CSCI 497P/597P: Computer Vision

Scott Wehrwein

Intrinsics, Extrinsics, and Stereo 1: Depth from Disparity

Goals

- Know a general projection matrix can be decomposed into intrinsics and extrinsics
- Understand how to calculate depth from disparity in a rectified stereo image pair.

Announcements

- Better notes on spherical warping are linked from last Friday's lecture on the course webpage.
- Takehome exam Friday—Monday, to cover material through last week.

Spherical reprojection

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

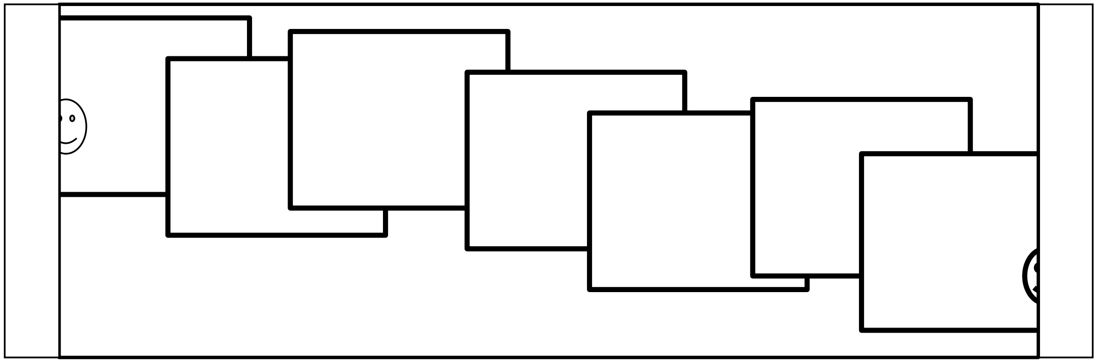
- Map image to spherical coordinates
  - need to know the focal length

Aligning spherical images

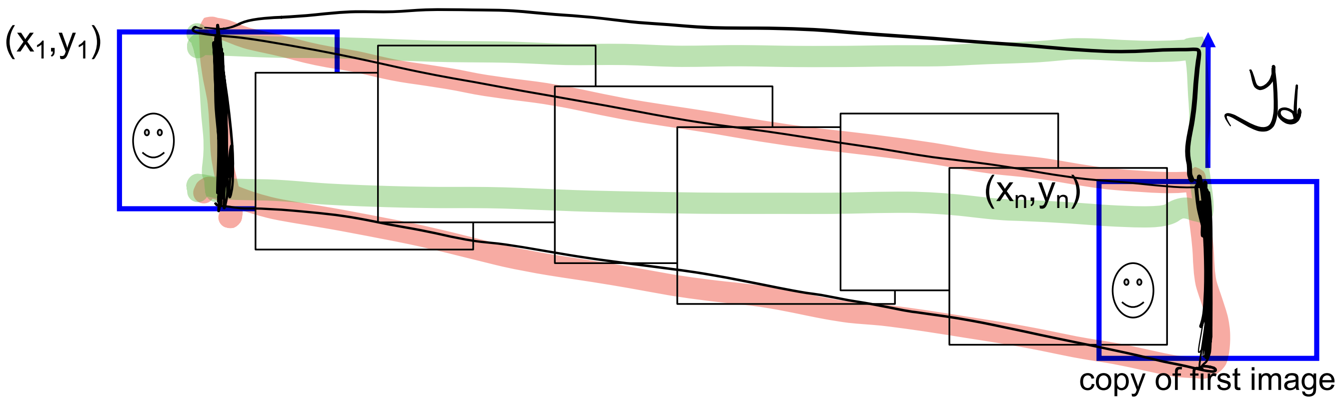
- Suppose we rotate the camera by θ about the vertical axis
  - How does this change the spherical image?

360 Problems: Drift

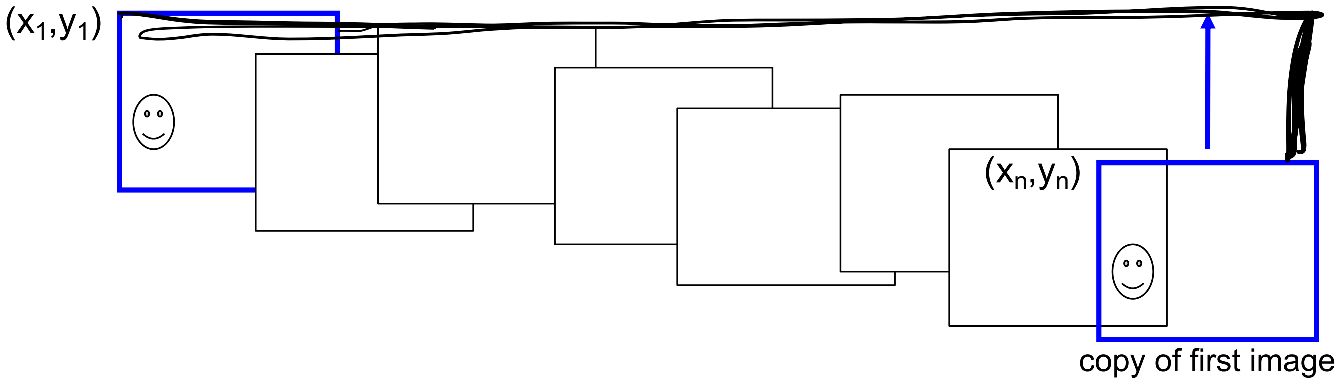


- Error accumulation
  - small errors accumulate over time

360 Problems: Drift

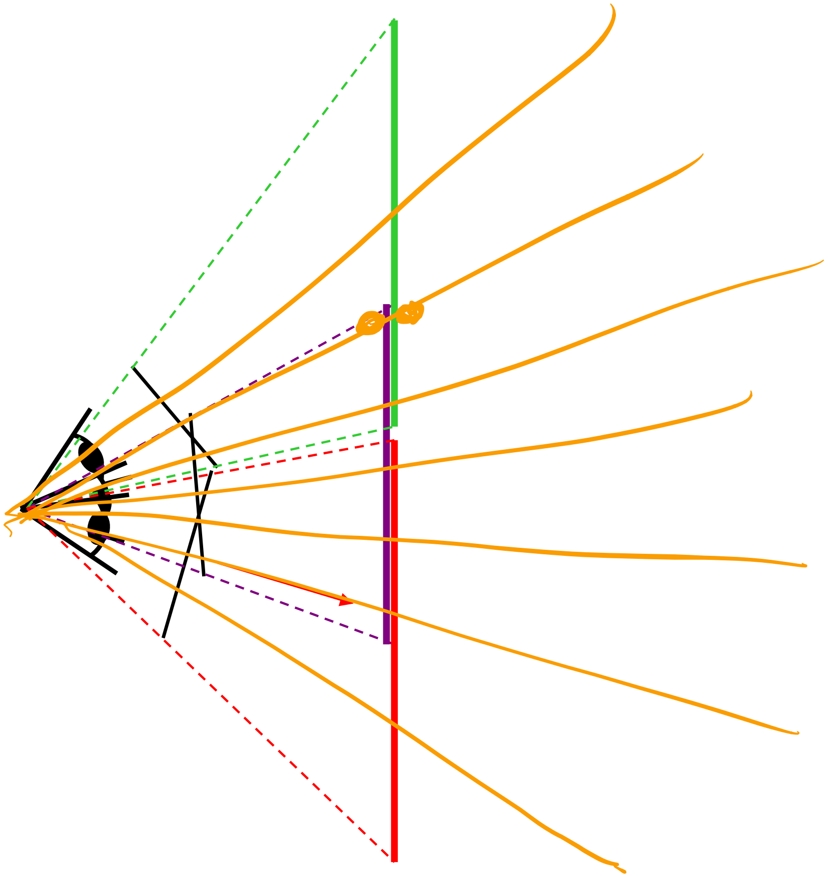


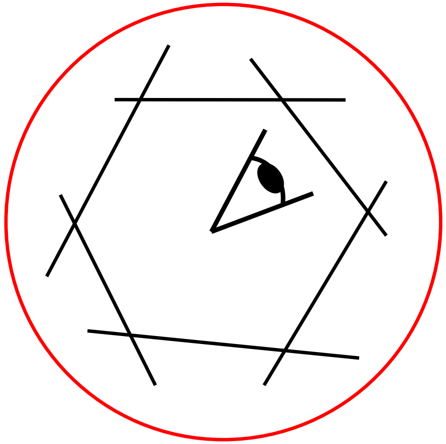
360 Problems: Drift



- Solutions
  - 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: $y' = y + ax$**
    - run a big optimization problem, incorporating this constraint
      - best solution, but more complicated
      - known as "bundle adjustment"

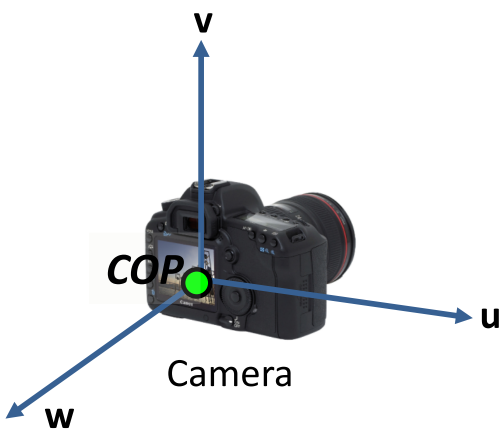
Panoramas require a common COP

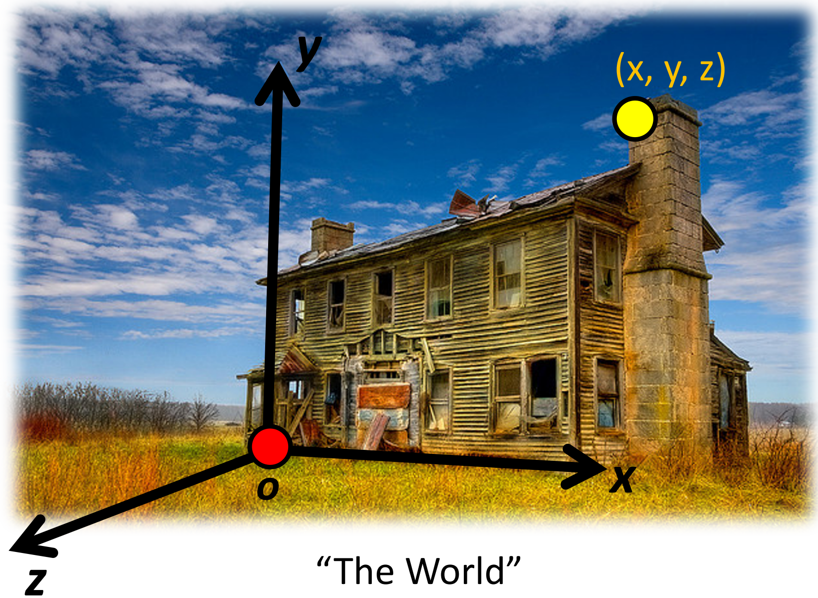




Camera(s) without a common COP

- With panoramas, we always assumed a common COP.
- How can we model the geometry of a camera in a separate world coordinate system?



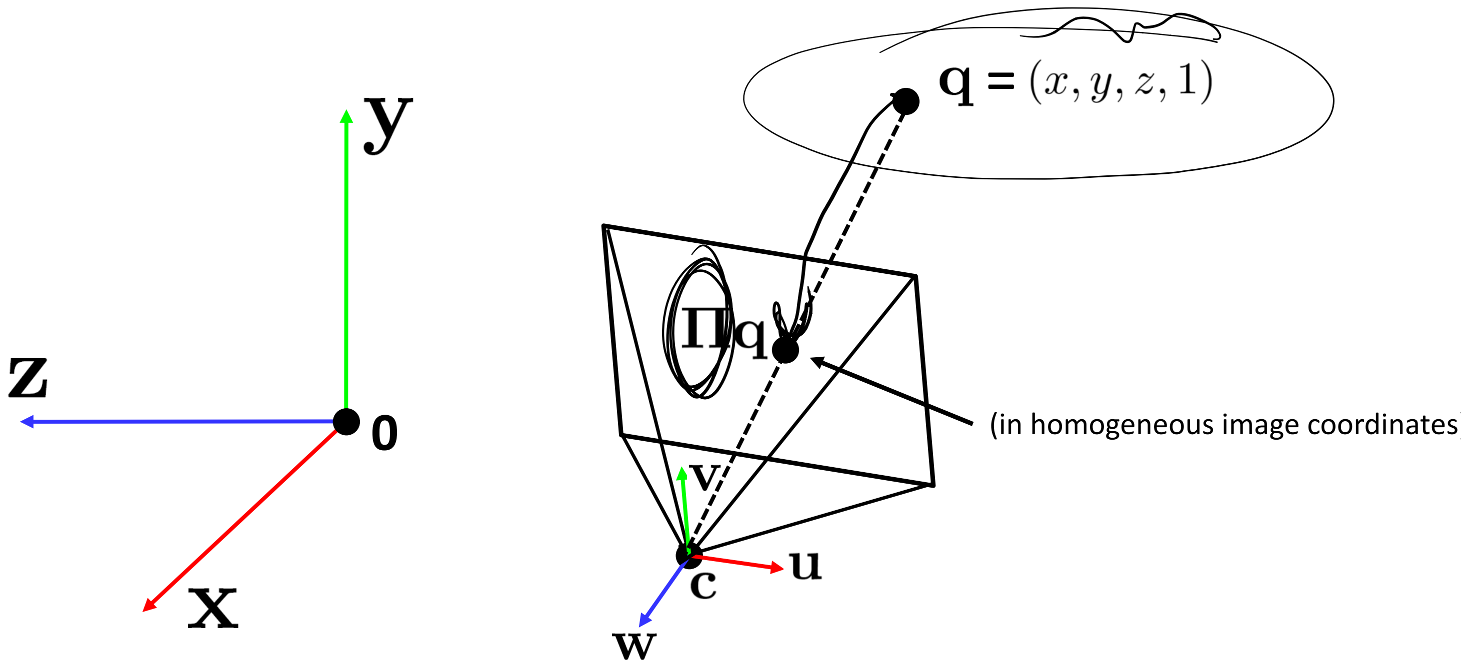


Two important coordinate systems:

1. *World* coordinate system
2. *Camera* coordinate system

How do we project a given point (x, y, z) in world coordinates?

Projection matrix



Intrinsic Camera Parameters

- Everything you need to get from **camera** coordinates to **pixel** coordinates:

4-to-3

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

focal, etc

Π

K (intrinsics)

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

- Getting more general:

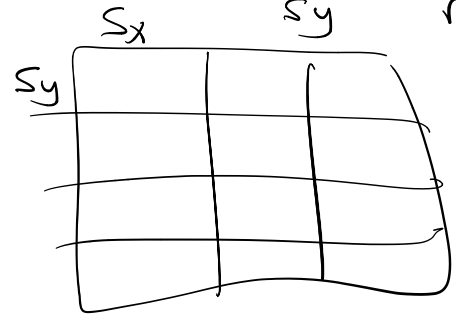
$$\mathbf{K} = \begin{bmatrix} -f & s & c_x \\ 0 & -\alpha f & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

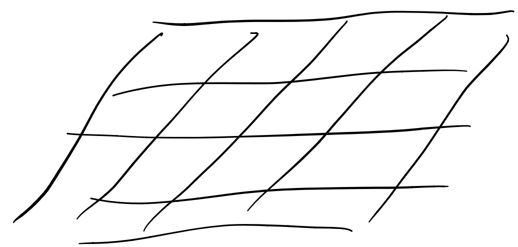
$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \leftarrow \Pi \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix}$$

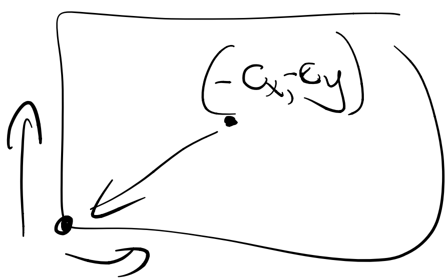
↗ image plane

↑ camera

$\alpha = \frac{s_x}{s_y}$ = aspect ratio







Intrinsic Camera Parameters

Everything you need to get from **camera** coordinates to **pixel** coordinates:

$$\underbrace{\begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\substack{\mathbf{K} \\ \text{(intrinsics)}}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

(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)

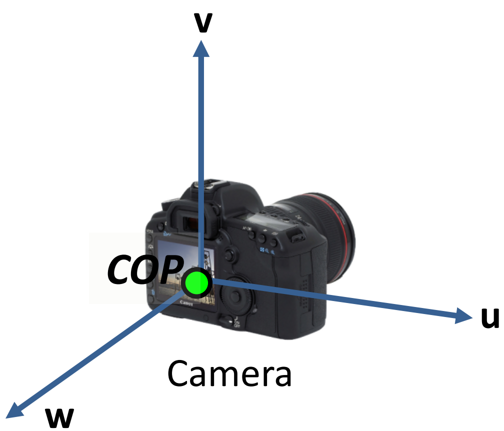
α : **aspect ratio** (1 unless pixels are not square)

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

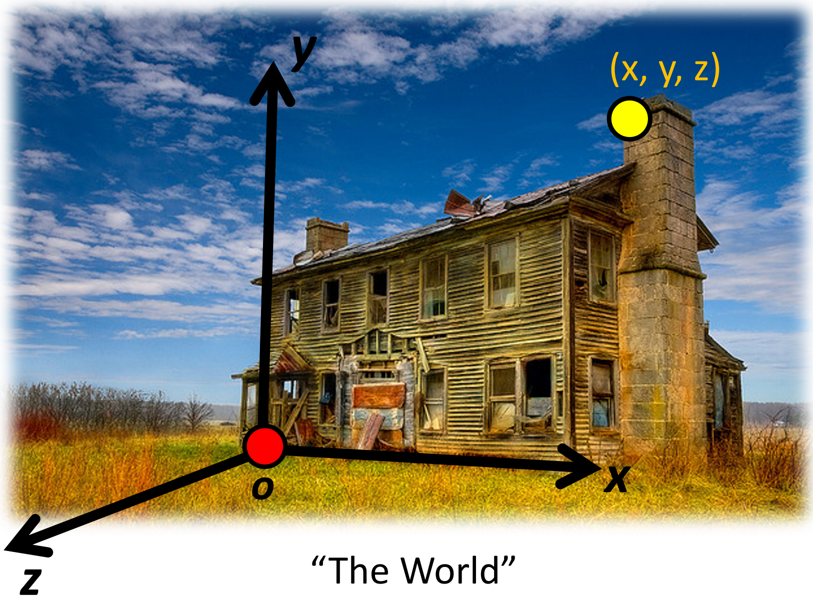
(c_x, c_y) : **principal point** ((0,0) unless optical axis doesn’t intersect projection plane at origin)

Camera(s) without a common COP

- With panoramas, we always assumed a common COP.
- How can we model the geometry of a camera in a separate world coordinate system?



Camera



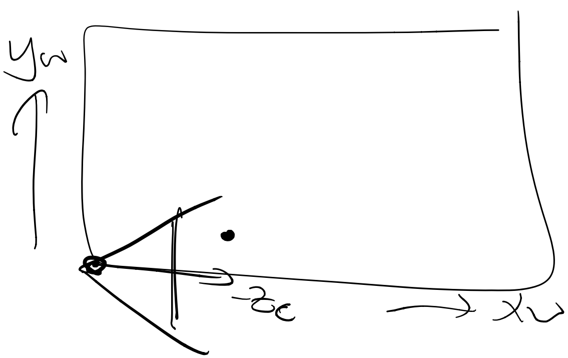
"The World"

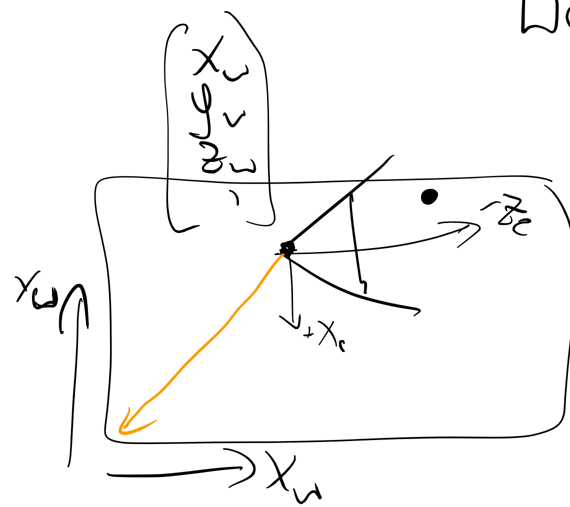
Two important coordinate systems:
1. *World* coordinate system
2. *Camera* coordinate system

How do we project a given point (x, y, z) in world coordinates?

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \Pi \\ K[1,\ 1,\ 1\ 0] \end{bmatrix} \begin{bmatrix} ? \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

img   camera   World





1. Translate $(0,0)_c$ to $(0,0)_w$

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3\times3} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x_w \\ 0 & 1 & 0 & -y_w \\ 0 & 0 & 1 & -z_w \\ 0 & 0 & 0 & 1 \end{bmatrix} \hat{P}$$

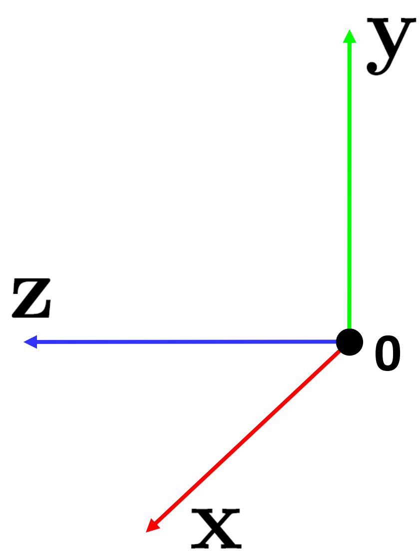
Extrinsic Camera Parameters

- Everything you need to get from **world** coordinates to **camera** coordinates

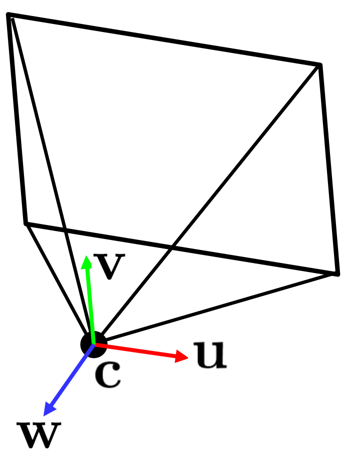
$$K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} & \mathbf{R} & & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3\times 3} & -\mathbf{c} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}$$

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)

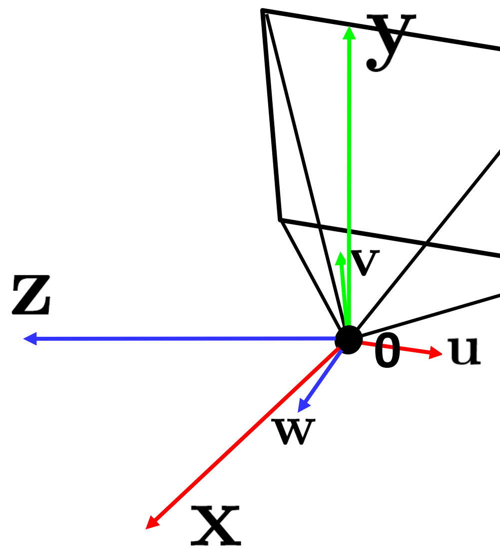


Step 1: Translate by -**c**



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)



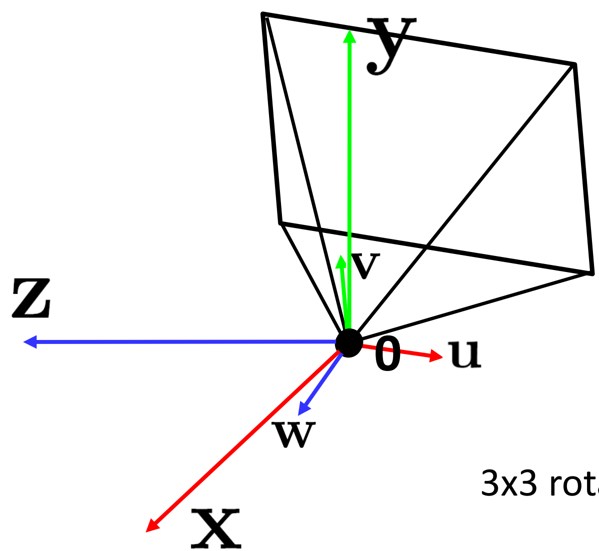
Step 1: Translate by -**c**

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**

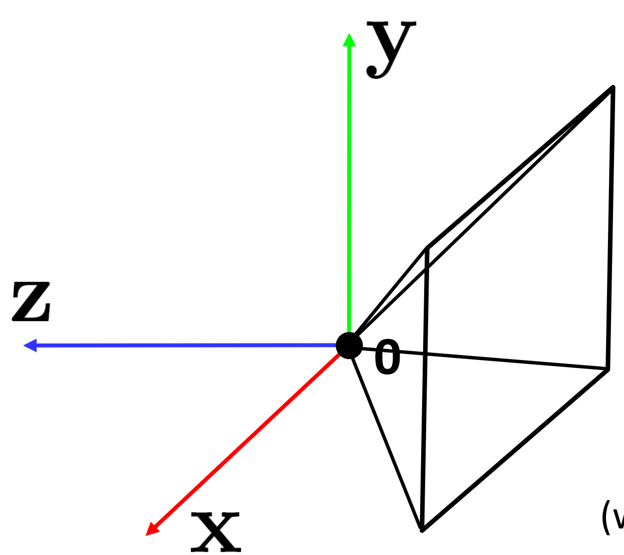
Step 2: Rotate by **R**

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

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**

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

(with extra row/column of [0 0 0 1])

Projection matrix: Putting it all together

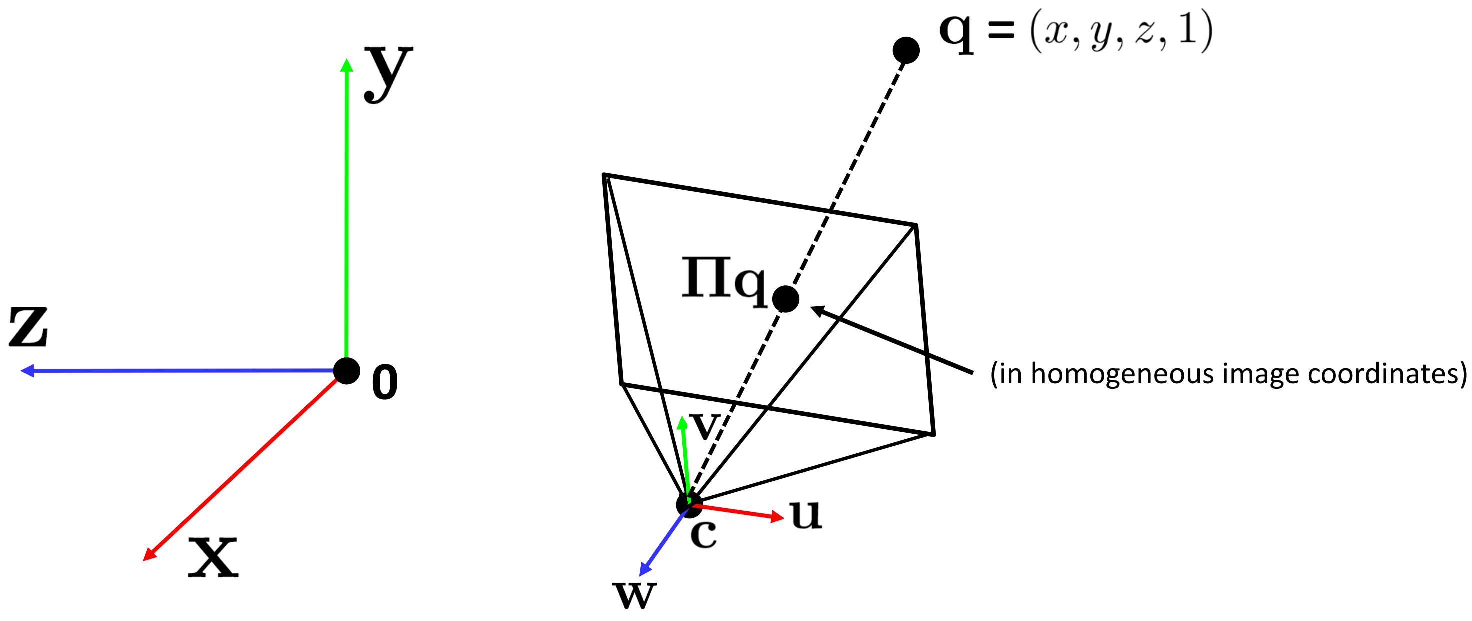
$$\mathbf{\Pi} = \mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} & \mathbf{R} & & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} & \mathbf{I}_{3\times 3} & & -\mathbf{c} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

intrinsics — projection — rotation — translation

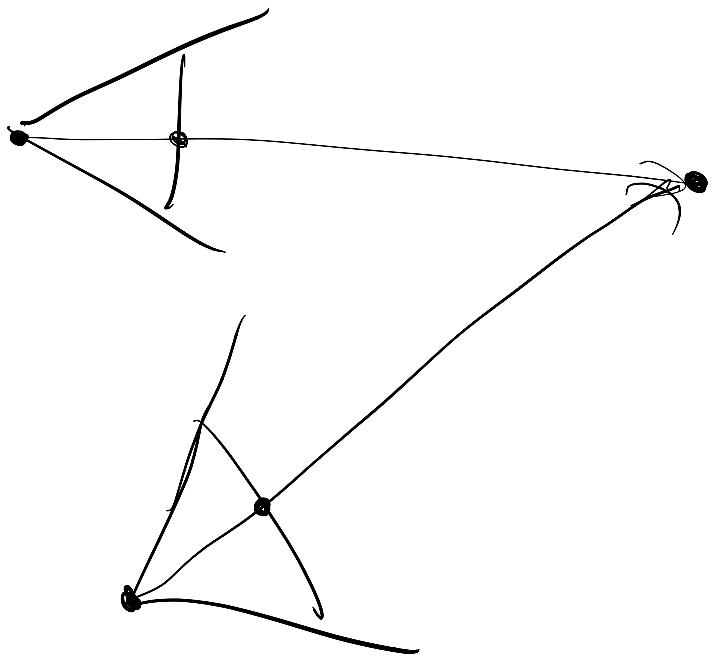
The **K** matrix converts 3D rays in the camera's coordinate system to 2D image points in image (pixel) coordinates.

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).

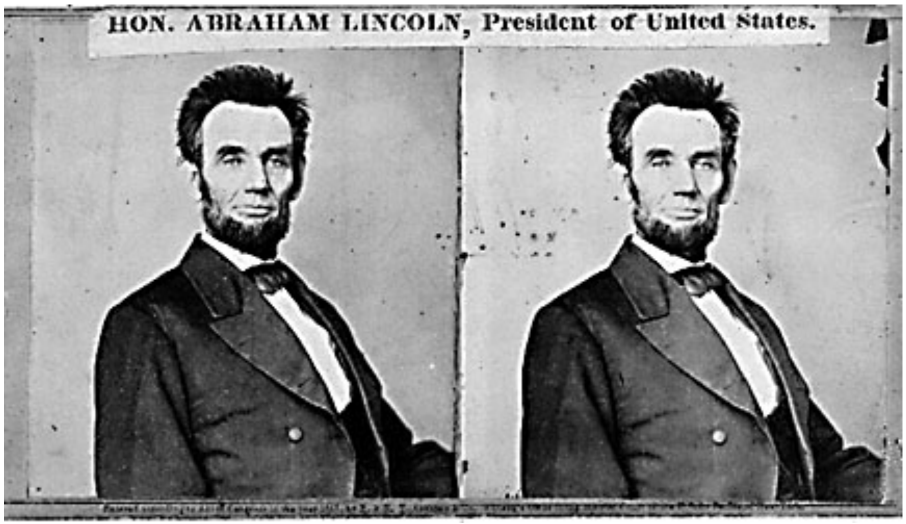
Projection matrix



What happens when cameras have different COPs?



Stereo



- Given two images from different viewpoints
  - How can we compute the depth of each point in the image?
  - Based on *how much each pixel moves* between the two images

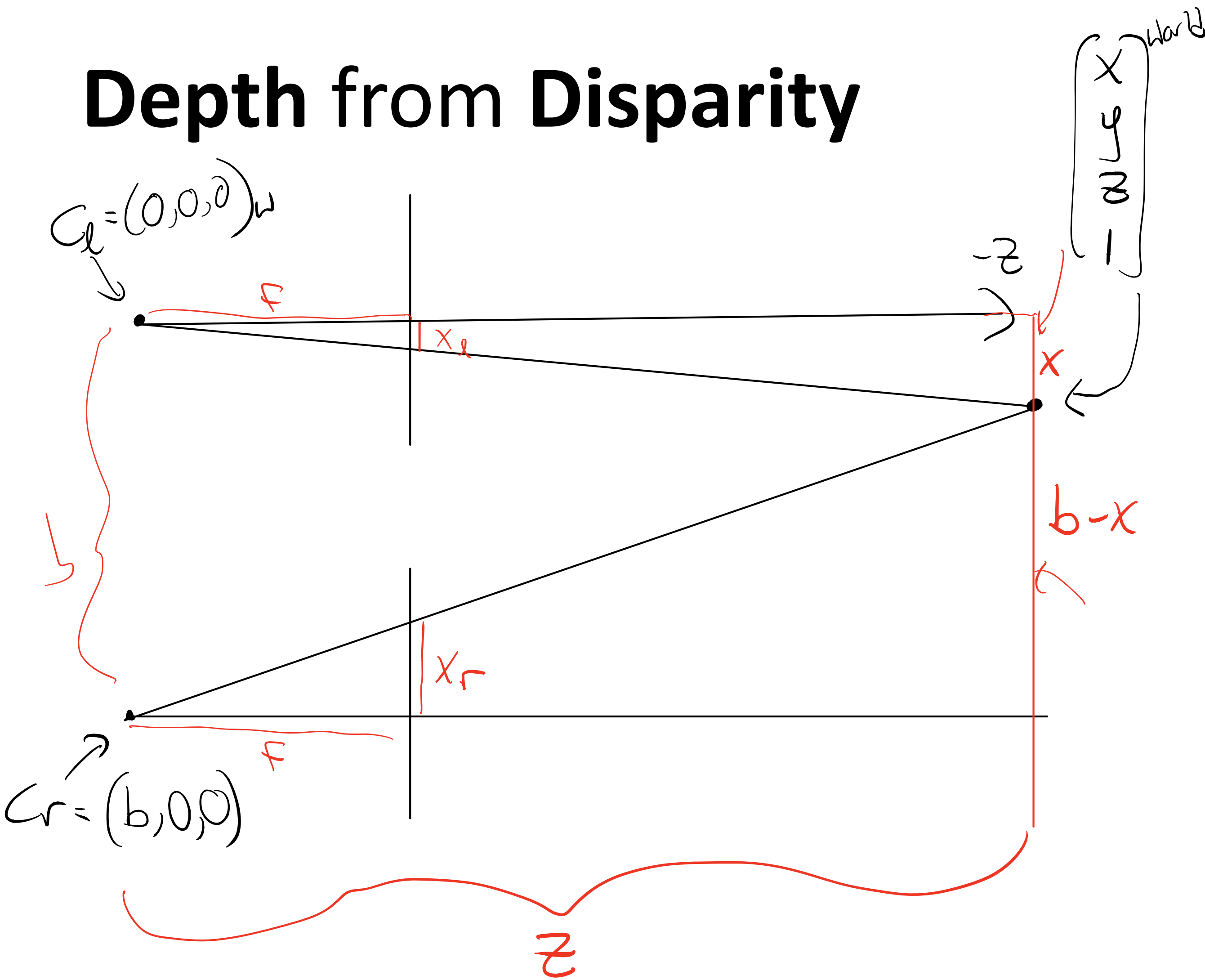
$$\text{depth} \propto \frac{1}{\text{disparity}}$$

Hypothesis generation time: what relationship do you expect to find between **depth** and **how much a pixel moves**?

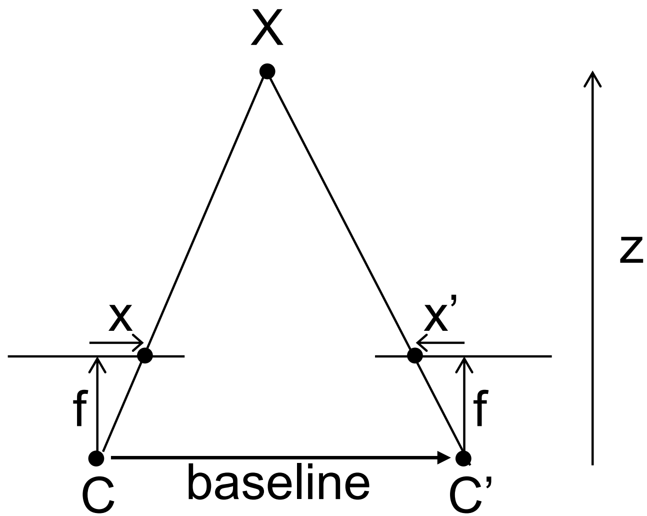
Depth from Disparity

Assumptions

- 2 cameras
- same f
- same PP
- COP off by b in X
  (b ↑ known)



Depth from disparity



$$disparity = x - x' = \frac{baseline * f}{z}$$