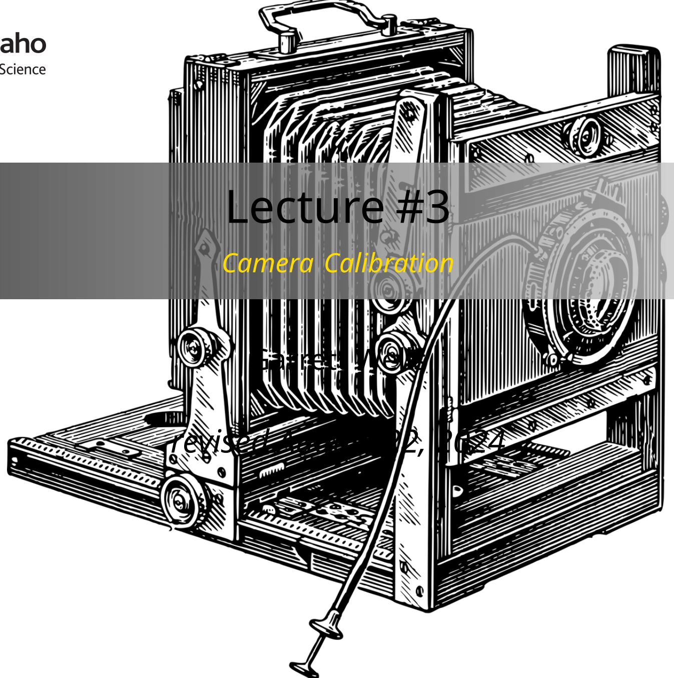
University of Idaho
Department of Computer Science
Coeur d'Alene



# Lecture #3

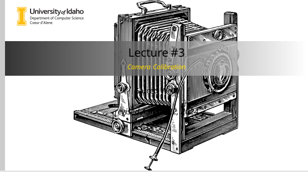
*Camera Calibration*

Garrett Wells

*revised August 22, 2024*

2024-08-22

Lecture #3



# Table of Contents I






Table of Contents I

Review

Cameras and Reality

Distortion
Radial
Tangential
Intentional Distortion

Camera Calibration

Background Theory

Camera Matrix
Camera Matrix Decomposition
Intrinsic Matrix
Rotation Matrix
Distortion Coefficients

# Cameras

*Review*

- Camera Obscura
- Modern Cameras
  1. Aperture
  2. Focal Length



Cameras
Review
- Camera Obscura
- Modern Cameras
  1. Aperture
  2. Focal Length

# Cameras

*Review*

- Camera Obscura
- Modern Cameras
    1. Aperture
    2. Focal Length
    3. **Shutter**



Cameras
Review
- Camera Obscura
- Modern Cameras
    1. Aperture
    2. Focal Length
    3. **Shutter**

# Cameras

*Review*

- Camera Obscura
- Modern Cameras
  1. Aperture
  2. Focal Length
  3. **Shutter**
  4. **Sensor**



Cameras
*Review*
- Camera Obscura
- Modern Cameras
  1. Aperture
  2. Focal Length
  3. **Shutter**
  4. **Sensor**

# Cameras and Reality

*Cameras see the world...*

**Camera**

Translates between the 3D world and a 2D image [1].
$x = PX$

Our goal is to use features in the 2D image(lines) to describe 3D objects.



Cameras and Reality
Cameras see the world...
Camera
Translates between the 3D world and a 2D image [1].
$x = PX$
Our goal is to use features in the 2D image(lines) to describe 3D objects.

# Cameras and Reality

*Cameras tend to lie about the world…*

- Cameras depend on lenses to bend and focus light
- Lenses are designed and manufactured by humans
- Lenses are prone to distortion



Cameras and Reality
Cameras tend to lie about the world…
- Cameras depend on lenses to bend and focus light
- Lenses are designed and manufactured by humans
- Lenses are prone to distortion

# Distortion

*Warping Reality*

- Distortion reduces fidelity
- Reference lines become curved

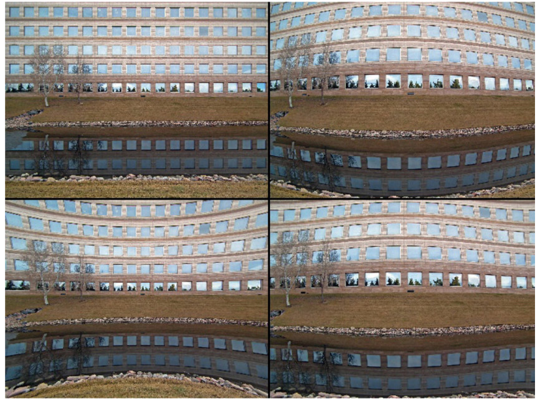


Figure: Examples of Distortion



Distortion
Warping Reality
- Distortion reduces fidelity
- Reference lines become curved

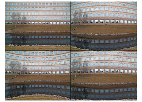


Figure: Examples of Distortion

## Radial Distortion

- Light bends more at edges than center
- Pincushion, Barrel, Mustache(Complex)

## Tangential

- Image sensor isn't parallel to image plane




Distortion
Types of Distortion
Radial Distortion
Light bends more at edges than center
Pincushion, Barrel, Mustache(Complex)
Tangential
Image sensor isn't parallel to image plane

# Radial Distortion

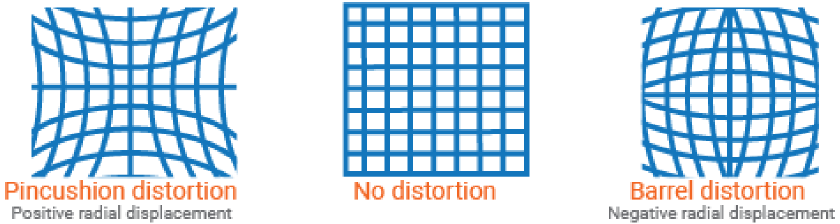
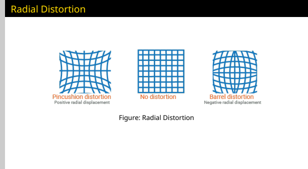



Figure: Radial Distortion





1. Radial distortion occurs when light rays bend more near the edges of a lens than they do at its center

# Tangential Distortion

Figure: Tangential Distortion Example

2024-08-22

Lecture #3
└─Distortion
  └─Tangential
    └─Tangential Distortion

Tangential Distortion

Figure: Tangential Distortion Example

# Tangential Distortion

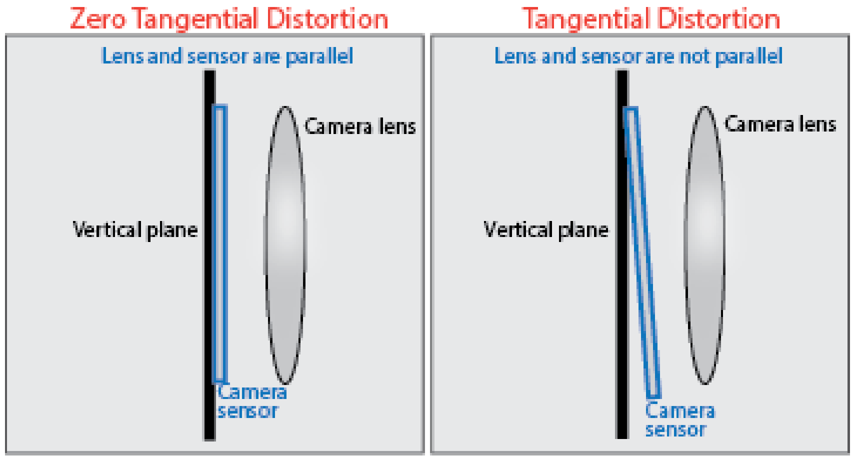
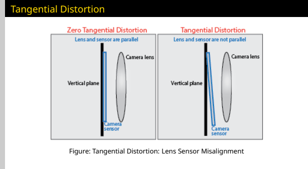



Figure: Tangential Distortion: Lens Sensor Misalignment






1. Occurs when the lens and sensor are not parallel.

# Intentional Radial Distortion

*Fish-eye Lenses*

Fish-eye lenses produce intentional distortion


Figure: Fish-Eye Lens

Figure: Artistic Fish-eye Perspective

2024-08-22

Lecture #3
└Distortion
  └Intentional Distortion
    └Intentional Radial Distortion

Intentional Radial Distortion
Fish-eye Lenses
Fish-eye lenses produce intentional distortion

Figure: Fish-Eye Lens

Figure: Artistic Fish-eye Perspective

# Intentional Tangential Distortion

*Tilt-Shift Lenses*

Tilt-shift lenses allow you to change lens alignment

Figure: Tilt Shift Lens

Figure: Tilted View of Hong Kong



# Intentional Tangential Distortion

*Tilt-Shift Lenses*

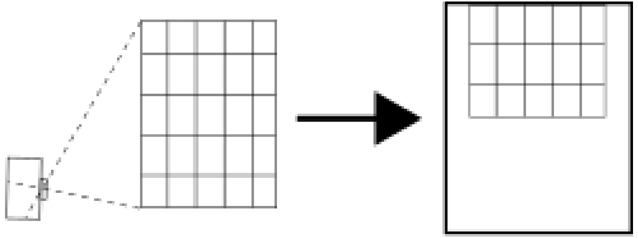


Figure: Level Camera



Intentional Tangential Distortion
Tilt-Shift Lenses

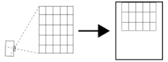


Figure: Level Camera

# Intentional Tangential Distortion

*Tilt-Shift Lenses*

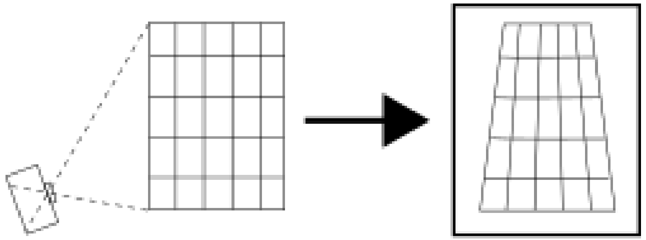
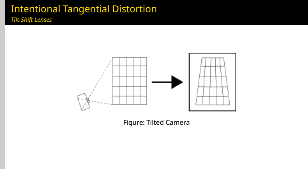


Figure: Tilted Camera





# Intentional Tangential Distortion

*Tilt-Shift Lenses*

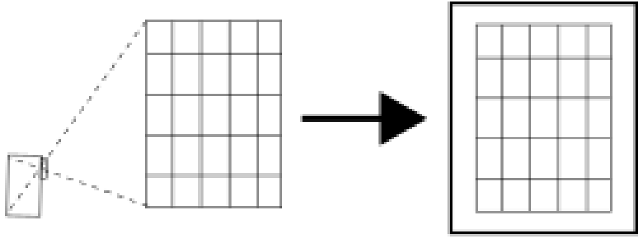


Figure: Shifted Camera

2024-08-22

Lecture #3
└Distortion
  └Intentional Distortion
    └Intentional Tangential Distortion

Intentional Tangential Distortion
Tilt-Shift Lenses

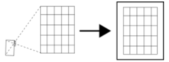


Figure: Shifted Camera

# Sources

Carnegie Mellon University's Computer Vision Slides



Sources
Carnegie Mellon University's Computer Vision Slides

# Coordinate Systems

1. Camera Coordinate System
2. Image Sensor's Coordinate System
3. World Space



Coordinate Systems

1. Camera Coordinate System
2. Image Sensor's Coordinate System
3. World Space

# Calibration Matrices

1. **Intrinsic Matrix:** camera ➞ image sensor
2. **Extrinsic Matrix:** world ➞ camera

*How do we map between our 3D world and the camera's 2D perspective?*




Calibration Matrices
1. **Intrinsic Matrix:** camera ➞ image sensor
2. **Extrinsic Matrix:** world ➞ camera
*How do we map between our 3D world and the camera's 2D perspective?*

# Ideal Case

*Pinhole Camera*

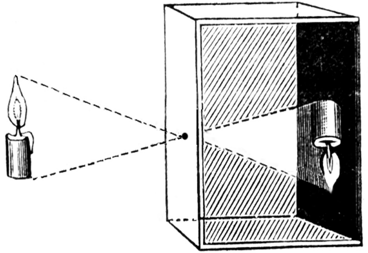


Figure: Pinhole Camera

- Straight line from image plane to world point
- Easy, how can we describe this “projection”?

2024-08-22

Lecture #3
└Camera Calibration
└Ideal Case

Ideal Case
Pinhole Camera

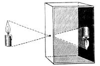


Figure: Pinhole Camera

- Straight line from image plane to world point
- Easy, how can we describe this “projection”?

# Terminology and Setup

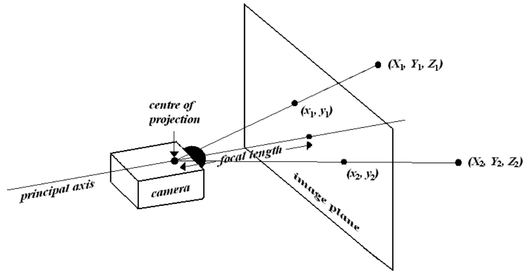
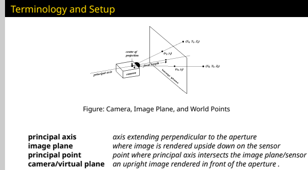



Figure: Camera, Image Plane, and World Points

| | |
|---|---|
| **principal axis** | *axis extending perpendicular to the aperture* |
| **image plane** | *where image is rendered upside down on the sensor* |
| **principal point** | *point where principal axis intersects the image plane/sensor* |
| **camera/virtual plane** | *an upright image rendered in front of the aperture .* |






1. For a pinhole camera(ideal case) the image plane is constructed at the aperture/point of convergence. Which should match the focal length.
2. **Correction:** Contrary to this diagram, we are referring to the “image plane” as the virtual image plane/camera.
3. **Virtual Plane:** must be $xf_{focallength}$ distance in front of the aperture, with scale $x : 1$(or $1 : 1$ at focal length), following rules of projective geometry [2].

## Projective Geometry

1. A system specialized for applications such as computer graphics.
2. System for scaling points in images(2D) or 3D models.

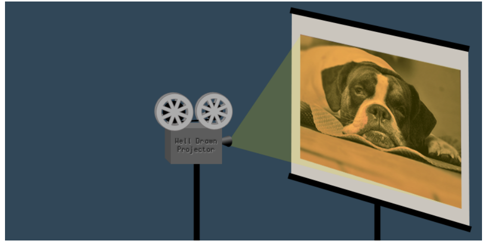



Figure: Projector Close

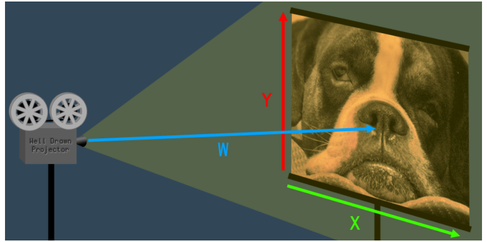



Figure: Projector Far: Where are our points?



Projective Geometry
Projective Geometry
1. A system specialized for applications such as computer graphics.
2. System for scaling points in images(2D) or 3D models.
Figure: Projector Close
Figure: Projector Far: Where are our points?

1. **Summary:** *Homogeneous coordinates allow any cartesian point to be projected when multiplied by a scalar. This is important because image coordinate* $(x, y)$ *corresponds to a world coordinate* $(x, y, z)$
2. If you think it is weird to include technology such as a projector in a mathematical system, remember, this geometry is made for applications such as computer graphics where scaling/projecting is common/necessary.
3. For our purposes, $W$ is the principal axis extending from, and perpendicular to the aperture

# Homogeneous Coordinates

## Projective Geometry

A four dimensional space consisting of $X$, $Y$, $Z$, and $W$ [3], [4].

## Homogeneous Coordinate

A coordinate (3D or 4D) with a dimension $W$, which represents projective space.

$(X, Y) \rightarrow (X \times W, Y \times W, W)$

$(X, Y, Z) \rightarrow (X \times W, Y \times W, Z \times W, W)$



Homogeneous Coordinates
Projective Geometry
A four dimensional space consisting of $X$, $Y$, $Z$, and $W$ [3], [4].
Homogeneous Coordinate
A coordinate (3D or 4D) with a dimension $W$, which represents projective space.
$(X, Y) \rightarrow (X \times W, Y \times W, W)$
$(X, Y, Z) \rightarrow (X \times W, Y \times W, Z \times W, W)$

# Homogeneous Coordinates

**Example**

2D → 3D Homogeneous
$(X, Y) \rightarrow (X \times W, Y \times W, W)$
$(1, 3)$ & $W = 1 \rightarrow (1, 3, 1)$

**Example**

3D → 4D Homogeneous
$(X, Y, Z) \rightarrow (X \times W, Y \times W, Z \times W, W)$
$(5, 2, 1)$ & $W = 3 \rightarrow (15, 6, 3, 3)$



Homogeneous Coordinates
Example
2D → 3D Homogeneous
$(X, Y) \rightarrow (X \times W, Y \times W, W)$
$(1, 3)$ & $W = 1 \rightarrow (1, 3, 1)$
Example
3D → 4D Homogeneous
$(X, Y, Z) \rightarrow (X \times W, Y \times W, Z \times W, W)$
$(5, 2, 1)$ & $W = 3 \rightarrow (15, 6, 3, 3)$

## Homogeneous Coordinates

*Intuitive/Weird Homogeneous Coordinate Properties*

- Original scale(1:1), is when $W = 1$
- $W < 0$ flips upside–down and back–to–front
- $W$ cannot equal 0



Homogeneous Coordinates
Intuitive/Weird Homogeneous Coordinate Properties
Original scale(1:1), is when $W = 1$
$W < 0$ flips upside–down and back–to–front
$W$ cannot equal 0

# Camera Matrix

$$x = PX$$

Figure: World To Image Coordinates

- ☞ $x$ denotes image coordinate (homogeneous)
- ☞ $X$ is the same coordinate, but in world space (homogeneous)
- ☞ *What is P?*



Camera Matrix

$$x = PX$$

Figure: World To Image Coordinates

- ☞ $x$ denotes image coordinate (homogeneous)
- ☞ $X$ is the same coordinate, but in world space (homogeneous)
- ☞ *What is P?*

# Camera Matrix

$$x = PX$$

Figure: World To Image Coordinates

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$



Camera Matrix

$$x = PX$$

Figure: World To Image Coordinates

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

1. **Math Check:** $[3 \times 4] \times [4 \times 1] = [3 \times 1]$, which is 3D vector.

# Camera Matrix

$$\begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure: Camera Matrix: Image & Camera Same Origin

,



Camera Matrix

$$\begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure: Camera Matrix: Image & Camera Same Origin

1. **Note:** $f_x$, $f_y$, are the same for the ideal case (pinhole camera/camera obscura) but as soon as we introduce a sensor/film, there is the possibility of misalignment.
2. According to [2], $f_x, f_y$ differ because of sensor flaws, non-uniform scaling, etc.
3. Some texts use single focal length and an "aspect ratio" to describe the amount of deviation from a square pixel. This approach separates camera geometry from distortion.

# Camera Matrix

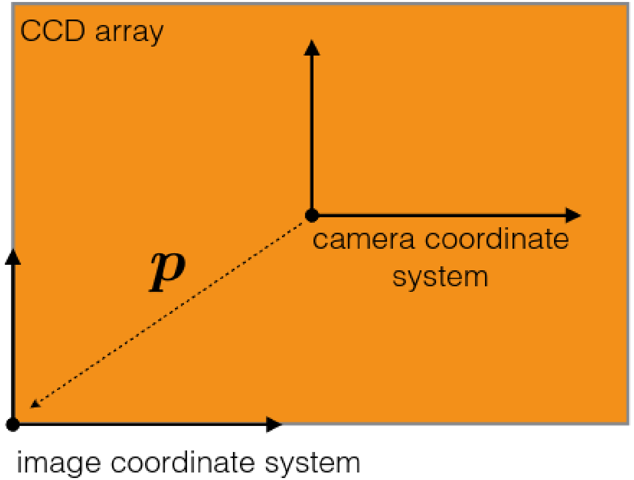
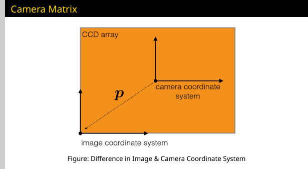



Figure: Difference in Image & Camera Coordinate System





# Camera Matrix

$$\begin{bmatrix} f_x & 0 & p_x & 0 \\ 0 & f_y & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure: Camera Matrix: Image Offset from Camera



Camera Matrix

$$\begin{bmatrix} f_x & 0 & p_x & 0 \\ 0 & f_y & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure: Camera Matrix: Image Offset from Camera

1. Here, we add $p_x$, $p_y$, which are the optical center/principal point offsets
2. These are used to shift the location of the center of the film/sensor relative to the pinhole/aperture.

# Remember The Goal

**Definition**

**Camera** Translates between the 3D world and a 2D image [1].

$$x = PX$$



Remember The Goal
Definition
Camera Translates between the 3D world and a 2D image [1].
$x = PX$

1. So far we have constructed a camera matrix that accounts for the internal geometry of the camera.
2. It doesn't model any of our lens distortion, or relate world space to the virtual image.

# Camera Matrix Decomposition

*Adding distortion correction to our model...*

## Camera Matrix Decomposed

$$x = PX$$

$$\blacktriangledown$$

$$P = KR[I| - C]$$



Camera Matrix Decomposition
*Adding distortion correction to our model...*
Camera Matrix Decomposed
$x = PX$
$\blacktriangledown$
$P = KR[I| - C]$

1. This is how we break down the camera matrix to model *intrinsic*(interior) and *extrinsic*(exterior/world space) relationships.

# Camera Matrix Decomposition

*In more detail*

$$P = KR[I| - C]$$

| | |
|---|---|
| $K$ | **intrinsic matrix** of the camera |
| $R[I\| - C]$ | **extrinsic matrix** |
| $R$ | **rotation** from world to camera coordinate system |
| $I$ | identity matrix |
| $-C$ | **translation** from camera to world coordinate |



Camera Matrix Decomposition
In more detail
$P = KR[I| - C]$
$K$ intrinsic matrix of the camera
$R[I| - C]$ extrinsic matrix
$R$ rotation from world to camera coordinate system
$I$ identity matrix
$-C$ translation from camera to world coordinate

# Intrinsic Matrix

$$\begin{bmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

Figure: Intrinsic Matrix, $K$

- Represent optical center and focal length of camera [5].
- $s$ represents *skew*
- $f_x$, $f_y$, represent *focal length in pixels*
- $p_x$, $p_y$, represent optical center (in pixels)



Intrinsic Matrix

$$\begin{bmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

Figure: Intrinsic Matrix, $K$

- Represent optical center and focal length of camera [5].
- $s$ represents *skew*
- $f_x$, $f_y$, represent *focal length in pixels*
- $p_x$, $p_y$, represent optical center (in pixels)

# Extrinsic Matrix

$$R\left[\begin{array}{ccc|c} 1 & 0 & 0 & -C_x \\ 0 & 1 & 0 & -C_y \\ 0 & 0 & 1 & -C_z \end{array}\right]$$

Figure: Extrinsic Matrix, $R[I|-C]$

- Represents the location of the camera in world space [5].
- $R$ is the rotation matrix(3x3), values defined in [6].
- Translate → rotate



Extrinsic Matrix

$$R\left[\begin{array}{ccc|c} 1 & 0 & 0 & -C_x \\ 0 & 1 & 0 & -C_y \\ 0 & 0 & 1 & -C_z \end{array}\right]$$

Figure: Extrinsic Matrix, $R[I|-C]$

- Represents the location of the camera in world space [5].
- $R$ is the rotation matrix(3x3), values defined in [6].
- Translate → rotate

# Extrinsic Matrix: Camera to World

*Coordinate Systems*

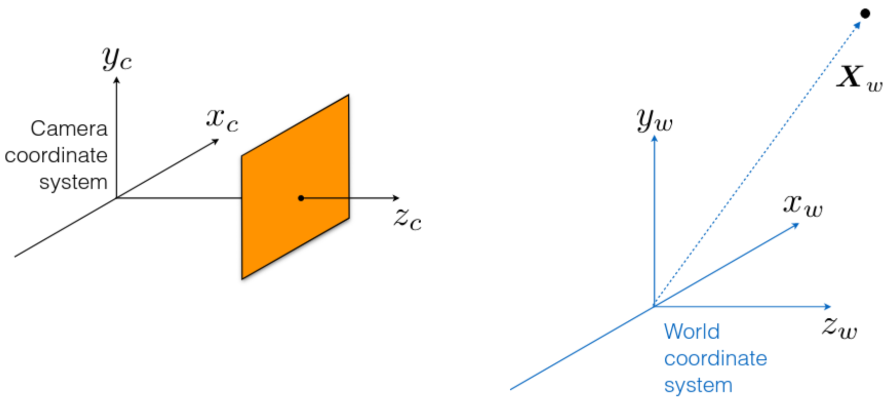
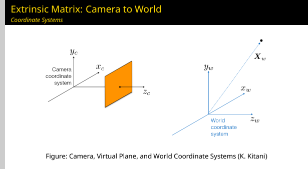



Figure: Camera, Virtual Plane, and World Coordinate Systems (K. Kitani)

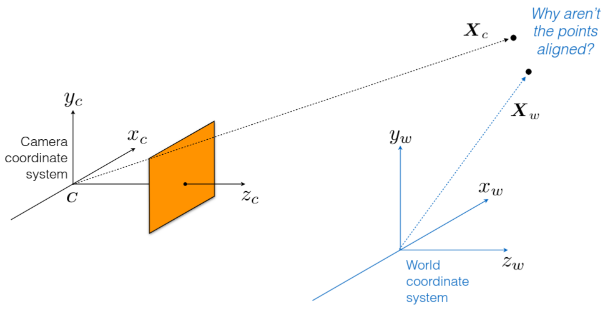





1. Figures from [1].

# Extrinsic Matrix: Camera to World

*Step 1: Translation*

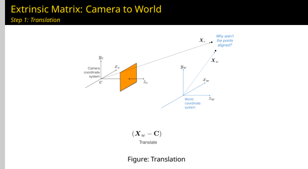



$$(\boldsymbol{X}_w - \mathbf{C})$$

Translate

Figure: Translation

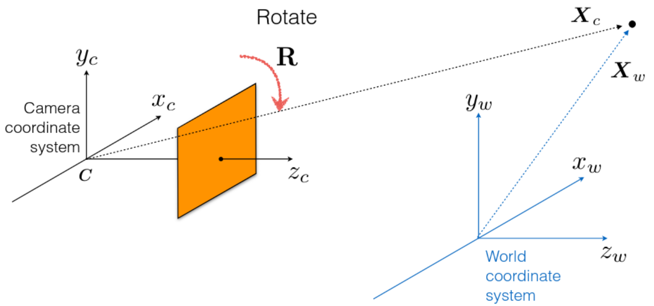





1. Figures from [1].

# Extrinsic Matrix: Camera to World

*Step 2: Rotation*

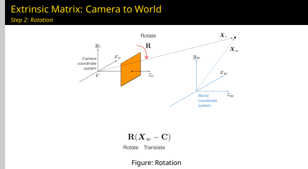



$$\underbrace{\mathbf{R}}_{\text{Rotate}}\underbrace{(\boldsymbol{X}_w - \mathbf{C})}_{\text{Translate}}$$

Figure: Rotation

2024-08-22

Lecture #3
└Camera Matrix
  └Intrinsic Matrix
    └Extrinsic Matrix: Camera to World



1. Figures from [1].

# Extrinsic Matrix: Camera to World

*Step 2: Rotation*

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Extrinsic Matrix: Camera to World
Step 2: Rotation

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1. This is a rotation about the Z-axis (axis of projection) only.
2. Figures from [1].

# Camera Matrix As Sequence of Operations

$$P = \overbrace{K}^{\text{Intrinsic Matrix}} \times \overbrace{[R \mid \mathbf{t}]}^{\text{Extrinsic Matrix}}$$

$$= \overbrace{\underbrace{\begin{pmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{pmatrix}}_{\text{2D Translation}} \times \underbrace{\begin{pmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\text{2D Scaling}} \times \underbrace{\begin{pmatrix} 1 & s/f_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\text{2D Shear}}}^{\text{Intrinsic Matrix}} \times \overbrace{\underbrace{\left( I \mid \mathbf{t} \right)}_{\text{3D Translation}} \times \underbrace{\left( \begin{array}{c|c} R & 0 \\ \hline 0 & 1 \end{array} \right)}_{\text{3D Rotation}}}^{\text{Extrinsic Matrix}}$$

Figure: Sequence of Operations (Simek, K.)



Camera Matrix As Sequence of Operations
Figure: Sequence of Operations (Simek, K.)

1. Another perspective on what is happening, what types of operations are being performed under the hood here.

# Extrinsic Matrix: Alternative Notation

**Key Difference:** Rotates ➞ translates

$$P = K[R|t]$$

where

$$t = -RC$$



Extrinsic Matrix: Alternative Notation
**Key Difference:** Rotates ➞ translates
$$P = K[R|t]$$
where
$$t = -RC$$

# Distortion Coefficients

Real lenses usually have some distortion, mostly radial distortion and slight tangential distortion. So, the above model is extended as:

$$
\begin{aligned}
&\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t \\
&x' = x/z \\
&y' = y/z \\
&x'' = x' \frac{1+k_1 r^2 + k_2 r^4 + k_3 r^6}{1+k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_1 x' y' + p_2(r^2 + 2x'^2) + s_1 r^2 + s_2 r^4 \\
&y'' = y' \frac{1+k_1 r^2 + k_2 r^4 + k_3 r^6}{1+k_4 r^2 + k_5 r^4 + k_6 r^6} + p_1(r^2 + 2y'^2) + 2p_2 x' y' + s_3 r^2 + s_4 r^4 \\
&\text{where} \quad r^2 = x'^2 + y'^2 \\
&u = f_x * x'' + c_x \\
&v = f_y * y'' + c_y
\end{aligned}
$$

$k_1$, $k_2$, $k_3$, $k_4$, $k_5$, and $k_6$ are radial distortion coefficients. $p_1$ and $p_2$ are tangential distortion coefficients. $s_1$, $s_2$, $s_3$, and $s_4$, are the thin prism distortion coefficients. Higher-order coefficients are not considered in OpenCV.



1. These images were included by the previous instructor, but I couldn't find a good link to the cited OpenCV documentation so they will appear uncited here. A close second might be the description in [7].
2. You won't need to do anything with these for this class. OpenCV calculates them during calibration.

# Distortion Coefficients Continued

$$s \begin{bmatrix} x''' \\ y''' \\ 1 \end{bmatrix} = \begin{bmatrix} R_{33}(\tau_x, \tau_y) & 0 & -R_{13}(\tau_x, \tau_y) \\ 0 & R_{33}(\tau_x, \tau_y) & -R_{23}(\tau_x, \tau_y) \\ 0 & 0 & 1 \end{bmatrix} R(\tau_x, \tau_y) \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix}$$

$$u = f_x * x''' + c_x$$
$$v = f_y * y''' + c_y$$

where the matrix $R(\tau_x, \tau_y)$ is defined by two rotations with angular parameter $\tau_x$ and $\tau_y$, respectively,

$$R(\tau_x, \tau_y) = \begin{bmatrix} \cos(\tau_y) & 0 & -\sin(\tau_y) \\ 0 & 1 & 0 \\ \sin(\tau_y) & 0 & \cos(\tau_y) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\tau_x) & \sin(\tau_x) \\ 0 & -\sin(\tau_x) & \cos(\tau_x) \end{bmatrix} = \begin{bmatrix} \cos(\tau_y) & \sin(\tau_y)\sin(\tau_x) & -\sin(\tau_y)\cos(\tau_x) \\ 0 & \cos(\tau_x) & \sin(\tau_x) \\ \sin(\tau_y) & -\cos(\tau_y)\sin(\tau_x) & \cos(\tau_y)\cos(\tau_x) \end{bmatrix}.$$

In the functions below the coefficients are passed or returned as

$$(k_1, k_2, p_1, p_2[, k_3[, k_4, k_5, k_6[, s_1, s_2, s_3, s_4[, \tau_x, \tau_y]]]])$$

vector. That is, if the vector contains four elements, it means that $k_3 = 0$ . The distortion coefficients do not depend on the scene viewed. Thus, they also belong to the intrinsic camera parameters. And they remain the same regardless of the captured image resolution. If, for example, a camera has been calibrated on images of 320 x 240 resolution, absolutely the same distortion coefficients can be used for 640 x 480 images from the same camera while $f_x$, $f_y$, $c_x$, and $c_y$ need to be scaled appropriately.

# Bibliography I


[1] K. Kitani, “Camera Matrix,” Sep. 15, 1975. pmid: 11.

[2] K. Simek, “Dissecting the Camera Matrix, Part 3: The Intrinsic Matrix ←,” (), [Online]. Available: https://ksimek.github.io/2013/08/13/intrinsic/ (visited on 08/22/2024).

[3] *Homogeneous coordinates*, in *Wikipedia*, Aug. 20, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Homogeneous_coordinates&oldid=1241392156 (visited on 08/20/2024).

[4] “Explaining Homogeneous Coordinates & Projective Geometry,” (), [Online]. Available: https://www.tomdalling.com/blog/modern-opengl/explaining-homogenous-coordinates-and-projective-geometry/ (visited on 08/21/2024).





Bibliography I

[1] K. Kitani, “Camera Matrix,” Sep. 15, 1975. pmid: 11.

[2] K. Simek, “Dissecting the Camera Matrix, Part 3: The Intrinsic Matrix ←,” (), [Online]. Available: https://ksimek.github.io/2013/08/13/intrinsic/ (visited on 08/22/2024).

[3] *Homogeneous coordinates*, in *Wikipedia*, Aug. 20, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Homogeneous_coordinates&oldid=1241392156 (visited on 08/20/2024).

[4] “Explaining Homogeneous Coordinates & Projective Geometry,” (), [Online]. Available: https://www.tomdalling.com/blog/modern-opengl/explaining-homogenous-coordinates-and-projective-geometry/ (visited on 08/21/2024).

# Bibliography II


[5] "What Is Camera Calibration? - MATLAB & Simulink," (), [Online]. Available: `https://www.mathworks.com/help/vision/ug/camera-calibration.html` (visited on 08/20/2024).

[6] *Rotation matrix*, in *Wikipedia*, Jul. 31, 2024. [Online]. Available: `https://en.wikipedia.org/w/index.php?title=Rotation_matrix&oldid=1237822022` (visited on 08/22/2024).

[7] "OpenCV: Camera Calibration," (), [Online]. Available: `https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html` (visited on 08/23/2024).





Bibliography II

[5] "What Is Camera Calibration? - MATLAB & Simulink," (), [Online]. Available: https://www.mathworks.com/help/vision/ug/camera-calibration.html (visited on 08/20/2024).

[6] Rotation matrix, in Wikipedia, Jul. 31, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Rotation_matrix&oldid=1237822022 (visited on 08/22/2024).

[7] "OpenCV: Camera Calibration," (), [Online]. Available: https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html (visited on 08/23/2024).