Simulation and Implementation of SLAM Drawing Based on ROS Wheeled Mobile Robot

Yufan Cui, Zijie Niu*

School of mechanical and electronic engineering, Northwest A&F University, Xianyang, China

*Corresponding author: niuzijie@nwsuaf.edu.cn

Abstract. In order to realize the establishment of the unknown terrain map in the indoor environment of the ROS wheeled mobile robot, this article uses Lidar, Raspberry Pi and other hardware to build the wheeled mobile robot, in the ROS environment, the robot is built by the URDF model, and the robot model is configured. Differential controller, keyboard control node, etc. Establish the simulation environment, use different simulation environments to visually simulate the robot, and simulate the two optimization algorithms in the SLAM algorithm to simulate the construction of the map. In order to explore the effect of SLAM's different optimization algorithms on the construction of graphics, the SLAM algorithm based on filter and the SLAM algorithm based on graph optimization are introduced respectively. The robot is selected. After the robot is built, three different experimental environments are selected. A variety of map-building algorithms are used to experiment, comparing the effects of G-mapping algorithm and Cartographer algorithm to build maps in a small-scale, large-scale and multi-obstacle environment. Use the image processing software under the Linux system to analyse the map created by the robot in the experimental environment, measure the coordinates of the robot feature points in the created map by the number of pixels, and compare the actual coordinates of the location of the special point in the actual environment to evaluate Construction effect. By comparing the effects of the two algorithms on the contours of various obstacles when building a map, the effect of mapping in a multi-obstacle environment is evaluated. The results show that the SLAM graph-building algorithm based on graph optimization has a more stable graph-building effect.

Keywords: ROS, robot, slam mapping, navigation.

1. Introduction
To complete people's heavy work, the rapid development of computer technology and automation technology in all walks of life derived a series of intelligent robots in recent years. Some semi-autonomous or even fully autonomous unmanned vehicles have entered our life. However, the control system and storage space become a problem because a large number of sensors are needed to ensure the safety of the navigation process. A set of operating system with strong program module portability and low code reuse rate is needed to reduce the problem of insufficient system space and high code reuse
rate caused by multi-sensor real-time work. Compared with the traditional computer system, ROS (robot operating system) has the advantages of low code reuse rate, distributed communication and no difference in program language, so more and more scholars begin to develop robots in ROS.

In the ROS system, the designed robot can be simulated in Gazebo simulation and Rviz simulation environment, which provides a reliable guarantee for the robot navigation in the actual complex environment [1]. Nowadays, there are also examples of agricultural robots using ROS framework, and ROS robots are gradually applied in agriculture [2]. Therefore, the research on the navigation mode of wheeled mobile robot based on ROS has important guiding significance and practical value for the reliability of unmanned driving technology.

2. Simulation of G-mapping algorithm

2.1. Principle of G-mapping algorithm

G-mapping algorithm is a filter-based optimization algorithm in SLAM algorithm. The algorithm separates localization from mapping. Firstly, the robot is located, and then the robot's surrounding environment is mapped [3]. However, SLAM algorithm is a real-time location and mapping algorithm, which regards location and mapping as joint events. In other words, the idea of SLAM algorithm is in line with the joint distribution probability: P (x, y), and the joint probability can be transformed into conditional probability by the knowledge of probability theory, that is: P (X, Y) =P(Y|X) •P(X). Using this idea for location and mapping, the robot odometer information and the obstacle position information detected by lidar need to be combined. The specific process of G-mapping algorithm is as follows: data input, establishment of motion model, scanning matching, suggested distribution, weight calculation, resampling, particle maintenance map, map update. The specific procedures of each step are as follows:

Input the data measured by the sensor: data input means that the obstacle information measured by the lidar and the odometer information of the robot are transmitted to the main control board, and the data are placed in the function package of G-mapping algorithm.

Establishment of motion model: using the existing input data, the initial pose state of the robot is obtained, and the pose state is defined as M time to predict the pose of particles at M + 1 time.

Scanning matching: because the obstacle information detected by the lidar sensor is not absolute but noisy, the predicted particles at M+1 time are scanned and matched with multiple default particles preset in G-mapping algorithm. By comparing with the default particles, the best position coordinates of M+1 can be selected. If the difference of scanning matching is too large, it will be estimated in the default particles to find the nearest particle as the best position coordinate.

Suggested distribution: combine the detection model with the dynamic model, select the surrounding position coordinate samples in a certain range from the best position under the previous scan matching to calculate the mean and variance, so that they conform to the Gaussian distribution under the set mean variance.

Weight calculation: calculate the weight proportion of all sampling position coordinates, and then carry out the normalization operation. After resampling, the weight still needs to be calculated.

Resampling: When resampling is needed, the number of sampling should be moderate. If the number of sampling is too much, it may lead to particle degradation.

Particle maintenance map: update the map information carried by each particle. Each particle represents a map of a moment. After the update, the particle trajectory is obtained.

Map update: calculate the weight proportion of particles, that means calculate the weight again to get the most reliable surrounding information of the robot motion particles and to store the updated map in the ROS function package.

2.2. Simulation of G-mapping algorithm

According to the above G-mapping algorithm theory support, write G-mapping function package, open the keyboard control robot model node, gazebo simulation environment node and G-mapping navigation function package. Open the Rviz visualization tool, and use the keyboard to control the movement of
the robot when simulating the robot model. In the Rviz visualization environment, the map of the robot model based on G-mapping algorithm can be obtained in real time.

After the terminal is turned on, the keyboard can be used to control the movement of the robot model, and in Rviz visualization tool is the map of surrounding environment created by simulation robot when applying G-mapping algorithm, as shown in Figure 1.

Figure 1. Map created by the G-mapping algorithm.

3. Simulation of Cartographer mapping algorithm

3.1. Principle of Cartographer algorithm

Cartographer is a graph optimization algorithm based on SLAM algorithm. The main difference between this algorithm and G-mapping algorithm is that it has a closed-loop detection part [4].

The algorithm takes the robot's pose as a point in the map, takes the information scanned by the lidar as an edge, saves the relative distance between the point and the edge, and then optimizes the graph, also known as back-end nonlinear optimization. Back end non-linear optimization is mainly to reduce or eliminate the cumulative error caused by too long drawing time. The specific methods are as follows:

The laser scan information obtained by the sensor is saved in the sub map. At the beginning, the cumulative error in the sub map is very small, and the stored laser scan information is more reliable and can be saved [5].

With the increase of construction time and space, the cumulative error will gradually accumulate, and the laser scanning information in the new sub map is no longer reliable. So the previously established sub map is used as a tool in closed-loop detection. When a new laser scanning information is saved in the new sub map, the new scanning information will be scanned and matched with the laser scanning information in the sub map in closed-loop detection.

After scanning matching, a certain range of windows will be selected near the position and pose estimation of the newly added map laser scanning information. The key content of Cartographer is the creation of local subgraph fusing multi-sensor data and the implementation of scan matching strategy for closed-loop detection.

3.2. Page Numbers

According to the above theoretical support of cartographer algorithm, write Cartographer function package, open the node of keyboard control robot model, node of gazebo simulation environment and function package of Cartographer navigation. Open the Rviz visualization tool, and use the keyboard to
control the robot movement when simulating the robot model. In the Rviz, the map of the robot model based on the cartographer algorithm can be obtained in real time.

After the terminal is turned on, the keyboard can be used to control the movement of the robot model, as shown in Figure 2, which is the simulation situation after starting each node. The model in gazebo is the real scene simulated by the robot, and the model in Rviz visualization tool is the map of the surrounding environment created by the robot using Cartographer algorithm.

![Figure 2. Cartographer algorithm simulation diagram.](image)

4. Construction of a real robot

Some hardware needs to be selected to control the robot smoothly. Because in the process of robot motion mapping, a PC is needed to monitor the image obtained by the robot. The main control board of the robot should have good communication function, which can obtain map information and transmit the data to the PC. Meanwhile, a set of control board specially controlling the wheel speed is needed as the lower computer to calculate the wheel speed change of the robot and transmit the wheel speed information of the robot to the main control board. Some sensors are also needed to obtain the environmental information of the robot.

After the robot is assembled, PC is used for distributed communication. In this experiment, the robot side is selected as the host and the PC side as the slave because the robot side has to deal with a lot of data [6]. After the IP address of the PC is added to the robot, the master and slave can communicate [7].

In this design, the main hardware used are Raspberry Pie 3B + development board, STM32F103, SLR A1, 25GA370 DC reducer motor, etc. As shown in Figure 3, it is the physical picture of the robot.

![Figure 3. Robot physical picture.](image)
5. Experimental process and results

In this chapter, the real robot is used to test G-mapping and Cartographer algorithms to evaluate the accuracy of robot positioning and the ability to identify a variety of obstacles. Experiment one is to evaluate the accuracy of robot positioning, select two different experimental environments with flat and good light, and place a feature object in the experimental environment. The two algorithms are applied to build the map in the same experimental environment, and the positioning coordinates and shape restoration degree of the feature items in the corresponding algorithm are analyzed and compared, so as to compare the accuracy of the two algorithms in the same environment. In Experiment 2, two algorithms were used to compare the ability of obstacle recognition in the same experimental environment. Four groups of obstacles were set up in this experimental environment, and the two algorithms were used to build the map respectively. The ability of obstacle recognition was evaluated according to the effect of the construction.

5.1. Robot positioning accuracy evaluation

In the simulation environment, the difference between the two algorithms is not great, because the robot odometer information is more ideal [1], so it is necessary to compare the advantages and disadvantages of the two algorithms in the actual environment. Considering the light and the scope of the experimental environment, this experiment selects the bedroom and storage room as the experimental environment. A feature object is placed in different experimental environments, and the mapping effect of two different mapping algorithms is evaluated by the shape and coordinate position of the object in the map.

5.1.1. The first experimental environment test. The first experimental environment is the bedroom. The experimental environment is shown in Figure 4. G-mapping algorithm and Cartographer algorithm are used to map the bedroom environment respectively. The mapping effect of the two algorithms is analysed and compared by comparing the contour and actual size of the drawing and the positioning point error of the feature object in the environment.

The keyboard is used to control the movement of the robot, so as to reduce the speed of the robot and prevent the robot from slipping due to high speed, which leads to the wrong operation of the odometer information. By controlling the movement of the robot in the experimental environment, the robot applies the G-mapping algorithm to build the first map, as shown in Figure 5.
Establish a plane coordinate system for the first experimental environment, as shown in Figure 5. The characteristic point is point A in the figure, and the actual coordinates of the point are (0.80m, 1.00m). Through image processing of the established map, measure the coordinates of point a in the map, and measure the effect of map establishment.

The image analysis software GIMP in Linux system is used to analyse the map. Each pixel of the map is 0.05m, and the size of the map can be calculated by the number of pixels. In the experimental environment, there is a cuboid whose position is measured to verify the effect of drawing. Each algorithm builds five maps in this environment, compares and analyses them respectively, because there may be accidental errors in the experiment.

The GIMP image processing software is installed in the virtual machine to measure the x-axis and y-axis pixels of the special points in the established PGM map file, as shown in Figure 6. Due to the limitation of lidar resolution, each pixel represents 0.05m, which means that the measurement error is within the range of 0.05m.
The x-axis and y-axis coordinates of the special point A can be obtained from the distance of the map pixels established by image analysis. The results are shown in Table 1 and Table 2 respectively.

|   | 1   | 2   | 3   | 4   | 5   |
|---|-----|-----|-----|-----|-----|
| G-mapping | 0.75m | 0.70m | 0.70m | 0.65m | 0.75m |
| Cartographer | 0.70m | 0.70m | 0.70m | 0.70m | 0.70m |

Table 2. Y-axis coordinate values of special point A under two algorithms.

|   | 1   | 2   | 3   | 4   | 5   |
|---|-----|-----|-----|-----|-----|
| G-mapping | 0.95m | 0.95m | 0.95m | 1.00m | 0.95m |
| Cartographer | 0.95m | 0.95m | 0.95m | 0.95m | 0.90m |

5.1.2. The second experimental environment test. The second experimental environment is the bedroom and storage room. Establish a plane coordinate system for the second experimental environment, as shown in Figure 7. The characteristic point is point B in the map, and the actual coordinates of the point are (0.50m, 0.70m). Through image processing of the established map, measure the coordinates of point B in the map, and measure the effect of map establishment.

According to the method of the last experiment, the x-axis and y-axis coordinates of the special point B can be obtained from the distance of the map pixels established by image analysis. The results are shown in Table 3 and Table 4 respectively.

|   | 1   | 2   | 3   | 4   | 5   |
|---|-----|-----|-----|-----|-----|
| G-mapping | 0.40m | 0.50m | 0.45m | 0.40m | 0.40m |
| Cartographer | 0.45m | 0.45m | 0.45m | 0.45m | 0.40m |

|   | 1   | 2   | 3   | 4   | 5   |
|---|-----|-----|-----|-----|-----|
| G-mapping | 0.60m | 0.65m | 0.60m | 0.60m | 0.65m |
| Cartographer | 0.60m | 0.60m | 0.60m | 0.60m | 0.60m |
5.2. **Experiment of obstacle recognition ability**

This experiment is carried out in the first experimental environment of the mapping and positioning accuracy experiment. In this experimental environment, several kinds of obstacles with different shape and size ratio are placed. The G-mapping algorithm and Cartographer algorithm are used to build the map respectively, and the accuracy of the obstacle contour established by the two algorithms is compared.

In the same obstacle environment, the G-mapping algorithm and the Cartographer algorithm were used to do five experiments respectively, and the maps generated each time were saved. The comparison of maps established by G-mapping algorithm and cartographer algorithm in multi obstacle environment is shown in Figure 8. Through qualitative analysis of several obstacle contours in the two algorithms, the precision of the two algorithms can be compared.

![Figure 8](image)

**Figure 8.** Comparison of the five constructions of the two algorithms.

The number of successful obstacle recognition by the two mapping algorithms is shown in Table 5. It can be seen that the detection effect of Cartographer algorithm is obviously better than that of G-mapping algorithm. However, the effect of the two algorithms on the contour of the cylinder model is not very ideal, and neither of them can completely create a circular contour. The reason may be that the diameter of the cylinder model is small. In the process of robot mapping, the odometer information must be combined. But the odometer information cannot be accurately accumulated, which makes it difficult to establish a complete small-diameter circular contour.

|                  | G-mapping | Cartographer |
|------------------|-----------|--------------|
| ball pen         | 0         | 3            |
| bench leg        | 3         | 5            |
| cuboid model     | 5         | 5            |
| cylinder model   | 5         | 5            |

**Table 5.** The number of obstacles successfully identified.

6. **Conclusions**

In this design, the overall research content is as follows:

The robot model in ROS environment is established, the configuration of various controllers in the simulation environment is realized, and the function package of robot model control is built. The robot URDF modeling language is studied and the robot modeling is completed. The differential controller is
configured and the robot keyboard control node is programmed. The robot model is successfully displayed in the gazebo simulation environment.

Two kinds of slam algorithms are simulated, which are G-mapping algorithm and Cartographer algorithm. After the theoretical analysis of the two algorithms, the function package program of the robot model in the virtual machine is modified and compiled. In the virtual machine, the two algorithms are simulated by gazebo and stage respectively. Through the keyboard control node to control the robot model of the two algorithms, the environment map of the two algorithms is successfully established.

Experiments are designed to compare the mapping effects of different slam optimization algorithms. Through the average and variance analysis of the coordinate values obtained from the multiple measurement results of special points in the experiment, the mapping effects of G-mapping algorithm and Cartographer algorithm are evaluated, which provides experience for unmanned mapping and navigation in ROS environment. The experimental results show that both algorithms can build a map with an error of less than 10 cm in a small area. However, if the experiment is carried out in a large range, the error of Cartographer algorithm is smaller. In the multi obstacle environment, the map built by Cartographer algorithm can recognize the obstacles more accurately.

Acknowledgments
This work was financially supported by National Key Research and Development Project (2019YFD1002400).

References
[1] Finegan, H, Jaffe, S, Leon, A, Lytle, K, & Lewin, G. C. (2019). Development of an Autonomous Agricultural Vehicle to Measure Soil Respiration. 2019 Systems and Information Engineering Design Symposium (SIEDS).
[2] Habibie, N, Nugraha, A. M., Anshori, A. Z., Ma'Sum, M. A., & Jatmiko, W. (2017). Fruit mapping mobile robot on simulated agricultural area in Gazebo simulator using simultaneous localization and mapping (SLAM). 2017 International Symposium on Micro-NanoMechatronics and Human Science (MHS).
[3] Hajjaj, S. S. H, & Sahari, K. S. M. (2017). Establishing remote networks for ros applications via port forwarding: a detailed tutorial. International Journal of Advanced Robotic Systems, 14 (3), 172988141770335.
[4] Hussain, R, & Zeadally, S. (2018). Autonomous cars: research results, issues, and future challenges. Communications Surveys & Tutorials, IEEE.
[5] Luo Hao. (0). Research on error correction method of inertial navigation system based on map matching. (medical interpretation, Beijing University of Technology) (Chinese).
[6] Puškár M, Fabian M, & Kadarova J. (2017)Autonomous vehicle with internal combustion drive based on the homogeneous charge compression ignition technology. International Journal of Advanced Robotic Systems,, 14 (5): 1729881417736896.
[7] Wang Z, Gong L, & Chen Q. (2016)Rapid developing the simulation and control systems for a multifunctional autonomous agricultural robot with ROS. International Conference on Intelligent Robotics and Applications. Springer, Cham: 26 - 39.
[8] Inder, Silva, & Shi. (2019). Learning control policies of driverless vehicles from uav video streams in complex urban environments. Remote Sensing, 11 (23), 2723.