Journal of Intelligent & Robotic Systems (2024) 110:144
https://doi.org/10.1007/s10846-024-02123-1

REGULAR PAPER

An Optimization on 2D-SLAM Map Construction Algorithm Based on LiDAR

Zhuoran Li[1] · Kazem Chamran[1] · Mustafa Muwafak Alobaedy[1] · Muhammad Aman Sheikh[2] · Tahir Siddiqui[3] · Abdul Ahad[4,5]

Received: 27 December 2023 / Accepted: 28 May 2024 / Published online: 28 September 2024

Abstract
When a mobile robot moves in an unknown environment, the emergence of Simultaneous Localization and Mapping (SLAM) technology becomes crucial for accurately perceiving its surroundings and determining its position in the environment. SLAM technology successfully addresses the issues of low localization accuracy and inadequate real-time performance of traditional mobile robots. In this paper, the Robot Operating System (ROS) robot system is used as a research platform for the 2D laser SLAM problem based on the scan matching method. The study investigates the following aspects: enhancing the scan matching process of laser SLAM through the utilization of the Levenberg–Marquardt (LM) method; improving the optimization map by exploring the traditional Hector-SLAM algorithm and 2D-SDF-SLAM algorithm, and employing the Weighted Signed Distance Function (WSDF) map for map enhancement and optimization; proposing a method for enhanced relocation using the Cartographer algorithm; establishing the experimental environment and conducting experiments utilizing the ROS robot system. Comparing and analyzing the improved SLAM method with the traditional SLAM method, the experiment proves that the improved SLAM method outperforms in terms of localization and mapping accuracy. The research in this paper offers a robust solution to the challenge of localizing and mapping mobile robots in unfamiliar environments, making a significant contribution to the advancement of intelligent mobile robot technology.

Keywords Simultaneous localization and Mapping · 2D LIDAR · Scan Matching · Map Optimization

1 Introduction

In the study of mobile robots, the issues of localization and map construction are crucial. If there are errors in the positioning and map construction, the mobile robot will encounter difficulties navigating autonomously using the constructed map and may fail to complete the task accurately. If the mobile robot is outdoors in an open area, it can use GPS (Global Positioning System) to obtain its absolute position. However, in indoor and covered outdoor environments, GPS will not work effectively. Therefore, in this case, it is expected that the mobile robot will be able to determine its own position based on the sensors it carries, without relying on the external environment. The positioning process of the mobile robot relies on the environment map, and the establishment of the environment map is based on the accurate positioning of the mobile robot. Therefore, the two problems of positioning and map establishment are not isolated; they are intrinsically linked. If these two problems are combined into one, it is called SLAM.

✉ Kazem Chamran
kazem.charman@city.edu.my

✉ Muhammad Aman Sheikh
msheikh@cardiffmet.ac.uk

Zhuoran Li
994512621@qq.com

Mustafa Muwafak Alobaedy
new.technology@hotmail.com

Abdul Ahad
Ahad9388@nwpu.edu.cn

1 Faculty of Information Technology, City University Malaysia, 46100 Petaling Jaya, Kuala Lumpur, Malaysia

2 Cardiff School of Technologies, Cardiff Metropolitan University, Cardiff, UK

3 University of Turku, Fi-20014 Turun Yliopisto, Finland

4 School of Software, Northwestern Polytechnical University, Xi'an, Shaanxi, People's Republic of China

5 Department of Electronics and Communication Engineering, Istanbul Technical University (ITU), 34467 Istanbul, Turkey

SLAM has been a popular research problem in the field of mobile robotics since it was first proposed in the literature by R. Smith, M. Self, and P. Cheeseman [1]. The SLAM problem refers to a scenario where a subject is equipped with specific sensors that construct a map of the environment and simultaneously determine its position in the map while moving, without prior knowledge of the environment. Solving the SLAM problem also truly enables the autonomy of mobile robots; therefore, SLAM technology is the key technology in mobile robots.

For instance, in the context of mobile robot navigation and control, SLAM technology plays a crucial role. It provides crucial support for tasks such as path planning and obstacle avoidance by utilizing sensor data for precise positioning and environment mapping. Combining SLAM technology with ILC (Iterative Learning Control) schemes enables the system to maintain accurate positioning even in dynamic environments or when dealing with uncertainties, while generating maps for adaptive control strategy adjustment [2]. SLAM algorithms can detect changes in the environment and unexpected obstacles. When combined with ILC's fault estimation and compensation techniques, these algorithms enable the system to respond promptly to faults, adapt to uncertainties and disturbances, and ensure continuous operation and performance [3, 4]. Meanwhile, the rich sensor data generated by SLAM technology can be utilized for data-driven control strategies, such as reinforcement learning-based methods or adaptive control techniques, to enhance control performance and autonomy [5].

In this article, the SLAM problem itself will be investigated with the aim of enhancing the performance and reliability of mobile robotic systems in various applications.

1.1 Problem Statements

SLAM is an important technology for mobile robots, enabling them to navigate and create accurate maps of their environment in real time. Although SLAM algorithms have made significant progress in recent years, there is still significant scope for optimization to improve their accuracy and efficiency. In this paper, potential areas for optimizing SLAM algorithms, strategies to enhance performance, and methods to evaluate their effectiveness will be explored.

(1) SLAM front-end issues

Occupying rasterized maps is one of the SLAM methods for building maps, which will be utilized in this paper. The raster size in this method is a key factor that influences the accuracy of the map. Therefore, this paper focuses on optimizing the accuracy of occupied rasterized maps in SLAM front-end map building.

(2) SLAM back-end problem

When the sensor scans the surface data, a certain amount of error will appear. Over time, this error accumulation increases. In response to the aforementioned issue, this paper builds upon the Cartographer algorithm developed by the Google team to further optimize and enhance its performance.

The research questions of this paper are specified below:

(1) Which SLAM algorithms can be optimized?
(2) How can SLAM algorithms be optimized to improve the accuracy of maps constructed by mobile robots?
(3) How can we test the optimized SLAM algorithm and evaluate its performance?

Optimizing SLAM algorithms is an ongoing task because it directly impacts the navigation and map-building capabilities of mobile robots. Through relevant SLAM algorithm optimization, map analysis, and real-world testing, improvements are ensured to meet the requirements of a wide range of applications. This is crucial for the success of various technologies, such as self-driving vehicles and warehouse robots, where SLAM plays a critical role.

1.2 Research Objectives

With the widespread use of mobile robots, the significance of SLAM techniques is increasing. However, while SLAM algorithms are constantly evolving, there is still much potential for improvement. This paper will focus on SLAM algorithms and how they can be optimized to enhance the accuracy of constructed maps.

The research objectives of this paper address the SLAM building problem as follows:

(1) To begin, it involves exploring SLAM algorithms that can be improved and analyzed.
(2) To enhance the accuracy of the constructed maps, it is imperative to optimize the SLAM algorithm.
(3) To assess the optimized algorithm, a comparative analysis is conducted by comparing it with the pre-optimized version.

By enhancing SLAM algorithms, more precise and dependable map construction can be achieved. This improvement aids in enhancing the performance of mobile robots in a variety of applications, including autonomous driving, UAV (Unmanned Aerial Vehicle) navigation, and warehouse automation. Therefore, this paper will explore how various aspects of the SLAM algorithm can be optimized to enhance the accuracy of map construction.

To evaluate the optimized SLAM algorithm, this paper will use a comparative analysis to contrast it with the unoptimized version. This comparison will help the study quantify the effect of optimization and determine the magnitude of improvements and where the greatest gains have been achieved. The quantitative comparison will offer a clear understanding of the benefits of the optimized SLAM algorithm in terms of accuracy, efficiency, and robustness. This will serve as a solid reference for future SLAM research and applications.

This series of studies aims to offer stronger and more reliable support for SLAM technology for mobile robots and autonomous systems, laying a solid foundation for achieving greater success in progressively complex environments. Whether in the field of scientific research or in commercial applications, the continuous improvement of SLAM technology will propel mobile robots forward, fostering innovation and enhancing convenience in people's lives and work.

2 Literature Review

2.1 Applications and Development

The growth of mobile robots has led to an increased demand for SLAM knowledge; however, the SLAM problem has been a challenge. To address this problem, some innovative solutions have been proposed, such as the Enhanced Localisation Solution (ELS) in SLAM localisation. This solution combines standard localisation techniques with machine learning techniques [6]. Mobile cloud computing can upload SLAM data to the cloud for mobile services. A request state-aware resource allocation technique has been proposed to enhance the mobile user experience using intelligent resource capacity prediction techniques [7]. Fog computing is used to enhance the efficiency of cloud computing. A population-based multi-objective meta-heuristic optimizer is proposed to facilitate resource allocation and scheduling in fog environments. This further enhances the cloud experience of SLAM technology for mobile users [8].

The semantic location prediction problem in SLAM research focuses on semantic descriptions of locations. Researchers utilize mobile phone data and machine learning algorithms to identify user-visited locations and establish relationships between locations and activities [9]. A new approach based on a fuzzy logic system in SLAM research was proposed to maximize the area coverage of tiling robots, which offers new insights for the advancement of floor cleaning robots [10].

Research that integrates SLAM techniques, machine learning, and robotics aims to enhance agricultural productivity by improving autonomous spatial localization and vision mapping techniques. This enables image processing and data analysis in agricultural environments [11]. In the military domain, path planning is a crucial procedure for Unmanned Combat Aerial Vehicles (UAVs). The map building function of SLAM technology is a prerequisite for path planning. Therefore, an enhanced symbiont search algorithm has been proposed to plan UAV paths more efficiently [12]. SLAM algorithms are introduced for reinforcement learning hardware implementation, specifically an FPGA (Field Programmable Gate Array) proximal policy optimization algorithm for designing and implementing a novel hardware architecture for control theory benchmarking [13].

To enhance road safety measures. Data mining and machine learning techniques are used to determine accident severity and propose prediction rules based on SLAM techniques [14]. In automated driving, SLAM techniques combined with Internet of Things (IoT) technologies are employed for accurate risk assessment and operational testing protocols [15]. Meanwhile, SLAM technology can be applied to edge computing and deep learning anomaly detection methods for intelligent motorway monitoring networks [16]. In SLAM research, a security framework based on a lightweight authentication scheme has been proposed for securing vehicle-to-vehicle communications [17]. In addition, the development of a curved lane detection algorithm based on a Bayesian framework in SLAM algorithms has significantly enhanced the efficiency of road marking detection [18]. Finally, to enhance the efficiency of transport infrastructure management, service-based cyberinfrastructures have been developed. These cyberinfrastructures can integrate multiple data sources, such as SLAM, to achieve the goal of ambient intelligence [19].

The research on these SLAM problems offers crucial insights and methods for addressing challenges in user localization, traffic safety, and environmental intelligence.

2.2 SLAM Classification

Depending on the sensors carried by the mobile robot, SLAM can be categorized into laser SLAM and vision SLAM [20, 21]. To better adapt to the environment, SLAM with multi-sensor fusion and SLAM combined with deep learning techniques are also now available [22, 23].

Laser SLAM systems typically incorporate either 2D or 3D LiDAR, and the selection depends on the complexity of the specific environment. In complex and variable environments, 3D LiDAR may be more appropriate, while in relatively simple environments, 2D LiDAR may be sufficient [24, 25].

The monocular camera has only one camera that records the two-dimensional image information of the environment and uses the principles of visual geometry to deduce the pose changes of the robot [26]. Binocular cameras have two cameras, similar to human eyes, and can provide more accurate distance estimations [27]. Depth cameras use infrared

sensor technology, similar to lidar, to measure distance by emitting and receiving light [28].

Wang et al. proposed a SLAM algorithm for mobile robots based on LiDAR and binocular vision [29]. The algorithm enhances the RBPF-SLAM method by integrating binocular vision, LiDAR, and odometer data to accomplish robot localization and navigation through information fusion. At the same time, researchers like Pan proposed a SLAM algorithm that integrates a monocular camera and an IMU, making full use of the accurate measurement of angular velocity and acceleration by the IMU, while also utilizing the image information collected by the camera [30]. Researchers like Yu have introduced magnetometer data based on the fusion of monocular vision and IMU. This data is integrated into the monocular vision SLAM algorithm to address the trajectory drift issue under pure rotation and enhance accuracy [31].

Deep learning was first proposed in 2006. It simulates the human learning process by constructing complex neural network models to replicate the structure of the human brain. These models are trained with large-scale data.

2.3 SLAM Framework

The 2D laser SLAM framework mainly includes front-end scan matching, back-end optimization, and map construction [1]. The front-end includes sensor data acquisition and scan matching, while the back-end includes optimization and loopback detection. Front-end scan matching utilizes the relationship between two consecutive frames of laser sensor scan data to estimate localization. Back-end optimization is utilized to minimize the cumulative error that occurs after scan matching. Loopback detection is employed to eliminate the cumulative error and reduce the map drift phenomenon by verifying if the estimated value of the current position matches the estimated value of the historical position. The map building module is used to generate environmental map information. The 2D laser SLAM framework is shown in Fig. 1.

2.4 Front-end Scanning Matching

The Iterative Closest Point (ICP) algorithm is one of the most widely used algorithms today [32]. This algorithm utilizes the minimization of the Euclidean distance between two frames of matching point clouds to retrieve the transformation information regarding the relative position. This algorithm also has a variant called PLICP (Point-To-Line ICP). Compared with the ICP algorithm, PLICP improves accuracy and speeds up the search for closed solutions [33].

Feature-based scan matching extracts implicit functions as features from scan data and works well on a closed graphical background [25]. Zhao et al. proposed a matching method based on quadratic curve characteristics. This method provides ample corner point features [34].

Scan matching algorithms based on mathematical properties are one of the most common approaches in SLAM. It utilizes mathematical models and optimization algorithms to analyze and align laser scanning data for estimating the robot's position and creating environment maps [35].

Correlation-based scan matching is a method in laser SLAM. This method involves searching the specified space in the established model and then calculating the position of the points to determine the variance [36].

2.5 Back-end Optimization

Filter-based two-dimensional laser SLAM method uses a recursive Bayesian estimation approach to construct incremental maps and achieve accurate positioning [37].

Graph optimization-based approach is a high-precision navigation and map-building technique widely used in the field of mobile robotics [38]. The method utilizes all the observed information to evaluate the current state of the robot and create a map.

2.6 Loopback Detection

Frame-to-frame loopback detection utilizes correlation scanning matching to compare the similarity between two frames

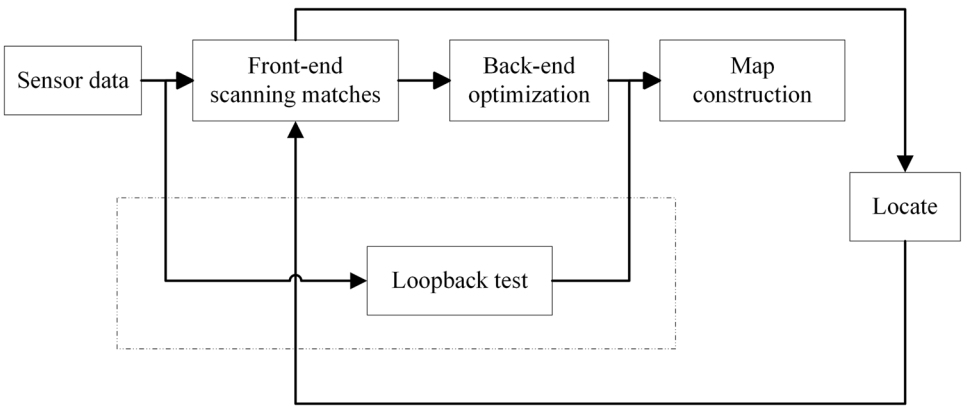


Fig. 1 2D laser SLAM framework

of LiDAR data to determine whether they form a loopback or not [36].

A representative algorithm for loopback detection of frames and subgraphs is the Cartographer algorithm. This algorithm reduces the redundant data generated during the detection process as it is a frame-to-subgraph detection method, thus improving the efficiency of loopback detection [39].

Olson proposed a multi-resolution scan matching method, which can reduce the accumulation of local errors but increases the amount of calculation [40]. Himetedt proposed a histogram-based matching method [41]. Nieto extracts feature points from scanned data and combines them with deep learning methods for matching [42]. Yin proposed a learning method based on twin neural networks for semi-automatic representation of LiDAR point clouds. KD (K-Dimensional) trees are then established to accelerate loopback testing [43].

2.7 Mainstream 2D Laser SLAM Algorithms

In 2002, Montebello and other researchers proposed the Fast-SLAM 1.0 algorithm. Fast-SLAM 1.0 incorporates the features of Kalman filtering for roadmap position estimation and particle filtering for system position estimation [44]. Fast-SLAM 1.0 utilizes the posterior probability of the robot's position and waypoints to predict the complete robot path's posterior probability through particle filtering. The traditional Extended Kalman Filter (EKF)-based SLAM suffers from two classical problems:

(1) Complexity increases dramatically as the number of waymarked points increases.
(2) The entire algorithm fails when incorrect observations occur.

In contrast, Fast-SLAM can effectively solve these problems. Therefore, the new Fast-SLAM 2.0 algorithm proposes an important method for function calculation to address this problem [45]. In 2007, Grisette proposed the Gmapping method to further optimize Fast-SLAM. Gmapping is currently one of the most widely used algorithms in the field of indoor robot two-dimensional laser SLAM. The algorithm uses RBPF (Rao-Blackwellized Particle Filters) to solve the SLAM problem, i.e., it locates first and builds the map later. In 2007, Grisette proposed the Gmapping method to further optimize Fast-SLAM. Gmapping is currently one of the most widely used algorithms in the field of indoor robot two-dimensional laser SLAM [46]. The algorithm uses RBPF (Rao-Blackwellised Particle Filters) to solve the SLAM problem, i.e., locate first and build the map later. Introduced in 2010, Core-SLAM is a lightweight SLAM method consisting of only 200 lines of code, which results in a small performance overhead [47].

In 1999, Gutmann et al. proposed a similar graph optimization framework, but ignored the sparsity of the matrix [48]. In 2010, Konolige et al. proposed the Karto-SLAM algorithm, which is the first open-source graph optimization-based algorithm that utilizes a highly optimized and non-iterative square root method decomposition to achieve a sparse decoupling solution [49]. General graph optimization methods typically necessitate an initial guess, leading to multiple iterations to determine the minimum of the local cost function. In 2011, Carlone et al. proposed an approximate linearized graph optimization method called Lago-SLAM, which does not require [50]. In 2016, Google introduced Cartographer, a laser SLAM algorithm based on graph optimization [39]. The algorithm is optimized for front-end data extraction and data processing in local SLAM to obtain more accurate sub-maps. In 2020, Li and other researchers proposed a novel map representation based on a regionalized Gaussian Processes (GP) map reconstruction algorithm using GP regression [51]. This approach allows for the use of concise mathematical techniques for position estimation and map updating.

In 2011, researchers such as Kohlbrecher proposed the Hector-SLAM algorithm, which is unique in that it requires no back-end optimization part and no additional sensor support, making it suitable for LiDAR systems that require a high update frequency and low measurement noise [52].

3 Methodology

3.1 SLAM front-end Optimization

In SLAM methods that rely on occupancy rasterized maps, the raster size not only constrains the map's accuracy but also affects the algorithm's memory usage and the accuracy of robot localization. To address this issue, this chapter offers a comprehensive study of Hector-SLAM and SDF (Signed Distance Field) maps, and suggests an enhanced WSDF map derived from them. The WSDF map combines a more detailed raster representation with weight information to enhance map accuracy and reduce memory consumption, thereby optimizing the performance of the SLAM system. Finally, the LM method is utilized to solve the scan matching problem. The positioning accuracy is enhanced by optimizing the robot's pose estimation.

3.1.1 Hector-SLAM Algorithm

The Hector-SLAM algorithm is a method that does not rely on odometer data but instead heavily depends on high-resolution and high-frequency LiDAR [52]. The map model utilized in this algorithm is based on a raster map and is

localized through scan matching. During scan matching, the Gauss–Newton method is used to obtain the optimal solution for scan matching and achieve self-localization. In order to prevent the Gauss–Newton method from getting trapped in local minima while solving the optimal positional transformation, the maps obtained from scanning matching are stored in multiple maps with varying resolutions and organized hierarchically based on accuracy, from low to high. When the Gauss–Newton method is used, the optimal solution is obtained by initiating the search from the map with the smallest resolution. Subsequently, the optimal solution of the map in this layer serves as the initial position estimation for the map in the preceding layer. The search progresses layer by layer in this manner.

This algorithm is more flexible, easy to deploy, and consumes less energy than other algorithms. The main goal of the Hector-SLAM algorithm is to achieve precise positioning while reducing the computational demands on the personal computer (PC). The flowchart of the Hector-SLAM algorithm is shown in Fig. 2:

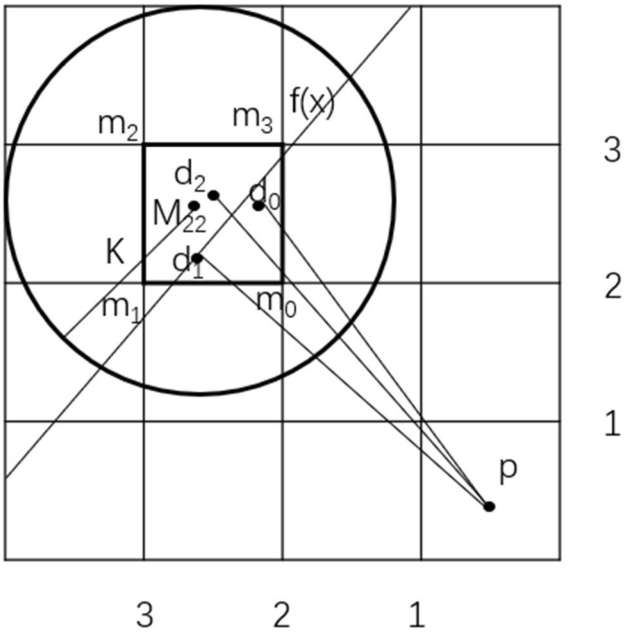


Fig. 3 SDF map principle as shown in Figure

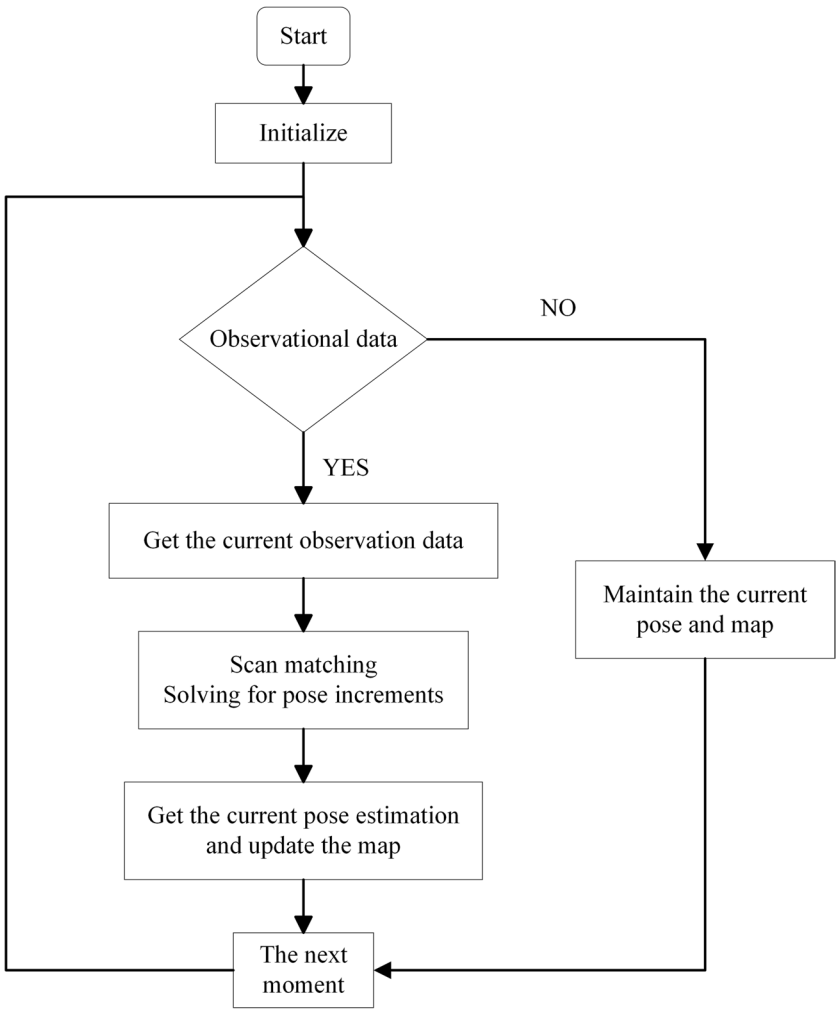


Fig. 2 Flowchart of Hector-SLAM algorithm

3.1.2 SDF Maps

In SDF maps, regression lines are used to describe the contours of detected objects, which are subsequently used to update the surrounding raster [53]. As shown in Fig. 3. SDF map principle as shown in, the robot is located at point P in the figure, and three of the LiDAR's scanning points are located in the grid M_{22}, which is d_1, d_2, d_3. The outline of the obstacle is approximated by fitting a straight line $f(x)$ based on the three scan points described above. The rasters within a radius of K centred on the raster M_{22} are updated and when the distance from the robot position P to the raster is greater than its distance to the straight line $f(x)$, the raster update is negative, and vice versa the raster update is positive. The contours of the obstacles are then described in terms of subgrid size accuracy using the SDF and always averaged over all measurements. When there are not enough available scan points in a raster, scan points located in neighbouring rasters are used for regression. Also, to resolve the errors on the x and y axes, a Deming regression was used to fit a straight line $f(x)$.

3.1.3 WSDF

In this paper, SDF maps are improved and introduced into raster maps, which use the average of all time measurements when updating object contours using the signed distance function. This method reduces the error caused by Gaussian sensor noise. However, at each scan update, not only the raster where the scanning point is located is updated, but also other rasters within a radius of the scanning point, so averaging over all raster updates introduces noise into the map. In this paper, we introduce the idea of using weighted distances in TSDF (Truncated Signed Distance Function) maps in 3D reconstruction, and weight each update to make the update values smoother and more efficient [54].

Assume that the updated value of a raster is d and the current weight of this raster is w. For the weight, the computation rule is shown in Eq. (1) below:

$$W = \left[\frac{\frac{1}{d^2} - \frac{1}{d_{\max}^2}}{\frac{1}{d_{\min}^2} - \frac{1}{d_{\max}^2}} \right] \cdot W_{\max} \tag{1}$$

where d_{min} and d_{max} are the upper and lower limits of the distance d, and W_{max} is the maximum value of the weights.

The size of the currently calculated weights determines whether to update or not, and the decision rule is as follows: if $w \geq r\% \cdot w_{last\_max}$, then update; otherwise, no update. Where w_{last_max} is the weight value of the most recent maximum update. The above rule integrates the value of the distance close to the maximum weight of the last update affected by noise.

The rule for updating at each time is shown in Eq. (2).

$$\begin{cases} D_{t+1} = \frac{W_t D_t + w_{t+1} d_{t+1}}{W_t + w_{t+1}} \\ W_{t+1} = W_t + w_{t+1} \end{cases} \tag{2}$$

In order to prevent W from reaching W_{max} quickly, w_{t+1} is usually made to.

3.1.4 LM Methodology

Using the Gauss–Newton method can suffer from the issue of inaccurately approximating local values and potential non-convergence. Therefore, in this paper, we use the LM method to solve the scan matching problem.

The LM method utilizes the concept of a trust region. Assume a reliable maximum displacement range (trust domain) centered on the solution. Find the optimal value of the approximation function of the objective function within the trust domain and solve for the optimal step size based on this optimal value. If the step results in a decrease in the objective function, then the step is considered desirable. Proceed with obtaining the step to continue to the next iteration. Otherwise, if the step does not lead to a decrease in the objective function, it is considered undesirable. In this case, the results of this iteration will be discarded, the range of the trust domain will be reduced, and the optimal value will be recalculated based on the new trust domain [55].

For SDF maps, scan matching is expressed as solving for the bitmap increment that minimizes the map value, the expression of which changes to Eq. (3). For the LM method, the solution to Eq. (3) is Eq. (4).

$$q^* = \underset{q}{argmin} \sum_{i=1}^{n} \left[M(S_i(q)) \right]^2 \tag{3}$$

$$\Delta q = (H + \mu I)^{-1} g \tag{4}$$

where $H = J^T J$, $g = J^T f(x)$, and H are Hessian matrices [56], J is the Jacobi matrix [57], I is the unit matrix, and μ is the penalty factor. The penalty factor μ is used to control the step size of each iteration. The value of μ has the following meaning:

When $\mu > 0$, $H + \mu I$ is a positive definite matrix, ensuring that Δx is the descent direction.

(1) If the value of μ is large, then Δq approaches the direction of gradient descent.

This is shown in Eq. (5).

$$\Delta q \approx -\frac{1}{\mu} g \tag{5}$$

(2) If the value of μ is small, then Δq is close to the direction of the Gauss–Newton method. shown in Eq. (6).

$$\Delta q \approx -H^{-1} \cdot g \tag{6}$$

Thus, the penalty factor affects both the direction of descent and the size of the descent step.

The quality of the step is determined by the ratio q of the actual decline ΔF of the objective function this time and its predicted decline ΔL this time.

(1) This iteration is valid when $q > 0$ and the penalty factor is reduced. shown in Eq. (7).

$$\begin{cases} \mu = \mu \cdot max\left\{\frac{1}{3}, 1-(2q-1)^3\right\} \\ v = 2 \end{cases} \tag{7}$$

(2) When $q \leq 0$, this iteration is invalid, and the penalty factor needs to be increased. shown in Eq. (8).

$$\begin{cases} \mu = \mu \cdot v \\ v = 2 \cdot v \end{cases} \tag{8}$$

v in Eq. is the parameter vector.

After using the LM method to solve the scanning matching to get the position increment at the current moment, the position $q_t = q_{t+1} + \Delta q$ of the robot at the current moment t in the world coordinate system of the environment map can be obtained, and then the data z_i can be scanned by the LiDAR at the moment t and then be used to build the map.

3.1.5 Improving the SLAM Algorithm

This section describes the final improved laser SLAM method based on the previously proposed WSDF maps by integrating the LM method into the WSDF maps during the scan matching phase. The algorithm is presented in Table 1.

The improved SLAM method's overall flow is briefly described below:

(1) Initialize the robot's position and the environment map, and set the relevant parameters.
(2) Acquire the laser data at the moment and convert it to the world coordinate system based on the robot's position in the previous moment.
(3) Scanning and matching involve matching the converted laser data to the existing map. The Levenberg–Marquardt (LM) method is used to solve the position increment of the robot through optimal matching, ultimately determining the robot's current position.

Table 1 Improved SLAM Algorithm

Improved SLAM Algorithm	
1.	Initialize robot pose q_0 and map M
2.	Initializing various parameters
3.	$t := 1$
4.	**while** true **do**
5.	**for all** t moment of lidar scanning data z_i **do**
6.	Convert z_i to the world coordinate system $S_i(q_{t-1})$
7.	**end for**
8.	Compute the scan matching objective function $f := \sum_{i=1}^{n} M\ (S_i(q_{t-1}))$ and its Jacobian matrix J
9.	Using the LM method to update the position $q_t := LM(q_{t-1}, f, J)$
10.	**for all** t moment of LiDAR data S_i **do**
11.	Update WSDF value $WSDF(M(S_i(q_t)))$
12.	Updated maps
13.	**end of**
14.	$t := t+1$
15.	**end while**

(4) After obtaining the robot's position at the current moment, update the value of the WSDF map and consequently the environment map.
(5) Return to step (2) for the next localization moment and map update until reaching the final moment, then the method concludes.

3.2 SLAM Back-end Optimization

3.2.1 Cartographer Algorithm

The Cartographer algorithm utilizes a graph optimization approach [39]. The fundamental concept of the method is to create a precise estimation of the robot's trajectory and map by analyzing the stored sensor data and the relationships among them. Subsequently, the algorithm operates based on the established constraints. In this method, the robot's poses are represented by nodes, and the spatial constraints between poses are represented by edges between nodes, resulting in a graph called a bitmap. Once the bit-posture map is completely constructed, the robot's trajectory and the constructed map are looped back and optimized by adjusting the robot's bit-posture to best satisfy the constraint relationships represented by the edges. The Cartographer algorithm framework is shown in Fig. 4.

For example, $\xi = (\xi_x, \xi_y, \xi_\theta)$ is used to denote the robot position, ξ_x and ξ_y are used to denote translations in directions x and y, and ξ_θ is used to denote rotations [58].

The LiDAR data are shown in Eq. (9).

$$H = \{h_k\}_{k=1.....k}, h_k \in R^2 \tag{9}$$

The initial point of the laser is $0 \in R^2$.

The laser data is mapped to the submap with the attitude transformation T_ξ and the coordinate system shown in Eq. (10).

$$T_\xi P = \begin{pmatrix} cos\xi_\theta & -sin\xi_\theta \\ sin\xi_\theta & cos\xi_\theta \end{pmatrix} P + \begin{pmatrix} \xi_x \\ \xi_y \end{pmatrix} \tag{10}$$

Subgraphs are generated from successive scanned laser data frames. When a new scanned data frame is inserted into the raster, the state of the raster is calculated. The raster is then categorized into two states: hit and miss.

Observed rasters are updated with probabilities according to Eq. (11), while unobserved rasters are assigned a probability according to Eq. (12).

$$odds(p) = \frac{p}{1-p} \tag{11}$$

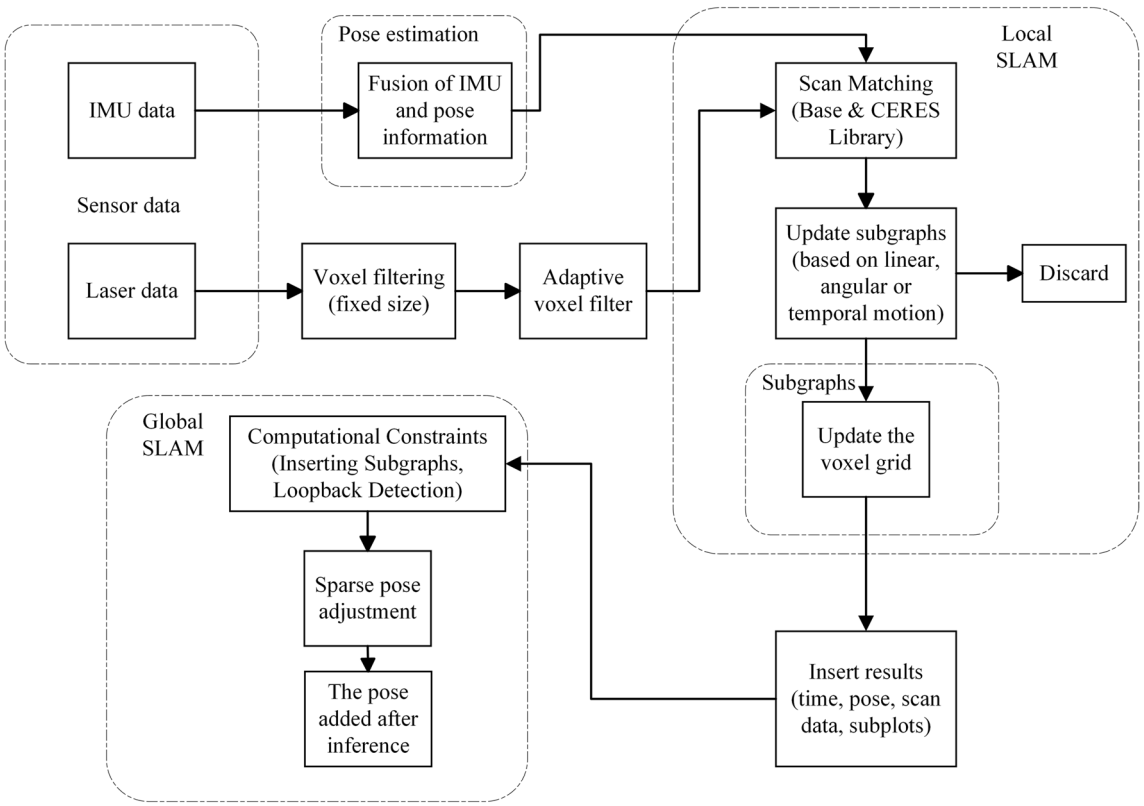


Fig. 4 Cartographer algorithm framework

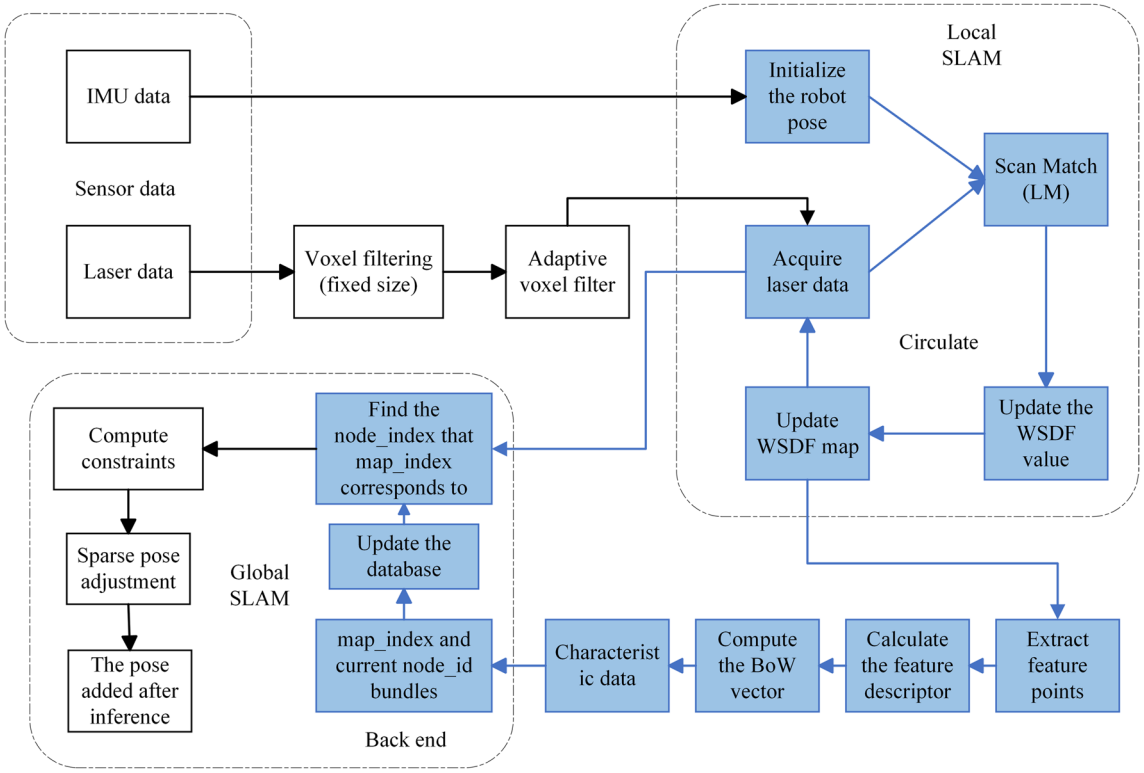


Fig. 5 Optimizing the Cartographer Algorithm

$$M_{new}(x) = clamp(odds^{-1}(odds(M_{old}(x)) \cdot odds(P_{hit}))) \quad (12)$$

The Ceres Solver optimization is performed before the laser data frame is inserted into the submap. As shown in Eq. (14).

$$\underset{\xi}{argmin} \sum_{k=1}^{k} \left(1 - M_{smooth}(T_{\xi} h_k)\right)^2 \quad (13)$$

Since the LiDAR scanning frame only matches the current sub-map, it will generate cumulative errors. Therefore, it needs to be optimized for the sub-map position and LiDAR data frame. As shown in Eq. (14).

$$\underset{[I]^m,[I]^s}{argmin} \frac{1}{2} \sum_{ij} \rho(E^2(\xi_i^m, \xi_j^s, \sum ij, \xi_{ij})) \quad (14)$$

The following Eqs. (15) and (16) refer to the submap bitmap and laser frame bitmap under specific constraints.

$$[I]^m = \{\xi_i^m\}_{i=1.....m} \quad (15)$$

$$[I]^s = \{\xi_j^s\}_{j=1.....n} \quad (16)$$

The relative position ξ_{ij} is given to the scanning frame j of the LiDAR to obtain the position in subgraph i, which is then optimally constrained together with the covariance matrix $\sum_{ij}$. The cost function is expressed through the residuals E This is shown in Eqs. (17) and (18).

$$E^2(\xi_i^m, \xi_j^s, \xi_{ij}) = e(\xi_i^m, \xi_j^s, \xi_{ij})^T \sum_{ij}^{-1} e(\xi_i^m, \xi_j^s, \xi_{ij}) \quad (17)$$

$$e(\xi_i^m, \xi_j^s, \xi_{ij}) = \xi_{ij} - \begin{pmatrix} R_{\xi_i^m}^{-1}(t_{\xi_i^m} - t_{\xi_j^s}) \\ \xi_{i;\theta}^m - \xi_{j;\theta}^s \end{pmatrix} \quad (18)$$

The branch delimitation algorithm in Cartographer system will accelerate loopback detection by primarily utilizing a lookup method for loopbacks. As shown in Eq. (19).

$$\xi^* = \underset{\xi \in w}{argmin} \sum_{k=1}^{k} M_{nearest}(T_{\xi} h_k) \quad (19)$$

In Eq. (3.37), w is the search window and $M_{nearest}$ is the extension of the M function. Determining the window near the new raster modifies the angle of the growth value ξ_{θ} and the maximum detection range d_{max} obtained by using LiDAR. As shown in Eqs. (20) and (21).

$$d_{max} = \underset{k=1....k}{max} \| h_k \| \quad (20)$$

$$\delta_{\theta} = arccos\left(1 - \frac{r^2}{2d_{max}^2}\right) \quad (21)$$

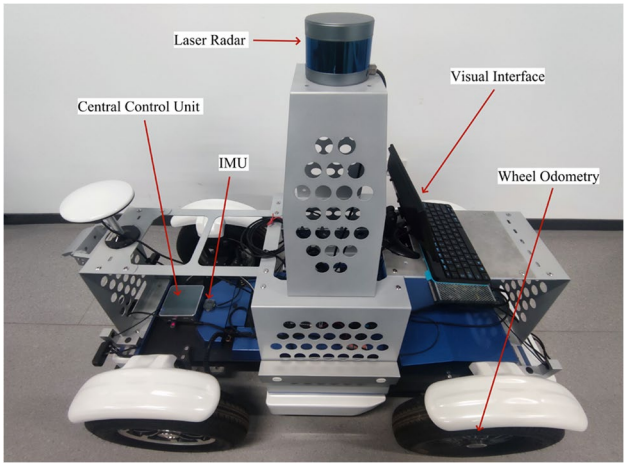


Fig. 6 Mobile robot

Table 2 Mobile robot parameters

Structural parameters	Numerical value
overall length	1608 mm
overall width	800 mm
tread size	front wheel 686 mm; rear wheel 692 mm
wheelbases	900 mm
weight	90 kg
minimum turning radius	2.54 m

Adjusted the size to fit the search window and cover it. As shown in Eqs. (22), (23), and (24).

$$w_x = \left[\frac{W_x}{r}\right] \tag{22}$$

$$w_y = \left[\frac{W_y}{r}\right] \tag{23}$$

$$w_\theta = \left[\frac{W_\theta}{\delta_\theta}\right] \tag{24}$$

A finite set W of windows will be searched based on the bit-wise estimate ξ_θ as the position, as shown in Eqs. (25) and (26).

$$\overline{W} = \{-W_x, ..., W_x\} \cdot \{-W_y, ..., W_y\} \cdot \{-W_\theta, ..., W_\theta\} \tag{25}$$

$$W = \{\xi 0 + (rj_x, rj_y, \delta_\theta j_\theta) : (j_x, j_y, j_\theta) \in \overline{W}\} \tag{26}$$

Since the search window limits the loopback speed, the calculation of δ -value is carried out by the branch bounding method for loopback optimization of the branch bounding algorithm based on depth-first search.

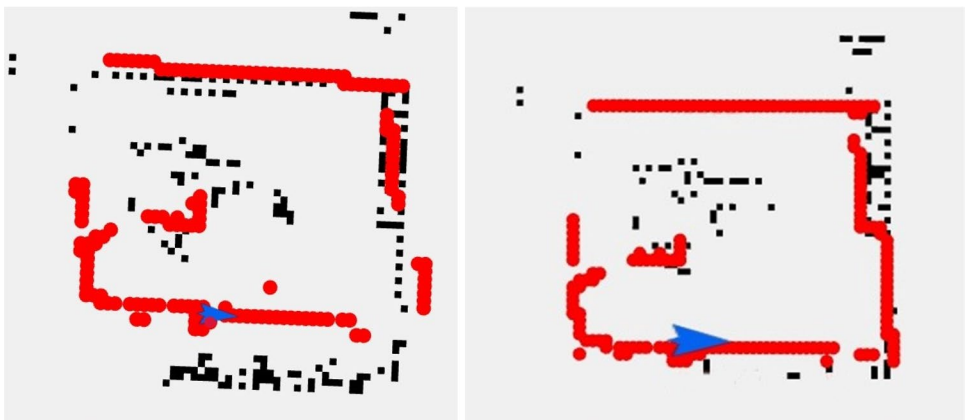
3.3 Optimizing the Cartographer Algorithm

The speed of loopback detection is accelerated with the introduction of branch partitioning in the back-end of SLAM. However, loopback detection still requires traversing all subgraphs, leading to a significant amount of computation. To address the issues of low relocation accuracy and lengthy loopback detection, we introduce an enhanced Cartographer algorithm.

As shown in Fig. 5 below, the black box represents the original framework of the Cartographer algorithm, while the blue box indicates the optimized and improved section.

The process of building an enhanced diagram is as follows:

(1) Initialize the robot's position and the environment map, and set the relevant parameters.



a. Hector-SLAM b. The optimization algorithm in this paper

Fig. 7 Scanning Matching Room Diagram

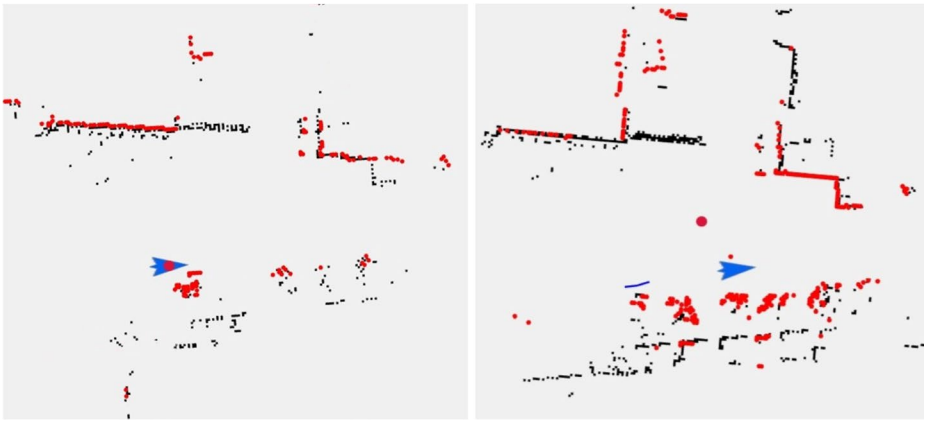


Fig. 8 Scanning Matching Outdoor Diagram

a. Hector-SLAM b. The optimization algorithm in this paper

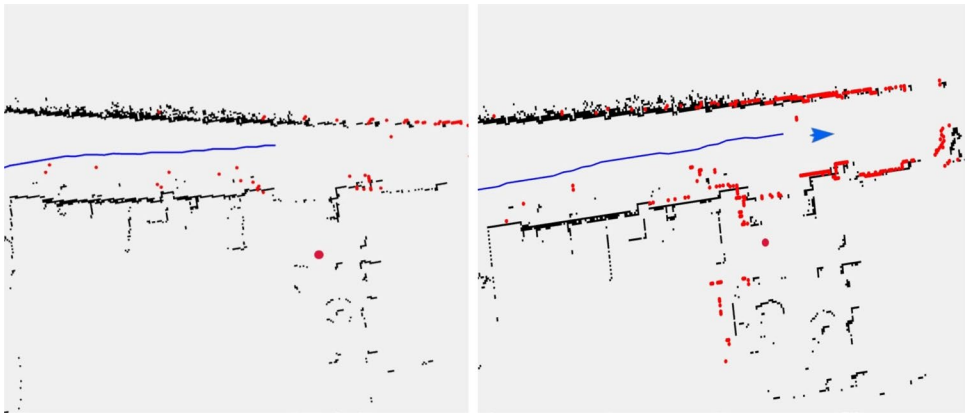
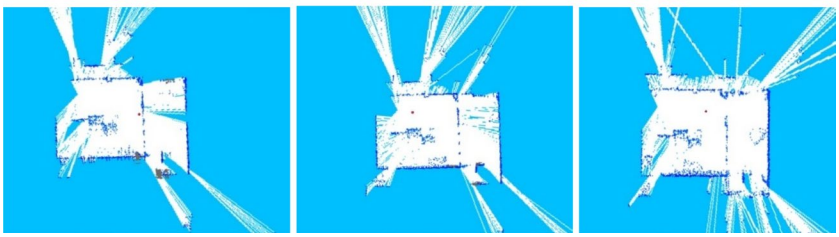


Fig. 9 Scanning Matching Gallery Diagram

a. Hector-SLAM b. The optimization algorithm in this paper

(2) Acquire the laser data at the current moment and convert it to the world coordinate system based on the robot's position in the previous moment.
(3) Scanning and matching involve matching the converted laser data with the existing map. The LM method is then used to calculate the robot's position increment based on the optimal matching, determining the robot's current position.
(4) After determining the robot's current position, update the WSDF map, and then update the environment map.
(5) Return to step (2) for the next moment of positioning and map update.
(6) Receive the WSDF map input information from the previous step, and then extract the feature points in the map in chunks. After chunking the feature points, utilize the quadtree to allocate the feature points and extract the maximum feature points to achieve a uniform distribution of feature points.
(7) Calculate the feature descriptor and compute the bag-of-words (BoW) vector of the frame.
(8) Transmit the feature data to the backend. The feature data includes a bag-of-words vector and a map index.
(9) In the backend, first bind the map index with the current node_id to node_to_map.
(10) Secondly, the bound information is updated in the database, which now stores the map_index of the nodes' locations.
(11) Finally, according to the map node obtained at the current moment, go to the database to find the historical map with the same node. After finding the map, the map_index corresponds to the node_index, and finally, the current laser data is applied to the node's bit position and matched with the map at that location.
(12) Save the map.

a. Indoor environment



b. Hector-SLAM c. Cartographer d. This paper algorithm

Fig. 10 Diagram of the indoor back-end optimization algorithm

In this study, firstly, the raster maps were optimized, and the SDF maps were improved and integrated into the raster maps. Subsequently, the WSDF maps were constructed by incorporating the concept of weighted distances from the TSDF maps. Weighting was applied to each update, resulting in smoother and more effective update values. Secondly, the Cartographer algorithm process is optimized. After incorporating the WSDF map into the framework, the received map information is chunked to extract feature points. The feature points are then assigned, and the maximum feature point is extracted. Feature descriptors are then calculated, and BoW vectors of the frame are computed. The feature data is transferred to the back-end, which includes the BoW ctor and the map index.

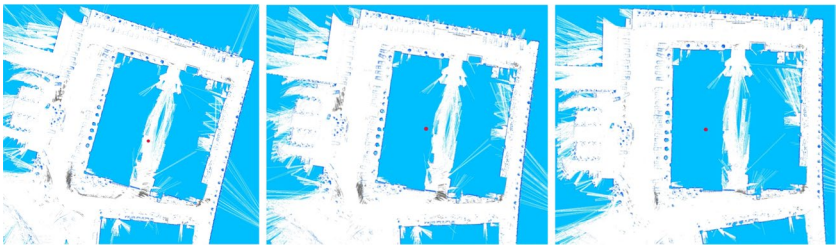
The optimized relocation process only requires loading the dictionary and the database, extracting features from the map, querying all historical images of the same node in the database, identifying features in the image, and then determining the corresponding node_index based on the image index. Subsequently, the current laser data is utilized to compare with the map of the node's position, eliminating the need to traverse sub-maps.

4 Results and Discussion

4.1 Experimental Platforms

The hardware platform of the experimental setup in this paper is a mobile robot that utilizes Ackermann steering. It comprises a Robosense-16 line lidar, IMU, wheel odometer, visual interface, and central control unit. The LiDAR sensor is mounted on the top of the mobile robot, while the IMU is fixed on the front axle of the mobile robot. The wheel odometer is set in the center of the wheel hub, the visual interface is fixed above the rear axle of the mobile robot, and the central controller is fixed in the center of the front axle of the mobile robot, as shown in Fig. 6. The main components of the mobile robot used in this paper are marked in the figure.

a. Outdoor scenes



b. Hector-SLAM c. Cartographer d. This paper algorithm

Fig. 11 Outdoor back-end optimization algorithm diagram

The processor model used in the mobile robot is Intel i5-10210U 1.60 GHz with 8 GB of memory. The system employed in the device is the ROS (Robot Operating System) system running on the Linux operating system, and the visualization platform is Rviz (Robot Visualization). The operating system utilized in the central control machine is Ubuntu 18.04.6 LTS. The software framework for acquisition and algorithm development is ROS, and the primary programming language employed is C++. In this paper, the software framework for data acquisition and algorithm development is ROS and the C++programming language. Table 2 presents the parameters of the autonomous mobile robot.

4.2 Experimental Results

In this section, the Hector-SLAM algorithm is experimentally compared with the optimized algorithm proposed in this paper to validate its effectiveness. The black area in the figure represents the actual wall, while the red area indicates the point cloud matching.

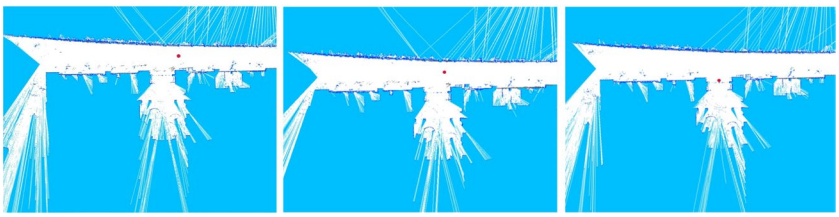
Firstly, the indoor environment scanning matching results are compared. As shown in Fig. 7.

Then, a comparison of the outdoor environment scanning matching results is presented. As shown in Fig. 8.

Finally, a comparison of the results of the environmental scanning and matching of the promenade-type linear road. As shown in Fig. 9.

As depicted in the figure, there is a clear discrepancy between the matching results of the red dot cloud and the actual wall surface in the indoor, outdoor, and corridor linear road environments of the Hector-SLAM algorithm. In contrast, the algorithm proposed in this paper minimizes misalignment, resulting in a nearly flat wall that significantly improves map construction. The experimental results demonstrate that the proposed algorithm can effectively construct the scan matching map in real-world scenarios.

a. Promenade environment



a. Hector-SLAM c. Cartographer d. This paper algorithm

Fig. 12 Diagram of the optimization algorithm for the back-end of the promenade

4.3 Back-End Optimization Algorithm Validation

Firstly, an indoor environment was chosen for this experiment to validate the algorithm, and the experimental results are shown in Fig. 10 below. From the figure, it can be seen that in the Hector-SLAM algorithm and Cartographer algorithm, there are relatively more grey areas, indicating that there are many unidentified areas. However, in contrast, the algorithm proposed in this paper yields more comprehensive results in the mapping process, with no apparent unidentified grey areas.

The second experimental scenario involves an outdoor environment with a large building footprint and uneven structures like manhole covers and slopes on the surrounding road surface. This setting offers a better opportunity to test the performance of the back-end optimization algorithm. The test scenario and experimental results are shown in Fig. 11 below.

The third experimental scenario involves a straight road designed as a promenade. The scenario map and map construction effect are shown in.

Figure 12 below. On one side of the straight promenade-style road, there was a lawn. Since similar information is often repeated in this area and there is no solid boundary wall, certain scenes may not be accurately recognized or may be ignored during the map construction process. Therefore, experiments in such a complex environment can more comprehensively evaluate the performance of the optimization algorithm proposed in this paper in terms of recognition capability and stability.

4.4 Discussion

4.4.1 Indoor Test Scenarios

Ten locations were randomly selected in the indoor scene, and each location was individually numbered from 1 to 10. Subsequently, experiments were conducted by applying Hector-SLAM, Cartographer, and the optimization algorithm proposed in this paper, respectively, to evaluate their performances at these 10 labeled locations. Calculate its MAE (Mean Absolute Error).

The MAE is calculated as the average of the absolute differences between the predicted and true values. It is calculated by adding up the absolute value of the error for each sample and dividing by the number of samples. It is one of the common measures of the error in a predictive model. It is the average absolute difference between the predicted value and the actual observed value. Since the mean absolute error is the average of the absolute value of each error, it does not cancel out the positive and negative errors, thus better reflecting the actual magnitude of the prediction error. The MAE is capable of handling outliers in a relatively robust manner, thus preventing adverse effects on performance evaluation. The MAE treats all observations equally.

The recorded data is shown in Table 3 below.

In the indoor experimental scenario. By comparing the absolute errors of the three algorithms, as shown in Fig. 13, it can be concluded that the algorithm proposed in this paper maintains an overall lower error level than

Table 3 Indoor data table

a. Hector-SLAM			
Position number	Measured distance on the graph/m	Actual measuring distance/m	Absolute error/m
1	8.836	8.876	0.040
2	4.549	4.554	0.005
3	5.886	5.854	0.032
4	1.156	1.054	0.102
5	3.443	3.342	0.101
6	1.863	1.916	0.053
7	2.376	2.273	0.103
8	1.863	1.801	0.062
9	2.771	2.687	0.084
10	10.335	10.341	0.006
b. Cartographer			
Position number	Measured distance on the graph/m	Actual measuring distance/m	Absolute error/m
1	8.869	8.876	0.007
2	4.550	4.554	0.004
3	5.813	5.854	0.041
4	1.025	1.054	0.029
5	3.442	3.342	0.100
6	1.836	1.916	0.080
7	2.328	2.273	0.055
8	1.699	1.801	0.102
9	2.614	2.687	0.073
10	10.326	10.341	0.015
c. This paper optimises the algorithm			
Position number	Measured distance on the graph/m	Actual measuring distance/m	Absolute error/m
1	8.884	8.876	0.008
2	4.552	4.554	0.002
3	5.840	5.854	0.014
4	1.025	1.054	0.029
5	3.439	3.342	0.097
6	1.969	1.916	0.053
7	2.328	2.273	0.055
8	1.846	1.801	0.045
9	2.727	2.687	0.040
10	10.338	10.341	0.003
d. MAE			
	Hector-SLAM	Cartographer	This paper optimises the algorithm
MAE	0.0588	0.0506	0.0346

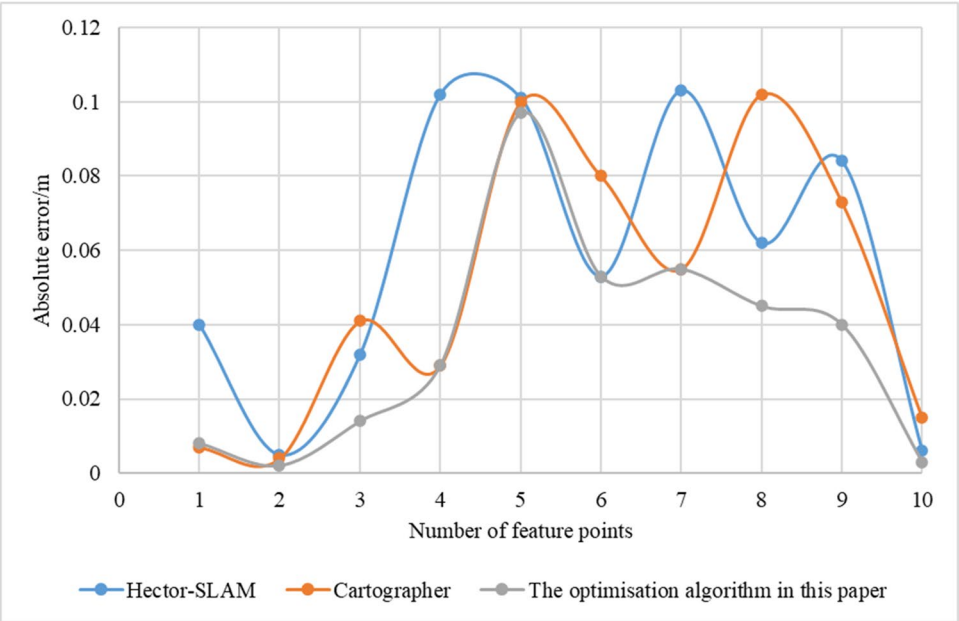


Fig. 13 Indoor Absolute Error Diagram

the other two algorithms. The specific percentage of error improvement is shown in Fig. 14, indicating that the SLAM algorithm proposed in this paper reduces the error in the map construction process by 41.16% compared to the Hector-SLAM algorithm and by 31.62% compared to the Cartographer algorithm.

4.5 Outdoor Test Scenarios

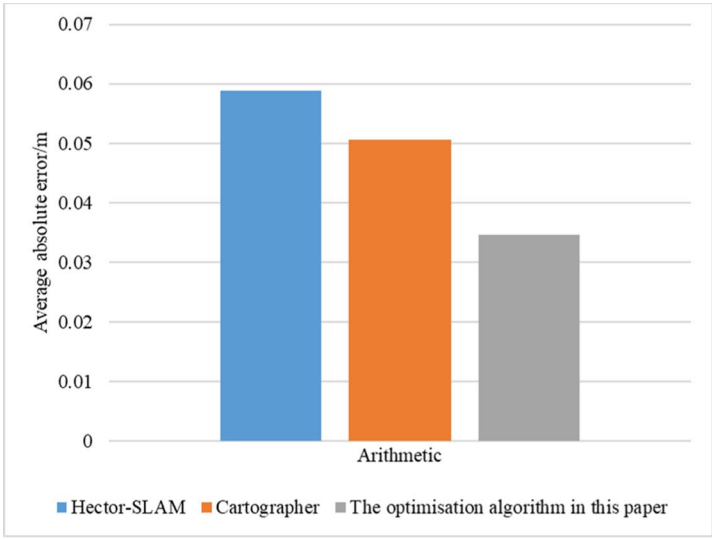
In this outdoor scenario experiment, the performance of the algorithm proposed in this paper was evaluated on complex and uneven road surfaces. The experiment was conducted around an outdoor building to simulate real-world conditions. In order to conduct error analysis, 10 test points were selected this time, and the corresponding data were recorded. The specific data is shown in Table 4 below.

In the outdoor experimental scenario, the absolute error results are shown in Fig. 15, which clearly demonstrate that the algorithm proposed in this paper is more accurate than the other two algorithms. The specific error improvement percentage is shown in Fig. 16, which demonstrates that the SLAM algorithm proposed in this paper reduces the error by 24.02% compared to the Hector-SLAM algorithm and by 10.19% compared to the Cartographer algorithm.

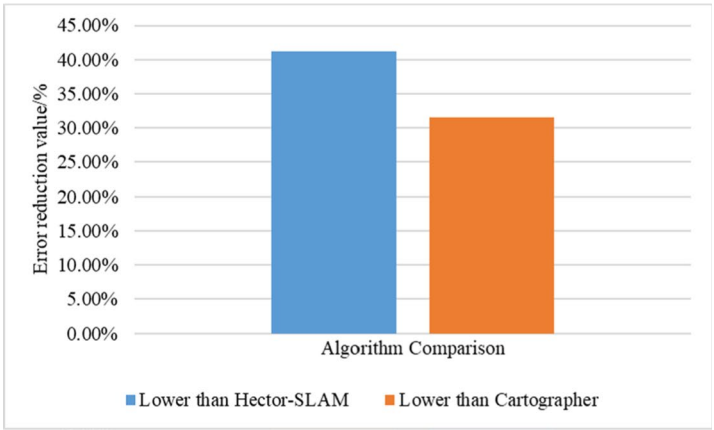
In this paper, the absolute error values and average absolute error values are calculated by comparing the detailed feature point analysis data of the Hector-SLAM algorithm, the Cartographer algorithm, and the algorithm proposed in this study. In the indoor test scenario, the SLAM algorithm proposed in this paper reduces the error in the map construction process by 41.16% compared with the Hector-SLAM algorithm and by 31.62% compared with the Cartographer algorithm. In the outdoor test scenario, the error of the proposed SLAM algorithm is reduced by 24.02% compared to the Hector-SLAM algorithm and by 10.19% compared to the Cartographer algorithm. The error analysis of the experimental data clearly shows that the proposed algorithm has less error compared to the other two algorithms.

5 Conclusion and Future

In this paper, the challenges faced by indoor and outdoor mobile robots during map construction are effectively addressed through a comprehensive study of the SLAM algorithm. The starting point of this paper is that the current hardware technology of mobile robots has become relatively mature, and the issue of power consumption has gradually decreased. This reduction in power consumption has lessened the constraints on the development of indoor and outdoor mobile robots. Therefore, the use of sensor technology and optimization algorithms to create high-precision maps is the key breakthrough in this research. By optimizing the map construction algorithm and enhancing the existing SLAM algorithm, this paper successfully achieves the creation of high-precision maps, reduces data errors, and optimizes the repositioning effect. The advancement of this technology not only offers a dependable foundation for the navigation of mobile robots and enhances their operational efficiency but also creates additional room for enhancing their performance in various work settings, enabling them to adapt more flexibly to changing environments. Against the backdrop of the current advancements in mobile robot technology, the



a. Comparative plot of mean absolute error



b. Comparison plot of algorithmic enhancements (Optimized algorithms in this paper compared to Hector-SLAM and Cartographer error degradation plots)

Fig. 14 Comparison of indoor algorithms

research findings of this paper will positively contribute to the ongoing development of the mobile robot field.

The future research directions of 2D laser SLAM are as follows.

1. Algorithm optimization

Enhance the real-time performance and robustness of SLAM, minimize computational load, and decrease the performance demands on the SLAM processing platform. Eliminate the motion aberration of LiDAR through the speed estimation compensation method and odometer interpolation model to enhance the accuracy of front-end scan matching. A novel sequence data processing flow is proposed to effectively reduce the phenomenon of map building drift and improve the system's robustness. Extract special edge or corner point features by segmenting the laser scanning data. An adaptive algorithm framework is designed to automatically adjust the parameters and strategies under varying environmental conditions. This enhances the robustness and adaptability of the system. Fusing semantic information with laser data to enhance the semantic representation and depth of understanding of the map. By combining semantic information, higher-level scene understanding, and robot behavior planning, we can achieve improved intelligence in the system.

2. Multi-sensor fusion

Table 4 Outdoor data table

a. Hector-SLAM			
Position number	Measured distance on the graph/m	Actual measuring distance/m	Absolute error/m
1	25.352	25.342	0.010
2	8.013	7.988	0.025
3	4.929	4.876	0.053
4	2.906	2.785	0.121
5	1.767	1.657	0.110
6	5.724	5.635	0.089
7	2.940	2.812	0.128
8	8.911	8.834	0.077
9	2.315	2.216	0.099
10	3.418	3.368	0.050
b. Cartographer			
Position number	Measured distance on the graph/m	Actual measuring distance/m	Absolute error/m
1	25.349	25.342	0.007
2	7.996	7.988	0.008
3	4.919	4.876	0.043
4	2.896	2.785	0.111
5	1.765	1.657	0.108
6	5.707	5.635	0.072
7	2.933	2.812	0.121
8	8.911	8.834	0.077
9	2.314	2.216	0.098
10	3.371	3.368	0.003
c. This paper optimises the algorithm			
Position number	Measured distance on the graph/m	Actual measuring distance/m	Absolute error/m
1	25.347	25.342	0.005
2	7.996	7.988	0.008
3	4.910	4.876	0.034
4	2.883	2.785	0.098
5	1.757	1.657	0.100
6	5.698	5.635	0.063
7	2.920	2.812	0.108
8	8.910	8.834	0.076
9	2.304	2.216	0.088
10	3.370	3.368	0.002
d. MAE			
	Hector-SLAM	Cartographer	This paper optimises the algorithm
MAE	0.0762	0.0648	0.0582

Fusion of multiple sensor data in the data acquisition stage can provide richer information for SLAM and improve the quality of map construction. Multi-sensor fusion is an inevitable trend. The 2D LiDAR is integrated with the depth camera to enhance the sensor's resistance to interference, expand detection range, and improve map construction accuracy. Aiming to address the issue of the invisible light distance effect in ultra-wideband (UWB) technology and the error accumulation in indoor positioning with LIDAR, a fusion of UWB absolute positioning technology and LIDAR relative positioning technology is employed to enhance the accuracy of positioning. Using wireless communication technology, data from multiple mobile robots or sensor nodes is centrally processed and

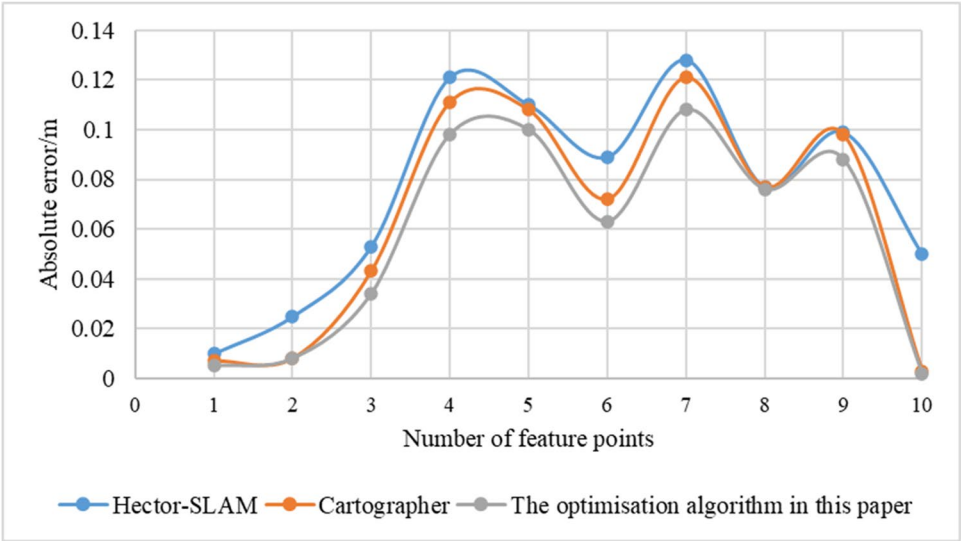
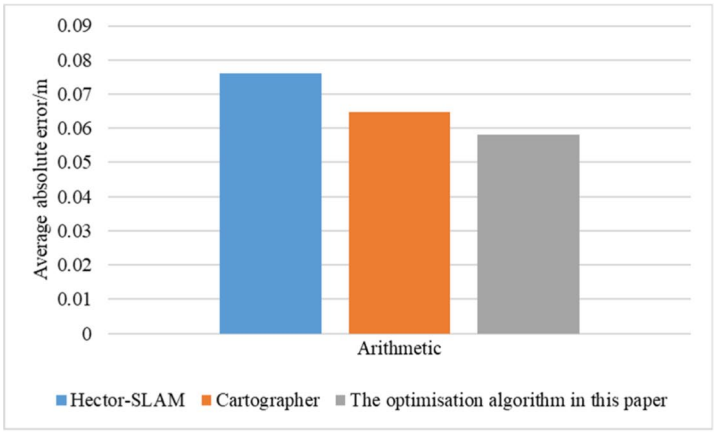
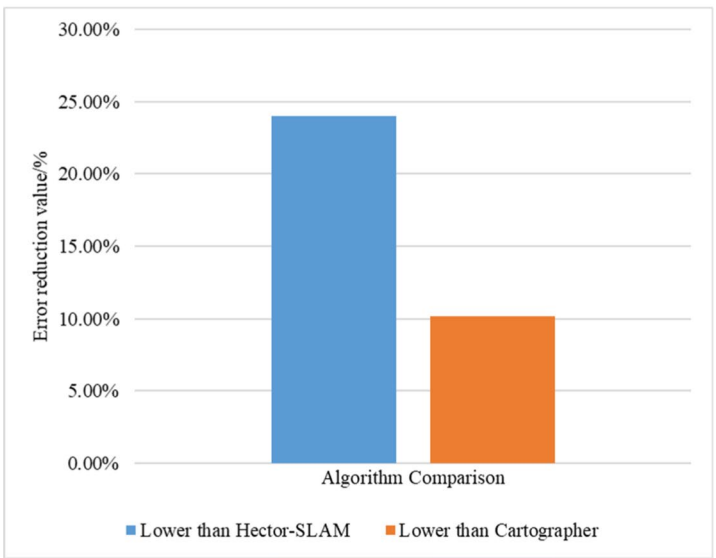


Fig. 15 Outdoor absolute error diagram



a. Comparative plot of mean absolute error



b. Comparison plot of algorithmic enhancements (Optimized algorithms in this paper compared to Hector-SLAM and Cartographer error degradation plots)

Fig. 16 Comparison of outdoor algorithms

fused. Through wireless communication, sensor fusion enables multi-robot cooperative positioning and map construction, expanding the application range and coverage of the SLAM system.

Acknowledgements The authors would like to thank City University Malaysia and Cardiff Metropolitan Univer- sity, UK for the support.

Author Contributions All the authors have equally contributed.

Funding N/A.

Data Availability The data is available and can be shared on request.

Code Availability The code is available and can be shared on request.

Declarations

Ethical Approval No human involved and all the research is carried out in an ethical manner.

Consent to Participate I hereby give my consent for the participate if any information required.

Consent for Publication I hereby give my consent for the results of this study to be published.

Conflicts of Interest/Competing Interests: The authors declare there is no conflict of interest.

.

References

1. Smith, R., Self, M., Cheeseman, P.: Estimating Uncertain Spatial Relationships in Robotics* *The research reported in this paper was supported by the National Science Foundation under Grant ECS-8200615, the Air Force Office of Scientific Research under Contract F49620–84-K-0007, and by General Motors Research Laboratories. In: Machine Intelligence and Pattern Recognition, vol. 5, J. F. Lemmer and L. N. Kanal, (eds.) in Uncertainty in Artificial Intelligence, vol. 5. pp. 435–461, North-Holland, (1988). https://doi.org/10.1016/B978-0-444-70396-5.50042-X
2. Guan, S., Zhuang, Z., Tao, H., Chen, Y., Stojanovic, V., Paszke, W.: Feedback-aided PD-type iterative learning control for time-varying systems with non-uniform trial lengths. Trans. Inst. Meas. Control. **45**(11), 2015–2026 (2023). https://doi.org/10.1177/01423312221142564
3. Stojanovic, V.: Fault-tolerant control of a hydraulic servo actuator via adaptive dynamic programming. MMC **3**(3), 016 (2023). https://doi.org/10.3934/mmc.2023016
4. Wang, R., Zhuang, Z., Tao, H., Paszke, W., Stojanovic, V.: Q-learning based fault estimation and fault tolerant iterative learning control for MIMO systems. ISA Trans. **142**, 123–135 (2023). https://doi.org/10.1016/j.isatra.2023.07.043
5. Tao, H., Zheng, J., Wei, J., Paszke, W., Rogers, E., Stojanovic, V.: Repetitive process based indirect-type iterative learning control for batch processes with model uncertainty and input delay. J. Proc. Contr. **132**, 103112 (2023). https://doi.org/10.1016/j.jprocont.2023.103112
6. Papandrea, M., Giordano, S.: Location prediction and mobility modelling for enhanced localization solution. J Ambient. Intell. Human. Comput. **5**(3), 279–295 (2014). https://doi.org/10.1007/s12652-013-0175-x
7. Durga, S., Daniel, E., Leelipushpam, P.G.J.: A novel request state aware resource provisioning and intelligent resource capacity prediction in hybrid mobile cloud. J. Ambient. Intell. Human. Comput. **13**(5), 2637–2650 (2022). https://doi.org/10.1007/s12652-021-03093-0
8. Subbaraj, S., Thiyagarajan, R., Rengaraj, M.: A smart fog computing based real-time secure resource allocation and scheduling strategy using multi-objective crow search algorithm. J Ambient. Intell. Human. Comput. **14**(2), 1003–1015 (2023). https://doi.org/10.1007/s12652-021-03354-y
9. Celik, S.C., Incel, O.D.: Semantic place prediction from crowd-sensed mobile phone data. J. Ambient. Intell. Human. Comput. **9**(6), 2109–2124 (2018). https://doi.org/10.1007/s12652-017-0549-6
10. Samarakoon, S.M.B.P., Muthugala, M.A.V.J., Elara, M.R.: Toward obstacle-specific morphology for a reconfigurable tiling robot. J. Ambient. Intell. Human. Comput. **14**(2), 883–895 (2023). https://doi.org/10.1007/s12652-021-03342-2
11. Tanco, M.M., Tejera, G., Di Martino, J.M.: Learning agriculture keypoint descriptors with triplet loss for visual SLAM. J. Ambient. Intell. Human. Comput. (2023). https://doi.org/10.1007/s12652-023-04681-y
12. Rajmohan, S., Ramasubramanian, N.: Improved symbiotic organisms search for path planning of unmanned combat aerial vehicles. J. Ambient. Intell. Human. Comput. **14**(4), 4289–4311 (2023). https://doi.org/10.1007/s12652-023-04540-w
13. Waseem, S.M., Roy, S.K.: FPGA implementation of proximal policy optimization algorithm for edge devices with application to agriculture technology. J. Ambient. Intell. Human. Comput. **14**(10), 14141–14152 (2023). https://doi.org/10.1007/s12652-022-04117-z
14. Ospina-Mateus, H., Quintana Jiménez, L.A., Lopez-Valdes, F.J., Berrio Garcia, S., Barrero, L.H., Sana, S.S.: Extraction of decision rules using genetic algorithms and simulated annealing for prediction of severity of traffic accidents by motorcyclists. J. Ambient. Intell. Human. Comput. **12**(11), 10051–10072 (2021). https://doi.org/10.1007/s12652-020-02759-5
15. Ravikumar, S., Kavitha, D.: IOT based autonomous car driver scheme based on ANFIS and black widow optimization. J. Ambient. Intell. Human. Comput. (2021). https://doi.org/10.1007/s12652-020-02725-1
16. Wang, J., Wang, M., Liu, Q., Yin, G., Zhang, Y.: Deep anomaly detection in expressway based on edge computing and deep learning. J. Ambient. Intell. Human. Comput. **13**(3), 1293–1305 (2022). https://doi.org/10.1007/s12652-020-02574-y
17. Tandon, R., Gupta, P.K.: SV2VCS: a secure vehicle-to-vehicle communication scheme based on lightweight authentication and concurrent data collection trees. J Ambient Intell Human Comput **12**(10), 9791–9807 (2021). https://doi.org/10.1007/s12652-020-02721-5

18. Fakhfakh, M., Chaari, L., Fakhfakh, N.: Bayesian curved lane estimation for autonomous driving. J Ambient Intell Human Comput **11**(10), 4133–4143 (2020). https://doi.org/10.1007/s12652-020-01688-7
19. De Amicis, R., Conti, G., Piffer, S., Prandi, F.: Service oriented computing for ambient intelligence to support management of transport infrastructures. J Ambient Intell Human Comput **2**(3), 201–211 (2011). https://doi.org/10.1007/s12652-011-0057-z
20. Suleymanoglu, B., Soycan, M., Toth, C.: Iindoor mapping: Experiences with lidar SLAM. Int. Arch. Photogramm. Remote. Sens. Spatial. Inf. Sci. **XLIII-B1**(2022), 279–285 (2022). https://doi.org/10.5194/isprsarchives-XLIII-B1-2022-279-2022.
21. Zhang, C., Fang, Z., Luo, X., Liu, W.: Accurate and robust visual SLAM with a novel ray-to-ray line measurement model. Image Vis. Comput. **140**, 104837 (2023). https://doi.org/10.1016/j.imavis.2023.104837
22. Li, R., Wang, S., Gu, D.: Ongoing evolution of visual SLAM from geometry to deep learning: Challenges and opportunities. Cogn. Comput. **10**(6), 875–889 (2018). https://doi.org/10.1007/s12559-018-9591-8
23. Xu, X., et al.: A Review of multi-sensor fusion SLAM systems based on 3D LIDAR. Remote Sensing **14**(12), 2835 (2022). https://doi.org/10.3390/rs14122835
24. Xia, Y., Lei, X., Pan, J., Chen, L., Zhang, Z., Lyu, X.: Research on orchard navigation method based on fusion of 3D SLAM and point cloud positioning. Front. Plant Sci. **14**, 1207742 (2023). https://doi.org/10.3389/fpls.2023.1207742
25 Zhao, J., Zhao, L., Huang, S., Wang, Y.: 2D Laser SLAM with general features represented by implicit functions. IEEE Robot. Autom. Lett. **5**(3), 3 (2020). https://doi.org/10.1109/LRA.2020.2996795
26. Slowak, P., Kaniewski, P.: Stratified Particle Filter Monocular SLAM. Remote. Sens. **13**(16), 3233 (2021). https://doi.org/10.3390/rs13163233
27. Chun, L., Hongfei, L., Qi, Z., Zhenzhen, M., Sisi, T., Yaping, W.: Binocular SLAM based on learning-based feature extraction. In: Proceedings of the 2020 3rd International Conference on Robot Systems and Applications, Chengdu China: ACM, pp. 25–29 (2020). https://doi.org/10.1145/3402597.3402602
28. Mirowski, P., Palaniappan, R., Ho, T. K.: Depth camera SLAM on a low-cost WiFi mapping robot. In: 2012 IEEE International Conference on Technologies for Practical Robot Applications (TePRA), pp. 1–6 (2012). https://doi.org/10.1109/TePRA.2012.6215673
29. Wang, X., He, L., Zhao, T.: Mobile robot for SLAM research based on lidar and binocular vision fusion. Chin. J. Sens. Actuat. **31**, 394–399 (2018). https://doi.org/10.3969/j.issn.1004-1699.2018.03.013
30. Pan, L., Tian, F., Ying, W., She, B.: Monocular Visual-Inertial SLAM with Camera-IMU Extrinsic Automatic Calibration and Online Estimation. In: Intelligent Robotics and Applications. H. Yu, J. Liu, L. Liu, Z. Ju, Y. Liu, and D. Zhou, (eds.) Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 706–721 (2019). https://doi.org/10.1007/978-3-030-27538-9_61
31. Yu, F., Yu, H., Wei, Y.: Research on robot positioning technology based on multi sensor. In: 2019 International Conference on Computer Network, Electronic and Automation (ICCNEA), pp. 480–485 (2019). https://doi.org/10.1109/ICCNEA.2019.00094
32. Djehaich, M., Ziane, H., Achour, N., Tiar, R., Ouadah, N.: SLAM-ICP with a Boolean method applied on a car-like robot. In: 2013 11th International Symposium on Programming and Systems (ISPS), pp. 116–121 (2013). https://doi.org/10.1109/ISPS.2013.6581476
33. Censi, A.: An ICP variant using a point-to-line metric. In: 2008 IEEE International Conference on Robotics and Automation pp. 19–25 (2008). https://doi.org/10.1109/ROBOT.2008.4543181
34. Zhao, J., Huang, S., Zhao, L., Chen, Y., Luo, X.: Conic feature based simultaneous localization and mapping in open environment via 2D Lidar. IEEE Access **7**, 173703–173718 (2019). https://doi.org/10.1109/ACCESS.2019.2956563
35. Biber, P., Strasser, W.: The normal distributions transform: a new approach to laser scan matching. In: Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453), 3, pp. 2743–2748 (2003). https://doi.org/10.1109/IROS.2003.1249285
36. Olson, E. B.: Real-time correlative scan matching. In 2009 IEEE International Conference on Robotics and Automation, pp. 4387–4393 (2009). https://doi.org/10.1109/ROBOT.2009.5152375
37. Duymaz, E.. Oğuz, A. E., Temeltaş, H.: Performance analysis of filter based airborne simultaneous localization and mapping methods. In: 2015 7th International Conference on Recent Advances in Space Technologies (RAST), pp. 157–162 (2015). https://doi.org/10.1109/RAST.2015.7208333
38. Jurić, A., Kendeš, F., Marković, I., I.Petrović, I.: A comparison of graph optimization approaches for pose estimation in SLAM. In: 2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO), pp. 1113–1118 (2021). https://doi.org/10.23919/MIPRO52101.2021.9596721
39. Hess, W., Kohler, D., Rapp, H., Andor, D.: Real-time loop closure in 2D LIDAR SLAM. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1271–1278 (2016). https://doi.org/10.1109/ICRA.2016.7487258
40. Olson, E.: M3RSM: Many-to-many multi-resolution scan matching. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 5815–5821 (2015). https://doi.org/10.1109/ICRA.2015.7140013
41. Himstedt, M., Frost, J., Hellbach, S., Böhme, H.-J., Maehle, E.: Large scale place recognition in 2D LIDAR scans using Geometrical Landmark Relations. In 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5030–5035 (2014). https://doi.org/10.1109/IROS.2014.6943277
42. Granström, K., Schön, T., Nieto, J., Ramos, F.: Learning to close loops from range data. Int. J. Robot. Res. **30**, 1728–1754 (2011). https://doi.org/10.1177/0278364911405086
43. Yin, H., Ding, X., Tang, L., Wang, Y., Xiong, R.: Efficient 3D LIDAR based loop closing using deep neural network. In: 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 481–486 (2017). https://doi.org/10.1109/ROBIO.2017.8324463
44. Montemarlo, M.: FastSLAM: A factored solution to the simultaneous localization and mapping problem. American Association for Artificial Intelligence (2002). https://doi.org/10.1007/s00244-005-7058-x
45. Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges. Proc. Int. Conf. Artif. intell. (2003). https://doi.org/10.1007/s00214-013-1423-z
46. Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE Trans. Rob. **23**(1), 34–46 (2007). https://doi.org/10.1109/TRO.2006.889486
47. Steux, B., Hamzaoui, O. E.: TinySLAM: A SLAM algorithm in less than 200 lines C-language program. In: 2010 11th International Conference on Control Automation Robotics & Vision, pp. 1975–1979 (2010). https://doi.org/10.1109/ICARCV.2010.5707402
48. Gutmann, J.-S., Konolige, K.: Incremental mapping of large cyclic environments. In: Proceedings 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation. CIRA'99 (Cat. No.99EX375), pp. 318–325 (1999). https://doi.org/10.1109/CIRA.1999.810068

49. Konolige, K., Grisetti, G., Kümmerle, R., Burgard, W., Limketkai, B., Vincent, R.: Efficient sparse pose adjustment for 2D mapping. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 22–29 (2010). https://doi.org/10.1109/IROS.2010.5649043
50. Carlone, L., Aragues, R., Castellanos, J., Bona, B.: A linear approximation for graph-based simultaneous localization and mapping. (2011). https://doi.org/10.15607/RSS.2011.VII.006
51. Li, B., Wang, Y.. Zhang, Y., Zhao, W., Ruan, J., Li, P. : GP-SLAM: laser-based SLAM approach based on regionalized Gaussian process map reconstruction. Auton. Robots. **44** (2020). https://doi.org/10.1007/s10514-020-09906-z
52. Kohlbrecher, S., von Stryk, O., Meyer, J., Klingauf, U.: A flexible and scalable SLAM system with full 3D motion estimation. In 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, pp. 155–160 (2011). https://doi.org/10.1109/SSRR.2011.6106777
53. Fossel, J.-D., Tuyls, K., Sturm, J.: 2D-SDF-SLAM: A signed distance function based SLAM frontend for laser scanners. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany: IEEE, pp. 1949–1955 (2015). https://doi.org/10.1109/IROS.2015.7353633
54. Kim, H., Moon, J., Lee, B.: RGB-to-TSDF: Direct TSDF Prediction from a single RGB Image for Dense 3D Reconstruction. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6714–6720 (2019). https://doi.org/10.1109/IROS40897.2019.8968566
55. Huang, X., Cao, H., Jia, B.: Optimization of levenberg marquardt algorithm applied to nonlinear systems. Processes **11**(6), 6 (2023). https://doi.org/10.3390/pr11061794
56. Lefkimmiatis, S., Unser, M.: A projected gradient algorithm for image restoration under Hessian matrix-norm regularization. In 2012 19th IEEE International Conference on Image Processing, pp. 3029–3032 (2012). https://doi.org/10.1109/ICIP.2012.6467538
57. Kautsky, J., Golub, G.H.: On the calculation of Jacobi Matrices. Linear. Algebra. Appl. **52–53**, 439–455 (1983). https://doi.org/10.1016/0024-3795(83)80028-7
58. Xia, T., Shen, X.: Research on parameter adjustment method of cartographer algorithm. In: 2022 IEEE 6th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), pp. 1292–1297 (2022). https://doi.org/10.1109/IAEAC54830.2022.9930054

Zhuoran Li received the B.E. degree from Shandong Agriculture and Engineering University, Jinan, China, in 2020. He is currently pursuing the master's degree with the Faculty of Information Technology, City University Malaysia. His research interests include intelligent driving, Internet of Things, reinforcement learning, decision making, and path planning technology of autonomous vehicle.

Kazem Chamran received a B.Eng. (Hons) degree in Electrical and Electronic Engineering from Sheffield Hallam University in 2010, a M.Sc. degree in Network and Communication Engineering from University Putra Malaysia (UPM) in 2015, and a Ph.D. in Computing from Sunway University in 2022. He has over seven years of experience in both academia and industry, developed applications that are still in use. Currently, he is the Head of Research and a Senior Lecturer at City University Malaysia in the Faculty of IT. He is also a member of several national and international committees and scientific societies, including IEEE and Malaysian research committees. His research interests lie in the areas of artificial intelligence, 5G- 6G communication, sustainable energy harvesting, machine learning, and IoT.

Mustafa Muwafak Alobaedy is a computer science expert with extensive experience in academia and the software development industry. He currently serves as an associate professor at City University Malaysia in the Faculty of Information Technology. With over nine years in academia and more than ten years in the software development industry, Alobaedy has developed profound expertise in discrete optimization problems, metaheuristic algorithms, software development, and data warehousing. He has published several journal and conference papers and is actively involved in teaching and supervising undergraduate and postgraduate students. His research focuses on optimization, blockchain, and software engineering.

Muhammad Aman Sheikh is a lecturer in Electronics and Computer Systems Engineering at Cardiff School of Technologies at Cardiff Metropolitan University, United Kingdom. Prior to joining Cardiff Metropolitan University UK, he was Program Leader and a Senior Lecturer at the School of Engineering and Technology, Sunway University. Previously he was a Senior Scientist in the R&D sector and was involved in enormous projects for PETRONAS. He is a Fellow of the Institute of Electrical and Electronic Engineers, Malaysia (MIEEE) and, since 2010, a Registered Engineer with the Board of Engineers. An active researcher, he has published extensively and established global collaborations by working with international collaborators and securing international grants. He was awarded the Graduate Assistant Scholarship during his M.Sc. and Ph.D. in Electrical and Electronics Engineering in 2013 and 2018, respectively. He also received prestigious awards during his Ph.D. candidature, including Best Graduate Assistant Award and High Achiever Award.

Abdul Ahad earned his PhD in Computer Science from Sunway University Malaysia (2022), an M.S. in Computer Science from the Virtual University of Pakistan (2017), and an M.Sc. in Computer Science from Swat University of Pakistan (2014). Currently, as a Postdoctoral Researcher at Northwestern Polytechnical University in Xi'an, China. He has been listed in the top 2% of scientists worldwide by Stanford University in 2023. He has also contributed to the academic community through workshops, conferences such as IEEE, Elsevier, Springer, and EAI, and numerous publications in esteemed journals and conferences on subjects central to his research interests, including 5G, 6G, smart healthcare, cyber security, and the Internet of Things (IoT). His research interests are focusing on the future of telecommunications and healthcare through the lenses of 5G and 6G technologies, smart healthcare, the Internet of Things (IoT), cyber security, machine learning, and deep learning.