Hindawi
Journal of Robotics
Volume 2021, Article ID 9988916, 11 pages
https://doi.org/10.1155/2021/9988916

Research Article

A Novel Self-Positioning Based on Feature Map Creation and Laser Location Method for RBPF-SLAM

Yubao Shen[1] **and Zhipeng Jiao**[2]

[1]*Informationt Technology Center, Hefei Preschool Education College, Hefei 230013, China*
[2]*Shenzhen Tencent Computer System Co., Ltd, Shenzhen 518000, China*

Correspondence should be addressed to Yubao Shen; dbsod2012@126.com

Received 23 March 2021; Revised 26 October 2021; Accepted 18 November 2021; Published 20 December 2021

Academic Editor: Keigo Watanabe

Aiming at the high computational complexity of the traditional Rao-Blackwellized Particle Filtering (RBPF) method for simultaneous localization and Mapping (SLAM), an optimization method of RBPF-SLAM system is proposed, which is based on lidar and least square line segment feature extraction as well as raster, reliability mapping continuity. Validation test results show that less storage in constructing a map with this method is occupied, and the computational complexity is significantly reduced. The effect of noise data on feature data extraction results is effectively avoided. It also solves the problem of error accumulation caused by noninteger grid size movement of unmanned vehicle in time update stage based on Markov positioning scheme. The improved RBPF-SLAM method can enable the unmanned vehicle to construct raster map in real time, and the efficiency and accuracy of map construction are significantly improved.

1. Introduction

Nowadays, artificial intelligence technologies, such as unmanned driving, UAV, virtual reality, and augmented reality, have entered people's life, which cannot be without the development of localization and map building technology (SLAM). SLAM based on lidar has become a widely used technology in unmanned vehicle positioning solutions due to its advantages of accurate measurement, no need to preset the scene, integration of multiple sensors, working in poor light environment, and generation of environment map for easy navigation. The technical basis of Lidar SLAM is the key to solving SLAM process, which mainly focuses on environmental feature extraction, data association, map creation, autonomous positioning, path planning, and other major problems.

The geometric representation of feature map in SLAM extracts more intuitive geometric features from the sensor's perception information of the environment size and uses these types of geometric information to describe the environment. The speed of feature map recognition determines the efficiency of unmanned vehicle positioning and map construction, which is the core problem of unmanned vehicle navigation and the basis of solving the movement control of unmanned vehicle. Currently, the widely used linear segment segmentation and extraction algorithms of laser scanning data include S&M (Split-and-Merge) algorithm, RANSAC (Random Sample Consensus) algorithm, LR (Line Regression) algorithm, IEPF (Iterative Endpoint Fit) algorithm, and EM (Expectation Maximization) algorithm, all of which are compared in detail in literature [1]. IEPF algorithm based on Hough transform is a common and efficient recursive algorithm to extract the point set of laser SLAM into line segment features. In image processing, in view of the low efficiency of Hough transform, literature [2] proposed a fast Hough transform method to compress parameter space. Einsele et al. [3] extracted line segment features by identifying matching units. Vandorpe et al. [4] studied a method of extracting line and angle features with lidar for dynamic map building. Jensefelt et al. [5] introduced a method that uses lidar to extract linear features for robot pose tracking in indoor environment. However, IEPF

algorithm has two disadvantages. First, it is difficult to determine the threshold value of line segment segmentation. If the threshold value is too large, the problem of undersegmentation will occur; if the threshold value is too small, the problem of oversegmentation will occur. Secondly, IEPF is sensitive to noise. The disadvantages of Hough transform lie in the large amount of calculation and are easy to extract similar features and the feature parameters of extracted line segment are not precise enough. In this paper, a line segment feature optimization extraction method based on Hough transform is proposed, which solves the problem that the extracted line segment feature parameters are not precise enough and provides a basis for SLAM map creation.

Lidar SLAM map creation refers to the process of building an environment map based on the pose and environmental information data continuously collected by the sensor carried by the robot itself during the moving process. Commonly used map representation models include feature map, raster map, topology map, and mixed map. Literature [6] elaborates the advantages and disadvantages of various maps. Common algorithms for lidar SLAM map creation are mainly based on filtering (EKF based map creation algorithm, RBPF based map creation algorithm, and FastSLAM based map creation algorithm) and graph optimization defense (PTAM-SLAM, SD-SLAM, ORB-SLAM, SVO-SLAM, RGBD-SLAM, etc.). The main disadvantage of PF-SLAM is that the computation amount increases exponentially with the increase of the number of landmarks, which makes it difficult for particle filter to be applied in SLAM. In order to reduce the computational complexity of PF-SLAM, Murphy et al. [7] proposed RPF-SLAM. The idea of RPF-SLAM is to decompose SLAM into two relatively independent problems: localization problem and pose-based map estimation problem. Pose estimation uses particle filtering, while pose-based map estimation uses EKF. This method combines the advantages of PF and EKF, reduces the computational complexity, and has a good effect on data association. On this basis, Monte Merlo et al. [8] proposed the FAST-Slam1.0 algorithm based on RBPF. Fast-Slam1.0 algorithm adopts the process model of SLAM as the importance function of particle sampling, so serious particle degradation phenomenon occurs in the algorithm during operation, resulting in large error between map construction and environment. Later, they proposed the FastSLAM2.0 algorithm for this problem [9], which constructed a Gaussian distribution function to replace the PROCESS model of SLAM as the importance function of particle sampling, alleviating the phenomenon of particle degradation, possibly preserving the historical pose of the unmanned vehicle, and reducing the error of positioning and mapping. Meanwhile, the convergence of the algorithm is proved theoretically for the first time. Literature [10] proposed a laser SLAM algorithm GMapping based on RBPF-SLAM, which added high-precision laser measurement data into the proposed distribution, making the proposed distribution closer to the actual posterior distribution and further reducing the error of positioning and mapping.

Lidar SLAM localization algorithms mainly include scanning matching method, extended Kalman filter (EKF) localization, Markov localization, and particle filtering localization. Markov positioning method [11] regards the positioning problem of unmanned vehicles as a discrete Markov process, and the position and pose of unmanned vehicles at every moment is a state of this process, which transfers between multiple states with the movement of unmanned vehicles. Hu Yuwen et al. [12] proposed a window-constraining Markov positioning method to reduce the updates through Markov positioning within the window scope. Li Yu [13] proposed a pose estimation algorithm based on point cloud matching, which adopted the classical ICP point cloud matching algorithm to scatter points in gaussian distribution near the rough estimation to realize particle updating and obtain the real pose of unmanned vehicles. This method is based on the classical RBPF-SLAM and improves the reliability and efficiency of the algorithm.

In this paper, a line segment feature description map is adopted to improve the steps of the traditional RPF-SLAM algorithm, and the line segment feature map is created based on the improved steps of the RPF-SLAM algorithm. In the particle sampling step, the line segment features are used for pose estimation to reduce the sampling area. In the weight updating step, the weight updating method based on similarity comparison is adopted to reduce the computational complexity. An observation update model based on similarity comparison is proposed, as well as a Markov localization method which can be located in a created line segment feature map. Analysis through simulation experiment shows that the method can effectively locate unmanned vehicles in feature map.

2. Creating Method of Line Feature Map Based on RBPF

Considering that there are a lot of line features in the environment where driverless vehicles are located, line segments are used to describe the edge features of buildings in the environment. This paper mainly solves two problems: line segment feature extraction and line segment feature map creation.

The key issue of line segment feature extraction is how to extract feature the point cloud data obtained by lidar at a certain time, extract all line segment features, and express them with parameters.

The key issue of creating line segment feature map is how to use a series of motion control information and sensor observation information of unmanned vehicle in the process of environmental motion to create environment line segment feature map.

2.1. Line Feature Extraction Algorithm Based on Hough Transform. In order to improve the fitting accuracy of line segment features and make the fitted line segments to describe the environment concisely and accurately, a line feature fitting method is proposed based on probability Hough transform fitting, voting box depth first search and merging similar features, and least square fitting.

The basic steps of the Algorithm 1 are presented as follows:

For the straight line shown in Figure 1, where the distance from the origin to the straight line is c, and the angle between the vertical line of the line and the positive direction of the axis is θ; then the line can be expressed by the following expression:

$$x \cos \theta + y \sin \theta = c. \quad (1)$$

In formula (1), if θ and c are taken as constant, and x and y as variable, the line can be represented by a point (θ, c). In the same way, all lines in the coordinate system can be represented by such a point, and all points (x, y) satisfying formula (2) can be approximately considered as in the characteristics of the line represented by the point (θ, c); δ is a small constant:

$$|x \cos \theta + y \sin \theta - c| < \delta. \quad (2)$$

Because of the limited measurement range of lidar, the distance c from the origin to all line features is also limited, so all possible linear features can be represented by a point in the region $\theta \in (0, \pi)$, $c \in [c_{\min}, c_{\max}]$. In the Hough transformation, the region is divided into small grids, each of which is called a ballot box and represents a straight line with the center point (θ, c) of the ballot box. When a point (x, y) in point cloud data is satisfied by formula (2), it is considered that the point is on the line feature represented by the ballot box, and the number of votes in the ballot box is increased by one. When all points vote, the number of votes in all ballot boxes is counted. If the number of votes in a certain ballot box is greater than a certain threshold, a large number of points are approximately on the line represented by the ballot box, so that the line features can be extracted. Due to the noise or the actual situation of the environment and other factors, the point cloud data is not accurate on the same line, so it may appear that a line feature is detected into multiple approximate line features. As a result, the similar line features need to be merged.

As shown in Figure 2, each small grid is a voting box with the size of $\Delta\theta \times \Delta c$. Each voting box represents a straight line. When any point cloud data converted to Cartesian coordinate system satisfies formula (2), the point will vote for the voting box. The darker grids in the figure denote the ballot boxes with more than average votes. Because of the error of the rangefinder and the feature in the environment is not necessarily accurate line, the point cloud data roughly on one line can fit several approximate line features. As shown in Figure 2, 12 approximate straight lines are fitted out in the dark area on the left side of Figure 2, and the parameters (θ, c) of these lines are similar. In this paper, we use the depth first search voting box method to sum up these point cloud data.

After the combination of Hough transform and similar lines based on depth first search, the main existing line features and the points through which these line features pass are determined, and the accurate parameters can be calculated by using the least square method.

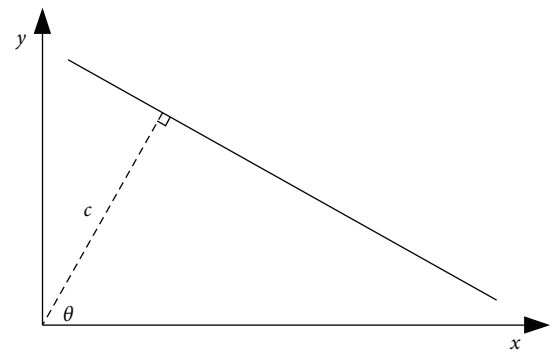


Figure 1: Line parameter.

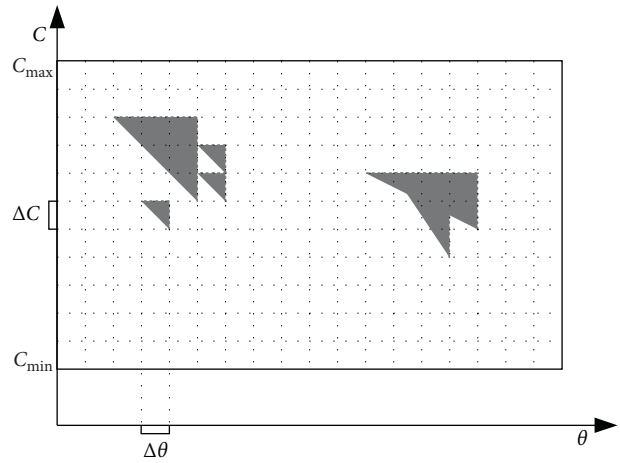


Figure 2: Hough transform ballot box.

2.2. Method of Creating Line Segment Feature Map Based on RBPF. The main reason for the excessive number of particles required by RBPF-SLAM is that the estimation of the position and attitude of unmanned vehicles in step 3 is not accurate enough. In this paper, the sampling link is improved. In order to make RBPF-SLAM adapt to the creation of line segment feature map, step 2 is added, and corresponding adjustment is made in step 6. In step 2, local line segment feature map is created by using the line segment feature extraction method described in the previous section. Finally, the new Algorithm 2 is divided into six steps.

During the sampling phase, the position and posture of the unmanned vehicle at the k moment is $X_k = (x_k, y_k, \varphi_k)^T$; its updating method is divided into two stages: prediction and correction. During the prediction stage, the estimated pose of the current moment $\hat{X}_k$ is estimated by using the position and attitude of the previous moment X_{k-1}, the odometer data of the previous moment O_{k-1}, and the odometer reading of the current moment O_k. During the calibration stage, the feature subgraphs of all terrain m_r that may be detected by the sensor around the global map are extracted from the constructed global map m_{k-1}, and the observation data of the sensor z_k are matched to obtain the pose of the unmanned vehicle X_k.

2.2.1. Posture Prediction. Assume that the difference between the time $k-1$ and the time k of the unmanned vehicle odometer is $o_k = O_k - O_{k-1}$, and it has the same dimension

as the position and attitude of the unmanned vehicle $X_{k-1} = (x_{k-1}, y_k - 1, \varphi_{k-1})^T$ at the moment $k-1$. The parameter o_k represents the displacement of the three dimensions of the unmanned vehicle, namely, the vector in the coordinate system of the last moment, the origin of the coordinate system is x, y, z, and the positive direction of the axis is $(x_{k-1}, y_{k-1})^T$. If the matrix can convert any point to the world coordinate system, then the displacement of the unmanned vehicle in the world coordinate system from time to time is the estimated position and posture of the unmanned vehicle according to the odometer data $\widehat{X}_k = X_{k-1} + M^* o_k$.

2.2.2. Posture Correction. After the estimated position $\widehat{X}_k$ is calculated, the environmental feature subgraphs m_r that may be observed by lidar sensor in this position and attitude are extracted from the constructed global map m_{k-1}. Specific practices are as follows: setting $\widehat{X}_k = (\widehat{x}_k, \widehat{y}_k, \widehat{\varphi}_k)^T$, the length of the laser radiation is set to be two times of the starting point $(\widehat{x}_k, \widehat{y}_k)^T$ of the laser range. When a ray intersects with a line segment feature, the line segment feature is added to the local feature subgraphs m_r. When a ray intersects with multiple line segment features, the intersection point $(\widehat{x}_k, \widehat{y}_k)^T$ and the nearest line segment feature are added to the local feature subgraph m_r. The above process is used to simulate the lidar scanning process, and the feature subgraph m_r is composed of all line segment features that may be observed by an unmanned vehicles at the estimated position. The unmanned vehicle position and surrounding environment are shown in Figure 3; X_k presents the real position and attitude of the unmanned vehicle, and its orientation is shown by the arrow in the figure. The semicircle with the center X_k is the observation field of the unmanned vehicle lidar; $\widehat{X}_k$ represents the estimated position and posture of the unmanned vehicle, which are considered as the range that can be observed by the unmanned vehicle lidar; through the method described in this paragraph, the local feature submap m_r at this time is obtained as shown in Figure 4.

After getting the feature subgraph m_r under the estimated position, the subimages m_k' extracted from the observation data Z_k are matched in the local feature subgraph m_k' to obtain the unmanned vehicle position and attitude.

The matching process is divided into two steps. Firstly, the orientation of the unmanned vehicle is calculated, and then the position of the unmanned vehicle is calculated. For a certain line segment f_p in m_k', if the orientation of unmanned vehicle is φ_k, the θ parameters of corresponding line segment f_p is m_r: f_p: $\theta + \varphi_k - 90$. The essence of calculating the orientation of unmanned vehicle is to find a φ_k line segment so that any line segment f_p can be found in m_r to make f_p: $\theta = f_p$: $\theta + \varphi_k - 90$. Set a function (x, y), in which x is the feature map and y is the angle. If the parameter θ of a certain line segment f_0: $\theta = [c_0, \theta_0, l_0, P_0,]$ is closest to the value y, then formula (3) $F(x, y) = (\theta_0 - y)^2$ denotes the degree of similarity with the angle of the middle line when the unmanned vehicle is facing φ_k:

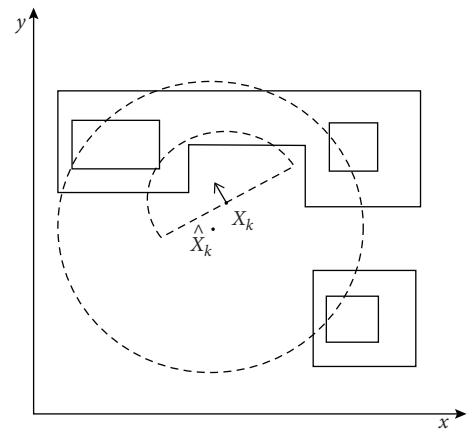


Figure 3: Map of the position and surrounding characteristics of unmanned vehicles.

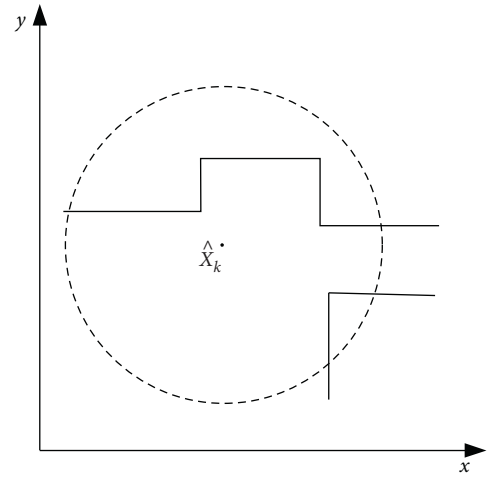


Figure 4: Local feature subgraph.

$$\sum_{f \in m_k} F(m_k, f\colon \theta + \varphi_k - 90). \tag{3}$$

Under the condition of no noise, if φ_k is the accurate orientation of the unmanned vehicle, the formula is equal to 0. For a certain line segment f_p in m_k', there should be a certain line segment f_p in m_r. When the unmanned vehicle is facing φ_k, the condition f_p: $\theta + \varphi_k - 90 = f_q$ is satisfied. Therefore, all possible values of φ_k are not continuous but discrete, and $\varphi_k \in S = \{f_q\colon \theta + 90 - f_q\colon \theta, f_q\colon \in m_r\}$; that is, the number of possible values of φ_k is equal to the number of middle line segments in m_r. After determining the value range of φ_k, as long as the minimum value φ_k of formula (3) is found in the set S, it can be used as the true orientation of unmanned vehicle.

After φ_k the orientation of the unmanned vehicle is determined, the position of the unmanned vehicle $(x_k, y_k)^T$ is calculated. Assuming that a certain line segment f_p in m_k' is the result of a certain line segment f_p in m_r observed by the unmanned vehicle in the real posture, the points on the upper part will fall on the ground after rotating according to the actual orientation of the unmanned vehicle and then

translating according to the actual position. After calculating the orientation of the unmanned vehicle in the previous step, the corresponding relationship f_q to f_p can be obtained, that is, which line segment in m_r should be closest to *each point in* z_k. Setting the real pose of the unmanned vehicle $X_k = (x_{k,} y_{k,} \varphi_{k,})^T$, φ_k is known; then, any point P_p in z_k will be changed into: rotating according to the actual orientation of the unmanned vehicle φ_k and then translating P_p according to the actual position $(x_k, y_k)^T$:

$$\begin{bmatrix} \cos(\varphi_k - 90), -\sin(\varphi_k - 90) \\ \sin(\varphi_k - 90), -\cos(\varphi_k - 90) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_k \\ y_k \end{bmatrix}. \tag{4}$$

Furthermore, the point will fall on top of f_q. To accurately calculate the real position of the unmanned vehicle $(x_k, y_k)^T$, the least square method is used to calculate the position $x_k\ y_k$, so that the distance between all points in z_k and the corresponding line segment in m_r is the minimum, even if the following formula is the smallest:

$$\sum_{p_p \in z_k} \left(p_q * \cos\left(f_q\colon \theta + p_q * \sin\left(f_q\colon \theta\right) - f_q\colon c\right)\right). \tag{5}$$

Among them, P_p is the point in z_k, P_p is converted into P_q by formula (4), and $(x_k, y_k)^T$ can be calculated. After the above steps, the real pose estimation of unmanned vehicle X_k at the current moment $X_k = (x_{k,} y_{k,} \varphi_{k,})^T$ can be calculated.

3. Location Method of Lidar Based on Markov

Markov localization method discretizes the position and attitude of the unmanned vehicles in grid and calculates the probability of the unmanned vehicles in this position and posture for each grid, which is called the reliability of the grid. At each time, the algorithm is divided into two stages. Firstly, on the basis of grid reliability of the previous time, the reliability is estimated by using the motion control information, and then the prediction reliability is corrected by using the observation information of sensors.

The position and posture of the unmanned vehicle is expressed as $X = (x, y, \varphi)$, (x, y) indicates the position, and φ indicates the posture of the unmanned vehicle. In the two-dimensional feature map, it is the direction angle of the unmanned vehicle. Further, $x \in [x_{\min}, x_{\max}]$, $y \in [y_{\min}, y_{\max}]$ where $x_{\min}, x_{\max}$, respectively, represent the minimum and maximum X values that can be obtained by the component of unmanned vehicle position, and $y_{\min}$, $y_{\max}$, respectively, represent the minimum and maximum y values that can be taken by the component of unmanned vehicle position. At the same time, each small grid is marked as $X_{k,i}, i \in I$, and I represents the index set of all grids.

If the reliability of each grid is recorded as $Bel(X_{k,i})$ and the reliability of the prediction of the unmanned vehicle $X_{k,i}$ at the grid is recorded as $\widehat{Bel}(X_{k,i})$, the time update equation after discretization of the pose space can be expressed as follows:

$$\widehat{Bel}\left(X_{k,i}\right) = \sum_{j \in I} p\left(X_{k,i} | X_{k-1,j}, u_k\right) Bel\left(X_{k-1,j}\right). \tag{6}$$

The observation renewal equation can be expressed as follows:

$$Bel\left(X_{k,i}\right) = \eta p\left(z_k | X_{k,i}, M\right) \widehat{Bel}\left(X_{k,j}\right), \tag{7}$$

where η is the normalization constant. Formula (6) gives the time update calculation method of the time update model based on the continuity of grid reliability mapping relationship. At time k, the reliability of all grids is calculated to complete the time update. This calculation process has a large amount of calculation because, firstly, the unmanned vehicle position and posture space has three degrees of freedom x, y, φ, and after discretization, there are a lot of grids, so each iteration needs to update the reliability of all grids; secondly, for example, in formula (6), when calculating the reliability of k any grid $X_{k,i}$ at any time, it is necessary to calculate all grids in the control input of unmanned vehicle at any time. The sum of the product of the distribution transferred to the grid $X_{k,i}$ and the time $k-1$ reliability.

To solve the problem of large amount of calculation in the process of time update, two solutions are proposed, respectively, for the above two reasons: for the problem of large number of grids, the method of dimension reduction is adopted. For the component of unmanned vehicle position and attitude space, that is, φ, the direction angle of unmanned vehicle can be obtained by reading the gyroscope installed on the unmanned vehicle. Therefore, the discrete unmanned parking space is transformed into a plane with two degrees of freedom x, y. Considering that each grid on the plane corresponds to its positioning reliability, if the coordinates of the grid center point $X_{k,i}$ are (x_i, y_i) , then there is a mapping relationship:

$$f_k(x_i, y_i) = \widehat{Bel}\left(X_{k,i}\right). \tag{8}$$

The reliability of all grids is calculated as a whole, and the grid reliability relationship $k-1$ is translated according to the motion control input of unmanned vehicle u_k to form a new grid reliability mapping relationship.

For the solution proposed above, the difficulty lies in that when the unmanned vehicle is translated by the integral multiple of the grid size under the control input, only the reliability of the grid needs to be reassigned according to formula (6). However, under the action of u_k, the unmanned vehicle does not necessarily translate according to the ideal integral multiple of the grid. For example, when the center point of one grid moves to the noncenter point of another grid, if the grid reliability is updated according to the above scheme, it will be the center point of the grid by default. Although this kind of error is very small, but the Markov location scheme is a continuous iterative process, the positioning at any K time depends on the positioning results at any $k-1$ time, so the error will accumulate continuously, making the positioning error larger and larger.

To eliminate the error caused by the nongrid integer multiple movement of the unmanned vehicle, this paper proposes a method of continuous grid belief mapping relationship. When updating any grid reliability, the new grid confidence is obtained by the translation of the unmanned

vehicle under the action of u_k, and then the new grid reliability is obtained by taking into account the uncertainty of motion on the mean confidence map.

The xy plane is composed of discrete grids, and each grid corresponds to a belief. Therefore, the grid confidence mapping relationship $f_k(x, y)$ can be expressed as a series of scatter points in 3D space, and its definition domain is the center point of all grids. As shown in Figure 5, the four points $q_{11}, q_{12}, q_{21}, q_{22}$ are the center points of the four grids, which are in the definition domain $f_k(x, y)$. However, if the point P is not in the definition domain, it cannot be taken.

The two-dimensional linear interpolation method is used to represent $f_k(x, y)$ the continuity $F_k(x, y)$, so that any point (x_0, y_0) can be taken from the four grid centers around ot $F_k(x_0, y_0)$. The conversion formula is as follows:

$$\begin{aligned} F_k(x_0, y_1) &= f_k(x_1, y_1)\frac{x_0 - x_1}{x_2 - x_1} + f_k(x_2, y_1)\frac{x_2 - x_0}{x_2 - x_1}, \\ F_k(x_0, y_2) &= f_k(x_1, y_2)\frac{x_0 - x_1}{x_2 - x_1} + f_k(x_2, y_2)\frac{x_2 - x_0}{x_2 - x_1}, \\ F_k(x_0, y_0) &= F_k(x_0, y_1)\frac{y_0 - y_1}{y_2 - y_1} + F_k(x_0, y_2)\frac{y_2 - y_0}{y_2 - y_1}. \end{aligned} \tag{9}$$

If the translation quantity of the unmanned vehicle under the action of control input is (x', y'), then $F_k(x, y)$ the relationship between time $k-1$ continuous reliability surface and time continuous reliability surface $F_{k-1}(x, y)$ is as follows:

$$F_k(x, y) = F_{k-1}(x - x', y - y'). \tag{10}$$

For the grid with any center point (x_i, y_i) at any time k, the relationship b between the initial reliability and the time k–2 continuous reliability surface is as follows:

$$Bel\left(\widehat{X}_{k,i}\right) = \frac{\int\int\int_D F_k(x, y)\mathrm{d}x\mathrm{d}y}{\Delta x * \Delta y}. \tag{11}$$

Let D be the area occupied by the grid in the xy plane $X_{k,i}$ and $\Delta x * \Delta y$ be the area of the grid. Considering that x' and y' can be arbitrary real numbers, this scheme can realize grid translation of any size, thus avoiding the error caused by discrete calculation.

Considering that u_k, displacement of the driverless vehicle (x', y'), is an uncertain value, it is necessary to use the calculated confidence grid for fuzzy operation, that is, convolution calculation. Convolution kernel is determined by the motion model of the unmanned vehicle, which represents the uncertainty degree of the unmanned vehicle transferring to another grid and its surrounding grid in u_k.

In the process of unmanned vehicle movement, positioning is only determined by u_k which is not accurate enough but also combined with each sensor observation Z_k to correct the positioning reliability of each grid. The calculation difficulty of formula (7) is the observation likelihood $P(Z_k|X_{k,i}, M)$, which represents the possibility of observing Z_k at the grid $X_{k,i}$ in the map.

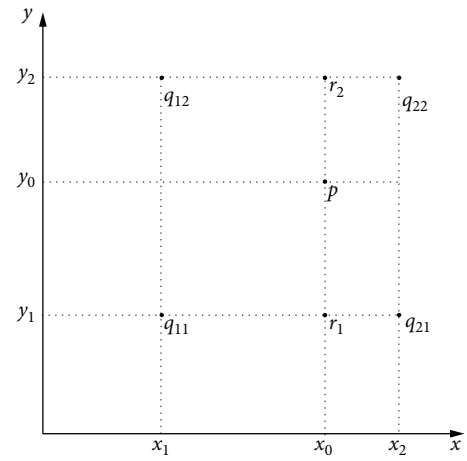


Figure 5: Two-dimensional interpolation.

In the existing feature map-based research, the usual way to calculate observation likelihood is to use NN or JCBB [14] methods to associate observation data with map data and then calculate likelihood values. However, the essence of these methods is only to give the only correlation assumption between observations and maps, which obviously does not conform to the multicorrelation situation that may exist in unmanned vehicle positioning. Therefore, this paper proposes a method to calculate the likelihood value without data association. The main idea of this method is as follows. Firstly, the line segment feature of the observed point cloud data is extracted, and then the line segment feature of the map that can be observed by the unmanned vehicle sensor in this position is extracted at the center point of the grid. The likelihood value is calculated by comparing the similarity between the two.

Suppose that the line segment set $F = \{f_i| = [c, q, \theta, l, p], i = 1, \ldots, n\}$, n is the number of line segments in the set, and each line segment f_i can be expressed as the equation shown in formula (1). Since the lidar can only observe the nearest point at any angle θ, a ray with an angle of α from the origin is drawn and intersected with a line segment in the line segment feature set to produce an intersection point p'. If the distance between p' and the origin is β, then the set F can be represented by the mapping relationship $G(\alpha) = \beta$.

Assuming that the line segment feature set F_{Z_k} obtained from the line segment feature extraction of the observation data Z_k of the unmanned vehicle is represented by the mapping relationship $G_{Z_k}(\alpha)$ and the linear feature set $F_{M(X_{k,i})}$ that the unmanned vehicle can observe in the map at the center of the grid $X_{k,i}$ is represented by the mapping relationship $G_{M(X_{k,i})}(\alpha)$, then the observation likelihood $P(Z_k|X_{k,i}, M)$, M) can be represented by the similarity between $G_{Z_k}(\alpha)$ and $G_{M(X_{k,i})}(\alpha)$. In order to calculate the similarity between $G_{Z_k}(\alpha)$ and c, this paper discretizes the two functions and then compresses them value by value. The likelihood value $P(Z_k|X_{k,i}, M)$ is calculated as follows:

$$\left(Z_k|X_{k,i},M\right)=\frac{1}{n}\sum_{i=1}^{n}\frac{\beta_{\max}-\left|G_{Z_k}(\alpha_i)-G_{M(X_{k,i})}(\alpha_i)\right|}{\beta_{\max}}. \tag{12}$$

Among them, $\beta_{\max}$ for unmanned vehicles can detect the maximum distance and $I=\{\alpha_i, i=1,\ldots,n\}$ is the angle of discretization.

4. Experiment

To evaluate the performance of the proposed methods in this paper, we test line segment extraction and create line segment feature map experiment, and the results are compared with other algorithms.

4.1. Line Segment Extraction Experiment. The point cloud data obtained by lidar will have noise due to various factors, and the result of line segment feature extraction algorithm will be affected. To verify the advantages of the proposed method in antinoise ability, the original lidar data without filtering is used for line feature extraction experiment.

The experimental results of segment feature extraction using IEPF are shown in Figure 6. The segmentation threshold of IEPF algorithm is set to 3 cm. It can be seen that there are clearly three lines in the point cloud data. Besides these three lines, there are some scattered noise points. Because IEPF has no antinoise ability, it can detect three lines by mistake. The principle of IEPF's sensitivity to noise has been elaborated in the first section of this paper.

The experimental results of the method described in this paper are shown in Figure 6, where the size of Hough transform grid is set to $(\pi/12)\times 5$; that is, $\Delta\theta=(\pi/12)$, $\Delta c=5$ cm; the error constant of voting box is $\delta=3$. It can be seen that the method described in this paper can simply and accurately restore the line features in the environment. The reason the method proposed in this paper has the ability to carry noise is that Hough transform determines the possible line segment features according to the number of votes. Therefore, only when there are a large number of points on the same line segment feature, the line segment feature can be extracted, and the noise distribution is random, which is not enough for the extraction conditions.

In this experiment, 100 frames of point cloud data are randomly extracted, among which there are 298 lines; that is, $N_e=298$. The experimental data of the two line segment feature extraction algorithms are shown in Table 1. Although IEPF algorithm can extract most of the actual line segment features, there are also a large number of erroneous extracted data due to noise.

In this paper, we use the same method in research [15] to describe the accuracy and error rate of line segment feature extraction algorithm. Formula (6) represents the accuracy rate of the algorithm, and formula (7) represents the error rate of the algorithm. Among them, N_t represents the number of real line segment features in the point cloud data, N_e represents the number of line segment features extracted by the line segment feature extraction algorithm, and N_m represents the number of line segment features extracted by the algorithm and the actual line segment features:

$$R_t=\frac{N_m}{N_t},\qquad R_f=\frac{N_e-N_m}{N_e}. \tag{13}$$

4.2. Experiment of Creating Line Segment Feature Map. To verify the effectiveness of the algorithm proposed in this paper, the data set is used in the experiments. The data set is collected by the unmanned vehicle driving in a circular corridor with LIDAR. The data set contains 551 sets of LIDAR point cloud data and odometer data.

In this paper, GMapping is used as the comparison method, and the same data set is used for experiment. Figure 6 shows the effect of GMapping on the data set. Figure 6(a) uses 30 particles to create a map, and Figure 6(b) creates a map with 60 particles. It can be seen that the map created with 60 particles is closer to the real situation.

Figure 7 shows the mapping effect of the method proposed in this paper on the dataset. It can be seen that the map created by the method in this paper can also reflect the real terrain well with fewer particles.

In addition to the advantage of requiring fewer particles, the proposed algorithm also needs less memory than GMapping algorithm in runtime and map storage. There are two reasons. On the one hand, SLAM algorithm based on particle filter saves a path and map for each particle. The proposed method needs fewer particles, so it can save memory. On the other hand, GMapping uses raster maps, and the memory needed to store the map increases with the increase of map area. In this paper, we use a line segment feature map. Each line segment in the map only needs four parameters to store, and the required memory will not be affected by the map size if there is no segment feature.

4.3. Markov Localization Experiment. To verify the effectiveness of the proposed localization method for unmanned vehicle (UAV) based on Markov in real situation, a random driving simulation experiment with noise is designed. The simulation environment in the experiment is shown in the square area shown in Figure 8,

In the figure, the side length is 80m; the initial pose of the unmanned vehicle is (10, 63, 0); the speed is 1.6 m/s; the positioning operation is performed once every second; the maximum measurement distance of the lidar is set to 10 m, the field of vision is set to 180 degree, and the laser scanning interval is 6 degrees. The size of the reliability grid is 1 m × 1 m.

The parameters of this experiment have noise to simulate the sensor error in real environment. The standard deviation of velocity measurement is 0.1 m/s, the standard deviation of lidar measurement is 0.05 m, and the standard deviation of angle measurement of lidar is 0.5 degrees.

Step 1: First, make the point cloud data sparse, and then Hough transform is used to preliminarily determine the features of line segments and the points belonging to these line segments.
Step 2: Search the voting box in Hough space by depth first algorithm, and merge the approximate line segments.
Step 3: Using the least square method to obtain accurate line features.

Algorithm 1: Line feature extraction algorithm based on Hough transform.

Step 1. Algorithm initialization: $k = 0$, the N system state samples are extracted from the prior distribution $p(X_0, m_0)$ as particles, the initial weight of each particle is $1/N$, and the size of particle set S_0 is N;
Step 2. Building a local map: In this step, segment feature representation and extraction method are used to construct the local line segment map in the field of vision of unmanned vehicle at the current time according to the observation data z_k of sensor. The sensor is 2D lidar, z_k, which is the point cloud data observed by lidar;
Step 3. Sampling: For all particles, the motion control input u_k and observation data z_k of unmanned vehicle are used to determine the sampling area;
Step 4. Weight update: For all particles, the weight of particles is calculated according to the unmanned vehicle position X_k, observation data z_k and the constructed map m_{k-1};
Step 5. Resampling: When the weight of a few particles is large and the weight of other particles is small, resampling is conducted to delete the particles with large error;
Step 6. Map update: For each particle, according to the estimated pose X_k in step 2, the local feature subgraph m_k' created in step 1 is completed to the constructed global map m_{k-1} to form a new global map m_k.

Algorithm 2: Algorithm of creating line feature map based on RBPF.

(a) (b)

Figure 6: GMapping the drawing effect. (a) Drawing effect of using 30 particles by GMapping. (b) Drawing effect of using 60 particles by GMapping.

Table 1: Comparison of line segment feature extraction data.

Sparsity degree	Accuracy (%)	Error rate (%)
IEPF	93.44	37.26
The proposed method	97.5	0

The simulation results are shown in Figure 9. It can be seen that when $k = 0$, the position and attitude of the unmanned vehicle are (10, 63, 0). In the reliability grid, the surrounding environment is a corridor parallel to the X-axis, and there are many areas with similar environments in the map. Therefore, the grid reliability of the corridor location attachment is high at the initial moment. When $k = 38$, after 38 iterations, each iteration will compare the observed data with the map in real time. The reliability of most regions has dropped to a very low value, only the grid (70, 63) has the highest reliability, and the real location (70.8, 63.5) of the unmanned vehicle is also in the grid. When $k = 76$, the real position of the unmanned vehicle is (41, 59); as shown in the figure, the reliability around the grid (41, 59) is significantly higher than that in other locations. From the above process, we can judge that with the movement in the environment, the unmanned vehicle continuously observes the surrounding environment, and the unmanned vehicle gradually converges to its real location from the assumed multiple positioning points at the beginning. The above experiments show that with the movement of the unmanned vehicle, the sensors observe increasingly information from the environment. Even if there are similar regions in the environment, the proposed localization algorithm based on Markov

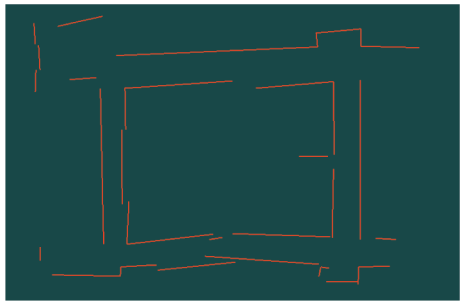


Figure 7: Drawing effect of using 10 particles by the proposed method.

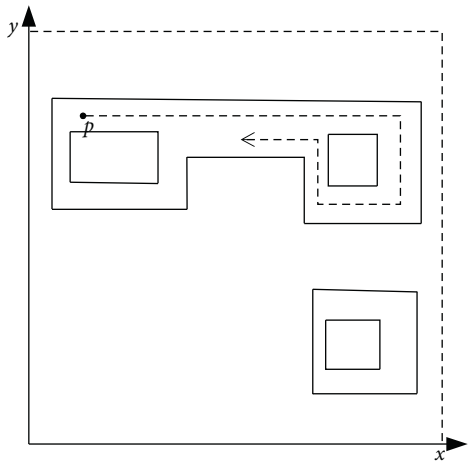
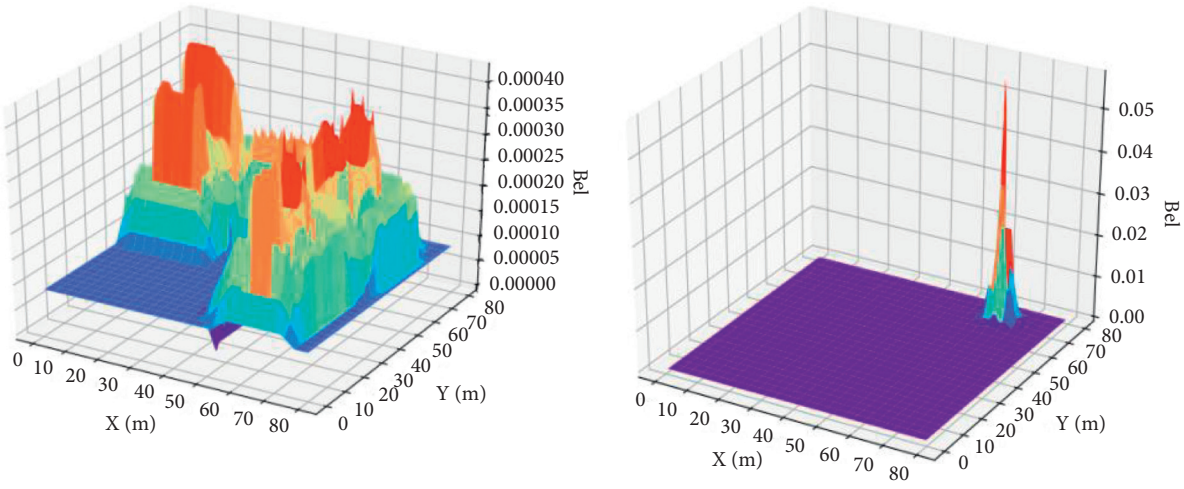


Figure 8: Random driving experiment noise.



(a) (b)

Figure 9: Continued.

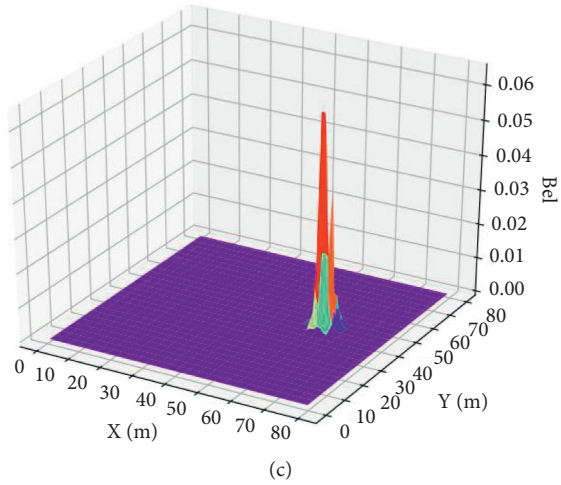


Figure 9: Reliability grid of random driving simulation experiment under noise. (a) $k = 0$, (b) $k = 38$, and (c) $k = 76$.

can converge the estimated position to the real location of the unmanned vehicle.

5. Conclusion

In this paper, the unmanned vehicle based on lidar is taken as the research background, and the research work is carried out in three aspects: the extraction of line segment, feature representation in feature map, the creation of line segment feature map, and the localization of unmanned vehicle. Through the analysis of simulation experiments, the proposed method can effectively realize the localization of the unmanned vehicle in the feature map. Compared with the contrast algorithm, the effect is better. In addition, there are still some problems in this paper. (1) In this paper, the map representation method only considers the line segment characteristics and cannot describe some other shape obstacles in the environment; (2) in the process of map construction, particle degradation occurs from time to time, although selective resampling measures have been taken, but the resampling time is still relatively frequent, resulting in high computational complexity; (3) in Markov localization, the super large map will cause huge calculation. In the follow-up work, we can divide the large-scale map into subgraphs and locate them on the subgraph. These issues will continue to be considered in the future.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research was funded by Research on Team Resource Sharing Mechanism Based on Big Data, Systematic Research Project on the construction of teachers' teaching innovation team in National Vocational Colleges (no. tx20200401).

References

[1] K. Demarsin, D. Vanderstraeten, T. Volodine, and D. Roose, "Detection of closed sharp feature lines in point clouds for reverse engineering applications," in *Proceedings of the International Conference on Geometric Modeling and Processing*, pp. 571–577, Springer, Berlin Germany, July 2006.

[2] Y. L. Huang, Y. T. Ye, and Z. L. Chen, "New method of fast hough transform for circle detection," *Journal of Electronic Measurement and Instrument*, vol. 24, no. 9, pp. 837–841, 2010.

[3] T. Einsele, "Real-time self-localization in unknown indoor environment using a panorama laser range finder," in *Proceedings of the International Conference on Intelligent Robots and Systems*, vol. 2, pp. 697–702, IEEE, Grenoble, France, September 1997.

[4] F. Tombari, N. Fioraio, T. Cavallari, S. Salti, A. Petrelli, and L. D. Stefano, "Automatic detection of pole-like structures in 3D urban environments," in *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4922–4929, Chicago, IL, USA, September 2014.

[5] J. G. Kang, S. Y. An, W. S. Choi, and S. Y. Oh, "Recognition and path planning strategy for autonomous navigation in the elevator environment," *International Journal of Control, Automation and Systems*, vol. 8, no. 4, pp. 808–821, 2010.

[6] C. X. Chen, J. Zhang, and X. Y. Ji, "Review of LIDAR slam technology and its application in autonomous vehicles," *Journal of Beijing Union University*, vol. 31, no. 4, pp. 61–69, 2017.

[7] Y. Yu, J. Li, H. Guan, C. Wang, and J. Yu, "Semiautomated extraction of street light poles from mobile LiDAR point-clouds," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 3, pp. 1374–1386, 2015.

[8] M. Montemerlo, S. Thrun, and D. Koller, "A factored solution to the simultaneous localization and mapping problem," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, pp. 593–598, Pittsburgh, PA, USA, July 2005.

[9] M. Montemerlo, S. Thrun, and D. Koller, "An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proceedings of the*

Sixteenth International Joint Conference on Artificial Intelligence, vol. 2, pp. 1151–1156, August 2003.

[10] 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.

[11] Y. Liu, S. Zhu, and B. Jin, "Study on markov localization algorithm for mobile robot:new modelling method for orientation sensor," *Journal of Zhejiang University(Engineering Science)*, vol. 39, no. 3, pp. 25–27, 2005.

[12] Y. W. Hu, Y. Jiang, and J. W. Gong, "Window constrained markov localization for unmanned ground vehicle," *Transactions of Beijing Institute for Technology*, vol. 34, no. 4, pp. 353–357, 2014.

[13] Y. Li, *Localization and Mapping on Urban Area Based on 3D Point Cloud of Autonomous Vehicles*, Beijing Institute of Technology, Beijing, China, 2016.

[14] Z. Wu and C. Zhao, "An optimized data association solution for SlAM," *Robot*, vol. 31, no. no3, pp. 217–223, 2009.

[15] V. Nguyen, S. Gächter, A. Martinelli, N. Tomatis, and R. Siegwart, "A comparison of line extraction algorithms using 2D range data for indoor mobile robotics," *Autonomous Robots*, vol. 23, no. 2, pp. 97–111, 2007.