Spatiotemporal calibration based on nonlinear optimization for heterogeneous information including GNSS raw data

Yanfang Shi[1,2], *Student Member*, Baowang Lian[1], Yonghong Zeng[2], *Fellow, IEEE* , Yugang Ma[2], *Senior Member*, Yangyang Liu[1]

***Abstract*—Achieving high-precision positioning through multi-source integration has become an inevitable trend in autonomous vehicle systems, and the spatiotemporal calibration of multi-source information is the primary prerequisite. This paper proposes a spatiotemporal calibration algorithm for the fusion system of GNSS data, LiDAR data, and visual data with the inertial sensor as the central coordinate system. Firstly, we use the pseudo-distance information of GNSS to construct the space-time calibration model of GNSS (Global Navigation Satellite System) relative to IMU (Inertial Measurement Unit). Secondly, based on the reprojection principle, we construct a spatiotemporal calibration model of visual images relative to the IMU. Then, according to the distance formula of the LiDAR (Light Detection and Ranging) points cloud, the space-time calibration model of the LiDAR points cloud relative to the IMU is established. Finally, we use the nonlinear optimization algorithm to obtain the spatiotemporal parameters. We have done extensive simulations based on simulated data and publicly available real-world datasets. The simulation results show that using the proposed calibration model yields spatiotemporal parameter accuracy superior to existing calibration algorithms and exhibits some degree of robustness to the noise in IMU data. It achieves approximately 40% improvement in position estimation accuracy with the open-source odometry and the real-world datasets while ensuring good safety and reliability under high computational efficiency.**

***Index Terms*—GNSS, spatiotemporal calibration, vision, LiDAR, fusion, IMU**

I. Introduction

MULTI-source fusion is an irreplaceable method in motor vehicle to achieve precise positioning. It has become a trend to utilize the collaboration of an increasing number of sensor data and compensate for the limitations of individual sensors to achieve positioning for motor vehicle. The measurements obtained by different sensors observing the same target may not be synchronized. Therefore, it is not wise to directly send the acquired measurements to the central processor for data fusion. It is necessary to transform the measurements to a unified spatio-temporal reference frame through calibration. If directly fusing data without temporal and spatial calibration, the positional estimation results would have more large errors, making the fusion less reliable. Hence, temporal and spatial calibration for heterogeneous information is necessary for data fusion.

A. Overview of Related Works and Problems

Space-time calibration has always been a research topic of great concern to researchers. However, there are several challenges in the current research on temporal and spatial registration of heterogeneous information.

Firstly, there is a wealth of literatures on specific sensor combinations such as Vision and LiDAR [1]–[7], Vision and Inertial [8], [9], LiDAR and LiDAR [10], Vision/LiDAR/Inertial [11], Vision/GPS/Inertial [12], GNSS and IMU [13], etc. These calibration algorithms are designed for specific combinations and have certain effects in determining particular fusion systems. In recent studies, language models have been used to solve real-world route optimization problems [14], [15], demonstrating their potential in handling complex data patterns. This approach can also be applied to multi-sensor data calibration by learning from historical data and implicit patterns to improve calibration accuracy. Drawing on the combined approach of sliding mode control and neural network estimators [16], we can introduce a more robust compensation mechanism in multi-sensor calibration to enhance the system's accuracy and stability. However, when new sensors are added, these calibration models become ineffective.

Secondly, some existing temporal and spatial registration methods are highly dependent on specific calibration targets or scenes, which severely limits their applicability. [17] requires spherical objects in the scene. In [18]–[21], spatial parameters are solved by using a checkerboard by combining 3D lines and plane correspondences with a minimal number of poses. [10] needs orthogonal planes or nearly orthogonal planes. The calibration method for the IMU in [22] requires the assistance of a moving horizon scheme to estimate parameters. The application environment of autonomous vehicle systems is extremely complex and variable, so the required scenarios for

This research is supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research & Development Programme (FCP-NUS-RG-2022-018). This research is also supported in part by National Natural Science Foundation of China under Grant 62171735, 62173276, 62101458, 62001392, 61803310 and 61801394, in part by the Natural Science Basic Research Program of Shaanxi under Grant 2022GY-097,2021JQ-122 and 2021JQ-693, and in part by China Postdoctoral Science Foundation under Grant 2020M673482 and 2020M673485. The author would like to express gratitude for the support and sponsorship provided by the China Scholarship Council (Liu Jin Xuan [2021]70).

1 Northwestern Polytechnical University, China, Shaanxi, 710072 (e-mail: syf1819285@mail.nwpu.edu.cn; bwlian@nwpu.edu.cn; yyliu@nwpu.edu.cn).

2 Institute for Infocomm Research, Agency for Science, Technology and Research, Singapore 138632, (e-mail: yhzeng@i2r.a-star.edu.sg; mayg@i2r.a-star.edu.sg).

calibration algorithms may not always be present. As a result, existing calibration algorithms may not be effective.

Thirdly, most existing temporal and spatial registration methods only consider spatial or time parameters, overlooking the significant issue of time offset or the coordinate difference between different sensor data. [1], [4], [6], [7], [10], [18]–[21], [23], [24] only has spatial calibration and ignores temporal calibration. [8] proposes an online time offset calibration model based on filtering, which only estimates the temporal parameters, ignoring the spatial parameters. In high-precision autonomous vehicle systems, an increasing amount of sensor data participates in fusion operations. The timing of different sensor data entering the central processor will inevitably have discrepancies, leading to time errors. Additionally, different sensors are positioned differently on the autonomous vehicle, and each sensor's output data is in its own coordinate system. Therefore, determining the spatial parameters between various sensor data is essential. Consequently, algorithms that only consider calculating one type of calibration parameter cannot meet the requirements of high-precision vehicle systems. Paper [25] proposes a spatio-temporal calibration method for LVLVS (Line Structured Light Vision Sensor). The first step is to calculate the spatial parameters. Then, on the premise of solving the spatial parameters, the time parameters are obtained through the reprojection error. Howerer, this method of calculating spatio-temporal parameters by stages will lead to the accumulation of errors and eventually affect the accuracy of spatio-temporal parameters.

Fourthly, and importantly, GNSS positioning, as an absolute positioning method, can be used for fusion and trajectory drift correction [26]. However, the primary prerequisite for GNSS raw data [27] to be involved in fusion is to address the temporal and spatial parameters. Unfortunately, there is not a particularly large amount of literatures on the calibration of GNSS raw data and other sensor data. Therefore, it is very important to choose a suitable temporal and spatial calibration technique for GNSS raw data.

B. Our Contributions

Inspired by [28], we propose the spatiotemporal calibration algorithm for GNSS data, LiDAR data and visual data. We use spline functions to interpolate the IMU output values, making them continuous. Then, combining other sensor data with the IMU output values and based on their positioning principles, we derive an error model. This error models are then used as an optimization factors in nonlinear optimization operation, thereby solving for the spatiotemporal calibration parameters. The key contributions are:

- We utilize GNSS pseudorange information and IMU output values to construct the spatiotemporal calibration model for GNSS data, and the proposed model relies on data from only one GNSS satellite, making it more adaptable in environments where GNSS satellite signals are not favorable.
- We utilize the curvature information of LiDAR points cloud to extract corner points and surface points, and construct the spatiotemporal calibration model for LiDAR point clouds based on the fact that the distances from points to lines or surfaces are the same under different coordinate system.
- We utilize the ORB (Oriented FAST and Rotated BRIEF) feature extraction algorithm, feature matching algorithm, and visual reprojection principle to construct the spatiotemporal calibration model for visual data.

We conduct extensive simulations based on simulated data and publicly available real-world datasets. The simulation results show that the combination of the proposed calibration models achieves higher precision in spatiotemporal parameters than existing calibration algorithms. Furthermore, when applying the proposed calibration algorithm to some existing open-source odometry systems, we analyze the performance of the proposed calibration model in terms of position estimation accuracy, maximum error, reliability, and computational efficiency. The experimental results fully demonstrate that the proposed calibration model can improve position estimation accuracy by approximately 40% while ensuring good safety and reliability under high computational efficiency.

II. Problem Formulation

Due to different preprocessing and initialization, each kind of sensor data consumes different times, so the time they arrive at the fusion module is generally different. It is crucial to identify this bias for heterogeneous information fusion, which is time calibration. On the other hand, the measurement of each sensor is relative to its coordinate system, so the data needs to be converted to the same coordinate system. Finding the transformation parameters between different coordinate systems is spatial calibration.

Overall, the proposed method uses the IMU coordinate as the central coordinate, assuming each sensor's time delay and conversion parameters relative to the IMU coordinate in advance. We use the temporal and spatial parameters to predict the measured values at a certain time in the future. Then, we subtract the actual measurement from the predicted measurement to get an error. Finally, we use the nonlinear optimization algorithm to find the minimum value of the error to get the time and space parameters. In this way, we can estimate time and space parameters within the theoretical framework of maximum likelihood estimation.

On the one hand, regarding the time parameter, inspired by the paper [29] published by Joern Rehder in 2016, we express the time delay of other sensor data relative to IMU as a variable quantity. Assuming the time delay is τ, we can describe the error as Eq. (1):

$$e_i = f_t - h_{p,(t+\tau_i)} \tag{1}$$

where, f_t is the actual measurement at timestamp t, $h_{p,(t+\tau_i)}$ is the predicted measurement at the moment t. The predicting model varies with the sensor. τ_i represents the time delay of different sensor data, in the following sections: τ_g represents the time delay of GNSS data, τ_l represents the time delay of LiDAR data, and τ_c represents the time delay of visual data. e_i represents the error of different sensor data.

On the other hand, regarding the spatial parameters, the transformation matrix $\mathbf{T}_{i,b}(t)$ from other coordinate systems i to the IMU coordinate system b is given by Eq. (2).

$$\mathbf{T}_{i,b}(t) = \begin{bmatrix} \boldsymbol{R}_{i,b}(t) & \boldsymbol{\Upsilon}_{i,b}(t) \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2}$$

where, $\boldsymbol{R}_{i,b}(t)$ and $\boldsymbol{\Upsilon}_{i,b}(t)$ respectively represent the 3×3 rotation matrix and the 3×1 translation vector from the coordinate system i to the coordinate system b.

The output frequency of the IMU data, usually around 200 Hz, is much higher than that of other sensor data, such as the camera and LiDAR data (around 10 Hz), the GNSS data (around 1 Hz). The schematic diagram of their relationship is shown in Fig. 1. N_b denotes the number of the inertial data, N_c represents the visual data, N_l represents the LiDAR data, and N_g represents the GNSS data. Here, we adopt the idea of the paper [29] to represent the data of IMU by a B-spline function of sixth-order, which is a piecewise fifth-degree polynomial. In this way, it avoids the complex operation and the error caused by pre-integration and can calculate the time and space parameters at any time.

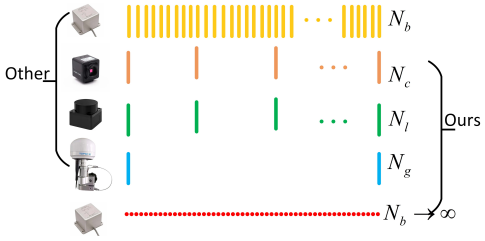


Fig. 1. Comparison diagram of the relationship of the four sensors in our paper and those in other studies. In other studies, the key difference lies in the frequency of the output data, while in our paper the output frequency of IMU is expressed by a set of analytic basis functions, which can predict the IMU output data at any time

Therefore, the Eq. (2) can be expressed as Eq. (3):

$$\mathbf{T}_{i,b}(t) = \begin{bmatrix} \boldsymbol{\Gamma}_{i,b}(t) & \boldsymbol{\Theta}_{i,b}(t) \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{3}$$

where $\boldsymbol{\Gamma}_{i,b}(t) = \boldsymbol{\Gamma}_{i,b}\{\boldsymbol{\phi}(t)\boldsymbol{\eta}_{(i,\varphi)}\}$, $\boldsymbol{\Theta}_{i,b}(t) = \boldsymbol{\phi}(t)\boldsymbol{\chi}_{(i,\nu)}$, $\boldsymbol{\Gamma}(\cdot)$ is a function that transforms the orientation parameters to the rotation matrix and $\boldsymbol{\phi}(t)$ is a set of B-spline functions. $\boldsymbol{\eta}_{(i,\varphi)}, \boldsymbol{\chi}_{(i,\nu)}$ are the coefficient vectors of the spline function $\boldsymbol{\phi}(t)$ regarding sensor data i. Then, finding the spatial-temporal parameters from the i coordinate system to the IMU coordinate system turns into finding $\boldsymbol{\eta}_{(i,\varphi)}$ and $\boldsymbol{\chi}_{(i,\nu)}$, where t includes time delay τ.

In nonlinear optimization, the design and selection of the residual block are crucial. Well-designed residual blocks can provide good numerical stability and convergence performance while capturing the critical differences between the estimated and observed values. Therefore, The first key to the proposed spatiotemporal calibration model is to design a predictive model for sensor data, subtract it from the actual measurements, and then obtain the residual module. The second key is to utilize nonlinear optimization methods to solve for $\boldsymbol{\eta}_{(i,\varphi)}$ and $\boldsymbol{\chi}_{(i,\nu)}$. Here, it is essential to clarify that we utilize the homogeneous coordinates system, as shown in Eq. (4).

$$\boldsymbol{P}_k = \begin{bmatrix} x_b & y_b & z_b & 1 \end{bmatrix}^\top \tag{4}$$

III. Error Model And Spatiotemporal Calibration

Based on the above analysis, the design of the residual block plays an important role in the accuracy of the time and space parameters. In the following, we design the residual blocks of GNSS, LiDAR and visual data relative to the IMU data.

A. Coordinate System And Transformation

The celestial coordinate systems involved in the paper and the relationship between them are shown in Fig. 2. The specific orientations of their coordinate axes and the conversion between them can be referenced in [30]. It should be noted, different from [30], we define that the ECEF (Earth-Centered, Earth-Fixed) (F) coordinate system and the ECI (Earth-Centered Inertial) (I) coordinate system coincide at the moment when the receiver receives the satellite signal. The

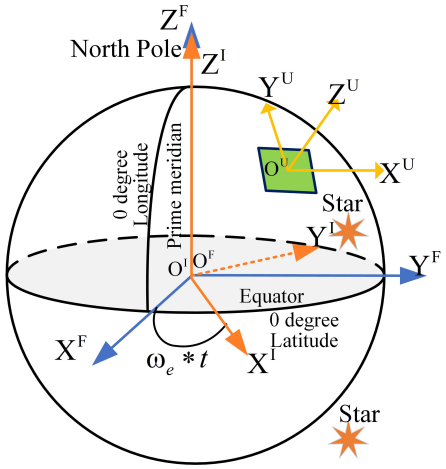


Fig. 2. Schematic diagram of the relationship between the coordinate system I, the coordinate system F and the coordinate system U and the specific orientation of their coordinate axes

sensor coordinate systems and the specific orientations of the coordinate axes mentioned in the paper are shown in Fig. 3. The carrier coordinate takes the center of the carrier as the origin. The rotation between the carrier coordinate system and the local coordinate system ENU (East-North-Up) (U) , is the carrier's attitude, namely, roll, pitch, and yaw. In this paper, we put the IMU sensor in the center of the carrier, so the IMU coordinate system and the carrier coordinate system are coincidental. In our system, we take the U coordinate system as the world coordinate system, and at time $t = 0$, the carrier coordinate system, the Inertial coordinate system, and the world coordinate system coincide. As the carrier moves, the carrier coordinate system and the Inertial coordinate system b move together, and the world coordinate system remains stationary.

B. Error Models Of Different Sensor Systems

In the following, we take GNSS, LiDAR, and visual data as examples to elaborate the construction of their residual models. The diagram is shown in Fig. 4.

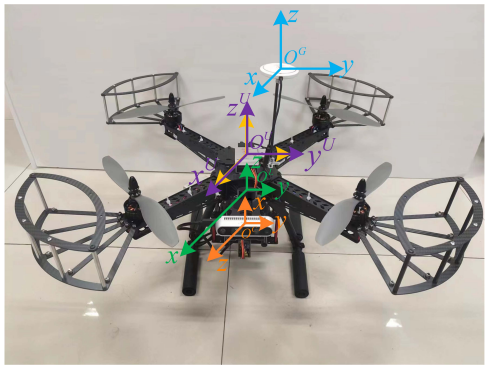


Fig. 3. Schematic diagram of the carrier coordinate system, the IMU coordinate system, the GNSS coordinate system, the LiDAR coordinate system and the camera coordinate system and the specific direction of their axes

1) The processing of the IMU data: We adopt the output model in [31], which includes the influence of gravity on acceleration, as illustrated in Eq. (5) and Eq. (6).

$$\widetilde{\boldsymbol{\omega}} = \boldsymbol{\omega} + \iota_1 + \upsilon_1 \tag{5}$$

$$\widetilde{\boldsymbol{a}} = \boldsymbol{q}_{bw}(\boldsymbol{a} + \vartheta) + \iota_2 + \upsilon_2 \tag{6}$$

$\boldsymbol{\omega}$ and $\boldsymbol{a}$ are the true values. $\widetilde{\boldsymbol{\omega}}$ and $\widetilde{\boldsymbol{a}}$ are the measured values. ι_1 and ι_2 are the deviations of acceleration and angular velocity, and their derivatives follow the Gaussian distribution. υ_1 and υ_2 are Gaussian white noise for acceleration and angular velocity, respectively. ϑ represents gravitational acceleration. $\boldsymbol{q}_{bw}$ represents the quaternion transformation from the Inertial coordinate system to the world coordinate system.

We can obtain the position $\boldsymbol{P}_b^I$ of the IMU sensor by adopting the algorithm in [32] combining the linear acceleration and angular velocity. Then, we use B-spline functions to interpolate the trajectory, making the trajectory continuous. The B-spline functions and interpolation schematics are shown in Fig. 5(a) and Fig. 5(b), respectively.

2) Error model of the GNSS raw data: Fig. 6 shows the schematic construction of the optimization factor for GNSS data. On the one hand, the receiver can obtain the travel time of the signal from the satellite to the receiver by calculating the number of pseudo-code offsets [33] through the correlation peaks of the original code and the received code, then multiplied by the speed of light σ, which we use as the actual measurement. As shown in Eq. (7), N_1 is the number of code. ρ represents the time for each code chip. These are functions of the GNSS receiver, not the focal point of our discussion, so we won't go into detailed elaboration.

$$D_f = \rho \cdot N_1 \cdot \sigma \tag{7}$$

Pseudo-range is not the accurate geometric distance, which includes a variety of errors generated during signal generation, propagation, and processing [34]. The errors at the satellite signal generation end mainly come from orbit errors and clock errors. Orbital errors come mainly from the influence of other stars, which cannot be accurately modeled in ephemeris, and the clock error is the error between the atomic clock on the satellite and the standard system time, which is monitored by the system control module and continuously corrected.

As the signal travels from the satellite to the receiver, it passes through the ionosphere and troposphere; the propagation speed of the signal is no longer the propagation speed in a vacuum. The signal will be delayed by the composition of the atmosphere and the path it travels. Satellite signals take different paths to reach the receiver, and this phenomenon is called the multipath effect. When the satellite signal arrives, the propagation time is obtained by comparing the emission time annotated by the satellite atomic clock with the less accurate local clock at the receiver. According to [34], there is Eq. (8):

$$D_p = \|\boldsymbol{P}_s^I - \boldsymbol{P}_r^I\| + c(\delta_r - \delta_s) + T_r^s + I_r^s + M_r^s + \varepsilon_r^s \tag{8}$$

T_r^s, I_r^s, M_r^s, ε_r^s are the effects of the troposphere, ionosphere, multipath effect and white Gaussian noise on pseudo-range, respectively, here represented by distance. δ_r and δ_s respectively represent the clock errors of the GNSS receiver and the satellite. $\boldsymbol{P}_s^I$, $\boldsymbol{P}_r^I$ denote the ECI coordinate values of the satellite and the receiver at the moment when the satellite signal arrives at the receiver, respectively. [30] provides a more detailed explanation of the GNSS receiver.

We multiply the coordinate values of the Inertial sensors $\boldsymbol{P}_b^I$ in the ECI coordinate system by the spatiotemporal parameters $\mathbf{T}_{b,g}(t+\tau_g)$ between the Inertial sensor and the GNSS receiver, and this yields the coordinate values of GNSS receiver $\boldsymbol{P}_r^I$ in the ECI coordinate system. As shown in Eq. (9).

$$\boldsymbol{P}_r^I = \mathbf{T}_{b,g}(t+\tau_g)\boldsymbol{P}_b^I \tag{9}$$

$\boldsymbol{P}_s^I$ is the ECI coordinate of the satellite when the receiver picks up the satellite signal. We define the ECI coordinate system coincide with the ECEF coordinate system when the receiver receives the signal. So, the ECI coordinates of the satellite at the time of receiving the satellite signal are the same as the ECEF coordinates, i.e., $\boldsymbol{P}_s^I = \boldsymbol{P}_s^F$. The ECEF coordinates of the satellite when generating the signal, $\boldsymbol{P}_{s,1}^F$, which is included in the ephemeris. However, due to the rotation of the Earth, the ECEF coordinates of the satellite at the reception time are not equal to that at the emission time, that is, $\boldsymbol{P}_{s,1}^F \neq \boldsymbol{P}_s^F$.

From Fig. 2, it can be observed that the satellite's ECEF coordinates at the transmission time and reception time differ by an angle corresponding to the Earth's rotation. The angular velocity of the Earth's rotation ω_e is known, and we have already obtained the satellite signal propagation time from ρN_1. Therefore, we can derive the angle of Earth's rotation, $\theta = \omega_e \rho N_1$. Thus, the ECI coordinate of the satellite at the receiving time can be obtained by the following Eq. (10):

$$\boldsymbol{P}_s^F = \mathbf{R}_z(\theta)\boldsymbol{P}_{s,1}^F \tag{10}$$

where $\mathbf{R}_z(\theta)$ represents a rotation about the z axis of the ECEF frame with magnitude θ.

Therefore, for a set of GNSS raw data, the error model is as shown in Eq. (11):

$$\begin{aligned} e_g &= D_p - D_f \\ &= \|\boldsymbol{P}_s^I - \boldsymbol{P}_r^I\| + c(\delta_r - \delta_s) + T_r^s + I_r^s + M_r^s + \varepsilon_r^s \\ &\quad - \rho N_1 c \\ &= \|\mathbf{R}_z(\omega_e(-\rho N_1))R_F^I \boldsymbol{P}_{s,1}^F - \mathbf{T}_{b,g}(t+\tau_g)\boldsymbol{P}_b^I\| \\ &\quad + c(\delta_r - \delta_s) + T_r^s + I_r^s + M_r^s + \varepsilon_r^s - \rho N_1 c \end{aligned} \tag{11}$$

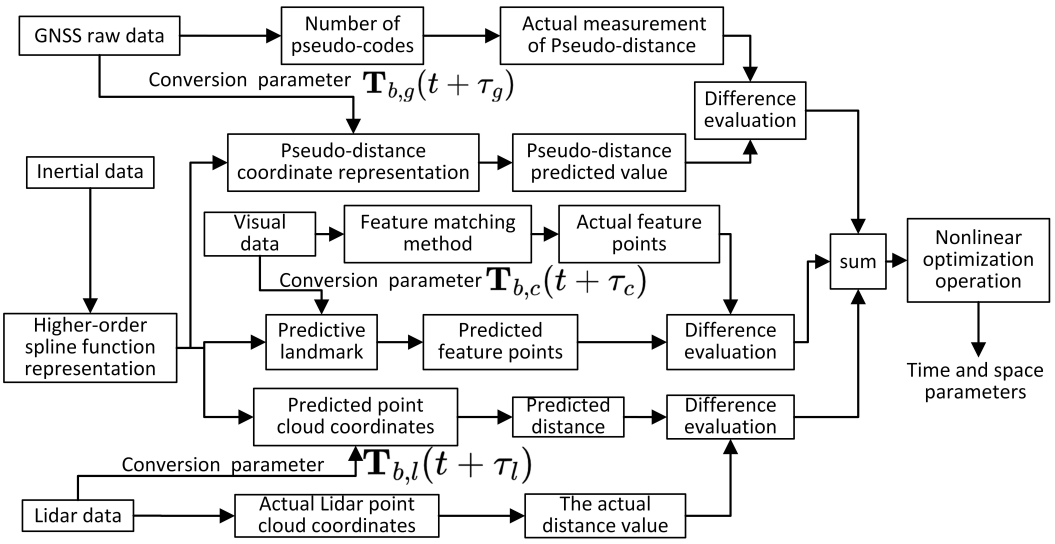
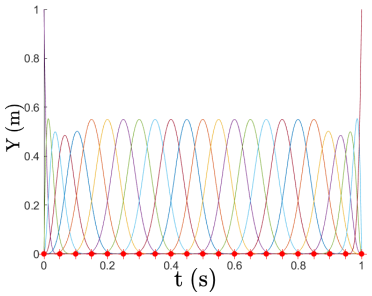
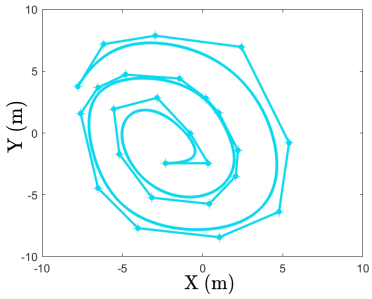


Fig. 4. Schematic diagram of the calibration process of GNSS data, Visual data, LiDAR data, and Inertial data



(a)



(b)

Fig. 5. (a) B-spline functions; (b) Control points and the interpolated trajectory

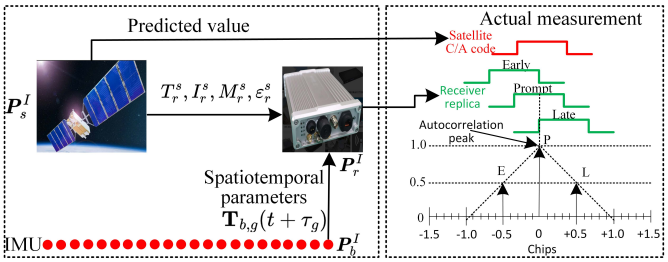


Fig. 6. Schematic diagram of optimization factor construction for spatiotemporal calibration parameters of GNSS raw data

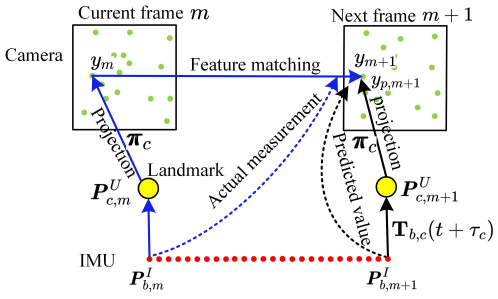


Fig. 7. Schematic diagram of the construction of optimization factor for spatiotemporal calibration parameters of visual data

For N_g sets of GNSS raw data, we obtain the average error, so the final error model is expressed as Eq. (12):

$$\overline{e_g} = \frac{1}{N_g}\sum_{\alpha=1}^{N_g}|D_{p,\alpha} - D_{f,\alpha}| \tag{12}$$

3) Error Model Of Visual Data: The construction of the optimization factor of visual data is shown in Fig. 7.

For visual data, what can be measured directly is the coordinates of pixels on the image at different times in the camera coordinate system. In addition, the ECI coordinates of the Inertial sensor are known, such as $\boldsymbol{P}^I_{b,m+1}$ and $\boldsymbol{P}^I_{b,m}$. As mentioned earlier, the ECEF coordinate system and the ECI coordinate system coincide at the moment of GNSS signal reception. Therefore, the coordinate values in the ECEF coordinate system of the Inertial sensor at the reception moment are equal to the coordinate values in the ECI coordinate system, denoted as $\boldsymbol{P}^I_{b,m+1} = \boldsymbol{P}^F_{b,m+1}$ and $\boldsymbol{P}^I_{b,m} = \boldsymbol{P}^F_{b,m}$. Then, we apply Eq. (13) to convert the values of the Inertial sensor from the ECEF coordinate system to the ENU coordinate system.

$$\boldsymbol{P}^U_{b,m} = \mathbf{R}^U_F \boldsymbol{P}^I_{b,m} \tag{13}$$

Where $\mathbf{R}_F^U$ represents the transformation from the ECEF coordinate system to the ENU coordinate system [30], as shown in Eq. (14).

$$\mathbf{R}_F^U = \begin{bmatrix} -\sin\varrho_1 & -\sin\varrho_2\cos\varrho_1 & \cos\varrho_2\cos\varrho_1 \\ \cos\varrho_1 & -\sin\varrho_2\sin\varrho_1 & \cos\varrho_2\sin\varrho_1 \\ 0 & \cos\varrho_2 & \sin\varrho_2 \end{bmatrix} \tag{14}$$

ϱ_1 and ϱ_2 denote the longitude and latitude of the reference point, respectively.

Then, by multiplying with the spatiotemporal parameters $\mathbf{T}_{b,c}(t+\tau_c)$ between the Inertial sensor and the camera, we obtain the coordinates of the camera in the world coordinate system $\boldsymbol{P}_{c,m}^U$, as shown in Eq. (15).

$$\boldsymbol{P}_{c,m}^U = \mathbf{T}_{b,c}(t+\tau_c)\boldsymbol{P}_{b,m}^U \tag{15}$$

We map this landmark point $\boldsymbol{P}_{c,m}^U$ to the pixel plane at the time m through projection to find out the corresponding pixel coordinates, as shown in Eq. (16) and $\boldsymbol{\pi}_c$ is the projection function of the camera.

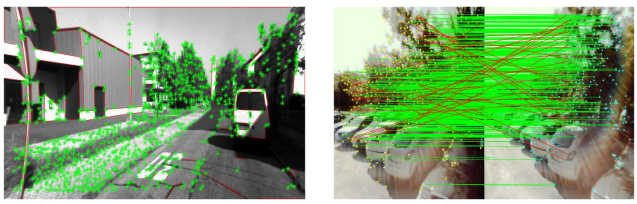
$$y_m = \boldsymbol{\pi}_c(\mathbf{T}_{b,c}(t+\tau_c)\boldsymbol{P}_{b,m}^U) \tag{16}$$

Then, we find the corresponding point of the current frame m on the next frame $m+1$ by the feature points matching algorithm and define this point as the actual measurement, as shown in Eq. (17).

$$y_{f,m+1} \Leftrightarrow \text{FeatureMatching}(y_m) \tag{17}$$

This paper first extracts ORB (Oriented FAST and Rotated BRIEF) feature points [35] from visual images. Only when the landmark point is projected to pixels corresponding to ORB feature points is it selected to participate in the reprojection process and feature-matching algorithm. ORB feature points combine two algorithms: FAST (Features from Accelerated Segment Test) [36] and BRIEF (Binary Robust Independent Elementary Features) [37].

ORB feature points locate areas of high-intensity variation in the image, and the generated descriptors grant ORB feature points rotational invariance, enabling them to maintain good performance even when the image undergoes rotation, which makes them highly effective in feature matching. Fig. 8(a) shows the ORB feature extraction results for KITTI 00 [38] sequence. As shown in Fig. 8(b), this is the matching results of ORB feature points.



(a) (b)

Fig. 8. (a) ORB feature point extraction results of a specific image of the KITTI dataset; (b) ORB feature points matching results of different images at different times of the Malaga dataset for the same place

We can obtain the predicted world coordinates of the camera at the $m+1$ frame, as shown in Eq. (18).

$$\boldsymbol{P}_{c,m+1}^U = \mathbf{T}_{b,c}(t+\tau_c)\boldsymbol{P}_{b,m+1}^U \tag{18}$$

The predicted pixels of the predicted landmark points on the $m+1$ frame can be obtained through the projection algorithm. As shown in Eq. (19):

$$\begin{aligned} y_{p,m+1} &= \boldsymbol{\pi}_c(\boldsymbol{P}_{c,m+1}^U) \\ &= \boldsymbol{\pi}_c(\mathbf{T}_{b,c}(t+\tau_c)\boldsymbol{P}_{b,m+1}^U) \end{aligned} \tag{19}$$

Therefore, for a single ORB feature point, the error is as Eq. (20):

$$\begin{aligned} e_c &= y_{p,m+1} - y_{f,m+1} \\ &= \boldsymbol{\pi}_c(\mathbf{T}_{b,c}(t+\tau_c)\boldsymbol{P}_{b,m+1}^U) \\ &\quad - \text{FeatureMatching}(\boldsymbol{\pi}_c(\mathbf{T}_{b,c}(t+\tau_c)\boldsymbol{P}_{b,m}^U)) \end{aligned} \tag{20}$$

If there are N_c sets of successfully matched ORB feature point pairs on the $m+1$ frame and the m frame, the error model of visual data is as shown in Eq. (21):

$$\overline{e_c} = \frac{1}{N_c}\sum_{\beta=1}^{N_c} |y_{p,\beta} - y_{f,\beta}| \tag{21}$$

4) Error Model Of LiDAR Points Cloud: For the LiDAR points cloud, what is already known is the position of the scanning points at the scanning time. For example, the points cloud representation of malaga2009_campus_2L is shown in Fig. 9, Fig. 10, Fig. 11 and Fig. 12 respectively, which depict the LiDAR points cloud in three-dimensional display, two-dimensional display, and two-dimensional depth display. We

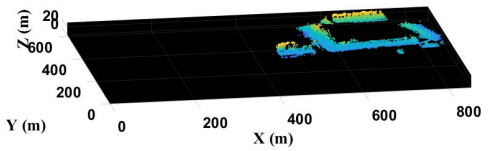


Fig. 9. Points cloud representation of the LiDAR points cloud for malaga2009_campus_2L in the Malaga dataset

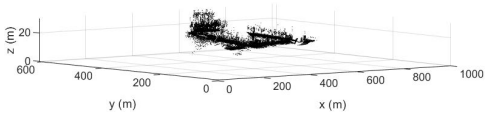


Fig. 10. Three-dimensional visualization of the LiDAR points cloud for malaga2009_campus_2L in the Malaga dataset

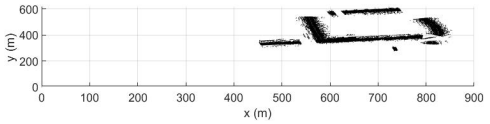


Fig. 11. Two-dimensional visualization of the LiDAR points cloud for malaga2009_campus_2L in the Malaga dataset

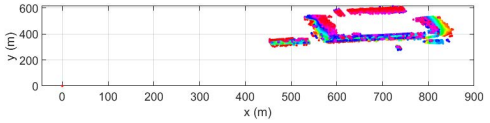


Fig. 12. Two-dimensional depth visualization of the LiDAR points cloud for malaga2009_campus_2L in the Malaga dataset

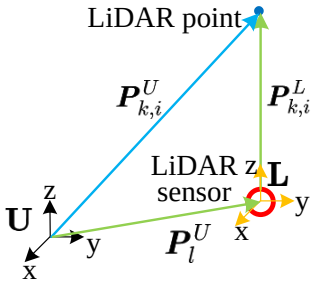


Fig. 13. Illustration of the relationship between the LiDAR sensor coordinates and the points cloud coordinates

employ the method [39] to calculate the curvature of the points cloud, as shown in Eq. (22).

$$\Omega = \frac{1}{|S|\|\boldsymbol{P}^L_{k,i}\|}\| \sum_{j\in S, j\neq i} (\boldsymbol{P}^L_{k,i} - \boldsymbol{P}^L_{k,j})\| \tag{22}$$

Ω denotes the curvature value of the i point on the k beam. $|S|$ is the set of consecutive points of i returned by the laser scanner in the same scan. Here, $\boldsymbol{P}^L_{k,i}$ represents the three-dimensional coordinates of the points cloud in the LiDAR coordinate system. The points are divided into surface and corner points according to their curvature values. We then compute the distance of the points to the previous scan plane or line. There are different distance calculation formulas for corner points and surface points [39], as shown in Eq. (23) and Eq. (24), respectively. If it is a corner point, the distance is from the point to the line; Otherwise, it refers to the distance to the surface. We refer to this distance as the actual measurements.

$$d_\varepsilon = \frac{|(\boldsymbol{P}^L_{k,n} - \boldsymbol{P}^L_{k-1,u}) \times (\boldsymbol{P}^L_{k,n} - \boldsymbol{P}^L_{k-1,v})|}{|(\boldsymbol{P}^L_{k-1,u} - \boldsymbol{P}^L_{k-1,v})|} \tag{23}$$

$$d_\zeta = \frac{\left|(\boldsymbol{P}^L_{k,n} - \boldsymbol{P}^L_{k-1,v}) \times (\boldsymbol{P}^L_{k-1,u} - \boldsymbol{P}^L_{k-1,w})\right|}{|(\boldsymbol{P}^L_{k-1,u} - \boldsymbol{P}^L_{k-1,v}) \times (\boldsymbol{P}^L_{k-1,u} - \boldsymbol{P}^L_{k-1,w})|} \tag{24}$$

The distance should be constant in different coordinate systems. We use the symbol $d_\varepsilon(\boldsymbol{P}^L_{k,n})$ or $d_\zeta(\boldsymbol{P}^L_{k,n})$ to denote the distance, as shown in Eq. (25). We refer to them as the actual distance measurements.

$$d_{f-c} = d_\varepsilon(\boldsymbol{P}^L_{k,n}), d_{f-s} = d_\zeta(\boldsymbol{P}^L_{k,n}) \tag{25}$$

The position of the IMU at the next scanning time is known, and the position of the LiDAR sensor in the world coordinate system at the next time can be obtained by the spatiotemporal parameters $\mathbf{T}_{b,l}(t+\tau_l)$ from the LiDAR to the IMU, as shown in Eq. (26).

$$\boldsymbol{P}^U_l = \mathbf{T}_{b,l}(t+\tau_l)\mathbf{R}^U_F\boldsymbol{P}^I_b \tag{26}$$

Now, we need to obtain the predicted position of the LiDAR points cloud at time k based on the predicted position of the LiDAR sensor in the world coordinate system. Fig. 13 illustrates the relationship between the LiDAR sensor position and the points cloud position.

In the LiDAR coordinate system, the coordinate of the LiDAR sensor is at the origin. Therefore, the LiDAR points cloud coordinates in the LiDAR coordinate system represent the offset of the points cloud relative to the LiDAR sensor. So, the coordinates of the LiDAR points cloud in the world coordinate system are obtained by adding the coordinates of the LiDAR sensor and the coordinates of the LiDAR points cloud, as shown in Eq. (27).

$$\boldsymbol{P}^U_{k,i} = \boldsymbol{P}^L_{k,i} + \boldsymbol{P}^U_l \tag{27}$$

Combining with Eq. (26), the coordinates of the LiDAR points cloud in the world coordinate system at time k is as shown in Eq. (28).

$$\boldsymbol{P}^U_{k,i} = \boldsymbol{P}^L_{k,i} + \mathbf{T}_{b,l}(t+d_l)\mathbf{R}^U_F\boldsymbol{P}^I_b \tag{28}$$

Then we plug this predicted position into the distance calculation formulas to get the predicted distance $d_p = d_\varepsilon(\boldsymbol{P}^U_{k,i})$ or $d_p = d_\zeta(\boldsymbol{P}^U_{k,i})$. Therefore, the error model of the LiDAR data is shown in Eq. (29).

$$\overline{e_l} = \frac{1}{N_l}\sum_{\kappa=1}^{N_l} |d_{p,\kappa} - d_{f,\kappa}| \tag{29}$$

Where N_l represents the number of points.

C. Spatiotemporal Calibration With Nonlinear Optimization

According to the above analysis, the overall cost function is as shown in Eq. (30):

$$\begin{aligned}\boldsymbol{\Psi} = &\frac{1}{N_g}\sum_{\alpha=1}^{N_g} |D_{p,\alpha} - D_{f,\alpha}| + \frac{1}{N_c}\sum_{\beta=1}^{N_c} \omega_\beta |y_{p,\beta} - y_{f,\beta}| \\ &+ \frac{1}{N_l}\sum_{\kappa=1}^{N_l} |d_{p,\kappa} - d_{f,\kappa}|\end{aligned} \tag{30}$$

Please note that the term ω_β represents the weight. This weight balances the influence of the other two terms, distance errors, with the term representing coordinate errors. We aim to ensure that these terms similarly impact the optimization variables. This weight is obtained through adaptation. The variables need to be optimized include: $\mathbf{T}_{b,l}(t+\tau_l)$, $\mathbf{T}_{b,c}(t+\tau_c)$, $\mathbf{T}_{b,g}(t+\tau_g)$. According to Eq. (3), we can transform the problem into solving: $\boldsymbol{\eta}_{(l,\varphi)}$, $\boldsymbol{\chi}_{(l,\nu)}$, $\boldsymbol{\eta}_{(c,\varphi)}$, $\boldsymbol{\chi}_{(c,\nu)}$, $\boldsymbol{\eta}_{(g,\varphi)}$, $\boldsymbol{\chi}_{(g,\nu)}$, τ_l, τ_c, τ_g. Since the cost function contains three types of errors, each with multiple instances, it is inevitable to encounter numerous outliers. These outliers can significantly affect the accuracy of the final optimized variables. Therefore, we employ the loss kernel function to handle these outliers, aiming to eliminate their influence on the final optimized results.

Then, the most crucial step is to calculate the iteration step length. The Gauss-Newton method utilizes the Eq. (31) to calculate the iteration step size.

$$(\mathbf{H} + \lambda\mathbf{I})\triangle X = \mathbf{g} \tag{31}$$

Here, $\mathbf{H}$ represents the second derivative, and $\mathbf{g}$ represents the first-order derivative of the cost function. The range of the value for λ is from 0.1 to 0.5. Here, we present the process of solving the spatiotemporal parameters of GNSS raw data. The same approach can be applied to other sensor data. According to the cost function, seven variables should be optimized. We choose the cost function's first and second derivatives that incorporate the seven iteration step sizes. When taking the derivative of the cost function concerning one iteration step size, the other six iteration step sizes are treated as known values. This way, we can obtain seven equations involving the seven iteration steps. As shown in Eq. (32).

$$(\frac{\partial^2\boldsymbol{\Psi}}{\partial^2\triangle x_i} + \lambda\mathbf{I})\triangle x_i = \frac{\partial\boldsymbol{\Psi}}{\partial\triangle x_i} \quad (i = 1, 2, \ldots, 7) \tag{32}$$

The value x_1, x_2, x_3, x_4, x_5, x_6, x_7 represents the three dimensions of vectors $\boldsymbol{\eta}_{(g,\varphi)}$ and $\boldsymbol{\chi}_{(g,\nu)}$ respectively along with a time deviation τ_g. $\frac{\partial^2 \boldsymbol{\Psi}}{\partial^2 \triangle x_i}$ is the second derivative of the cost function $\boldsymbol{\Psi}$, denoted as $\mathbf{H}$ in Eq. (31) and $\frac{\partial \boldsymbol{\Psi}}{\partial \triangle x_i}$ is the first derivative of the cost function $\boldsymbol{\Psi}$, denoted as $\mathbf{g}$ in Eq. (31). $\frac{\partial^2 \boldsymbol{\Psi}}{\partial^2 \triangle x_1}$ is the second derivative of the cost function when treating $\triangle x_1$ as an unknown variable and $(\triangle x_2, \ldots, \triangle x_7)$ as known variables. The other equations are similar. In this way, we can obtain seven equations that involve $(\triangle x_1, \ldots, \triangle x_7)$. By simultaneously solving these seven equations, we can determine the specific value of $(\triangle x_1, \ldots, \triangle x_7)$. Then, we obtain the next variable value by combining this step size with the previous value. We substitute this value into the cost function to evaluate whether the obtained result satisfies the convergence criteria, as shown in Eq. (33).

$$\|\boldsymbol{\Psi}\| \leq 10^{-9} \tag{33}$$

If the convergence condition is satisfied, we stop the computation. Otherwise, we continue calculating the step size and iterating. In the optimization, considering the computational time constraints in practical scenarios, we set the maximum number of iterations to 100. If, after 100 iterations, the final optimized result still does not meet the requirements of the convergence criteria, we will reset the initial values and restart the iteration again.

Finally, when the convergence criterion is met, the optimized variables and the last iteration's step size constitute our final optimization result. This is because when calculating the step size, we use the derivative of the cost function with the addition of the step size.

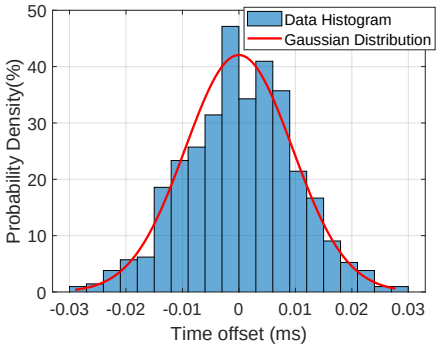
IV. Model Verification

In this section, we evaluate the proposed calibration model from two aspects. Firstly, we assess the accuracy and uncertainty of the spatiotemporal parameters. Secondly, we examine the robustness against IMU noise.

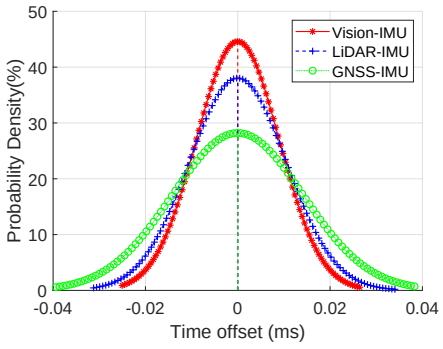
A. Accuracy Of The Spatiotemporal Parameter Model

The simulated sensor setup consists of one monocular camera and one LiDAR sensor with 16 scan lines, an Inertial sensor, and a GNSS receiver. The camera has a resolution of 1600 by 1200 and a focal length of 1032 px. The input frequency settings for Visual, GNSS, and LiDAR data are 20 Hz, 1 Hz, and 10 Hz, respectively. We represent the position curve of IMU by using 50 basis functions per second [40]. All visual feature points add white noise with a standard deviation of 0.5 pixel. We use the method in [41] to obtain the intrinsic parameter of the camera. Here, we use the algorithm in Toolbox[1] to calculate the approximate position of each image as our initial values.

Here, we use the GPS satellite data from reference [42] as the actual GNSS raw data. We use points cloud recorded from an actual LiDAR sensor for the ranging data. Then, we interpolate these points cloud to get the data used in the simulation. The IMU and camera are rigidly connected, and we adopt the Internal noise parameters of IEEE Aerospace and Electronic Systems Society analysis [31] attached to the accelerometer and gyroscope, respectively. First, we preset the time deviation, translation parameter, and rotation parameter of the camera, LiDAR, and GNSS data relative to the IMU sensor and then use the proposed algorithm to estimate these parameters. We set the Monte Carlo simulation to 500 times and the time offset range to be $-5ms \backsim 5ms$. We plot the histogram distribution of the estimation errors, as shown in Fig. 14(a). It can be observed that the distribution of estimation errors follows a Gaussian distribution. Fig. 14(b) presents the statistical results of estimation errors in the time offset relative to IMU data for visual, LiDAR, and GNSS raw data.



(a)



(b)

Fig. 14. (a) Histogram and Gaussian distribution plot of estimation errors in the time offset of visual data relative to IMU data; (b) Gaussian distribution results of estimation errors in the time offset of visual data, LiDAR data and GNSS raw data relative to the IMU data

Using the same method, we analyzed the estimation errors of the rotational and translational parameters of visual data, LiDAR data, and GNSS raw data relative to the IMU data, as shown in Table I. According to Table I, the proposed model can correctly estimate the time offset of any combination of sensors. Its standard deviation (0.03, 0.04, 0.05) (ms) is smaller than that of reference [40] 0.14 (ms), indicating that the proposed model can correctly estimate the time offset of any sensor relative to the IMU sensor. Its error is within the acceptable uncertainty range.

As for the estimation error of translation, for visual and LiDAR data, the maximum error is 0.65 (mm), which is

[1]http://www.vision.caltech.edu/bouguetj/calib_doc/

TABLE I
ESTIMATED RESULTS OF DIFFERENT SENSOR COMBINATIONS, G REPRESENTS THE PRESET BASELINE, E_μ REPRESENTS THE MEAN OF THE ESTIMATION ERRORS AND E_σ REPRESENTS THE STANDARD DEVIATION OF THE ESTIMATION ERRORS

Sensor	Type			Result
Vison	τ_l	G		0
		E_μ		0
		E_σ		0.03
	$\mathbf{T}_{b,c}$	$\boldsymbol{\Upsilon}_{b,c}$	G	$[56 \quad 94 \quad -28]$
			E_μ	$[0.15 \quad 0.84 \quad -0.56]$
			E_σ	$[0.23 \quad 0.65 \quad 0.11]$
		$\boldsymbol{R}_{b,c}$	G	$[35° \quad 76° \quad 163°]$
			E_μ	$[0.0021° \quad 0.0053° \quad 0.0002°]$
			E_σ	$[0.0041° \quad 0.0026° \quad 0.0061°]$
LiDAR	τ_l	G		0
		E_μ		0
		E_σ		0.04
	$\mathbf{T}_{b,l}$	$\boldsymbol{\Upsilon}_{b,l}$	G	$[13 \quad 39 \quad 154]$
			E_μ	$[0.26 \quad 0.65 \quad -0.33]$
			E_σ	$[0.43 \quad 0.15 \quad 0.81]$
		$\boldsymbol{R}_{b,l}$	G	$[76° \quad 125° \quad 43°]$
			E_μ	$[0.0047° \quad 0.0033° \quad 0.0025°]$
			E_σ	$[0.0032° \quad 0.0014° \quad 0.0027°]$
GNSS	τ_l	G		0
		E_μ		0
		E_σ		0.05
	$\mathbf{T}_{b,g}$	$\boldsymbol{\Upsilon}_{b,g}$	G	$[-86 \quad 23 \quad -147]$
			E_μ	$[-1.92 \quad 2.45 \quad -2.42]$
			E_σ	$[0.64 \quad 0.78 \quad 0.31]$
		$\boldsymbol{R}_{b,g}$	G	$[38° \quad 87° \quad 82°]$
			E_μ	$[0.0003° \quad -0.0024° \quad 0.0042°]$
			E_σ	$[0.0014° \quad 0.0032° \quad 0.0057°]$

[1] The time variable is measured in ms, the translational variable is in mm and the rotational variable is in degree.

smaller than reference [17], whose error is basically more than 1 (mm), and it is also smaller than the maximum estimation error in reference [40], that is 0.73 (mm). All these indicate that compared with the calibration model in reference [40] and [17], the translation parameter estimated by the proposed model is more accurate. Moreover, the maximum standard deviation is 0.81 (mm), which is less than 0.98 (mm) in [40]. Therefore, it can be inferred that the uncertainty range of translation estimation for visual and LiDAR data is smaller than that in [40], which is within a reasonable and acceptable range. For GNSS raw data, the mean estimation error is around 2.5 (mm). While this is slightly larger than the estimation errors of visual and LiDAR data, it is still within an acceptable range.

As for the rotation parameter, its maximum error is $0.0053°$, which is lower than that in [17], except for the interpolation case, and it is also smaller than the maximum error in [40] $0.0098°$. Therefore, we can conclude that the accuracy of the proposed algorithm is higher than that of references [40] and [17]. The maximum standard deviation of estimation error is $0.0061°$, which is also less than $0.0086°$ in [40]. Therefore, the error uncertainty of the estimated value is also within a reasonable and acceptable range. In summary, it can be concluded that the proposed spatiotemporal calibration model is better than [40] and [17] regarding accuracy and uncertainty.

B. Robustness To Noise in IMU Data

As mentioned earlier, as shown in Eq. (5) and Eq. (6), IMU data is easily influenced by Gaussian white noise υ_1, υ_2 and random errors ι_1, ι_2, which cause error accumulation easily and affect the quality of spatiotemporal parameters. Therefore, we need to consider the robustness of the proposed calibration model to noise in the IMU data. We add a certain proportion of noise to the raw data of the IMU. υ_1, υ_2 and $\dot{\iota_1}, \dot{\iota_2}$ follow a normal distribution, and we assume the mean values for the first set of noise to be $\mu_1, \mu_2, \mu_{\iota_1}, \mu_{\iota_2}$. The mean values for the second and third sets of noise are $2\{\mu_1, \mu_2, \mu_{\iota_1}, \mu_{\iota_2}\}$ and $3\{\mu_1, \mu_2, \mu_{\iota_1}, \mu_{\iota_2}\}$. The variance remains unchanged. We conduct 500 simulations on each group of IMU data with noise and make statistics on the estimated errors to test the robustness of the proposed calibration model to IMU noise.

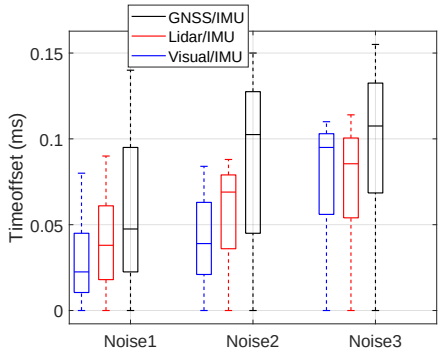
For the convenience of comparison, we unify the vector form of the estimation errors of the rotation and translation parameters into the scalar form. We calculate the mean value of the sum of squares of each dimension of the errors and then take the root mean square of the results. Finally, all the estimated errors under each noise are counted and represented by a box plot. Fig. 15(a), Fig. 15(b) and Fig. 16(a) are box plots of the statistical results of the estimated errors of time offset, rotation parameter and translation parameter respectively.

As seen from Fig. 15(a), with the noise increase, the median line and maximum value slightly rise. For example, for the calibration error of GNSS data, the maximum values are 0.14 (ms), 0.15 (ms) and 0.16 (ms) respectively. According to [40], the standard deviation is about 0.14 (ms). According to this standard, the increase of 0.01 (ms) is within the range of error uncertainty, which indicates that the estimation error of time offset does not change much with the increase of noise. Therefore, the time parameter of the proposed spatiotemporal calibration model is not sensitive to the change in noise intensity, and it has a certain resistance to IMU noise.

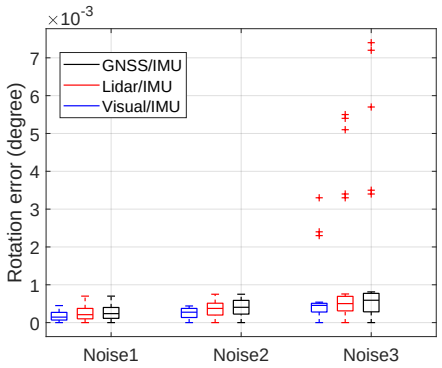
From Fig. 15(b), it can be observed that when the mean of noise in IMU data is twice of the original noise, the estimation errors do not increase significantly. However, when the mean of noise in IMU data triples of the original noise, outliers become significant, with GNSS data being the most affected. Fig. 16(a) shows that when the mean of noise in IMU data doubles, the translation estimation errors of visual data and LiDAR data are relatively small. In contrast, when the mean of noise in IMU data triples, outliers in GNSS data significantly increase, with most estimation errors exceeding 3.5 (mm). Therefore, the proposed calibration algorithm exhibits a certain level of robustness to noise in the raw data of the IMU.

V. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed calibration model in terms of scale and position estimation accuracy using the simulated data. We also apply the proposed algorithm to some open-source programs for performance evaluation using the publicly accessible datasets.

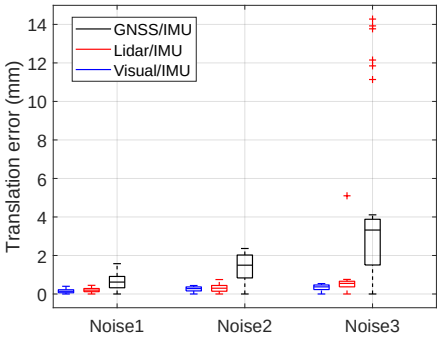


(a)

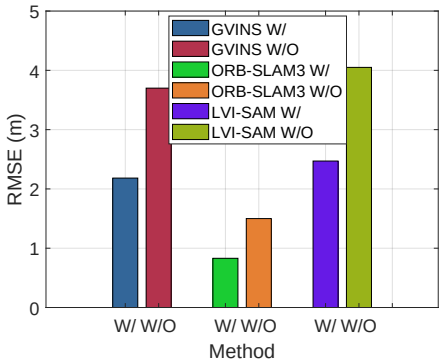


(b)

Fig. 15. (a) Statistical results of estimation errors of time parameters under three kinds of IMU noise; (b) Statistical results of estimation errors of rotation parameters under three kinds of IMU noise



(a)



(b)

Fig. 16. (a) Statistical results of estimation errors of translation parameter under three kinds of IMU noise; (b) The RMSE values for position estimation errors, using the proposed calibration algorithm and without it, in the actual programs with the real-world data as input

A. *Performance Of Position Estimation*

We accurately model LiDAR and visual data and use the GPS satellite data from reference [42]. The input frequencies for visual and LiDAR data are 20 Hz and 10 Hz, respectively. A white noise with a standard deviation of 0.5 pixel is added to all visual feature points. IMU and camera are rigidly connected, and we add noise to the IMU data with one-tenth the intensity of the raw data to the accelerometer and gyroscope, respectively. The LiDAR points cloud is accompanied by a white noise with a standard deviation of 0.5 (mm). By controlling the input of the time parameter and space parameter derived from the proposed model in the paper, we conduct experiments under the following four conditions: w/o time and space parameters, w/o time and w/ space parameters, w/ time and w/o space parameters; w/ time and w/o space parameters. The other conditions in the simulation remain the same. The simulation results are shown in Fig. 17(a), Fig. 17(b), Fig. 17(c) and Fig. 17(d).

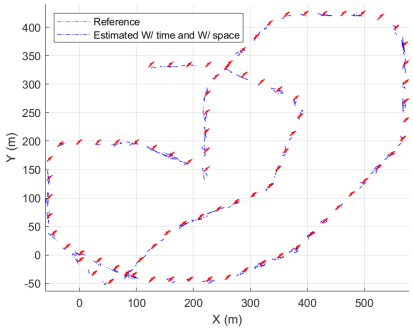
It can be seen from the comparison between Fig. 17(b) and Fig. 17(c) that the estimated trajectory is closer to the reference trajectory after the time parameter is added, indicating that the addition of the proposed time parameter improves the position estimation accuracy of the fusion system. By comparing Fig. 17(b) and Fig. 17(d), it can be concluded that the proposed spatial parameters effectively improve position estimation accuracy.

We run 500 simulations and count all the errors to draw CDF (Cumulative Distribution Function) in four cases, as shown in Fig. 18(a). It can be seen that the errors of position estimation are about 1 (m) at most when the time and space parameters are added, while those are more than 15 (m) when the time and space parameters are not adopted. The position estimation errors are slightly improved by adding any of the parameters. The RMSE results of position estimation errors under the four conditions are shown in rows 2 to 5 of Tabel. II.

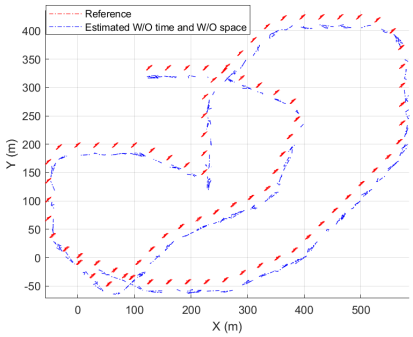
After adopting the time and space calibration of the proposed model, the position estimation accuracy of the fusion system is significantly improved, which is about 21 times lower than the RMSE value without calibration. The proposed space-time calibration models of the GNSS, LiDAR, and visual data are theoretically feasible and effective.

B. *Performance Of Scale Estimation*

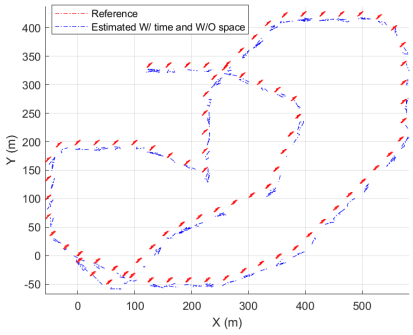
Monocular cameras are popular in low-cost fusion systems due to their simple structure and low cost. However, it is impossible to determine an object's actual size in a single



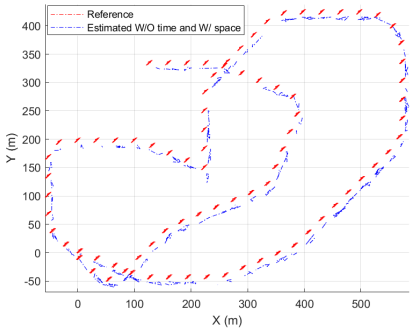
(a)



(b)

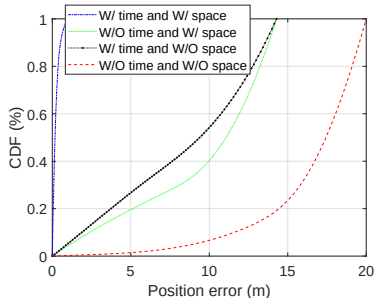


(c)

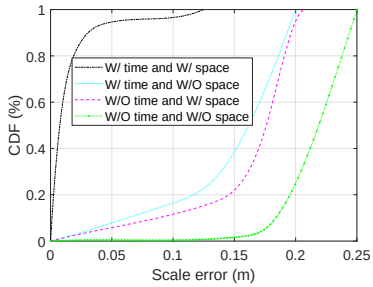


(d)

Fig. 17. (a) W/ time and W/ space parameters; (b) W/O time and W/O space parameters; (c) W/ time and W/O space parameters; (d) W/O time and W/ space parameters



(a)



(b)

Fig. 18. (a) Schematic diagram of statistical results of position estimation errors under four conditions; (b) The statistical analysis results of scale estimation error in four cases

TABLE II
RMSE OF ESTIMATION ERRORS UNDER DIFFERENT CONDITIONS

Mode	Mode of comparison			RMSE (m)
Sim	Pos	W/ time and W/ space		**0.8328**
		W/ time and W/O space		10.7247
		W/O time and W/ space		11.5641
		W/O time and W/O space		17.8529
	Sca	W/ time and W/ space		**0.0672**
		W/O time and W/ space		0.1834
		W/ time and W/O space		0.1783
		W/ Otime and W/O space		0.2346
Exp	Implemented	GVINS	W/	**2.1830**
			W/O	3.700 [30]
		ORB-SLAM3	W/	**0.0415**
			W/O	0.0750 [35]
		LVI-SAM	W/	**2.4705**
			W/O	4.0500 [43]

[1] Sim represents simulation, Exp represents experiment, Pos represents position, and Sca represents scale.

image. The relative depth of the object can be measured by the parallax formed by the camera's motion. However, the trajectory and map estimated by monocular SLAM will differ from the actual trajectory and map by a factor of scale. The image cannot determine the accurate scale alone, called scale uncertainty. In the simulation, we design a fusion system of a monocular camera, LiDAR points cloud, and Inertial data, and the IMU data is used to assist the LiDAR points cloud for distortion correction and visual feature points for tracking. This fusion system aims to design a depth estimation algorithm for a spatial rectangular plane of unknown size.

The primary premise of the fusion system is spatiotemporal calibration. By controlling the time and space parameters derived from the space-time calibration model proposed in the paper, we simulate in the following four situations: w/o time and w/o space parameters, w/ time and w/o space parameters, w/o time and w/ space parameters, w/ time and w/ space parameters. Other conditions are consistent in the simulation. The original picture is shown in Fig. 19, and the simulation results are shown in Fig. 20. As seen from Fig. 20, A, B, C, and D represent scale estimations of the same location under four different conditions. It is evident that when both temporal and spatial parameters are present, as in point D, the estimation is more refined, indicating a higher accuracy in scale estimation. Points B and C show slightly higher accuracy in scale estimation compared to point A. Therefore, either temporal or spatial parameters can enhance the accuracy of scale estimation. In order to facilitate quantitative analysis, we conduct 500 simulations and count all the scale estimation errors in four cases. The results are shown in Fig. 18(b).

Fig. 19. Original image for scale estimation

Fig. 18(b) shows that the maximum scale estimation error after adding spatiotemporal parameters is about 0.13 (m). Most of the errors are concentrated between 0 (m) and 0.04 (m) , while the scale estimation errors without them are basically above 0.15 (m), and most of them are concentrated between 0.2 (m) and 0.25 (m). After adding any of these parameters, the precision of scale estimation is improved slightly, but the effect is still not ideal. This shows that both time and space parameters are essential for the precision of scale estimation. The RMSE values of the scale estimation errors are shown in rows 6 to 9 of Table. II. The precision of scale estimation without temporal and spatial calibration is about 3.5 times higher than that with temporal and spatial calibration. It is proved that the proposed space-time calibration models play a vital role in improving the precision of scale estimation, and the proposed calibration models of LiDAR data and visual data are feasible and effective in improving the accuracy of scale estimation in theory.

C. Performance of applications in real-world projects

In this section, we validate the application performance of the proposed calibration model using the publicly available datasets in the actual open-source programs. The experiments in this paper are conducted on a laptop configured with a 7-1165G7 processor at 2.800GHz and 32GB of RAM.

1) Applied to GVINS: GVINS [30] is a nonlinear optimized fusion system of tightly coupled GNSS raw measurements, visual data, and Inertial data. The first step of the fusion system is to calibrate the GNSS raw measurements, visual data, and Inertial data. We compare the estimated trajectories with and without the proposed calibration algorithm with the complex-environment dataset[2] of GVINS as input. The complex-environment dataset is a collection of GNSS raw measurements and ground truth by Shen Shaojie's team at the Hong Kong University of Science and Technology, which contains GNSS raw measurement, visual, and Inertial data. The total time taken by the proposed calibration algorithm is 9.8 (ms). The estimated trajectory is shown in Fig. 21.

We conduct a statistical analysis of all position estimation errors, and their RMSE values are presented in the 10th row of Tabel. II, as shown in Fig. 16(b). Therefore, we can conclude that, in practical open-source programs, using the proposed calibration algorithm can improve position estimation accuracy by about 41%.

2) Applied to ORB-SLAM3: ORB-SLAM3 [35] is a tightly coupled fusion system of the visual and Inertial data. Firstly, the space-time calibration of visual and Inertial data is carried out. In this section, we use the MH-04-difficult dataset of EuRoC[3] as input, run the ORB-SLAM3 program and compare position estimation errors in both scenarios, with the proposed calibration algorithm and without the proposed calibration algorithm. The EuRoC dataset is a Vision/Inertial dataset collected on a miniature aircraft (MAV). The frame rate is 100Hz, which can achieve millimeter accuracy, which we use as the baseline position. The total time taken by the proposed calibration algorithm is 3.9 (ms). The estimated trajectory of ORB-SLAM3 with the proposed calibration model is shown in Fig. 22.

The RMSE values of the position estimation errors are shown in lines 12 and 13 of Table. II, as shown in Fig. 16(b). For ease of observation, we magnify the RMSE values by a factor of 20 here. Similar to the previous analysis of GVINS, we can conclude that the proposed calibration algorithm for the visual data can improve the accuracy of position estimation by about 45% compared to that without the proposed calibration algorithm.

3) Applied to LVI-SAM: LVI-SAM [43] is a tightly coupled fusion system of LiDAR-Vision-Inertial data. The first step is to calibrate the LiDAR, visual, and Inertial data. In this section, we input the Jackal dataset[4], run the LVI-SAM program, and compare the position estimation errors. The Jackal dataset is gathered by mounting the sensor suite on a Clearpath Jackal uncrewed ground vehicle (UGV), which includes a feature-rich environment, beginning and ending at the same position. The total time taken by the proposed calibration algorithm is 6.2 (ms). The estimated trajectory of LVI-SAM using the proposed calibration model is shown in Fig. 23.

[2]http://https://github.com/HKUST-Aerial-Robotics/GVINS-Dataset/
[3]https://projects.asl.ethz.ch/datasets/doku.php?id=kmavvisualinertialdatasets
[4]https://github.com/TixiaoShan/LVI-SAM

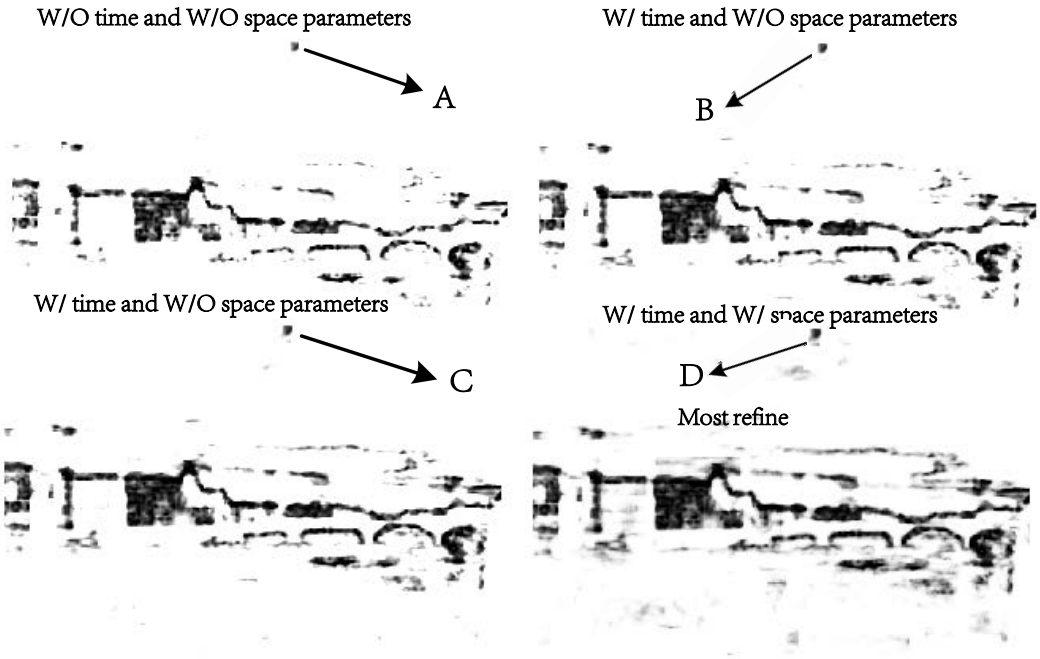


Fig. 20. Scale estimation results under four conditions: W/O time and W/O space parameters; W/ time and W/O space parameters; W/ time and W/O space parameters; W/ time and W/ space parameters

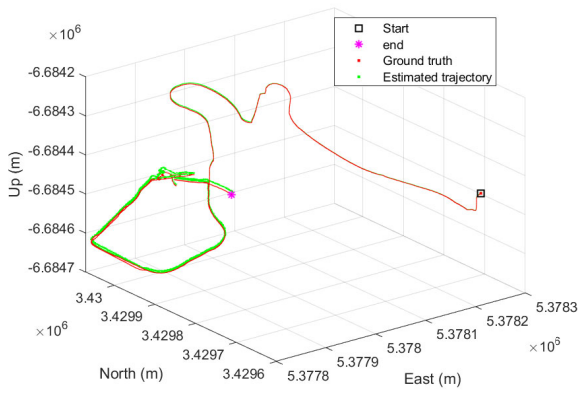


Fig. 21. The comparison between the reference trajectory and the estimated trajectory obtained by applying the proposed calibration model to the GVINS program using the complex_environment dataset as input

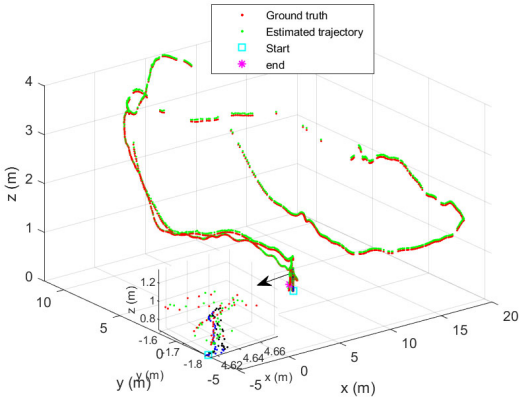


Fig. 22. The comparison between the reference trajectory and the estimated trajectory obtained by applying the proposed calibration model to the ORB-SLAM3 program with the MH-04-difficult dataset as input

The RMSE values of the position estimation errors are shown in lines 14 and 15 of Table. II. According to the same analytical logic of GVINS and ORB-SLAM3, we can see that the proposed LiDAR-visual-inertial calibration algorithm can improve the position estimation accuracy of the fusion system by 39% compared that without the proposed calibration algorithm.

Safety and reliability are crucial aspects for applications such as autonomous driving and drones, as any errors or instability can lead to serious consequences. Therefore, we use maximum error to assess the safety of the proposed algorithm and calculate the confidence intervals of the position estimation errors to determine its reliability. The maximum error of the proposed calibration algorithm when applied to GVINS is 2.357 (m), with the 95% confidence interval of 2.057 + 0.234 (m), where 2.057 (m) is the average estimated error; when applied to ORB-SLAM3, the maximum error is 0.053 (m), with the confidence interval of 0.0406 + 0.024 (m); and when applied to LVI-SAM, the maximum error is 2.638 (m), with the confidence interval of 2.324 + 0.286 (m).

The maximum error values from the application of the proposed algorithm to actual programs indicate that the system does not pose a danger or loss in the event of anomalies, demonstrating good safety. The error bar chart of the confidence intervals is shown in Fig. 24. The 95% confidence intervals indicate that after applying the proposed algorithm

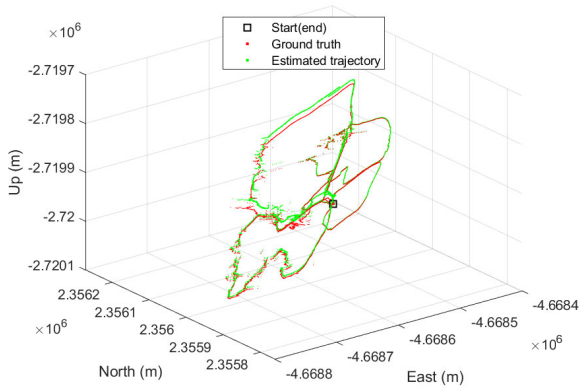


Fig. 23. The comparison between the reference trajectory and the estimated trajectory obtained by applying the proposed calibration model to the LVI-SAM program using the Jackal dataset as input

to actual programs, the system can consistently output accurate and consistent data over long-term use, indicating high reliability.

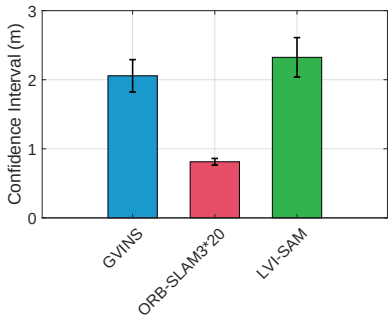


Fig. 24. The confidence interval for the position error of the proposed algorithm applied to practical programs. For easier observation, the confidence interval for the position estimation error after applying the proposed algorithm to the ORB-SLAM3 program is magnified by 20 times

VI. Conclusions and Future Researches

This paper proposes a spatiotemporal calibration algorithm for the fusion system of GNSS data, LiDAR data, and visual data, using the IMU sensor as the central coordinate system. Through simulations, it verifies that the accuracy of the spatiotemporal parameters and the error uncertainty of the proposed algorithm outperform existing methods. Additionally, experimental results demonstrate the robustness of the proposed calibration model against noise in IMU data. By applying the proposed calibration algorithm to three real programs with actual datasets, its superior performance is validated in terms of position estimation accuracy, computational efficiency, safety, and reliability. In the future, we will explore additional types of sensor data and design their optimization factors relative to the IMU data.

References

[1] R. Ishikawa, T. Oishi, and K. Ikeuchi, "Lidar and camera calibration using motions estimated by sensor fusion odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 2018, pp. 7342–7349.

[2] B. Fu, Y. Wang, X. Ding, Y. Jiao, L. Tang, and R. Xiong, "Lidar-camera calibration under arbitrary configurations: Observability and methods," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 6, pp. 3089–3102, 2019.

[3] M. R. Nowicki, "Spatiotemporal calibration of camera and 3d laser scanner," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6451–6458, 2020.

[4] T. Scott, A. A. Morye, P. Piniés, L. M. Paz, I. Posner, and P. Newman, "Choosing a time and place for calibration of lidar-camera systems," in *Proceeding of IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4349–4356.

[5] J. Beltrán, C. Guindel, A. de la Escalera, and F. García, "Automatic extrinsic calibration method for lidar and camera sensor setups," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 17 677–17 689, 2022.

[6] C. Yuan, X. Liu, X. Hong, and F. Zhang, "Pixel-level extrinsic self calibration of high resolution lidar and camera in targetless environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7517–7524, 2021.

[7] Z. Yang and S. Shen, "Monocular visual–inertial state estimation with online initialization and camera–imu extrinsic calibration," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 39–51, 2016.

[8] M. Li, H. Yu, X. Zheng, and A. I. Mourikis, "High-fidelity sensor modeling and self-calibration in vision-aided inertial navigation," in *Proceeding of IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 409–416.

[9] Y. Yang, P. Geneva, K. Eckenhoff, and G. Huang, "Degenerate motion analysis for aided ins with online spatial and temporal sensor calibration," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2070–2077, 2019.

[10] D.-G. Choi, Y. Bok, J.-S. Kim, and I. S. Kweon, "Extrinsic calibration of 2-d lidars using two orthogonal planes," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 83–98, 2015.

[11] Z. Taylor and J. Nieto, "Motion-based calibration of multimodal sensor extrinsics and timing offset estimation," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1215–1229, 2016.

[12] W. Lee, K. Eckenhoff, P. Geneva, and G. Huang, "Intermittent GPS-aided VIO: Online initialization and calibration," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5724–5731.

[13] S. Bai, J. Lai, P. Lyu, B. Wang, X. Sun, and W. Yu, "An enhanced adaptable factor graph for simultaneous localization and calibration in gnss/imu/odometer integration," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 9, pp. 11 346–11 357, 2023.

[14] Y. Liu, F. Wu, Z. Liu, K. Wang, F. Wang, and X. Qu, "Can language models be used for real-world urban-delivery route optimization?" *The Innovation*, vol. 4, no. 6, 2023.

[15] X. Qu, H. Lin, and Y. Liu, "Envisioning the future of transportation: Inspiration of chatgpt and large models," *Communications in Transportation Research*, vol. 3, 2023.

[16] Y. Fei, P. Shi, Y. Li, Y. Liu, and X. Qu, "Formation control of multi-agent systems with actuator saturation via neural-based sliding mode estimators," *Knowledge-Based Systems*, vol. 284, p. 111292, 2024.

[17] J. Kümmerle, T. Kühner, and M. Lauer, "Automatic calibration of multiple cameras and depth sensors with a spherical target," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 2018, pp. 1–8.

[18] L. Zhou, Z. Li, and M. Kaess, "Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 2018, pp. 5562–5569.

[19] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *Proceeding of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3, 2004, pp. 2301–2306.

[20] A. Geiger, F. Moosmann, Ö. Car, and B. Schuster, "Automatic camera and range sensor calibration using a single shot," in *Proceeding of IEEE international conference on robotics and automation*, 2012, pp. 3936–3943.

[21] A. Kassir and T. Peynot, "Reliable automatic camera-laser calibration," in *Australasian Conference on Robotics and Automation*, vol. 2010, 2010.

[22] F. Girrbach, R. Zandbergen, M. Kok, T. Hageman, G. Bellusci, and M. Diehl, "Towards in-field and online calibration of inertial navigation systems using moving horizon estimation," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 4338–4343.

[23] A. Perez-Yus, E. Fernandez-Moral, G. Lopez-Nicolas, J. J. Guerrero, and P. Rives, "Extrinsic calibration of multiple RGB-D cameras from

line observations," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 273–280, 2017.
[24] W. Lee, P. Geneva, Y. Yang, and G. Huang, "Tightly-coupled GNSS-aided visual-inertial localization," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 9484–9491.
[25] F. Han, Q. Zhang, B. Fu, T. Yang, Y. Wang, and R. Xiong, "Multiconstraint spatial and temporal calibration of rotating line structured light vision sensor," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–11, 2021.
[26] Y. Shi, B. Lian, Y. Zeng, and E. Kurniawan, "Robust loop-closure algorithm based on gnss raw data for large-scale and complex environments," *IEEE Transactions on Vehicular Technology*, pp. 1–15, 2024.
[27] W. Jianing, L. Baowang, and X. Zhe, "Weak GPS signal acquisition method based on DBZP," *Journal of systems engineering and electronics*, vol. 29, no. 2, pp. 236–243, 2018.
[28] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.
[29] J. Rehder, R. Siegwart, and P. Furgale, "A general approach to spatiotemporal calibration in multisensor systems," *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 383–398, 2016.
[30] S. Cao, X. Lu, and S. Shen, "GVINS: Tightly coupled GNSS–visual–inertial fusion for smooth and consistent state estimation," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2004–2021, 2022.
[31] W. DRAFT, "Draft recommended practice for precision centrifuge testing of linear accelerometers," 2009.
[32] D. Pan, J. Shao, S. Zhang, S. Zhang, B. Chang, H. Xiong, and W. Zhang, "Slam-based forest plot mapping by integrating imu and self-calibrated dual 3-d laser scanners," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–13, 2023.
[33] H. Zhao, B. Lian, and J. Feng, "Adaptive beamforming and phase bias compensation for GNSS receiver," *Journal of Systems Engineering and Electronics*, vol. 26, no. 1, pp. 10–18, 2015.
[34] E. D. Kaplan and C. Hegarty, *Understanding GPS/GNSS: principles and applications*. Artech house, 2017.
[35] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
[36] D. G. Viswanathan, "Features from accelerated segment test (fast)," in *Proceedings of the 10th workshop on image analysis for multimedia interactive services, London, UK*, 2009, pp. 6–8.
[37] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to sift or surf," in *2011 International conference on computer vision*, 2011, pp. 2564–2571.
[38] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis.Pattern Recognit*, 2012.
[39] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.
[40] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proceeding of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1280–1286.
[41] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.
[42] K. Borre and D. Akos, "A software-defined gps and galileo receiver: single-frequency approach," in *Proceedings of the 18th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2005)*, 2005, pp. 1632–1637.
[43] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in *Proceeding of IEEE international conference on robotics and automation (ICRA)*, 2021, pp. 5692–5698.