

Tightly-coupled 6DoF localization in complex environments with GNSS raw data

Yanfang Shi^{1,2}, *Student Member*, Baowang Lian¹, Yonghong Zeng², *Fellow, IEEE*, Ernest Kurniawan², *Senior Member*

Abstract—In large-scale urban environments, precise six-degree-of-freedom (6DOF) pose estimation is essential for vehicles and robots to perform autonomous driving and exploration, as well as to achieve high intelligence and full autonomy of Unmanned Aerial Vehicles (UAV). Achieving 6DOF pose estimation in Global Navigation Satellite System (GNSS)-denied environments is challenging. The performance of relative 6DOF localization systems based on Light Detection and Ranging (LiDAR), vision, and inertial data is easily affected by environmental conditions, leading to error accumulation and a significant decrease in estimation accuracy in complex environments. To address this issue, we propose a tightly coupled framework based on nonlinear optimization for vision, LiDAR, inertial, and GNSS raw data. In the experimental section, we validate the effectiveness of the proposed optimization factor model for GNSS data, LiDAR data, and visual data in improving position and orientation estimation accuracy through simulations. Additionally, we use real datasets to compare the proposed algorithm with several existing open-source programs in terms of computational efficiency, pose estimation accuracy, worst-case scenarios, and reliability. The experimental results show that, although the total processing time increases, the position estimation accuracy and orientation estimation accuracy of the proposed fusion algorithm improve by at least 58.0%. Overall, the proposed tightly-coupled algorithm outperforms the existing methods.

Index Terms—GNSS, Tightly-coupled, Vision, LiDAR, Fusion, Inertial.

I. INTRODUCTION

IN large-scale urban environments, road surfaces inevitably have undulations, with constant inclines and declines. Therefore, vehicles and robots require precise 6DOF pose estimation for autonomous driving and exploration [1]. Additionally, pose estimation of aircraft during taxiing or parking on airport surfaces plays an important role in airport control and handling special situations [2]. Thus, accurate and reliable pose estimation is a primary issue for autonomous driving and navigation of unmanned systems [3].

In scenarios such as dense forests, tunnels, and underground parking garages, GNSS often fails [1]. Relative positioning systems based on LiDAR and vision also have their own

degradation environments. Inertial sensors experience a decline in estimation accuracy due to error accumulation. Therefore, when used alone, these sensors perform inadequately in complex scenarios. Through data fusion algorithms, combining multiple sensors can achieve more accurate and stable pose estimation, enhancing the robustness of autonomous unmanned navigation systems in challenging environments [3].

Here, it is important to note that the concept of orientation proposed in this paper is equivalent to the meaning denoted as attitude in some other papers, while pose and 6DoF localization collectively represent the combination of position and orientation.

A. Related work

Sensor fusion is a crucial approach for achieving accurate positioning and orientation in unmanned systems in harsh environments and has consistently been a topic of research interest among scholars. Broadly, it can be categorized into loosely coupled and tightly coupled methods. Loosely-coupled methods refer to handling the estimation results of the sensors, while tightly-coupled process the measurements from the sensors. Below, we briefly summarize the research work.

1) *LiDAR-based fusion*: Reference [4] proposes a lightweight ground-optimized odometer based on LiDAR data, which can achieve real-time estimation on low-power embedded systems. Compared with LiDAR Odometry and Mapping (LOAM) [5], this method can reduce the computation cost with the same or better accuracy. Tixiao Shan in [6] proposes a framework for tightly-coupled LiDAR and inertial odometry via smoothing and mapping, LiDAR-Inertial Odometry and Mapping with Smoothing and Mapping (LIO-SAM). Scan-matching at a local scale instead of a global scale significantly improves the real-time performance of the system. [7] and [8] design loosely-coupled frameworks based on Visual-LiDAR Odometry and Mapping (V-LOAM). LiDAR-based methods, for example, loosely-coupled [4] and tightly-coupled such as [6], [9], [10] typically fail when operating in structure-less environments or degenerate situations, such as long motorways, tunnels and open spaces [11]. Paper [12] introduces a novel method for high-precision positioning by integrating GNSS, INS, and LiDAR data with trajectory-layer enhancement, significantly improving accuracy in complex environments, but it may face challenges in terms of data fusion complexity and sensor calibration. Paper [13] proposes a continuous-time, tightly coupled integration approach for LiDAR, IMU, and radar odometry, achieving accurate and smooth state estimation, but

This research is supported in part by National Natural Science Foundation of China under Grant 62173276, 62101458, 62171375, 62001392, 61803310 and 61801394. This research is also supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research & Development Programme. 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; bwliao@nwpu.edu.cn).

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

it may face challenges related to computational complexity and sensor calibration.

2) *Visual-based fusion*: Visual-based system, for example, filter-based method [14] and optimization-based pipeline [15]–[18] cannot recover the metric scale and poses. Therefore, [19] proposed an improvement of Oriented FAST and Rotated BRIEF Simultaneous Localization and Mapping (ORB-SLAM) [20] named ORB-SLAM2, adding the support of a stereo camera and RGBD camera for depth estimation. In [15], Antoni Rosinol designs a visual/Inertial odometer framework by constructing grid reconstruction and three-dimensional semantic tags. Compared with ORB-SLAM [20], Visual-Inertial Navigation System Mono (VINS-Mono) [16], and Open Keyframe-based Visual-Inertial SLAM (OKVIS) [21], it improves flexibility and computational efficiency to a certain extent. However, visual-based systems will suffer from bias, noise, and drift in harsh environments [22], which leads to a rapid performance decline in complex environments.

There are also some studies [23]–[26] which use filtering or optimization methods to tightly couple LiDAR plane or point features with visual feature points and introduce the sliding windows to improve robustness and real-time performance further. Nevertheless, there is still a particular gap compared with the requirements in complex environments.

In a word, vision-based methods are inherently affected by light, weather, and viewpoint changes. Their performance deteriorates considerably when the vision sensor fails under severe conditions in harsh environments. LiDAR-based pipelines are sensitive to changes in viewpoint or translation and have poor adaptability. Therefore, none of the LiDAR-based and visual-based methods can provide robust and high-precision position and orientation estimation in harsh environments. Moreover, a severe drawback of the relative methods is that they rely on the relative motion between frames for positioning and orientation estimation. As a result, the previous frame's error accumulates in the next frame, which leads to a decrease in estimation accuracy, so they cannot adapt to the complex environments.

Some researchers began to try combining GNSS absolute positioning and relative methods to improve position and orientation estimation accuracy in harsh environments.

3) *Loosely-coupled fusion involving GNSS*: In [27]–[32], the position values are derived from the GNSS raw data and then fused with visual data. Rui Sun in [33] presents a new comparison-based algorithm of pseudo-range in order to improve GNSS/IMU integrated positioning. Xiao Hui in [34] presents one GNSS/INS/Vision solution, including vision-based detection, tracking, position estimation and optimization. Shaojie Shen in [35] proposes a loosely-coupled pipeline that fuses visual, inertial, LiDAR and GNSS data by Unscented Kalman Filter (UKF) algorithm to produce a smooth and consistent trajectory in different environments. However, in the above methods, GNSS positioning results and other relative methods are loosely coupled, and their performance is greatly affected by GNSS positioning. However, GNSS is susceptible to electromagnetic interference signals and environmental obstructions that hinder achieving precise positioning. Therefore, loosely coupled approaches cannot achieve satisfactory performance in complex environments. Reference [36] integrates

LiDAR, IMU, GNSS, and camera data with 3D digital maps, achieving high localization accuracy and robustness but facing challenges in dynamic environments or areas lacking detailed maps. LGVINS (LiDAR-GPS-Visual and Inertial System) [37] combines LiDAR, GPS, visual, and inertial data for robust UAV state estimation, excelling in accuracy and robustness across diverse environments but facing challenges with sensor integration complexity and reliance on high-quality sensor data. Paper [38] presents a GPS-free, cooperative localization method for UAVs to track a nonstationary target using bearing measurements, with the advantage of distributed execution and guaranteed error decay, but it faces challenges in terms of communication synchronization and the lack of target speed information in some scenarios.

4) *Tightly-coupled fusion involving GNSS*: Therefore, researchers began integrating GNSS raw data and relative methods in a tightly coupled method. In [39]–[41], the Extended Kalman Filter (EKF) method is used to fuse GNSS raw data, visual data, IMU data and [42]–[44] adopt the optimization method to integrate GNSS raw data, visual, IMU data. In [41], the conversion between local coordinates and global coordinates needs to be better addressed; otherwise, it will seriously affect the accuracy of the pose estimation of the fusion system. Shaozu Cao in [45] proposes a coarse-to-fine initialization procedure to efficiently calibrate the transformation online and initialize GNSS states within a short window. Paper [46] presents a TFG-based invariant EKF for tightly-coupled GNSS/INS/LiDAR integration, improving accuracy and convergence, with real-world experiments showing over 35% better positioning accuracy and faster convergence compared to existing methods, but further improvements are needed for robust centimeter-level positioning. The advantage lies in its superior convergence and positioning accuracy, while the main drawback is its limited robustness for centimeter-level precision.

Reference [47] includes tightly coupled and loosely coupled algorithms based on GNSS and IMU. In the loosely coupled algorithm, the pose estimation accuracy of the fusion system is affected by the number of available satellites. In the tightly coupled system, while this limitation is addressed, the Inertial Navigation System (INS) should implement satellite ephemeris equations, requiring more computation by the INS computer, and the pose estimation accuracy of the fusion system is severely impacted by the accumulation of INS errors.

B. Our contribution

Based on the nonlinear optimization [16], we propose a tightly coupled framework incorporating GNSS raw data to address the challenges in complex environments, such as drift, low accuracy, and lack of degrees of freedom. The key contributions are:

- We introduce a tightly coupled framework that first estimates the spatial parameters of the sensor data and then integrates GNSS, vision, LiDAR, and inertial data, leveraging their complementary strengths to achieve continuous and accurate position and orientation estimation in complex environments.

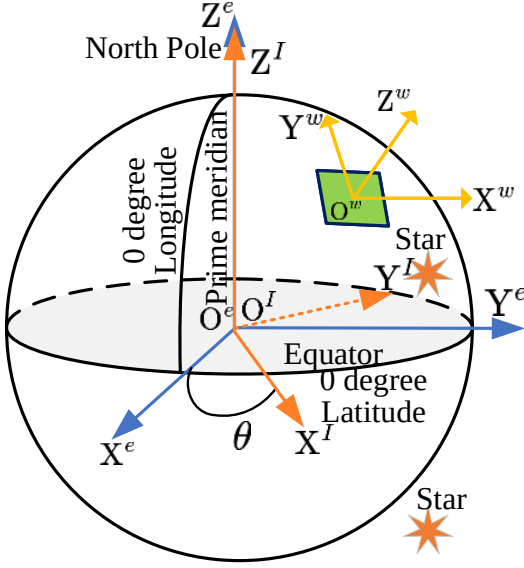


Fig. 1. Schematic diagram of the relationship between celestial coordinate systems

- We design a GNSS data optimization factor based on the pseudo-range calculation formula of GNSS data, allowing the fusion algorithm to leverage the absolute positioning advantage of GNSS even in complex scenarios with only one usable satellite. Additionally, we design an inertial data optimization factor using the preintegration principle of inertial data.
- We design a visual optimization factor model based on the principle of visual reprojection. Additionally, we construct a LiDAR point-cloud optimization factor model using point-to-surface (line) distance errors from LiDAR data. This allows the fusion algorithm to maintain continuous high-precision position and orientation estimation even in complex scenarios with degraded visual data or degraded LiDAR data.

In the simulations, we validate the proposed optimization factor models for GNSS data, LiDAR data, visual data, and inertial data. The results demonstrate the effectiveness of the proposed optimization factor models in improving pose estimation accuracy. Additionally, we use publicly available datasets to compare the proposed fusion algorithm with some existing open-source algorithms in terms of pose estimation accuracy, computational efficiency, consistency, and worst-case scenarios. The results show that the proposed fusion algorithm outperforms the existing open-source algorithms.

II. NOTATION AND DEFINITIONS

A. Coordinate system

The celestial coordinate systems [45] involved in the text are shown in Fig. 1.

- ECI coordinate system (Earth-Centered Inertial, I): The inertial coordinate system remains stationary or moves in a straight line at a constant speed (without acceleration) in space. All inertial devices produce measurement results in the direction of the measurement axis within the inertial

coordinate system. The origin of the coordinate system is the Earth's center of mass, with the z-axis aligned with the Earth's rotational axis pointing towards the geographic North Pole, the x-axis lying in the equatorial plane and pointing towards the vernal equinox, and the y-axis forming a right-handed Cartesian coordinate system.

- ECEF coordinate system (Earth-Centered, Earth-Fixed, e): The Earth-Centered, Earth-Fixed (ECEF) coordinate system shares the same origin and z-axis definition as the ECI coordinate system, but the ECEF coordinate system rotates synchronously with the Earth (hence the name "Earth-Fixed"). The origin coincides with the Earth's center of mass, the z-axis points toward the geographic North Pole, the x-axis points toward the intersection of the equator and the prime meridian (Greenwich Meridian), and the y-axis lies in the equatorial plane, forming a right-handed Cartesian coordinate system with the x-axis and z-axis.
- ENU (East-North-Up, w) Coordinate System: ENU is referred to as the geographic or navigation coordinate system. The origin is the center of the inertial device's coordinate system (the intersection of the three axes of the inertial device), with the y-axis pointing horizontally to true north, the x-axis pointing horizontally to the east, and the z-axis forming a right-handed Cartesian coordinate system with the x-axis and y-axis. The z-axis is perpendicular to the Earth's ellipsoid surface, pointing either upwards (toward the sky) or towards the Earth's center.
- The sensor coordinate systems mentioned in the paper are shown in Fig. 2. In the figure, x_i, y_i, z_i represents the coordinate axes, where c, l, g, b denotes the camera coordinate system (the yellow color), the LiDAR coordinate system (the green color), the GNSS receiver coordinate system (the blue color), and the inertial coordinate system (the pink color, coincides with the carrier's coordinate system), respectively. n is the ENU coordinate system (the purple color), which is also the world coordinate system w for the proposed algorithm.

B. Notation

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. In the proposed algorithm, the spatial parameters involved are $\mathbf{T}_b^g$, $\mathbf{T}_b^l$, and $\mathbf{T}_b^c$, which represent the spatial parameters of the GNSS receiver, LiDAR, and camera relative to the IMU, respectively. $\mathbf{T}_{(k,k+1)}$ represents the pose transformations between different time instances $k, k+1$. To achieve higher precision in pose estimation, we first need to obtain these spatial parameters.

We add a "1" to the end of a three-dimensional vector, transforming it into a four-dimensional vector, which is called homogeneous coordinate, so that the rotation matrix and translation vector form a 4×4 transformation matrix. In the paper, we use homogeneous coordinates and transformation

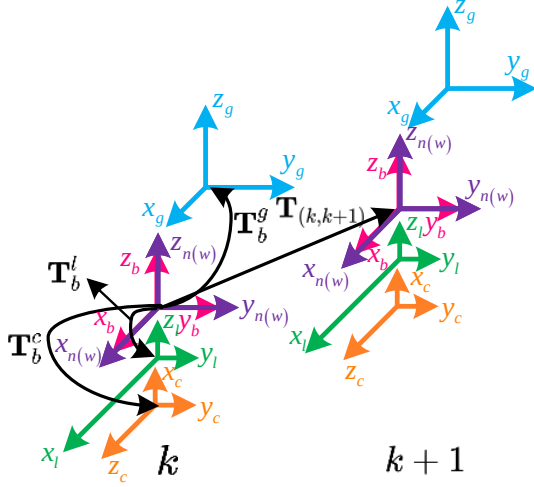


Fig. 2. Schematic diagram of the relationship between sensor coordinate systems

matrices. The forms of $\mathbf{T}_{(k,k+1)}$, $\mathbf{T}_b^c$, $\mathbf{T}_b^l$, $\mathbf{T}_b^g$ are as shown in Eq. (1).

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{\Gamma} \\ \mathbf{0}^\top & 1 \end{bmatrix} \quad (1)$$

The matrix $\mathbf{R}$ represents the rotation matrix of dimensions 3×3 , indicating the carrier's orientation, while the vector $\mathbf{\Gamma}$ with dimensions 3×1 represents the carrier's translation. Additionally, the carrier's rotation matrix can be represented using alternative methods such as quaternions, rotation axes, and Euler angles, and their conversion relationships are detailed in reference [48]. Eq. (2) encapsulates the relationship between two positions, where both $\mathbf{P}_k$ and $\mathbf{P}_{k+1}$ are three-dimensional vectors in moment k and $k+1$ respectively.

$$\begin{bmatrix} \mathbf{P}_{k+1} \\ 1 \end{bmatrix} = \mathbf{T}_{(k,k+1)} \begin{bmatrix} \mathbf{P}_k \\ 1 \end{bmatrix} \quad (2)$$

By using homogeneous coordinates and the transformation matrix, the composition of two transformations can be expressed in linear form.

C. State estimation

We solve the spatial and pose parameters in two steps. First, we determine the spatial parameters of the LiDAR data $\mathbf{T}_b^l$, vision data $\mathbf{T}_b^c$, and GNSS data $\mathbf{T}_b^g$ relative to the IMU. In this step, the states ξ inside the window can be summarized as Eq. (3).

$$\xi = \{\mathbf{x}_{n+1}, \mathbf{x}_{n+2}, \dots, \mathbf{x}_{n+N_1}\}, \quad (3)$$

$$\mathbf{x}_i = \{\mathbf{T}_b^l, \mathbf{T}_b^c, \mathbf{T}_b^g, \mathbf{q}_{b,i}^w, \mathbf{v}_{b,i}^w, \mathbf{P}_{b,i}^w\} \quad (4)$$

$$i \in [n+1, n+N_1] \quad (5)$$

$n+1$ and $n+N_1$ represent the start and end of the sliding window, respectively. N_1 represents the number of keyframes in the sliding window.

Second, building upon the results of the first step, we estimate the pose parameters of the vehicle $\mathbf{T}_{(i,i+1)}$ in Eq. (6).

$$\xi = \{\mathbf{x}_{n+N_1}, \mathbf{x}_{n+N_1+1}, \dots, \mathbf{x}_{n+N_2}\}, \quad (6)$$

$$\mathbf{x}_i = \{\mathbf{q}_{b,i}^w, \mathbf{v}_{b,i}^w, \mathbf{P}_{b,i}^w\} \quad (7)$$

$$i \in [n+N_1, n+N_2] \quad (8)$$

$n+N_1$ and $n+N_2$ represent the start and end of the sliding window, respectively. $N_2 - N_1$ represents the number of keyframes in the sliding window.

III. PROBLEM FORMULATION

Inspired by [16] and [45], consider a certain state ξ and an observation $\mathbf{r}_i$ related to this variable, due to the presence of noise, the observation follows a probabilistic distribution $p(\mathbf{r}_i | \xi)$. When multiple observations are made and each measurement is independent of the others, the likelihood of the measurements $\mathbf{r} = (\mathbf{r}_1, \dots, \mathbf{r}_n)^\top$ is given as follows:

$$p(\mathbf{r} | \xi) = \prod_i p(\mathbf{r}_i | \xi) \quad (9)$$

If prior information about the robot's state $p(\xi)$, such as GPS or wheel encoder data, is known, then according to Bayes' theorem, the posterior probability is given as follows:

$$p(\xi | \mathbf{r}) = \frac{p(\mathbf{r} | \xi)p(\xi)}{p(\mathbf{r})} \quad (10)$$

By using maximum a posteriori (MAP) estimation, the optimal estimate of the system state is obtained as follows:

$$\xi_{\text{MAP}} = \arg \max_{\xi} p(\xi | \mathbf{r}) \quad (11)$$

In the posterior Eq. (11), the denominator is independent of the state variable and can be discarded. The MAP estimation then becomes as follows:

$$\xi_{\text{MAP}} = \arg \max_{\xi} \prod_i p(\mathbf{r}_i | \xi) p(\xi) \quad (12)$$

Thus, we have:

$$\xi_{\text{MAP}} = \arg \min_{\xi} \left[-\sum_i \log p(\mathbf{r}_i | \xi) - \log p(\xi) \right] \quad (13)$$

If we assume that the observations follow a multivariate Gaussian distribution, it is given as follows:

$$p(\mathbf{r}_i | \xi) = \mathcal{N}(\mu_i, \Sigma_i), p(\xi) = \mathcal{N}(\mu_\xi, \Sigma_\xi) \quad (14)$$

Then, we have the following equation:

$$\xi_{\text{MAP}} = \arg \min_{\xi} \sum_i \|\mathbf{r}_i - \mu_i\|_{\Sigma_i}^2 + \|\xi - \mu_\xi\|_{\Sigma_\xi}^2 \quad (15)$$

Thus, we transform the MAP problem into the least squares problem. Then we can apply the nonlinear optimization methods to solve the problem.

From the derivation of the MAP estimation, it is evident that as the trajectory grows, the dimension of the observation matrix increases significantly [49]. To ensure computational efficiency, inspired by the [50], we adopt the sliding window method, which involves selecting a certain amount of data for computation. As the trajectory progresses, new data is

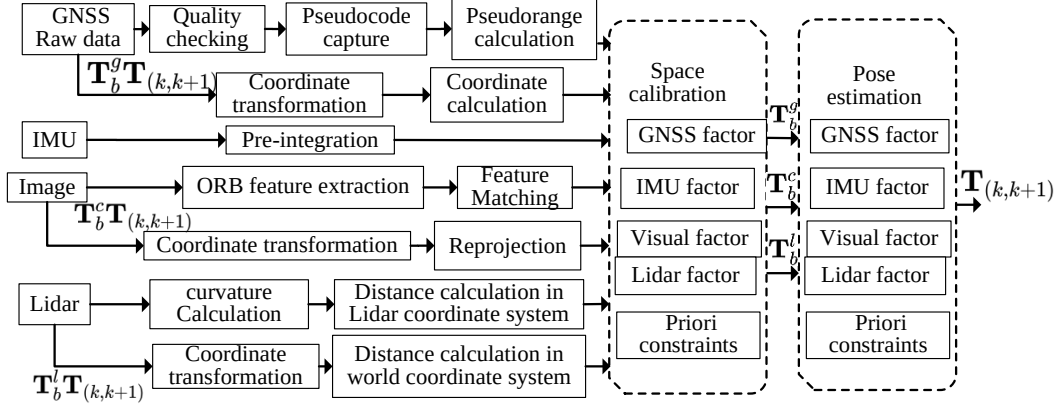


Fig. 3. The flowchart of the tightly coupled algorithm proposed in this paper for GNSS data, LiDAR data, visual data, and inertial data

gradually added while removing old data. However, the data that is removed still has an impact on the current computation. To ensure the accuracy of the computational results, we need to account for this influence.

Therefore, for the state estimation in this paper, the loss function of the proposed algorithm is as follows:

$$\begin{aligned} \xi^* &= \arg \min_{\xi} \{ \|\mathbf{r}_p - \mathbf{J}_p \xi\|_{\Sigma_p}^2 + \sum_i \|\mathbf{f}_i(\mathbf{z}_i)\|_{\Sigma_i}^2 \} \\ &= \arg \min_{\xi} \{ \|\mathbf{r}_p - \mathbf{J}_p \xi\|_{\Sigma_p}^2 + \|\mathbf{f}_b(\mathbf{z}_b)\|_{\Sigma_b}^2 + \|\mathbf{f}_g(\mathbf{z}_g)\|_{\Sigma_g}^2 \\ &\quad + \|\mathbf{f}_v(\mathbf{z}_v)\|_{\Sigma_v}^2 + \|\mathbf{f}_l(\mathbf{z}_l)\|_{\Sigma_l}^2 \} \end{aligned} \quad (16)$$

Where $\mathbf{f}_v^\top(\mathbf{z}_v)$, $\mathbf{f}_l^\top(\mathbf{z}_l)$, $\mathbf{f}_g^\top(\mathbf{z}_g)$ and $\mathbf{f}_b^\top(\mathbf{z}_b)$ represent the errors of the predicted and measured values obtained for the camera, LiDAR, GNSS and inertial data respectively. The discrepancies between the two values are used as an optimization factors that participates in the nonlinear optimization. The establishment of the error model directly affects the accuracy of the estimation. $\mathbf{r}_p$ is the prior information passed to the current optimization process by the variables removed in the sliding window algorithm.

Therefore, the core of the proposed fusion algorithm lies in deriving predictive and measurement models for each sensor data and the prior information.

IV. THE CONSTRUCTION PROCESS OF OPTIMIZATION FACTORS

In this section, we provide a detailed introduction to the prior factor and the optimization factors for the four types of sensor data. The specific workflow of the proposed algorithm is shown in Fig. 3, and the schematic diagram depicting the relationships among the five optimization factors is presented in Fig. 4. It should be noted that the prior factor is not the focus of innovation in this paper. To maintain the completeness, we merely elucidate the specific usage of the prior factor. Detailed derivation methods can be found in [49]–[51].

A. Prior information factor

As the fusion system continues to explore new environments, it encounters new sensor data and environmental features. Consequently, the number of least squares residuals

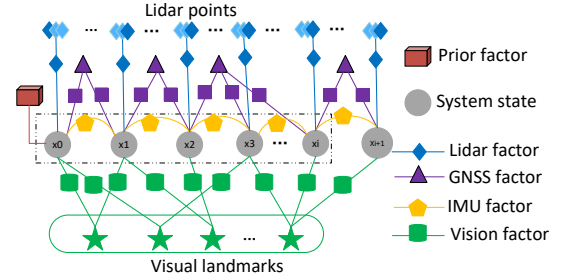


Fig. 4. Schematic diagram depicting the relationships of the five optimization factors involved in the optimization

increases, the information matrix grows larger, and the computational load continues to rise. To keep the number of optimization variables within a manageable range, the sliding window algorithm is needed to dynamically add or remove the optimization variables. Directly discarding variables and their corresponding measurements result in loss of information. The appropriate approach is to use marginal probabilities to transfer the information carried by the discarded variables to the remaining variables.

To provide clear explanations, we use a simple and specific example below to illustrate how the prior information is calculated and incorporated into the new least squares problem.

The least squares system in Eq. (17) corresponds to the factor graph shown in Fig. 5(a).

$$\xi = \arg \min_{\xi} \frac{1}{2} \sum_i \|\mathbf{r}_i\|_{\Sigma_i}^2 \quad (17)$$

where,

$$\xi = \begin{bmatrix} \xi_1 \\ \xi_2 \\ \vdots \\ \xi_6 \end{bmatrix}, \mathbf{r} = \begin{bmatrix} \mathbf{r}_{12} \\ \mathbf{r}_{13} \\ \mathbf{r}_{14} \\ \mathbf{r}_{15} \\ \mathbf{r}_{56} \end{bmatrix} \quad (18)$$

The Gauss-Newton solution corresponding to the above least squares problem is shown as follows:

$$\underbrace{\mathbf{J}^\top \Sigma^{-1} \mathbf{J}}_{\mathbf{H} \text{ or } \Lambda} \delta \xi = \underbrace{\mathbf{J}^\top \Sigma^{-1} \mathbf{r}}_{\mathbf{b}} \quad (19)$$

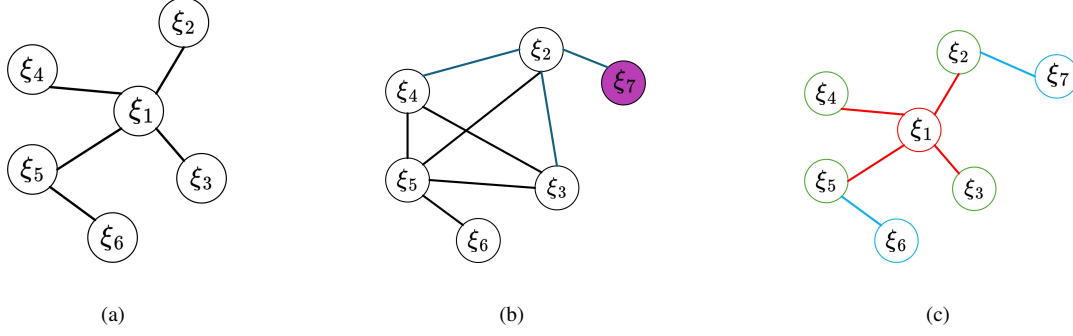


Fig. 5. (a) Circles: represent vertices, the variables that need to be optimized; Edges: represent the residuals constructed between vertices. There are unary edges (connected to a single vertex), binary edges, and multi-ary edges....; (b) The factor graph after adding the new variable ξ_7 ; (c) Red indicates the marginalized variables and measurement constraints; green represents the retained variables related to the marginalized variables; and blue denotes the variables that are not related to the marginalized variables.

Note that, $\Lambda = \Sigma_{new}^{-1} \neq \Sigma^{-1}$ reflects the variance of the solution, while Σ^{-1} reflects the variance of the residuals. Additionally, the Jacobian matrix in the equation is shown as follows:

$$\mathbf{J} = \frac{\partial \mathbf{r}}{\partial \boldsymbol{\xi}} = \begin{bmatrix} \frac{\partial \mathbf{r}_{12}}{\partial \boldsymbol{\xi}} \\ \frac{\partial \mathbf{r}_{13}}{\partial \boldsymbol{\xi}} \\ \frac{\partial \mathbf{r}_{14}}{\partial \boldsymbol{\xi}} \\ \frac{\partial \mathbf{r}_{15}}{\partial \boldsymbol{\xi}} \\ \frac{\partial \mathbf{r}_{56}}{\partial \boldsymbol{\xi}} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \\ \mathbf{J}_3 \\ \mathbf{J}_4 \\ \mathbf{J}_5 \end{bmatrix}, \mathbf{J}^\top = [\mathbf{J}_1^\top \quad \mathbf{J}_2^\top \quad \mathbf{J}_3^\top \quad \mathbf{J}_4^\top \quad \mathbf{J}_5^\top] \quad (20)$$

The matrix multiplication in Eq. (19) can be written as a summation, as shown in Eq. (21).

$$\sum_{i=1}^5 \mathbf{J}_i^\top \Sigma_i^{-1} \mathbf{J}_i \delta \boldsymbol{\xi} = - \sum_{i=1}^5 \mathbf{J}_i^\top \Sigma_i^{-1} \mathbf{r}_i \quad (21)$$

Since each residual is only related to certain state variables, the Jacobian matrix of unrelated terms will be zero when taking derivatives. For example:

$$\mathbf{J}_2 = \frac{\partial \mathbf{r}_{13}}{\partial \boldsymbol{\xi}} = \begin{bmatrix} \frac{\partial \mathbf{r}_{13}}{\partial \xi_1} & 0 & \frac{\partial \mathbf{r}_{13}}{\partial \xi_3} & 0 & 0 & 0 \end{bmatrix} \quad (22)$$

$$\Lambda_2 = \mathbf{J}_2^\top \Sigma_2^{-1} \mathbf{J}_2 \quad (23)$$

$$= \begin{bmatrix} \left(\frac{\partial \mathbf{r}_{13}}{\partial \xi_1} \right)^\top \Sigma_2^{-1} \frac{\partial \mathbf{r}_{13}}{\partial \xi_1} & 0 & \left(\frac{\partial \mathbf{r}_{13}}{\partial \xi_1} \right)^\top \Sigma_2^{-1} \frac{\partial \mathbf{r}_{13}}{\partial \xi_3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \left(\frac{\partial \mathbf{r}_{13}}{\partial \xi_3} \right)^\top \Sigma_2^{-1} \frac{\partial \mathbf{r}_{13}}{\partial \xi_1} & 0 & \left(\frac{\partial \mathbf{r}_{13}}{\partial \xi_3} \right)^\top \Sigma_2^{-1} \frac{\partial \mathbf{r}_{13}}{\partial \xi_3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (24)$$

Similarly, $\Lambda_1, \Lambda_3, \Lambda_4, \Lambda_5$ can be computed and they are also sparse.

The observational information between the new variable ξ_7 and the old variable ξ_2 form the residual $\mathbf{r}_{27}$. The factor graph is shown in Fig. 5(b). The new residual, combined with the information left before marginalization, constructs a new least squares system.

As shown in Fig. 5(c), at time $t \in [0, k]$, the system state is $\xi_i, i \in [1, 6]$. At time $k+1$, new measurements and state variables ξ_7 are added.

At time k , after completing the least squares optimization, the variable ξ_1 is marginalized. The marginalized state is denoted as $\boldsymbol{\varpi}_m$, and the remaining variables $\xi_i, i \in [2, 5]$ are denoted as $\boldsymbol{\varpi}_r$. After marginalization, all variables and corresponding measurements of $\boldsymbol{\varpi}_m$ will be discarded. At the same time, this information is transferred to the retained variables $\boldsymbol{\varpi}_r$ through the marginalization. Here, the information from the marginalized variables is not related to ξ_6 . At time $k+1$, new state variables ξ_7 (denoted as $\boldsymbol{\varpi}_n$) and the corresponding observations are introduced, and a new round of least squares optimization begins.

Before marginalization, the least squares information matrix constructed from variable $\boldsymbol{\varpi}_m$ and the corresponding measurement S_m is as follows:

$$\Lambda_m(k) = \begin{bmatrix} \Lambda_{mm}(k) & \Lambda_{mr}(k) \\ \Lambda_{rm}(k) & \Lambda_{rr}(k) \end{bmatrix} = \sum_{i,j \in S_m} \mathbf{J}_{ij}^\top(k) \Sigma_{ij}^{-1} \mathbf{J}_{ij}(k) \quad (25)$$

$$\mathbf{b}_m(k) = \begin{bmatrix} \mathbf{b}_{mm}(k) \\ \mathbf{b}_{mr}(k) \end{bmatrix} = - \sum_{i,j \in S_m} \mathbf{J}_{ij}^\top(k) \Sigma_{ij}^{-1} \mathbf{r}_{ij}(k) \quad (26)$$

After marginalization, the measurement information of variable $\boldsymbol{\varpi}_m$ is transferred to variable $\boldsymbol{\varpi}_r$:

$$\Lambda_p(k) = \Lambda_{rr}(k) - \Lambda_{rm}(k) \Lambda_{mm}^{-1}(k) \Lambda_{mr}(k) \quad (27)$$

$$\mathbf{b}_p(k) = \mathbf{b}_{mr}(k) - \Lambda_{rm}(k) \Lambda_{mm}^{-1}(k) \mathbf{b}_{mm}(k) \quad (28)$$

The subscript p denotes *prior*, meaning that this information constructs a prior for variable $\boldsymbol{\varpi}_r$.

At time $k+1$, the new residual $\boldsymbol{\varpi}_{27}$, prior information $\mathbf{b}_p(k), \Lambda_p(k)$, and residual $\mathbf{r}_{56}$ construct a new least squares problem, as follows:

$$\Lambda(k+1) = \Pi^\top \Lambda_p(k) \Pi + \sum_{i,j \in S_a(k+1)} \mathbf{J}_{ij}^\top(k+1) \Sigma_{ij}^{-1} \mathbf{J}_{ij}(k+1) \quad (29)$$

$$\mathbf{b}(k+1) = \Pi^\top \mathbf{b}_p(k) - \sum_{i,j \in S_a(k+1)} \mathbf{J}_{ij}^\top(k+1) \Sigma_{ij}^{-1} \mathbf{r}_{ij}(k+1) \quad (30)$$

Where $\Pi = [\mathbf{I}_{\text{dim}} \ \boldsymbol{\omega}_r \ 0]$ is used to expand the dimensions of the matrix. S_a represents the measurements other than those marginalized, such as $\mathbf{r}_{56}, \mathbf{r}_{27}$.

B. Inertial factor model

According to reference [16], the optimization factor for the inertial sensor is given by Eq. (31).

$$\mathbf{f}_b(\mathbf{z}_b) = \begin{bmatrix} \mathbf{R}_k^w (\mathbf{P}_{k+1}^w - \mathbf{P}_k^w - \mathbf{v}_k^w \Delta t + \frac{1}{2} \mathbf{g}^w \Delta t^2) - \hat{\alpha}_{k,k+1} \\ \mathbf{R}_k^w (\mathbf{v}_{k+1}^w - \mathbf{v}_k^w + \mathbf{g}^w \Delta t) - \hat{\beta}_{k,k+1} \\ 2 [\mathbf{q}_k^{w-1} \otimes \mathbf{q}_{k+1}^w \otimes (\hat{\gamma}_{(k,k+1)})^{-1}]_{xyz} \\ \chi_{a,k+1} - \chi_{a,k} \\ \chi_{\tau,k+1} - \chi_{\tau,k} \end{bmatrix} \quad (31)$$

Where $\hat{\alpha}_{k,k+1}$, $\hat{\beta}_{k,k+1}$, and $\hat{\gamma}_{k,k+1}$ are preintegration factors, as shown in Eq. (32), Eq. (33) and Eq. (34).

$$\hat{\alpha}_{k,k+1} = \iint_{t \in [k,k+1]} \mathbf{R}_t^k (\hat{\mathbf{a}}_t - \chi_{a,t} - \mathbf{n}_a) dt^2 \quad (32)$$

$$\hat{\beta}_{k,k+1} = \int_{t \in [k,k+1]} \mathbf{R}_t^k (\hat{\mathbf{a}}_t - \chi_{a,t} - \mathbf{n}_a) dt \quad (33)$$

$$\hat{\gamma}_{k,k+1} = \int_{t \in [k,k+1]} \frac{1}{2} \boldsymbol{\Omega}(\hat{\boldsymbol{\tau}}_t - \chi_{\tau,t} - \mathbf{n}_\tau) dt \quad (34)$$

where,

$$\boldsymbol{\Omega}(\boldsymbol{\tau}) = \begin{bmatrix} \boldsymbol{\tau}^\wedge & \boldsymbol{\tau} \\ -\boldsymbol{\tau}^\top & 0 \end{bmatrix}, \boldsymbol{\tau}^\wedge = \begin{bmatrix} 0 & -\tau_z & \tau_y \\ \tau_z & 0 & -\tau_x \\ -\tau_y & \tau_x & 0 \end{bmatrix} \quad (35)$$

The symbol $[\cdot]_{xyz}$ represents the imaginary part of the quaternion. Where the parameters k and $k+1$ represent preintegration's starting and ending moments. Multiple sets of IMU data exist between k and $k+1$ moment respectively. $\mathbf{P}_{k+1}^w$, $\mathbf{v}_k^w$ and $\mathbf{R}_k^w$ are translation, velocity and rotation in the world coordinate system in k moment respectively. $\mathbf{g}^w$ is the gravity vector in the world frame. χ_a and χ_τ are accelerometer bias and gyroscope bias respectively.

The inertial coordinate system coincides with the carrier coordinate system in the proposed fusion system. The carrier's world coordinate value at the previous moment is the world coordinate of the inertial sensor $\mathbf{P}_k^w$. We multiply this value by rotation and translation $\mathbf{T}_{(k,k+1)}$ and get the world coordinates of the inertial sensor at $k+1$ moment $\mathbf{P}_{k+1}^w$, as shown in Eq. (36).

$$\begin{bmatrix} \mathbf{P}_{k+1}^w \\ 1 \end{bmatrix} = \mathbf{T}_{(k,k+1)} \begin{bmatrix} \mathbf{P}_k^w \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \boldsymbol{\Gamma} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_k^w \\ 1 \end{bmatrix} \quad (36)$$

Then, by taking the derivative of $\mathbf{P}_{k+1}^w$, we can obtain the velocity at time $k+1$, as shown in Eq. (37).

$$\dot{\mathbf{P}}_{k+1}^w = \mathbf{v}_{k+1}^w \quad (37)$$

Combining with the rotation matrix $\mathbf{R}_k^w$ and using the conversion relationships between rotation matrices and quaternions [48], we can determine the quaternion at time $k+1$. Subsequently, plugging in the translation, velocity, and quaternion $\mathbf{q}_k^w$ at time $k+1$ into Eq. (31), we can get the error model for the inertial data $\mathbf{f}_b(\mathbf{z}_b)$.

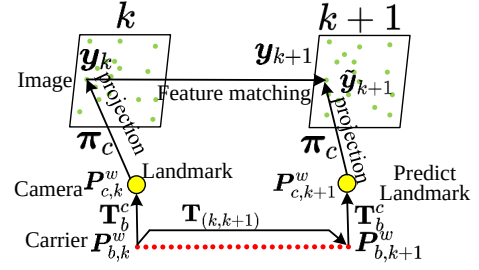


Fig. 6. Schematic diagram of the construction process of visual factor

C. Vision factor model

The construction process of the visual factor is shown in Fig. 6. At time $k+1$, the pixel coordinates can be directly measured. Meanwhile, using the principle of visual re-projection, pixel coordinates can be obtained by calculating the three-dimensional landmark points, allowing for the acquisition of the error between the predicted and measured pixel values.

We first extract Oriented FAST and Rotated BRIEF (ORB) feature points [18] 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: Features from Accelerated Segment Test (FAST) [52] and Binary Robust Independent Elementary Features (BRIEF) [53].

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. 7(a) shows the ORB feature extraction results for KITTI 00 [54] sequence. Fig. 7(b) is the matching result of ORB feature points. It can be observed that there are errors in the matching results. In our paper, We combine the Hamming distance (45) and the distance ratio (0.75) to filter valid matches and reduce the number of incorrect matches.

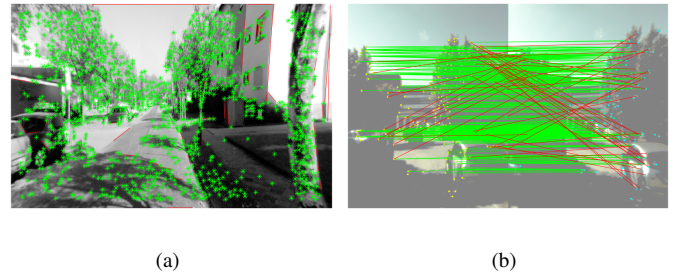


Fig. 7. (a) ORB feature point extraction results of a specific image in the KITTI dataset; (b) ORB feature points matching results of different images at different time in Malaga dataset.

Firstly, we multiply the world coordinate position of the carrier at the k moment $\mathbf{P}_{b,k}^w$ by the spatial parameter $\mathbf{T}_b^c$ between the inertial sensor and the camera. We can find the position of the camera in the world coordinate system at the k time $\mathbf{P}_{c,k}^w$. Through the projection algorithm, we can get the

pixel coordinates of this landmark on the k picture, as shown in Eq. (38).

$$\mathbf{y}_k = \pi_c(\mathbf{P}_{c,k}^w) = \pi_c(\mathbf{T}_b^c \mathbf{P}_{b,k}^w) \quad (38)$$

where, π_c is the projection function of the camera. Then, we adopt the feature points matching method [18] to find the pixel coordinates $\mathbf{y}_{k+1}$ of the $k+1$ moment corresponding to the pixel $\mathbf{y}_k$ of the k moment. As shown in Eq. (39), this projection point is the actual measurement.

$$\mathbf{y}_{k+1} \Leftrightarrow \text{Match}(\mathbf{y}_k) \quad (39)$$

Meanwhile, we multiply the world coordinate position of the carrier of the k time $\mathbf{P}_{b,k}^w$ by rotation and translation of the carrier between the two moments $\mathbf{T}_{(k,k+1)}$. We can obtain the world coordinate position of the carrier at the $k+1$ time $\mathbf{P}_{b,k+1}^w$. In the same way as the k moment, we can also get the pixel coordinates of the camera on the $k+1$ picture, as shown in Eq. (40). We take this projection point as the predicted value.

$$\tilde{\mathbf{y}}_{k+1} = \pi_c(\mathbf{P}_{c,k+1}^w) = \pi_c(\mathbf{T}_b^c \mathbf{T}_{(k,k+1)} \mathbf{P}_{b,k}^w) \quad (40)$$

Therefore, the vision optimization factor is as shown in Eq. (41).

$$\begin{aligned} \mathbf{f}_v(\mathbf{z}_v) &= \sum_{i=1}^F (\tilde{\mathbf{y}}_{i+1} - \mathbf{y}_{i+1}) \\ &= \sum_{i=1}^F \{ \pi_c(\mathbf{T}_b^c \mathbf{T}_{(i,i+1)} \mathbf{P}_{b,i}^w) - \text{Match}(\pi_c(\mathbf{T}_b^c \mathbf{P}_{b,i}^w)) \} \end{aligned} \quad (41)$$

F represents the number of landmark points in the sliding window.

D. LiDAR factor model

The three-dimensional coordinates of the **LiDAR point cloud** can be obtained directly in the LiDAR coordinate system. According to [5], we can obtain the curvature values of the **point cloud** as shown in Eq. (42).

$$\eta = \frac{1}{|S| \cdot \|\mathbf{P}_{\kappa,\nu}^L\|} \left\| \sum_{o \in S, o \neq \nu} (\mathbf{P}_{\kappa,\nu}^L - \mathbf{P}_{\kappa,o}^L) \right\| \quad (42)$$

where, L represents the LiDAR coordinate system, κ represents the sweep moment, ν represents one of the point clouds during the κ sweep moment. S is the set of consecutive points of ν returned by the LiDAR scanner in the same scan. η is the curvature value of the ν point of the sweep κ .

We extract a certain number of corner and plane points based on the maximum and minimum curvature values and set specific thresholds. Then, we calculate the distances of the corner points or plane points to their corresponding lines or surfaces using Eq. (43) and Eq. (44), respectively.

$$d_\varepsilon = \frac{|(\mathbf{P}_{(\kappa+1,\mu)}^L - \mathbf{P}_{(\kappa,\nu)}^L) \times (\mathbf{P}_{(\kappa+1,\mu)}^L - \mathbf{P}_{(\kappa,o)}^L)|}{|\mathbf{P}_{(\kappa,\nu)}^L - \mathbf{P}_{(\kappa,o)}^L|} \quad (43)$$

$$d_\zeta = \frac{\left| \frac{(\mathbf{P}_{(\kappa,\nu)}^L - \mathbf{P}_{(\kappa,o)}^L) \times (\mathbf{P}_{(\kappa,\nu)}^L - \mathbf{P}_{(\kappa,h)}^L)}{(\mathbf{P}_{(\kappa,\nu)}^L - \mathbf{P}_{(\kappa,o)}^L) \times (\mathbf{P}_{(\kappa,o)}^L - \mathbf{P}_{(\kappa,h)}^L)} \right|}{\left| \frac{(\mathbf{P}_{(\kappa+1,\mu)}^L - \mathbf{P}_{(\kappa,\nu)}^L) \times (\mathbf{P}_{(\kappa+1,\mu)}^L - \mathbf{P}_{(\kappa,o)}^L)}{(\mathbf{P}_{(\kappa,\nu)}^L - \mathbf{P}_{(\kappa,o)}^L) \times (\mathbf{P}_{(\kappa,o)}^L - \mathbf{P}_{(\kappa,h)}^L)} \right|} \quad (44)$$

Where $\mathbf{P}_{\kappa,\nu}^L$ represents the three-dimensional coordinate of the **point cloud** in the LiDAR coordinate system.

The distance should remain the same in different coordinate systems. We can directly obtain the distance values in the LiDAR coordinate system by substituting the **point cloud** coordinates into Eq. (43) and Eq. (44), which serves as the actual measurement values, we denote it as $d_\varepsilon(d_\zeta)_k$.

At k moment, the position of the carrier (the inertial sensor's position) $\mathbf{P}_{b,k}^w$ in the world coordinate system is known. Multiplying it by the carrier's transformation $\mathbf{T}_{(k,k+1)}$ yields the carrier's position at $k+1$ moment in the world coordinate system, as shown in Eq. (45).

$$\mathbf{P}_{b,k+1}^w = \mathbf{T}_{(k,k+1)} \mathbf{P}_{b,k}^w \quad (45)$$

Subsequently, multiplying this value by the transformation parameters $\mathbf{T}_b^l$ between the LiDAR sensor and the inertial sensor provides the LiDAR sensor's position at time $k+1$ in the world coordinate system $\mathbf{P}_{l,k+1}^w$, as shown in Eq. (46).

$$\mathbf{P}_{l,k+1}^w = \mathbf{T}_b^l \mathbf{T}_{(k,k+1)} \mathbf{P}_{b,k}^w \quad (46)$$

Now, we obtain the predicted position of the **LiDAR point cloud** $\mathbf{P}_{k+1,\nu}^w$ at time $k+1$ based on the predicted position of the LiDAR sensor in the world coordinate system $\mathbf{P}_{l,k+1}^w$. Fig. 8 illustrates the relationship between the LiDAR sensor and the **point cloud** positions.

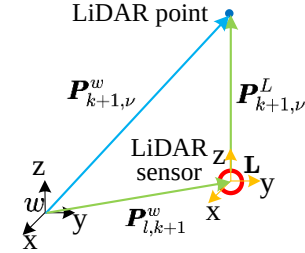


Fig. 8. Illustration of the relationship between LiDAR sensor coordinates and **point cloud** coordinates

In the LiDAR coordinate system, the coordinate of the LiDAR sensor is at the origin of the LiDAR sensor. Therefore, the **LiDAR point cloud** coordinates in the LiDAR coordinate system represent the offset of the **point cloud** relative to the LiDAR sensor. So, the coordinates of the **LiDAR point cloud** in the world coordinate system are obtained by adding the coordinates of the LiDAR sensor and the coordinates of the **LiDAR point cloud**, as shown in Eq. (47).

$$\mathbf{P}_{k+1,\nu}^w = \mathbf{P}_{k+1,\nu}^L + \mathbf{P}_{l,k+1}^w \quad (47)$$

Combining with Eq. (46), the coordinates of the **LiDAR point cloud** in the world coordinate system at time $k+1$ is as shown in Eq. (48).

$$\mathbf{P}_{k+1,\nu}^w = \mathbf{P}_{k+1,\nu}^L + \mathbf{T}_b^l \mathbf{T}_{(k,k+1)} \mathbf{P}_{b,k}^w \quad (48)$$

By substituting Eq. (47) and Eq. (48) into Eq. (43) and Eq. (44), we can obtain the distance of the **LiDAR point cloud** to the line (plane) in the world coordinate system. We refer to this as the predicted value and denote it as $d_\varepsilon(d_\zeta)_{k+1}$.

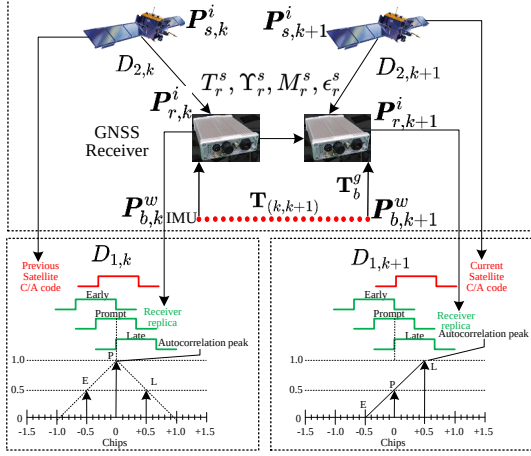


Fig. 9. Illustration of the calculation process for GNSS optimization factor

Therefore, the optimization factor of the **LiDAR point cloud** is shown in Eq. (49).

$$f_l(z_l) = \sum_{i=1}^{\varkappa} (d_{\varepsilon_k} - d_{\varepsilon_{k+1}}) + \sum_{j=1}^{\varsigma} (d_{\zeta_k} - d_{\zeta_{k+1}}) \quad (49)$$

It should be noted that based on the curvature values of the **point cloud**, we set a certain threshold and select the **point cloud** that falls within this threshold range. $\varkappa$ and ς represent the number of corner points and surf points, respectively.

E. GNSS factor model

The construction process of the GNSS optimization factor is shown in Fig. 9. First, we perform correlation operations between the pseudocode received by the receiver and the replicated pseudocode to find the correlation peak. This allows us to determine the satellite signal transmission time. By subtracting this from the time the receiver received the satellite signal, we obtain a time difference. Multiplying this time difference by the speed of electromagnetic waves gives the pseudorange. Additionally, the coordinates of the satellite are known. We assume the position of the receiver, and then we can use the distance errors caused by various delays in the ephemeris to obtain the pseudorange. Then we obtain the pseudorange observation formula, for the detailed introduction, please refer to [55] and [56]. We take the difference between the pseudorange equations at two different moments to obtain the GNSS optimization factor.

According to [55] and [56], the pseudorange observation equation is shown in Eq. (50).

$$\Theta = \|\mathbf{P}_s^I - \mathbf{P}_r^I\| + O_r^s + T_r^s + \Upsilon_r^s + M_r^s + v_c \delta_r + \epsilon_r^s \quad (50)$$

where O_r^s includes orbit errors and satellite clock errors, which are monitored and continuously corrected by the system control module; Υ_r^s and T_r^s represent the errors caused by the ionosphere and troposphere's impact on the propagation path, respectively; M_r^s is the error caused by the multipath effect; Υ_r^s , T_r^s , and M_r^s are all included in the ephemeris parameters; ϵ_r^s is the delay caused by Gaussian white noise, and $v_c \delta_r$ is

the delay caused by the receiver's clock error. v_c is the speed of electromagnetic wave.

According to [55] and [56], we can derive the corresponding formulation for the transmission time of the GPS signal k_s , as shown in Eq. (51).

$$k_s = \text{TOW} + (30\mathfrak{J} + \mathfrak{T}) \times 0.020 + (\mathfrak{J} + \frac{\text{CP}}{1023} \times 0.001) \quad (51)$$

Where TOW is the time of week in seconds, corresponding to the GPS time at the start of the next subframe; $\mathfrak{J}$ is the number of words in the navigation message data; $\mathfrak{T}$ is the number of bits in the navigation message; $\mathfrak{J}$ is the integer number of C/A code cycles in the navigation message; and CP is the code phase measurement value of the received signal fragment in the current C/A code cycle.

According to [55] and [56], the pseudorange is given by Eq. (52):

$$\Theta = v_c(k_r - (k_s + \delta(k_s))) \quad (52)$$

where k_r is the time when the receiver receives the signal, and $\delta(k_s)$ is the satellite clock error, with $\delta(k_s)$ included in the ephemeris.

In Eq. (50), $\|\mathbf{P}_s^I - \mathbf{P}_r^I\|$ represents the geometric distance between the satellite and the receiver. $\mathbf{P}_s^I$ and $\mathbf{P}_r^I$ denote the ECI coordinates of the satellite and the receiver, respectively, when the satellite signal arrives at the receiver. I represents the ECI coordinates. The ECEF coordinates of the satellite at the transmission time $\mathbf{P}_s^e$ can be obtained from the ephemeris. The ECEF coordinates of the satellite at the reception time can be determined using Eq. (53).

$$\mathbf{P}_s^e = \mathbf{R}_z(\theta) \mathbf{P}_s^e \quad (53)$$

Where $\mathbf{R}_z(\theta)$ represents a rotation about the z axis of the ECEF frame with magnitude θ [45]. θ is the angle of Earth's rotation during the time interval between the satellite signal transmission and reception, as shown in Eq. (54).

$$\theta = \omega_e(k_r - (k_s + \delta(k_s))) \quad (54)$$

In this paper, we assume that the ECEF and ECI coordinate systems coincide at the moment when the satellite signal is received. Therefore, by combining Eq. (53) and Eq. (54), we obtain the ECI coordinates of the satellite at the reception time.

At time $k+1$, we have the following equation:

$$\Theta_{k+1} = \|\mathbf{P}_{s,k+1}^I - \mathbf{P}_{r,k+1}^I\| + O_r^{s,k+1} + T_r^{s,k+1} + \Upsilon_r^{s,k+1} + M_r^{s,k+1} + v_c \delta_r^{k+1} + \epsilon_r^{s,k+1} \quad (55)$$

At time k , there are the following formula, as shown in Eq. (56).

$$\Theta_k = \|\mathbf{P}_{s,k}^I - \mathbf{P}_{r,k}^I\| + O_r^{s,k} + T_r^{s,k} + \Upsilon_r^{s,k} + M_r^{s,k} + v_c \delta_r^k + \epsilon_r^{s,k} \quad (56)$$

where,

$$\begin{aligned} \mathbf{P}_{s,k+1}^I &= \mathbf{R}_z(\theta_1) \mathbf{P}_{s,k+1}^e, \mathbf{P}_{r,k+1}^I = \mathbf{R}_w^I \mathbf{P}_{r,k+1}^w \\ \mathbf{P}_{s,k}^I &= \mathbf{R}_z(\theta_2) \mathbf{P}_{s,k}^e, \mathbf{P}_{r,k}^I = \mathbf{R}_w^I \mathbf{P}_{r,k}^w \\ \theta_1 &= \omega_e((k+1)_r - ((k+1)_s + \delta((k+1)_s))) \\ \theta_2 &= \omega_e(k_r - (k_s + \delta(k_s))) \end{aligned} \quad (57)$$

In our paper, we consider the ENU coordinate system as the world coordinate system, and we align the ECI and ECEF coordinate systems at the time of reception. Therefore, $\mathbf{R}_w^I$ is equivalent to $\mathbf{R}_e^n$. According to [45], $\mathbf{R}_e^n$ is shown in Eq. (58).

$$\mathbf{R}_e^n = \begin{bmatrix} -\sin \lambda & -\sin \phi \cos \lambda & \cos \phi \cos \lambda \\ \cos \lambda & -\sin \phi \sin \lambda & \cos \phi \sin \lambda \\ 0 & \cos \phi & \sin \phi \end{bmatrix} \quad (58)$$

λ and ϕ denote the reference point's longitude and latitude.

Then, we obtain the coordinates of the GNSS receiver in the world coordinate system at moments $k+1$ and k using Eq. (59).

$$\mathbf{P}_{r,k+1}^w = \mathbf{T}_b^g \mathbf{T}_{(k,k+1)} \mathbf{P}_{b,k}^w, \mathbf{P}_{r,k}^w = \mathbf{T}_b^g \mathbf{P}_{b,k}^w \quad (59)$$

We substitute Eq. (57), Eq. (58) and Eq. (59) into Eq. (55) and Eq. (56), and then take the difference between Eq. (55) and Eq. (56), as shown in Eq. (60).

$$\begin{aligned} & \Theta_{k+1} - \Theta_k = \\ & \|\mathbf{R}_z(\theta_1) \mathbf{P}_{s,k}^e - \mathbf{R}_w^i \mathbf{T}_b^g \mathbf{P}_{b,k}^w\| \\ & + O_r^{s,k+1} + T_r^{s,k+1} + \Upsilon_r^{s,k+1} \\ & + M_r^{s,k+1} + c\delta_r^{k+1} + \epsilon_r^{s,k+1} \\ & - \|\mathbf{R}_z(\theta_2) \mathbf{P}_{s,k}^e - \mathbf{R}_w^i \mathbf{T}_b^g \mathbf{T}_{(k,k+1)} \mathbf{P}_{b,k}^w\| \\ & - O_r^{s,k} - T_r^{s,k} - \Upsilon_r^{s,k} \\ & - M_r^{s,k} - c\delta_r^k - \epsilon_r^{s,k} \end{aligned} \quad (60)$$

For the same receiver, the clock error is equal within a certain period, meaning that $\delta_r^{k+1} = \delta_r^k$. By subtracting the right-hand side from the left-hand side of Eq. (60), we can obtain the optimization factor for GNSS data $\mathbf{f}_g(\mathbf{z}_g)$.

V. THE PROCESS OF ESTIMATING SPATIAL PARAMETERS AND POSE PARAMETERS

The proposed algorithm employs a factor graph-based non-linear optimization approach. Through the analysis above, we obtain the optimization factors for GNSS raw data, LiDAR data, and visual data, which define the loss function for the algorithm. As discussed in section "state estimation", we can transform the state estimation problem into a least-squares problem. It is important to note that we use the Gauss-Newton method improved by Levenberg and Marquardt (LM), which introduces a damping factor during the solution process [57], [58], as shown below:

$$(\mathbf{J}^\top \mathbf{J} + \vartheta \mathbf{I}) \Delta \mathbf{x}_{\text{lm}} = -\mathbf{J}^\top \mathbf{f} \text{ with } \vartheta \geq 0 \quad (61)$$

The initial value of μ_0 is set as shown in Eq. (62).

$$\vartheta_0 = \iota \max \{ (\mathbf{J}^\top \mathbf{J})_{ii} \} \quad (62)$$

Typically, set $\iota \sim [10^{-8}, 1]$ as needed.

The damping factor update strategy is determined by the ratio factor, as shown in Eq. (63).

$$\rho = \frac{F(\mathbf{x}) - F(\mathbf{x} + \Delta \mathbf{x}_{\text{lm}})}{L(0) - L(\Delta \mathbf{x}_{\text{lm}})} \quad (63)$$

$F(\mathbf{x})$ represents the value of the objective function at the current state $\mathbf{x}$. $F(\mathbf{x} + \Delta \mathbf{x}_{\text{lm}})$ is the objective function value after applying the update step $\Delta \mathbf{x}_{\text{lm}}$ obtained through the

Levenberg-Marquardt algorithm. $L(0)$ represents the predicted value of the objective function without applying the step, and $L(\Delta \mathbf{x}_{\text{lm}})$ is the predicted value after the update based on the quadratic approximation. Where,

$$\begin{aligned} L(0) - L(\Delta \mathbf{x}_{\text{lm}}) &= -\Delta \mathbf{x}_{\text{lm}}^\top \mathbf{J}^\top \mathbf{f} - \frac{1}{2} \Delta \mathbf{x}_{\text{lm}}^\top \mathbf{J}^\top \mathbf{J} \Delta \mathbf{x}_{\text{lm}} \stackrel{\text{b-J}^\top \mathbf{f}}{=} \\ &= -\frac{1}{2} \Delta \mathbf{x}_{\text{lm}}^\top (-2\mathbf{b} + (\mathbf{J}^\top \mathbf{J} + \vartheta \mathbf{I} - \vartheta \mathbf{I}) \Delta \mathbf{x}_{\text{lm}}) \\ &= \frac{1}{2} \Delta \mathbf{x}_{\text{lm}}^\top (\vartheta \Delta \mathbf{x}_{\text{lm}} + \mathbf{b}) \end{aligned} \quad (64)$$

In the proposed algorithm, we adopt the damping factor update strategy introduced by Marquardt in 1963, as shown in Eq. (65).

$$\begin{aligned} & \text{if } \rho < 0.25, \vartheta := \vartheta * 2 \\ & \text{elseif } \rho > 0.75, \vartheta := \vartheta / 3 \end{aligned} \quad (65)$$

If $\rho < 0$, the damping factor should be increased to reduce the step size. If the value $\rho > 0$ and relatively large, the damping factor should be decreased, allowing the LM algorithm to approach Gauss-Newton for faster convergence. If ρ is a small positive number, the damping factor should be increased to reduce the iteration step size.

It is essential to clarify that Σ represents the covariance and serves two purposes. Firstly, it normalizes the error factors from different types of sensor data, allowing them to be combined in a dimensionless manner. Secondly, it represents the expression of error uncertainty, providing a weighted representation of different types of errors.

In the proposed algorithm, the error variance is set to 1 pixel for visual data [20], [59], 10 *cm* for GNSS data [60], and 5 *mm* for LiDAR data [61], [62].

Regarding the clock error δ_r of the GNSS receiver, because the two moments involved in the optimization process are very close, we consider the clock errors at the two moments to be equal. This way, they cancel out in Eq. (60).

VI. EXPERIMENTAL RESULTS

We conduct simulations and real-world experiments to verify the performance of our proposed fusion algorithm. In the simulations, we theoretically validate the effectiveness of the four optimization factors (GNSS data, LiDAR data, visual data, and inertial data) in improving position and attitude estimation accuracy. Then, using real-world datasets, we compare the accuracy, consistency, latency, and worst-case performance of the proposed algorithm's position and attitude estimation with several advanced open-source algorithms. The experiments in this paper are conducted on a laptop configured with a 7-1165G7 processor at 2.800GHz and 32GB of RAM.

A. Simulation

1) *Setup*: The simulation environment is a cubic space of $200 \text{ m} \times 200 \text{ m} \times 200 \text{ m}$, containing randomly generated 3D landmarks. These landmarks are projected onto a virtual camera operating at 20 Hz, with a horizontal field of view (FOV) of 75 degrees and a vertical FOV of 55 degrees, generating approximately 200 visible feature points per frame.

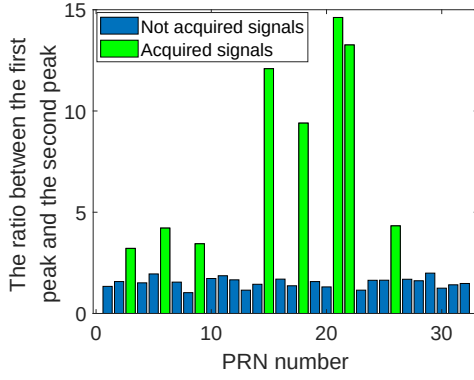


Fig. 10. The sequence numbers of the captured satellites

Additional white noise with a standard deviation of 0.5 pixels is added to all feature points.

A virtual IMU operating at 200 Hz is rigidly connected to the camera, moving along a predefined 3D path. The accelerometer and gyroscope have white noise standard deviations of 0.005 m/s^2 and 0.005 rad/s , respectively, and random walk biases with standard deviations of $2.5 \times 10^{-4} \text{ m/s}^2$ and $2.5 \times 10^5 \text{ rad/s}$.

The LiDAR has a horizontal FOV of 360 degrees and a vertical FOV ranging from -15 degrees to 15 degrees, with a scanning frequency of 20 Hz. Each point has Gaussian white noise with a standard deviation of $\sigma = 0.02 \text{ m}$.

Simultaneously, a virtual GNSS receiver operating at 10 Hz uses historical broadcast ephemeris data, which includes orbit information and time parameters of GNSS satellites. Ephemeris data provides precise satellite positions and velocities at specific times. Based on the satellite orbit information, we calculate the spatial positions of each satellite at a particular time. The trajectory and position of the virtual receiver in the simulation are known, allowing for the calculation of distances between the receiver and each satellite using three-dimensional coordinates. Pseudorange represents distance and can be obtained by multiplying the time taken for satellite signals from transmission to reception by the speed of light. By adding errors in signal propagation (such as satellite clock errors, atmospheric delays, etc.) to the distance between the receiver and satellite positions, pseudorange measurements can be simulated. We add white noise to the generated pseudorange. In the simulation, the standard deviation of pseudorange noise is set to 1 m .

Here, we use the GPS satellite data from reference [63], and the acquisition results and the geometric distribution of these satellite signals are shown in Fig. 10 and Fig. 11, respectively.

2) *Analysis of position estimation accuracy:* With all other simulation conditions hold constant, we evaluate the performance of the proposed optimization factor models by analyzing the results obtained when different optimization factors are involved in the computation. As shown in Fig. 12, it demonstrates the positioning estimation results for four scenarios: including all factors, excluding GNSS factor, excluding both GNSS and visual factors, and excluding GNSS, visual and LiDAR factors.

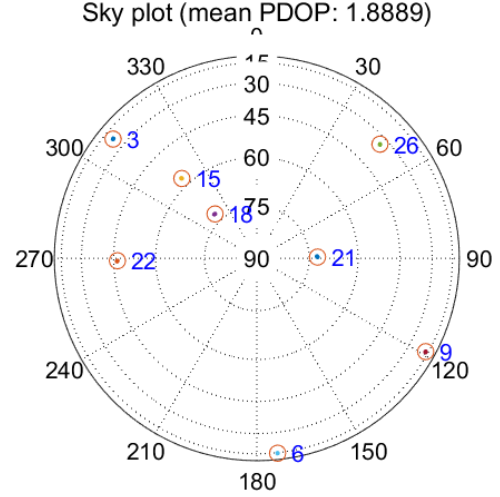


Fig. 11. The Position Dilution of Precision (PDOP) factor of the captured satellites

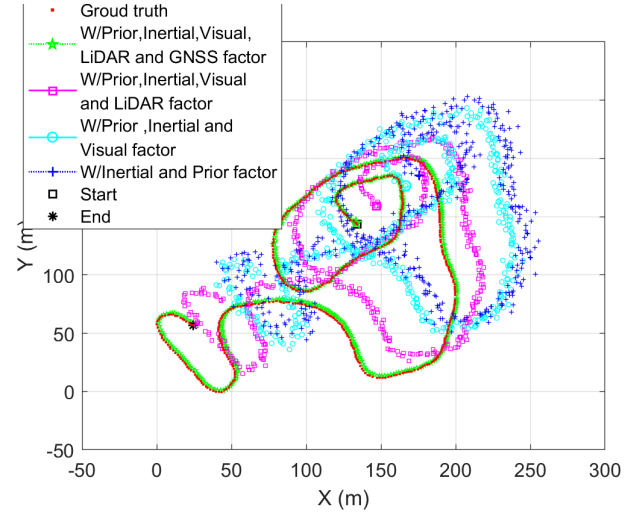


Fig. 12. Illustration of the discrepancy between the estimated trajectories and the reference trajectory under different optimization factors in the optimization

Fig. 12 reveals that the estimated trajectory with all optimization factors (green line) is closest to the reference trajectory (red line). The trajectory with only the inertial factor and prior information (blue line) is farthest from the reference trajectory, indicating the most significant estimation error. Adding each optimization factor brings the estimated trajectory closer to the reference trajectory.

We divide the estimated trajectory into eight segments, conduct a statistical analysis of the estimation error for each segment, and calculate the Root Mean Square Error (RMSE) for each segment under four different conditions, as shown in Fig. 13(a).

As shown in Fig. 13(a), if only inertial factor and prior information are involved in the optimization, the RMSE of the estimation error remains around 50 (m). With each additional optimization factor, the RMSE decreases. When GNSS data, LiDAR data, visual data, and inertial data are all involved in the optimization, the RMSE of the estimation error reduces

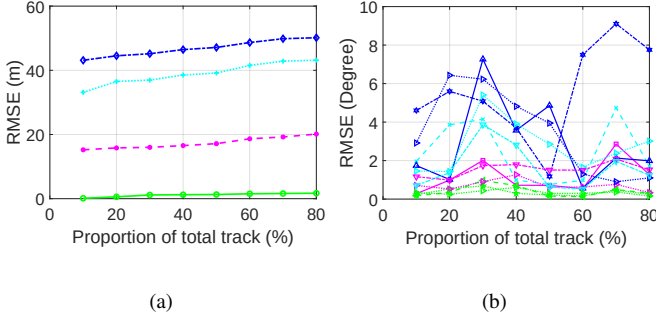


Fig. 13. (a) Segmented statistical RMSE values of the position estimation error under four different conditions; (b) Segmented statistical RMSE values of the attitude estimation error under four different conditions

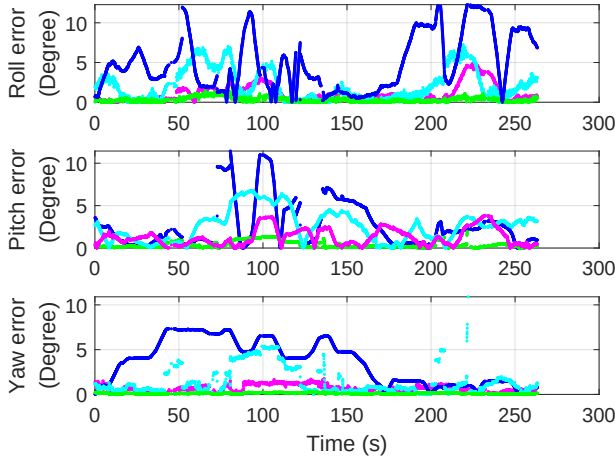


Fig. 14. Attitude estimation errors under four different conditions

to around 1 (m). This theoretically proves the feasibility and effectiveness of the proposed optimization factors in improving position estimation accuracy.

3) *Analysis of orientation estimation accuracy:* Similar to the analysis method for position estimation error, we statistically analyze the attitude estimation error under four different conditions: including all optimization factors, excluding GNSS optimization factor, excluding both GNSS and LiDAR factor, and excluding GNSS, LiDAR, and visual factor, as shown in Fig. 14. Here, the color representation is the same as in Fig. 12. The segmented error statistical analysis RMSE values are shown in Fig. 13(b).

It can be observed that with the addition of optimization factors, the RMSE values of the attitude estimation errors decrease. When all optimization factors are included (green line), the angle estimation error is minimal, around 0.2 degree. In contrast, the angle estimation error is most significant for the case involving only inertial and prior factor (blue line), reaching approximately 10 degrees. The inclusion of LiDAR and visual factor tends to improve the estimation errors. Thus, it can be concluded that the proposed optimization factors of GNSS data, visual data, LiDAR data, and inertial data are theoretically feasible and effective in improving the accuracy of attitude estimation.

B. Comparison with existing algorithms

In this section, we use three publicly available datasets to compare the position estimation performance and attitude estimation performance of the proposed algorithm with several advanced open-source algorithms from four aspects. The four aspects are accuracy, real-time performance, stability, and the worst-case performance. In this paper, the accuracy is evaluated by calculating the RMSE of the estimation error, stability is assessed by the size of the 95% confidence interval, the worst-case performance is measured by the maximum estimation error, and the real-time performance is evaluated by the time taken from signal input to the estimation results.

1) *M2DGR dataset:* We utilize the M2DGR dataset [64], a novel large-scale dataset obtained by equipping a ground robot with a comprehensive sensor suite. This suite includes fish-eye cameras, infrared cameras, visual-inertial sensors, Inertial Measurement Units (IMU), LiDAR sensor, a consumer-grade GNSS receiver, and a GNSS-IMU navigation system with real-time kinematic (RTK) signals. These sensors collectively contribute to the data collection process on the ground robot. The dataset contains multiple sequences covering a variety of scenarios, including both indoor and outdoor environments. We use the M2DGR_street_02 dataset to compare the pose estimation performance of GVINS [45], VINS-fusion [31], VINS-Mono [16], ORB-SLAM2 [19], LIO-SAM [6], LVI-SAM [22], W/O GNSS factor, and the proposed algorithm. W/O GNSS factor refers to the fusion algorithm proposed in this paper, but without the GNSS factor.

The estimated trajectories using the proposed algorithm and several existing algorithms, along with the reference trajectory, are shown in Fig. 15. It should be noted that the position error of ORB-SLAM2 is too large (150.427 m), rendering it meaningless for comparison. Therefore, we do not include its trajectory in the graph. The RMSE values from the statistical analysis of the estimation errors are presented in Table. I, with the first row showing the RMSE values for position estimation error, and the second row showing the RMSE values for attitude estimation error.

From the first row data of Table. I, it can be seen that the RMSE of the position estimation error for the proposed algorithm (1.284 m) is the lowest, and compared to the RMSE of the W/O GNSS factor algorithm (3.057 m), it shows a 58.0% improvement. This indicates that the GNSS optimization factor proposed in this paper is highly effective in improving the position estimation accuracy of the fusion system. Furthermore, the fusion algorithms primarily based on visual data, such as ORB-SLAM2 (150.427 m), VINS-Mono (25.264 m), and VINS-Fusion (35.297 m), have significantly higher RMSE values compared to the fusion algorithms primarily based on LiDAR data, such as LVI-SAM (3.592 m) and LIO-SAM (4.063 m). This suggests that the LiDAR data in the M2DGR_street_02 dataset is more reliable than the visual data, indicating that this is a vision-degraded environment. This proves that the performance of fusion algorithms based on visual and LiDAR data is affected by challenging environments. This further proves the effectiveness of the proposed optimization factors for LiDAR and visual data in improving

TABLE I
THE QUANTITATIVE COMPARISON OF THE PERFORMANCE OF THE PROPOSED ALGORITHM AND SEVERAL EXISTING FUSION ALGORITHMS IN POSE ESTIMATION ON THE M2DGR_STREET_02 DATASET

Metric\Method	GVINS	VINS-fusion	ORB-SLAM2	VINS-Mono	LIO-SAM	LVI-SAM	Proposed	W/O GNSS factor
Accuracy	6.781 18.356	35.297 68.237	150.427 86.247	25.264 35.367	4.063 2.947	3.592 2.582	1.284 0.354	3.057 5.967
Confidence interval	5.876 ± 1.626 16.276 ± 3.540	33.157 ± 3.260 65.128 ± 4.555	145.427 ± 7.952 83.128 ± 4.060	23.127 ± 3.939 33.478 ± 3.445	3.064 ± 1.208 2.356 ± 0.996	3.091 ± 0.767 2.034 ± 0.824	1.076 ± 0.261 0.338 ± 0.058	2.874 ± 0.475 4.996 ± 1.586
Latency	152.2	172.5	98.6	124.8	205.7	276.1	198.5	147.4
Worst-case	8.526 20.236	39.924 71.461	158.237 89.528	28.125 39.972	6.267 3.757	5.926 3.476	1.526 0.561	4.342 6.924

Each small cell contains two sets of data. The top shows the position estimation error in meter (m), and the bottom shows the attitude estimation error in degree (Degree). The third row represents the total processing time in millisecond (ms).

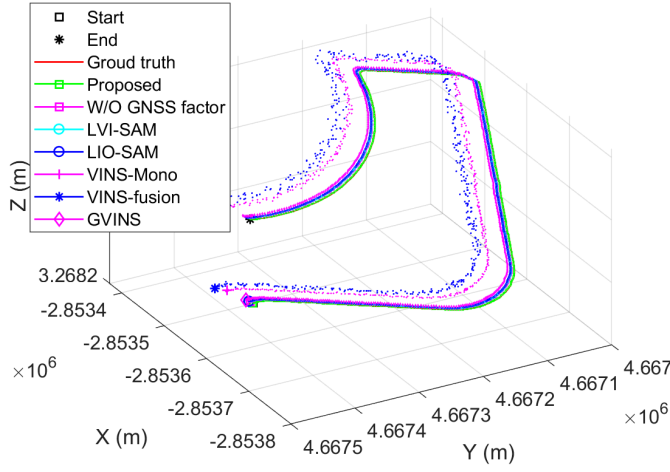


Fig. 15. The estimated trajectories of the proposed algorithm and several existing algorithms on the M2DGR_street_02 dataset, along with the reference trajectory

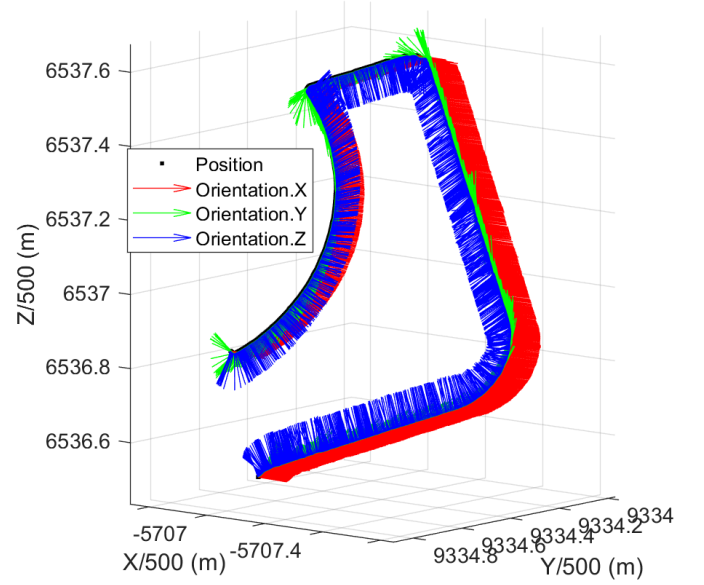


Fig. 16. The Six Degree of Freedom (6-DoF) diagram of the proposed algorithm on the M2DGR_street_02 dataset

position estimation accuracy.

From the RMSE values of attitude estimation error in the second row, the proposed algorithm (0.354 Degree) has the lowest value among the various algorithms, indicating that the proposed algorithm achieves the highest accuracy in attitude estimation. Here, we represent the position and attitude of the vehicle using the Six Degree of Freedom (6-DoF) model, as shown in Fig. 16. The 6-DoF algorithm flow is presented in Algorithm. 1. The variations of the red, green, and blue lines represent the changes in the vehicle's pitch angle, roll angle, and yaw angle, respectively.

The second row in Table. I shows the 95% confidence intervals for the estimates. We represent the confidence intervals using the error bar charts, with the error bar charts for position estimation and attitude estimation shown in Fig. 17(a) and Fig. 17(b), respectively. It should be noted that the pose estimation errors of algorithms ORB-SLAM2, VINS-fusion and VINS-Mono are too large. To facilitate the analysis of the consistency between the proposed algorithm and several other algorithms, we scale down the position estimation and attitude estimation results of ORB-SLAM2, VINS-fusion and VINS-Mono by a certain proportion.

From Fig. 17(a) and Fig. 17(b), it can be seen that the 95% confidence intervals for position estimation and attitude

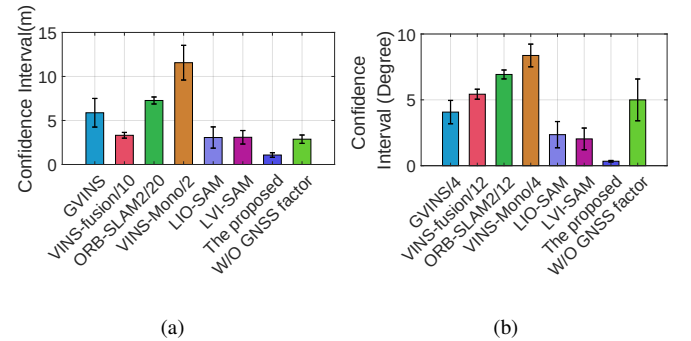


Fig. 17. (a) The 95% confidence intervals of the position estimation for the proposed algorithm and several existing algorithms on the M2DGR_street_02 dataset; (b) The 95% confidence intervals of the attitude estimation for the proposed algorithm and several existing algorithms on the M2DGR_street_02 dataset.

estimation of the proposed algorithm are the smallest, indicating that the proposed algorithm has good consistency, meaning that its pose estimation data has high reliability. This suggests that the pose estimation from the proposed algorithm

Algorithm 1 Simplified 6DOF animation

Input: P : position vector, R : rotation matrix, optional arguments: SamplePlotFreq, View, CreateAVI, [numPlotSamples dummy] = size(P)

Output: 3D animation and optionally an AVI file

- 1: Parse inputs and optional arguments.
 - 2: Reduce P and R based on SamplePlotFreq.
 - 3: Create 3D figure using 'figure()', configure axis with 'axis equal' and 'view()'.
 - 4: Set up lighting with 'lighting phong'.
 - 5: **if** CreateAVI **then**
 - 6: Initialize AVI file using 'VideoWriter()'.
 - 7: **end if**
 - 8: Plot initial position using 'plot3()' and create axis arrows with 'quiver3()'.
 - 9: **for all** samples $i = 1, 2, \dots, \text{numPlotSamples}$ **do**
 - 10: Update position and orientation using 'set()' for plot and quiver handles.
 - 11: **if** CreateAVI **then**
 - 12: Capture frame using 'getframe()' and add it to AVI.
 - 13: **end if**
 - 14: **end for**
 - 15: **if** CreateAVI **then**
 - 16: Close AVI file using 'close()'.
 - 17: **end if**
 - 18: **return** the 3D animation handle.
-

is more robust. In autonomous driving and navigation, the high consistency of the pose estimation information from the proposed algorithm can enhance the system's decision-making capabilities, ensuring safety and reliability.

The third row in Table. I shows the total time consumption, which refers to the total time taken from signal input to pose output. It can be seen that the total time consumption of the proposed algorithm (198.5 ms) is not the smallest among the algorithms. The shortest time is from the ORB-SLAM2 algorithm (98.6 ms), as it only processes visual and inertial data, but its pose estimation error is too large to be meaningful for comparison. Compared to LVI-SAM and LIO-SAM, the proposed algorithm achieves higher estimation accuracy while consuming less time. Compared to the W/O GNSS factor algorithm, the proposed algorithm takes about 50 ms more, which is due to the additional processing of the GNSS data. However, the position estimation accuracy improves by 58.0%, and the attitude estimation accuracy improves by 94.1%. Given the accuracy improvements, the increase in total time consumption is worthwhile. Similarly, compared to the GVINS algorithm, the total time consumption increases by more than 40 ms, but given the improvement in accuracy, this increase in time is also worthwhile.

The fourth row in Table. I lists the maximum estimation errors of the several fusion algorithms, representing the worst-case performance. It can be seen that the maximum values of the position estimation error and attitude estimation error for the proposed algorithm are the smallest compared to the other algorithms. This means that the proposed algorithm is more robust, with the strongest resistance to interference,

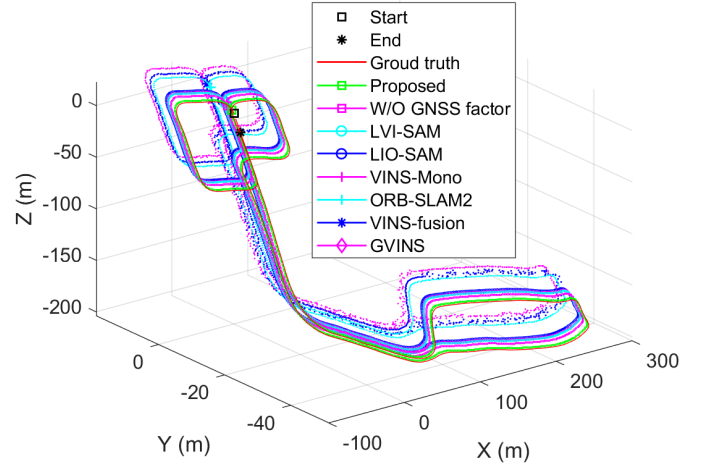


Fig. 18. The estimated trajectories of the proposed algorithm and several existing algorithms on the MVSEC_day1 dataset, along with the reference trajectory

reducing the risk of significant deviations at critical moments. Therefore, the proposed algorithm can adapt to more complex environments and has greater applicability.

2) *MVSEC dataset*: The Multi Vehicle Stereo Event Camera dataset [65] is a collection of data designed for the development of novel 3D perception algorithms for event based cameras. Stereo event data is collected from car, motorbike, hexacopter and handheld data, and fused with LiDAR, IMU, motion capture and GPS to provide ground truth pose and depth images. In addition, they provide images from a standard stereo frame based camera pair for comparison with traditional techniques. The dataset was collected from four different vehicles, in both indoor and outdoor environments, in day and night settings. All hexacopter sequences have motion capture ground truth from an indoor Vicon area and outdoor Qualisys area, while the other sequences have ground truth generated by fusing LiDAR information with IMU and GPS. The camera clocks are synchronized by a hardware trigger generated by the left camera and send to the right camera. In addition to events, the cameras also each generate IMU and frame based image measurements. In addition, a Velodyne LiDAR and stereo frame based camera with IMU (VI Sensor) is mounted with the DAVIS cameras. When available, the ground truth pose is also captured using an indoor (Vicon, left) or outdoor (Qualisys, right) motion capture system.

The estimated trajectories of the proposed algorithm and several existing algorithms are shown in Fig. 18, while the 6DOF trajectory of the proposed algorithm is illustrated in Fig. 19. The statistical analysis of the position estimation errors and attitude estimation errors of the proposed algorithm and several existing algorithms are shown in Table. II. From the RMSE values in the first row of Table. II, it can be seen that the proposed algorithm outperforms several other algorithms in both position estimation accuracy and attitude estimation accuracy. The 95% confidence intervals for the position estimation and the attitude estimation are shown in Fig. 20(a) and Fig. 20(b) respectively. From Fig. 20(a) and Fig. 20(b), it can be observed that the confidence intervals

TABLE II

THE QUANTITATIVE COMPARISON OF THE PERFORMANCE OF THE PROPOSED ALGORITHM AND SEVERAL EXISTING FUSION ALGORITHMS IN POSE ESTIMATION RESULTS ON THE MVSEC_DAY1 DATASET

Metric\Method	GVINS	VINS-fusion	ORB-SLAM2	VINS-Mono	LIO-SAM	LVI-SAM	Proposed	W/O GNSS factor
Accuracy	3.497 2.569	10.479 6.486	8.573 2.257	12.521 3.389	4.127 10.212	3.256 5.954	0.984 0.945	2.478 2.548
Confidence interval	3.076 $\pm$ 0.926 2.462 $\pm$ 0.989	9.074 $\pm$ 1.760 5.405 $\pm$ 1.123	7.128 $\pm$ 1.752 1.749 $\pm$ 0.751	10.138 $\pm$ 2.939 3.154 $\pm$ 0.550	3.468 $\pm$ 0.908 9.127 $\pm$ 2.112	2.728 $\pm$ 0.867 5.079 $\pm$ 1.108	0.764 $\pm$ 0.361 0.840 $\pm$ 0.141	2.129 $\pm$ 0.475 2.209 $\pm$ 0.481
Latency	212.3	194.5	143.7	163.8	286.4	356.7	245.8	198.7
Worst-case	5.287 3.625	13.569 7.157	9.327 3.351	16.854 4.461	5.246 12.324	4.248 6.843	1.245 1.297	2.967 3.126

Each small cell contains two sets of data. The top shows the position estimation error in meter (m), and the bottom shows the attitude estimation error in degree (Degree). The third row represents the total processing time in millisecond (ms).

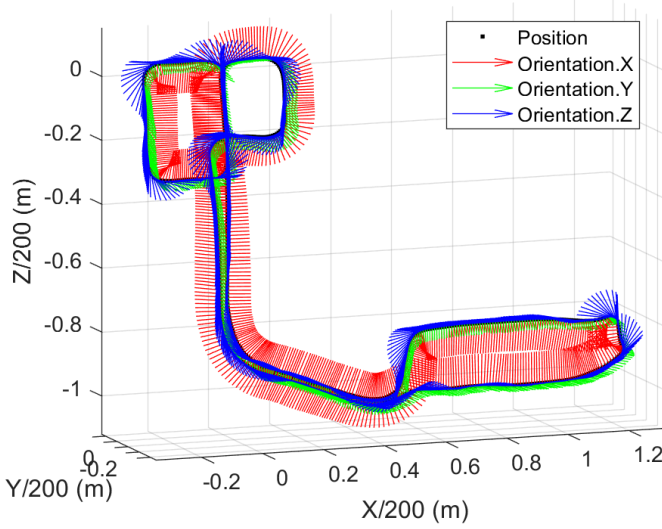


Fig. 19. The Six Degree of Freedom (6-DoF) diagram of the proposed algorithm on the MVSEC_day1 dataset

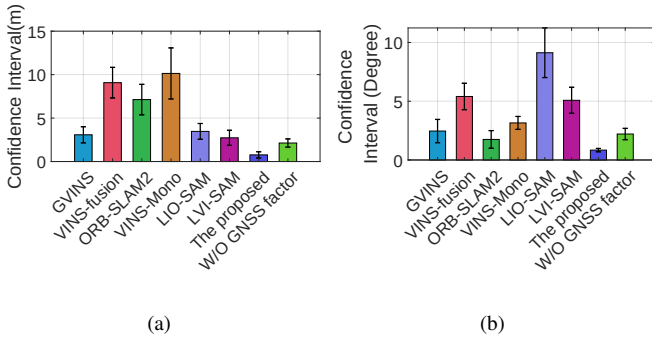


Fig. 20. (a) The 95% confidence intervals of the position estimation for the proposed algorithm and several existing algorithms on the MVSEC_day1 dataset; (b) The 95% confidence intervals of the attitude estimation for the proposed algorithm and several existing algorithms on the MVSEC_day1 dataset.

of the proposed algorithm are smaller than those of the other algorithms. Additionally, the data in the fourth row of Table. II indicates that the maximum errors of the proposed algorithm are also smaller than that of the other algorithms.

From the data in the third row of Table. II, it can be concluded that although the total time of the proposed al-

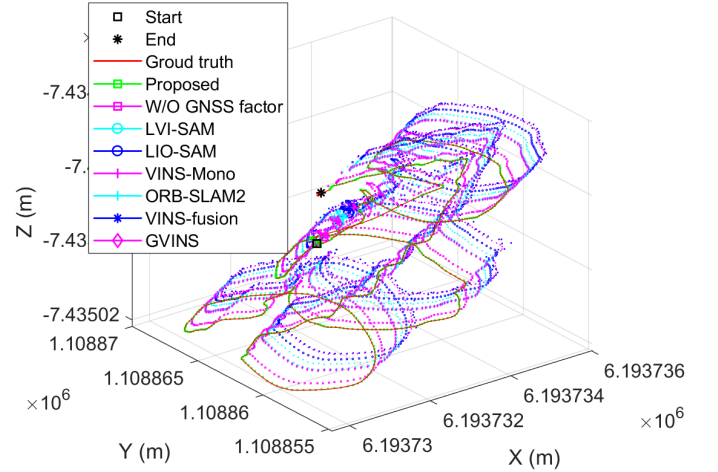


Fig. 21. The estimated trajectories of the proposed algorithm and several existing algorithms on the NTU VIRAL_eee 03 dataset, along with the reference trajectory

gorithm (245.8 ms) increases, for example, by 102.1 (ms) compared to ORB-SLAM2 (143.7 ms), the position estimation accuracy improves by 88.5%, and the attitude estimation accuracy increases by 58.1%. Compared to the W/O GNSS factor algorithm, the total time increases by 47.1 ms, while the position estimation accuracy improves by 60.3%, and the attitude estimation accuracy increases by 62.9%. Therefore, the increase of about 100 ms in total time is worthwhile considering the improvement in estimation accuracy.

Therefore, the proposed algorithm outperforms several existing algorithms in terms of estimation accuracy, consistency, and reliability on the MVSEC_day1 dataset.

3) *NTU VIRAL dataset*: NTU VIRAL is a dataset collected from the Unmanned Aerial Vehicle (UAV) platform [66], which is equipped with an extensive set of sensors: Two 3D LiDARs, Two time-synchronized cameras, Multiple Inertial Measurement Units (IMUs), Four Ultra-wideband (UWB) nodes on UAV, ranging to three anchor nodes. The flight tests are conducted in a variety of both indoor and outdoor conditions.

The estimated trajectories of the proposed algorithm and several existing algorithms, along with the reference trajectory, are shown in Fig. 21. The 6DOF trajectory of the proposed algorithm is shown in Fig. 22. The statistical analysis of the

TABLE III
THE QUANTITATIVE COMPARISON OF THE PERFORMANCE OF THE PROPOSED ALGORITHM AND SEVERAL EXISTING FUSION ALGORITHMS IN POSE ESTIMATION RESULTS ON THE NTU VIRAL_eee03 DATASET

Metric\Method	GVINS	VINS-fusion	ORB-SLAM2	VINS-Mono	LIO-SAM	LVI-SAM	Proposed	W/O GNSS factor
Accuracy	1.451 3.276	1.877 6.247	1.075 2.258	1.745 5.582	1.353 3.257	1.172 2.472	0.284 0.253	0.857 1.224
Confidence interval	1.124 ± 0.427 2.865 ± 0.724	1.687 ± 0.353 5.828 ± 1.025	0.857 ± 0.362 2.048 ± 0.476	1.627 ± 0.389 4.954 ± 0.966	1.198 ± 0.305 2.927 ± 0.541	0.925 ± 0.335 2.121 ± 0.657	0.226 ± 0.073 0.238 ± 0.043	0.674 ± 0.351 1.096 ± 0.486
Latency	263.7	193.4	156.9	231.4	297.4	304.2	217.8	185.1
Worst-case	1.725 4.569	2.121 7.258	1.347 2.687	2.367 7.569	1.778 4.025	1.551 3.148	0.351 0.327	1.197 1.914

Each small cell contains two sets of data. The top shows the position estimation error in meter (m), and the bottom shows the attitude estimation error in degree (Degree). The third row represents the total processing time in millisecond (ms).

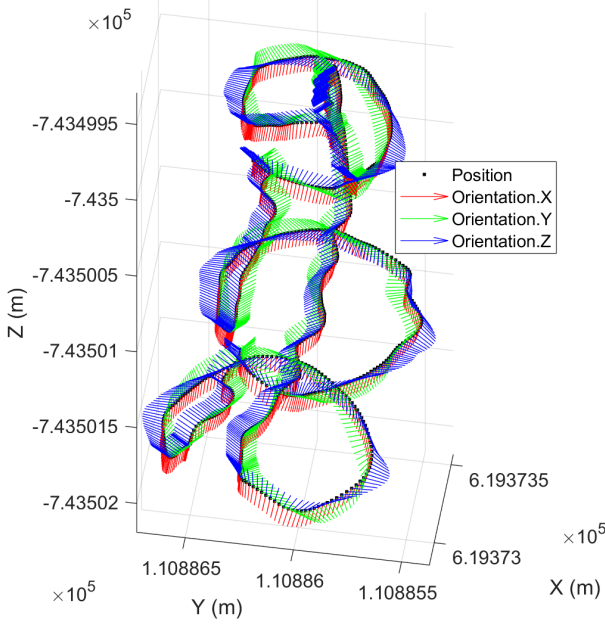


Fig. 22. The Six Degree of Freedom (6-DoF) diagram of the proposed algorithm on the NTU VIRAL_eee03 dataset

pose estimation errors for the proposed algorithm and several existing algorithms are shown in Table. III. From the first row data of Table. 22, it can be seen that the proposed algorithm has the highest position estimation accuracy (0.284 m) among all the algorithms, followed by the W/O GNSS factor algorithm (0.857 m). ORB-SLAM2 comes next, with an RMSE of 1.075 m. However, the W/O GNSS factor algorithm's attitude estimation accuracy is 1.224 (Degree), while ORB-SLAM2's is 2.258 (Degree), and the proposed algorithm achieves 0.253 (Degree). This represents an improvement of 79.3% compared to the W/O GNSS factor algorithm and 88.8% compared to ORB-SLAM2 algorithm. Therefore, the proposed algorithm demonstrates superior pose estimation accuracy compared to the existing algorithms. The 95% confidence intervals for the position estimation accuracy and attitude estimation accuracy of these algorithms are shown in Fig. 23(a) and Fig. 23(b) respectively. It is evident that, compared to other algorithms, the proposed algorithm has the best consistency.

The proposed algorithm's maximum position estimation error and maximum attitude estimation error are 0.351 (m)

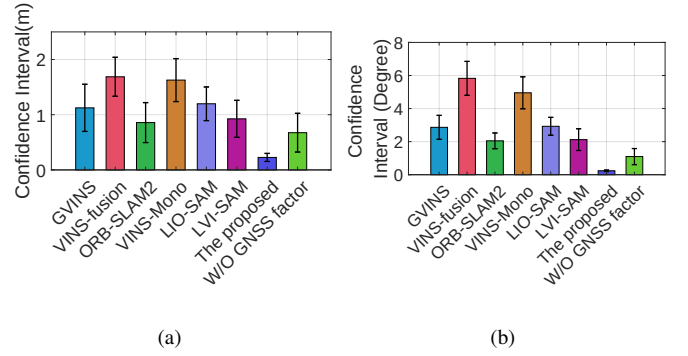


Fig. 23. (a) The 95% confidence intervals of the position estimation for the proposed algorithm and several existing algorithms on the NTU VIRAL_eee03 dataset; (b) The 95% confidence intervals of the attitude estimation for the proposed algorithm and several existing algorithms on the NTU VIRAL_eee03 dataset.

and 0.327 (Degree), respectively, which are the smallest among all the algorithms. This indicates that, compared to the other algorithms, the proposed algorithm has the strongest ability to mitigate risks.

In terms of computational efficiency, the proposed algorithm has a total time of 217.8 (ms), which has an increase of 32.7 (ms) compared to the total time of the W/O GNSS factor algorithm (185.1 ms) and an increase of 60.9 (ms) compared to the ORB-SLAM2 algorithm (156.9 ms). Considering the overall improvement in pose estimation accuracy and the reliability of the estimates, the increase in total time is justified.

4) *Robustness Analysis:* We compare and analyze the robustness and applicability of the proposed algorithm and existing algorithms in complex environments from two aspects. One aspect is whether the algorithm can successfully perform positioning in complex environments, and the second aspect is the improvement in pose estimation accuracy. First, on the M2DGR dataset, we use position estimation accuracy as an example. From the first row of Table. I, we can see that ORB-SLAM2, VINS-fusion, and VINS-Mono are vision-based fusion frameworks. Among them, the one with the smallest RMSE is 25.264 (m), which does not meet the general positioning accuracy requirements for autonomous systems, indicating positioning failure. The proposed algorithm, however, achieves a positioning accuracy of 1.284 (m), meeting the requirement. On the other hand, LVI-SAM and LIO-

SAM are LiDAR based fusion algorithms, with the smallest positioning accuracy being 3.592 (m), while the proposed algorithm achieves an accuracy of 1.284 (m), improving by 64.3%. Similarly, compared to GVINS (6.781 m), the proposed algorithm improves positioning accuracy by 81.1%, showing stronger adaptability in complex environments. We can also analyze the improvement in attitude estimation accuracy and the improvement in pose estimation accuracy on other datasets. In summary, the proposed algorithm demonstrates stronger robustness and adaptability in complex environments compared to existing algorithms.

VII. CONCLUSIONS

This paper proposes a tightly coupled algorithm based on GNSS data, LiDAR data, visual data, and inertial data. We use three actual datasets to compare the pose estimation performance of the proposed algorithm with several existing algorithms across four aspects. The results demonstrate that the proposed algorithm outperforms the existing algorithms in practical applications. However, the total computation time of the proposed algorithm is somewhat increased compared to some existing algorithms. In the future, we will continue to research optimization algorithms, layered processing, parallel processing, and adaptive algorithms to minimize the computation time of the fusion algorithm, thereby further enhancing the capabilities of the proposed algorithm in practical applications.

REFERENCES

- [1] Y. Gao, J. Yuan, J. Jiang, Q. Sun, and X. Zhang, "Vido: A robust and consistent monocular visual-inertial-depth odometry," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 3, pp. 2976–2992, 2022.
- [2] D. Fu, S. Han, W. Li, and H. Lin, "The pose estimation of the aircraft on the airport surface based on the contour features," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 2, pp. 817–826, 2022.
- [3] S. Chen, F. Hu, Z. Chen, and H. Wu, "Correction method for uav pose estimation with dynamic compensation and noise reduction using multi-sensor fusion," *IEEE Transactions on Consumer Electronics*, 2023.
- [4] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized Lidar odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2018, pp. 4758–4765.
- [5] 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.
- [6] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled Lidar Inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2020, pp. 5135–5142.
- [7] Z. Wang, J. Zhang, S. Chen, C. Yuan, J. Zhang, and J. Zhang, "Robust high accuracy Visual-Inertial-Laser SLAM system," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2019, pp. 6636–6641.
- [8] M. Camurri, M. Ramezani, S. Nobili, and M. Fallon, "Pronto: A multi-sensor state estimator for legged robots in real-world scenarios," *Frontiers in Robotics and AI*, vol. 7, p. 68, 2020.
- [9] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3d Lidar Inertial odometry and mapping," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2019, pp. 3144–3150.
- [10] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu, "LINS: A Lidar-Inertial state estimator for robust and efficient navigation," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2020, pp. 8899–8906.
- [11] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer, "Super odometry: IMU-centric LiDAR-Visual-Inertial estimator for challenging environments," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2021, pp. 8729–8736.
- [12] Z. Wen, X. Guo, Z. Zheng, S. Liu, T. Zhang, X. Xia, and Y. Li, "TI-gilns: A trajectory-layer-enhanced gnss/ins/lidar integrated approach towards reliable positioning and accurate accuracy quantification," *IEEE Internet of Things Journal*, 2024/11, Early access.
- [13] S. Li, X. Li, S. Chen, Y. Zhou, and S. Wang, "Continuous-time lidar/imu/radar odometry for accurate and smooth state estimation via tightly coupled integration," *IEEE Internet of Things Journal*, 2024/8, Early access.
- [14] P. Geneva, K. Eickenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A research platform for Visual-Inertial estimation," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2020, pp. 4666–4672.
- [15] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2020, pp. 1689–1696.
- [16] T. Qin, P. Li, and S. Shen, "VINS-mono: A robust and versatile monocular Visual-Inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, 2018.
- [17] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular Visual-Inertial odometry algorithms for flying robots," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2018, pp. 2502–2509.
- [18] 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.
- [19] R. Mur-Artal and J. D. Tardós, "ORM-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
- [20] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.
- [21] S. Leutenegger, "OKVIS 2.0 for the FPV drone racing VIO competition 2020," 2020.
- [22] 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.
- [23] Y. Yang, P. Geneva, X. Zuo, K. Eickenhoff, Y. Liu, and G. Huang, "Tightly-coupled aided Inertial navigation with point and plane features," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2019, pp. 6094–6100.
- [24] X. Zuo, Y. Yang, P. Geneva, J. Lv, Y. Liu, G. Huang, and M. Pollefeys, "LIC-Fusion 2.0: LiDAR-Inertial-Camera odometry with sliding-window plane-feature tracking," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2020, pp. 5112–5119.
- [25] W. Shao, S. Vijayarangan, C. Li, and G. Kantor, "Stereo Visual Inertial LiDAR simultaneous localization and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2019, pp. 370–377.
- [26] J. Graeter, A. Wilczynski, and M. Lauer, "LIMO: Lidar-monocular visual odometry," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2018, pp. 7872–7879.
- [27] Y. Yu, W. Gao, C. Liu, S. Shen, and M. Liu, "A GPS-aided omnidirectional visual-inertial state estimator in ubiquitous environments," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2019, pp. 7750–7755.
- [28] W. Lee, K. Eickenhoff, P. Geneva, and G. Huang, "Intermittent GPS-aided VIO: Online initialization and calibration," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2020, pp. 5724–5731.
- [29] R. Mascaro, L. Teixeira, T. Hinzmann, R. Siegwart, and M. Chli, "GOMSF: Graph-optimization based multi-sensor fusion for robust UAV pose estimation," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2018, pp. 1421–1428.
- [30] W. Yan, L. Bastos, J. A. Gonçalves, A. Magalhães, and T. Xu, "Image-aided platform orientation determination with a GNSS/low-cost IMU system using robust-adaptive Kalman filter," *GPS Solutions*, vol. 22, pp. 1–12, 2018.
- [31] T. Qin, S. Cao, J. Pan, and S. Shen, "A general optimization-based framework for global pose estimation with multiple sensors," *arXiv preprint arXiv:1901.03642*, 2019.
- [32] G. Cioffi and D. Scaramuzza, "Tightly-coupled fusion of global positional measurements in optimization-based visual-inertial odometry," in *Proc. IEEE/RSJ Int. Conf. Intell.Robots Sys.*, 2020, pp. 5089–5095.
- [33] R. Sun, J. Wang, Q. Cheng, Y. Mao, and W. Y. Ochieng, "A new IMU-aided multiple GNSS fault detection and exclusion algorithm for integrated navigation in urban environments," *GPS Solutions*, vol. 25, pp. 1–17, 2021.
- [34] X. Hu, D. Olesen, and P. Knudsen, "Toward high-quality magnetic data survey using UAV: development of a magnetic-isolated vision-based positioning system," *GPS Solutions*, vol. 25, pp. 1–17, 2021.
- [35] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV," in *Proc. IEEE Int. Conf.Robot. Autom.*, 2014, pp. 4974–4981.

- [36] E. Moumier, M. Elhabiby, M. Korenberg, and A. Noureldin, "Lidar-based multi-sensor fusion with 3d digital maps for high-precision positioning," *IEEE Internet of Things Journal*, 2024/11, Early access.
- [37] M. Irfan, S. Dalai, P. Trslie, M. C. Santos, J. Riordan, and G. Dooly, "Lgvins: Lidar-gps-visual and inertial system based multi-sensor fusion for smooth and reliable uav state estimation," *IEEE Transactions on Intelligent Vehicles*, 2024/9, Early access.
- [38] F. Zhang, X. Shao, and W. Zhang, "Cooperative fusion localization of a nonstationary target for multiple uavs without gps," *IEEE Systems Journal*, vol. 18, no. 4, pp. 1951–1962, 2024.
- [39] D. H. Won, E. Lee, M. Heo, S. Sung, J. Lee, and Y. J. Lee, "GNSS integration with vision-based navigation for low GNSS visibility conditions," *GPS solutions*, vol. 18, pp. 177–187, 2014.
- [40] D. H. Won, E. Lee, M. Heo, S.-W. Lee, J. Lee, J. Kim, S. Sung, and Y. J. Lee, "Selective integration of GNSS, vision sensor, and INS using weighted DOP under GNSS-challenged environments," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 9, pp. 2288–2298, 2014.
- [41] T. Li, H. Zhang, Z. Gao, X. Niu, and N. El-Sheimy, "Tight fusion of a monocular camera, MEMS-IMU, and single-frequency multi-GNSS RTK for precise navigation in GNSS-challenged environments," *Remote Sensing*, vol. 11, no. 6, p. 610, 2019.
- [42] G. Huang, "Visual-Inertial navigation: A concise review," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2019, pp. 9572–9582.
- [43] W. Wen, T. Pfeifer, X. Bai, and L.-T. Hsu, "It is time for factor graph optimization for GNSS/INS integration: Comparison between FGO and EKF," *arXiv preprint arXiv:2004.10572*, 2020.
- [44] Z. Gong, R. Ying, F. Wen, J. Qian, and P. Liu, "Tightly coupled integration of GNSS and vision SLAM using 10-dof optimization on manifold," *IEEE Sensors Journal*, vol. 19, no. 24, pp. 12 105–12 117, 2019.
- [45] S. Cao, X. Lu, and S. Shen, "GVINS: Tightly coupled GNSS–Visual–Inertial fusion for smooth and consistent state estimation," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2004–2021, 2022.
- [46] C. Xia, X. Li, F. He, S. Li, and Y. Zhou, "Accurate and rapidly-convergent gnss/ins/lidar tightly-coupled integration via invariant ekf based on two-frame group," *IEEE Transactions on Automation Science and Engineering*, 2024/10, Early Access.
- [47] G. Giorgi, *Attitude Determination*. Cham: Springer International Publishing, 2017, pp. 781–809. [Online]. Available: https://doi.org/10.1007/978-3-319-42928-1_27
- [48] M. D. Shuster *et al.*, "A survey of attitude representations," *Navigation*, vol. 8, no. 9, pp. 439–517, 1993.
- [49] T.-C. Dong-Si and A. I. Mourikis, "Consistency analysis for sliding-window visual odometry," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 5202–5209.
- [50] G. Sibley, L. Matthies, and G. Sukhatme, "Sliding window filter with application to planetary landing," *Journal of Field Robotics*, vol. 27, no. 5, pp. 587–608, 2010.
- [51] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.
- [52] 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.
- [53] 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.
- [54] 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.
- [55] P. Ward, "The natural measurements of a gps receiver," in *Proceedings of the 51st Annual Meeting of The Institute of Navigation (1995)*, 1995, pp. 67–85.
- [56] E. D. Kaplan and C. Hegarty, *Understanding GPS/GNSS: principles and applications*. Artech house, 2017.
- [57] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of applied mathematics*, vol. 2, no. 2, pp. 164–168, 1944.
- [58] D. W. Marquardt, "An algorithm for least-squares estimation of non-linear parameters," *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.
- [59] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.
- [60] Y. Liu, W. Liu, X. Zhang, Y. Liang, X. Tao, and L. Ma, "An improved gnss ambiguity best integer equivariant estimation method with laplacian distribution for urban low-cost rtk positioning," *Satellite Navigation*, vol. 5, no. 1, p. 12, 2024.
- [61] J. Levinson and S. Thrun, "Robust vehicle localization in urban environments using probabilistic maps," in *2010 IEEE international conference on robotics and automation*. IEEE, 2010, pp. 4372–4378.
- [62] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Voxelized gicp for fast and accurate 3d point cloud registration," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 054–11 059.
- [63] 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.
- [64] J. Yin, A. Li, T. Li, W. Yu, and D. Zou, "M2dgr: A multi-sensor and multi-scenario slam dataset for ground robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2266–2273, 2022.
- [65] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3d perception," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2032–2039, 2018.
- [66] T.-M. Nguyen, S. Yuan, M. Cao, Y. Lyu, T. H. Nguyen, and L. Xie, "Ntu viral: A visual-inertial-ranging-lidar dataset, from an aerial vehicle viewpoint," *The International Journal of Robotics Research*, vol. 41, no. 3, pp. 270–280, 2022.