

Universal IMU-Centric Spatiotemporal Calibration Algorithm for Heterogeneous Information

Yanfang Shi; Baowang Lian

School of Electronics And Information
Northwestern Polytechnical University
Shaanxi, China

syf1819285@mail.nwpu.edu.cn
bwliao@nwpu.edu.cn

Yonghong Zeng; Yugang Ma

Institute for Infocomm Research
Singapore, Singapore

yhzeng@i2r.a-star.edu.sg
mayg@i2r.a-star.edu.sg

Yangyang Liu

School of Electronics And Information
Northwestern Polytechnical University
Shaanxi, China

yyliu@nwpu.edu.cn

Abstract—Spatiotemporal calibration is an essential problem in the fusion system with heterogeneous multi-source information. Therefore, a universal spatiotemporal calibration algorithm for heterogeneous information is much needed. This paper proposes a universal spatiotemporal calibration technique with the inertial sensor as the central coordinate system. Firstly, it employs a high-order spline interpolation method to transform the output data of the inertial sensor into a continuous form. Subsequently, the calibration model is established by combining the output data from other sensors. The paper provides detailed descriptions of the spatiotemporal calibration models for LiDAR (Light Detection and Ranging) data, and visual data, respectively. In the simulations, the proposed calibration models are applied to existing open-source programs using publicly available datasets. The results demonstrate that, with the adoption of the proposed calibration algorithm, the position estimation accuracy of the fusion system can be improved by 39%.

Index Terms—Spatiotemporal calibration, Vision, LiDAR, Inertial

I. INTRODUCTION

MULTI-source fusion is an irreplaceable method in motor vehicles to achieve precise positioning. It is necessary to transform the measurements to a unified spatio-temporal reference frame through calibration. Without temporal and spatial registration and directly fusing data, the positional estimation results would have more significant errors, making fusion less preferable. Hence, temporal and spatial registration for heterogeneous information is necessary for data fusion.

This research is supported in part by National Natural Science Foundation of China under Grant 62173276, 62171735, 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. This research is also supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research & Development Programme (FCP-NTU-RG-2022-021, FCP-ASTAR-TG-2022-003). The author would like to express gratitude for the support and sponsorship provided by the China Scholarship Council (Liu Jin Xuan [2021]70). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore and Infocomm Media Development Authority.

Space-time calibration has always been a research topic of great concern to researchers.

1) *Calibration For Specific Combinations:* [1] proposes calibration methods for camera-LiDAR (Light Detection and Ranging). Fuzhang Han [2] proposes a spatiotemporal calibration method for a LVLVS (Line structured light vision sensor). In literature [3], the authors proposed a calibration method for GPS (Global Positioning System), visual, and inertial data.

2) *Calibration Depending On The Scene:* The method [4] requires spherical objects in the scene. In [5], spatial parameters are solved using a checkerboard by combining 3D lines and plane correspondences with a minimal number of poses. The calibration method for the IMU (Inertial Measurement Unit) in [6] requires the assistance of a moving horizon scheme to estimate parameters.

3) *Calibration For Different Output:* Lipu Zhou in [5], [7] only calculates the spatial parameters, ignoring the calculation of time parameters. Fuzhang Han [2] proposes a spatiotemporal calibration method for LVLVS, divided into two stages, which will lead to the accumulation of errors and eventually affect the accuracy of spatiotemporal parameters. In [8], the calibration method proposed for GNSS (Global Navigation Satellite System) raw data, visual data, and inertial data outputs includes translation parameters and yaw angle estimation, with no estimation of time parameters.

Therefore, for fusion systems dealing with heterogeneous information, it is urgent and crucial to choose a universal spatiotemporal calibration method that is not dependent on the scene and can adapt to the increasing variety of sensor types.

II. OUR CONTRIBUTION

Inspired by [9], this paper proposes a universal spatiotemporal calibration method based on nonlinear optimization centered around the IMU sensor's coordinate system.

- The proposed model is scene-agnostic and combines time and spatial parameters through high-order spline

interpolation of IMU data, irrespective of sensor type. Therefore, it is theoretically applicable to any sensor data.

- We employ the re-projection principle to build the calibration model for visual data, and utilize point-to-line or point-to-plane distances to establish the calibration model for LiDAR point clouds.
- The simulation results show that applying the proposed calibration algorithm to some existing open-source odometry systems can improve position estimation accuracy by 39%, which fully demonstrates the feasibility of the proposed calibration model in practical applications.

III. PROBLEM FORMULATION

On the one hand, regarding the time parameter, inspired by the paper [10] 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 d , we can describe the error expression as Eq. (1):

$$e_j = y_j - h_{p,(t_j+d)} \quad (1)$$

where, y_j is the actual measurement at timestamp t_j , $h_{p,(t_j+d)}$ is the predicted measurement at the moment t_j . The predicting model varies with the sensor.

On the other hand, regarding the spatial parameters, the transformation matrix from other coordinate systems k to the IMU coordinate system b is given by Eq. (2).

$$\mathbf{T}_{k,b}(t) = \begin{bmatrix} \mathbf{R}_{k,b}(t) & \mathbf{t}_{k,b}(t) \\ \mathbf{0}^T & 1 \end{bmatrix} \quad (2)$$

Here, we adopt the idea of the paper [10] to represent the data of IMU by a B-spline function of sixth-order.

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

$$\mathbf{T}_{k,b}(t) = \begin{bmatrix} \mathbf{C}_{k,b}(t) & \mathbf{p}_{k,b}(t) \\ \mathbf{0}^T & 1 \end{bmatrix} \quad (3)$$

Where $\mathbf{C}_{k,b}(t) = \mathbf{C}(\varphi(t)) = \mathbf{C}(\phi_t(t)\mathbf{c}_{(k,\varphi)})$, $\mathbf{p}_{k,b}(t) = \phi_t(t)\mathbf{c}_{(k,t)}$, $\mathbf{C}(\cdot)$ is a function that transforms the orientation parameters to the rotation matrix and $\phi_t(t)$ is a set of B-spline functions. Then, finding the spatial-temporal parameters from the k coordinate system to the IMU coordinate system turns into finding $\mathbf{c}_{(k,\varphi)}$ and $\mathbf{c}_{(k,t)}$, where t includes time delay d , which represents the time parameter.

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 obtain a residual module that participates in the optimization process. The second key to the proposed spatiotemporal calibration model is to utilize nonlinear optimization methods to solve for $\mathbf{c}_{(k,\varphi)}$ and $\mathbf{c}_{(k,t)}$.

IV. ERROR MODEL AND SPATIOTEMPORAL CALIBRATION

The celestial coordinate system can be found in reference [11]. In our system, we take the n coordinate system as the world coordinate system.

A. Error Models Of Different Sensor Systems

In the section, we take LiDAR, and visual data as examples to elaborate the construction of the residual module separately.

1) *The Processing of the IMU Data:* In this paper, we adopt the output data model from [12], which includes the influence of gravity on acceleration, as illustrated in Eq. (4) and Eq. (5).

$$\tilde{\omega} = \omega + b^g + n^g \quad (4)$$

$$\tilde{\mathbf{a}} = \mathbf{q}_{bw}(\mathbf{a}^w + g^w) + b^a + n^a \quad (5)$$

ω and $\mathbf{a}^w$ are the true values. $\tilde{\omega}$ and $\tilde{\mathbf{a}}$ are the measured values that are used in the simulation. b^g and b^a are the deviations of acceleration and angular velocity, and their derivatives follow a Gaussian distribution. n^a and n^g are Gaussian white noise for acceleration and angular velocity, respectively. g^w represents gravitational acceleration. $\mathbf{q}_{bw}$ represents the quaternion transformation from the inertial coordinate system to the world coordinate system.

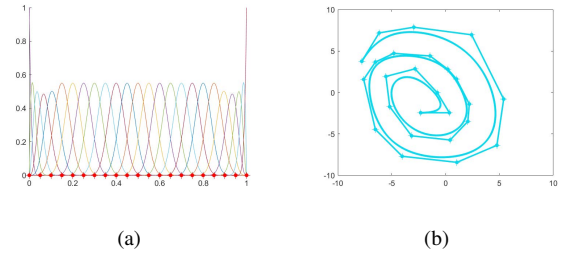


Fig. 1. (a) Schematic diagram of B-spline functions; (b) Schematic diagram of control points and the interpolated trajectory.

We can obtain the position $\mathbf{P}_b^i$ of the IMU sensor by employing the algorithm from [13] using 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. 1(a) and Fig. 1(b), respectively.

2) *Error Model Of Visual Data:* The construction process of the optimization factor of visual data is shown in Fig. 2.

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 $\mathbf{P}_{b,m+1}^i$ and $\mathbf{P}_{b,m}^i$. The coordinate values in the ECEF coordinate system at the moment of reception of the inertial sensor are equal to the coordinate values in the ECI coordinate system, $\mathbf{P}_{b,m+1}^i = \mathbf{P}_{b,m+1}^e$ and $\mathbf{P}_{b,m}^i = \mathbf{P}_{b,m}^e$. Then, we apply Eq. (6) to convert the values of the inertial sensor in the ECEF coordinate system into values in the ENU (East-North-Up) coordinate system, which corresponds to the world coordinate system.

$$\mathbf{P}_{b,m}^n = \mathbf{R}_e^n \mathbf{P}_{b,m}^i \quad (6)$$

Where $\mathbf{R}_e^n$ represents the transformation parameters between the ECEF coordinate system and the ENU coordinate system [11]. Then, by multiplying with $\mathbf{T}_{b,c}(t_m + d_c)$, we obtain

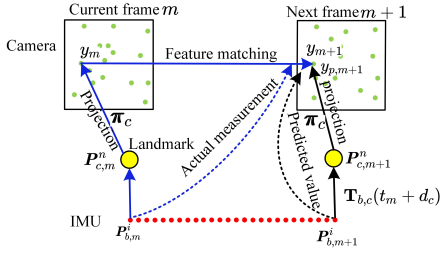


Fig. 2. Schematic diagram of the construction of optimization factor for spatiotemporal calibration parameters of visual data

the coordinates of the camera in the world coordinate system $\mathbf{P}_{c,m}^n$, as shown in Eq. (7).

$$\mathbf{P}_{c,m}^n = \mathbf{T}_{b,c}(t_m + d_c)\mathbf{P}_{b,m}^n \quad (7)$$

We map this landmark point $\mathbf{P}_{c,m}$ to the pixel plane at the time m through projection to find out the corresponding pixel coordinates, as shown in Eq. (8) and π_c is the projection function of the camera.

$$y_m = \pi_c(\mathbf{T}_{b,c}(t_m + d_c)\mathbf{P}_{b,m}^n) \quad (8)$$

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. (9).

$$y_{m+1} \Leftrightarrow \text{FeatureMatching}(y_m) \quad (9)$$

This paper first extracts ORB (Oriented FAST and Rotated BRIEF) feature points [14] from visual images. Fig. 3(a) shows the ORB feature extraction results for KITTI 00 [15] sequence, and Fig. 3(b) is the matching result of ORB feature points.

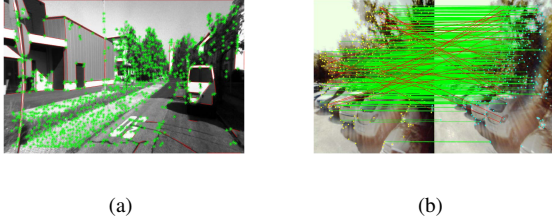


Fig. 3. (a) ORB feature point extraction results of a specific image in the KITTI dataset; (b) ORB feature points matching results of different images at different times in the Malaga dataset.

Similarly, using the same approach as Eq. (7), we can obtain the predicted world coordinates of the camera at the $m+1$ frame, as shown in Eq. (10).

$$\mathbf{P}_{c,m+1}^n = \mathbf{T}_{b,c}(t_m + d_c)\mathbf{P}_{b,m+1}^n \quad (10)$$

The predicted pixels of the predicted landmark points on the $m+1$ frame can be obtained through the projection algorithm. As shown in Eq. (11):

$$\begin{aligned} y_{p,m+1} &= \pi_c(\mathbf{P}_{c,m+1}^n) \\ &= \pi_c(\mathbf{T}_{b,c}(t_m + d_c)\mathbf{P}_{b,m+1}^n) \end{aligned} \quad (11)$$

Therefore, for a single ORB feature point, the error model, including space and temporal parameters, is as Eq. (12):

$$\begin{aligned} e_c &= y_{p,m+1} - y_{m+1} \\ &= \pi_c(\mathbf{T}_{b,c}(t_m + d_c)\mathbf{P}_{b,m+1}^n) \\ &\quad - \text{FeatureMatching}(\pi_c(\mathbf{T}_{b,c}(t_m + d_c)\mathbf{P}_{b,m}^n)) \end{aligned} \quad (12)$$

If there are N_c sets of successfully matched ORB feature point pairs on the $m+1$ frame and the m frame, then the error model for the spatiotemporal parameters is as shown in Eq. (13):

$$e_C = \frac{1}{N_c} \sum_{j=1}^{N_c} |y_{p,j} - y_j| \quad (13)$$

3) *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. 4. In this paper, we employ the method described in

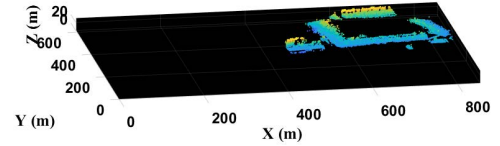


Fig. 4. Points cloud representation of LiDAR points cloud for malaga2009_campus_2L in the Malaga dataset

reference [16] to calculate the curvature of the points cloud. Different distance calculation formulas exist for corner and surface points [16]. The distance should be constant in different coordinate systems. We use the symbol $d_\epsilon(\mathbf{P}_{k,n}^L)$ or $d_\zeta(\mathbf{P}_{k,n}^L)$ to denote the distance, as shown in Eq. (14), we refer to them as the actual distance measurements.

$$d_{m-c} = d_\epsilon(\mathbf{P}_{k,n}^L), d_{m-s} = d_\zeta(\mathbf{P}_{k,n}^L) \quad (14)$$

On the other hand, 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 $\mathbf{T}_{b,l}(t_j + d_l)$ from the LiDAR to the IMU, as shown in Eq. (15).

$$\mathbf{P}_l^n = \mathbf{T}_{b,l}(t_j + d_l)\mathbf{R}_e^n \mathbf{P}_b^i \quad (15)$$

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. 5 illustrates the relationship between the LiDAR sensor and point cloud positions.

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. As a result, 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. (16).

$$\mathbf{P}_{k,i}^n = \mathbf{P}_{k,i}^L + \mathbf{P}_l^n \quad (16)$$

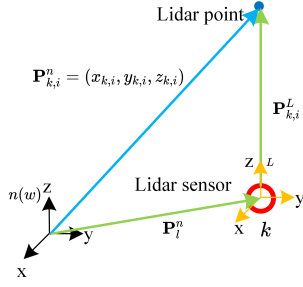


Fig. 5. Illustration of the relationship between LiDAR sensor coordinates and point cloud coordinates

Combining with Eq. (15), the coordinates of the LiDAR points cloud in the world coordinate system at time k is as shown in Eq. (17).

$$\mathbf{P}_{k,i}^n = \mathbf{P}_{k,i}^L + \mathbf{T}_{b,l}(t_j + d_l)\mathbf{R}_e^n \mathbf{P}_b^i \quad (17)$$

Then we plug this predicted position into the distance calculation formulas to get the predicted distance $d_p = d_\varepsilon(\mathbf{P}_{k,i}^n)$ or $d_p = d_c(\mathbf{P}_{k,i}^n)$. Therefore, the error model of the spatiotemporal calibration is shown in Eq. (18).

$$e_L = \frac{1}{N_l} \sum_{j=1}^{N_l} |d_{p,j} - d_j| \quad (18)$$

Where N_l represents the number of points in the operation.

B. Spatiotemporal Calibration With Nonlinear Optimization

According to the above analysis, the overall cost function is as shown in Eq. (19):

$$E_{all} = \frac{1}{N_c} \sum_{j=1}^{N_c} \omega_j |y_{p,j} - y_j| + \frac{1}{N_l} \sum_{j=1}^{N_l} |d_{p,j} - d_j| \quad (19)$$

The term ω_j represents a weight that ranges from 0 to 1. 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: $\mathbf{T}_{b,l}(t_j + d_l)$, $\mathbf{T}_{b,c}(t_j + d_c)$, $\mathbf{T}_{b,g}(t_j + d_g)$. According to Eq. (3), we can transform the problem into solving: $\mathbf{c}_{(l,\varphi)}$, $\mathbf{c}_{(l,t)}$, $\mathbf{c}_{(c,\varphi)}$, $\mathbf{c}_{(c,t)}$, $\mathbf{c}_{(g,\varphi)}$, $\mathbf{c}_{(g,t)}$, d_l , d_c , d_g .

The most crucial step is to calculate the iteration step length. The Gauss-Newton method utilizes the Eq. (20) to calculate the iteration step size.

$$(\mathbf{H} + \lambda \mathbf{I}) \Delta X = \mathbf{g} \quad (20)$$

Here, $\mathbf{H}$ represents the second derivative, and $\mathbf{g}$ represents the first-order derivative of the cost function. We set the variable λ to a fixed value of 0.5. Here, we present the process of solving the spatiotemporal parameters of LiDAR data relative to the inertial data. The same approach can be applied to other sensor data. According to the cost function and our analysis, seven variables should be optimized. In this case, we use the

TABLE I
RMSE OF ESTIMATION ERRORS UNDER DIFFERENT CONDITIONS

Mode	Mode of comparison		RMSE (m)
Exp	Open-source	ORB-SLAM3	W/
			W/O
		LVI-SAM	W/
			W/O

¹ Exp represents experiment.

cost function's first and second derivatives, incorporating 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. (21).

$$\left(\frac{\partial^2 E_{all}}{\partial^2 \Delta x_i} + \lambda \mathbf{I} \right) \Delta x_i = \frac{\partial E_{all}}{\partial \Delta x_i} \quad (i = 1, 2, \dots, 7) \quad (21)$$

The value $x_1, x_2, x_3, x_4, x_5, x_6, x_7$ represents the three dimensions of vectors $\mathbf{c}_{(g,\varphi)}$ and $\mathbf{c}_{(g,t)}$ respectively along with a time deviation d_g . $\frac{\partial^2 E_{all}}{\partial^2 \Delta x_i}$ is the second derivative of the cost function E_{all} , denoted as $\mathbf{H}$ in Eq. (20) and $\frac{\partial E_{all}}{\partial \Delta x_i}$ is the first derivative of the cost function E_{all} , denoted as $\mathbf{g}$ in Eq. (20). $\frac{\partial^2 E_{all}}{\partial^2 \Delta x_1}$ is the second derivative of the cost function when treating Δx_1 as an unknown variable and $(\Delta x_2, \dots, \Delta x_7)$ as known variables. The other equations are similar. In this way, we can obtain seven equations that involve $(\Delta x_1, \dots, \Delta x_7)$. By simultaneously solving these seven equations, we can determine the specific value of $(\Delta x_1, \dots, \Delta x_7)$. 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. (22).

$$\|E_{all}\| \leq 10^{-9} \quad (22)$$

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.

V. SIMULATIONS AND COMPARISONS

In this section, we apply the proposed algorithm to some open-source programs for performance evaluation using publicly accessible datasets.

1) *Applied to ORB-SLAM3*: ORB-SLAM3 [14] is a tightly coupled fusion system of visual and inertial data. In this section, we use the MH-04-difficult dataset of EuRoC¹ as input. The estimated trajectory of ORB-SLAM3 using the proposed calibration model is shown in Fig. 6.

The RMSE values of the position estimation errors of the two calibration algorithms are shown in Table. I. Through computation, we can conclude that, in practical open-source programs, using the proposed vision calibration algorithm can

¹ <https://projects.asl.ethz.ch/datasets/doku.php?id=kamavvisualinertialdatasets>

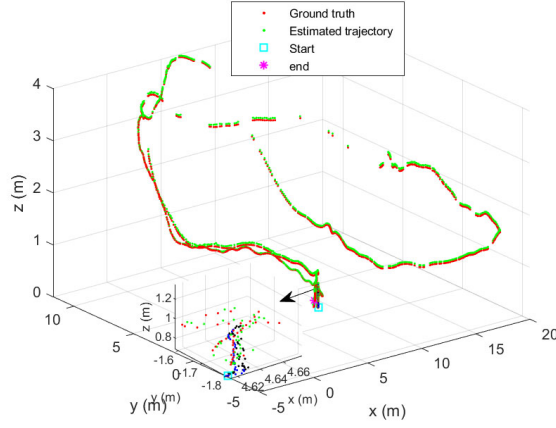


Fig. 6. The comparison between the reference trajectory and the estimated trajectory obtained by applying the proposed calibration model to the ORB-SLAM3 program using the MH-04-difficult dataset as input

improve position estimation accuracy by about 45% compared to without the proposed calibration algorithm.

2) *Applied to LVI-SAM*: LVI-SAM [17] is a tightly coupled fusion system of LiDAR-Vision-Inertial data. In this section, we input the Jackal dataset², run the LVI-SAM program, and compare the position estimation errors. The estimated trajectory of LVI-SAM using the proposed calibration model is shown in Fig. 7.

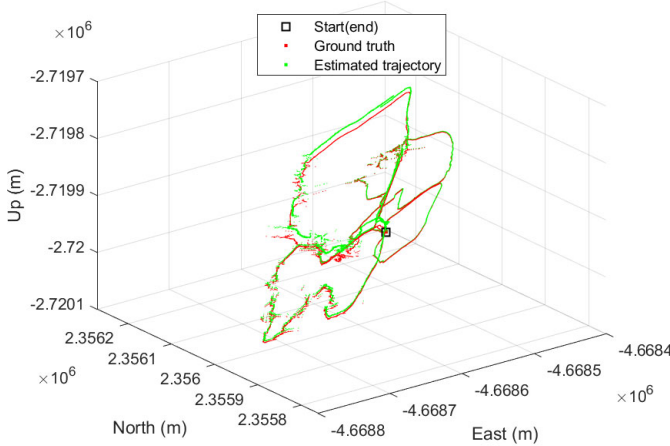


Fig. 7. The comparison between the reference trajectory and the estimated trajectory obtained by applying the proposed calibration model to the LVI-SAM program using the Jackal dataset as input

The RMSE values of the position estimation errors are shown in Table. I. We can conclude that the proposed LiDAR-visual-inertial calibration algorithm can improve the position estimation accuracy by 39% compared to that without the proposed calibration algorithm.

VI. CONCLUSIONS AND FUTURE RESEARCHES

This paper proposes a spatiotemporal calibration algorithm based on nonlinear optimization with the IMU sensor as the central coordinate system. The simulation results demonstrate the feasibility and effectiveness of the proposed algorithm in practical applications. In the future, we will apply the proposed algorithm to a wider range of sensor data to validate its broad applicability.

REFERENCES

- [1] J. Beltrán, C. Guindel, A. de la Escalera, and F. García, "Automatic extrinsic calibration method for lidar and camera sensor setups," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17 677–17 689, 2022.
- [2] 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 Trans. Instrum. Meas.*, vol. 70, pp. 1–11, 2021.
- [3] W. Lee, K. Eickenhoff, P. Geneva, and G. Huang, "Intermittent GPS-aided VIO: Online initialization and calibration," in *Proc. IEEE Int. Conf. Robot. Autom.*, IEEE, 2020, pp. 5724–5731.
- [4] J. Kümmerle, T. Kühner, and M. Lauer, "Automatic calibration of multiple cameras and depth sensors with a spherical target," in *Proc. IEEE/RSSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–8.
- [5] 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/RSSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 5562–5569.
- [6] F. Gierbach, 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.
- [7] 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.
- [8] W. Lee, P. Geneva, Y. Yang, and G. Huang, "Tightly-coupled GNSS-aided visual-inertial localization," in *Proc. IEEE Int. Conf. Robot. Autom.*, IEEE, 2022, pp. 9484–9491.
- [9] 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.
- [10] J. Rehder, R. Siegwart, and P. Furgale, "A general approach to spatiotemporal calibration in multisensor systems," *IEEE Trans. Robot.*, vol. 32, no. 2, pp. 383–398, 2016.
- [11] 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.
- [12] W. DRAFT, "Draft recommended practice for precision centrifuge testing of linear accelerometers," 2009.
- [13] 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 Trans. Geosci. Remote Sens.*, vol. 61, pp. 1–13, 2023.
- [14] 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 Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [15] 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.
- [16] 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.
- [17] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 5692–5698.

²<https://github.com/TixiaoShan/LVI-SAM>

**Authors' Responses to Reviewers' Comments for paper 60974,
"Universal IMU-Centric Spatiotemporal Calibration Algorithm for Heterogeneous
Information"**

by Yanfang Shi, Baowang Lian, Yonghong Zeng, Yugang Ma and Yangyang Liu.

The authors are extremely grateful to the chairs and the reviewers for their careful reading of our manuscript and their helpful comments. We have incorporated these comments and made necessary changes with the highlights in blue typeface in the revision. Please see the detailed responses in this document and the revised manuscript. Below are our responses to the reviewers' comments in the exact order they were listed in the reviews.

Responses to Chairs:

- Q0.0:** *The reviewer(s) have suggested some major revisions to your manuscript. Therefore, I invite you to respond to the reviewer(s)' comments and revise your manuscript.*
- R0.0:** The authors appreciate the chairs for giving us a chance to revise the manuscript. According to your suggestions, we have addressed the reviewers' comments and revised our manuscript.

Responses to Reviewer 1:

- Q1.0:** *This paper proposes a universal spatiotemporal calibration algorithm for heterogeneous sensor data fusion centered around the inertial measurement unit (IMU). The key ideas are: Represent IMU data as continuous trajectories using B-spline interpolation. This allows establishing calibration models between IMU and other sensors irrespective of sensor types. Establish sensor-specific calibration models by predicting measurements using IMU data and minimizing errors between predicted and actual measurements through nonlinear optimization. Detailed models are presented for cameras and LiDARs. Simulation results on public datasets demonstrate improved trajectory estimation accuracy when applying the proposed calibration to existing open source odometry systems. **Strengths:** Scene-agnostic and sensor-agnostic approach allows wide applicability to fusing various heterogeneous sensors. Does not require specialized calibration objects or motions. Detailed calibration models presented for cameras and LiDARs. Approach can likely be extended to other sensors like radar. Solid theoretical grounding in spline interpolation, reprojection errors, point-to-line distances etc. Quantitative simulations demonstrate improved accuracy, validating the proposed ideas.*
- R1.0:** Thanks for carefully reviewing the manuscript and the valuable comments.
- Q1.1:** *No real-world physical experiments to complement the simulations. Actual performance may differ.*

R1.1: The experiments in this paper were conducted under the Robot Operating System (ROS), which is close to a real operating system. Some very popular odometry systems we know, such as ORB-SLAM3 and LVI-SAM, have been validated under the ROS system and have been demonstrated to be effective in practical applications. ROS has wide applications, and the datasets used in the experiments are all from real environments, encompassing many complex and challenging scenarios such as low lighting conditions, weak textures, obstacles, and others.

Q1.2: *Handling of feature extraction and data association (e.g. ORB features for cameras) is glossed over. Noise in these steps can impact optimization.*

R1.2: Thank you for the comment. This paper employs matching ORB feature points. The ORB feature points were introduced in the following reference [1], which corresponds to citation [14] in the article. ORB feature points combine two algorithms: FAST (Features from Accelerated Segment Test) and BRIEF (Binary Robust Independent Elementary Features). The extraction process of FAST feature points involves certain parameters, such as the radius of the circle (3 pixels), brightness percentage (20%), and consecutive count (12). The selection of these parameters is based on the following reference [2]. The parameter involved in the BRIEF algorithm, namely the number of descriptor pairs around the feature point, is 256. This choice is referenced from the following document [3].

[1] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, ORB-SLAM3: An accurate open-source library for visual, visualinertial, and multimap slam, *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874-1890, 2021.

[2] 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. 68.

[3] 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.

The specific process is as follows:

We choose a pixel point and assume its brightness is I_p . We set a threshold value T , for example, 20% of I_p . Taking this pixel as the center, we select 16 pixels on a circle with a radius of 3 pixels, as shown in Fig. 1(a). If consecutive N_1 points are on the chosen circle whose brightness is more significant than $I_p + T$ or less than $I_p - T$, pixel p can be considered a feature point. N_1 is selected as 12. Repeat these four steps for every pixel, performing the same operation.

The computation method of the BRIEF descriptor involves selecting N_2 point pairs around the feature point P in a specific pattern, for example, choosing N_2 as 256. The grayscale comparison results of these N_2 point pairs are combined to form the descriptor, as shown in Fig. 1(b). Therefore, an ORB feature point consists of two components: pixel intensity and descriptor, as shown in Eq. (1).

$$ORB_F = \{I_p, V_p\} \quad (1)$$

Among these, I_p represents the grayscale value of the feature point, while V_p is a vector

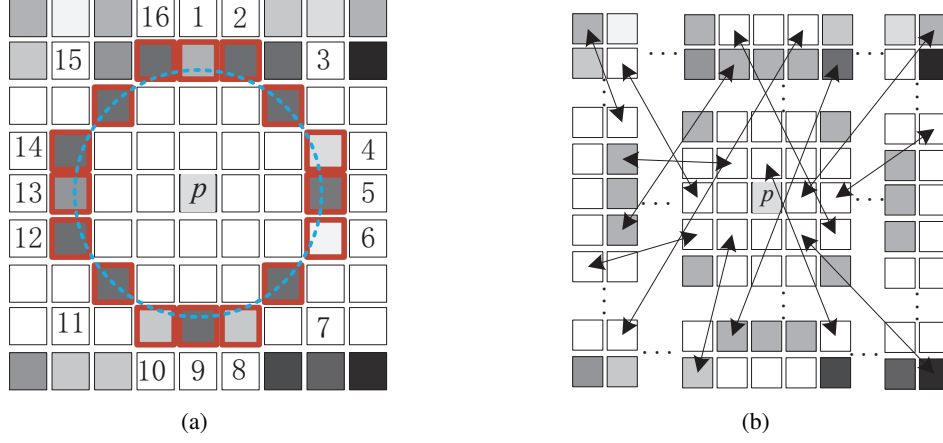


Fig. 1: (a) Illustration of the selection process for FAST feature points; (b) Illustration of the descriptor generation process.

containing 256 elements, each being either 1 or 0. 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 calibration algorithm.

There is certainly noise in the visual data, so we set certain thresholds for the ORB feature matching algorithm. [The extraction of ORB features is not the innovation of this paper. Due to space constraints, it is not detailed in this paper.](#) In the text, Fig.3a and Fig.3b show the ORB feature extraction results for KITTI sequence and ORB feature points matching results of different images at different times in the Malaga dataset respectively.

Q1.3: *Limited details provided on optimization methodology. More specifics on solver, initialization etc. would help reproducibility.*

R1.3: Thank you for the comment. In Section B of the fourth part of the document, a specific elucidation of the nonlinear optimization process is presented. The final cost function is obtained according to the proposed algorithm, as illustrated in Eq. 19 in the text. Based on the Gauss-Newton solving process described in Eq. 20, each unknown coefficient is solved, as detailed in Eq. 21. The specific relationships among coefficients during the solving process are highlighted in the text.

The proposed calibration algorithm can be applied to various combinations of sensors in different fusion systems, selecting initial values based on known physical ranges and definitions.

Q1.4: *No ablation studies to analyze the impact of individual components like spline order, outlier rejection etc.*

R1.4: Thank you for the comment. In our algorithm, we also conducted experiments with various spline interpolation orders. Orders lower than 6 lacked sufficient degrees of freedom, while

orders higher than 6 increased complexity, affecting computational efficiency. Taking these factors into consideration, we chose a spline order of 6. The IMU pose is parameterized as a 6×1 spline, using three degrees of freedom for orientation and three for translation.

Regarding outlier rejection, through the interpolation method, we can filter out a certain number of outlier points, but there may still be some outlier points that cannot be removed, requiring further investigation. The innovation of this paper lies in establishing a spatiotemporal calibration model of other sensor data relative to IMU data. Therefore, the handling of IMU outlier values is not the focus of the research in this paper, and is not extensively discussed.

Q1.5: *Perform physical experiments with an IMU and other sensors rigidly mounted to validate the approach.*

R1.5: Thank you for the comment. The comparative experiments in this paper were conducted on popular open-source odometry systems such as ORB-SLAM3 and LVI-SAM, both implemented in the ROS system, a dedicated operating system for robots. The datasets used were collected in real-world environments, with IMU sensors and other sensors being rigidly connected. In our experiments, we compared the position estimation errors with and without the proposed calibration algorithm, while keeping other conditions consistent. After applying the proposed calibration algorithm, there was an improvement in position estimation accuracy, demonstrating the effectiveness of the proposed calibration algorithm in practical odometry applications. We also conducted comparative experiments on the accuracy of the obtained spatiotemporal parameters and those obtained by existing calibration algorithms. The precision of the proposed calibration algorithm surpasses that of existing calibration algorithms. Due to space constraints, we did not present all the experimental results. The ultimate goal of the proposed calibration algorithm is to be applied in practical odometry systems, so we only showcased the trajectory estimation results of two datasets on two odometry systems.

Q1.6: *Provide more specifics on data association, outlier rejection, optimization details to improve reproducibility. Study the impact of different spline orders, outlier rejection techniques, initialization methods on calibration accuracy. Overall the paper presents a promising sensor-agnostic calibration approach with useful simulation results. With additional experimental validation and implementation details, this could become a valuable contribution towards multi-sensor fusion and localization.*

R1.6: Thank you for the comment. The algorithm centers around the IMU sensor to derive the spatiotemporal parameters of other sensors relative to the IMU. The derivation process of the spatiotemporal calibration model for each sensor data establishes the association between other sensor data and IMU data, which constitutes Section Four of the paper. As for outlier handling, the specifics of nonlinear optimization, and the treatment of initial values, we have addressed them in previous responses.

Responses to Reviewer 2:

Q2.0: *This article proposes a universal spatiotemporal calibration technique with the inertial sensor as the central coordinate system. The technique uses high-order spline interpolation to convert the output data of the sensor, and combines it with the output data of other sensors to establish a calibration model, achieving spatiotemporal calibration of heterogeneous information. The proposed algorithm model is scene independent and achieves sensor type independence by combining temporal and spatial parameters. Therefore, theoretically applicable to any sensor data and applied to some existing open-source odometry systems to verify the feasibility of the algorithm. The work is well written and technologically sound, however, the reviewer thinks that there are some flaws in the simulation and result part that need to be improved before acceptance. Please find specific comments below.*

R2.0: Thank you for your careful review of our manuscript.

Q2.1: *The experimental results show the estimation results of the proposed method in two simulation systems, but no comparative experimental reference is provided. Only the actual values are given. Should comparative data be added to demonstrate the effectiveness of the proposed method?*

R2.1: Thank you for the comment. The comparative experiments in this paper were conducted on popular open-source odometry systems such as ORB-SLAM3 and LVI-SAM, both implemented in the ROS system, a dedicated operating system for robots. The datasets used were collected in real-world environments, with IMU sensors and other sensors being rigidly connected. In our experiments, we compared the position estimation errors with and without the proposed calibration algorithm, while keeping other conditions consistent. After applying the proposed calibration algorithm, there was an improvement in position estimation accuracy, demonstrating the effectiveness of the proposed calibration algorithm in practical odometry applications. We also conducted comparative experiments on the accuracy of the obtained spatiotemporal parameters and those obtained by existing calibration algorithms. The precision of the proposed calibration algorithm surpasses that of existing calibration algorithms. Due to space constraints, we did not present all the experimental results. The ultimate goal of the proposed calibration algorithm is to be applied in practical odometry systems, so we only showcased the trajectory estimation results of two datasets on two odometry systems.

Q2.2: *Figure 6 contains a small coordinate graph. What is the meaning of the simulation results in the figure?*

R2.2: Thank you for the comment. The magnified section in Figure 6 compares partial estimated trajectories with partial reference trajectories, where blue dots represent the starting portion of the reference trajectory, and black dots represent the starting portion of the estimated trajectory. The purpose of the magnified section is to allow readers to clearly see that there is still a certain

gap between our estimated trajectory and the reference trajectory, indicating that position errors still exist. Of course, the accuracy of position estimation in a complete odometry system is influenced by many factors. At least from the results of this experiment, it can be seen that the position estimation accuracy after calibrating with the proposed calibration algorithm is within an acceptable range, which also demonstrates the feasibility of the proposed calibration algorithm.

Q2.3: *The simulation results require more detailed explanations, not just descriptions of experimental data in graphs and tables.*

R2.3: Thank you for the comment. The comparative experiments in this paper were conducted on popular open-source odometry systems such as ORB-SLAM3 and LVI-SAM, both implemented in the ROS system, a dedicated operating system for robots. The datasets used were collected in real-world environments, with IMU sensors and other sensors being rigidly connected. In our experiments, we compared the position estimation errors with and without the proposed calibration algorithm, while keeping other conditions consistent. After applying the proposed calibration algorithm, there was an improvement in position estimation accuracy, demonstrating the effectiveness of the proposed calibration algorithm in practical odometry applications.

Q2.4: *The simulation experiment section of the article did not provide detailed experimental environment parameters.*

R2.4: Thank you very much for your thorough review. The datasets used in the experimental section of this paper are publicly available datasets collected in specific real-world environments, and references are provided in the text. The open-source odometry systems used in the experiments are also popular and cited. The sensor settings for collecting dataset can be found online, but due to space constraints, we did not include them in the paper.

Q2.5: *In addition to content related issues, some textual issues also require the author's attention, such as: 1) For example, there are some text errors on page 3, such as "described in [16] to calculate" – > "described in reference [16] to calculate".*

R2.5: Thank you very much for your careful inspection. We have made modifications according to your suggestions, as highlighted in the text.

Q2.6: *The paragraph containing the formula in the article contains an incorrect use of ".".*

R2.6: Thank you very much for your careful inspection. We have changed $C(\bullet)$ to $C(\cdot)$, as highlighted in the text.

Responses to Reviewer 3:

Q3.0: *1. The background of the spatiotemporal calibration, fusion system, and heterogeneous multi-source information is not adequate, which can be provided more for potential readers.*

R3.0: Thank you for your careful review of our manuscript. Each sensor has its own shortcomings. For the high-precision requirements of the current unmanned systems' position and attitude estimation, it is an inevitable trend to utilize multisource fusion to meet these demands. However, a significant issue with multisource fusion is that each sensor's data preprocessing time differs, resulting in a time offset when reaching the central processor at different times. Moreover, since each sensor is located at a different position on the unmanned vehicle, and each sensor's data is relative to its own coordinate system, it is necessary to transform the data from different sensors into a unified coordinate system. Thus, there exists a spatial parameter. The process of determining the time offset and spatial parameters between different sensors is the spatiotemporal calibration process. Spatiotemporal calibration is the primary prerequisite for achieving high-precision estimation in multisource fusion systems. The spatiotemporal calibration model proposed in this paper arises from such a context.

Q3.1: *The paper lacks relevant parameter information about the specific simulation settings, while the contribution part should be extended to show more insights from the authors' side.*

R3.1: Thank you for the comment. The datasets used in the experimental section of this paper are publicly available datasets collected in specific real-world environments, and references are provided in the text. The open-source odometry systems used in the experiments are also popular and cited. The sensor settings for collecting dataset can be found online, but due to space constraints, we did not include them in the paper.

The calibration algorithm proposed in this paper can provide researchers with broader perspectives, allowing its application to other types of sensors. By utilizing the localization principles and output data of other sensors, we can obtain corresponding optimization factors. Then, we employ the nonlinear optimization algorithm to obtain the spatiotemporal parameters of other sensors relative to the inertial sensor. The paper points out this aspect in the conclusion section, as highlighted in the article.