applied sciences

Article

IMU-Aided High-Frequency Lidar Odometry for Autonomous Driving

Hanzhang Xue [1], Hao Fu [1] and Bin Dai [1,2,*]

[1] College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China; xuehanzhang13@nudt.edu.cn (H.X.); fuhao@nudt.edu.cn (H.F.)

[2] Unmanned Systems Research Center, National Innovation Institute of Defense Technology, Beijing 100071, China

* Correspondence: bindai.cs@gmail.com

Received: 14 March 2019; Accepted: 6 April 2019; Published: 11 April 2019

Abstract: For autonomous driving, it is important to obtain precise and high-frequency localization information. This paper proposes a novel method in which the Inertial Measurement Unit (IMU), wheel encoder, and lidar odometry are utilized together to estimate the ego-motion of an unmanned ground vehicle. The IMU is fused with the wheel encoder to obtain the motion prior, and it is involved in three levels of the lidar odometry: Firstly, we use the IMU information to rectify the intra-frame distortion of the lidar scan, which is caused by the vehicle's own movement; secondly, the IMU provides a better initial guess for the lidar odometry; and thirdly, the IMU is fused with the lidar odometry in an Extended Kalman filter framework. In addition, an efficient method for hand–eye calibration between the IMU and the lidar is proposed. To evaluate the performance of our method, extensive experiments are performed and our system can output stable, accurate, and high-frequency localization results in diverse environment without any prior information.

Keywords: ego-motion estimation; hand-eye calibration; IMU; lidar odometry; sensor fusion

1. Introduction

Precise and high-frequency localization is one of the key problems for autonomous vehicles. In recent years, the fusion of Global Navigation Satellite System (GNSS) and Inertial Measurement Unit (IMU) has been the most popular localization method. However, the GNSS/IMU system will fail in some environments where GNSS signals suffer from satellite blockage or multipath propagation [1], such when an autonomous vehicle is driving in a tunnel, in the mountains, or in an environment with electromagnetic interference. Thus, an alternative localization method is necessary when the GNSS signal is unavailable or is of poor quality.

Currently, most autonomous vehicles are equipped with light detection and ranging (lidar) device, which is a promising sensor that could accurately calculate the range measurements of the surroundings. Some recent navigation approaches have begun to use lidar as a complementary sensor for the GNSS/IMU localization system. A typical lidar odometry algorithm could roughly be divided into three steps: a pre-processing step, which tries to compensate the intra-frame distortion caused by the vehicle's own movement; an intermediate step, which is the main body of the lidar odometry algorithm, and a last step, which outputs the localization result.

As the lidar mostly spins at 10 Hz, its output frequency is thus less than 10 Hz, which might not be fast enough to meet the needs of other modules, like planning or control. To remedy this, some approaches try to combine the lidar with other sensors, such as IMU. In these approaches, the IMU and the lidar odometry are usually combined in an Extended Kalman Filter (EKF) framework, or by using a factor graph representation [2–7] (as shown in the left of Figure 1). In this paper, we claim that IMU information can not only be fused with the lidar odometry at the output level, but it can also aid lidar odometry in the pre-processing level and the intermediate level (as shown in the right of Figure 1).

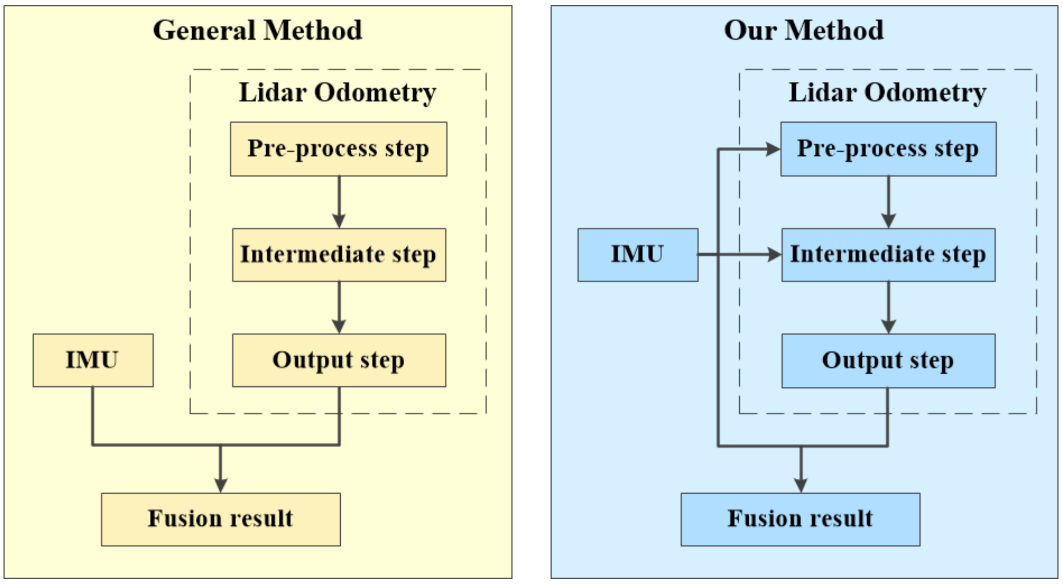


Figure 1. A typical lidar odometry algorithm can be roughly divided into three steps: a pre-processing step, an intermediate step, and an output step. Most existing lidar/Inertial Measurement Unit (IMU) fusion approaches try to fuse lidar odometry and the IMU in the output level [2–7]. In this paper, we try to utilize the IMU information in all the three steps.

In the pre-processing step, as the vehicle itself is moving during the lidar spin, the generated point clouds will be distorted. This intra-frame distortion is usually ignored. In this paper, we show that the high-frequency IMU information could naturally help correct this distortion, as long as the IMU and lidar are well-calibrated. We also claim that **the problem of intra-frame correction is in fact equivalent to inter-frame registration**, and propose an interesting two-step correction procedure which does not rely on the IMU. In the intermediate step, the IMU could also provide a good initial guess for the lidar odometry algorithm. Past works usually use the registration result from the previous two scans as the initial guess. We will do experiments to compare these two approaches. In the output step, we try to combine the lidar odometry with the IMU in the EKF framework.

In summary, we make the following contributions in this paper:

- We propose an efficient method for hand–eye calibration between the IMU and the lidar, which is a pre-requisite for making good use of the IMU.
- In the pre-processing step of the lidar odometry algorithm, we emphasize that the intra-frame correction is equivalent to inter-frame registration, and propose an efficient method for utilizing the IMU information to rectify the distorted point clouds.
- In the intermediate step, we use IMU to obtain the initial guess for the lidar odometry algorithm, and compare this approach with previous methods which usually use the registration results from the previous two scans as the initial guess.
- By combining the lidar odometry with the IMU information, we obtain a complete localization system which can output stable, accurate, and high-frequency localization results at a frequency of around 40 Hz.

The rest of this paper is organized as follows. In Section 2, we introduce some related works. Section 3.1 provides an overview of the proposed approach, and introduces the coordinate systems and notations in our approach. The details of the proposed approach are presented from Sections 3.2–3.6. The experimental results and comparison with state-of-the-art approaches are presented in Section 4. Finally, the conclusions are summarized in Section 5.

2. Related Works

Lidar odometry has been a hot topic in recent years. Lidar odometry is, in essence, equivalent to scan matching, which tries to estimate the relative transform between two frames of point clouds. The scan matching methods can be classified into three categories: point-based, mathematical property-based, and feature-based. The Iterative Closest Point (ICP) algorithm [8] and its variants [9] are the most popular point-based methods which estimate the relative transform through finding the nearest points between the two scans. For the mathematical property-based method, such as the Normal Distribution Transform (NDT) algorithm and its variants (D2D-NDT [10] and P2D-NDT [11]), the observed points are modeled as a set of Gaussian probability distribution. For the feature-based method, there have been various features applied, such as the line segment feature [12], corner feature [13], and grid feature [14]. Zhang et al. proposed a method called Lidar Odometry And Mapping (LOAM) [15,16], which extracts feature points located on sharp edges or planar surfaces, and matches these feature points using either the point-to-line distance or the point-to-plane distance.

All these lidar odometry algorithms can only output localization results at a frequency of less than 10 Hz, which might not be fast enough. Some recent approaches have started to fuse the lidar odometry with the IMU using either the Kalman Filter framework [2–5,17,18] or the factor graph representation [6,7].

In [2], the authors proposed a localization system which combines a 2D Simultaneous Localization and Mapping (SLAM) subsystem with a 3D navigation subsystem based on IMU. This system can estimate the accurate state of an unmanned ground vehicle (UGV), but the usage of 2D SLAM is not suitable for the off-road environment. In [3], a hybrid lidar odometry algorithm was proposed which combines the feature-based scan matching method with the ICP-based scan matching method, and the lidar odometry result was then fused with the IMU in an EKF framework. In [6], a bunch of IMU measurements were considered as a relative constraint using the pre-integration theory, and the results of lidar odometry were fused with IMU measurements in a factor graph framework. However, in all these previous works, the IMU information was only fused with the lidar odometry in the result level; whilst in this paper, we try to utilize the IMU information to aid the whole process of the lidar odometry algorithm.

3. The Proposed Approach

3.1. Coordinate Systems and Notations

The aim of this work is to accurately estimate the pose of the vehicle. Three coordinate systems are used in our approach, which are defined as follows:

- World Coordinate System $\{W\}$. In $\{W\}$, we define the coordinates of a point by latitude, longitude, and altitude, the x-axis points to the east, the y-axis points to the north, and the z-axis points to the sky, following the right-hand rule. This coordinate system is defined in a global scale and will never change.
- Body Coordinate System $\{B\}$. This coordinate is fixed with the IMU, which is installed in the center of two rear wheels of the vehicle. The x-axis points to the right, the y-axis points to the forward, and the z-axis points to the upward, following the right-hand rule.
- Lidar coordinate system $\{L\}$. The lidar is mounted on top of the vehicle, and the origin of $\{L\}$ is located at the center of the lidar. The three axes of this coordinate system are the same as in $\{B\}$.

Figure 2 shows the coordinate system $\{W\}$ and $\{B\}$ in the 2D plane and the relationship between them. The body coordinate system $\{B\}$ changes with the movement of the vehicle. $\boldsymbol{T}_{\mathrm{k}}^{\mathrm{WB}}$ represents the transformation between the coordinate system $\{W\}$ and $\{B\}$ at time t_{k}, which can also denote the pose of the vehicle at time t_{k}. On the contrary, $\boldsymbol{T}_{\mathrm{k}}^{\mathrm{BW}}$ transforms the coordinate system $\{B\}$ to $\{W\}$ and $\boldsymbol{T}_{\mathrm{k}}^{\mathrm{BW}} = \left(\boldsymbol{T}_{\mathrm{k}}^{\mathrm{WB}}\right)^{-1}$. Moreover, $\triangle\boldsymbol{T}_{\mathrm{k-1,k}}^{\mathrm{B}}$, which could be obtained from the IMU, represents the relative transformation of the vehicle between time t_{k-1} and t_k. The relationship between $\triangle T^{B}_{k-1,k}$ and T^{WB}_{k} can be expressed as:

$$\triangle \boldsymbol{T}^{B}_{k-1,k} = \left(\boldsymbol{T}^{WB}_{k-1}\right)^{-1} \boldsymbol{T}^{WB}_{k}. \tag{1}$$

In addition, the relationship between the coordinate system $\{B\}$ and $\{L\}$ is shown in Figure 3.

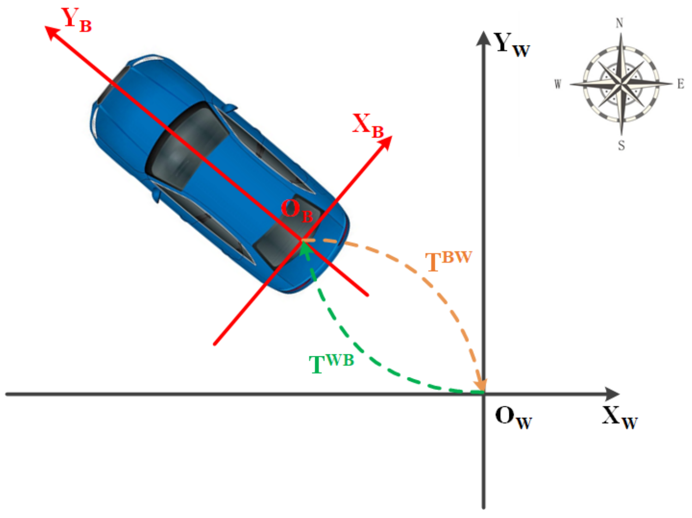


Figure 2. The relative relationship between the world coordinate system $\{W\}$ and the body coordinate system $\{B\}$ in the 2D plane, where the black axis indicates $\{W\}$ and the red axis indicates $\{B\}$.

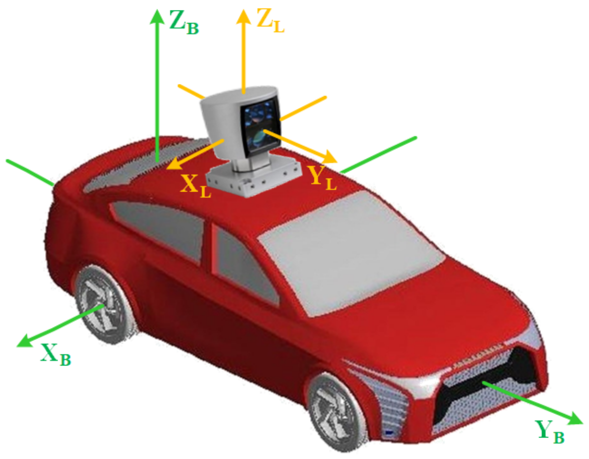


Figure 3. The relationship between the body coordinate system $\{L\}$ and the lidar coordinate system $\{B\}$, where the yellow axis indicates $\{L\}$ and the green axis indicates $\{B\}$.

We use $\boldsymbol{T}^{WL}_{k}$ to denote the pose of the lidar at time t_k. $\triangle \boldsymbol{T}^{L}_{k-1,k}$ represents the relative transformation of the lidar between time t_{k-1} and t_k. The transformation $\boldsymbol{T}^{BL}$, which transforms the coordinate system $\{B\}$ to $\{L\}$, can be obtained from the hand–eye calibration.

The transformation $\boldsymbol{T}$ can be expressed by:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{p} \\ \boldsymbol{0}^T & 1 \end{bmatrix}, \tag{2}$$

where $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$ represents the rotational matrix and $\boldsymbol{p} \in \mathbb{R}^3$ is the translational vector.

In *Lie Group* [19], $\boldsymbol{T}$ belongs to the *Special Euclidean group* $\mathrm{SE}(3)$ and the rotational matrix $\boldsymbol{R}$ belongs to the *Special Orthogonal group* $\mathrm{SO}(3)$. Correspondingly, the $\mathfrak{se}(3)$ and $\mathfrak{so}(3)$ are two kinds of *Lie Algebra*. $\boldsymbol{\theta} \in \mathfrak{so}(3)$ indicates the rotation angle which can be expressed as $\boldsymbol{\theta} = \theta \boldsymbol{n}$, where θ represents the magnitude of the rotation angle and $\boldsymbol{n}$ denotes the rotation axis. The pose can thus be represented by $\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\theta} & \boldsymbol{p} \end{bmatrix}^T \in \mathfrak{se}(3)$.

The exponential map and logarithmic map can be used to derive the relationship between the *Lie Group* and the *Lie Algebra*. For example, $\boldsymbol{T}$ can be represented by the exponential map of the $\boldsymbol{\xi}$, namely, $\boldsymbol{T} = \exp(\boldsymbol{\xi}^{\wedge})$. Conversely, $\boldsymbol{\xi}$ can be represented by the logarithmic map of the $\boldsymbol{T}$, namely, $\boldsymbol{\xi} = \log(\boldsymbol{T})^{\vee}$.

The exponential map between $\boldsymbol{\theta}$ and $\boldsymbol{R}$ can be formulated by using the Rodriguez formula:

$$\boldsymbol{R} = \cos\theta \cdot \boldsymbol{I} + (1 - \cos\theta) \cdot \boldsymbol{n} \cdot \boldsymbol{n}^T + \sin\theta \cdot \boldsymbol{n}^{\wedge}. \tag{3}$$

The operator $\wedge$ maps a vector to a skew-symmetric matrix as Equation (4), and the operator $\vee$ maps a skew symmetric matrix back to a vector.

$$\boldsymbol{\theta}^{\wedge} = \begin{bmatrix} \theta^x \\ \theta^y \\ \theta^z \end{bmatrix}^{\wedge} = \begin{bmatrix} 0 & -\theta^z & \theta^y \\ \theta^z & 0 & -\theta^x \\ -\theta^y & \theta^x & 0 \end{bmatrix}. \tag{4}$$

The notation of the data used in our system are specified as follows. Each type of data used in our system is composed of the timestamp and the data itself. The timestamp is defined as the time when the data is generated. t^{B} denotes the timestamp of the IMU measurement and t^{L} represents the timestamp of the lidar point clouds. $f_{\mathrm{k}}^{\mathrm{L}}$ indicates all points generated from the lidar during $[t_{\mathrm{k-1}}^{\mathrm{L}}, t_{\mathrm{k}}^{\mathrm{L}}]$. In addition, the i-th point generated by the b-th laser beam is denoted as $\boldsymbol{X}_{(\mathrm{i,b,k})}^{\mathrm{L}}$ in the lidar coordinate system $\{L\}$. The number of points received per laser beam is denoted as S_b and $i \in [1, S_b]$.

3.2. Motion Prior from IMU and Wheel Encoder

It is well-known that the accelerometer in the IMU is affected by the gravity, and it needs to be integrated twice to obtain the translational vector [20]. In contrast, the gyro in the IMU directly measures the angular velocity, and it only needs to be integrated once to obtain the rotation angle. It is reasonable to believe that the angular output from the IMU is much more accurate than the translational measurement.

For the ground vehicle, besides IMU, it is also equipped with the wheel encoder. The wheel encoder directly measures the rotation of the wheel, and thus provides a much more accurate translational measurement.

In this paper, we try to combine the angular output from the IMU and the translational output from the wheel encoder to obtain the motion prior.

The motion model of the vehicle can be simplified as Figure 4. We mount the IMU at the center of two rear wheels, and mount two wheel encoders on the left and right rear wheels, respectively. The black car denotes the position of the vehicle at the previous moment $t_{\mathrm{k-1}}^{\mathrm{B}}$, and the blue car indicates its position at the current time $t_{\mathrm{k}}^{\mathrm{B}}$. $\triangle sl$ and $\triangle sr$ represent the traveled distances of the left rear wheel and the right rear wheel during the short period of time.

Firstly, we compute the rotation matrix $\boldsymbol{R}_{\mathrm{k}}^{\mathrm{WB}}$ by triaxial angular velocities of the IMU. We introduce the following kinematic model [21]:

$$\frac{\mathrm{d}\boldsymbol{R}_{\tau}^{\mathrm{BW}}}{\mathrm{d}t} = \boldsymbol{R}_{\tau}^{\mathrm{BW}} \begin{bmatrix} \omega_{\tau}^{x} \\ \omega_{\tau}^{y} \\ \omega_{\tau}^{z} \end{bmatrix}^{\wedge}. \tag{5}$$

In Equation (5), $\boldsymbol{R}_{\tau}^{\mathrm{BW}}$ indicates the rotation matrix from the coordinate system $\{B\}$ to $\{W\}$. ω_{τ}^{x}, ω_{τ}^{y}, and ω_{τ}^{z} represent the instantaneous angular velocities of the IMU.

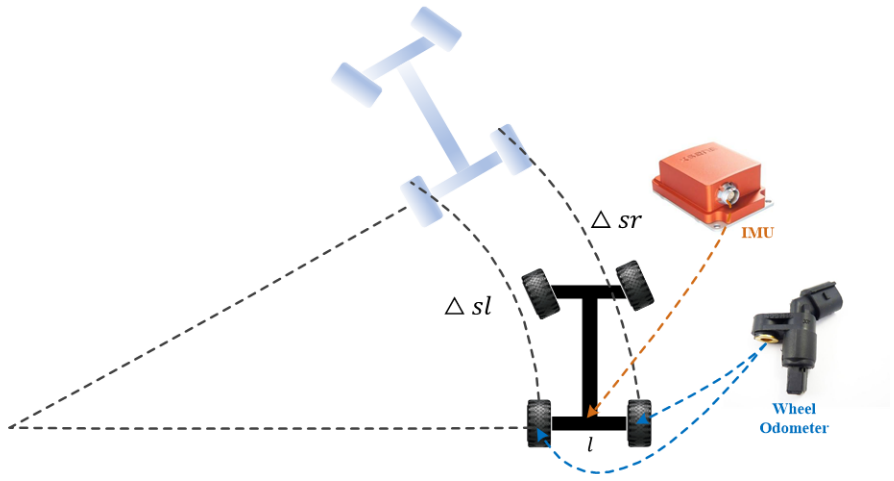


Figure 4. Schematic diagram of the simplified vehicle motion model. The black car represents the previous position of the vehicle, and the blue car indicates its current position. During a time period, the left rear wheel travels a distance of $\triangle sl$, and the right rear wheel travels $\triangle sr$.

By assuming the angular velocity of the IMU constant during $[t^{\mathrm{B}}_{\mathrm{k}-1}, t^{\mathrm{B}}_{\mathrm{k}}]$, we integrate Equation (5) from $t^{\mathrm{B}}_{\mathrm{k}-1}$ to $t^{\mathrm{B}}_{\mathrm{k}}$ and solve the differential equation. The rotation matrix $\boldsymbol{R}^{\mathrm{BW}}_{\mathrm{k}}$ can be obtained by:

$$\boldsymbol{R}^{\mathrm{BW}}_{\mathrm{k}} = \boldsymbol{R}^{\mathrm{BW}}_{\mathrm{k}-1}\,\mathrm{Exp}\left((\boldsymbol{\omega}_{\mathrm{k}} \triangle t)^{\wedge}\right). \tag{6}$$

To compute the translational component $\boldsymbol{p}^{\mathrm{WB}}_{\mathrm{k}}$ of the pose $\boldsymbol{T}^{\mathrm{WB}}_{\mathrm{k}}$, we approximate the traveled distance $\triangle s$ of the vehicle to be the average of the two rear wheels. $\triangle s$ indicates the relative displacement of the vehicle during $[t^{\mathrm{B}}_{\mathrm{k}-1}, t^{\mathrm{B}}_{\mathrm{k}}]$ in the body coordinate system $\{B\}$ and the instantaneous speed of the vehicle can be denoted as $v = \triangle s/(t^{\mathrm{B}}_{\mathrm{k}} - t^{\mathrm{B}}_{\mathrm{k}-1})$. We assume the velocity of the vehicle to be constant during $[t^{\mathrm{B}}_{\mathrm{k}-1}, t^{\mathrm{B}}_{\mathrm{k}}]$, where the wheel encoder always records the distance traveled along the heading direction of the vehicle. Thus, the instantaneous velocity vector can be denoted as $\boldsymbol{v}^{\mathrm{B}}_{\mathrm{k}} = [v\ \ 0\ \ 0]^{T}$. The relationship between the velocity vector $\boldsymbol{v}^{\mathrm{W}}_{\mathrm{k}}$ in the coordinate system $\{W\}$ and the velocity vector $\boldsymbol{v}^{\mathrm{B}}_{\mathrm{k}}$ in the coordinate system $\{B\}$ is:

$$\boldsymbol{v}^{\mathrm{W}}_{\mathrm{k}} = \boldsymbol{R}^{\mathrm{WB}}_{\mathrm{k}}\,\boldsymbol{v}^{\mathrm{B}}_{\mathrm{k}}. \tag{7}$$

Using $\boldsymbol{p}^{\mathrm{WB}}_{\mathrm{k}-1}$, $\boldsymbol{R}^{\mathrm{WB}}_{\mathrm{k}}$, and $\boldsymbol{v}^{\mathrm{W}}_{\mathrm{k}}$, the translational component $\boldsymbol{p}^{\mathrm{WB}}_{\mathrm{k}}$ can be computed as:

$$\begin{aligned} \boldsymbol{p}^{\mathrm{WB}}_{\mathrm{k}} &= \boldsymbol{p}^{\mathrm{WB}}_{\mathrm{k}-1} + \boldsymbol{v}^{\mathrm{W}}_{\mathrm{k}}\ (t_{\mathrm{k}} - t_{\mathrm{k}-1}) \\ &= \boldsymbol{p}^{\mathrm{WB}}_{\mathrm{k}-1} + \boldsymbol{R}^{\mathrm{WB}}_{\mathrm{k}} \begin{bmatrix} \triangle s & 0 & 0 \end{bmatrix}^{T}. \end{aligned} \tag{8}$$

Since both the IMU and the wheel encoder output data at a frequency of 100 Hz, we use two separate buffers to cache these two kinds of data. These two sources of information are then fused by Equations (6) and (8). This fusion process only involves a few operators, and usually takes less than 1 millisecond. Therefore, the fusion output, which we term as the motion prior data, is also generated at 100 Hz.

3.3. Pre-Processing Step 1: Lidar-IMU Hand–Eye Calibration

As described before, the lidar point clouds are represented in the lidar coordinate system $\{L\}$. Taking the lidar point clouds of two frames $f^{\mathrm{L}}_{\mathrm{k}-1}$ and $f^{\mathrm{L}}_{\mathrm{k}}$ as inputs, we can calculate $\triangle\boldsymbol{T}^{\mathrm{L}}_{\mathrm{k}-1,\mathrm{k}}$ by the lidar odometry algorithm. $\triangle\boldsymbol{T}^{\mathrm{L}}_{\mathrm{k}-1,\mathrm{k}}$ indicates the relative transformation in the coordinate system $\{L\}$. However, the $\triangle\boldsymbol{T}^{\mathrm{B}}_{\mathrm{k}-1,\mathrm{k}}$ obtained from the IMU is expressed in the body coordinate system $\{B\}$.

Therefore, in order to utilize the IMU information, we need to establish the relationship between $\{L\}$ and $\{B\}$ first.

The calculation of the relative transformation $\boldsymbol{T}^{\mathrm{BL}}$ between $\{L\}$ and $\{B\}$ is a typical hand–eye calibration problem. Based on $\boldsymbol{T}^{\mathrm{BL}}$, we can transform a point from the lidar coordinate system $\{L\}$ to the body coordinate system $\{B\}$:

$$\boldsymbol{X}^{\mathrm{B}}_{(\mathrm{i,b,k})} = \boldsymbol{T}^{\mathrm{BL}} \boldsymbol{X}^{\mathrm{L}}_{(\mathrm{i,b,k})} \,. \tag{9}$$

According to [22], we can formulate the hand–eye calibration problem between the coordinate system $\{L\}$ and $\{B\}$ as:

$$\triangle \boldsymbol{T}^{\mathrm{B}}_{\mathrm{k,k-1}}\, \boldsymbol{T}^{\mathrm{BL}} = \boldsymbol{T}^{\mathrm{BL}}\, \triangle \boldsymbol{T}^{\mathrm{L}}_{\mathrm{k,k-1}} \,. \tag{10}$$

$\triangle \boldsymbol{T}^{\mathrm{B}}_{\mathrm{k-1,k}}$ and $\triangle \boldsymbol{T}^{\mathrm{L}}_{\mathrm{k-1,k}}$ are inputs of Equation (10). Let the timestamps of frames $f^{\mathrm{L}}_{\mathrm{k}}$ and $f^{\mathrm{L}}_{\mathrm{k-1}}$ be $\tau^{\mathrm{L}}_{\mathrm{k}}$ and $\tau^{\mathrm{L}}_{\mathrm{k-1}}$, respectively. We firstly find the nearest timestamps $\tau^{\mathrm{B}}_{\mathrm{k}}$ and $\tau^{\mathrm{B}}_{\mathrm{k-1}}$ from the IMU, then $\boldsymbol{T}^{\mathrm{WB}}_{\mathrm{k}}$ and $\boldsymbol{T}^{\mathrm{WB}}_{\mathrm{k-1}}$ can be obtained. The relative transformation $\triangle \boldsymbol{T}^{\mathrm{B}}_{\mathrm{k-1,k}}$ can be computed by Equation (1). We then use $\triangle \boldsymbol{T}^{\mathrm{B}}_{\mathrm{k-1,k}}$ as the initial guess and $\triangle \boldsymbol{T}^{\mathrm{L}}_{\mathrm{k-1,k}}$ can be obtained from the lidar odometry.

Equation (10) can be solved by calculating the rotational and translational component separately. The rotational component is:

$$\triangle \boldsymbol{R}^{\mathrm{B}}_{\mathrm{k,k-1}}\, \boldsymbol{R}^{\mathrm{BL}} = \boldsymbol{R}^{\mathrm{BL}}\, \triangle \boldsymbol{R}^{\mathrm{L}}_{\mathrm{k,k-1}} \,, \tag{11}$$

and the translational component is:

$$\triangle \boldsymbol{R}^{\mathrm{B}}_{\mathrm{k,k-1}}\, \boldsymbol{p}^{\mathrm{BL}} + \triangle \boldsymbol{p}^{\mathrm{B}}_{\mathrm{k,k-1}} = \boldsymbol{R}^{\mathrm{BL}}\, \triangle \boldsymbol{p}^{\mathrm{L}}_{\mathrm{k,k-1}} + \boldsymbol{p}^{\mathrm{BL}} \,. \tag{12}$$

There are many methods to solve Equations (11) and (12). According to [23], the method proposed by Tsai et al. [24] exhibits the best performance. We therefore chose to use this method.

In our approach, multiple pairs of transformation $\triangle \boldsymbol{T}^{\mathrm{L}}_{\mathrm{i,i-1}}$ and $\triangle \boldsymbol{T}^{\mathrm{B}}_{\mathrm{i,i-1}}$ are needed. To reduce the error of hand–eye calibration [25,26], the following requirements need to be met:

- To avoid the intra-frame distortion caused by the vehicle's own movement, the frames should be chosen when the vehicle is stationary.
- The relative rotation angle of the two frames needs to be greater than 90 degrees, and the relative translation distance should be small.
- The rotational axes for these multiple transformation pairs should not be parallel.

After completing the hand–eye calibration, each point of the frame $f^{\mathrm{L}}_{\mathrm{k}}$ can be transformed by Equation (9) and be represented in the coordinate system $\{B\}$.

3.4. Pre-Processing Step 2: Intra-Frame Correction

As mentioned in Section 3.1, it usually takes 100 ms for the lidar to spin a full circle. During this time, the vehicle also keeps moving. Some of these data are received at the previous moment $t^{\mathrm{L}}_{\mathrm{k}} - t_j$ ($t_j \in (0, 100)$ (ms)) and their coordinates are represented in the coordinate system corresponding to the moment $t^{\mathrm{L}}_{\mathrm{k}} - t_j$. These points can be denoted as $\boldsymbol{X}^{\mathrm{B}}_{(\mathrm{i,b,k-j})}$. If we want to represent these points in the coordinate system corresponding to time $t^{\mathrm{L}}_{\mathrm{k}}$, we need to transform the point $\boldsymbol{X}^{\mathrm{B}}_{(\mathrm{i,b,k-j})}$ to $\boldsymbol{X}^{\mathrm{B}}_{(\mathrm{i,b,k})}$. This transformation is usually called "correction" or "rectification" of the lidar scan.

There are two common ways to rectify the distorted point clouds. One utilizes the information of IMU [16,27]. We make some improvements in this method which is described in Section 3.4.1. The other one uses the previous registration results [15], which is presented in Section 3.4.2. Furthermore, we also propose an interesting two-step correction procedure which does not rely on the IMU in Section 3.4.3.

3.4.1. Rectification with the Aid of IMU

In this method, we try to use IMU to rectify the raw points. Let the pose at time $t^{\mathrm{L}}_{\mathrm{k}}$ and $t^{\mathrm{L}}_{\mathrm{k-1}}$ be $\boldsymbol{T}^{\mathrm{WB}}_{\mathrm{q}}$ and $\boldsymbol{T}^{\mathrm{WB}}_{\mathrm{p}}$. Since the IMU data is generated at a frequency of 100 Hz and the lidar point clouds are generated at a frequency of 10 Hz, there are always nine frames of IMU data between the pose $\boldsymbol{T}_{\mathrm{p}}^{\mathrm{WB}}$ and $\boldsymbol{T}_{\mathrm{q}}^{\mathrm{WB}}$. Therefore, ten relative transformations $\triangle\boldsymbol{T}_{\mathrm{m-1,m}}^{\mathrm{B}}$ $(m \in (p,q])$ can be obtained by Equation (1) and each $\triangle\boldsymbol{T}_{\mathrm{m-1,m}}^{\mathrm{B}}$ represents the relative transformation during $[t_{\mathrm{m-1}}^{\mathrm{B}}, t_{\mathrm{m}}^{\mathrm{B}}]$, where $t_{\mathrm{m}}^{\mathrm{B}} - t_{\mathrm{m-1}}^{\mathrm{B}}$ equals to 10 ms.

Since the rotational velocity of the lidar is constant during $[t_{\mathrm{k-1}}^{\mathrm{L}}, t_{\mathrm{k}}^{\mathrm{L}}]$, the rotation angle of the lidar per 10 ms is also fixed. We can separate all points of a frame $f_{\mathrm{k}}^{\mathrm{L}}$ into ten sets, and the ID of points in each set can be denoted as $\mathrm{i} \in [\frac{m-1-p}{q-p} \times S_b, \frac{m-p}{q-p} \times S_b]$. Let the total number of points in each set be S_e, which is equal to $S_b/10$. We further assume that the motion of the vehicle is constant during $[t_{\mathrm{m-1}}^{\mathrm{B}}, t_{\mathrm{m}}^{\mathrm{B}}]$. Each set of points now correspond to a unique relative transformation $\triangle\boldsymbol{T}_{\mathrm{m,m-1}}^{\mathrm{B}}$ which can also be denoted as $exp\left(\triangle\boldsymbol{\xi}_{\mathrm{m-1,m}}^{\wedge}\right)$.

We linearly interpolated $exp\left(\triangle\boldsymbol{\xi}_{\mathrm{m-1,m}}^{\wedge}\right)$ to transform point $\boldsymbol{X}_{\mathrm{(i,b,k-j)}}^{\mathrm{B}}$ to $\boldsymbol{X}_{\mathrm{(i,b,m-1)}}^{\mathrm{B}}$. Then, $\boldsymbol{X}_{\mathrm{(i,b,m-1)}}^{\mathrm{B}}$ will be transformed to $\boldsymbol{X}_{\mathrm{(i,b,k)}}^{\mathrm{B}}$ by $\triangle\boldsymbol{T}_{\mathrm{q,m-1}}^{\mathrm{B}}$. The whole rectification process can be denoted as:

$$\hat{\boldsymbol{X}}_{\mathrm{(i,b,k)}}^{\mathrm{B}} = \triangle\boldsymbol{T}_{\mathrm{q,m-1}}^{\mathrm{B}} \cdot \exp\left(\triangle\boldsymbol{\xi}_{\mathrm{m-1,m}} \cdot \frac{i\%S_e}{S_e}\right)^{\wedge} \cdot \boldsymbol{X}_{\mathrm{(i,b,k-j)}}^{\mathrm{B}}, \tag{13}$$

where $A\%B$ is used to compute the remainder when A is divided by B, and $\hat{\boldsymbol{X}}_{\mathrm{(i,b,k)}}^{\mathrm{B}}$ represents the undistorted point after rectification.

3.4.2. Rectification by Previous Registration Results

This approach assumes that the motion of the vehicle is constant during $\left[t_{\mathrm{k-2}}^{\mathrm{L}}, t_{\mathrm{k}}^{\mathrm{L}}\right]$. Thus, it can assume that the relative transformation $\triangle\boldsymbol{T}_{\mathrm{k-1,k-2}}^{\mathrm{B}}$ between frame $f_{\mathrm{k-1}}^{\mathrm{L}}$ and $f_{\mathrm{k-2}}^{\mathrm{L}}$ is the same as the relative transformation $\triangle\boldsymbol{T}_{\mathrm{k,k-1}}^{\mathrm{B}}$ between frame $f_{\mathrm{k}}^{\mathrm{L}}$ and $f_{\mathrm{k-1}}^{\mathrm{L}}$. Therefore, we can linearly interpolate the relative transformation $exp\left(\triangle\boldsymbol{\xi}_{\mathrm{k-1,k-2}}^{\wedge}\right)$ to rectify the raw distorted points in frame $f_{\mathrm{k}}^{\mathrm{L}}$ as:

$$\hat{\boldsymbol{X}}_{\mathrm{(i,b,k)}}^{\mathrm{B}} = \exp\left(\triangle\boldsymbol{\xi}_{\mathrm{k-1,k-2}} \cdot \frac{i}{S_b}\right)^{\wedge} \cdot \boldsymbol{X}_{\mathrm{(i,b,k-j)}}^{\mathrm{B}}. \tag{14}$$

3.4.3. Rectification by Registering Two Distorted Point Clouds

In this approach, we note that intra-frame correction is in fact quite related to inter-frame registration. As the task of intra-frame correction is to estimate the lidar motion between 0 ms and 100 ms, inter-frame registration tries to estimate the motion between 100 ms and 200 ms. Both frame *a* (corresponding to the lidar date generated between 0 ms and 100 ms) and frame *b* (generated between 100 ms and 200 ms) are distorted point clouds. It is reasonable to assume that the degree of distortion of these two frames are quite similar. We could thus directly register these two frames without any intra-frame correction first, and then use the registration result to correct each frame.

Let the relative transformation obtained from registering two distorted point clouds be $\triangle\widetilde{\boldsymbol{T}}_{\mathrm{k,k-1}}^{\mathrm{B}}$. We assumed that $\triangle\widetilde{\boldsymbol{T}}_{\mathrm{k,k-1}}^{\mathrm{B}}$ can approximate the lidar motion between time $\left[t_{\mathrm{k}}^{\mathrm{L}}, t_{\mathrm{k-1}}^{\mathrm{L}}\right]$ and linearly interpolate $exp\left(\triangle\widetilde{\boldsymbol{\xi}}_{\mathrm{k,k-1}}^{\wedge}\right)$ to rectify the raw distorted points of $f_{\mathrm{k}}^{\mathrm{L}}$, as in Equation (15).

$$\hat{\boldsymbol{X}}_{\mathrm{(i,b,k)}}^{\mathrm{B}} = \exp\left(\triangle\widetilde{\boldsymbol{\xi}}_{\mathrm{k,k-1}} \cdot \frac{i}{S_b}\right)^{\wedge} \cdot \boldsymbol{X}_{\mathrm{(i,b,k-j)}}^{\mathrm{B}}. \tag{15}$$

The undistorted frame after rectification is denoted as $\hat{f}_{\mathrm{k}}^{\mathrm{L}}$. In Section 4, we do experiments to compare these three approaches.

3.5. Intermediate Step: Inter-Frame Registration

To register two frames of point clouds, we could use any off-the-shelf scan registration approaches, such as ICP, NDT, and LOAM. To successfully apply these approaches, it always requires an accurate initial guess for the estimated transformation. Just as mentioned previously, the problem of inter-frame registration is, in fact, quite related to intra-frame correction. Therefore, the approaches mentioned in Section 3.4 can also be applied here. We can either use the registration result of the previous two frames, or use the IMU data directly. We undertake experiments to compare these two approaches, presented in Section 4.

3.6. Output Step: Fusion in an EKF Framework

3.6.1. Basic Concepts

The Extended Kalman filter (EKF) [28] is one of the most commonly used data fusion algorithms. It is a recursive state estimator for the nonlinear system, and the state is assumed to satisfy a Gaussian distribution. A typical EKF algorithm consists of two steps: a prediction step, and an update step.

In the prediction step, the previous state $\boldsymbol{x}_{\mathrm{k}-1}$ is used to compute a priori estimate of the current state by:

$$\begin{aligned} \bar{\boldsymbol{x}}_{\mathrm{k}} &= \boldsymbol{f}\left(\hat{\boldsymbol{x}}_{\mathrm{k}-1}\right), \\ \bar{\boldsymbol{P}}_{\mathrm{k}} &= \boldsymbol{F}\,\hat{\boldsymbol{P}}_{\mathrm{k}-1}\,\boldsymbol{F}^{T} + \boldsymbol{R}, \end{aligned} \tag{16}$$

where R is the covariance of the process noise. $\bar{\boldsymbol{x}}_{\mathrm{k}}$ and $\bar{\boldsymbol{P}}_{\mathrm{k}}$ are the mean and covariance matrices of the prior distribution of the state $\boldsymbol{x}_{\mathrm{k}}$, respectively. $\hat{\boldsymbol{x}}_{\mathrm{k}-1}$ and $\hat{\boldsymbol{P}}_{\mathrm{k}-1}$ are the mean and covariance matrices of the posterior Gaussian distribution of the state $\boldsymbol{x}_{\mathrm{k}-1}$. By performing a first-order Taylor expansion to the function $\boldsymbol{f}$ at the mean of the state $\boldsymbol{x}_{\mathrm{k}-1}$, the first-order partial derivative can be denoted as the Jacobian matrix $\boldsymbol{F}$:

$$\boldsymbol{F} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}_{\mathrm{k}-1}}\right|_{\hat{\boldsymbol{x}}_{\mathrm{k}-1}}. \tag{17}$$

In the update step, the priori estimate of the current state is combined with the current observation $\boldsymbol{z}_{\mathrm{k}}$ to refine the state estimate and obtain the posteriori state estimate $\boldsymbol{x}_{\mathrm{k}}$, which can be formulated as:

$$\begin{aligned} \boldsymbol{K}_{\mathrm{k}} &= \bar{\boldsymbol{P}}_{\mathrm{k}}\,\boldsymbol{H}^{T}\left(\boldsymbol{H}\,\bar{\boldsymbol{P}}_{\mathrm{k}}\,\boldsymbol{H}^{T} + \boldsymbol{Q}\right)^{-1}, \\ \hat{\boldsymbol{x}}_{\mathrm{k}} &= \bar{\boldsymbol{x}}_{\mathrm{k}} + \boldsymbol{K}_{\mathrm{k}}\left(\boldsymbol{z}_{\mathrm{k}} - \boldsymbol{h}\left(\bar{\boldsymbol{x}}_{\mathrm{k}}\right)\right), \\ \hat{\boldsymbol{P}}_{\mathrm{k}} &= \left(\boldsymbol{I} - \boldsymbol{K}_{\mathrm{k}}\boldsymbol{H}\right)\bar{\boldsymbol{P}}_{\mathrm{k}}, \end{aligned} \tag{18}$$

where $\boldsymbol{Q}$ represents the covariance of the measurement noise. $\boldsymbol{K}_{\mathrm{k}}$ is the Kalman gain and the posteriori state estimation $\boldsymbol{x}_{\mathrm{k}} \sim \mathcal{N}\left(\hat{\boldsymbol{x}}_{\mathrm{k}}, \hat{\boldsymbol{P}}_{\mathrm{k}}\right)$. Similarly, a first-order Taylor expansion to the function $\boldsymbol{h}$ can be performed at the mean of the prior distribution of the state $\boldsymbol{x}_{\mathrm{k}}$. The first-order partial derivative could be defined as the Jacobian matrix $\boldsymbol{H}$:

$$\boldsymbol{H} = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}_{\mathrm{k}}}\right|_{\bar{\boldsymbol{x}}_{\mathrm{k}}}. \tag{19}$$

In this section, We use an EKF framework to fuse the IMU and the results obtained from the lidar odometry, as shown in Figure 5. Let the motion prior model presented in Section 3.2 be the system model and the results from the lidar odometry (which is denoted as the red star in Figure 5) be the measurements. The system model and measurement model are described as follows.

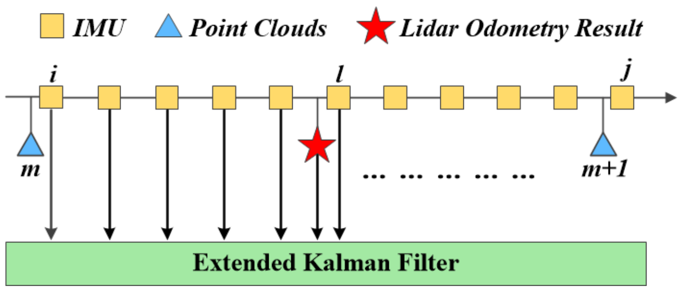


Figure 5. The diagram of the proposed fusion framework. The yellow rectangle represents the IMU data, and the blue triangle indicates the lidar point clouds. The red star denotes the localization result output from the lidar odometry.

3.6.2. System Model

Since the main purpose of our system is to obtain the pose of the vehicle, we could use the *Lie Algebra* $\boldsymbol{\xi}$ to define the state vector $\boldsymbol{x}_\mathrm{k}$ at time t_k as:

$$\boldsymbol{x}_\mathrm{k} = \begin{bmatrix} \boldsymbol{\theta}_\mathrm{k} & \boldsymbol{p}_\mathrm{k} \end{bmatrix}^T \in \mathfrak{se}(3)\,. \tag{20}$$

By utilizing the logarithmic map and the approximate Baker-Campbell-Hausdorff (BCH) formula [21], the $\boldsymbol{R}_\mathrm{k}^\mathrm{WB}$ obtained from the Equation (6) can be converted to:

$$\boldsymbol{\theta}_\mathrm{k} = \boldsymbol{J}_l^{-1}\left(\boldsymbol{\theta}_{\mathrm{k}-1}\right) \cdot \left(\boldsymbol{\omega}_\mathrm{k} \triangle t\right) + \boldsymbol{\theta}_{\mathrm{k}-1}\,, \tag{21}$$

where $\boldsymbol{J}_l^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right)$ denotes the right BCH Jacobian of SO(3) and it equals to:

$$\boldsymbol{J}_l^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) = \frac{\theta_\mathrm{k}}{2}\cot\frac{\theta_\mathrm{k}}{2}\boldsymbol{I} + \left(1 - \frac{\theta_\mathrm{k}}{2}\cot\frac{\theta_\mathrm{k}}{2}\right)\boldsymbol{n}_\mathrm{k}\boldsymbol{n}_\mathrm{k}^T - \frac{\theta_\mathrm{k}}{2}\boldsymbol{n}_\mathrm{k}^\wedge\,, \tag{22}$$

and $\boldsymbol{J}_l^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right)$ can be denoted as:

$$\boldsymbol{J}_l^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) = \begin{bmatrix} J_1^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) & J_2^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) & J_3^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) \\ J_4^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) & J_5^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) & J_6^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) \\ J_7^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) & J_8^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) & J_9^{-1}\left(\boldsymbol{\theta}_\mathrm{k}\right) \end{bmatrix}. \tag{23}$$

Combining the Equations (8), (21) and (23), the state transition equation could be defined as:

$$\boldsymbol{x}_\mathrm{k} = \boldsymbol{x}_{\mathrm{k}-1} + \begin{bmatrix} \boldsymbol{J}_l^{-1}\left(\boldsymbol{\theta}_{\mathrm{k}-1}\right) \cdot \boldsymbol{\omega}_\mathrm{k} \triangle t \\ \boldsymbol{R}_\mathrm{k}^\mathrm{WB}(1:3,1) \cdot \triangle s \end{bmatrix}, \tag{24}$$

where $\boldsymbol{R}_\mathrm{k}^\mathrm{WB}$ is computed by θ_k^x, θ_k^y and θ_k^z according to Equation (3) and $\boldsymbol{R}_\mathrm{k}^\mathrm{WB}(1:3,1)$ means the first column of the $\boldsymbol{R}_\mathrm{k}^\mathrm{WB}$.

According to Equations (17) and (24), $\boldsymbol{F}$ can be derived as:

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{F}_{\theta\theta} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{F}_{p\theta} & \boldsymbol{I}_{3\times3} \end{bmatrix} \in \mathbb{R}^{6\times6}\,, \tag{25}$$

where $\boldsymbol{F}_{\theta\theta} \in \mathbb{R}^{3\times3}$, and it can be computed by:

$$\boldsymbol{F}_{\theta\theta} = \boldsymbol{I}_{3\times3} + \begin{bmatrix} \left(\frac{\partial \boldsymbol{J}_l^{-1}(\boldsymbol{\theta}_{\mathrm{k}-1})}{\partial \theta_{\mathrm{k}-1}^{x}} \cdot \boldsymbol{\omega}_{\mathrm{k}} \triangle t\right)^T \\ \left(\frac{\partial \boldsymbol{J}_l^{-1}(\boldsymbol{\theta}_{\mathrm{k}-1})}{\partial \theta_{\mathrm{k}-1}^{y}} \cdot \boldsymbol{\omega}_{\mathrm{k}} \triangle t\right)^T \\ \left(\frac{\partial \boldsymbol{J}_l^{-1}(\boldsymbol{\theta}_{\mathrm{k}-1})}{\partial \theta_{\mathrm{k}-1}^{z}} \cdot \boldsymbol{\omega}_{\mathrm{k}} \triangle t\right)^T \end{bmatrix}^T . \tag{26}$$

$\boldsymbol{F}_{p\theta}$ can be computed by:

$$\boldsymbol{F}_{p\theta} = \begin{bmatrix} \frac{\partial \boldsymbol{R}_{\mathrm{k}}^{\mathrm{WB}}(1{:}3{,}1)}{\partial \theta_{\mathrm{k}-1}^{x}} & \frac{\partial \boldsymbol{R}_{\mathrm{k}}^{\mathrm{WB}}(1{:}3{,}1)}{\partial \theta_{\mathrm{k}-1}^{y}} & \frac{\partial \boldsymbol{R}_{\mathrm{k}}^{\mathrm{WB}}(1{:}3{,}1)}{\partial \theta_{\mathrm{k}-1}^{z}} \end{bmatrix} . \tag{27}$$

The prior probability distribution of the state $\boldsymbol{x}_{\mathrm{k}}$ could then be obtained by the Equation (16).

3.6.3. Measurement Model

The measurement vector $\boldsymbol{z}_{\mathrm{k}}$ is denoted as:

$$\boldsymbol{z}_{\mathrm{k}} = \begin{bmatrix} \theta_{z_{\mathrm{k}}}^{x} & \theta_{z_{\mathrm{k}}}^{y} & \theta_{z_{\mathrm{k}}}^{z} & p_{z_{\mathrm{k}}}^{x} & p_{z_{\mathrm{k}}}^{z} & p_{z_{\mathrm{k}}}^{z} \end{bmatrix}^T . \tag{28}$$

According to Figure 5, the lidar odometry algorithm starts to run at time $t_{\mathrm{m}}^{\mathrm{L}}$, but the result is obtained at time t_{k}, which is several tens of milliseconds after $t_{\mathrm{m}}^{\mathrm{L}}$. We assume that the pose of the vehicle at time $t_{\mathrm{m}}^{\mathrm{L}}$ and t_{k} were approximately equal to $\boldsymbol{T}_{\mathrm{i}}^{\mathrm{WB}}$ and $\boldsymbol{T}_{l}^{\mathrm{WB}}$. Both $\boldsymbol{T}_{\mathrm{i}}^{\mathrm{WB}}$ and $\boldsymbol{T}_{l}^{\mathrm{WB}}$ can be obtained from the IMU data.

The relationship between the measurement $\boldsymbol{z}_{\mathrm{k}}$ and the real state $\boldsymbol{x}_{\mathrm{k}}$ of the vehicle at time t_{k} can be indicated by $\triangle \boldsymbol{T}_{\mathrm{i},l}^{\mathrm{B}}$, which can also be denoted as $\triangle \boldsymbol{\xi}_{\mathrm{i},l} = \begin{bmatrix} \triangle \boldsymbol{\theta}_{\mathrm{i},l} & \triangle \boldsymbol{p}_{\mathrm{i},l} \end{bmatrix}^T$. The observation equation could then be represented as:

$$\boldsymbol{z}_{\mathrm{k}} = \begin{bmatrix} \boldsymbol{\theta}_{\mathrm{k}} \\ \triangle \boldsymbol{p}_{\mathrm{i},l} \end{bmatrix} + \begin{bmatrix} \boldsymbol{J}_l^{-1}(\boldsymbol{\theta}_{\mathrm{k}}) \cdot \triangle \boldsymbol{\theta}_{\mathrm{i},l} \\ \triangle \boldsymbol{R}_{\mathrm{i},l}^{\mathrm{B}} \cdot \boldsymbol{p}_{\mathrm{k}} \end{bmatrix} . \tag{29}$$

According to Equations (19) and (29), $\boldsymbol{H}$ can be derived as:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_{\theta\theta} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{H}_{pp} \end{bmatrix} , \tag{30}$$

where $\boldsymbol{H}_{pp} = \triangle \boldsymbol{R}_{\mathrm{i},l}^{\mathrm{WB}} \in \mathbb{R}^{3\times3}$, $\boldsymbol{H}_{\theta\theta} \in \mathbb{R}^{3\times3}$ and $\boldsymbol{H}_{\theta\theta}$ can be computed by:

$$\boldsymbol{H}_{\theta\theta} = \begin{bmatrix} \left(\frac{\partial \boldsymbol{J}_l^{-1}(\boldsymbol{\theta}_{\mathrm{k}})}{\partial \theta_{\mathrm{k}}^{x}} \cdot \triangle \boldsymbol{\theta}_{\mathrm{i},l}\right)^T \\ \left(\frac{\partial \boldsymbol{J}_l^{-1}(\boldsymbol{\theta}_{\mathrm{k}})}{\partial \theta_{\mathrm{k}}^{y}} \cdot \triangle \boldsymbol{\theta}_{\mathrm{i},l}\right)^T \\ \left(\frac{\partial \boldsymbol{J}_l^{-1}(\boldsymbol{\theta}_{\mathrm{k}})}{\partial \theta_{\mathrm{k}}^{z}} \cdot \triangle \boldsymbol{\theta}_{\mathrm{i},l}\right)^T \end{bmatrix}^T . \tag{31}$$

Based on the covariance matrix $\bar{\boldsymbol{P}}_{\mathrm{k}}$ from Equation (16) and the Jacobian matrix $\boldsymbol{H}$ from Equation (30), the Kalman gain $\boldsymbol{K}_{\mathrm{k}}$ and the posteriori state estimation could be computed by the Equation (18).

This fusion framework is summarized in Algorithm 1.

Algorithm 1: The fusion framework of our system.

```
input  : The history pose set $\{T_i^{WB}\}$ and pose $T_{B_k}^{WB}$ at time $t_k^B$ from the IMU; pose $T_{L_k}^{WB}$ at time
         $t_k^L$ from the lidar odometry; previous pose $T_{k-1}^{WB}$
output : Current pose $T_k^{WB}$
1  begin
2      use $T_{B_k}^{WB}$ and $T_{k-1}^{WB}$ to do the state transition by Equation (16) and obtain the prior
        distribution $\bar{x}_k$;
3      if have $T_{L_k}^{WB}$ then
4          use $\bar{x}_k$, $T_{L_k}^{WB}$ and $\{T_i^{WB}\}$ to compute the posterior distribution $\hat{x}_k$ by Equation (18)
5          output the $T_k^{WB}$ by $\hat{x}_k$
6      end
7      else
8          output the $T_k^{WB}$ by $\bar{x}_k$
9      end
10 end
```

4. Experimental Results

The platform used in the following experiments is shown in Figure 6. The vehicle is equipped with a Velodyne HDL-64E lidar, a Xsens MTI300 IMU, two photoelectric encoders, a NovAtel SPAN-CPT GNSS/INS system, and a Nova 5100 industrial computer. The lidar is mounted on the top of the car, and it has 64 beams which could generate over 1.3 million points per second with a high range accuracy. The horizontal Field of View (FOV) is 360°, the vertical FOV is 26.8°, and the maximal detection distance is 120 m. The NovAtel GNSS/INS system provides localization results with centimeter-level accuracy, which can be served as the ground truth.

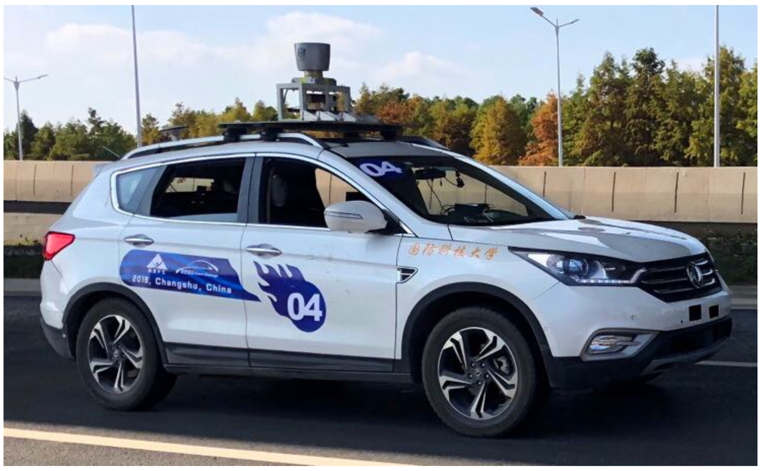


Figure 6. Our experimental platform. It is equipped with a Velodyne HDL-64E Lidar, a Xsens MTI300 IMU, two photoelectric encoders, a NovAtel SPAN-CPT GNSS/INS system, and a Nova 5100 industrial computer.

4.1. Experiments on Hand–Eye Calibration

As described in Section 3.3, in order to perform the lidar-IMU hand–eye calibration, we need to utilize multiple pairs of relative transformations $\{\triangle T_{k-1,k}^{B}, \triangle T_{k-1,k}^{L}\}$.

We choose the frame pairs $\{f_i^L, f_{i-1}^L\}$ before and after the vehicle makes a turn. In addition, all the frames are collected when the vehicle is stationary. Figure 7 shows several transformation pairs we have chosen. In each sub-figure, the left part is the registration result before the hand–eye calibration, whilst the right half is the result after hand–eye calibration. It can be seen that f_i^L and f_{i-1}^L are much better-registered in the right half of Figure 7a–d than the left half, which indicates the importance of hand–eye calibration.

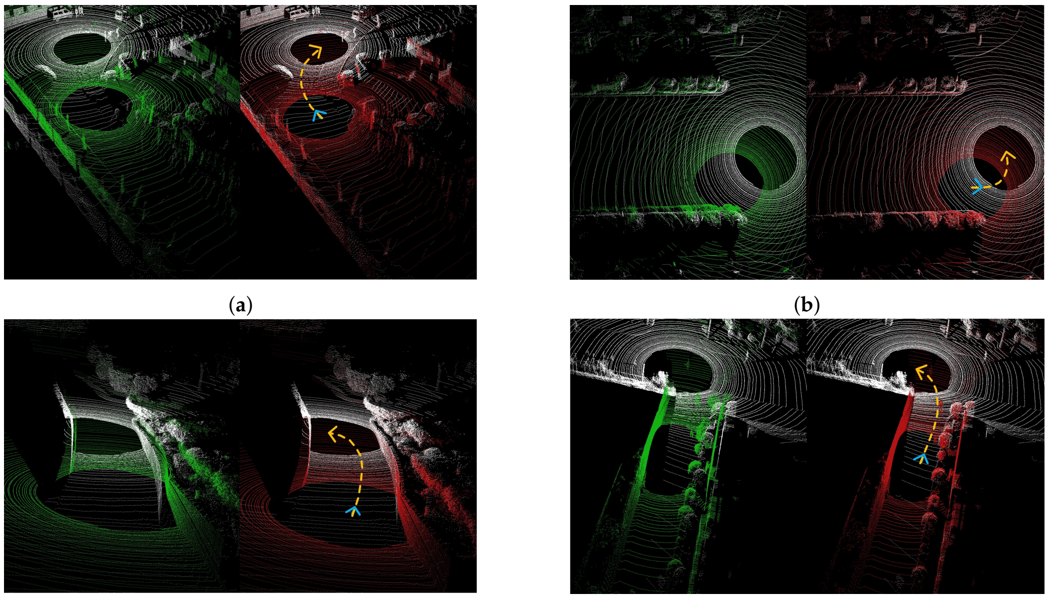

(**a**) (**b**) (**c**) (**d**)

Figure 7. Four typical lidar pairs chosen to perform the hand–eye calibration. In each sub-figure, the left part is the registration result before the hand–eye calibration, whilst the right half is the result after hand–eye calibration. The yellow dotted line indicates the movement trajectory of the vehicle, while the yellow arrow and the blue arrow denote the orientation of the vehicle.

Since the ground truth of $\boldsymbol{T}^{\mathrm{BL}}$ is impossible to be obtained, we could not verify the calibration result directly. However, we could use the IMU data to assemble different frames together, and the quality of the assembled scene can be used to evaluate the accuracy of the hand–eye calibration result. Figure 8 shows the assembled scene when the vehicle makes a turn at the intersection. Figure 8a is the assembled result before hand–eye calibration, which is very vague, while the result after the hand–eye calibration in Figure 8b is much clearer. This suggests that the result of our hand–eye calibration is accurate.

To quantitatively test the effect of hand–eye calibration, we choose the LOAM algorithm as the lidar odometry algorithm, and compare the estimated trajectories with the ground-truth trajectory generated by the GNSS/INS system. Figure 9 illustrates the localization result before and after hand–eye calibration. The translation error and rotation error are shown in Figure 9b,c, where the horizontal axis represents the traveled distance and the vertical axis shows the errors. It is obvious that the translation error and the rotation error are significantly reduced after the hand–eye calibration.

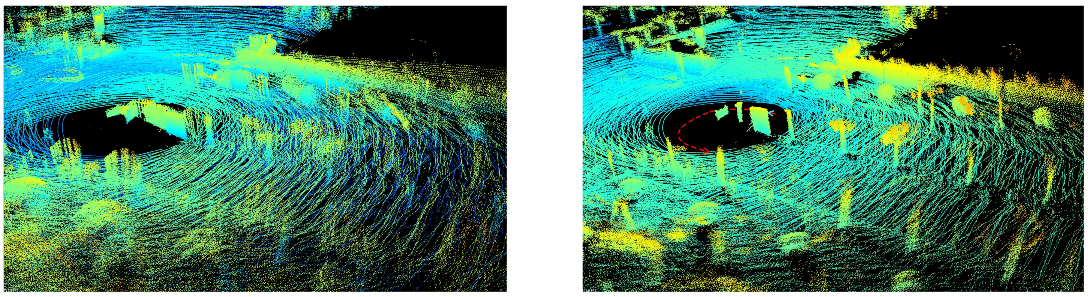

(**a**) (**b**)

Figure 8. The 3D reconstruction result when the vehicle turns around at the intersection. The left (**a**) is the result before hand–eye calibration, and the right (**b**) is the result after hand–eye calibration.

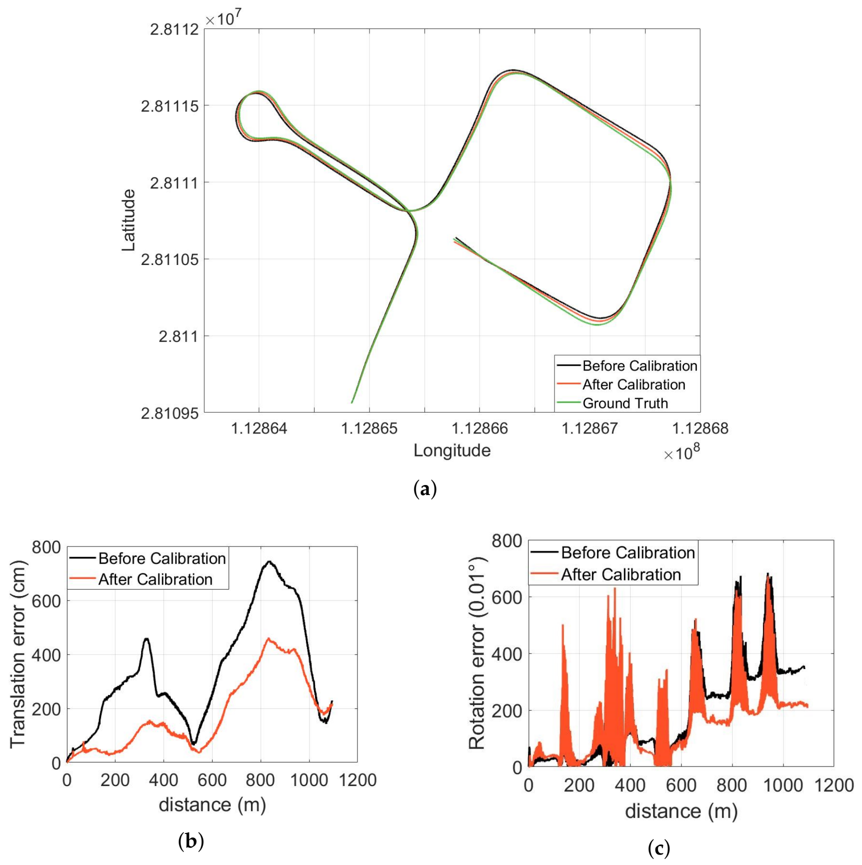


Figure 9. The localization results before and after hand–eye calibration, where (**a**) compares the trajectories before/after hand–eye calibration with the ground truth; (**b**) compares the translation error before/after calibration, and rotation error is shown in (**c**). In this experiment, the UGV drives at a speed around 25 km/h, and the overall distance is about 1.1 km.

4.2. Experiments on Intra-Frame Correction

Based on the hand–eye calibration results, we could also use the 3D reconstruction results to qualitatively test the three rectification methods introduced in Section 3.4. In order to test the result of the rectification, the vehicle should run fast to amplify the intra-frame distortion. Therefore, we carry out this experiment when the vehicle drives at around 60 km/h, and the results are shown in Figure 10.

Figure 10a–d represent different 3D reconstruction results when the vehicle is making a turn. Figure 10a is the result without any rectification, and the reconstructed scene is quite blurry. Figure 10b–d show the results when the distorted point clouds are rectified by the previous registration result, by registering two distorted point clouds and by the IMU data. The results obtained by these three approaches are all much better than the one without rectification. Moreover, the average processing time for these three methods applied on 2000 frames of point clouds is displayed in Table 1. It is seen that the approach of registering two distorted point clouds is the most computational expensive approach.

To quantitatively test the effect of intra-frame correction, LOAM is also used as the lidar odometry algorithm, and the localization accuracy is compared with the ground-truth trajectory. Figure 11 compares the localization results with/without rectification. The trajectories generated by the three rectification methods and without rectification are compared with the ground truth in Figure 11a, where “Method 1” represents rectification by previous registration, “Method 2” is the approach of registering two distorted point clouds, and “Method 3” indicates rectification by IMU data. In addition, Figure 11b,c compare the translation error and rotation error of these approaches.

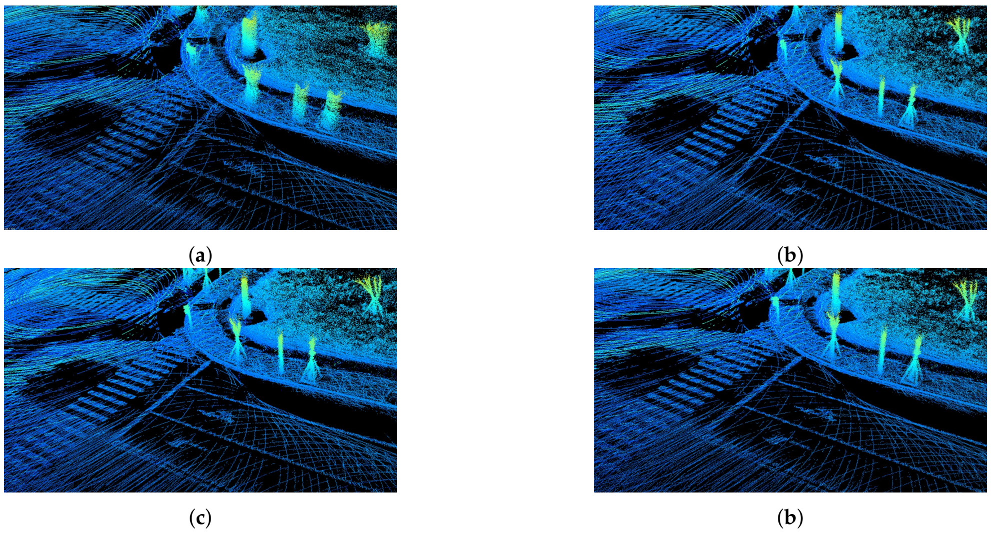
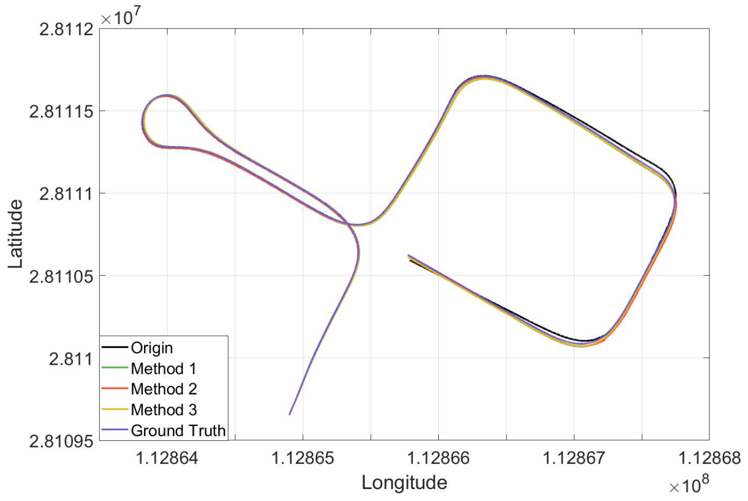
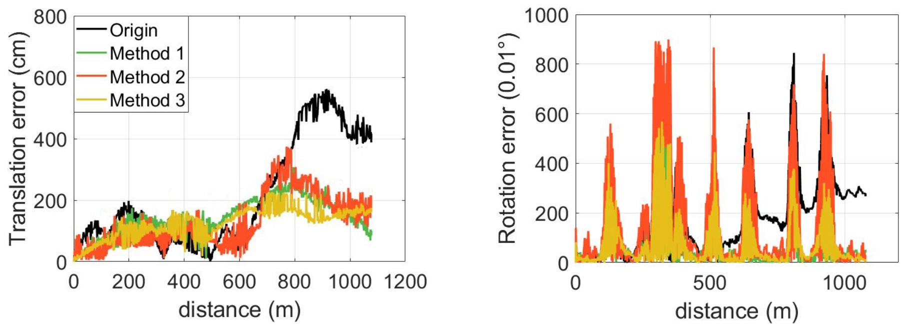

(**a**) (**b**) (**c**) (**b**)

Figure 10. The 3D reconstruction results to test the three rectification methods. (**a**) represents the result without any rectification, and (**b**–**d**) show the results when the distorted point clouds are rectified by the previous registration result, by registering two distorted point clouds and by the IMU data, respectively.



(**a**)



(**b**) (**c**)

Figure 11. The localization results with/without rectification, where (**a**) compares the trajectories generated by the three rectification methods (Methods 1–3 represent rectification by the previous registration result, by registering two distorted point clouds and by the IMU data, respectively), the trajectory generated without rectification (Origin) and the ground truth; and (**b**,**c**) compare the translation error and rotation error among them, respectively. In this experiment, the UGV drives at a speed of around 60 km/h.

It can be seen that both the translation error and the rotation error are significantly reduced after rectification, and "Method 3" exhibits the best performance, followed by "Method 1" and "Method 2". A detailed error comparison is presented in Table 2.

Table 1. Computational efficiency of three rectification methods.

Rectification Methods	Average Processing Time (ms)
By previous registration	2.565
By registering two distorted point clouds	68.504
By IMU data	4.546

Table 2. Error comparison of three rectification methods.

Rectification Methods	Mean Relative Position Error
Method 1	0.79%
Method 2	0.94%
Method 3	0.57%
Without Rectification	2.28%

4.3. Experiments on Inter-Frame Registration

4.3.1. Tests with Different Methods of Assigning Initial Guess

In this experiment, two methods of assigning initial guess for the lidar odometry algorithm are compared—namely, using the IMU data or the registration result from the previous two scans. In both approaches, the raw distorted point clouds are rectified by the IMU data, and LOAM is used as the lidar odometry algorithm.

The tests are conducted when the vehicle drives at different speeds: a relatively slow speed of around 25 km/h, and a faster speed of around 60 km/h. A detailed error comparison is shown in Table 3. It can be seen that using the IMU data as the initial guess always produces a better result than using the registration result from the previous two scans, especially when the vehicle drives at a faster speed.

Table 3. Error comparison of different methods at the intermediate step.

Speed (km/h)	Methods of Assigning Initial Guess	Mean Relative Position Error
25	IMU	0.31%
	Previous regist	0.46%
60	IMU	0.57%
	Previous regist	1.32%

4.3.2. Tests with Different Lidar Odometry Algorithms

Here, we compare three popular lidar odometry algorithms: ICP, NDT, and LOAM. During the experiments, the point clouds are rectified by the IMU data. The vehicle drives at a speed around 25 km/h, and the overall distance is about 1.1 km. A detailed error comparison is shown in Table 4. It is obvious that LOAM performs much better than NDT and ICP.

Table 4. Error comparison of three lidar odometry algorithms.

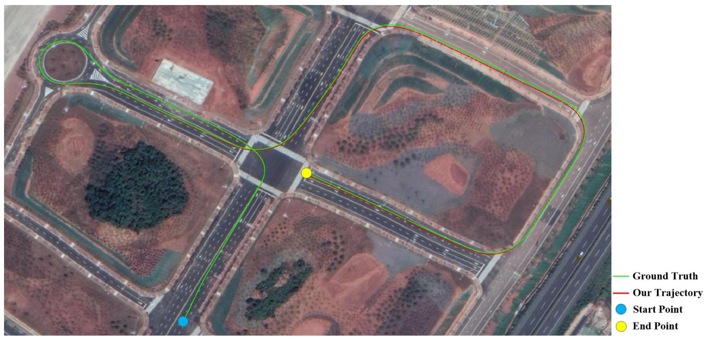
Lidar Odometry Methods	Mean Relative Position Error
ICP	6.28%
NDT	3.85%
LOAM	0.31%

4.3.3. Tests in Both Urban and Off-Road Environments

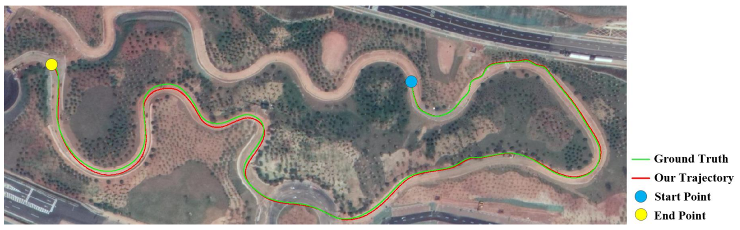
Our algorithm is tested in both urban and off-road environment. We choose to use LOAM as the lidar odometry algorithm. The point clouds are rectified by the IMU data. In the urban environment, the vehicle drives at a speed of around 25 km/h, and the overall distance is about 1.1 km. In the off-road environment, the vehicle drives at a speed of around 15 km/h and the overall distance is about 1.0 km. The trajectories generated by our method are compared with the ground truth in Figure 12. A detailed error comparison is shown in Table 5. It is seen that our approach is applicable for both the urban and off-road environments, whilst the error in the off-road scenario is slightly higher than the urban environment.

Table 5. Error comparison of diverse environments.

Environment	Distance (km)	Mean Relative Position Error
Urban	1.097	0.31%
Off-road	0.983	0.93%



(**a**)

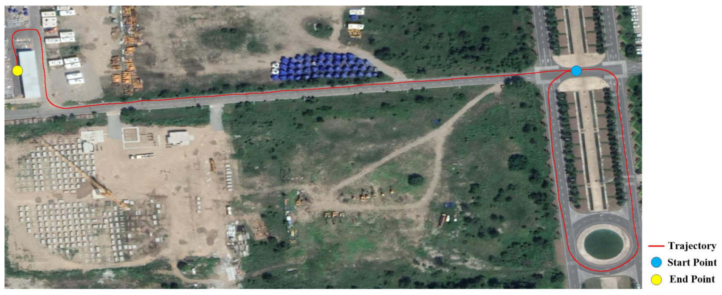


(**b**)

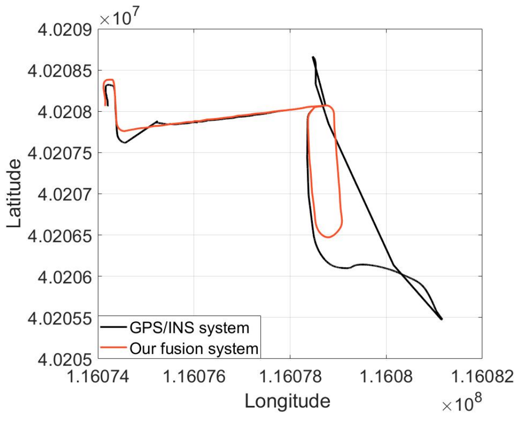
Figure 12. The results in the urban environment and off-road environment. (**a**) shows the trajectory generated in the urban environment when the UGV drives at a speed of around 25 km/h, and (**b**) shows the trajectory generated in the off-road environment when the UGV drives at a speed of around 15 km/h.

4.4. Experiments in GNSS-Denied Environment

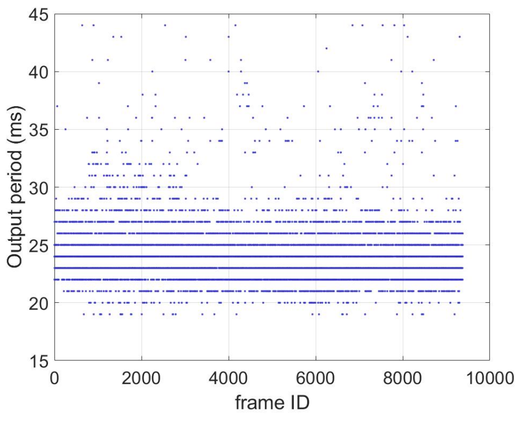
Finally, we test our whole localization system in the GNSS-Denied environment. Figure 13a shows the trajectory generated by our localization system when the UGV drives at a speed of 20 km/h. By projecting the trajectory to the satellite image, we find that the position drift is quite small. Figure 13b compares the trajectories generated by our localization system and by the disturbed GNSS/INS system. Figure 13c shows the output frequency of our localization system. The average output frequency is around 39.7 Hz. In summary, no matter whether the GNSS signal is available or not, our system can output precise and high-frequency localization results.



(**a**)



(**b**)



(**c**)

Figure 13. The results in a GNSS-deniedenvironment. (**a**) shows the trajectory generated by our system when the UGV drives at a speed around 20 km/h and the overall distance is about 1.3 km. (**b**) compares the trajectories generated by our system and by the disturbed GNSS/INS system. (**c**) shows the output period of our localization system during the whole process, and the average output frequency is about 39.7 Hz.

5. Conclusions

In this paper, we present a novel approach for combining the IMU data with lidar odometry. The proposed localization system could generate stable, accurate, and high-frequency localization results. Since our system does not recognize the loop closure, we can further extend our work to enable the ability of loop-closure detection.

Author Contributions: Conceptualization, H.F.; Data curation, H.X.; Formal analysis, H.X.; Funding acquisition, B.D.; Investigation, H.X.; Methodology, H.F. and H.X.; Project administration, B.D.; Resources, H.F. and B.D.; Software, H.F. and H.X.; Supervision, B.D.; Validation, H.F.; Original draft, H.X.; Review and editing, H.F.

Funding: This work was supported by the National Natural Science Foundation of China under No. 61790565.

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

References

1. Kos, T.; Markezic, I.; Pokrajcic, J. Effects of multipath reception on GPS positioning performance. In Proceedings of the Elmar-2010, Zadar, Croatia, 15–17 September 2010; pp. 399–402.
2. Kohlbrecher, S.; von Stryk, O.; Meyer, J.; Klingauf, U. A flexible and scalable SLAM system with full 3D motion estimation. In Proceedings of the 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, Kyoto, Japan, 1–5 November 2011; pp. 155–160.
3. Gao, Y.; Liu, S.; Atia, M.M.; Noureldin, A. INS/GPS/LiDAR integrated navigation system for urban and indoor environments using hybrid scan matching algorithm. *Sensors* **2015**, *15*, 23286–23302. [CrossRef] [PubMed]

4. Tang, J.; Chen, Y.; Niu, X.; Wang, L.; Chen, L.; Liu, J.; Shi, C.; Hyyppä, J. LiDAR scan matching aided inertial navigation system in GNSS-denied environments. *Sensors* **2015**, *15*, 16710–16728. [CrossRef] [PubMed]
5. Liu, S.; Atia, M.M.; Gao, Y.; Givigi, S.; Noureldin, A. An inertial-aided LiDAR scan matching algorithm for multisensor land-based navigation. In Proceedings of the 27th International Technical Meeting of the Satellite Division of the Institute of Navigation, Tampa, FL, USA, 8–12 September 2014; pp. 2089–2096.
6. Ye, H.; Liu, M. LiDAR and Inertial Fusion for Pose Estimation by Non-linear Optimization. *arXiv* **2017**, arXiv:1710.07104.
7. Pierzchala, M.; Giguere, P.; Astrup, R. Mapping forests using an unmanned ground vehicle with 3D LiDAR and graph-SLAM. *Comput. Electron. Agric.* **2018**, *145*, 217–225. [CrossRef]
8. Besl, P.J.; McKay, N.D. A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 239–256. [CrossRef]
9. Pomerleau, F.; Colas, F.; Siegwart, R.; Magnenat, S. Comparing ICP variants on real-world data sets. *Auton. Robots* **2013**, *34*, 133–148. [CrossRef]
10. Stoyanov, T.; Magnusson, M.; Andreasson, H.; Lilienthal, A.J. Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations. *Int. J. Robot. Res.* **2012**, *31*, 1377–1393. [CrossRef]
11. Magnusson, M.; Lilienthal, A.; Duckett, T. Scan registration for autonomous mining vehicles using 3D-NDT. *J. Field Robot.* **2007**, *24*, 803–827. [CrossRef]
12. Nguyen, V.; Gachter, S.; Martinelli, A.; Tomatis, N.; Siegwart, R. A comparison of line extraction algorithms using 2D range data for indoor mobile robotics. *Auton. Robots* **2007**, *23*, 97–111. [CrossRef]
13. Aghamohammadi, A.; Taghirad, H. Feature-Based Laser Scan Matching for Accurate and High Speed Mobile Robot Localization. In Proceedings of the EMCR, Freiburg, Germany, 19–21 September 2007.
14. Furukawa, T.; Dantanarayana, L.; Ziglar, J.; Ranasinghe, R.; Dissanayake, G. Fast global scan matching for high-speed vehicle navigation. In Proceedings of the 2015 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), San Diego, CA, USA, 14–16 September 2015; pp. 37–42.
15. Zhang, J.; Singh, S. LOAM: Lidar Odometry and Mapping in Real-time. In Proceedings of the Robotics: Science and Systems, Berkeley, CA, USA, 12–16 July 2014; pp. 109–111.
16. Zhang, J.; Singh, S. Low-drift and real-time lidar odometry and mapping. *Auton. Robots* **2017**, *41*, 401–416. [CrossRef]
17. Hemann, G.; Singh, S.; Kaess, M. Long-range GPS-denied aerial inertial navigation with LIDAR localization. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 1659–1666.
18. Wan, G.; Yang, X.; Cai, R.; Li, H.; Zhou, Y.; Wang, H.; Song, S. Robust and Precise Vehicle Localization Based on Multi-Sensor Fusion in Diverse City Scenes. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 4670–4677.
19. Barfoot, T.D. *State Estimation for Robotics*; Cambridge University Press: Cambridge, UK, 2017; pp. 231–234.
20. Kepper, J.H.; Claus, B.C.; Kinsey, J.C. A Navigation Solution Using a MEMS IMU, Model-Based Dead-Reckoning, and One-Way-Travel-Time Acoustic Range Measurements for Autonomous Underwater Vehicles. *IEEE J. Ocean. Eng.* **2018**, 1–19. [CrossRef]
21. Forster, C.; Carlone, L.; Dellaert, F.; Scaramuzza, D. On-Manifold Preintegration for Real-Time Visual–Inertial Odometry. *IEEE Trans. Robot.* **2017**, *33*, 1–21. [CrossRef]
22. Wu, L.; Ren, H. Finding the Kinematic Base Frame of a Robot by Hand-Eye Calibration Using 3D Position Data. *IEEE Trans. Autom. Sci. Eng.* **2017**, *14*, 314–324. [CrossRef]
23. Shah, M.; Eastman, R.D.; Hong, T. An overview of robot-sensor calibration methods for evaluation of perception systems. In Proceedings of the Workshop on Performance Metrics for Intelligent Systems, College Park, MD, USA, 20–22 March 2012; pp. 15–20.
24. Tsai, R.Y.; Lenz, R.K. A new technique for fully autonomous and efficient 3D robotics hand/eye calibration. *IEEE Trans. Robot. Autom.* **1989**, *5*, 345–358. [CrossRef]
25. Dornaika, F.; Horaud, R. Simultaneous robot-world and hand-eye calibration. *IEEE Trans. Robot. Autom.* **1998**, *14*, 617–622. [CrossRef]
26. Huang, K.; Stachniss, C. On Geometric Models and Their Accuracy for Extrinsic Sensor Calibration. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation, Brisbane, QLD, Australia, 21–25 May 2018; pp. 1–9.

27. Scherer, S.; Rehder, J.; Achar, S.; Cover, H.; Chambers, A.; Nuske, S.; Singh, S. River mapping from a flying robot: State estimation, river detection, and obstacle mapping. *Auton. Robots* **2012**, *31*, 189–214. [CrossRef]
28. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]