Design of Mobile Robot Based on Cartographer SLAM Algorithm
Qing GAO¹, Hao JIA², Ya LIU² and Xin-cheng TIAN²,*

¹Enterprise Management and Informatization, Weichai Power Co., Ltd., Weifang, China 261061
²School of Control Science and Engineering, Shandong University, Jinan, China 250061
*Corresponding author

Abstract. Aiming at some problems existing in the development and industrialization of intelligent autonomous mobile robot, the SLAM mobile robot platform is designed and developed in this paper. Cartographer is used as the SLAM algorithm to realize the mapping and localization of the robot in the unknown environment. What’s more, the new hardware design architecture is proposed, and the bottom control module is designed to be applied to the autonomous mobile robot. On the one hand, the design can reduce the workload of the non-algorithm part of the main control unit and improve the real-time of mapping. On the other hand, it solves the problem of communication clogging and enhances the compatibility of the system. Finally, an autonomous test platform is built to test the mapping and localization functions of the robot. The results of mapping, computational complexity and robustness of Cartographer SLAM and Hector SLAM algorithms are analyzed.

Introduction
Intelligent mobile robots are of great significance in accelerating the promotion of intelligent upgrading in various industries and upgrading the level of intelligence in various industries. Simultaneous Localization and Mapping (SLAM) is the core technology to realize intelligent mobile robot, so it has very important research value and theoretical significance. In recent years, with the support of relevant policies and the promotion of market forces, intelligent mobile robots have begun to be applied on a small scale in various industries. From unmanned driving, mine detection, to household sweeping robots, SLAM mobile robots have shown their applications in military, industrial production and daily life [1].

SLAM mobile robots have been applied in deterministic environment such as indoor environment. However, in roadways with similar characteristics and under special conditions such as smoke and dust, there are still some problems, such as poor real-time performance and large accumulated errors, which are far from meeting the requirements of rescue and exploration work. Therefore, it is of great significance to develop SLAM mobile robots that can adapt to harsh environments.

Research on Cartographer SLAM Algorithm
Cartographer is a SLAM algorithm based on laser radar sensor introduced by Google in October 2016. This algorithm can generate a real-time raster map with a resolution of 5 cm from the data measured by sensors such as laser rangefinder. Cartographer adopts SLAM theoretical framework based on graph optimization method, which is divided into front-end and back-end parts. The front-end is responsible for data association and closed-loop detection. Data association mainly deals with local data relations, and resolves the matching between the consecutive data frames and the related attitude estimation problems. Closed-loop detection is mainly aimed at judging the matching problem between the current pose of the robot and the pose of the previously visited region by using the data obtained by the sensor. The creation of pose map can be completed through the above two processes. Because of the observation noise of the sensor and the error of scanning matching itself, the pose map obtained by the front-end will be biased, so it needs to be corrected by the back-end graph optimization part. Back-end processing does not directly process the observation data of the sensor,
but only optimizes the pose map created by the front-end, and obtains the maximum likelihood estimation of the pose, that is, the optimal pose sequence [2,3]. The algorithm framework is shown in Fig. 1.

Figure 1. SLAM algorithm framework based on graph optimization.

Pose of autonomous mobile robot can be represented by \( \xi=(\xi_x,\xi_y,\xi_\theta) \), \( \xi_x \) and \( \xi_y \) represent the translations in direction \( x \) and \( y \), respectively. \( \xi_\theta \) represents the amount of rotation in a two-dimensional plane. The data measured by the lidar sensor is recorded as \( H = \{h_k\}_{k=1,...,K} \in \mathbb{R}^2 \).

The initial laser point is 0 and satisfies the requirement of \( 0 \in \mathbb{R}^2 \). The pose transformation from the scanned data frame of lidar to the submap is recorded as \( T_\xi \), and can be mapped to a submap coordinate system by Eq. 1.

\[
T_\xi p = \begin{pmatrix}
\cos \xi_\theta & -\sin \xi_\theta \\
\sin \xi_\theta & \cos \xi_\theta
\end{pmatrix} p + \begin{pmatrix}
\xi_x \\
\xi_y
\end{pmatrix}
\]

(1)

Continuous scanning lidar data frame can generate a submap, which uses probability grid to express the map model. When the new scanned data is inserted into the probabilistic grid, the state of the grid will be calculated. Each grid has two states: hit and miss. If the grid is hit, the adjacent grids are inserted into the hit set, and all relevant points on the connection line between the scan center and the scan point are added to the lost set [4]. Set a probability value for the grid that has not been observed before, and update the probability of the observed grid according to Eq. 3.

\[
\text{odds}(p) = \frac{p}{1-p}
\]

(2)

\[M_{\text{new}}(x) = \text{clamp}(\text{odds}^{-1}(\text{odds}(M_{\text{old}}(x)) \cdot \text{odds}(p_{\text{hit}))))
\]

(3)

Before inserting the laser scanning frame into the submap, it is necessary to optimize the pose of the scanning frame and the current submap by Ceres Solver solver, so that the above problem can be transformed into solving the non-linear least squares problem.

\[
\arg \min_{\xi} \sum_{k=1}^{K} (1 - M_{\text{smooth}}(T_\xi h_k))^2
\]

(4)

Because the scanning frame of lidar only matches the current submap, and the environment map is composed of a series of submaps, so there will be accumulated errors. Cartographer algorithm optimizes the pose of all the lidar data frames and submaps by Sparse Pose Adjustment (SPA). The pose of lidar data frames inserted into submaps are cached into memory for closed-loop detection. All scanning frames and submaps are used for closed-loop detection when submaps do not change. The
mathematical expression of the optimization problem constructed by Cartographer algorithm using sparse pose adjustment method is as follows:

$$\arg\min_{\Xi, \Xi'} \frac{1}{2} \sum_{ij} \rho(E^2(\xi_i^m, \xi_j^s; \sum_{ij}, \xi_{ij}))$$

$$\Xi^m = \{\xi_i^m\}_{i=1,\ldots,m}$$ and $$\Xi^s = \{\xi_j^s\}_{j=1,\ldots,n}$$ represent the pose of submap and scanning frame respectively under certain constraints. $$\xi_{ij}$$ represents the matching position of scanning frame $$j$$ in the submap and constitutes optimization constraints together with its related covariance matrix $$\sum_{ij}$$. The cost function of this constraint is expressed by residual and can be calculated by Eq. 6.

$$E^2(\xi_i^m, \xi_j^s; \sum_{ij}, \xi_{ij}) = e(\xi_i^m, \xi_j^s; \xi_{ij})^T \sum_{ij}^{-1} e(\xi_i^m, \xi_j^s; \xi_{ij}),$$

$$e(\xi_i^m, \xi_j^s; \xi_{ij}) = \xi_{ij} - \left( R_{\xi_i^m}^{-1}(t_{\xi_i^m} - t_{\xi_j^s}) \right)_{\xi_{ij}, \theta = \theta_{ij}}$$

In addition, the Cartographer algorithm also uses branch and bound scan matching algorithm to accelerate the process of closed-loop detection and relative pose calculation.

$$\xi^* = \arg\max_{\xi \in \mathcal{W}} \sum_{k=1}^{K} M_{\text{nearest}}(T_{\xi} h_k)$$

$$W$$ represents the search window and $$M_{\text{nearest}}$$ is the extension of $$M$$ function in the previous section. Identify a window around a new grid, and determine the maximum range of the point set by constantly modifying the angle increment $$\delta_\theta$$ and the maximum range $$d_{\text{max}}$$ of the sensor [5]. Eq. 8 and Eq. 9 can be derived from the Pythagorean Theorem.

$$d_{\text{max}} = \max_{k=1,\ldots,K} \| h_k \| $$

$$\delta_\theta = \arccos(1 - \frac{r^2}{2d_{\text{max}}^2})$$

The integer step length is calculated by the size of the search window so that it can cover the entire search window.

$$w_x = \left[ \begin{array}{c} W_x \ \ r \\ r \end{array} \right], \quad w_y = \left[ \begin{array}{c} W_y \ \ r \\ r \end{array} \right], \quad w_\theta = \left[ \begin{array}{c} W_\theta \ \ \delta_\theta \end{array} \right].$$

A finite set of search windows $$W$$ is formed with the estimation of pose $$\xi_0$$ as the center.

$$\widetilde{W} = \{-w_x, \ldots, w_x\} \times \{-w_y, \ldots, w_y\} \times \{-w_\theta, \ldots, w_\theta\}$$

$$W = \left\{ \xi_0 + (r_j, r_j, \delta) : (j, j, j, j) \in \widetilde{W} \right\}$$

Branch and bound method can efficiently calculate the value of $$\xi^*$$, but the selection speed of search window size is too slow. Therefore, the branch-and-bound scan matching algorithm based on depth-first search is used to optimize the algorithm [6].
**Design of Hardware System**

**Architecture of System Hardware**

In this paper, the Jetson TX1 embedded platform of Nvidia Corporation is chosen as the main control board [7]. The data collected by lidar is sent to SLAM system through Ethernet interface. SLAM system runs on Nvidia TX1 board. The bottom control module upload the motion state and pose information of the robot to the main control board. The path planning system transfers the calculated robot speed to the chassis drive system through the bottom control module. Thus, the control of mobile robot platform is realized. At the same time, the Cartographer algorithm running on SLAM system sends incremental maps and robot pose to remote industrial computer via WiFi or Ethernet for real-time display. The overall scheme of the hardware is shown in Fig. 2.

![Hardware architecture of SLAM robot platform.](image)

Figure 2. Hardware architecture of SLAM robot platform.

After the above work, the whole hardware system is built. The physical and 3D model of SLAM mobile robot platform is shown in Fig. 3.

![Physical and 3D models of SLAM mobile robot platform.](image)

Figure 3. Physical and 3D models of SLAM mobile robot platform.

**Design of Software System**

**Architecture of System Software**

The system software design mainly includes three parts: the bottom control system, system peripheral drive system and the SLAM system. The bottom control system is developed under the integrated environment of MDK 5 and interacts with SLAM system by writing embedded programs. Driver system of system peripheral and SLAM system are designed and developed based on ROS.
Software Design of SLAM System Based on ROS

The software design of SLAM system mainly completes the transplantation of Cartographer algorithm to the self-built SLAM mobile robot platform. Cartographer algorithm is mainly divided into two parts: front-end and back-end. The front-end mainly completes the work of establishing submaps. A certain number of submaps can be used as basic units to form closed-loop detection. Finally, the process of graph optimization is accelerated by branch and bound algorithm. The program implementation process is as follows: First, the function AdHorizonLaserFan() of the local_trajectory_builder is called. This function calculates the robot pose predicted by the pose interpolator as the initial pose to the real-time scanner matcher. Then, the attitude optimized by the real-time scanner matcher is used as the initial pose of the Ceres solver. Finally, scan-to-map is implemented through the Ceres solver to get the final result, and the pose interpolator is updated.

Experimental Verification

Test Environment

In order to test and verify the SLAM mobile robot in multi-dimension, the 2nd floor of Innovation Building of Shandong University is chosen as the test environment. The test scene is about 76 meters long and 38.5 meters wide, and the building area is about 2926 square meters. Because of the large scale of the scene tested and the variety of ground materials, the validity of SLAM algorithm can be tested comprehensively. The test scenario for mobile robots is shown in Fig. 4.

![Figure 4. The test scenario for mobile robots.](image)

Test Scheme

Firstly, the robot is guaranteed to run normally in the whole test environment, and the robustness of the two algorithms is tested and compared by manual operation to make the robot rotate at high speed until its rotation acceleration is too large. In order to ensure that the two algorithms run under the same experimental conditions, the Rosbag function package is used to record the sensor data during the experiment, and then the two SLAM algorithms are run through the Jetson TX1 main control board. Finally, Hector algorithm and Cartographer algorithm are used to build environment maps of test scenes. In order to evaluate the mapping accuracy of the two algorithms, the building structure in the measurement environment is measured, and the measured values are compared with those in RVIZ.

Experimental Results and Analysis

According to the above experimental scheme, the mapping results of Cartographer SLAM algorithm and Hector SLAM algorithm are obtained, as shown in Fig. 5. Fig. 5(a) is a local enlargement of the trajectory of the robot when it rotates at high speed, so the trajectory at Fig. 5(a) is confused. It is the high-speed rotation that results in the overlap of rotation in Hector SLAM algorithm, as shown in Fig. 5(d) and Fig. 5(e). Under the same experimental conditions, the Cartographer SLAM algorithm has a better mapping results, and the boundary of obstacles on the environmental map is clearer, without burrs. The real-world view of the data acquisition process of the robot is shown in Fig. 6.
Figure 5. Comparison of mapping results between Cartographer algorithm and Hector algorithm.

Figure 6. The real-world view of the data acquisition process of the robot.

Conclusion
This paper mainly completes the following tasks based on mobile robot platform. Firstly, the framework of graph optimization SLAM algorithm is introduced, and the Cartographer algorithm based on graph optimization method is deduced theoretically. Secondly, the hardware design architecture of the system is introduced, and the new hardware architecture is proposed, which enhances the compatibility of the system. Then the software design of the bottom control module and
the Cartographer SLAM algorithm is introduced in detail, so as to complete the task of building the experimental platform. Finally, through the contrast experiments between Hector SLAM algorithm and Cartographer SLAM algorithm, it is proved that Cartographer SLAM algorithm performs better in mapping accuracy, robustness and real-time performance.

Acknowledgement

This research was financially supported by the Major Innovation Projects in Shandong Province under Grant No.2017CXGC0601.

References

[1] Siegwart R, Nourbakhsh I R, Scaramuzza D, et al. Introduction to autonomous mobile robots[M]. MIT press, 2011.

[2] Grisetti G, Kummerle R, Stachniss C, et al. A tutorial on graph-based SLAM[J]. IEEE Intelligent Transportation Systems Magazine, 2010, 2(4): 31-43.

[3] Krinkin K, Filatov A, yom Filatov A, et al. Evaluation of Modern Laser Based Indoor SLAM Algorithms[C]//2018 22nd Conference of Open Innovations Association (FRUCT). IEEE, 2018: 101-106.

[4] Xu B, Liu Z, Fu Y, et al. Research of cartographer laser SLAM algorithm[C]//LIDAR Imaging Detection and Target Recognition 2017. International Society for Optics and Photonics, 2017, 10605: 1060509.

[5] Kohlbrecher S, Von Stryk O, Meyer J, et al. A flexible and scalable slam system with full 3d motion estimation[C]//2011 IEEE International Symposium on Safety, Security, and Rescue Robotics. IEEE, 2011: 155-160.

[6] Hess W, Kohler D, Rapp H, et al. Real-time loop closure in 2D LIDAR SLAM[C]//2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016: 1271-1278.

[7] Nikolskiy V P, Stegailov V V, Vecher V S. Efficiency of the Tegra K1 and X1 systems-on-chip for classical molecular dynamics[C]//2016 International Conference on High Performance Computing & Simulation (HPCS). IEEE, 2016: 682-689.