ORIGINAL RESEARCH **Open access**

Real-Time Visual–Inertial SLAM Using Enhanced Point and Line Features under Challenging Indoor Conditions

Paolo Ricci[1*], Marco De Luca[1], Giulia Ferraro[2], Antonio Russo[1], Luca Bianchi[3]

Abstract

Given the shortcomings of conventional point-line feature visual-inertial simultaneous localization and mapping (SLAM) systems, which suffer from limited accuracy and cannot run in real time in indoor environments with weak texture and fluctuating lighting, this paper introduces an inertial SLAM technique that leverages point-line vision, specifically designed for indoor environments with weak texture and fluctuating lighting. Initially, Bilateral Filtering is used in conjunction with the Speeded Up Robust Features (SURF) point feature extraction and the Fast Library for Approximate Nearest Neighbors (FLANN) algorithms to enhance the robustness of the extracted point features. Next, a minimum density threshold and a length-suppression parameter selection strategy for line features are developed, and geometric-constraint-based line-feature matching is integrated to improve the efficiency of line-feature processing. Moreover, the visual-inertial parameters and biases are initialized through the maximum a posteriori estimation approach. Ultimately, simulation experiments compare the suggested method against the traditional tightly-coupled monocular visual-inertial odometry employing point and line features (PL-VIO) algorithm. The simulation results indicate that the proposed inertial SLAM method based on point-line vision for indoor environments with weak texture and illumination supports effective real-time operation, achieving an average positioning accuracy increase of 22% and a 40% improvement in situations with illumination variations and blurred images.

Keywords Feature extraction, Simultaneous localization and mapping (SLAM), Fast bilateral filtering, SURF algorithm, Nearest-neighbor algorithm, Geometric constraints

*Correspondence:
Paolo Ricci
paolo.ricci@unina.it
[1] Department of AI for Artichoke and Fennel Crop Modeling, Faculty of Agriculture, University of Naples Federico II, Naples, Italy
[2] Department of Machine Learning for Thrips and Aphid Prediction, Faculty of Agriculture, University of Bologna, Bologna, Italy
[3] Department of Smart Water Pump Scheduling, Faculty of Agriculture, University of Milan, Milan, Italy

Introduction

Over the past few decades, the visual simultaneous localization and mapping (SLAM) [1] algorithm has emerged as a fundamental technology for enabling autonomous navigation in mobile robots. This approach enables mobile robots to acquire their pose and environmental data using suitable sensors in indoor areas where GPS signals are unreliable, allowing them to build maps, understand their surroundings, and independently carry out designated tasks.

Visual-inertial SLAM systems are classified into two categories depending on how vision and IMU data are combined: loosely coupled and tightly coupled. In the loosely coupled strategy, motion estimation is conducted separately by each module before fusion, as demonstrated in Refs. [2] and [3]. In contrast, the tightly coupled strategy jointly optimizes the raw measurements from both the camera and IMU, leading to more reliable globally consistent outcomes.

This is an open access article distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License (CC BY-NC-SA 4.0), which permits non-commercial use, distribution, reproduction, and adaptation in any medium or format, provided that appropriate credit is given to the original author(s) and source, a link to the license is provided, and any changes made are indicated. Adapted material must be distributed under the same license.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

Well-known tightly coupled SLAM systems include VINS-Mono [4] and ORB-SLAM2 [5]. Drawing on these foundations, researchers have developed various enhanced approaches [6, 7]. Among the feature types employed, point features are the most prevalent. Nevertheless, point feature detectors such as the Scale Invariant Feature Transform (SIFT) [8], Features from Accelerated Segment Test (FAST) [9], and the Oriented FAST and Rotated BRIEF (ORB) [10] frequently fail to identify enough point features in low-texture indoor settings like corridors, windows, or plain white walls. Additionally, factors such as rapid camera motion, insufficient texture, and lighting shifts can substantially degrade both the quality and quantity of point-feature correspondences.

Incorporating line features offers a practical solution to these limitations. Unlike point features, indoor man-made environments are rich in edges, straight structures, and distinct line segments. Line features exhibit greater resistance to illumination changes and rotational variations, providing additional geometric information that enhances the robustness and precision of SLAM systems that rely primarily on point features.

Vakhitov *et al.* [11] developed the EPn-PL and OPnPL algorithms, which integrate point and line features. Zuo *et al.* [12] used Plücker coordinates to represent line features in the front-end of a point-line SLAM system, while relying on minimal orthogonality for line parameterization during back-end optimization. Pumarola *et al.* [13] integrated line features into the ORB-SLAM framework and proposed their use for system initialization. Gomez-Ojeda *et al.* [14] were the first to release an open-source SLAM framework based on combined point and line features, giving balanced consideration to both in the loop closure module. At present, key works in this area include PL-VIO [15], PLS-VIO [16], PL-VINS [17], among others. However, SLAM systems that combine point and line features often suffer reduced efficiency at the front-end stage, where each image frame requires full feature extraction and matching. Adding line features further raises the computational demands of these steps, lowering the system's overall speed and reliability.

To overcome these challenges, this paper proposes an indoor visual-inertial SLAM algorithm centered on point-line features. The adoption of bilateral filtering and the SURF [18] feature extraction algorithm enhances the system's tolerance to lighting variations. Feature matching is performed using the fast nearest-neighbor algorithm, delivering superior accuracy compared to the baseline SLAM system. Compared with the standard point-line SLAM approach, the presented method achieves greater accuracy. Adjustments were made to the Line Segment Detector (LSD) [19] algorithm to mitigate its high computational cost and slow performance.

Additionally, the complicated original matching procedure was streamlined. Geometric constraints, combined with random-sampling consistency algorithms, were applied to reduce processing burden during feature matching, thereby improving real-time capabilities. Stepwise joint initialization was also implemented to enhance algorithmic stability. The goal of this research is to advance the real-time performance, robustness, and precision of SLAM systems operating in challenging environments.

The rest of the paper is organized as follows. The algorithm framework is proposed in Section 2. Section 3 proposes the image processing method in detail and describes the initialization work. Lastly, the experimental results are presented in Section 4 to verify the advantage demonstrated in Section 3.

The Algorithm Framework

This study concentrates on the front-end image processing and visual-inertial initialization methods, while retaining the standard nonlinear optimization structure in the back-end. The primary innovation appears in the marked region of **Figure 1**. As depicted in **Figure 1**, the overall system is constructed upon the PL-VIO framework. The current work centers on the point and line feature extraction plus matching procedures inside the data preprocessing stage:

© 2026 The Author(s). | Published by Agro Intelligence Research Press

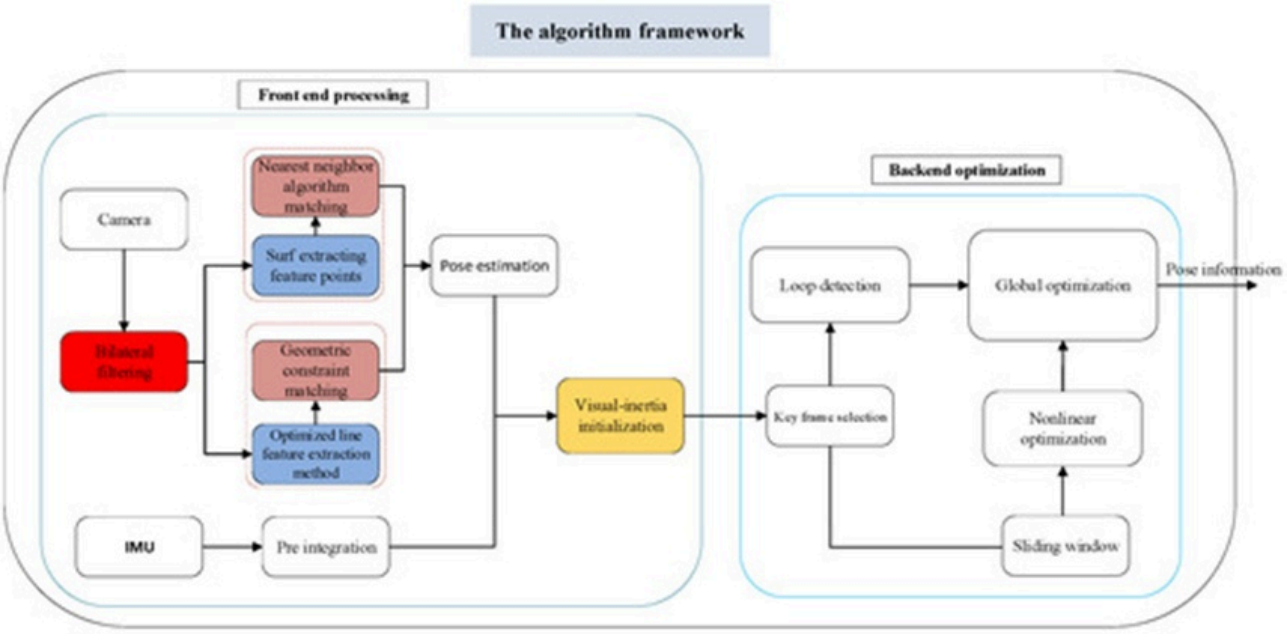


Figure 1. The algorithm framework.

(1) Thanks to its ability to detect reliable features despite shifts in translation, rotation, or viewpoint, the SURF algorithm is selected here for point feature detection. Conventional optical flow methods prove inadequate when lighting changes rapidly. Instead, this paper applies the fast nearest-neighbor algorithm (FLNN) [20], which connects smoothly with SURF parameters.

(2) For real-time feature extraction and matching, the LSD algorithm was first created to capture object outlines but falls short for locating line segments in three-dimensional space. As a result, LSD parameters are modified, and an adaptive constraint technique for line segments is suggested to sharply cut down the duration of line feature matching. Inspired by Gomez-Ojeda [21], this work introduces constraints based on pole geometry and point-line affine invariants to accelerate line-feature handling.

(3) Nonlinear optimization-based point-line feature SLAM commonly faces extended inertial initialization periods and reduced stability. To counter this, the paper implements a stepwise joint initialization [21] strategy that boosts stability while shortening the initialization phase.

Front-End Processing

Figure 2 illustrates the workflow for handling point and line features.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

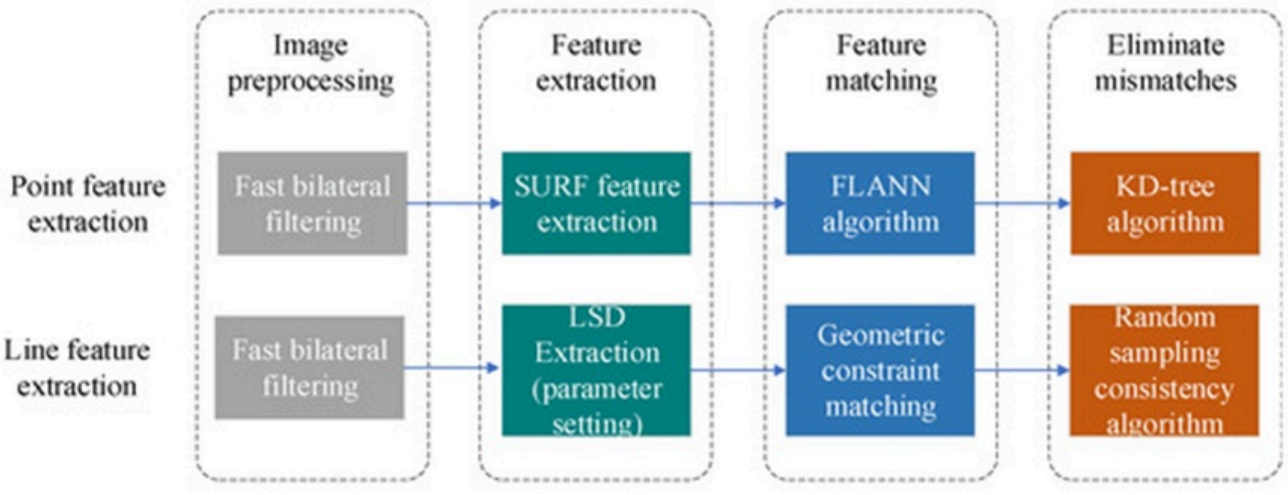


Figure 2. Point line feature processing flow chart.

The image processing

Before detecting point and line features, it is necessary to remove noise and various false points or lines. Given the local spatial adjacency and intensity similarity among neighboring pixels, bilateral filtering is performed for preprocessing [21]. This filter achieves edge retention through the joint use of spatial and range kernels in convolution, formulated as follows:

$$H_I = \frac{1}{W}\sum_{x\in N} G_{\sigma_d}(\|x-y\|)G_{\sigma r}(I_x - I_y)I_y \quad (1)$$

Where:

$$W = \sum_{x\in N} G_{\sigma_d}(\|x-y\|)G_{\sigma r}(I_x - I_y)$$
$$G_{\sigma_d} = e^{-\frac{1}{2}\left(\frac{d(x,y)}{\sigma_d}\right)^2} \quad (2)$$
$$G_{\sigma_r} = e^{-\frac{1}{2}\left(\frac{\delta(I_x,I_y)}{\sigma_r}\right)^2}$$

y stands for the target pixel value, N defines the square window around pixel y in image I, and x refers to any neighboring pixel. Ix and Iy represent pixel intensities. $G\sigma d$ functions as the spatial kernel, $\|x-y\|$ measures spatial separation, $G\sigma r$ acts as the intensity similarity kernel, σd is the spatial Gaussian deviation, and σr is the range Gaussian deviation. d (x, y) and δ (Ix, Iy) signify the Euclidean pixel distance and the intensity difference between Ix and Iy.

Consequently, in smooth image zones where $Ix-Iy$ shows little variation, the range weight nears 1, and spatial weighting prevails, mimicking standard Gaussian smoothing. At edges, large $Ix-Iy$ differences lower the domain coefficient, diminishing the kernel contribution in that area and thereby protecting boundary details.

The SURF detector operates via scale-space analysis and applies a Gaussian-smoothed Hessian matrix for point detection. For an image location (x,) at scale σ, the Hessian matrix takes the form:

© 2026 The Author(s). | Published by Agro Intelligence Research Press

$$H(I,\sigma) = \begin{bmatrix} L_{xx}(I,\sigma) & L_{xy}(I,\sigma) \\ L_{xy}(I,\sigma) & L_{yy}(I,\sigma) \end{bmatrix} \quad (3)$$

Where $(I,)$, $Lxy(I,\sigma)$, and $Lyy(I,\sigma)$ are the convolutions of the image using the second-order Gaussian derivatives at location I.

SURF replaces the Gaussian filter with an approximate Box filter. This simplifies filtering to basic sum and difference calculations of pixel values across image sub-regions, accomplished quickly via repeated integral-image queries. The approximated Hessian determinant [18] per pixel is expressed as:

$$\det(H) = D_{xx}(x)D_{yy}(x) - (0.9D_{xy}(x))^2 \quad (4)$$

By adopting the Box filter in place of the Gaussian, the algorithm multiplies Dxy by 0.9 to balance approximation inaccuracies. Discrete pixel templates replace the filters; the X-, Y-, and XY-direction templates are shown in **Figure 3**. Computations yield Dxx via the X template, Dyy via the Y template, and Dxy via the XY template. These values are then inserted into **Eq. 4** to determine the Hessian determinant for every pixel.

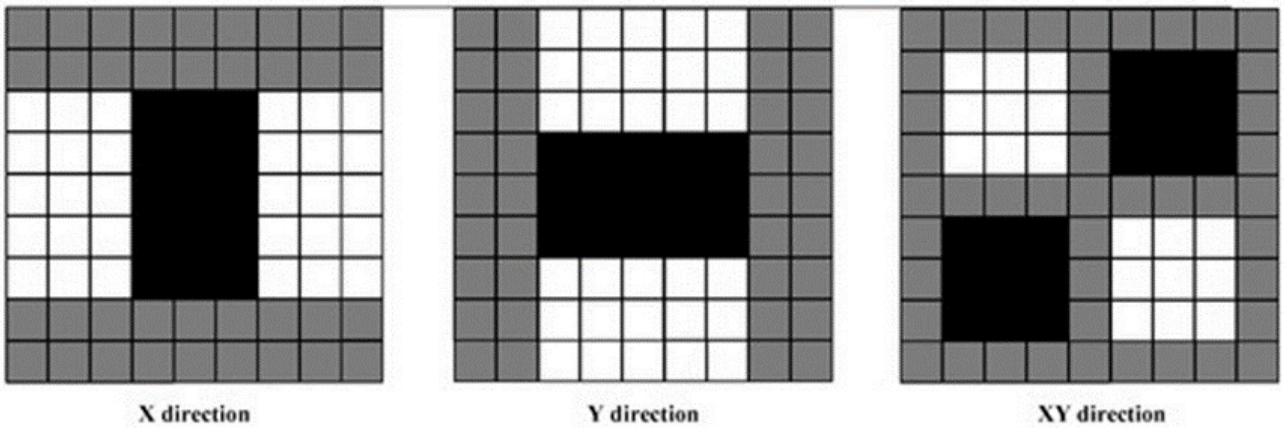


Figure 3. Box filter image processing template.

The Box filter image processing template [18] is presented below.

Starting from the source image, scale pyramids form as the square Box filter window is progressively enlarged. Convolution is sped up using integral images, enabling rapid computation of Hessian determinants. Tests confirm that doubling the Box filter size—equivalent to a pyramid level of 2—delivers the best balance of computation time and extraction performance.

During point feature detection, the SURF Hessian threshold is fixed at 800. Results from the baseline extraction appear in **Figure 4**. After applying bilateral filtering before SURF **(Figure 5)**, the number of invalid points on white walls and floors is notably reduced.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

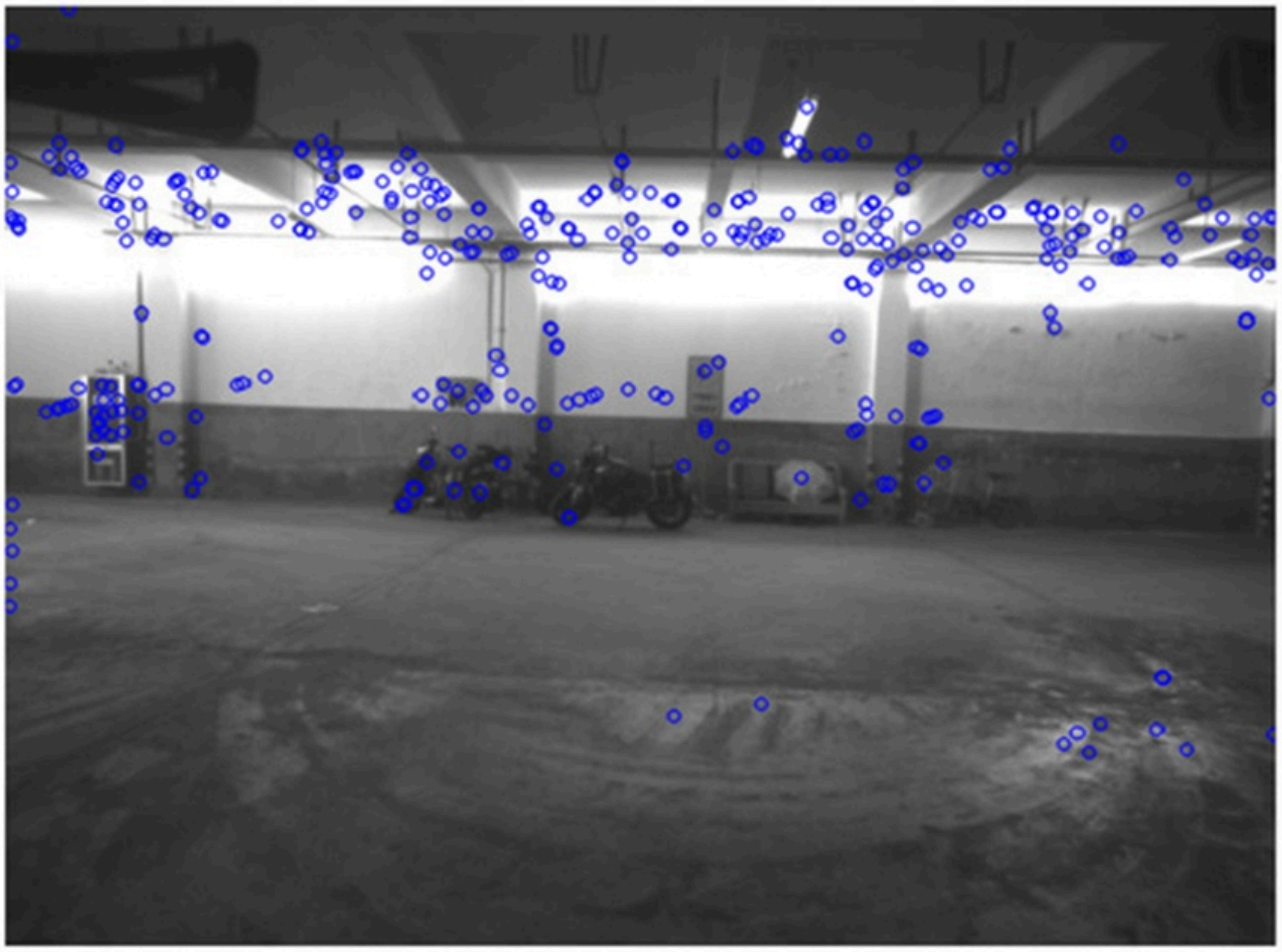


Figure 4. SURF feature extraction before bilateral filtering.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

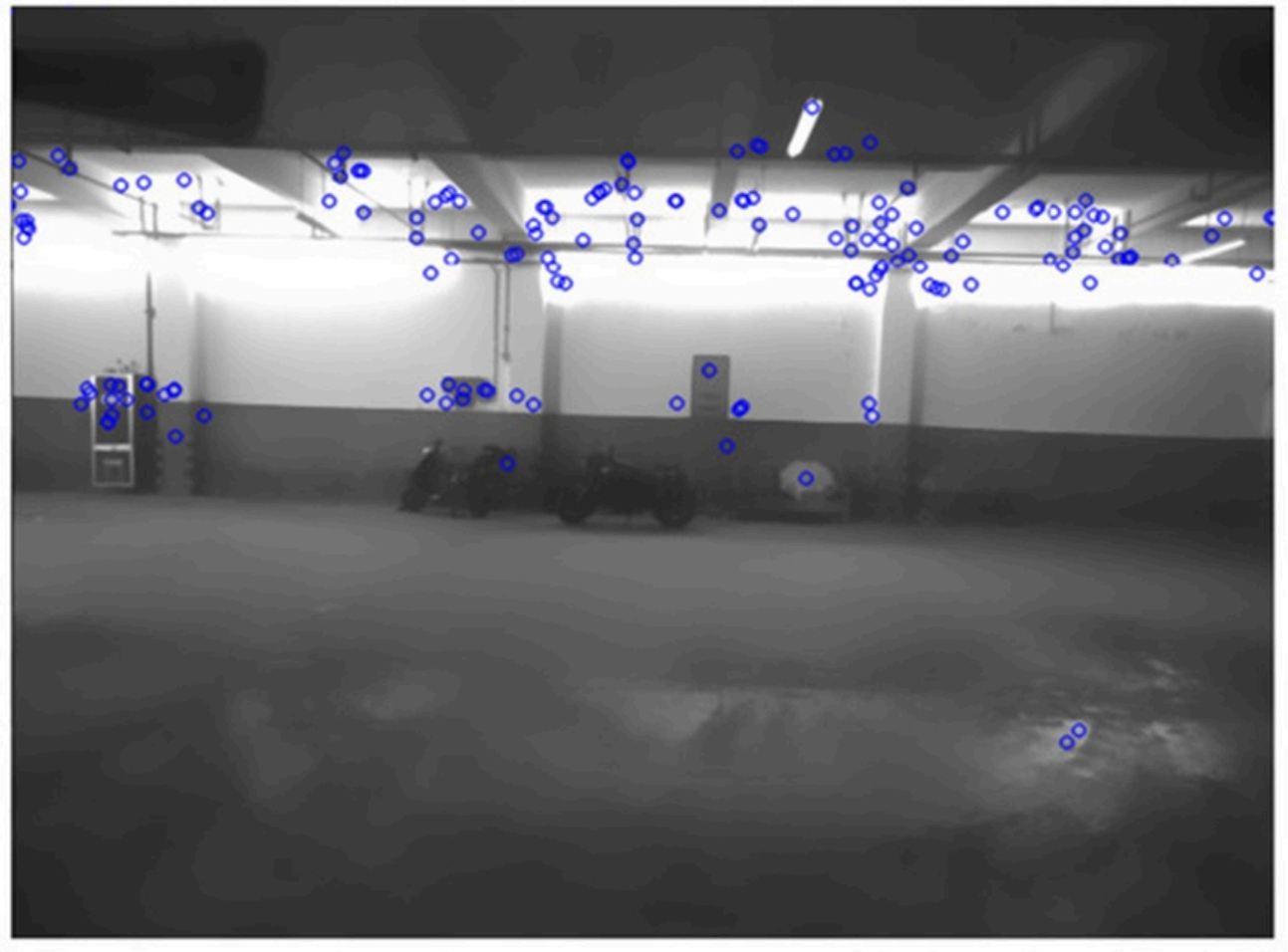


Figure 5. SURF feature extraction after bilateral filtering.

Feature matching in this paper relies on the FLANN algorithm, executed through the following stages:

(1) Point features of the initial image form the training set, and those of the subsequent image serve as the query set. Euclidean distances are computed across all pairs between the two sets.

(2) Nearest and second-nearest neighbors are kept by comparing distances, discarding all other candidates. The Euclidean distance formula is:

$$D(x,y) = \|x,y\| = \sqrt{\sum_{t=1}^{d} (X_t - Y_t)^2} \quad (5)$$

(3) Matching pairs are preserved only when the nearest and second-nearest distances meet **Eq. 6**; otherwise, they are rejected. The ratio parameter (0 < ratio < 1) acts as the decision threshold for distance differences. Higher ratio values increase pair counts at the cost of accuracy, while lower values decrease counts but raise accuracy. Testing shows that a ratio of 0.6 produces around 150 matches with limited errors.

Figure 6 demonstrates that standard SURF matching generates many pairs but still includes mismatches. Results from the current algorithm are displayed in **Figure 7**. Incorporating FLANN clearly eliminates several erroneous and redundant point matches, yielding improved overall matching quality.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

$$\frac{\text{Nearest Euclidean Distance}}{\text{Sub-near Euclidean distance}} < ratio \quad (6)$$

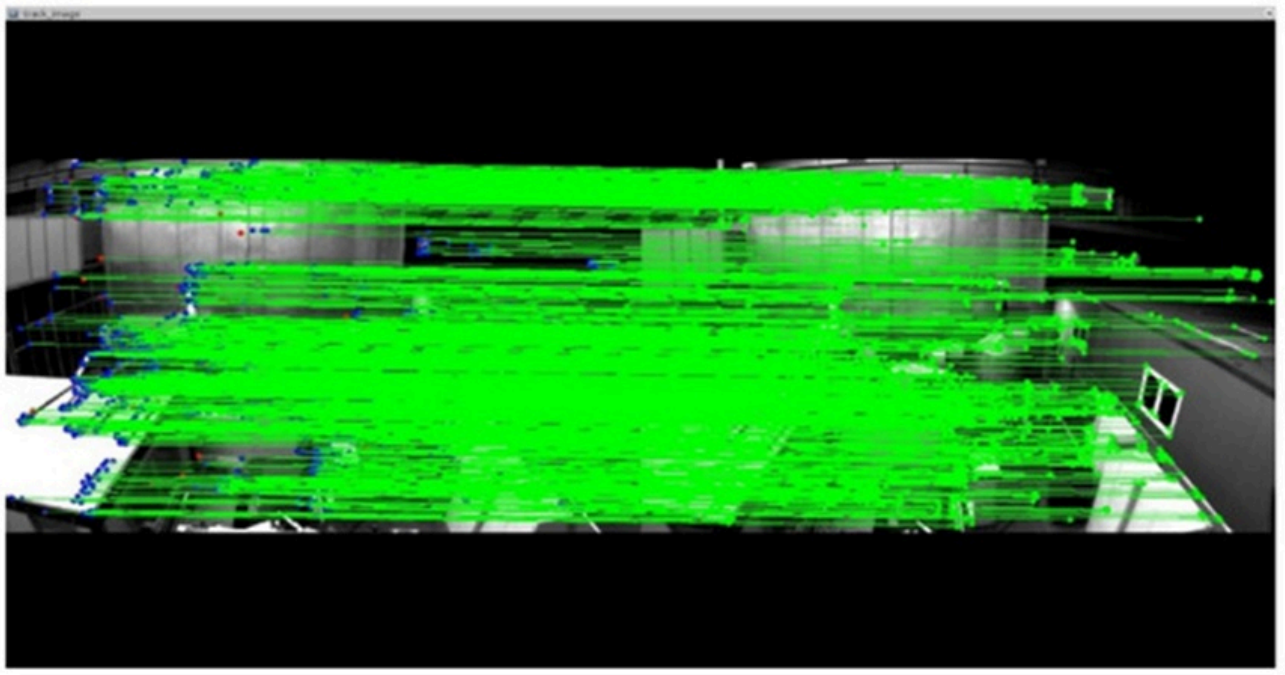


Figure 6. SURF is matched with the original algorithm.

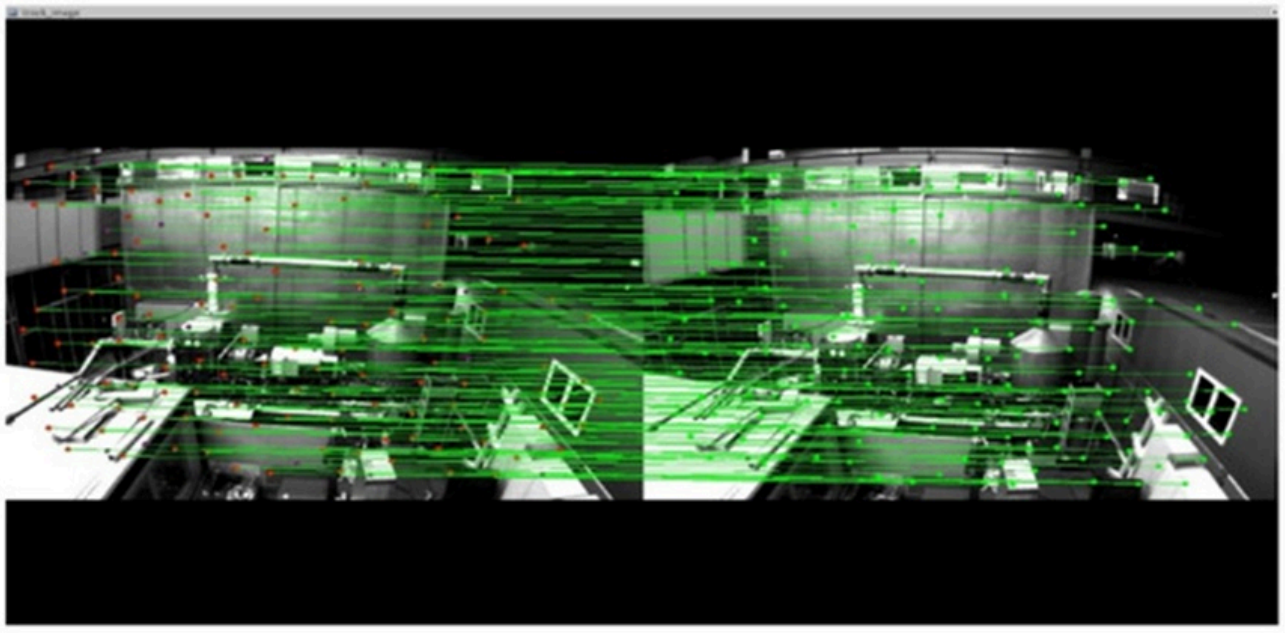


Figure 7. FLANN algorithm matching.

Proper tuning of LSD internal settings—including pyramid count, scale factor, and minimum density threshold—can greatly enhance extraction velocity. Confirmation shows faster processing when pyramid layers n = 3, the scale factor is 0.5, and the minimum density threshold is 0.5.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

To improve line feature quality, a length threshold filters out extended segments useful for pose calculation while removing short, unreliable segments with minimal impact. Each line segment length must fulfill:

$$\begin{cases} len_{l_i} \geq \{len_{\min} = \beta \bullet [\max(W_I, H_I)]\}, i \in \{1, 2, \ldots, k\} \\ \beta = 1 - e^{-0.001 \cdot k} \end{cases} \quad (7)$$

Where $lenli$ denotes the length of segment i, lenmin is the minimum segment length, WI gives the input frame width, HI gives the input frame height, • indicates ceiling operation, β is the length coefficient, and k is the count of line features in the present frame.

Although PL-VINS [17] applies an analogous approach to line quality, its empirical length factor does not adapt well to wide variations in segment numbers. Large factors risk too few trackable segments during sparse cases, while small factors leave excess short segments in dense cases. The present approach links segment count to the length factor through an equation . Low segment counts raise the threshold, preserving more candidates for tracking. High counts lower the threshold, limiting short-segment interference with pose accuracy. This yields stronger adaptability of the screening method across differing line quantities. In the end, the line feature extraction speed is three times faster than that of the unmodified LSD algorithm.

This paper introduces a line-feature-matching approach that utilizes geometric constraints. Line segments are defined as lk={sk,ek}, with sk and ek indicating the beginning and end points. The reference line segment is denoted as li, the query line segment as lj, and the candidate matching pair as (li,). Parallelism among line segment pairs is captured through the angle θij formed by the pair:

$$\theta_{tf} = arc\tan\left(\left\|\vec{I}_t \times \vec{I}_f\right\| / \vec{I}_t \cdot \vec{I}_f\right), \vec{I}_k = \frac{s_k - e_k}{\|s_k - e_k\|_2} \quad (8)$$

Polar geometry constraints are described via the angle formed between the midpoint flow vector x_{ij}=m_i −m_j,=(s_k+e_k)/2 of each line segment and the axes in a 2D coordinate frame. For matching across frames, the relevant epipolar geometry constraint becomes:

$$\theta_{tf}^{f} = \arcsin\left(\|x_{tf} \times \eta_y\| / \|x_{tf}\|\right) \quad (9)$$

θ_{ij}^{s} and θ_{ij}^{f} denote the angles of the midpoint flow direction xij relative to the X and Y coordinate axes. ηx and ηy represent the direction vectors along the X and Y axes.

The method exploits the uniqueness property of tracking, in which one segment in the reference frame links to exactly one segment in the query frame. A matching vector ω is defined between the reference segment i and the query segment lj.

$$\omega_t = \left[\omega_{t0} \ \ldots, \omega_{tf} \ \ldots, \omega_{tn}\right]^T \quad (10)$$

Successful matching between i and lj sets the corresponding component ωi of ωij to 1; otherwise, it is set to 0.

A line segment pair (li,) qualifies as a successful match if the matching error and objective value equal βij and b.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

$$\beta_{ij} = \begin{bmatrix} \theta_{ij} \\ \theta_{ij}^{epip} \\ \rho_{ij} \\ \mu_{ij} \end{bmatrix}, \; b = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \quad (11)$$

Here, μij and ϱij indicate the length ratio and projection ratio between the two lines, θij is the angle between the paired segments, and $epip$ refers to the epipolar constraint from **Eqs. 8 and 9**.

$$A_i = [\beta_{i0}, \cdots, \beta_{ij}, \cdots, \beta_{in}] \quad (12)$$

The uniqueness of the tracking solution is assured only when the linear system $Ai\omega i=b$ holds, meaning the sum of all elements in the matching vector wi equals 1. Thus, the line-matching task becomes an optimization problem subject to these constraints. This work derives a sparse optimal solution using the $L1$ norm, as given in **Eq. 13**.

$$\min_{\omega_i} \lambda \|\omega_i\|_1 + \frac{1}{2} \|A_i \omega_i - b\|_1 \quad (13)$$

Upon solving **Eq. 13**, the resulting sparse vector ωi undergoes normalization. The largest element ωim in ωi identifies line segment lm in the query frame as the optimal correspondence for reference segment li.

Lastly, the random sampling consistency (RANSAC) [18] algorithm removes erroneous matches and unnecessary line features.

As illustrated in **Figure 8** [22], the modified LSD algorithm extracts distinct line features even in low-texture, dimly lit scenes with sparse details. While the total count of line features decreases somewhat, the matching duration drops from 70 ms to 20 ms **(Figure 9)**.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

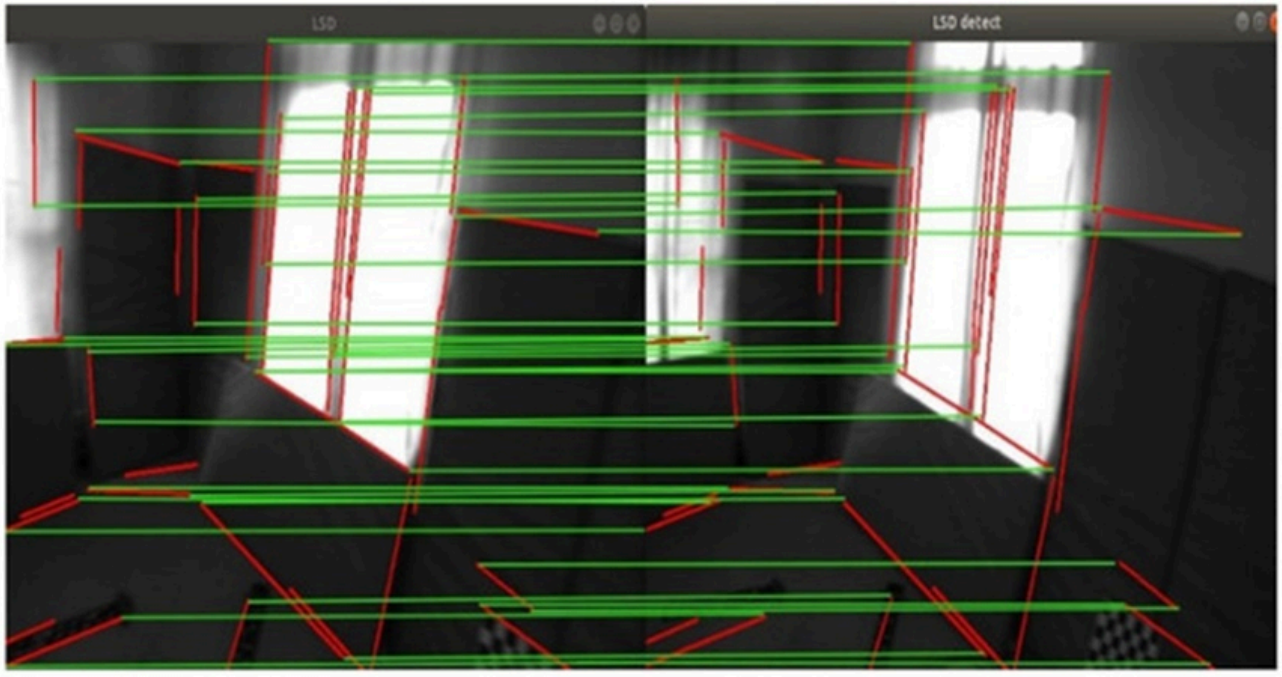


Figure 8. Line segment geometric constraint matching.

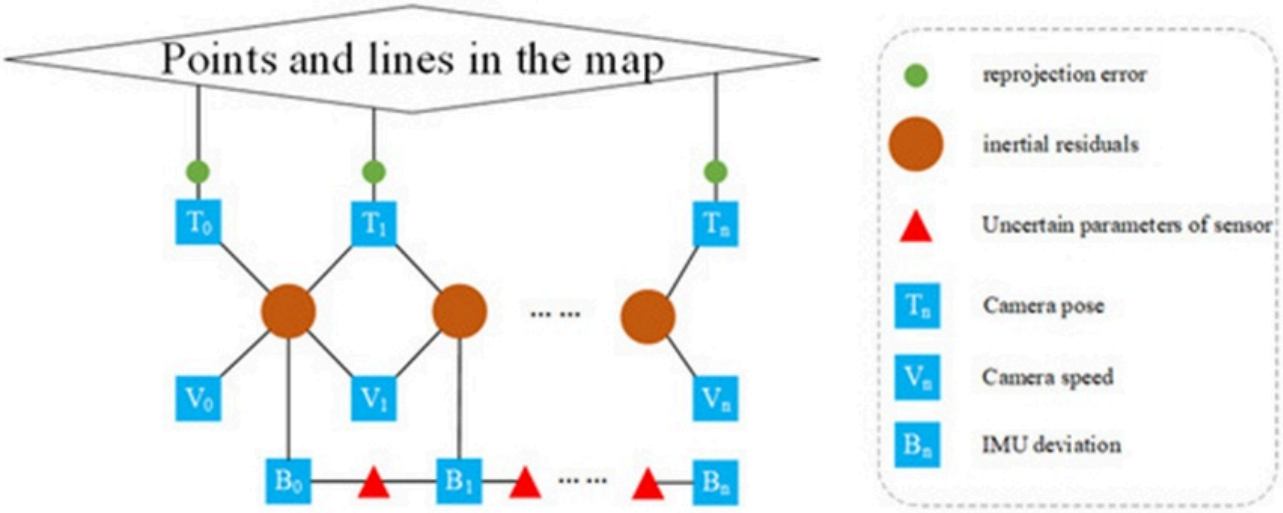


Figure 9. Visual-Inertial initialization.

Visual-inertia initialization

This part estimates visual-inertial parameters and biases via a progressive joint initialization procedure, drawing on the approach from ORB-SLAM3 [20]. Initially, visual-inertial Bundle Adjustment (BA) optimization generates a preliminary map containing several camera poses and sparse point clouds. Camera pose Rt remains fixed within this map and serves to compute IMU motion residuals in the subsequent phase.

In the IMU-only initialization phase, the objective is to derive optimal estimates of the inertial variables. Using maximum a posteriori estimation, only stationary camera paths and inertial quantities across keyframes are considered. These inertial quantities are incorporated into the IMU-only state vector $\mathbb{V}k$, detailed in **Eq. 14**:

© 2026 The Author(s). | Published by Agro Intelligence Research Press

$$\mathbb{V}_k = \{\mathcal{U}, g, \mathcal{B}, \bar{v}_{0:k}\} \quad (14)$$

$\mathcal{U}$ denotes the scale factor from the pure visual initialization phase. g represents the gravity vector, $\mathcal{B}$ stands for the constant accelerometer and gyroscope biases during initialization, and 0:k is the scaled camera velocity spanning the first to last keyframe, initially derived from the fixed camera path.

Focusing solely on inertial readings as described above:

$$\mathcal{X}_{0:k} \doteq \{\mathcal{X}_{0:1}, \mathcal{X}_{0:2}, \cdots, \mathcal{X}_{k-1:k}\} \quad (15)$$

Consequently, the maximized posterior is expressed in **Eq. 16**.

$$p(\mathbb{V}_k \mid \mathcal{X}_{0:k}) \propto p(\mathcal{X}_{0:k} \mid \mathbb{V}_k)\, p(\mathbb{V}_0) \quad (16)$$

Here, ($\mathcal{D}$0:k|$\mathbb{V}k$) indicates the likelihood, and ($\mathbb{V}$0) denotes the prior.

Leveraging Markov properties, where measurements at any instant rely solely on the current state, and applying standard measurement independence, **Eq. 16** simplifies to **Eq. 17**.

$$p(\mathbb{V}_0) \sqcap p\left(x_{tf} \mid s_t, s_f\right) \sqcap p(Z_t \mid s_t) \quad (17)$$

With measurement independence, the pure inertial maximum a posteriori estimation is given by **Eq. 18**.

$$\mathbb{V}'_k = \underbrace{\arg\max}_{\mathbb{V}_k} \left(p(\mathbb{V}_0) \prod_{t=1}^{k} p(\chi_{t-1:t} \mid \mu, g, \mathcal{B}, \bar{v}_{t-1}, \bar{v}_t) \right) \quad (18)$$

The estimate $\mathbb{V}'k$ maximizes **Eq. 17** or equivalently minimizes the negative log-posterior. Negative logarithms of IMU pre-integration terms and priors, combined with Gaussian errors, yield a sum-of-squared-residuals form. This leads to the final optimization in **Eq. 19**.

$$\mathbb{V}_k = \underbrace{\arg\min}_{v_k} \left(\gamma_p{}^2 + \sum_{t=1}^{k} \gamma_{\psi_{-1}{}^{2}\Sigma_{n-1,}} \right) \quad (19)$$

Unlike **Eq. 14**, the optimization in **Eq. 19** excludes visual residuals and instead employs the prior residual γp to drive IMU biases toward zero, with covariances from **Eq. 15** supplied by IMU data.

Post-inertial optimization, poses, velocities, and 3D points are adjusted to the recovered scale and aligned so the Z-axis matches the estimated gravity direction. Biases are refreshed, and IMU pre-integration is redone to minimize linearization inaccuracies. IMU parameters in **Eq. 15** emerge from combined visual and inertial constraints. Following initialization, fresh keypoints and IMU readings enable continued localization and mapping.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

Experimental Results and Their Analysis

Experiments were conducted on the DJI Manifold2-C platform with the following specifications: Intel® Core™ i7-8559U CPU running at 2.7–4.5 GHz, 8 GB of memory, no GPU, Ubuntu 16.04 LTS 64-bit OS with ROS Kinetic. Performance assessment used the EuRoC public dataset [23], featuring 11 sequences across easy, medium, and hard categories that account for lighting, speed, and flight variations to comprehensively test the method.

Figure 10 displays the per-frame average processing time for each component across the three algorithms on EuRoC data.

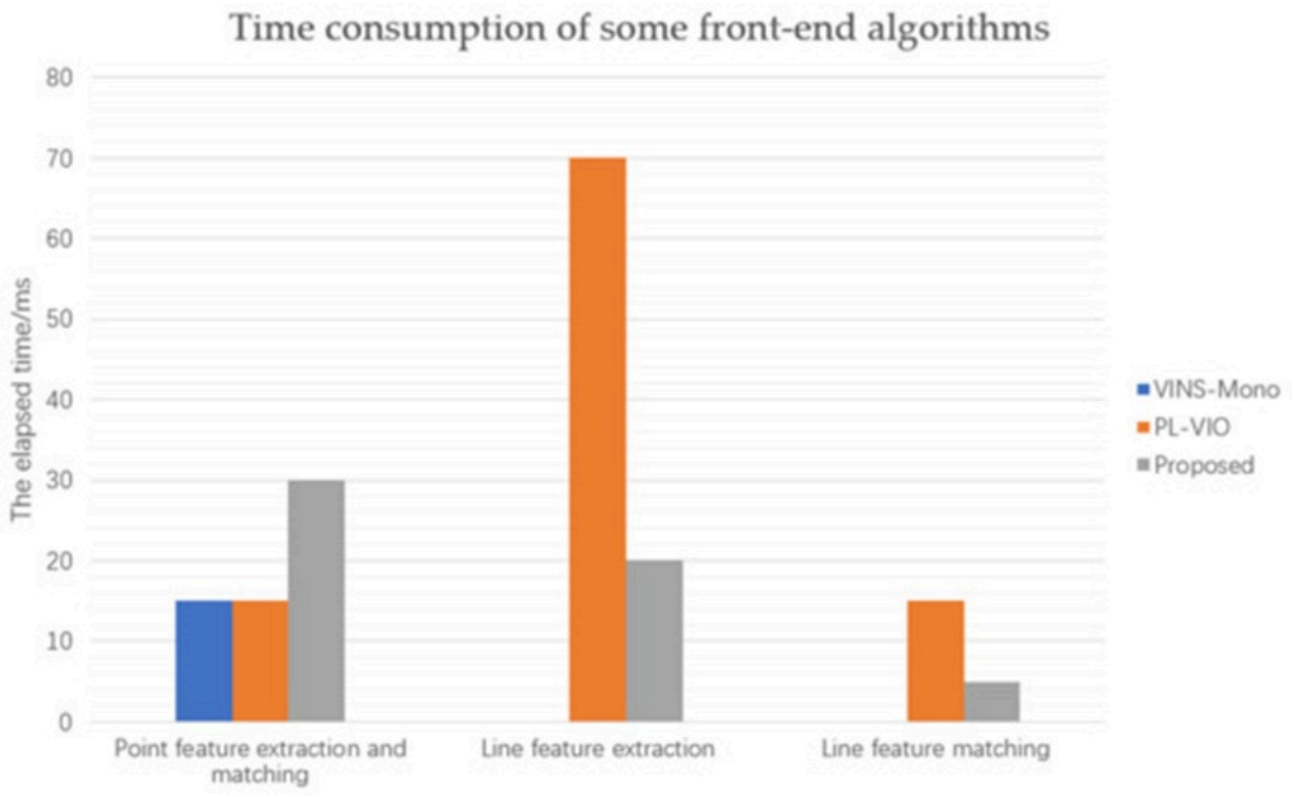


Figure 10. Consumption time of the front-end algorithm.

From **Figure 10**, SURF combined with FLANN for point features takes 30 ms, exerting minimal impact on real-time performance while delivering enhanced positioning precision and robustness. Line feature extraction in the proposed approach averages 20 ms per frame, with the combined point-line extraction and matching thread totaling 55 ms. This reduction in front-end runtime enables real-time execution of the full system.

Motion estimation errors for VINS-Mono, PL-VIO, and the current algorithm on EuRoC (across 10 subsequences) appear in **Table 1**. Absolute Trajectory Error (ATE) serves as the accuracy metric.

Table 1. Motion estimation errors of three algorithms in the EuRoC dataset (RMSE unit: m) and absolute trajectory error for both algorithms.

Serial	VINS-Mono	PL-VIO	Ours
MH_01_easy	0.355	0.129	0.145
MH_02_easy	0.343	0.183	0.150

© 2026 The Author(s). | Published by Agro Intelligence Research Press

MH_03_medium	0.558	0.262	0.218
MH_04_difficult	0.595	0.377	0.224
MH_05_difficult	0.587	0.283	0.257
V1_01_easy	0.359	0.166	0.085
V1_03_difficult	0.529	0.221	0.148
V2_01_easy	0.280	0.109	0.128
V2_02_medium	0.546	0.174	0.150
V2_03_difficult	0.572	0.319	0.183

Analysis of alignment and pose errors from data 3 in **Table 1** reveals that MH_01_easy and V2_01_easy sequences possess ample point-line features, good illumination, and moderate motion. The proposed method's strengths target low-texture and high-speed conditions, leading PL-VIO to outperform it slightly on those two sequences. The presented algorithm achieves the best results in all remaining cases. Superior accuracy is particularly evident in the texture-deficient MH_04_difficult sequence and the rapid-motion V1_03_difficult sequence.

Figure 11 depicts a scenario with abrupt lighting shifts, minimal-feature elements like windows and doorways, and intense motion—one of the most demanding environments that closely matches the targeted conditions and tests algorithmic limits.



Figure 11. V1_03_difficult data scenario [23].

To objectively assess the positioning performance of the three methods in this setting, the EVO toolkit was employed for visualization.

Figures 12-14 compare planar trajectories against ground truth for VINS-Mono, PL-VIO, and the proposed algorithm on this dataset.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

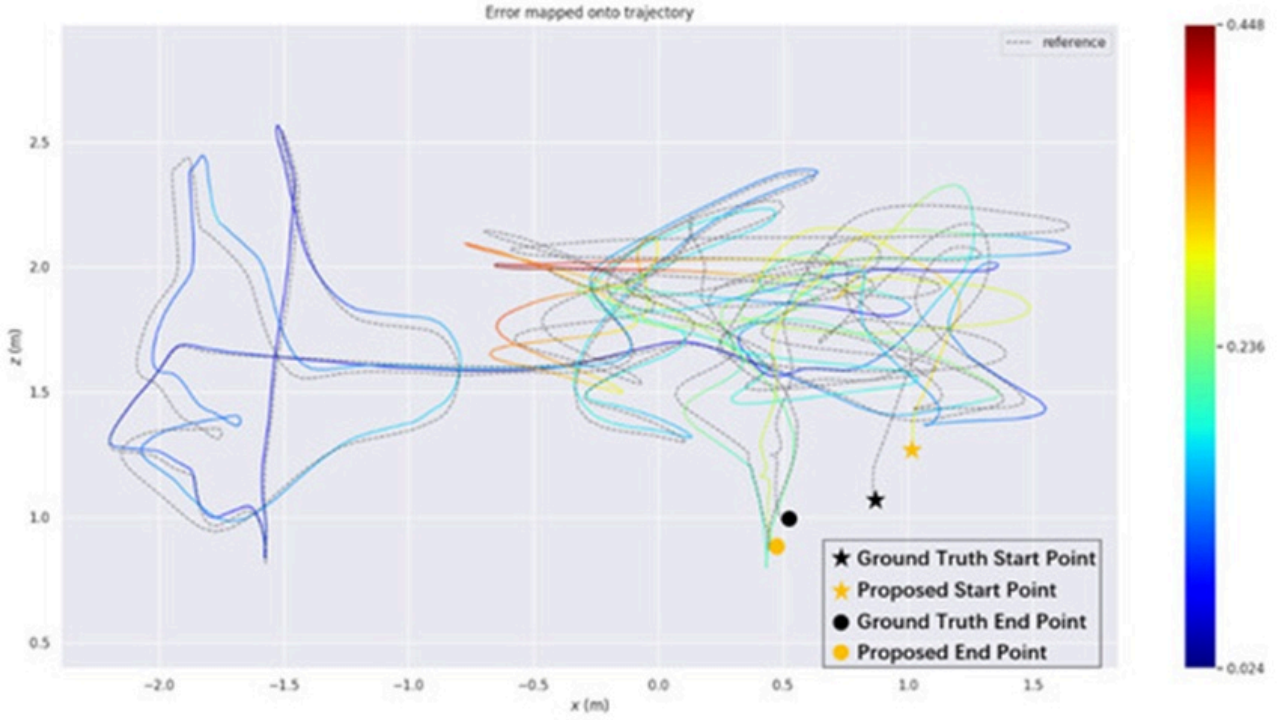


Figure 12. The VINS-Mono shows the absolute position error in the V1_03_difficult plane.

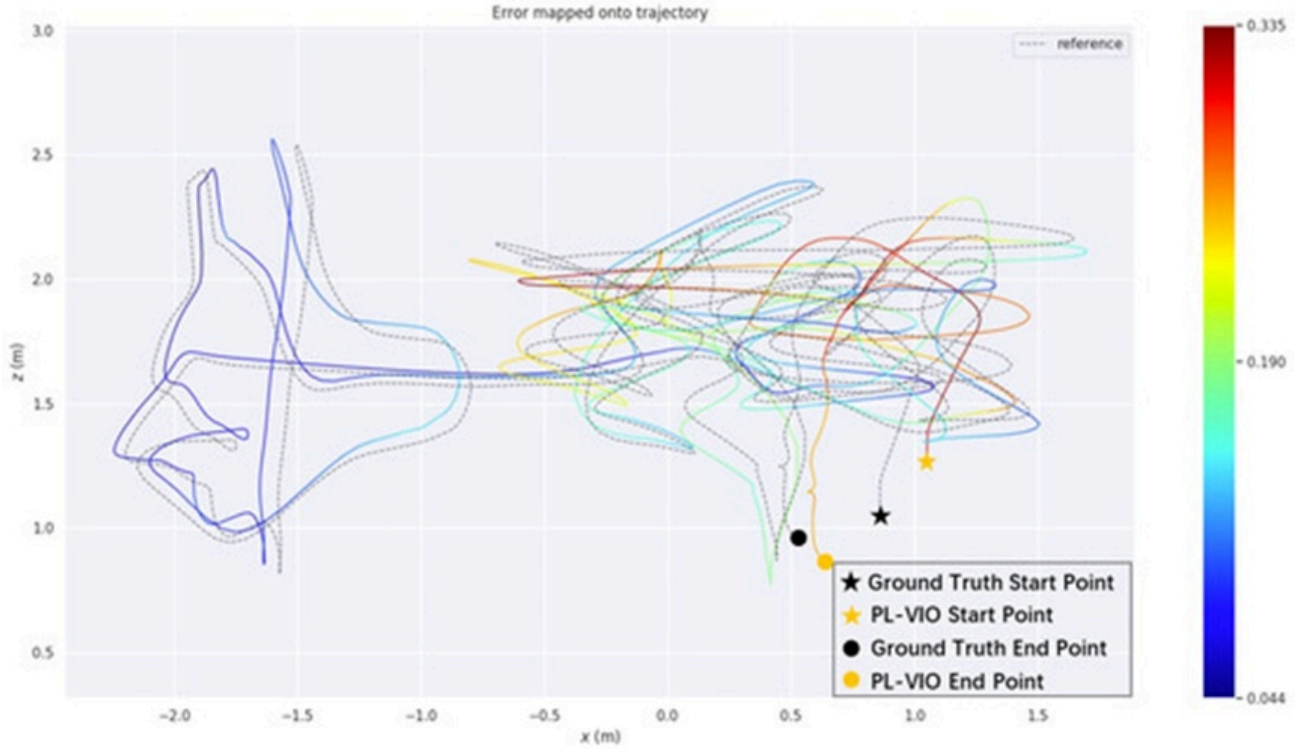


Figure 13. The P L-VIO shows the absolute position error in the V1_03_difficult plane.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

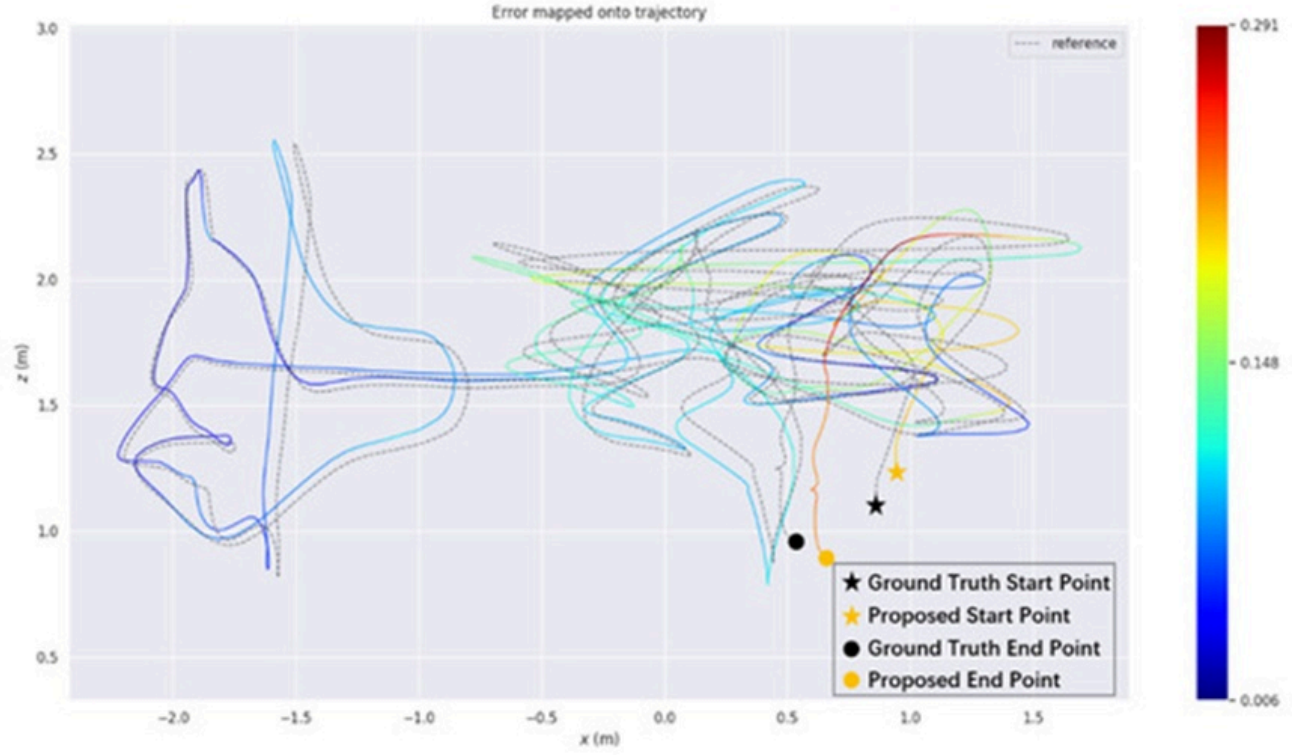


Figure 14. The proposed method shows the absolute pose error in the V1_03_difficult plane.

These figures illustrate Absolute Pose Error (APE) across the sequences. The color gradient from dark blue to red indicates the magnitude of the error, with the band extremes showing the maximum and minimum error values. Tracks nearer to dark blue exhibit smaller deviations from truth. The V1_03_difficult sequence involves indoor movement in a dimly lit environment. Comparisons of planar paths with reality for the proposed method and PL-VIO are shown in the associated figures. The resulting APE values are 0.236 m for VINS-Mono, 0.190 m for PL-VIO, and 0.148 m for the current algorithm, demonstrating closer alignment with ground truth and greater stability.

Real-time errors for the three algorithms on V1_03_difficult are plotted in **Figure 15**.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

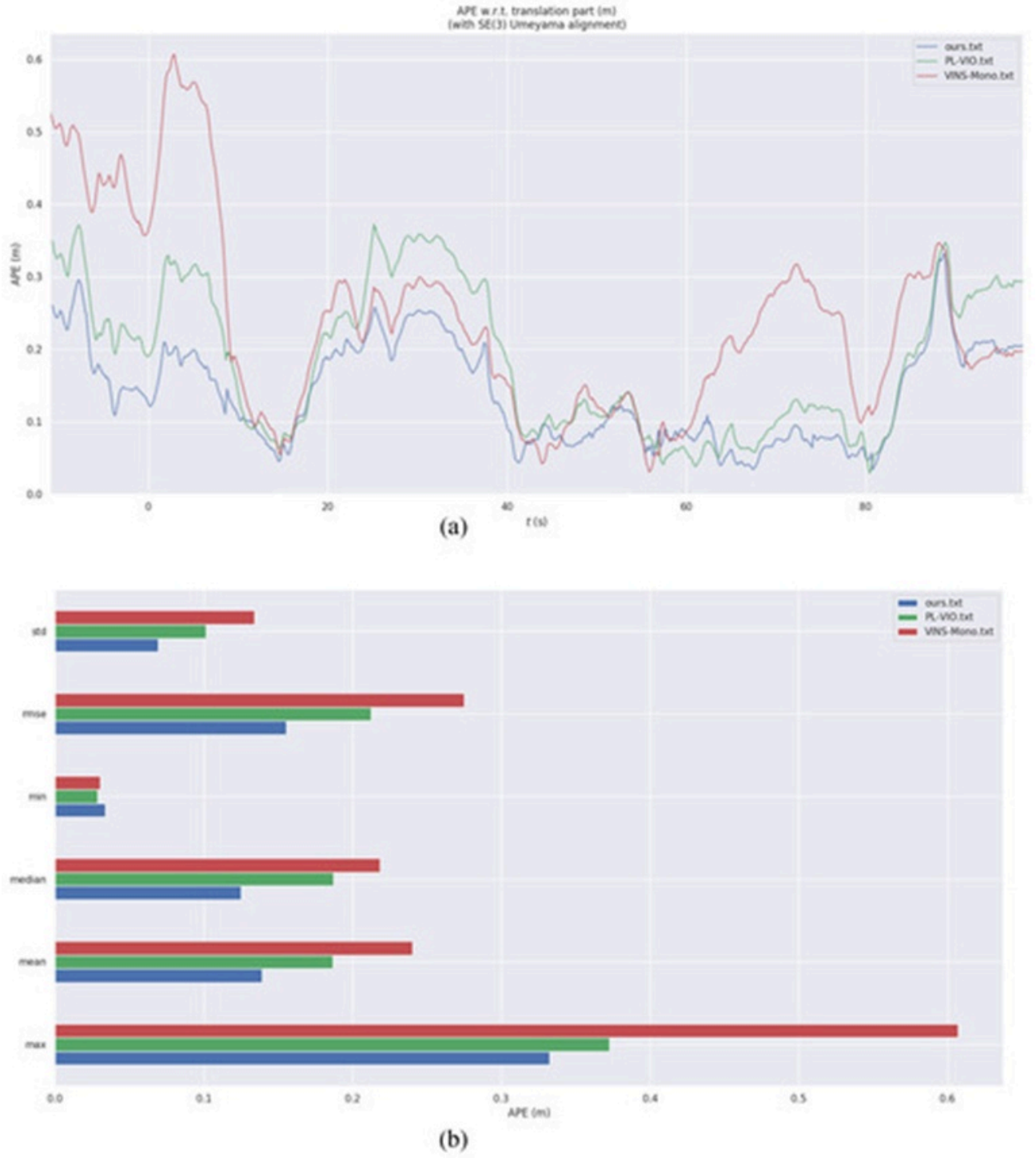


Figure 15. The real-time errors of the three algorithms in V1_03_difficult: (a) the real-time absolute pose error in V1_03_difficult, (b) shows the root mean square error, median error, mean error, extreme error, and sum of squares of error of the three algorithms.

Figure 15a shows smoother trajectories for the proposed algorithm than for the alternatives. **Figure 15b** summarizes root mean square, median, mean, maximum, and squared error sums, where the presented method excels across all metrics.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

Figure 16 indicates tighter error clustering, consistent behavior, and improved robustness for this algorithm. In summary, it surpasses the other two approaches overall.

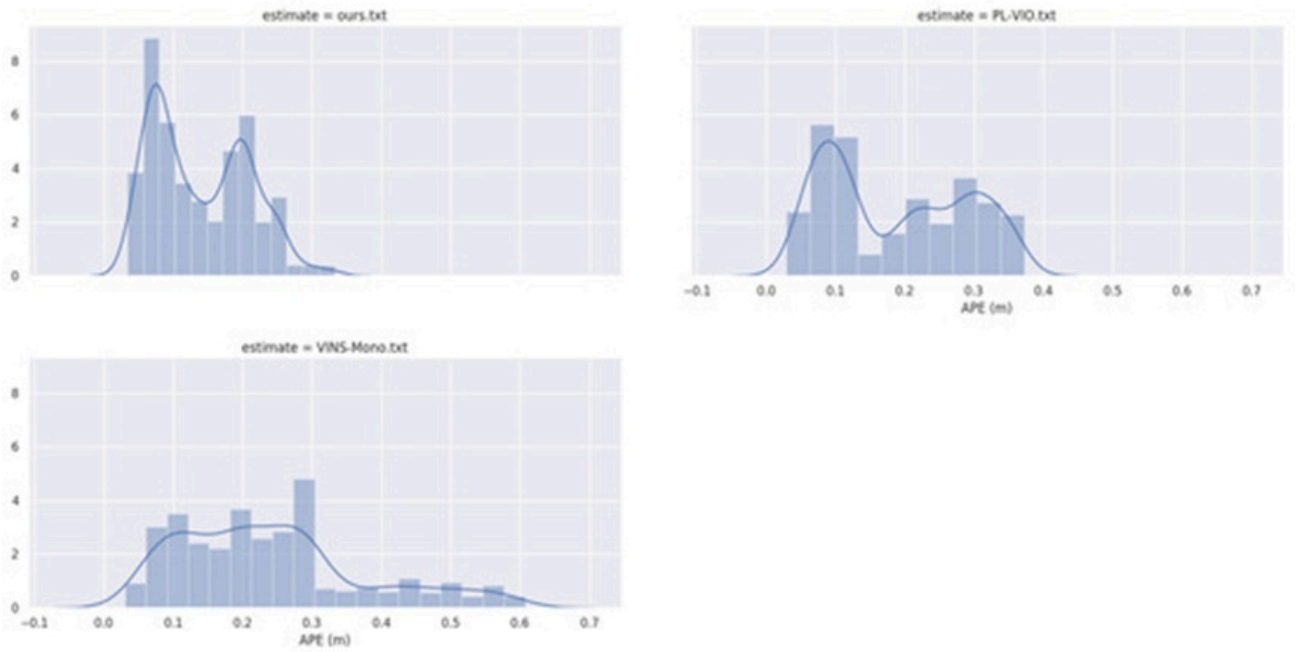


Figure 16. Range of absolute pose error distribution in V1_03_difficult.

Testing occurred in the underground parking garage at Northwest China University of Technology. The environment features open spaces bordered by white walls, which complicates point-feature detection. Variable lighting across locations and inconsistent UAV speeds add further challenges. A quadrotor UAV equipped with a computing unit and a D453i camera (30 Hz image rate, 200 Hz IMU rate) was used to evaluate real-world weak-texture behavior. All three algorithms ran online. The test area and hardware setup are displayed in **Figure 17**.



Figure 17. Experimental field and equipment.

The procedure involves launching the localization system, flying the UAV for a set duration with start and landing points coinciding, generating real-time trajectories and maps, and measuring final planar distances from the origin for each method.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

In **Figures 18a(1)–a(3)**, PL-VIO and the proposed algorithm detect more line features than VINS-Mono in expansive white-wall areas lacking sufficient points. From **Figures 18b(1) and 18b(2)**, after removing mismatches and excess lines, the current method retains fewer but higher-quality segments, thereby lowering the overall mismatch rate.

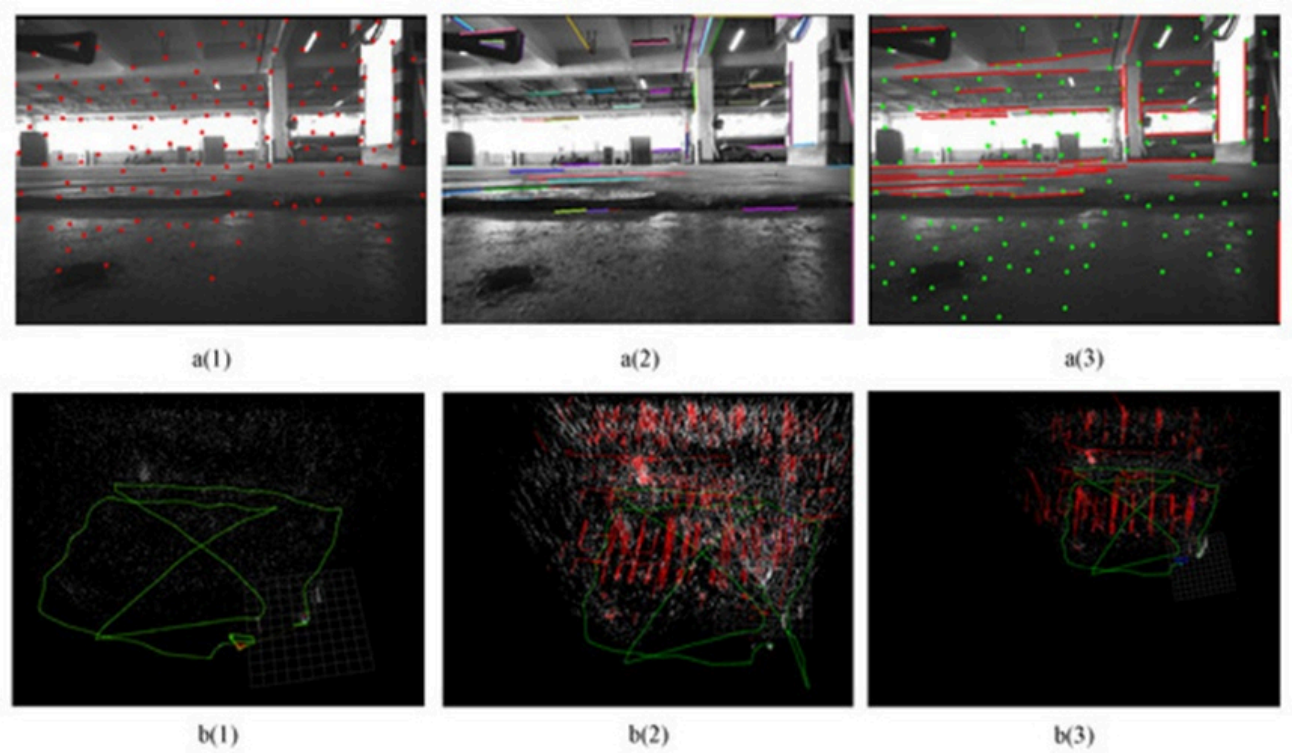


Figure 18. Actual scene experiment: a(1) VINS-Mono algorithm for feature extraction of images in real scenes, a(2) PL-VIO algorithm for feature extraction of images in real scenes, a(3) The proposed algorithm for feature extraction of images in real scenes, b(1) The real-time localization trajectory of the VINS-Mono algorithm is in the actual scene, b(2) The real-time localization trajectory of PL-VIO algorithm is in the actual scene, and b(3) The real-time localization trajectory of the proposed algorithm is in the actual scene.

Actual flight paths for the three algorithms appear in **Figure 19**.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

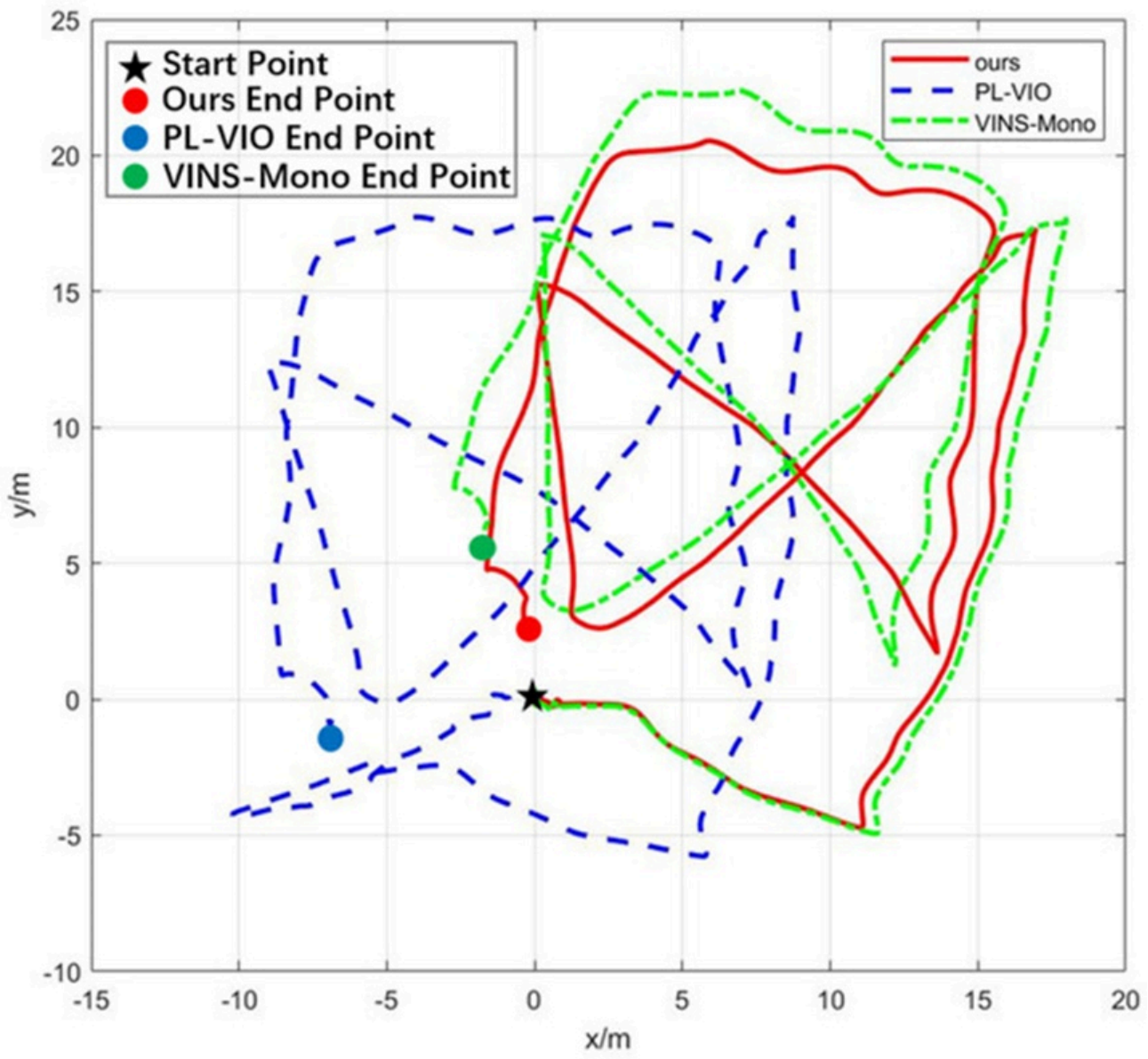


Figure 19. Plar trajectories of the three algorithms in the actual scenario.

Final distances to origin measured 2.902 m for the proposed algorithm, 6.99 m for PL-VIO, and 5.963 m for VINS-Mono. The new method thus achieves smaller practical errors, improving accuracy by 51.33% over VINS-Mono and 58% over PL-VIO.

Figure 19 shows PL-VIO trajectories drifting shortly after takeoff, as line-feature SLAM struggles to complete initialization quickly. Stability returns later with accumulating features. By refining image handling and initialization, the proposed algorithm rapidly computes accurate parameters, yielding stable UAV paths and high precision from the start. While VINS-Mono avoids divergence, its accuracy lags behind the current approach.

Figures 20 and 21 capture post-initialization pose variations for the three systems.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

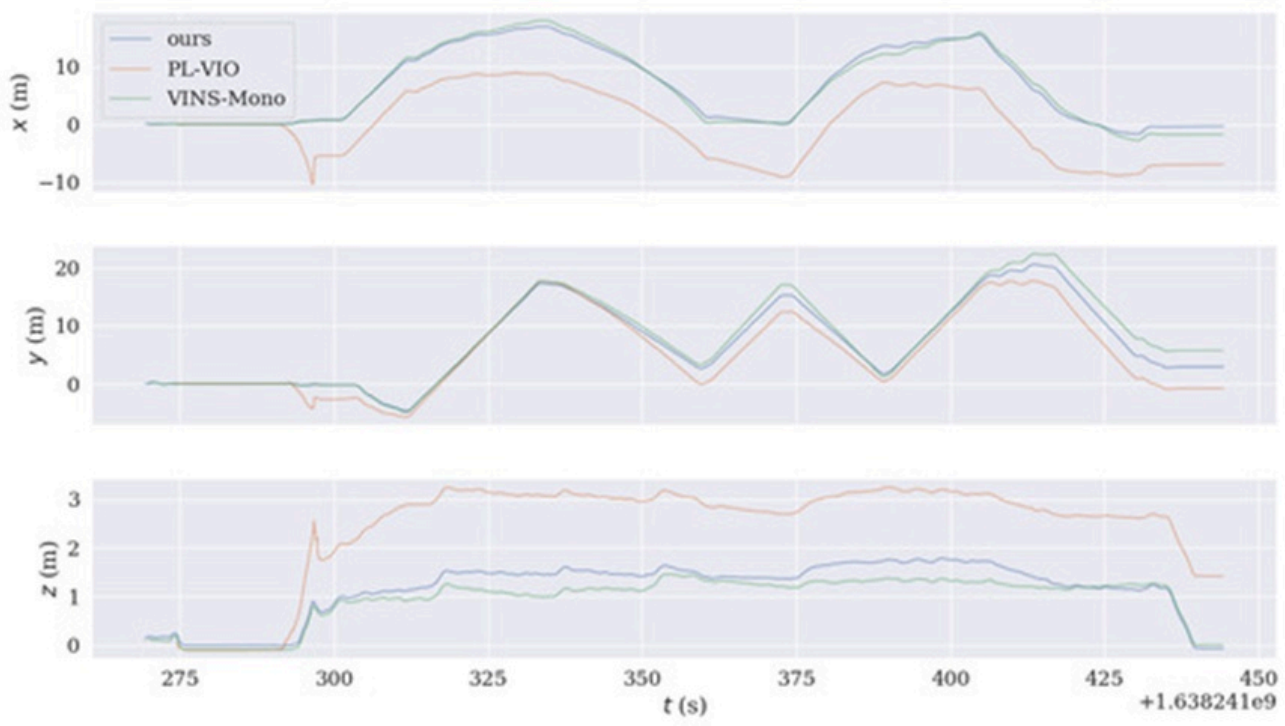


Figure 20. The displacement of the three algorithms on the X, Y, and Z axes, respectively.

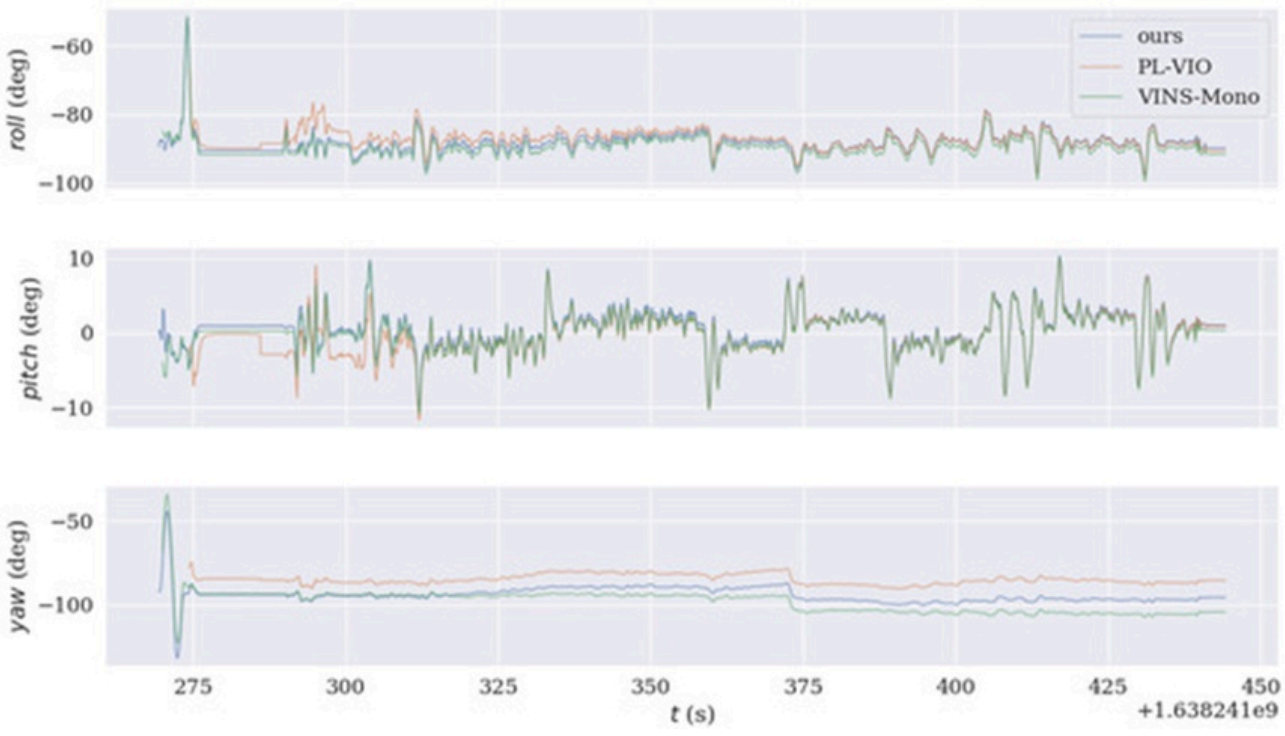


Figure 21. Attitude angle changes for three algorithms.

Comparisons confirm shorter initialization time and smoother motion for the proposed algorithm compared to the baselines.

© 2026 The Author(s). | Published by Agro Intelligence Research Press

Experimental evaluations demonstrate that the new method effectively addresses pose estimation in texture-poor areas, delivering superior initialization, feature processing, and matching. It exhibits top performance in real environments. Integrating line features with IMU data markedly boosts overall accuracy and reliability.

Conclusion

This paper proposes an innovative point-line visual-inertial SLAM technique. Validation relies on the EuRoC public dataset and actual UAV flights. Key findings include:

(1) Incorporating bilateral filtering into the SLAM front-end preprocessing stage effectively suppresses image noise and eases point-line feature detection.

(2) To handle camera-motion blur, SURF detection paired with FLANN matching is utilized. Although point-feature processing time increases modestly, localization precision improves substantially.

(3) Careful parameter tuning of the LSD line extraction algorithm, combined with a length suppression approach, triples extraction speed. A geometric constraint-based line-matching technique is introduced, reducing line segment matching time by 66.66%.

(4) Stepwise joint initialization significantly shortens startup time, raises accuracy, and produces smoother positioning trajectories.

Acknowledgements

The authors wish to express their appreciation to the National Natural Science Foundation of China (NSFC) (61603297) and the Natural Science Foundation of Shaanxi Province (2020JQ-219).

Conflict of interest

None

Financial support

This research was funded by the National Natural Science Foundation of China (NSFC) (61603297) and the Natural Science Foundation of Shaanxi Province (2020JQ-219).

Ethics statement

None

Received: 14 Sep 2024 Revised: 19 Nov 2024 Accepted: 05 Dec 2024
Published online: 25 November 2026

Rights and permissions

© 2026 The Author(s). | Published by Agro Intelligence Research Press

Open Access The author(s) retain copyright. This article is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. It may be shared and adapted for non-commercial purposes with appropriate attribution, an indication of changes, and distribution of adaptations under the same license. Third-party material may be subject to separate terms identified in its credit line. View the license at https://creativecommons.org/licenses/by-nc-sa/4.0/.

References

1. Latif Y, Doan AD, Chin TJ, Reid I. Sprint: Subgraph place recognition for intelligent transportation. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA); Paris, France; 2020. p. 5408–14.

2. Bryson M, Sukkarieh S. Building a robust implementation of bearing-only inertial SLAM for a UAV. J Field Robot. 2010;24:113–43.

3. Geneva P, Eckenhoff K, Huang G. A linear-complexity EKF for visual-inertial navigation with loop closures. In: Proceedings of the International Conference on Robotics and Automation (ICRA); Montreal, QC, Canada; 2019.

4. Qin T, Li P, Shen S. VINS-Mono: A robust and versatile monocular visual-inertial state estimator. IEEE Trans Robot. 2018;34:1004–20.

5. Mur-Artal R, Tardós JD. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Trans Robot. 2016;33:1255–62.

6. Burdziakowski P. Towards precise visual navigation and direct georeferencing for MAV using ORB-SLAM2. In: Proceedings of the Baltic Geodetic Congress (BGC Geomatics); Gdansk, Poland; 2017. p. 394–8.

7. Aguilar WG, Rodríguez GA, Álvarez L, Sandoval S, Quisaguano FJ, Limaico A. Visual SLAM with a RGB-D camera on a quadrotor UAV using on-board processing. In: International Work-Conference on Artificial Neural Networks; Cham, Switzerland: Springer; 2017. p. 596–606.

8. Lowe D. Distinctive image features from scale-invariant key points. Int J Comput Vis. 2003;20:91–110.

9. Rosten E, Drummond T. Machine learning for high-speed corner detection. In: Proceedings of the European Conference on Computer Vision; Graz, Austria; Springer: Berlin/Heidelberg, Germany; 2006. p. 430–43.

10. Rublee E, Rabaud V, Konolige K, Bradski G. ORB: An efficient alternative to SIFT or SURF. In: Proceedings of the International Conference on Computer Vision; Barcelona, Spain; 2011. p. 2564–71.

11. Vakhitov A, Funke J, Moreno-Noguer F. Accurate and linear time pose estimation from points and lines. In: Proceedings of the European Conference on Computer Vision; Graz, Austria; Springer: Cham, Switzerland; 2016. p. 583–99.

12. Zuo X, Xie X, Liu Y, Huang G. Robust visual SLAM with point and line features. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); Vancouver, BC, Canada; 2017. p. 1775–82.

13. Pumarola A, Vakhitov A, Agudo A, Sanfeliu A, Moreno-Noguer F. PL-SLAM: Real-time monocular visual SLAM with points and lines. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA); Singapore; 2017. p. 4503–8.

14. Gomez-Ojeda R, Moreno FA, Zuniga-Noël D, Scaramuzza D, Gonzalez-Jimenez J. PL-SLAM: A stereo SLAM system through the combination of points and line segments. IEEE Trans Robot. 2019;35:734–46.

15. 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.

16. Wen H, Tian J, Li D. PLS-VIO: Stereo vision-inertial odometry based on point and line features. In: Proceedings of the IEEE International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS); Vientiane, Laos; 2020. p. 1–7.

17. Fu Q, Wang J, Yu H, Ali I, Guo F, He Y, et al. PL-VINS: Real-time monocular visual-inertial SLAM with point and line. arXiv. 2020;arXiv:2009.07462.

18. Tomasi C, Manduchi R. Bilateral filtering for gray and color images. In: Proceedings of the IEEE Sixth International Conference on Computer Vision; Bombay, India; 1998. p. 839–46.

19. Von Gioi RG, Jakubowicz J, Morel JM, Randall G. LSD: A fast line segment detector with a false detection control. IEEE Trans Pattern Anal Mach Intell. 2008;32:722–32.

20. Muja M, Lowe DG. Fast approximate nearest neighbors with automatic algorithm configuration. In: Proceedings of the International Conference on Computer Vision Theory and Application Vissapp; Lisboa, Portugal; 2009;2:331–40.

21. Campos C, Elvira R, Rodríguez JJG, Montiel JMM, Tardós JD. ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM. IEEE Trans Robot. 2021;37:1874–90.

22. Chaudhury KN, Sage D, Unser M. Fast O (1) bilateral filtering using trigonometric range kernels. IEEE Trans Image Process. 2011;20:3376.

23. Burri M, Nikolic J, Gohl P, Schneider T, Rehder J, Omari S, et al. The EuRoC micro aerial vehicle datasets. Int J Robot Res. 2016;35:1157–63.

© 2026 The Author(s). | Published by Agro Intelligence Research Press