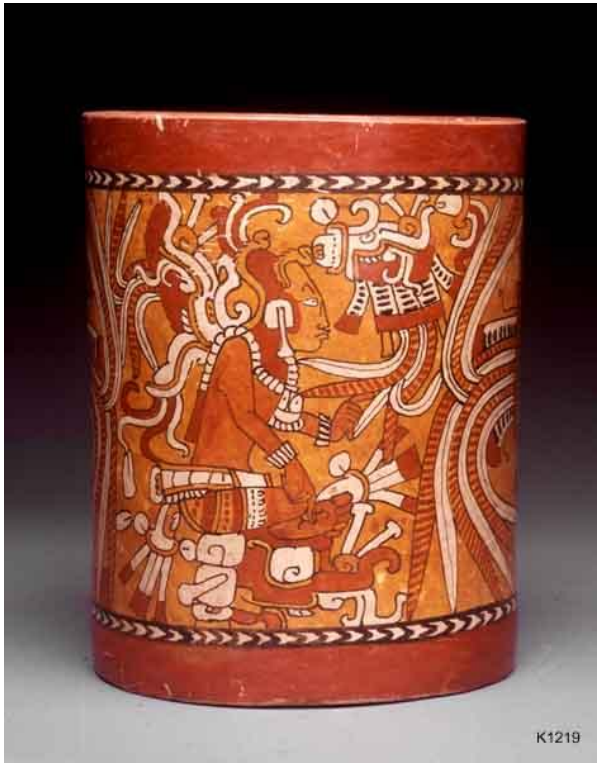
- Basic approach
  - Take a photo of a parabolic mirror with an orthographic lens (Nayar)
  - Or buy one a lens from a variety of omnicam manufacturers…
    - See http://www.cis.upenn.edu/~kostas/omni.html

Tilt-shift

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

Titlt-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”

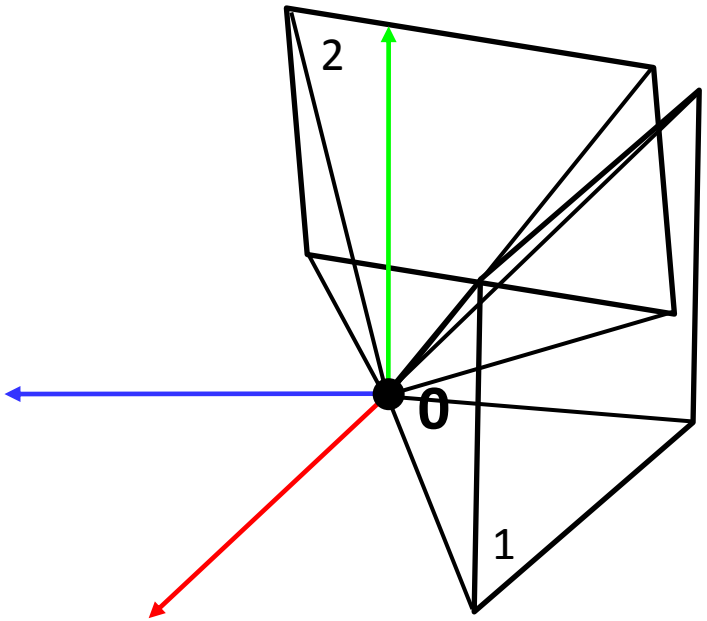
Back to mosaics

- How to we align the images?

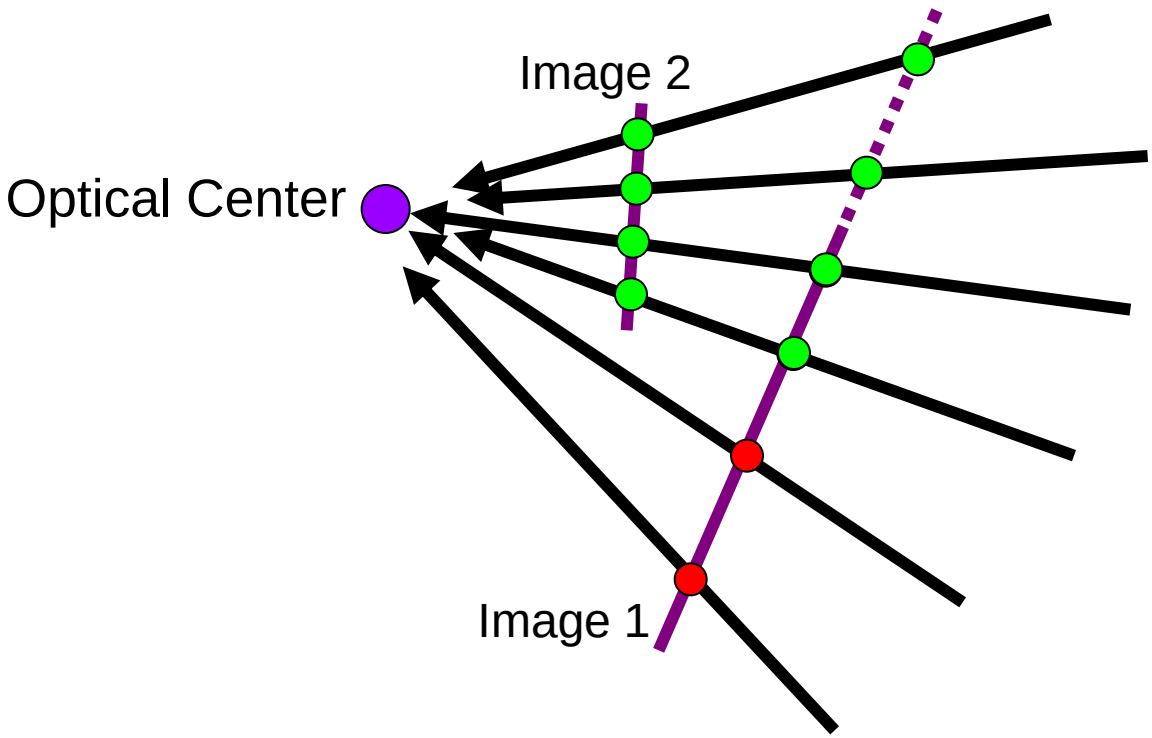
Creating a panorama

- 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



Geometric Interpretation of Mosaics



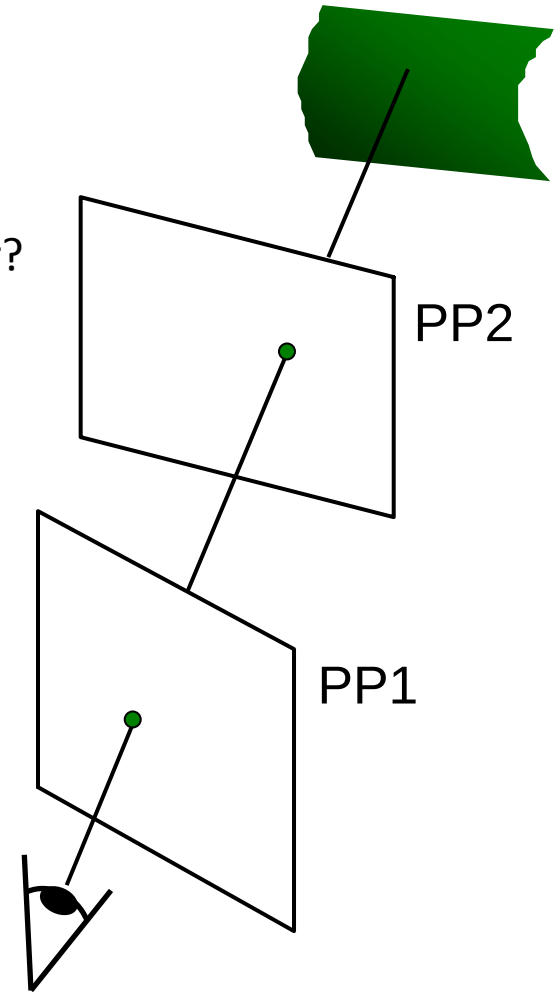
- If we capture all 360º of rays, we can create a 360º panorama
- The basic operation is *projecting* an image from one plane to another
- The projective transformation is scene-INDEPENDENT
  - This depends on all the images having the same optical center

Image reprojection

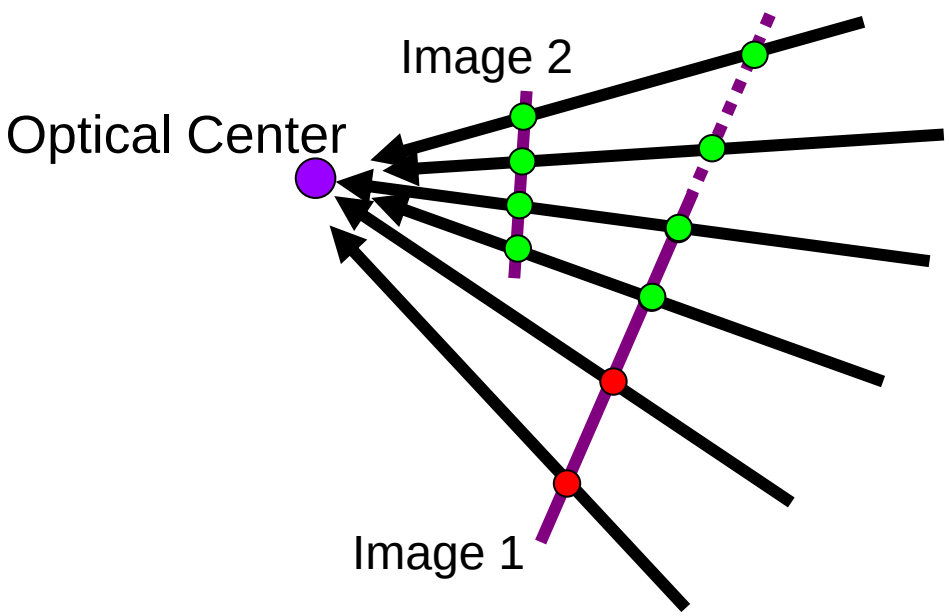
- Basic question
  - How to relate two images from the same camera center?
    - how to map a pixel from PP1 to PP2

Answer

- Cast a ray through each pixel in PP1
- Draw the pixel where that ray intersects PP2



What is the transformation?



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

3D ray coords (in camera 2)

image coords (in image 2)

$$\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} = \mathbf{R}_1 \mathbf{R}_2^{\mathrm{T}} \mathbf{K}_2^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

3D ray coords (in camera 1)

image coords (in image 2)

How do we transform image 2 onto image 1's projection plane?

image 1	image 2
$\mathbf{K}_1$	$\mathbf{K}_2$
$\mathbf{R}_1 = \mathbf{I}_{3\times 3}$	$\mathbf{R}_2$

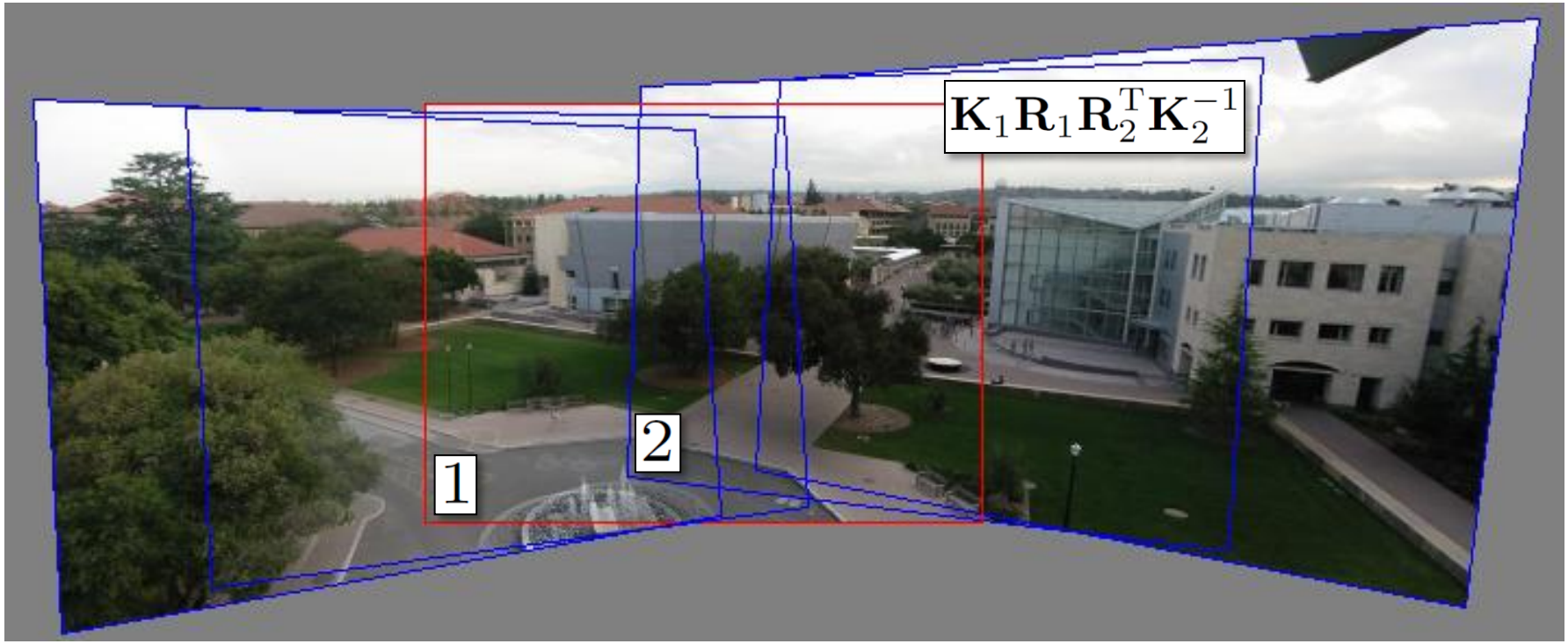
$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \sim \mathbf{K}_1 \mathbf{R}_1 \mathbf{R}_2^{\mathrm{T}} \mathbf{K}_2^{-1} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

image coords (in image 1)

3x3 homography

image coords (in image 2)

Image alignment



Can we use homography to create a 360 panorama?