


Article

A Method of Calibration for the Distortion of LiDAR Integrating IMU and Odometer

Qiuxuan Wu ^{1,*} , Qinyuan Meng ¹, Yangyang Tian ², Zhongrong Zhou ¹, Cenfeng Luo ¹, Wandeng Mao ², Pingliang Zeng ¹, Botao Zhang ¹ and Yanbin Luo ¹

¹ School of Automation, Hangzhou Dianzi University, Hangzhou 310018, China

² Electric Power Research Institute of State Grid Henan Electric Power Company, Zhengzhou 450018, China

* Correspondence: wuqx@hdu.edu.com

Abstract: To improve the motion distortion caused by LiDAR data at low and medium frame rates when moving, this paper proposes an improved algorithm for scanning matching of estimated velocity that combines an IMU and odometer. First, the information of the IMU and the odometer is fused, and the pose of the LiDAR is obtained using the linear interpolation method. The ICP method is used to scan and match the LiDAR data. The data fused by the IMU and the odometer provide the optimal initial value for the ICP. The estimated speed of the LiDAR is introduced as the termination condition of the ICP method iteration to realize the compensation of the LiDAR data. The experimental comparative analysis shows that the algorithm is better than the ICP algorithm and the VICP algorithm in matching accuracy.

Keywords: motion distortion; IMU; sensor fusion; odometer; ICP



Citation: Wu, Q.; Meng, Q.; Tian, Y.; Zhou, Z.; Luo, C.; Mao, W.; Zeng, P.; Zhang, B.; Luo, Y. A Method of Calibration for the Distortion of LiDAR Integrating IMU and Odometer. *Sensors* **2022**, *22*, 6716. <https://doi.org/10.3390/s22176716>

Academic Editors: Yong Liu and Xingxing Zuo

Received: 25 July 2022

Accepted: 2 September 2022

Published: 5 September 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Exact pose estimation is the key technology for mapping, location, and navigation in the field of the mobile robot [1], which can provide the message of the robot's position and gesture in real time. Sensors used to obtain robot pose estimates include LiDAR, cameras, wheel encoders, IMUs, etc. According to the different sensors the robot is equipped with, SLAM technology is divided into visual SLAM and laser SLAM. Although the sensor used in visual SLAM has low cost and rich image information, it has a great impact on the normal operation of the camera under weak- or no-light conditions. What is more, because the image information is too rich, the algorithm requires high processor performance. Now, the mainstream mobile robots are still dominated by laser sensors [2], such as the unmanned delivery vehicle of JD and the "prime" unmanned delivery vehicle of Amazon.

A 2D LiDAR estimates the pose of the sensor by scan-matching two adjacent frames of laser data [3]. However, only relying on 2D laser SLAM to estimate the pose of the robot has many limitations. The frequency of the system output estimated pose is low, and the running time becomes longer, which will generate a large cumulative error and eventually affects the positioning and map construction of the robot. A cartographer algorithm [4] is developed using a SICK radar, and the frame rate can reach more than 100 Hz. The motion distortion can be ignored, so there is no distortion correction algorithm module. However, the frame rate of most LiDAR is around 10 Hz. Without distortion correcting, there will be distortion error appearing in LiDAR data, which is hard to eliminate through loopback detection and back-end optimization, etc. The research on this issue has great practical significance. Many domestic and foreign works have been conducted on removing motion distortion and false match of LiDAR data in recent years. Yoon et al. [5] proposed an unsupervised parameter learning in the Gaussian variational inference setting, which combines classical trajectory estimation of mobile robots and deep learning on rich sensor data to learn a complete estimator via the deep network. However, it requires a large amount of calculation, the captured laser

data cannot complete feature extraction or matching when the environment is not clearly structured, and the real-time and robustness are poor. Therefore, it is only suitable for small indoor scenes with clear structure instead of open large outdoor scenes. Hyeong et al. [6] proposed an ICP (Iterative Closest Points, iterative closest point) outlier rejection scheme to compare the laser data of the scanned environment and select matching points and reject the algorithm that does not match parts. The ICP algorithm needs to be provided with an initial value, and the matching accuracy of the ICP algorithm directly depends on whether the initial value is accurate. However, in the process of acquiring the surrounding environment, the laser is often accompanied by the motion of the robot. Especially when the laser frame rate is small, the captured laser data will produce motion distortion, and there will be a large error with the real environment over time. Xue et al. [7] proposed a simultaneous fusion of IMU, wheel encoder, and LiDAR to estimate the own motion of a moving vehicle. However, this method does not propose a countermeasure for discontinuous laser scanning. Bazer and Cherfaoui [8] proposed a method for correcting errors caused by time stamp errors during sensor data acquisition, but this scheme assumes that the scanning angle of the laser is fixed and the quadratic interpolation assumption is too simplistic, which cannot meet the complex outdoor environment. Hong et al. [9] proposed a new approach to enhancing ICP algorithms by updating speed, which estimates the speed of the LiDAR through ICP iterations, and uses the estimated speed to compensate for scan distortion due to motion. Although it considers the motion of the robot into consideration, its assumption of uniform motion is too ideal; for low-frame rate LiDAR, the assumption of uniform motion does not hold.

Aiming at the above problems, this paper proposes an improved algorithm for estimated speed scan matching that integrates an IMU and odometer. This algorithm is called Iao_ICP (ICP that integrates IMU and Odometer) in this paper. The main contributions of this paper are as follows: (1) The algorithm uses the linear interpolation method to obtain the pose of LiDAR, which solves the alignment problem of the discontinuous laser scan data. (2) The data fused by the IMU and the odometer provides a better initial value for the ICP, and the estimated speed of the LiDAR is introduced as the iterative value of the ICP method to realize the termination condition of LiDAR data compensation.

The rest of the paper is organized as follows: Firstly, the causes of motion distortion in the traditional ICP algorithm are analyzed. Secondly, the incremental information of the wheel odometer and the angular velocity information of the IMU are integrated into the pose estimation. Finally, through data sets and physical experiments, the effectiveness of the proposed algorithm in removing motion distortion and improving the accuracy of map construction is demonstrated.

2. Causes of LiDAR Motion Distortion

The mechanical LiDAR is driven by an internal motor to rotate the radar ranging core 360° clockwise to obtain the surrounding environment data. Each frame of laser data is encapsulated by the data information obtained by a certain number of discrete laser beams, and the laser data of each frame is not obtained instantaneously. The data distortion of LiDAR is related to the motion state of the robot which carries LiDAR. When laser scanning is accompanied by the motion of the robot, the laser data of each angle is not obtained instantaneously. When the scanning frequency of the LiDAR is relatively low, the motion distortion of the laser frame caused by the motion of the robot cannot be ignored [10].

The current domestic LiDAR rotation frequency is about 5–10 Hz. When the robot carrying the LiDAR is stationary, the measurement data of the LiDAR has no error, but in the SLAM system, the robot is often in a state of motion. Take the environment shown in Figure 1 as an example. It can be seen that the distance data of each laser beam are collected in different poses, as shown in the pose of points A and B. Suppose the robot is moving at a constant speed, the solid curved arrow indicates that the LiDAR rotation direction is clockwise, and the solid long straight arrow indicates that the LiDAR moves from point A to point B along the X direction. Then, in the case of no motion distortion correction during this period, the LiDAR data will have a motion distortion error of Δx .

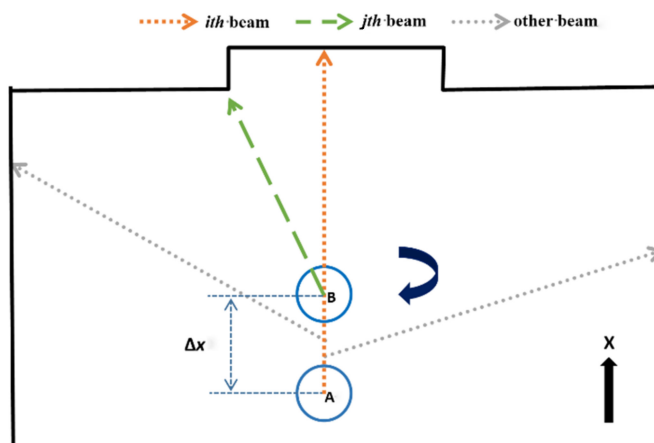


Figure 1. The acquisition process of one frame of LiDAR data.

As described above, when the robot obtained a frame of LiDAR data, the laser is obtained at point A, and the laser is obtained at point B. However, when general LiDAR drives package data, it is assumed that all laser beams of a frame of LiDAR data are obtained in the same pose and instantaneously, that is, all laser beams are obtained from point A data. Its pose actually produces motion changes, and each laser point is generated on a different reference pose, which eventually causes the environmental distortion of the laser collection. As Figure 2 shows, the left picture is the actual environment, while the dotted line in the picture on the right is the true value, and the solid line is the LiDAR data with motion distortion.

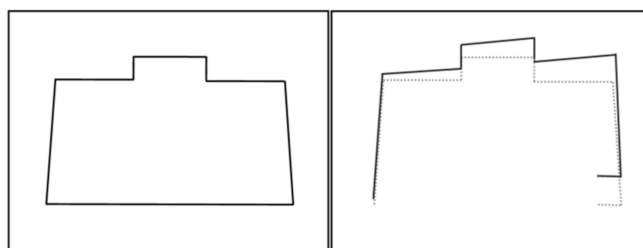


Figure 2. LiDAR motion distortion.

3. Principle of ICP Algorithm

The ICP algorithm [11] was first developed by Beals and McKay in 1992. The ICP algorithm is essentially an optimal registration method based on the least-squares method. ICP first matches each point of the target laser data with the closest point of the reference laser data and finds the rotation matrix R and translation matrix p , which are used to convert the two. Afterward, the laser matching is iteratively optimized by repeatedly generating pairwise closest points until the convergence accuracy requirements for correct registration are met. The ICP algorithm first needs to determine an initial pose, and the selected initial value will have an important impact on the final registration result. The algorithm may fall into a local optimum instead of a global minimum if the initial value is not chosen properly.

Given $X = \{x_1, x_2, \dots, x_{N_x}\}$ as a frame of laser data, $P = \{p_1, p_2, \dots, p_{N_p}\}$ as the laser data of adjacent frames, and $T = \{T_1, T_2, \dots, T_i\}$ as the transformation matrix of laser data of adjacent frames, x_i and p_i indicate the coordinates of the laser spot, N_x and N_p indicate the number of laser dots, and i indicates the frame number of laser data. This paper defined a minimizing objective Function (1) to transform P through the coordinates, and cover the maximum to X [11].

$$E(R, p) = \frac{1}{N_p} \sum_{i=1}^{N_p} \| x_i - Rp_i - t \|^2 \tag{1}$$

The resulting transformation matrix T can be described as (2):

$$T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \quad (2)$$

The processing steps of the given objective function are shown as follows:

Step1: Solving the mean value of LiDAR data X and P :

$$U_x = \frac{1}{N_x} \sum_{i=1}^{N_x} X_i, U_p = \frac{1}{N_p} \sum_{i=1}^{N_p} P_i;$$

Step2: Remove the translation of LiDAR data X and P to distributed laser data around the mean value:

$$x'_i = x_i - U_x, p'_i = p_i - U_p$$

Step3: Define matrix, and make SVD decomposition of it, where H is the matrix to be decomposed by SVD, U and V are the two non-singular matrices decomposed, and $\sigma_1, \sigma_2,$ and σ_3 are the three singular values decomposed, respectively:

$$H = \sum_{i=1}^{N_p} x'_i p'^T_i = U \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} V^T$$

Step4: Calculate the solution of the objective function:

$$R = UV^T, p = u_x - Ru_p$$

Since the ICP algorithm uses the closest point as the corresponding point, the initial result may be different from the real environment. However, the results converge to the base environment by repeating this process. The LiDAR scan data for frame i , namely, X , are shown in Figure 3a. The LiDAR scan data for frame $i + 1$, namely, P , are shown in Figure 3b. The first step of ICP iteration is shown in Figure 3c. The closest point between X and P is found as Figure 3d shows. The first matching estimated transformation and updated P by $p'_i = T_1 p_i$, which is shown in Figure 3e. The X and P matched after many iterations, as Figure 3f shows. Final pose estimation is solved through the transformation of $T = T_n T_{n-1} \cdots T_2 T_1 (i = 1, \cdots, n)$, namely:

$$x_i = T_n T_{n-1} \cdots T_2 T_1 p_i = T p_i \quad (3)$$

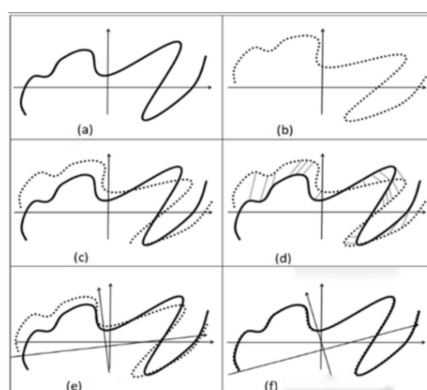


Figure 3. The principle of ICP algorithm. (a) Frame i (b) Frame $i + 1$ (c) Start matching (d) Find adjacent (e) First match (f) After multiple iterations of matching.

4. Estimation Speed Scan Matching Algorithm Based on IMU and Odometer

A wheeled odometer and IMU are introduced to compensate for motion distortion of laser data caused by robot moves. Direct measurement of displacement and angle

information through a wheel odometer or direct measurement of angular velocity and linear acceleration through an IMU [12], then integrate them, respectively, to obtain the displacement and angle information. In the ideal conditions, the wheel odometer or IMU has a high-precision local pose estimation ability because of the high pose update frequency (higher than 200 Hz) of the above sensors, which can accurately reflect the motion of the robot in real time [13]. What is more, these two types of sensors are completely decoupled from the robot state estimation, which can prevent the introduction of errors. However, on the one hand, during the actual movement of the robot, the wheels will slip and the accumulated error will occur, which leads to a certain deviation in the obtained odometer angle data when only the encoder is used, and the error increases with the running time and the stroke increases. On the other hand, the linear acceleration accuracy of the IMU is poor, though it has high angular velocity measurement accuracy, and the local accuracy of the quadratic integral is still very poor, which leads to a certain deviation of obtained displacement data. Therefore, this paper proposes the Iao_ICP algorithm, and the algorithm framework is shown in Figure 4. First, the information of the IMU and the odometer is fused, and the pose of the LiDAR is obtained using the linear interpolation method to remove most of the motion distortion. Then, scan matching of LiDAR data is conducted using the ICP method. Data fused by the IMU and odometer provide a better initial value for ICP, and estimated speed is introduced as a termination condition for iteration of the ICP method [14]. The matching result is used as the correct value, and the error value of the odometer is obtained. The error value is evenly distributed to each point, and the position of the laser point is corrected again, so as to further determine the pose of the laser point.

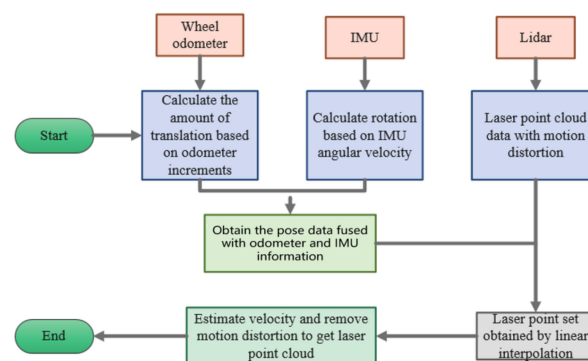


Figure 4. The architecture diagram of the Iao_ICP algorithm.

4.1. Pose Estimation with Fusion of IMU and Odometer

The chassis control system of the mobile robot reads the IMU data and the odometer data. Each time the IMU data are read, the odometer data can also be obtained without considering the problem of time synchronization. That means the IMU pose queue and odometer pose queue maintain strict alignment, which can directly fuse both to generate a new pose queue. However, the update frequency of low-cost LiDAR is generally only 5–10 Hz, which leads to the new pose queue after fusion cannot maintain strict alignment with the pose queue of laser frames. Although there is no way to obtain the pose of the laser frame directly from the fused pose queue since the pose queues of the two are not strictly aligned, the pose of the laser frame can be obtained by linearly interpolating the fused pose queue. Below are the detailed steps to obtain the estimated pose based on the linear interpolation method by fusing the IMU and odometer data:

Step1: As the start time of the current laser frame, the end time of the current laser frame, and the time interval between two laser beams have been known. Odometer data and IMU data are stored in a queue in the same chronological order, and the team leader is the earliest. There are oldest odometer and IMU data timestamps, and latest odometer and

IMU data timestamps. First, solve the new queue generated by fusing odometry and IMU data within the above timestamps. The fusion expressions are shown below:

$$\begin{cases} Odom_Imu_List[i].x = OdomList[i].x \\ Odom_Imu_List[i].y = OdomList[i].y \\ Odom_Imu_List[i].\theta = ImuList[i].\theta \end{cases} \quad (4)$$

In the formula, $Odom_Imu_List [i]$ is the fused pose data at the t_i moment, $OdomList [i]$ is the odometer pose data at the t_i moment, $ImuList [i]$ is the IMU pose data at the t_i moment, and $x, y,$ and θ are the X-axis data, Y-axis data, and angle data in the pose data, respectively.

Step2: Solve the emission pose corresponding to each laser in the current frame of laser data, namely, to solve the robotic pose at the time of $\{t_s, t_s + \Delta t, \dots t_s + i\Delta t \dots t_e\}$. It is reasonable to assume that the robot moves at a uniform speed during the data update of the fusion of two adjacent frames due to the high update frequency of odometer data and IMU data. Linear interpolation can be used on this assumption, as shown in Figure 5.

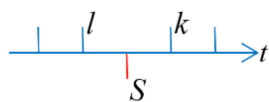


Figure 5. Linear interpolation of laser pose.

Suppose there are corresponding fused pose queues at the time of l, k for laser data, but not at the time of s , and the value of s is greater than l , and less than k . Then, solve the pose of robot p_s, p_m, p_e corresponding to the three moments t_s, t_m, t_e ($t_s < t_m < t_e$). The pose of the first laser beam can be calculated with the Formula (5). In the same way, the emission pose of the last laser beam and the laser beam at the middle time can be obtained.

$$\begin{cases} p_l = Odom_Imu_List[l] \\ p_k = Odom_Imu_List[k] \\ p_s = p_l + \frac{p_k - p_l}{k - l}(s - l) \end{cases} \quad (5)$$

Step3: Following the method in the Step2, p_m and p_e can be solved. Further assumed, the robot performs uniform acceleration motion during a frame of laser data. Thus, the pose of the robot is a quadratic function of time, as Figure 6 shows. Thus, using the known robot pose p_s, p_m, p_e as the independent variable, a quadratic curve function $P(t) = At^2 + Bt + C$ ($t_s < t < t_e$) can be obtained by interpolation, and A, B, C are the coefficients of the quadratic function. Next, the value of every time $\{t_s, t_s + \Delta t, \dots t_s + i\Delta t \dots t_e\}$ can be substituted into a curve, and the pose of each laser point data in global coordinate system $\{p_{t_s}, p_{t_s + \Delta t}, \dots p_{t_s + i\Delta t} \dots p_{t_e}\}$ can be obtained.

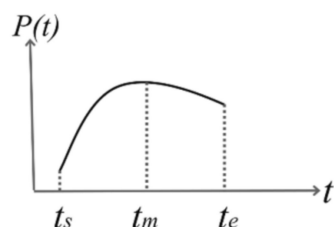


Figure 6. Pose function graph.

Step4: The relative pose (array form) of the laser point in the global coordinate system is converted into a pose change matrix. Then, convert the coordinate information in the radar coordinate system x_i to the coordinates in the global coordinate system, as Formula (6) shows.

$$x'_i = V2T(p_i)x_i \quad (6)$$

In the above Formula (6), function $V2T(p_i)$ is a whole, indicating that the relative pose in the form of an array p_i is converted into a pose transformation matrix in the form of a matrix. By the coordinate information in radar system x_i left multiplication corresponding matrix p_i , the coordinate of the radar coordinate system x_i can be translated into the coordinate in the global coordinate system x'_i , because p_i is the pose in the global coordinate system.

Step5: According to the coordinates of the scanning point corresponding to each laser beam in the global coordinate system x'_i , the laser data of the laser scan point in the LiDAR coordinate system can be solved with Formula (7).

$$\begin{cases} x'_i = (p_x, p_y) \\ range = \sqrt{p_x \cdot p_x + p_y \cdot p_y} \\ angle = \text{atan2}(p_y, p_x) \end{cases} \quad (7)$$

For the first equation above, p_x, p_y are the coordinates of the i th frame of laser data in the LiDAR coordinate system on the X- and Y-axis, respectively.

For the second equation above, the coordinates p_x and p_y on the x and y axes of the laser x_i frame in the laser coordinate system are known. The distance point x_i from the origin of the laser coordinate system can be found according to the "Pythagorean Theorem".

For the third equation above, p_x and p_y have been found, and the angle between point x_i and X-axis can be solved according to inverse trigonometric functions. The specific implementation process of the algorithm is shown in Algorithm 1.

Algorithm 1: A Pose Estimation Algorithm Based on IMU and Odometer

Input: Odometer pose queue $OdomList[i]$, IMU pose queue, and laser pose queue x_i

Output: laser pose queue X_n

1: **for** $i = 1:n$ **do**

2: $Odom_Imu_List[i].x = OdomList[i].x$;

$Odom_Imu_List[i].y = OdomList[i].y$;

$Odom_Imu_List[i].\theta = ImuList[i].\theta$; //fuse the data of odometer and IMU pose queue, then put into $Odom_Imu_List[i]$

3: **end for**

4: $p_s = \text{LinerInterp}(Odom_Imu_List[t_s])$;

$p_m = \text{LinerInterp}(Odom_Imu_List[t_m])$;

$p_e = \text{LinerInterp}(Odom_Imu_List[t_e])$; //Perform linear interpolation on the fusion pose of the start, end and intermediate moments, $\text{LinerInterp}()$ is function used to make linear interpolation

5: $P(t) = P(t) = At^2 + Bt + C$; //Substitute p_s, p_m, p_e into above formula in order, and the coefficients of quadratic curve functions A, B, C can be solved.

6: **for** $i = 1:n$ **do**

7: $p_i = Ai^2 + Bi + C$; //solve the pose of each laser point in global coordinate system p_i

8: $x'_i = V2T(p_i)x_i = (p_x, p_y)$; //obtain the pose of each laser point in the global coordinate system x'_i

9: $X_n = (range, angle) = (\sqrt{p_x * p_x + p_y * p_y}, \text{atan2}(p_y, p_x))$; //compose a new laser point set X_n

10: **end for**

4.2. Estimated Velocity and Laser Data Pose Compensation

To remove the motion distortion of the laser point cloud data, the speed of the robot needs to be estimated. Since the scanning period of LiDAR is about 0.1 s, it can be assumed that the speed of the robot is constant during this scanning period, and V_i is used to indicate the velocity in the LiDAR coordinate system at t_i time. Firstly, estimate the velocity V_i from the relative motion transformation between two adjacent frames of laser data X_i and X_{i-1} , supposing that n indicates the number of laser points of laser data X_i . The time interval

between two adjacent frames of laser points is Δt . x_0, x_1, \dots, x_n is the laser point of laser X_i , $t_{x_j} - t_{x_{j-1}} = \Delta t_s$ ($j = 0, 1, \dots, n - 1$).

Therefore, the estimated velocity V_i is:

$$V_i = \frac{T2V\left(T_{i-1}^{-1}T_i\right)}{\Delta t} \approx \frac{1}{\Delta t} \lg T_{i-1}^{-1}T_i \quad (8)$$

In Formula (8), $T_{i-1}^{-1}T_i$ is a whole, indicating the pose difference of the robot from $i-1$ time to i time in the radar coordinate system, and $T2V\left(T_{i-1}^{-1}T_i\right)$ indicates a way to convert the pose difference from matrix form to array form.

The pose of frame i and laser point j is:

$$T(t_i + j\Delta t_s) = T_i \cdot V2T(V_i \cdot j\Delta t_s) = T_i e^{j\Delta t_s V_i} \quad (9)$$

In Formula (9), $j\Delta t_s$ is the duration of laser point cloud data in frame i from laser point 0th to laser point j .

V_i is the origin velocity of laser point data in frame i .

$V_i \cdot j\Delta t_s$ is the pose difference of frame i laser data cloud from laser point 0th to laser point k .

$V2T(V_i \cdot j\Delta t_s)$ is the conversion of relative pose difference from array form to matrix form.

$T_i \cdot V2T(V_i \cdot j\Delta t_s)$ is to obtain the pose of laser point j in frame i by using frame i of laser point cloud data right-multiplied by the pose difference from the initial pose of the 0th laser point.

Substitute the above Formula (9) into Formula (3), the laser point cloud data collection X_i is converted into \bar{X}^* , and \bar{X}^* is the laser point cloud data collection after speed compensation.

$$\bar{X}^* = \left\{ e^{j\Delta t_s V_i} p_j \mid j = 0, 1, \dots, n \right\} \quad (10)$$

For some types of LiDAR, it takes 100 ms to perform a scan with a scan angle of 360° , which takes the estimation of robot motion later than the actual movement. To prevent this kind of delay, a backward compensation scheme can be used. Take the time corresponding to the last laser point as the reference time, the corresponding time of each laser point can be converted. With the above conditions, Formula (9) can be revised into:

$$T[t_i - (n - j)\Delta t_s] = T_i e^{(n-j)\Delta t_s (-V_i)} \quad (11)$$

Formula (10) can be revised into:

$$\bar{X} = \left\{ e^{(n-j)\Delta t_s (-V_i)} x_j \mid j = 0, 1, \dots, n \right\} \quad (12)$$

The specific implementation process of the algorithm is shown in Algorithm 2.

Algorithm 2: Estimating velocity and removing motion distortion from laser point cloud data combined with ICP

Input: the queue of laser pose X_n

Output: motion transformation matrix of adjacent laser frames T

1: $V = V_i$ //speed initialization

2: **do**

3: $T_{\Delta t_s} = e^{\Delta t_s(-V_i)}$ //the motion transformation matrix T is estimated by the speed of the two adjacent frames of laser light

4: **for** $j = 1 : n$ **do** //traverse all laser points in the current laser frame

5: $T_{j\Delta t_s} = T_{(j-1)\Delta t_s} T_{\Delta t_s}$ //calculate the motion transformation matrix of each laser point

6: $\bar{x}_{i_j} = T_{j\Delta t_s} x_{i_j}$ //Motion transformation for each laser point

7: **end for**

8: $T = \text{ICP}(\bar{X}^{-1}, \bar{X}_i, T)$ //iterative matching via ICP

9: $V = V_i$ //renew the value of velocity

10: $V_i = 1/\Delta \lg T$ //do the next round of speed estimation

11: **While** $\|V - V_i\| > e$ //when the speed error value is greater than the threshold e , execute the loop

5. Positioning Accuracy Evaluation of Laser Odometry after Motion Distortion Calibration

This experiment utilizes the sequences b0_2014_07_11_10_58_16 (denoted as ①), b0_2014_07_11_11_00_49 (denoted as ②), and b0_2014_07_21_12_42_53 (denoted as ③) in the Cartographer public dataset. The laser odometry accuracy of the Iao_ICP algorithm and the original Cartographer algorithm is quantitatively evaluated by executing this. Figure 7 shows the mapping effect of the Iao_ICP algorithm on the sequence. The processor of the test equipment is Intel (R) Core (TM) i5–5200 CPU 2.20 GHz and it has 8 GB RAM.

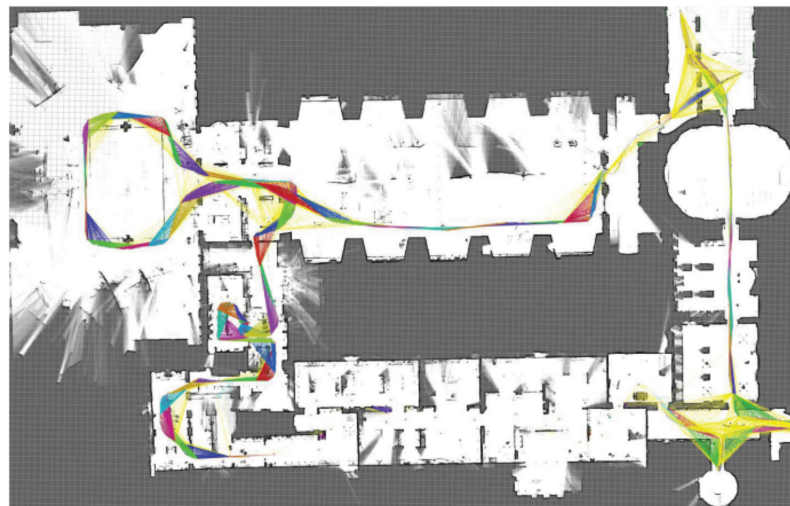


Figure 7. Mapping based on the b0_2014_07_11_11_00_49 sequence.

The analysis is performed by comparing the data calculated by the Iao_ICP algorithm with the Cartographer data set. Table 1 lists the absolute trajectory errors calculated by these two algorithms [15]. In addition, the Iao_ICP algorithm was used to calculate the relative trajectory error and compared with the relative trajectory error of the original Cartographer algorithm.

Table 1. Comparison results of absolute trajectory error between Iao_ICP algorithm and Cartographer algorithm.

Sequence	Algorithm	RMSE (m)	Average (m)	Maximum (m)	Minimum (m)
①	Iao_ICP	0.0179	0.0103	0.01356	0.0011
	Cartographer	0.023	0.0147	0.01428	0.0024
②	Iao_ICP	0.0166	0.0091	0.1510	0.0008
	Cartographer	0.0197	0.0096	0.1663	0.0011
③	Iao_ICP	0.0158	0.0089	0.1233	0.0003
	Cartographer	0.0193	0.0092	0.1349	0.0012

Using sequence ① for testing, the comparison of the relative trajectory error results obtained is shown in Figure 8.

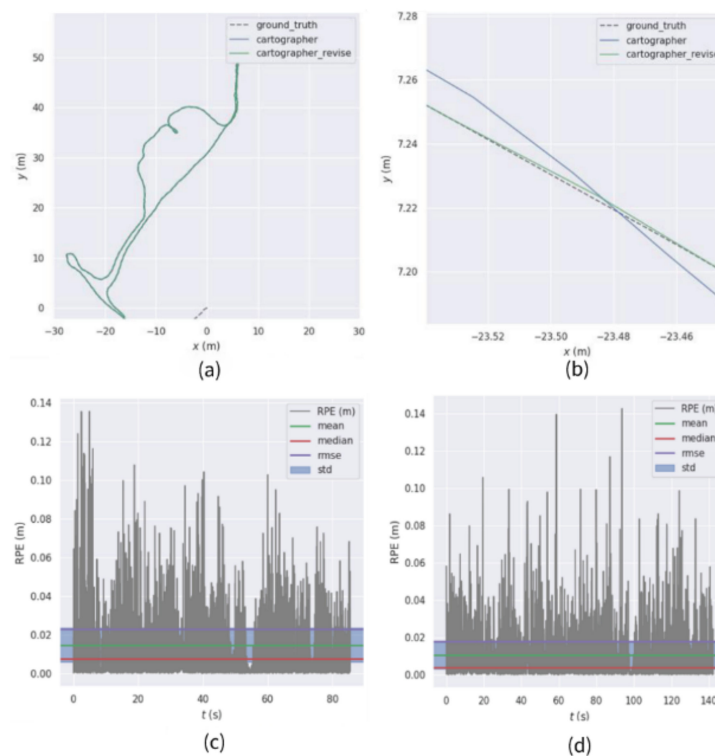


Figure 8. Comparison of relative trajectory errors of sequence ①. (a) Cartographer, improvement scheme, and real trajectory comparison (b) Local trajectory map (c) Absolute trajectory error of Cartographer (d) Absolute trajectory error of the improved scheme.

The obtained comparison of relative trajectory error results is shown in Figure 9 by using a sequence for testing.

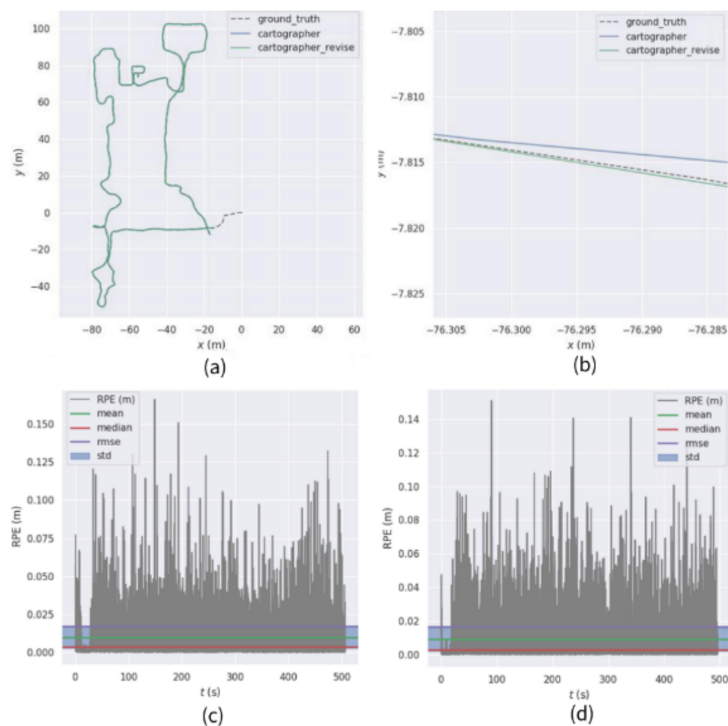


Figure 9. Comparison of relative trajectory errors of sequence ②. (a) Cartographer, improvement scheme, and real trajectory comparison (b) Local trajectory map (c) Absolute trajectory error of Cartographer (d) Absolute trajectory error of the improved scheme.

The obtained comparison of relative trajectory error results is shown in Figure 10 by using a sequence for testing.

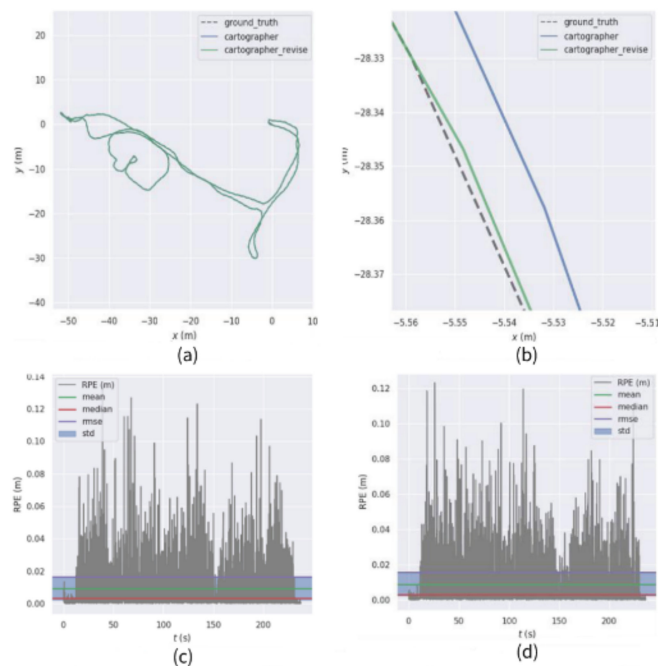


Figure 10. Comparison of relative trajectory errors of sequence ③. (a) Cartographer, improvement scheme, and real trajectory comparison (b) Local trajectory map (c) Absolute trajectory error of Cartographer (d) Absolute trajectory error of the improved scheme.

From the sequence ① test results, it can be seen that, from the RMSE index, the root-mean-square error of the Iao_ICP algorithm is 0.0179 m, and the root-mean-square error of the original Cartographer algorithm is 0.0230 m. Compared with the original Cartographer algorithm, the root-mean-square error of the Iao_ICP algorithm is reduced by 22.06%. The average error of the Iao_ICP algorithm is 0.0044 m smaller than that of the original Cartographer algorithm. The maximum absolute trajectory error of the original Cartographer algorithm is 0.1428 m in this sequence, and the maximum absolute trajectory error of the Iao_ICP algorithm is 0.1357 m. The minimum absolute trajectory error of the Cartographer original algorithm is 0.0024 m, and the minimum absolute trajectory error of the Iao_ICP algorithm is 0.0011 m. It can be seen from the above data that the Iao_ICP algorithm has a smaller relative trajectory error than the original Cartographer algorithm in sequence ①.

6. Physical Experiment Analysis

This experiment uses a small wheeled differential car as the mobile robot platform.

As shown in Figure 11, the platform configuration is as follows: wheeled robot, embedded development board, 16-line RS-LIDAR-16 scanner, IPMS-IG IMU. Among them, the wheeled robot is driven by four wheels and two motors. The embedded development board uses STM32f103 as the main controller, and it is also equipped with a motor driver module and an MPU6050 module. RS-LiDAR-16 adopts a hybrid solid-state LiDAR, which integrates 16 laser transceiver components. The measurement distance is up to 150 m, the measurement accuracy is within ± 2 cm, the number of output points is up to 300,000 points/s, the horizontal angle is 360° , the vertical measurement is 360° , and the angle is $\pm 15^\circ$. IMU integrates three-axis acceleration and angular velocity sensors, which can measure the real-time pose of the robot, and has the advantages of high precision, high frequency, low power consumption, and strong real-time performance. This experiment realizes the conversion of 3D LiDAR to 2D LiDAR by projecting the 16-line data of 3D LiDAR onto a fixed plane. Since the real motion trajectory of the robot cannot be accurately obtained in the real scene, this experiment judges and tests the cumulative error of the robot pose during the mapping process of the Iao_ICP algorithm according to the loopback effect. The movement of the robot is controlled by the handle in this experiment.

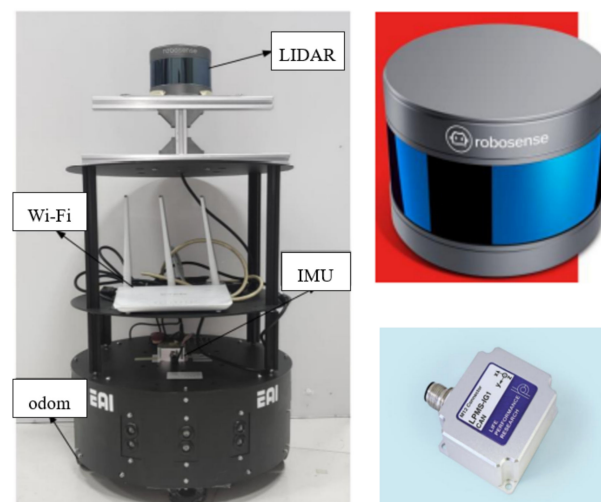


Figure 11. Mobile experiment platform.

The real environment is a rectangular hall corridor with a length of about 43 m, a width of about 51 m, and a building area of about 2193 m², as shown in Figure 12 above. It is easy to measure the actual size of the object and compare the data with the mapping accuracy of the test algorithm. Due to the cabinets, building supports, stair entrances, elevator entrances, and other objects in the environment have a strong structure, the effectiveness

and robustness of the algorithm for eliminating laser motion distortion and mapping accuracy can be tested in the above environment. There are also the following reasons: the test scene is relatively large, and there are long straight corridors, transparent glass, flowing crowds, and other factors in the environment that may easily interfere with the test of mapping. The smooth marble floor increases the accumulation of pose errors during the movement of the robot.



Figure 12. Experimental real scene.

To compare the mapping accuracy of the Iao_ICP algorithm and the original Cartographer algorithm, 10 highly structured objects were selected in the test scene for measurement and analysis. Figures 13 and 14 are the mapping effect of the original Cartographer algorithm and the mapping effect of the Iao_ICP algorithm. First, the actual size of the object is measured by a handheld laser rangefinder. The map measurements displayed in the rviz plugin for algorithmic mapping are measured. Finally, the relative error and absolute error of the two algorithms are calculated. The measurement data and error values of the above two algorithms are shown in Tables 2 and 3 below. Figure 15 is a comparison chart of the relative error of the two algorithms.

It can be seen from Figures 13 and 14 that the original Cartographer algorithm has a large pose error product in this experimental scene. Although a loop can be formed, the effect of eliminating local errors on the map is not good. The Iao_ICP algorithm removes motion distortion from most laser data by fusing wheel odometer and IMU information. At the same time, the laser scan data are compensated by estimating the speed of the robot and ICP algorithm. The Iao_ICP algorithm not only effectively removes motion distortion, but also eliminates the accumulation of pose errors caused by tire slippage during robot motion. Figure 14 shows that the map constructed by the Iao_ICP algorithm has no confusion, no burrs, and clear structural features. It can clearly express the surrounding environment information, and the map ghost is small. It can be seen that the mapping effect of the Iao_ICP algorithm is better than that of the original Cartographer algorithm. Combined with the error data analysis in Tables 2 and 3, and Figure 15, it can be seen that the average relative error of the Iao_ICP algorithm is much smaller than that of the original Cartographer algorithm, and the relative error is mostly concentrated below 1%. The error is stable, and there is no mutation.

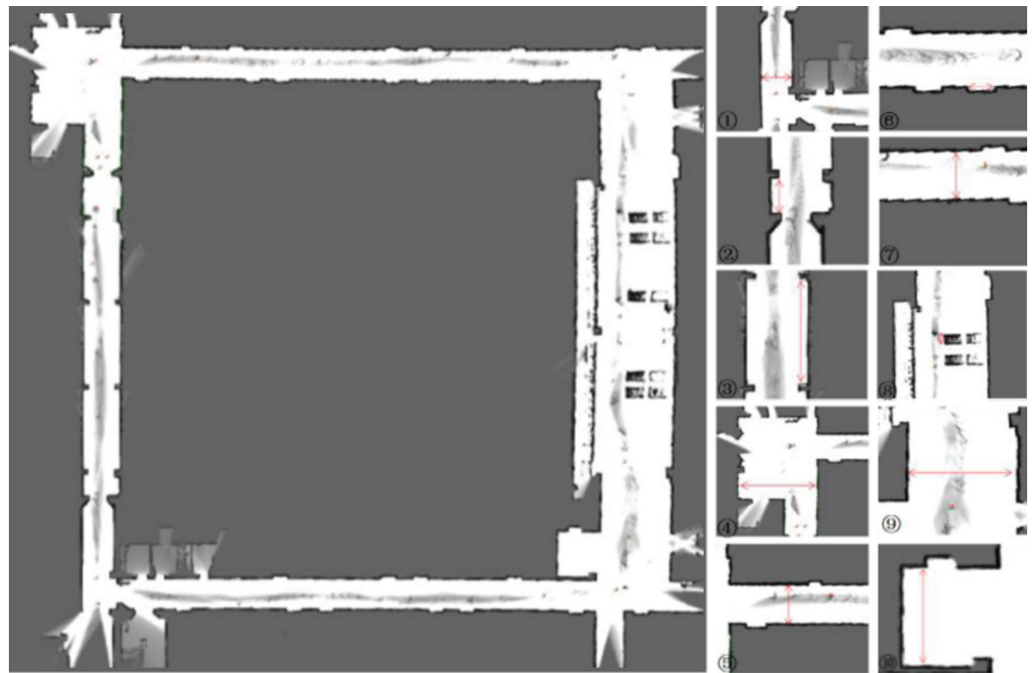


Figure 13. Mapping effect of Cartographer.

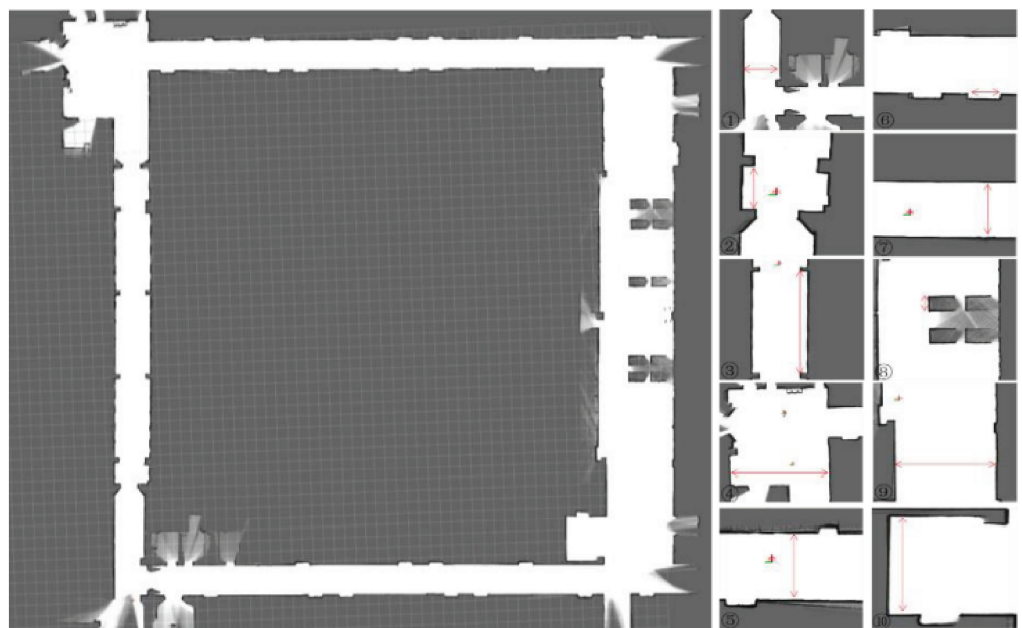


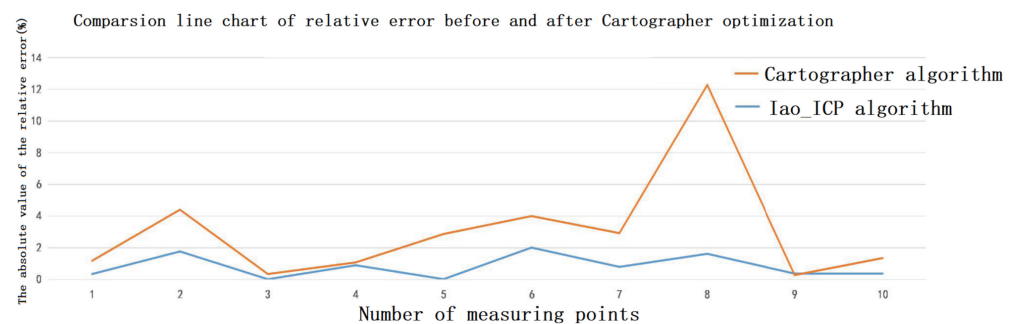
Figure 14. Mapping effect of Iao_ICP.

Table 2. Cartographer original algorithm mapping error table.

Measuring Point	Measured Value (cm)	Figure Measured Values (cm)	Absolute Error (cm)	Relative Error (%)
1	284.700	288.074	−3.374	−1.185107
2	195.000	186.400	8.600	4.410256
3	712.200	709.709	2.491	0.349761
4	812.000	803.200	8.800	1.083743
5	271.000	263.200	7.800	2.878228
6	136.300	130.840	5.460	4.005869
7	272.300	264.320	7.980	2.930591
8	76.500	85.895	−9.395	−12.281045
9	629.200	627.426	1.774	0.281945
10	402.700	397.230	5.470	1.358331

Table 3. Iao_ICP algorithm mapping error table.

Measuring Point	Measured Value (cm)	Figure Measured Values (cm)	Absolute Error (cm)	Relative Error (%)
1	284.700	283.700	1.000	0.351246
2	195.000	197.540	3.460	1.774358
3	712.200	712.363	-0.163	-0.022886
4	812.000	819.340	-7.340	-0.903940
5	271.000	270.928	0.072	0.026568
6	136.300	133.549	2.751	2.018341
7	272.300	270.116	2.184	0.802056
8	76.500	77.744	-1.244	-1.626143
9	629.200	626.829	2.371	0.376827
10	402.700	404.365	-1.665	-0.413459

**Figure 15.** Line chart of relative error comparison of two algorithms.

7. Conclusions

For the problem of removing laser motion distortion, in the case of wheel slippage and accumulated error, the traditional method of directly measuring displacement and angle information based on the wheel odometer, and the odometer angle data obtained by the encoder, will have a certain deviation. In addition, with the traditional method of directly measuring the angular velocity and linear acceleration based on the inertial navigation unit, and then integrating the displacement and angle information, due to the poor accuracy of the linear acceleration of the IMU, the local accuracy of the quadratic integration is still very poor. Therefore, the displacement data obtained will also have a certain deviation. The Iao_ICP algorithm proposed in this paper uses the linear interpolation method to obtain the pose of the LiDAR, which solves the alignment problem of discontinuous laser scan data. Data fused by IMU and odometer provide a better initial value for ICP. The estimated speed is introduced as the termination condition of the ICP method iteration to realize the compensation of the LiDAR data. The experiment uses a small wheeled mobile robot to collect data and compare and analyze results in a corridor environment to verify the original Cartographer algorithm and the Iao_ICP algorithm. Finally, the experimental data show that the algorithm proposed in this paper can effectively remove laser motion distortion, improve the accuracy of mapping, and reduce the cumulative error.

Author Contributions: Conceptualization, Q.W. and Q.M.; methodology, Q.M.; software, Z.Z.; validation, Q.W., C.L. and P.Z.; formal analysis, Z.Z.; investigation, Q.M.; resources, Q.W.; data curation, B.Z.; writing—original draft preparation, Q.M.; writing—review and editing, Z.Z.; visualization, C.L.; supervision, Y.L.; project administration, W.M.; funding acquisition, Y.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Science and Technology Project of the State Grid Corporation of China “Research and application of visual and auditory active perception and collaborative cognition technology for smart grid operation and maintenance scenarios” (Grant: 5600-202046347A-0-0-00).

Institutional Review Board Statement: This study was conducted in accordance with the Declaration of Helsinki, and approved by the Ethics Committee of Hangzhou Dianzi University (protocol code CE022108 22/02/26).

Informed Consent Statement: Informed consent will be obtained from all subjects involved in this study.

Data Availability Statement: Data will be made available upon request from the authors.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, P.; Shengze, W.; Michael, K. π -SLAM:LiDAR Smoothing and Mapping With Planes. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation, Xi'an, China, 19–22 November 2021; pp. 5751–5757.
2. Rui, H.; Yi, Z. High Adaptive LiDAR Simultaneous Localization and Mapping. *J. Univ. Electron. Sci. Technol. China* **2021**, *50*, 52–58.
3. Xin, L.L.; Xunyu, Z.; Xiafu, P.; Zhaohui, G.; Xungao, Z. Fast ICP-SLAM Method Based on Multi-resolution Search and Multi-density Point Cloud Matching. *Robot* **2020**, *42*, 583–594.
4. Hess, W.; Kohler, D.; Rapp, H.; Andor, D. Real-time loop closure in 2D LI-DAR SLAM. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016.
5. David, J.Y.; Haowei, Z.; Mona, G.; Hugues, T.; Timothy, D.B. Unsupervised Learning of LiDAR Features for Use in a Probabilistic Trajectory Estimator. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation, Xi'an, China, 19–22 November 2021; pp. 1154–1161.
6. Jo, J.H.; Moon, C.-b. Development of a Practical ICP Outlier Rejection Scheme for Graph-based SLAM Using a Range Finder. *Korean Soc. Precis. Eng.* **2019**, *20*, 1735–1745. [[CrossRef](#)]
7. Xue, H.; Fu, H.; Dai, B. IMU-aided high-frequency LiDAR odometry for autonomous driving. *Appl. Sci.* **2019**, *9*, 1506. [[CrossRef](#)]
8. Bezet, O.; Cherfaoui, V. Time error correction for range scanner data. In Proceedings of the International Conference on Information Fusion, New York, NY, USA, 10–13 July 2006.
9. Hong, S.; Ko, H.; Kim, J. VICP:velocity updating iterative closest point algorithm. In Proceedings of the IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–8 May 2010; pp. 1893–1898.
10. Zhang, J.; Singh, S. Low-drift and real-time LiDAR odometry and mapping. *Auton Robot.* **2017**, *41*, 401–406. [[CrossRef](#)]
11. Arun, K.S.; Huang, T.S.; Blostein, S.D. Least-squares fitting of two 3-D point sets. *IEEE Trans. Pattern Anal. Mach. Intell.* **1987**, *9*, 698–700. [[CrossRef](#)] [[PubMed](#)]
12. Xiaohui, J.; Wenfeng, X.; Jinyue, L.; Tiejun, L. Solving method of LiDAR odometry based on IMU. *J. Instrum.* **2021**, *42*, 39–48.
13. Xu, Z.; Zhi, W.; Can, C.; Zhixuan, W.; Renxin, H.; Jian, W. Design and Implementation of 2D LiDAR Positioning and Mapping System. *Opt. Technol.* **2019**, *45*, 596–600.
14. Yokozuka, M.; Koide, K.; Oishi, S.; Banno, A. LiTAMIN2: Ultra Light LiDAR-based SLAM using Geometric Approximation applied with KL-Divergence. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation, Xi'an, China, 19–22 November 2021; pp. 11619–11625.
15. Liwei, L.; Xukang, Z.; Xiuhua, L.; Zijun, Z. Research on Map Evaluation of 2D SLAM Algorithm for Low-Cost Mobile Robots. *Comput. Simul.* **2021**, *38*, 291–295.