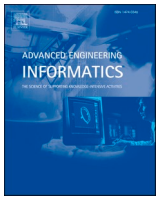
Advanced Engineering Informatics 60 (2024) 102370

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei



Cognitive robotics: Deep learning approaches for trajectory and motion control in complex environment

Muhammad Usman Shoukat [a,b], Lirong Yan [a,b,*], Di Deng [a], Muhammad Imtiaz [a], Muhammad Safdar [c], Saqib Ali Nawaz [d]

[a] Hubei Key Laboratory of Advanced Technology for Automotive Components, School of Automotive Engineering, Wuhan University of Technology, Wuhan 430070, China
[b] National Energy Key Laboratory for New Hydrogen-Ammonia Energy Technologies, Foshan Xianhu Laboratory, Foshan 528200, China
[c] Intelligent Transportation Systems Research Center, Wuhan University of Technology, Wuhan 430063, China
[d] School of Information and Communication Engineering, Hainan University, Haikou 570228, China

ARTICLE INFO

Keywords:
Simultaneous localization and mapping
YOLOv5
Mobile robot
Obstacle detection
Fingerprint sequence
Graph optimization

ABSTRACT

Simultaneous Localization and Mapping (SLAM) is the research hotspot of robot positioning and navigation. In a large-scale complex environment, closed-loop detection by vision or lidar has low reliability and high computational cost. To solve this problem, a graph optimization SLAM algorithm based on YOLOv5 (You Only Look Once version 5) and Wi-Fi fingerprint sequence matching is proposed. The proposed method utilizes fusion deep learning approaches to enhance the accuracy and robustness of closed-loop detection to navigate the robot. The algorithm uses an effective object detection network and the fingerprint sequence for closed-loop detection to figure out the dynamic semantic information within a scene. Therefore, the traditional matching based on fingerprint point pairs is extended to include matching of fingerprint sequences. This can greatly reduce the probability of closed-loop misjudgment, ensuring the accuracy of closed-loop detection and meeting the accuracy requirements of the SLAM algorithm in a wide range of complex environments. The proposed algorithm is verified with two sets of experimental data (the robot starts from different starting points): the accuracy of the proposed algorithm is 22.95% higher than that of the first set of data compared with the Gaussian similarity method; the second group of data increased by 39.19%. The experimental results show that the proposed method improves the accuracy and robustness of mobile robot localization and mapping.

1. Introduction

In the context of globalization and technology, logistics faces growing challenges. Logistics efficiency, cost reduction, and transportation safety and accuracy are important demands. SLAM is a key deep learning technology for mobile robots to achieve autonomous localization and navigation after entering complex environments. In recent years, SLAM has gradually become a hot issue in robotics [1,2]. Researchers worldwide have proposed many SLAM solutions, including graph-based SLAM, which converts the problem into maximum likelihood estimate and is considered one of the most effective methods [3–6]. Graph-based SLAM represents the environment as a graph with nodes representing robot poses and landmarks and edges representing limitations. The robot builds the graph by observing features along its path. Closed loop detection is the key to realizing graph optimization in SLAM and a difficult point in the SLAM problem. Proper closed-loop detection can fix the accumulated error of the odometer to get a consistent map. On the other hand, the wrong closed-loop detection results will mess up the subsequent map optimization, which will fail to create the map [7]. MonoSLAM [8], PTAM [9], RatSLAM [10], DTAM [11], KinectFusion [12], and ORB-SLAM [13] are just a few of the many outstanding visual SLAM methods that have aided in the advancement of SLAM technology. Robotics has a pressing need for real-time, high-precision SLAM technology, which has recently experienced explosive growth.

To realize autonomous motion, a robot needs to have the ability to perceive the surrounding scene and its position in the scene. The current SLAM system mainly relies on sensors such as cameras, inertial measurement unit (IMU), LiDAR, etc. Kundu et al. [14] proposed a method to detect objects' stationary or moving states using multi-view geometric

* Corresponding author.
E-mail address: lirong.yan@whut.edu.cn (L. Yan).

https://doi.org/10.1016/j.aei.2024.102370
Received 16 August 2023; Received in revised form 29 November 2023; Accepted 17 January 2024
Available online 26 January 2024

constraints and sensors. Wei et al. [15] suggested a camera motion model-based method for a dynamic region removal-based binocular vision SLAM algorithm. At present, 2D laser radar SLAM approach can promptly generate a 2D grid map. However, this map only shows the geometric characteristics of the environment within a plane and does not provide an accurate information about obstacles in space [16,17]. The use of 3D LiDAR SLAM and visual SLAM requires the projection of the generated spatial point cloud to obtain a 2D grid map [18]. Chen et al. [19] proposed that the aerial view can be obtained by height projection of a 3D point cloud by slicing, and the front view can be obtained by projection according to the cylindrical coordinate system. Hu et al. [20] proposed integrating semantic maps with loop detection to solve semantic maps and improve the dictionary by combining motion feature points to remove dynamic objects. Han et al. [21] noted that SLAM helps robots dynamically map and localize in complex 3D surroundings. They propose a multimodal intelligent logistics robot system that combines 3D CNN, and visual SLAM for path planning and control.

Jiao et al. [22] proposed a deep learning-based VSLAM method for indoor dynamic scenes that utilizes GCNv2 (geometric correspondence network) and a lightweight ESPNetV2 neural network for semantic segmentation of images. Then, improved motion consistency detection is used to remove dynamic features and obtain static feature points. Soares et al. [23] also tested deep learning techniques, including YOLOv3 and Mask R-CNN (region-based convolutional neural network) methods, in dynamic SLAM systems without camera motion. They found that the object detection algorithm based on YOLOv3 has improved real-time performance and accuracy. At the same time, Kazerouni et al. [24] demonstrate that YOLOv5′s advanced object detection capabilities allow real-time object identification and tracking. The combination of these methods not only improves robots' perception of their surroundings and gives them the ability to make on-the-fly changes to their trajectory planning and motion control. Wang et al. [25] proposed a semantic SLAM system in dynamic scenarios that combines deep learning methods with LUT (look-up table) SLAM and utilizes the YOLOv3 object detection algorithm to detect and remove specific moving objects, generating a dense point cloud map that eliminates moving objects. More research into SLAM and YOLOv5′s motion control capabilities will help robots navigate a complex environment. Fig. 1 shows that a full SLAM context includes tracking on the front end, back-end optimization, object detection, and map reconstruction.

However, the indoor mobile robot is limited by its working environment and its own structure, and it cannot observe the ground from some visual angles. The method of ground fitting cannot effectively separate the ground obstacles. Use the RGB-D (red, green, blue depth) camera and the ORB_SLAM2 (Oriented FAST and rotated BRIEF_SLAM2) [26] algorithm to generate a 3D dense point cloud. Based on the motion and structural characteristics of indoor mobile robot, the relationship between the point cloud coordinate system generated by the camera and the real-space ground plane is discussed, and a method of preparing a 2D grid map by dimension reduction projection is proposed. The system has been improved on the original SLAM framework by adding potential target detection threads based on deep learning and feature point relative velocity calculation threads. The improved system technique as discussed in Fig. 1. The latter realization is a beneficial supplement to the current SLAM's 2D occupation grid map preparation. Traditional visual SLAM algorithms haven't achieved breakthrough progress since 2017 due to these inevitable issues: 1) The traditional algorithm is not much robust in adverse conditions, such as poor illumination or large changes in illumination; 2) if the camera moves too quickly, the traditional algorithm easily loses the tracking target; and 3) traditional algorithms cannot recognize foreground objects, and no good solution exists. This paper proposes a graph optimization SLAM algorithm based on YOLOv5 and Wi-Fi fingerprint sequence matching in large-scale complex environments. By integrating SLAM and YOLOv5, robots can not only perceive and comprehend their surroundings but also dynamically respond to changes in real-time, optimizing their trajectory planning and motion control strategies. In summary, this study makes the following key contributions:

1) Using the dynamic time warping (DTW) algorithm to realize the matching of Wi-Fi fingerprint sequences, which ensures the accuracy of closed-loop detection;
2) The proposed module uses YOLOv5′s data for object detection to enhance the map created by the SLAM system. Graph SLAM based on YOLOv5 and Wi-Fi fingerprint information, it meets the algorithm accuracy requirements of SLAM in large-scale complex environments.
3) In order to enhance robot-edge server SLAM collaboration, this study designed a unique graph-based system. To ensure the algorithm is correct, two different sets of experimental data were used. We simplify the process into two phases: first, robot data is grouped together, and then, the edge is offloaded.

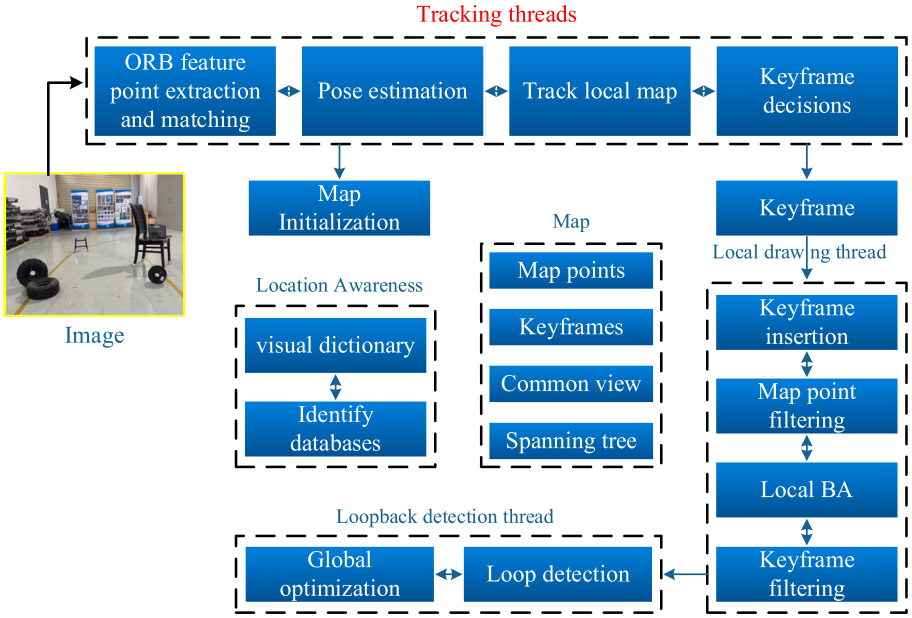


Fig. 1. Schematic diagram of an indoor SLAM system based on deep learning.

The remaining parts of this article are structured as: The framework and methodology, including graph SLAM, deep learning approaches, matching Wi-Fi fingerprint sequence and training of uncertainty models, are discussed in Sections II and III, respectively. The evaluation in Section IV takes into account both the experiment and result analysis. In Section V, we come to a conclusion.

2. Framework

2.1. Graph SLAM

The graph optimization SLAM algorithm for fingerprint sequence matching proposed in this article is mainly divided into two steps: generation of the uncertainty model and the graph SLAM algorithm. The graph framework [27] is widely used today to tackle the SLAM problem. The restrictions are defined by a quadratic equation, and the matrix form of the graph is used to express this. The whole graph optimization SLAM algorithm flow is shown in Fig. 2.

To begin with, Gaussian similarity is used to calculate the degree of similarity between any two close fingerprint points, and an uncertainty model of similarity and distance is established through a training process; second, the similarity of two remote fingerprint points is calculated. When the degree of similarity exceeds a certain threshold, a Gaussian closed loop is obtained, and the two fingerprint points are selected as the starting points for the two fingerprint sequences. The DTW algorithm is used to calculate the similarity of the two fingerprint sequences. If the level of similarity between fingerprint sequences reaches a specific threshold, a closed loop of the fingerprint sequences is achieved. Furthermore, Gauss closed loop and fingerprint sequence closed loop are added to the pose map as constraints. Further data collection results in the insertion of new edges to the matrix, which serve as additional restrictions. Given that each node can communicate with only some of its neighbors, time is typically a linear function of the number of constraints and nodes.

2.2. Deep learning approaches

With the development of deep neural networks, research in the field of object detection has been divided into two directions [28]. The first technique is a two-stage object detection algorithm that is based on target domains, like Faster RCNN. The second method is a single-stage object detection method that is based on regression calculation, like YOLOv5. This article selects the YOLOv5 network to improve the real-time performance of graph SLAM systems because it has the smallest depth and feature map width in the YOLO single-stage object detection method, as shown in Fig. 3.

Once the offline training phase is over, the graph SLAM system starts to process the camera image. Initially, it enters the latent complex target detection thread, which relies on deep learning techniques. The system utilizes the YOLOv5 model, which has been trained in advance, to identify and label the latent indoor complex environment target by generating a detection box. To improve the recognition capabilities of

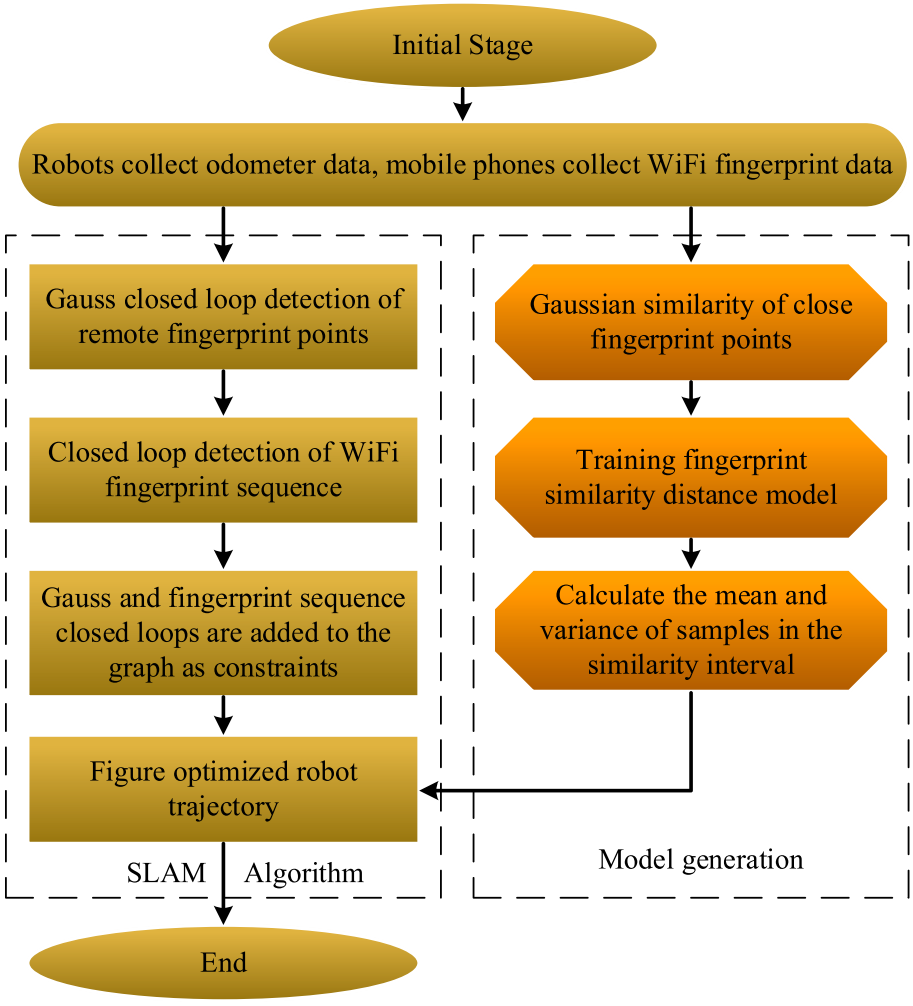


Fig. 2. Graph optimization SLAM algorithm's flowchart.

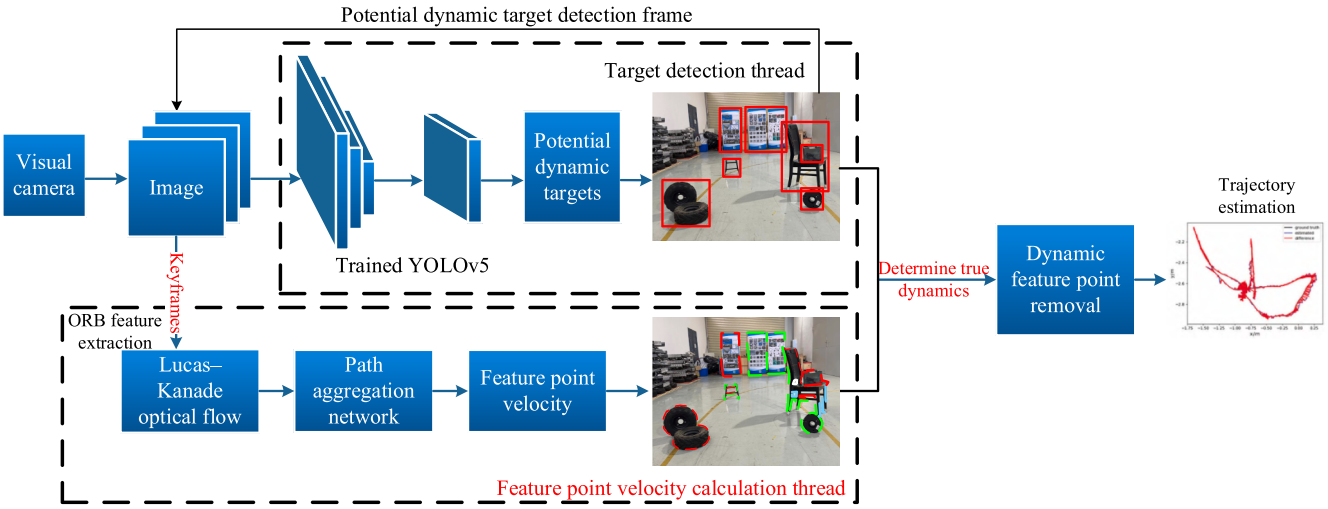


Fig. 3. Framework of a graph-based SLAM with object detection.

YOLOv5 in real locations, the pre-trained network was trained for the second time using a data set of 1000 real laboratory scenes. After the graph is built, the optimal set of robot postures that fulfill the criteria must be calculated. Thus, there are two main components to the graph-based SLAM architecture: the front and the rear. The initial phase of SLAM is concerned with the graph building using the sensor data in its raw form. On the other hand, the SLAM backend uses graph optimization to determine the most probable posture configuration given the restrictions. The graph optimization algorithm is then used to optimize the pose map to get the best path for the robot.

3. Methodology

In this article, we use Wi-Fi data for SLAM in an unknown environment and locate ourselves through the map features (Wi-Fi fingerprints) repeatedly observed by intelligent devices during the movement. In SLAM based on graph optimization, we construct a pose map from the collected sensor data, where vertices represent poses and edges represent constraints between vertices; one is obtained from the odometer data at consecutive times; the other constraint can be obtained by matching the observation data at different times (also known as closed-loop detection). This research chooses the YOLOv5 network for target detection because indoor complex environment targets with prior information can match the SLAM system's accuracy requirements. Our application uses RGB-D TUM (Technical University of Munich) datasets for pre-training due to its robotic nature. Because there is some error in the observation data, all constraints are tied to an uncertain parameter, so the SLAM problem based on graph optimization is changed into the optimization of posture to minimize the error caused by constraints. Chair, tool box, cable roller, iron stand, and posters are target objects. After offline training, when the graph optimization SLAM system gets the camera image, it enters the deep learning-based possible dynamic target detection thread and utilizes the trained YOLOv5 model to identify and mark the detection frame.

Use (x_t, y_t) to represent the 2D position information of the robot at time t, and θ_t to represent the orientation information. Let C denote the set of constructed constraints. The goal of graph-based SLAM is to find the optimal configuration of poses that minimizes the following function:

$$\sum_{(i,j)\in C}^{1} \left(Z_{ij} - \widehat{Z}_{ij}(x_i, x_j)\right)^T \sum_{ij}^{-1} \left(Z_{ij} - \widehat{Z}_{ij}(x_i, x_j)\right) \tag{1}$$

where, Z_{ij} represents the rigid body transformation between vertices i and j, $\sum$ the covariance matrix represents the uncertainty of this transformation. For the laser beam, the transformation Z_{ij} can be estimated by scanning matching (e.g., iterative closest point). For visual sensors, this transformation can be obtained by feature matching. Given the received signal strength (RSS) of an access point (AP), it is easy to determine whether a user has re-accessed a certain area because each AP has a unique hardware address.

However, it is very difficult to estimate the relative positional relationship between the two measurement points by signal strength because RSS itself does not contain any distance or direction information. Once the actual dynamic feature points are removed, the static feature points are utilized to match the two frames of images, and graph-based SLAM's BA (bundle adjustment optimizes all graph variables to improve robot pose and landmark position estimates.) technique matches the camera's six-degree-of-freedom position and 3D landmark points. At the same time, optimization is performed to calculate the camera pose $\widehat{T}$ of the current frame and the coordinates $\widehat{p}$ of the landmark point by minimizing the reprojection error. The objective function is as:

$$(\widehat{T}, \widehat{p}) = \underset{(T,p)}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{j=1}^{k} \left\| z_{i,j} - f\left(T_{i,i-1}, p_j\right) \right\|^2 \tag{2}$$

here, $z_{i,j}$ represents the pixel coordinates of the feature point of the static landmark point p_j in the i^{th} frame image, $f(*)$ is the mapping function, and $T_{i,i-1}$ represents the transformation matrix from the $(i-1)^{th}$ frame to the i^{th} frame. A DTW and loop detection algorithm are used to solve drift and optimize camera pose to improving the accuracy and robustness of the graph-based SLAM system. The Gaussian method is used to minimize reprojection errors. After obtaining the relative motion speed of feature points, it is fused with the potential dynamic target detection thread based on deep learning to determine the motion state of feature points using the following equation:

$$\text{status} = \begin{cases} true, v_i^k > v_i^{th} \\ false, v_i^k \le v_i^{th} \end{cases} \tag{3}$$

where, v_i^k represents the motion velocity of feature point k in the i^{th} frame of the image, and v_i^{th} is the overall scene motion velocity of the i^{th} frame of the image. We approximate the average motion velocity of feature points in the scene outside the potential dynamic target area between two frames to the camera's motion velocity and set it as the threshold value. Status represents the motion state of the current feature point k, while "true" represents the actual motion and "false" indicates relative stillness.

In addition, the multipath effect and the objects in the environment have a great influence on the signal propagation. The closed-loop detection using the location fingerprint method can overcome the influence of the environment on the signal propagation. A fingerprint point describes the access point (AP) detected at a certain location and the corresponding signal strength. Just like human fingerprints and DNA (deoxyribonucleic acid), Wi-Fi fingerprints can be used as features to uniquely describe a certain location, and the distance between two locations can be measured by the similarity of fingerprint point pairs. Nevertheless, in a large-scale complex environment, it is far from enough to only use the matching of fingerprint point pairs. Since the Gaussian similarity calculation method only considers the same AP scanned by two fingerprint points, it may lead to closed-loop misjudgment in the matching of fingerprint point pairs (two points close to each other but with low similarity). Extending the traditional matching based on fingerprint point pairs to the matching of fingerprint sequences can reduce the probability of closed-loop misjudgment, therefore ensuring the accuracy of the algorithm.

3.1. Matching Wi-Fi fingerprint sequence

Since the fingerprint sequence contains multiple fingerprint data, the amount of information is richer than that of a single fingerprint point pair. Therefore, the closed-loop detection is proposed through the matching of Wi-Fi fingerprint sequences, and the closed-loop detection of two fingerprint sequences is realized by using the DTW algorithm, which can greatly reduce the positioning error. The DTW algorithm is an elastic matching algorithm. By looking at how two fingerprint sequences are related to each other, dynamic programming is used to change how the two sequences are related to each other. This gives an optimal path that maximizes the similarity between the two sequences along the path. The process of localization refers to the determination of an active agent's position on a map. This work builds an environmental model using fingerprints of places. This model locates the robot. Fingerprint matching simplifies fingerprint-based localization. The fingerprint approach minimizes perceptual aliasing and enhances place distinctiveness by merging data from all robot sensors. The following sections compare fingerprint-based localization approaches. Fig. 4 shows the fingerprint-based localization process.

This section describes fingerprints of places, which represent geometric framework nodes. Compact topological representation facilitates high-level symbolic reasoning for map development and navigation. Firstly, Gaussian function is used to calculate the similarity of two fingerprint points, and the fingerprint at time t is expressed as $\boldsymbol{F}_t = (f_t, x_t)$, where $\boldsymbol{x}_t = (x_t, y_t, \theta_t)$ represents the odometer posture of the robot at time t; $\boldsymbol{f}_t$ represents the fingerprint information collected at the position $\boldsymbol{x}_t$. Specifically, the fingerprint information includes the hardware address of the AP scanned at time t and the signal strength $\boldsymbol{f}_t = (f_{t,1}, f_{t,2}, \cdots, f_{t,Z})$, where Z represents the number of AP scanned at time t. The Gaussian function is used to calculate the similarity of the two-fingerprint points $\boldsymbol{F}_i$ and $\boldsymbol{F}_j$:

$$S_{\text{Gauss}}(i,j) = \text{sim}(\boldsymbol{F}_i, \boldsymbol{F}_j) = \frac{1}{Z} \cdot \sum_{z=1}^{Z} \exp\left\{ -\frac{(f_{i,z} - f_{j,z})^2}{2\sigma^2} \right\} \tag{4}$$

If the Gaussian similarity of the two fingerprint points is greater than a certain threshold v_s, the two fingerprint points are taken as the starting points of the two fingerprint sequences, and closed-loop detection is performed by calculating the similarity of the two fingerprint sequences.

It is assumed that the two fingerprint sequences are $\boldsymbol{L} = \{l_1, l_2, \cdots, l_n\}$ and $\boldsymbol{K} = \{k_1, k_2, \cdots, k_m\}$, respectively. In order to calculate the similarity of the two fingerprint sequences, the first step is to find the corresponding relationship between the two sequences. Take the fingerprint sequence $\boldsymbol{L}$ as the horizontal axis and the fingerprint sequence $\boldsymbol{K}$ as the vertical axis, and construct a similarity matrix $\boldsymbol{S}$ of $n \times m$ as shown in equation (5). In the similarity matrix, there are multiple corresponding paths for the two fingerprint sequences. Take one of them as an example, the path is represented by blue dots, i.e., $\boldsymbol{U} = \{u_1, u_2, \cdots, u_{r-1}, u_r, u_{r+1}, \cdots, u_{R-1}, u_R\}$ as shown in Fig. 5, R represents the length of the path $\boldsymbol{U}$, which satisfies $\max(n,m) \le R \le n+m-1$; where, u_r is the coordinate of the r^{th} point of the path $u_r = (i_r, j_r)$, which indicates that the i_r fingerprint point of the fingerprint sequence $\boldsymbol{L}$ corresponds to the j_r fingerprint point of the fingerprint sequence $\boldsymbol{K}$. Then the similarity between two points $s(u_r) = s(l_{i_r}, k_{j_r}) = S_{\text{Gauss}}(l_{i_r}, k_{j_r})$, the similarity matrix $\boldsymbol{S}$ between all corresponding points of the two fingerprint sequences can be obtained as:

$$\boldsymbol{S} = \begin{Bmatrix} s(l_1,k_1) & s(l_1,k_2) & \cdots & s(l_1,k_m) \\ s(l_2,k_1) & s(l_2,k_2) & \cdots & s(l_2,k_m) \\ \vdots & \vdots & & \vdots \\ s(l_n,k_1) & s(l_n,k_2) & \cdots & s(l_n,k_m) \end{Bmatrix} \tag{5}$$

The effective path in the similarity matrix shall meet the following constraint conditions, and the schematic diagram of the constraint conditions is shown in Fig. 6.

1) Boundary conditions: the starting point is $S(1,1)$ and the ending point is $S(n,m)$, that is, the effective path must start from the lower left corner and end at the upper right corner.
2) Continuity: the path in Fig. 6 from u_r to the next point u_{r+1} needs to meet $i_{r+1} - i_r \le 1$, $j_{r+1} - j_r \le 1$. As shown in circle I in Fig. 6, in order to ensure continuity, when arriving at point $S(i,j)$ along the path, the previous point must pass through one of points $S(i-1,j-1)$, $S(i-1,j)$, $S(i,j-1)$, that is, the point at a certain time can only match the points at the same time and adjacent times, and cannot over the matching, as shown in circle II in Fig. 6.
3) Monotonicity: from w_r to the next point u_{r+1}, $i_r \le i_{r+1}$, $j_r \le j_{r+1}$, which makes the path monotonous with time, and there is no time reversal, as shown in circle III in Fig. 6.

Apparently, there are many effective paths $\boldsymbol{U}$ that meet the constraint conditions, and DTW needs to find the best path among them to maximize the cumulative value of similarity between two sequences along the path. Use $S_{seq}(L,K)$ to express the similarity between the fingerprint sequences L and K, that is, the similarity corresponding to the optimal path. The solution process of $S_{seq}(L,K)$ is as:

$$\begin{cases} r(i,j) = S_{\text{Gauss}}(i,j) + \max\{r(i-1,j-1), r(i-1,j), r(i,j-1)\} \\ S_{seq}(L,K) = r(n,m) \end{cases} \tag{6}$$

where, $r(i,j)$ denotes the cumulative value of similarity on the path from $S(1,1)$ to $S(i,j)$ in the similarity matrix S. $S_{\text{Gauss}}(i,j)$ represents the Gaussian similarity $S_{\text{Gauss}}(l_i, k_j)$ corresponding to the current grid point $S(i,j)$, that is, the Gaussian similarity between the two fingerprint points l_i and k_j. To reach point $S(i,j)$, you can only start from $S(i-1,j-1)$, $S(i-1,j)$ or $S(i,j-1)$, $\max\{r(i-1,j-1), r(i-1,j), r(i,j-1)\}$ means to select the point with the largest cumulative value of similarity among the three points $S(i-1,j-1)$, $S(i-1,j)$ and $S(i,j-1)$, as the

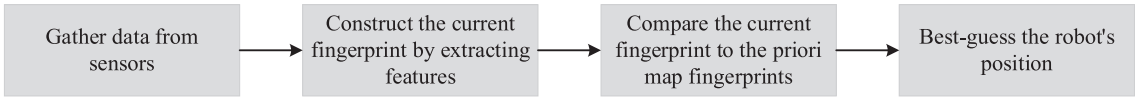


Fig. 4. Typical steps in fingerprint-based localization.

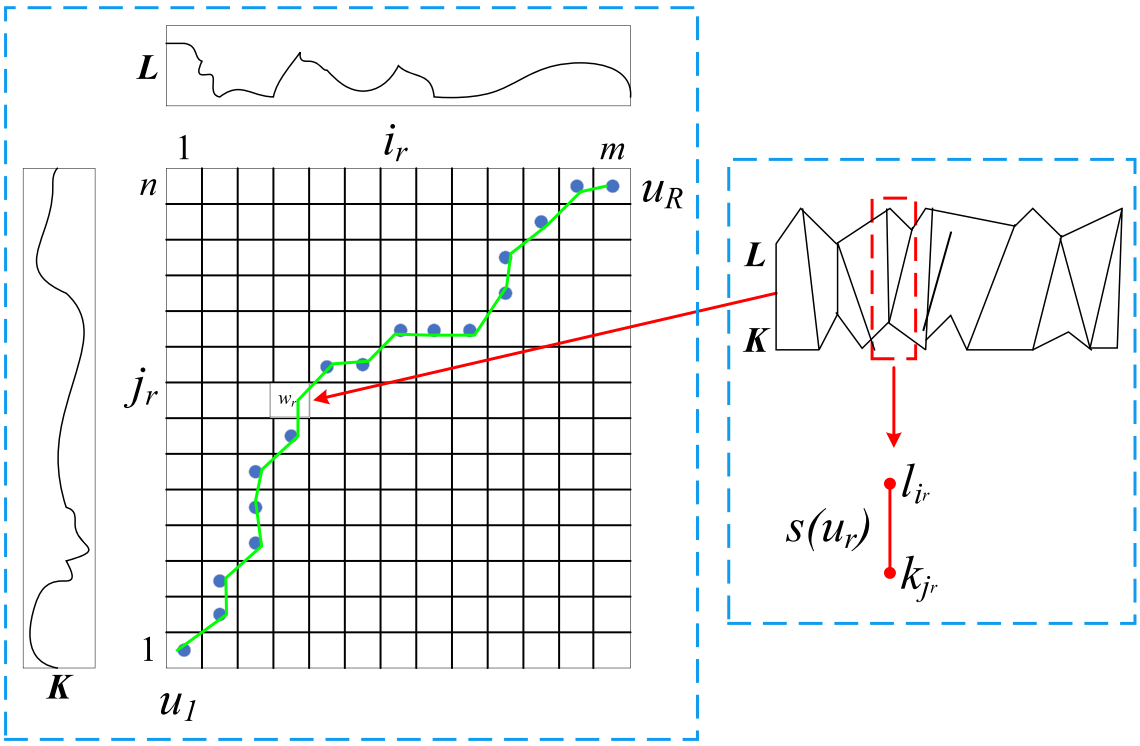


Fig. 5. Schematic diagram of paths in similarity matrix.

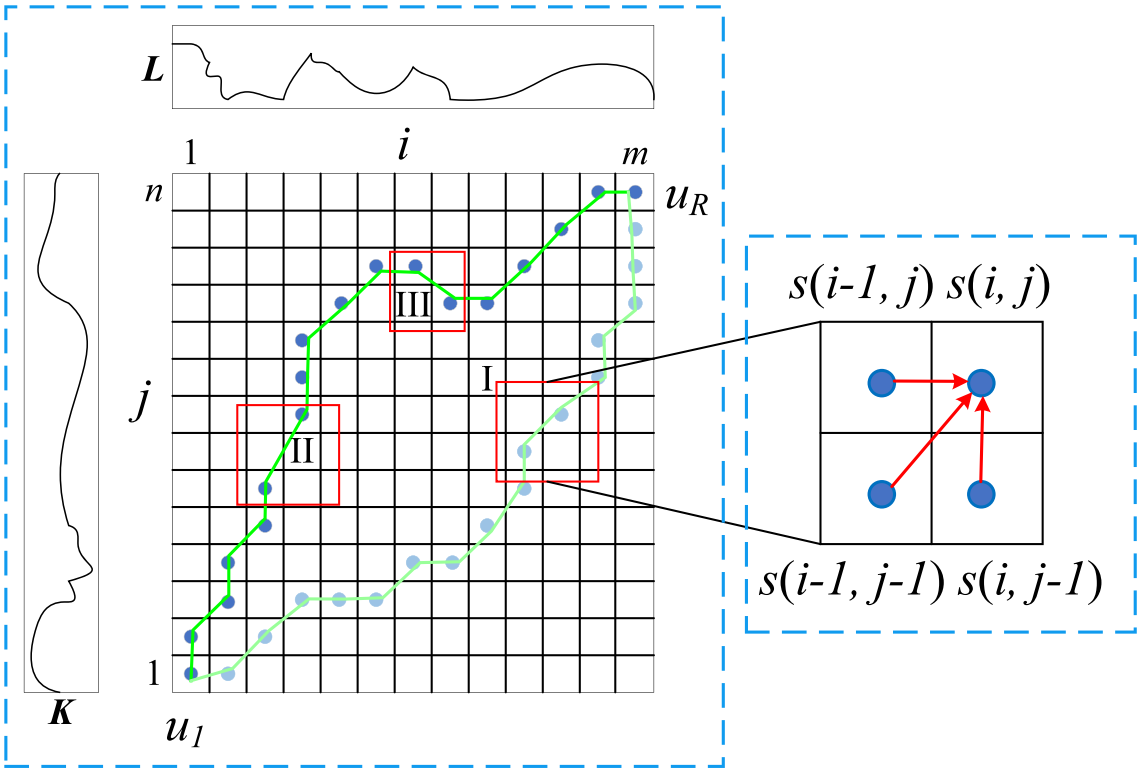


Fig. 6. Visual representation of the constraint conditions of effective paths in similarity matrix.

starting point. Repeat the above process and traverse the entire similarity matrix S, and you can find a path from $S(1,1)$ and, starting from the optimal path of $S(n,m)$, the value of $r(n,m)$ is finally obtained, and the similarity $S_{seq}(L,K)$ between the fingerprint sequences L and K is also obtained.

After solving $S_{seq}(L,K)$, the obtained optimal path is expressed by $\widehat{U} = \{\widehat{u}_1, \widehat{u}_2, \cdots, \widehat{u}_{r-1}, \widehat{u}_r, \widehat{u}_{r+1}, \cdots, \widehat{u}_{R-1}, \widehat{u}_R\}$, so the similarity between the two fingerprint sequences L and K can be as:

$$S_{seq} = \frac{1}{R}\cdot\sum_{r-1}^{R} s(\widehat{u}_r) \tag{7}$$

$$\frac{1}{R}\cdot\sum_{r-1}^{R} s(\widehat{u}_r) = \frac{1}{R}\sum_{r-1}^{R}\prod_{z=1}^{Z}\exp\left\{-\frac{[l_z(i_r)-k_z(j_r)]^2}{2\sigma^2}\right\} \tag{8}$$

If the similarity value of the two fingerprint sequences is higher than a certain threshold value v_s, the two positions are considered to be the same, and a closed-loop fingerprint sequence is detected. However, in the actual process, these two positions are often impossible to be in the same place, so an uncertainty covariance parameter is added to compensate for this error. The uncertainty covariance matrix is usually a symmetric matrix. In practical applications, a diagonal matrix with a large value can be selected. The size of the diagonal matrix elements indicates the degree of neglect of this error. If the similarity of the two fingerprint sequences is very high and the corresponding ground real value is less than 3 m, it indicates that the actual distance between the two fingerprint sequences is very close. It is considered that the error at this time is small and can be ignored, so the diagonal array element is set to 1. Both the Gaussian closed loop and the fingerprint sequence closed loop are added to the pose map as constraints. For the Gaussian closed loop and the fingerprint sequence closed loop, the diagonal matrix elements are set as the reciprocal of the variance of the similarity samples. The uncertainty model of similarity and distance is set up through a training process. This is done to improve the accuracy of the system and find out the mean and variance of the similarity samples.

3.2. Training of uncertainty model

Each edge in a graph-based SLAM needs a specification of its uncertainty. For the edge constructed by an odometer, its uncertainty can be obtained from the motion model. In the YOLOv5 network, the backbone network is the core component of the entire model, which determines the performance and speed of the model. But, the running YOLOv5 in prediction mode does not necessitate a persistent network connection. So, in order to improve the accuracy of the system, an uncertainty model of similarity and distance is established through a training process. The effectiveness of the method proposed in this study was verified using SLAM's commonly used TUM dataset. It is divided into low-dynamic sitting sequences and high-dynamic walking sequences [29]. Compared with the graph-based SLAM system, the accuracy of the SLAM system was evaluated using absolute trajectory error (ATE) and relative pose error (RPE). ATE represents the error between the actual trajectory coordinates and the estimated coordinates, and RPE represents the error between the real pose transformation and the estimated pose transformation. The root mean square error is:

$$e_{\mathrm{ATE}} = \sqrt{\frac{1}{n}\sum_{i}^{n}\left(x_i - \widetilde{x}_i\right)^2} \tag{9}$$

$$e_{\mathrm{RPE}} = \sqrt{\frac{1}{n}\sum_{i,j}\left(\delta_{i,j} - \widetilde{\delta}_{i,j}\right)^2} \tag{10}$$

here, x_i represents the real position of the camera; $\widetilde{x}_i$ is the estimated position of the camera. $\delta_{i,j}$ is the real pose transformation value at time i and j; $\widetilde{\delta}_{i,j}$ represents the relative pose transformation value estimated by the SLAM system.

Although the odometer may have accumulated errors over time, it is very accurate over a short period of time. It is known that the odometer is still very accurate within the range of 30 m. Therefore, the similarity of all fingerprints with a distance of less than 30 m is calculated. So, we get K training data:$\{S_k, d_k\}_{k=1}^{K}$, where S_k stands for how similar two fingerprints are and d_k stands Euclidean distance. Then the binning method is used to train the samples, and a model is obtained to represent the uncertainty of the distance corresponding to a certain similarity. That is, given a similarity S, calculate the mean $\widehat{d}(s)$ and variance $var(s)$ of all samples with an interval of b from the similarity S, and calculate the covariance matrix $\sum$ in formula (1) through the mean and variance.

$$\widehat{d}(s) = \frac{1}{c(b(s))}\sum_{k\in b(s)} d_k \tag{11}$$

$$var(s) = \frac{1}{c(b(s))}\sum_{k\in b(s)}\left(d_k - \widehat{d}(s)\right)^2 \tag{12}$$

where, $c(b)$ counts the number of samples whose similarity falls in interval b. See Fig. 7 for the uncertainty model established by this method.

4. Experiment and result analysis

In order to verify the performance of the algorithm proposed in this article, two sets of experiments were carried out in the Hubei key laboratory of advanced technology for automotive components test lot (about 100 m^2) of Wuhan University of Technology in China. The designed data acquisition platform such as Bulldog is shown in Fig. 8. The robot development platform, bulldog automated guided vehicle, is selected as the experimental verification platform. The robot obtains odometer information through the chassis, the system environment is Ubuntu 18.04 (developed by British company Canonical), and the data acquisition frequency is set to 10 Hz. The robot is equipped with a ground mobile platform, LiDAR, industrial camera, differential GPS, IMU, laser sensor, and six cell phones. The laser sensor is used to realize AMCL (adaptive monte Carlo localization) and takes the attitude obtained as the real value of the mobile robot on the ground in the environment. The platform has indoor and outdoor positioning and navigation capabilities, developed based on an open-source robot operating system (ROS), providing rich sensor interfaces. It is a time-of-flight active (i.e., a sensor that emits energy into the environment and measures the environmental reaction) sensor that performs a two-dimensional scan of its surrounds. The time it takes for the light to go from its source to an obstruction and back again is what the time-of-flight principal measures. To determine the distance to an object, one measures how long it takes for a laser pulse to travel from the emitter to the receiver. Sensor has a 180-degree field of view. Since the laser range sensors are mounted side by side, the robot may estimate distance in any direction other than directly to the left and right (see Fig. 8). Cell phones collect Wi-Fi fingerprint data at a frequency of 0.5 Hz. The robot started

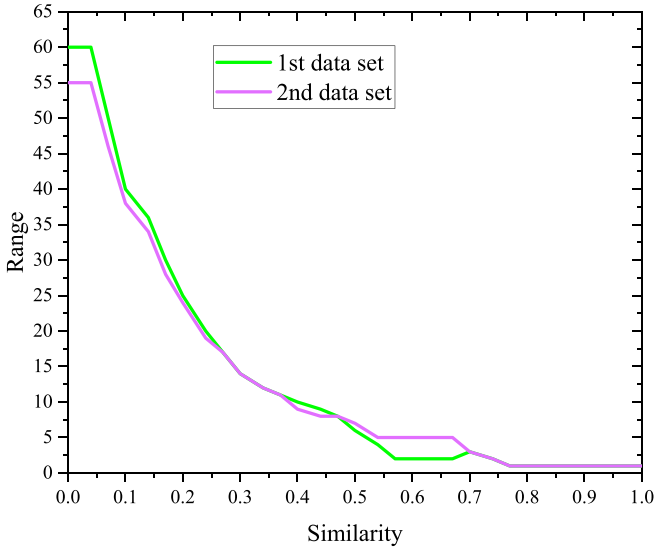


Fig. 7. Fingerprint similarity distance model.

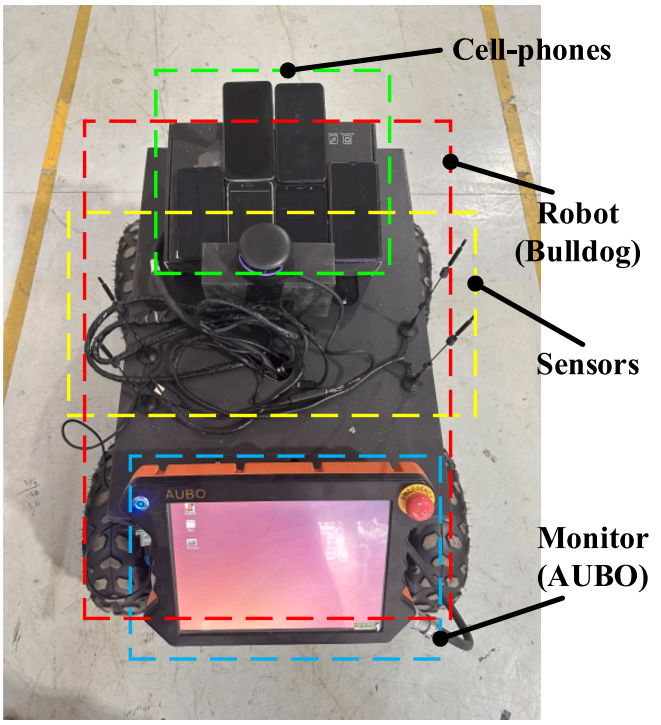


Fig. 8. Bulldog data acquisition platform.

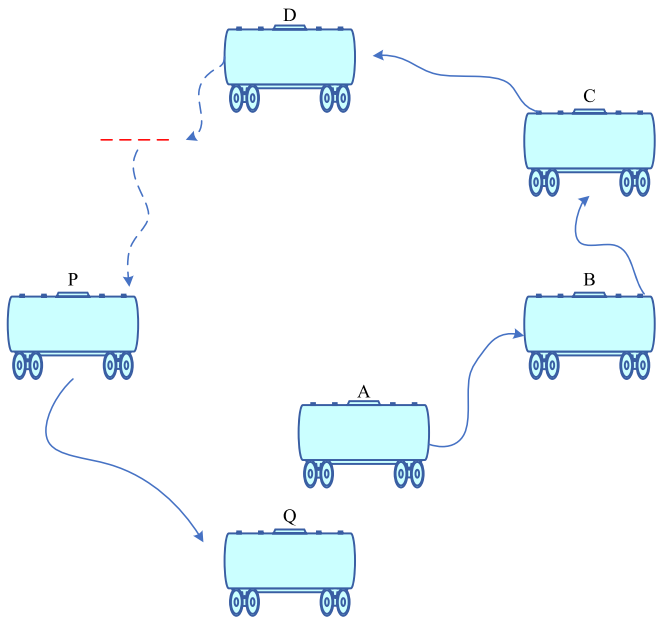


Fig. 9. The loop closing problem, after exploring its surroundings, the robot eventually reaches destination Q. The key finding is whether or not the robot has revisited a previously explored location.

Fig. 10. Panorama with bulldog automated guided vehicle, ground observation points and obstacles to be separated.

from different starting points and moved at an average speed of 0.4 *m/s*. Two sets of experimental data were collected. This scheme enables Bulldog to have autonomous driving ability, be able to avoid obstacles, plan driving paths, recognize path signs, and respond accordingly.

Based on graph optimization SLAM and YOLOv5 transformations, the robot processes and analyzes the surrounding point cloud information and multi-frame image information and integrates relevant speed, angle, mileage, and other information to achieve positioning and navigation. The three stages are involved in fingerprint development. The initial stage in creating a fingerprint is extracting the various features (such as vertical edges, corners, and color patches) from the sensors. The retrieved features are ranked in order from 0 to 360 degrees in terms of their angular position. Second, the concept of a "virtual feature" is introduced. This is because an edge in the unwrapped omnidirectional image corresponds to a corner in the laser scanner's output. The concept of empty space is introduced to fingerprints to better describe the vast ($>20^{\circ}$) angular gaps between consecutive fingerprint elements. Fingerprints become more distinguishable when the order of their features in a series is taken into account. The production of a fingerprint is complete with this final addition. Mobile robots use exteroceptive sensor data to interact with their environment. Since sensors are not perfect, the results will always have some degree of error. The graph optimization SLAM's primary problem is identifying a previously visited spot if the robot returns. Since the robot loops, this is called the closing the loop problem. So, for topological maps, the robot should detect a previously visited node if it returns to it (see Fig. 9).

First, the method of closed-loop fingerprint sequence ($n = 20$, $m = 25$) is compared with the method of Gaussian closed-loop only, and the algorithm accuracy is greatly improved. The robot creates the topological map as it moves. The probability distribution may split when the robot returns to a node. Two hypotheses should appear: one for the robot's new node (Fig. 9, node Q) and one for the previously produced node already in the map (e.g., in Fig. 9, node A). The program monitors the two largest probability distributions. If the localizer's probability distribution has two peaks in the same direction, a loop is found. The two possibilities are localized until one remains to find where the loop was closed. The location of the experiment can be seen in Fig. 10.

As can be seen, eight observation locations have been chosen for the ground and the various obstacles that are present in the environment, seven of which are located on the ground, and one of which is scattered on the obstacle and obstacle plane, as shown in Fig. 10. Make the mobile robot travel around the room from different positions; create a point cloud map to cut off the ground; and repeat 10 times. With this method, the ground observation points can be intercepted, the obstacle observation points can be retained, and the correct separation can be achieved. The DTW algorithm is used to extract the objects with obstacles and corresponding planes intercepted on the ground three times, and the ground observation points are omitted. The probability of correct separation is 70 %. Length and angle errors measure map accuracy. We take four measurement line segments and two measurement angles, including one included angle, and compare them to the actual scene. See Table 1 for the experimental results. It can be seen from Table 1 that if v_s is too large or too small, the accuracy of the algorithm will be reduced. For different experimental data, the optimal similarity threshold is also different. When v_s is greater than 0.70, the system has not detected the closed loop, resulting in a large error, which is 18.99 *m* (the first group of data) and 10.22 *m* (the second group of data). When$v_s = 0.10$, the SLAM algorithm accuracy of the first group of data is the highest, which can reach 3.10 m; when$v_s = 0.20$, the SLAM algorithm accuracy of the second group of data is the highest, which can reach 2.68 m. Then, Gauss's closed loop and fingerprint sequence closed loop are added to

Table 1
Effect of different closed-loop detection methods and v_s on the accuracy of SLAM algorithm.

Data sets	v_s	SLAM algorithm accuracy / m					
		Gauss	Fingerprint sequence	Gauss + fingerprint sequence	Gauss + fingerprint sequence (add interference)		
					Delete some APs	Gaussian noise 3	Gaussian noise 5
1st	0.10	**4.02**	**3.25**	3.19	**3.34**	3.39	3.31
	0.20	4.42	3.40	**3.08**	3.46	**3.03**	**3.10**
	0.30	4.23	3.44	3.42	3.52	3.33	3.46
	0.40	4.41	3.59	3.44	3.94	3.57	3.79
	0.50	4.44	4.10	3.98	4.32	4.17	4.30
	0.60	5.76	5.82	5.68	5.65	4.99	5.45
	0.70	18.99	18.99	18.99	18.99	18.99	18.99
2nd	0.10	9.14	5.65	5.26	5.89	5.14	6.33
	0.20	3.92	**2.68**	**2.37**	2.83	**2.42**	**2.62**
	0.30	**3.89**	2.82	2.66	**2.75**	2.57	2.68
	0.40	4.19	3.88	3.03	3.50	3.32	3.75
	0.50	4.44	4.95	4.23	4.35	3.99	4.37
	0.60	8.89	8.10	7.87	7.86	7.86	7.73
	0.70	10.22	10.22	10.22	10.22	10.22	10.22

the pose map as constraints. The results obtained by using this method are as follows: The first group of data is 3.08 *m*, which is 22.95 % higher than the method using Gauss only (accuracy: 4.02 *m*); the second set of data is 2.37 *m*, which is 39.19 % higher than the method using only Gauss (accuracy: 3.89 *m*).

In order to verify whether the algorithm accuracy of the experimental environment will be affected after the interference, three kinds of interference are added to the original Wi-Fi signal strength. The first is to delete some APs based on the originally scanned APs. A total of 8618 fingerprint points were collected in the experiment. The number of APs scanned by each fingerprint point is within the range of (40140), so 5 APs are randomly deleted from the AP scanned by each fingerprint point. The second is to add Gaussian noise with a variance of 3 to the signal strength. The third is to add Gaussian noise with a variance of 5 to the signal strength. The experimental results are shown in Table 1. Under three interference conditions, the accuracy of the SLAM algorithm obtained from the first group of experimental data is 3.34 *m*, 3.03 *m*, and 3.10 *m*, and the accuracy of the SLAM algorithm obtained from the second group of experimental data is 2.75 *m*, 2.42 *m*, and 2.62 *m*. The experimental results show that the whole SLAM algorithm has high stability when the environment is disturbed. Although the original Wi-Fi data (signal strength and number of APs) are processed to some extent, the applicability and stability of the whole SLAM algorithm are verified because there is reliable fingerprint closed-loop detection to eliminate the deviation of the data in the later stage, and the trajectory of the robot is optimized by the above optimization algorithm.

As shown in Table 2, it can be seen that the number of closed loops corresponding to the fingerprint sequence is less than the number of closed loops corresponding to Gauss. This is because the fingerprint sequence contains multiple continuous fingerprint points. The original single fingerprint point can form a closed loop, but now it needs multiple fingerprint points to form a closed loop, which leads to a great reduction in the number of closed loops obtained by the algorithm. Therefore, the number of "fingerprint closed loop" edges added to the pose map is also less. Moreover, the similarity threshold v_s also has a certain impact on the closed-loop data. The smaller the similarity threshold v_s, the greater the number of closed loops. The larger the similarity threshold v_s, the fewer the closed loops. When the similarity threshold v_s is greater than 0.70, the system cannot detect closed loops, so the number of closed loops is 0.

Table 2
Effect of different similarity detection methods and v_s on the number of closed-loop.

v_s	Number of closed loops (first group)		Number of closed loops (second group)	
	Gaussian	Fingerprint sequence	Gaussian	Fingerprint sequence
0.10	124,130	95,611	123,983	85,482
0.20	46,242	31,268	24,345	18,875
0.30	24,463	19,833	9376	7556
0.40	9310	4572	3035	1448
0.50	1780	612	609	247
0.60	36	0	32	0
0.70	0	0	0	0
0.80	0	0	0	0
0.90	0	0	0	0

It can be seen from the results of two groups of experimental data that the optimized robot trajectory is basically consistent with the actual value on the ground. The Gaussian + fingerprint sequence is the closest to the actual value on the ground. The robot is shown a variety of indoor settings, including open and closed doors. At this stage, the robot is accumulating the values of its sensor variables that are indicative of a certain condition. The parameters of parametric models can then be determined using this data set δ.

As a result of recent breakthroughs in SLAM, researchers have taken a keen interest in bringing these techniques inside to improve indoor localization. In addition to facilitating the resolution of multi-dimensional optimization problems, deep learning permits the autonomous learning of complicated, non-linear relations or features inside a given data set. Complex filtering or machine learning algorithms address multi-path effects and device-dependent noise signal volatility. Due to algorithm performance restrictions, classical learning approaches for scalable localization in complicated and hierarchical contexts are difficult. Real-time fingerprint processes are impossible due to their great dimensionality. The dimension of a raw WiFi RSS file is the number of APs scanned in the area of interest. Because each AP covers a limited area, many RSS database elements are frequently empty. DTW's deeper functions that map input to output enable fingerprint-based indoor localization by learning signal fluctuations and ambient dynamics [30]. DNN algorithms can extract complicated fingerprint patterns from a huge set of noise samples, resulting in high accuracy.

In addition, the influence of the length of the fingerprint sequence on the accuracy of the proposed SLAM algorithm is also being studied. The Gauss + fingerprint sequence method is adopted to take five groups of data, respectively. The corresponding algorithm results are shown in Table 3. It can be seen that when $n = 20$ and $m = 25$, the algorithm accuracy is the highest. When the fingerprint sequence is too short, the amount of fingerprint information contained in it is not enough, resulting in the inaccuracy of closed-loop detection. When the length of the fingerprint sequence is too long, the computational complexity will be increased due to the large amount of fingerprint data, and the accuracy of the SLAM algorithm will be reduced. When three kinds of interference are added to the experimental data, the accuracy of the

Table 3
Effect of the length of fingerprint sequence on the accuracy of SLAM algorithm.

(n,m)	SLAM algorithm accuracy (first group, v_s= 0.10) / m				SLAM algorithm accuracy (second group, v_s= 0.20) / m			
	Gauss + fingerprint sequence	Gauss + fingerprint sequence (add interference)			Gauss + fingerprint sequence	Gauss + fingerprint sequence (add interference)		
		Delete some Aps	Gaussian noise 3	Gaussian noise 5		Delete some APs	Gaussian noise 3	Gaussian noise 5
(10,15)	3.40	3.74	3.45	3.49	2.67	2.86	3.11	3.25
(20,25)	**3.08**	**3.24**	**3.03**	**3.10**	**2.37**	**2.75**	2.43	2.62
(30,35)	3.52	3.39	3.59	3.57	2.96	2.91	**2.40**	**2.58**
(40,45)	3.56	3.83	3.75	3.80	3.00	3.47	2.82	3.24
(50,55)	3.64	4.02	3.96	4.22	3.25	3.40	3.36	3.59

optimal SLAM algorithm obtained from the first group of experimental data is 3.23 *m*, 3.03 *m*, and 3.10 *m*, 2.37 *m*, and the accuracy of the SLAM algorithm obtained from the second group of experimental data is 2.75 *m*, 2.40 *m*, and 2.58 *m*. The experimental results show that the optimized algorithm has high stability in the case of interference. Even though the length of the sequence that the optimal algorithm needs to be accurate has changed, it is still much better than the Gaussian method, which is based on a single fingerprint point.

5. Conclusion

This article presents a graph optimization SLAM algorithm based on deep learning and Wi-Fi fingerprint sequence matching. First, the Wi-Fi fingerprint information is collected by the cell phone on the robot, and the DWT algorithm realizes the closed-loop detection of the fingerprint sequence. Then, the Gaussian closed-loop and the closed-loop of the fingerprint sequence are added to the pose map as constraints, and the trajectory of the robot is optimized by the graph optimization algorithm. Additionally, YOLOv5 algorithm is employed to identify potential targets by using existing information. Then graph-based SLAM tracking thread uses Wi-Fi fingerprint information and YOLOv5 to minimize complex feature points. Finally, the optimized trajectory of the robot is obtained. This study combines primary data sets with TUM dynamic data sets to simulate and evaluate the real targets within indoor environments. The proposed algorithm is verified by two sets of experimental data, and the positioning accuracy can reach 3.08 *m* and 2.37 *m*, respectively, which is 22.95 % and 39.19 % higher than the method using only Gauss. This paper shows that the algorithm significantly improves SLAM accuracy in a large-scale complex environment. In follow-up research, the integration of digital twin and ROS transformations is employed to enhance precision, facilitating the application of this robotic technology in the healthcare sector.

Funding

This work was supported in part by the Natural Science Foundation of China (61876137), and Foshan Xianhu Laboratory of the Advanced Energy Science and Technology Guangdong Laboratory (XHD2020-003).

CRediT authorship contribution statement

Muhammad Usman Shoukat: Visualization, Software, Data curation. **Lirong Yan:** Supervision, Project administration, Investigation, Funding acquisition. **Di Deng:** Visualization, Validation, Formal analysis. **Muhammad Imtiaz:** Visualization, Software, Data curation. **Muhammad Safdar:** Visualization, Software, Data curation. **Saqib Ali Nawaz:** Visualization, Methodology, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

The author(s) wish to acknowledge Prof. Fuwu Yan (Hubei Key Laboratory of Advanced Technology for Automotive Components, Wuhan University of Technology, Wuhan 430070, China) for providing the facilities and funding needed for this research work.

References

[1] Stachniss, C., Leonard, J. J., & Thrun, S. (2016). Simultaneous localization and mapping. In Springer Handbook of Robotics (pp. 1153-1176). Springer, Cham.
[2] H.A. Hashim, A.E. Eltoukhy, Nonlinear filter for simultaneous localization and mapping on a matrix lie group using imu and feature measurements, IEEE Trans. Syst. Man Cybernet.: Syst. 52 (4) (2021) 2098–2109.
[3] J. Vallvé, J. Solà, J. Andrade-Cetto, Pose-graph SLAM sparsification using factor descent, Robot. Autonomous Syst. 119 (2019) 108–118.
[4] Mendes, E., Koch, P., & Lacroix, S. (2016, October). ICP-based pose-graph SLAM. In 2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR) (pp. 195-200). IEEE.
[5] K. Harsányi, A. Kiss, T. Szirányi, A. Majdik, MASAT: A fast and robust algorithm for pose-graph initialization, Pattern Recogn. Lett. 129 (2020) 131–136.
[6] B. Fang, G. Mei, X. Yuan, L. Wang, Z. Wang, J. Wang, Visual SLAM for robot navigation in healthcare facility, Pattern Recogn. 113 (2021) 107822.
[7] L. Chang, X. Niu, T. Liu, J. Tang, C. Qian, GNSS/INS/LiDAR-SLAM integrated navigation system based on graph optimization, Remote Sens. (Basel) 11 (9) (2019) 1009.
[8] A.J. Davison, I.D. Reid, N.D. Molton, O. Stasse, MonoSLAM: Real-time single camera SLAM, IEEE Trans. Pattern Anal. Mach. Intell. 29 (6) (2007) 1052–1067.
[9] Klein, G., & Murray, D. (2007, November). Parallel tracking and mapping for small AR workspaces. In 2007 6th IEEE and ACM international symposium on mixed and augmented reality (pp. 225-234). IEEE.
[10] Milford, M. J., Wyeth, G. F., & Prasser, D. (2004, April). RatSLAM: a hippocampal model for simultaneous localization and mapping. In IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 (Vol. 1, pp. 403-408). IEEE.
[11] Newcombe, R. A., Lovegrove, S. J., & Davison, A. J. (2011, November). DTAM: Dense tracking and mapping in real-time. In 2011 international conference on computer vision (pp. 2320-2327). IEEE.
[12] Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., & Fitzgibbon, A. (2011, October). Kinectfusion: Real-time dense surface mapping and tracking. In 2011 10th IEEE international symposium on mixed and augmented reality (pp. 127-136). Ieee.
[13] R. Mur-Artal, J.M.M. Montiel, J.D. Tardos, ORB-SLAM: a versatile and accurate monocular SLAM system, IEEE Trans. Rob. 31 (5) (2015) 1147–1163.
[14] Kundu, A., Krishna, K. M., & Sivaswamy, J. (2009, October). Moving object detection by multi-view geometric techniques from a single camera mounted robot. In 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 4306-4312). IEEE.
[15] T. Wei, X. Li, Binocular vision SLAM algorithm based on dynamic region elimination in dynamic environment, Robot 42 (3) (2020) 336–345.
[16] Yagfarov, R., Ivanou, M., & Afanasyev, I. (2018, November). Map comparison of lidar-based 2d slam algorithms using precise ground truth. In 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV) (pp. 1979-1983). IEEE.
[17] Wijaya, S. F. A. E., Purnomo, D. S., Utomo, E. B., & Anandito, M. A. (2019, September). Research Study of Occupancy Grid map Mapping Method on Hector SLAM Technique. In 2019 International Electronics Symposium (IES) (pp. 238-241). IEEE.

[18] Hull, G. (2017). Real-time occupancy grid mapping using LSD-SLAM (Doctoral dissertation, Stellenbosch: Stellenbosch University).
[19] Chen, X., Ma, H., Wan, J., Li, B., & Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (pp. 1907-1915).
[20] Hu, M., Li, S., Wu, J., Guo, J., Li, H., & Kang, X. (2019, July). Loop closure detection for visual SLAM fusing semantic information. In 2019 Chinese Control Conference (CCC) (pp. 4136-4141). IEEE.
[21] Z. Han, Multimodal intelligent logistics robot combining 3D CNN, LSTM and visual SLAM for path planning and control, Front. Neurorob. 17 (2023) 1285673.
[22] J. Jiao, C. Wang, N. Li, Z. Deng, W. Xu, An adaptive visual dynamic-SLAM method based on fusing the semantic information, IEEE Sens. J. 22 (18) (2021) 17414–17420.
[23] Soares, J. C. V., Gattass, M., & Meggiolaro, M. A. (2019, December). Visual SLAM in human populated environments: exploring the trade-off between accuracy and speed of YOLO and Mask R-CNN. In 2019 19th International Conference on Advanced Robotics (ICAR) (pp. 135-140). IEEE.
[24] I.A. Kazerouni, L. Fitzgerald, G. Dooly, D. Toal, A survey of state-of-the-art on visual SLAM, Expert Syst. Appl. 205 (2022) 117734.
[25] Z. Wang, Q. Zhang, J. Li, S. Zhang, J. Liu, A computationally efficient semantic slam solution for dynamic scenes, Remote Sens. (Basel) 11 (11) (2019) 1363.
[26] R. Mur-Artal, J.D. Tardós, Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras, IEEE Trans. Rob. 33 (5) (2017) 1255–1262.
[27] F. Lu, E. Milios, Globally consistent range scan alignment for environment mapping, Auton. Robot. 4 (4) (1997) 333–349.
[28] S.A. Nawaz, J. Li, U.A. Bhatti, M.U. Shoukat, R.M. Ahmad, AI-based object detection latest trends in remote sensing, multimedia and agriculture applications, Front. Plant Sci. 13 (2022) 1041514.
[29] Sturm, J., Engelhard, N., Endres, F., Burgard, W., & Cremers, D. (2012, October). A benchmark for the evaluation of RGB-D SLAM systems. In 2012 IEEE/RSJ international conference on intelligent robots and systems (pp. 573-580). IEEE.
[30] W. Zhang, K. Liu, W. Zhang, Y. Zhang, J. Gu, Deep neural networks for wireless localization in indoor and outdoor environments, Neurocomputing 194 (2016) 279–287.