# Camera Model

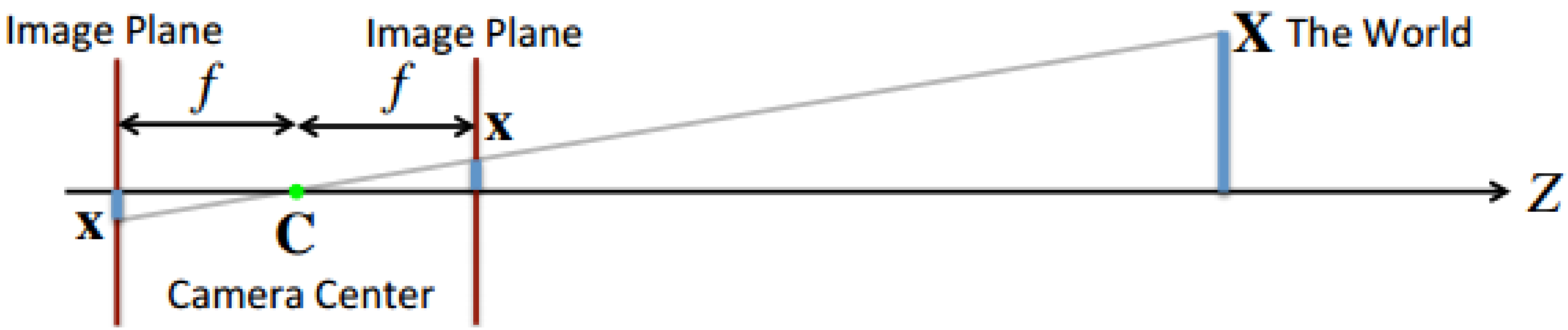
**Chul Min Yeum**
Assistant Professor
Civil and Environmental Engineering
University of Waterloo, Canada

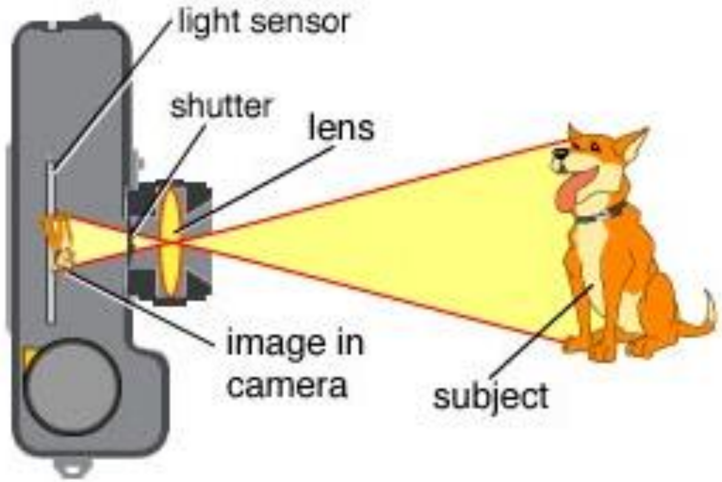
**CIVE 497 – CIVE 700: Smart Structure Technology**

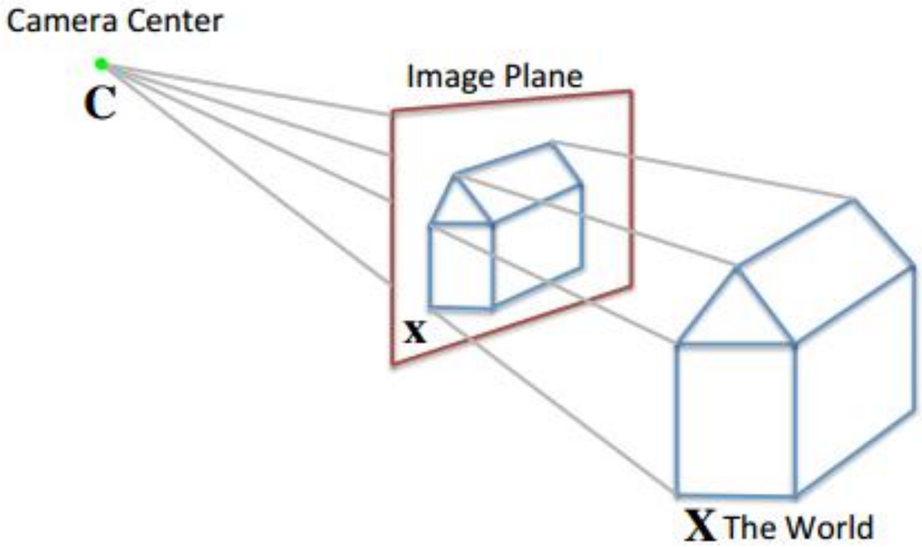
**Last updated: 2024-03-12**

UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING

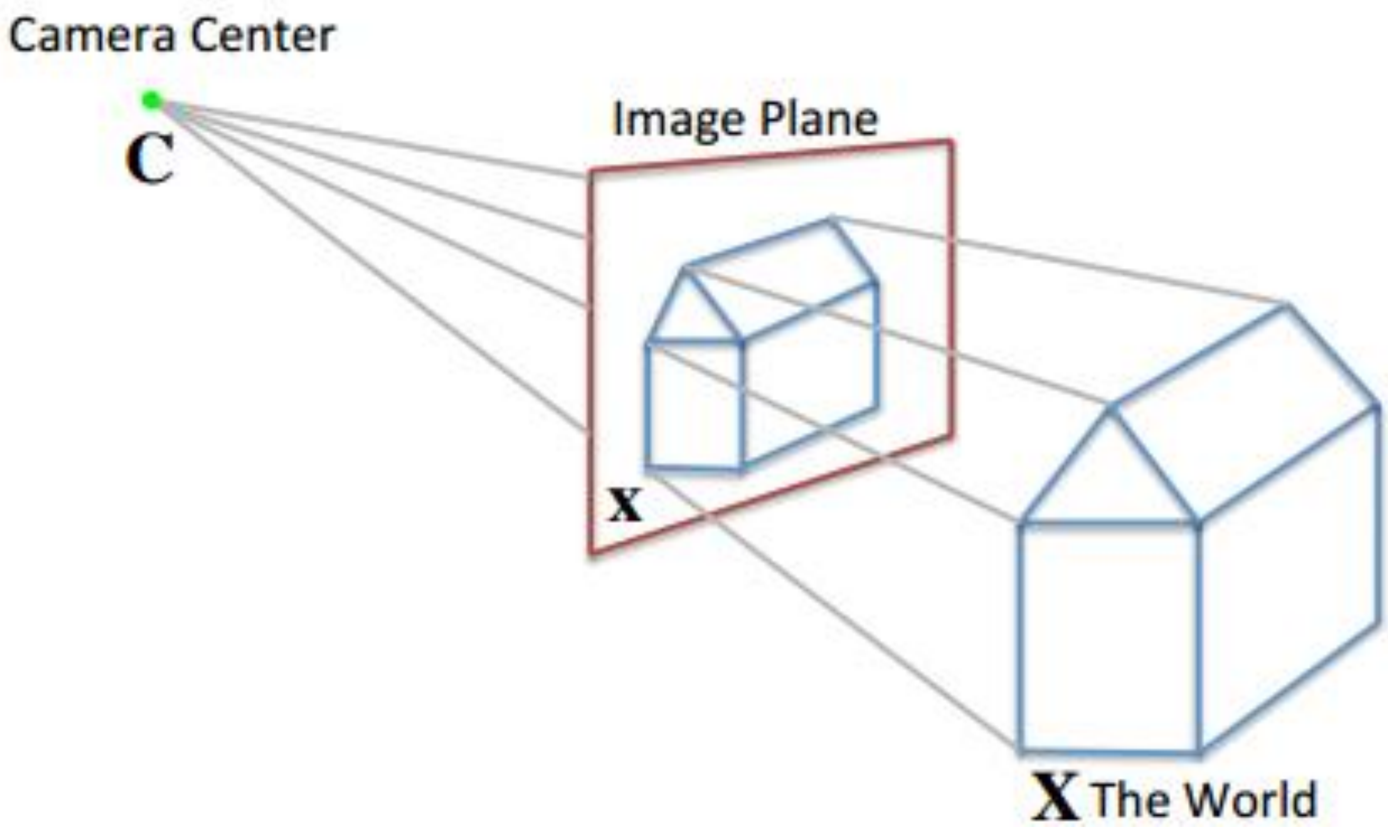
# (Review) Pinhole Camera Model



# How to Model Projection




**How to mathematically model the projection of 3D scenes on images**

An arbitrary homogeneous vector representative of a point is of the form $\mathbf{x} = (x_1, x_2, x_3)^T$, representing the point $(x_1/x_3, x_2/x_3)^T$ in $\mathbb{R}^2$.

Line equation, $ax + by + c = 0$, in $\mathbb{R}^2$ is represented as $\mathrm{l} = (a, b, c)^\intercal$ in the homogeneous coordinate.

# Representing Points in 3D

An arbitrary homogeneous vector representative of a point is of the form $\mathbf{x} = (x_1, x_2, x_3, x_4)^T$, representing the point $(x_1/x_4, x_2/x_4, x_3/x_4)^T$ in $\mathbb{R}^3$.

**Example)**

$$\mathbf{x_1} = \begin{pmatrix} 5 \\ 3 \\ 2 \\ 1 \end{pmatrix} \quad \mathbf{x_2} = \begin{pmatrix} 10 \\ 6 \\ 4 \\ 2 \end{pmatrix} \quad \mathbf{x_3} = \begin{pmatrix} 5k \\ 3k \\ 2k \\ k \end{pmatrix}, k \neq 0$$

**<u>up to a scale</u>**

$\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}$ and $\mathbf{x_4}$ indicate the same point of (5, 3, 2) in $\mathbb{R}^3$

# Representing Plane in 3D

$$\pi_1 x + \pi_2 y + \pi_3 z + \pi_4 = 0$$

Plane equation in $\mathbb{R}^3$

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3, \pi_4)^\intercal$$

Plane representation in HC

**Example)** $\boldsymbol{\pi}_1 = \begin{pmatrix} 2 \\ 1 \\ 4 \\ 3 \end{pmatrix}$ $\boldsymbol{\pi}_2 = \begin{pmatrix} 4 \\ 2 \\ 8 \\ 6 \end{pmatrix}$ $\boldsymbol{\pi}_3 = \begin{pmatrix} 2k \\ 1k \\ 4k \\ 3k \end{pmatrix}, k \neq 0$

$\boldsymbol{\pi}_1, \boldsymbol{\pi}_2$, and $\boldsymbol{\pi}_3$indicate the same plane of $2x + y + 4z + 3 = 0$ in $\mathbb{R}^3$

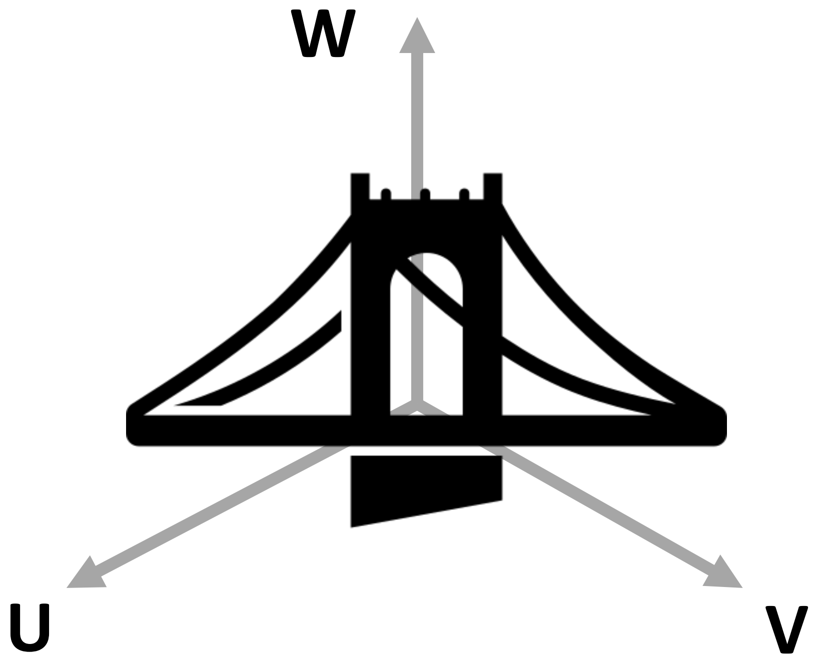
# Image Geometry (World Coordinate)

Object of interest in World

Coordinate System (U, V, W)

Null Island (0°N 0°E)

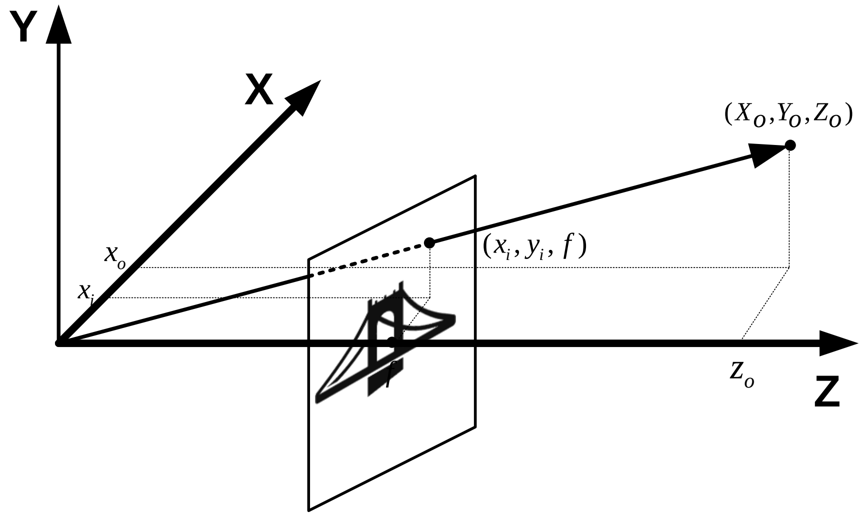

Custom coordinate



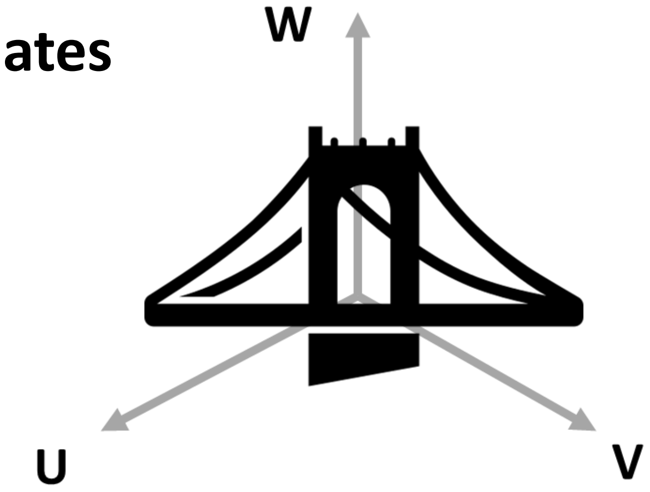

**Q.** Example of world coordinate?

# Image Geometry (Pixel Coordinate)

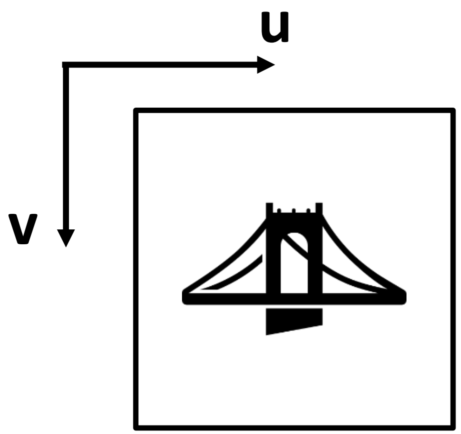
## Camera coordinates




## World coordinates

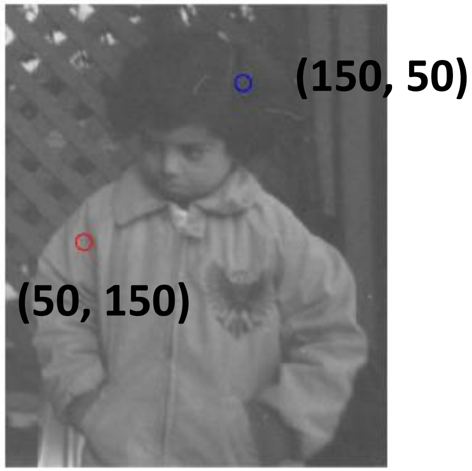



## Pixel coordinates




Our image gets digitized

into pixel coordinates (u,v)



# Forward Projection

| World Coordinates | | Camera Coordinates | | Pixel Coordinates |
|---|---|---|---|---|
| **U**<br>**V**<br>**W** | ⇨ | **X**<br>**Y**<br>**Z** | ⇨ | **u**<br>**v** |

<u>We will make a mathematical model to describe how 3D World points get projected into 2D pixel coordinates</u>

# Backward Projection

| World Coordinates | | Camera Coordinates | | Pixel Coordinates |
|---|---|---|---|---|
| **U** | | **X** | | **u** |
| **V** | ← | **Y** | ← | **v** |
| **W** | | **Z** | | |

Much of vision concerns trying to derive backward projection equations to recover 3D scene structure from images.

**Scene from images**

# Projection Matrix (Camera Matrix)

$$\begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

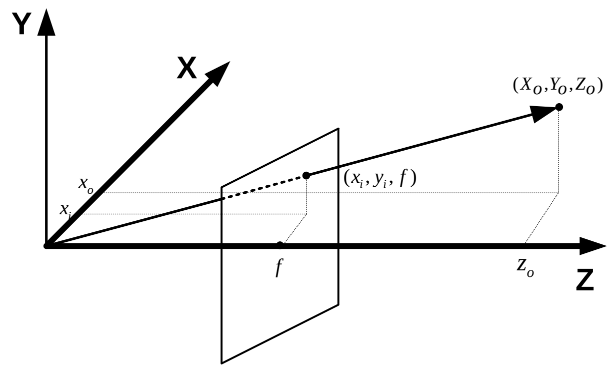
Homogenous coordinate system

$$\mathbf{x} = \mathbf{P_i X}$$

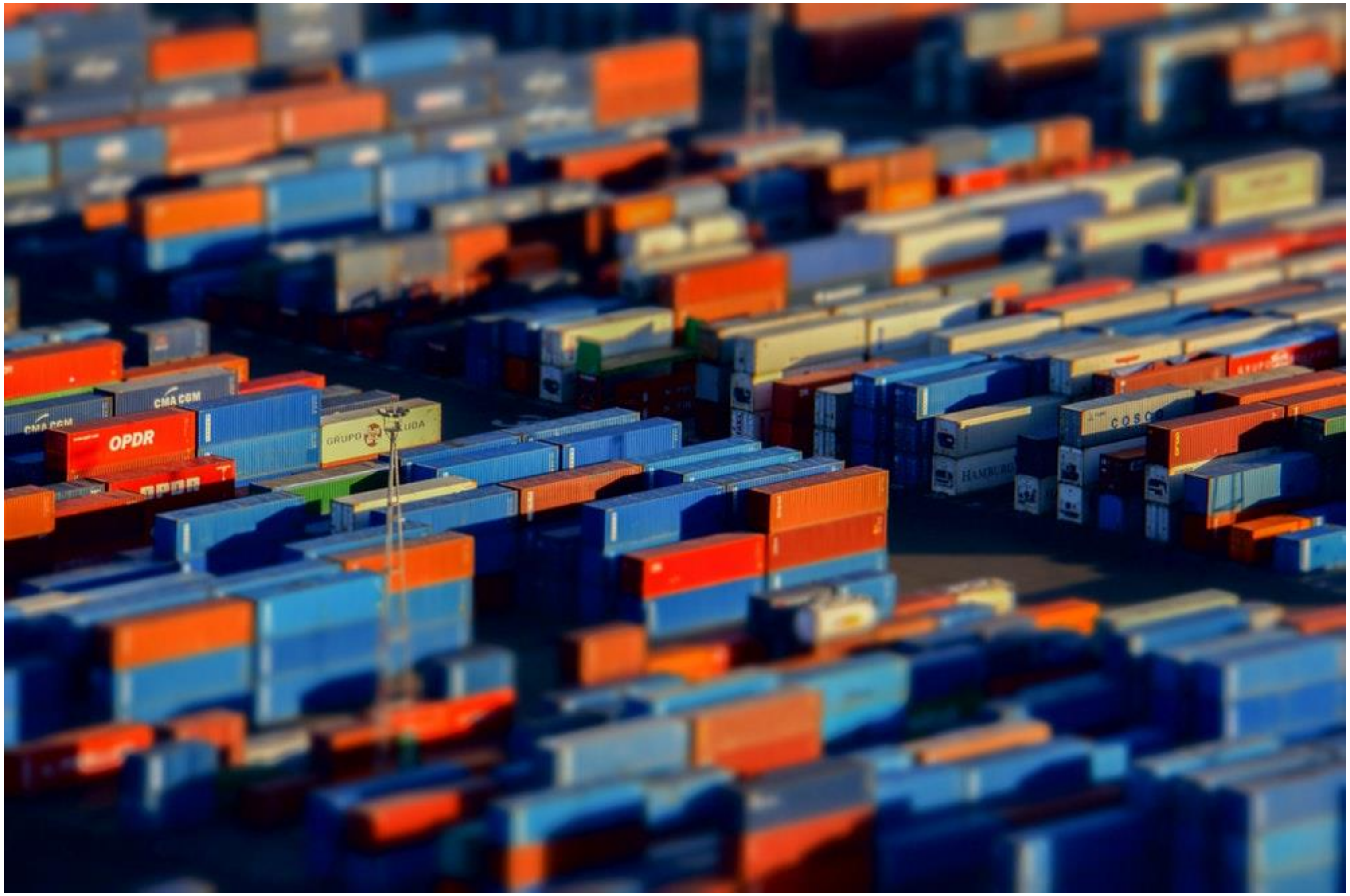
**2D point** **3D point**

- **If we knew a projection matrix in an image, we can compute the image point corresponding to the world point**
- **Although we knew an image points, we cannot find an unique world point.**

**Q.** Meaning of a 3 x 4 matrix?

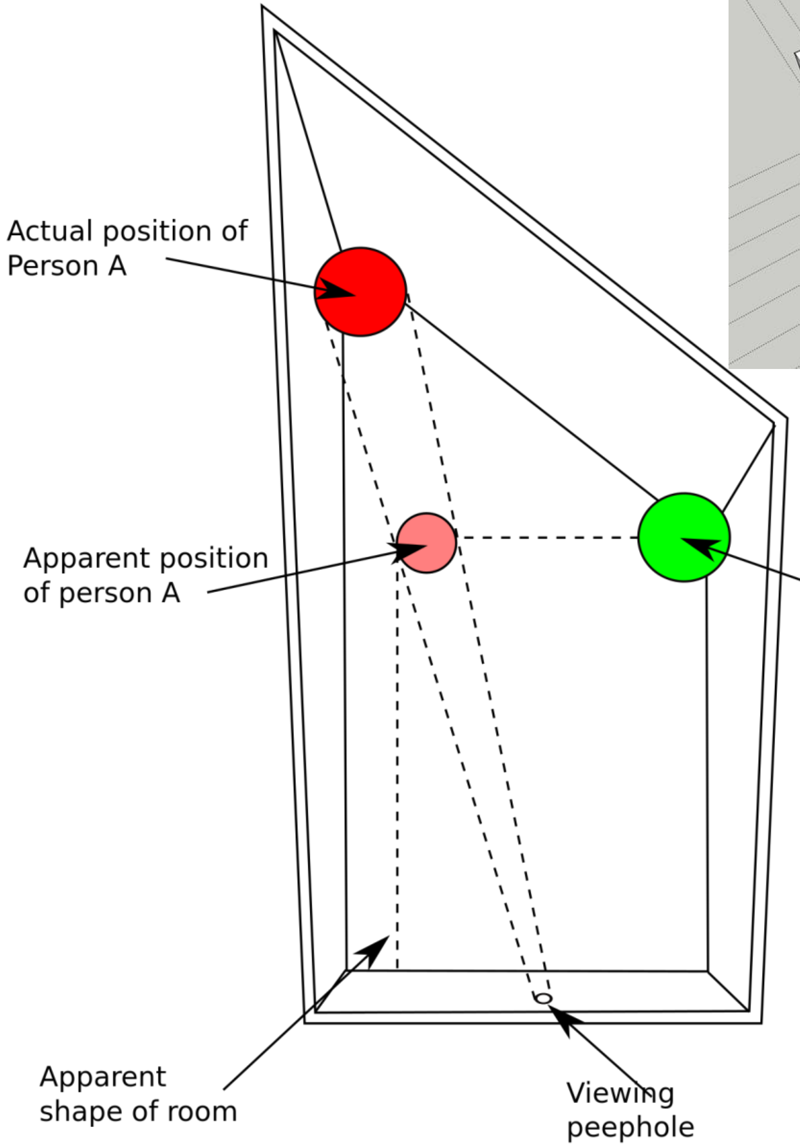


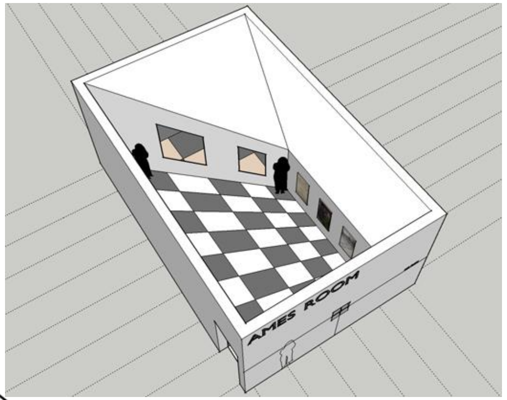
# Example: Unknown Scale



# Example: Ames room

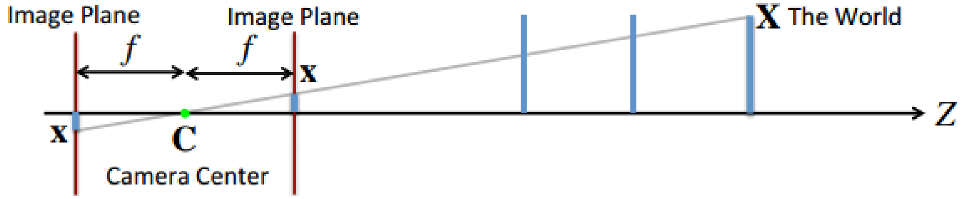
Supplement



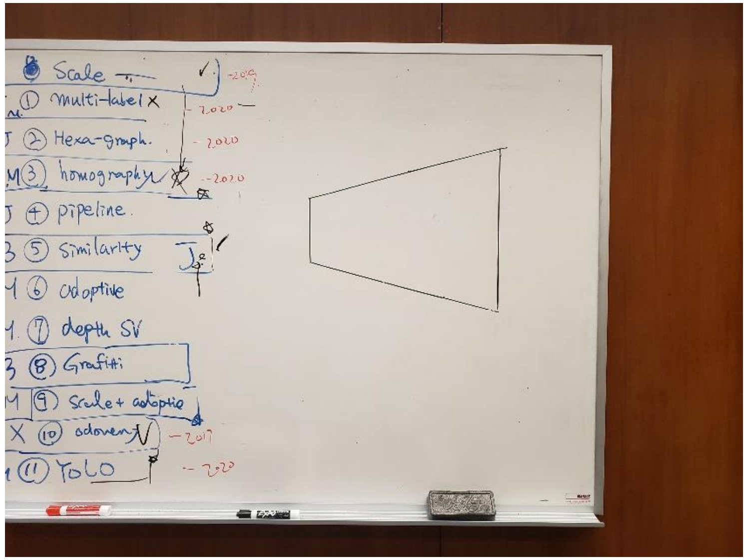

Smaller when farther away

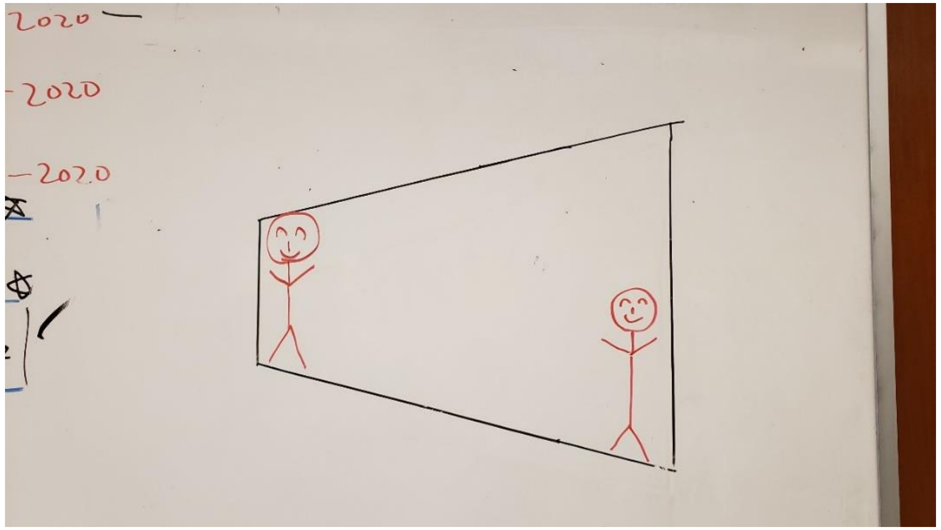
Illusion?

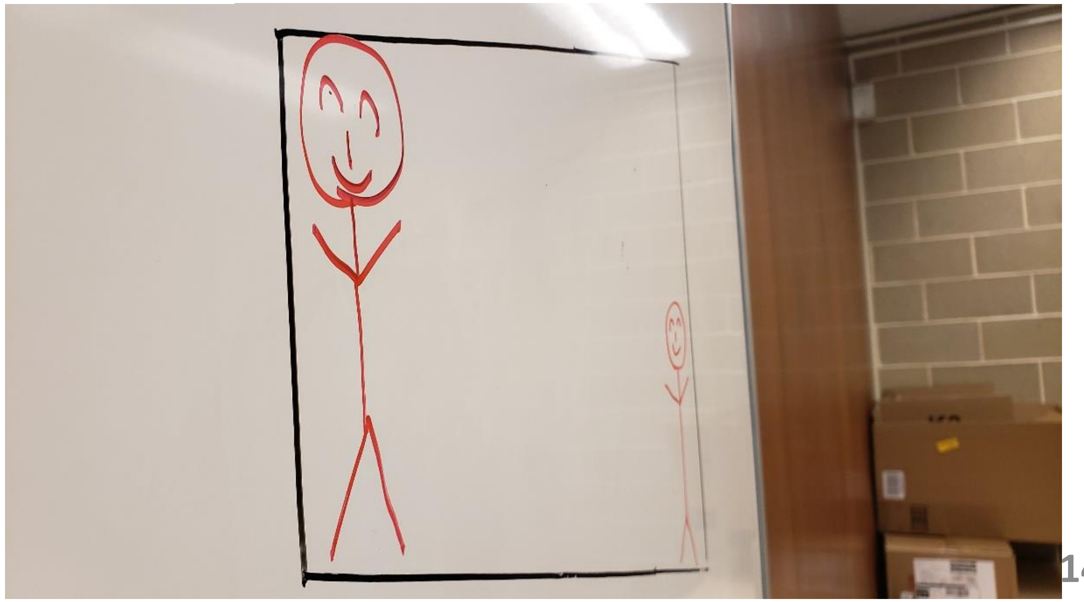


# Example: Chul Min’s Room

Supplement

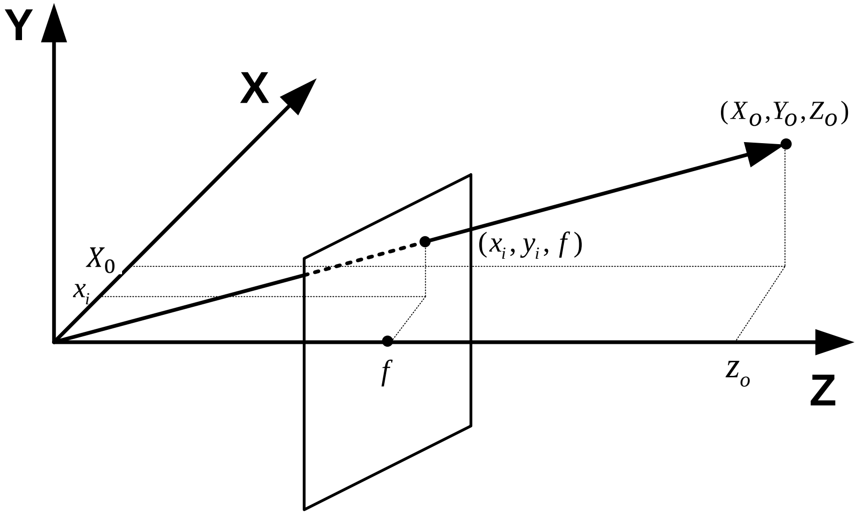


https://www.youtube.com/watch?v=54Oy75Bnu_Q

# Camera Model: A Simple Case

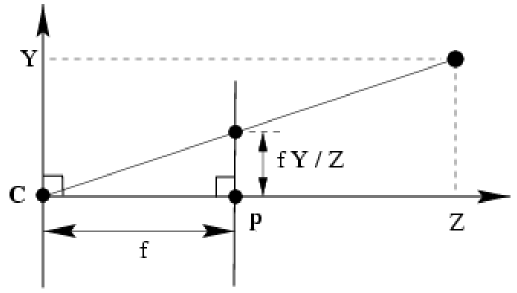



$$\frac{x_i}{f} = \frac{X_o}{Z_0} \quad , \quad \frac{y_i}{f} = \frac{Y_o}{Z_0}$$

$$x_i = f\frac{X_o}{Z_0} \quad , \quad y_i = f\frac{Y_o}{Z_0}$$

**Up to scale !!!**

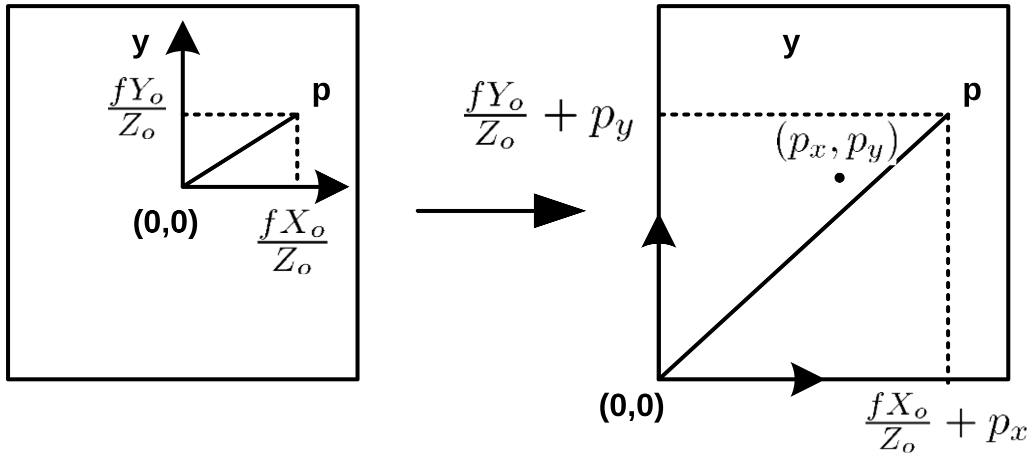
$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} fX_o \\ fY_o \\ Z_o \end{bmatrix} = \begin{bmatrix} \boxed{f} & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}$$




- Any point on the ray OP is projected onto a same image point.
- Only valid at homogeneous coordinate (up to scale)

**1 Unknown**

# Camera Model: Pixel Coordinate w.r.t. the Image Origin




$$x_i = f\frac{X_o}{Z_0} + p_x \quad , \quad y_i = f\frac{Y_o}{Z_0} + p_y$$

$$\begin{bmatrix} fX_o + Z_0 p_x \\ fY_o + Z_0 p_y \\ Z_o \end{bmatrix} = \begin{bmatrix} \boxed{f} & 0 & \boxed{p_x} & 0 \\ 0 & f & \boxed{p_y} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} = \mathbf{K} \left[ \ \mathbf{I} \ \middle| \ \vec{0} \ \right] \vec{X}$$

where the first three columns of the matrix form $\mathbf{K}$.

If the image center (principal point) is not the center of the image, **3 Unknowns**

# Physical Scale VS Pixel Scale

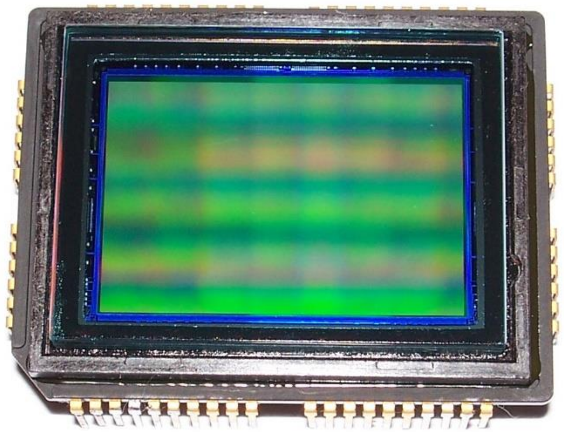
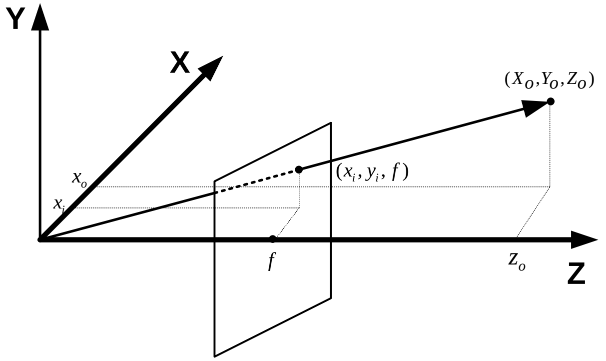


Image sensor




$$x_i = f\boxed{\frac{X_o}{Z_0}} + p_x \quad , \quad y_i = f\boxed{\frac{Y_o}{Z_0}} + p_y$$

**Non-dimensional number** **Non-dimensional number**

$$\begin{bmatrix} fX_o + Z_0 p_x \\ fY_o + Z_0 p_y \\ Z_o \end{bmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} = \mathbf{K} \left[ \ \mathbf{I} \ \middle| \ \vec{0} \ \right] \vec{X}$$

When the projection matrix is represented by a pixel scale, the image points are also represented by a pixel scale.

**Q.** What is the assumption?

Let's have $m_x$ pixel/unit length along X and $m_y$ pixel/unit length along Y (because the pixel may not be square, but most case, it is square)

$$\begin{bmatrix} I \\ J \\ 1 \end{bmatrix} = \begin{bmatrix} m_x x_i \\ m_y y_i \\ 1 \end{bmatrix} = \begin{bmatrix} m_x f X_o + Z_o m_x p_x \\ m_y f Y_o + Z_o m_y p_y \\ Z_o \end{bmatrix} = \begin{bmatrix} m_x f X_o + Z_o x_0 \\ m_y f Y_o + Z_o y_0 \\ Z_o \end{bmatrix} = \begin{bmatrix} m_x f & 0 & x_0 & 0 \\ 0 & m_y f & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}$$

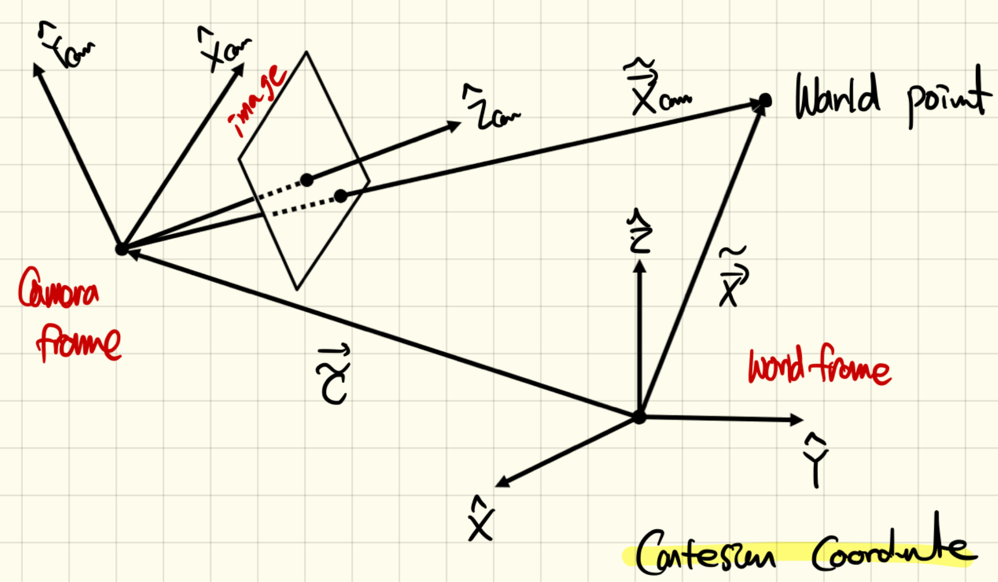
$$\begin{aligned} \vec{x} &= \mathbf{K} [\ \mathbf{I} \mid \vec{0}\ ]\, \vec{X} \\ &= \begin{bmatrix} m_x f & 0 & x_0 & 0 \\ 0 & m_y f & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \vec{X} \end{aligned}$$

We do not know physical locations where the point is mapped on the image sensor, but we know their pixel locations.

where $\mathbf{K} = \begin{bmatrix} m_x f & 0 & x_0 \\ 0 & m_y f & y_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boxed{\alpha} & 0 & \boxed{x_0} \\ 0 & \boxed{\beta} & \boxed{y_0} \\ 0 & 0 & 1 \end{bmatrix}$

**4 Unknowns**

# Camera Model : Full Configuration




$$\tilde{\vec{X}}_{cam} = R(\tilde{\vec{X}} - \tilde{\vec{C}})$$

R: (3x3) $\tilde{\vec{C}}$: 3x1 ①

$$\vec{X}_{cam} = \begin{pmatrix} \tilde{\vec{X}}_{cam} \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{\vec{C}} \\ \vec{0}^T & 1 \end{bmatrix} \begin{pmatrix} \tilde{\vec{X}} \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{\vec{C}} \\ \vec{0}^T & 1 \end{bmatrix} \vec{X}$$

$$\vec{x}_{cam} = K[I \mid \vec{0}]\, \vec{X}_{cam} = K[I \mid \vec{0}] \begin{bmatrix} R & -R\tilde{\vec{C}} \\ \vec{0}^T & 1 \end{bmatrix} \vec{X}$$

$\tilde{\vec{X}}_{cam} = R\tilde{\vec{X}}$ World frame → Camera frame. Rotation

$\tilde{\vec{X}}_{cam} = \tilde{\vec{X}} - \tilde{\vec{C}}$ World frame → Camera frame Translation.

We move the origin first and rotate the axes.

②

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & *^1 \\ R_{21} & R_{22} & R_{23} & *^2 \\ R_{31} & R_{32} & R_{33} & *^3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} R_{11} & R_{12} & R_{13} & *^1 \\ R_{21} & R_{22} & R_{23} & *^2 \\ R_{31} & R_{32} & R_{33} & *^3 \end{bmatrix} = [R \mid -R\tilde{\vec{C}}]$$

where $\begin{pmatrix} *^1 \\ *^2 \\ *^3 \end{pmatrix} = -R\tilde{\vec{C}}$

$$\vec{x}_{cam} = K[R \mid -R\tilde{\vec{C}}]\,\vec{X}$$
$$= KR[I \mid -\tilde{\vec{C}}]\,\vec{X}$$

# Camera Model : Full Configuration (Continue)

Translation

$$\vec{x}_{cam} = \mathbf{K}\,[\,\mathbf{I}\,|\,\vec{0}\,]\,\vec{X}$$
$$= \mathbf{K}\,[\,\mathbf{I}\,|\,\vec{0}\,]\,(\tilde{X} - \vec{C})$$

Translation & rotation

$$\vec{x}_{cam} = \mathbf{K}\,[\,\mathbf{I}\,|\,\vec{0}\,]\,\mathbf{R}\vec{X}$$
$$= \mathbf{K}\,[\,\mathbf{I}\,|\,\vec{0}\,]\,\mathbf{R}(\tilde{X} - \vec{C})$$

$$\vec{x}_{cam} = KR\,[\,I_{3\times3}\,|\,-\tilde{\vec{C}}\,]\,\vec{X}$$

Pixel Coordinate (homogeneous)

Camera Internal parameter

World to Camera rotation (physical)

World to Camera translation (physical)

World point (homogeneous)

$$\vec{x}_{cam} = P\vec{X}.$$
$$= [p_1\ p_2\ p_3\ p_4]\vec{X}.$$

- **Rotation matrix (3)**
- **Camera center (3)**
- **Focal length (1~2)**
- **Principal points (2)**

**9 (10) Unknowns**

$$\mathbf{x} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\mathbf{X}$$

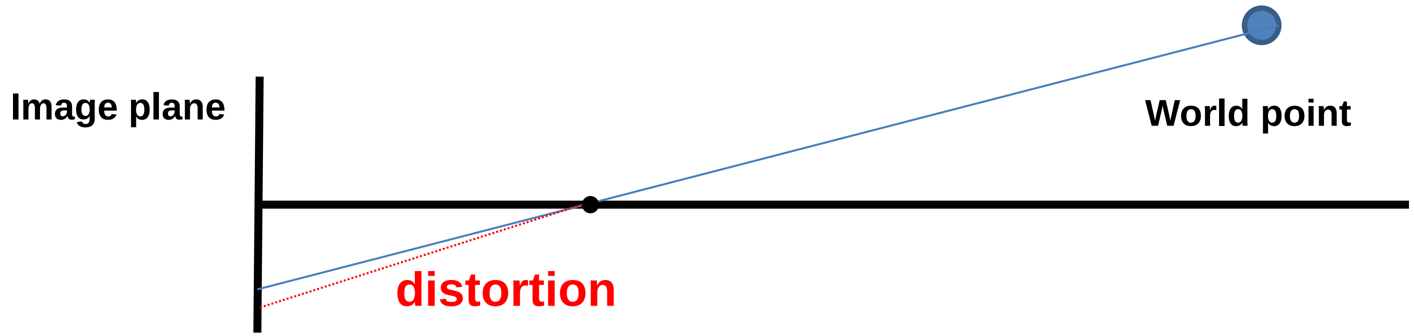
$$\begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

# Radial distortion (due to optics of the lens)

$$r^2 = \|\mathbf{x}\|^2 = x^2 + y^2$$

$$\mathbf{x}' = (1 + \boxed{k_1} r^2 + \boxed{k_2} r^4)\mathbf{x}$$

before after




**9 (10) + # of distortion parameters unknowns**

# Radial Distortion Model in MATLAB

Supplement

Any real world point $P(X,Y,Z)$ can be defined with respect to some 3-D world origin.

Relative to a camera lens, this 3-D point can be defined as p0, which is obtained by rotating and translating P.

$$p_0 = (x_0, y_0, z_0) = RP + t$$

The 3-D point p (0) is then projected into the camera's image plane as a 2D point, (x (1) , y (1) ).

$$x_1 = \frac{x_0}{z_0}, \; y_1 = \frac{y_0}{z_0}$$

When a camera captures an image, it does not precisely capture the real points, but rather a slightly distorted version of the real points which can be denoted (x (2) , y (2) ). The distorted points can be described using the following function:

$$x_2 = x_1\left(1 + k_1 r^2 + k_2 r^4\right) + 2p_1 x_1 y_1 + p_2\left(r^2 + 2{x_1}^2\right)$$

$$y_2 = y_1\left(1 + k_1 r^2 + k_2 r^4\right) + 2p_2 x_1 y_1 + p_1\left(r^2 + 2{y_1}^2\right)$$

where:

k (1) , k (2) = radial distortion coefficients of the lens

p (1) , p (2) = tangential distortion coefficients of the lens

r = $\sqrt{{x_1}^2 + {y_1}^2}$

https://www.mathworks.com/help/symbolic/examples/developing-an-algorithm-for-undistorting-an-image.html

# Review All Terms

- Rotation matrix (3)
- Translation matrix (3)
- Focal length (2)
- Principal points (2)
- Lens distortion parameters (2~4)

Exterior (External)

Interior (Internal)

# Example: Pix4D

| | |
|---|---|
| Camera Model Name | Field used to type the camera model name. It is recommended to type the name as follow:<br>*camera_name_focal_length_sensor_widthxsensor_height* |
| Image Width [pixel] | Image width in pixels. |
| Image Height [pixel] | Image height in pixels. |
| Focal Length [pixel] | Focal length in pixels (if defined in pixels, the mm value is automatically computed and added to the corresponding field). |
| Principal Point x [pixel] | Principal point x coordinate in pixels (if defined in pixels, the mm value is automatically computed and added to the corresponding field). |
| Principal Point y [pixel] | Principal point y coordinate in pixels (if defined in pixels, the mm value is automatically computed and added to the corresponding field). |
| Sensor Width [mm] | Sensor width in mm. |
| Sensor Height [mm] | Sensor height in mm. |
| Pixel Size [μm] | Pixel size in μm. |
| Focal Length [mm] | Focal length in mm (if defined in mm, the pixel value is automatically computed and added to the corresponding field). |
| Principal Point x [mm] | Principal point x coordinate in mm (if defined in mm, the pixel value is automatically computed and added to the corresponding field). |
| Principal Point y [mm] | Principal point y coordinate in mm (if defined in mm, the pixel value is automatically computed and added to the corresponding field). |
| Radial Distortion R1 | Radial distortion R1 parameter of the lens (optional, it is recommended to leave the distortion parameters to 0). |
| Radial Distortion R2 | Radial distortion R2 parameter of the lens (optional, it is recommended to leave the distortion parameters to 0). |
| Radial Distortion R3 | Radial distortion R3 parameter of the lens (optional, it is recommended to leave the distortion parameters to 0). |
| Tangential Distortion T1 | Tangential distortion T1 parameter of the lens (optional, it is recommended to leave the distortion parameters to 0). |
| Tangential Distortion T2 | Tangential distortion T2 parameter of the lens (optional, it is recommended to leave the distortion parameters to 0). |

# Slide Credits and References


- Lecture notes: Robert Collins
- Hartley, Richard, and Andrew Zisserman. Multiple view geometry in computer vision. Cambridge university press, 2003.