COM SCI 188

Intro to Robotics

Lecture 6

Yuchen Cui

Winter 2026

Agenda

- Announcements

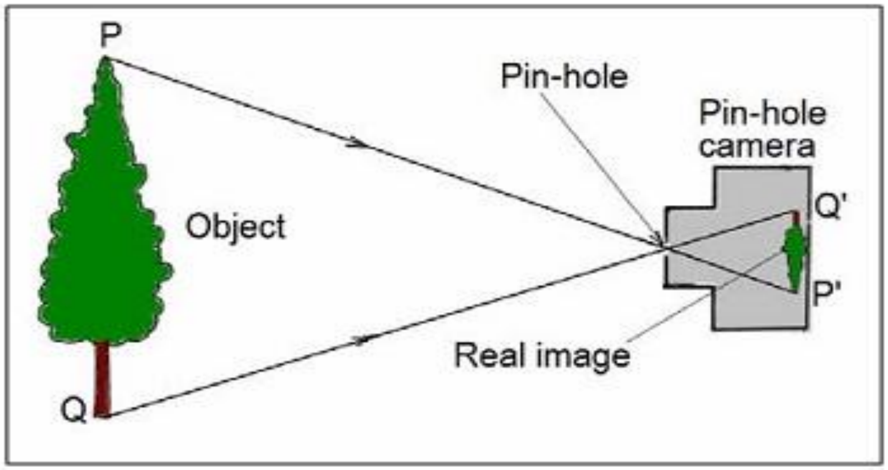
- Cameras: 2D & 3D
- Camera Calibration
- <break>
- Computer Vision Basics
  - Image Processing
  - Convolutional Neural Networks

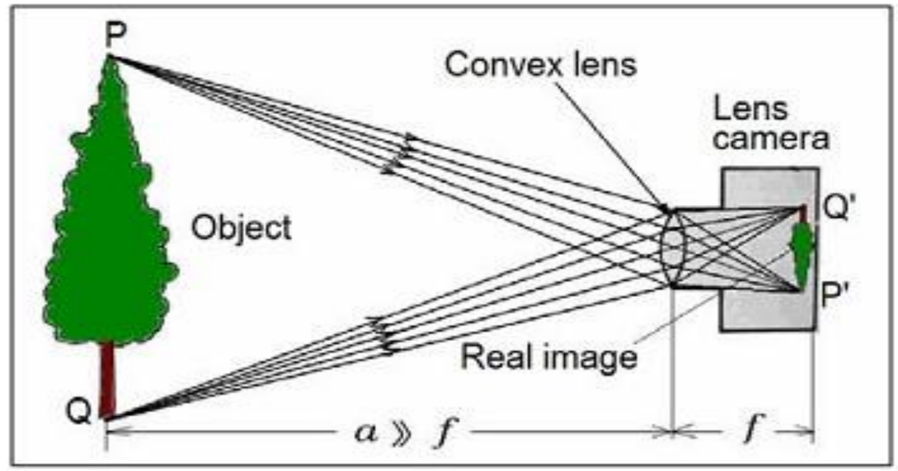
Announcements

- **Coding Assignment 1 – ddl extended to Sunday**
- Problem Set 2 due next Friday

- Discussion
  - More on Quaternions & PID control
  - Practice problems

Recap: Cameras





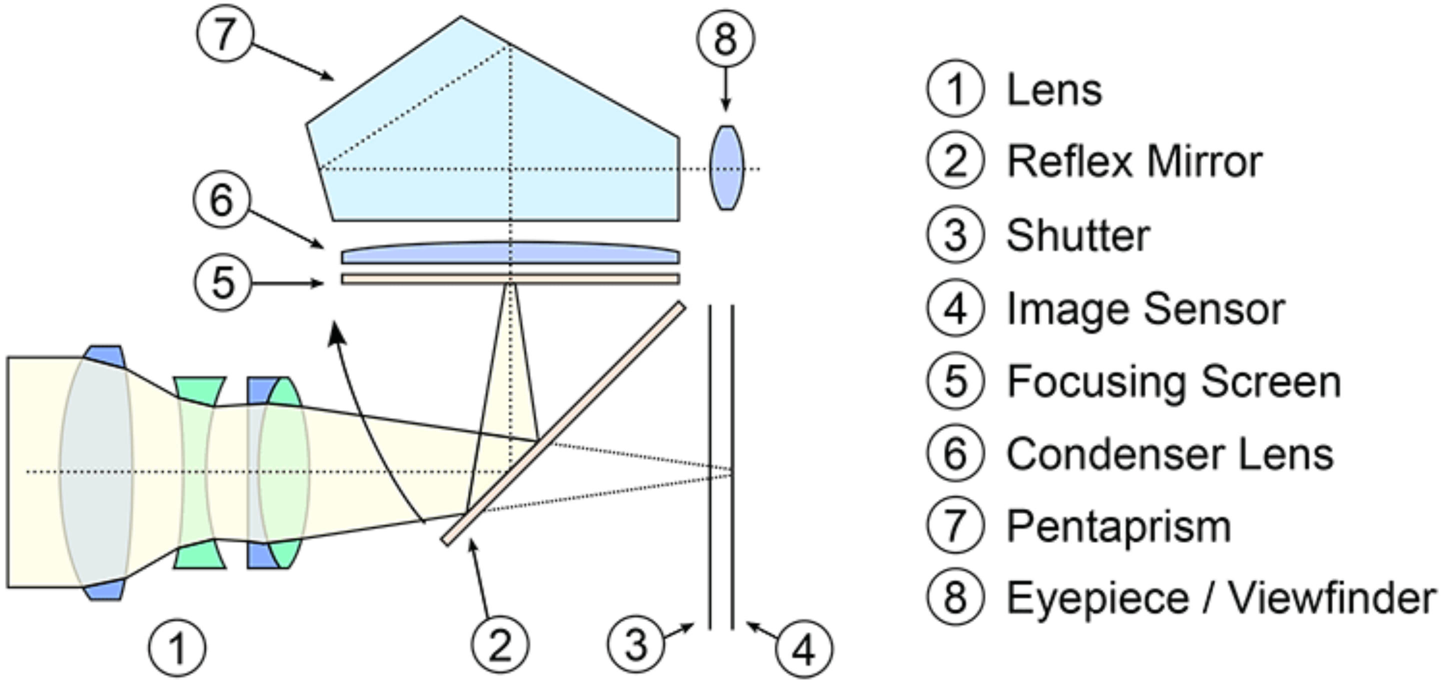
Pinhole Camera

- Geometric camera
- Pinhole passes light -> dim image
- No focus plane (blur is the same at all distances)

Lens Camera

- Optical camera
- Lens refracts light -> bright image
- Perfect focus at one plane

Digital Single-Lens Reflex (DSLR) Camera

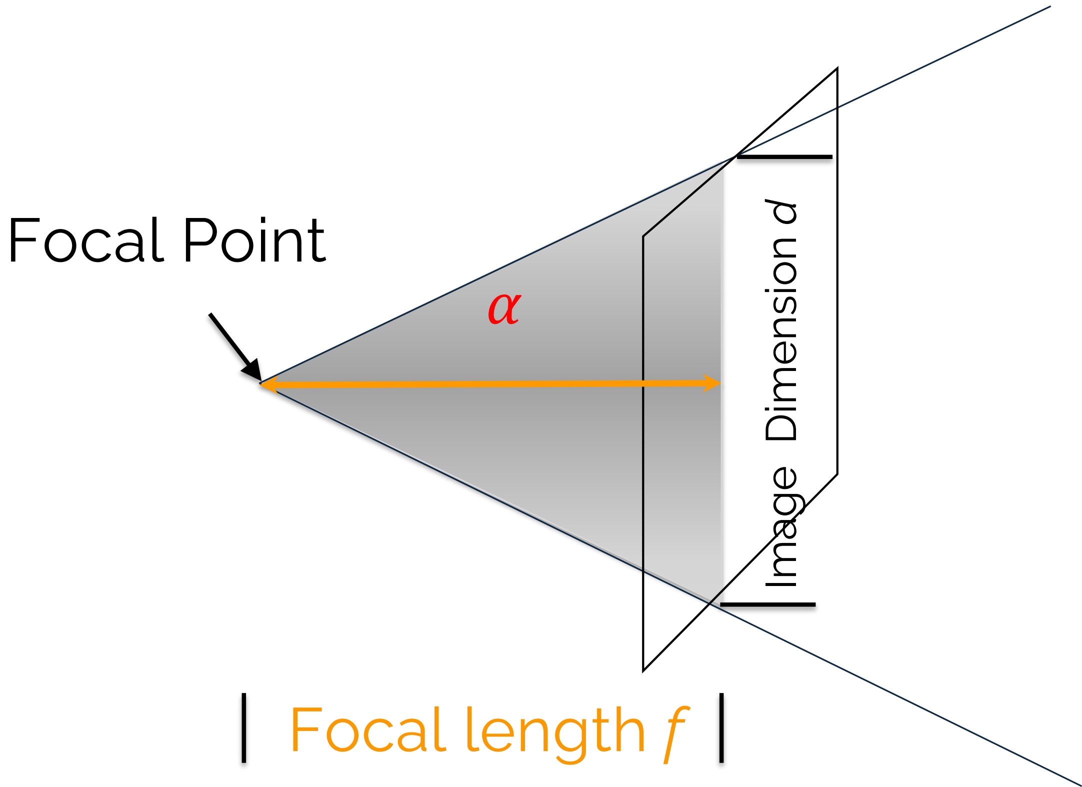


Pinhole Camera Geometry

Motivation

- Physics of real cameras are all different (too tedious to model all of them).
- But they all try their best to approximate pinhole camera.
- So in most of computer vision subjects, we model all cameras mathematically as a pinhole camera.

Field of View



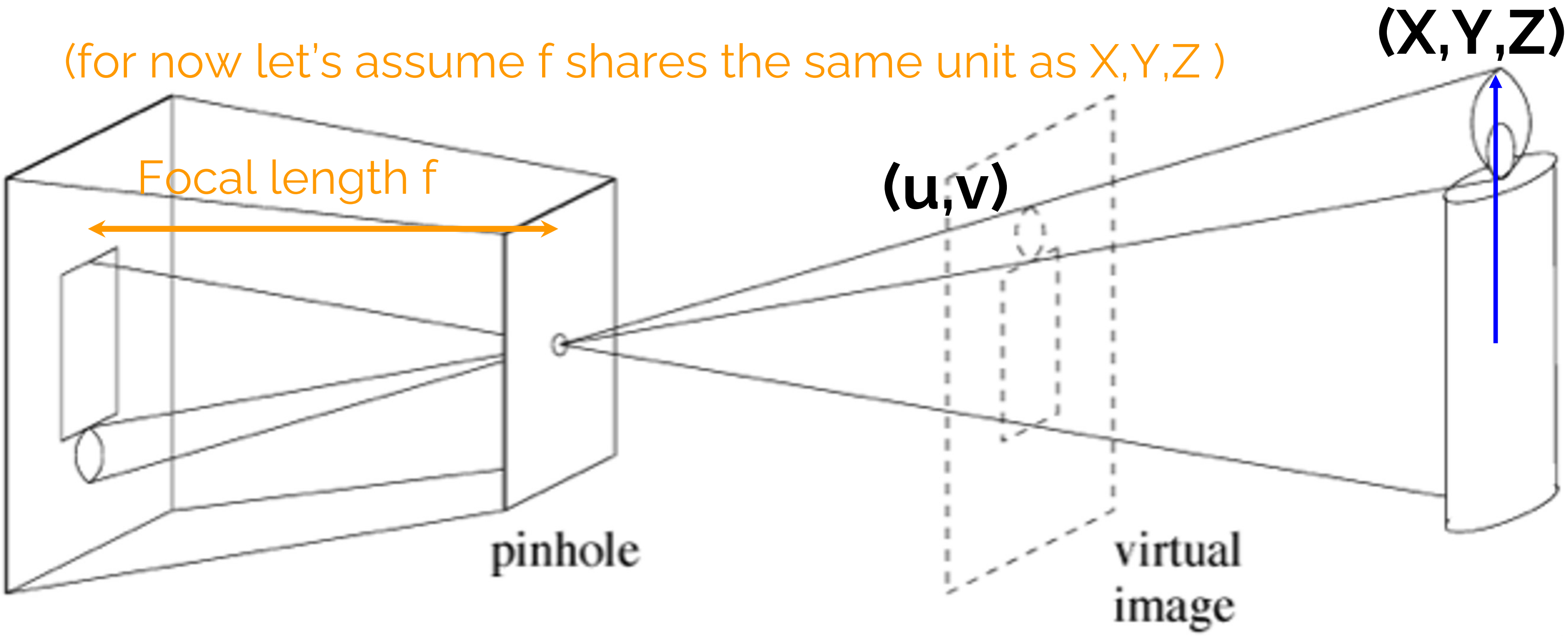
Distance from the focal point to image plane

FoV: $\alpha = 2\,\mathrm{atan}\,[d/(2f)]$

Unit of FoV α is degree

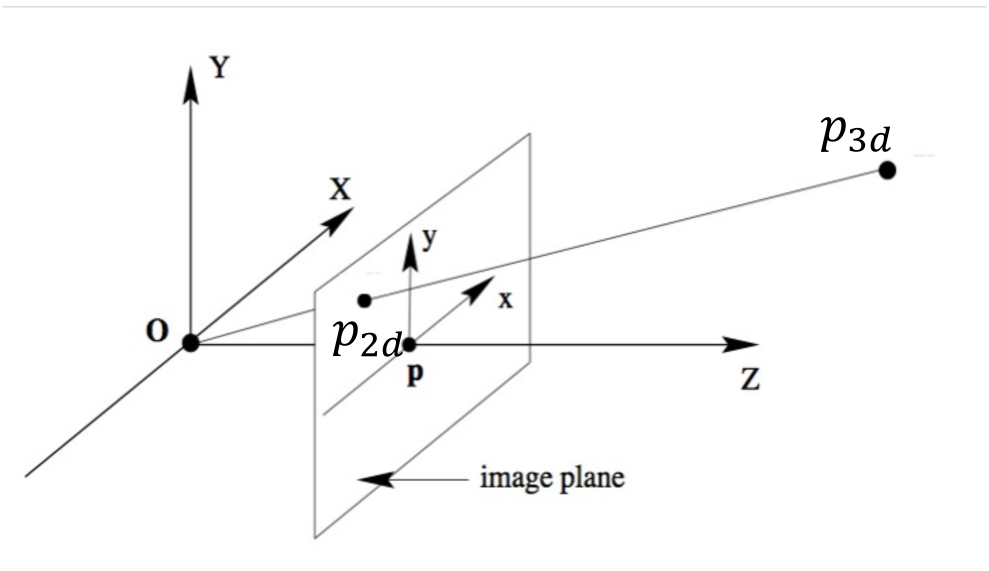
Each camera has two FoV:
Vertical & horizontal FoV

Focal Length



How to compute the 2D pixel location (u,v) image from the 3D location X,Y,Z?

Camera Projection:



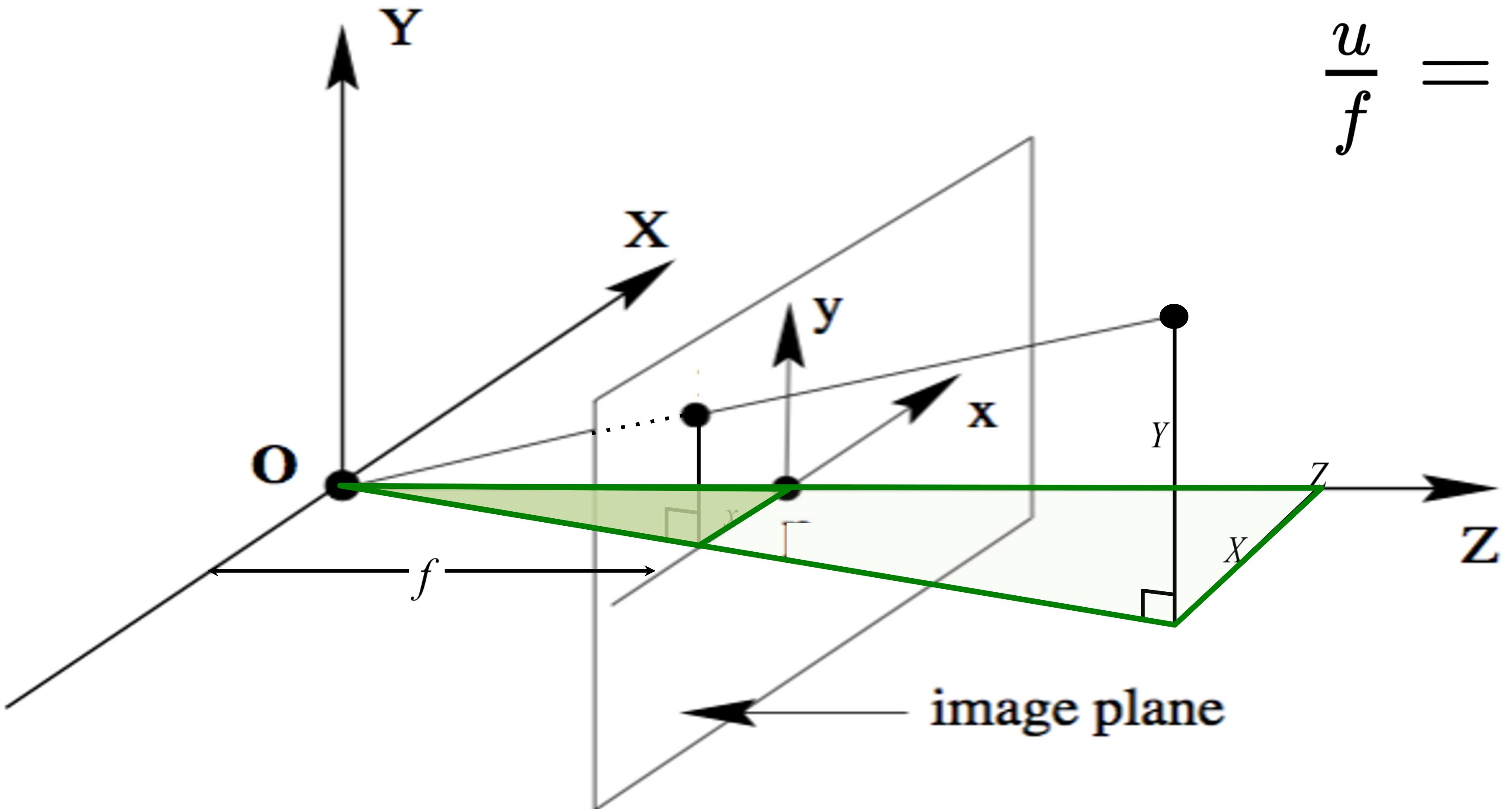
Camera coordinate:

Camera center is origin.

$$p_{2d} = \begin{bmatrix} u \\ v \end{bmatrix}$$

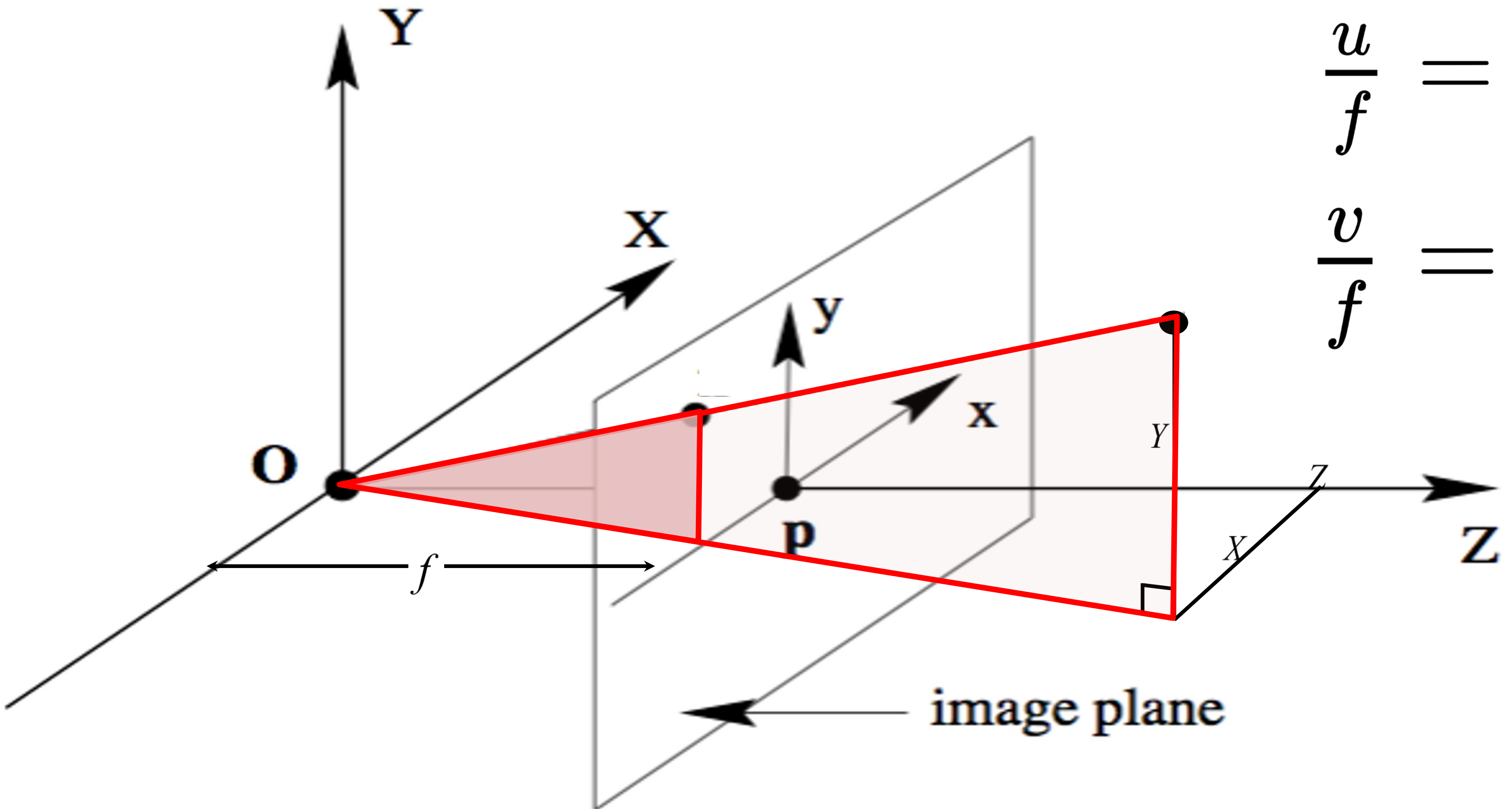
$$p_{3d} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Camera Projection:



$$\frac{u}{f} = \frac{X}{Z}$$

Camera Projection:



$$\frac{u}{f} = \frac{X}{Z}$$

$$\frac{v}{f} = \frac{Y}{Z}$$

Camera Projection:

$$u = \frac{fX}{Z}$$

$$v = \frac{fY}{Z}$$

$$\Leftrightarrow \quad \lambda \begin{bmatrix} u \\ v \\ f \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Rewrite in Homogeneous coordinates

Cartesian Coordinate

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix}$$

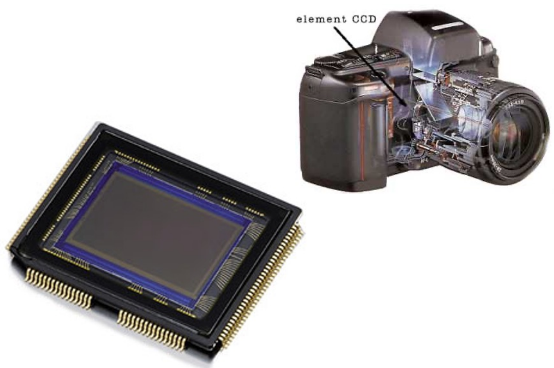
homogeneous coordinates

Equivalent
(w is non-zero scalar)

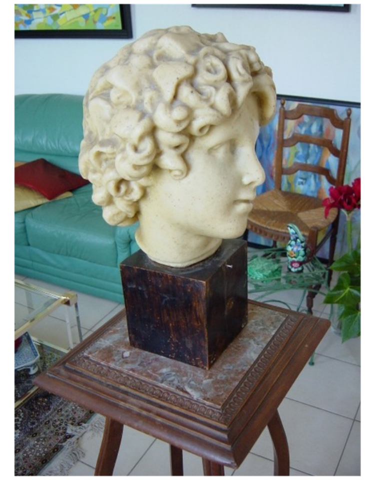
Convert from homogenous coordinate back to 2D coordinate

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} \Rightarrow \left(\frac{a}{c}, \frac{b}{c}\right)$$

Camera Projection:



If we let f to take care transform from word unit to image unit.
The unit of f is pixel



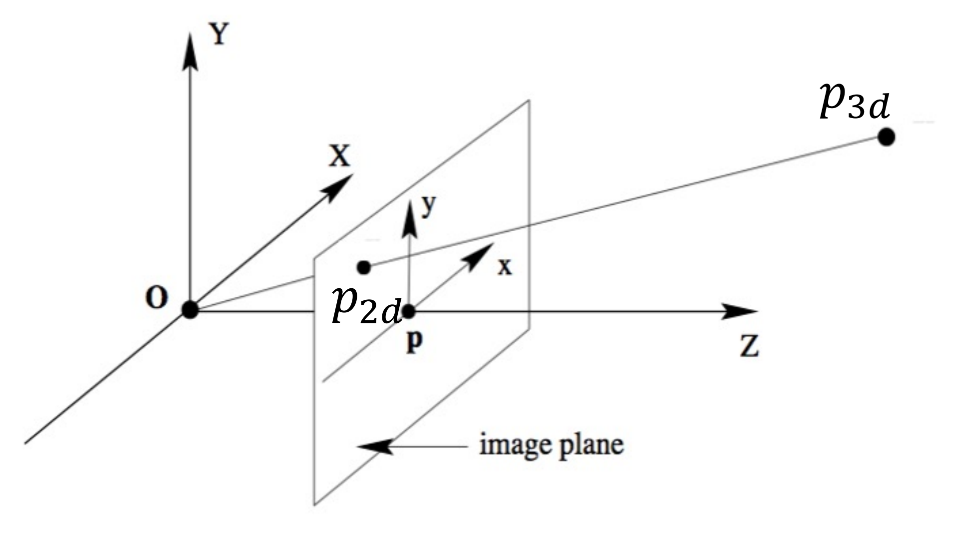
World Unit: e.g. Meters

Image Unit: Pixels

$$u = \frac{fX}{Z} \qquad v = \frac{fY}{Z}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Exercise



$$p_{3d} = \begin{bmatrix} 3 \\ 1 \\ 5 \end{bmatrix}$$

f = 500

$$p_{2d} = \begin{bmatrix} u \\ v \end{bmatrix}$$

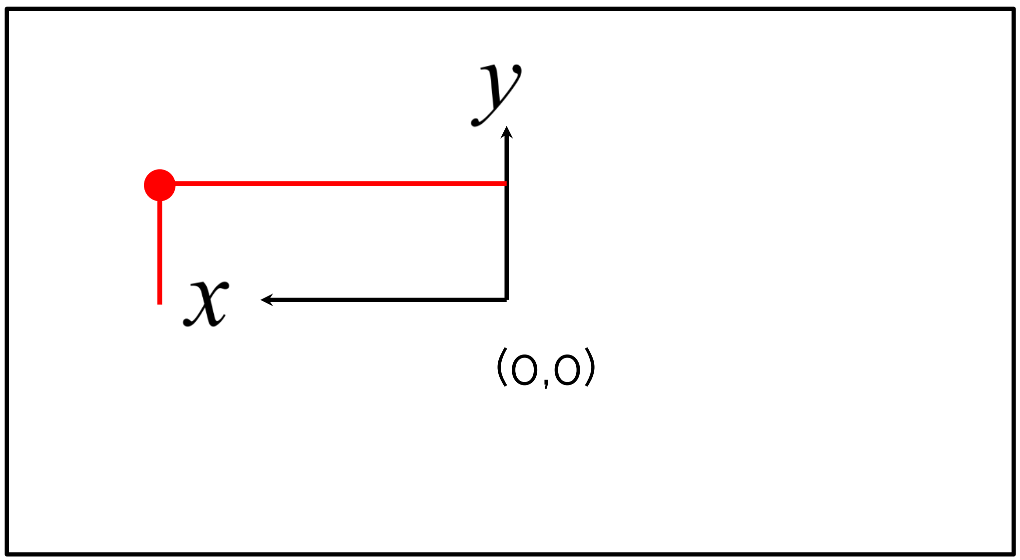
what's u,v?

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Image Coordinate

Picture

u = 300
v = 100



Until now, we use 2D coordinate conventions that are **consistent** with the 3D Camera coordinate. **However, if your application uses a different 2D coordinate, you'll need to further transform the (u,v)**

Image Coordinate (Example1)

$^{B}P = \begin{bmatrix} 300 \\ 100 \end{bmatrix}$

What is ^{A}P?
What is $^{A}T_{B}$?

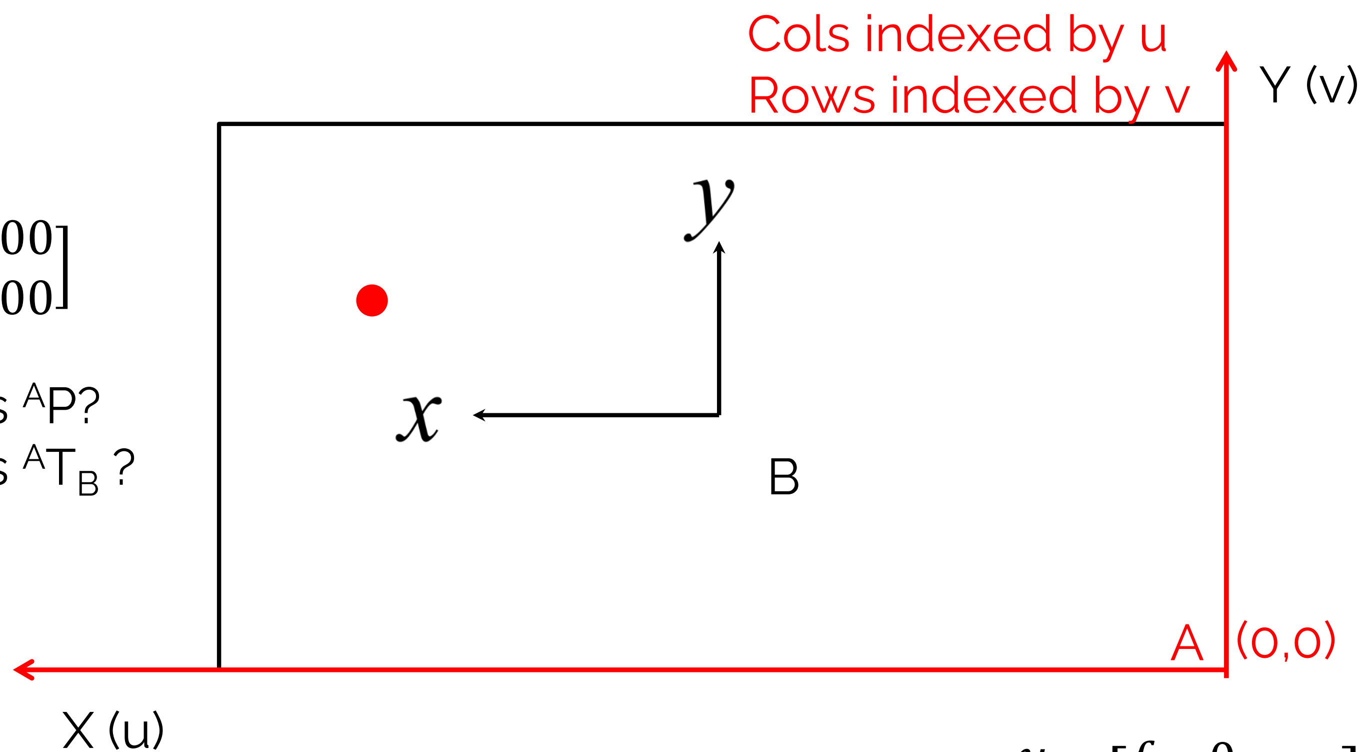


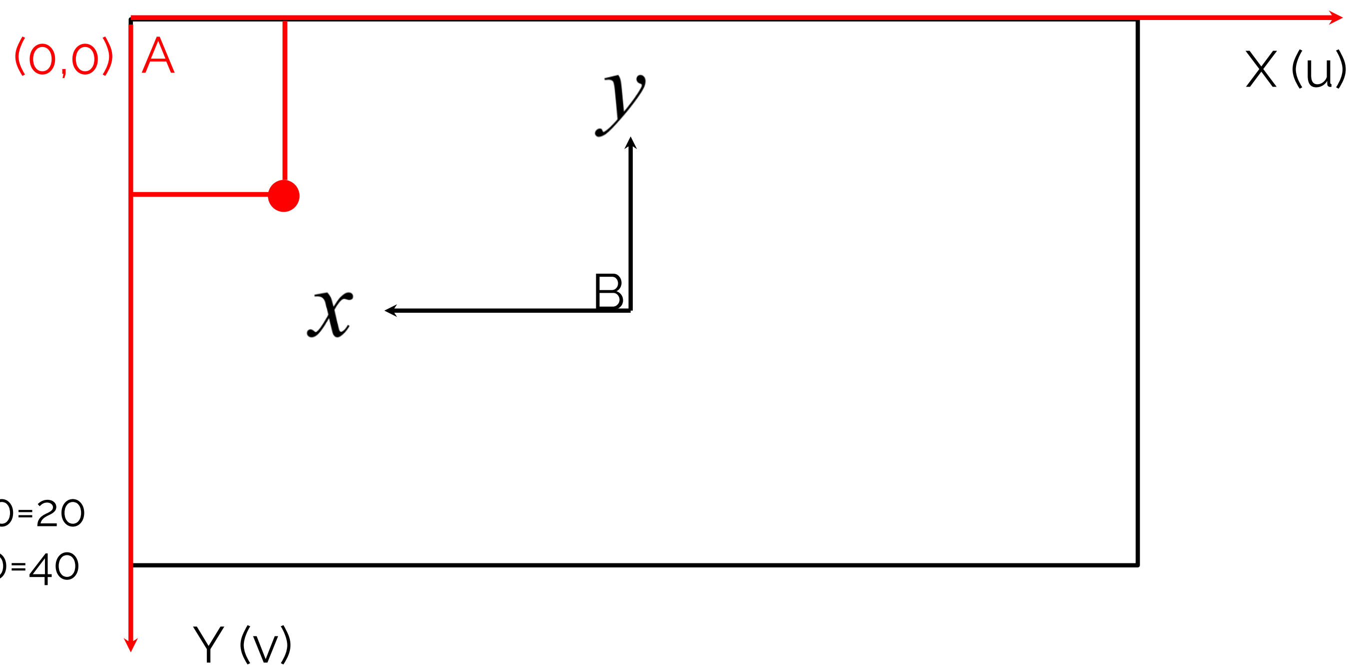
Image size
W = 640
H = 480

u' = 300+320=620
v' = 100+240=340

Matrix form

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

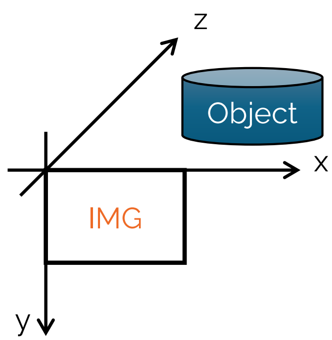
Image Coordinate (Example2)



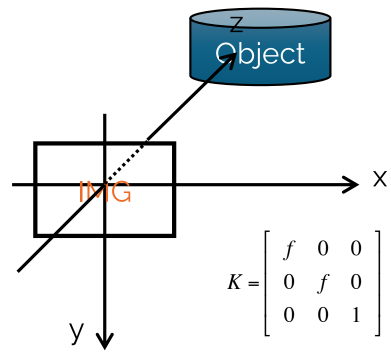
u = 300
v = 100

u' = -1*300+320=20
v' = -1*100+240=40

Popular Camera Coordinate System

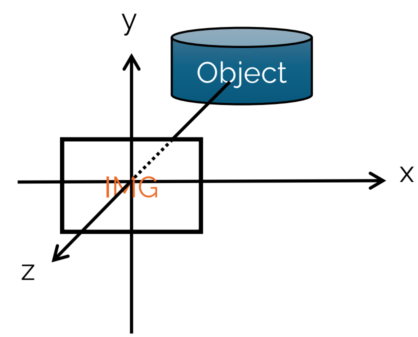


Computer Vision Researchers



$$K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

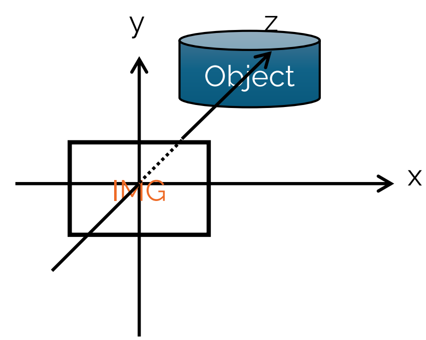
Lazy Grad Students



Graphics Guru ☺
OpenGL

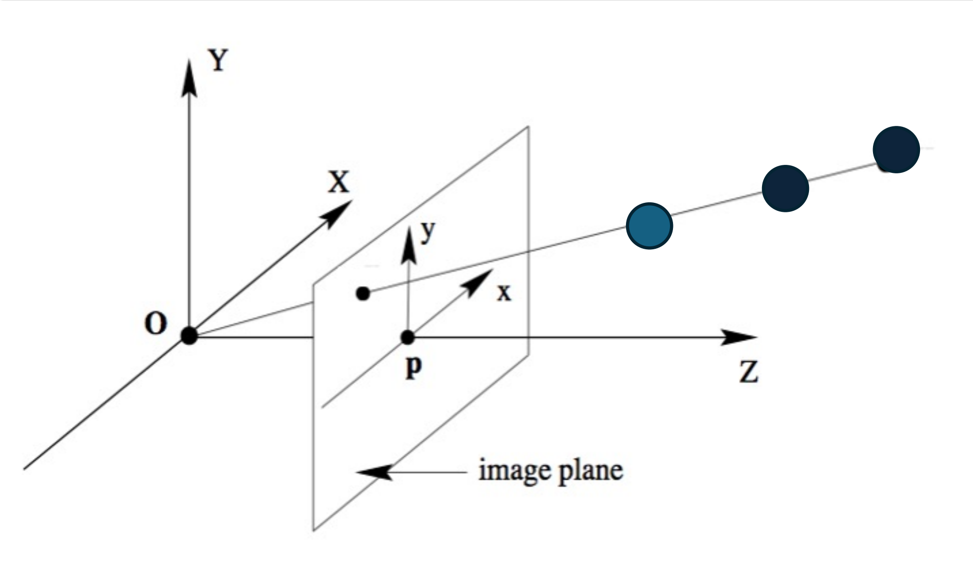
Right-Handed Coordinate System

Evil Microsoft DirectX ☺
= Left-Handed Coordinate System



Camera Projection:

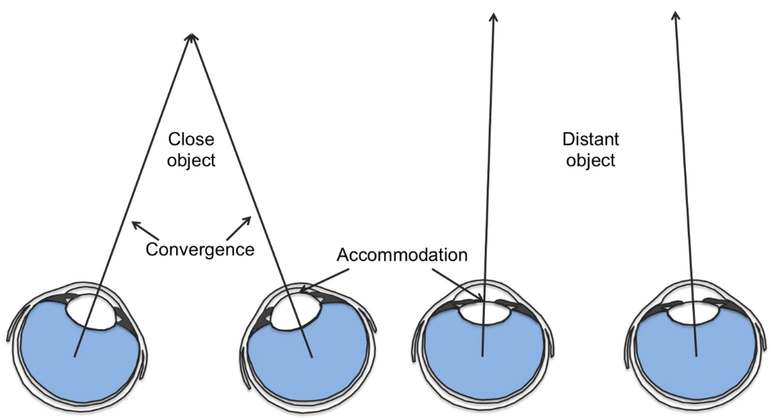
$$\lambda \begin{bmatrix} u \\ v \\ f \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$



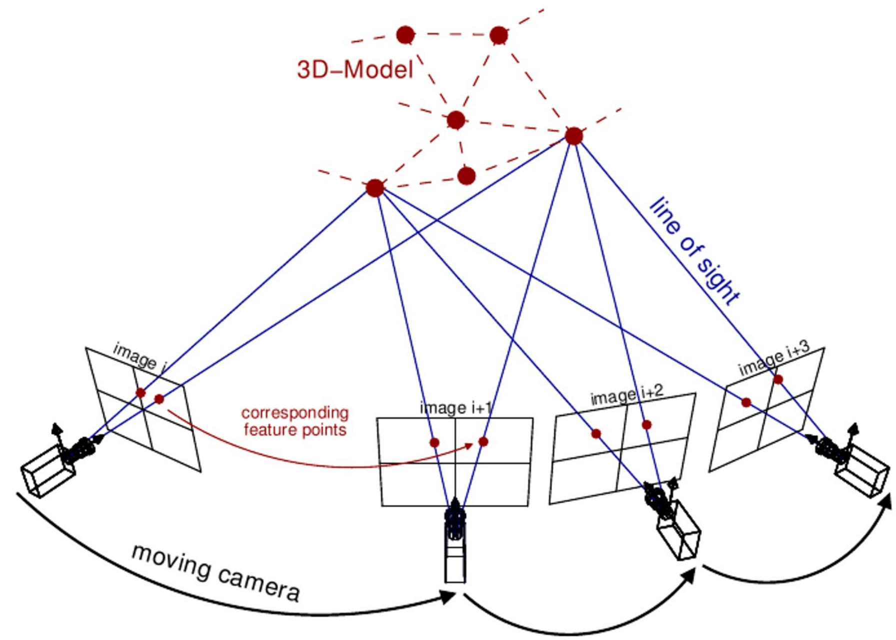
All the points on the same ray, projected to the same pixels [u,v]
Only the point with smallest distance is observed in the image.

3D Vision

How do humans & animals perceive depth?



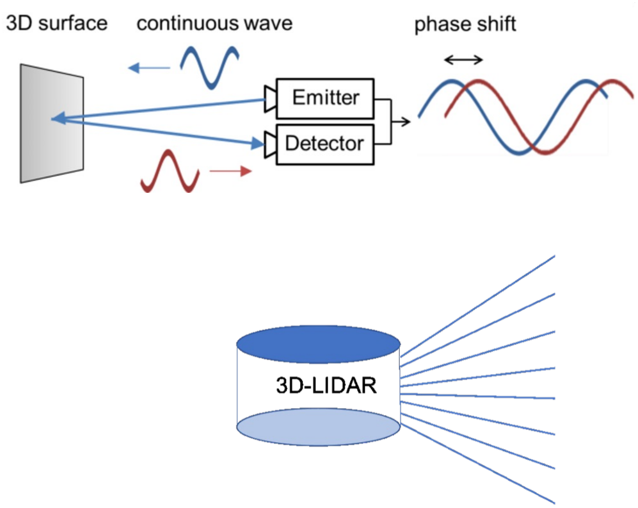
Binocular Vision



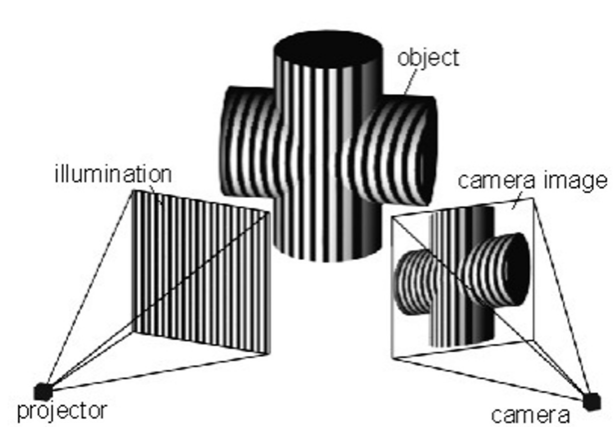
Structure from Motion (SfM)

How do robots perceive depth?

Stereo Camera

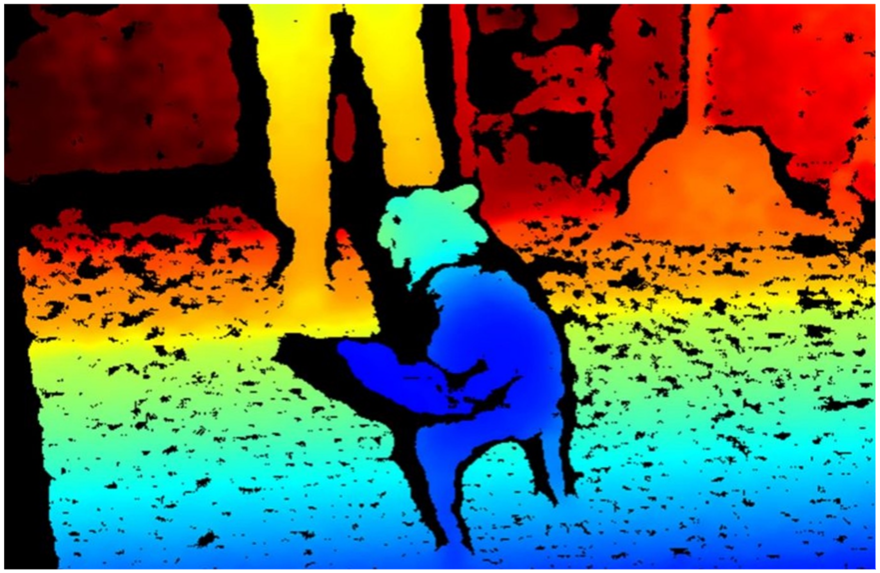


Time of Flight

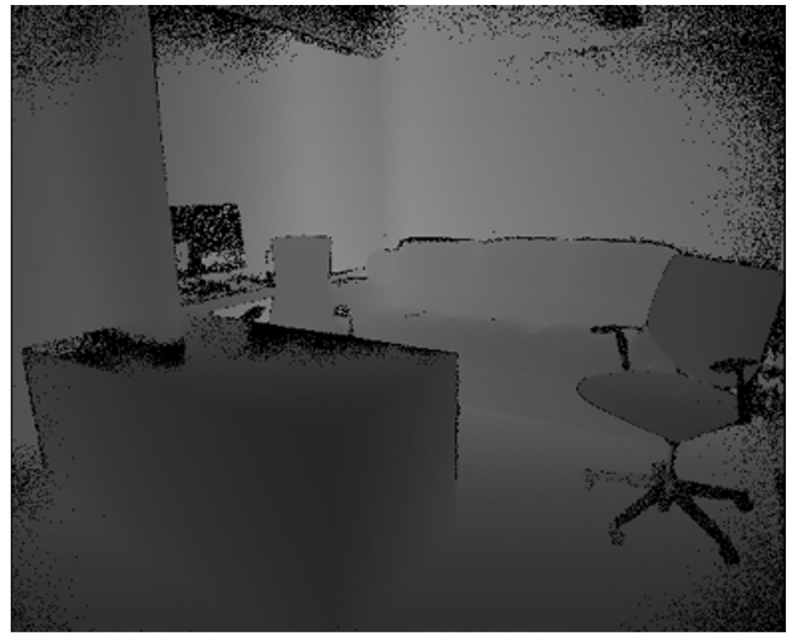


Structured Light

Intel Realsense

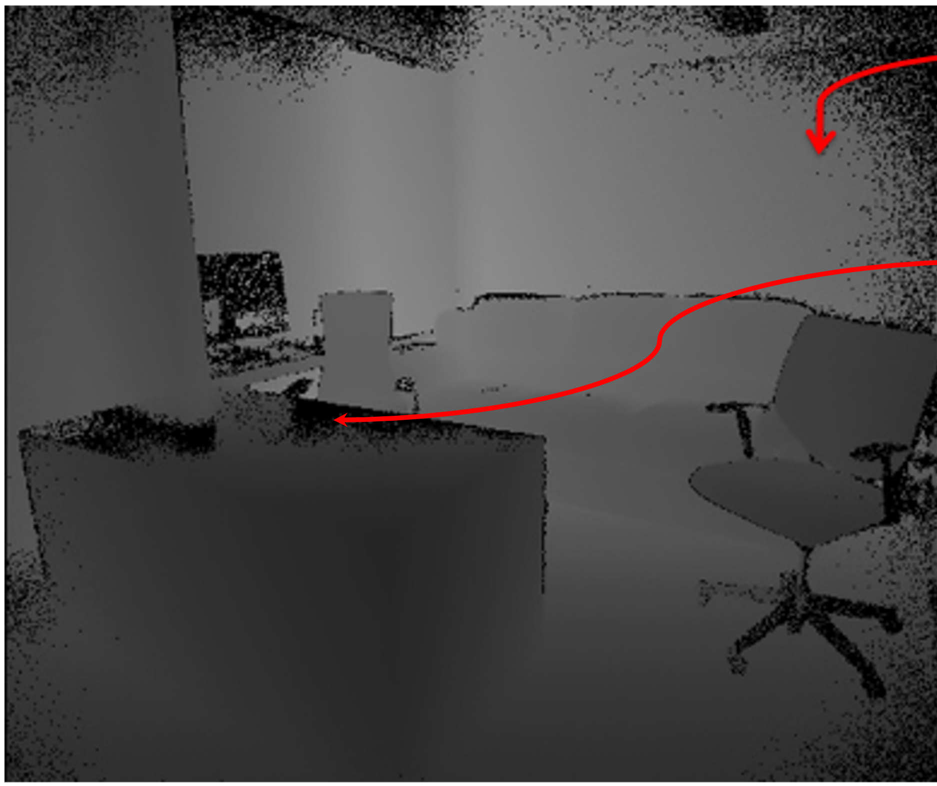


2D to 3D projection



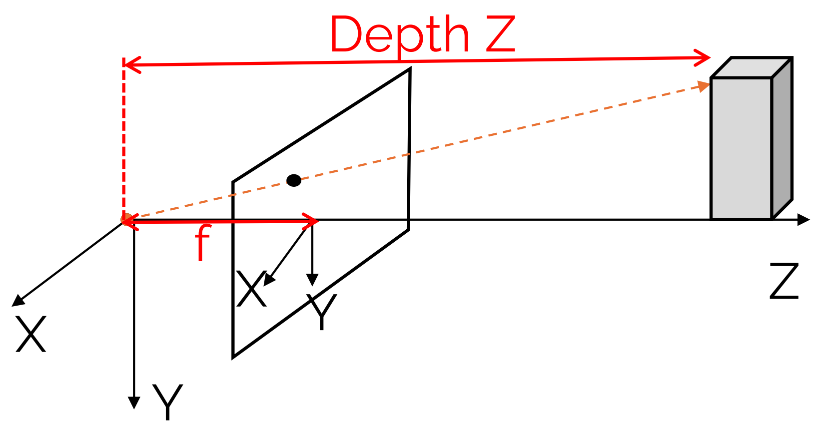
Knowing just 2D coordinate (u,v), we don't have enough information to compute the 3D point location (X,Y,Z).
However, with additional depth channel we can (RGB-D image).

RGB-D Image

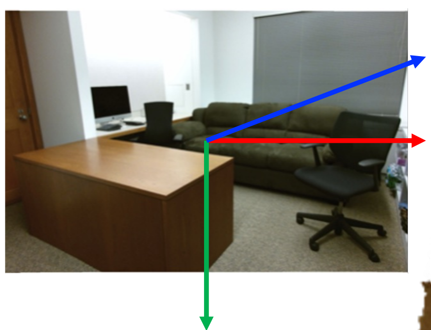


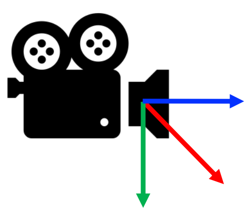
Depth Image (1xHxW):
Each pixel record depth value Z
(in meter or millimeter)

Missing depth = 0

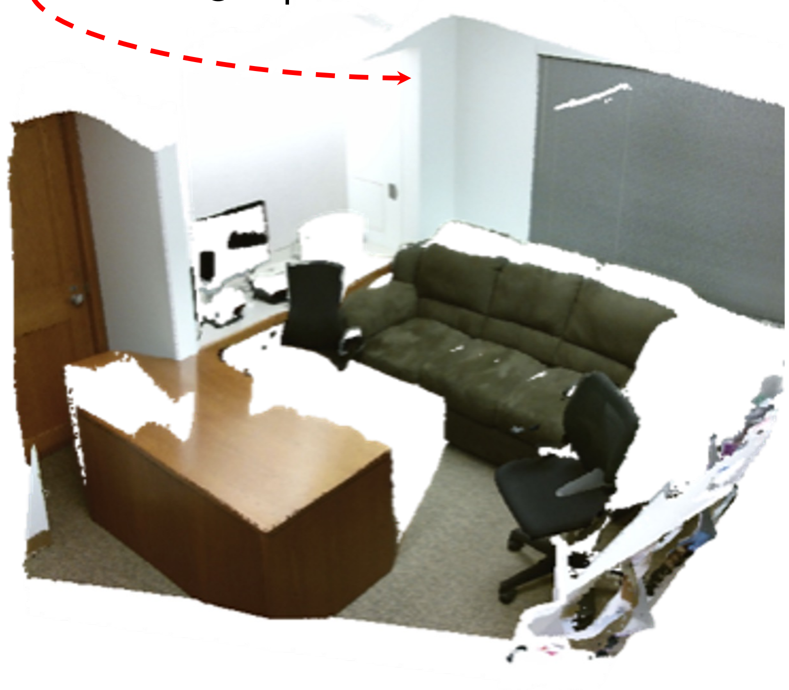


RGB-D Image Representation





3D point cloud: one pixel corresponding to one 3D point



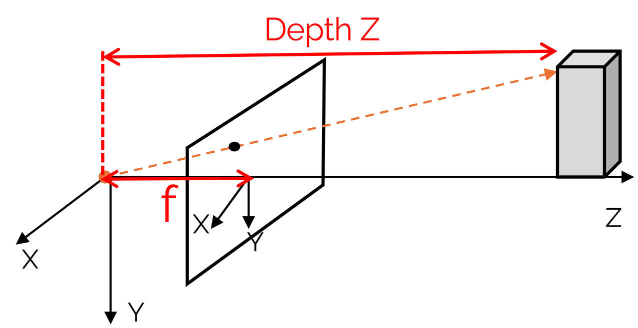
Depth Image → 3D Point Clouds:

A pixel with
- Image coordinate (u,v)
- Depth value = Z
- Focal focal length f

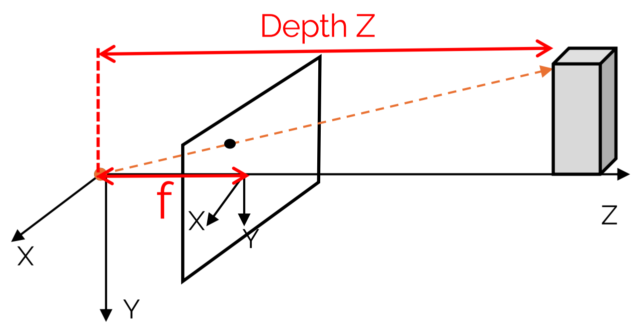
Its 3D location (X,Y,Z) in camera coordinate can be computed by:

$$X = \frac{u}{f} * Z, \qquad Y = \frac{v}{f} * Z$$

(Z : reading from depth image)



Summary:



3D point (X,Y,Z) ⟶ 2D image coordinate (u,v)

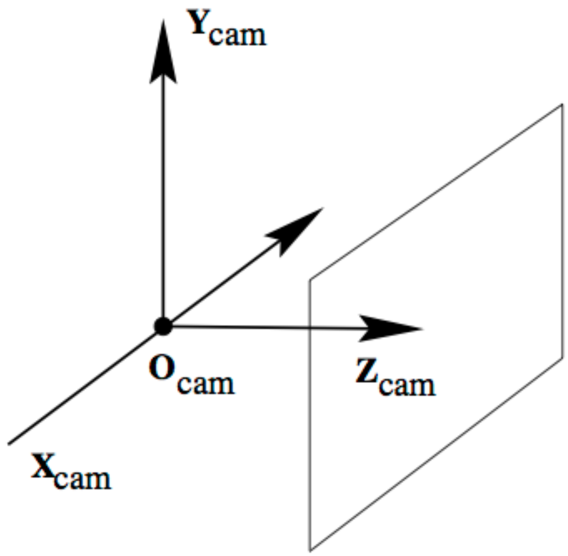
$$u = \frac{fX}{Z} \qquad v = \frac{fY}{Z}$$

Depth Image (u,v, Z) ⟶ 3D Point Clouds

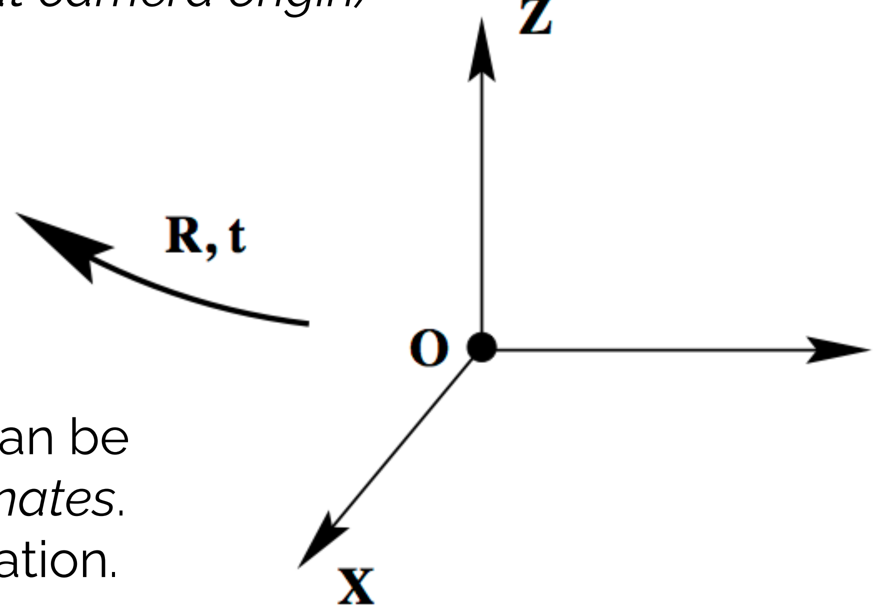
$$X = \frac{u}{f} * Z, \qquad Y = \frac{v}{f} * Z$$

(Z : reading from depth image)

World Coordinate to Camera Coordinate

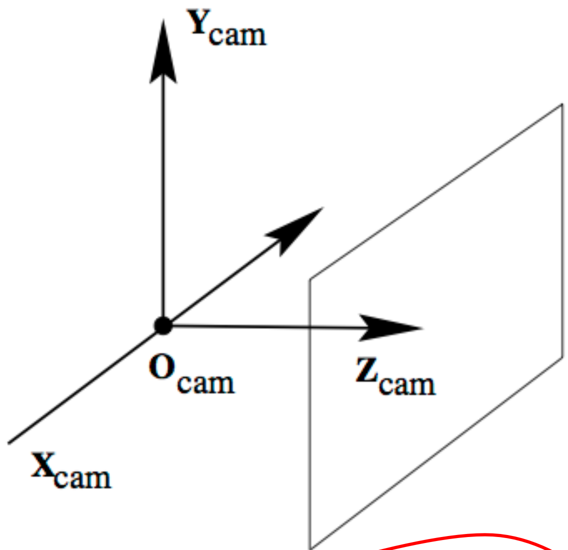


In order to apply the camera model we described so far, the 3D point (X,Y,Z) must be expressed in <u>*camera coordinates*</u> *(i.e., centered at camera origin)*



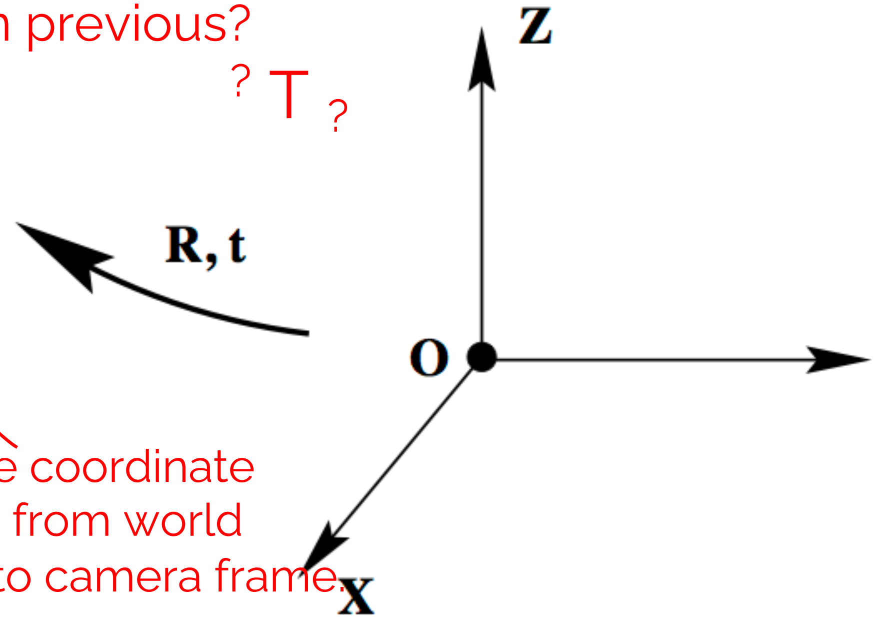
However, the world coordinate can be different from the *camera coordinates*. Requires an additional transformation.

World Coordinate to Camera Coordinate



How to representation this transformation?
Remember rigid transform in previous?

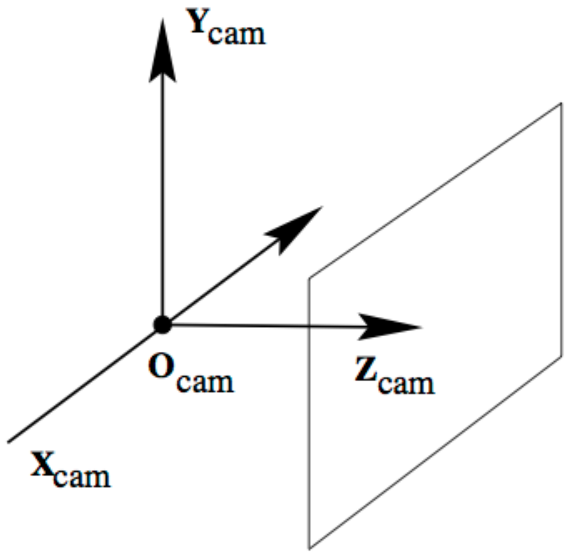
? T ?



$$\begin{pmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

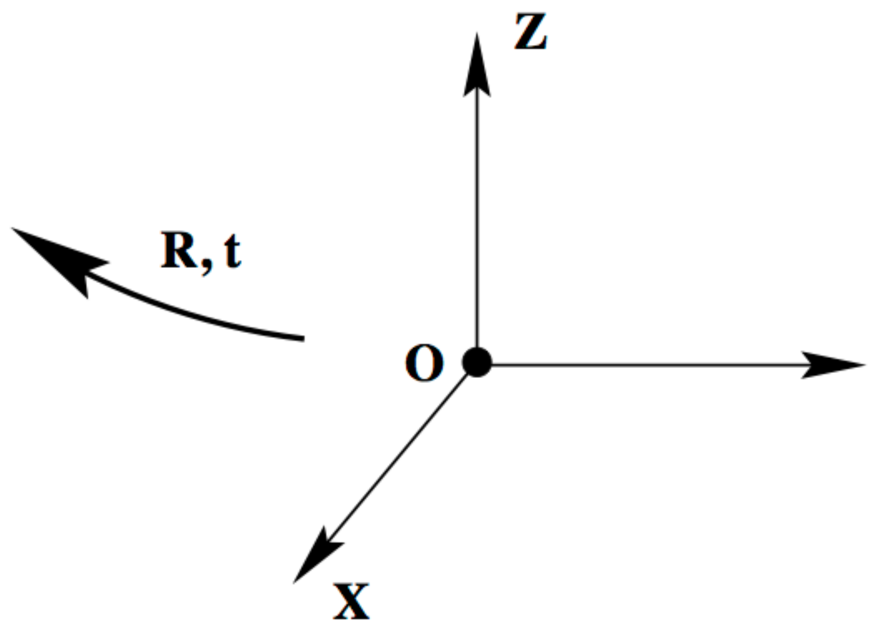
Change coordinate frames from world frame to camera frame.

World Coordinate to Camera Coordinate



$^{Camera}T_{world}$

Is this the pose of camera in world frame?



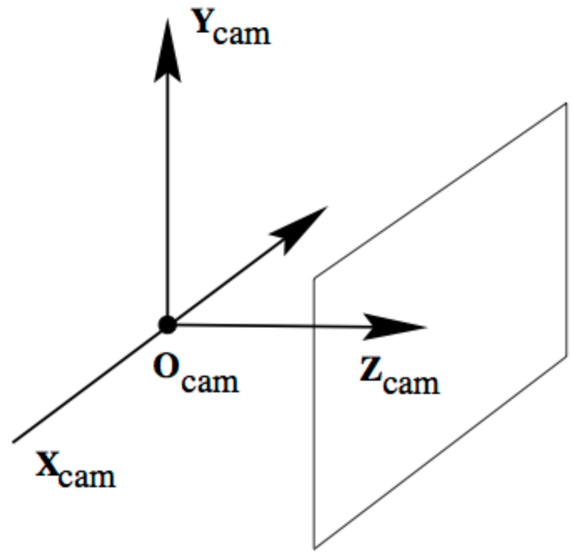
$$\begin{pmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

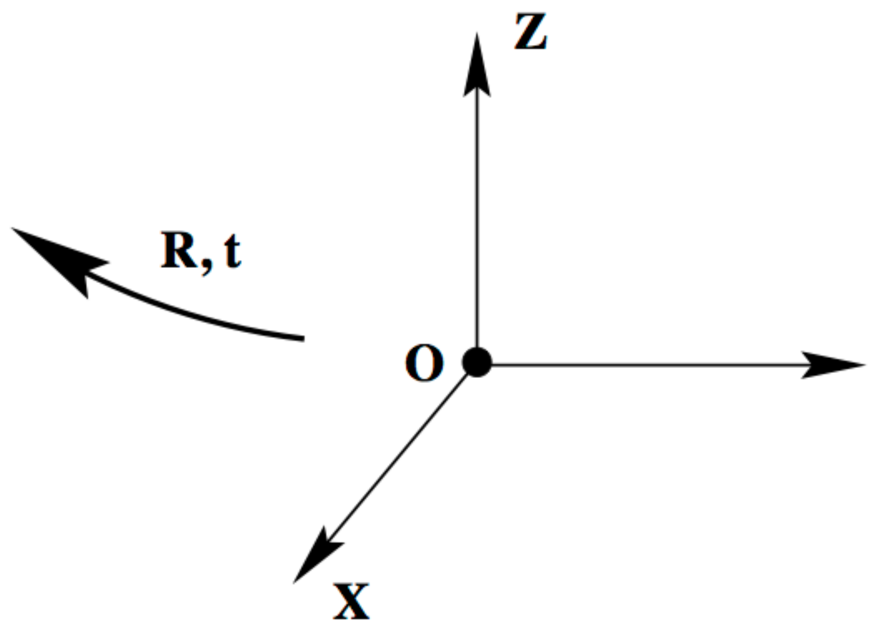
World Coordinate to Camera Coordinate

$^{Camera}T_{world}$

No, it is inverse of camera pose

Camera pose in world Frame should be $^{w}T_{c}$





$$\begin{pmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Camera: Putting everything together

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\mathbf{K}}_{\text{Intrinsic Matrix}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \underbrace{\underbrace{\left[\begin{array}{c|c} I & \boldsymbol{t} \\ \hline \mathbf{0} & 1 \end{array}\right]}_{\text{translation}} \times \underbrace{\left[\begin{array}{c|c} R & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right]}_{\text{rotation}}}_{\text{Camera Extrinsic Matrix [R|t]}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Camera: Putting everything together

Reduce Vector Dimension

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underset{\text{Intrinsic Matrix}}{\mathbf{K}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \underbrace{\underset{\text{translation}}{\left[\begin{array}{c|c} I & \boldsymbol{t} \\ \hline \mathbf{0} & 1 \end{array}\right]} \times \underset{\text{rotation}}{\left[\begin{array}{c|c} R & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right]}}_{\text{Camera Extrinsic Matrix [R|t]}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} fx & 0 & u_0 \\ 0 & fy & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Transforming 2D coordinate system.

We have been assuming fx = fy in the previous slides. However, in realworld camera they can be different

Camera: Putting everything together

$$
\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\underbrace{\mathbf{K}}_{\text{Intrinsic Matrix}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \underbrace{\underbrace{\left[\begin{array}{c|c} I & \boldsymbol{t} \\ \hline \mathbf{0} & 1 \end{array}\right]}_{\text{translation}} \times \underbrace{\left[\begin{array}{c|c} R & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right]}_{\text{rotation}}}_{\text{Camera Extrinsic Matrix [R|t]}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{\substack{\text{Camera Parameter} \\ \text{Camera Projection Matrix}}}
$$

$$
\mathbf{x} \overset{\mathbf{P}}{=} \mathbf{PX}
$$

Camera: Putting everything together

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} \mid \mathbf{t}\end{bmatrix}\mathbf{X}$$

- Map a 3D point X into a 2D coordinate in image x
- How to describe its *pose* in the world? (extrinsic matrix)
- How to describe its internal parameters? (intrinsic matrix)

That’s it for today!

Questions?