2016 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration

3D Indoor Map Building with Monte Carlo Localization in 2D Map

Lei Zhao, Zhun Fan, Wenji Li, Honghui Xie, Yang Xiao
Department of Electronic Engineering
Shantou University
Guangdong, Shantou 515063
E-mail: 14lzhao1@stu.edu.cn

***Abstract*—In this paper, we propose a 3D indoor map building method based on Monte Carlo localization in 2D map. The traditional 3D SLAM mainly adopts the visual odometry technology for robot localization. However, the visual localization has a poor real-time performance. Besides, in some special scenarios, such as corridors, the visual localization may generate matching errors, resulting in cumulative errors. These errors will lead to a wrong robot localization. The Monte Carlo localization based on lidar in 2D map can achieve a higher localization accuracy. Therefore, we use the above method to replace the visual localization while using a kinect to collect 3D environment information. To study the performance of the proposed method, we make some experiments and compare with the popular open source RGB-D SLAM system based on visual localization provided by Felix Endres et al. in 2014. The experimental results demonstrate that our method has better effect in the indoor corridor environment with less features.**

***Keywords*-lidar, kinect, visual localization, Monte Carlo localization, 3D indoor map building**

I. INTRODUCTION

Mobile robot simultaneous localization and mapping (SLAM) is one of the key technologies to realize autonomous navigation of the robot [1]. As the mobility of a robot and the complexity of an indoor environment have been increasing, the robot must consider the 3D information of the environment in order to determine its mobile strategy. 3D SLAM technology is an important approach to improve the robot mobility and perception ability.

With the development of image processing and computer vision technology, the localization technology based on vision sensors is widely applied in the robot indoor 3D SLAM [2]. Kinect, is a vision sensor which can provide color and depth information of the environment [3]. By implementing feature extraction and matching of the collected image, the visual odometry can execute motion estimation to locate the robot [4][5]. But the process of image feature extraction and matching consumes a large amount of computational resources which reduces the real-time performance. Besides, due to the visual localization depends on the image information, there exists matching errors under some special scene, such as corridors. At the same time, only relying on a single sensor, for example the kinect, the cumulative errors of the system can't be corrected [6].

In a robot navigation system, a robot uses a lidar and a wheeled odometry, and then combine them with Monte Carlo localization (MCL) algorithm which can achieve a higher localization accuracy in 2D map. Therefore, we adopt the above method to replace the visual localization which can avoid the process of image feature extraction and matching, and reduces the time-consuming. Meanwhile, the multi sensor information fusion among kinect, lidar and odometry can correct the cumulative errors of the system. The MCL in a 2D map can afford us the robot poses at different time. By calculating the relationship among these poses, we can get the pose transition matrices. These transition matrices are served for the 3D point cloud registration [7]. After that, we finally get the indoor 3D map.

The rest of this paper is organised as follows. Section II introduces the Monte Carlo localization theory. Section III discusses the 3D indoor mapping. In section IV, we make some experiments to verify the effect of the proposed method and compare with the popular open source RGB-D SLAM system based on visual localization provided by Felix Endres et al. in [8]. The last section summarizes the work of this paper.

II. MONTE CARLO LOCALIZATION THEORY

MCL is an algorithm that can determine a robot position in a 2D map. By using the particle filter, it is applicable to both local and global localization problems [9]. It is easy to implement and tend to work well across abroad range of localization problems. Due to the robot motion model $p(x_t|x_{t-1}, u_{t-1})$ and measurement model $p(z_t|x_t)$ are applied to the prediction stage and update stage of MCL, we will introduce these two models at first [10] .

A. Robot motion model

Robot motion model uses the relative motion information as measured by the robots internal odometry. More specifically, in the time interval $(t-1, t]$, the robot advances from x_{t-1} to x_t. The odometry reports back to us a related advance from $\overline{x}_{t-1} = (\overline{x} \quad \overline{y} \quad \overline{\theta})$ to $\overline{x}_t = (\overline{x}' \quad \overline{y}' \quad \overline{\theta}')$. This affords us relative odometry information by the following equations:

$$\delta_{rot1} = \arctan 2(\overline{y}' - \overline{y}, \overline{x}' - \overline{x}) - \overline{\theta} \tag{1}$$

978-1-5090-3575-5/16 $31.00 © 2016 IEEE
DOI 10.1109/ICIICII.2016.29

$$\delta_{trans} = \sqrt{(\overline{x} - \overline{x}')^2 + (\overline{y} - \overline{y}')^2} \tag{2}$$

$$\delta_{rot2} = \overline{\theta}' - \overline{\theta} - \delta_{rot1} \tag{3}$$

Where the letters with a bar indicate that these are odometry measurements embedded in a robot internal coordinate whose relation to the global world coordinates is unknown. The robot motion in the time interval $(t, t-1]$ is approximated by a rotation δ_{rot1} , followed by a translation δ_{trans} and a second rotation δ_{rot2} . To model the motion error, we assume that the true values of the rotation and translation are obtained from the measured ones by subtracting independent noise sample (ε_{b^2}) :

$$\hat{\delta}_{rot1} = \delta_{rot1} - \text{sample}(\alpha_1\delta_{rot1}^2 + \alpha_2\delta_{trans}^2) \tag{4}$$

$$\hat{\delta}_{trans} = \delta_{trans} - \text{sample}(\alpha_3\delta_{trans}^2 + \alpha_4\delta_{rot1}^2 + \alpha_4\delta_{rot2}^2) \tag{5}$$

$$\hat{\delta}_{rot2} = \delta_{rot2} - \text{sample}(\alpha_1\delta_{rot2}^2 + \alpha_2\delta_{trans}^2) \tag{6}$$

Where ε_{b^2} is a zero-mean noise variable with variance b^2. The parameters α_1 to α_4 are robot-specific error parameters. Consequently, we get the next moment position, x_t :

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} \hat{\delta}_{trans}\cos(\theta + \hat{\delta}_{rot1}) \\ \hat{\delta}_{trans}\sin(\theta + \hat{\delta}_{rot1}) \\ \hat{\delta}_{rot1} + \hat{\delta}_{rot2} \end{pmatrix} \tag{7}$$

Where the $x_{t-1} = (x \quad y \quad \theta)^T$ represents last moment pose. According to the above formula, every particle that is transformed by the odometry motion model will be distributed according to $p(x_t|x_{t-1}, u_{t-1})$.

B. Measurement model

After having a priori information about the pose of the robot, the measurement probability $p(z_t|x_t)$ can be calculated as follows.

Assuming m is the known map information and $z_t^i (i = 1, 2, 3, \ldots, K)$ refers to an individual measurement. The probability $p(z_t|x_t, m)$ is obtained as the product of the individual measurement likelihoods:

$$p(z_t|x_t, m) = \prod_{k=1}^{K} p(z_t^k|x_t, m) \tag{8}$$

If $z_t^k \neq z_{max}$, the individual measurement likelihood $p(z_t^k|x_t, m)$ can be calculated as follows:

$$x_{z_t^k} = x + x_{k,sens}\cos\theta - y_{k,sens}\sin\theta + z_t^k\cos(\theta + \theta_{k,sens}) \tag{9}$$

$$y_{z_t^k} = y + y_{k,sens}\cos\theta + x_{k,sens}\sin\theta + z_t^k\sin(\theta + \theta_{k,sens}) \tag{10}$$

$$dist = \min_{x',y'}\{\sqrt{(x_{z_t^k} - x')^2 + (y_{z_t^k} - y')^2} \mid \langle x', y'\rangle \text{occupied in } m\} \tag{11}$$

$$p(z_t^k|x_t, m) = z_{hit} \cdot \text{prob}(dist, \sigma_{hit}) + \frac{z_{random}}{z_{max}} \tag{12}$$

Where the function $\text{prob}(dist, \sigma_{hit})$ computes the probability of $dist$ under a zero-centered Gaussian distribution with standard deviation σ_{hit}. $x_{k,sens}$ and $y_{k,sens}$ represent the location of the sensor relative to the robot center. $\theta_{k,sens}$ refer to the angular orientation of the sensor beam relative to the robot heading direction. z_{hit}, z_{random}, z_{max} denote the intrinsic parameters of the observe model.

C. Monte Carlo localization

MCL has a variety of different versions. In this paper, we use KLD-Sampling MCL. KLD-sampling can adapt the number of particles over time [10]. KLD-sampling is derived from the Kullback-Leibler divergence, which is a measure of the difference between two probability distributions [11]. At each iteration of the particle filter, KLD-sampling determines the number of samples such that, with probability $1 - \delta$, the error between the true posterior, and the sample-based approximation is less than ε.

The algorithm takes as input the previous sample set χ_{t-1} along with the map m and the most recent control u_t and measurement z_t. In a nutshell, KLD-sampling MCL generates particles until the following statistical bound is satisfied:

$$M_\chi := \frac{k-1}{2\varepsilon}\left\{1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}}z_{1-\delta}\right\}^3 \tag{13}$$

This bound is based on the volume of the state space that is covered by particles. The volume covered by particles is measured by a histogram. Each bin in the histogram H is either empty or occupied by at least one particle. Initially, each bin b in the histogram H is set to empty. And then each particle i is drawn from the previous sample set χ_{t-1}:

$$draw\ i\ with\ probability\ \propto\ \omega_{t-1}^i \tag{14}$$

Based on this particle, a new particle is predicted, weighted, and inserted into the new sample set χ_t :

$$x_t^{[M]} = \text{sample\_motion\_model}(u_t, x_{t-1}^{[i]}) \tag{15}$$

$$\omega_t^{[M]} = \text{measurement\_model}(z_t, x_t^{[M]}, m) \tag{16}$$

$$\chi_t = \chi_t + \langle x_t^{[M]}, \omega_t^{[M]} \rangle \tag{17}$$

If the newly generated particle falls into an empty bin of the histogram, then $k = k + 1$ and the bin is marked as non-empty. Thus, k measures the number of histogram bins filled with at least one particle. If $k > 1$, then we use the equation (13) to calculate the M_χ. The algorithm generates new particles until $M < M_\chi$ or $M < M_{\chi min}$. Finally, by KLD-Sampling MLC described as above, we get the next moment particle set χ_t.

III. 3D INDOOR MAP BUILDING

Initially, through odometry and lidar, the control information u_t and observe information z_t are acquired. Then KLD-Sampling MCL exploits these information to locate the robot in the 2D map that constructed in advance. As the robot moves, we can get its pose x_t at different times in the 2D map. The following equation describes the pose transformation process between the last moment and next moment:

$$x_t = R \cdot x_{t-1} + t \tag{18}$$

Where R is a rotation matrix and t is a translation vector. Meanwhile, R and t represent the posture of the robot and kinect, since the position of kinect relative to the robot is fixed. The kinect collects a pair of 2D color images and depth images whenever the robot pose x_t is obtained. After that, the color and depth images can be used to build a local 3D map which is appeared by point cloud. The following equations express how to convert 2D images to 3D point cloud:

$$z = d/s \tag{19}$$

$$x = (u - c_x) \cdot z/f_x \tag{20}$$

$$y = (v - c_y) \cdot z/f_y \tag{21}$$

where c_x, c_y, f_x, f_y represent kinect internal parameters [12]. d is the depth information and s is the scaling factor. u and v are the position of the image pixels. x, y, z are the position of the 3D point cloud. Assuming a robot obtains the local 3D point cloud p_{t-1} and p_t in pose x_{t-1} and x_t respectively, then we can implement point cloud registration for the p_{t-1} and p_t . Point cloud registration, in essence, is a process of point cloud transformation, which is often described by a transform matrix:

$$T = \begin{bmatrix} R_{3\times3} & t_{3\times1} \\ O_{1\times3} & 1 \end{bmatrix} \in R^{4\times4} \tag{22}$$

IV. RESULTS AND COMPARISON

A. Experiments and Results

To prove the effect of the 3D indoor map building method based on Monte Carlo localization in 2D map, this paper has conducted experiments on a Turtlebot2 robot with a low cost RPlidar lidar and a kinect as the sensors. The experimental site is Guangdong Key Laboratory of Digital Signal and Image Processing, Science and technology building, 4 floor, Shantou University. The experiment environment scenario is shown in Figure 1. The experiment consists of two parts:

1) 2D map building and localization: In this paper, we choose Gmapping to build our 2D map. Gmapping is a ROS package that is convenient to use [13][14][15]. It uses highly efficient Rao-Blackwellized particle filer(RBPF) to learn grid maps from laser range data [16][17]. The constructed 2D map is shown in Figure 2. The black lines represent obstacles in the environment, for example the walls and the doors, which is infeasible area that the robot should be avoided. The white and gray area represent feasible and unknown region respectively.

Once the 2D map is obtained, we can locate the robot in the 2D map by using KLD-Sampling MCL. Figure 3 shows the initial localization situation. In the beginning the robot doesn't know where it is, therefore it has the same probability to appear at anyplace in the 2D map. The green dots in Figure 3 represent particles and each particle represents a possible location of the robot. The black dot is the robot model and its place denotes the locating result estimated by the particles that processed by the KLD-Sampling MCL. Through several movements, the robot gradually determines its position by using KLD-Sampling MCL. Figure 4 shows the localization result and we can see that the particle gathered in the vicinity of the estimated position.

2) 3D indoor map building: The specific 3D indoor map is the corridor shown in Figure 1. The corridor is about 30 meters long and 2 meters wide.

In the experiment, we run the program of the proposed method in Linux operating system. The robot is controlled remotely with a gamepad. It starts at one end of the corridor, then it advances in the middle of the corridor and stops moving when it reaches the elevator. The Turtlebot2 publishes a pose information by using a ROS topic when it moves 0.2 meters away. At the same time, a local 3D point cloud will be built and merged with the local 3D point cloud which obtained in the last pose. The whole 3D indoor corridor map is shown in Figure 5. The elevator and the door on both sides of the corridor can be easily identified in the 3D map.

B. Comparison

We make a comparison between the proposed method and RGB-D SLAM system provided by Felix Endres et al.

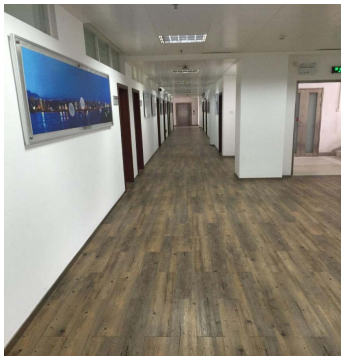

Figure 1. Experimental site: Guangdong Key Laboratory of Digital Signal and Image Processing, Science and technology building, 4 floor, Shantou University

Figure 2. The constructed 2D map of experimental site by using Gmapping

in [8]. In the experiment, we turn on the RGB-D SLAM program in Linux operating system and also control the robot with a gamepad. The robot trajectory is similar to the last experiment. However, due to the existence of error, the robot trajectory of these two experiments can't be exactly the same.

The 3D indoor map results are shown in Figure 6. In the middle of the picture, the intersection of the yellow lines represent robot pose and the yellow lines represent the transform relationship between these poses. The poses and their relationship are being constantly optimized by using General Graph Optimization(g2o) [18].

Due to the experimental environment mainly consists of white walls and brown doors, it is lack of features. The RGB-D SLAM system generates many matching and cumulative errors. And these errors generated by the visual localization are so serious that the g2o algorithm can't correct them which result in a messy 3D indoor corridor map shown in Figure 6.

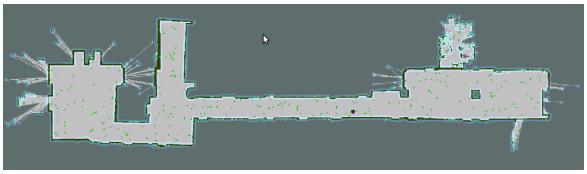

Figure 3. Initial localization situation of the robot in 2D map

Figure 4. The robot localization situation after several movements

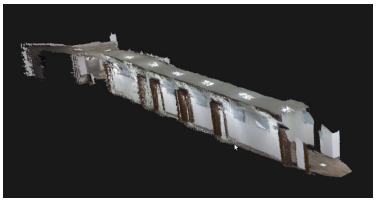

Figure 5. 3D indoor corridor map based on Monte Carlo localization in 2D map

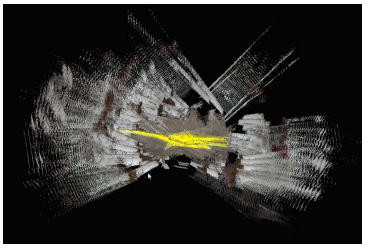

Figure 6. 3D indoor corridor map using RGB-D SLAM system provided by Felix Endres et al.

V. Conclusion

In this paper, we design a 3D indoor map building method based on Monte Carlo localization in 2D map. The MCL in 2D map can achieve a higher localization accuracy and avoids some shortcomings of the visual localization, especially in the terms of time consuming and cumulative errors. To verify the effect of suggested method, we make some experiments to test the performance of proposed method and the popular open source RGB-D SLAM system based on visual localization. The experimental results show that the proposed method can achieve better localization which results in a better 3D map.

The future work will mainly focus on the application of 3D indoor map in the robot navigation and optimizing the pose and the pose transform matrix of the robot.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant (61175073, 61300159, 61332002, 51375287), in part by the Guangdong High-Level University Project Green Technologies for Marine Industries, in part by the Science and Technology Planning Project of Guangdong Province (2013B011304002) and in part by the Project of Educational Commission of Guangdong Province, China2015KGJHZ014).

REFERENCES

[1] S. Thrun, J. J .Leonard, Simultaneous localization and mapping, in Springer handbook of robotics, pp.871-889, Springer Berlin Heidelberg, 2008.

[2] J. Fuentes-Pacheco, J. Ruiz-Ascencio, J. M. Rendn-Mancha, Visual simultaneous localization and mapping: a survey, Artificial Intelligence Review, vol. 43, no. 1, pp.55-81, 2015.

[3] E. E. Hitomi, J. V. L. Silva, G. C. S. Ruppert, 3D scanning using RGBD imaging devices: A survey, Developments in Medical Image Processing and Computational Vision, pp.379-395, Springer International Publishing, 2015.

[4] D. Scaramuzza, F. Fraundorfer, Visual odometry [tutorial], IEEE Robotics & Automation Magazine, vol. 18, no. 4, pp. 80-92, 2011.

[5] F. Fraundorfer, D. Scaramuzza, Visual odometry: Part II: Matching, robustness, optimization, and applications, IEEE Robotics & Automation Magazine, vol. 19, no. 2, pp. 78-90, 2012.

[6] J. C. K. Chow, D. D. Lichti, J. D. Hol, et al., Imu and multiple rgb-d camera fusion for assisting indoor stop-and-go 3d terrestrial laser scanning, Robotics, vol. 3, no. 3, pp.247-280, 2014.

[7] http://docs.pointclouds.org/trunk/group__registration.html

[8] F. Endres, J. Hess, J. Sturm, et al., 3-D mapping with an RGB-D camera, IEEE Transactions on Robotics, vol. 30, no. 1, pp.177-187, 2014.

[9] F. Dellaert, W. Burgard, D. Fox, et al., Monte Carlo localization for mobile robots, Proceedings of the IEEE International Conference on Robotics and Automation, pp.1322-1328, Detroit, USA, 1999.

[10] S. Thrun, W. Burgard, D. Fox, Probabistic Robotics, The MIT Press, September, 2005.

[11] J. M. Joyce, Kullback-leibler divergence, International Encyclopedia of Statistical Science, pp.720-722, Springer Berlin Heidelberg, 2011.

[12] Z. Zhang, A flexible new technique for camera calibration, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 11, pp. 1330-1334, 2000.

[13] http://www.ros.org/gmapping

[14] G. Grisetti, C. Stachniss, W. Burgard, Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling, in Proceedings of the 2005 IEEE International Conference on Robotics and Automation , pp. 2443C2448, Barcelona, Spain, 2005.

[15] G. Grisetti, C. Stachniss, W. Burgard, Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters, IEEE Transactions on Robotics, vol. 23, no. 1, pp. 34-46, 2007.

[16] Murphy, P. Kevin, Bayesian map learning in dynamic environments, in Proc. Conf. Neural Inf. Process. Syst, pp.1015C1021, 1999.

[17] A. Doucet, J. de Freitas, K. Murphy, S. Russel, Rao-Blackwellized particle filtering for dynamic Bayesian networks, in Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence, pp. 176C183, 2000.

[18] R. Kmmerle, G. Grisetti, H. Strasdat, et al., g 2 o: A general framework for graph optimization, IEEE International Conference on Robotics and Automation (ICRA), pp. 3607-3613, 2011.