

Article

GPLVINS: Tightly Coupled GNSS-Visual-Inertial Fusion for Consistent State Estimation with Point and Line Features for Unmanned Aerial Vehicles

Xinyu Chen, Shuaixin Li , Ruifeng Lu and Xiaozhou Zhu * 

Defense Innovation Institute, Chinese Academy of Military Science, Beijing 100071, China; chenxinyu_1123@sina.com (X.C.); lsx_navigation@sina.com (S.L.); ruifeng_lu123@sina.com (R.L.)

* Correspondence: zhuxiaozhou.niidt@gmail.com; Tel.: +86-176-1018-7799

Highlights

What are the main findings?

- Proposes the GPLVINS system for UAVs, which builds a tightly coupled GNSS-visual-inertial nonlinear optimization framework by fusing point and line features on the basis of GVINS. This addresses the issue of insufficient feature extraction in traditional point-feature VIO under texture-sparse environments and enhances the stability of UAV 6-DoF pose estimation.
- Optimizes the traditional LSD line feature extraction algorithm: short line segments are filtered out via non-maximum suppression and length threshold screening. This not only reduces computational cost but also integrates line reprojection residuals into the optimization process, further improving positioning accuracy.

What are the implication of the main findings?

- By comparing the performance of GPLVINS with GVINS, PL-VIO, and VINS-Fusion in indoor, outdoor, and indoor–outdoor transition scenarios, GPLVINS demonstrates superior positioning performance compared with other algorithms. Our system can handle complex situations such as drastic changes in lighting, loss of GNSS signals, or feature degradation, making it more suitable for the practical operational requirements of UAVs.
- Offers a more reliable state estimation scheme for UAV autonomous navigation. Particularly in GNSS-constrained or visually sparse feature scenarios, the incorporation of line features supplements environmental constraints and reduces the risk of pose drift, laying a foundation for subsequent extensions to stereo vision and adaptation to larger-scale textureless scenarios.



Academic Editor: Pablo Rodríguez-González

Received: 26 September 2025

Revised: 12 November 2025

Accepted: 14 November 2025

Published: 17 November 2025

Citation: Chen, X.; Li, S.; Lu, R.; Zhu, X. GPLVINS: Tightly Coupled GNSS-Visual-Inertial Fusion for Consistent State Estimation with Point and Line Features for Unmanned Aerial Vehicles. *Drones* **2025**, *9*, 801. <https://doi.org/10.3390/drones9110801>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Abstract

The employment of linear features to enhance the positioning precision and robustness of point-based VIO (visual-inertial odometry) has attracted mounting attention, especially for UAV (unmanned aerial vehicle) applications where reliable 6-DoF pose estimation is critical for autonomous navigation, mission execution, and safety. This paper presents GPLVINS—GNSS (global navigation satellite system)-point-line-visual-inertial navigation system—a UAV-tailored enhancement of the nonlinear optimization-based GVINS (GNSS-visual-inertial navigation system). Unlike GVINS, which struggles with feature extraction in weak-texture environments and depends entirely on point features, GPLVINS innovatively integrates line features into its state optimization framework to enhance robustness and accuracy. While existing studies adopt the LSD (line segment detector) algorithm for line feature extraction, this approach often generates numerous short line segments

in real-world scenes. Such an outcome not only increases computational costs but also degrades pose estimation performance. In order to address this issue, the present study proposes an NMS (non-maximum suppression) strategy for the refinement of LSD. The line reprojection residual is then formulated as the distance between point and line, which is incorporated into the nonlinear optimization process. Experimental validations on open-source datasets and self-collected UAV datasets across indoor, outdoor, and indoor-outdoor transition scenarios demonstrate that GPLVINS exhibits superior positioning performance and enhanced robustness for UAVs in environments with feature degradation or drastic lighting intensity variations.

Keywords: GPLVINS; line feature; sensor fusion; UAV; UAV navigation; pose estimation

1. Introduction

With the rapid expansion of UAV applications, UAVs are increasingly operating in complex and unstructured environments. In these scenarios, precise 6-DoF pose estimation is not only essential but mission-critical. Even minor positioning drift can lead to UAV collision with obstacles, mission failure, or safety hazards. The widely used VINS-Mono [1], often deployed in UAVs, employs visual-inertial fusion with factor graph optimization to accurately track trajectories, maintaining robustness during rapid movements. In contrast, GVINS [2] tightly couples GNSS data based on VINS-Mono, which effectively enhances the pose estimation performance of UAVs in outdoor environments. Although traditional GNSS systems perform effectively in outdoor scenarios with robust GNSS signals, they encounter significant challenges in environments where UAVs frequently operate, such as GNSS-denied indoor spaces, weak-signal urban canyons, and sparse feature environments. In such environments, the loss of GNSS signals causes the system to degrade to VINS (visual-inertial navigation system). Meanwhile, sparse point features or low-texture environments pose severe challenges to the feature extraction and tracking capabilities of the VINS system, leading to frequent localization drift or even complete positioning failure in UAVs that rely solely on GVINS—risks that are unacceptable for autonomous UAV operations. To address the issue of GVINS positioning failure in low-texture environments, we incorporate line constraints into the GVINS framework to enhance both the positioning accuracy and the system robustness. This forms the basis of the proposed GPLVINS algorithm for UAV-integrated positioning.

The LSD [3] algorithm in OpenCV is commonly used for line feature extraction, but its efficacy to detect large numbers of short line segments increases computational costs and impairs real-time performance. This issue is particularly prominent in UAV applications, as short line segments raise computational complexity and hinder real-time processing. Additionally, short segments are also difficult to track and match, which reduces positioning accuracy and leads to unstable UAV flight. To mitigate this, this study optimizes the LSD algorithm using NMS and a length threshold to filter out short line segments and improve extraction efficiency. The line reprojection residual is defined as the distance from the midpoint of a line segment to its reprojected line, which is integrated into the nonlinear optimization process to provide more reliable constraints for UAV pose estimation.

Building on improvements to GVINS, this paper introduces GPLVINS. The key contributions of this study are as follows:

- Based on GVINS, GPLVINS incorporates line constraints to enhance the localization performance and robustness of the system in low-texture environments, effectively addressing the problem of UAV positioning failure in such settings.

- The LSD line feature detection algorithm is optimized using an NMS strategy to filter short line segments—a modification that significantly improves algorithm efficiency and ensures favorable real-time performance, fully meeting the requirements of real-time UAV applications.
- A self-developed UAV was utilized to independently collect experimental datasets, and these datasets, together with open-source datasets, were used to validate the performance of GPLVINS. Experiments demonstrate that GPLVINS achieves superior positioning in indoor, outdoor, and transition scenarios, with enhanced robustness under feature degradation and lighting changes.

2. Related Work

The academic research on multi-source information fusion positioning technologies for UAV is vast, with VIO being one of the most prominent approaches for UAV navigation. VIO fuses visual and inertial data to provide solutions, and the fusion methods are generally categorized into two groups: those based on the extended Kalman filter (EKF) and those based on graph optimization. A prominently used EKF-based VINS approach for UAVs is the multiple state constraint Kalman filter (MSCKF) [4]. Patrick et al. introduced OpenVINS [4], an open-source EKF-based visual-inertial estimation framework widely adopted in UAV research, which has undergone continuous refinement by subsequent researchers. For instance, Yang et al. enhanced the positioning accuracy of OpenVINS for UAVs by integrating line features [5], addressing the limitations of point-only features in UAV-relevant weak-texture scenes. Bai et al. proposed a modified ORB-SLAM2 algorithm [6], which fuses IMU data with wheel encoder data using the EKF to enhance the performance of VIO. On the other hand, graph optimization-based VINS methods jointly optimize various measurements to estimate the optimal state. These methods are preferred for many high-precision UAV applications. To control optimization time, historical states and measurements are commonly marginalized, while sliding window optimization is employed for recent states [1,2,7,8]. This paper focuses on graph optimization-based methods tailored for UAV navigation. Based on the types of visual features utilized, current approaches are categorized into point-based methods and point-line hybrid methods.

Over the recent years, numerous feature point-based approaches for UAVs have been proposed based on the ShiTomasi corner extraction method [9], such as [1,2,10,11]. Among these, GVINS is highly representative in UAV applications, as it tightly fuses GNSS raw data with image and IMU data, and is capable of achieving high-precision positioning results for outdoor UAV missions. However, this method is highly dependent on feature point extraction, which may lead to low-precision pose estimation in degraded environments such as weak-texture scenes. This is a major limitation for UAVs operating in diverse real-world scenarios. The aim of this study is to make GVINS less dependent on feature point extraction and improve its performance in environments with weak textures. The original pure point feature optimization framework has been integrated with line features, which improves positioning robustness through joint minimization of line reprojection residuals, making it suitable for UAV positioning in complex environments.

In weak-texture environments, relying solely on feature points may be insufficient for high-precision UAV pose estimation, necessitating additional scene constraints from other geometric features in the environment, such as line features. For instance, both PL-VIO [12] and PLS-VIO [13] incorporate line features into their original point-based VIO frameworks to enhance the algorithms' positioning accuracy and robustness in weak-texture environments. Notably, these two algorithms directly employ the LSD algorithm [3] for line feature extraction. This approach proves to be extremely time-consuming, which

is incompatible with the constraints of UAV on-board computing resources. Currently, the application of the LSD algorithm has become a practical bottleneck for the practical implementation of point-line fused VINS systems in UAVs. This study aims to improve the LSD algorithm and enhance the detection speed of line features to meet the requirements of UAV flight.

Since cameras and IMUs only impose relative constraints between two states, this leads to the problem of cumulative drift, especially during long-term UAV operation of VINS systems. The GNSS data can be used to reduce such drift and establish a connection between the local coordinate system and the global coordinate system. The approaches are mainly categorized into two types: loose coupling and tight coupling. For example, Ref. [14] presents a state estimation systems that achieves loose coupling of GNSS, visual, and inertial data via an EKF framework. Ref. [2] tightly couples raw GNSS measurement data with VINS-Mono, significantly eliminating the long-term drift of VINS-Mono while demonstrating strong robustness in complex environments. The GPLVINS proposed in this study is a drone-oriented VINS solution developed based on GVINS. This approach simultaneously integrates point and line features, tightly coupling raw GNSS observations with visual and inertial data, and employs nonlinear optimization to solve the system state.

3. Method

Building on GVINS, we propose GPLVINS, which integrates line features into GVINS to enhance its robustness and performance for UAV applications. This paper focuses on the fusion of these line features with a design that considers UAV on-board computing constraints, with the GPLVINS system's architecture being illustrated in Figure 1. In this paper, UAV sensor configurations include a monocular camera, IMU, and GNSS receiver.

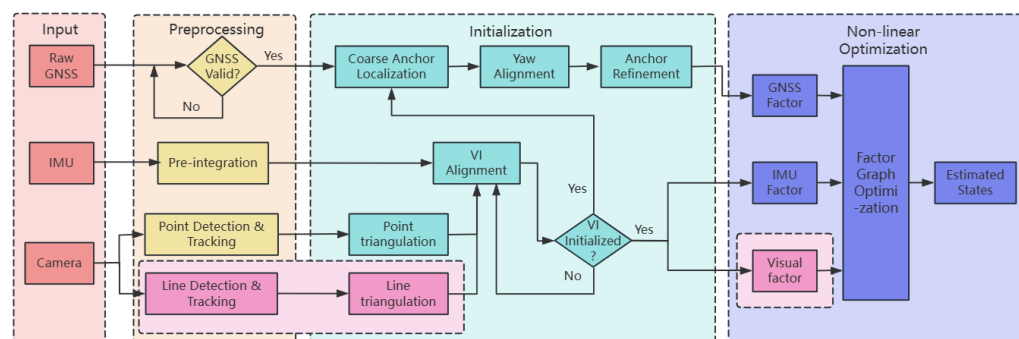


Figure 1. The block diagram of the GPLVINS system, where the pink box indicates the improved part based on GVINS. Our system processes raw data inputs and feeds them into a nonlinear optimizer, estimating system states through sliding window optimization.

First of all, three coordinate systems are defined: the world coordinate system π_w , the IMU coordinate system π_b , and the camera coordinate system π_c . Gravity aligns with the Z-axis of the world coordinate system. The IMU and camera have six degrees of freedom, and their extrinsic parameters are assumed to be known.

This section briefly introduces GVINS, which tightly couples raw GNSS measurement information, visual information, and inertial information for drift-free UAV positioning. It involves three steps: data preprocessing, system initialization, and nonlinear optimization. During preprocessing, GNSS signals are filtered, and image features are extracted and tracked. Initialization aligns image and inertial coordinates, then uses a coarse-to-fine approach to determine and refine an anchor point, enabling absolute position calculation from relative data. Measurements are modeled within a factor graph to constrain system states. GVINS's multi-sensor fusion ensures stable output in challenging environments,

reducing VIO drift and maintaining accuracy under noise. The subsequent focus is on GPLVINS, especially improvements in line feature processing and optimization. Other details can be found in [2].

3.1. MAP Estimation

The core of problem modeling involves the conversion of measurements from visual, inertial, and GNSS sensors into probabilistic constraints on the system state. The utilization of nonlinear optimization facilitates the minimization of measurement residuals, thereby enabling the estimation of the state. The approach under discussion is regarded as a maximum a posteriori (MAP) estimation method. Under the assumptions that all measurements are mutually independent and each measurement's noise adheres to a zero-mean Gaussian distribution, the MAP problem can be rephrased as minimizing the sum of multiple cost terms, with each term corresponding to a specific measurement.

$$\begin{aligned}
 \mathcal{X}^* &= \arg \max_{\mathcal{X}} p(\mathcal{X} \mid \mathbf{z}) \\
 &= \arg \max_{\mathcal{X}} p(\mathcal{X}) p(\mathbf{z} \mid \mathcal{X}) \\
 &= \arg \max_{\mathcal{X}} p(\mathcal{X}) \prod_{i=1}^n p(\mathbf{z}_i \mid \mathcal{X}) \\
 &= \arg \min_{\mathcal{X}} \left\{ \|\mathbf{r}_p - \mathbf{H}_p \mathcal{X}\|^2 + \sum_{i=1}^n \|\mathbf{r}(\mathbf{z}_i, \mathcal{X})\|_{\mathbf{P}_i}^2 \right\}
 \end{aligned} \tag{1}$$

where $\mathcal{X}$ represents the system state, which is described in detail in Section 3.4. $\mathbf{z}$ is the collection of n independent measurements from different sensors. $\{\mathbf{r}_p, \mathbf{H}_p\}$ contains the prior information of $\mathcal{X}$. $\mathbf{r}(\cdot)$ represents the residual function of each measurement and $\|\cdot\|_{\mathbf{P}}$ is the Mahalanobis norm.

Such formulas can be represented by a factor graph. So we decompose the optimization problem into independent factors of correlation states and measurements. Section 3.4 provides a detailed introduction to the optimization factors.

3.2. Data Preprocessing

3.2.1. Preprocessing of Raw GNSS Data

Validity checks are performed on GNSS signals, including whether the received signals originate from the four major satellite systems (GPS, GLONAS, Galileo, and BeiDou), with respect to ephemeris validity, ephemeris timeliness, signal measurement validity, satellite tracking status, and satellite elevation angle. Only the screened GNSS data can proceed to subsequent processing. The remaining poor GNSS data will be discarded.

3.2.2. IMU Data Preprocessing

IMU data undergoes pre-integration to extract relative motion between consecutive frames, independent of initial state. The midpoint method calculates relative displacement, rotation, and velocity change, with Jacobian and covariance matrices updated dynamically to accurately characterize error propagation. This offers reliable constraints and uncertainty estimates for optimization. Detailed formulas are presented in [1].

3.2.3. Image Data Preprocessing

The process involves two parts: point feature detection and tracking and line feature detection and tracking. Point features are detected with ShiTomasi [9] and tracked using KLT [15], and outliers were removed via RANSAC-based epipolar constraints. For line features, the LSD algorithm is improved by applying non-maximum suppression to eliminate short segments, with a length threshold being used to refine results. Extracted segments

are sorted by length, and only those exceeding the minimum length $L_{\min}$ are retained for tracking and optimization. The length threshold $L_{\min}$ is modeled as:

$$L_{\min} = \lceil \eta \cdot \min(H, W) \rceil \quad (2)$$

where η is the scaling factor, based on the smaller of image height H or width W , with the ceiling function $\lceil \cdot \rceil$ being applied. It has a significant impact on the time of line feature extraction. To balance accuracy and efficiency, $\eta = 0.1$ is used, resulting in the line detection being shown in Figure 2. The improved LSD filters short segments and speeds up extraction.



Figure 2. Effect diagram of line feature detection where η is set to 0.1. For the frame from the self-collected subway station dataset, short line segments have been filtered out, leaving only the long, easy-to-track line segments.

Line feature matching and tracking are based on [16,17]. LBD (Line band descriptor) is used to create descriptors for each line segment, capturing key properties across multiple dimensions into vectors. KNN (K-nearest neighbors) matches these descriptors to find corresponding line segments in different frames, enabling line feature tracking.

3.3. System Initialization

3.3.1. Feature Triangulation

It consists of feature point and line feature triangulation, following similar principles. Point triangulation constructs a projection equation from initial and observation frames, solved via SVD (singular value decomposition) for 3D coordinates. Line triangulation involves back-projecting 2D line segments into 3D space; each segment corresponds to a plane, and the intersection of these planes from different frames gives the 3D line. The specific process can be seen in Figure 3. The positions p_1 and p_2 represent the line segment's locations in initial and observed frames. To select the best observation frame, we use the angle between the normal vectors $\mathbf{n}_i$ and $\mathbf{n}_j$ of the matched planes. And we choose the frame with the smallest cosine value of this angle.

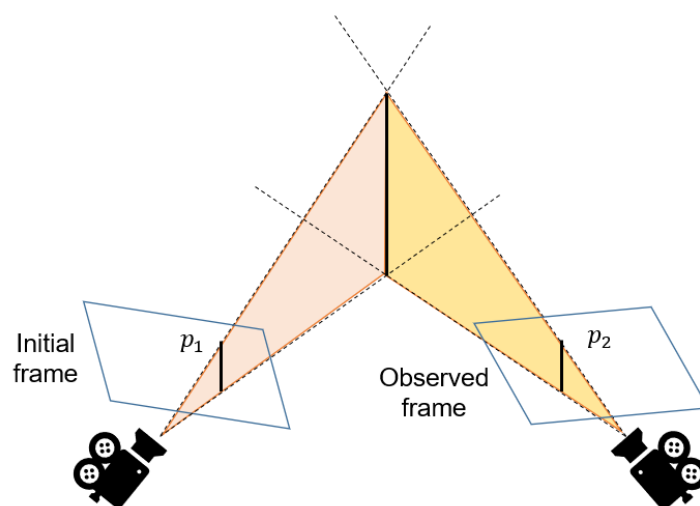


Figure 3. Schematic diagram of the line feature triangulation principle. Through this operation, the positions of feature line segments in 3D space can be obtained.

3.3.2. Visual-Inertial Alignment

After IMU pre-integration and image processing, VI (visual-inertial) alignment begins by matching the scaled visual structure with IMU data for collaboration, as detailed in [2]. It involves: first, creating an error equation based on pre-integrated rotation and gyroscope biases, solving it via least squares to estimate biases; second, forming constraint equations from IMU and visual data to derive key parameters like scale, gravity, and velocity; and third, using the scale to align the visual trajectory with IMU results and orienting gravity with the Z-axis of π_w for unified alignment. This step aligns the IMU coordinate system with the camera coordinate system and aligns the Z-axis of the world coordinate system with gravity. The world coordinate system and the ENU (east–north–up) coordinate system can be aligned with only a yaw angle difference remaining now.

3.3.3. GNSS Initialization

To fuse global GNSS measurements with local image and IMU data, we need an anchor point that serves as the focal point connecting the global coordinate system and the local coordinate system. The position of that point within both coordinate systems must be known. This section presents a coarse-to-fine GNSS initialization approach consisting of three steps.

First of all, a coarse ECEF (Earth-centered, Earth-fixed) coordinate of the reference point is obtained via SPP (single point positioning) for rough localization. Second, the yaw angle ψ is calibrated using low-noise Doppler measurements to determine the orientation between the ENU coordinate system and the local world coordinate system. As long as the yaw angle is determined, the local world coordinate system can be converted to the ENU coordinate system through rotation. An optimization is formulated for the initial yaw:

$$\underset{\delta t, \psi}{\text{minimize}} \sum_{k=1}^n \sum_{j=1}^{p_k} \left\| r_{\mathcal{D}}(\tilde{\mathbf{z}}_{r_k}^{s_j}, \mathcal{X}) \right\|_{\sigma_{r_k, dp}^{s_j}}^2 \quad (3)$$

where n represents the sliding window size, p_k represents the number of satellites observed in k -th epoch inside the window, $r_{\mathcal{D}}$ is the Doppler residual, $\tilde{\mathbf{z}}_{r_k}^{s_j}$ is the raw Doppler shift observation value of the receiver for the j -th satellite at the k -th epoch, $\mathcal{X}$ is the system state, and $\sigma_{r_k, dp}^{s_j}$ is the standard deviation of the Doppler observation noise for the j -th satellite at the k -th epoch. In this step, the velocity $\mathbf{v}_b^w$ is fixed to the VIO result, and the receiver clock

drifting rate δt_k is assumed to be constant inside the window. The coarse anchor will be used to compute the direction vector κ_r^s and rotation $\mathbf{R}_n^e$.

Finally, the coarse ECEF coordinates are refined by integrating VIO data, aligning the trajectory with the world coordinate system. This step takes the VIO position result as prior information and optimizes the following problem using sliding window measurements.

$$\underset{\delta \mathbf{t}, \mathbf{p}_{anc}^e}{\text{minimize}} \left(\sum_{k=1}^n \sum_{j=1}^{p_k} \left\| r_{\mathcal{P}}(\tilde{\mathbf{z}}_{r_k}^{s_j}, \mathcal{X}) \right\|_{\sigma_{r_k, pr}^{s_j}}^2 + \sum_{k=1}^n \left\| \mathbf{r}_{\mathcal{T}}(\tilde{\mathbf{z}}_{k-1}^k, \mathcal{X}) \right\|_{\mathbf{D}_{t,k}}^2 \right) \quad (4)$$

$\delta \mathbf{t}$ is the receiver clock biases, $\mathbf{p}_{anc}^e$ is the refined anchor point coordinate, $r_{\mathcal{P}}$ is the code pseudorange residual, $\tilde{\mathbf{z}}_{r_k}^{s_j}$ is the raw code pseudorange observation value, $\sigma_{r_k, pr}^{s_j}$ is the standard deviation of the code pseudorange observation noise, $\mathbf{r}_{\mathcal{T}}$ is the clock bias residual, $\tilde{\mathbf{z}}_{k-1}^k$ is the raw clock bias observation value, and $\mathbf{D}_{t,k}$ is the covariance matrix related to this residual. Solving this problem refines the anchor point coordinate and receiver clock bias for each GNSS epoch, completing the initialization phase of the estimator. If available, RTK trajectories can serve as ground truth to evaluate positioning accuracy.

3.4. Nonlinear Optimization

Next, the observation data of all sensors are jointly optimized to estimate the state variables. Our system employs a sliding window optimization strategy to estimate the state of key frames within the window. The system state vector to be estimated is denoted as $\mathcal{X}$, which comprises three major categories: body motion state, visual feature parameters, and global correlation parameters. The system state to be estimated is $\mathcal{X}$.

$$\begin{aligned} \mathcal{X} &= [\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_n, \rho_0, \rho_1, \dots, \rho_m, \mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_l, \psi]^\top \\ \mathbf{x}_k &= [\mathbf{p}_{b_k}^w, \mathbf{v}_{b_k}^w, \mathbf{q}_{b_k}^w, \mathbf{b}_a, \mathbf{b}_w, \delta \mathbf{t}, \dot{\delta t}]^\top, k \in [0, n] \end{aligned} \quad (5)$$

Among them, n represents the number of key frames within the sliding window, m represents the number of visual feature points within the window, l represents the number of feature line segments within the window, $\mathbf{x}_n$ represents the body motion state of the n -th keyframe, ρ_m represents the inverse depth of the m -th feature point, $\mathbf{o}_l$ represents the four-parameter orthogonal parameterization of the l -th 3D line feature, and ψ represents the yaw angle between the world coordinate system and the ENU coordinate system, serving as the core parameter linking the local and global coordinate systems. The body motion state $\mathbf{x}_k$ at each keyframe integrates two core types of parameters: IMU motion state and GNSS clock parameters. $\mathbf{x}_k$ contains the position $\mathbf{p}_{b_k}^w$, velocity $\mathbf{v}_{b_k}^w$, orientation $\mathbf{q}_{b_k}^w$, acceleration bias $\mathbf{b}_a$, gyroscope bias $\mathbf{b}_w$, GNSS receiver clock bias $\delta \mathbf{t}$, and clock drift rate $\dot{\delta t}$ of the n -th frame node. Due to the satellite data used being sourced from four major satellite navigation systems: Beidou, Galileo, GLONASS and GPS, the clock biases of the different systems are different. The clock bias $\delta \mathbf{t}$ is expressed as:

$$\delta \mathbf{t} = [\delta t_1, \delta t_2, \delta t_3, \delta t_4]^\top \quad (6)$$

The factor graph of GPLVINS is displayed in Figure 4, and the following is an introduction to each optimization factor.

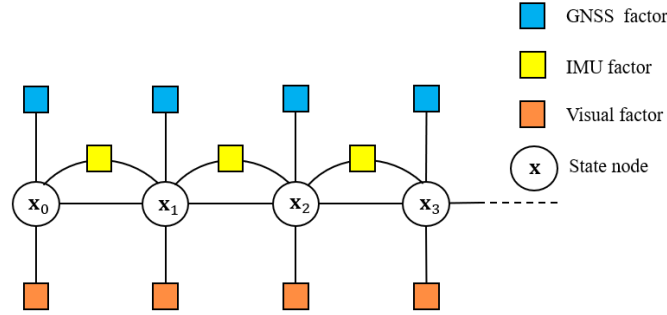


Figure 4. The factor graph of The nonlinear optimization problem. System states are circles. GNSS factors are blue squares. IMU pre-integration factors are yellow squares, and visual factors are orange squares.

3.4.1. IMU Factor

For the IMU data within the timestamp interval $[t_k, t_{k+1}]$, through a series of derivations in [2], the residual of the IMU pre-integration measurement is modeled as:

$$\begin{aligned} \mathbf{r}_B(\mathbf{z}_{b_{k+1}}^{b_k}, \mathcal{X}) &= [\delta \mathbf{a}_{b_{k+1}}^{b_k}, \delta \boldsymbol{\beta}_{b_{k+1}}^{b_k}, \delta \boldsymbol{\theta}_{b_{k+1}}^{b_k}, \delta \mathbf{b}_a, \delta \mathbf{b}_g]^\top \\ &= \begin{bmatrix} \mathbf{R}_w^{b_k} (\mathbf{p}_{b_{k+1}}^w - \mathbf{p}_{b_k}^w + \frac{1}{2} \mathbf{g}^w \Delta t_k^2 - \mathbf{v}_{b_k}^w \Delta t_k) - \hat{\mathbf{a}}_{b_{k+1}}^{b_k} \\ \mathbf{R}_w^{b_k} (\mathbf{v}_{b_{k+1}}^w + \mathbf{g}^w \Delta t_k - \mathbf{v}_{b_k}^w) - \hat{\boldsymbol{\beta}}_{b_{k+1}}^{b_k} \\ 2 [\mathbf{q}_{b_k}^{w-1} \otimes \mathbf{q}_{b_{k+1}}^w \otimes (\hat{\mathbf{y}}_{b_{k+1}}^{b_k})^{-1}]_{xyz} \\ \mathbf{b}_{ab_{k+1}} - \mathbf{b}_{ab_k} \\ \mathbf{b}_{wb_{k+1}} - \mathbf{b}_{wb_k} \end{bmatrix} \end{aligned} \quad (7)$$

where $\delta \mathbf{a}_{b_{k+1}}^{b_k}$, $\delta \boldsymbol{\beta}_{b_{k+1}}^{b_k}$, and $\delta \boldsymbol{\theta}_{b_{k+1}}^{b_k}$ represent the pre-integration residuals of position, velocity, and attitude. $\delta \mathbf{b}_a$ and $\delta \mathbf{b}_g$ represent the residuals of the accelerometer bias and gyroscope bias.

3.4.2. Visual Factor

First, the feature point reprojection factor is introduced. In the image data preprocessing stage, feature corner points are detected in image frames and then further tracked using the Lucas–Kanade method. The projection process can be defined as:

$$\tilde{\mathcal{P}} = \pi_c \left(\mathbf{R}_b^c \left(\mathbf{R}_w^b \mathbf{x}^w + \mathbf{p}_w^b \right) + \mathbf{p}_b^c \right) + \mathbf{n}_c \quad (8)$$

$\tilde{\mathcal{P}} = [u, v]^\top$ is the coordinate of the feature point on the 2D image plane, $\mathbf{x}^w$ is the coordinate of its 3D point in π_w , and $\pi_c(\cdot)$ is the projection function of camera. $\mathbf{n}_c$ is the noise. For the feature point l observed in the i -th and j -th frames, with inverse depth ρ_l , the residual can be modeled as:

$$\mathbf{r}_C(\tilde{\mathbf{z}}_l, \mathcal{X}) = \tilde{\mathcal{P}}_l^{c_j} - \pi_c(\hat{\mathbf{x}}_l^{c_j}) \quad (9)$$

where $\hat{\mathbf{x}}_l^{c_j}$ is defined as:

$$\hat{\mathbf{x}}_l^{c_j} = \mathbf{R}_b^c \left(\mathbf{R}_{b_j}^{b_i} \left(\mathbf{R}_w^b \left(\mathbf{R}_{b_i}^b \left(\mathbf{R}_c^b \frac{1}{\rho_l} \pi_c^{-1}(\tilde{\mathcal{P}}_l^{c_i}) + \mathbf{p}_c^b \right) + \mathbf{p}_{b_i}^w \right) + \mathbf{p}_{b_j}^w \right) + \mathbf{p}_b^c \right) \quad (10)$$

$\tilde{\mathcal{P}}_l^{c_j}$ represents the image coordinates of feature point l observed in image frame j within the camera coordinate system, whereas $\hat{\mathbf{x}}_l^{c_j}$ represents the three-dimensional position of

feature point l in π_c , estimated based on the current system state and then projected onto the image plane using the projection function of camera $\pi_c(\cdot)$. The difference between these two values constitutes the point reprojection residual.

Second, we are going to introduce the line reprojection factor. First, the line geometric transformation is defined. $\mathbf{T}_w^c = [\mathbf{R}_w^c, \mathbf{t}_w^c]$ represents the transformation from π_w to π_c . Therefore, the line coordinates in the camera coordinate system is defined as:

$$\mathbf{L}_c = \begin{bmatrix} \mathbf{n}_c \\ \mathbf{d}_c \end{bmatrix} = \mathbf{T}_w^c \mathbf{L}_w = \begin{bmatrix} \mathbf{R}_w^c & [\mathbf{t}_w^c]_{\times} \mathbf{R}_w^c \\ 0 & \mathbf{R}_w^c \end{bmatrix} \begin{bmatrix} \mathbf{n}_w \\ \mathbf{d}_w \end{bmatrix} \quad (11)$$

$[\cdot]_{\times}$ represents the skew-symmetric matrix. Transforming the line coordinate $\mathbf{L}_c$ to the image plane, we can then yield the projected line, which can be expressed as:

$$\mathbf{l} = [l_0, l_1, l_2]^T = \mathbf{K} \mathbf{n}_c \quad (12)$$

$\mathbf{K}$ is the line projection matrix. We can compute $\mathbf{n}_c$ from Equation (11). For line feature j observed in the initial frame and the i -th frame, the line reprojection residual model is modeled as the distance from the midpoint m of line segment j observed in image frame i to the projected line. The residual can then be defined as:

$$\mathbf{r}_L(\tilde{\mathbf{z}}_{\mathcal{L}_j}^{c_i}, \mathcal{X}) = \mathbf{d}(\mathbf{m}, \mathbf{l}) = \frac{\underline{\mathbf{m}}^T \mathbf{l}}{\sqrt{l_1^2 + l_2^2}} \quad (13)$$

$\mathbf{d}(\mathbf{m}, \mathbf{l})$ represents the distance function from a point to a line, and $\underline{\mathbf{m}}$ represents the homogeneous coordinate of the midpoint m of the line segment.

3.4.3. GNSS Factor

First, the pseudorange factor is defined that when the signal is received, the ECI coordinate system coincides with the ECEF coordinate system, i.e., $\mathbf{p}_r^E = \mathbf{p}_r^e$. Due to the Earth's rotation, the ECEF coordinate system during signal transmission differs from that at the moment of signal reception. The ECEF coordinate system during signal transmission is defined as $(\cdot)^{e'}$. Therefore, the satellite position has to be determined by Equation (11):

$$\mathbf{p}_s^E = \mathbf{R}_z(-\omega_E t_f) \mathbf{p}_s^{e'} \quad (14)$$

where ω_E is the Earth's rotational angular velocity, t_f is the signal propagation time, and $\mathbf{R}_z(\theta)$ is a rotation matrix around the z-axis of the ECI coordinate system. The pseudorange residual for a single-code pseudorange measurement of satellite s_j at time t_k can then be modeled as:

$$r_{\mathcal{P}}(\tilde{\mathbf{z}}_{r_k}^{s_j}, \mathcal{X}) = \|\mathbf{R}_z(\omega_E t_f) \mathbf{p}_s^{e'} - \mathbf{p}_{r_k}^E\| + c(\mathbf{1}_{s_j}^T \delta \mathbf{t}_k - \Delta t^{s_j}) + T_{r_k}^{s_j} + I_{r_k}^{s_j} - \tilde{P}_{r_k}^{s_j} \quad (15)$$

where r_k represents the raw satellite signal at time t_k . $T_{r_k}^{s_j}$ and $I_{r_k}^{s_j}$ are the tropospheric delay and ionospheric delay, respectively.

Second, the Doppler factor is introduced. The velocity of the receiver in the ECEF coordinate system can be obtained from the velocity in π_w through the following equation:

$$\mathbf{v}_r^e = \mathbf{R}_n^e \mathbf{R}_w^n \mathbf{v}_b^w \quad (16)$$

Analogous to the pseudorange measurement, the ECI coordinate system is defined as the ECEF coordinate system at the moment of satellite signal reception, so $\mathbf{v}_r^E = \mathbf{v}_r^e$. The

velocity of the satellite in the ECEF coordinate system during signal transmission, $\mathbf{v}_s^{e'}$, can then be expressed in the ECI coordinate system as:

$$\mathbf{v}_s^E = \mathbf{R}_z(-\omega_E t_f) \mathbf{v}_s^{e'} \quad (17)$$

So the residual of the Doppler measurement for satellite s_j at time t_k can be modeled as:

$$r_{\mathcal{D}}(\tilde{\mathbf{z}}_{r_k}^{s_j}, \mathcal{X}) = \frac{1}{\lambda} \mathbf{k}_{r_k}^{s_j T} (\mathbf{R}_z(-\omega_E t_f) \mathbf{v}_{s_j}^{e'} - \mathbf{v}_{r_k}^E) + \frac{c}{\lambda} (\delta t_k - \Delta t^{s_j}) + \Delta \tilde{f}_{r_k}^{s_j} \quad (18)$$

where r_k represents the raw satellite signal at time t_k .

4. Experiment

This section validates the improved performance of the modified LSD algorithm and evaluates GPLVINS's positioning accuracy for UAVs using the open-source dataset and self-collected UAV data. All experiments simulate real UAV operating conditions. For outdoor UAV tests, RMSE based on RTK (real-time kinematic) ground truth is used. A high-precision RTK system provides accurate latitude, longitude, and altitude. The output of the evaluated algorithm is compared with RTK data, and RMSE (root mean square error) quantifies error and accuracy. The RMSE formula is:

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}} \quad (19)$$

where n is the quantity of estimated values, y_i is the i -th true value, and $\hat{y}_i$ is the i -th estimated value.

For indoor and indoor–outdoor switch evaluations, a cumulative error method is used. The UAV starts at a point, follows a route, and returns to form a loop. Accuracy and stability are assessed by the ratio of the position difference between start and end to the total travel length. The cumulative error formula is:

$$e = \frac{d}{l} \times 100\% \quad (20)$$

where d is the distance from the UAV trajectory's starting point to its ending point and l is the total length of the UAV trajectory. In the dataset, the IMU frequency is 200 Hz, the camera frequency is 20 Hz, the GNSS receiver frequency is 10 Hz, the LSD line detection threshold η is 0.1, and the optimization window size of both GPLVINS and GVINS are 10. All experiments were conducted on an Intel Core i7-10870U CPU @2.20 GHz. GPLVINS, GVINS, PL-VIO, and VINS-Fusion were run on the Ubuntu 20.04 system with ROS Noetic installed.

4.1. Performance Validation Experiments for the Modified LSD

In the line feature extraction stage, this study modifies the LSD algorithm using the NMS method. A length threshold is set to eliminate short line segments, and only those with lengths exceeding this threshold proceed to the subsequent feature tracking stage. To verify the performance improvement of the modified LSD algorithm, experiments are conducted on a self-collected subway station dataset. Specifically, GPLVINS and PL-VIO (which uses LSD for line feature extraction) are run on the MH-05-difficult test set within the EuRoC dataset. The improvement of the modified LSD is evaluated by comparing the average time consumed for line feature extraction.

Table 1 shows that the modified LSD algorithm achieves a substantial speed improvement of approximately 75%. Additionally, the line feature tracking speed of GPLVINS has also increased. This is because short line segments that are difficult to track are filtered out during the line feature extraction stage, thereby saving some tracking time. Further-

more, the output frequency of GPLVINS state estimation results is maintained at 10 Hz, a frequency that is adequate to meet the positioning requirements for the majority of drone flight missions. On average, the GPLVINS algorithm consumes 476 MB of memory and utilizes 533% of total CPU resources. The computational demands of this algorithm are not particularly stringent, and most drones possess the necessary capabilities to execute it.

Table 1. Comparison of average runtime Between GPLVINS and PL-VIO. Experimental results indicate that the time consumption of the modified LSD algorithm is significantly reduced. The bold numbers indicate that the corresponding algorithm has better performance in the comparison.

Algorithm Name	GPLVINS	PL-VIO
Point Feature Detection and Tracking/ms	12	12
Line Feature Detection (LSD)/ms	16 (Modified)	65
Line Feature Tracking/ms	9	15
Output Frequency of State Estimation Results/Hz	10	5

4.2. Experimental Evaluation of Outdoor Scenarios

The dataset used in the outdoor scenario is from the open-source dataset of [2], which was recorded at the Fok Ying Tung Sports Center. The evaluation method is the RMSE evaluation method based on RTK ground truth. The GPLVINS algorithm and GVINS algorithm were run on this dataset, respectively, and the trajectory data generated by the positioning algorithms were analyzed against the RTK trajectory data in the dataset. As shown in Figure 5, by carefully observing the algorithm trajectories and RTK trajectories, it is evident that the trajectory of GPLVINS is closer to the RTK trajectory.

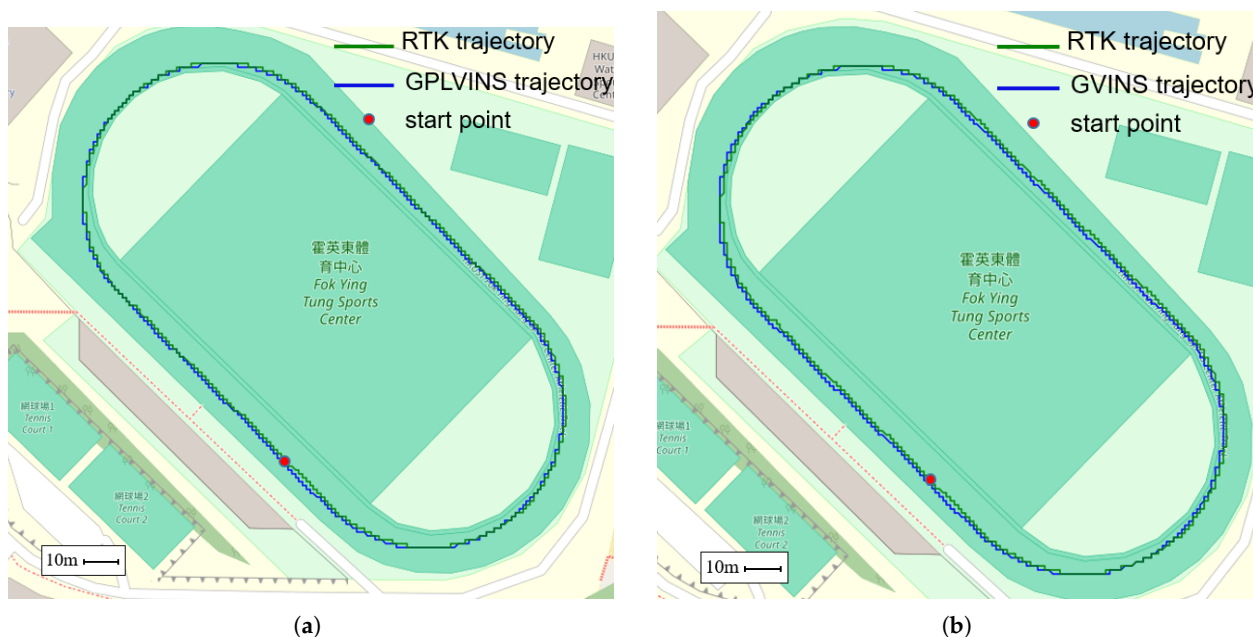


Figure 5. Trajectory comparison of GPLVINS and GVINS on the open-source dataset of [2]: (a) GPLVINS and (b) GVINS. With the RTK trajectory being regarded as the ground truth, the closer the trajectory of the algorithm is to the RTK trajectory, the higher its positioning accuracy. In (a), the trajectory of GPLVINS is closer to the green RTK trajectory, which is more clearly observable in the straight section of the runway.

We use RTK data as the true values to analyze the errors in the latitude, longitude, and altitude data produced by the GPLVINS and GVINS algorithms. We create error curves for each of these three components and calculate the RMSE for each.

The error curves in Figure 6 show that the new line feature constraints enhance the accuracy of UAV altitude measurements. The discrepancy in the error curves is primarily reflected in the altitude error graph. In the initial phase, the two curves also show a similar variation pattern: they first increase and then decrease to the minimum value around the 500th data point. In the subsequent phase, the altitude error of the GPLVINS algorithm remains consistently below 0.2 m, maintaining a relatively low level. In contrast, the altitude error of the GVINS algorithm begins to increase continuously around the 1600th data point, even exceeding 0.6 m at its maximum, before starting to decrease around the 2400-th data point.

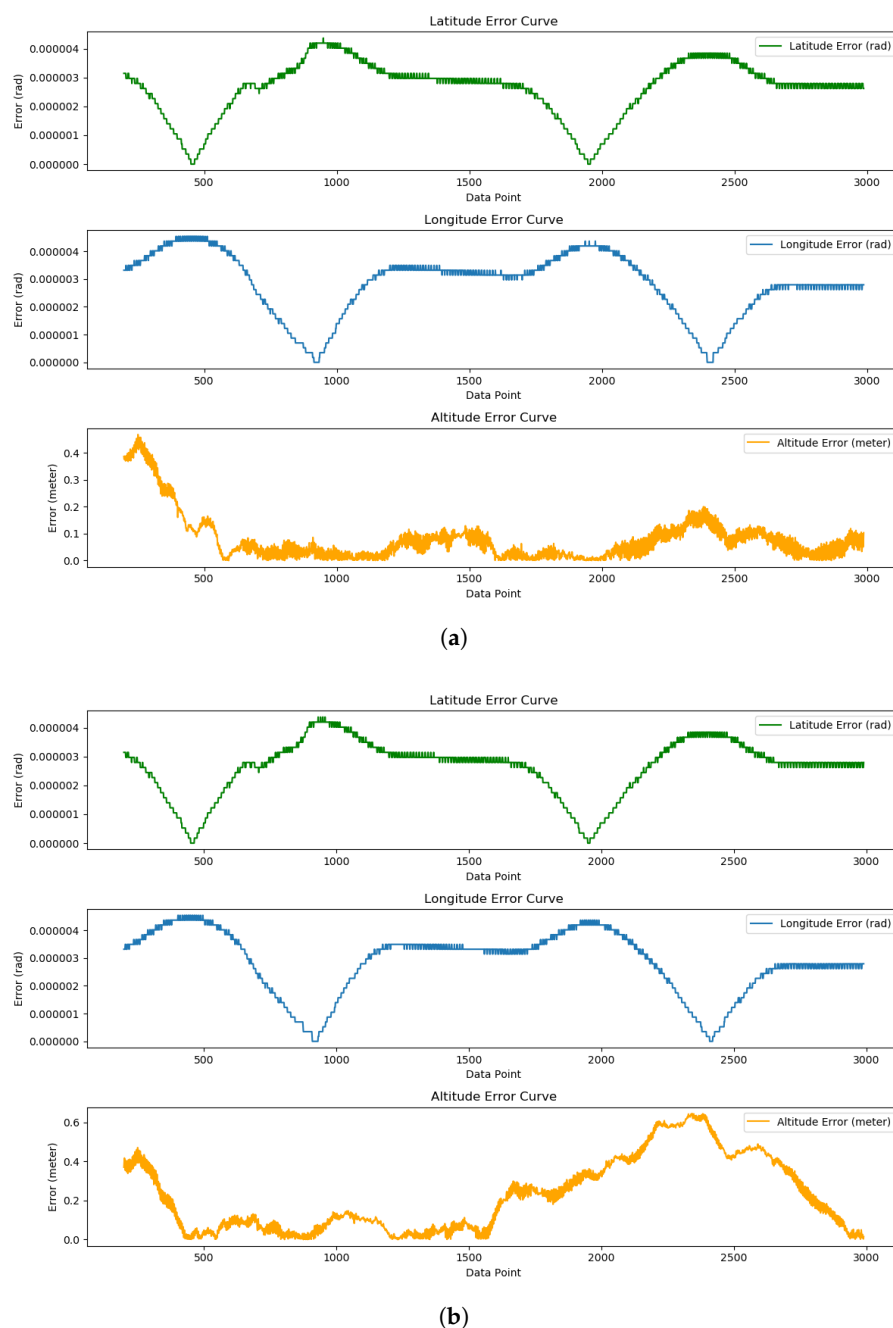


Figure 6. Error curves of GPLVINS and GVINS: (a) GPLVINS and (b) GVINS. No discernible difference can be observed from the longitude and latitude error curves. In contrast, the altitude error curve indicates that the altitude positioning accuracy of GPLVINS is higher than that of GVINS, which is consistent with the analysis result derived from RMSE.

The RMSE results for both algorithms were analyzed. Table 2 shows that GPLVINS has lower RMSEs than GVINS in latitude, longitude, and altitude data. Of these three dimensions, the improvement in altitude estimation accuracy is more significant, while the improvements in latitude and longitude estimation accuracy are limited. The reason for this lies in the fact that in outdoor environments with sufficient GNSS signals and distinct features, GNSS constraints and point constraints have already kept positioning error at a low level, which results in limited performance improvement from the addition of line constraints. In contrast, when GNSS signals are poor or in low-texture environments, the performance improvement brought by line constraints becomes more pronounced, which can be observed in subsequent experiments. The RMSE results confirm the earlier observation that GPLVINS algorithm trajectories align more closely with RTK trajectories. Thus, GPLVINS offers better positioning performance than GVINS in outdoor environments.

Table 2. RMSE results of latitude, longitude, and altitude for two algorithms. When translated into linear distances, the RMSE values of longitude and latitude also lie in the 10^{-6} -m order of magnitude—leading to very small positioning errors. In this case, line constraints yield limited improvements in the positioning accuracy of longitude and latitude while delivering more significant improvements in the positioning accuracy of altitude.

Evaluation Data Type	GPLVINS	GVINS
Latitude/ $^{\circ}$	2.806859×10^{-6}	2.813002×10^{-6}
Longitude/ $^{\circ}$	3.047753×10^{-6}	3.069171×10^{-6}
Altitude/ m	0.117652	0.292413

A paired *t*-test was conducted on the RMSE results of the two algorithms to ensure the reliability of the comparison results. The results are shown in Table 3. The findings suggest that the observed discrepancy in performance between the two algorithms is not attributable to random factors. Rather, the difference is statistically significant and reproducible, thereby substantiating the hypothesis that the positioning performance of GPLVINS in outdoor environments is indeed superior to that of GVINS.

Table 3. Results of paired *t*-tests for RMSE between GPLVINS and GVINS. The null hypothesis states that the mean difference between the RMSEs of the two algorithms is 0. The alternative hypothesis states that the mean difference between the RMSEs of the two algorithms is less than 0. If *p* is less than α , the null hypothesis is rejected and the alternative hypothesis accepted, meaning GPLVINS achieves better positioning performance than GVINS.

RMSE Type	t-Statistic	p-Value	Significance ($\alpha = 0.05$)
Latitude RMSE	4.129	3.75×10^{-5}	Significant ($p < 0.05$)
Longitude RMSE	8.856	1.45×10^{-18}	Highly Significant ($p < 0.05$)
Altitude RMSE	41.63	4.93×10^{-295}	Extremely Significant ($p \ll 0.05$)

4.3. Experimental Evaluation of Indoor and Outdoor Scenarios

Three UAV data sequences were recorded in this test, with the recording scenarios including a conference room, an underground parking lot, and a subway station, respectively (see Figure 7).

Figure 8 shows the recording equipment, including a GNSS receiver, a monocular camera, and an IMU, all handheld during data collection. The data collection process simulates low-altitude flight of a UAV. The three datasets have different features: the indoor conference room dataset lacks GNSS signals, with challenging features like white columns, windows, LED displays, and reflections. The underground parking lot dataset features indoor–outdoor transitions, with low light and no GNSS, causing potential drift at exits.

The subway station dataset also involves indoor–outdoor transitions, with dim lighting, weak GNSS signals, and significant brightness changes at exits.

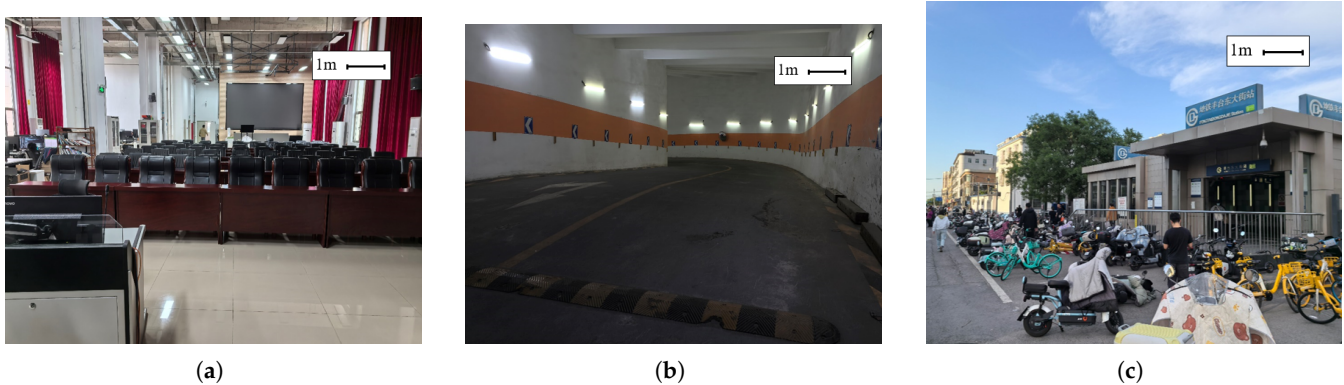


Figure 7. Real-scene photos of recording scenarios: (a) conference room; (b) underground parking lot; (c) subway station. Sparse features in partial scenarios and GNSS signal failure will introduce cumulative errors to trajectory estimation, leading the algorithm-estimated trajectory to exhibit a discrepancy between its start and end points. This phenomenon is thus used to evaluate the performance of the algorithm.



Figure 8. Equipment used for recording datasets. The UAV is equipped with a monocular camera, an IMU, a GNSS receiver, and a LiDAR on its top, where the LiDAR was not used in this experiment.

Next, GPLVINS and GVINS were run on the three sequences, respectively, and the cumulative error evaluation method was used to evaluate the performance of GPLVINS. The result trajectories are shown in Figure 9.

Jitter occurs in the bottom turn of the conference room trajectory due to a black LED display in the camera image, which degrades the environment and impairs feature detection and tracking. Consequently, the GPLVINS trajectory self-intersects after this turn, unlike the GVINS trajectory. As the actual UAV recording route intersected itself, GPLVINS demonstrates superior robustness to feature degradation compared with GVINS.

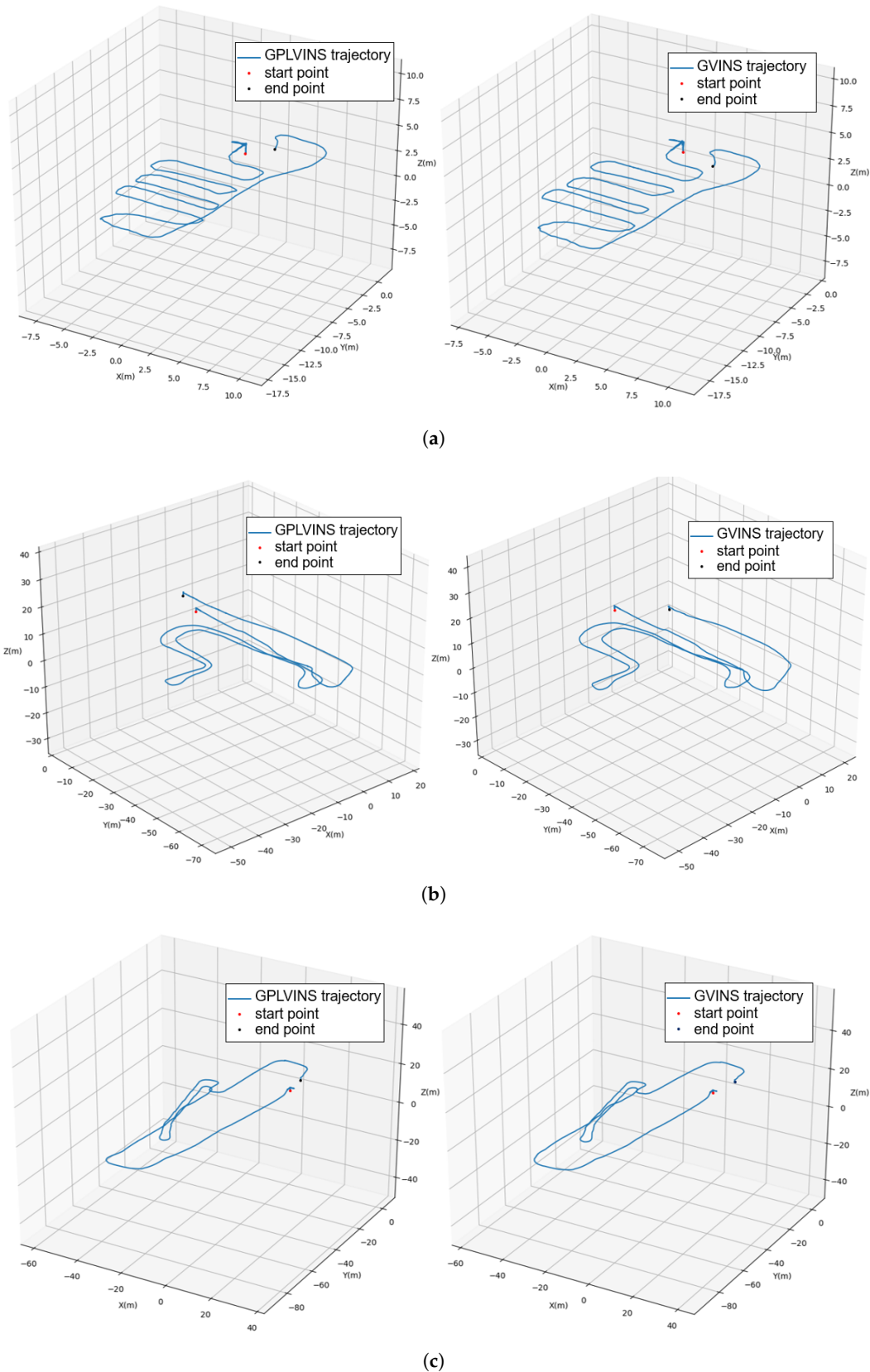


Figure 9. Trajectory plots of GPLVINS and GVINS: (a) trajectory plot of the conference room dataset; (b) trajectory plot of the underground parking lot dataset; (c) trajectory plot of the subway station dataset. The turn near the endpoint in (b) marks the exit of the underground parking garage, where trajectory drift is clearly visible. The downward portion of the trajectory in (c) corresponds to the interior of the subway station.

Both algorithms drift noticeably when the UAV exits the underground parking due to lighting changes affecting feature detection. However, GPLVINS drifts less, indicating better stability. Inside the parking lot, both trajectories show good performance in dim environments with no significant drift. In the subway station, GVINS incorrectly indicates the stairs are reached earlier than GPLVINS, which does not match reality. This suggests GVINS's indoor performance is worse, while GPLVINS remains stable, consistent with the conference room results.

In order to ensure the accuracy and rigor of the experimental results, this study also employed the EuRoC dataset for testing purposes, incorporating PL-VIO and VINS-Fusion as comparison algorithms. VINS-Fusion adopts the monocular + IMU mode, and the playback speed of the dataset for PL-VIO is set to $0.1\times$. The window size of GPLVINS, GVINS, PL-VIO, and VINS-Fusion are set to 10. The cumulative error analysis of the UAV trajectory data yields the results presented in Table 4. Due to significant illumination variations in the parking lot and subway station datasets, VINS-Fusion failed to estimate the pose at the exit of the parking lot and the exit of the subway station, respectively. In contrast, PL-VIO also performed poorly in these two datasets, showing a striking difference from its performance in the EuRoC datasets. The primary cause is also attributed to illumination changes. However, GPLVINS and GVINS, which benefit from GNSS constraints in outdoor scenarios, achieved relatively better performance. Synthesizing the experimental results from both self-collected datasets and the EuRoC datasets, it can be concluded that the positioning performance of the GPLVINS algorithm is superior to that of other algorithms, except in the MH-03-medium dataset—making it more suitable for real-world UAV missions.

Table 4. Cumulative error evaluation results of different algorithms in various datasets. In the self-collected datasets, the positioning performance of GPLVINS is significantly superior to that of other algorithms. In the EuRoC datasets, the positioning performance of GPLVINS outperforms other algorithms except in the MH-03-medium dataset. The bold numbers indicate that the corresponding algorithm has better performance in the comparison.

Dataset Name	GPLVINS	GVINS	PL-VIO	VINS-Fusion
Conference Room	2.277%	4.054%	2.871%	3.184%
Parking Lot	2.263%	6.172%	7.264%	-
Subway Station	3.103%	4.537%	7.143%	-
MH-02-easy	0.248%	0.471%	0.285%	0.264%
MH-03-medium	0.201%	0.238%	0.203%	0.188%
MH-05-difficult	0.339%	0.651%	0.471%	0.529%

5. Conclusions

This paper introduces a tightly integrated system tailored for UAVs that combines camera, IMU, and GNSS data in a nonlinear optimization framework, using point and line features. The LSD algorithm is enhanced with non-maximum suppression, and line feature constraints are added to improve accuracy. Experiments demonstrate that GPLVINS exhibits excellent positioning performance in outdoors, indoors, and transitional environments, displaying strong robustness in challenging environments. These advantages are crucial for real-world UAV operations.

However, GPLVINS nevertheless exhibits inherent limitations. In environments with extremely sparse texture, drastic illumination variations, or highly dynamic scenarios, even the improved LSD algorithm struggles to extract a sufficient number of valid line features. This results in an increase in line constraint errors or even the complete absence of such constraints and may even induce localization drift due to insufficient features. During the initialization phase, the system relies on Doppler shift measurements to constrain

the yaw offset. When the receiver velocity is lower than the noise level of the Doppler shift, the constraint on the yaw offset becomes ineffective, making it difficult to achieve reliable estimation of the yaw angle. Therefore, the system requires a minimum velocity of 0.3 m/s during the initialization phase to ensure effective yaw angle constraint. The current line extraction method can be further optimized; developing a more stable, efficient scheme suited for UAV pose estimation is crucial. Future work will involve attempting to replace LSD with line feature extraction algorithms that are more conducive to state estimation, extending monocular vision to stereo vision, and conducting tests in large-scale, low-texture, and low-illumination indoor scenarios to better leverage line constraints.

Author Contributions: Conceptualization, X.Z.; methodology, S.L.; software, R.L.; validation, X.C.; formal analysis, X.C.; resources, X.C.; writing—original draft preparation, X.C.; writing—review and editing, X.C.; visualization, X.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data available on request due to restrictions because the collected data contains classified information.

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

References

1. Qin, T.; Li, P.; Shen, S. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [\[CrossRef\]](#)
2. Cao, S.; Lu, X.; Shen, S. GVINS: Tightly Coupled GNSS–Visual–Inertial Fusion for Smooth and Consistent State Estimation. *IEEE Trans. Robot.* **2022**, *38*, 2004–2021. [\[CrossRef\]](#)
3. von Gioi, R.G.; Jakubowicz, J.; Morel, J.-M.; Randall, G. LSD: A Fast Line Segment Detector with a False Detection Control. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 722–732. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Geneva, P.; Eckenhoff, K.; Lee, W.; Yang, Y.; Huang, G. OpenVINS: A Research Platform for Visual-Inertial Estimation. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 4666–4672.
5. Yang, Y.; Geneva, P.; Eckenhoff, K.; Huang, G. Visual-Inertial Odometry with Point and Line Features. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 2447–2454.
6. Bai, Y.; Yang, F.; Liu, T.; Zhang, J.; Hu, X.; Wang, Y. Research on Lidar Vision Data Fusion Algorithm Based on Improved ORB-SLAM2. In Proceedings of the 2025 4th International Conference on Artificial Intelligence, Internet of Things and Cloud Computing Technology (AloTC), Guilin, China, 8–10 August 2025; pp. 174–179.
7. Leutenegger, S.; Lynen, S.; Bosse, M.; Siegwart, R.; Furgale, P. Keyframe-based visual-inertial odometry using nonlinear optimization. *Int. J. Robot. Res.* **2014**, *34*, 314–334. [\[CrossRef\]](#)
8. Duan, C.; Liu, R.; Li, N.; Li, S.; Tang, Q.; Dai, Z.; Zhu, X. Tightly Coupled RTK-Visual-Inertial Integration with a Novel Sliding Ambiguity Window Optimization Framework. *IEEE Trans. Intell. Transp. Syst.* **2025**, *early access*. [\[CrossRef\]](#)
9. Shi, J. Good features to track. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 21–23 June 1994; pp. 593–600.
10. Stumberg, L.V.; Usenko, V.; Cremers, D. Direct sparse visualinertial odometry using dynamic marginalization. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 2510–2517.
11. Xia, C.; Li, X.; Li, S.; Zhou, Y. Invariant-EKF-Based GNSS/INS/Vision Integration with High Convergence and Accuracy. *IEEE/ASME Trans. Mechatron.* **2024**, *early access*. [\[CrossRef\]](#)
12. He, Y.; Zhao, J.; Guo, Y.; He, W.; Yuan, K. PL-VIO: Tightly-Coupled Monocular Visual-Inertial Odometry Using Point and Line Features. *Sensors* **2018**, *18*, 1159. [\[CrossRef\]](#) [\[PubMed\]](#)
13. Wen, H.; Tian, J.; Li, D. PLS-VIO: Stereo Vision-inertial Odometry Based on Point and Line Features. In Proceedings of the 2020 International Conference on High Performance Big Data and Intelligent Systems, Shenzhen, China, 23–23 May 2020; pp. 1–7.
14. Angelino, C.V.; Baraniello, V.R.; Cicala, L. UAV position and attitude estimation using IMU, GNSS and camera. In Proceedings of the 2012 15th International Conference on Information Fusion, Singapore, 9–12 July 2012; pp. 735–742.
15. Baker, S.; Matthews, I. Lucas-kanade 20 years on: A unifying framework. *Int. J. Comput. Vis.* **2004**, *56*, 221–255. [\[CrossRef\]](#)

16. Kaehler, A.; Bradski, G. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2016.
17. Zhang, L.; Koch, R. An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency. *J. Vis. Commun. Image Represent.* **2013**, *24*, 794–805. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.