Line Segment Extraction and Polyline Mapping for Mobile Robots in Indoor Structured Environments Using Range Sensors

Ankit A. Ravankar *, Abhijeet Ravankar ***, Takanori Emaru *, and Yukinori Kobayashi *

Abstract: Robot mapping and exploration tasks are crucial for many robotic applications and allow mobile robots to autonomously navigate in unknown environments. An accurate model of the environment is, therefore, essential for the robots to localize and perform navigation. In this paper, we present a line segment based mapping of indoor environments using range sensors for solving the simultaneous localization and mapping problem. The proposed method uses an extended Kalman filter. The extracted lines are merged to represent different structures in the environment correctly, and we show the results of our mapping method on simulated and real data sets. The experimental results demonstrate that the proposed method is capable of building an accurate line segment map of the environment for robot navigation.

Key Words: robot mapping, line segment extraction, noise modeling, simultaneous localization and mapping, extended Kalman filter.

1. Introduction

Exploration and mapping of mobile robots in unknown environments has several robotic applications such as in search and rescue operations, patrolling, warehouse picking, delivery robots, service robots in homes and public places, and most recently autonomous driving vehicles. For a truly autonomous behavior, mobile robot needs to perceive the environment using exteroceptive sensors such as range sensors and camera and use the sensor information to build the map of the environment. From the map information, the robot can estimate its position in the given environment or, in other words, localize itself in the map. The two problems of mapping and localization needs to be solved in tandem to each other, and one cannot be solved without the other. The problem is called as SLAM or simultaneous localization and mapping and is a very active area of research in robotics [1]. Numerous solutions to the SLAM problem had been proposed in the last 15 years, and it is an active area of research [2]. Some of the most popular methods use probabilistic filters such as particle filters or extended Kalman filters (EKF) to estimate the correct pose of the robot in the world while incrementally building the map of the surrounding [1]. The problem is difficult because the sensor data coming from the robot is corrupted by noise [3], and these errors which accumulate over time need to be corrected. The size and dynamics of the environment also play a crucial role in accurate mapping. It is, therefore, essential to model such uncertainties arising from the sensor data, and continuously update the state of the robot over time [4]. There are many different ways of modeling the environment. The most popular method uses an occupied grid map, where obstacles are modeled as occupied cells and free spaces as unoccupied cells in a probability grid [5]. Other methods include using cameras and depth sensors that can represent the environment as a 3D model [6]–[8].

A popular sensor commonly used for robot mapping is the sweeping 2D range sensors or light detection and ranging (LiDAR) that gives distance information to the obstacles as range and bearing. The map can either be represented as a point map or a feature map. The advantage of using features over point maps is the significant reduction in memory usage for storing the map. With respect to the size of the environment, storing all the features from the sensor might not be a good idea, and, therefore, many pieces of research have used feature-based maps for environment modeling [2]. Features extracted from raw range sensors, e.g., lines and planes, can simplify the geometry of the surrounding and provide easily readable maps. The line segment is the simplest of such features that can be used to represent the environment. All the raw data from the sensor can be encoded into poly-line features, and this can significantly reduce the computation time, as well as memory required to store features for a large area mapping. Line features for SLAM have been previously used in researches such as [9]–[12]. In particular, indoor scenarios can be modeled by line segments because most indoor environments are well structured. Entities such as walls, rooms, and edges can easily be represented into poly-lines and used for robot localization. However, one major problem with such environments is that it becomes difficult to extract features from the sensor data when the robot is operating in densely cluttered and noisy areas with many obstacles. Incorrect detection and wrong data association can then lead to imperfect maps and result in localization failures. In the present work, we have dealt with this problem efficiently wherein pre-processing of sensor data is performed to avoid any incorrect detection and association.
In this paper, we present a line segment based mapping algorithm for robots operating in indoor areas. Our line segment method uses a modified Hough transform algorithm, which is popular in computer vision for detecting lines and circles. The method is based on the probabilistic Hough transform, and we combine it with clustering to further speed up the detection rate. Uncertainties and noise errors from the line are modeled to accurately extract line segments from sensor data. Similar looking line features are merged to get the poly-line map. The extraction incorporates a noise model from the range sensor and is fused with the EKF-SLAM algorithm to handle the robot pose uncertainties. The algorithm is tested in simulated and real environments, and the results are verified through experiments. The rest of the paper is structured as follows. Section 2 describes the line segment extraction method. Section 3 shows line segment integration with EKF SLAM. Section 4 shows the experimental results and analysis. Lastly, Section 5 concludes the article.

2. Line Segment Extraction

Feature extraction from raw range sensor data is an important task for localization and exploration in unknown environments. Many researchers in the past have used different methods for line extraction from range data. A detailed literature review of different methods can be found in [13]. Some authors have used linear regression, random sample consensus (RANSAC), and expectation-maximization (EM) based methods, and others have used the Hough transform for line extraction from sonar and range sensor data. Many of the mentioned methods are directly used on the sensor data, and thus their computational cost is higher. To eliminate high computation cost from feature extraction, we do pre-processing on the raw sensor to remove noise and cluster data.

2.1 Pre-Processing

Before extracting the line segment, pre-processing of the raw sensor data is performed to eliminate any spurious points or noise from the data using methods as described in [14]. A laser scanner scans the environment in a counterclockwise direction and outputs $N$ points as

\[ p_k = (d_k \cos \phi_k, d_k \sin \phi_k), \quad k = 1, 2, \ldots, N, \quad (1) \]

where $d_k$ is the distance to the object, and $\phi_k$ is the scanning angle of the scan point with respect to the sensor’s $x$-axis (Fig. 1). Here, $\theta_{\text{min}}$ and $\theta_{\text{max}}$ are the range sensor’s minimum and maximum angular range and $X_G, Y_G$ is the global frame. The maximum and minimum range distance for the laser sensor are $d_{\text{max}}$ and $d_{\text{min}}$ respectively. Firstly, the scan data is clustered, and breakpoints are detected. These breakpoints represent discontinuities in the sensor measurement arising from open doors, corridors, and empty areas in the map. The breakpoint detection algorithm uses a distance threshold to find the breaks between the scan points and is given as

\[ \|p_{k+1} - p_k\| < D_{\text{th}}, \quad (2) \]

where $D_{th}$ is the distance threshold parameter. If the condition is satisfied, then the two points belong to the same cluster; else, the points ($p_{k+1}$ and $p_k$) are selected as breakpoints with $p_{k+1}$ as the initial point for the next cluster. The value of $D_{th}$ has been adopted from the study conducted in [11] and is given by

\[ D_{\text{th}} = d_k \frac{\sin(\Delta \phi)}{\sin(\lambda - \Delta \phi)} + 3\sigma_d. \quad (3) \]

where $\Delta \phi$ is the angular resolution of the laser scanner and $\sigma_d$ accounts for the sensor noise associated with $d_k$. The term $\lambda$ is chosen statistically and is provided by the sensor manufacturer. From different studies, the value of $\lambda = 10^\circ$ is found to perform well for indoor scenarios [10]. Once the clusters have been split using the distance breakpoint, the iterative endpoint fitting (IEPF) algorithm is used to further divide the detected clusters into several line segments [15]. If the clustering step generates $U$ clusters, $S^c = (S_j \mid j = 1, 2, \ldots, U)$, IEPF recursively splits the cluster $S_j$ into two subsets, $S^1_j$ and $S^2_j$. The algorithm is described as follows. For each cluster $S$ with $S^c$, an optimal line segment $L_s$ with its covariance matrix $P_s$ is found.

A number of algorithms can be adapted for line extraction from the range sensor, however, as discussed earlier, most of them are directly used without considering the sensor noise and uncertainties in measurement that results in poor performance. Our approach to incrementally learning the line segments in the environment adopts several key ideas from studies such as [16], [17] that considered weighted line fitting of range finder data with some uncertainty. We integrated the line fitting with localization uncertainty and used EKF for robot pose uncertainty modeling. In this work, we used the Hough transform algorithm for line segment detection. The Hough transform is a popular method in computer vision for detecting line segments from images [18]. These line features are detected in images by a voting process in a parameter space where object candidates are obtained as local maxima in the accumulator space and the lines.
are extracted against the maximum voted point. A modified version of the Hough transform (probabilistic Hough transform [PHT]) was used instead of the standard Hough transform. The PHT minimizes the amount of computation time needed to detect the lines by exploiting the difference in the fraction of votes needed to detect lines (segments) reliably with a different number of supporting points [19]. The standard Hough transform iterates over all the points and adds voting. This requires a considerable amount of computing power. For an input image with \( M \) pixels as edge points and a parameter space of \( N_r \times N_b \) accumulators, the computational complexity is \( O(MN_rN_b) \) for the voting space and \( O(N_rN_b) \) for the search space. This complexity and memory requirement grows further in higher dimensions.

On the other hand, the PHT algorithm uses random sampling of edge points. As \( m < M \), this drastically reduces the complexity of detection to \( O(mN_rN_b) \). A smaller value of \( m \) will lead to faster detection. By reducing the number of voting points in the accumulator space, PHT can work in real-time for line detection. Our method can detect line segments instead of lines in cluttered environments. However, PHT does not consider noise and uncertainty into account when detecting lines. This results in a higher than required number of lines detected from the sensor measurement. Our improvement and contribution to the line detection are that we do clustering in the accumulator space to determine the most optimum line from the maximum voted point, and slightly less than maximum voted points. The reason is that there are always points in the accumulator space which do not correspond to the local maxima; however their contribution to the line extraction cannot be ignored. The line is drawn against the mean point in the accumulator space after the clustering process. This ensures that only one line is detected and not many, which is not the case with SHT or PHT. Furthermore, to reduce the number of lines detected, lines generated from similar features, e.g., long walls, are merged together based on their overlapping criteria as shown in Fig. 3. Line segments \( A_iB_i \) and \( C_iD_i \), \( i = 1, 2, 3 \), are the line segments detected by the PHT algorithm. Based on the commonly shared points between the two line segments, they are merged into a single dominant line \( M_iN_i \). Note that in Fig. 3 case (d), no common dominant line is detected. The ‘gap’ criteria decides whether the lines should be merged or not. The value of ‘gap’ is chosen to be 10 cm in our case. In other words, if the gap between two line segments is greater than this value, the lines will not be merged and rather treated as individual lines (feature). After determining the dominant line, we can proceed with uncertainty modeling for line segment detection. The flow of the line segment extraction is shown in Fig. 4.

### 2.2 Sensor Noise Modeling

For a given data set \( R = \{r_i\} \) given by (1), measured distance \( d_k \) is computed by noise \( \epsilon_{d_k} \) as

\[
\hat{d}_k = d_k + \epsilon_{d_k}.
\]

Similarly, measurement angle \( \varphi_k \) is computed by noise \( \epsilon_{\varphi_k} \):

\[
\hat{\varphi}_k = \varphi_k + \epsilon_{\varphi_k}.
\]

Therefore, the point \( p_k \) in (1) can be written as

\[
\hat{p}_k = (d_k + \epsilon_{d_k}, \cos(\varphi_k + \epsilon_{\varphi_k})/\sin(\varphi_k + \epsilon_{\varphi_k})).
\]

Let us assume all noise terms are zero-mean Gaussian random variables. The corresponding covariances can be defined as \( \sigma_{d_k}^2 \) and \( \sigma_{\varphi_k}^2 \). Assuming \( \epsilon_{\varphi_k} \ll 1^\circ \), and, \( \epsilon_{d_k} \) and \( \epsilon_{\varphi_k} \) as independent, the covariance of point \( p_k \) can be defined as

\[
P_{p_k} = \begin{bmatrix}
\frac{1}{2} \left[ 2 \sin^2 \varphi_k - \sin 2 \varphi_k \right] + \\
\frac{1}{2} \left[ 2 \cos^2 \varphi_k \sin^2 \varphi_k \right]
\end{bmatrix}
\]

Further, assuming \( \sin \varphi_k \approx \varphi_k \), \( \cos \varphi_k \approx 1 \), and, \( \epsilon_{d_k}, \epsilon_{\varphi_k} \approx 0 \), we can use \( d_k \) and \( \varphi_k \) as good estimates for the quantities \( \hat{d}_k \) and \( \hat{\varphi}_k \), respectively.

### 2.3 Initial Line Segment Estimation

In polar format, the line can be defined as \( L \):

\[
L = \begin{bmatrix} \rho \ a \end{bmatrix}^T,
\]

and in the Cartesian coordinate system, the same line is given as

\[
x \cos a + y \sin a = \rho.
\]

Here, \( \rho \) is the normal distance to the line and \(-\pi \leq a \leq \pi\) is the orientation of the line with the sensor frame as shown in Fig. 5.

To determine the segment from the line, consider a line with the orientation \( a \) in the coordinate frame \( \rho_i - \psi_i \) that is obtained after rotating the origin to angle \( a \) as shown in Fig. 6. The endpoints of the line segment are scalar values given as \( \varphi_a, \psi_a \) in the coordinate frame \( \rho_i - \psi_i \). The line segment can be given as [16]

\[
S = \begin{bmatrix} \rho \ a \ \varphi_a \ \psi_a \end{bmatrix}^T.
\]

In order to get multiple endpoints, the above equation can be augmented to represent multiple endpoints on the same line given as
2.5 Weighted Line Fitting Method

In this work, we adopted the weighted line fitting model as described in [16], [17]. To approximate accurate line segment detection from the initial line using the PHT algorithms as discussed earlier, the line parameters are computed using the maximum likelihood formulation

\[
L(\rho, \alpha) = \sum_{k=1}^{n} \frac{[\delta_{pk}]^2}{P_{\rho_k}},
\]

(14)

where \(P_{\rho_k}\) is the covariance matrix of a measured point \(p_k\) given by (7) as

\[
P_{\rho_k} = \begin{bmatrix} P_{\rho} & P_{\rho\alpha} & P_{\rho\psi} \\ P_{\rho\alpha} & P_{\alpha} & P_{\alpha\psi} \\ P_{\rho\psi} & P_{\alpha\psi} & P_{\psi} \end{bmatrix},
\]

(23)

and \(P_{\rho}\), \(P_{\alpha}\), and \(P_{\psi}\) are the computed noise for orientation uncertainty \(\epsilon_{\alpha}\) and distance uncertainty \(\epsilon_{\rho}\) are calculated as

\[
\hat{\rho} = \frac{\sum_{k=1}^{n} \hat{\rho}_k}{\sum_{k=1}^{n} \rho_k},
\]

(17)

and \(\hat{\psi}\) is the measurement point position along the line (Fig. 7) and calculated as

\[
\hat{\psi}_k = \hat{\rho}_k \sin(\hat{\alpha} - \hat{\psi}_k).
\]

(18)

The line heading depends upon the angle \(\alpha\) and the reference frame in which the calculations are performed. Therefore, the center of rotation uncertainty \(\hat{\psi}_r\) with independent random variables \(\epsilon_{\psi}\) and distance uncertainty \(\epsilon_{\rho}\) are calculated as

\[
\hat{\psi}_r = \frac{\sum_{k=1}^{n} \hat{\psi}_k}{\sum_{k=1}^{n} \rho_k}.
\]

(16)

\[
\hat{\psi}_r = \frac{\sum_{k=1}^{n} \hat{\psi}_k}{\sum_{k=1}^{n} \rho_k}.
\]

(15)

The line position covariance are then computed as

\[
P_L = \begin{bmatrix} E[\epsilon_{\rho}^2] & E[\epsilon_{\rho}\epsilon_{\psi}] & E[\epsilon_{\psi}^2] \\ E[\epsilon_{\rho}\epsilon_{\psi}] & E[\epsilon_{\psi}^2] & E[\epsilon_{\rho}] \\ E[\epsilon_{\psi}^2] & E[\epsilon_{\rho}] & E[\epsilon_{\psi}] \end{bmatrix} = \begin{bmatrix} P_{\rho} & P_{\rho\psi} & P_{\psi} \\ P_{\rho\psi} & P_{\psi} & P_{\psi} \\ P_{\psi} & P_{\psi} & P_{\psi} \end{bmatrix},
\]

(24)

\[
\epsilon_L = [\epsilon_{\rho} \epsilon_{\psi}]^T.
\]

(25)
Here, $E$ is the expectation operator, $P_{\rho\rho}$ is the distance variance, $P_{\alpha\alpha}$ is the orientation variance, and $P_{\rho\alpha}$ and $P_{\alpha\rho}$ are corresponding variances and satisfy the symmetric positive definite property ($P_{\rho\rho} = P_{\alpha\alpha}$). The covariances are given as below:

$$P_{\rho\rho} = \frac{1}{\sum_{k=1}^{n} \rho_{k}^2},$$

(26)

$$P_{\alpha\alpha} = \frac{1}{\sum_{k=1}^{n} \alpha_{k}^2},$$

(27)

$$P_{\rho\alpha} = -P_{\alpha\rho} = \sum_{k=1}^{n} \frac{\delta\rho_{k}}{P_{\rho\rho}},$$

(28)

### 2.7 Line Merging

Similar line segments can be further sorted and merged to save computation after their estimation. For lines detected from different poses, it is important to transform lines and covariances to the respective reference frames. Consider lines $L_1$ and $L_2$ found in scans taken at two sensor poses $[0]$ and $[i]$ respectively (see Fig. 9). Then, we have

$$L_1 = \begin{bmatrix} p_0 & \alpha_0 \end{bmatrix}^T, \quad L_2 = \begin{bmatrix} p_1 & \alpha_1 \end{bmatrix}^T.$$  

(29)

Let $\hat{\gamma}_{0i}$ be an independent measurement of the robot’s pose $[i]$ with respect to pose $[0]$:

$$\hat{\gamma}_{0i} = \begin{bmatrix} x_0 & y_0 \end{bmatrix} = \begin{bmatrix} x_i & y_i \end{bmatrix}.$$  

(30)

To transfer the parameters of $L_2$ from pose $[0]$ to pose $[i]$ as a new line $L$, we calculate

$$L = \begin{bmatrix} \rho_i & \alpha_i \end{bmatrix}^T $$

(31)

$$= \begin{bmatrix} \rho_i + x_0 \cos(\alpha_i + \gamma_0) + \sin(\alpha_0 + \gamma_0) & \alpha_0 + \gamma_0 \end{bmatrix}^T.$$  

Consider the line $L_0 = [p_0 \ \alpha_0]$ with respect to the sensor’s local frame $[i]$ and represented as $r_i = [x_i \ y_i \ \theta_i]^T$. Here, $x_i$ and $y_i$ describe the position and $\theta_i$ describes the orientation of the robot with respect to the positive x-axis. The transform to the global reference frame ($r_0$) would be represented as

$$L_0 = \begin{bmatrix} p_0 & \alpha_0 \end{bmatrix}^T = \begin{bmatrix} \rho_i + \delta\rho_i & \alpha_i + \theta_i \end{bmatrix}^T,$$  

(32)

where

$$\delta\rho_i = x_i \cos(\alpha_i + \theta_i) + y_i \sin(\alpha_i + \theta_i).$$  

(33)

Figure 10 shows the line covariance transferred from local to global frame. The values of $\delta\rho$ and $\delta\theta$ represent the reference frame displacement in $(\rho\theta)$ frame. For the line $L_i$, the covariance matrix $P_{Li}$ at pose $r_i$ with the covariance matrix $P_{r_i}$, the transformed covariance matrix can be written as

$$P_{Li} = H_i P_{Li}^T + K_i P_{i} K_i^T,$$  

(34)

where the matrices $H_i$ and $K_i$ are given as

$$H_i = \begin{bmatrix} 1 & 0 \\ \delta\theta_i & 1 \end{bmatrix},$$  

(35)

and

$$K_i = \begin{bmatrix} 0 & \cos(\alpha_i + \theta_i) & \sin(\alpha_i + \theta_i) \\ \cos(\alpha_i + \theta_i) & 1 & 0 \end{bmatrix}.$$  

(36)

The value of $\delta\rho_i$ can be calculated from (33), and $\delta\theta_i$ is given by

$$\delta\theta_i = y_i \cos(\alpha_i + \theta_i) - x_i \sin(\alpha_i + \theta_i).$$  

(37)

Please note that the transformation matrix $H_i$ that transforms the line covariance matrix from the local to the global reference frame is independent of the reference pose. There always exists a value $H_P$ for which we can diagonalize the covariance matrix $P_L$ as

$$H_P = \begin{bmatrix} 1 & 0 \\ \psi_P & 1 \end{bmatrix},$$  

(38)

$$P_L = \begin{bmatrix} 1 & 0 \\ \psi_P & 1 \end{bmatrix} \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix} \begin{bmatrix} 1 & \psi_P \\ 0 & 1 \end{bmatrix},$$  

(39)

where $\psi_P$ is the center of rotation uncertainty with the estimate defined in (17) and is given by

$$\psi_P = -\frac{P_{\rho\theta}}{P_{\alpha\theta}}.$$  

(40)

Figure 11 shows the pose covariance $P_{\rho_i}$ transformed into the direction of the line and normal to $P_{\rho\rho}$. To confirm whether the given sets of lines are good to merge, a chi-squared ($\chi^2$) test is performed that states a hypothesis test whether the two line segments lie on the same line. After extracting the two line segments using the methods described earlier, the test is performed as follows:

$$\chi^2 = (\delta L)^T (P_{L1} + P_{L2})^{-1} (\delta L) < 3,$$  

(41)

where

$$\delta L = L_1 - L_2.$$  

(42)
From the chi-squared test (41), if the condition is satisfied and the lines are sufficiently similar to be merged, the lines $L_1$ and $L_2$ are merged using the maximum likelihood formulation and the coordinates $(L_{m}^i)$ and the uncertainty estimates $(P_{m}^i)_{ij}$ of the newly merged line are

$$L_m^i = P_{Lm}^i[(P_{L1}^i)^{-1}L_1^i]^{-1} + [(P_{L2}^i)^{-1}L_2^i]^{-1},$$

$$P_{m}^i = [(P_{L1}^i)^{-1} + (P_{L2}^i)^{-1}]^{-1}.$$

The endpoints are obtained by projecting the existing endpoints onto the newly merged line.

### 2.8 Odometry Noise Model

For the mobile robot equipped with the range sensor, the change in pose from $r_i$ to $r_j$ is measured as $r_{ij}$. The covariance matrix for this transform is given as

$$P_{rij} = \begin{bmatrix} P_{xx} & P_{xy} & P_{x\theta} \\ P_{yx} & P_{yy} & P_{y\theta} \\ P_{x\theta} & P_{y\theta} & P_{\theta\theta} \end{bmatrix}.$$  

(45)

Assuming smaller displacement for the transform $r_{ij}$ and noise measurement in $x$, $y$, and $\theta$ as independent random variables, the covariance can be stated as

$$P_{rij} = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_{\theta}^2 \end{bmatrix}. $$

(46)

Here $\sigma_x^2$, $\sigma_y^2$, and $\sigma_{\theta}^2$ represent the corresponding position variances in $x$, $y$, and $\theta$.

Let the initial covariance at position $r_i$ be $P_{ri}$ and for displacement $r_{ij}$ be $P_{rij}$. Then the combined covariance $P_{rij}$ in the global frame is written as

$$P_{rij} = GP_{ri}G^T + KP_{rij}K^T $$

(47)

with

$$G = \begin{bmatrix} 1 & 0 & -y_i \cos \theta - x_i \sin \theta \\ 0 & 1 & -y_i \sin \theta - x_i \cos \theta \\ 0 & 0 & 1 \end{bmatrix}, $$

(48)

and

$$K = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. $$

(49)

### 3. EKF Based Line Segment Detection

In the previous section, we have exhaustively modeled the line segment with range uncertainties that results in more robust line extraction considering the sensor noise. Using the methods described earlier and assuming smaller displacement of the sensor, the algorithm will extract and merge the lines to make a global line map. However, in real applications, the range sensor is fixed on a moving robot, and the challenge is to map the environment by taking into account the uncertainties from all robot poses. In this section, we show the integration of the line segment method with the extended Kalman filter (EKF) SLAM. The EKF is a non-linear state filter that uses a Bayes filter to linearize the estimate of the current mean and covariance. EKF-SLAM is considered to be the de-facto standard algorithm for solving the robot navigation problem. A major problem with the EKF algorithm is that, as the size of the map increases, or the number of stored features in the state vector increases, the consistency of the SLAM algorithm decreases with an increase in computation cost. Earlier researches on using line features with the EKF have been limited to small to medium-sized environment. In this work, by fusing the line segment uncertainty model with the robot’s motion model, our aim is to map large environments with good accuracy. Figure 12 describes the integration of the proposed line segment algorithm into the EKF-SLAM. The map is continuously updated until no more new features can be found in the environment. The following subsection will discuss the integration of our proposed line segment method with EKF-SLAM.

### 3.1 Map Representation

In the EKF SLAM algorithm, the map is a representation of the environment and stored as a state vector with the robot pose (or state) and the features (lines, points, corners, etc.) from the environment as a Gaussian variable. The EKF filter recursively predicts and corrects the robot’s state as it moves in the environment and observes features. Features are continuously matched to see if the observation was made earlier, and if not, new features are added to the state vector. The robot state vector at time $t$ with line features is represented as

$$\mu_t = [r^T \ L^1 \ L^2 \ \cdots \ L^n]^T, $$

(50)

where

$$r^T = [x_p^T \ y_p^T \ \theta_p^T]^T, $$

and $L_i^T = [p_{1i} \ \cdots \ p_{ni} \ a_{1i} \ \cdots \ a_{ni} \ \theta_{1i} \ \cdots \ \theta_{ni}]^T, i = 1, \ldots, n,$ are the line features extracted from the environment. The covariance matrix of the state vector is given as

$$Q = \begin{bmatrix} Q_{\mu} & Q_{\mu L} & \cdots & Q_{\mu \theta} \\ Q_{\mu \theta} & Q_{L} & \cdots & Q_{\mu \theta \theta} \\ \vdots & \vdots & \ddots & \vdots \\ Q_{\mu \theta} & Q_{\theta \theta} & \cdots & Q_{\theta \theta} \end{bmatrix}. $$

(51)

### 3.2 EKF Prediction

From the robot’s motion model, its displacement from time $t-1$ to $t$ can be calculated using the wheel encoder information. Let the right ($s_R$) and left wheel ($s_L$) displacements are given by $s_t$:

$$s_t = \begin{bmatrix} s_R \\ s_L \end{bmatrix}, $$

(52)

and the corresponding covariance matrix with this motion is given by

$$S_{st} = \begin{bmatrix} \sigma_{st}^2 & 0 \\ 0 & \sigma_{st}^2 \end{bmatrix}. $$

(53)
and the state vector and the covariance matrices are updated as
\[
\hat{\mu}_t = g(\hat{\mu}_{t-1}, x_t), \quad (54)
\]
\[
\hat{Q}_t = G_p Q_{t-1} G_p^T + G_w S_c G_w^T, \quad (55)
\]
where \(G_p = \frac{\partial g(\mu_{t-1}, x_t)}{\partial \mu_{t-1}}\) and \(G_w = \frac{\partial g(\mu_{t-1}, x_t)}{\partial x_t}\) are the Jacobian matrices.

### 3.3 Feature Association

As the robot moves, line features are extracted from the range sensor and added to the state vector. If the robot sees a previously explored feature, the EKF can do data association and correct the predicted state. After the line segments are detected from \(z'\) observations, the line feature is defined as \(L^j = [\rho_f, \alpha_f]^T\), and
\[
z_j = h(\hat{\mu}_t) = \begin{bmatrix} \hat{\mu}_{j}^T \\ \hat{\sigma}_{j}^T \end{bmatrix} = \begin{bmatrix} \rho_f^j - \hat{x}_p \cos \alpha_f^j - \hat{y}_p \sin \alpha_f^j \\ \alpha_f^j - \hat{\theta}_p \end{bmatrix}. \quad (56)
\]
The covariance matrix for \(\hat{z}_j\) is given by
\[
\hat{Q}_j = H_j^T \hat{Q}_j H_j^T, \quad (57)
\]
Here, \(H_j = \frac{\partial h(\hat{\mu}_t)}{\partial \mu}\) and \(\hat{Q}_j\) is the covariance of the predicted state vector. A similar chi-squared test based on artificial distance criteria is performed to check if the feature can be matched to that with a stored feature for data association:
\[
d_{ij} = (z_j - \hat{z}_i)^T (Q + \hat{Q}_j)^{-1} (z_j - \hat{z}_i) < \chi^2, \quad (58)
\]
\[
Q_j = \begin{bmatrix} \sigma_{\rho_i}^2 & \sigma_{\rho_i \alpha_i} \\ \sigma_{\rho_i \alpha_i} & \sigma_{\alpha_i}^2 \end{bmatrix}. \quad (59)
\]
Since, our proposed method of line segment extraction only detects a single line from the sensor data, the data association step in the EKF is improved as the matches can be made accurately. This also improves the loop closure detection when the robot comes back to a previously visited area, and the map is closed using the correct data association.

### 3.4 EKF Correction

The EKF performs the correction step for the associated feature \(z'\) (line feature) scanned at time \(t\) and updates the state vector and the covariance matrix. It is given as
\[
K_j = \hat{Q}_j H_j^T (H_j \hat{Q}_j H_j^T + Q_j)^{-1}, \quad (60)
\]
\[
\mu_t = \hat{\mu}_t + K_j (z_j - \hat{z}_j), \quad (61)
\]
\[
\hat{Q}_t = (I - K_j H_j) \hat{Q}_j. \quad (62)
\]
Here, \(K_j\) is called the Kalman gain, and it is determined by calculating the difference of uncertainties between the real and estimated observations.

### 3.5 Feature Addition

New features that do not get associated with previously observed features are added to the state vector continuously and are given as
\[
L^{n+1} = \begin{bmatrix} \rho_f^{n+1} \\ \alpha_f^{n+1} \end{bmatrix} = \begin{bmatrix} \rho_f^i + x_p \cos \alpha_f^{n+1} + y_p \sin \alpha_f^{n+1} \\ \theta_p + \alpha_f^i \end{bmatrix}. \quad (63)
\]
shows the result of the adaptive breakpoint detection segmentation of the range sensor data. The sensor data is clustered into several segments (shown in ellipses), and from these segments, the lines are extracted using the proposed method. The bottom figure shows our line segment extraction result. As can be seen, the proposed method can accurately estimate the lines from the range data.

4.1 Simulation Result

The simulation environment is as shown in Fig. 14. The environment is made of straight-line walls, and objects such as tables and shelves populate the environment. The size of the environment is approximately 16m × 18m. Simulation tests were conducted to test the performance of the proposed algorithm with different noise levels, which is usually not possible with a real sensor. The simulation also gives the flexibility of testing different parameters that are suitable for a similar environment in indoor or outdoor areas (e.g., very cluttered and densely populated or simple with straight walls). Also, ground truth data is used to test the SLAM trajectory accuracy. Initially, the robot was teleoperated inside the environment to ensure that all areas of the map are covered. After that, the data (laser scans and odometry) was fed into our line segment and SLAM algorithm. The corrected scan map of the environment is as shown on the right. All measurements are in meters. As can be seen, the SLAM algorithm correctly mapped all the areas. The line segment extraction and the poly-line map of the environment using our proposed method are shown in Fig. 15. A total of 67 lines were extracted during the mapping process, and they correctly matched the laser scan map data to represent the environment as a segment map. The corrected trajectory of the robot by EKF SLAM is also presented. The accuracy of the corrected trajectory is in the range of ±5 cm. It should be noted that elements in the map such as tables with legs are neglected when mapping with line segments because the laser clusters produced by such elements are small and can be safely ignored. However, when using the same map for navigation, such elements should be modeled as obstacles, or the robot will bump into obstacles. To solve this, the threshold for scan cluster that determines the minimum number of laser points required to make the line segment can be decreased to detect the line segment from smaller data clusters. This will, however, result in more number line segments extracted.

4.2 Real Environment

To test our algorithm, we did a similar experiment in a real environment with an actual robot. The environment is a long corridor in the Engineering Building of Hokkaido University and has several rooms on either side of the corridor. The total length of the corridor is around 70 m and the width of 10 m. Figure 16 shows the test environment and the robot (Pioneer-P3DX) that was used for the experiments. The environment is cluttered with obstacles and therefore produces a lot of sensor noise when collecting the data. Nevertheless, the proposed method can handle the noise levels in the sensor measurement and can correctly detect line segments from the laser scans. The robot was moved from one end of the corridor to the other end with several closing loops. Figure 17 shows the scan map produced by the SLAM algorithm after post-processing. A lot of noise can be seen around obstacles such as walls, chairs, and tables. These noises can be further processed and removed by the methods described in [14], [20]. Raw odometry (x and y) and orientation (θ) measurements from the robot are shown in Fig. 18. The line segment map of the same area is shown in Fig. 19. To keep the obstacle positions in the map, smaller segments were extracted around the static obstacles. Our algorithm could accurately model the environment with line segments, and the map can be used for navigation. The raw odometry from the robot and the corrected SLAM odometry is shown as well in Fig. 19. By using the EKF SLAM and the segment detection, the proposed method can correct significant drifts arising due to the robot’s wheel slippage. A feature of our proposed method is that redundant and duplicate lines are eliminated by using the weighted approach, and hence the data association...
between features is enhanced. This significantly reduces the wrong segments or duplicate feature detection when the robot is moving in areas that have many obstacles and improves the overall quality of the map. More features will lead to a very large covariance matrix in the EKF step, and this increases the computation cost of the algorithm as more and more features are added to the feature state vector. To improve this point, we performed another test in a similar area. Figure 20 shows the result of the test. The figure on the left is from the standard line segment EKF-SLAM without any noise modeling of the sensor data. On the right is the result of the proposed method with weighted line segment extraction and noise modeling of LiDAR data. From the result, it is evident that the proposed method generates far less number of line segments in the map as compared to the standard EKF-SLAM. The map is also more consistent with the environment because segments are fused to form a longer feature. The total time cost comparison of the two methods is presented in Fig. 21. As is evident, since there are lesser number of features to store, the time is significantly lower than the standard method. Using the proposed mapping method for medium to large areas, real-time performance can be achieved. Moreover, from Fig. 20, it can also be seen that in the case of the standard EKF-SLAM, the path of the robot crosses the feature (represented as a circle). The result of this will be that, during localization, free space will be modeled as an obstacle, and the robot will have to find an alternative path. On the other hand, in the proposed method, the line segment length threshold can be accurately set allowing narrow and smaller sections of the scene to be accurately reflected in the global map. The algorithm can also detect multiple loop closures. The EKF-SLAM position and uncertainty errors over time are shown in Fig. 22. EKF SLAM is able to correct the odometric drift errors and keep the positional uncertainties in check. The standard deviations in $x$ and $y$ are also shown in the figure.

5. Conclusion

In this paper, we have presented a mapping algorithm for robots operating in indoor environments. We presented a line segment extraction algorithm using a modified Hough transform and its integration with the EKF-SLAM algorithm. The line segment detection algorithm takes into consideration the sensor noise and is, therefore, more robust to mapping in cluttered environments. By reducing the number of points in the accumulator voting space of the Hough transform, the proposed method can extract single lines from range data. The extracted segments from the range sensor are merged with similar line segments and used for data association during the EKF step. Experiments in simulation and real environment prove that the proposed method can build good quality line maps in real environments that can be used for autonomous robot navigation.

References

[1] S. Thrun, W. Burgard, and D. Fox: Probabilistic robotics, *Intelligent Robotics and Autonomous Agents*, The MIT Press, 2005.
[2] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza,
J. Neira, I. Reid, and J.J. Leonard: Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age, *Trans. Robot.*, Vol. 32, No. 6, pp. 1309–1332, Dec. 2016.

[3] A.A. Ravankar, Y. Hoshino, T. Emaru, and Y. Kobayashi: Map building from laser range sensor information using mixed data clustering and singular value decomposition in noisy environment, 2011 IEEE/SICE International Symposium on System Integration (SII), pp. 1232–1238, 2011.

[4] Y.-T. Wang, C.-C. Peng, A.A. Ravankar, and A. Ravankar: A single LiDAR-based feature fusion indoor localization algorithm, *Sensors*, Vol. 18, No. 4, p. 1294, 2018.

[5] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit: FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges, *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI’03, pp. 1151–1156, 2003.

[6] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse: MonoSLAM: Real-time single camera SLAM, *IEEE Transactions on Pattern Analysis & Machine Intelligence*, Vol. 29, No. 6, p. 1052–1067, 2007.

[7] R. Mur-Artal, J.M.M. Montiel, and J.D. Tardos: ORB-SLAM: A versatile and accurate monocular SLAM system, *IEEE Transactions on Robotics*, Vol. 31, No. 5, pp. 1147–1163, 2015.

[8] A.A. Ravankar, A. Ravankar, T. Emaru, and Y. Kobayashi: A hybrid topological mapping and navigation method for large area robot mapping, 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), pp. 1104–1107, 2017.

[9] Y.-H. Choi, T.-K. Lee, and S.-Y. Oh: A line feature based SLAM with low grade range sensors using geometric constraints and active exploration for mobile robot, *Autonomous Robots*, Vol. 24, No. 1, pp. 13–27, 2008.

[10] S.-Y. An, J.-G. Kang, L.-K. Lee, and S.-Y. Oh: SLAM with salient line feature extraction in indoor environments, 2010 IEEE 11th International Conference on Control Automation Robotics & Vision, pp. 410–416, 2010.

[11] G.A. Borges and M.-J. Aldon: Line extraction in 2D range images for mobile robots, *J. Intell. Robot. Syst.*, Vol. 40, No. 3, pp. 267–297, 2004.

[12] M. Dakulovic, S. Iles, and I. Petrovic: Exploration and mapping of unknown polygonal environments based on uncertain range data, *Automatika*, Vol. 52, No. 2, pp. 118–131, 2011.

[13] V. Nguyen, A. Martinelli, N. Tomatis, and R. Siegwart: A comparison of line extraction algorithms using 2D laser rangefinder for indoor mobile robotics, 2005 IEEE/SICE International Conference on Intelligent Robots and Systems, pp. 1929–1934, 2005.

[14] A.A. Ravankar, Y. Hoshino, A. Ravankar, L. Jixin, T. Emaru, and Y. Kobayashi: Algorithms and a framework for indoor robot mapping in a noisy environment using clustering in spatial and Hough domains, *International Journal of Advanced Robotic Systems*, Vol. 12, No. 3, p. 27, 2015.

[15] R.O. Duda and P.E. Hart: *Pattern Classification and Scene Analysis*, John Willey & Sons, 1973.

[16] S.T. Pfister, S.I. Roumeliotis, and J.W. Burdick: Weighted line fitting algorithms for mobile robot map building and efficient data representation, 2003 *IEEE International Conference on Robotics and Automation*, Vol. 1, pp. 1304–1311, 2003.

[17] S.T. Pfister: Algorithms for mobile robot localization and mapping, incorporating detailed noise modeling and multi-scale feature extraction, Ph.D. dissertation, California Institute of Technology, 2006.

[18] A. Ravankar, A.A. Ravankar, Y. Hoshino, T. Emaru, and Y. Kobayashi: On a hopping-points SVD and Hough transform-based line detection algorithm for robot localization and mapping, *International Journal of Advanced Robotic Systems*, Vol. 13, No. 3, p. 98, 2016.

[19] N. Kiriya, Y. Eldar, and A.M. Bruckstein: A probabilistic Hough transform, *Pattern Recognition*, Vol. 24, No. 4, pp. 303–316, 1991.

[20] A. Ravankar, Y. Kobayashi, A. Ravankar, and T. Emaru: A connected component labeling algorithm for sparse LiDAR data segmentation, 2015 *IEEE 6th International Conference on Automation, Robotics and Applications (ICARA)*, pp. 437–442, 2015.