# Simultaneous Localization and Mapping: A Project Review

***Abstract*—This article reports a method for Simultaneous Localization and Mapping. To make a texture map, we used a particle filter model using odometry, inertial, 2-D laser range, and RGBD measurements from a two-minute activity of a differential-drive robot. As a result, we reconstructed the texture map and the trajectory which starts and ends in the same room.**

***Keywords—Simultaneous Localization and Mapping, Particle Filter, Texture Mapping***

## I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is a technique in robotic mapping and navigation for both constructing a map of an unknown environment and keeping track of a robot's location within the map at the same time. This technique is used in various applications such as a driverless car, autonomous robotic vacuum cleaners, and planetary rovers.

In this article, our goal is to make a texture map of a building using data of odometry, inertial measurement unit, 2-D laser range, and RGBD measurements from a two-minute activity of a differential-drive robot. The parameters are fitted for a training activity dataset and are tested for other activity datasets.

We used the particle filter algorithm which includes mapping steps, predicting steps and updating steps. The method deals with the uncertainty of a situation with a set of particles, each of which behaves on a stochastic process. To avoid particle depletion, in which most of the updated particle weights become close to zero because the finite set of particles are not accurate hypotheses, a resampling algorithm is included in the method.

This article reports the problem formulation, the technical approach, and the result of the project.

## II. PROBLEM FORMULATION

### A. Differential-drive Robot

The target agent is a differential-drive robot with encoder counters, an inertial sensor, a 2-D Lidar scanner, and an RGBD camera. Fig.1 shows the robot. The height, width, and depth of the robot are 514.35 mm, 476.25 mm, and 584.20 mm respectively. The wheel diameter is 254 mm, and the distance between the left and right wheels is 393.7 mm.

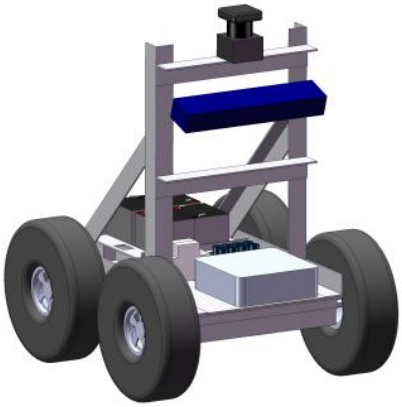

Fig.1. The target differential drive robot, equipped with encoders, IMU, 2-D LIDAR scanner, and an RGBD camera.

### B. Encoders

Each of the four encoder counters is set at each wheel. The frequency is 40 Hz, and there are 360 tics per revolution. Considering these and the robot hardware spec above, the meters per tic is

$$\frac{\pi * 254\ [mm]}{360\ [tic/revolution]} = 0.0022\ [m/tic]$$

Given encoder counts FR, FL, RR, RL corresponding to the four wheels, the right wheels travel a distance of

$$v_r dt = \frac{(FR + RR)}{2} * 0.0022\ [m/sec]$$

and the left wheels travel a distance of

$$v_l dt = \frac{(FL + RL)}{2} * 0.0022\ [m/sec].$$

The mean value of these is the distance the robot travels. Although the yaw rate can be also calculated from these theoretically, we use IMU to compute it since wheels are likely to slip when the robot rotates.

### C. Inertial Measurement Unit

Linear acceleration and angular velocity data are provided from an inertial measurement unit. The data is noisy because of high frequency vibrations of the robot. Therefore, the data is preprocessed with a moving average, which works as a low-pass filter. The window width of the moving average is 0.1 second.

### D. 2-D LIDAR

The robot has a horizontal LIDAR scanner, Hokuyo UTM-30LX, with 270-degree field of view and maximum range of 30m provides distances to obstacles in the environment. Each

LIDAR scan contains 1081 measured ranges. Since too near or too distant information often lower the performance of the algorithm, the range R is preprocessed with upper and lower thresholds as

$$R_{min} < R < R_{max}$$

where $R_{min} = 0.1\ m, R_{max} =$ 8 m.

### E. Kinect

An RGBD camera provides both RGB images and disparity images. The camera is located at (0.18, 0.005, 0.36)m with respect to the robot center and has orientation roll = 0 rad, pitch = 0.36 rad, and yaw 0.021 rad.

The depth camera and rgb camera are not in the same location. Given the values *d* at location the *(i, j)-th* pixel of the disparity image, the depth and associated color are obtained by the following mapping:

$$dd = (-0.00304 * d + 3.31)$$

$$\text{depth} = \frac{1.03}{\text{dd}}$$

$$\text{rgbi} = \frac{\text{i} * 526.37 + \text{dd} * (-4.5 * 1750.46) + 19276.0}{585.051}$$

$$\text{rgbj} = \frac{\text{j} * 526.37 + 16662.0}{585.051}$$

where (rgbi, rgbj) denotes the corresponding pixel on the RGB image.

A point (Xw, Yw, Zw) in the world coordinate associated with a point (u, v) in the RGB image is obtained by the following mapping:

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & P_{wc} \\ 0 & 1 \end{bmatrix} R_{oc}^T \begin{bmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} fs_u & fs_\theta & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \frac{1}{Z_o} [I_3 \quad 0] \begin{bmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{bmatrix}$$

where the intrinsic camera parameters are

*fsu=585.05108211,*

*fsv=585.05108211,*

*fsθ=0,*

*cu=242.94140713,*

*cv=315.83800193.*

RGB colors of the points whose Zw is less than 0.3 m are plotted on the texture map.

## III. Technical Approach

The method consists of four part: Mapping, Localization Prediction, Localization Update, and Texture Mapping. The method uses 20 particles.

### A. Mapping

In Mapping step, the lidar data is transformed to the body frame by the following function:

$$\begin{bmatrix} X_b \\ Y_b \end{bmatrix} = \text{f}(\text{z}_\text{t}, \theta) = \begin{bmatrix} z_t cos\theta \\ z_t sin\theta \end{bmatrix}$$

where $\text{z}_\text{t}$ and θ denote the range and angle. The points in the body frame are transformed to the world frame by the following mapping:

$$\begin{bmatrix} X_w \\ Y_w \end{bmatrix} = \text{R}_{\theta_i} \begin{bmatrix} X_b \\ Y_b \end{bmatrix} + \mu_\text{i}$$

where $\mu_\text{i}$ and $\theta_i$ represent the position and direction of the *i*-th particle. The (Xw, Yw) cells are recognized as occupied and the cells between $\mu_\text{i}$ and (Xw, Yw) are recognized as empty. The empty cells are computed with Bresenham's line algorithm.

The odds ratio of a cell $m_i$ is defined as

$$o(m_i|z_{0:t}, x_{0:t}) := \frac{\text{p}(m_i = 1|z_{0:t}, x_{0:t})}{\text{p}(m_i = 0|z_{0:t}, x_{0:t})}$$

$$= \frac{\text{p}_\text{h}(z_{0:t}|\ m_i = 1, x_t)}{\text{p}_\text{h}(z_{0:t}|m_i = 0, x_t)} o(m_i|z_{0:t-1}, x_{0:t-1})$$

Therefore the log-odds ratio is defined as

$$\lambda_{i,t} := \log o(m_i|z_{0:t}, x_{0:t})$$

$$= \lambda_{\text{i,t}-1} + \log \frac{\text{p}_\text{h}(z_{0:t}|\ m_i = 1, x_t)}{\text{p}_\text{h}(z_{0:t}|m_i = 0, x_t)}$$

Hence, we initialize the log-odds ratio with all zero and update them by adding a constant value if the cell is recognized as occupied and by subtracting the value if empty.

### B. Localization Prediction

The probability distribution of particle filter is transformed as:

$$P_{t|t}(x_t) = \sum_{k=1}^{N} \alpha_{t|t}^k \, \delta(x_t; \mu_{t|t}^k)$$

$$P_{t+1|t}(x) = \int P_f(x|s, u_t) \, P_{t|t}(s) ds$$

$$= \sum_{k=1}^{N} \alpha_{t+1|t}^k \, \delta(x; \mu_{t+1|t}^k)$$

The positions and directions of the particles are predicted by using the encoder counts and IMU information.

$$\mu_{\text{i,t+1}} = \mu_{i,t} + \frac{v_r + v_l}{2} dt \begin{bmatrix} cos\theta_{i,t} \\ sin\theta_{i,t} \end{bmatrix} + N_\mu$$

$$\theta_{\text{i,t+t}} = \theta_{i,t} + \omega_t dt + N_\theta$$

where $\omega_t$ represents the yaw rate and $N_\theta$ and $N_\mu$ denotes Gaussian random noise.

### C. Localization Update

The probability distribution of particle filter is updated as:

$$P_{t+1|t+1}(x) = \frac{P_h(z_{t+1}|x)P_{t+1|t}(x)}{\int P_h(z_{t+1}|s)P_{t+1|t}(s)\,ds}$$

$$= \sum_{k=1}^{N} \frac{\alpha_{t+1|t}^k P_h\left(z_{t+1}\middle|\mu_{t+1|t}^k\right)}{\Sigma_j \alpha_{t+1|t}^j P_h\left(z_{t+1}\middle|\mu_{t+1|t}^j\right)} \delta(x;\mu_{t+1|t}^k)$$

The map correlation is computed as follows:

$$corr(y,m) = \sum_i \mathbf{1}\{y_i = m_i\}$$

where $y_i$ and $m_i$ represent the lidar observation and current prediction for the *i*-th cell.

The weights of the particles are updated by multiplying the softmax of *corr*:

$$\alpha_{i,t+1} \propto \frac{e^{corr_i}}{\Sigma_j e^{corr_j}} \alpha_{i,t}$$

$$\sum_i \alpha_{i,t+1} = 1$$

To avoid particle depletion, in which most of the updated particle weights become close to zero because the finite set of particles are not accurate hypotheses, a resampling algorithm is included in the method if the effective number of particles are below a threshold.

$$N_{eff} = \frac{1}{\sum_i \alpha_{i,t}^2}$$

The resampling is done with the stratified resampling algorithm in Table.1.

Table.1 The stratified resampling algorithm

| Stratified (low variance) resampling |
|---|
| 1: **Input**: particle set $\{\mu^{(k)}, \alpha^{(k)}\}_{k=1}^N$ |
| 2: **Output**: resampled particle set |
| 3: $j \leftarrow 1$, $c \leftarrow \alpha^{(1)}$ |
| 4: **for** $k = 1, \ldots, N$ **do** |
| 5: $u \sim \mathcal{U}\left(0, \frac{1}{N}\right)$ |
| 6: $\beta = u + \frac{k-1}{N}$ |
| 7: **while** $\beta > c$ **do** |
| 8: $j = j + 1$, $c = c + \alpha^{(j)}$ |
| 9: add $\left(\mu^{(j)}, \frac{1}{N}\right)$ to the new set |

### D. Texture Mapping

Color segmentation is done by a single Gaussian model in RGB color space.

A point (Xw, Yw, Zw) in the world coordinate associated with a point (u, v) in the RGB image is obtained by the following mapping:

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_{wc} & P_{wc} \\ 0 & 1 \end{bmatrix} R_{oc}^T \begin{bmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} X_o \\ Y_o \\ Z_o \end{bmatrix} = Z_o \begin{bmatrix} fs_u & fs_\theta & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

where the depth information of the disparity image are used as $Z_o$.

RGB colors of the points whose Zw is less than 0.3 m are plotted on the texture map.

## IV. Results

### A. Map and Trajectory

The method was evaluated on three cases: Case A, B, and C. The parameters are fitted for Case A. The dataset for Case C is the one provided us as the test dataset. Fig.2, Fig.3 and Fig.4 show the result of the reconstructed maps and the trajectories on them in Case A, B, and C. In all cases, the reconstructed map represents the structure of the building and the robot starts and ends at the same point and our method succeeds in reconstructing the maps.

The GIF files are included in the submitted file.

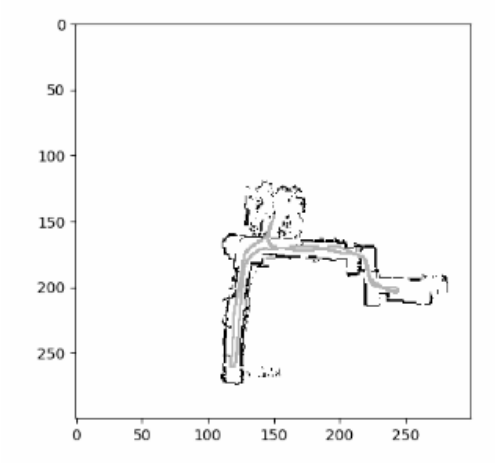


Fig.2. The reconstructed map and the trajectory in Case A.

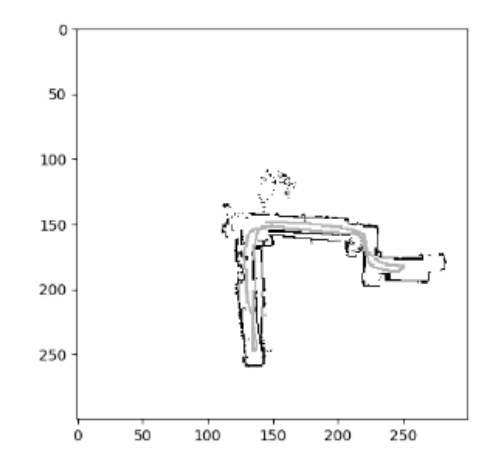


Fig.3. The reconstructed map and the trajectory in Case B.

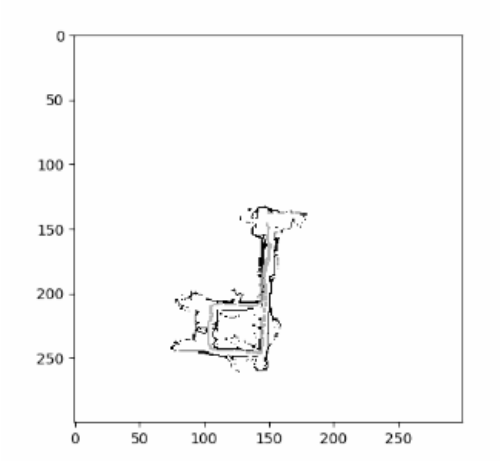


Fig.4. The reconstructed map and the trajectory in Case C.

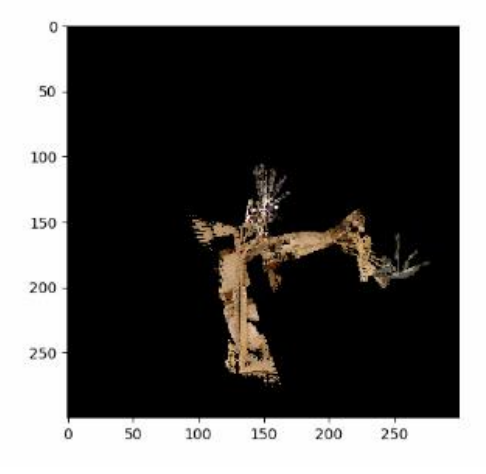


Fig.5. The reconstructed texture map in Case A.

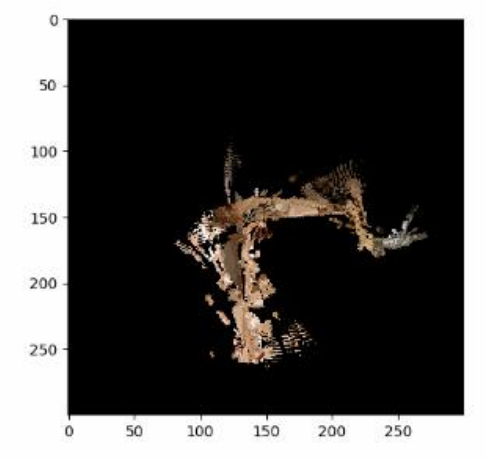


Fig.6. The reconstructed texture map in Case B.

### *B. Texture Map*

Fig.5 and Fig.6 show the reconstructed texture map. The method succeeds in reconstructing the shape of the map and find the corresponding colors of the floor, such as the yellow-brown floor, gray floor, and red floor. The texture mapping step in Case C is not evaluated due to the lack of its RGBD data.

The GIF files are included in the submitted file.

## V. DISCUSSION

The aim of this article was to assess the SLAM method for reconstructing a map, a trajectory, and a texture map. We used the particle filter algorithm which includes mapping steps, predicting steps and updating steps. The method uses a set of particles and includes resampling step to avoid particle depletion. With data of encoder counts, IMU, 2-D Lidar, and RGBD from a differential-drive robot, the method was evaluated for three cases.

This article clearly demonstrates that the method can reconstruct a map and a trajectory and make a texture map; the reconstructed map represents the structure of the building, the trajectory shows the robot starts and ends at the same place, and the texture map has the correct color features of the floor. For higher performance, the method can use more particles and optimize the parameters for the robot, the building, the target activities, and the intrinsic parameters.