深蓝学院
shenlanxueyuan.com

Autonomous Valet Parking（AVP） Theory and Practice
自主代客泊车理论与实践

Lecture 3： Camera Model

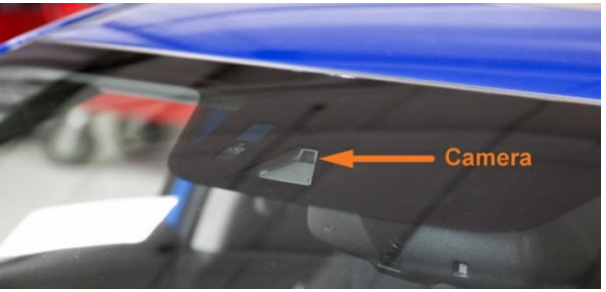
Lecturer Tong QIN

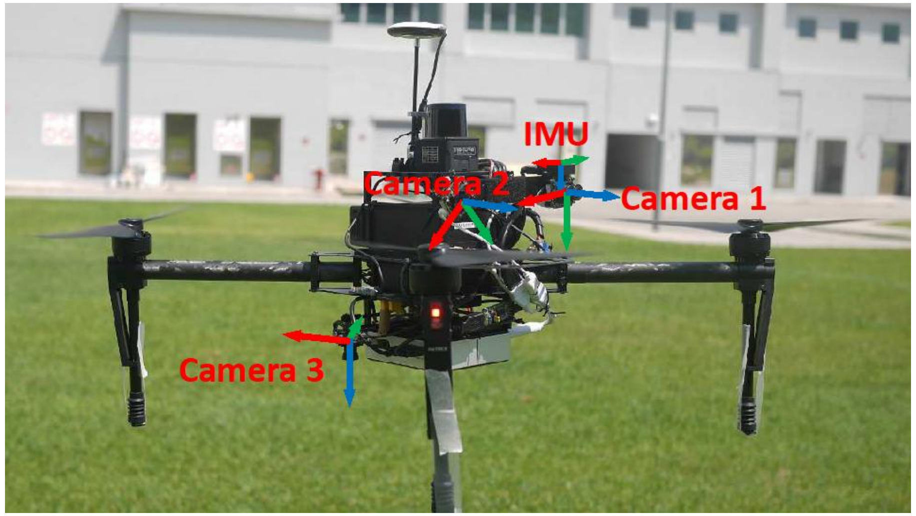
Associate Professor
Shanghai Jiao Tong University
Global Institute of Future Technology
qintong@sjtu.edu.cn

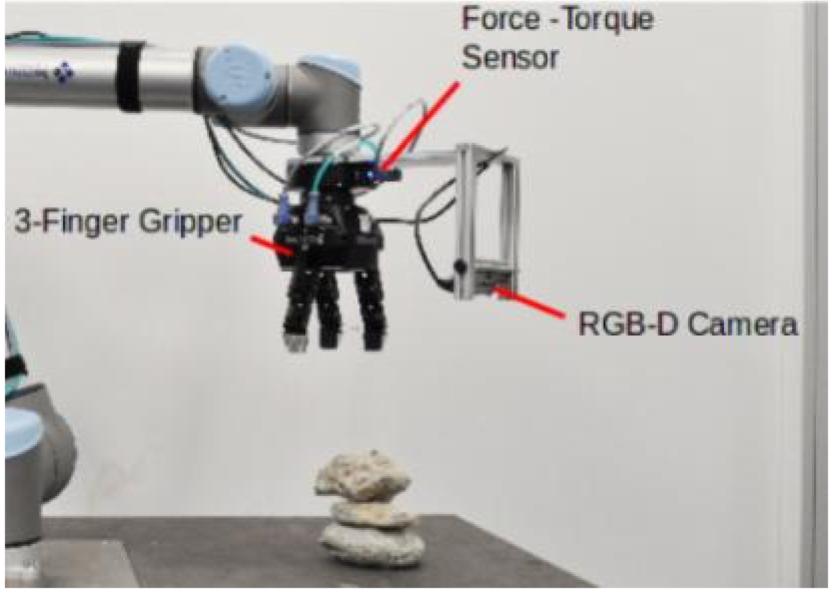
Content

1. Camera Background
2. Pinhole Camera Model
3. Distortion
4. Fisheye Camera Model (MEI)
5. IPM (Inverse Projective Mapping)
6. Assignment

Robot Sees with Cameras

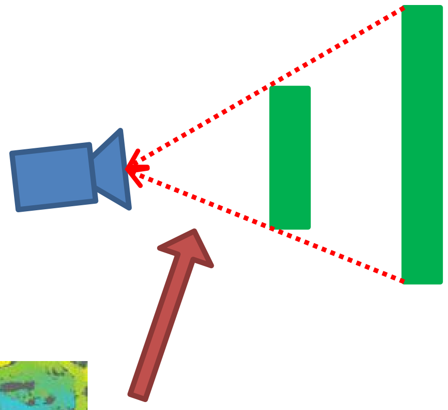






Cameras

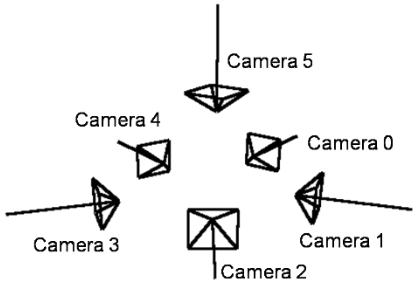
- Monocular
  - Simplest setup
  - Depth unknown



- Stereo
  - Able to compute depth
  - Depth accuracy affected by baseline, resolution, and calibration

- Multi-Camera
  - Overlapping / Non-overlapping field-of-view

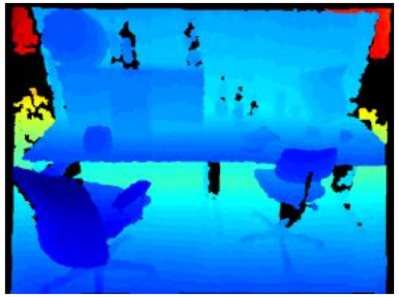
Omnidirectional multi-camera system
Ladybug



Relative position and posture of each camera

Cameras

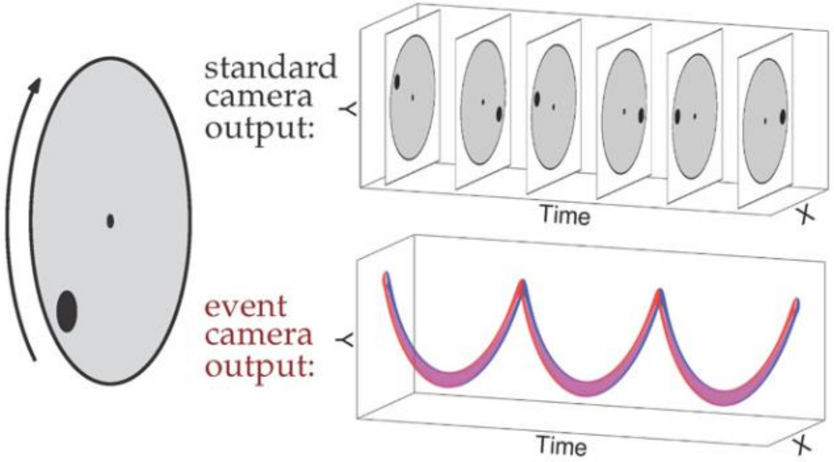
- RGB-D Sensor
  - Great depth in limited range
  - Does not work outdoors

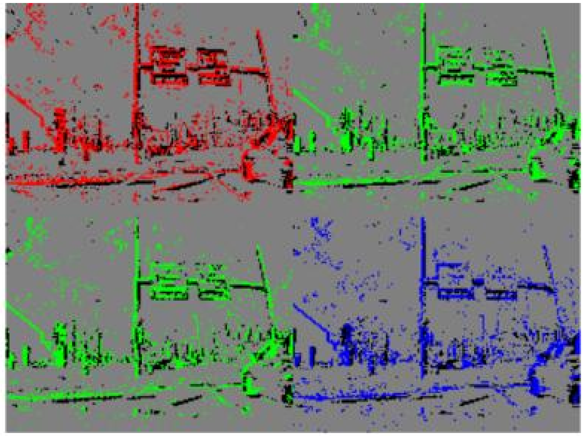


- Omnidirectional camera
  - 360 capture
  - Strong distortion

Cameras

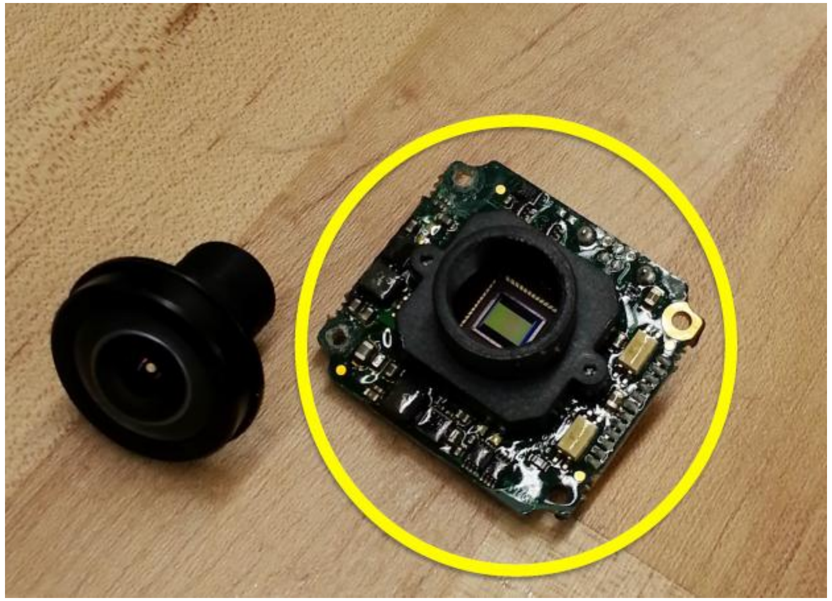
- Event Camera
  - High Temporal Resolution
  - Lack of Rich Visual Information

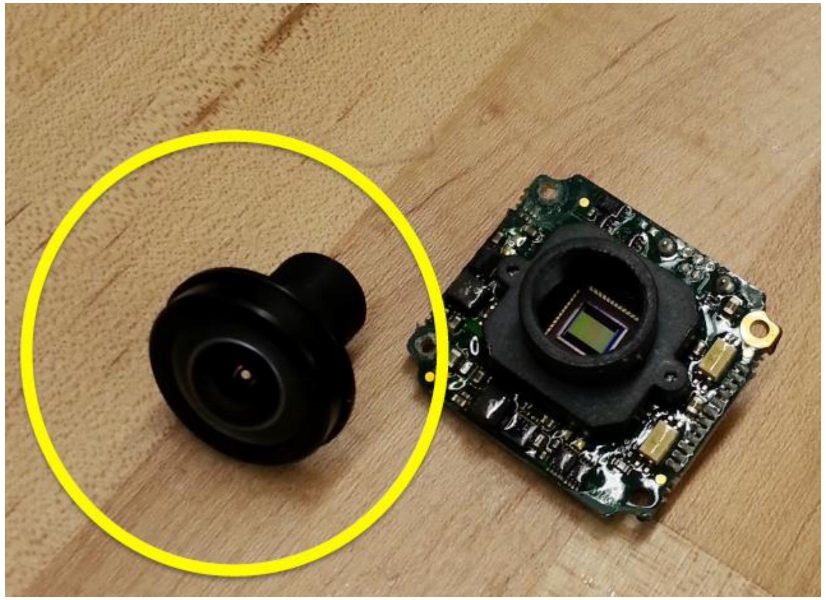




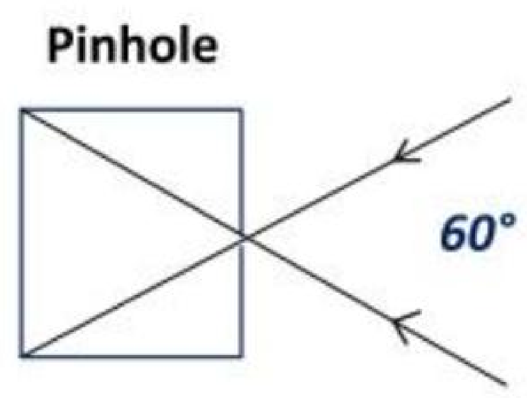
Cameras

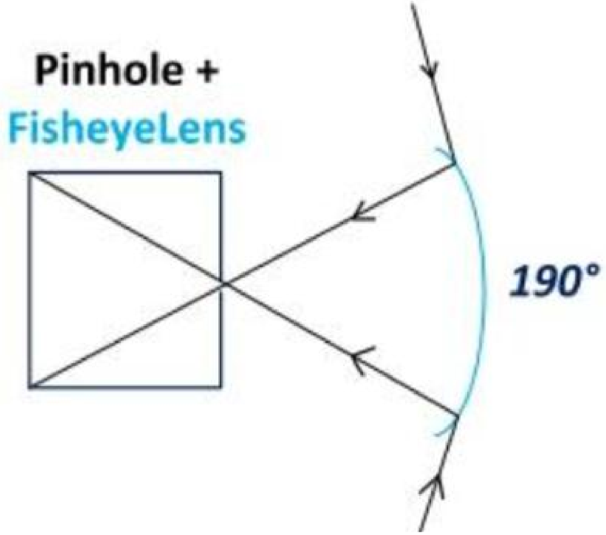
- Sensor
- Lens

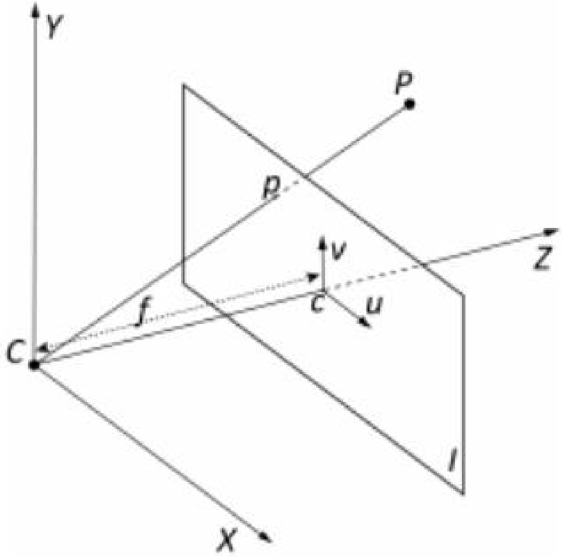




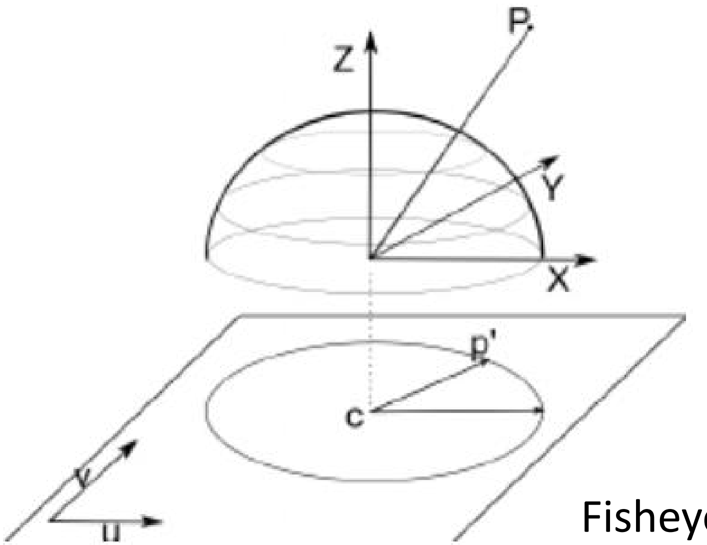
Cameras







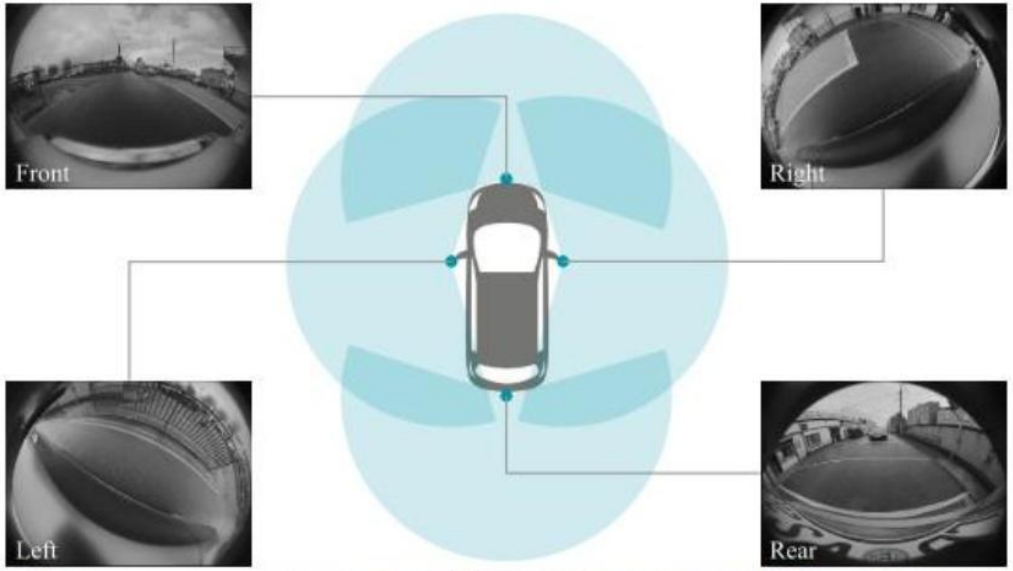
Pinhole model



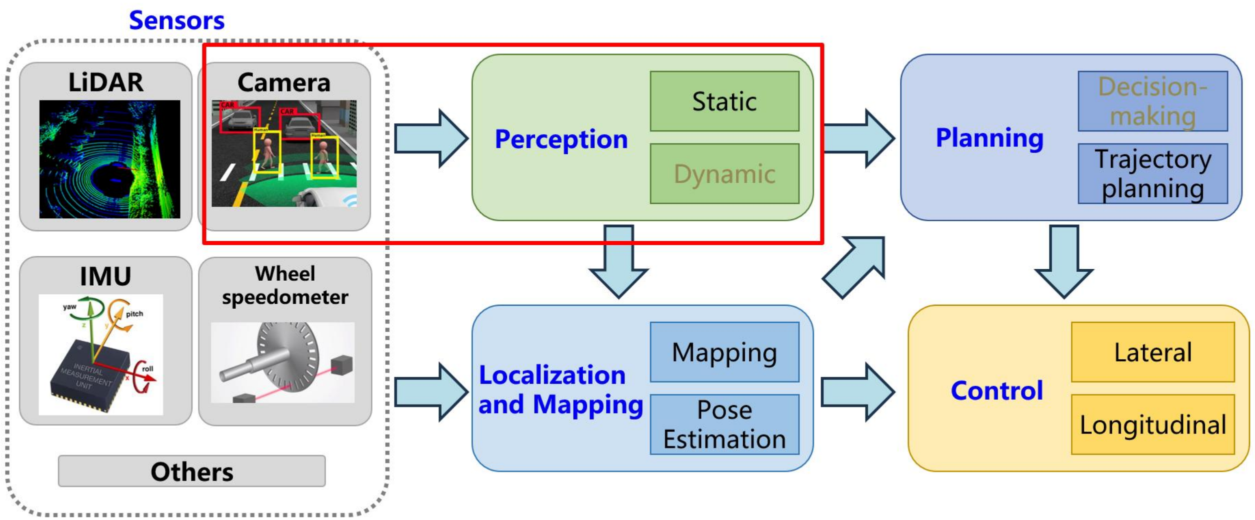
Fisheye model

Fisheye model

Cameras for AVP



AVP Architecture

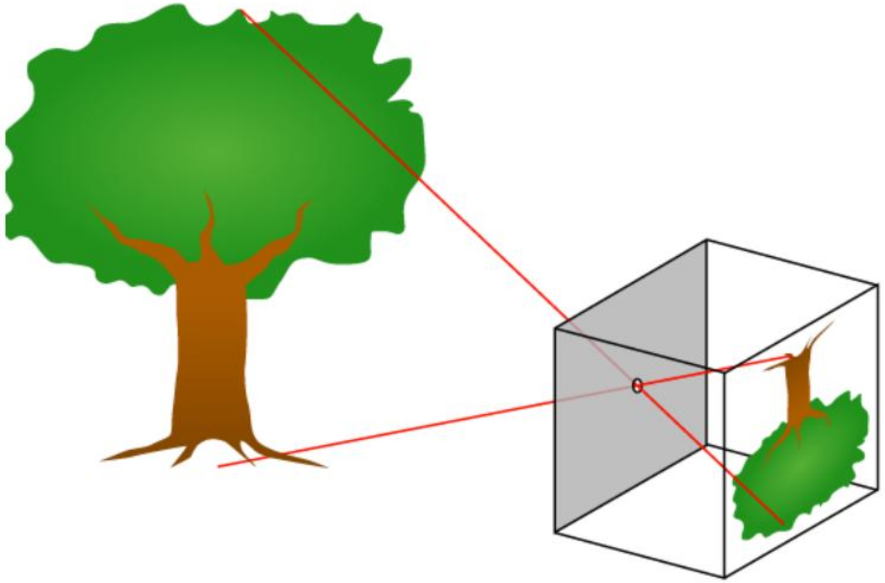


Content

1. Camera Background
2. Pinhole Camera Model
3. Distortion
4. Fisheye Camera Model (MEI)
5. IPM (Inverse Projective Mapping)
6. Assignment

Pin-hole Camera Model

Pinhole imaging

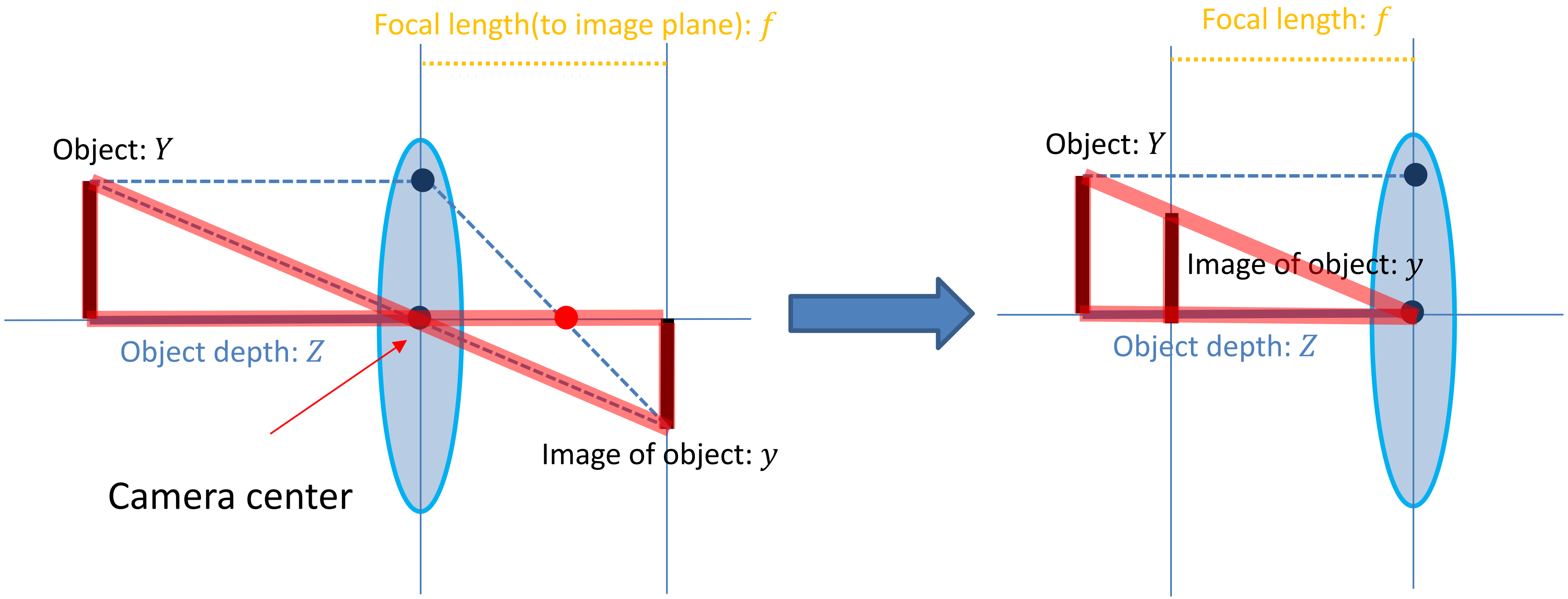


Light from an object passes through a small hole (the pinhole) and projects an inverted image onto a surface or screen on the other side.

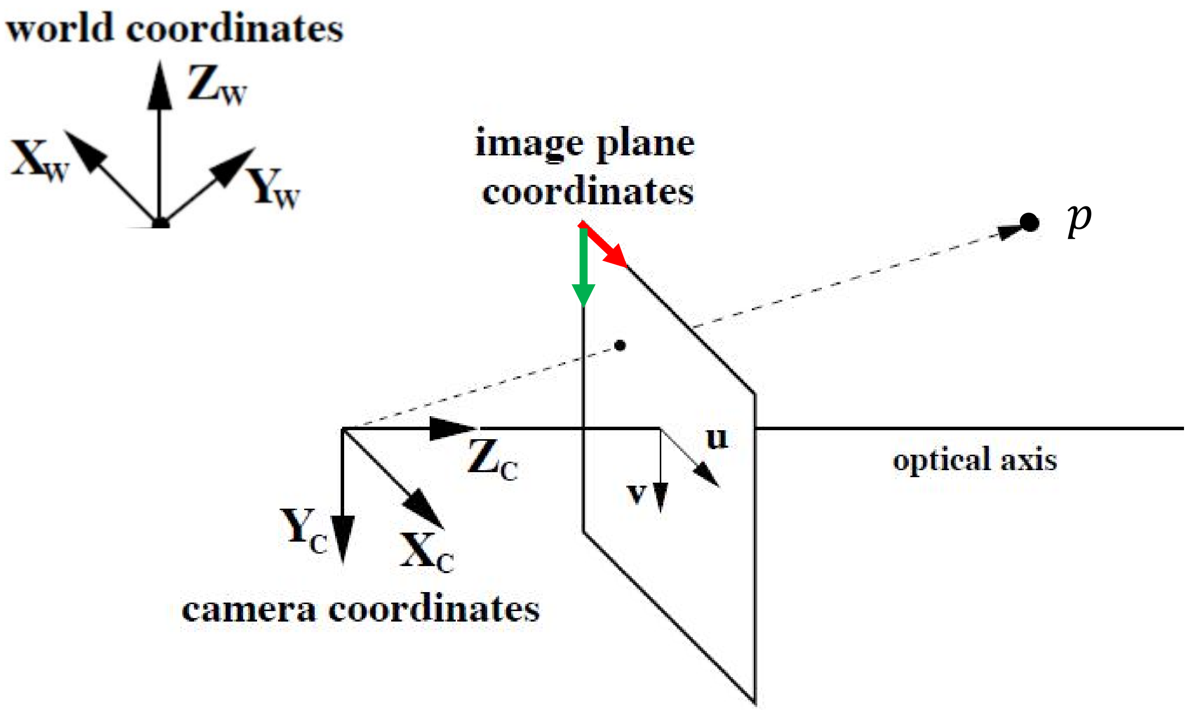
Pin-hole Camera Model

- Assume that the image plane is in front of the lens:

$$-y = f\frac{Y}{Z}$$



Camera Coordinate System



Coordinates：

- World ${}^{w}p = [x_w, y_w, z_w]$
- Camera ${}^{c}p = [x_c, y_c, z_c]$
- Image plane ${}^{I}p = [u, v]$

What's the relationship between ${}^{w}p$, ${}^{C}p$, ${}^{I}p$?

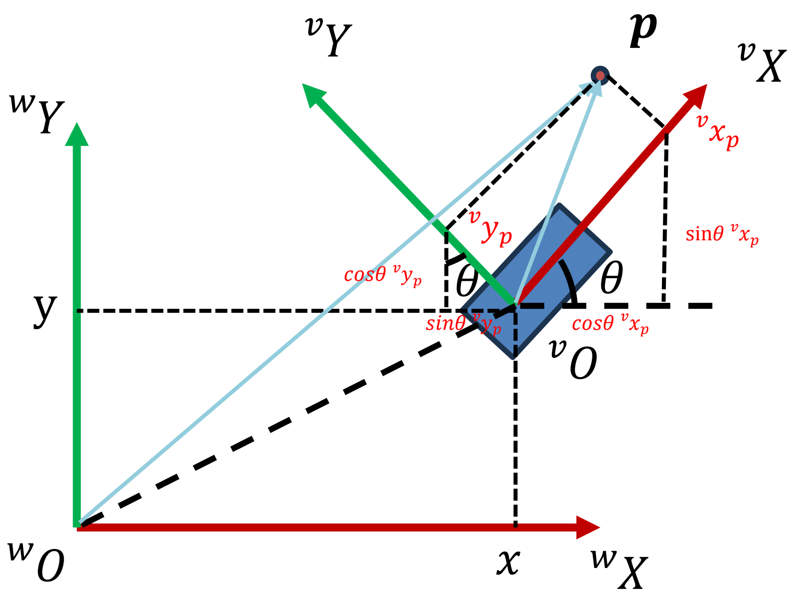
3D Coordinate Transformation

- Recall： 2D Transformations
  - World (Global) Coordinate
  - Vehicle (Local) Coordinate

Vehicle 2D State: $s = [x,\ y,\ \theta]$

$$^{w}\boldsymbol{p} = \underbrace{\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}}_{^{w}R_{v}} {}^{v}\boldsymbol{p} + \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{^{w}t_{v}}$$

$$^{v}\boldsymbol{p} = \underbrace{\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}^{T}}_{^{v}R_{w}} {}^{w}\boldsymbol{p} \underbrace{- \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}^{T} \begin{bmatrix} x \\ y \end{bmatrix}}_{^{v}t_{w}}$$



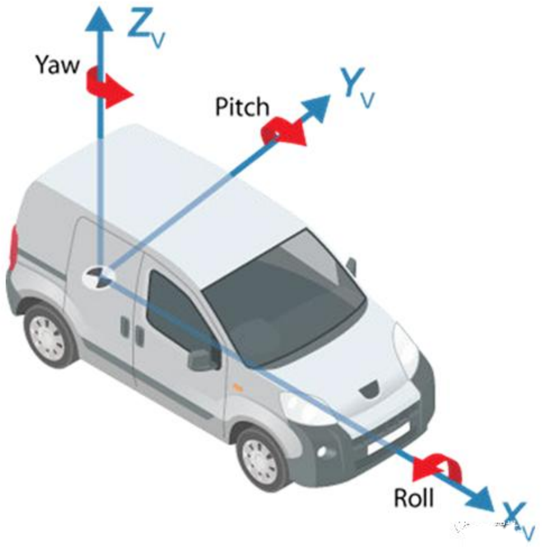
$^{w}R_{v}$, $^{w}t_{v}$: rotation matrix and translation vector from vehicle frame to world frame

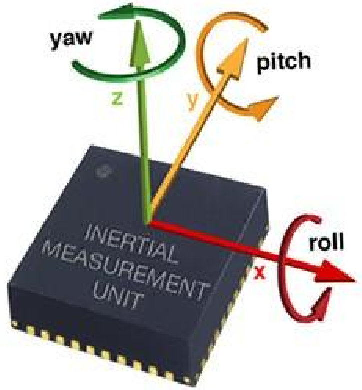
$^{v}R_{w}$ $^{v}t_{w}$: rotation matrix and translation vector from world frame to vehicle frame

3D Coordinate Transformation

Rotation Presentation

- ➢ Euler Angle
  - Roll(横滚) /Pitch(俯仰) / Yaw(偏航)





- ➢ Rotation Matrix

$$R = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix}_{3\times3}$$

- ➢ Quaternion

$$q = [w, x, y, z]^{\top}$$

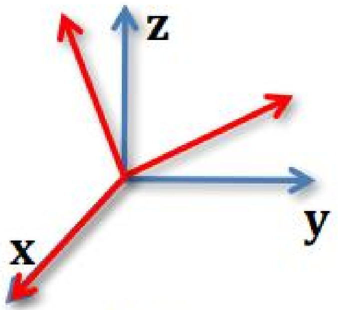
Quaternion -> R

$$\mathbf{R} = \begin{bmatrix} q_w^2 + q_x^2 - q_y^2 - q_z^2 & 2(q_x q_y - q_w q_z) & 2(q_x q_z + q_w q_y) \\ 2(q_x q_y + q_w q_z) & q_w^2 - q_x^2 + q_y^2 - q_z^2 & 2(q_y q_z - q_w q_x) \\ 2(q_x q_z - q_w q_y) & 2(q_y q_z + q_w q_x) & q_w^2 - q_x^2 - q_y^2 + q_z^2 \end{bmatrix}$$

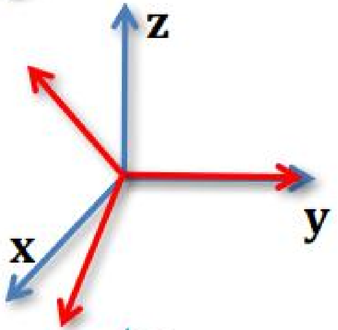
3D Coordinate Transformation

Correspondence: Rotation Matrix - Euler Angles

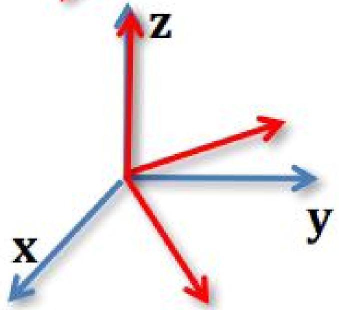
- $\boldsymbol{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$



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

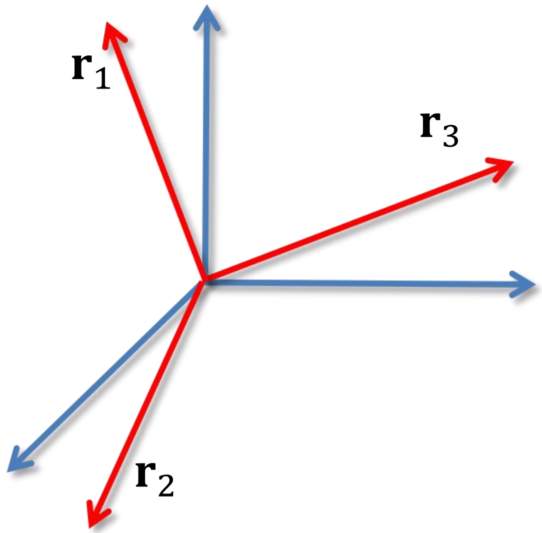


- $\boldsymbol{R}_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$



3D Coordinate Transformation

- Let $\mathbf{R} = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$ be a rotation matrix
- Orthogonal:
  - $\mathbf{r}_i{}^T \cdot \mathbf{r}_j = \begin{cases} 0 \text{ if } i \neq j \\ 1 \text{ if } i = j \end{cases}$
  - $\mathbf{R} \cdot \mathbf{R}^T = \mathbf{I}$
- Special orthogonal:
  - $\det\mathbf{R} = \mathbf{r}_1{}^T \cdot (\mathbf{r}_2 \times \mathbf{r}_3) = \mathbf{r}_1{}^T \cdot \mathbf{r}_1 = 1$
- The set of all rotations forms the Special Orthogonal Group
  - Special orthogonal group
  - 3D rotations: $SO(3)$
  - 2D rotations: $SO(2)$
  - $SO(n) = \{\mathbf{R} \in \mathbb{R}^{n\times n} | \mathbf{R} \cdot \mathbf{R}^T = \mathbf{I}, \det\mathbf{R} = 1\}$

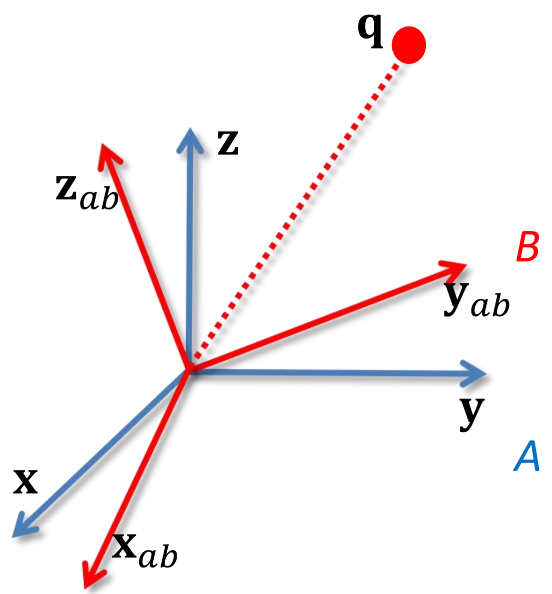


3D Coordinate Transformation

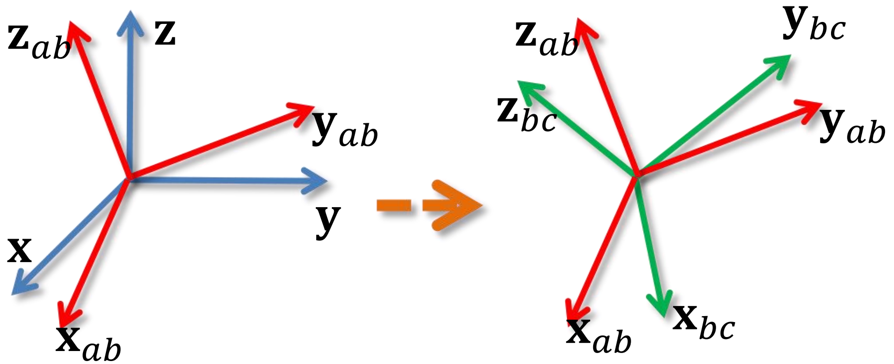
- A transformation that rotates the coordinates of a point from frame *B* to frame *A*
  - Let $\mathbf{q}_b = [x_b,\ y_b,\ z_b]^T \in \mathbb{R}^3$ be coordinate of point $\mathbf{q}$ in frame *B*
  - Let $\mathbf{q}_a$ be coordinate of point $\mathbf{q}$ in frame *A*
  - $\mathbf{q}_a = x_b \cdot \mathbf{x}_{ab} + y_b \cdot \mathbf{y}_{ab} + z_b \cdot \mathbf{z}_{ab}$

$$= [\mathbf{x}_{ab}, \mathbf{y}_{ab}, \mathbf{z}_{ab}] \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} = {}^a\mathbf{R}_b \cdot \mathbf{q}_b$$

Rotation matrix from B to A

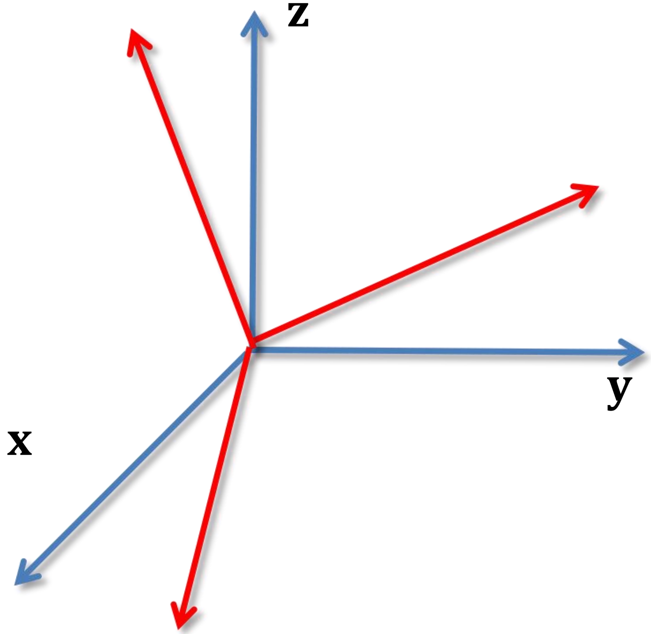


- Composition Rule
  - ${}^a\mathbf{R}_c = {}^a\mathbf{R}_b \cdot {}^b\mathbf{R}_c$



3D Coordinate Transformation

- Any rotation can be described by three successive rotations about linear independent axes
- However, this is an almost 1-1 transform with singularities:
  - $\boldsymbol{R}_z(\psi) \cdot \boldsymbol{R}_x(\phi) \cdot \boldsymbol{R}_y(\theta) \Rightarrow \boldsymbol{R}$
  - $\boldsymbol{R}_z(\psi) \cdot \boldsymbol{R}_x(\phi) \cdot \boldsymbol{R}_y(\theta) \nLeftarrow \boldsymbol{R}$



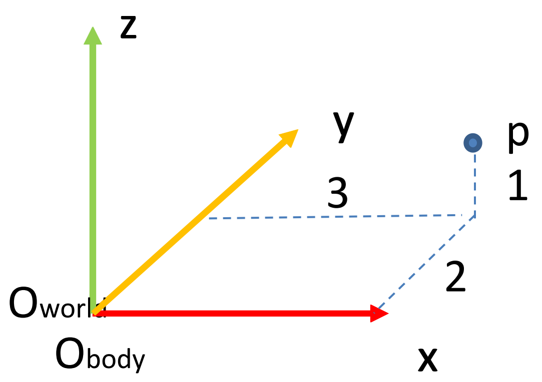
3D Coordinate Transformation

- Different Euler angle conversions:

Proper Euler angles	Tait-Bryan angles
$X_1Z_2X_3 = \begin{bmatrix} c_2 & -c_3s_2 & s_2s_3 \\ c_1s_2 & c_1c_2c_3 - s_1s_3 & -c_3s_1 - c_1c_2s_3 \\ s_1s_2 & c_1s_3 + c_2c_3s_1 & c_1c_3 - c_2s_1s_3 \end{bmatrix}$	$X_1Z_2Y_3 = \begin{bmatrix} c_2c_3 & -s_2 & c_2s_3 \\ s_1s_3 + c_1c_3s_2 & c_1c_2 & c_1s_2s_3 - c_3s_1 \\ c_3s_1s_2 - c_1s_3 & c_2s_1 & c_1c_3 + s_1s_2s_3 \end{bmatrix}$
$X_1Y_2X_3 = \begin{bmatrix} c_2 & s_2s_3 & c_3s_2 \\ s_1s_2 & c_1c_3 - c_2s_1s_3 & -c_1s_3 - c_2c_3s_1 \\ -c_1s_2 & c_3s_1 + c_1c_2s_3 & c_1c_2c_3 - s_1s_3 \end{bmatrix}$	$X_1Y_2Z_3 = \begin{bmatrix} c_2c_3 & -c_2s_3 & s_2 \\ c_1s_3 + c_3s_1s_2 & c_1c_3 - s_1s_2s_3 & -c_2s_1 \\ s_1s_3 - c_1c_3s_2 & c_3s_1 + c_1s_2s_3 & c_1c_2 \end{bmatrix}$
$Y_1X_2Y_3 = \begin{bmatrix} c_1c_3 - c_2s_1s_3 & s_1s_2 & c_1s_3 + c_2c_3s_1 \\ s_2s_3 & c_2 & -c_3s_2 \\ -c_3s_1 - c_1c_2s_3 & c_1s_2 & c_1c_2c_3 - s_1s_3 \end{bmatrix}$	$Y_1X_2Z_3 = \begin{bmatrix} c_1c_3 + s_1s_2s_3 & c_3s_1s_2 - c_1s_3 & c_2s_1 \\ c_2s_3 & c_2c_3 & -s_2 \\ c_1s_2s_3 - c_3s_1 & c_1c_3s_2 + s_1s_3 & c_1c_2 \end{bmatrix}$
$Y_1Z_2Y_3 = \begin{bmatrix} c_1c_2c_3 - s_1s_3 & -c_1s_2 & c_3s_1 + c_1c_2s_3 \\ c_3s_2 & c_2 & s_2s_3 \\ -c_1s_3 - c_2c_3s_1 & s_1s_2 & c_1c_3 - c_2s_1s_3 \end{bmatrix}$	$Y_1Z_2X_3 = \begin{bmatrix} c_1c_2 & s_1s_3 - c_1c_3s_2 & c_3s_1 + c_1s_2s_3 \\ s_2 & c_2c_3 & -c_2s_3 \\ -c_2s_1 & c_1s_3 + c_3s_1s_2 & c_1c_3 - s_1s_2s_3 \end{bmatrix}$
$Z_1Y_2Z_3 = \begin{bmatrix} c_1c_2c_3 - s_1s_3 & -c_3s_1 - c_1c_2s_3 & c_1s_2 \\ c_1s_3 + c_2c_3s_1 & c_1c_3 - c_2s_1s_3 & s_1s_2 \\ -c_3s_2 & s_2s_3 & c_2 \end{bmatrix}$	$Z_1Y_2X_3 = \begin{bmatrix} c_1c_2 & c_1s_2s_3 - c_3s_1 & s_1s_3 + c_1c_3s_2 \\ c_2s_1 & c_1c_3 + s_1s_2s_3 & c_3s_1s_2 - c_1s_3 \\ -s_2 & c_2s_3 & c_2c_3 \end{bmatrix}$
$Z_1X_2Z_3 = \begin{bmatrix} c_1c_3 - c_2s_1s_3 & -c_1s_3 - c_2c_3s_1 & s_1s_2 \\ c_3s_1 + c_1c_2s_3 & c_1c_2c_3 - s_1s_3 & -c_1s_2 \\ s_2s_3 & c_3s_2 & c_2 \end{bmatrix}$	$Z_1X_2Y_3 = \begin{bmatrix} c_1c_3 - s_1s_2s_3 & -c_2s_1 & c_1s_3 + c_3s_1s_2 \\ c_3s_1 + c_1s_2s_3 & c_1c_2 & s_1s_3 - c_1c_3s_2 \\ -c_2s_3 & s_2 & c_2c_3 \end{bmatrix}$

3D Coordinate Transformation

- World frame (fixed)
- Body frame (moving)



The two coordinate systems coincide.

P is a fixed point in body frame.

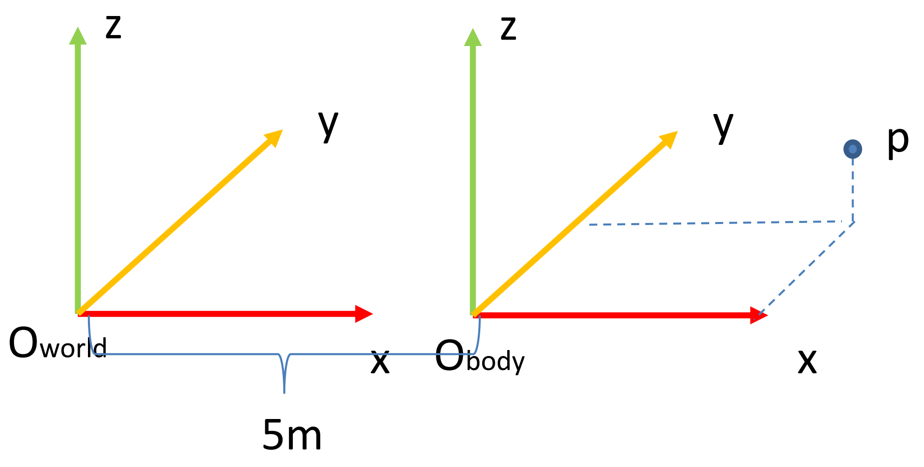
$$ {}^{b}p = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} $$

$$ {}^{w}p = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{{}^{w}R_b} {}^{b}p + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}}_{{}^{w}t_b} \qquad {}^{w}p = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} $$

3D Coordinate Transformation

- World frame (fixed)
- Body frame (moving)

P is a fixed point in body frame.



$$^{b}p = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}$$

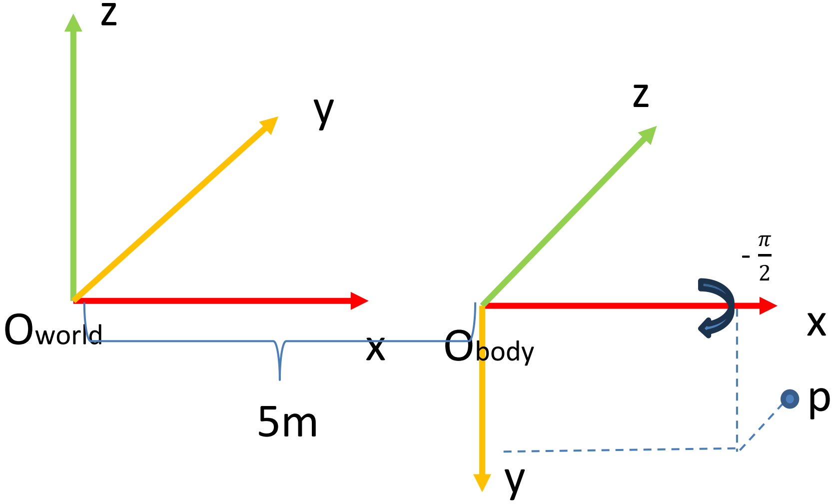
$$^{w}p = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{^{w}R_b} {}^{b}p + \underbrace{\begin{bmatrix} 5 \\ 0 \\ 0 \end{bmatrix}}_{^{w}t_b} \qquad {}^{w}p = \begin{bmatrix} 8 \\ 2 \\ 1 \end{bmatrix}$$

3D Coordinate Transformation

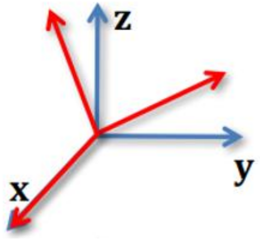
- World frame (fixed)
- Body frame (moving)

P is a fixed point in body frame.

$$^{b}p = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}$$



$$\boldsymbol{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

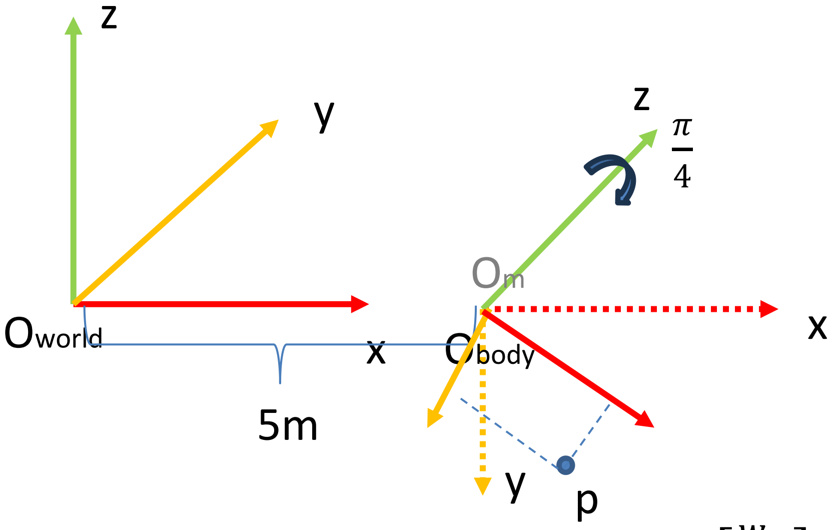


$$^{w}p = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}}_{^{w}R_b} {}^{b}p + \underbrace{\begin{bmatrix} 5 \\ 0 \\ 0 \end{bmatrix}}_{^{w}t_b} \qquad {}^{w}p = \begin{bmatrix} 8 \\ 1 \\ -2 \end{bmatrix}$$

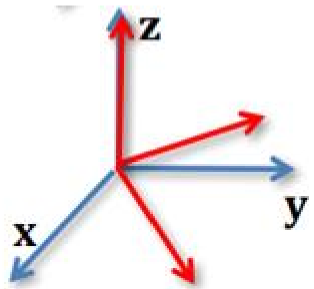
3D Coordinate Transformation

- World frame (fixed)
- Body frame (moving)

P is a fixed point in body frame.



$$\boldsymbol{R}_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



$${}^{b}p = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}$$

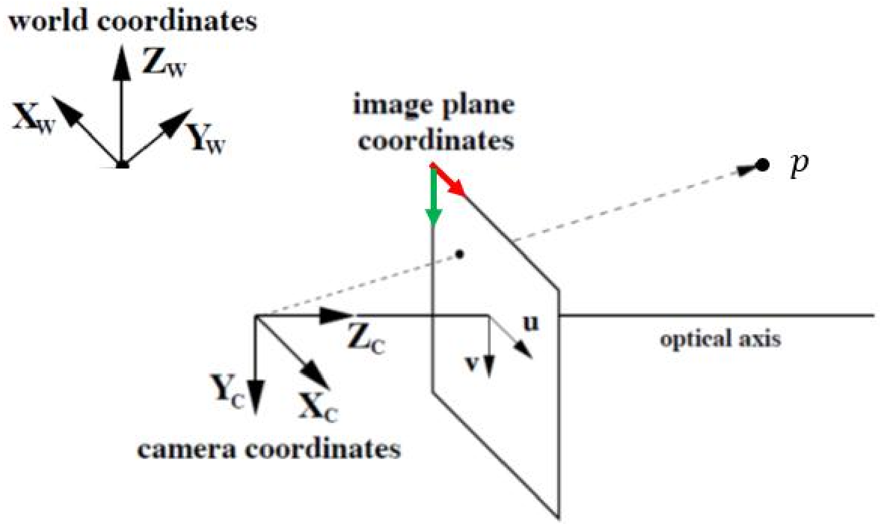
$$\begin{bmatrix} {}^{w}p \\ 1 \end{bmatrix} = \overbrace{\begin{bmatrix} {}^{w}\boldsymbol{R}_{b_m} & {}^{w}t_{b_m} \\ \boldsymbol{0} & 1 \end{bmatrix}}^{{}^{w}\boldsymbol{T}_{b_m}} \overbrace{\begin{bmatrix} {}^{b_m}\boldsymbol{R}_{b} & {}^{b_m}t_{b} \\ \boldsymbol{0} & 1 \end{bmatrix}}^{{}^{b_m}\boldsymbol{T}_{b}} \begin{bmatrix} {}^{b}p \\ 1 \end{bmatrix}$$

$${}^{b_m}p = \underbrace{\begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{{}^{b_m}\boldsymbol{R}_{b}} {}^{b}p + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}}_{{}^{b_m}t_{b}}$$

$${}^{w}p = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}}_{{}^{w}\boldsymbol{R}_{b_m}} {}^{b_m}p + \underbrace{\begin{bmatrix} 5 \\ 0 \\ 0 \end{bmatrix}}_{{}^{w}t_{b_m}} = \begin{bmatrix} 5 + \frac{\sqrt{2}}{2} \\ 1 \\ -\frac{5\sqrt{2}}{2} \end{bmatrix}$$

Perspective Projection

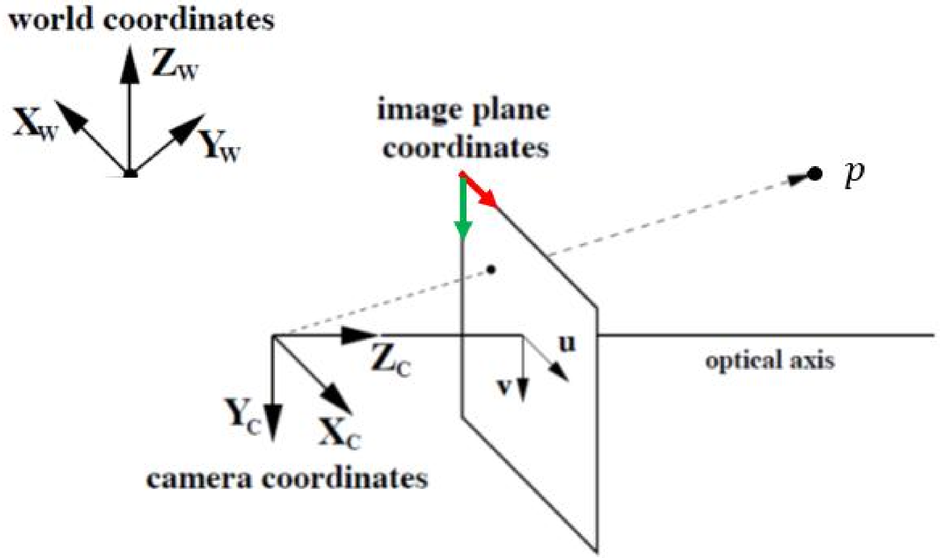
- Optical axis is the z-axis
- The image plane (u, v) is perpendicular to the optical axis
- Intersection of the image plane and the optical axis is the image center (u_0, v_0)
- f is the distance of the image plane from the origin (in pixels)
- Point in Camera Coordinate: ${}^{c}p = [x_c, y_c, z_c]$
- Formulation:
  - $u = \frac{f x_c}{z_c} + u_0$
  - $v = \frac{f y_c}{z_c} + v_0$



Perspective Projection

- Matrix form:

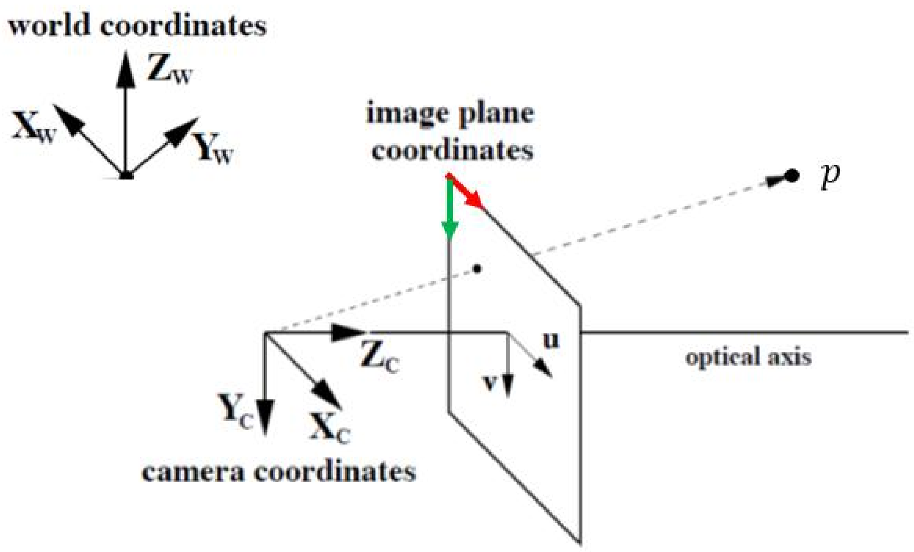
$$\lambda\begin{pmatrix}u\\v\\1\end{pmatrix}=\begin{pmatrix}f & 0 & u_o\\0 & f & v_o\\0 & 0 & 1\end{pmatrix}\begin{pmatrix}x_c\\y_c\\z_c\end{pmatrix}$$



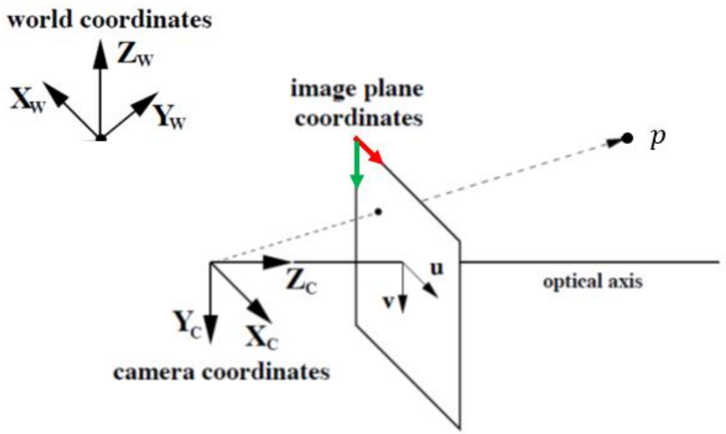
Perspective Projection

- From camera to world:

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = \begin{pmatrix} {}^c\boldsymbol{R}_w & {}^c\boldsymbol{t}_w \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix}$$



Perspective Projection

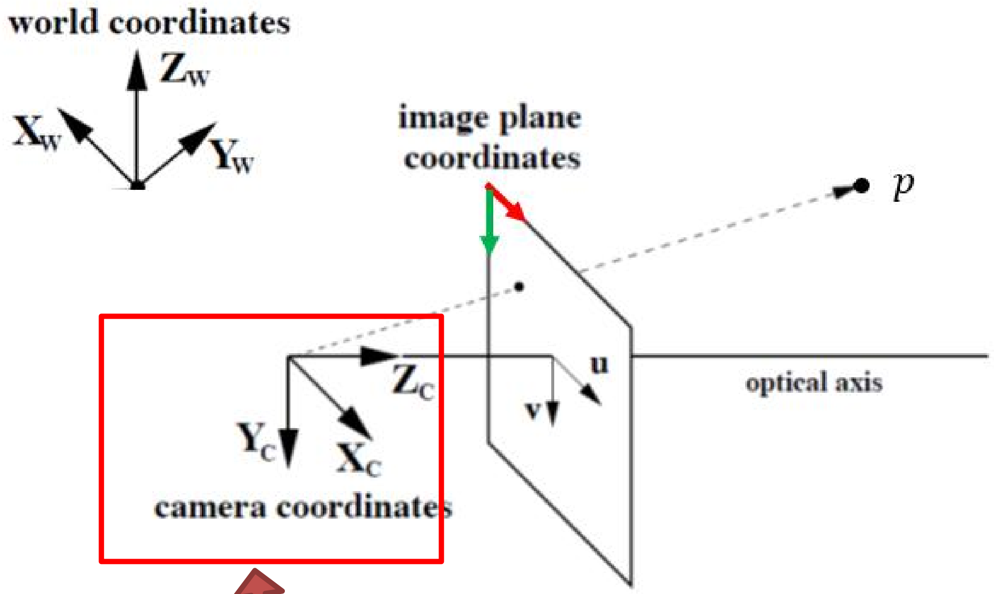


$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & u_o \\ 0 & f & v_o \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix}$$

- From camera to world:

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = \boxed{\begin{pmatrix} {}^c\boldsymbol{R}_w & {}^c\boldsymbol{t}_w \\ 0 & 1 \end{pmatrix}} \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix}$$

Pin-hole Camera Model



$$\boxed{\lambda} \boxed{\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}} = \boxed{\begin{pmatrix} f & 0 & u_o \\ 0 & f & v_o \\ 0 & 0 & 1 \end{pmatrix}} \boxed{\begin{pmatrix} {}^{c}\boldsymbol{R}_{w} & {}^{c}\boldsymbol{t}_{w} \end{pmatrix}} \boxed{\begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix}} = P \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix}$$

Depth

Pixel Values

Intrinsic Calibration Matrix

(Inverse) Camera Pose

${}^{w}\boldsymbol{R}_{c}$ ${}^{w}\boldsymbol{t}_{c}$

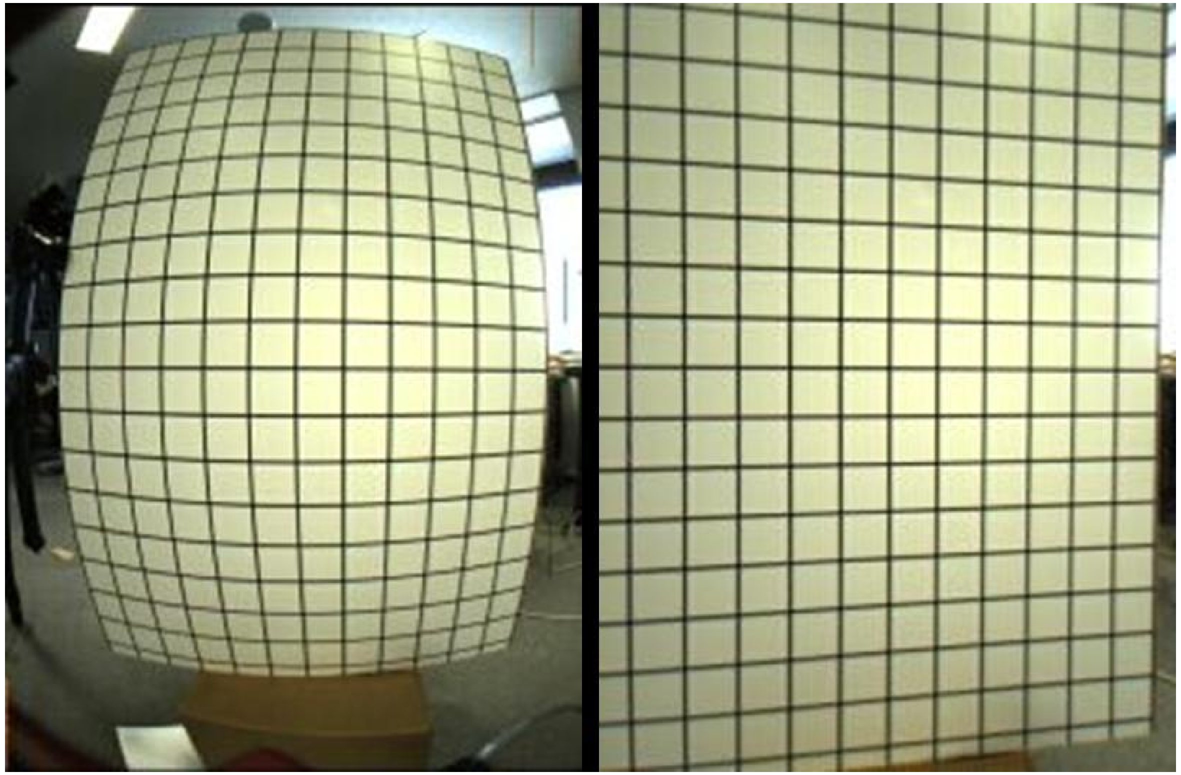
World Point

Content

1. Camera Background
2. Pinhole Camera Model
3. Distortion
4. Fisheye Camera Model (MEI)
5. IPM (Inverse Projective Mapping)
6. Assignment

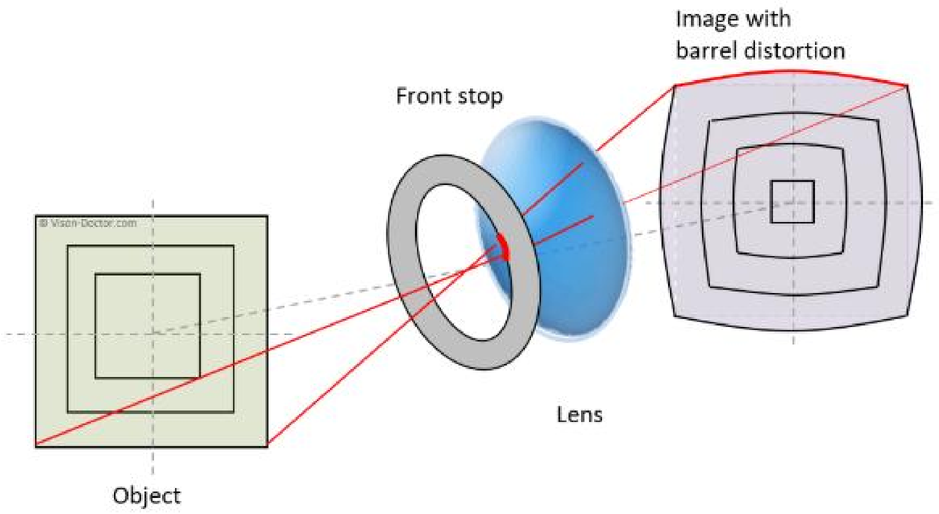
Camera Calibration

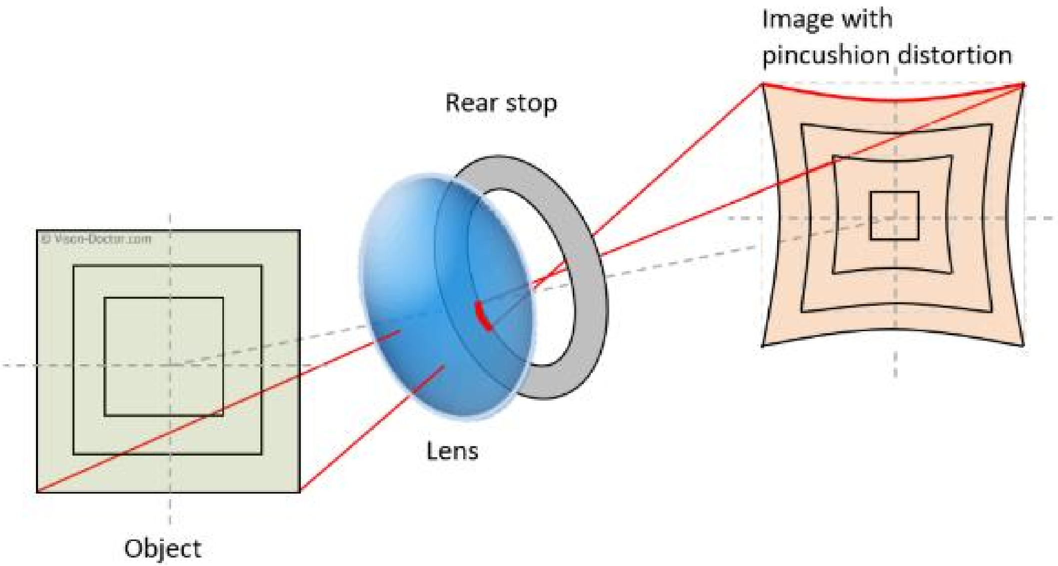
- Straight lines should be straight



Distortion model

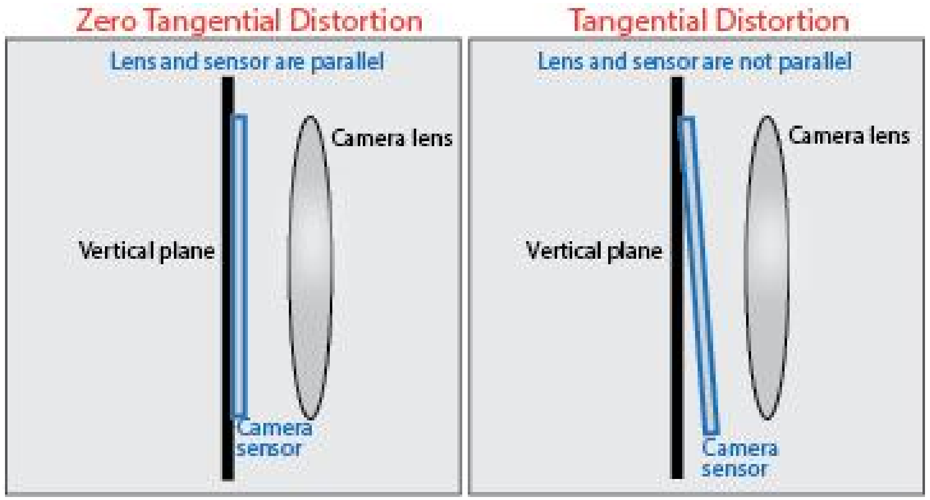
- Tangential and radial distortion model
  - Radial distortion

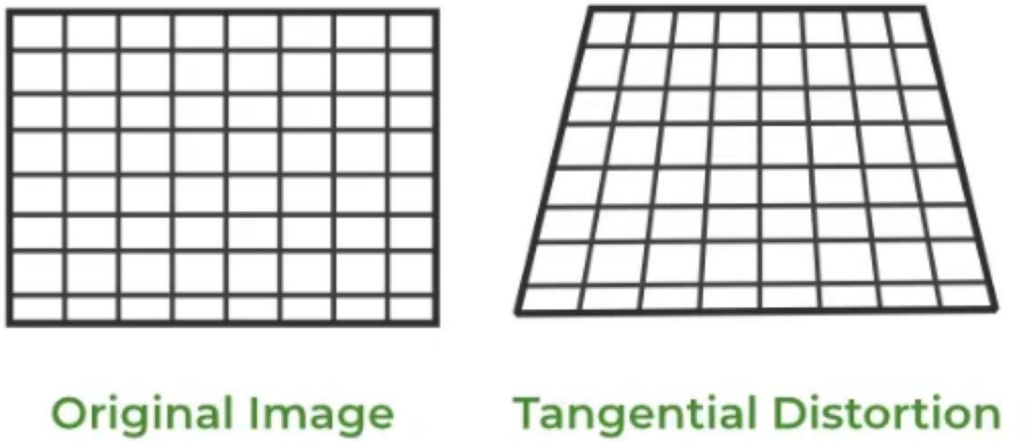




Distortion model

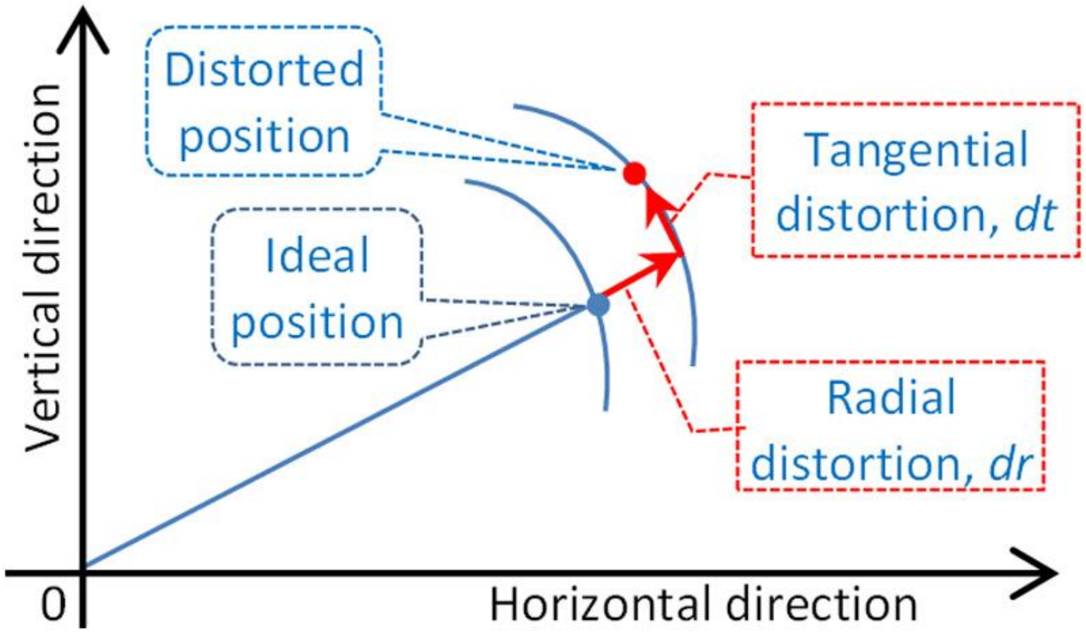
- Tangential and radial distortion model
  - Tangential distortion





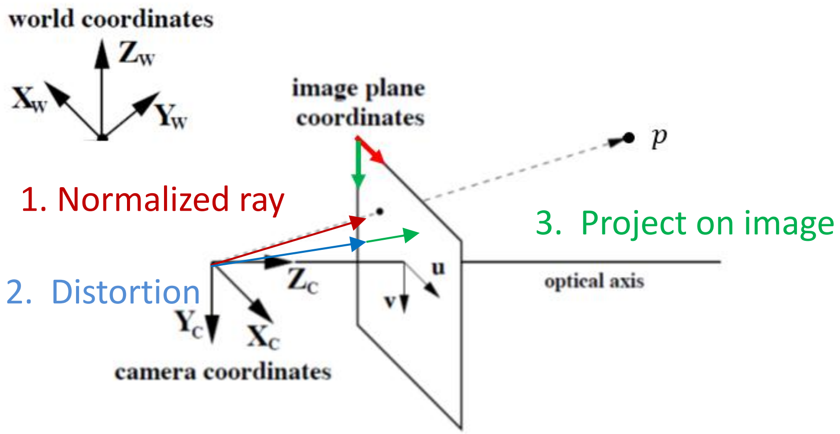
Camera Calibration

- Tangential and radial distortion model



Distortion model

- Tangential and radial distortion model
  - $k1,\ k2,\ [k3,\ [k4,\ [k5,\ [k6]]]]$ are the radial coefficients
  - $p1$ and $p2$ are the tangential distortion coefficients



1. Project to unit plane (z = 1)

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x_c/z_c \\ y_c/z_c \end{bmatrix}$$

2. Distortion:
(Distortion coefficient D)

$$r^2 = x'^2 + y'^2$$

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} x' \frac{1+k_1 r^2 + k_2 r^4 + k_3 r^6}{1+k_4 r^2 + k_5 r^4 + k_6 r^6} + 2p_1 x' y' + p_2(r^2 + 2x'^2) \\ y' \frac{1+k_1 r^2 + k_2 r^4 + k_3 r^6}{1+k_4 r^2 + k_5 r^4 + k_6 r^6} + p_1(r^2 + 2y'^2) + 2p_2 x' y' \end{bmatrix}$$

3. Image Projection:
(Camera matrix K)

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x x'' + c_x \\ f_y y'' + c_y \end{bmatrix}$$

Projection Example

- Pinhole + Tangential and radial Distortion

$$ {}^{c}p = [6,\ 4,\ 2] $$


```yaml
%YAML:1.0
---
calibration_Time: "2024年12月12日 星期四 16时32分21秒"
nrOfFrames: 57
image_Width: 640
image_Height: 480
board_Width: 7
board_Height: 5
square_Size: 1.0000000149011612e-01
Camera_Matrix: !!opencv-matrix
   rows: 3
   cols: 3
   dt: d
   data: [ 5.6865194941315929e+02, 0., 3.2513762614210782e+02, 0.,
       7.6375465300688415e+02, 2.3177702561676526e+02, 0., 0., 1. ]
Distortion_Coefficients: !!opencv-matrix
   rows: 1
   cols: 5
   dt: d
   data: [ 1.1726086573014034e-02, -5.2021897536855632e-01,
       -4.8681362994895585e-03, 9.6693728067371231e-04,
       2.2217257704057318e+00 ]
```


camera_calib.yaml

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

$$ (k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3) $$

$$ \begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} x' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} + 2p_1x'y' + p_2(r^2+2x'^2) \\ y' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} + p_1(r^2+2y'^2) + 2p_2x'y' \end{bmatrix} $$

1. Project to unit plane : $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x_c/z_c \\ y_c/z_c \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$

2. Distortion:$r^2 = 9 + 4 = 13$

$$ \begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} x'(1 + k_1r^2 + k_2r^4 + k_3r^6) + 2p_1x'y' + p_2(r^2 + 2x'^2) \\ y'(1 + k_1r^2 + k_2r^4 + k_3r^6) + p_1\left(r^2 + 2y'^2\right) + 2p_2x'y' \end{bmatrix} $$

3. Image Projection:

$$ \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x x'' + c_x \\ f_y y'' + c_y \end{bmatrix} = \begin{bmatrix} 568.65\, x'' + 325.13 \\ 763.75 y'' + 231.77 \end{bmatrix} $$

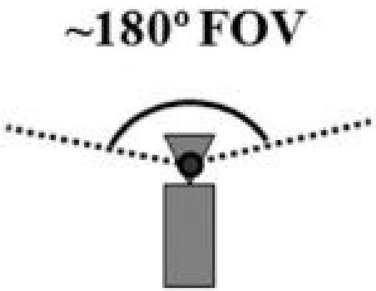
Content

1. Camera Background
2. Pinhole Camera Model
3. Distortion
4. Fisheye Camera Model (MEI)
5. IPM (Inverse Projective Mapping)
6. Assignment

Fisheye Camera

- Dioptric camera

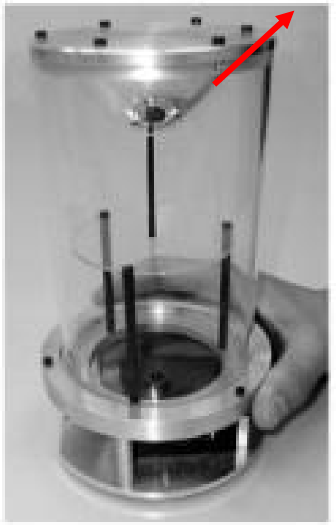
  折射

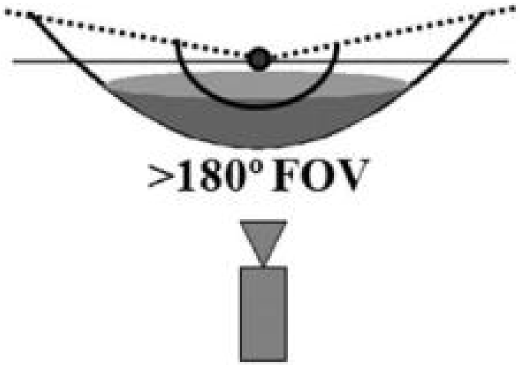


- Catadioptric camera

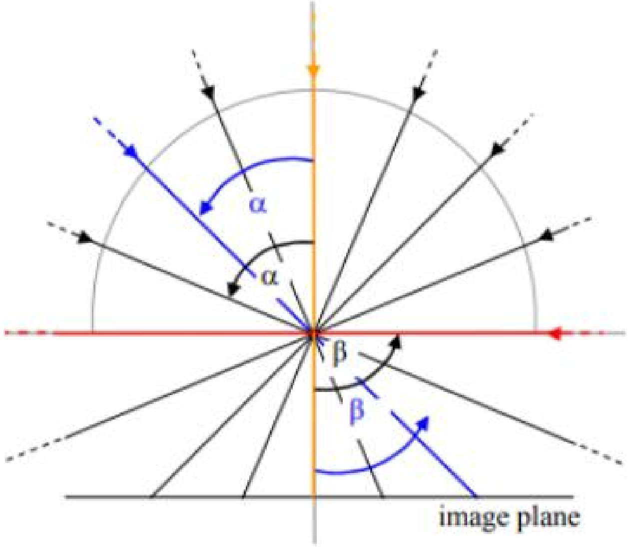
  反射折射

mirror

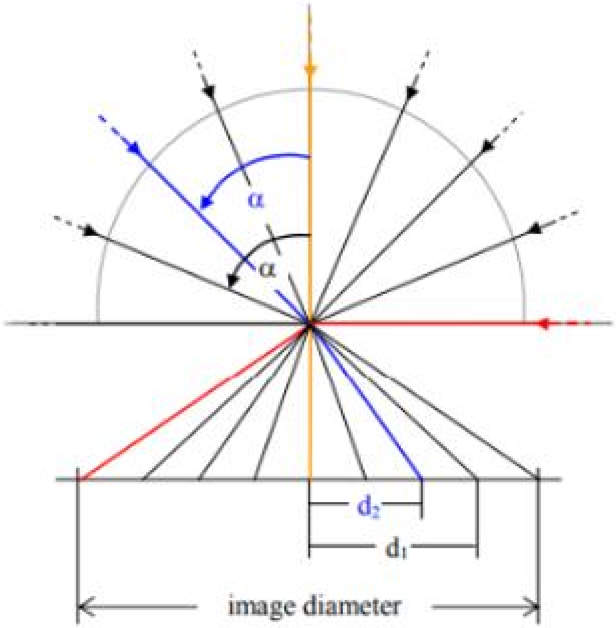




Fisheye Camera Model

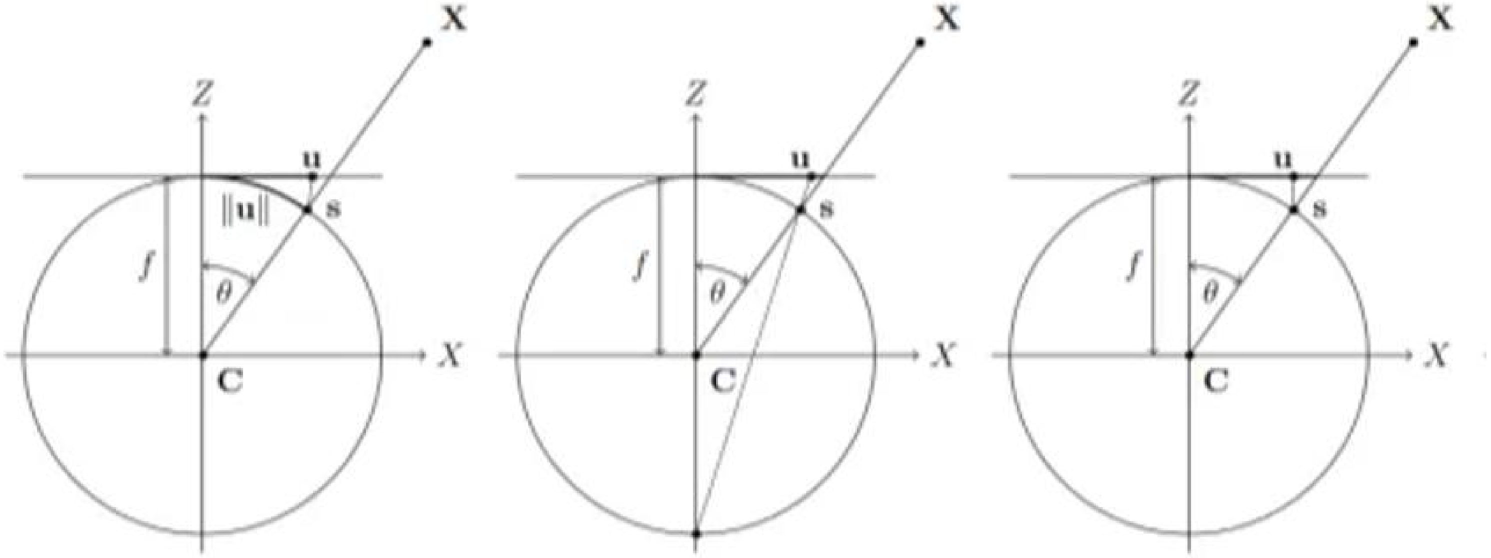


Pinhole perspective model



Fisheye perspective model

Classical Fisheye Camera Models



... ...

(a) Equidistant
等距

$$u = f\theta$$

(b) Stereographic
体视

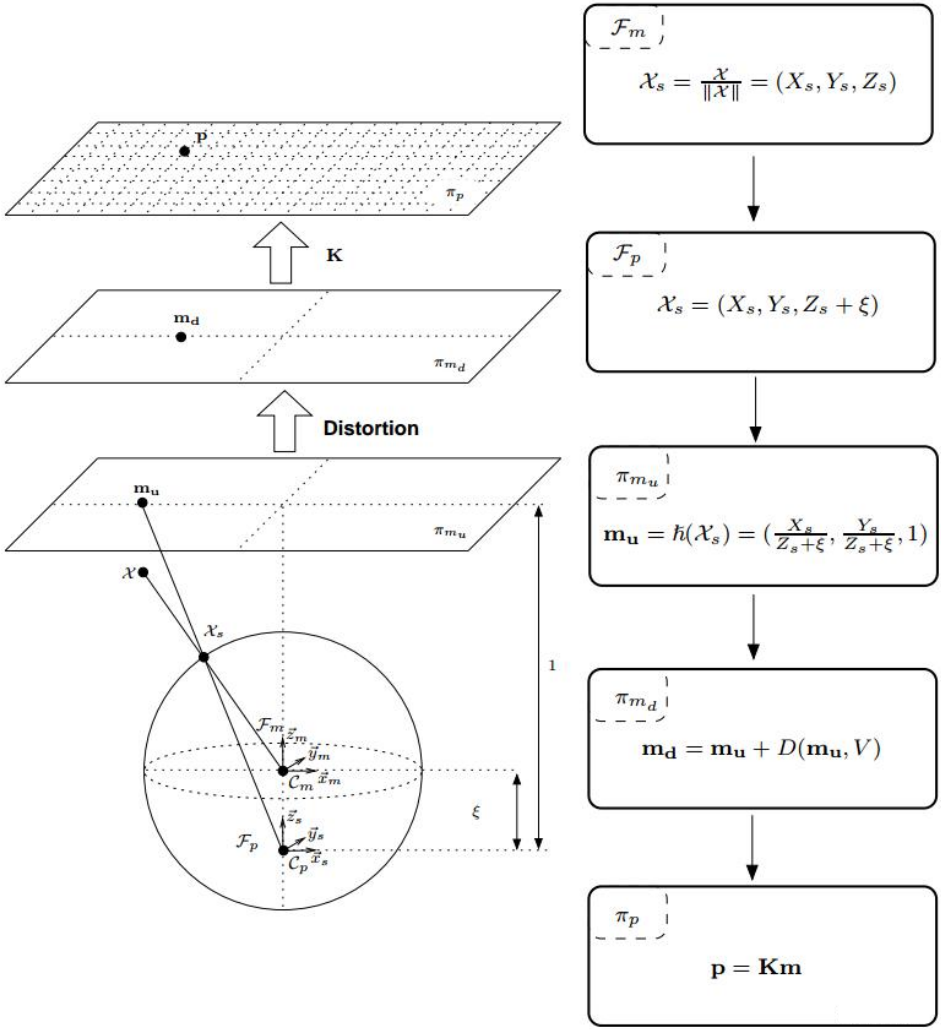
$$u = 2f\tan\frac{\theta}{2}$$

(c) Orthographic
正交

$$u = f\sin\theta$$

OpenCV::Fisheye

MEI [1] Camera Model



1. 在 mirror 坐标系上的世界点映射到单位圆上：

$$(\mathcal{X})_{\mathcal{F}_m} \rightarrow (\mathcal{X}_s)_{\mathcal{F}_m} = \frac{\mathcal{X}}{\|\mathcal{X}\|} = (X_s, Y_s, Z_s)$$

2. 将这些点转换为以 $\mathcal{C}_p = (0, 0, \xi)$ 为中心的新参考系上：

$$(\mathcal{X}_s)_{\mathcal{F}_m} \rightarrow (\mathcal{X}_s)_{\mathcal{F}_p} = (X_s, Y_s, Z_s + \xi)$$

3. 将它们投影到归一化图像平面上：

$$\mathbf{m}_u = \left(\frac{X_s}{Z_s + \xi}, \frac{Y_s}{Z_s + \xi}, 1\right) = \hbar(\mathcal{X}_s)$$

4. 添加上径向和切向畸变：

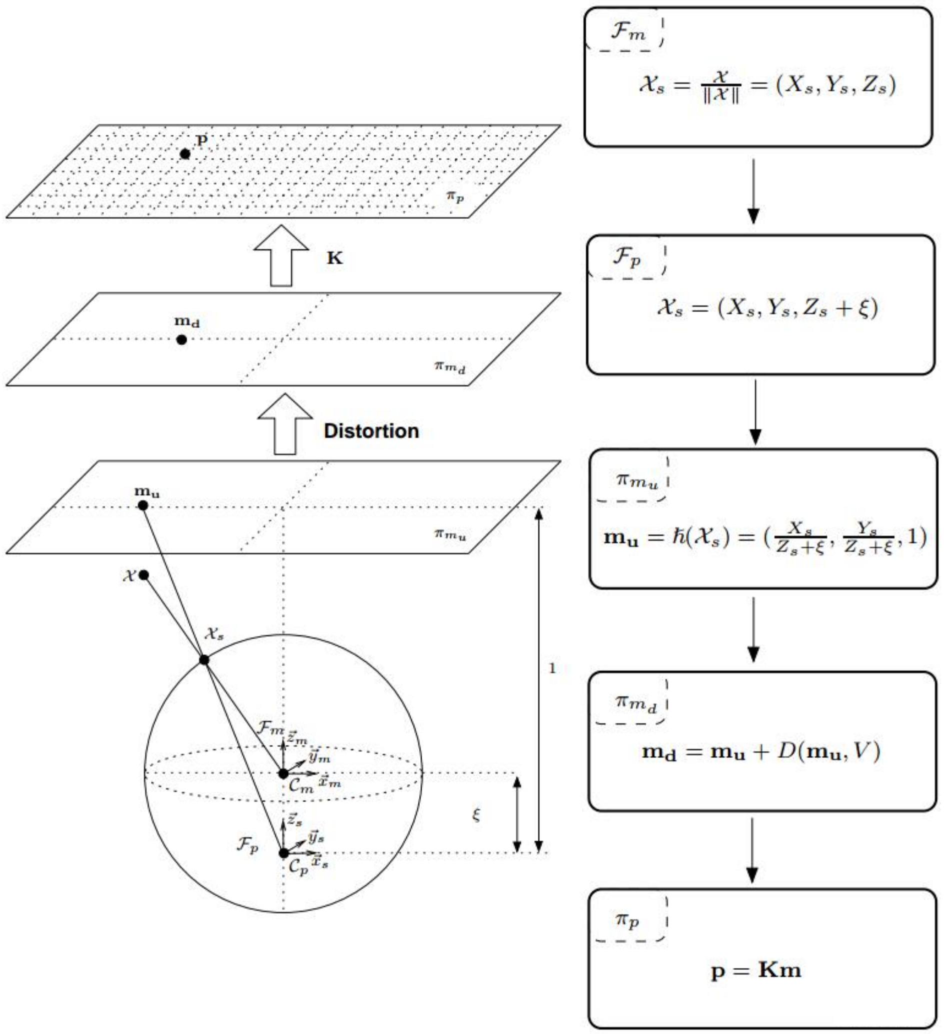
$$\mathbf{m}_d = \mathbf{m}_u + D(\mathbf{m}_u, V)$$

5. 最终投影涉及一个广义相机投影矩阵 $\mathbf{K}$(其中 γ 为广义焦距，(u_0, v_0) 为主点，s 斜度)：

$$\mathbf{p} = \mathbf{K}\mathbf{m} = \begin{bmatrix} \gamma & \gamma s & u_0 \\ 0 & \gamma r & v_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{m} = k(\mathbf{m})$$

[1] C. Mei and P. Rives, "Single View Point Omnidirectional Camera Calibration from Planar Grids," *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Rome, Italy, 2007, pp. 3945-3950, doi: 10.1109/ROBOT.2007.364084.

Example: MEI Camera Model



0_intrinsic.yaml


```yaml
%YAML:1.0
---
model_type: MEI
camera_name: camera
image_width: 1920
image_height: 1080
mirror_parameters:
   xi: 1.0
distortion_parameters:
   k1: 0.0
   k2: 0.0
   p1: 0.0
   p2: 0.0
projection_parameters:
   gamma1: 935.307436087
   gamma2: 935.307436087
   u0: 960
   v0: 540
```


$$^{c}p = [2,\ 2,\ 1]$$

1. Normalized ray vector:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x_c/\|p\| \\ y_c/\|p\| \\ z_c/\|p\| \end{bmatrix} = \begin{bmatrix} 0.67 \\ 0.67 \\ 0.33 \end{bmatrix}$$

2. Change to C_p cooridinate:

$$\begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' + \xi \end{bmatrix} = \begin{bmatrix} 0.67 \\ 0.67 \\ 1.33 \end{bmatrix}$$

3. Project to unit plane:

$$\begin{bmatrix} x''' \\ y''' \end{bmatrix} = \begin{bmatrix} x''/z'' \\ y''/z'' \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

4. Distortion:

5. Image projection:

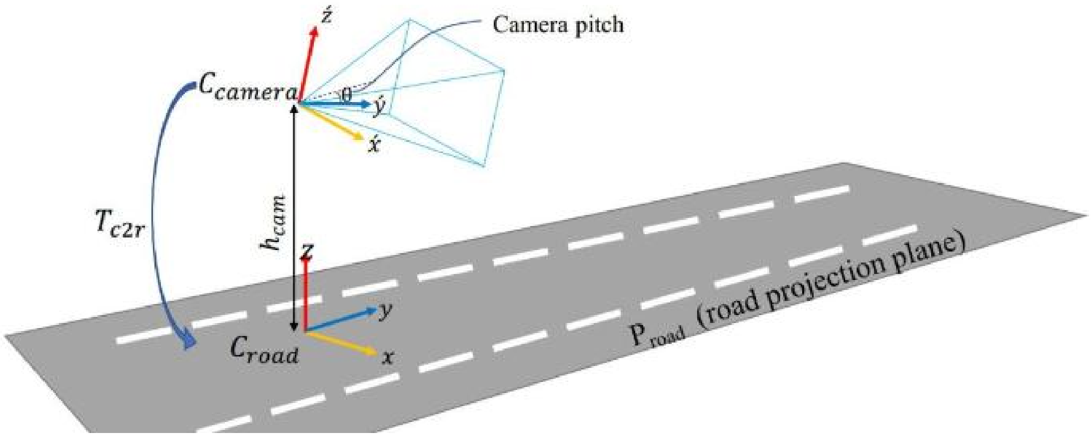
$$\begin{bmatrix} u \\ y \end{bmatrix} = \begin{bmatrix} f_x x''' + c_x \\ f_y y''' + c_y \end{bmatrix} = \begin{bmatrix} 935.3\ x''' + 960 \\ 935.3\ y''' + 540 \end{bmatrix} = \begin{bmatrix} 1427.66 \\ 1007.66 \end{bmatrix}$$

Content

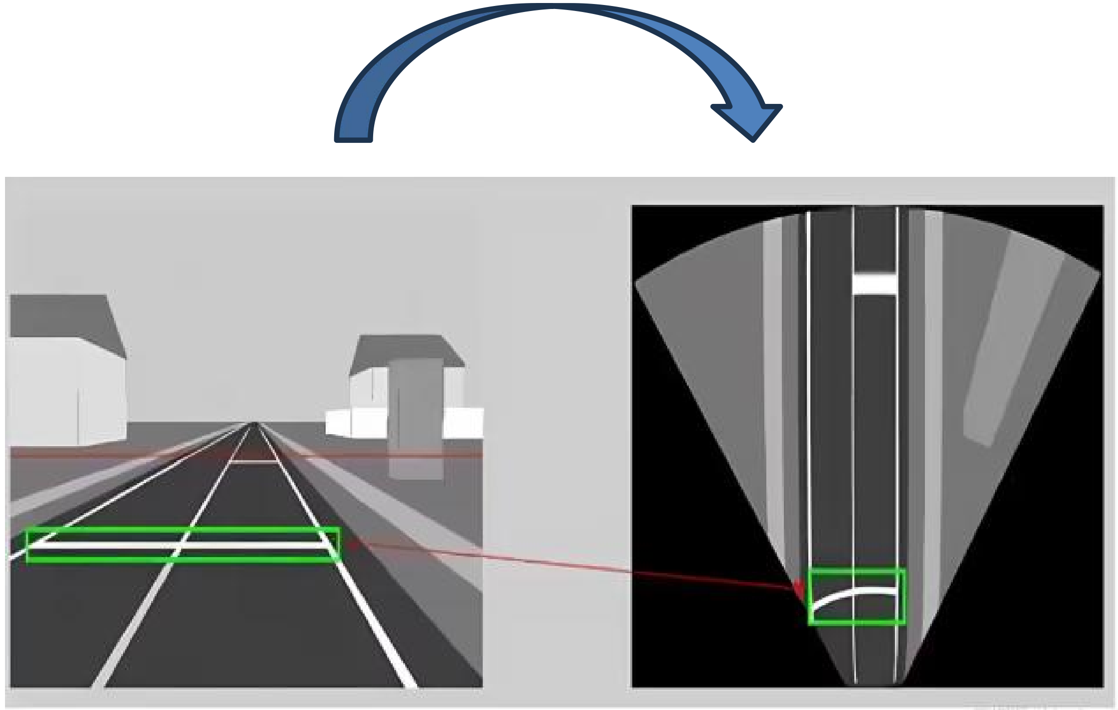
1. Camera Background
2. Pinhole Camera Model
3. Distortion
4. Fisheye Camera Model (MEI)
5. IPM (Inverse Projective Mapping)
6. Assignment

IPM Image

- IPM (Inverse Projective Mapping)



Inverse Projective



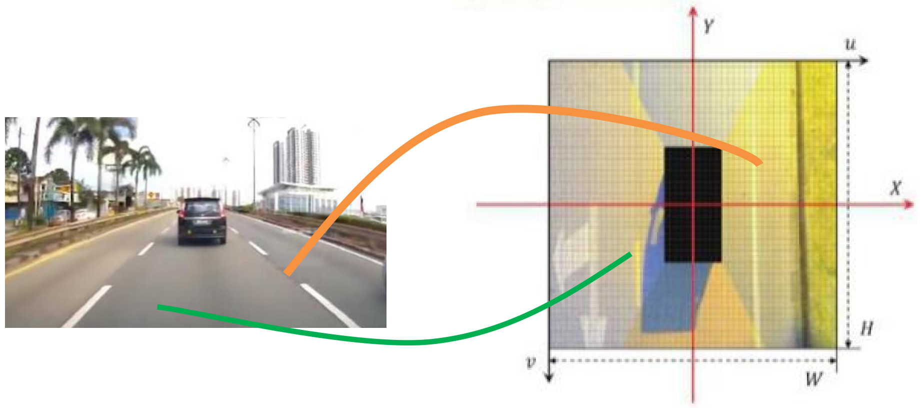
IPM Image

Method I:

r, g, b →

$(u,\ v)$ $\xrightarrow[\text{Inverse projection}]{\text{Camera model}}$ $\lambda(x_c, y_c, 1)$ $\xrightarrow{{}^{v}R_c\ ,\ {}^{v}t_c}$ $(x_v, y_v, 0)$ → (u_{ipm}, v_{ipm})

Raw image plane | Camera coordinate | Vehicle coordinate | IPM plane



Method II:

← r, g, b

(u_{ipm}, v_{ipm}) → $(x_v, y_v, 0)$ $\xrightarrow{{}^{c}R_v\ ,\ {}^{c}t_v}$ (x_c, y_c, z_c) $\xrightarrow[\text{projection}]{\text{Camera model}}$ $(u,\ v)$

IPM plane | Vehicle coordinate | Camera coordinate | Raw image plane

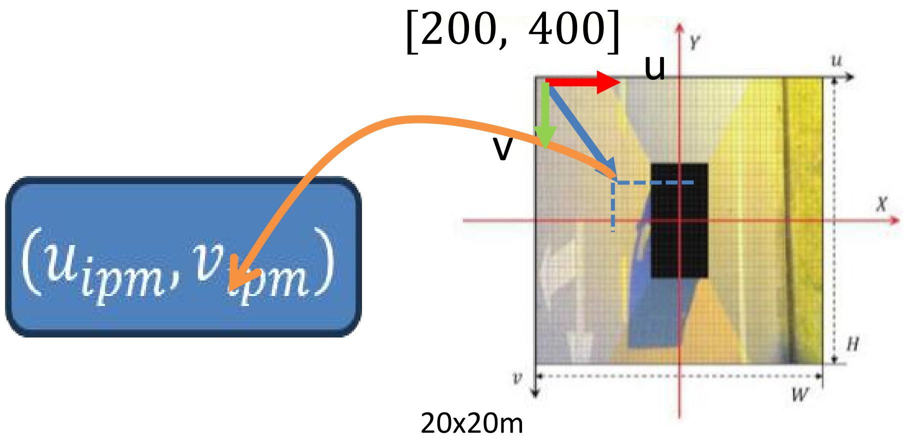
IPM Image

Method II:

$$(u_{ipm}, v_{ipm}) \longrightarrow (x_v, y_v, 0) \xrightarrow{{}^cR_v,\ {}^ct_v} (x_c, y_c, z_c) \xrightarrow[\text{projection}]{\text{Camera model}} (u, v)$$

r, g, b

IPM plane | Vehicle coordinate | Camera coordinate | Raw image plane



20x20m
1000x1000 pixel
0.02m/pixel

1. Choose one point on IPM plane, for example: $\begin{bmatrix} u_{ipm} \\ v_{ipm} \end{bmatrix} = \begin{bmatrix} 200 \\ 400 \end{bmatrix}$

2. Vehicle coordinate: $\begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} -(500 - u_{ipm}) * 0.02 \\ (500 - v_{ipm}) * 0.02 \\ 0 \end{bmatrix} = \begin{bmatrix} -6 \\ 2 \\ 0 \end{bmatrix}$

3. Camera coordinate: $\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} {}^cR_v & {}^ct_v \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \begin{bmatrix} {}^vR_c & {}^vt_c \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix}$

4. Project to image plane: $\begin{bmatrix} u \\ v \end{bmatrix} = \pi(p_c, K, D)$

0_extrinsic.yaml

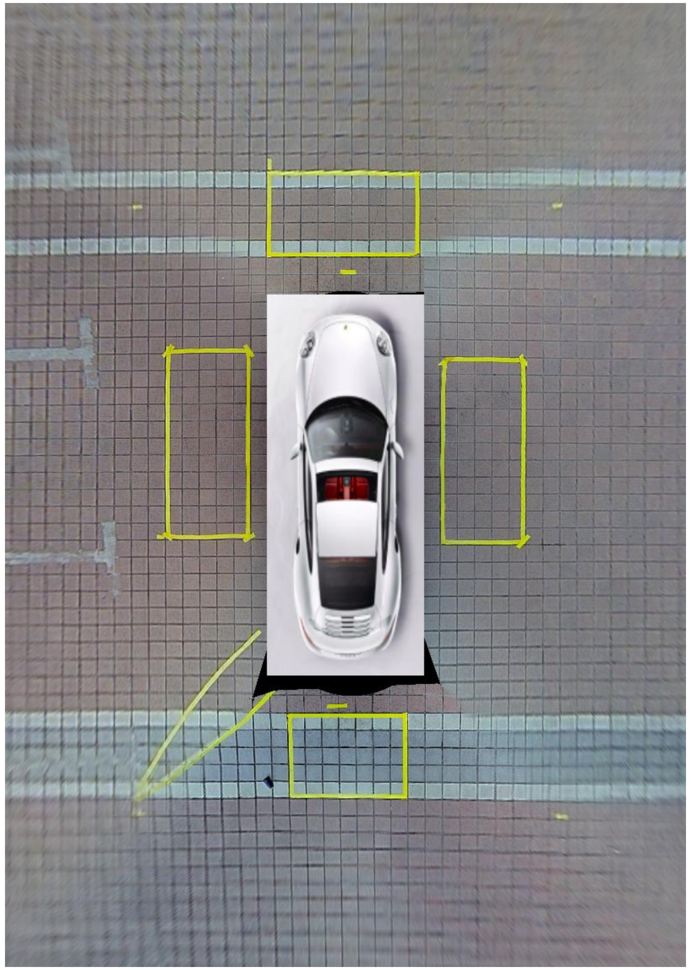

```yaml
%YAML:1.0
---
frame_id: cam0
transform:
  translation:
    x: 0.0
    y: 2.5
    z: 1.05
  rotation:
    x: -0.866
    y: 0.0
    z: 0.0
    w: 0.5
```


${}^vt_c$

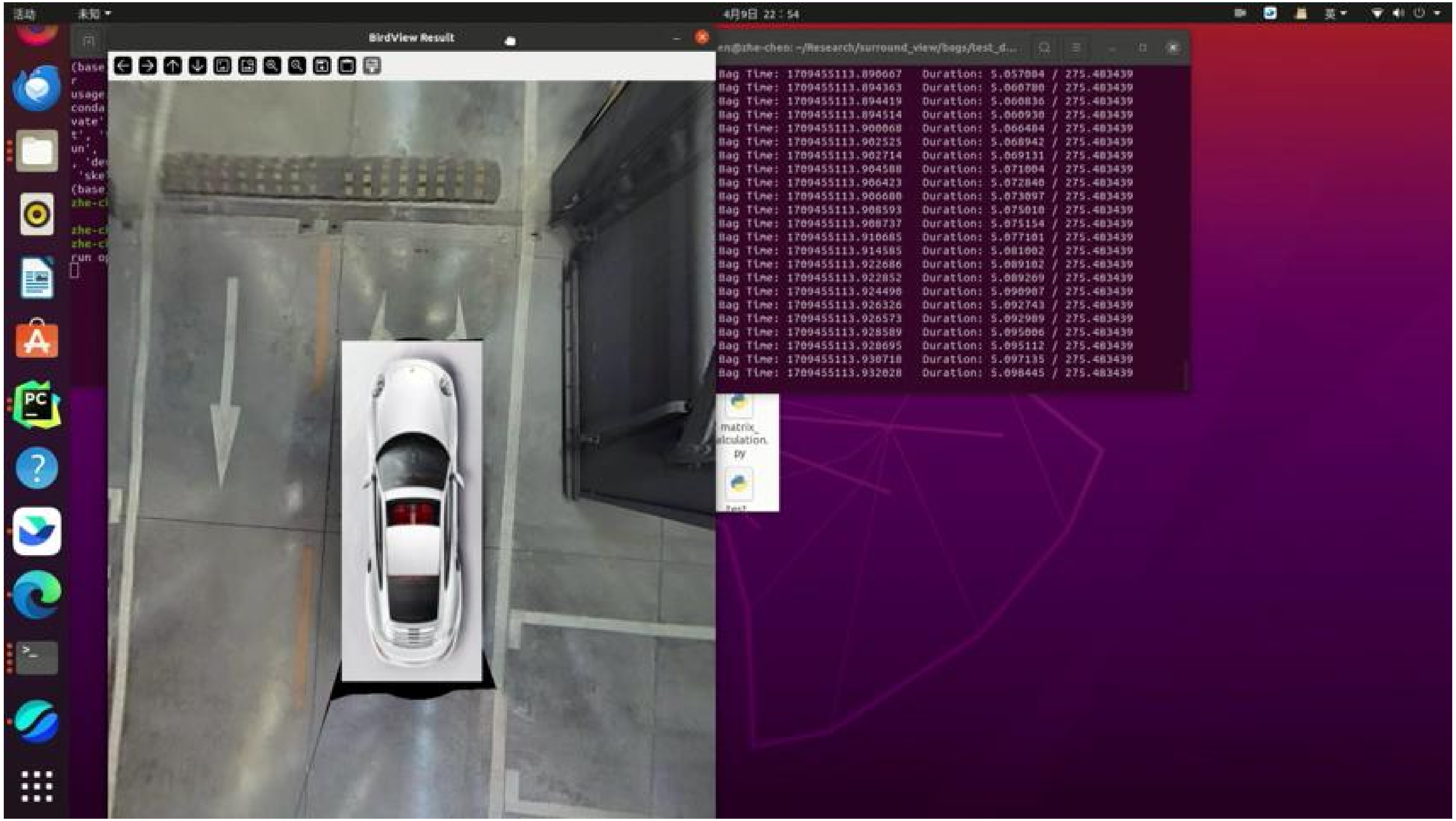
${}^vR_c$

IPM Image Stitching

Weight fusion



IPM Image Stitching



Content

1. Camera Background
2. Pinhole Camera Model
3. Distortion
4. Fisheye Camera Model (MEI)
5. IPM (Inverse Projective Mapping)
6. Assignment

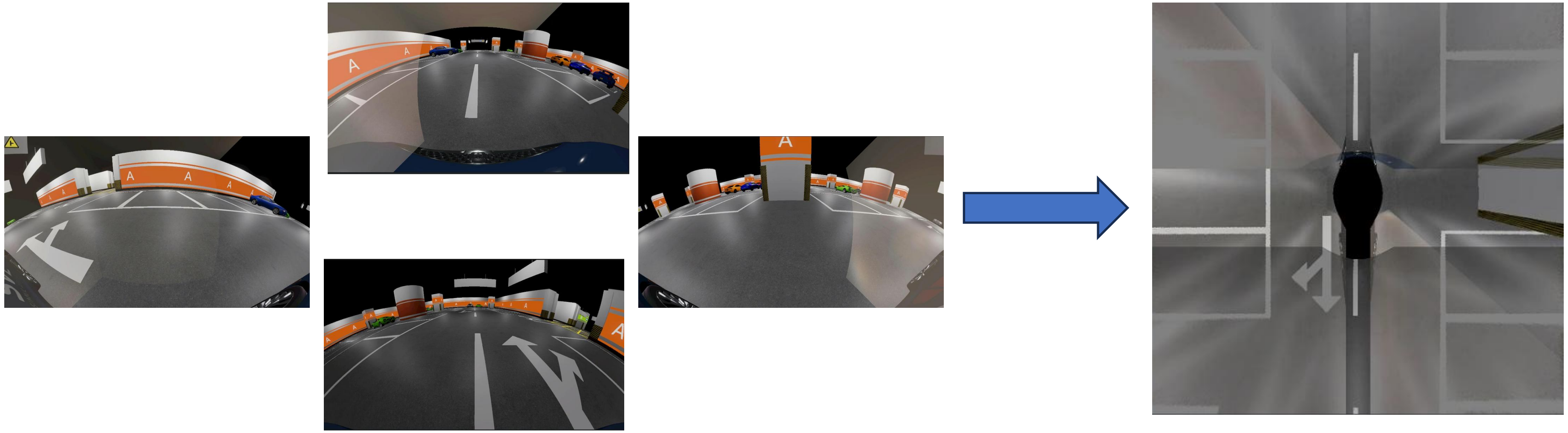
Assignment

Implement the IPM generation:

- Complete the function *GenerateIPMImage()*

```
cv::Mat IPM::GenerateIPMImage(const std::vector<cv::Mat>& images) const {
```

- Expected result:



深蓝学院
shenlanxueyuan.com

感谢聆听!

Thanks for Listening