# Particle Filter SLAM

Baoqian Wang
Department of Electrical and Computer Engineering
University of California, San Diego
San Diego, CA, 92093
Email:bawang@ucsd.edu

***Abstract*—This project investigates using particle filter in simultaneous localization and mapping (SLAM) problem. In particular, the robot is localized using the particle filter which consists of prediction step and update step. Moreover, the environment where the robot is operated is estimated using Bayesian rule with Lidar data. The experimental studies are conducted using the real Lidar data ,images, etc., collected by the THOR humanoid robot. The proposed algorithm shows promising performance in SLAM problem of the robot.**

## I. INTRODUCTION

Simultaneous localization and mapping is a hot topic in robotics area. It concerns with two questions of the robot, 'where am I?' and 'what does the environment looks like?'. It plays an important role in many robotic applications such as navigation, environment reconstruction and so on.

In the literature, the SLAM has been widely studied. Early SLAM approaches were mostly based on maximum likelihood estimation (MLE) [1], maximum a posterior estimation (MAP) and bayesian inference (BI)[2]. As the representations of the robot states, map, observations and control inputs affect the SLAM, different SLAM approaches are adopted for different representations. For instance, the map can be represented by landmark-based map, occupancy grid, surfels and polygonal mesh. Popular occupancy grid SLAM algorithms include Fast SLAM[3], which uses a particle filter to maintain the robot trajectory pdf and log-odds mapping to maintain a probabilistic map for every particle. Moreover, another algorithm namely kinect fusion [4] matches consecutive RGBD point clouds using the iterative closest point algorithm and updates a grid discretization of the truncated signed distance function (TSDF) representing the scene surface via weighted averaging. Moreover, examples for popular landmark-based SLAM algorithms include Rao-Blackwellized Particle Filter[5], Kalman Filter[6], Factor Graphs SLAM [7].

In this project, we focus on occupancy grid SLAM algorithms. In particular, the environment is represented using grid map, which uses 1 and -1 to indicate if the cell is occupied or not, respectively. Then the particle filter is adopted to predict and update the states of the robot. The environment is obtained through Bayesian inference with Lidar data.

The rest of this paper is organized as follows. Section II formulates the SLAM problem. Section III presents the particle filter SLAM approaches. The performance of the approaches are evaluated comprehensively in Section IV. Section V concludes the paper with a brief summary.

## II. PROBLEM FORMULATION

In this section, the Particle SLAM problem is formulated.

### A. Map Representation

The environment is represented with an occupancy grid map. In particular, the environment is divided into a regular grid with $n$ cells. A vector $\mathbf{m} \in \mathbb{R}^n$ is used to model the environment. The $i$-th entry indicates whether the $i$-th cell is free ($m_i$=-1) or occupied ($m_i$=1) (see Figure 1 as an illustration).

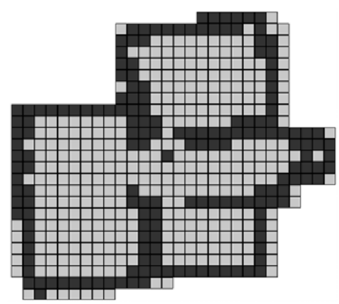


Figure 1. Illustration of occupancy grid map

### B. Observation Model

The observation model relates the robot state $\boldsymbol{x}$ and the surrounding enviroment $\boldsymbol{m}$ with the sensor observation $\boldsymbol{z}$ subject to measurement noise $\boldsymbol{v}$, which is captured by

$$\mathbf{z}_t = h\left(\mathbf{x}_t, \mathbf{m}_t, \mathbf{v}_t\right) \tag{1}$$

Due to the measurement noise, the observation $\mathbf{z}_t$ is a random variable and can be described by the following conditional probability density function (pdf)

$$\boldsymbol{z_t} \sim p_h\left(\cdot|\mathbf{x}_t, \mathbf{m}_t\right) \tag{2}$$

### C. Motion Model

The motion model in this project is based on odometry data including the sensor data such as wheel encoders, IMU, camera and laser. The model is represented by

$$\boldsymbol{x}_{t+1|t}^{(k)} = f\left(\boldsymbol{x}_{t|t}^{(k)}, \mathbf{u}_t + \boldsymbol{\epsilon}_t\right) \tag{3}$$

where $x$ is the state of the robot, $f$ is the differential-drive motion model, $u_t = (v_t \ , \ \omega_t)$ is the linear and angular velocity input (either known or obtained from the Encoders and IMU), and $\epsilon_t$ N $(0, \epsilon)$ is a 2-D Gaussian motion noise

Similarly, the state $\mathbf{x}$ of the robot is also a random variable, which is represented by

$$\boldsymbol{x}_{t+1|t}^{(k)} \sim p_f \left( \cdot | \boldsymbol{x}_{t|t}^{(k)}, \mathbf{u}_t \right) \tag{4}$$

### *D. SLAM Problem*

Then the SLAM problem consists of two parts, mapping and localization, respectively. The mapping is formulated as, given the robot trajectory $\boldsymbol{x}_{0:t}$ and a sequence of observations $\boldsymbol{z}_{0:t}$, build an occupancy grid map $\boldsymbol{m}$ of the environment which is represented by $p\left(\mathbf{m}|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right)$. The localization problem is formulated as $p\left(\mathbf{x}_{0:t}|\mathbf{z}_{0:t}, \mathbf{m}, \mathbf{u}_{0:t-1}\right)$. Overall, the SLAM problem is to obtain the $p\left(\mathbf{x}_{0:t}, \mathbf{m}|\mathbf{z}_{0:t}, \mathbf{u}_{0:t-1}\right)$, which is captured by

$$\begin{aligned} p\left(\mathbf{x}_{0:t}, \mathbf{m}|\mathbf{z}_{0:t}, \mathbf{u}_{0:t-1}\right) = p\left(\mathbf{x}_{0:t}|\mathbf{z}_{0:t}, \mathbf{u}_{0:t-1}\right) \\ \prod_i p(\mathbf{m}_i|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}) \end{aligned} \tag{5}$$

## III. Technical Approaches

In this section, the approaches to estimate the robot states $\mathbf{x}_{0:T}$ and the environment $\mathbf{m}$ are presented. In particular, the particle filter algorithm is used to localize the robot while the environment is obtained through Bayesian Inference.

### *A. Coordinate Transformation*

The robot operating in the 3-D space are related to several coordinate frames including the body frame, sensor frame and the world frame. In this project, the transformation between the sensor frame and the world frame is needed for localization and mapping, which is described by

$$x^{\mathcal{W}} = \begin{bmatrix} R & p \\ \mathbf{0} & 1 \end{bmatrix} x^{\mathcal{S}} \tag{6}$$

where $x^{\mathcal{W}}$ is the vector in world frame, $x^{\mathcal{W}}$ is the vector in sensor frame, $p$ is the vector of sensor in world frame, $R$ is the rotation matrix specified in Euler angles [8].

### *B. Localization*

In this subsection, the localization of the robot using Particle Filter is introduced. The main idea of the particle filter localization is to use a delta-mixture to represent the pdf of the robot state at time $t$, which is described by

$$p_{t|t}\left(\mathbf{x}_t\right) := p\left(\mathbf{x}_t|\mathbf{z}_{0:t}, \mathbf{u}_{0:t-1}\right) \approx \sum_{k=1}^{N} \alpha_{t|t}^{(k)} \delta\left(\mathbf{x}_t; \boldsymbol{\mu}_{t|t}^{(k)}\right) \tag{7}$$

where $\delta\left(\mathbf{x}; \mu^{(k)}\right)$ is the delta function with weights $\alpha^(k)$ which are described by the following equation

$$\delta\left(\mathbf{x}; \boldsymbol{\mu}^{(k)}\right) := \begin{cases} \infty & \mathbf{x} = \boldsymbol{\mu}^{(k)} \\ 0 & \text{else} \end{cases} \qquad \text{for } k = 1, \ldots, N \tag{8}$$

**Algorithm 1:** Stratified (low variance) resampling

**Input:** particle set $\left\{\boldsymbol{\mu}^{(k)}, \alpha^{(k)}\right\}_{k=1}^{N}$
**Output:** resampled particle set

$j \leftarrow 1, \ c \leftarrow \alpha^(1)$
**for** $i = 1 : N$ **do**
  $u \sim \mathcal{U}\left(0, \frac{1}{N}\right)$
  $\beta = u + \frac{k-1}{N}$
  **while** $\beta > c$ **do**
    $j = j + 1$**,** $c = c + \alpha^{(j)}$
  add $\left(\mu^{(j)}, \frac{1}{N}\right)$ to the new set

The Particle Filter localization consists of three steps, the resampling step, the prediction step and update step, which are described as follows

*1) Resampling:* In the resampling procedure, given particles $\left\{\mu_{t|t}^{(k)}, \alpha_{t|t}^{(k)}\right\}$, for $k = 1, ..., , N_{t|t}$, if $N_{eff} := \frac{1}{\sum_{k=1}^{N}\left(\alpha_{t|t}^{(k)}\right)^2} \leq N_{\text{threshold}}$, create a new set, $\left\{\bar{\mu}_{t|t}^{(k)}, \bar{\alpha}_{t|t}^{(k)}\right\}$ for $k = 1, ..., N_{t+1|t}$ using the stratified resampling method described as follows.

*2) Prediction:* In the prediction step, the state of the robot at the next time step is predicted based on the motion model, which is represtend by

$$\begin{aligned} p_{t+1|t}(\mathbf{x}) &= \int p_f\left(\mathbf{x}|\mathbf{s}, \mathbf{u}_t\right) \sum_{k=1}^{N_{t|t}} \alpha_{t|t}^{(k)} \delta\left(\mathbf{s}\boldsymbol{\mu}_{t|t}^{(k)}\right) d\mathbf{s} \\ &= \sum_{k=1}^{N_{t|t}} \alpha_{t|t}^{(k)} p_f\left(\mathbf{x}|\boldsymbol{\mu}_{t|t}^{(k)}, \mathbf{u}_t\right) \end{aligned} \tag{9}$$

Since the $p_{t+1|t}(\mathbf{x})$ is a mixture pdf with components $p_f\left(\mathbf{x}|\boldsymbol{\mu}_{t|t}^{(k)}, \mathbf{u}_t\right)$, it can be approximated by drawing samples from it by following the two procedures, the resampling procedure and prediction procedure. In particular, Then in the prediction procedure, the motion model is applied to each $\overline{\boldsymbol{\mu}}_{t|t}^{(k)}$ by drawing $\boldsymbol{\mu}_{t+1|t}^{(k)} \sim p_f\left(\cdot|\overline{\boldsymbol{\mu}}_{t|t}^{(k)}, u_t\right)$ and set $\alpha_{t+1|t}^{(k)} = \bar{\alpha}_{t|t}^{(k)}$.

*3) Update:* In the update step, the predicted state of the robot is updated using the Bayes rule with Lidar data which is described as follows,

$$\begin{aligned} p_{t+1|t+1}(\mathbf{x}) &= \frac{p_h\left(\mathbf{z}_{t+1}|\mathbf{x}\right) \sum_{k=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(k)} \delta\left(\mathbf{x}; \boldsymbol{\mu}_{t+1|t}^{(k)}\right)}{\int p_h\left(\mathbf{z}_{t+1}|\mathbf{s}\right) \sum_{j=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(j)} \delta\left(\mathbf{s}; \boldsymbol{\mu}_{t+1|t}^{(j)}\right) d\mathbf{s}} \\ &= \sum_{k=1}^{N_{t+1|t}} \left[ \frac{\alpha_{t+1|t}^{(k)} p_h\left(\mathbf{z}_{t+1}|\boldsymbol{\mu}_{t+1|t}^{(k)}\right)}{\sum_{j=1}^{N_{t+1|t}} \alpha_{t+1|t}^{(j)} p_h\left(\mathbf{z}_{t+1}|\boldsymbol{\mu}_{t+1|t}^{(j)}\right)} \right] \\ &\quad \delta\left(\mathbf{x}\boldsymbol{\mu}_{t+1|t}^{(k)}\right) \end{aligned} \tag{10}$$

The laser correlation model is used to obtain the $p_h\left(\mathbf{z}_{t+1}|\boldsymbol{\mu}_{t+1|t}^{(k)}, \mathbf{m}\right)$. In particular, the observation model is proportional to the similarity $\mathrm{corr}(\mathbf{y}, \mathbf{m})$ between the transformed scan $\boldsymbol{y}$ and the grid $\boldsymbol{m}$. The correlation is computed by

$$\mathrm{corr}(y, m) := \sum_i \nVdash \{m_i = y_i\} \tag{11}$$

where $\nVdash$ is the indicator function. Then the observation model is described as

$$p_h(\mathbf{z}|\mathbf{x}, \mathbf{m}) = \frac{e^{\mathrm{corr}(\mathbf{y},\mathbf{m})}}{\sum_{\mathbf{v}} e^{\mathrm{corr}(\mathbf{v},\mathbf{m})}} \propto e^{\mathrm{corr}(\mathbf{y},\mathbf{m})} \tag{12}$$

The localization algorithm is summarized as follows:

---

**Algorithm 2:** Particle Filter Localization

---

**Input:** Particle set $\left\{\boldsymbol{\mu}_t^{(k)}, \alpha_t^{(k)}\right\}_{k=1}^N$, Global frame odometry difference $\Delta o_t = o_{t+1} - o_t, \forall t \in \{0, 1, ...., T\}$, Current map $m_t$, Laser scan $z_t$, Head angle $s_t$

**Output:** Updated particles $\left\{\boldsymbol{\mu}_{t+1}^{(k)}, \alpha_{t+1}^{(k)}\right\}_{k=1}^N$

```
// Step 1: Resampling
```

**if** $N_{eff} := \frac{1}{\sum_{k=1}^N \left(\alpha_{t|t}^{(k)}\right)^2} \leq N_{threshold}$ **then**

  $\left\{\boldsymbol{\mu}_{t+1}^{(k)}, \alpha_{t+1}^{(k)}\right\}_{k=1}^N \leftarrow$ StratifiedResampling($\left\{\boldsymbol{\mu}_{t+1}^{(k)}, \alpha_{t+1}^{(k)}\right\}_{k=1}^N$)

```
// Step 2: Prediction
```

**for** $i = 1 : N$ **do**

  $\boldsymbol{\mu}_{t+1}^{(k)} = \boldsymbol{\mu}_t^{(k)} + \Delta o_t + \omega_t^N$

  $\omega_t^N \sim \mathbb{N}(0, W_{3\times 3})$

```
// Step 3: Update
```

**for** $i = 1 : N$ **do**

  $z_t^W$= transform2WorldFrame($\mu_{t+1}^{(i)}$,$z_t$,$s_t$)

  Remove points that hit the ground

  Find cells $y_t$ in the $m_t$ corresponding to the world frame scan $z_t^W$

  $l$=corr($y_t$,$m_t$)

  $\alpha_{t+1}^k$=$\alpha_t^k exp^l$

**for** $i = 1 : N$ **do**

  $\alpha_{t+1}^k$=$\frac{\alpha_{t+1}^k}{\sum_{j=1}^N \alpha_{t+1}^j}$

**Return** $\left\{\boldsymbol{\mu}_{t+1}^{(k)}, \alpha_{t+1}^{(k)}\right\}_{k=1}^N$

---

### C. Mapping

In occupancy grid mapping, the probability of an occupied cell is represented by

$$\gamma_{i,t} := p\left(m_i = 1|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right) \tag{13}$$

By using the Bayes rule, the probability of occupied cell is further transformed to

$$\begin{aligned}\gamma_{i,t} &= p\left(m_i = 1|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right) \\ &= \frac{1}{\eta_t} p_h\left(\mathbf{z}_t|m_i = 1, \mathbf{x}_t\right) p\left(m_i = 1|\mathbf{z}_{0:t-1}, \mathbf{x}_{0:t-1}\right) \\ &= \frac{1}{\eta_t} p_h\left(\mathbf{z}_t|m_i = 1, \mathbf{x}_t\right) \gamma_{i,t-1}\end{aligned} \tag{14}$$

Then the probability of the free cell is represented by

$$\begin{aligned}(1 - \gamma_{i,t}) &= p\left(m_i = -1|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right) \\ &= \frac{1}{\eta_t} p_h\left(\mathbf{z}_t|m_i = -1, \mathbf{x}_t\right)(1 - \gamma_{i,t-1})\end{aligned} \tag{15}$$

The odds ratio of a binary random variable $m_i$ updated over time is

$$\begin{aligned}o\left(m_i|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right) :&= \frac{p\left(m_i = 1|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right)}{p\left(m_i = -1|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right)} = \frac{\gamma_{i,t}}{1 - \gamma_{i,t}} \\ &= \frac{p_h\left(\mathbf{z}_t|m_i = 1, \mathbf{x}_t\right)}{p_h\left(\mathbf{z}_t|m_i = -1, \mathbf{x}_t\right)} \frac{\gamma_{i,t-1}}{1 - \gamma_{i,t-1}}\end{aligned} \tag{16}$$

Then estimating the pdf of $m_i$ conditioned on $z_{0:t}$ means accumulating the log-oods ratio, which is captured by

$$\begin{aligned}\lambda\left(m_i|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right) :&= \log o\left(m_i|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right) \\ &= \log\left(g_h\left(\mathbf{z}_t|m_i, \mathbf{x}_t\right) o\left(m_i|\mathbf{z}_{0:t-1}, \mathbf{x}_{0:t-1}\right)\right) \\ &= \lambda\left(m_i|\mathbf{z}_{0:t-1}, \mathbf{x}_{0:t-1}\right) + \log g_h\left(\mathbf{z}_t|m_i, \mathbf{x}_t\right) \\ &= \lambda\left(m_i\right) + \sum_{s=0}^t \log g_h\left(\mathbf{z}_s|m_i, \mathbf{x}_s\right)\end{aligned} \tag{17}$$

To obtain the $\log g_h\left(\mathbf{z}_t|m_i, \mathbf{x}_t\right)$, the Bayes rule can be used, in particular,

$$\begin{aligned}g_h\left(\mathbf{z}_t|m_i, \mathbf{x}_t\right) &= \frac{p_h\left(\mathbf{z}_t|m_i = 1, \mathbf{x}_t\right)}{p_h\left(\mathbf{z}_t|m_i = -1, \mathbf{x}_t\right)} \\ &= \frac{p\left(m_i = 1|\mathbf{z}_t, \mathbf{x}_t\right) p\left(m_i = -1\right)}{p\left(m_i = -1|\mathbf{z}_t, \mathbf{x}_t\right) p\left(m_i = 1\right)}\end{aligned} \tag{18}$$

where $\frac{p(m_i=1|\mathbf{z}_t,\mathbf{x}_t)}{p(m_i=-1|\mathbf{z}_t,\mathbf{x}_t)}$ specifies the accuracy of the observation.

The map pmf $\gamma_{i,t}$ can be recovered from the log-odds $\lambda_{i,t}$, which is described by

$$\gamma_{i,t} = p\left(m_i = 1|\mathbf{z}_{0:t}, \mathbf{x}_{0:t}\right) = 1 - \frac{1}{1 + exp(\lambda_{i,t})} \tag{19}$$

The mapping algorithm is then summarized as follows:

### D. SLAM

The SLAM is to perform the localization and mapping simultaneously. Based on the Particle filter localization algorithm and mapping algorithm. The particle filter SLAM algorithm is described as

## IV. Results

In this section, the proposed particle filter SLAM algorithm is implemented on the real data of THOR humanoid robot (see Figure 2 as an illustration).

**Algorithm 3:** Mapping

**Input:** Particle $\{\boldsymbol{\mu}_t, \alpha_t\}$, Current map $m_t$, Laser scan $z_t$, Head angle $s_t$

**Output:** Updated map $\boldsymbol{m}_t$

$z_t^W$= transform2WorldFrame($\mu_{t+1}^{(i)}$,$z_t$,$s_t$)

Find cells $y_t$ in the $m_t$ corresponding to the world frame scan $z_t^W$

Update the log-odds map using Equation 17 and 18

**Return** $m_t$

**Algorithm 4:** Particle Filter SLAM

**Input:** Laser scan $z_{0:T}$, Transformation matrix $T_l^w$ from Lidar frame to global frame, Global frame odometry difference $\Delta o_t = o_{t+1} - o_t, \forall t \in \{0, 1, ...., T\}$

**Output:** Robot state trajectories $\boldsymbol{x}_{0:T}$, Map $m$

```
// Step 1: Initialization
```

Initialize particles $\mu_0^{(k)} = [0, 0, 0]^T, \alpha_0^{(k)} = \frac{1}{N}, \forall k \in \{1, ..., N\}$ Initialize map $m_0$ Initialize state vector $\boldsymbol{x}$

**for** *t=0:T* **do**

```
// Step 2: Localization
```

$\left\{\boldsymbol{\mu}_{t+1}^{(k)}, \alpha_{t+1}^{(k)}\right\}_{k=1}^{N} \leftarrow$ $ParticleFilterLocalization(\left\{\boldsymbol{\mu}_t^{(k)}, \alpha_t^{(k)}\right\}_{k=1}^{N},$ $\Delta o_t = o_{t+1} - o_t, \forall t \in \{0, 1, ...., T\}$, $m_t$, $z_t$, $s_t)$

$l$= $\underset{k}{argmax}\ \ \alpha_{t+1}^{k}$

$\hat{\mu}_{t+1}$=$\mu_{t+1}^{l}$

$\boldsymbol{x}.append(\hat{\mu}_{t+1})$

```
// Step 2: Mapping
```

$\boldsymbol{m}_t \leftarrow Mapping(\hat{\mu}_t, \alpha_t, m_t, z_t, s_t)$

**Return** $m_T, \boldsymbol{x}$

### A. Parameter configurations and data descriptions

The parameters of the robot sensor configurations are described as follows. The center of the robot mass is kept at $0.93m$. The head is $0.33m$ above the center mass. The Lidar is $0.15m$ above the head. The data used in this project includes the Lidar sensor data, odemetry data and the head rotation angle data. Five different data sets are provided.

### B. Localization and mapping results

The final localization and mapping results on five different data sets are shown in Figures 3, 4, 5, 6, 7, respectively. As we can see from these figures, the gray area indicates the occupied environment while the white area is the free environment. The blue line is the robot trajectory. The results show that the proposed particle filter slam algorithm is promising in localizing the robot while mapping the environment. The trajectory of the robot and the corresponding mapping are reasonable and match with each other.

To further evaluate the performance of the proposed SLAM algorithm, the mapping over time of train data 0

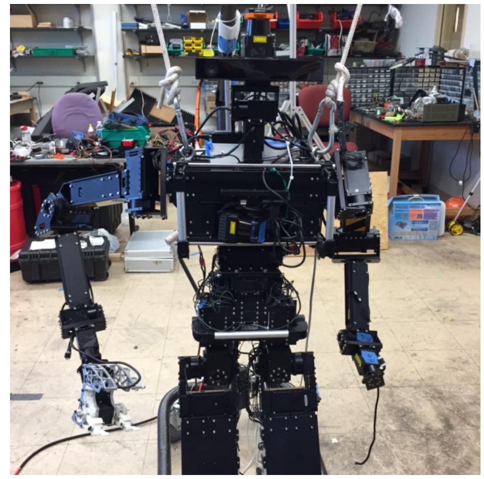


Figure 2. THOR humanoid robot

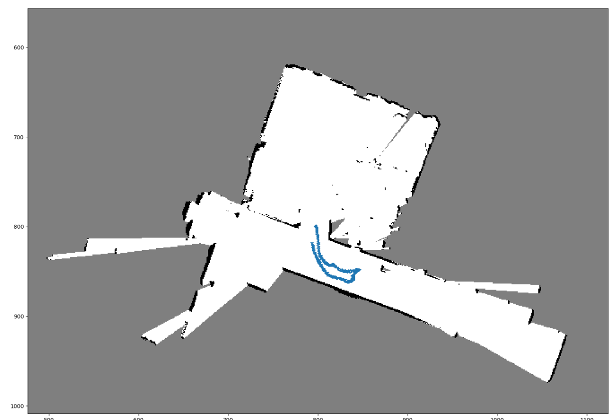



Figure 3. Mapping and localization on train data 0.

and train data 1 are shown in Figure 8 and 10 respectively. Moreover, the trajectory of the robot of train data 0 and train data 1 are shown in Figure 9 and 11. As we can see, the trajectories of robot are noisy, which are caused by the noise in observation model and the motion model. However, the pattern of the trajectory captures the movement of robot well. The corresponding mapping results also match with the robot trajectory.

## V. Conclusion

This project investigates the particle filter SLAM algorithm. In particular, the Bayes inference is used for occupancy grid mapping while the robot is localized using particle filter. The proposed algorithm is tested on real Lidar data on a humanoid robot. The results show that the algorithm is promising in localizing the robot and mapping the environment.

## References

[1] Y. Tian, H. Suwoyo, W. Wang, D. Mbemba, and L. Li, "An aekf-slam algorithm with recursive noise statistic based on mle and em," *Journal of Intelligent & Robotic Systems*, vol. 97, no. 2, pp. 339–355, 2020.

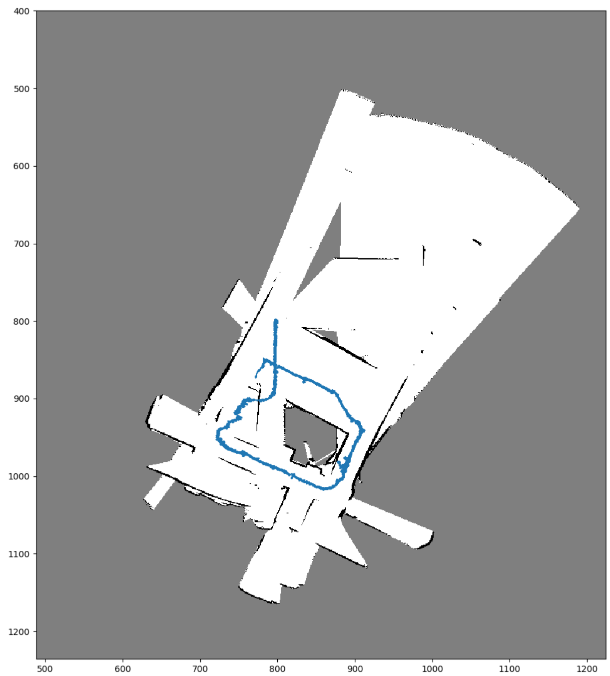


Figure 4. Mapping and localization on train data 1.

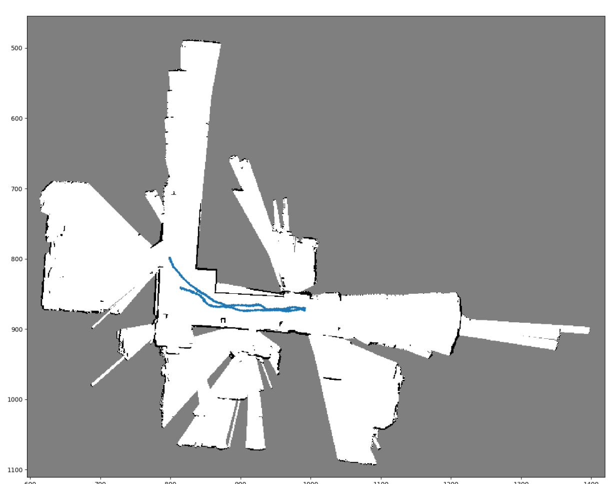


Figure 5. Mapping and localization on train data 2.


[2] J. Mullane, B.-N. Vo, M. D. Adams, and B.-T. Vo, "A random-finite-set approach to bayesian slam," *IEEE transactions on robotics*, vol. 27, no. 2, pp. 268–282, 2011.

[3] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," *Aaai/iaai*, vol. 593598, 2002.

[4] H. Roth and M. Vona, "Moving volume kinectfusion." in *BMVC*, vol. 20, no. 2, 2012, pp. 1–11.

[5] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[6] S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended kalman filter based slam," *IEEE Transactions on robotics*, vol. 23, no. 5, pp. 1036–1049, 2007.

[7] N. Carlevaris-Bianco, M. Kaess, and R. M. Eustice, "Generic node

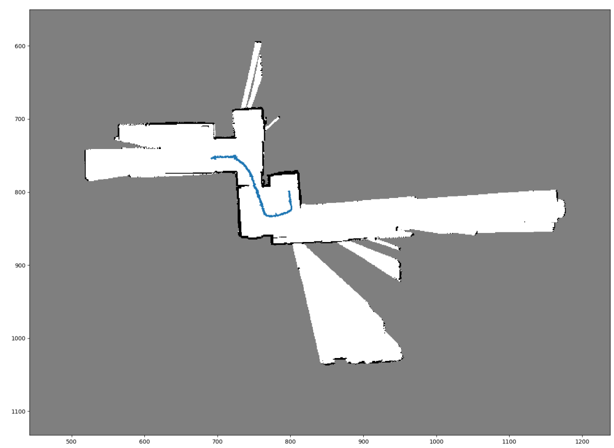



Figure 6. Mapping and localization on train data 3.

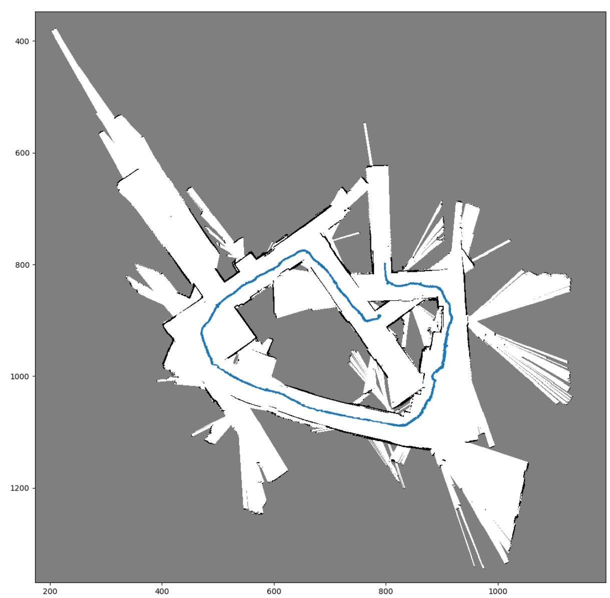


Figure 7. Mapping and localization on train data 4.


removal for factor-graph slam," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1371–1385, 2014.

[8] "Rotation matrix," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Rotation$_{matrix}$

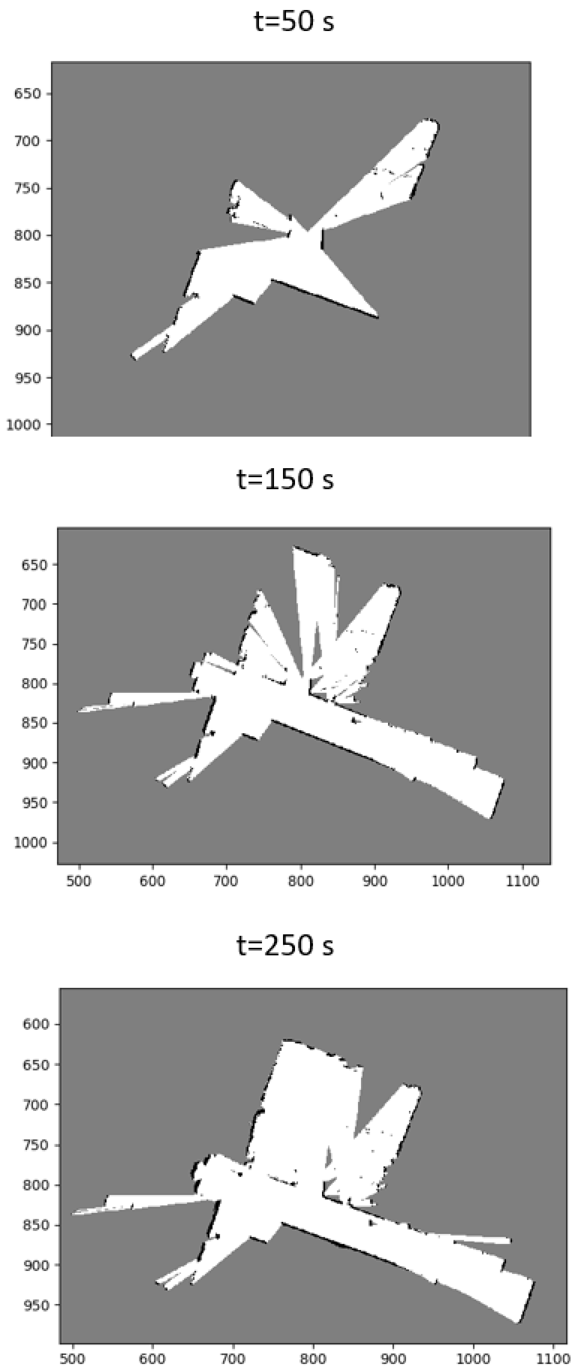
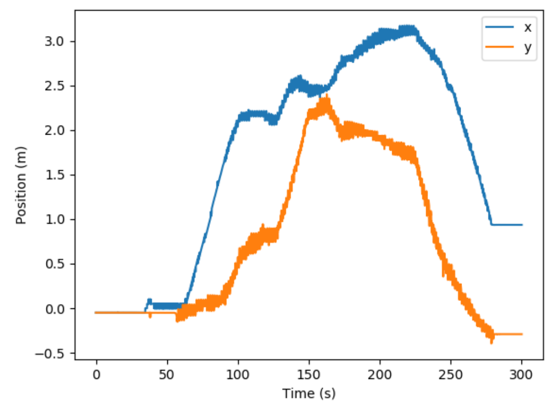
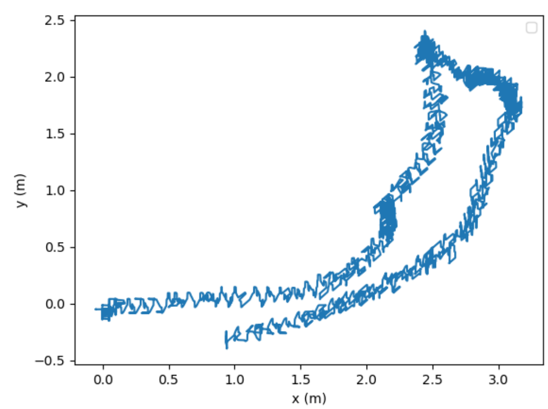



Figure 8. Mapping result over time of train data 0.




(a)




(b)

Figure 9. a) Robot trajectory over time of train data 0 and b) Robot trajectory in 2-D space of train data 0.

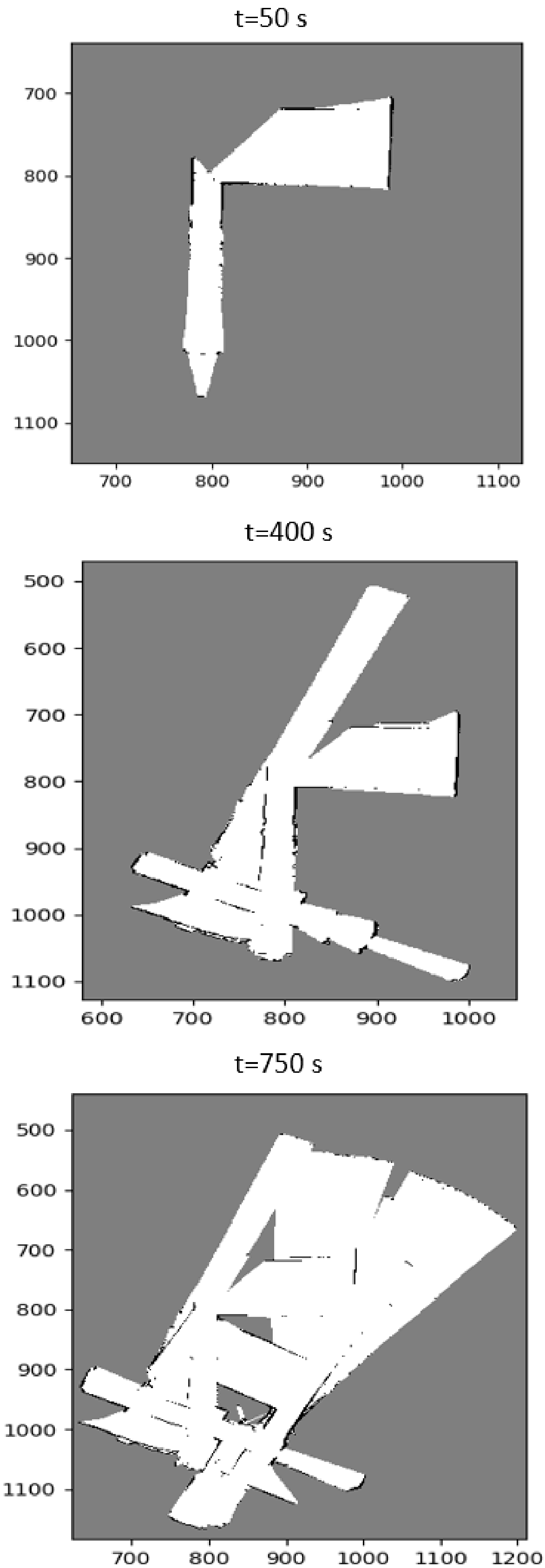
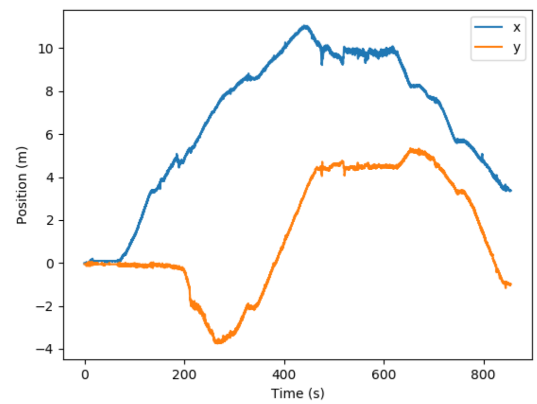
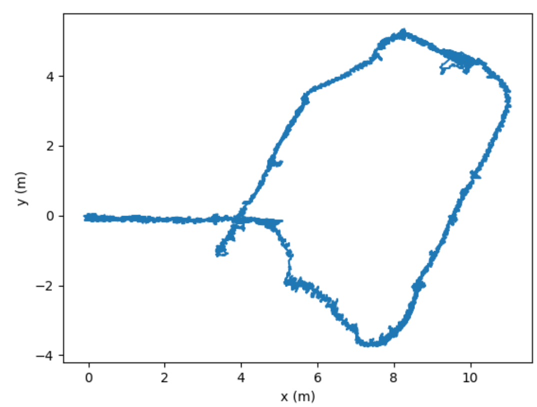



Figure 10. Mapping result over time of train data 1.




(a)




(b)

Figure 11. a) Robot trajectory over time of train data 1 and b) Robot trajectory in 2-D space of train data 1.