Camera Model

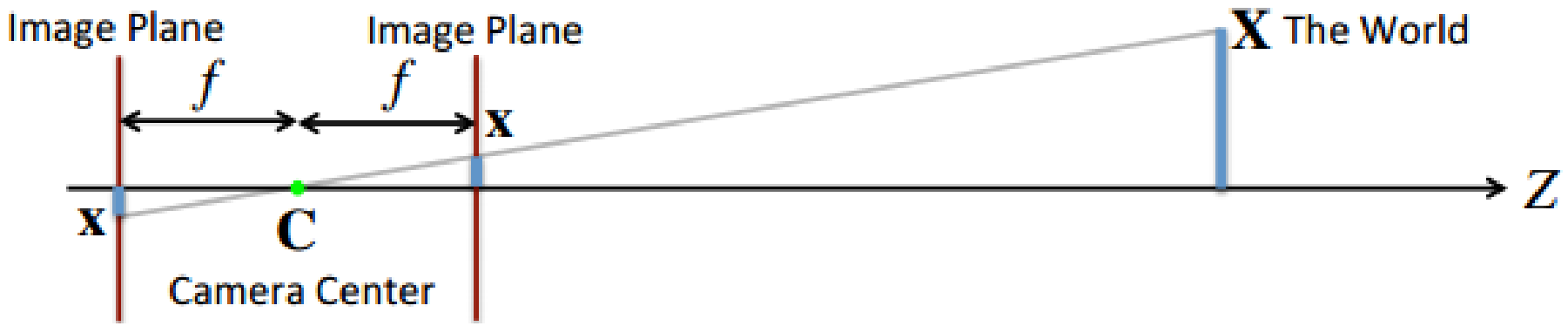
Chul Min Yeum
Assistant Professor
Civil and Environmental Engineering
University of Waterloo, Canada

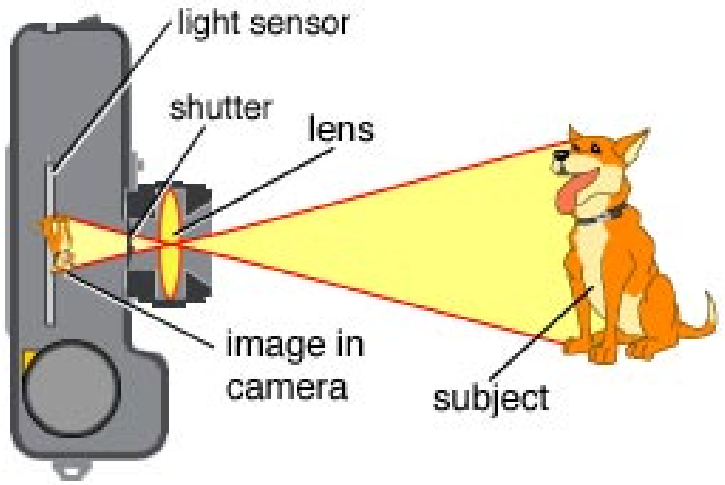
CIVE 497 – CIVE 700: Smart Structure Technology

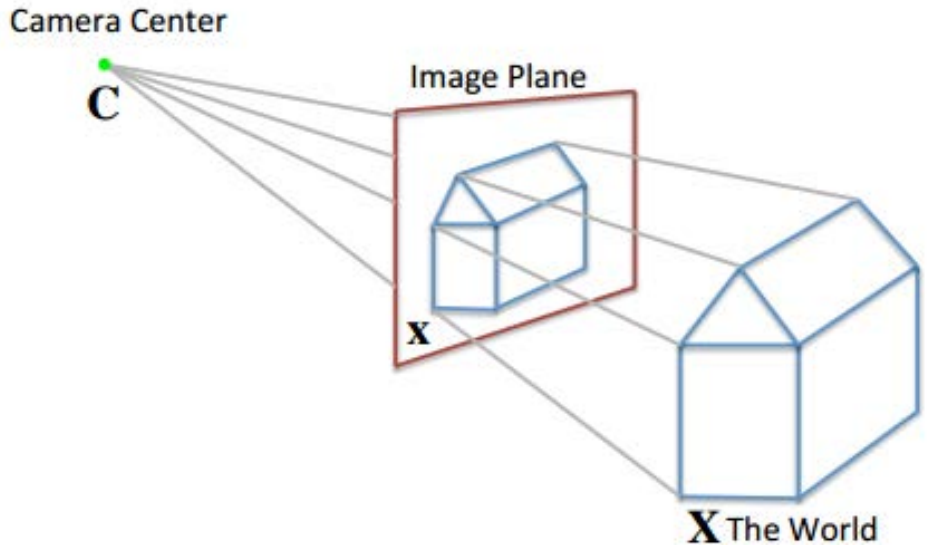
UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING

Last updated: 2019-03-07

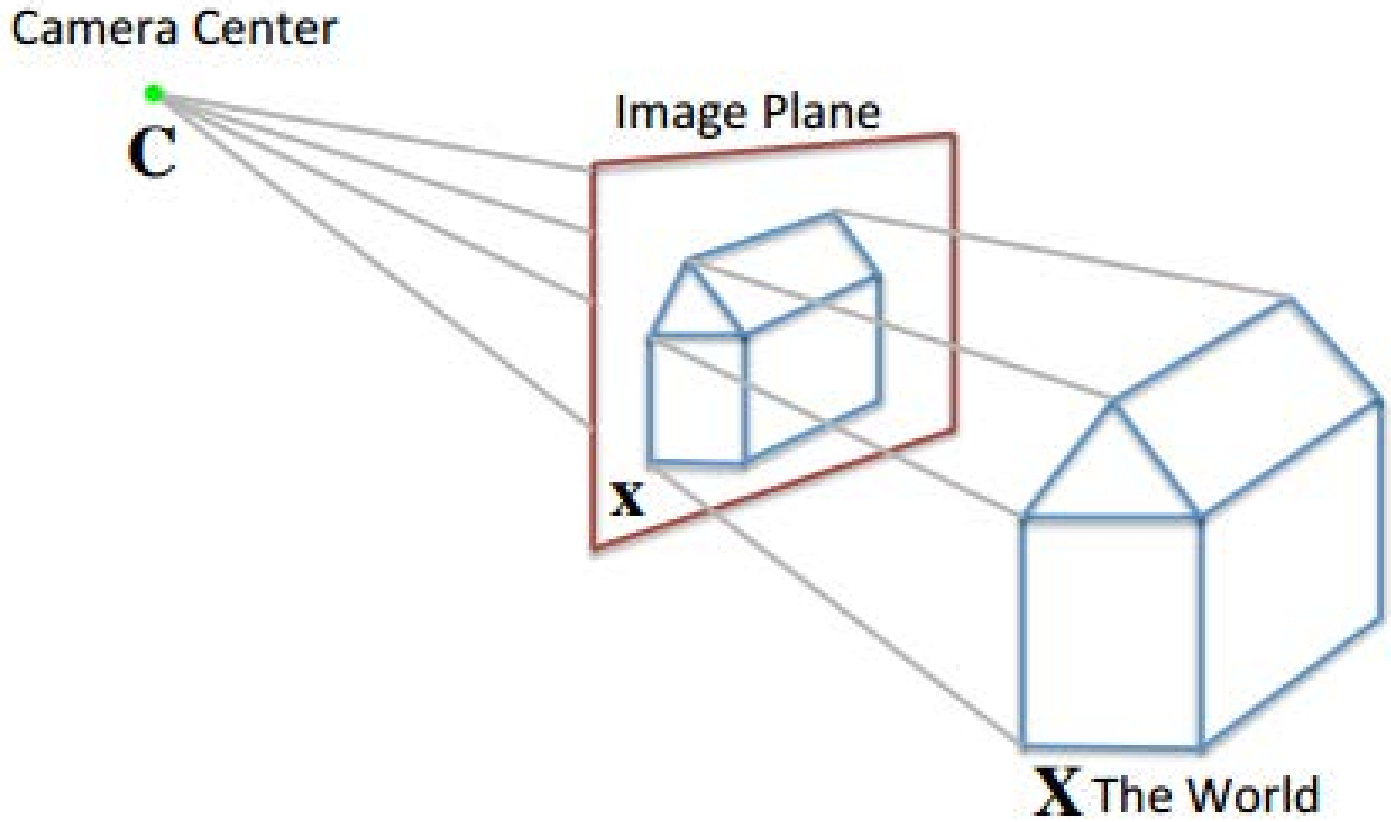
(Review) Pinhole Camera Model





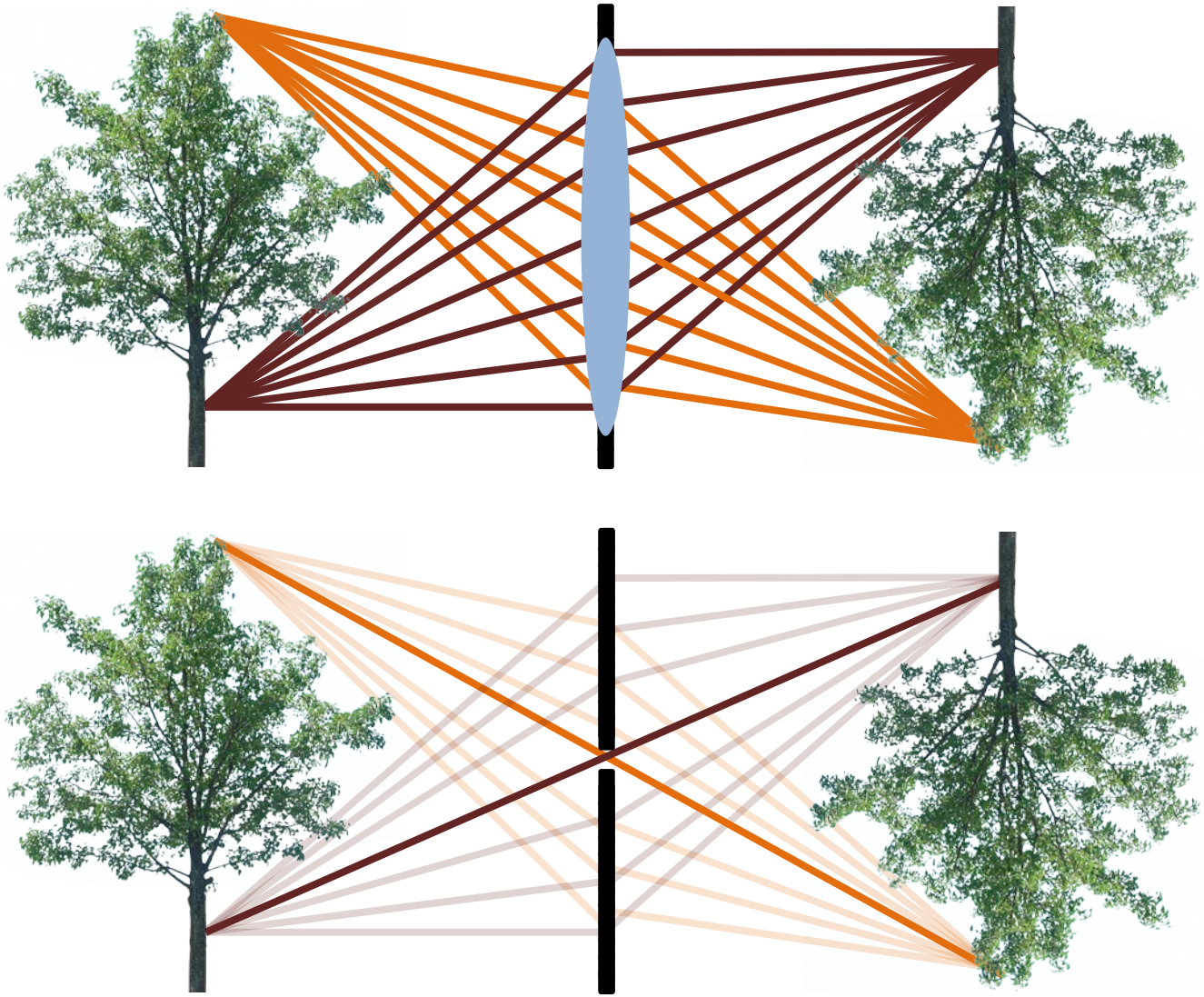


How to Model Projection



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

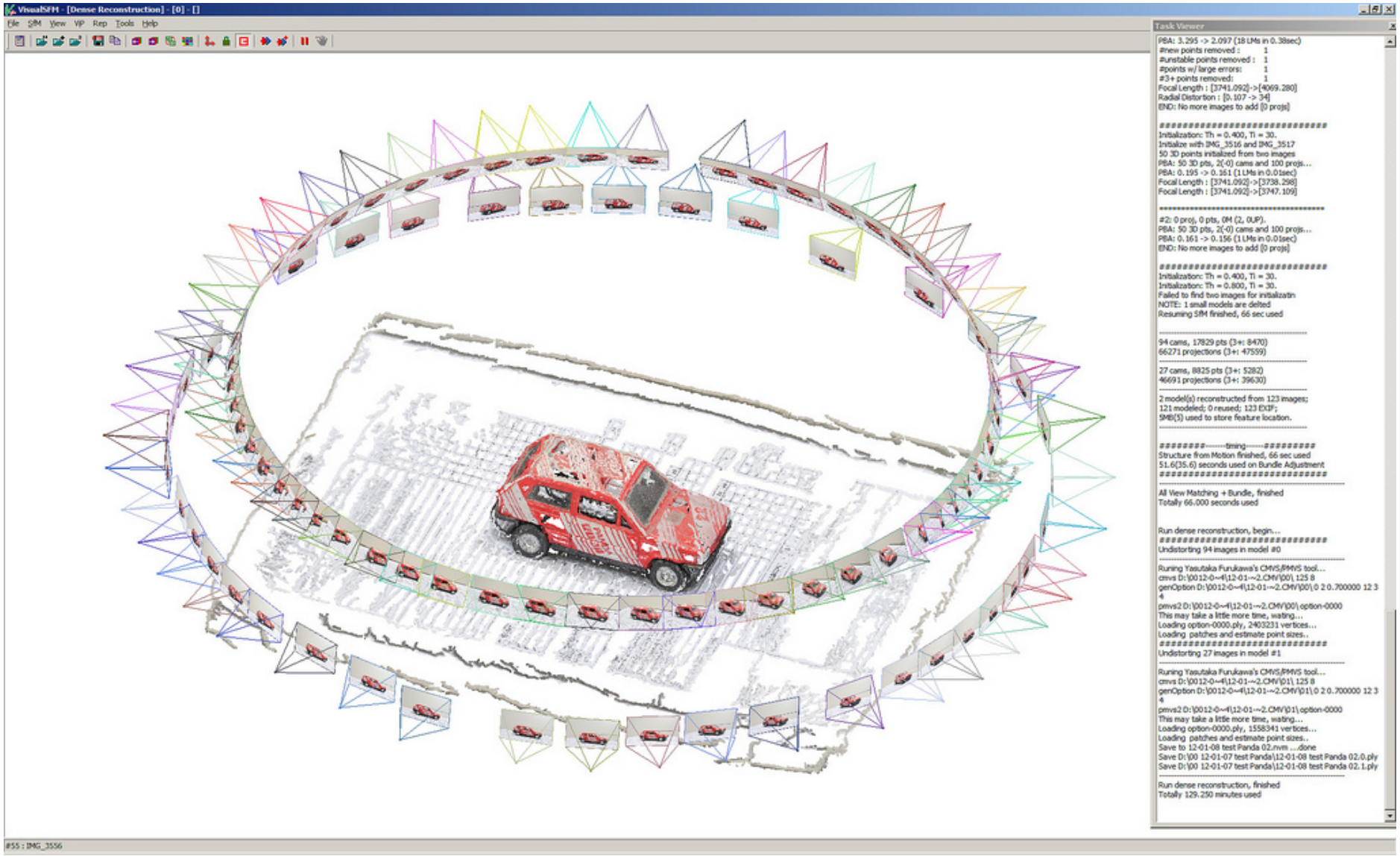
(Review) Both Lens and Pinhole Camera



We can derive properties and descriptions that hold for both camera models if:

- We use only central rays.
- We assume the lens camera is in focus.
- We assume that the focus distance of the lens camera is equal to the focal length of the pinhole camera.

Camera Representation



(Review) Homogeneous Coordinate

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) $\mathbf{x_1} = \begin{pmatrix} 5 \\ 3 \\ 2 \\ 1 \end{pmatrix}$ $\mathbf{x_2} = \begin{pmatrix} 10 \\ 6 \\ 4 \\ 2 \end{pmatrix}$ $\mathbf{x_3} = \begin{pmatrix} 5k \\ 3k \\ 2k \\ k \end{pmatrix}, k \neq 0$ **up to a scale**

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

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

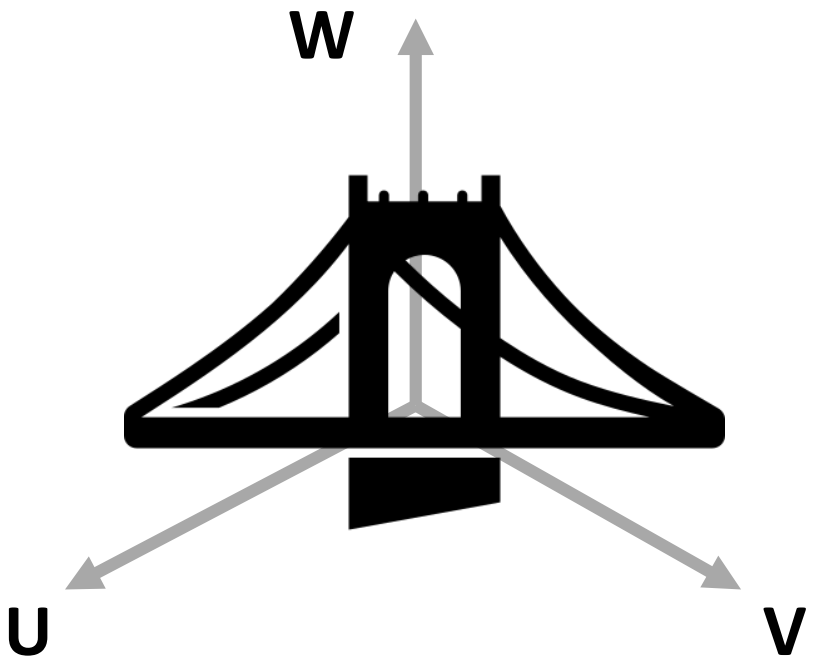
Example) $\mathbf{M_1} = \begin{pmatrix} 2 \\ 1 \\ 4 \\ 3 \end{pmatrix}$ $\mathbf{M_2} = \begin{pmatrix} 4 \\ 2 \\ 8 \\ 6 \end{pmatrix}$ $\mathbf{M_3} = \begin{pmatrix} 2k \\ 1k \\ 4k \\ 3k \end{pmatrix}, k \neq 0$

$\mathbf{M_1}, \mathbf{M_2}$, and $\mathbf{M_3}$indicate the same plane of $2x + y + 4z + 3 = 0$ in $\mathbb{R}^3$

Image Geometry (World Coordinate)

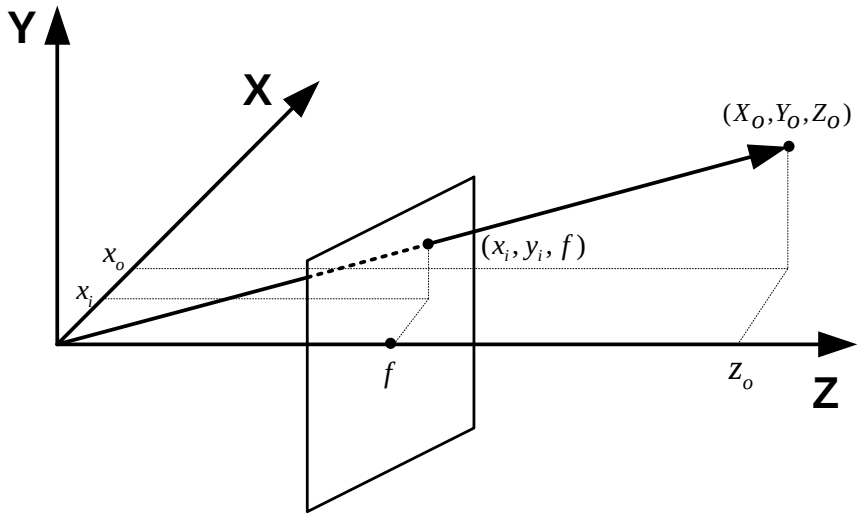
Object of interest in World

Coordinate System (U, V, W)

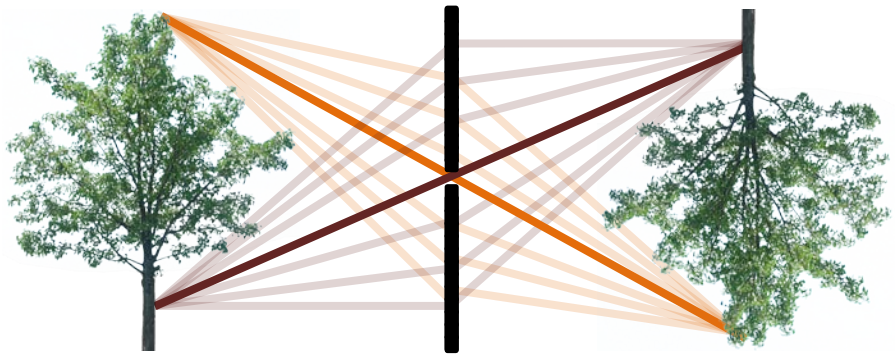


example of world coordinate?

Image Geometry (Camera Coordinate)



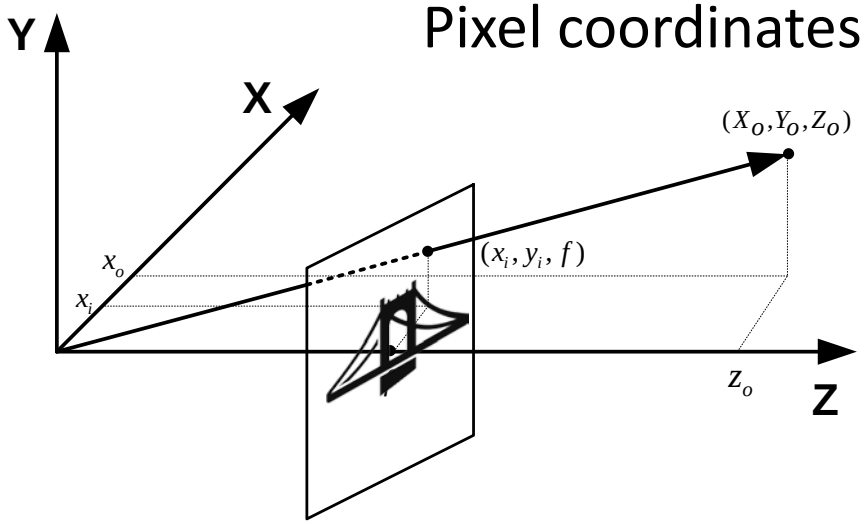
Camera Coordinate System (X, Y, Z)



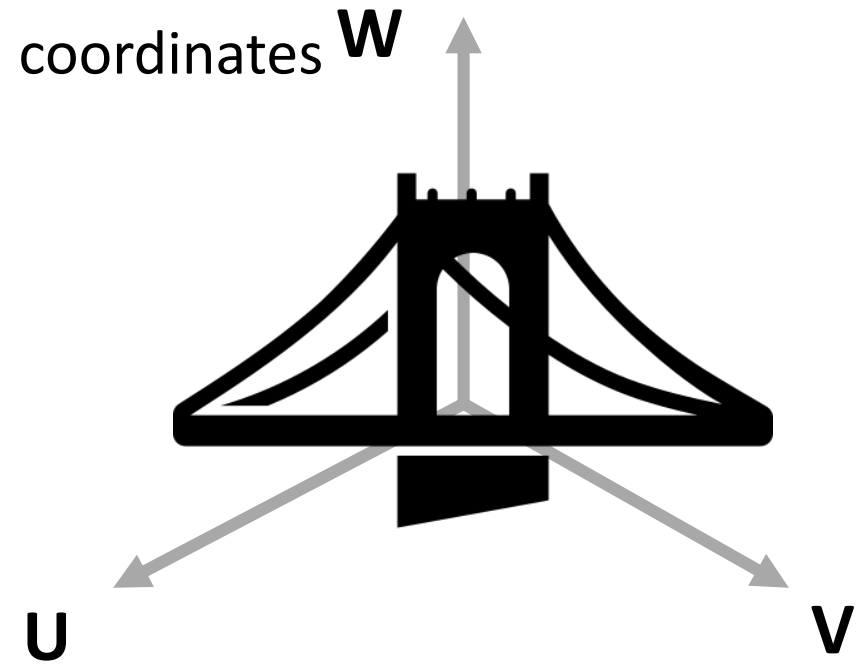
Film Coordinate system (x, y)

Image Geometry (Pixel Coordinate)

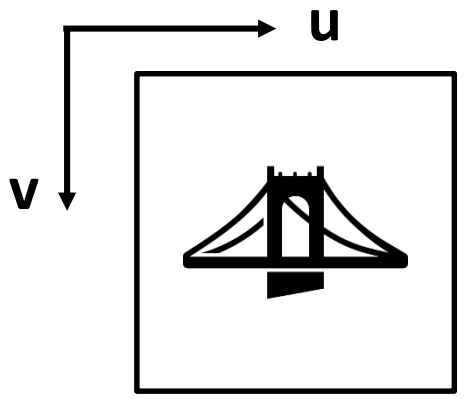
Camera coordinates



World coordinates

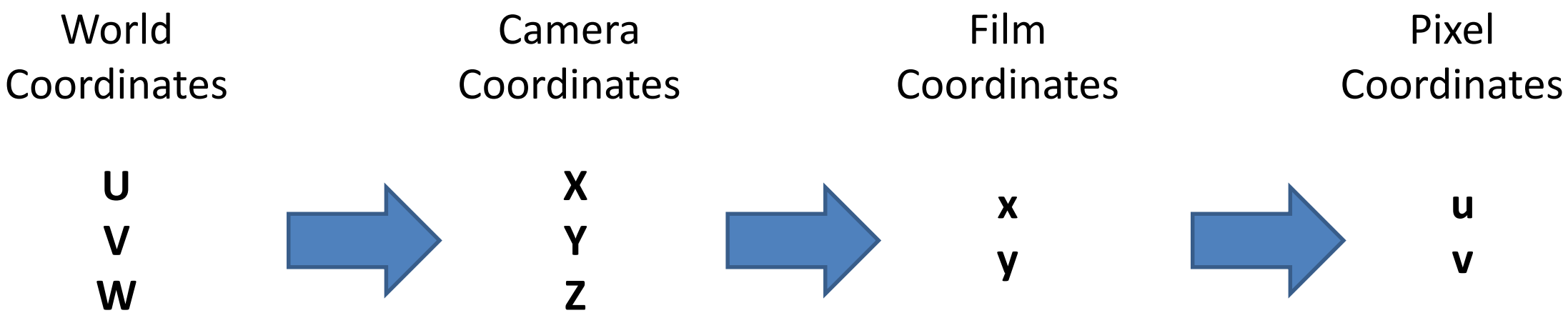


Pixel coordinates



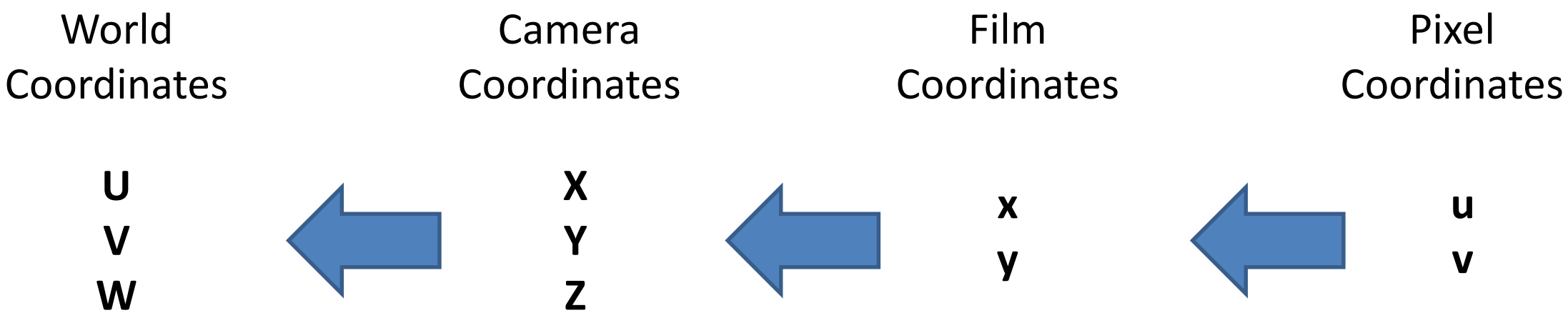
Pixel coordinates: Our image gets digitized into pixel coordinates (u,v)

Forward Projection



We want a mathematical model to describe how 3D World points get projected into 2D pixel coordinates

Backward Projection



Much of vision concerns trying to

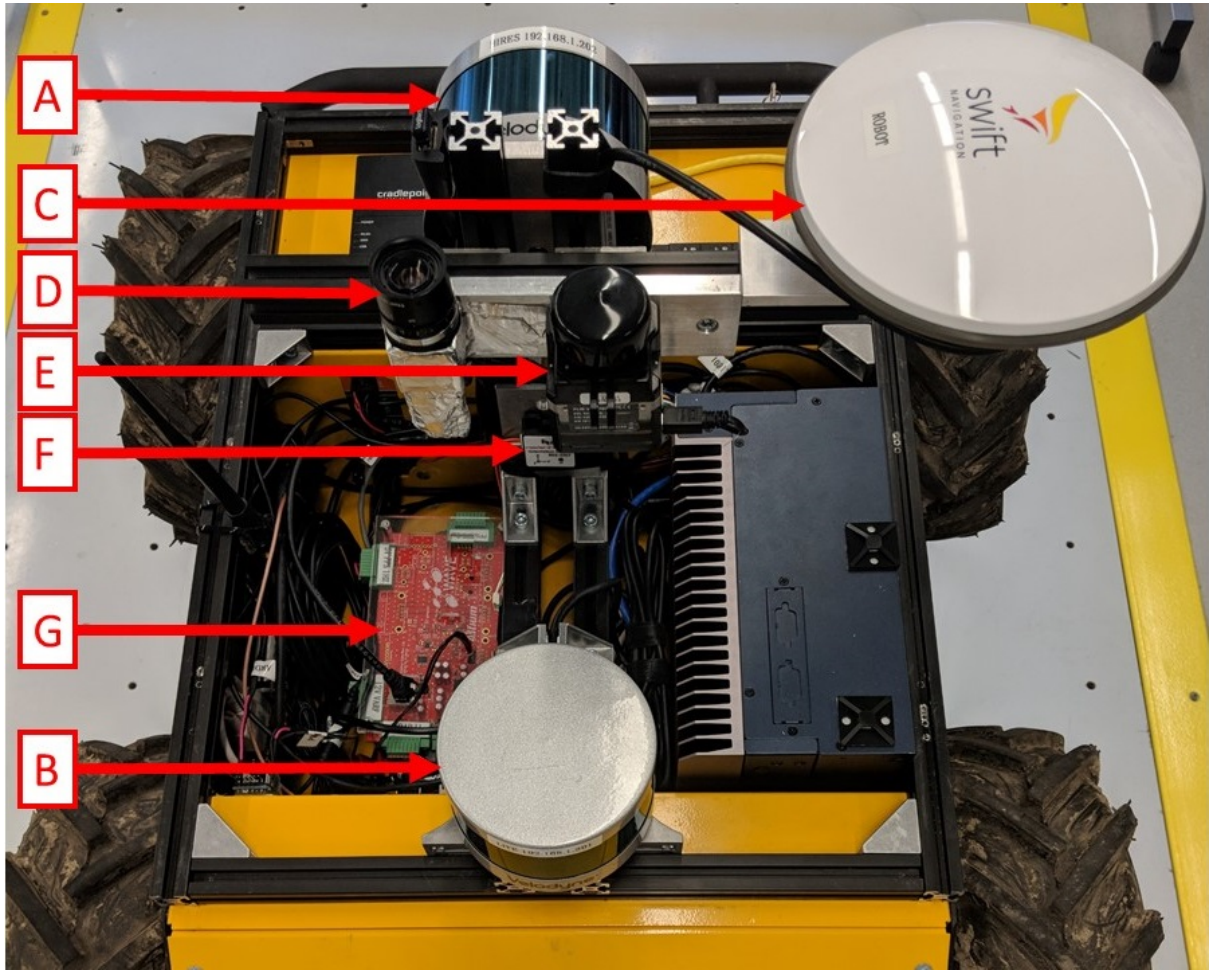
derive backward projection equations to

recover 3D scene structure from images.

Scene from images

Example: Coordinate Transformation

https://www.youtube.com/watch?v=YCihWIBIGZ8



Projection Matrix (Camera Matrix)

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

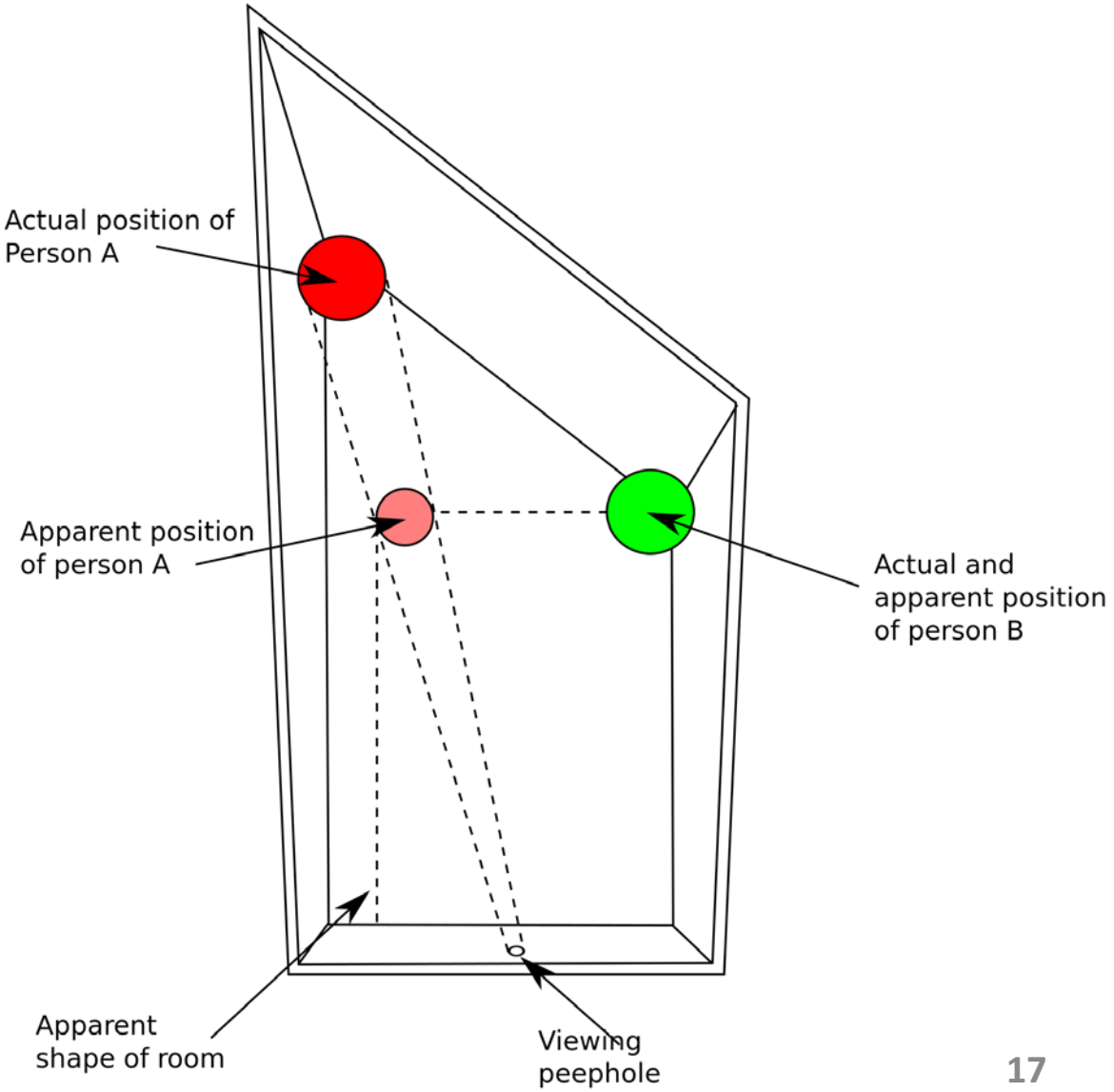
- **If we knew a projective matrix in an image, we can compute the image point corresponding to the world point**
- **Although we knew an image points, we cannot find an unique world point.**

$$\mathbf{x} = \mathbf{P}_i \mathbf{X}$$

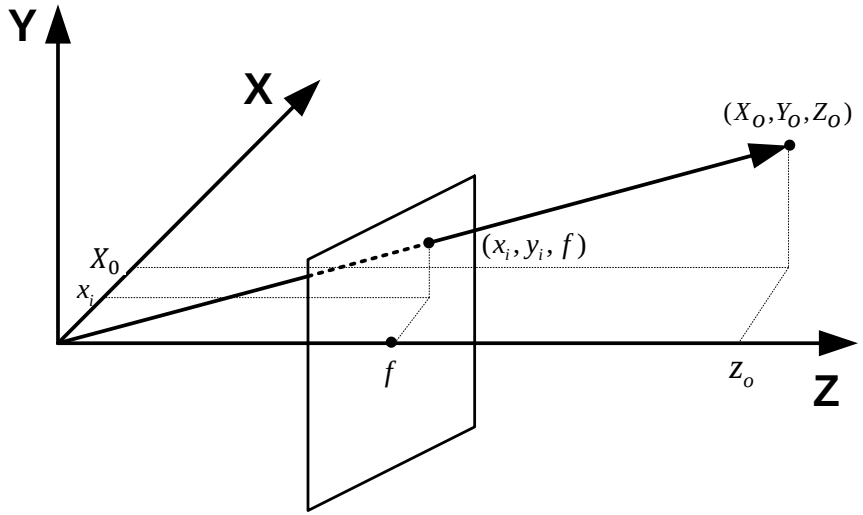
2D point **3D point**

Example: Unknown Scale

Example: Ames room



Camera Model: A Simple Case

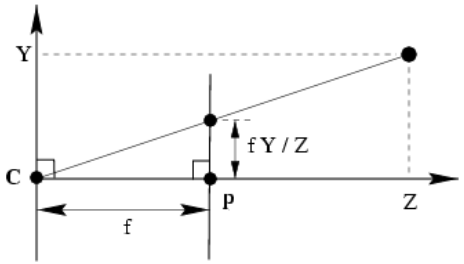


$$\frac{x_i}{f} = \frac{X_o}{Z_0} \;,\; \frac{y_i}{f} = \frac{Y_o}{Z_0}$$

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

Up to scale !!!

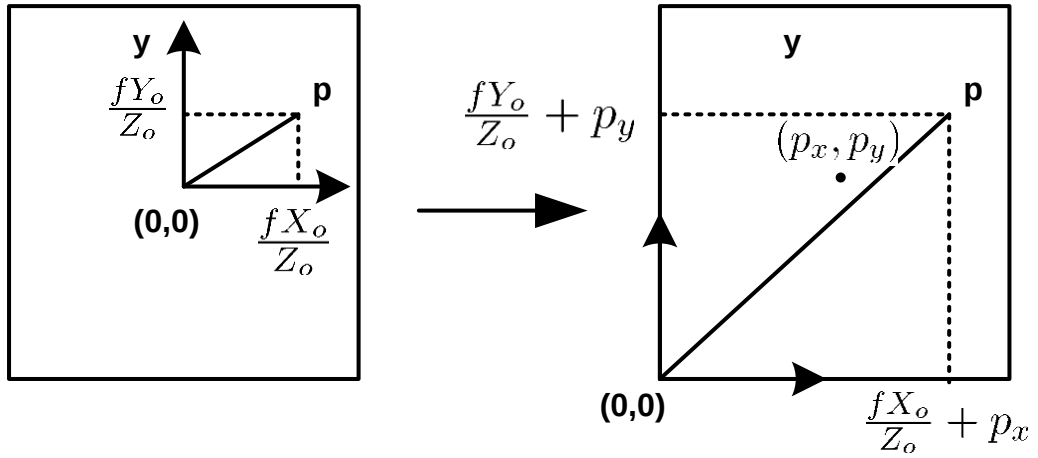
$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} fX_o \\ fY_o \\ Z_o \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}$$



- Any point on the ray OP is projected onto a same image point.
- Only valid at homogeneous coordinate

1 Unknown

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



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

$$\mathbf{K}$$

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

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

(Optional) Camera Model : When Different Pixel/Unit Ratios in X and Y Directions

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)

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

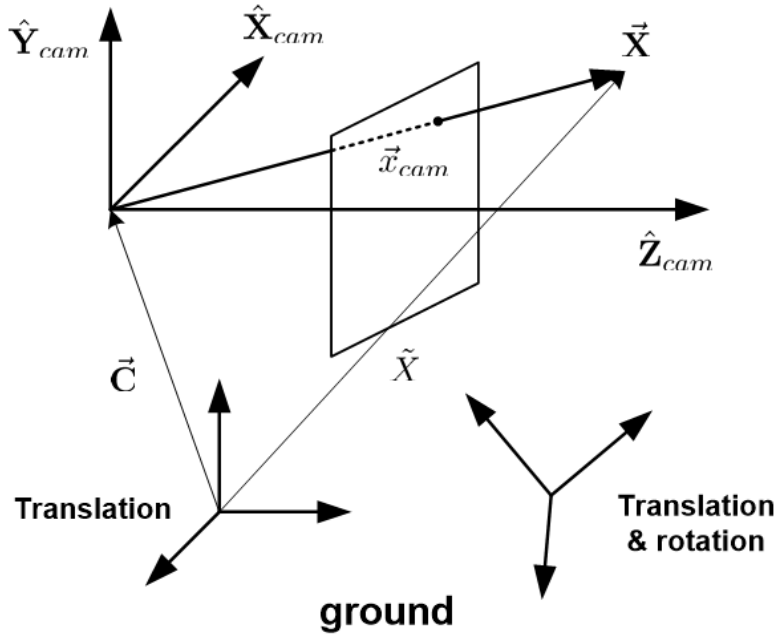
$$\vec{x} = \mathbf{K} \left[\mathbf{I} \mid \vec{0} \right] \vec{X}$$

$$= \begin{bmatrix} m_x f & 0 & x_0 & 0 \\ 0 & m_y f & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \vec{X}$$

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

4 Unknowns

Camera Model : Full Configuration



Translation

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

Translation & rotation

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

$$\mathbf{P} = \left[\ \vec{P}_1 \quad \vec{P}_2 \quad \vec{P}_3 \quad \vec{P}_4\ \right]$$
$$= \mathbf{KR}\left[\ \mathbf{I}\ \middle|\ -\vec{C}\ \right]$$

Multiview geometry

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

$$\mathbf{x} = \mathbf{P}\,\mathbf{X}$$

9 (10) Unknowns

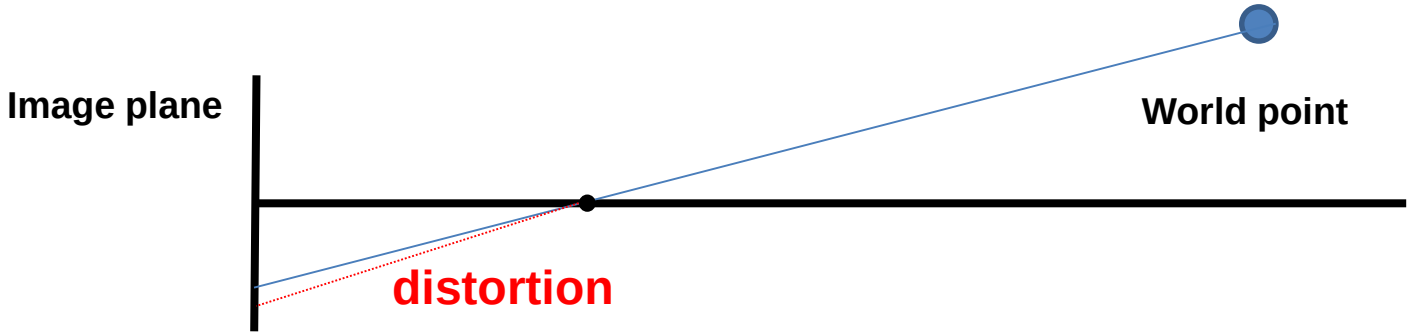
Radial Distortion

Radial distortion (due to optics of the lens)

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

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

before after



9 (10) + # of distortion parameters unknowns

Example: Lens Distortion Correction

Example: Camera Distortion Model in MATLAB

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:

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

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

where:

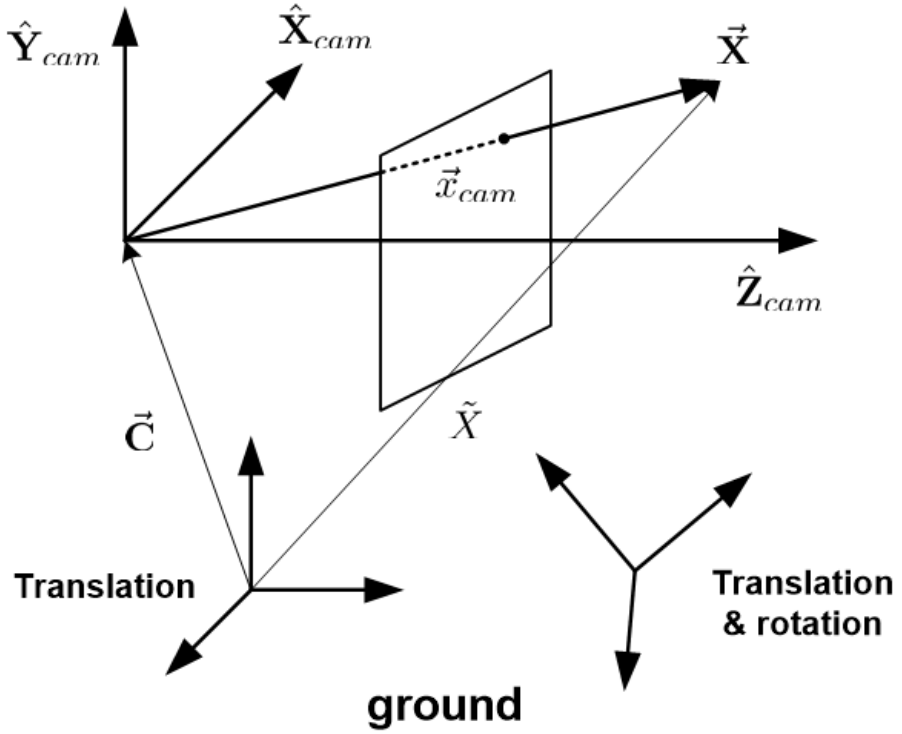
k (1) , k (2) = radial distortion coefficients of the lens

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

Camera Matrix



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

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

With respect to the world coordinate

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: VisualSfM (http://ccwu.me/vsfm/doc.html)

The camera model and coordinate system

Be careful it is slightly different with other SFM software (such as bundler).

The internal camera model has 8 parameters (7 if radial distortion is disabled)
  Given camera K[R T], K = [f, 0 0; 0 f 0; 0 0 1], radial distortion r, and a 3D point X.
  The reprojection in the image is [x, y, z]' = K (RX + T) -> (x/z, y/z)'
  Let the measurement be (mx, my), which is relative to principal point (typically image center)
  The distortion factor is r2 = r * (mx * mx + my * my)
  The undistorted measurement is (1 + r2) * (mx, my)
  Then, the reprojection error is (x/z - (1 + r2) mx, y /z - (1 + r2) my)

NOTE that the parameters saved in NVM file is slightly different with the internal representation. Instead, NVM saves the following for each camera:
  f, R (as quaternion), C = - R'T, rn = r * f * f.
The PBA code includes functions for loading the NVM file and convert the camera parameters.

The radial distortion model is different with other softwares:
* Dan Costin provides an efficient code for undistorting the images under this model.

As for the image coordinate system, X-axis points right, and Y-axis points downward, so Z-axis points forward.

The principal points are assumed to be at image centers except when using a single fixed calibration. When using fixed calibration [fx, cx, fy, cy], the error is estimated in a transformed image coordinate system.
  K = [fx 0 0; 0 fx 0; 0 0 1], Kc = [1 0 cx; 0 fy/fx cy; 1],
  You can see that Kc * K = [fx 0 cx; 0 fy cy; 0 0 1];
  Let (u, v, 1)' be an original feature location (relative to top-left corner of images)
  The measurement is defined as (mx, my, 1)' = Inv(Kc) * (u, v, 1)'

Note: the feature locations saved in NVM files are still relative to the image centers rather than the calibrated principal point [cx, cy];

Example: Pix4D

https://support.pix4d.com/hc/en-us/articles/202560169#gsc.tab=0

Camera Model Name	Field used to type the camera model name. It is recommended to type the name as follow: *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).

Projective Matrices of Multiple Images

i : Image number

$$x = P_i X$$

2D point **3D point**

- **If we knew a projective matrix in each image, we can compute the image point corresponding to the world point**

- **If we knew more two image points indicating same world point, we can compute the location of the world point. (triangulation)**

Slide Credits and References

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