Autonomous Valet Parking（AVP）Theory and Practice

自主代客泊车理论与实践

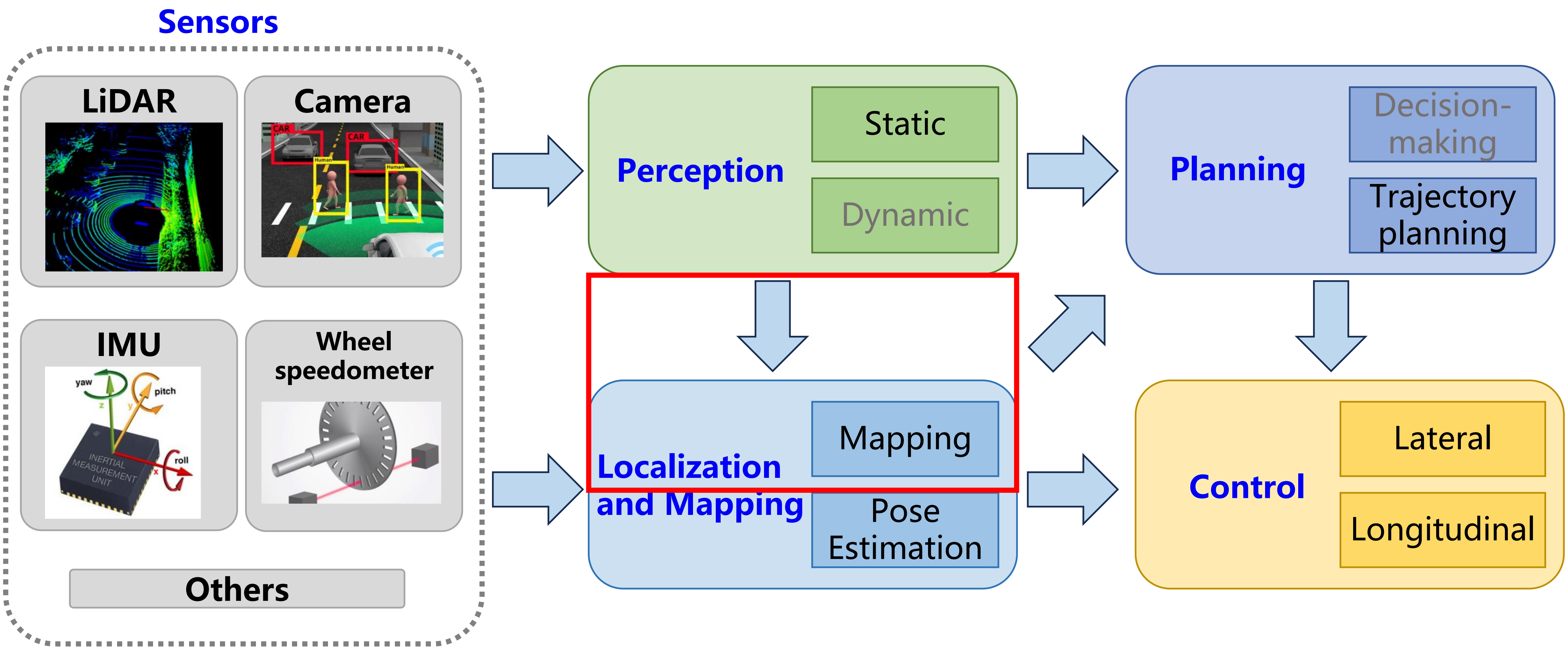
Lecture 5：Parking Map Construction

Lecturer Tong QIN

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

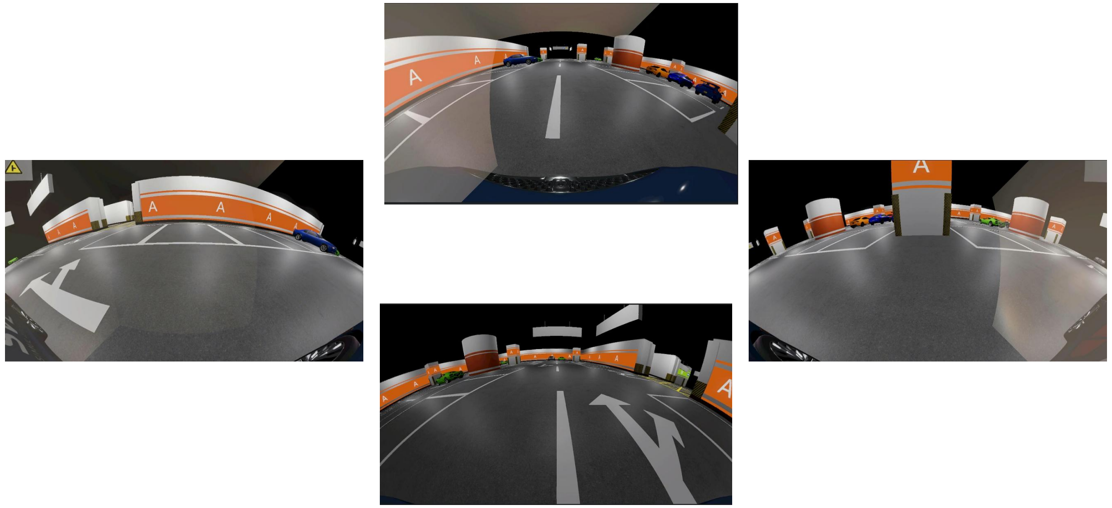
Content

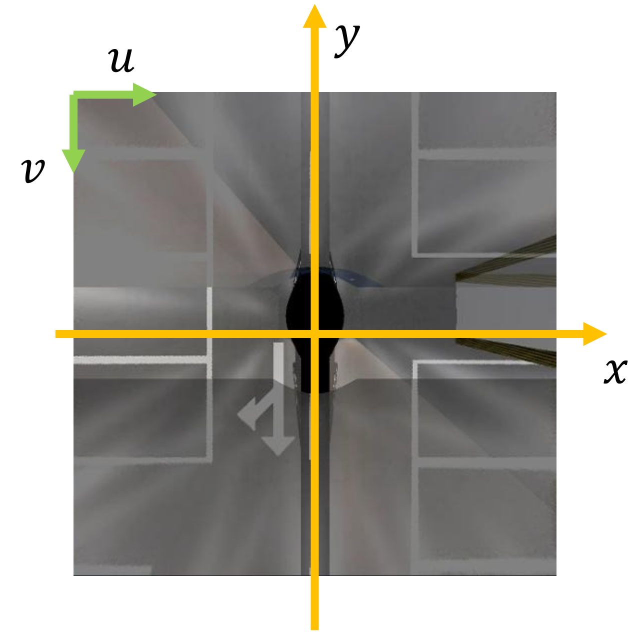
AVP Architecture



Content

Recall





Camera image coordinate $\longrightarrow$ IPM image coordinate

$$\begin{bmatrix} u \\ v \end{bmatrix} \xleftarrow[\pi(p_c, K, D)]{Intrinsic\ parameter\quad K, D} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}_{z_c > 0} \xleftarrow{Extrinsic\ parameter\quad {}^cR_v\ ,\ {}^ct_v} \begin{bmatrix} x_{v_c} \\ y_{v_c} \\ z_{v_c} \end{bmatrix}_{= 0} \xleftarrow{\substack{20 \times 20m \\ 1000 \times 1000\ pixel \\ 0.02m/pixel}} \begin{bmatrix} u_{ipm} \\ v_{ipm} \end{bmatrix}$$

Camera image coordinate

Camera coordinate

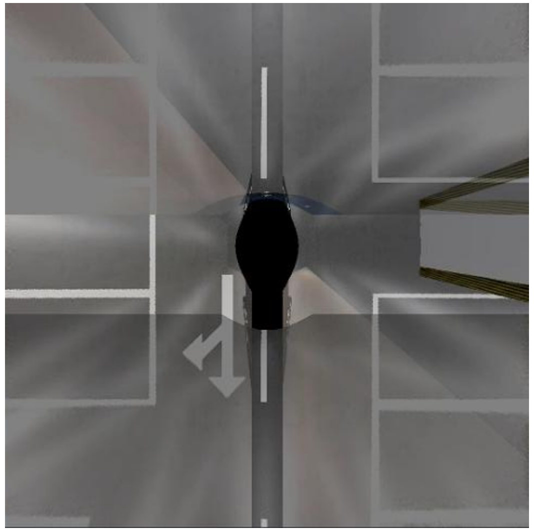
Vehicle center coordinate

IPM image coordinate

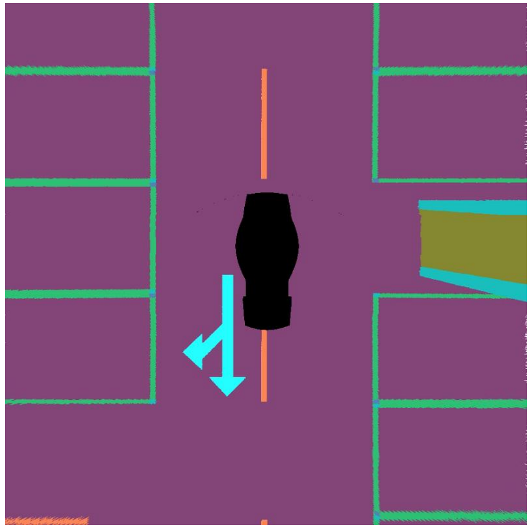
$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} {}^cR_v & {}^ct_v \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ \vdots \end{bmatrix}$$

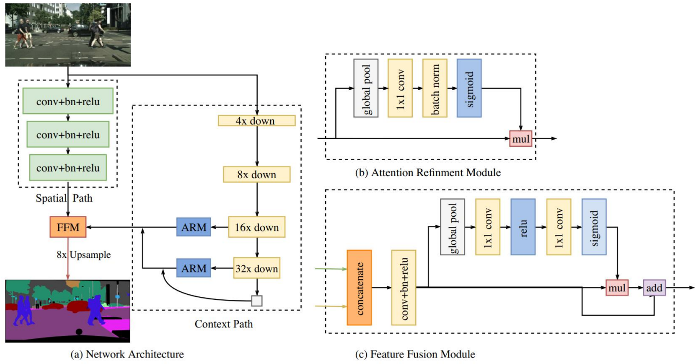
Recall

- Semantic feature detection



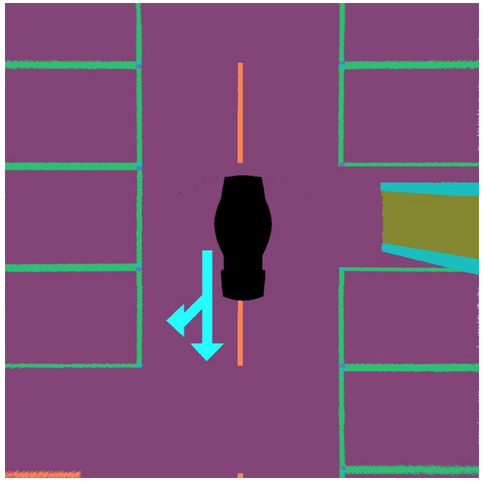
Semantic Segmentation



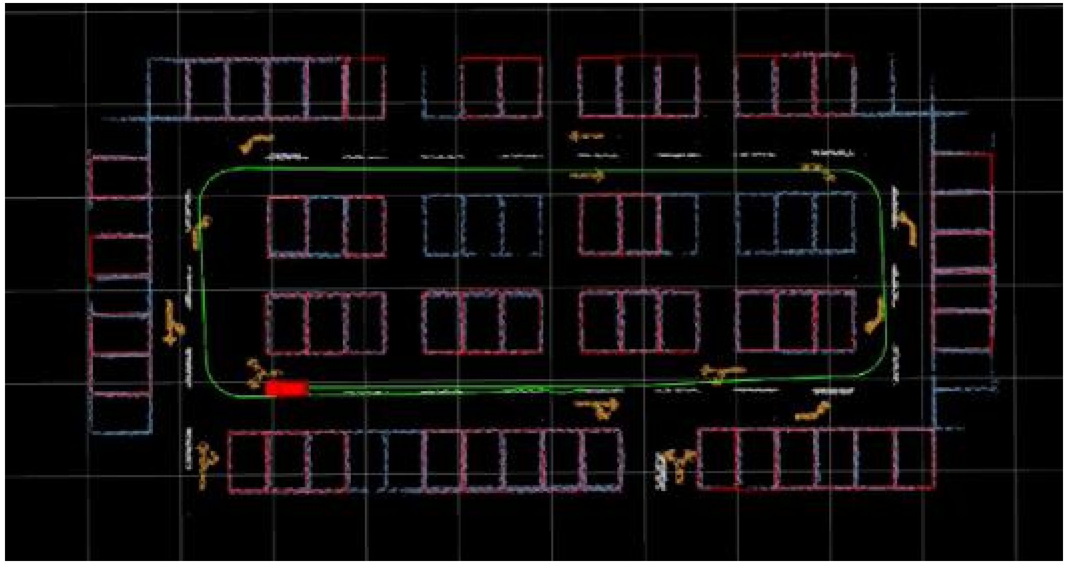


Coordinate Transformation

- Goal



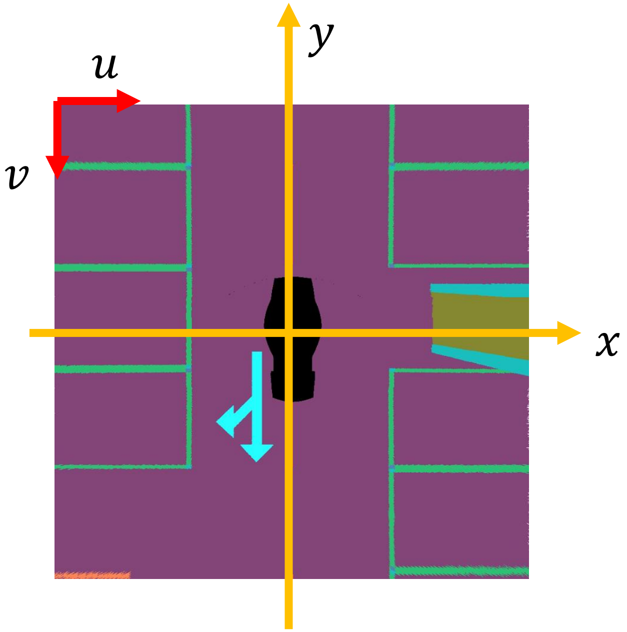
IPM image coordinate

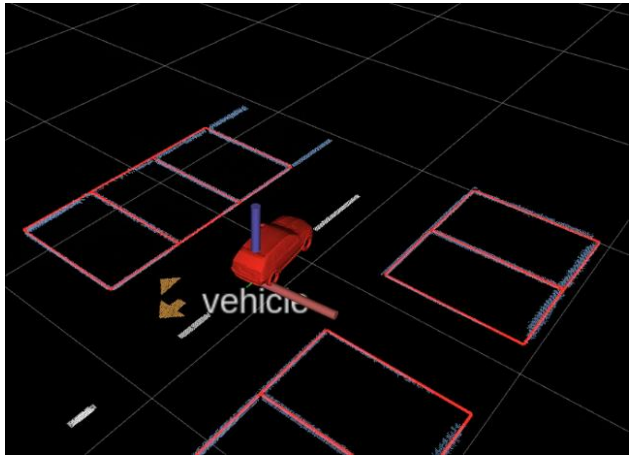


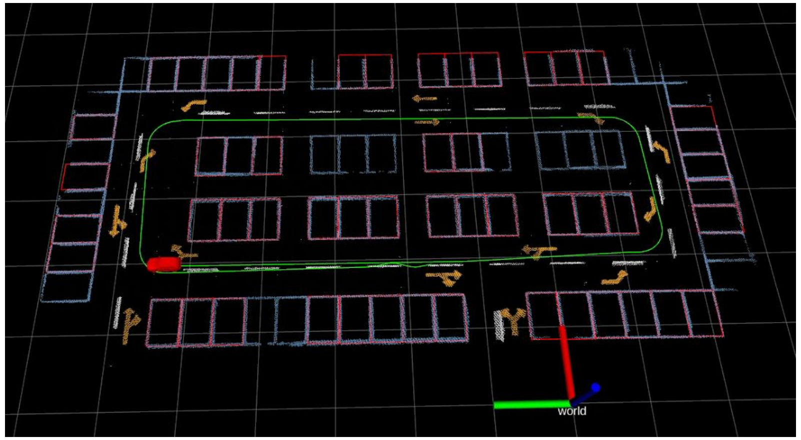
World (Global) coordinate

Semantic Map Construction

- Coordinates





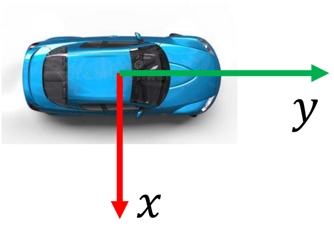


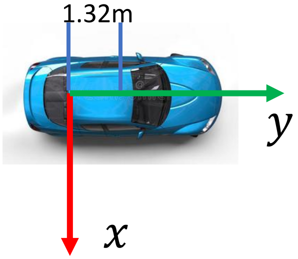
World (Global) coordinate

IPM image coordinate

Vehicle center coordinate

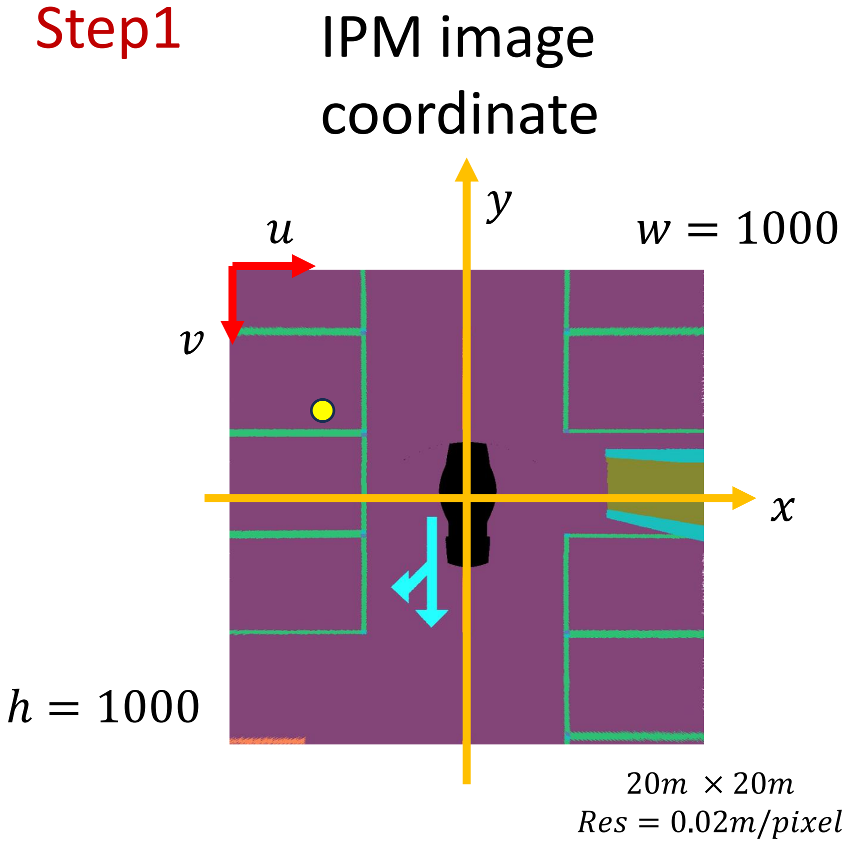
Vehicle rear coordinate

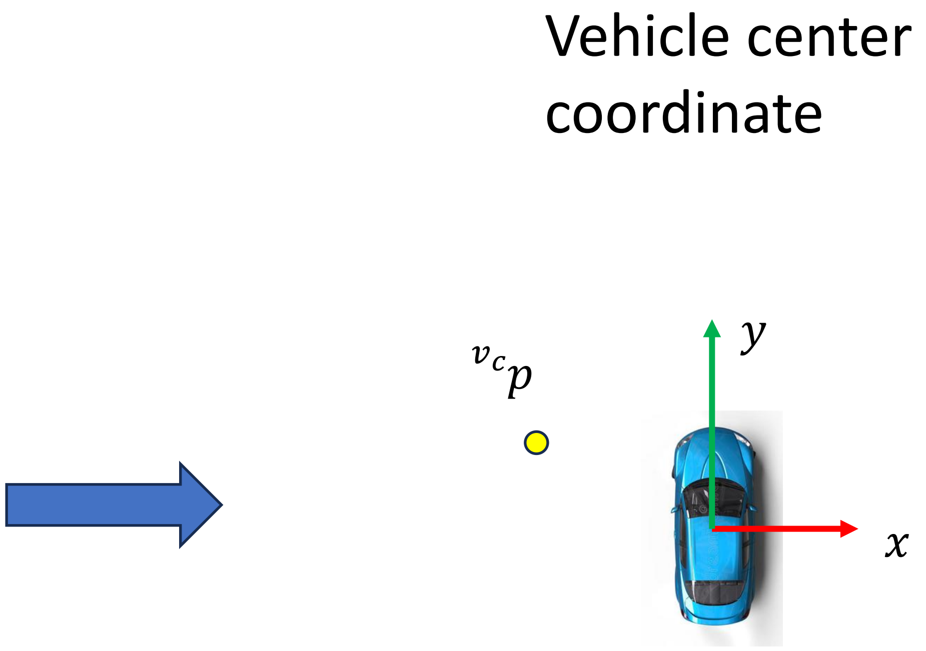




Coordinate Transformation

- Step1





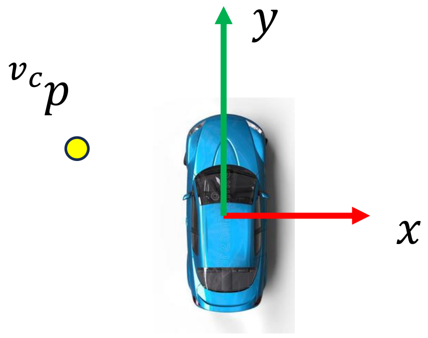
1. Choose one point on IPM plane, for example: $\begin{bmatrix} u_{ipm} \\ v_{ipm} \end{bmatrix} = \begin{bmatrix} 200 \\ 400 \end{bmatrix}$

2. Vehicle center coordinate: ${}^{v_c}p = \begin{bmatrix} x_{v_c} \\ y_{v_c} \\ z_{v_c} \end{bmatrix} = \begin{bmatrix} -\left(500 - u_{ipm}\right) * 0.02 \\ \left(500 - v_{ipm}\right) * 0.02 \\ 0 \end{bmatrix} = \begin{bmatrix} -6 \\ 2 \\ 0 \end{bmatrix}$

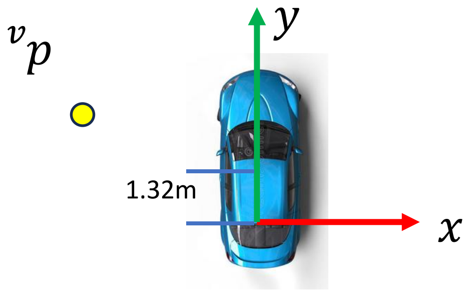
Coordinate Transformation

- Step2

Vehicle center coordinate



Vehicle (rear) coordinate



3. Vehicle (rear) coordinate: ${}^{v}p = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} x_{v_c} \\ y_{v_c} + 1.32 \\ 0 \end{bmatrix} = \begin{bmatrix} -6 \\ 3.32 \\ 0 \end{bmatrix}$

$$\begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \begin{array}{c} \begin{array}{cc} \quad {}^{v}R_{v_c} \quad & \quad {}^{v}t_{v_c} \end{array} \\ \begin{bmatrix} \boxed{\begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}} & \boxed{\begin{matrix} 0 \\ 1.32 \\ 0 \end{matrix}} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix} \end{array} \begin{bmatrix} x_{v_c} \\ y_{v_c} \\ z_{v_c} \\ 1 \end{bmatrix}$$

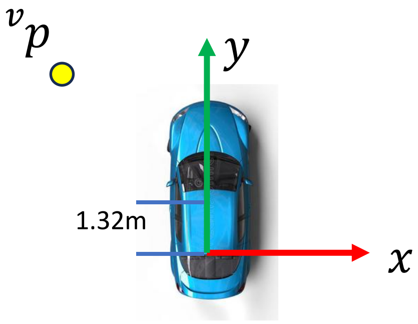
Coordinate Transformation

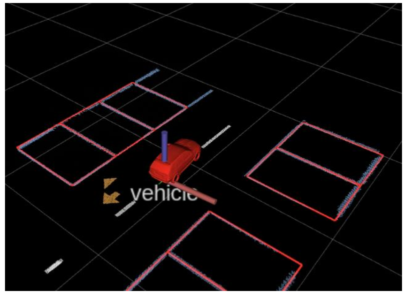
- Step3

- Assume we know vehicle's pose ${}^{w}R_{v}$, ${}^{w}t_{v}$
  - odometry

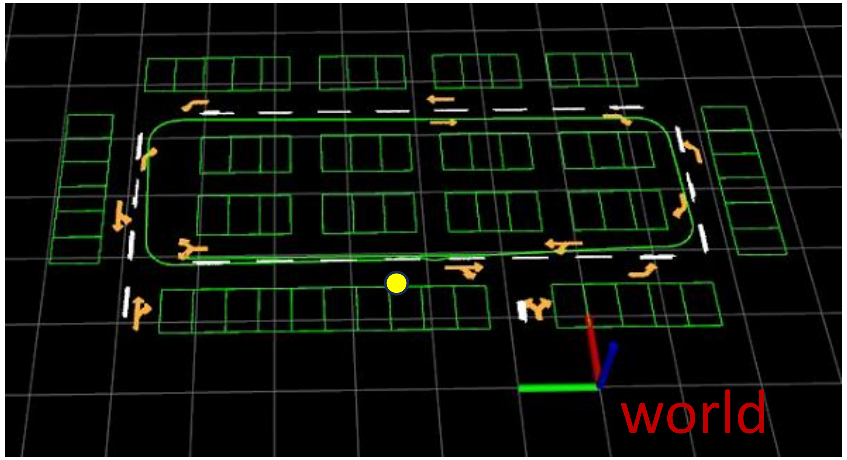
Vehicle rear coordinate

World (Global) coordinate





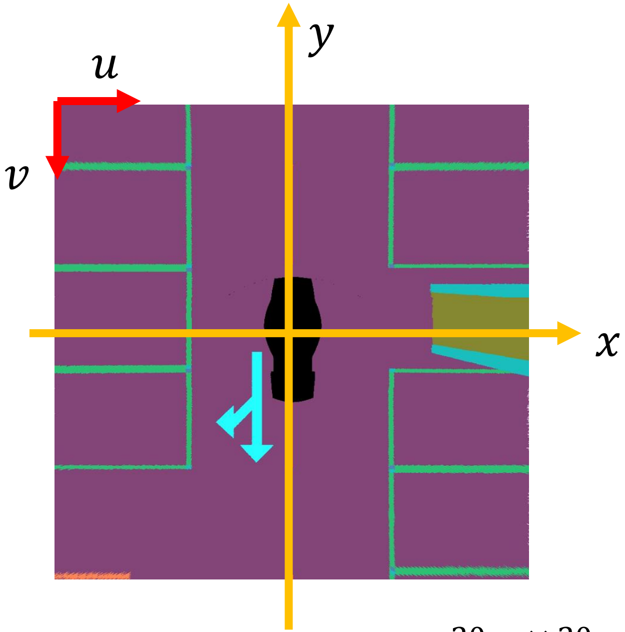
${}^{w}R_{v}$, ${}^{w}t_{v}$

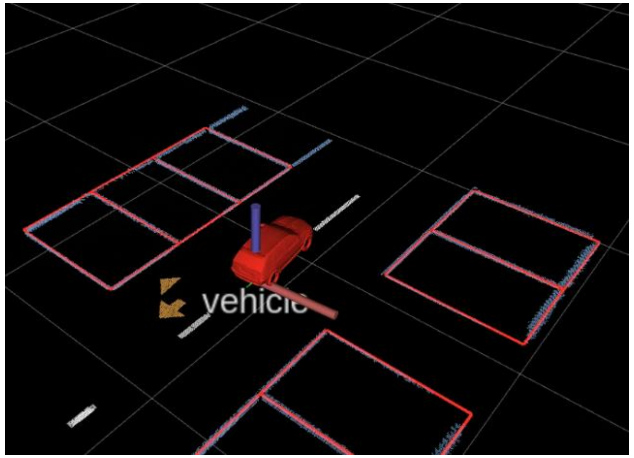


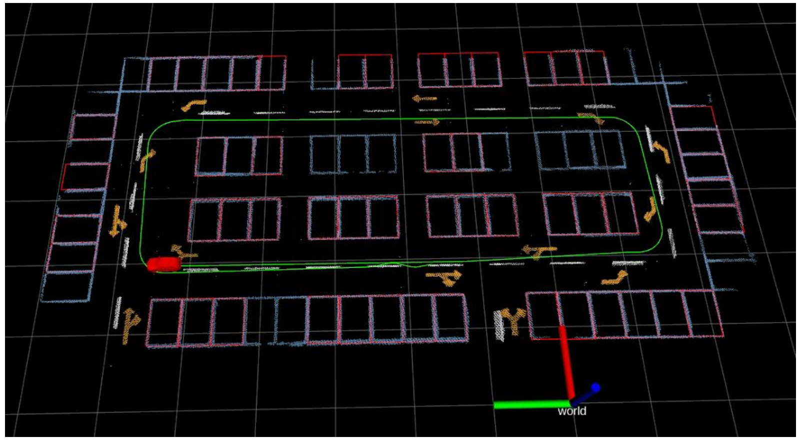
4. Vehicle (rear) coordinate to Global coordinate: $\begin{bmatrix} {}^{w}p \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{w}R_{v} & {}^{w}t_{v} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix}$ ${}^{v}p$

Coordinate Transformation

- Pipeline



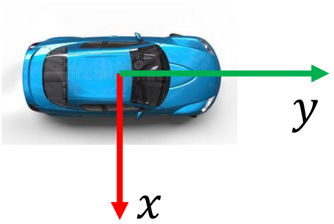




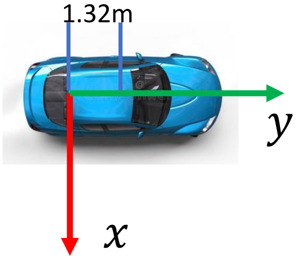
IPM image coordinate

$20m \times 20m$
$Res = 0.02m/pixel$

Vehicle center coordinate



Vehicle rear coordinate



${}^{w}R_v$, ${}^{w}t_v$

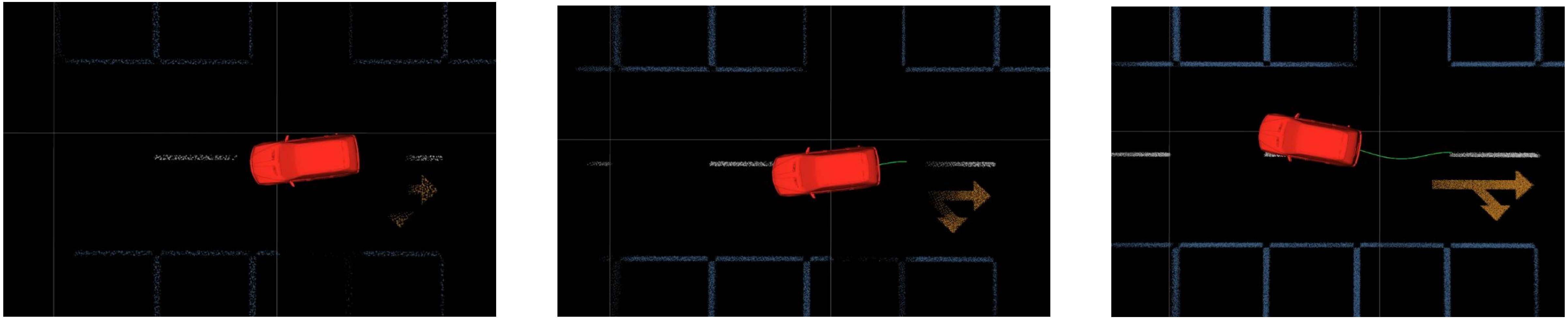
World (Global) coordinate

Content

1. Coordinate Transformation
2. Semantic Map Construction
3. Parking Spot Detection
4. Assignment

Semantic Map Construction

- Map increasing



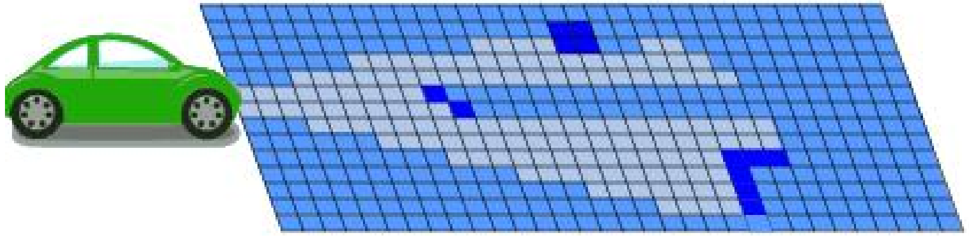
t $\qquad$ $t+1$ $\qquad$ $t+2$

Semantic points increase linearly

- Redundant points

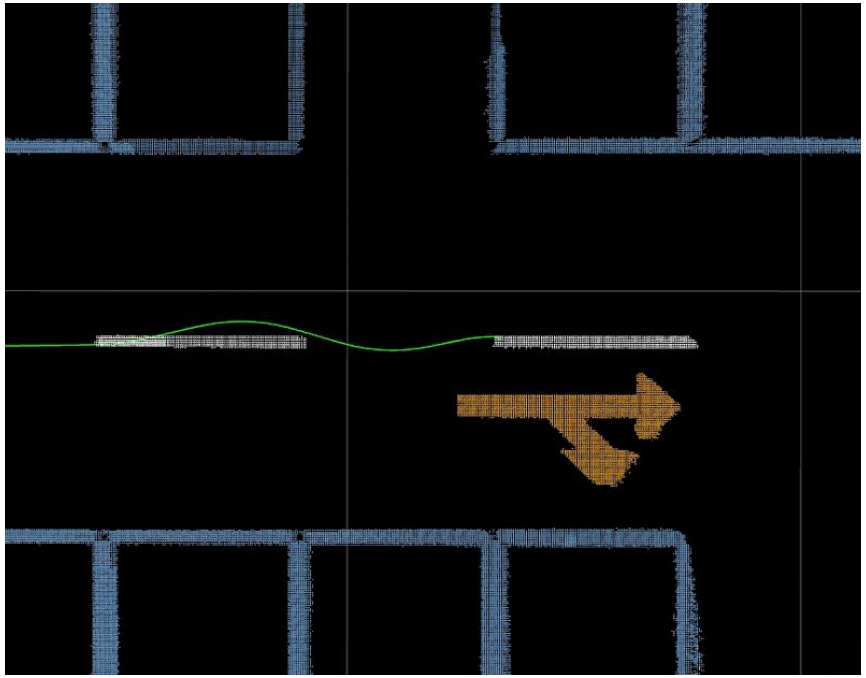
Semantic Map Construction

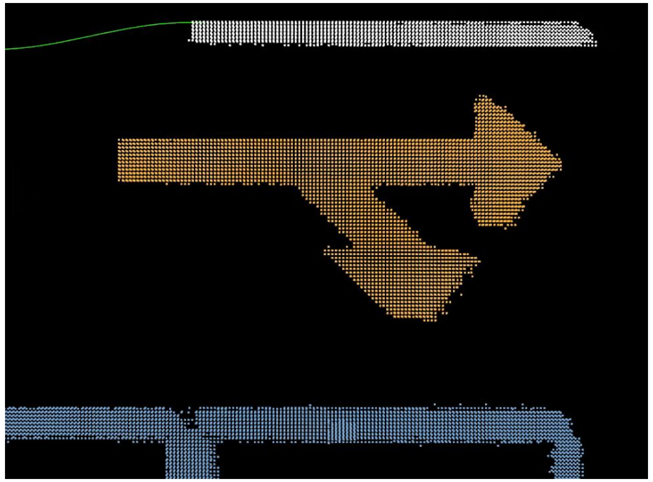
- Map sparsity
  - Grid map

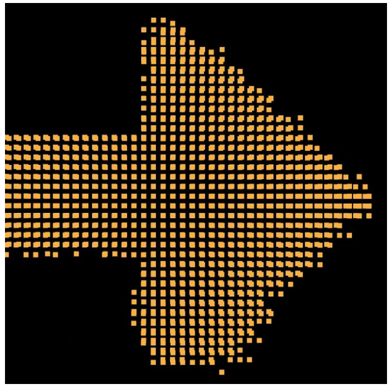


$0.04 \times 0.04m$

One label in one grid



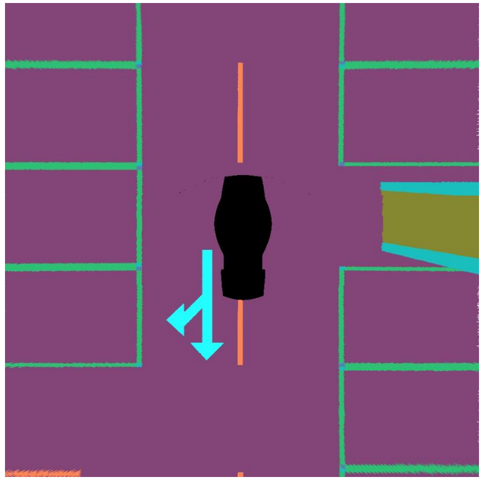


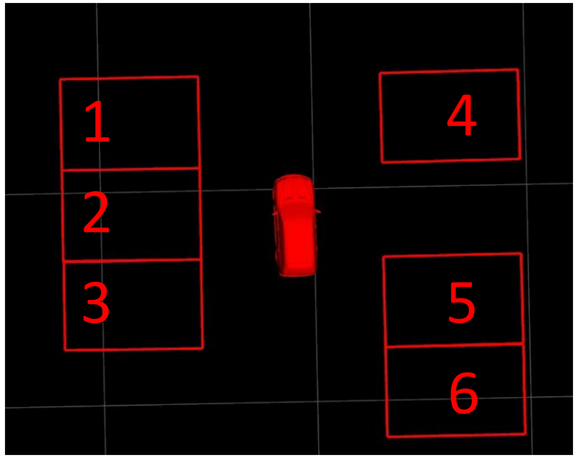


Content

Parking Spot Detection

- Goal
  - Extract the instance of parking spot





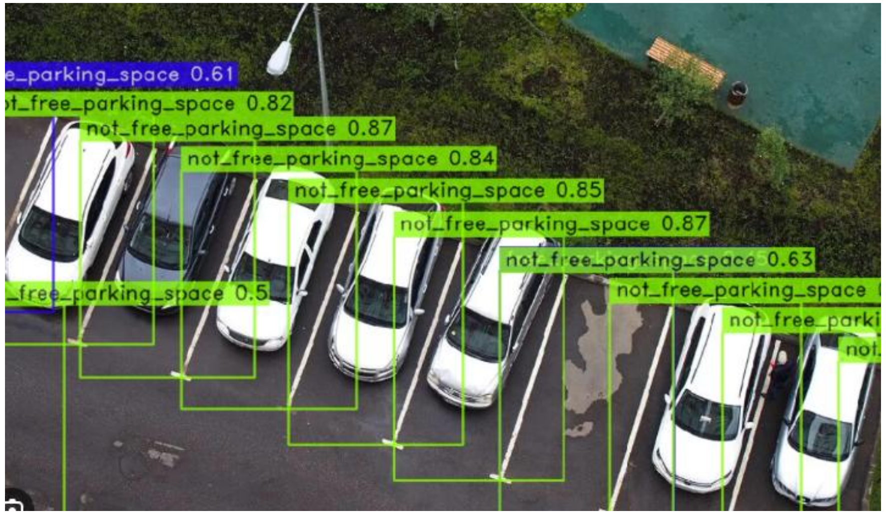
points
标量点

instance
实例

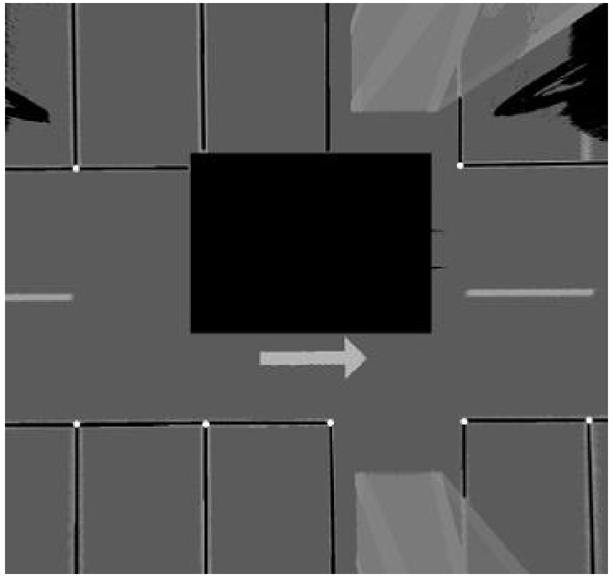
Parking Spot Detection

- Method

  - Neural network-based method



  - Rule-based method

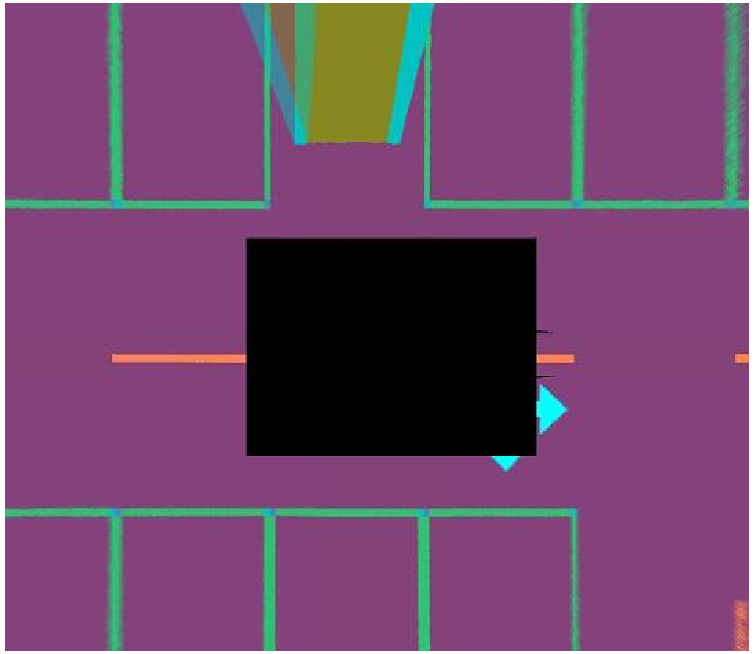


Geometric characteristic

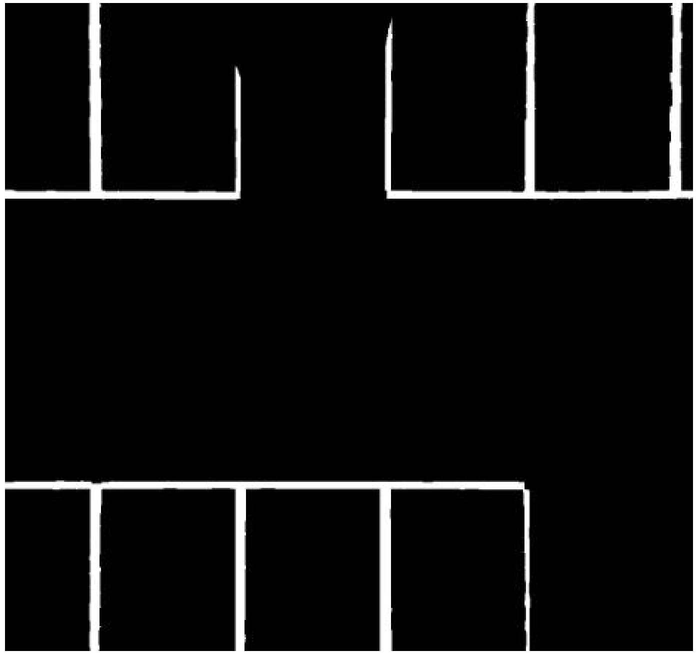
- Line->Corner->Rectangle

Parking Spot Detection

- Rule-based method
  - Line extraction

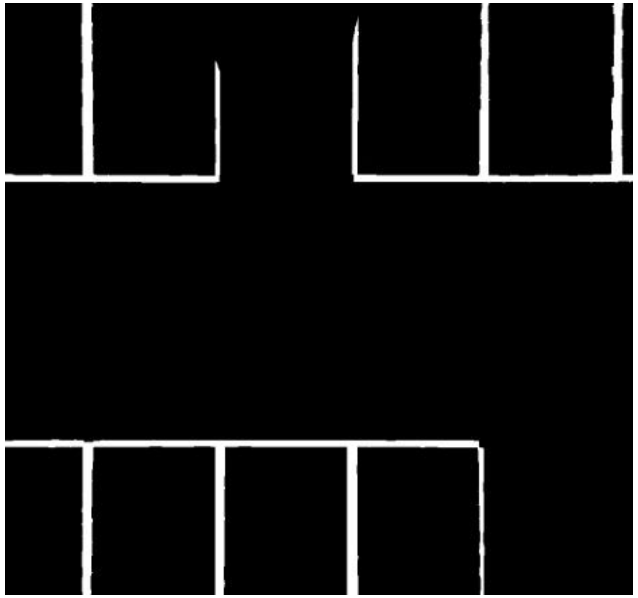


Label = slot line

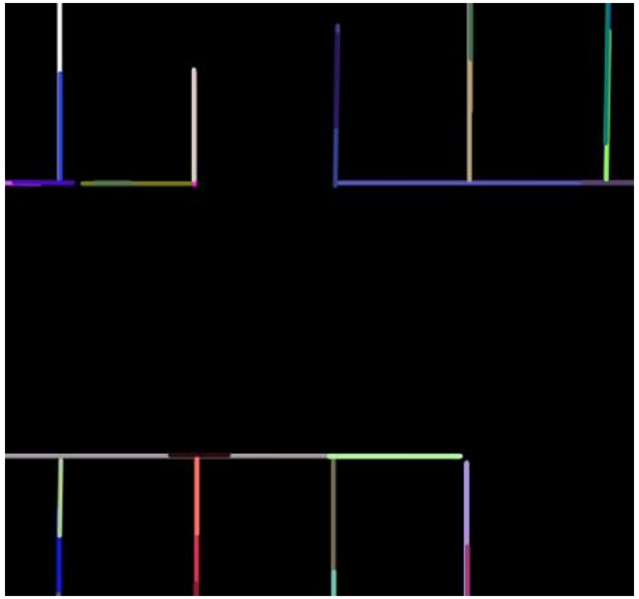


Parking Spot Detection

- Rule-based method
  - Line extraction



Hough Transform

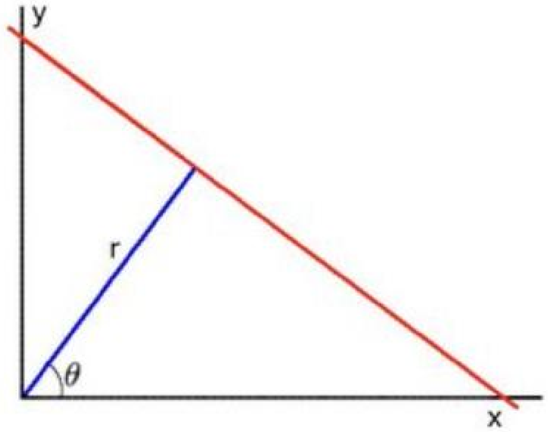


Parking Spot Detection

- Rule-based method
  - Line extraction
    - Hough Transform

1. As you know, a line in the image space can be expressed with two variables. For example:

   a. In the **Cartesian coordinate system:** Parameters: (m, b).
   b. In the **Polar coordinate system:** Parameters: (r, θ)



For Hough Transforms, we will express lines in the *Polar system*. Hence, a line equation can be written as:

$$y = \left(-\frac{\cos\theta}{\sin\theta}\right)x + \left(\frac{r}{\sin\theta}\right)$$

Arranging the terms: $r = x\cos\theta + y\sin\theta$

1. In general for each point (x_0, y_0), we can define the family of lines that goes through that point as:

$$r_\theta = x_0 \cdot \cos\theta + y_0 \cdot \sin\theta$$

Meaning that each pair (r_θ, θ) represents each line that passes by (x_0, y_0).

Parking Spot Detection

- Rule-based method
  - Line extraction
    - Hough Transform

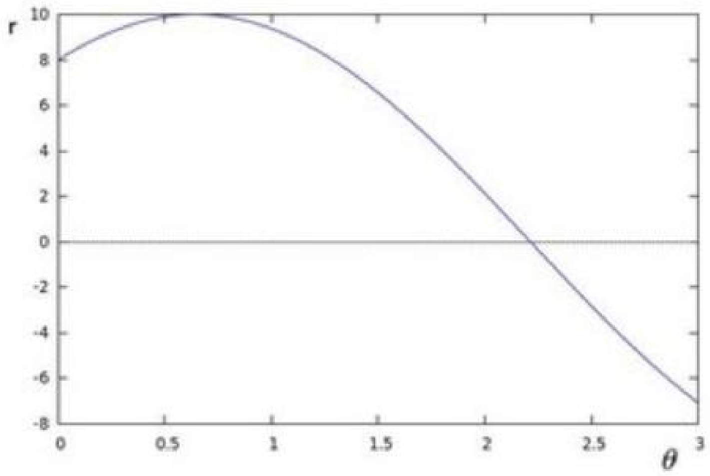
Arranging the terms: $r = x\cos\theta + y\sin\theta$

1. In general for each point (x_0, y_0), we can define the family of lines that goes through that point as:

$$r_\theta = x_0 \cdot \cos\theta + y_0 \cdot \sin\theta$$

Meaning that each pair (r_θ, θ) represents each line that passes by (x_0, y_0).

2. If for a given (x_0, y_0) we plot the family of lines that goes through it, we get a sinusoid. For instance, for $x_0 = 8$ and $y_0 = 6$ we get the following plot (in a plane θ - r):

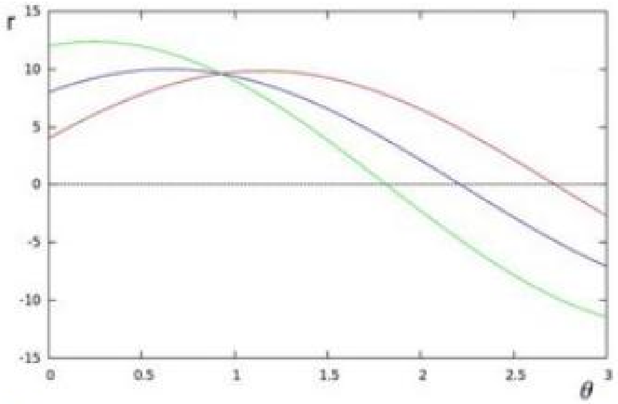


We consider only points such that $r > 0$ and $0 < \theta < 2\pi$.

Parking Spot Detection

- Rule-based method
  - Line extraction
    - Hough Transform

3. We can do the same operation above for all the points in an image. If the curves of two different points intersect in the plane θ - r, that means that both points belong to a same line. For instance, following with the example above and drawing the plot for two more points: $x_1 = 4$, $y_1 = 9$ and $x_2 = 12$, $y_2 = 3$, we get:

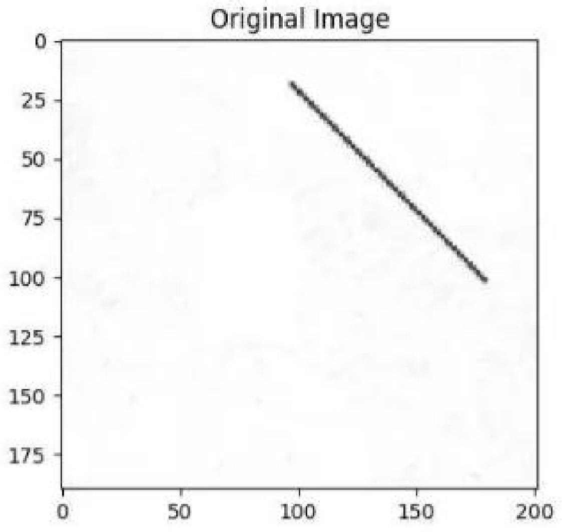


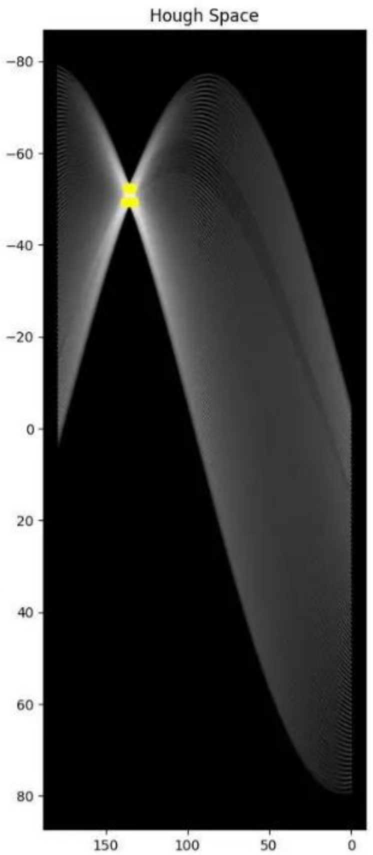
The three plots intersect in one single point $(0.925, 9.6)$, these coordinates are the parameters (θ, r) or the line in which (x_0, y_0), (x_1, y_1) and (x_2, y_2) lay.

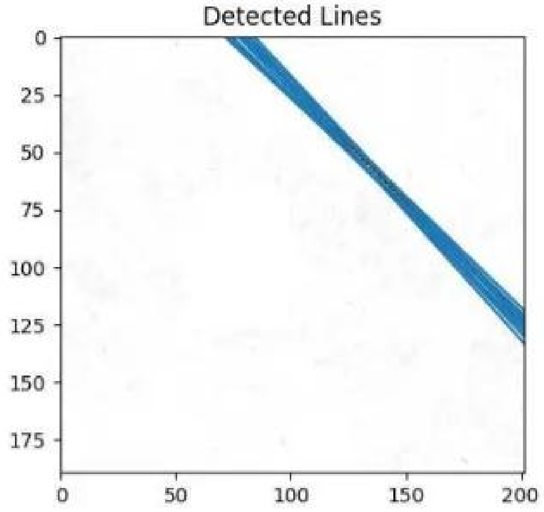
Parking Spot Detection

cv::HoughLinesP

- Rule-based method
  - Line extraction
    - Hough Transform



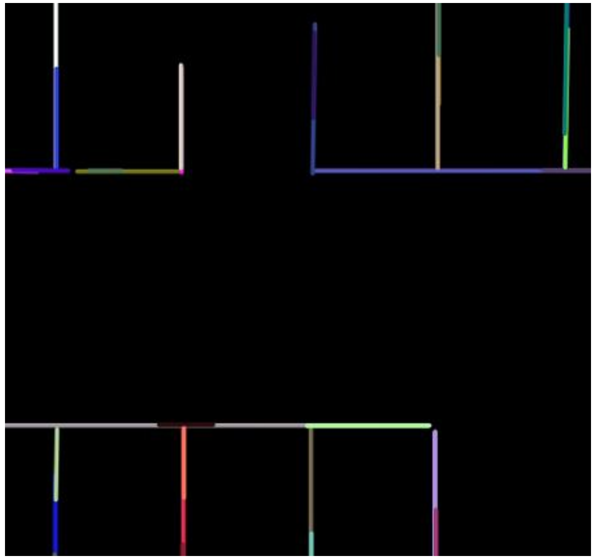


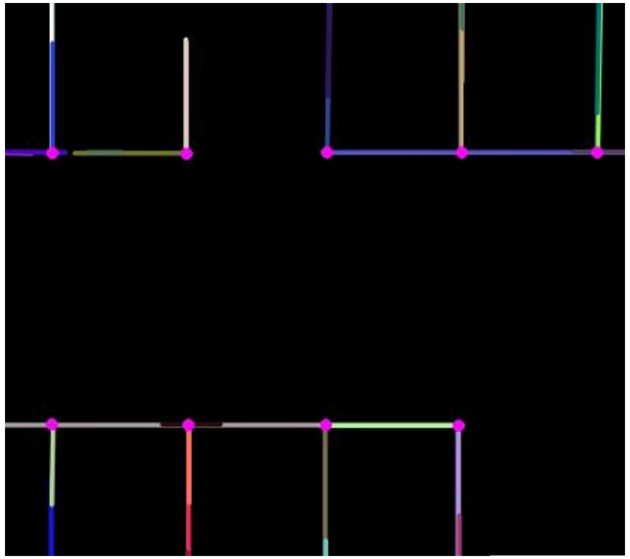


Parking Spot Detection

- Rule-based method
  - Line extraction
  - Corner extraction

- Iterate every two lanes
  - Distance threshold
    - Intersection angle threshold （~90°）
    - Check intersection point label
- Merge too-closed corner points





Intersection angle

$$\cos\theta = \frac{\overrightarrow{A_1A_2} \cdot \overrightarrow{B_1B_2}}{|\overrightarrow{A_1A_2}||\overrightarrow{B_1B_2}|}$$

$$a_1x + b_1y = c_1$$

$$a_2x + b_2y = c_2$$

Intersection point

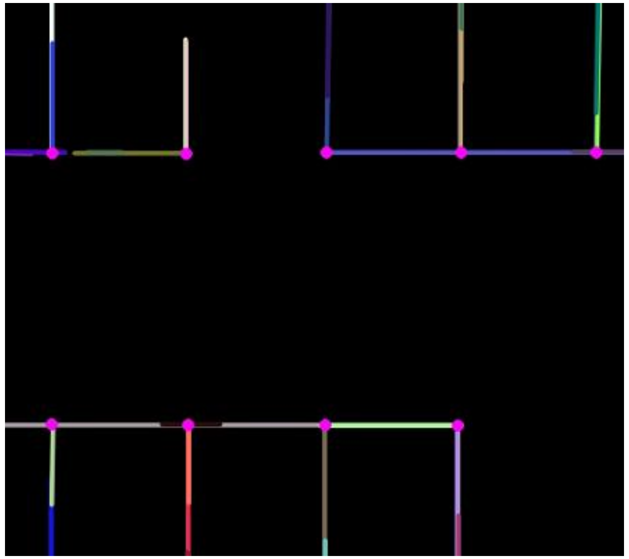
$$x = \frac{c_1b_2 - c_2b_1}{a_1b_2 - a_2b_1}$$

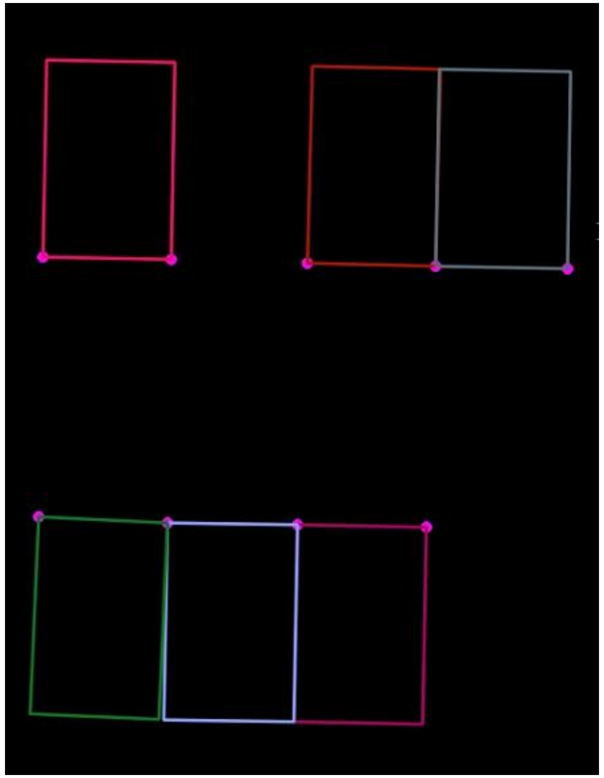
$$y = \frac{a_1c_2 - a_2c_1}{a_1b_2 - a_2b_1}$$

Parking Spot Detection

Slot: 6.4x4.2m   210x320pixel

- Rule-based method
  - Line extraction
  - Corner extraction
  - Slot fitting



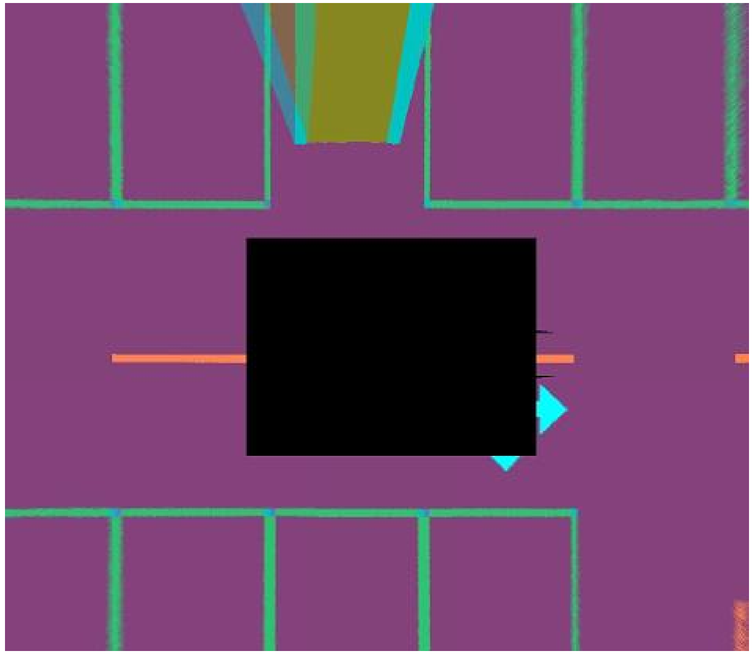


- Iterate every two corners
  - 两点距离是否和车位短边接近？(4.2m)
  - &两点组成的短边是否有一定比例的点的*label*为库位线
    - 计算短边对应的两个垂线 （注意垂线正、反方向）
    - 垂线上是否有大量的点的*label*为库位线
      - 若是，根据车位长度（6.4m）推测另外两个库位角点

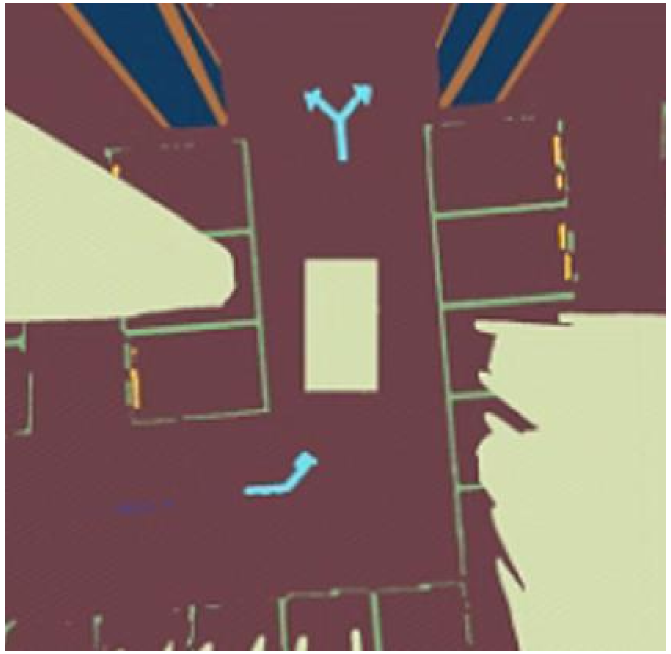
Parking Spot Detection

- Rule-based method
  - Line extraction
  - Corner extraction
  - Slot fitting

Ideal case

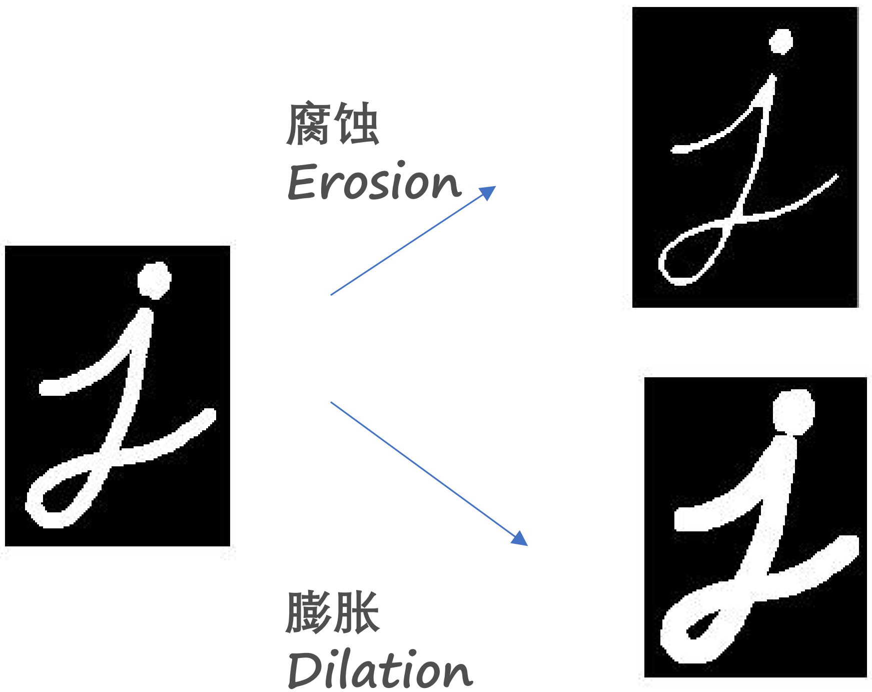


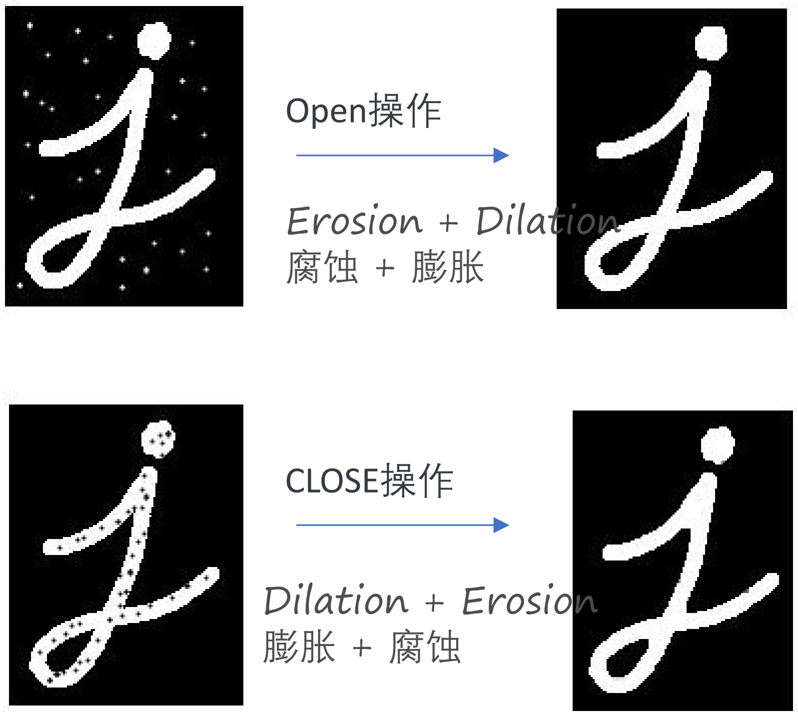
Real case : Noise and blur



Parking Spot Detection

- Rule-based method
  - Enhancement and denoising
  - Line extraction
  - Corner extraction
  - Slot fitting





Content

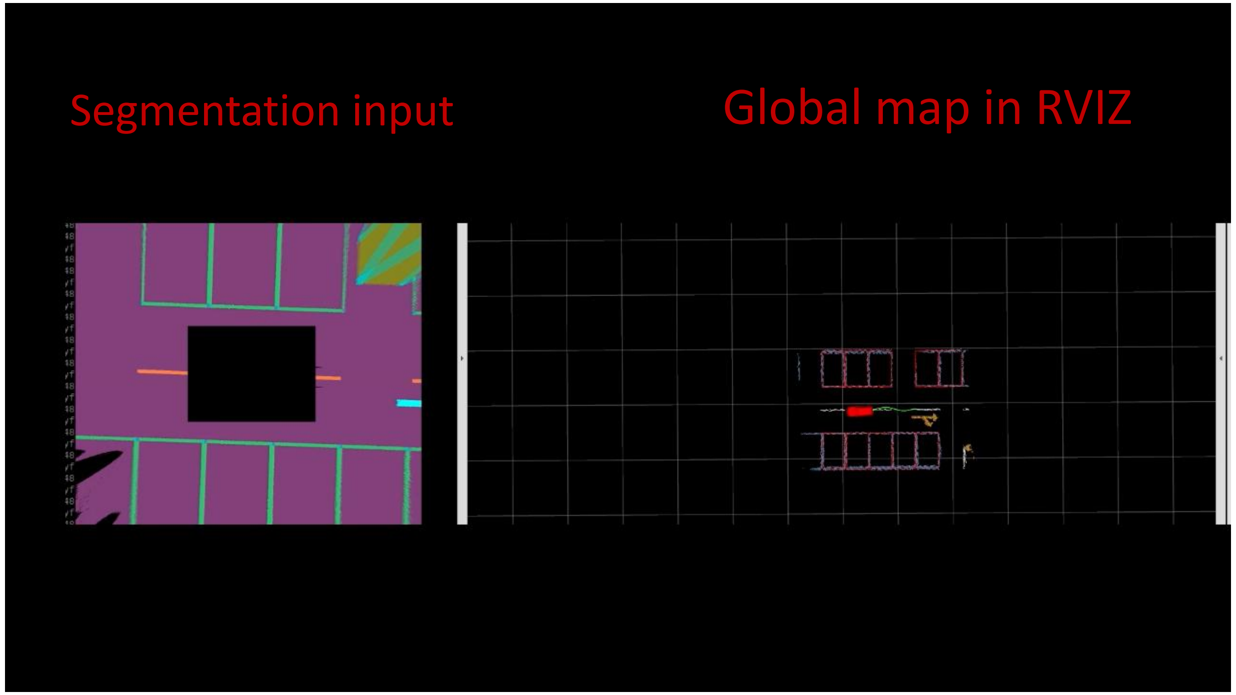
Assignment

Building your AVP map

Input：

- Segmentation result for every IPM image
- Vehicle pose (vehicle rear in global frame) ${}^{w}R_{v}$, ${}^{w}t_{v}$

Expected Output:



- Complete C++ functions:
  - *ipmPlane2Global()*
    - Transformation from IPM image plane to global coordinate
  - *detectSlot()*
    - Extract slot from lanes

深蓝学院
shenlanxueyuan.com

感谢聆听!

Thanks for Listening!