- Joint.mat
  - Joint angles
  - pos: Matrix of positions (Maybe you don’t need)
  - ts: timestamps (absolute time)
  - gyro: angular velocity of body
  - acc: acceleration of body
  - rpy: roll, pitch and yaw angles of body
  - Head_angles = [Neck angle(yaw), head angle(Pitch)];
  - Ft_l, ft_r : Torque/Force sensor of left and right foot (Maybe you don’t need)

- lidar.mat
  - t: 1.4268e+09(absolute time)
  - ~~rsz: 4324~~ (You don't need it)
  - pose: [0 0 0] (global odometry)
  - res: 0.0044 (radian, resolution) (theta = -135:0.25:135)
  - rpy: [-0.0120 -0.0164 -0.1107] (IMU roll pitch yaw)
  - scan: [1x1081 single] (Scan data, range -135deg to 135 deg)
- lidar.rpy and joint.rpy are identical l for the same time stamps

- Odometry
  - lidar{i}.pose: [x, y, theta]
  - +x: forward from robot
  - +y: left from robot
  - +z: up from robot
  - theta: rotation around +z

- Relative pose based on the odometry $(o_{t+1} \ominus o_t)$
  - Given global odometry
  - Find delta x, delta y and delta theta

$$\begin{bmatrix} O_{x\_t} \\ O_{y\_t} \end{bmatrix} = \begin{bmatrix} \cos\theta_{t-1} & \sin\theta_{t-1} \\ -\sin\theta_{t-1} & \cos\theta_{t-1} \end{bmatrix} \times \begin{bmatrix} x_t - x_{t-1} \\ y_t - y_{t-1} \end{bmatrix}$$

$$O_{\theta\_t} = \theta_t - \theta_{t-1}$$

Transform to local coordinate frame!!!!

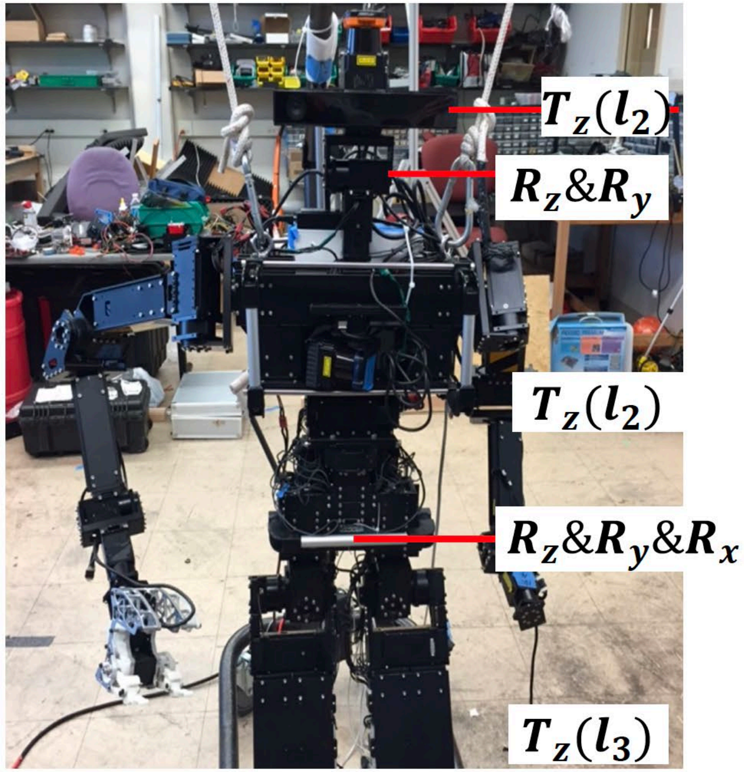
- Adding random noise
  - Random noises (mu = 0, sigma = σ) to odometry
  - For N particles, generate N random noise values

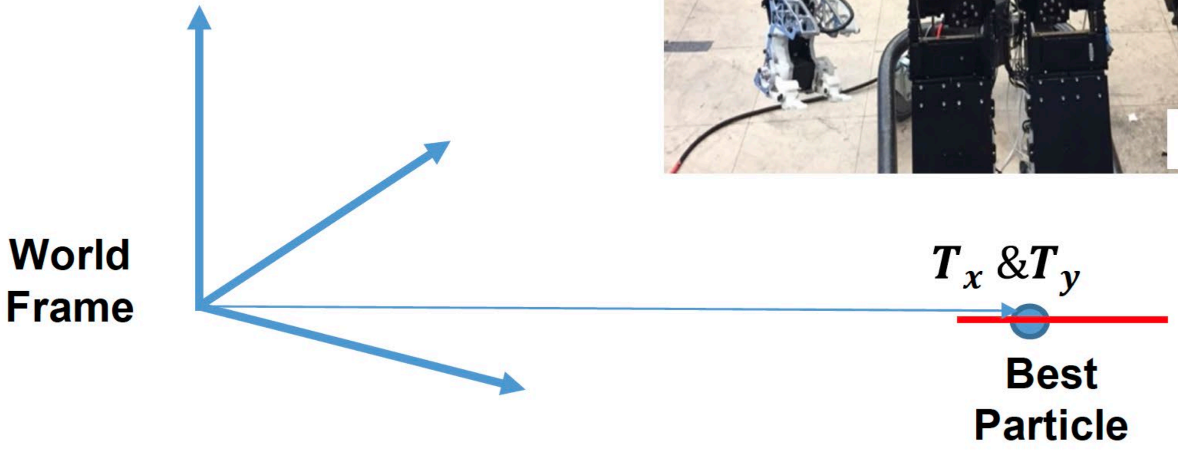
$$p_{t+1} = p_t \oplus (o_{t+1} \ominus o_t)$$

- Prediction$[P_{x_t}, P_{y_t}, P_{\theta_t}]$
  - $\begin{bmatrix} P_{x\_t} \\ P_{y\_t} \end{bmatrix} = \begin{bmatrix} P_{x_{t-1}} \\ P_{y_{t-1}} \end{bmatrix} + \begin{bmatrix} \cos P_{\theta_{t-1}} & -\sin P_{\theta_{t-1}} \\ \sin P_{\theta_{t-1}} & \cos P_{\theta_{t-1}} \end{bmatrix} \times \begin{bmatrix} O_{x\_t} \\ O_{y\_t} \end{bmatrix}$
  - $P_{\theta\_t} = P_{\theta_{t-1}} + O_{\theta\_t}$
  - Note : $P_{\theta_{t-1}}$ is the heading of a particle

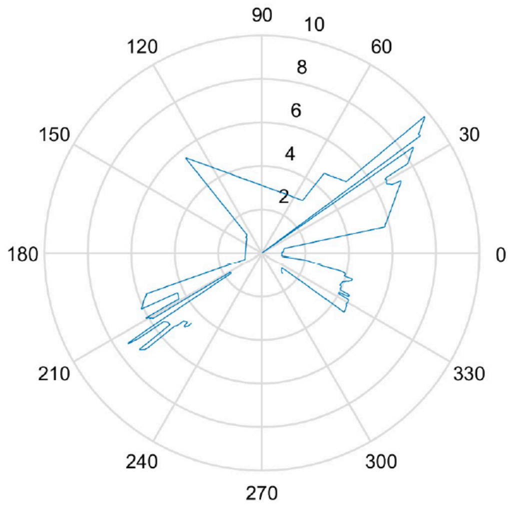
- The number of particles : 50 ~ 150
- Map resolution : 0.05
- Noise (when interval is 10)
  - Normal random([0.01, 0.01, 0.5*pi/180])
  - If norm(odo) < ε, noise is also zero
- Log odds Parameter
  - logOddOcc = 3, logOddFree = -1
  - Maximum : 120
  - Minimum :-120

- Input
  - $\boldsymbol{pf(x, y, \theta)}$
  - Body roll and pitch: $\boldsymbol{r, p}$
  - Head yaw and pitch





$$T = T_{xyz}(pf(x), pf(y), l_3)R_z(pf(\theta))R_y(p)R_x(r)T_z(l_2)R_z(head\_yaw)R_y(head\_pitch)T_z(l_1)$$

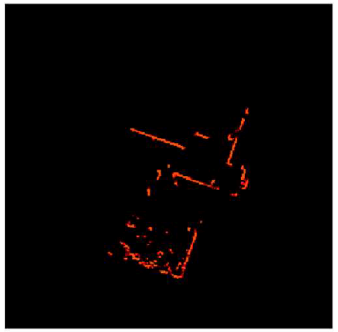


$$\text{scan} = [x_1, y_1, z_1; \ \dots \ x_{1081}, y_{1081}, z_{1081}]$$

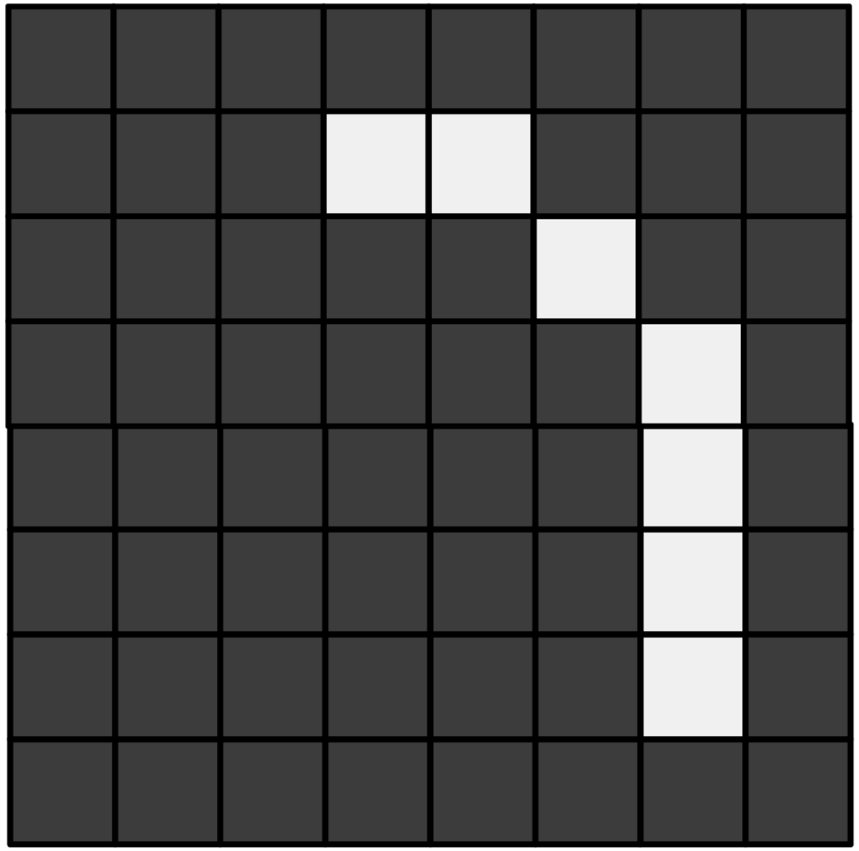
Polar Coordinate

Cartesian Coordinate w. r. t lidar frame

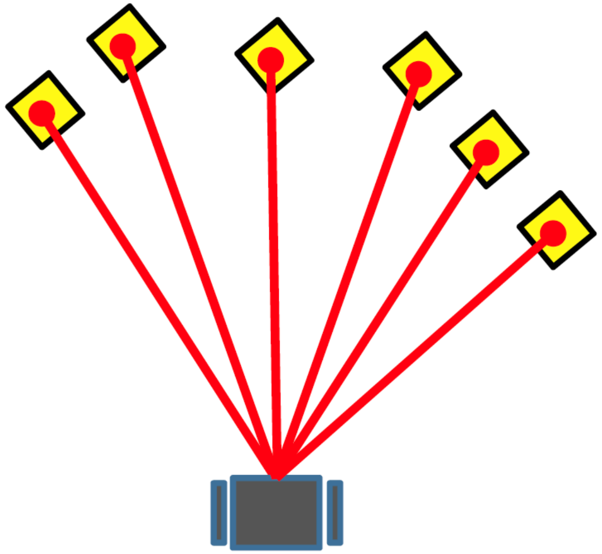
$$\text{scan\_world} = \mathrm{T} \times scan$$



Map

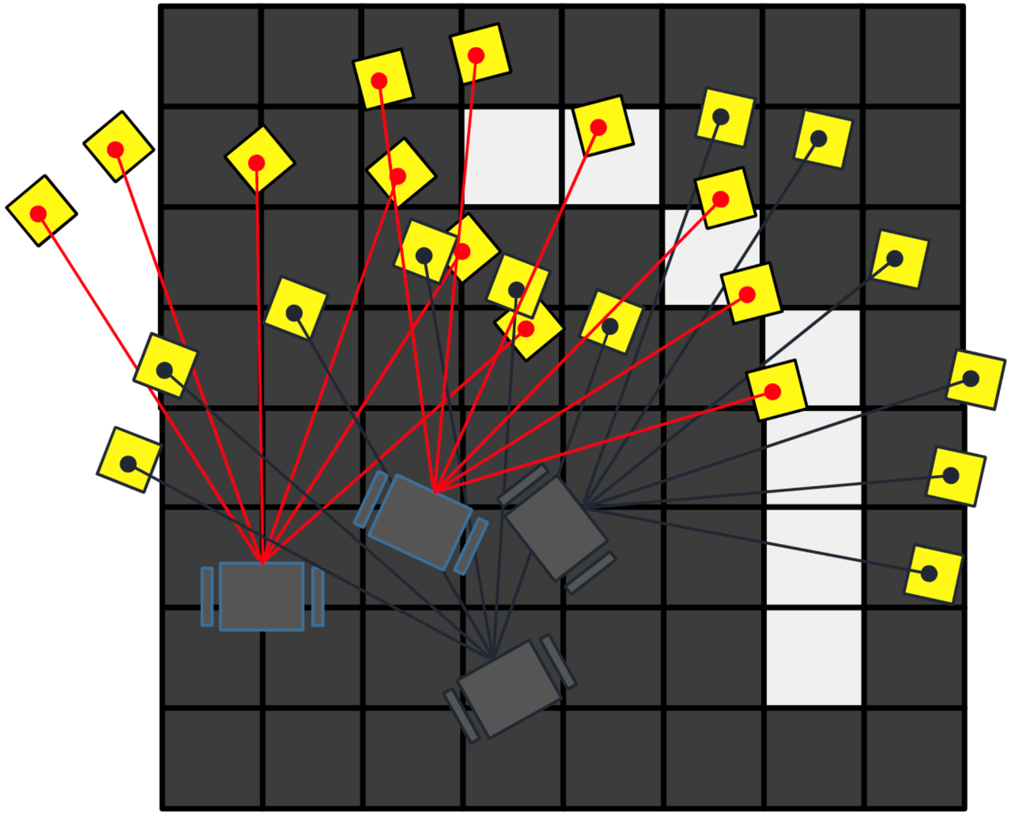


Range measurement

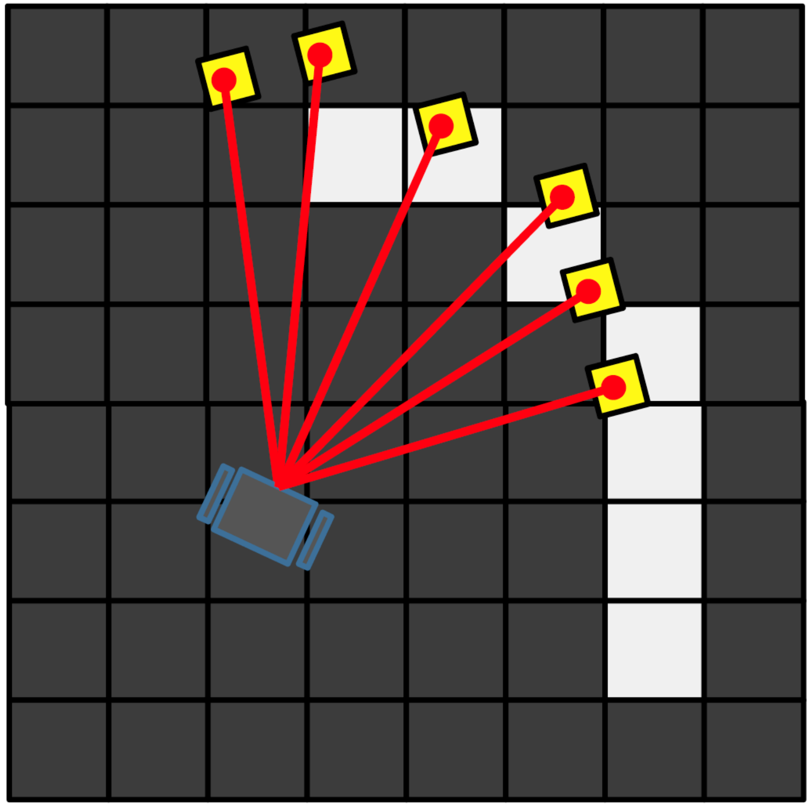


Robot

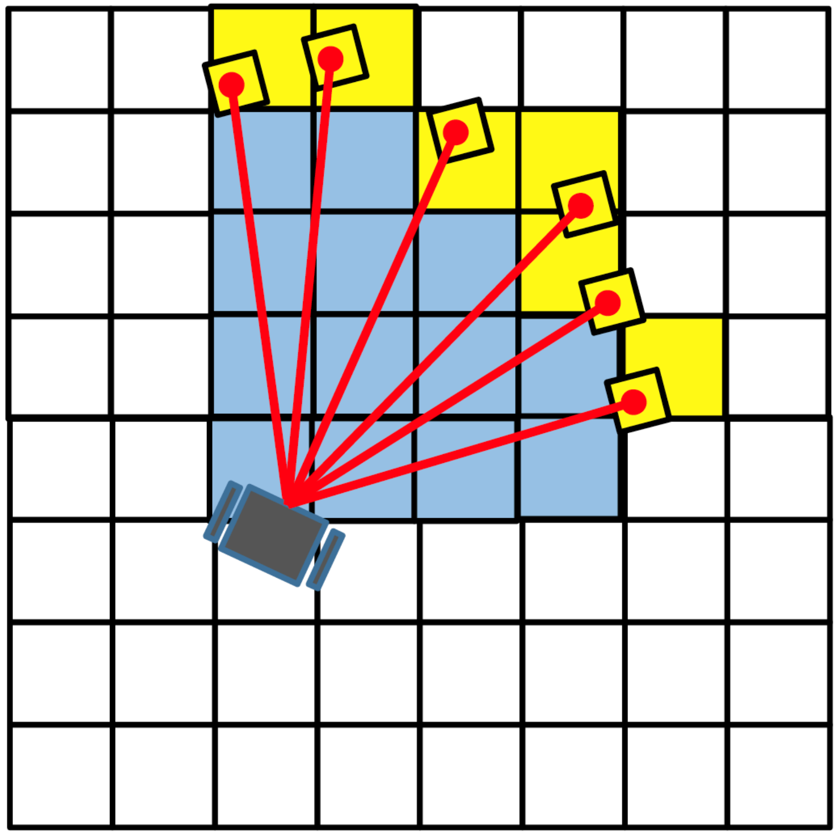
- Correlation-based Matching
  - Generate hypotheses (particles)



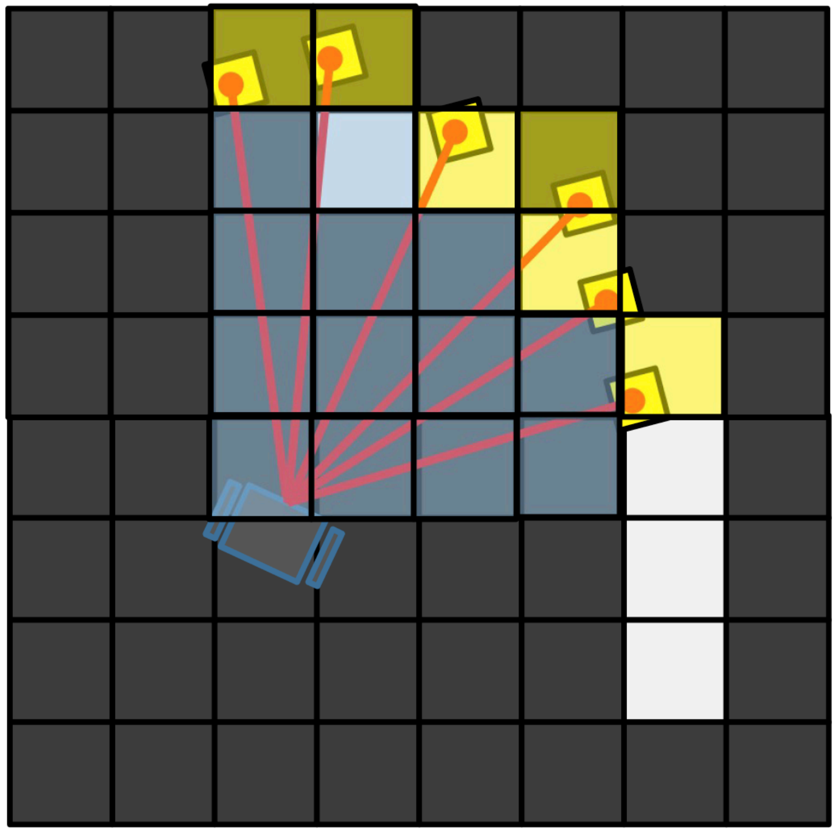
- Correlation-based Matching
  - For each hypothesis



- Correlation-based Matching
  - Build a local map from the measurement in a form that can be compared with the global map



- Correlation-based Matching
  - Evaluate hypotheses
    - score the hypothesis



- Correlation-based Matching (Find the best*)
  - Among all the hypotheses, choose the one that has the largest score in order to represent your current location