Autonomous Valet Parking（AVP）Theory and Practice

自主代客泊车理论与实践

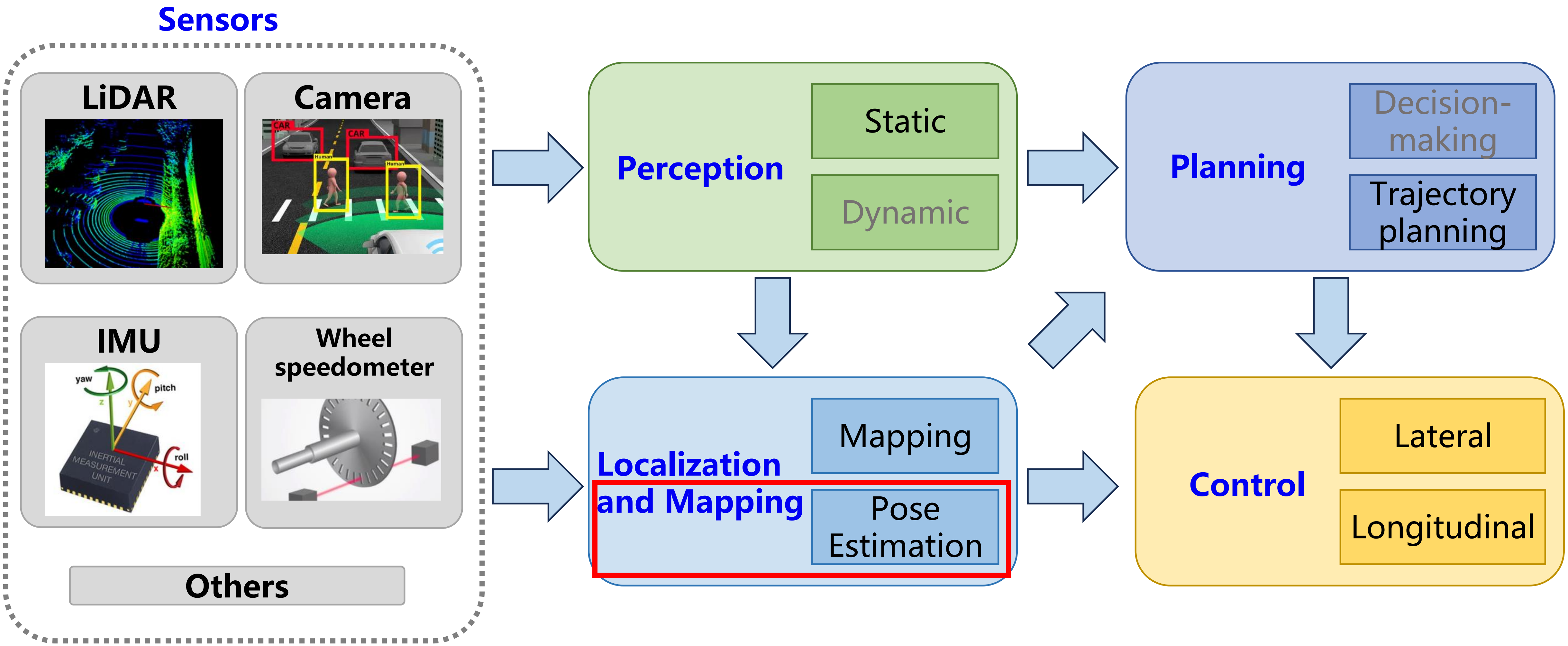
Lecture 6：Semantic Localization

Lecturer Tong QIN

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

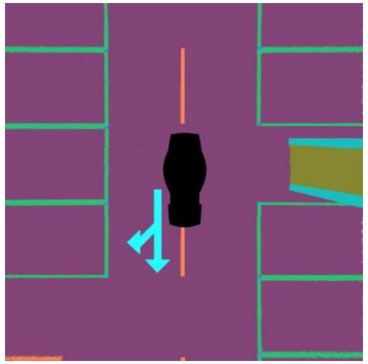
Content

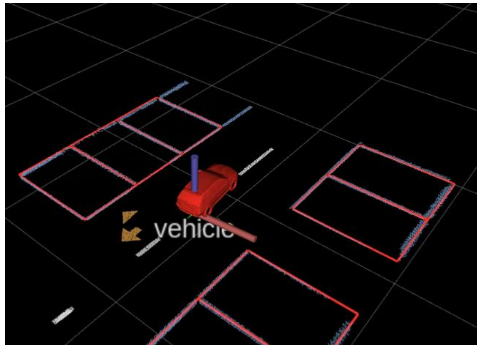
AVP Architecture



Content

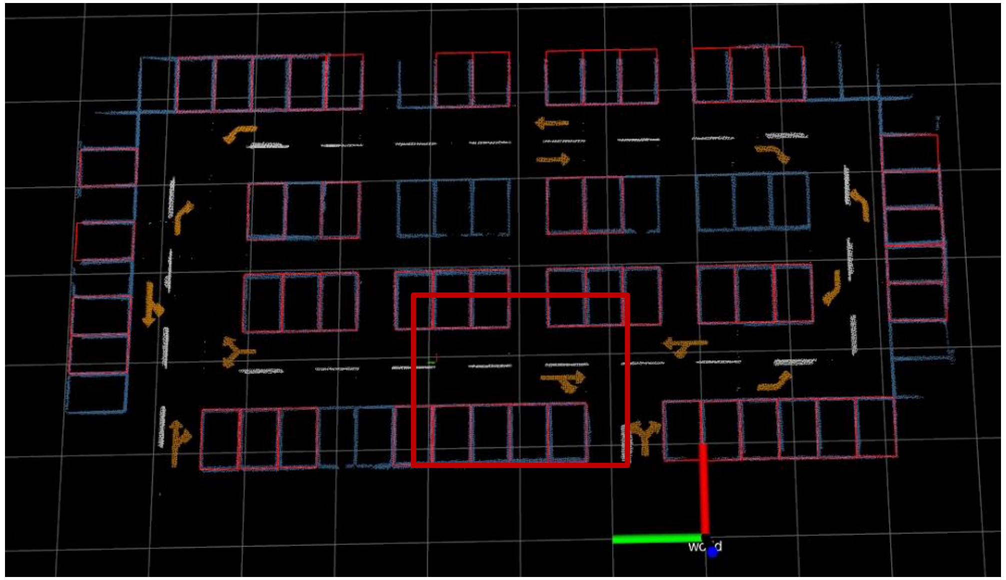
Point Cloud Registration
(3D-3D Pose Estimation)



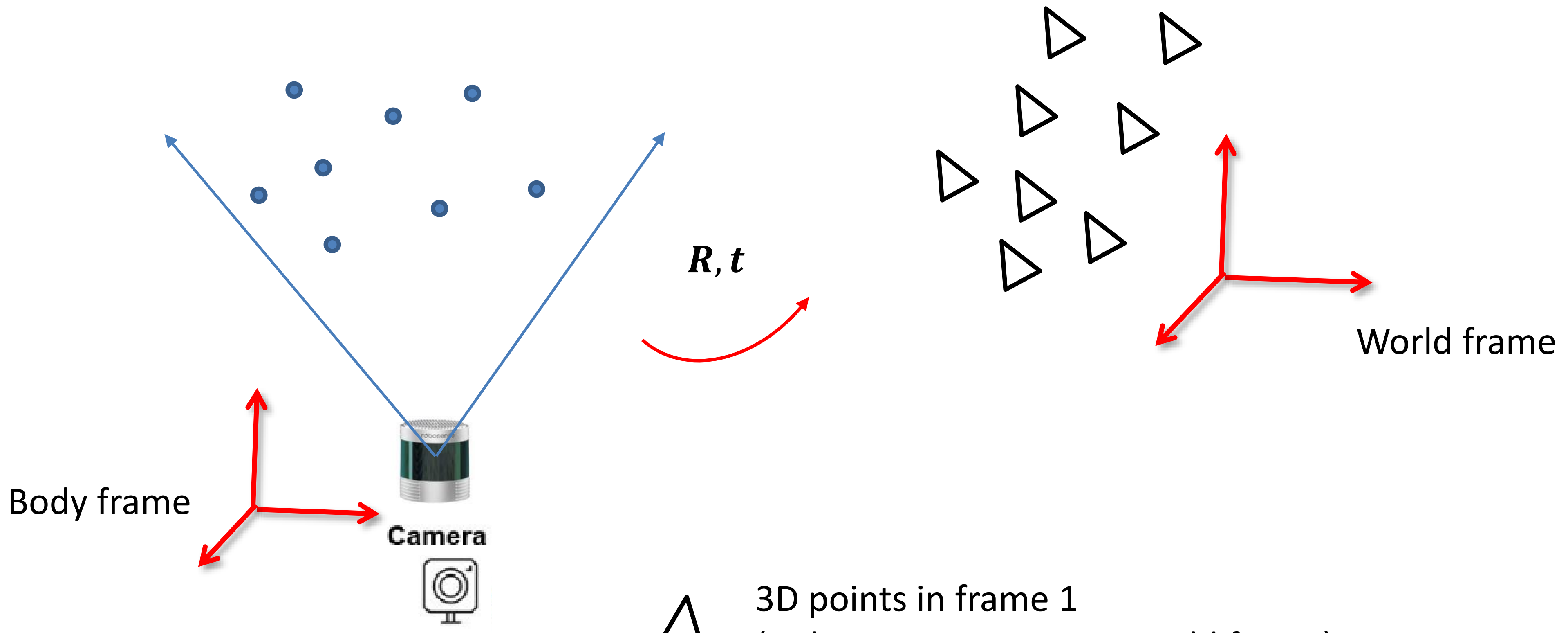


Where the vehicle is？

AVP Map



3D-3D Pose Estimation

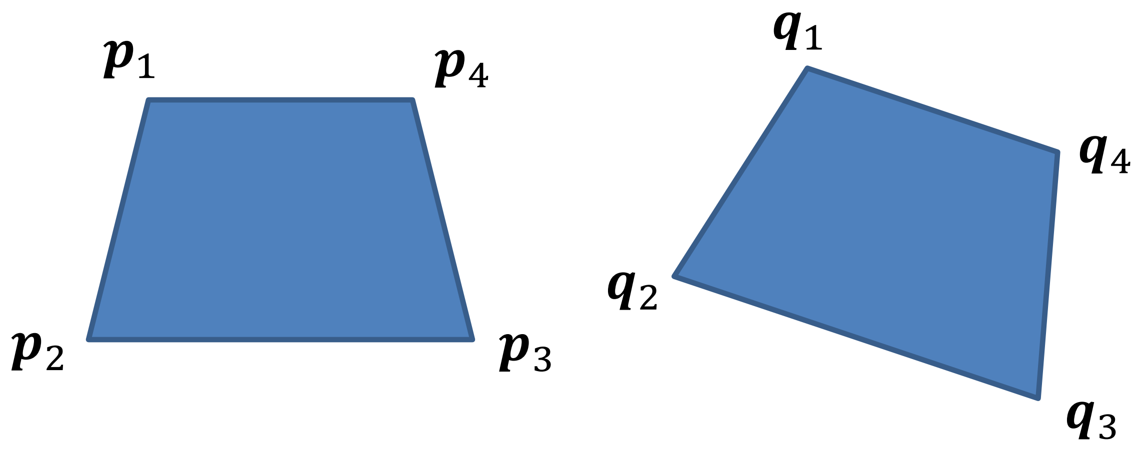


3D points in frame 1
(or known 3D points in world frame)

3D points in frame 2
(or 3D point observations in body frame)

3D-3D

3D-3D Pose Estimation

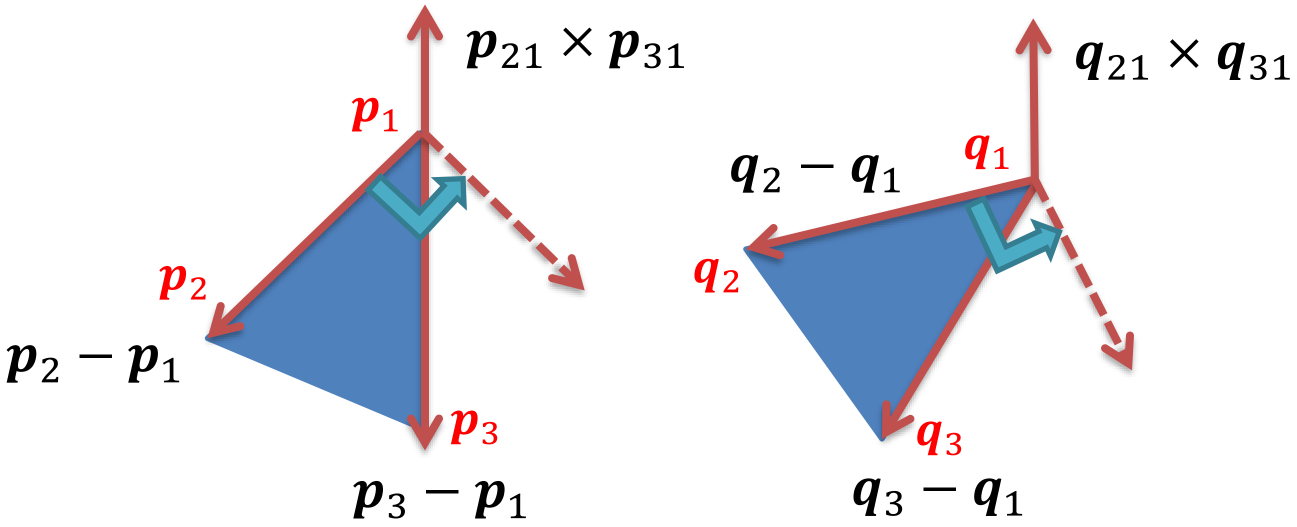


- How do we solve for $\boldsymbol{R}$, $\boldsymbol{t}$ from point correspondences?

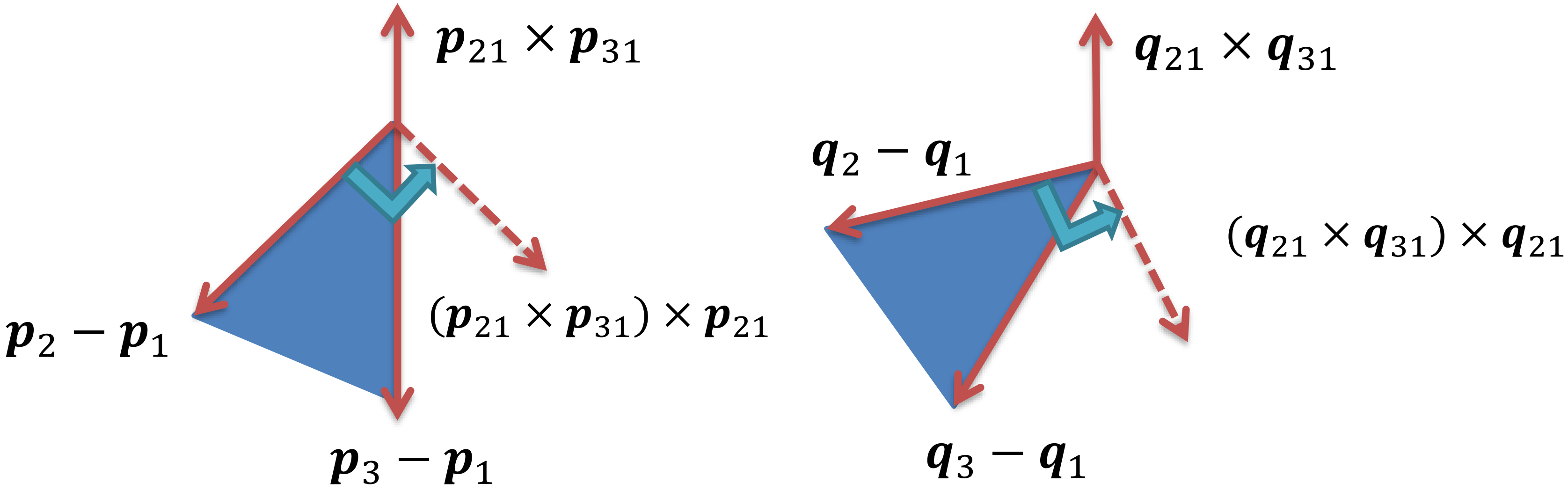
$$\boldsymbol{p}_i = \boldsymbol{R}\boldsymbol{q}_i + \boldsymbol{t}$$

3D-3D Pose Estimation

- What is the minimal number of points needed?



3D-3D Pose Estimation



- Three non-collinear points suffice: each triangle $\boldsymbol{p}_{i=1\ldots3}$ and $\boldsymbol{q}_{i=1\ldots3}$ make an orthogonal basis

$$(|\boldsymbol{p}_{21}| \quad |(\boldsymbol{p}_{21} \times \boldsymbol{p}_{31}) \times \boldsymbol{p}_{21}| \quad |\boldsymbol{p}_{21} \times \boldsymbol{p}_{31}|)$$

and

$$(|\boldsymbol{q}_{21}| \quad |(\boldsymbol{q}_{21} \times \boldsymbol{q}_{31}) \times \boldsymbol{q}_{21}| \quad |\boldsymbol{q}_{21} \times \boldsymbol{q}_{31}|)$$

- Rotation between two orthogonal bases is unique.

3D-3D Pose Estimation

- We solve a minimization problem for N >= 3 point correspondences:

$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i}^{N} \left\| \left(\boldsymbol{R}\boldsymbol{p}_s^i + \boldsymbol{t} \right) - \boldsymbol{p}_t^i \right\|^2$$

- After differentiating with respect to $\boldsymbol{t}$,

$$\frac{\partial F}{\partial t} = \sum_{i=1}^{N} 2(R \cdot p_s^i + t - p_t^i)$$

$$= 2Nt + 2R\sum_{i=1}^{N} p_s^i - 2\sum_{i=1}^{N} p_t^i$$

- Set the derivative equals to zero.

$$t = \frac{1}{N}\sum_{i=1}^{N} p_t^i - R\frac{1}{N}\sum_{i=1}^{N} p_s^i$$

$$= \bar{p}_t - R\bar{p}_s$$

3D-3D Pose Estimation

$$\hat{p} = p - \bar{p}$$

$$F(R) = \sum_{i=1}^{N} ||R \cdot \hat{p}_s^i - \hat{p}_t^i||^2$$

- We solve a minimization problem for N >= 3 point correspondences:

$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i}^{N} \left\| \left(\boldsymbol{R}\boldsymbol{p}_s^i + \boldsymbol{t} \right) - \boldsymbol{p}_t^i \right\|^2$$

- After differentiating with respect to $\boldsymbol{t}$,

$$\frac{\partial F}{\partial t} = \sum_{i=1}^{N} 2(R \cdot p_s^i + t - p_t^i)$$

$$= 2Nt + 2R \sum_{i=1}^{N} p_s^i - 2 \sum_{i=1}^{N} p_t^i$$

- Set the derivative equals to zero.

$$t = \frac{1}{N} \sum_{i=1}^{N} p_t^i - R \frac{1}{N} \sum_{i=1}^{N} p_s^i$$

$$= \bar{p}_t - R\bar{p}_s$$

3D-3D Pose Estimation

- We replace the optimal t into function, rewrite the objective function as

$$F(R) = \sum_{i=1}^{N} ||R \cdot \hat{p}_s^i - \hat{p}_t^i||^2 \qquad \hat{p} = p - \bar{p}$$

$$\begin{aligned}
||R \cdot \hat{p}_s^i - \hat{p}_t^i||^2 &= (R \cdot \hat{p}_s^i - \hat{p}_t^i)^T (R \cdot \hat{p}_s^i - \hat{p}_t^i) \\
&= ({\hat{p}_s^i}^T R^T - {\hat{p}_t^i}^T)(R \cdot \hat{p}_s^i - \hat{p}_t^i) \\
&= {\hat{p}_s^i}^T R^T R \hat{p}_s^i - {\hat{p}_t^i}^T R \hat{p}_s^i - {\hat{p}_s^i}^T R^T \hat{p}_t^i + {\hat{p}_t^i}^T \hat{p}_t^i \\
&= ||\hat{p}_s^i||^2 + ||\hat{p}_t^i||^2 - {\hat{p}_t^i}^T R \hat{p}_s^i - {\hat{p}_s^i}^T R^T \hat{p}_t^i \\
&= ||\hat{p}_s^i||^2 + ||\hat{p}_t^i||^2 - 2{\hat{p}_t^i}^T R \hat{p}_s^i
\end{aligned}$$

$$R^* = \arg\min_R \left(-2 \sum_{i=1}^{N} {\hat{p}_t^i}^T R \hat{p}_s^i\right) \qquad R^* = \arg\max_R \left(\sum_{i=1}^{N} {\hat{p}_t^i}^T R \hat{p}_s^i\right)$$

3D-3D Pose Estimation

$$R^* = \arg\max_R \left(\sum_{i=1}^{N} {\hat{p}_t^i}^T R \hat{p}_s^i\right)$$

$$\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} a_{ii} = a_{11} + a_{22} + \cdots + a_{nn}$$

$$\sum_{i=1}^{N} {\hat{p}_t^i}^T R \hat{p}_s^i = trace(P_t^T R P_s)$$

- Based on the definition of matrix multiplication and trace, the problem can be transformed into

$$R^* = \arg\max_R trace(P_t^T R P_s)$$

- Some useful mathematics
  - $\mathrm{tr}(\boldsymbol{AB}) = tr(\boldsymbol{BA})$
  - $\mathrm{tr}(\boldsymbol{A}) = tr(\boldsymbol{A}^T)$
  - $\mathrm{tr}(\boldsymbol{A}+\boldsymbol{B}) = tr(\boldsymbol{A}) + tr(\boldsymbol{B})$

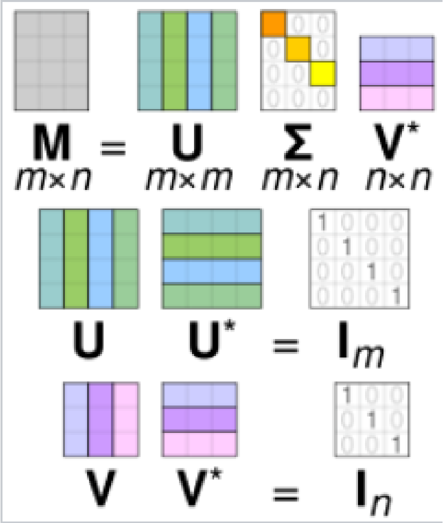
3D-3D Pose Estimation

- The 3D-3D pose problem reduced to

$$R^* = \arg\max_R trace(P_t^T R P_s)$$

$$\begin{aligned} trace(P_t^T R P_s) &= trace(R P_s P_t^T) \\ &= trace(RH) \\ &= trace(RU\Sigma V^T) \\ &= trace(\Sigma V^T R U) \end{aligned}$$

Singular value decomposition



- Note that V, U, and R are all orthogonal matrices, so $V^T R U$ is also an orthogonal matrix.

$$M = V^T R U = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix}$$

$$\begin{aligned} trace(\Sigma V^T R U) &= trace(\Sigma M) \\ &= \sigma_1 m_{11} + \sigma_2 m_{22} + \sigma_3 m_{33} \end{aligned}$$

3D-3D Pose Estimation

- non-negativity of singular values
- orthogonal matrices (where the absolute values of the elements are at most 1)

$$trace(\Sigma V^T RU) = trace(\Sigma M) = \sigma_1 m_{11} + \sigma_2 m_{22} + \sigma_3 m_{33} \leq \sigma_1 + \sigma_2 + \sigma_3$$

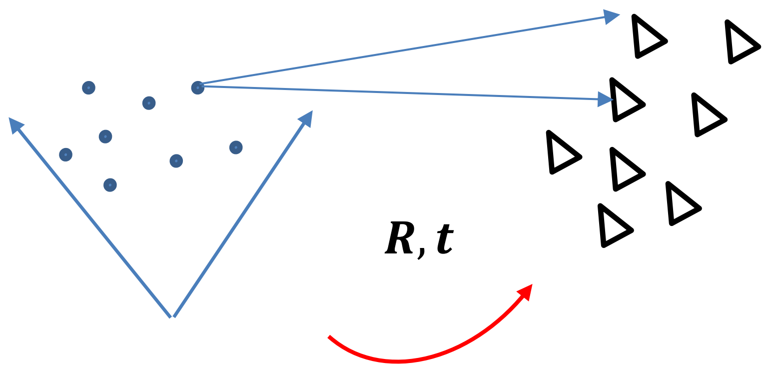
- maximum occurs only when M is the identity matrix

$$V^T RU = I$$
$$R = VU^T$$

- For orthogonal matrices $\det(Q) = \pm 1$
- R is special $SO(3)$, $\det(R) = 1$
  - If $\det(R) = 1$ $\quad R^* = VU^T$
  - If $\det(R) = -1$

$$R^* = V\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & -1\end{bmatrix}U^T \Rightarrow R^* = V\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & \det(VU^T)\end{bmatrix}U^T$$

3D-3D Pose Estimation



- 3D-3D pose estimation problem:

$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i}^{N} \left\| \left(\boldsymbol{R}\boldsymbol{p}_s^i + \boldsymbol{t} \right) - \boldsymbol{p}_t^i \right\|^2$$

- Closed-form solution:

$$\boldsymbol{R} \quad = \mathrm{V} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{bmatrix} U^T$$

$$\mathbf{t} = \frac{1}{N} \sum\nolimits_{i}^{N} \boldsymbol{p}_t^i - \boldsymbol{R} \frac{1}{N} \sum\nolimits_{i}^{N} \boldsymbol{p}_s^i = \overline{\boldsymbol{p}}_t - \boldsymbol{R}\overline{\boldsymbol{p}}_s$$

3D-3D Pose Estimation

- How to obtain 3D-3D data association?

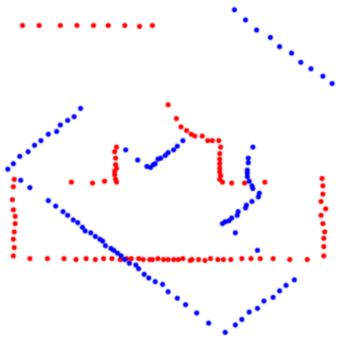
???

$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i}^{N} \left\| \left(\boldsymbol{R}\boldsymbol{p}_s^i + \boldsymbol{t} \right) - \boldsymbol{p}_t^i \right\|^2$$

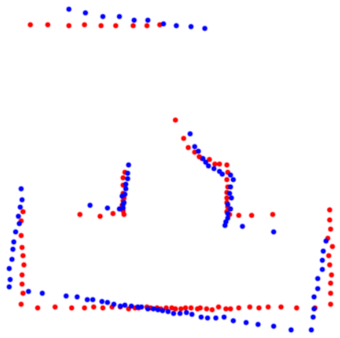
Content

3D-3D Pose Estimation

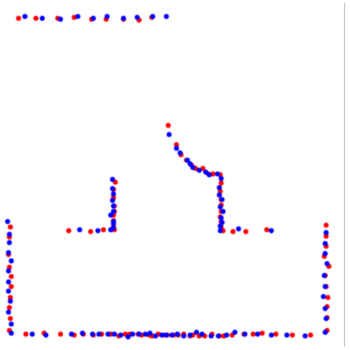
- How to obtain 3D-3D data association?
  - “Soft” data association directly from point clouds
  - The Iterative Closest Point (ICP) algorithm
    - Search of nearest neighbors
      - implementation: $O(MN)$
    - Need to speed up
    - K-d Tree: $O(MlogN)$



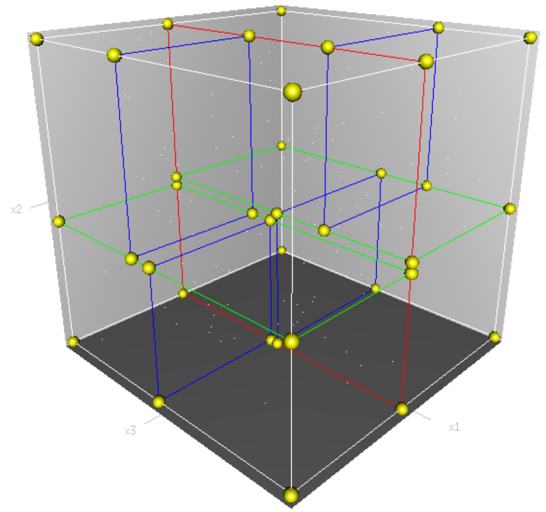
K=0



K=1



K=2

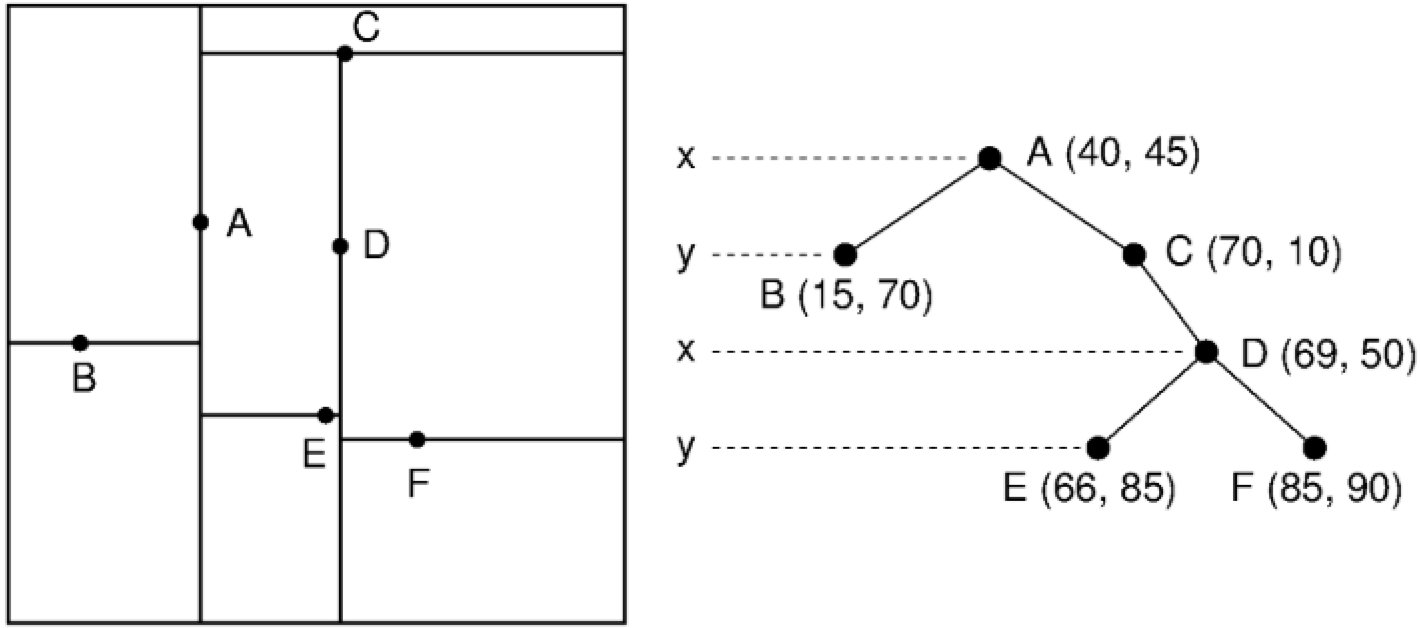


KD Tree

- kd树(K-dimension tree)是一种对k维空间中的实例点进行存储以便对其进行快速检索的树形数据结构。

- kd树是是一种二叉树，表示对k维空间的一个划分，构造kd树相当于不断地用垂直于坐标轴的超平面将K维空间切分，构成一系列的K维超矩形区域。kd树的每个结点对应于一个k维超矩形区域。

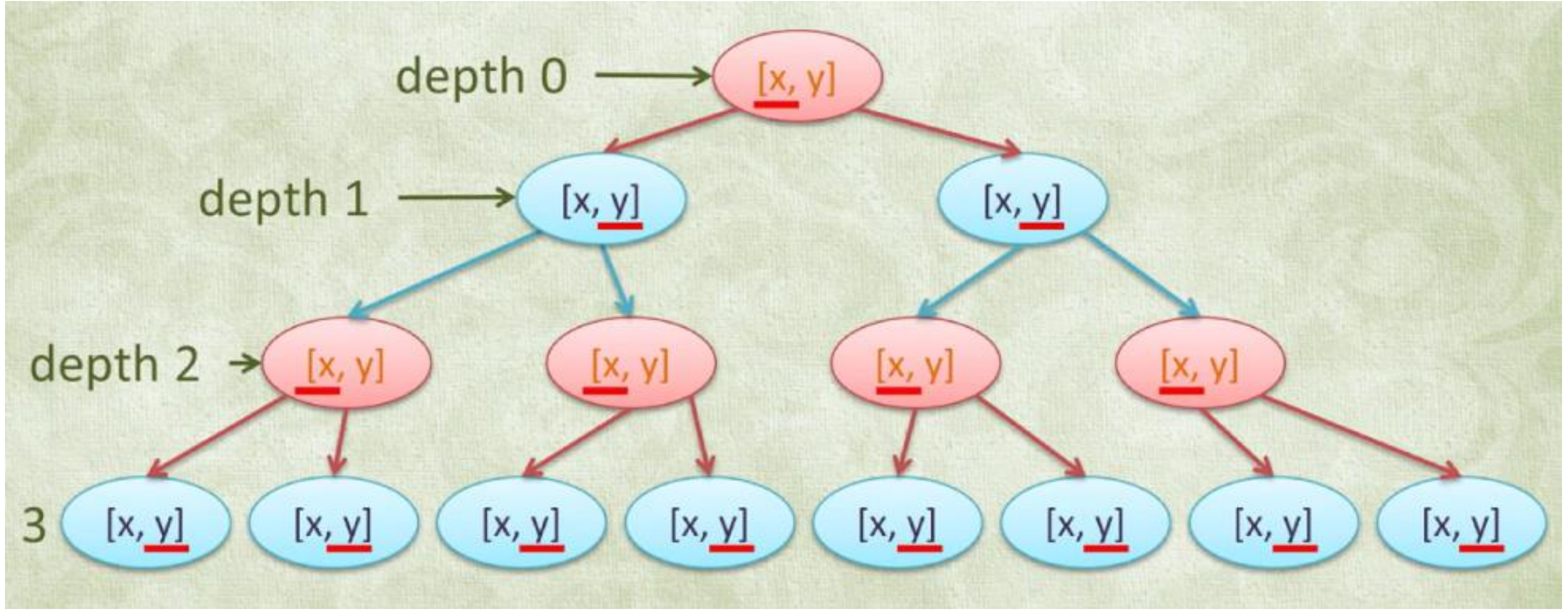
- 利用kd树可以省去对大部分数据点的搜索，从而减少搜索的计算量。



KD Tree

构建 KD Tree

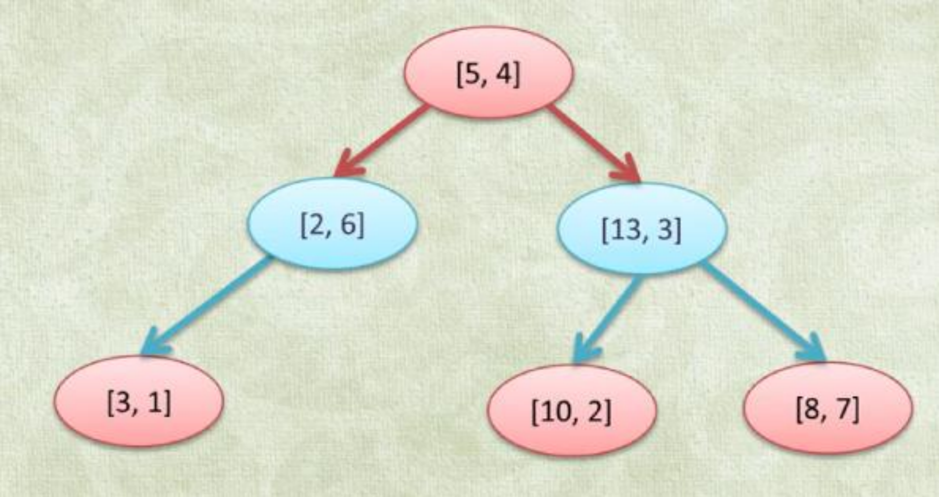
二维[x, y]

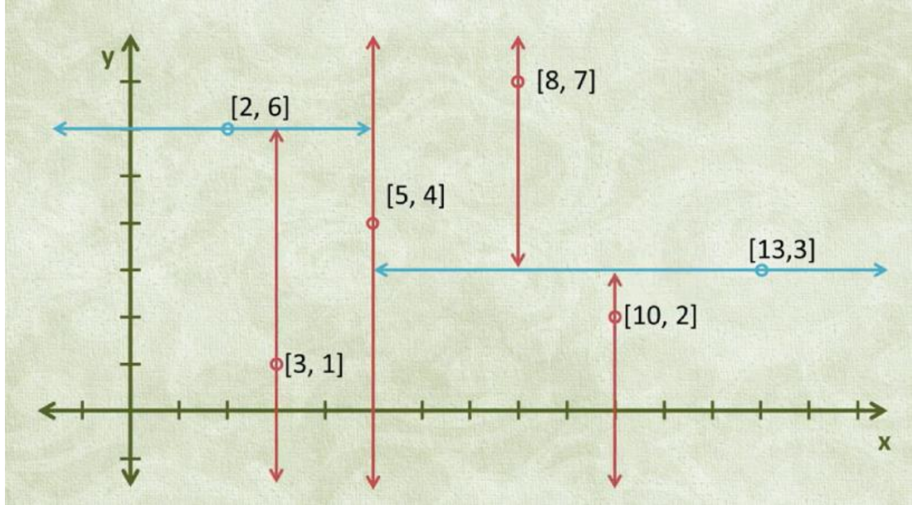


KD Tree

构建 KD Tree

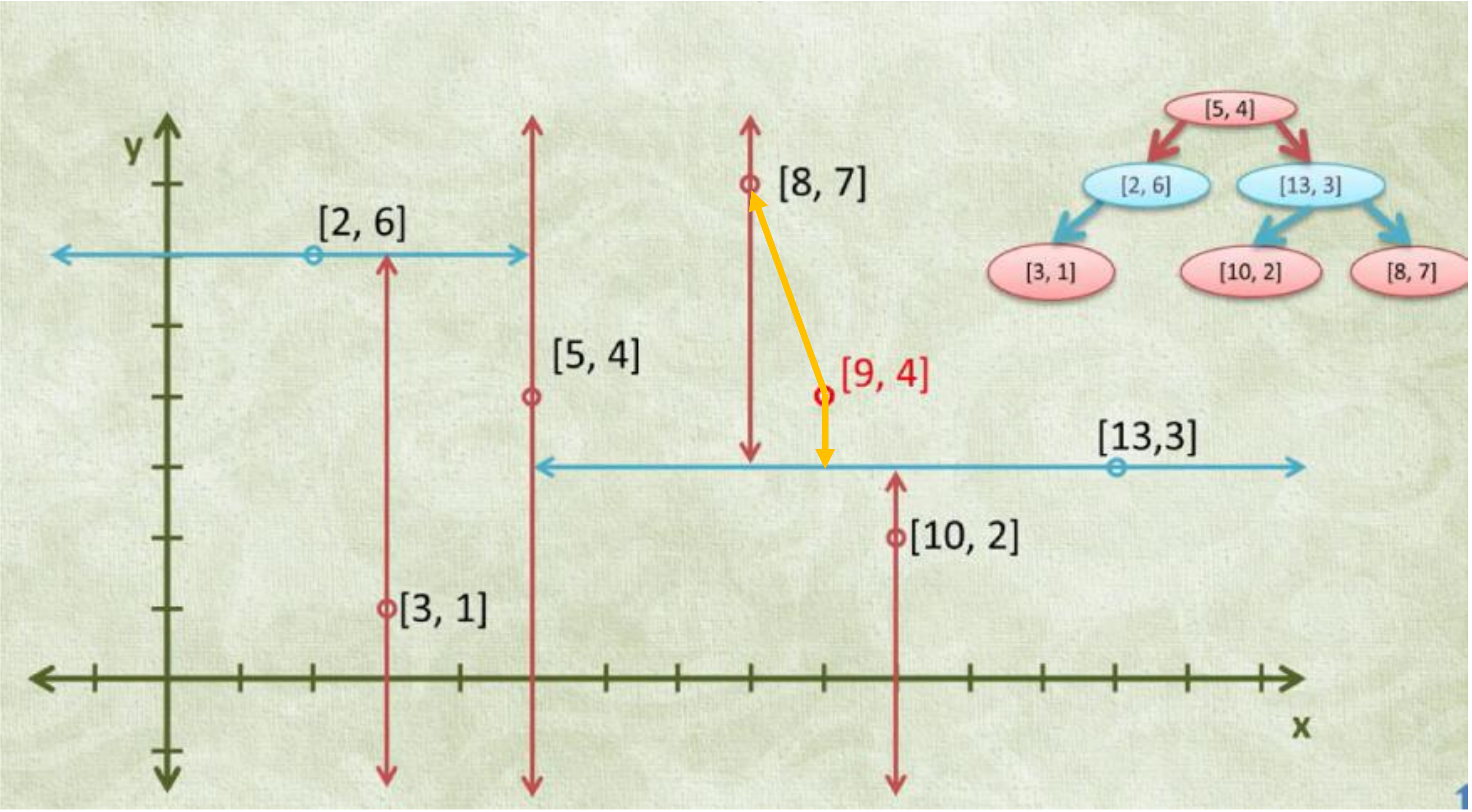
[5, 4] [2, 6] [13,3] [8, 7] [3, 1] [10, 2]





KD Tree

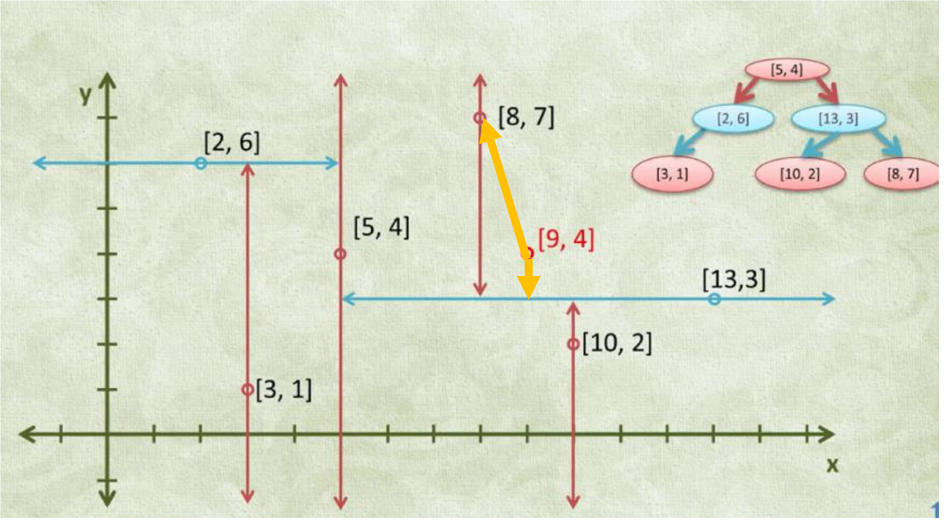
KD Tree 最近点查询 [9, 4]



KD Tree

伪代码 回溯法

```
bestNode, bestDist = None, inf
def NearestNodeSearch(curr_node):
    if curr_node == None:
        return
    if distance(curr_node, q) < bestDist:
        bestDist = distance(curr_node, q)
        bestNode = curr_node
    if q_i < curr_node_i:
        NearestNodeSearch(curr_node.left)
    else:
        NearestNodeSearch(curr_node.right)
    if |curr_node_i - q_i| < bestDist:
        NearestNodeSearch(curr_node.other)
```

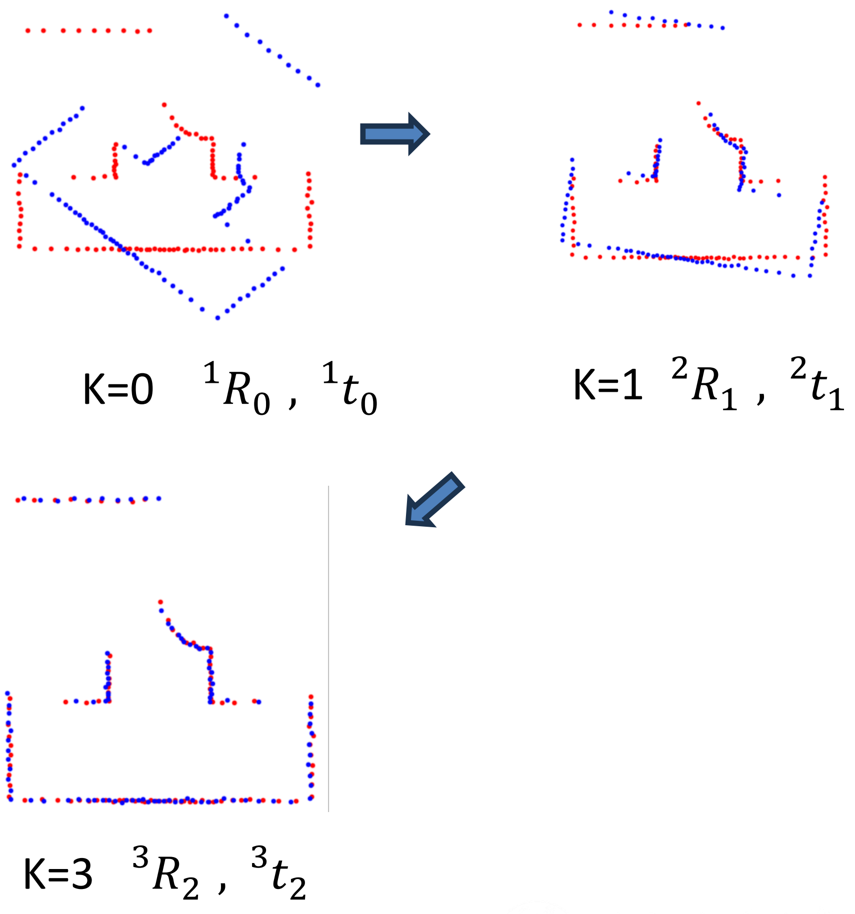


3D-3D Pose Estimation

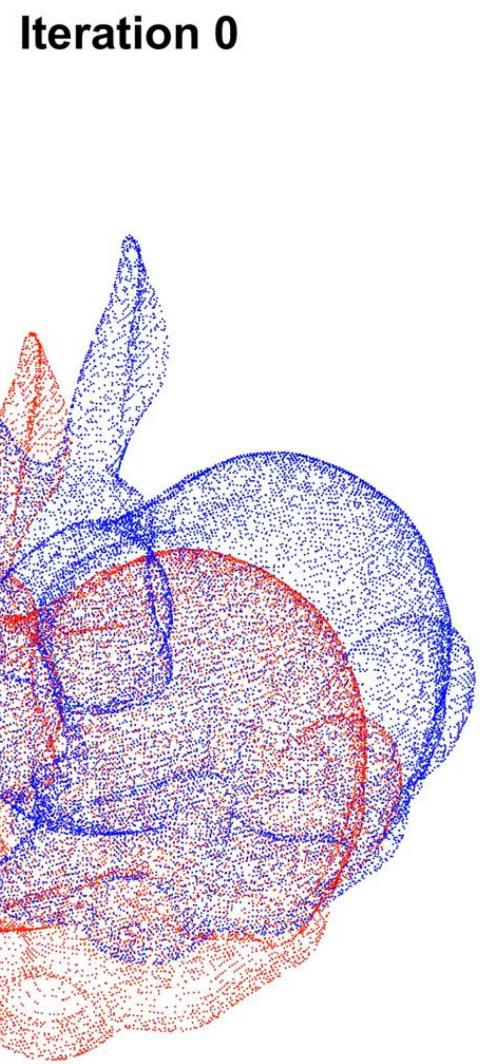
$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i}^{N} \left\| \left(\boldsymbol{R}\boldsymbol{p}_s^i + \boldsymbol{t} \right) - \boldsymbol{p}_t^i \right\|^2$$

- How to obtain 3D-3D data association?
  - “Soft” data association directly from point clouds
  - The Iterative Closest Point (ICP) algorithm
    1. Start with some initial guess of rotation and translation
    2. For each point in pointcloud1, find its nearest neighbor in pointcloud2 based on the current estimated rotation and translation by KD tree
    3. Refine the rotation and translation based on the latest data association
    4. Iterate from step 2 until converge

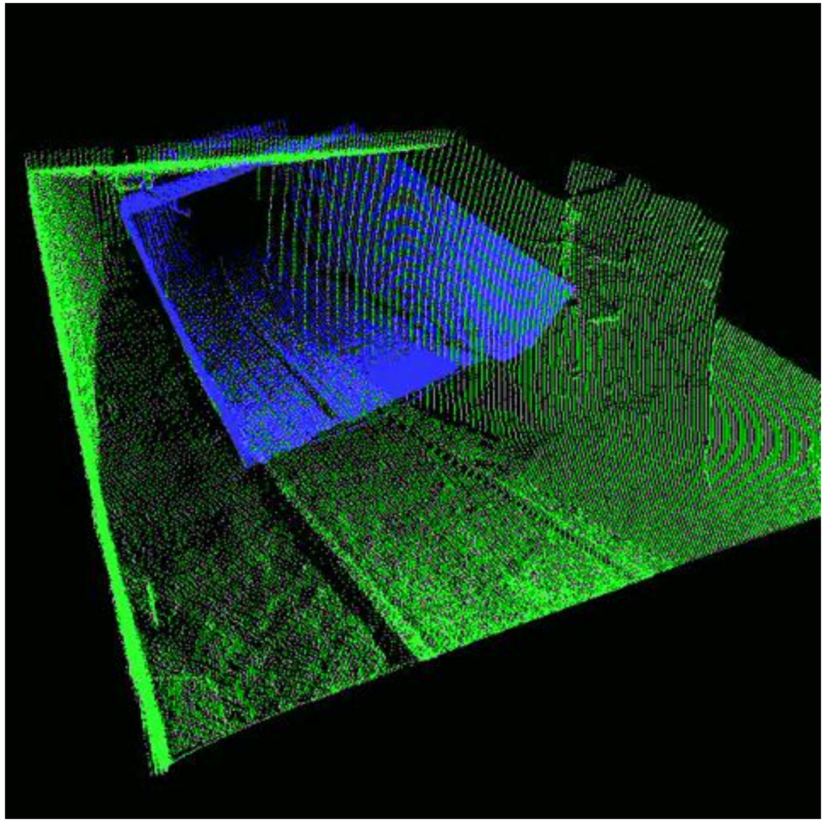
Greedy algorithm

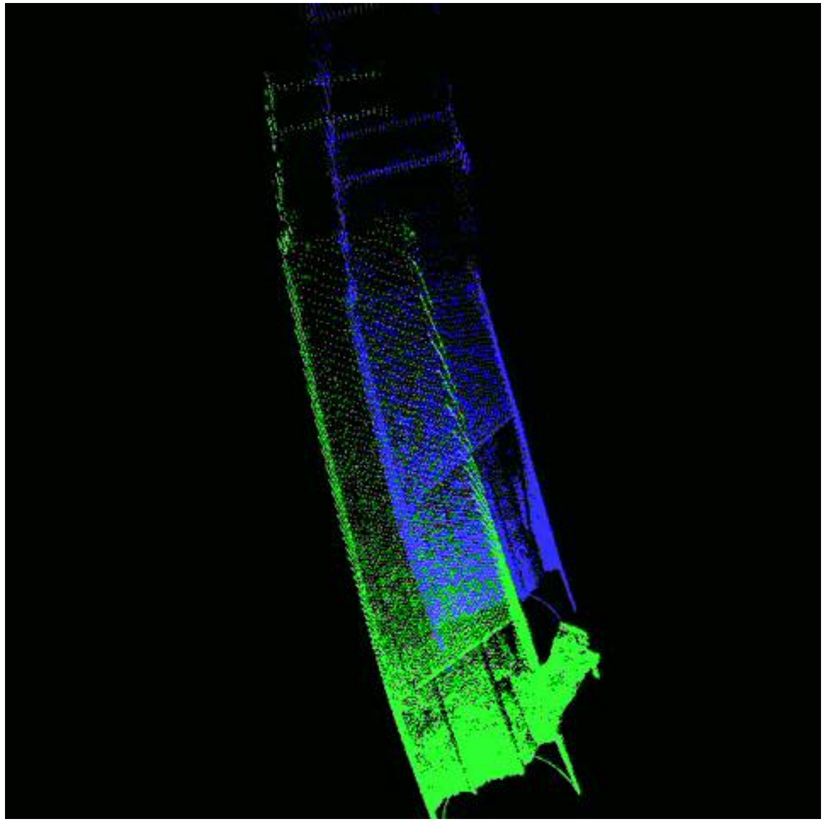


Point Cloud Registration (ICP)



3D-3D Registration of Point Cloud

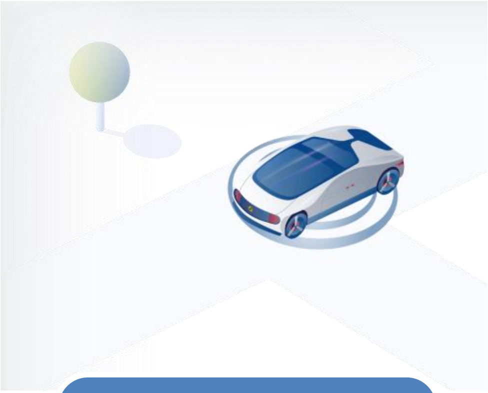




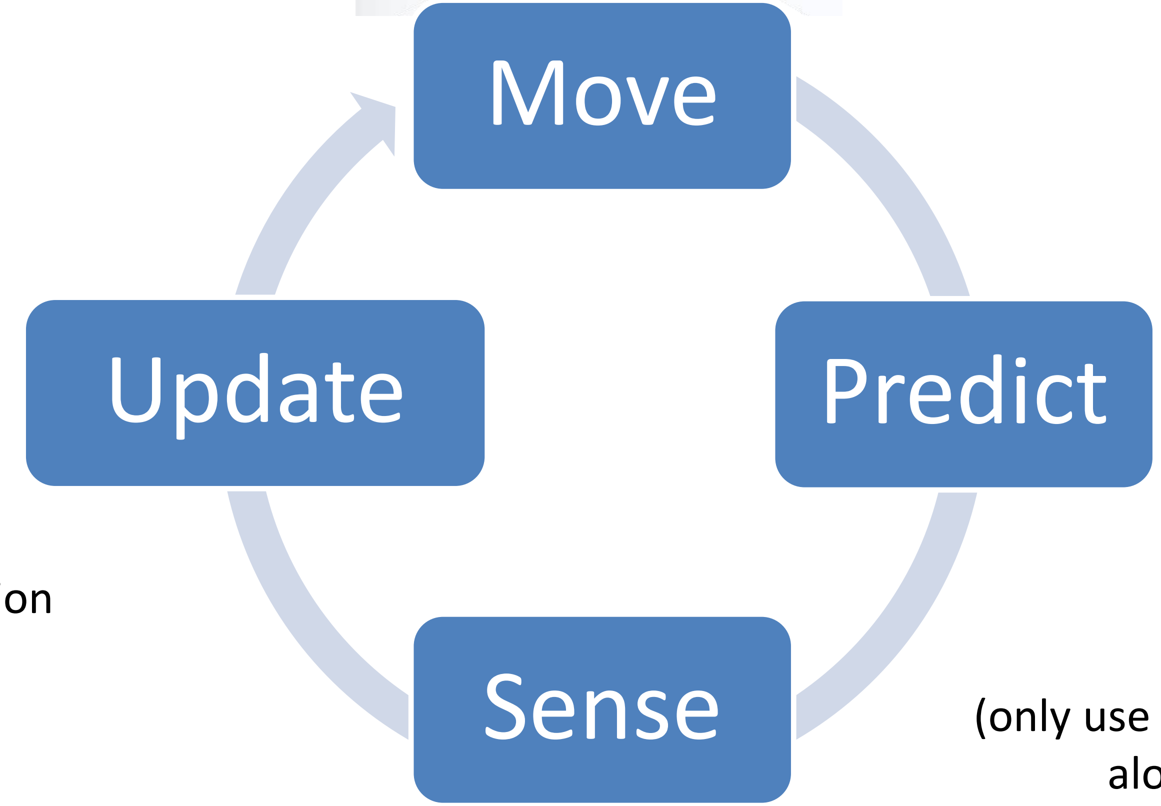
Content

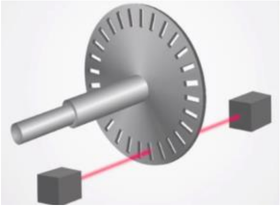
1. Semantic Point Registration
2. Nearest Neighbor Search：KD-Tree
3. Vision-IMU-Wheel Fusion EKF
4. Assignment

Vision-IMU-Wheel Fusion EKF

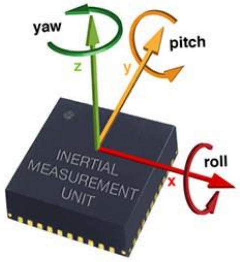


$$\boldsymbol{x} = [x, y, \theta]$$

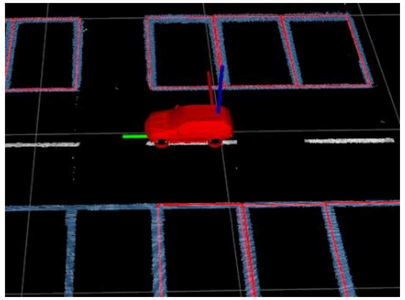




Wheel odometer

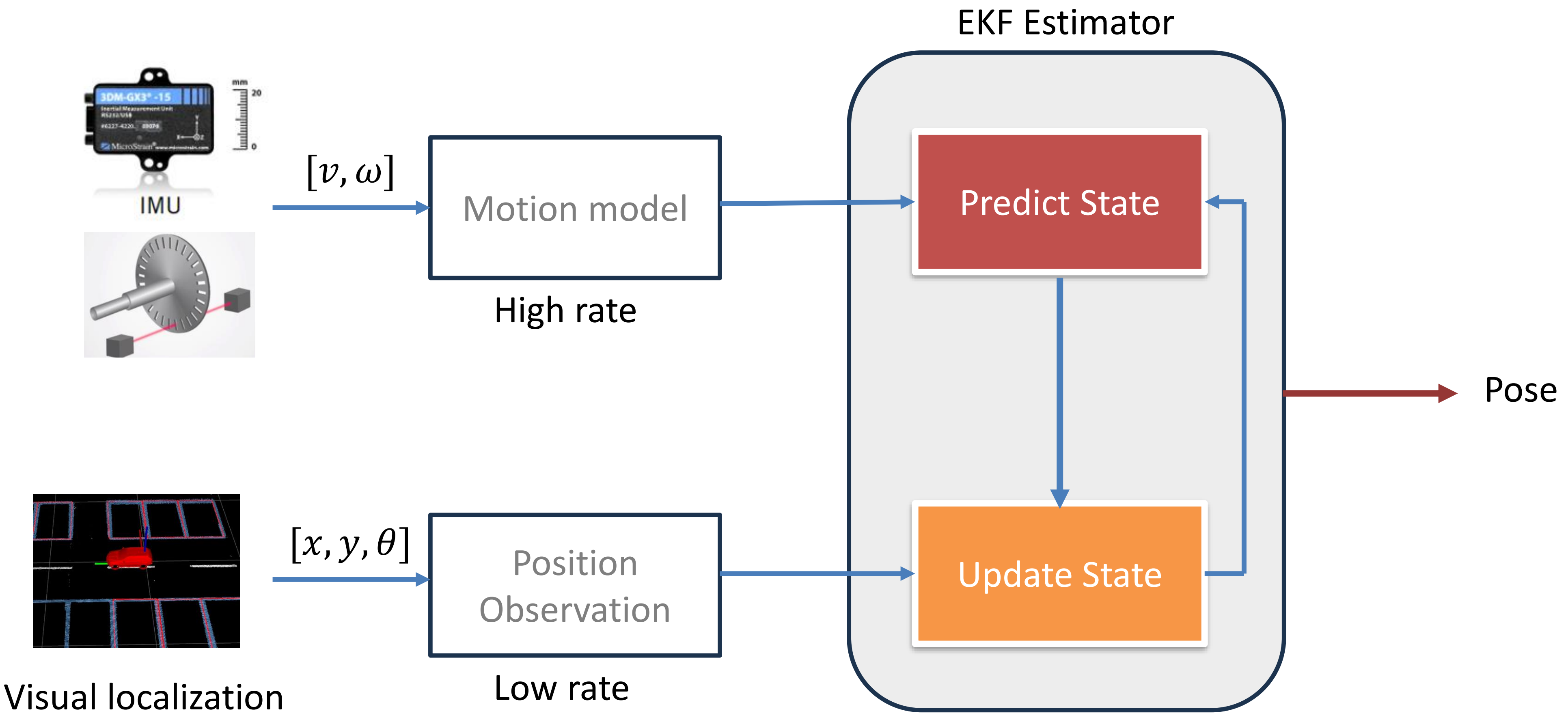


IMU
(only use angular velocity
along z-axis)

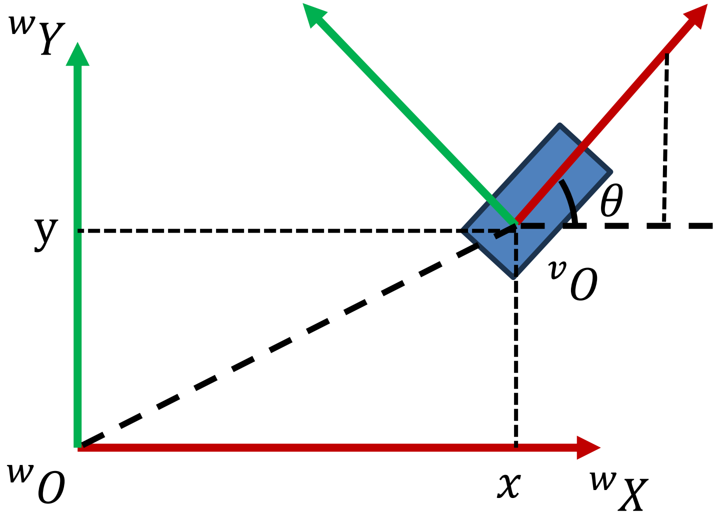


Visual semantic localization

Vision-IMU-Wheel Fusion EKF



Vision-IMU-Wheel Fusion EKF



Vehicle 2D State: $\boldsymbol{x} = [x, y, \theta]$

Motion signal: $\boldsymbol{u} = [v, \omega]$ velocity, angular velocity

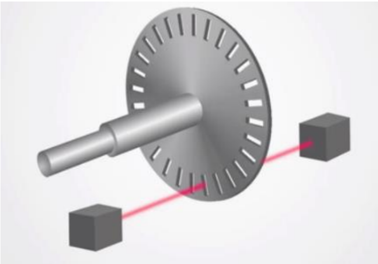
with Gaussian white noise

$n_v \sim N(0, Q_v),\ n_\omega \sim N(0, Q_\omega)$

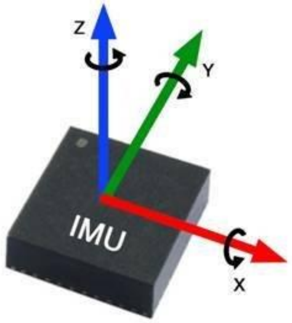
Motion model:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{n}) = \begin{bmatrix} \cos(\theta) \cdot (v + n_v) \\ \sin(\theta) \cdot (v + n_v) \\ \omega + n_\omega \end{bmatrix}$$

Motion sensor:



Velocity



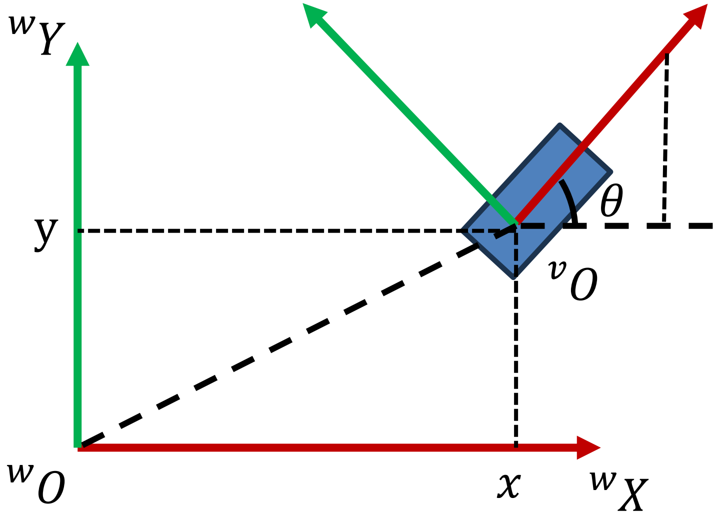
Angular velocity

Linearization:

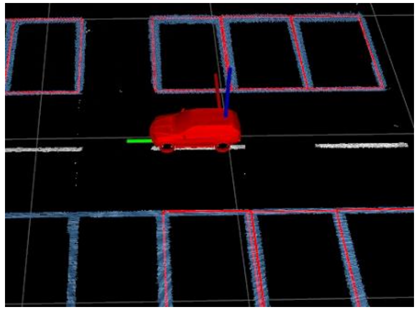
$$A_t = \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_{t-1}, \boldsymbol{u}_t, 0} = \begin{bmatrix} 0 & 0 & -v \cdot \sin(\theta) \\ 0 & 0 & v \cdot cos(\theta) \\ 0 & 0 & 0 \end{bmatrix}$$

$$U_t = \left.\frac{\partial f}{\partial \boldsymbol{n}}\right|_{\boldsymbol{x}_{t-1}, \boldsymbol{u}_t, 0} = \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{bmatrix}$$

Vision-IMU-Wheel Fusion EKF



Semantic Localization:



Position and orientation

Vehicle 2D State: $\boldsymbol{x} = [x, y, \theta]$

Measurement： $\boldsymbol{z} = [x, y, \theta]$ position and yaw angle

with Gaussian white noise

$\boldsymbol{n}_z \sim N(0, R_z)$

Measurement model:

$$z_t = g(x_t, n_t) = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + \boldsymbol{n}_z$$

Linearization:

$$C_t = \left.\frac{\partial g}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_{t},0} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$W_t = \left.\frac{\partial f}{\partial \boldsymbol{n}}\right|_{\boldsymbol{x}_{t},0} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Vision-IMU-Wheel Fusion EKF

- Motion Model:
  - $\dot{x} = f(x, u, n)$
  - $n_t \sim N(0, Q_t)$

  Assumptions
  - $A_t = \left.\frac{\partial f}{\partial x}\right|_{\mu_{t-1}, u_t, 0}$
  - $U_t = \left.\frac{\partial f}{\partial n}\right|_{\mu_{t-1}, u_t, 0}$

  Linearization
  - $F_t = I + \delta t\, A_t$
  - $V_t = \delta t\, U_t$

  Discretization
- Prediction step:
  - $\bar{\mu}_t = \mu_{t-1} + \delta t\, f(\mu_{t-1}, u_t, 0)$
  - $\bar{\Sigma}_t = F_t\, \Sigma_{t-1}\, F_t^T + V_t\, Q_t\, V_t^T$

- Measurement Model:
  - $z_t = g(x_t, n_t)$
  - $n_t \sim N(0, R_t)$

  Assumptions
  - $C_t = \left.\frac{\partial g}{\partial x}\right|_{\bar{\mu}_t, 0}$
  - $W_t = \left.\frac{\partial g}{\partial n}\right|_{\bar{\mu}_t, 0}$

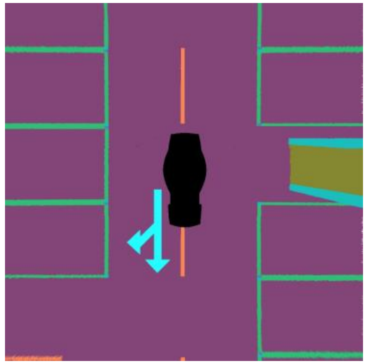
  Linearization
- Update step:
  - $K_t = \bar{\Sigma}_t\, C_t^T\, (C_t\, \bar{\Sigma}_t\, C_t^T + W_t\, R\, W_t^T)^{-1}$
  - $\mu_t = \bar{\mu}_t + K_t\, (z_t - g(\bar{\mu}_t, 0))$
  - $\Sigma_t = \bar{\Sigma}_t - K_t\, C_t\, \bar{\Sigma}_t$

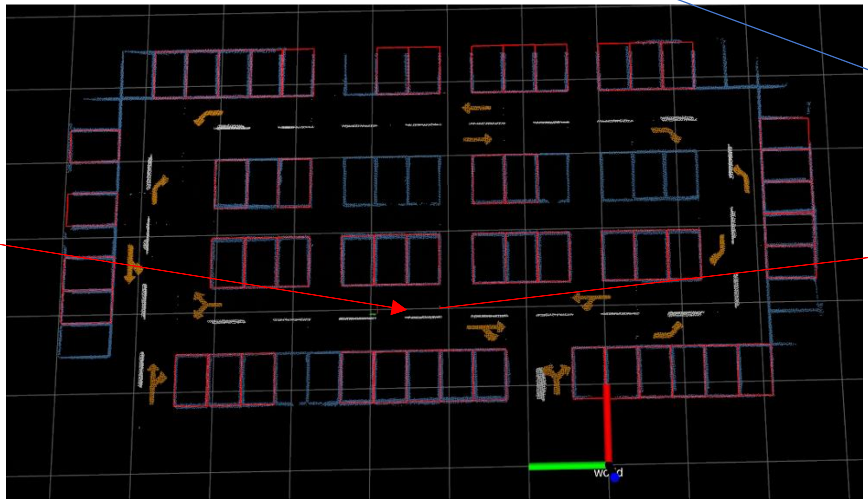
Content

Assignment

- Bag file
  - Segmentation image
  - Imu
  - Wheel speed
- AVP Map

- Coding task：
  - ICP Localization
  - Semantic + IMU + Wheel speed fusion





EKF

深蓝学院
shenlanxueyuan.com

感谢聆听！

Thanks for Listening