Research on Sensor Fusion-Based Calibration and Real-Time Point Cloud Mapping Methods for Laser LiDAR and IMU

He Huang [1], Xin Zhang [1], Junxing Yang [1]*, Yao Fu [1], Junyang Bian [1]

[1] School of Geomatics and Urban Spatial Informatics, Beijing University of Civil Engineering and Architecture, Beijing, China - huanghe@bucea.edu.cn, zx2806404866@163.com, yangjunxing@bucea.edu.cn, 18801132264@163.com, bjy1066702005@163.com

KEY WORDS: Inertial Measurement Unit, LiDAR, Joint Calibration, Multi-sensor Fusion, Simultaneous Localization and Mapping.

ABSTRACT:

Simultaneous Localization and Mapping (SLAM), as one of the core technologies in autonomous driving, provides environment perception, decision-making, and path planning to ensure safe vehicle operation. This paper proposes a tightly-coupled fusion mapping method based on LiDAR and Inertial Measurement Unit (IMU). By introducing IMU, performing extrinsic calibration, and time synchronization with the LiDAR, the issue of mapping drift caused by inconsistent coordinate systems is addressed, leading to improved mapping accuracy. Furthermore, to compensate for IMU zero-bias errors, IMU preintegration is employed to calibrate the point cloud and optimize the initial values of the LiDAR odometry. A novel iVox voxel-based point cloud registration method is used to enhance registration efficiency without compromising mapping accuracy. Lastly, the Iterative Extended Kalman Filter (IEKF) is incorporated into the back-end state estimation to propagate and correct the estimated state using IMU data, enabling precise state estimation and map updates. This research is of significant importance in addressing the challenges of low mapping accuracy and real-time performance in single-sensor SLAM, providing an effective solution for environment perception and path planning in autonomous driving systems.

1. INTRODUCTION

Multi-sensor fusion SLAM offers several advantages: mutual verification and complementarity, improved localization and mapping accuracy, and robustness. It expands the applicability range by overcoming the limitations of single sensors in specific situations and reduces time and computational costs, enhancing real-time performance and efficiency.

In recent years, various multi-sensor fusion SLAM algorithms have emerged. VINS-Mono is a framework based on visual-inertial SLAM that achieves tight coupling by fusing pre-integrated IMU data and camera feature data. LIO-SAM is a lidar-inertial SLAM framework that utilizes keyframe-based extraction of plane and edge features and reduces drift through GPS and loop closure constraints. LVI-SAM combines the ideas of VINS and LIO-SAM algorithms, utilizing lidar data for initialization and loop closure detection to improve robustness. LIMO is a laser-assisted visual SLAM method that can obtain depth information for feature points, improving feature point matching accuracy. LIC-FUSION proposes advanced methods for outlier removal and fusion of lidar plane features, but it has lower mapping efficiency. TVL-SLAM is a tightly coupled visual-lidar SLAM algorithm that avoids degeneracy situations through cross-checking. R2LIVE and R3LIVE are SLAM frameworks based on lidar-inertial and visual-inertial odometry, capable of building global map geometry and fusing visual data.

In conclusion, the multi-sensor fusion SLAM mapping method has gained increasing attention in recent years. Due to the complementary advantages of IMU and LiDAR sensors, SLAM methods based on the fusion of IMU and LiDAR have been widely applied and developed rapidly. Among the majority of IMU and LiDAR fusion SLAM systems, the joint calibration of IMU and LiDAR is a prerequisite for mapping. In the mapping process, the front-end point cloud registration utilizes a tree-based K-NN nearest neighbour search method, which leads to a longer computational time. Additionally, the back end adopts Kalman filter state estimation, which results in lower mapping accuracy and efficiency. To address these issues, this paper proposes improvements to the corresponding algorithms in both the front-end and back-end of the mapping process. Experiments were designed and conducted at the experimental site, and validation experiments were performed in real-world scenarios. Finally, the efficiency and accuracy of the experiments are analyzed.

* Junxing Yang

2. METHOD

2.1 Joint Calibration

In this experiment, three right-handed Cartesian coordinate systems are involved: the World Coordinate System (W), the LiDAR Coordinate System (L), and the IMU Coordinate System (I). These coordinate systems can be transformed into each other.

Conversion of IMU and LiDAR Coordinate Axes: The point cloud data obtained from the LiDAR scanner is in the LiDAR's coordinate system. When the robot collects point cloud data at different time instances during motion, the data needs to be transformed into the world coordinate system for estimating state transformations. The transformation relationship between the LiDAR coordinate system and the world coordinate system is represented by the matrix R. Please note that the provided translation may need further adjustment based on the context of the surrounding text.

$$\mathbf{T}_L^W = (tx_L^W, ty_L^W, tz_L^W, \alpha x_L^W, \beta y_L^W, \phi z_L^W) = (p_L^W, q_L^W) \quad (1)$$

The obtained transformation matrix $\mathbf{T}_L^W$ between the LiDAR coordinate system and the world coordinate system is given by equation (2):

$$X^W = RX^L + q_L^W \quad (2)$$

Where x and y are the coordinates of the point cloud data in the world coordinate system and the LiDAR coordinate system, respectively.

In the IMU coordinate system, the positive North direction, positive East direction, and positive Up direction are defined according to the right-hand orthogonal rule. The transformation between the IMU coordinate system and the LiDAR coordinate system is performed by solving the transformation matrix mapping between them. During system operation, this transformation matrix is used to map the current state of the point cloud data.

In the IMU coordinate system, the OX -axis points towards the positive North, the OZ -axis points towards the positive East, and the OY -axis is determined by the right-hand orthogonal rule concerning the OZ axes. The transformation between the IMU coordinate system and the LiDAR coordinate system is performed by solving the transformation matrix mapping between them. During system operation, this transformation matrix is used to map the current state of the point cloud data.

Let the transformation matrix between the two coordinate systems be denoted as A. If the IMU rotates around the X , Y and Z axes by angles γ , θ and φ respectively such that the two coordinate systems align, then A can be solved using the following equation (3):

$$A = \begin{bmatrix} \cos\theta\cos\varphi & \sin\theta & -\cos\theta\sin\varphi \\ -\cos\gamma\cos\varphi\sin\theta & \cos\gamma\cos\theta & \cos\gamma\sin\varphi\sin\theta + \sin\gamma\cos\varphi \\ \sin\gamma\cos\varphi\sin\theta + \cos\gamma\sin\varphi & -\sin\gamma\cos\theta & -\sin\gamma\sin\varphi\sin\theta + \cos\gamma\cos\varphi \end{bmatrix} \quad (3)$$

The matrix A is an orthogonal matrix, which means its transpose can represent the transformation matrix from the reference frame to the IMU frame.

2.1.1 IMU Error Model and Preintegration: IMU sensors have inherent accuracy errors, such as accelerometer bias, gyroscope temperature drift, mechanical vibrations, and susceptibility of magnetometers to external magnetic field interference. The installation environment, motion dynamics, and algorithmic processing can also introduce disturbances that affect the accuracy of IMU output. The errors of an IMU can be classified into systematic errors and random errors. Systematic errors include bias errors, scale factor errors, misalignment and non-orthogonality errors, nonlinearity errors, and temperature errors. Random errors include random walk errors and bias instability. In a multi-sensor fusion system, IMU data is collected at high frequency with inherent errors, and it is used to integrate the attitude and velocity information, which is then fused with lidar data for pose estimation. However, IMU integration introduces integration errors and requires significant computational resources. To address this issue, IMU preintegration techniques can be employed. This involves storing the IMU velocity and displacement increments as constants over small time intervals and incorporating the preintegration results as constraints in the optimization problem. This reduces the computational burden of reintegration and improves the efficiency of the optimization process. Therefore, IMU preintegration is an effective approach that enhances the accuracy and efficiency of pose estimation in laser SLAM.

2.1.2 Timestamp Synchronization: Due to the different operating frequencies of various sensors, in this experiment, the IMU has a sampling frequency of 1000Hz, while the LiDAR has a sampling frequency of 10Hz. To ensure synchronization, a common host provides a reference timestamp. Each sensor adds timestamp information to its independently collected data based on its calibrated timestamp. There are two methods for timestamp synchronization: hard synchronization and soft synchronization.

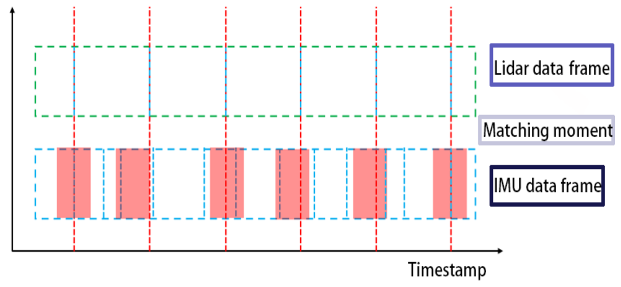

Figure 1. Principle of timestamp generation for LiDAR and IMU.

Figure 1 illustrates an example of time synchronization between LiDAR and IMU data. The time sequence records the sampling moments of each sensor, where the horizontal axis represents timestamps. The green dashed box represents the LiDAR sampling frequency, while the blue dashed box represents the IMU sampling frequency. When the LiDAR completes a sample, it searches for the nearest IMU data point in time to establish time matching between the two data sources, as indicated by the red box in the figure. Through time synchronization, data fusion and analysis from different sensors can be achieved, enhancing the accuracy of perception and control.

However, this method of time synchronization may encounter some challenges. As mentioned earlier, the nearest neighbour moment mentioned in red text is often ambiguous, and the time difference between the nearest neighbour moment and the LiDAR sampling time can be significant, leading to suboptimal time synchronization results. Additionally, as the number of sensors requiring time synchronization increases, the difficulty of matching and the precision of synchronization decrease.

Compared to soft synchronization, hard synchronization demonstrates better synchronization performance and higher matching accuracy.

2.2 IMU-LiDAR Tight-Coupling Mapping Method

In the vast majority of multi-sensor fusion systems, a tree-based structure is employed for nearest-neighbour matching. This paper introduces an improvement to the iterative closest point (ICP) algorithm, which mainly focuses on voxel-based nearest neighbour matching. This approach eliminates the need for constructing, iterating, balancing, and removing nodes in the tree-based structure, thus preserving the efficiency of the Laser-Inertial Odometry (LIO) system. For the backend state estimation, the proposed method utilizes forward computation using IMU data and backward computation using LiDAR data to calculate residuals. Iterative extended Kalman filtering is applied for data iteration and refinement. Figure 2 illustrates the mapping process flowchart for this chapter.

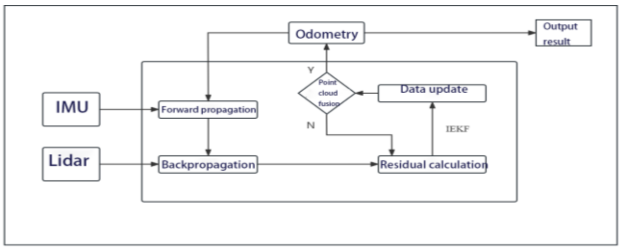

Figure 2.Technical approach.

2.2.1 Sparse Voxel-Based Nearest Neighbor Search: In the fusion system of LiDAR and IMU, strict K-nearest neighbour (KNN) search and range search are not required, especially considering the rough position estimation obtained from IMU measurements in the LIO system. The main advantage of the k-d tree is its ability to provide strict K-NN and range search results by conditionally partitioning high-dimensional space using hyperplanes. However, in some cases, the search algorithm may traverse very distant branches of the tree in search of potential nearest neighbours, which is unlikely to be useful for local plane coefficient estimation. On the other hand, voxel-based algorithms limit the search range to a predefined value, so discarding such neighbours does not affect the majority of residuals. Additionally, the construction, iteration, balancing, and removal of k-d tree nodes impact the efficiency of LIO, whereas these issues do not exist in voxel-based approaches. In the voxel-based approach, a conservative insertion and passive deletion strategy is used, and it is not mandatory to update.

Based on the aforementioned issues, this paper proposes an incremental voxel structure (iVox) as a point cloud spatial data structure in SLAM, which supports incremental insertion and parallel approximate K-NN queries. In the fusion SLAM system of LiDAR and IMU, K-NN is commonly used for point cloud registration in the front-end LIO module.

In iVox, the K-NN search is limited to a predefined range and divided into three steps. Given an iVox structure V and a query point P1, find the voxel index and neighbouring voxels, represented as S. Iterate through each voxel in S and search for up to K nearest points in each voxel. Merge the search results and select the best K nearest points. Since the algorithm has already been parallelized at the point cloud level, there is no need to perform parallel searches for each voxel here. The K-NN search in iVox is simple and efficient, although it is not as strict as tree-based algorithms. However, it is sufficient for LIO applications.

2.3 Fusion Mapping Method Based on Iterative Extended Kalman Filtering

This paper proposes an iterative extended Kalman filter-based method for LIDAR-IMU fusion mapping. The approach includes a preprocessing step that selects feature points from the raw LIDAR data for subsequent state estimation processes. Preintegration is utilized to calibrate the point cloud and provide initial values for optimizing the LIDAR odometry.

2.3.1 Iterative State Update: The propagated state $\hat{x}_k$ and covariance $\hat{p}_k$ impose a prior Gaussian distribution on the unknown state x_k, where $\hat{p}_k$ represents the covariance of the error state given by the equation below:

$$X_k \ominus \hat{X}_k = \left(\hat{X}_k^{\kappa} \oplus \tilde{X}_k^{\kappa}\right) \ominus \hat{X}_k = \hat{X}_k^{\kappa} \ominus \hat{X}_k + J^{\kappa} \tilde{X}_k^{\kappa} \quad (4)$$
$$: \; \mathrm{N}\left(0, \hat{P}_k\right)$$

where J^{κ} is the partial derivative of concerning evaluated at zero, and the specific formula for J^{κ} is given by equation (5):

$$J^{\kappa} = \begin{bmatrix} A\left(\delta^G \theta_{I_k}\right)^{-T} & 0_{3\times15} & 0_{3\times3} & 0_{3\times3} \\ 0_{15\times3} & I_{15\times15} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times15} & A\left(\delta^I \theta_{L_k}\right)^{-T} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times15} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \quad (5)$$

$\delta^G \theta_{I_k}$ and $\delta^I \theta_{L_k}$ represent the error states of the IMU's pose and external rotation matrix, respectively. For the first iteration of the Kalman filter, $\hat{X}_k^{\kappa} = \hat{X}_k$ and $J^{\kappa} = I$. In addition to the prior distribution, the state distribution of the measurement model is as follows:

$$-V_j = Z_j^{\kappa} + H_j^{\kappa} X_k^{\kappa} : \; \mathrm{N}\left(0, R_j\right) \quad (6)$$

The posterior distribution of the state X_k is obtained by combining the prior distribution from Equation (4) with the measurement model from Equation (5). The maximum a posteriori estimate (MAP) $\tilde{X}_k^{\kappa}$ can also be expressed as follows:

$$\min_{\tilde{X}_k^{\kappa}} \left(\left\| X_k \ominus \hat{X}_k \right\|_{\hat{P}_k}^2 + \sum_{j-1}^{m} \left\| Z_j^{\kappa} \right\|_{R_j}^2 \right) \quad (7)$$

In this case, where $\left\| X \right\|_M^2 = X^T M^{-1} X$, the maximum a posteriori (MAP) estimation problem can be solved using iterative Kalman filtering.

$$K = \left(H^T R^{-1} H + P^{-1}\right)^{-1} H^T R^{-1} \quad (8)$$

$$\hat{X}_k^{\kappa+1} = \hat{X}_k^{\kappa} \oplus \left(-\kappa Z_k^{\kappa} - \left(I - \kappa H\right)\left(J^{\kappa}\right)^{-1}\left(\hat{X}_k^{\kappa} \ominus \hat{X}_k\right)\right) \quad (9)$$

In equation (9), for ease of computation, we simplify some parts of the formulas as $H = \left[H_1^{\kappa T}, \ldots H_m^{\kappa T}\right]^T$, $R = diag\left(R_1, \ldots R_m\right)$, $P = \left(J^{\kappa}\right)^{-1} \hat{P}_k \left(J^{\kappa}\right)^{-T}$, $Z_k^{\kappa} = \left[Z_1^{\kappa T}, \ldots Z_m^{\kappa T}\right]^T$. Kalman filtering is typically computed through $K = PH^T\left(HPH^T + R\right)^{-1}$, which requires transposing matrices of state dimensions. We repeat the above process until the iteration reaches the target threshold and then stop, completing the iteration as shown in equation (10):

$$\left\| \hat{X}_k^{\kappa+1} \ominus \hat{X}_k \right\| < \varepsilon \quad (10)$$

By updating the state X, the point cloud collected by the k-th scan of the LiDAR sensor is transformed into the global map, completing the iterative update of the local map into the global map. Figure 3 illustrates the process of point cloud local map state update.

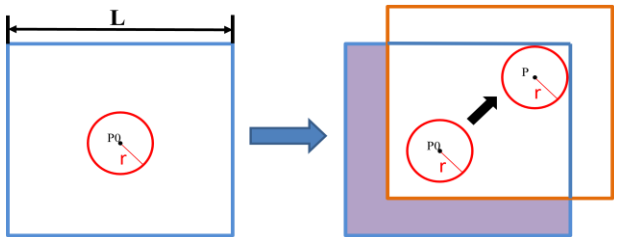

Figure 3. Updating the state of the point cloud local map.

```
State estimation based on IEKF：
Input: Previous output of $\bar{X}_{k-1}, \bar{P}_{k-1}$；
    Raw LiDAR point cloud from the current scan；
    IMU measurements ( $a_m, w_m$ ) from the current scan；
1、Compute the state estimate $\hat{X}_k$ and covariance value $\hat{P}_k$ using the forward calculation equations  and；
2、Perform motion compensation using the backward calculation equations；
3、$\kappa = -1, \hat{X}_k^{\kappa=0} = \hat{X}_k$
4、Iterate and repeat the process
    $\kappa = \kappa + 1$；
    According to equation (5) and $P = \left(J^{\kappa}\right)^{-1} \hat{P}_k \left(J^{\kappa}\right)^{-T}$ , calculate $J^{\kappa}$;
    Calculate the residual $Z_j^{\kappa}$ and covariance $H_j^{\kappa}$ based on equations  and；
    Calculate the state update $\hat{X}_k^{\kappa+1}$ based on equation (9)；
5、$\left\| \hat{X}_k^{\kappa+1} \ominus \hat{X}_k \right\| < \varepsilon$
6、$\bar{X}_k = \hat{X}_k^{\kappa+1}$，$\bar{P}_k = (I - KH)P$
Output: Current best estimate $\bar{X}_k$ and $\bar{P}_k$ .
```

Table 1. The process of state estimation algorithm based on IEKF is as follows.

After converting the transformed LiDAR point cloud data into a local map, the local map is then updated into the global map. This iterative process, using the Iterative Kalman Filter, continuously optimizes the error state of the tightly-coupled fusion system between LiDAR and IMU, thereby improving the robustness and accuracy of the SLAM system.

3. EXPERIMENTAL RESULTS AND ANALYSIS

3.1 Experimental Platform and Objects

This chapter primarily focuses on the experimental validation of the proposed algorithm and provides comparative experiments to demonstrate the feasibility and robustness of the improvements made in this study. To validate the LIDAR and IMU fusion SLAM mapping algorithm proposed in this paper, data collection was conducted within the experimental site. Furthermore, the algorithm was tested in real-world scenarios at the site to evaluate its performance in practical field conditions.

3.1.1 Experimental Platform Introduction： To validate the feasibility of the algorithm proposed in this paper, a self-built robot platform was used in the laboratory. The platform consists of a robot base, LiDAR, camera, inertial measurement unit (IMU), and GPS, as shown in Figure 4. The detailed specifications of the main hardware facilities used in this experiment are listed in Table 2.

Sensor Types	Detailed Introduction	Working Environment
Songling Robot HUNTER	It has similar features to a car, supports integration with the ROS system, and has the advantages of long battery life and minimal wear. It has a maximum payload of 200kg. It is an Ackermann front-wheel steering line-controlled chassis, which allows for high-speed operation under this configuration. It has a minimum braking distance of 0.2m and is suitable for autonomous driving development.	It can work around the clock, provided that there is sufficient battery power
Hesa Pandar 40	The detection distance is 200m with a 20% reflectivity. It has a vertical field of view angle of 23°, providing a viewing angle range from -16° to +7°. The minimum vertical resolution is 0.33°, and the minimum horizontal angular resolution is 0.2°.	It is recommended to operate in a relatively stable environment with good lighting, strong target surface reflection, and suitable temperature and humidity conditions. It is not suitable for operation in rainy or snowy weather.
OEM7 six-axis IMU	It consists of three accelerometers and three gyroscopes. It can measure the acceleration and angular velocity of an object.	It needs to work in a relatively stable environment with a good GPS signal, moderate temperature, weak magnetic field and electromagnetic interference, and minimal vibration to ensure measurement accuracy and reliability.

Table 2. Introduction of experimental platform hardware.

Figure 4. Experimental equipment.

3.1.2 Software Platform Introduction: The software platform used in this study was Ubuntu 18.04. The system was compiled under the Robot Operating System (ROS), which facilitated the display of node information during algorithm execution. Visualization and graphical representation were achieved using RVIZ, allowing for a visual representation of the algorithm's operation.

3.1.3 Selection of Experimental Site: To validate the feasibility of the proposed algorithm in a real-world environment, we collected our experimental data at the test site. The satellite image of the campus is shown in Figure 5.

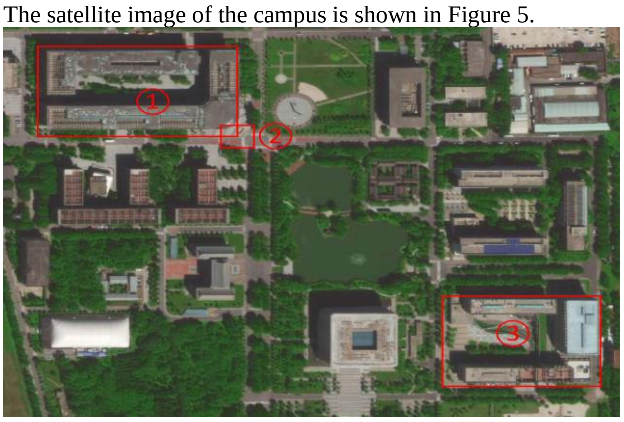

Figure 5. Experimental site.

The data collected at Site 1 was used for evaluating the SLAM algorithm and had a total length of approximately 800m. The data collected at Site 2 was utilized for IMU calibration as well as the calibration between IMU and LiDAR. Site 3 was chosen for collecting data to evaluate the improved point cloud registration algorithm, with a total length of around 500m.

3.2 IMU and LiDAR Calibration Experiment

3.2.1 IMU Self-Calibration Experiment and Result Analysis: In this section, we conducted an IMU self-calibration experiment to obtain the zero-offset errors of the three axes of the IMU. The open-source imu_utiles toolbox was used for this purpose. The specific steps of the experiment were as follows: First, the IMU was kept stationary in place for two minutes. Then, it was rotated in any direction, followed by a two-second static period. This process was repeated 50 times.

Obtained IMU Accelerometer Calibration Results:

The IMU was subjected to a calibration experiment to obtain the zero-offset errors of the three axes of the accelerometer. The open-source imu_utiles toolbox was used for this purpose. The specific steps of the experiment were as follows: First, the IMU was kept stationary in place for two minutes. Then, it was rotated in any direction, followed by a two-second static period. This process was repeated 50 times.

The obtained IMU accelerometer calibration results are shown in Figure 6.

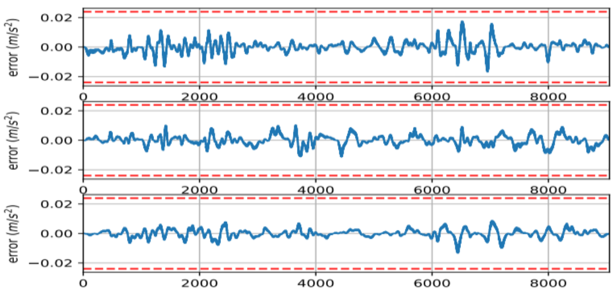

Figure 6. Errors in acceleration for each axis of the IMU.

From Figure 6, it can be observed that the errors in the accelerometer readings of the IMU fluctuate within the range of -0.02 to +0.02 over a certain period, with a maximum value not exceeding 0.02. Based on this, the average zero-offset error of the IMU accelerometer can be determined as:

$$[0.035698, 0.085627, 0.009563]^T \tag{11}$$

Errors in Angular Velocity for the IMU are shown in Figure 14:

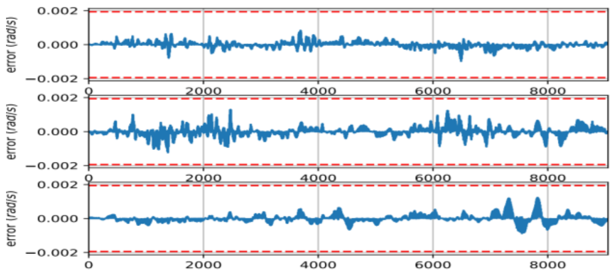

Figure 7. Errors in angular velocity for each axis of the IMU.

From Figure 7, it can be observed that the errors in the angular velocity measurements of the IMU fluctuate within the range of -0.002 to +0.002. The average error can be calculated as:

$$[-0.000567, 0.000264, 0.000056]^T \tag{12}$$

IMU Gyroscope Errors are shown in Figure 8:

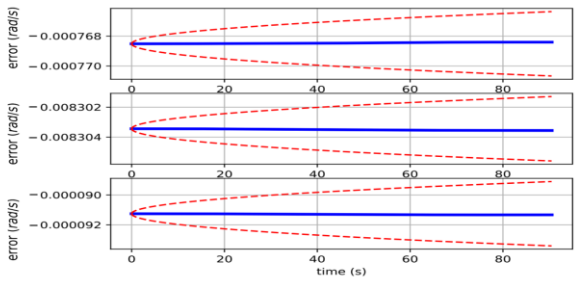

Figure 8. Errors in the gyroscope for the IMU.

The obtained gyroscope bias is:

$$\left[-1.3568e^{-06}, 1.7632e^{-06}, 1.4362e^{-06}\right]^T \tag{13}$$

The gyroscope errors of the IMU are influenced by factors such as zero-drift, which refers to the instrument itself or temperature effects; random noise, which is the interference on the gyroscope signal from electromagnetic fields or vibrations; vibration interference, which is external mechanical vibrations; and power supply voltage variations. However, the impact of these factors on the instrument is minimal. As can be seen from the figure, the gyroscope errors of the IMU are very small, with minimal angular bias errors. Therefore, the cumulative effect of these errors on the IMU can be considered negligible.

3.2.2 LIDAR and IMU Calibration Experiment Results and Analysis: In this study, the opencalib toolbox was used for the calibration of the LIDAR and IMU, employing a self-calibration method. This method minimizes the eigenvalues of the covariance matrix, ensuring that the feature points in the local map are distributed on the same plane or edge. By minimizing the distance between feature points and feature planes or edges, the extrinsic calibration from the LIDAR to the IMU is optimized. The Bundle Adjustment (BA) algorithm is then utilized to minimize the distance between each plane feature point and its corresponding plane. The experimental setup for data collection should follow the procedure depicted in Figure 9.

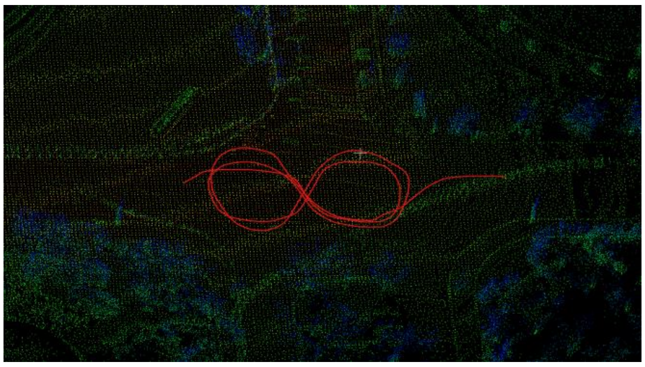
Figure 9. Field data collection trajectory route map.

This study collected data four times and conducted calibration on each dataset. The results obtained are shown in the following table:

Serial number	Roll	Pitch	Yaw	X(cm)	Y(cm)	Z(cm)
1	-0.007	0.004	-1.608	-4.652	2.322	15.261
2	-0.009	0.006	-1.670	-4.591	2.467	15.167
3	-0.008	0.004	-1.687	-4.851	2.239	15.988
4	-0.013	0.005	-1.593	-4.425	2.355	15.462

Table 3. The results of the four calibrations for the LIDAR and IMU calibration are as follows.

The final calibration results, obtained by averaging the values from the four calibration sessions, are presented in Table 4:

Serial number	Roll	Pitch	Yaw	X(cm)	Y(cm)	Z(cm)
1	-0.093	-0.004	-1.639	-4.630	2.346	15.169

Table 4. The final calibration results for the LIDAR and IMU are presented in the following table.

By incorporating the calibration results into the algorithm, it is evident that the mapping accuracy improves significantly, eliminating the occurrence of drift in the generated maps. Please refer to Figures 10 and 11 for a visual representation.

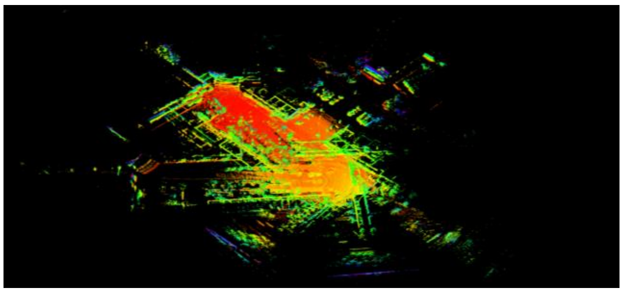
Figure 10. Experimental image before calibration.

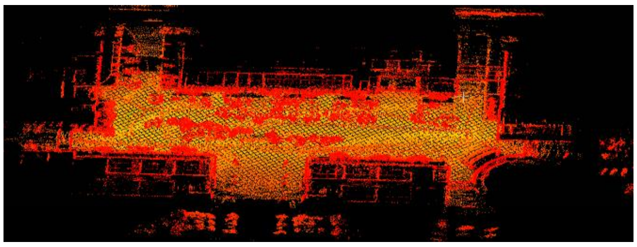
Figure 11. Experimental image after calibration.

3.3 Experimental Results and Analysis of the SLAM Algorithm

3.3.1 Efficiency Analysis of Improved Point Cloud Registration and Backend Optimization Experiment: To validate the robustness of the laser SLAM algorithm with the inclusion of IMU sensors and the feasibility of the proposed multi-sensor fusion system, this study conducted SLAM mapping in a real outdoor campus environment. Figures 12 and 13 illustrate the mapping results of the proposed algorithm in different campus scenarios.

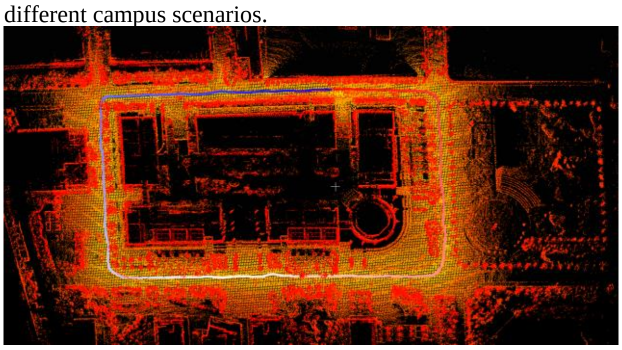
Figure 12. Mapping results in an outdoor scene 1.

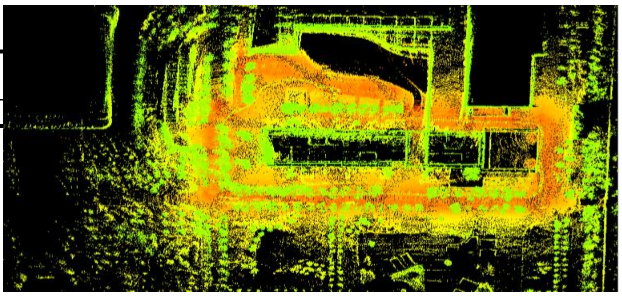
Figure 13. Mapping results in an outdoor scene 2.

From the above two effect images, it can be observed that our proposed algorithm exhibits no significant drift. There is no evident point cloud overlap for edges, buildings, and feature points. The maps generated by our algorithm exhibit clearer contours, and the pose estimation remains stable throughout the subsequent mapping process, achieving closure successfully.

To further evaluate the efficiency and accuracy of our proposed improved sparse voxel-based point cloud registration and backend mapping, a comparative experiment was conducted against the Lego-Loam algorithm. The comparative experiment involved comparing the runtime of the algorithms and their overall accuracy.

Table 5 presents the time consumed by the point cloud registration process, which involves K-NN nearest neighbour search, and the point cloud mapping process, which encompasses state estimation and fusion of IMU and lidar data. By analyzing the runtime of these three processes in point cloud mapping, the optimization of our algorithm in both the frontend and backend stages can be effectively verified.

Algorithm	Point cloud registration time (ms)	Odometry processing time (ms)	One frame of point cloud mapping time (ms)	Total processing time (ms)
The algorithm in this paper	19.362	9.365	76.681	105.408
Lego-Loam	27.654	9.694	89.642	127.990

Table 5. Analysis of runtime comparison.

Data analysis reveals that our proposed algorithm outperforms the Lego-Loam algorithm by reducing the total runtime for a single frame of point cloud data by 22.582ms. The reduction in runtime is primarily concentrated in the point cloud registration stage (8.292ms) and the mapping stage (12.961ms). Specifically, the improved point cloud registration method significantly reduces the registration time during data preprocessing, resulting in a 29.99% improvement in point cloud registration. Furthermore, the adoption of iterative extended Kalman filtering in the mapping stage enables efficient iteration and processing of point clouds, leading to the rapid generation of local point cloud maps. However, it should be noted that our algorithm exhibits a 17.64% increase in runtime compared to Lego-Loam.

3.3.2 Precision Analysis: In terms of accuracy, the evaluation mainly focuses on the drift error per hundred meters. Table 6 presents the absolute trajectory errors for the first scene, with the maximum relative trajectory error being 0.139%, the minimum relative trajectory error being 0.089%, and the average trajectory error being 0.114%. The relative trajectory error fluctuates around 0.110%. These results demonstrate that our proposed algorithm achieves high scene accuracy, with a relative trajectory error of 0.114%. These findings highlight the high precision of our algorithm in the given scene, as evidenced by the low relative trajectory error of 0.114%.

Serial number	APE（m）							RPE（%）
	Max	Mean	Median	Min	Rmse	Sse	Std	
1	1.9507	0.8268	0.7946	0.1682	0.8691	4241.4618	0.2677	0.1091
2	1.4243	0.6501	0.6881	0.0466	0.7115	2842.8620	0.2891	0.0892
3	2.2327	1.0051	0.9757	0.0528	1.1116	6938.5590	0.4748	0.1393
4	2.0422	0.8098	0.6281	0.1252	0.9403	4965.1192	0.4780	0.1185
Avg	1.9124	0.8230	0.7716	0.0982	0.9081	4747.0005	0.3774	0.1142

Table 6. Absolute trajectory error for scene 1.

Serial number	APE（m）							RPE（%）
	Max	Mean	Median	Min	Rmse	Sse	Std	
1	2.6786	0.8697	0.6370	0.0457	1.0494	4723.6321	0.5872	0.2105
2	3.8995	1.8139	1.4866	0.0811	2.0640	18271.8649	0.9849	0.4133
3	1.4059	0.3622	0.3188	0.0301	0.4044	701.7439	0.1800	0.0817
4	3.9205	0.8038	0.5324	0.0148	1.1331	5507.4360	0.7987	0.2273
Avg	2.9761	0.9624	0.7437	0.0429	1.1627	7301.1692	0.6377	0.2325

Table7. Absolute trajectory error for scene 2.

Table 7 presents the absolute trajectory error for Scene 2. The maximum relative trajectory error is 0.4133%, the minimum relative trajectory error is 0.0817%, and the average trajectory error is 0.2325%. The relative trajectory error fluctuates around 0.2300%. These results indicate that the proposed algorithm achieves high accuracy in Scene 2, with a relative trajectory error of 0.2325%.

In summary, both Table 6 and Table 7 demonstrate that the proposed algorithm exhibits low absolute and relative trajectory errors, indicating high precision. Qualitative experiments were conducted comparing the proposed algorithm with the Lego-Loam algorithm using data from Scene 2. The qualitative experiment results are shown in Table 8 and Table 9.

Please note that the content of Table 8 and Table 9 was not provided, so I cannot translate it for you.

Serial number	The algorithm in this paper		Lego-Loam	
	APE(m)	PRE(%)	APE(m)	PRE(%)
1	0.8691	0.1091	1.3594	0.1700
2	0.7115	0.0892	3.2486	0.4061
3	1.1112	0.1393	2.3941	0.2993
4	0.9404	0.1185	2.1136	0.2642
AVG	0.9082	0.1142	2.2790	0.2849

Table8. Experimental results for algorithm comparison in scene 1.

Serial number	The algorithm in this paper		Lego-Loam	
	APE(m)	PRE(%)	APE(m)	PRE(%)
1	1.0494	0.2105	1.6542	0.3308
2	2.0640	0.4133	2.0364	0.4073
3	0.4045	0.0817	1.2235	0.2447
4	1.1331	0.2273	1.4563	0.3641
AVG	1.1627	0.2325	1.5936	0.3185

Table9. Experimental results for algorithm comparison in Scene 2

The experimental results demonstrate that in both Scene 1 and Scene 2, using the data collected on the campus, the errors of the two algorithms have minimal impact on the mapping results. The proposed algorithm exhibits lower relative trajectory error and absolute trajectory error compared to the Lego-Loam algorithm, indicating higher accuracy and stronger robustness.

The experimental results demonstrate that in both Scene 1 and Scene 2, using the data collected on the campus, the errors of the two algorithms have minimal impact on the mapping results. The proposed algorithm exhibits lower relative trajectory error and absolute trajectory error compared to the Lego-Loam algorithm, indicating higher accuracy and stronger robustness.

Furthermore, the proposed iVox voxel-based point cloud registration and iterative extended Kalman filtering algorithms were compared experimentally. Finally, the fusion algorithm was tested in a real campus environment to validate its feasibility. A comparison experiment with the Lego-Loam algorithm was conducted to evaluate the performance in terms of time consumption, robustness, and accuracy. The analysis of the computational time for 10 single-frame point clouds concluded that the fusion SLAM algorithm based on iVox point cloud registration and iterative extended Kalman filtering exhibited a 17.64% improvement in time consumption compared to Lego-Loam.In terms of accuracy, self-testing and comparative experiments were conducted on buildings A and F on the campus. The experimental results showed that the absolute trajectory errors were 0.8230m and 0.8038m, and the relative trajectory errors were 0.1142% and 0.2325% for Scene 1 and Scene 2, respectively. The comparative experiments further confirmed the higher accuracy and robustness of the proposed algorithm.

4. CONCLUSION

This article presents research on multi-sensor SLAM, which combines IMU and LiDAR sensors, to address the issues of motion drift and low mapping efficiency in the mapping process of the LiDAR SLAM algorithm. The article introduces the models of IMU and LiDAR and derives the IMU preintegration through analysis of the IMU model. The IMU self-calibration and the joint calibration experiment of IMU and LiDAR are completed to solve the mapping drift caused by inconsistent coordinate systems in the multi-sensor fusion process. In the SLAM front-end, the voxel method iVox is introduced to replace the traditional tree-based point cloud registration. The back end utilizes iterative extended Kalman filtering to fuse the data from IMU and LiDAR sensors, achieving sensor fusion in the algorithm. The efficiency of mapping is evaluated by measuring the time consumption of mapping for multiple single-frame point clouds. The experimental results show that the efficiency is improved by 29.99% in the point cloud registration stage, 14.46% in the single-frame point cloud mapping stage, and 17.64% in total time consumption. To validate the effectiveness and feasibility of the proposed algorithm, this study conducted data collection in three selected areas of the test site under real-world environments. Qualitative and quantitative analysis is performed on the collected data, and algorithm testing is conducted in various campus scenarios. The average absolute trajectory errors are 0.9081m and 1.1627m, and the average relative trajectory errors are 0.114% and 0.233% respectively. A comparison of accuracy is also made with the Lego-Loam algorithm, showing that the proposed algorithm has a relative trajectory error of 0.171% lower than Lego-Loam in scenario one and 0.086% lower in scenario two. This confirms the advantages of the proposed algorithm in terms of robustness and accuracy. Through this research, we have successfully proposed a multi-sensor SLAM algorithm that integrates IMU and LiDAR sensors and demonstrated its robustness and accuracy through experiments. This research has important practical implications for improving the mapping efficiency and accuracy of LiDAR SLAM algorithms.

ACKNOWLEDGEMENTS

This research was funded by the National Natural Science Foundation of China (Grant Numbers 42201483) and the China Postdoctoral Science Foundation (Grant Numbers 2022M710332).

REFERENCES

Chou, C.C., Chou, C.F., 2022. Efficient and Accurate Tightly-Coupled Visual-Lidar SLAM. *IEEE Transactions on Intelligent Transportation Systems*, 23(9), 14509–14523. doi.org/10.1109/TITS.2021.3130089.

Durrantwhyte, H.F., Bailey, T., 2006. Simultaneous localization and mapping (SLAM): part II. *IEEE Robotics & Automation Magazine*, 13(2), 99–110. doi: 10.1109/MRA.2006.1638022.

Elamin, A., Abdelaziz, N., El-Rabbany, A., 2022. A GNSS/INS/LiDAR Integration Scheme for UAV-Based Navigation in GNSS-Challenging Environments. *Sensors 22*, 9908. doi.org/10.3390/s22249908.

Gentil, C. L., Vidal-Calleja, T., Huang, S., 2018. 3D Lidar-IMU Calibration Based on Upsampled Preintegrated Measurements for Motion Distortion Correction. *In 2018 IEEE International Conference on Robotics and Automation (ICRA)* , 2149-2155. doi: 10.1109/ICRA.2018.8460179.

Hou, L., Xu, X., Ito, T., et al. , 2022. An Optimization-Based IMU/Lidar/Camera Co-calibration Method. *In 2022 7th International Conference on Robotics and Automation Engineering (ICRAE)* , 118–122. doi: 10.1109/ICRAE56463.2022.10056217.

Lin, J., Zhang, F. , 2022. R3LIVE: A Robust, Real-time, RGB-colored, LiDAR-Inertial-Visual tightly-coupled state Estimation and mapping package. *In 2022 International Conference on Robotics and Automation (ICRA),* 10672-10678. doi: 10.1109/ICRA46639.2022.9811935.

Li, S., Wang, L., Li, J., et al., 2021. 3D LiDAR/IMU Calibration Based on Continuous-time Trajectory Estimation in Structured Environments. in *IEEE Access* , 35(8), 1242–1264. doi: 10.1109/ACCESS.2021.3114618.

Lv, J., Xu, J., Hu, K., et al., 2020. Targetless Calibration of LiDAR-IMU System Based on Continuous-time Batch Estimation. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* , 9968-9975. doi: 10.1109/IROS45743.2020.9341405.

Shan, T., Englot, B., 2019. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* , 4758-4765.doi: 10.1109/IROS.2018.8594299.

Strobl, K. H., Hirzinger, G. , 2007. Optimal Hand-Eye Calibration. *In IEEE/RSJ International Conference on Intelligent Robots and Systems* , 4647-4653, doi: 10.1109/IROS.2006.282250.

Zhang, J., Singh, S., 2018. Laser-visual-inertial Odometry and Mapping with High Robustness and Low Drift. *Journal of Field Robotics*, 35(8), 1242–1264.

Zou, Q., Sun, Q., Chen, L., et al., 2021. A Comparative Analysis of LiDAR SLAM-Based Indoor Navigation for Autonomous Vehicles. *IEEE Transactions on Intelligent Transportation Systems*,23(7), 6907-6921. doi: 10.1109/TITS.2021.3063477.