Received 22 May 2024, accepted 7 June 2024, date of publication 17 June 2024, date of current version 30 October 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3415416

RESEARCH ARTICLE

A Multi-Sensor Deep Fusion SLAM Algorithm Based on TSDF Map

YIBO CAO, ZHENYU DENG, ZEHAO LUO, AND JINGWEN FAN
School of Software, South China Normal University, Guangzhou, Guangdong 510631, China

Corresponding author: Yibo Cao (422705879@qq.com)

This work was supported in part by the Key Technology Research (Second Batch) Program of Ningbo under Grant 2021E007; and in part by the Scientific Research Foundation of Graduate School, South China Normal University, under Grant 21RJKC15.

ABSTRACT Traditional 2D Simultaneous Localization and Mapping (SLAM) algorithms commonly use occupancy grid map models, which are susceptible to Gaussian noise. The constraint information in the backend optimization process is limited. Sensor data utilization at various stages is also incomplete. To address these issues, this paper proposes a multi-sensor deep fusion SLAM method based on the Truncated Signed Distance Function (TSDF) map. Firstly, the inertial sensing unit (IMU) is pre-integrated, and then the distortion correction of the laser point cloud is corrected by using the posture obtained after pre-integration. In the front-end, the Unscented Kalman Filter (UKF) method is used to fuse odometry, IMU data, and LiDAR scan matching results to obtain the pose information of the robot. The backend uses IMU pre-integration factor, loopback detection, and laser point cloud registration to enhance constraints for global map pose optimization and achieve deep fusion of multi-sensor data. The map model uses a TSDF map, which constructs obstacle edges through weighted fusion and linear interpolation, and it truncates the grid cells around obstacles, thereby reducing the influence of Gaussian noise. The performance of Karto-SLAM, Cartographer, and the proposed algorithm is verified by comparing the public dataset and the dataset collected in the real environment. The results show that the proposed method effectively avoids the ghosting phenomenon of traditionally occupied raster maps and reduces Gaussian noise in terms of mapping. In terms of positioning accuracy, the effect of back-end optimization is enhanced by a multi-constraint relationship, which reduces the relative and absolute pose errors of the real trajectory. Our method improves localization accuracy by an average of 9% compared to Cartographer and by an average of 34% compared to Karto-SLAM.

INDEX TERMS Truncate signed distance function map, multi-sensor fusion, simultaneous location and mapping, inertial measurement unit pre-integrated.

I. INTRODUCTION

In recent years, with the advancement of hardware and improvements in computer technology, the application scope of mobile robots has expanded significantly. Among these applications, Simultaneous Localization and Mapping (SLAM) serve as the foundation for achieving autonomous navigation in mobile robots [1], [2], [3]. Common SLAM systems are generally divided into five parts: sensor data acquisition and processing, front-end odometry, back-end optimization, loop closure detection, and map construction. LiDAR and other auxiliary sensors collect data, which is then processed and analyzed by the front-end odometry. Using scan matching algorithms, the relative pose transformation between consecutive LiDAR data frames is quickly estimated. However, the computed pose contains cumulative errors and is not accurate enough. The back-end optimization is responsible for global trajectory optimization to obtain precise poses and build a globally consistent map. Throughout this process, loop closure detection is continuously executed to recognize revisited scenes, achieve loop closure, and eliminate cumulative errors.

During the SLAM process, the inherent characteristics of a single sensor impose limitations on the perception of the

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

surrounding environment and the estimation of the robot's pose [4]. This limitation raises concerns about the accuracy and robustness of SLAM algorithms. For instance, common single-line lidars, due to their lower sampling frequency, are prone to motion distortion during robot movement. Additionally, their reliance on data association between consecutive frames can affect the matching efficiency of lidar point clouds in environments with similar features and limited measurement ranges. Inertial Measurement Units (IMUs) exhibit drawbacks such as zero-offset, susceptibility to temperature influences, and cumulative errors. Wheel encoders, commonly used as odometry sensors, may introduce significant errors in dead reckoning due to wheel slippage or uneven road surfaces. The adoption of a multi-sensor fusion approach allows for the integration of inertial and odometric data, providing prior information for lidar point cloud matching, correcting lidar data distortion, and mitigating accumulated errors. Simultaneously, the long-term stable distance information from lidar data can reduce cumulative errors. Therefore, the effective fusion of multi-sensor data, achieving complementarity and the integration of non-homogeneous data, remains a focal point in current research on SLAM algorithms for mobile robots [5], [6], [7], [8].

Scan matching technology, which aligns consecutive frames of laser point clouds to estimate pose transformations, was initially implemented using the Iterative Closest Point (ICP) algorithm [9]. This method searches for corresponding matching points between frames based on the nearest neighbor principle and constructs a cost function for optimization. However, this approach suffers from drawbacks such as high computational complexity and dependence on initial values. Point-to-Line ICP (PL-ICP) is an optimized version of ICP that differs in its use of the minimum point-to-line distance for ICP computation, ensuring faster convergence speed and higher accuracy [10]. Wei et al. [11] utilized the PL-ICP algorithm in the early stages of Hector-SLAM to adjust its orientation and position, achieving a certain degree of correction for trajectory and pose. However, due to the absence of a loop closure detection process, this method still cannot eliminate trajectory deviation caused by accumulated errors. Several scholars have proposed the fusion of IMU observation information with lidar, odometry, and camera data at the front end of SLAM. However, most of these approaches achieve fusion at a specific stage and do not fully leverage the IMU's observation data [12], [13], [14]. Literature [15] introduced Ultra Wide Band (UWB) as prior information into point cloud registration, effectively ensuring the point cloud matching process. However, this information was not integrated into backend optimization, leading to insufficient fusion of sensor data. The Cartographer algorithm, developed by Google, is a laser SLAM algorithm based on a graph optimization framework. It employs filtering methods during data preprocessing to fuse IMU and odometry data. Subsequently, during frontend frame-to-frame matching, lidar data interpolation corrects distortions in lidar point cloud data, followed by the construction of a series of sub-maps [16]. However, its backend only achieves loop closure constraints through frame-to-submap and submap-to-submap matching relationships. As a result, this method fails to fully utilize IMU and odometry data for multi-sensor depth fusion, overly relying on lidar data and leading to inaccurate or failed loop closures in environments such as long corridors. This paper not only fuses odometry data, IMU data, and LiDAR scan matching results in the front-end to output higher precision pose information of the robot, but also enhances constraints for global map pose optimization in the back-end through IMU pre-integration factors, loop closure detection, and inter-frame matching. By fully utilizing data from IMU and other sensors, the approach achieves deep multi-sensor fusion.

In the realm of data fusion, existing fusion methods include weighted averaging, filter-based methods, Bayesian estimation, and neural networks, among others [17], [18], [19], [20]. The use of filtering methods allows for the clever fusion of observation and estimation data, effectively managing errors and constraining them within a certain range. Filtering methods are currently widely used in multi-sensor data fusion. Common filtering techniques include Kalman Filtering (KF) [21], Extended Kalman Filtering (EKF), and Unscented Kalman Filtering (UKF) [22]. EKF is an extension of Kalman Filtering designed to handle nonlinear systems. UKF, in contrast to EKF, avoids the computation of Jacobian matrices by using a set of poses containing mean and covariance, enhancing system robustness and real-time performance.This paper effectively enhances the overall system performance by using the Unscented Kalman Filter (UKF) to fuse IMU data, odometry data, and LiDAR scan matching results.

In the selection of map representation models, grid maps and point clouds are the mainstream approaches for 2D laser SLAM map representation, as seen in algorithms such as Hector SLAM [23], Gmapping [24], Karto-SLAM [25], and the Cartographer algorithm. Probability grid maps use probability values to represent subgrids in a static environment. Their state updates involve simple addition and subtraction operations, making them easy to construct. However, their resolution cannot be changed, and real-time storage of probability values for all subgrids in the environmental map leads to a sharp increase in memory consumption. Additionally, the fixed resolution in the configuration file can result in jagged and overlapping surfaces of constructed obstacles, leading to suboptimal mapping results. Truncated Signed Distance Function (TSDF) provides an implicit computation method for establishing the surface of obstacles, allowing for memory savings and smoother boundary processing. Fossel et al. [26] introduced the 2D Signed Distance Function (SDF) map into laser SLAM algorithms for the first time, but the SDF map update strategy was not well-developed. Subsequent work by Daun et al. [27], Fu et al. [28], and others proposed new SLAM algorithms based on the TSDF map model and improved the map update strategy. This paper selects TSDF as the map model, effectively addressing issues such as susceptibility to Gaussian noise and poor mapping performance associated with probabilistic grid maps.

In summary, addressing the challenges in 2D laser SLAM, there is practical value in the algorithmic research on multi-sensor fusion for handling multi-sensor data and closing the loop with backend constraints. Building upon the foundation of the Cartographer algorithm, map modeling, and graph optimization research, this paper proposes a method for deep fusion based on sensors such as IMU and Lidar. The backend is augmented with additional constraints, and the map model selection involves real-time localization and mapping using a Truncated Signed Distance Function (TSDF) to ensure the comprehensive utilization of multiple data sources.

II. ALGORITHM OVERVIEW

A. ALGORITHM PROCESS

The proposed algorithm takes IMU data, odometry data, and 2D laser data as input and outputs the pose information of the carrier robot while continuously constructing a more refined Truncated Signed Distance Function (TSDF) map. The algorithm can be divided into the following five components:

1) Data Preprocessing: Data preprocessing involves synchronizing timestamps between IMU data and lidar point cloud data. Subsequently, preintegration processing is applied to the IMU data to obtain the relative pose transformation matrix of the robot. Based on this, distortion correction is performed on the lidar point cloud data to address low-frequency distortions.
2) Data Fusion: Utilizing Unscented Kalman Filter (UKF), the algorithm combines odometry, IMU information, and the results of lidar point cloud scan matching. This process outputs more accurate pose information for the carrier robot while providing a good initial value for subsequent optimization and loop closure processes.
3) Scan Matching: The algorithm employs the Point-to-Line Iterative Closest Point (PL-ICP) algorithm for inter-frame matching computations. This involves registering laser point clouds with the map to update the map through point clouds and map alignment.
4) TSDF Map Construction: Using Truncated Signed Distance Function (TSDF) as the map model, the algorithm employs a weighted approach to update a grid with multiple observed values.
5) Loop Closure Detection and Multi-Constraint Backend Optimization: Around the carrier robot, a window W is constructed to search for the best match between laser points and the TSDF map.If no matching constraint within the window results in an error smaller than a threshold, loop closure is considered unsuccessful. If successful, the loop closure constraint is added to the backend optimization. Frame-to-frame matching constraints, IMU preintegration factor constraints, and loop closure constraints are inserted into the backend to construct the target error function for optimization. The algorithm iteratively solves this function, achieving correction of the current submap pose node and optimization of the global map.

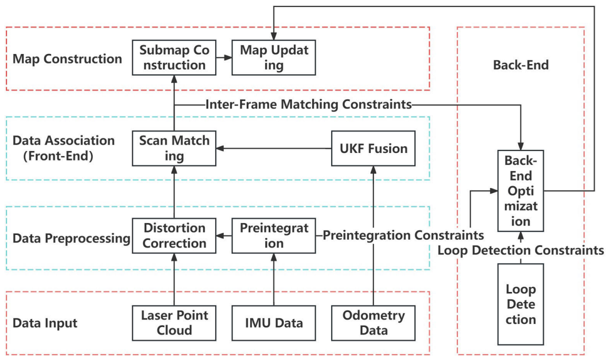


FIGURE 1. Block diagram of the system.

B. THE UNIQUENESS AND SUPERIORITY OF THE ALGORITHM

As shown in Table 1, the superiority of this algorithm mainly lies in the integrated multi-sensor data and the use of TSDF map, which is embodied in the following three points:

- Deep Fusion of Multi-Sensor Data: The method proposed in this paper uses the Unscented Kalman Filter (UKF) in the front-end to fuse odometry, IMU data, and LiDAR scan matching results to obtain the robot's pose information. In the back-end, IMU pre-integration factors, loop closure detection, and LiDAR point cloud matching are used to enhance the constraints for global map pose optimization, achieving deep fusion of multi-sensor data.
- Utilization of TSDF Maps: The paper employs Truncated Signed Distance Function (TSDF) maps, which enable the construction of obstacle edges through weighted fusion and linear interpolation. Additionally, around obstacles, grid cells are truncated, reducing the impact of Gaussian noise and avoiding the occurrence of map ghosting commonly associated with traditional occupancy grid maps. This approach effectively mitigates the influence of Gaussian noise.
- Improved Localization Accuracy: Through the fusion of multi-sensor data in the front-end, the algorithm outputs pose information with higher precision. Additionally, the enhancement of back-end optimization through multiple constraint relationships significantly reduces both relative and absolute position errors in the actual trajectory.

III. ALGORITHM DESIGN

In the experimental process, we define the global world coordinate system as Σ_M, and the coordinate system of

TABLE 1. Comparison of the characteristics of the algorithm in this paper with cartographer and Karto-SLAM.

Feature/Method	Ours	Cartographer	Karto-SLAM
Front-end sensor fusion	UKF fusion odometer, IMU data and lidar scanning matching results	The IMU and odometer data were fused using filter	No multi-sensor data fusion
Backend optimization	Global map pose optimization was enhanced with IMU pre-integration factor, closed-loop detection, and LiDAR point cloud matching data	The IMU data is not used on the backend	No multi-sensor data fusion
Map Representation	TSDF map, reduce Gaussian noise and avoid ghosting phenomenon	Occupancy grid maps and is susceptible to Gaussian noise.	Occupancy grid maps and is susceptible to Gaussian noise.
Positioning accuracy	On average, 9% higher than Cartographer and 34% higher than Karto-SLAM	The relative error is small, but IMU data is not fully utilized	The positioning accuracy is the lowest, especially when the error is more obvious in the outdoor environment

the carrier robot as Σ_M. The coordinate systems of the lidar and wheel odometry can be kept consistent with the robot coordinate system Σ_M through a stable coordinate transformation relationship. As for the IMU, since it is rigidly attached to the carrier robot, its measurement data can be considered as data in the carrier robot coordinate system Σ_M. Assuming that within the sampling period $[t_{k-1}, t_k]$ of the lidar, the collected lidar point cloud is represented as S_k, and the dataset collected by the IMU is denoted as $\Omega_{(k-1,k)}$. $\Omega_{(k-1,k)}$ consists of N sets of accelerations and angular increments in the Σ_M coordinate system $[a_{k_n}^W, \omega_{k_n}^W]$, where n = 1,2,3...N. If the global pose node of the system at time t_k is represented as X $=[x_k^W, y_k^W, \theta_k^W]$, indicating the robot's pose in the Σ_W coordinate system at time t_k, where θ_k^W is the yaw angle. If v_k^W represents the linear velocity of the robot in the Σ_W coordinate system at time t_k, the system model from pose node X_{k-1} to X_k at time t_k can be expressed as shown in equation 1:

$$X_k = \begin{bmatrix} x_k^W \\ y_k^W \\ \theta_k^W \end{bmatrix} = \begin{bmatrix} x_{k-1}^W - v_k^W \Delta t sin(\boldsymbol{\omega}_k^W \Delta t) \\ y_{k-1}^W \Delta t cos(\boldsymbol{\omega}_k^W \Delta t) \\ \theta_{k-1}^W + \boldsymbol{\omega}_k^W \Delta t \end{bmatrix} \quad (1)$$

A. DATA PREPROCESSING

1) IMU PREINTEGRATION

The observation equation for an Inertial Measurement Unit (IMU) sensor can be expressed as follows:

$$\begin{cases} a_{k_n}^W = R_{k_n}^{WM} a_{k_n}^M + b_{k_n}^a + \eta^a \\ \omega_{k_n}^{*W} = \omega_{k_n}^W + b_{k_n}^g + \eta^g \end{cases} \quad (2)$$

In equation 2, $b_{k_n}^a$ and $b_{k_n}^g$ represent the static biases (zero biases) of the accelerometer and gyroscope, respectively. η^a and η^g denote the observation noise of the accelerometer and gyroscope, while $\omega_{k_n}^{*W}$ represents the raw angular velocity observation. Here, $b_{k_n}^a$ and $b_{k_n}^g$ are both two-dimensional vectors, and $R_{k_n}^{WM}$ signifies the rotation matrix from the Σ_M coordinate system to the Σ_M coordinate system. During a single frame scan period of the lidar, the IMU data output frequency is much higher than the lidar scan data output frequency. Therefore, processing the IMU observation data through preintegration within the lidar single-frame data period can form inter-frame pose constraints. The relevant formulas are as follows:

$$\begin{cases} v_{k_{n+\delta}}^W = v_{k_n}^W + \left(\boldsymbol{R}_{t_{n+\delta}}^{WM} \boldsymbol{a}_{t_n}^M + \boldsymbol{b}_{t_n}^a + \boldsymbol{\eta}^a\right) \times \Delta t \\ \theta_{k_{n+\delta}}^W = \boldsymbol{\theta}_{k_n}^W + \left(\boldsymbol{\omega}_{k_n}^M + \boldsymbol{b}_{k_n}^g + \boldsymbol{\eta}^g\right) \times \Delta t \end{cases} \quad (3)$$

From equation 3, we can obtain the motion control quantity $u_{k+\delta} = \left[v_{k_{n+\delta}}^W, \theta_{k_{n+\delta}}^W\right]$, If the static biases in the IMU remain constant over the time interval Δt, then the preintegration observation model can be obtained by calculating $u_{k+\delta}$.

2) LIDAR POINT CLOUD DISTORTION CORRECTION

In the process of sampling a single-frame point cloud data with a lidar, there is a time delay. If the pose of the mobile robot changes within the lidar sampling period, the coordinates of the lidar in the Σ_W frame will also undergo corresponding changes. To ensure the accuracy of the laser input data and the precision of localization and mapping algorithms, it is necessary to perform distortion correction on the lidar point cloud.

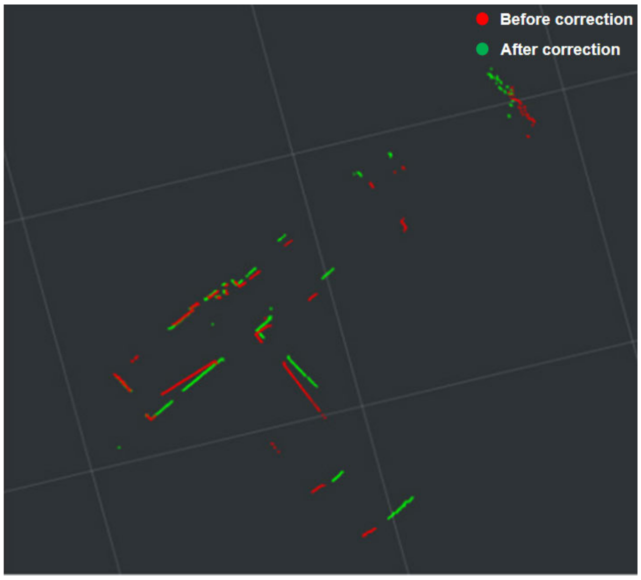


FIGURE 2. Correction effect of single-frame point cloud distortion.

The IMU data acquisition frequency can reach approximately 100-200 Hz, allowing for high-precision acquisition of the mobile robot's pose transformation during the scanning process in the short term. Therefore, IMU can be used to assist in the distortion correction of lidar point clouds. Before performing correction, it is necessary to synchronize the timestamps of the two sensors. Within the lidar single-frame sampling period $[t_{k-1}, t_k]$, the IMU can compute the relative pose transformation $T_{t_{k-1}}^{t_k}$ through preintegration. Assuming that the mobile robot undergoes constant linear motion during the time interval $t \in [t_{k-1}, t_k]$, on this basis, linear interpolation is used to calculate the pose transformation of any lidar point relative to the first lidar point. Subsequently, the current lidar point is transformed into the coordinate system corresponding to the first lidar point. The relevant computational formula (equation 4) is as follows:

$$\begin{cases} \boldsymbol{T}_i = \dfrac{t_k - t_i}{t_k - t_{k-1}} T_{t_{k-1}}^{t_k} \\ \boldsymbol{h}_i^* = \boldsymbol{T}_i \boldsymbol{h}_i \end{cases} \tag{4}$$

In the equation, t_i represents the timestamp of the i-th lidar point in a single-frame point cloud, T_i is the pose transformation matrix of that lidar point relative to the first lidar point in the frame, h_i denotes the position of the lidar point without distortion correction, and h_i^* represents the position obtained after correction. The distortion correction of the lidar point cloud is accomplished by utilizing equation 4 to transform the pose of all lidar points in a single frame, the distortion correction of the lidar point cloud is accomplished. Experimental results indicate that the distortion correction effect of a single-frame lidar point cloud is illustrated in Figure 2.

B. UNSCENTED KALMAN FILTER (UKF) DATA FUSION

The Unscented Kalman Filter (UKF) utilizes the unscented transform to reconstruct the statistical characteristics of the state vector through Sigma points. It employs a nonlinear system model for updating the state of the robot system. In comparison to the Extended Kalman Filter (EKF), UKF is more adept at reducing computational complexity and linearization errors [19]. The fusion of the Inertial Measurement Unit (IMU) preintegration factors and the pose obtained from local lidar scan matching based on UKF results in a higher-precision pose state quantity. Notably, due to the relatively low precision of the linear accelerometer measurements in the IMU, the angular velocity increments measured by the gyroscope are more accurate. Therefore, the IMU data is directly combined with wheel odometry. The spatial state model for the mobile robot is expressed as Formula 5.

$$\begin{cases} \boldsymbol{X}_k = f(\boldsymbol{X}_{k-1}, \boldsymbol{u}_k) + \boldsymbol{W}_{k-1} \\ \boldsymbol{Z}_k = \boldsymbol{h}(\boldsymbol{X}_k) + \boldsymbol{V}_k \end{cases} \tag{5}$$

In the equation 5, W_{k-1} represents the noise sequence of the motion model at time t_{k-1}, V_k represents the measurement noise sequence at time t_k, and u_k denotes the motion transformation matrix obtained by IMU preintegration at time t_k. The process of data fusion using UKF involves several key steps, including initialization, computation of Sigma points, updating the state through the input observation equation, calculating the Kalman filter gain K_k, and updating the mean and covariance of the robot's pose.

1) The initialization of data requires setting the augmented state vector x_0^α and the augmented matrix Σ_{k-1}^α as follows:

$$x_0^\alpha = \left[x_0, 0, 0\right], \quad \sum_{k-1}^{\alpha} = \begin{vmatrix} \Sigma_{k-1} & 0 & 0 \\ 0 & \boldsymbol{Q} & 0 \\ 0 & 0 & \boldsymbol{R} \end{vmatrix} \tag{6}$$

The term x_0 represents the initial state vector of the mobile robot, Q denotes the covariance of the motion model noise, and R represents the covariance of the observation model noise, specifically the observation noise covariance of the lidar.

2) The sigma point set is sampled from the state vector x_{k-1} of the mobile robot, as shown in Formula 7. Utilizing the state vector of the mobile robot at time t_{k-1} and the preintegration results from the IMU, the state vector of the mobile robot at time t_k is obtained. Simultaneously, the mean μ_k and covariance matrix Σ_k are computed, as shown in Formula 8.

$$x_k^\alpha = \left[x_{k-1}^\alpha, x_{k-1}^\alpha + \sqrt{(n+1)\Sigma_{k-1}^\alpha}, x_{k-1}^\alpha - \sqrt{(n+1)\Sigma_{k-1}^\alpha}\right] \tag{7}$$

$$\begin{cases} x_k = f(x_{k-1}, u_k) \\ \mu_k = \sum_{i=0}^{2N} \omega_i^m x_k \\ \Sigma_k = \sum_{i=0}^{2N} \omega_i^c (x_{k,j} - \mu_k)(x_{k,j} - \mu_k)^T \end{cases} \tag{8}$$

In the equation 8, ω_i^m represents the weight factor of the IMU observation, and ω_i^c represents the weight factor of the observation obtained from laser point cloud matching.

3) Performing state updates through the input observation equation involves utilizing the observations obtained from local lidar scan matching and calculating the corresponding mean and covariance. The relevant formulas are as follows, where P_k represents the covariance of the observation, and $\Sigma_k^{1,2}$ represents the cross-covariance between prediction and observation.

$$\begin{cases} z_{k,i} = \boldsymbol{h}(\boldsymbol{x}_k) \\ z_{k|k-1} = \sum_{i=0}^{2N} \omega_i^m \boldsymbol{Z}_{k,i} \\ \boldsymbol{P}_k = \sum_{i=0}^{2N} \omega_i^c (\boldsymbol{Z}_{k,i} - z_{k|k-1})(\boldsymbol{Z}_{k,i} - z_{k|k-1})^T \\ \boldsymbol{\Sigma}_k^{1,2} = \sum_{i=0}^{2N} \omega_i^c (\boldsymbol{x}_{k,i} - \boldsymbol{\mu}_k)(\boldsymbol{Z}_{k,i} - z_{k|k-1})^T \end{cases} \tag{9}$$

4) Calculate the Kalman filter gain K_k, and update the mean μ_k and covariance Σ_k. The calculation formulas are as follows:

$$\begin{cases} K_k = \Sigma_k^{1,2} P_k^{-1} \\ \mu_k^* = \mu_k + K_k(\mathbf{z}_k - \mathbf{z}_{k|k-1}) \\ \Sigma_k^* = \Sigma_k - K_k P_k K_k^T \end{cases} \tag{10}$$

C. SCAN MATCHING

The ICP algorithm employs the point-to-point Euclidean distance between adjacent frames as the error function, leading to slow convergence and significant computational resource consumption. PL-ICP improves upon this by using the point-to-line Euclidean distance as the error function, enhancing registration accuracy while also accelerating the solution process. If we assume there are two adjacent frames of laser point cloud data to be matched, where the target point cloud sequence is represented as U={$\mathbf{u}_1, \mathbf{u}_2 \cdots \mathbf{u}_\mathrm{n}$} and the to-be-matched point cloud sequence is represented as V = {$\mathrm{v}_1, \mathrm{v}_2 \cdots \mathrm{v}_\mathrm{n}$}, then their transformation relationship satisfies:

$$\begin{cases} v^* = Rv + t \\ R = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \\ \boldsymbol{t} = (\Delta x \quad \Delta y)^T \end{cases} \tag{11}$$

where v^* represents the point cloud v transformed by a matrix, resulting in an approximate point cloud u. θ is the rotation angle of point cloud U relative to point cloud V, and Δx and Δy represent the offsets in the x-axis and y-axis directions, respectively. Based on this, the error function is constructed as follows:

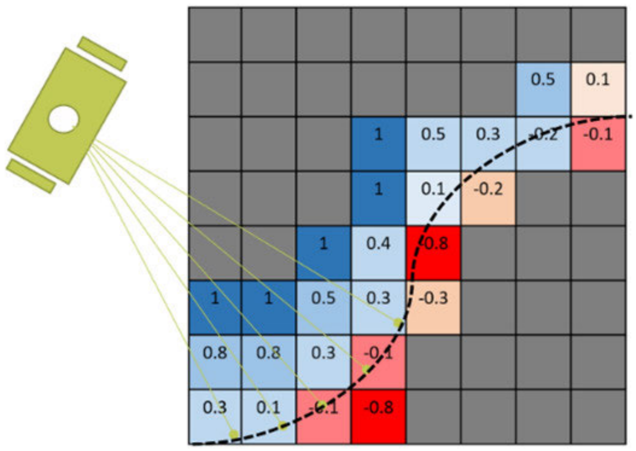


FIGURE 3. TSDF map model.

$$E(R, T) = \sum \left(n_i^T \left[R(\theta_{k+1}) u_i + t_{k+1} - v_n^i \right] \right) \tag{12}$$

In the above equation, n_i^T represents the normal vector of the nearest line, and k represents the number of iterations. During the solution process, the pre-integration results from the Inertial Measurement Unit (IMU) can be utilized as initial values for the pre-matching process. To minimize $E(R, T)$, the Levenberg-Marquardt (LM) algorithm can be employed for solving. The convergence threshold and maximum number of iterations can be set during the solution process and added as constraint factors to the back-end optimization.

D. MAP CONSTRUCTION

Traditional 2D navigation maps are often represented as occupancy grid maps, while TSDF maps provide an alternative grid-based mapping model. The occupancy grid map model is based on Bayesian theory, dividing the environment into equally sized finite grids and assigning each grid one of two states: free (Free, p = 0, no obstacle) or occupied (Occupied, p = 1, obstacle present). When a laser scan passes through the corresponding environment grid, the explored area's probability values are continuously updated based on specific formulas, thereby completing the map construction. The TSDF map model is an improvement over the Signed Distance Function (SDF). *SDF* is a signed distance field, while *TSDF* restricts its *sdf* values to be within -1 and 1 using a specific formula. The core idea of the TSDF map is to establish a truncated signed distance function map based on grids, where each grid's function value represents its distance to the obstacle surface. Figure 3 illustrates a TSDF map based on grids, with the obstacle represented as a curved line. It uses laser scan points within the same (or multiple) grids to fit the obstacle surface contour.

The definitions of the *sdf* and *tsdf* functions are as follows:

$$\begin{cases} sdf_i(x) = laser_i(x) - dist_i(x) \\ tsdf_i(x) = \max\left(-1, \min\left(1, \frac{sdf(x)}{t}\right)\right) \end{cases} \tag{13}$$

In the above equation, $laser_i$ represents the distance measured by the i-th laser beam, $dist_i$ represents the distance between the grid and the origin of the distance sensor, and the *sdf* function is applied to all grids traversed by the i-th laser. Starting from the laser origin, the *sdf* values of the grids decrease, and when the *sdf* is 0, the grid corresponding to the obstacle surface is assigned a value of 0. *tsdf* constrains the *sdf* values within the range of -1 and 1. When multiple observation lasers pass through the same grid, the update method for $tsdf(x)$ is as follows:

$$TSDF_i(x) = \frac{W_{i-1}(x)TSDF_{i-1}(x) + w_i(x)tsdf(x)}{W_{i-1}(x) + w_i(x)} \tag{14}$$

$$W_i(x) = W_{i-1}(x) + w_i(x) \tag{15}$$

In the above equation, $TSDF_i(x)$ and $TSDF_{i-1}(x)$ represent the values before and after fusion, respectively. $tsdf(x)$ represents the latest $tsdf(x)$ value for the grid $w_i(x)$ represents the weight calculated in the current iteration, and $w_{i-1}(x)$ represents the sum of weights from previous iterations. The essence of the update formula is an iterative form of a weighted least squares solution.

E. LOOP DETECTION AND BACK-END OPTIMIZATION

1) LOOP DETECTION

Loop Detection is a crucial component in Simultaneous Localization and Mapping (SLAM). By determining whether the current position of the robotic platform corresponds to a previously visited location, constraints are established by associating the pose of historical frames with the current frame. This process proves instrumental in correcting accumulated errors in trajectory during the extensive mapping process. Common loop detection algorithms include inter-frame loop detection, inter-frame, and submap loop detection, as well as inter-submap loop detection. The prevailing approach involves inter-frame and submap loop detection, which not only avoids the occurrence of erroneous closed loops between frames but also mitigates the computational burden associated with inter-submap loop detection. Cartographer, by constructing pose graph constraints between frames and sub-maps and accumulating them, performs global optimization when a sufficient number of constraints are available. This optimization effectively eliminates the cumulative errors in the occupancy grid map. Moreover, Cartographer employs a branch-and-bound method to expedite the computation of loop closure constraints. With suitable adjustments, this method is also applicable to Truncated Signed Distance Function (TSDF) maps. Due to the rotation invariance property of laser data, the process of detecting loop closure constraints between frames and sub-maps involves finding the optimal match between the laser point cloud and the TSDF values of the map. This necessitates determining the best match ξ^* within a window W around the current pose of the platform. Assuming the current laser frame is denoted as H, and its laser point position at time k is denoted as h_k, the following relationship needs to be satisfied:

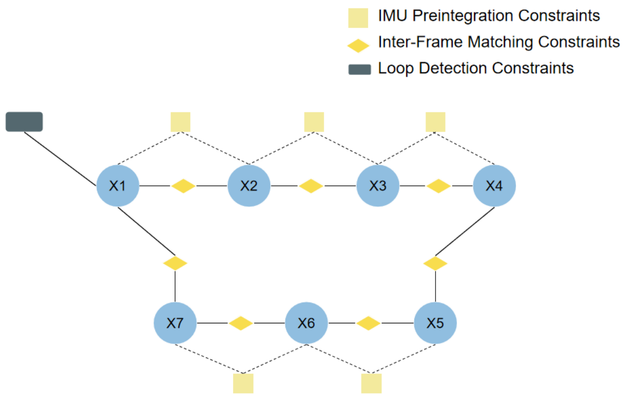

IMU Preintegration Constraints
Inter-Frame Matching Constraints
Loop Detection Constraints
X1
X2
X3
X4
X7
X6
X5

FIGURE 4. **Back-end constraint diagram.**

$$\begin{cases} \xi^* = arg \min\limits_{\xi \in W} \sum\limits_{k=1}^{K} |\phi_N(T_\xi h_k)| \\ s.t. \sum\limits_{k=1}^{K} |\phi_N(T_\xi h_k)| < e_{\max} \end{cases} \tag{16}$$

In the equation, ϕ_N represents the nearest neighbor interpolation points modeling the boundaries of obstacles in the Truncated Signed Distance Function (TSDF) map, and $e_{\max}$ is the set error threshold. If there is no matching relationship within window W that aligns the laser point positions with the submap, the incorporation of the current loop closure constraint into the back end is abandoned. To mitigate the computational burden introduced by brute-force searching for the optimal registration ξ^*, the search process employs the same branch-and-bound acceleration rules as Cartographer.

2) MULTI-CONSTRAINT BACK-END OPTIMIZATION

Back-end optimization involves utilizing constraint information generated across various system components to obtain the optimal state, adjust pose nodes, and correct the global map. The prevalent method employed today is graph optimization for pose graphs. The pose graph formed by the constraints in this paper's back-end can be abstracted as shown in Figure 4.

The nodes in the graph represent the state of the robotic platform. Assuming an initial error term for the robotic platform as e_0, nodes x_m and x_n correspond to the poses constructed for the laser data of the m-th and n-th frames in the world coordinate system, resulting in the relative transformation $\mathrm{T}_{\mathrm{m,n}}$ between the nodes. IMU preintegration provides the initial pose transformation $\mathrm{x}_{\mathrm{m,n}}$ for the laser radar. Considering the error between the observed and expected values of $\mathrm{T}_{\mathrm{m,n}}$, the formula for calculating this error term can be expressed as:

$$T_{m,n} = \begin{cases} R_m^T(\boldsymbol{p}_n - \boldsymbol{p}_i) \\ \theta_n - \theta_m \end{cases} \tag{17}$$

$$\boldsymbol{R}_m^T = \begin{bmatrix} \cos\theta_m & -\sin\theta_m \\ \sin\theta_m & \cos\theta_m \end{bmatrix} \tag{18}$$

$$e_1 = e_{mn}(x_m, x_n) = x_{m,n} - T_{m,n} \tag{19}$$

In the above equation, p_n represents the position of the robot in the world coordinate system, R_m^T is the transpose of the rotation matrix, and θ_m is the yaw angle. The information matrix for the IMU constraint factor is defined as Σ_1^{-1}, representing the weight assigned to the IMU constraint. The error term described above, based on the graph optimization model objective function, is formulated as follows:

$$F(x) = \sum \boldsymbol{e}_{mn}(\boldsymbol{x}_m, \boldsymbol{x}_n)^T \Sigma_1^{-1} \boldsymbol{e}_{mn}(\boldsymbol{x}_m, \boldsymbol{x}_n) \tag{20}$$

The ultimate goal of back-end optimization is to find the optimal pose x_1^* that minimizes the aforementioned error function. Therefore, it can be expressed as:

$$x_1^* = \arg\min \sum e_{mn}(x_m, x_n)^T \Sigma_1^{-1} e_{mn}(x_m, x_n) \tag{21}$$

Similarly, during the process of implementing laser point cloud registration and loop closure detection, inter-frame matching constraints are incorporated into the back-end optimization. The error term for these constraints is defined as e_2, with an information matrix $\sum_2^{-1}$. In the case of loop closure detection, which involves matching frames with sub-maps, the pose transformation relationship resulting from their matching is also integrated into the back-end optimization. The corresponding error term is denoted as e_3, with an information matrix $\sum_3^{-1}$. Thus, we have:

$$x_2^* = \arg\min \sum \boldsymbol{e}_2^T \Sigma_2^{-1} \boldsymbol{e}_2 \tag{22}$$

$$x_3^* = \arg\min \sum e_3{}^T \Sigma_3^{-1} e_3 \tag{23}$$

Combining all error terms, the final residual for the state of the robotic platform is represented in equation 23. This is the target value to be solved through nonlinear least squares iterations.

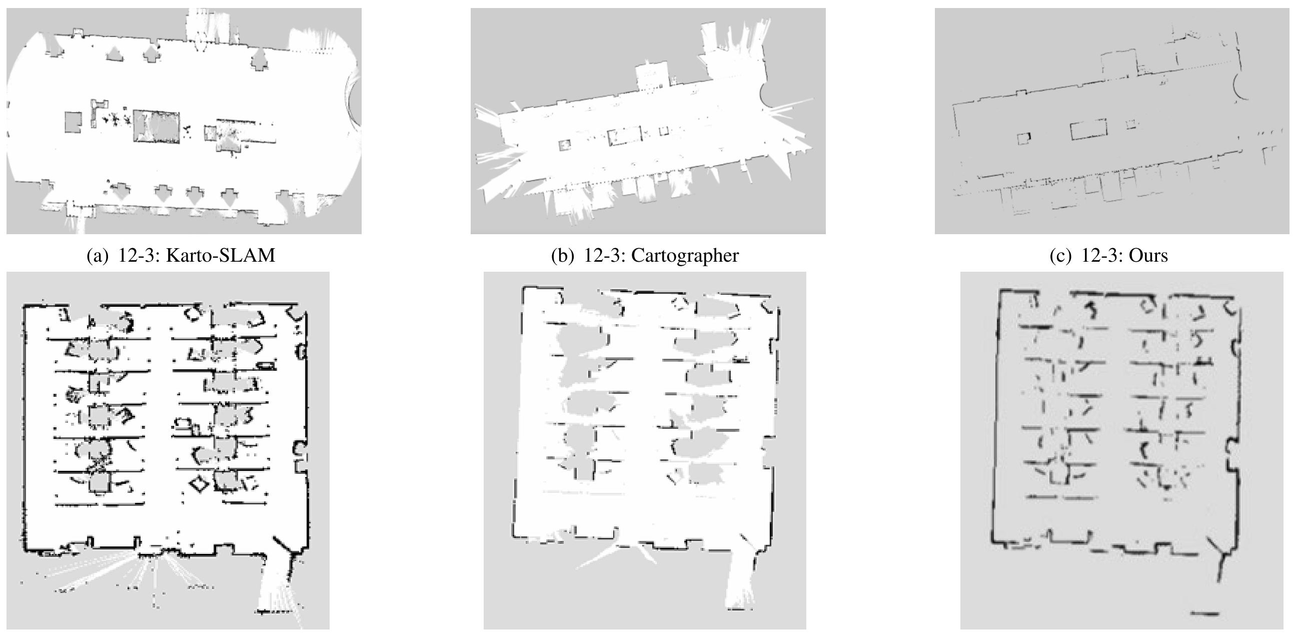
$$\mathbf{x}^* = \arg\min(\|\mathbf{e_0}\|^2 + \|\mathbf{e_1}\|^2 + \|\mathbf{e_2}\|^2 + \|\mathbf{e_3}\|^2) \tag{24}$$

IV. ANALYSIS OF EXPERIMENTAL RESULTS

To assess the algorithm's performance, comparative experiments were conducted using open-source datasets and datasets recorded in real-world environments, focusing on mapping quality and localization accuracy. The system ran on the Ubuntu 20.04 and ROS (Robot Operating System) platform, playing back data in the form of offline rosbags. The recorded datasets '11-7' and '11-9' in a real environment were collected from the teaching building corridor at the School of Software, South China Normal University. The robotic platform used was Turtlebot2, equipped with Slamtec's RPLidar A3, an inertial measurement unit (IMU), and wheel encoders. The hardware configuration included an AMD R5-5600H processor with a clock frequency of 3.30 GHz and 16 GB of RAM.The comparison algorithms selected are Karto-SLAM and Cartographer. In terms of multi-sensor data fusion, Karto-SLAM does not perform multi-sensor data fusion, whereas Cartographer, although it fuses IMU and other sensor data in the front-end to improve localization accuracy, does not utilize IMU and other sensor data in the back-end. In contrast, this paper uses the UKF in the front-end to fuse odometry, IMU data, and LiDAR scan matching results to obtain the pose information of the robot. In the back-end, IMU pre-integration factors, loop closure detection, and inter-frame matching of LiDAR point clouds enhance the constraints for global map pose optimization, achieving deep fusion of multi-sensor data. In terms of mapping, both Karto-SLAM and Cartographer are algorithms based on occupancy grid maps, which are susceptible to Gaussian noise and other influences. However, the TSDF map selected by the algorithm in this paper can effectively avoid map ghosting and the influence of Gaussian noise. Therefore, these two algorithms are chosen as comparison algorithms in this paper to experimentally validate the feasibility of the proposed algorithm.

A. MAPPING EFFECT ANALYSIS

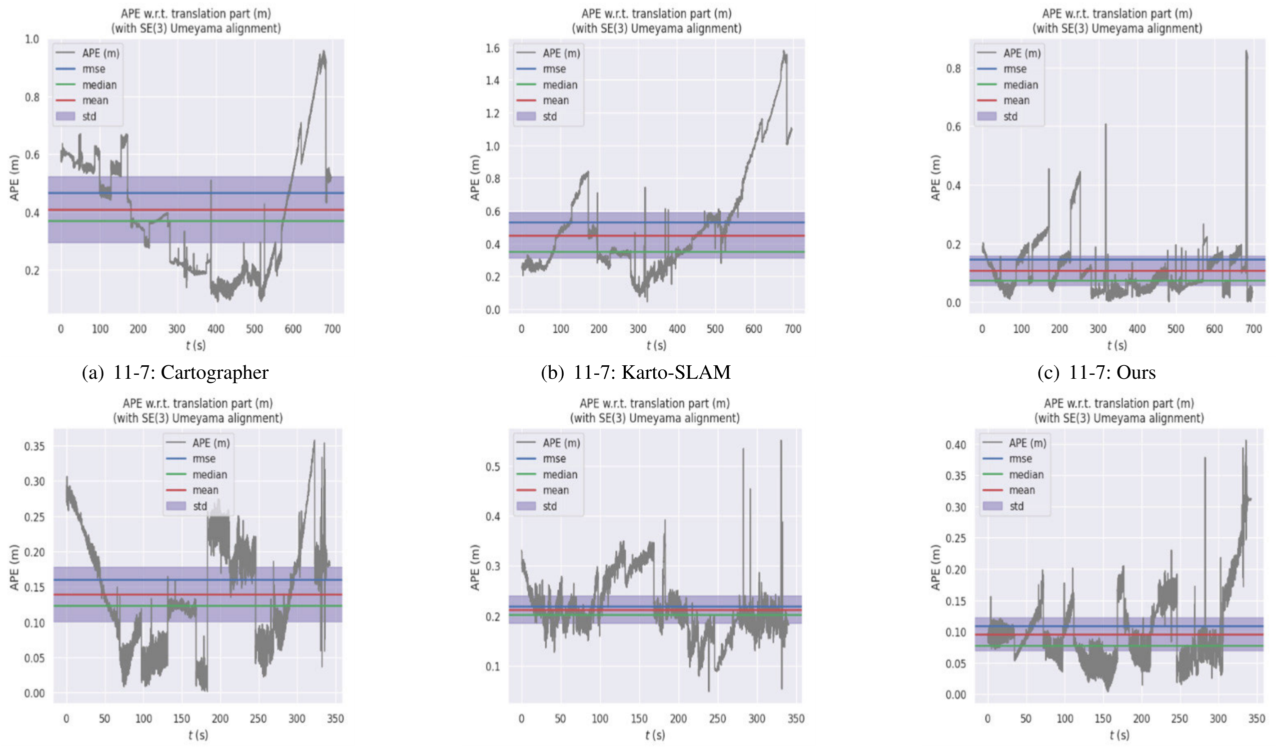
In terms of mapping effectiveness, experiments were conducted using publicly available datasets, specifically sequences 12-3 and 12-11. The tests involved running Karto-SLAM, Cartographer, and the algorithm proposed in this paper. The mapping results obtained for the respective algorithms are illustrated in Figure 5. Dataset sequences 12-3 represent an outdoor environment with a large scope, while dataset sequences 12-11 represent a common indoor scenario with complex obstacles. From the mapping results, it can be observed that, when processing the same dataset sequence, Karto-SLAM is heavily affected by Gaussian noise, leading to significant artifacts at the edges of obstacles. Cartographer, while capable of producing good mapping results, still show some influence from Gaussian noise. In our method, we utilize a TSDF map where, during the update of TSDF values, a weighted averaging method is employed to fuse multiple observation values. This smoothing technique helps mitigate the influence of noise, resulting in more accurate and stable TSDF values. Additionally, with each update, corresponding weights are adjusted. The increase in weights enhances the reliability of results from multiple observations, thereby reducing the impact of single Gaussian noise on the final TSDF values. Moreover, within the TSDF, only distance values within a certain range are considered, and values beyond this range are truncated. This reduces the influence of noise from distant sources and saves some amount of memory consumption. As shown in Figure 5, in the complex indoor environment of dataset sequences 12-11, the proposed method successfully avoids significant artifacts at the edges of obstacles and addresses the Gaussian noise problem effectively.



(a) 12-3: Karto-SLAM (b) 12-3: Cartographer (c) 12-3: Ours

(d) 12-11: Karto-SLAM (e) 12-11: Cartographer (f) 12-11: Ours

FIGURE 5. Comparison of algorithm mapping effects.



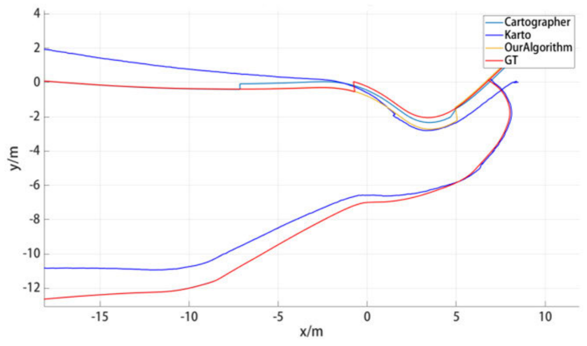
(a) 11-7: Cartographer (b) 11-7: Karto-SLAM (c) 11-7: Ours

(d) 11-9: Cartographer (e) 11-9: Karto-SLAM (f) 11-9: Ours

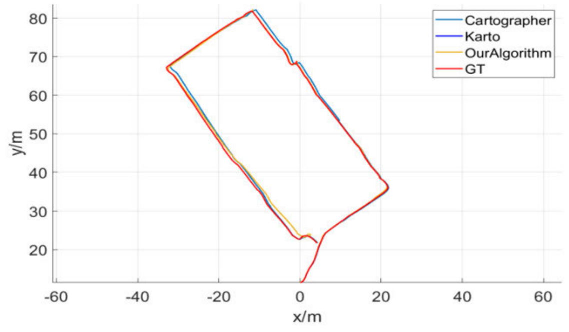
FIGURE 6. Comparison of absolute pose error curves.

B. MOTION TRAJECTORY AND LOCALIZATION ACCURACY ANALYSIS

Due to the availability of ground truth data, the collected datasets 11-7, 11-9, and the publicly available dataset 12-3 were used for localization accuracy analysis. Figure 7 illustrates the motion trajectories and ground truth comparisons of Karto-SLAM, Cartographer, and the algorithm proposed in this paper for datasets 12-3 and 11-7. Further calculations resulted in absolute pose error (APE) curves and relative pose error (RPE) for the three algorithms in



(a) 12–3 trajectory comparison



(b) 11-7 trajectory comparison

FIGURE 7. Comparison of motion trajectories.

TABLE 2. Comparison of RPE data.

Algorithm	Datasets	RMSE	Mean	S.D.	Max
Ours	11-7	**0.0115**	0.00401	**0.01077**	**0.85746**
	11-9	**0.01157**	0.00431	**0.01074**	**0.28643**
	12-3	**0.01633**	**0.0072**	**0.01467**	0.61281
Cartographer	11-7	0.0120	**0.00363**	0.0114	0.904713
	11-9	0.0117	**0.00430**	0.0109	0.29916
	12-3	0.01937	0.00937	0.01695	0.75732
Karto-SLAM	11-7	0.01680	0.00922	0.01405	0.8680
	11-9	0.01787	0.01264	0.01263	0.33378
	12-3	0.02485	0.01941	0.01552	**0.60946**

comparison with the ground truth trajectory, as shown in Figure 6 and Table 2. RPE is effective in evaluating trajectory drift within fixed time intervals, while APE calculates the difference between the ground truth trajectory and the localization results, providing an intuitive representation of algorithm performance.

From the APE error curves in Figure 6 and the RPE evaluation metrics in Table 2, it can be observed that Karto-SLAM exhibits the lowest localization accuracy, particularly in outdoor scenarios where the error values are more pronounced, leading to significant trajectory drift and a lack of robust loop closure constraints. Cartographer, which integrates IMU data through a pose extrapolator in the front end, performs better in localization, leveraging prior information from the IMU sensor. The algorithm proposed in this paper incorporates depth fusion of sensor information and integrates IMU preintegration constraints into the back-end optimization. From the relative pose error and absolute pose error shown in Figure 6 and Table 2, the algorithm demonstrates the best localization performance, with a more accurate fitting of the output motion trajectory to the ground truth trajectory.

V. CONCLUSION

This paper analyzes the deficiencies of commonly used 2D SLAM algorithms primarily based on laser data in complex indoor environments. Considering aspects such as point cloud registration, representation of map models, incomplete utilization of sensor data, and back-end optimization, a multi-sensor depth fusion SLAM method based on TSDF maps is proposed. The method is evaluated and tested using data collected from both open-source and real-world environments. Experimental results demonstrate significant advantages in mapping quality and localization accuracy. The proposed approach leverages IMU preintegration-derived poses for pre-matching in laser point cloud distortion correction and registration. This improves the quality of the system's input data and provides a good initial value for laser point cloud registration, accelerating the matching process. In the back-end optimization process, the utilization of multiple constraints effectively reduces localization errors. Constrained by research conditions and time limitations, the depth and breadth of this study await further enhancement. While establishing 2D TSDF maps in outdoor and high real-time scenarios can avoid Gaussian noise from traditional occupancy grid maps and save some memory consumption, the algorithm's complexity in updating obstacle surfaces requires a certain amount of computational resources, posing a disadvantage in terms of real-time performance. Future work aims to draw insights from other researchers in the field and considers improving the strategy for TSDF map updates to mitigate the substantial consumption of computational resources during the mapping process.

REFERENCES

[1] C. Qian, H. Zhang, J. Tang, B. Li, and H. Liu, "An orthogonal weighted occupancy likelihood map with IMU-aided laser scan matching for 2D indoor mapping," *Sensors*, vol. 19, no. 7, p. 1742, Apr. 2019.

[2] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, "SplaTAM: Splat, track & map 3D Gaussians for dense RGB-D SLAM," 2023, *arXiv:2312.02126*.

[3] C. Yan, D. Qu, D. Xu, B. Zhao, Z. Wang, D. Wang, and X. Li, "GS-SLAM: Dense visual SLAM with 3D Gaussian splatting," 2023, *arXiv:2311.11700*.

[4] J. Wu and X. Song, "Review on development of simultaneous localization and mapping technology," *J. Shandong Univ. Sci.*, vol. 51, pp. 16–31, Jan. 2021.

[5] M. B. Alatise and G. P. Hancke, "A review on challenges of autonomous mobile robot and sensor fusion methods," *IEEE Access*, vol. 8, pp. 39830–39846, 2020.

[6] W. Xuanbin, L. Xingxing, L. Jianchi, F. Shaoquan, L. Shengyu, and Z. Yuxuan, "Tightly-coupled stereo visual-inertial-LiDAR SLAM based on graph optimization," *Acta Geodaetica et Cartographica Sinica*, vol. 51, no. 8, p. 1744, 2022.

[7] X. Xu, L. Zhang, J. Yang, C. Cao, W. Wang, Y. Ran, Z. Tan, and M. Luo, "A review of multi-sensor fusion SLAM systems based on 3D LiDAR," *Remote Sens.*, vol. 14, no. 12, p. 2835, Jun. 2022.

[8] Y. Jia, H. Luo, F. Zhao, G. Jiang, Y. Li, J. Yan, Z. Jiang, and Z. Wang, "Lvio-fusion: A self-adaptive multi-sensor fusion SLAM framework using actor-critic method," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 286–293.

[9] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," *Proc. SPIE*, vol. 1611, pp. 586–606, Apr. 1992.

[10] A. Censi, "An ICP variant using a point-to-line metric," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2008, pp. 19–25.

[11] W. Wei, B. Shirinzadeh, S. Esakkiappan, M. Ghafarian, and A. Al-Jodah, "Orientation correction for Hector SLAM at starting stage," in *Proc. 7th Int. Conf. Robot Intell. Technol. Appl. (RiTA)*, Nov. 2019, pp. 125–129.

[12] F. Jiang, J. Chen, and S. Ji, "Panoramic visual-inertial SLAM tightly coupled with a wheel encoder," *ISPRS J. Photogramm. Remote Sens.*, vol. 182, pp. 96–111, Dec. 2021.

[13] J. Cheng, Y. Jin, Z. Zhai, X. Liu, and K. Zhou, "Research on positioning method in underground complex environments based on fusion of binocular vision and IMU," *Sensors*, vol. 22, no. 15, p. 5711, Jul. 2022.

[14] K. Xiao, W. Yu, W. Liu, F. Qu, and Z. Ma, "High-precision SLAM based on the tight coupling of dual LiDAR inertial odometry for multi-scene applications," *Appl. Sci.*, vol. 12, no. 3, p. 939, Jan. 2022.

[15] G. Schouten and J. Steckel, "RadarSLAM: Biomimetic SLAM using ultra-wideband pulse-echo radar," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2017, pp. 1–8.

[16] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LiDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 1271–1278.

[17] Z. Yajun, W. Jun, W. Shaoming, S. Jinping, and L. Peng, "Review of the method for distributed multi-sensor multi-target tracking," *J. Radar*, vol. 12, no. 1, pp. 1–17, 2022.

[18] M. R. U. Saputra, C. X. Lu, P. P. B. de Gusmao, B. Wang, A. Markham, and N. Trigoni, "Graph-based thermal–inertial SLAM with probabilistic neural networks," *IEEE Trans. Robot.*, vol. 38, no. 3, pp. 1875–1893, Jun. 2022.

[19] J. Zhang, L. Yuan, T. Ran, Q. Tao, and L. He, "Bayesian nonparametric object association for semantic SLAM," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5493–5500, Jul. 2021.

[20] Y. Zhang, F. Tosi, S. Mattoccia, and M. Poggi, "GO-SLAM: Global optimization for consistent 3D instant reconstruction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 3727–3737.

[21] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.

[22] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proc. IEEE Adapt. Syst. Signal Process., Commun., Control Symp.*, Oct. 2000, pp. 153–158.

[23] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot.*, Nov. 2011, pp. 155–160.

[24] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, Feb. 2007.

[25] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient sparse pose adjustment for 2D mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 22–29.

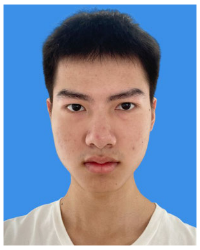
[26] J.-D. Fossel, K. Tuyls, and J. Sturm, "2D-SDF-SLAM: A signed distance function based SLAM frontend for laser scanners," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 1949–1955.

[27] K. Daun, S. Kohlbrecher, J. Sturm, and O. von Stryk, "Large scale 2D laser SLAM using truncated signed distance functions," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot. (SSRR)*, Sep. 2019, pp. 222–228.

[28] X. Fu, Z. Fang, X. Xiao, Y. He, and X. Liu, "Improved signed distance function for 2D real-time SLAM and accurate localization," 2021, *arXiv:2101.08018*.

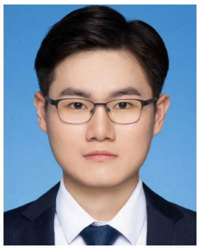
YIBO CAO was born in Hengyang, Hunan, China, in 1971. He received the B.S. and M.S. degrees from the Department of Electrical and Mechanical Engineering, Xi'an University of Electronic Science and Technology, in 1994 and 1997, respectively, the Ph.D. degree from the School of Mechanical Engineering, South China University of Technology, Guangzhou, China, in 2007, and the Ph.D. degree from the Department of Precision Instruments, Tsinghua University, Beijing, China, in 2009.

He joined the Software College, South China Normal University, in 2009, where he is currently an Associate Professor. His research interests include AI artificial intelligence technology research, optical electromechanical integration technology, the Internet of Things technology, pattern recognition, and simultaneous localization and mapping (SLAM).



ZHENYU DENG was born in Shaoyang, Hunan, China, in 2001. He is currently pursuing the degree with the School of Software, South China Normal University. His current research interests include computer vision, vision simultaneous localization, path tracking, mobile robots, and autonomous robot navigation.

ZEHAO LUO was born in Puning, Guangdong, China, in 2000. He is currently pursuing the degree with the School of Software, South China Normal University. His current research interests include computer vision, vision simultaneous localization, and mapping and indoor positioning



JINGWEN FAN was born in Ganzhou, Jiangxi, China, in 1998. He is currently pursuing the degree with the School of Software, South China Normal University. His current research interests include computer vision, vision simultaneous localization, mapping, and indoor positioning.