ISSN 2075-1087, Gyroscopy and Navigation, 2023, Vol. 14, No. 4, pp. 339–352. © Pleiades Publishing, Ltd., 2023.

Real-Time Visual-Inertial Odometry Based on Point-Line Feature Fusion

G. Yang[a], W. D. Meng[a, *], G. D. Hou[a], and N. N. Feng[a]

[a] *Xi'an University of Posts and Telecommunications, Xi'an, 710121 Republic of China*

**e-mail: mengwd1999@163.com*

Received July 19, 2023; revised November 23, 2023; accepted December 20, 2023

Abstract—To improve the localization accuracy and tracking robustness of monocular feature-based visual SLAM systems in low-texture environments, a visual-inertial odometry method combining line features and point features is proposed, taking advantage of the easy availability of line features in real-world environments and the high accuracy of feature-based methods. The combination of point and line features ensures accurate positioning of the SLAM system in low-texture environments, while the inclusion of IMU data provides prior information and scale information. The pose is optimized by minimizing the reprojection error of point and line features and the IMU error using bundle adjustment. An improved EDlines algorithm is introduced, which incorporates a pixel chain length suppression process to enhance the effectiveness of extracted line features and reduce the rate of line feature misalignment. Experimental results on the public EuRoC dataset and TUM RGB-D dataset show that the proposed method meets the real-time requirements and has higher localization accuracy and robustness compared with the visual SLAM method based on single point feature or the method adding traditional line features.

Keywords: simultaneous localization and mapping, visual-inertial odometry, point-line features, nonlinear optimization

DOI: 10.1134/S2075108724700068

1. INTRODUCTION

With the long-term development of Simultaneous Localization and Mapping (SLAM) and the advancement of modern computer vision techniques, SLAM technology has gradually been applied in various fields such as unmanned aerial vehicles, autonomous driving, domestic service robots, augmented reality, and virtual reality. Visual SLAM (VSLAM), based on vision, utilizes cameras to construct maps of the environment and perform localization, offering advantages such as real-time capability and rich information. However, compared to mainstream SLAM methods that primarily rely on lidar as the main sensor, visual SLAM suffers from issues of poor robustness and insufficient accuracy. Therefore, sensor fusion approaches incorporating wheel odometry, satellite navigation systems, and inertial measurement units (IMU) into visual SLAM systems have become a research focus. The combination of vision and IMU, as a low-cost solution, holds great potential for development. Cameras provide abundant environmental information, enabling the creation of 3D models and the recognition of self-position [1]. In addition, IMU provide self-motion information to help recover scale in monocular camera setups and obtain absolute pose estimation. The fusion of vision and IMU can be categorized into loosely coupled and tightly coupled approaches, depending on whether the data from both sensors are directly used for pose estimation. Tightly coupled methods, which establish constraint models using data from both sensors for overall pose estimation, are more commonly used due to their higher robustness. The fusion of vision and IMU can also be classified into filter-based and optimization-based approaches [2]. Filter-based methods, with the Extended Kalman Filter as a typical example, are more suitable for situations with limited computational resources and simple state variables. Optimization-based methods offer higher accuracy but require more computational resources. The Multi-State Constraint Kalman Filter (MSCKF) proposed by the Google team integrates visual states into the system state, achieving visual and IMU tightly coupled estimation within the framework of the Kalman filter [3]. Building upon ORB-SLAM [4], authors of ORB-SLAM incorporate IMU pre-integration and visual constraints to construct a graph optimization framework, reducing localization errors using the idea of covisibility [5]. VINS-Mono achieves better localization performance than other visual-inertial SLAM methods by employing loosely coupled initialization and tightly coupled back-end nonlinear optimization [6]. However, its front end, based on optical flow, is sensitive to environmental variations and lacks robustness.

Visual-inertial odometry demonstrates significant improvements in accuracy and robustness compared to single-sensor visual odometry. The methods for recovering camera motion based on feature forms in visual odometry can be divided into feature-based and direct methods. Direct methods estimate camera motion based on image intensity changes, making them highly sensitive to lighting variations and prone to tracking failures during fast camera motion. On the other hand, feature-based methods are more robust to lighting variations and fast motion but suffer from poor feature extraction performance in low-texture scenarios, which may result in the loss of tracking and difficulties in motion estimation.

In real-world scenarios, indoor environments with weak textures typically include objects such as ceilings, beams, appliances, and furniture, which exhibit linear features. These linear features are less sensitive to lighting changes and possess rotational invariance, making them suitable for providing compensatory constraints in weak texture environments. Scholars both domestically and internationally have conducted extensive research in this area. The authors of PL-SLAM implemented a real-time monocular SLAM system that combines point and line features based on ORB-SLAM [7]. Initialization can be achieved by detecting line features in only three consecutive frames. However, the real-time performance of PL-SLAM is poor due to the use of LSD [8] for line feature extraction, and its accuracy is not high. PL-SVO [9] incorporates line feature processing into the framework of SVO, where the proposed odometry eliminates the need for continuously extracting and matching features in subsequent frames and allows for rapid line segment tracking. PL-VIO [10] improves the accuracy and robustness of the system by adding line features to sliding window optimization based on VINS-Mono. Similarly, PL-VINS integrates line features into VINS-Mono. The authors made improvements to the LSD algorithm by adjusting hidden parameters and introducing length suppression, resulting in a threefold increase in speed compared to the original LSD algorithm [11]. Structure PLP-SLAM [12] utilizes both point and line features for self-localization and incorporates a Plane Segmentation and Reconstruction module to construct a structured map with semantic features. To process the variable line length, the neural network can enhance the line descriptor by understanding the geometry of the line, and use the line feature to improve the visual positioning effect of the point feature [13]. Manhattan-SLAM [14] proposed a new method for robust tracking in both Manhattan world scenes and non-Manhattan world scenes and uses point, line, and planar features to achieve pose estimation in the above two scenes. LGNET [15] model employs neural networks to augment initial features and integrate structural features to preserve intricate details. The integration of semantic SLAM or the incorporation of neural network models into SLAM systems necessitates platforms with sufficient computational capabilities, such as those equipped with on-board processing units or coupled with graphics processing units (GPU). These systems hold greater promise following advancements in lightweight computing technologies.

To solve the problem that traditional point feature methods are easy to track failure and cannot restore camera motion in a low-texture environment, we propose a method in this paper. The system optimizes the pose by binding adjustment to minimize IMU pre-integrating constraints and point-line re-projection errors in successive frames. Compared with the traditional method using only a single point feature, we add line features to improve the robustness and accuracy in light changes or weak texture environments. The main contributions of this paper are as follows: (1) The EDlines [16] line segment extraction algorithm is improved to provide the system with effective and significant line feature constraints, and the EDlines algorithm with length suppression is added to reduce the extraction of short line segments and line segment matching, improve the success rate of line segment matching, and make it meet the real-time requirements of the tracking process. (2) The point-line re-projection error is constructed and added to the nonlinear optimization framework of bundle adjustment (BA) with IMU pre-integrating constraints, effectively integrating information from visual and inertial sensors.

In this paper, we first verify the parameters and performance of the improved EDlines algorithm, and then compare the proposed method with two current advanced monocular VIO methods: VI ORB-SLAM and PL-VIO on the EuRoC dataset, and compare it with PL-SLAM and ORB-SLAM on the TUM RGB-D dataset. The detailed evaluation results are given in the following.

2. FUSION OF VISUAL-INERTIAL ODOMETRY AND LINE FEATURE

In this paper, the VI-SLAM system tracks the relative motion between devices, their position, and orientation in the Earth's inertial coordinate system $(\cdot)^E$ remain undetermined. Therefore, the coordinate system of the first keyframe determined by the V-SLAM system is typically adopted as the world coordinate system $(\cdot)^W$ (Fig. 1), where the direction of the gravity acceleration aligns with the world coordinate system's z-axis. $(\cdot)^B$ is the carrier coordinate system defined to be the same as the IMU coordinate system. And camera coordinate system is defined as $(\cdot)^C$. Rotations are represented using rotation matrices R or quaternions q. A point in the world coordinate system or the real 3D world is denoted as $\mathbb{R}^3$, while a point on the

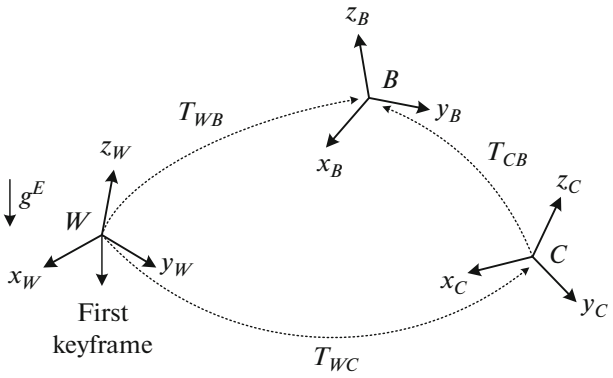

Fig. 1. Relationships among the world $(\cdot)^W$, camera $(\cdot)^C$, and IMU body $(\cdot)^B$ reference frames. The g_E and g_W are respectively the vectors of gravitational acceleration in the earth's inertial frame $(\cdot)^E$ and in the world frame $(\cdot)^W$.

image plane is denoted as $\mathbb{R}^2$. The gravity acceleration vector in the world coordinate system is represented as $g^W = [0\ \ 0\ \ g]^T$ and its direction aligns with the coordinate system's z-axis. If a vector describes the relative transformation from one reference frame to another frame, a right subscript is appended to indicate the transformed frame, such as P_{CB} for the vector that defines the translation from the camera frame to the IMU body frame. For the rotation matrix, the orientation from the camera frame to the IMU body frame is denoted as R_{CB}. The camera and IMU are rigidly attached to a common bracket. The transformation $T_{CB} = [R_{CB}|P_{CB}](4\times 4)$ between the camera frame $(\cdot)^C$ and the IMU body frame $(\cdot)^B$ is usually unknown and has to be calibrated.

2.1. Algorithm Framework

The algorithm framework of this paper, as shown in Fig. 2, consists mainly of tracking and optimization parts. The monocular camera captures images and IMU data, which are input to the tracking thread. ORB feature points and improved EDlines feature lines are extracted for each frame. The feature lines extracted by the improved EDlines algorithm are matched using LBD descriptors. Simultaneously, IMU pre-integration [17, 18] provides the current frame's pose and velocity, completing the initial pose estimation. At this point, image feature points and feature lines are reprojected to build a local map. The tracking thread decides on new keyframes and inputs them to the optimization thread. The keyframes are inserted and matched with the local map to construct landmarks. By combining the observed positions of the landmarks with prior information, local bundle adjustment optimization is performed on the point-line reprojection error and IMU pre-integration error to refine the pose and remove outlier features. This completes the optimization thread.

2.2. IMU Pre-Integration

The inertial measurement unit usually includes a 3-axis gyroscope and a 3-axis accelerometer, which can measure the rotation Angle value and acceleration value of the sensor for the inertial coordinate system. The measurement model is as follows:

$$\tilde{\omega}_{WB}^B = \omega_{WB}^B + b^g + \eta^g, \tag{1}$$

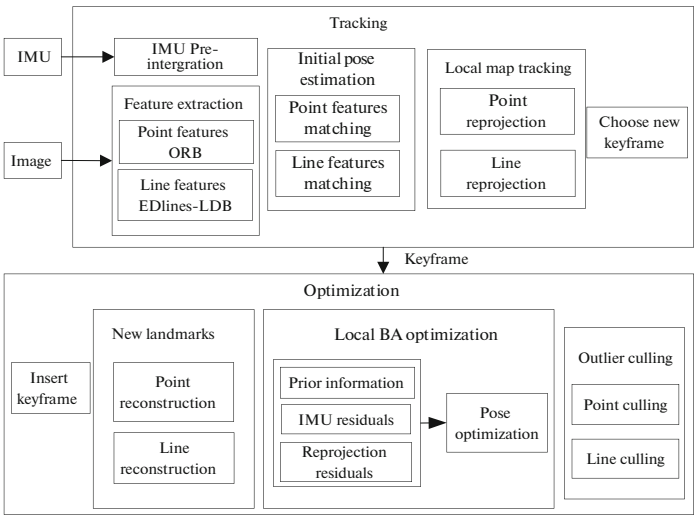

Fig. 2. Flowchart of nonlinear optimized visual-inertial odometry based on point line feature.

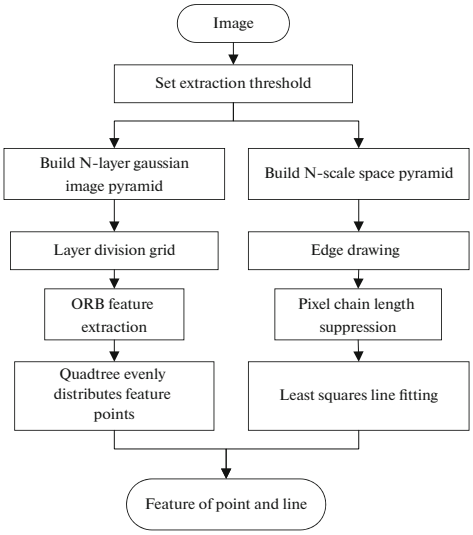

Fig. 3. Flowchart of point and line feature extraction.

$$\tilde{a}^B = R_{WB}\left(a^W - g^W\right) + b^a + \eta^a, \tag{2}$$

where $\tilde{\omega}_{WB}^B$ is the observed value of angular velocity, $\tilde{a}^B$ is the observed value of acceleration, b^g is the gyroscope angular velocity zero bias, b^a is the acceleration zero bias, η^g and η^a are white sensor noise, g^W is the acceleration of gravity in the world coordinate system. The concept of IMU pre-integral was first proposed in Article [17], which is a method of operation in Euclidean space. The author of Article [18] extended the method on manifold space based on Article [17] and obtained the relative motion increment independent of the initial state of the system. Given two consecutive keyframes of i and j at a given time, the orientation R_{WB}, position P_{WB}, and velocity V_{WB} of the corresponding IMU measurement can be calculated using the following formula to describe the multiple measurement values in the period:

$$R_{WBj} = R_{WBi}\prod_{k=i}^{j-1}\mathrm{Exp}\left(\left(\omega_k^B - b_k^g - \eta_k^g\right)\Delta t\right), \tag{3a}$$

$$\begin{aligned} P_{WBj} = P_{WBi} + \sum_{k=i}^{j-1}\Bigg(V_{WBk}\Delta t + \frac{1}{2}g^W\Delta t^2 \\ + \frac{1}{2}R_{WBk}\left(a_k^B - b_k^g - \eta_k^g\right)\Delta t^2\Bigg), \end{aligned} \tag{3b}$$

$$V_{WBj} = V_{WBi} + g^W\Delta t_{ij} + \sum_{k=i}^{j-1}R_{WBk}\left(a_k^B - b_k^g - \eta_k^g\right)\Delta t. \tag{3c}$$

Where Δt is the IMU data sampling interval and $\Delta t_{ij} = (j-i)\Delta t$. The $\mathrm{Exp}(\cdot)$ is the exponential map operator that maps Lie algebra $so(3)$ to the Lie group. When the influence of IMU measurement noise is ignored, concerning the zero bias $\bar{b}_i^{(\cdot)}$ at time i, the zero bias increment at time j is $\delta b_i^{(\cdot)}$. The expressions in (3) can be rewritten as:

$$R_{WBj} = R_{WBi}\Delta\bar{R}_{ij}\mathrm{Exp}\left(J_{\Delta\bar{R}_{ij}}^g\delta b_i^g\right), \tag{4a}$$

$$\begin{aligned} P_{WBj} = P_{WBi} + V_{WBi}\Delta t_{ij} + \frac{1}{2}g^W\Delta t_{ij}^2 \\ + R_{WBi}\left(\Delta\bar{P}_{ij} + J_{\Delta\bar{P}_{ij}}^g\delta b_i^g + J_{\Delta\bar{P}_{ij}}^a\delta b_i^a\right), \end{aligned} \tag{4b}$$

$$\begin{aligned} V_{WBj} = V_{WBi} + g^W\Delta t_{ij} \\ + R_{WBi}\left(\Delta\bar{V}_{ij} + J_{\Delta\bar{V}_{ij}}^g\delta b_i^g + J_{\Delta\bar{V}_{ij}}^a\delta b_i^a\right), \end{aligned} \tag{4c}$$

where the Jacobians $J_{(\cdot)}^g$ and $J_{(\cdot)}^a$ describe how the measurements change due to a change in the bias estimation [18]. The biases $\bar{b}_i^g$, $\bar{b}_i^a$ remain constant during the pre-integration period. The pre-integration terms $\Delta\bar{R}_{ij}$, $\Delta\bar{P}_{ij}$, $\Delta\bar{V}_{ij}$ are independent of the state and gravity at time i, and they can be computed directly from the IMU output between two keyframes, as follows:

$$\Delta\bar{R}_{ij} = \prod_{k=i}^{j-1}\mathrm{Exp}\left(\left(\omega_k^B - \bar{b}_i^g\right)\Delta t\right), \tag{5a}$$

$$\Delta\bar{P}_{ij} = \sum_{k=i}^{j-1}\left(\Delta\bar{V}_{ik}\Delta t + \frac{1}{2}\Delta\bar{R}_{ik}\left(a_k^B - \bar{b}_i^a\right)\Delta t^2\right), \tag{5b}$$

$$\Delta\bar{V}_{ij} = \sum_{k=i}^{j-1}\Delta\bar{R}_{ik}\left(a_k^B - \bar{b}_i^a\right)\Delta t. \tag{5c}$$

2.3. Point and Line Feature Extraction

ORB features are commonly used for extracting point features, while in visual SLAM, the LSD algorithm is often employed for detecting and extracting line features. The algorithm relies on the gradients of each pixel and performs pixel merging. The dependence on gradients and pixel merging requires significant computational resources, which affects the real-time performance of visual SLAM. To enable fast extraction of line features, this paper utilizes the more efficient EDlines algorithm. The line features are matched using the LBD (line band descriptor). The workflow for extracting point and line features is illustrated in Fig. 3.

Firstly, the extraction threshold is set, and the pixel points with a gray value greater than or less than the threshold in the image are extracted to realize the image binarization and convert the image into the black and white binary image for the next stage of image processing. The N-layer Gaussian pyramid helps in extracting features at different scales, while the N-scale space pyramid enables analysis and processing at different resolutions. Having both pyramids allows for more comprehensive image analysis and facilitates tasks such as object detection, recognition,

and image matching across different scales [19]. On each layer, Oriented FAST feature points are extracted. Subsequently, a quadtree algorithm is applied to ensure the uniform distribution of Oriented FAST feature points across each layer of the pyramid. Finally, the quadtree algorithm is applied again on the original image to achieve a uniform and dispersed distribution of feature points while reducing redundancy.

2.4. Line Feature Detection

The traditional line feature detection algorithm often detects a large number of short line segments in complex scenes. In the case of weak textures, the insufficient constraints of visual SLAM can be aggravated by the presence of cluttered line features, increasing the computational burden for line detection, extraction, description, and matching. Consequently, the probability of incorrect line feature matching also increases. We incorporated a length suppression step into the original EDlines line segment extraction algorithm to effectively eliminate line segments below the specified threshold. Effective long-line features are more likely to be consistently detected and extracted across multiple images, contributing more to positioning accuracy. This algorithm aims to generate clear, continuous, and reasonably long edge fragments. Given a grayscale input image, the steps of the improved EDlines algorithm are as follows:

Edge Drawing: First, the input image undergoes Gaussian filtering (using a Gaussian kernel as 5×5 with a default variance of 1). Then, the gradients and directions of the pixels are calculated using the following equations:

$$g_x(x,y) = \frac{L(x+1,y) - L(x,y) + L(x+1,y+1) - L(x,y+1)}{2}, \tag{6a}$$

$$g_y(x,y) = \frac{L(x,y+1) - L(x,y) + L(x+1,y+1) - L(x+1,y)}{2}, \tag{6b}$$

$$g(x,y) = \sqrt{g_y(x,y)^2 + g_y(x,y)^2}, \quad \theta = \arctan\frac{g_x(x,y)}{-g_y(x,y)}. \tag{6c}$$

Here, $g(x,y)$ and θ represent the gradient magnitude and direction of a pixel, respectively. Pixels are selected as anchor points (likely edge pixels) if their gradients are greater than the gradients of their adjacent pixels in the same direction and no less than μ. Finally, these anchor points are connected to form initial pixel chains.

Length Suppression: The initial pixel chains are subjected to length filtering, where chains shorter than a specified threshold are discarded. This step ensures that the extracted features consist primarily of long-line segments. Let $a\ (x_a, y_a)$ and $b\ (x_b, y_b)$ represent the endpoints of a pixel chain:

$$\begin{cases} \text{len}_n = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2} \\ \text{len}_n \geq \text{len}_{\min} = \sigma \min(W, H). \end{cases} \tag{7}$$

Here, $\text{len}_{\min}$ is the minimum pixel chain length, W and H represent the image width and height, and σ is a Length scale factor.

Least Squares Line Fitting: The pixel chains obtained from the Edge Drawing algorithm are segmented into one or multiple line segments. Each segment is then fitted using the least squares method. If the error exceeds a certain threshold, a new line segment is generated. The process is repeated for the remaining pixels in the chain until all anchor points are processed:

$$\min \sum_{a=1}^{m} |\delta_a| = \sum_{a=1}^{m} |\varphi(x_a) - y_a|. \tag{8}$$

Where x_a and y_a are pixel coordinates at the end of adjacent line segments.

2.5. Point and Line Feature Representation and Reprojection Error Model

A line in three-dimensional space can be represented using Plücker coordinates, which is a six-dimensional vector: $L = (m^T, d^T)^T \in \mathbb{R}^6$, where $m \in \mathbb{R}^3$ is the normal vector, $d \in \mathbb{R}^3$ is the moment vector, and they satisfy the condition $n^T d = 0$. Assuming a line L_W, in the world coordinate system, it can be transformed into a line L_C in the camera coordinate system using the information matrix:

$$T_{CW} = \begin{bmatrix} R_{CW} & t_{CW} \\ 0 & 1 \end{bmatrix}. \tag{9}$$

Here, $R_{CW}(3\times3)$ and $t_{CW}(3\times1)$ are the rotation and translation matrices (or the extrinsic parameters) that transform from the world coordinate system to the camera coordinate system. The projection of the line onto the camera plane can be described by the projection equation:

$$L_C = \begin{bmatrix} m_C \\ d_C \end{bmatrix} = T_{CW} L_{CW} = \begin{bmatrix} R_{CW} & [t_{CW}] \times R_{CW} \\ 0 & R_{CW} \end{bmatrix} \begin{bmatrix} m_W \\ d_W \end{bmatrix}. \tag{10}$$

As shown in Fig. 4, when a new line segment is observed in two different images, it is represented using Plücker coordinates. In Fig. 4b, the three-dimensional line L is observed and labeled as cl and

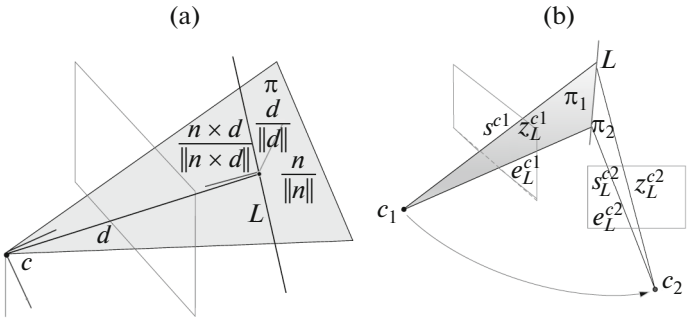

Fig. 4. (a) Plücker coordinates, and (b) line segment reprojection error model.

c2 in two camera views at different time instances, represented by z_L^{c1} and z_L^{c2}, respectively.

In this paper, T_{CB} is assumed that the extrinsic matrix, which represents the transformation between the camera coordinate system and the body coordinate system (or IMU coordinate system), is known. The state of an image frame i is denoted as $\left\{R_{WBi}, P_{WBi}, V_{WBi}, b_i^g, b_i^a\right\}$, and the observed position of a landmark named k in the image frame is denoted as $\hat{p}_{ik}$. The uncertainty of the landmark position is assumed to be one pixel. Thus, the reprojection error of the image feature point can be obtained as follows:

$$\begin{aligned} e_{ik}^p &= \hat{p}_{ik} - \frac{1}{Z_C} K \begin{bmatrix} I_3 & 0_{3\times 1} \end{bmatrix} \begin{bmatrix} \ell_k^C \\ 1 \end{bmatrix} \\ &= \hat{p}_{ik} - \frac{1}{Z_C} K \begin{bmatrix} I_3 & 0_{3\times 1} \end{bmatrix} T_{CB} \begin{bmatrix} R_{WBi} & P_{WBi} \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \ell_k \\ 1 \end{bmatrix}. \end{aligned} \tag{11}$$

Here, K is the camera's intrinsic matrix, z_C is the third component of the landmark's coordinates named ℓ_k^C in the camera coordinate system, which satisfies:

$$\ell_k^C = R_{CB} R_{WBi}^{-1} \left(\ell_k - p_i\right) + t_{CB}. \tag{12}$$

The projection of the landmark L_C onto the image plane yields l_C, which can be represented in Plücker coordinates as follows:

$$l_C = K_L m_C = \begin{bmatrix} f_y & 0 & 0 \\ 0 & f_x & 0 \\ -f_y c_x & -f_x c_y & f_x f_y \end{bmatrix} m_C = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix}. \tag{13}$$

Here, f_x, f_y and c_x, c_y are the camera's focal length, and principal point coordinates, respectively. The K_L is the intrinsic matrix used to project a 3D line on the image plane. Based on Fig. 4b, the reprojection error of the three-dimensional world line segment onto the two-dimensional plane of the current image frame can be defined as:

$$e_{ik}^l = \begin{bmatrix} \dfrac{x_s^T l_c}{\sqrt{l_1^2 + l_2^2}} & \dfrac{x_e^T l_c}{\sqrt{l_1^2 + l_2^2}} \end{bmatrix}^\top, \tag{14}$$

where $\{x_s, x_e\}$ are the 2D starting point and ending point of an extracted line segment.

2.6. Tightly Coupled VIO Based on Point-Line Feature

In visual SLAM, Bundle Adjustment is used to optimize camera poses or 3D maps by minimizing the reprojection error on the image plane. The nonlinear optimization process of Bundle Adjustment can be represented using factor graphs (Fig. 5) [20, 21]. In Fig. 5a, circular nodes represent camera poses or landmark points that need to be optimized, while square nodes represent visual observations, serving as constraints between nodes. In this paper, in addition to point features, line features are also included as constraints. For visual-inertial SLAM, a tightly coupled framework using factor graphs is also employed to optimize all state variables, as shown in Fig. 6b. Additionally, IMU pre-integration is incorporated to constrain the IMU states over continuous periods.

The fusion optimization stage plays a pivotal role in the overall mileage calculation method. The main innovation in this paper's optimization stage lies in utilizing line features as constraint conditions to compensate for the insufficient feature points in weak texture environments, thereby providing effective constraints. Additionally, an optimization iteration is performed using a re-projection error model of point-line features. The resulting state from the iteration is then utilized as prior information for optimizing subsequent frames, serving as a reference to reduce computational burden and achieve improved optimization results.

Once the camera pose is estimated, the map points and lines in the local map will be projected, and key points and key lines will be matched. Then, we optimize the current frame j by minimizing the feature reprojection error of all matched points and lines and IMU error. When performing tracking immediately after the map update, the IMU error term connects

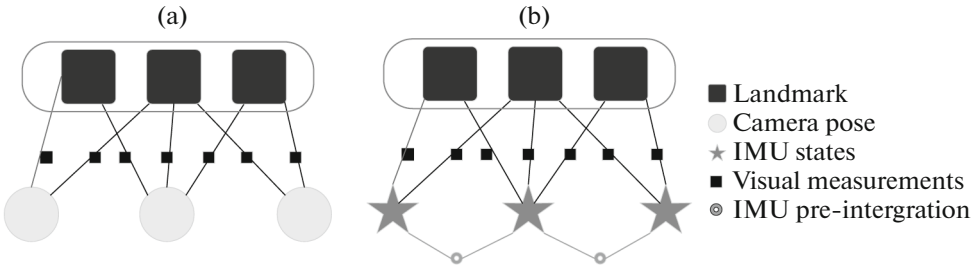

Fig. 5. Factor (a) represents the visual SLAM constraint relationship, and (b) represents the visual inertial SLAM constraint relationship.

the current frame j with the latest keyframe i. The estimated state of frame j is given by $\theta = \left\{R_{WBj}, P_{WBj}, V_{WBj}, b_j^g, b_j^a\right\}$:

$$\theta^* = \arg\min_{\theta}\left(\sum_k E_{proj}(k,j) + E_{IMU}(i,j)\right). \quad (15)$$

Here, E_{proj} is the error function for the reprojection of feature points and feature lines:

$$E_{proj}(k,j) = \rho\left(\left(x^k - \pi\left(\ell_k^C\right)\right)^\top \sum_k \left(x^k - \pi\left(\ell_k^C\right)\right)\right) + \rho\left(\left(x^k - \pi(L_C)\right)^\top \sum_k \left(x^k - \pi(L_C)\right)\right). \quad (16)$$

The IMU error term E_{IMU} is defined as:

$$E_{IMU}(i,j) = \rho\left(\sum_k \left[e_R^\top e_V^\top e_P^\top\right] \sum_I \left[e_R^\top e_V^\top e_P^\top\right]^\top\right) + \rho\left(e_b^\top \sum_R e_b\right), \quad (17)$$

where Σ_k is the information matrix associated with the key-feature scale, Σ_I is the information matrix of the pre-integrated IMU measurements, Σ_R is the information matrix of the random walk bias, and ρ represents the Huber robust cost function [18]. In this paper, the Gauss-Newton algorithm implemented in g2o is used to solve this optimization problem [22].

After optimization, the estimated state and Hessian matrix obtained are used as priors for the next optimization step. Assuming no map update, the next frame $j+1$ will be optimized by incorporating the estimated state from frame j and using the prior computed at the end of the previous optimization:

$$\theta = \{R_{WBj}, P_{WBj}, V_{WBj}, b_j^g, b_j^a, R_{WBj+1}, P_{WBj+1}, V_{WBj+1}, b_{j+1}^g, b_{j+1}^a\}, \quad \theta^* = arg\min_{\theta} \times\left(\sum_k E_{proj}(k, j+1) + E_{IMU}(j, j+1) + E_{prior}(j)\right), \quad (18)$$

$$E_{prior}(j) = \rho\left(\left[e_R^\top e_V^\top e_P^\top e_b^\top\right] \sum_p \left[e_R^\top e_V^\top e_P^\top e_b^\top\right]^\top\right), \quad (19)$$

$$e_R = \mathrm{Log}\left(\bar{R}_{BWj} R_{WBj}\right) \quad e_V = \bar{V}_{WBj} - V_{WBj}, \quad (20a)$$

$$e_P = \bar{P}_{WBj} - P_{WBj} \quad e_b = \bar{b}_j - b_j. \quad (20b)$$

In the above equation, $(\cdot)$ and Σ_p are the estimated state and Hessian matrix generated from the previous work. After this optimization step, frame j will be marginalized [23]. The algorithm utilizes priors and continuously connects consecutive frame optimizations until a map change occurs. At that point, the prior becomes invalid, and the tracking process will again connect the current frame with the most recent keyframe to continue the optimization computation.

3. EXPERIMENTAL VERIFICATION AND ANALYSIS

The objective of this study is to validate the proposed fusion of improved line feature-based visual-

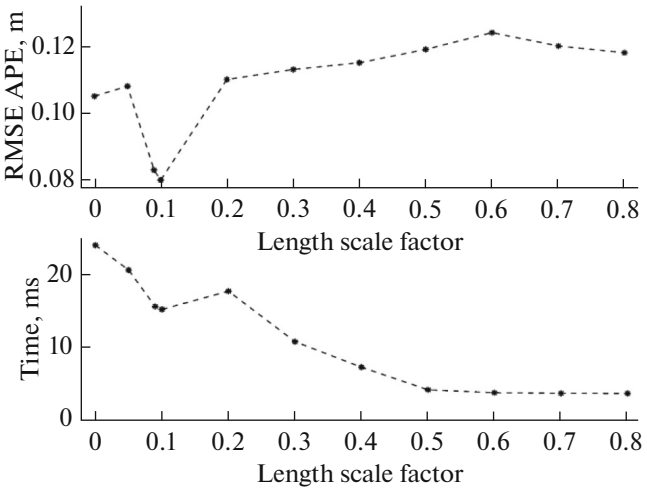

Fig. 6. Length suppression studies: Time means the average execution time of the line feature tracking and mean absolute pose error (APE) for different length scale factor values on MH_05.

Table 1. EuRoC dataset characteristics

Name	Length/duration	Note
V1_01_easy	58.6 m/144 s	Slow motion, bright scene
V1_02_medium	75.9 m/83.5 s	Fast motion, bright scene
V1_03_difficult	79.0 m/10 5s	Fast motion, motion blur
V2_01_easy	36.5 m/112 s	Slow motion, bright scene
V2_02_medium	83.2 m/115 s	Fast motion, bright scene
V2_03_difficult	86.1 m/115 s	Fast motion, motion blur
MH_01_easy	80.6 m/182 s	Good texture, bright scene
MH_02_easy	73.5 m/150 s	Good texture, bright scene
MH_03_medium	130.9 m/132 s	Fast motion, bright scene
MH_04_difficult	91.7 m/99 s	Fast motion, dark scene
MH_05_difficult	97.6 m/111 s	Fast motion, dark scene

inertial odometry for enhancing localization accuracy and robustness in low-texture environments compared to traditional feature point-based visual odometry. In this paper, two datasets, EuRoC [24] and TUM RGB-D [25], are used to test the proposed method. All experiments were conducted on a computer with an AMD 4800HS (2.90 GHz) processor, 16GB of memory, and an Ubuntu 16.04 operating system.

3.1. Experiment of Line Segment Extraction Algorithm

To evaluate the performance of the proposed line feature extraction algorithm and the improvement it brings to the visual SLAM system, five sub-sequences from the EuRoC dataset were selected for three experiments: MH_03_medium, V1_02_medium, MH_05_difficult, V1_03_difficult, and V2_03_difficult. The EuRoC dataset consists of 11 sub-sequences that are divided into three levels of difficulty based on different influencing factors, namely easy, medium, and difficult [24]. The specific division of difficulty levels is presented in Table 1.

First, the gradient calculation in ED is followed by setting a threshold μ to eliminate pixels that do not contain edge elements (refer to Section 2.3). The maximum quantization error between two successive pixels is 2, which occurs when the errors of adjacent pixels are +1 and −1. With an angle tolerance of 22.5° [16], the gradient threshold can be calculated as follows:

$$\mu = \frac{1}{\sin(22.5)} = 5.22.$$

The threshold value of the length suppression is selected for experiments. Corresponding to (7), the image size output by the two public data sets used in the experiment is $W = 752$ in width and $H = 480$ in height. The results are obtained by repeated experiments under MH_05_difficult sequences by the algorithm in this paper. The results are shown in Fig. 6. While the σ directly affects the extraction description and matching time, the larger the value of σ, the fewer line segments will be extracted. From the perspective of efficiency, the best value interval of σ is [0.07, 0.12]. In this experiment, σ is set to 0.09, which satisfies most of the excellent performance achieved in the experiment.

After obtaining the best parameters, we compare LSD, EDlines, and the improved EDlines algorithm proposed in this paper in OpenCV. The average computation time per frame, the number of extracted line features, and the matching rate were statistically analyzed. Table 2 presents the results. The proposed line extraction algorithm had an average computation time of 26.2 ms per frame. Compared to the original EDlines algorithm (average of 22.8 ms per frame), there was an additional 3.4 ms per frame required for line feature extraction. However, the LSD algorithm had an average extraction time of 51.8 ms per frame, which was 227% of the original EDlines algorithm's time and 197% of the proposed algorithm's time.

From Table 2, it can be observed that the LSD algorithm extracted the highest number of line features, followed by the original EDlines algorithm, and the proposed improved EDlines algorithm extracted the fewest. By incorporating a length suppression step into the EDlines line segment extraction algorithm to mitigate the extraction of short line segments, there has been a significant reduction in the number of extracted line segments and an increased proportion of longer line segments. Consequently, this enhancement has led to an improved success rate in line segment matching (i.e., matching line features between two image frames). We define m_s as the count of successfully matched line segments, m as the total number of extracted line segments, and calculate the matching rate as $m_s/m \times 100\%$, the values of m_s and m will be given after OpenCV has finished processing the image. The proposed algorithm extracted 63% fewer line features per frame compared to the LSD algorithm and a 52% reduction compared to the original

Table 2. Comparison of extracted line feature data by algorithms

Dataset	LSD			EDlines			Ours		
	Time/ms	amount	Matching rate, %	Time, ms	amount	Matching rate, %	Time, ms	amount	Matching rate, %
MH_03	56.8	571	72.3	25.7	426	82.0	32.7	256	84.6
V1_02	52.9	487	83.1	23.1	352	81.3	27.8	194	91.2
MH_05	48.1	524	57.5	24.0	411	73.2	24.2	163	85.6
V1_03	54.5	445	62.4	21.3	348	68.7	22.9	132	79.8
V2_03	46.8	367	74.7	19.8	324	77.8	23.6	145	82.5

EDlines algorithm. Moreover, the proposed improved EDlines algorithm achieved a 16% higher matching rate than the LSD algorithm and a 9.7% improvement over the original EDlines algorithm. The effectiveness of the improved EDlines algorithm in line feature extraction compared to the original EDlines algorithm is illustrated in Fig. 7. In this experiment, the output frequency of the EuRoC dataset is about 20 Hz, and the time consumption of feature point extraction is about 13 ms. The time consumption of the point-line feature extraction thread of the algorithm in this paper is about 28 ms in total, which meets the requirements of real-time operation, while the line feature extraction based on LSD is time-out.

The results showed that both LSD and the original EDlines algorithm extracted a significant amount of redundant line segments, such as short lines and neighboring lines with high similarity. These redundant line segments can negatively impact the real-time performance of the visual SLAM system and increase the probability of matching errors, consuming computational resources. The experimental results of line feature extraction demonstrated that, although the proposed improved EDlines algorithm slightly increased the extraction time compared to the original algorithm, it ensured real-time performance while extracting more effective features. This reduced the computational burden for optimizing line reprojection errors and maintained algorithm real-time capability. Additionally, the proposed algorithm achieved higher-quality extracted line segments and an 11% improvement in feature-matching success rate compared to the original algorithm, demonstrating its enhanced robustness.

3.2. Experiment on Effectiveness of Line Segments

To verify the inclusion of line features and the effect of improving line features, the trajectory error of monocular SLAM is compared with other state-of-the-art systems in this section. The TUM RGB-D dataset provides real-world indoor image sequences, including low-contrast and low-texture environments or different texture environments. Four different classes in this dataset were selected, with a total of 13 sequences. Compared with PL-SLAM [7] which was also based on point-line features and ORB-SLAM [4] which was only based on point features, the experiment was carried out.

The effectiveness of adding line segments with a well-formulated reprojection error in SLAM can be found in the fr1_floor of Table 3. This shows that the

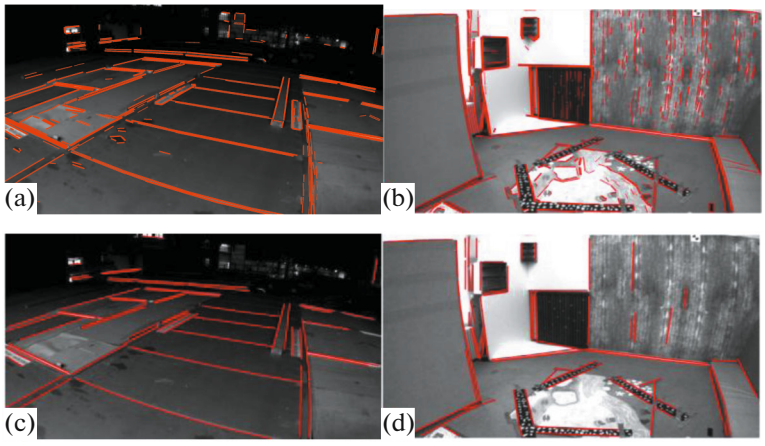

Fig. 7. (a) and (b): EDlines extract line segment effects, (c) and (d): EDlines are improved to extract line segment effects.

Table 3. Motion estimation errors (cm) of each algorithm running under the TUM RGB-D dataset. Each result from ours was calculated as the average of over 5 executions

TUM RGB-D Sequence		PL-SLAM		ORB-SLAM	Ours	
		line	point + line	point	line	point + line
Testing	fr1_xyz	1.46	1.21	1.38	1.29	**1.08**
	fr2_xyz	1.49	0.43	**0.36**	0.67	0.59
Handheld SLAM	fr1_floor	9.42	7.59	8.71	5.88	**3.32**
	fr2_desk	4.32	–	2.36	1.94	**1.75**
	fr2/360_kidnap	60.11	3.92	4.99	29.39	**2.43**
Texture	fr3_str_tex_far	1.25	**0.89**	0.98	1.14	1.09
	fr3_str_tex_near	7.47	**1.25**	1.55	1.86	1.43
	fr3_nstr_tex_far	37.60	–	–	18.33	**4.67**
	fr3_nstr_tex_near	1.58	2.06	2.88	1.66	**1.41**
	fr3_long_office	5.33	1.97	4.05	3.28	**1.52**
Dynamic object	fr3_sit_halfsph	9.05	1.31	1.48	4.72	**1.13**
	fr3_walk_xyz	–	1.54	1.64	2.30	**1.21**
	fr3_walk_halfsph	–	1.60	2.09	–	**1.36**
AVG		12.64	2.16	2.70	**6.035**	**1.77**

Symbol "–" means that the tracking is lost at some point and a significant portion of the sequence is not processed by the system.

line segment significantly improves the performance of monocular SLAM in the absence of feature points in low-texture and low-contrast environments. Moreover, when there is a sufficient number of point features to support the system's tracking capabilities, incorporating line features can enhance the positioning accuracy of the monocular SLAM. The results presented in Table 3 demonstrate the excellent scale and rotation invariance of line features in the pure plane scene with distance transformation, which can significantly enhance the performance of monocular SLAM algorithms, such as fr2_desk, fr3_str_tex_far, fr3_nstr_tex_far, and fr3_nstr_tex_near. The real-time performance of feature extraction and the validity of extracted line segments are rigorously evaluated in the context of tracking dynamic objects. The proposed system demonstrates exceptional performance across various scenarios, including fr3_sit_halfsph, fr3_walk_xyz, and fr3_walk_halfsph, and the same conclusion can be inferred from Table 4 presented in the subsequent section.

3.3. Open Dataset Simulation Experiment

The present study introduces a visual-inertial coupled algorithm that incorporates line features into VI ORB-SLAM [5]. To evaluate the algorithm's effectiveness, it is compared with the point feature-based VI ORB-SLAM and the point-line feature-based IMU fusion algorithm PL-VIO [10] in terms of pose accuracy. The simulation experiments are conducted on 11 sub-sequences from the EuRoC dataset. The motion estimation errors of different algorithms in the EuRoC data set (a total of 11 sub-sequences) are shown in Table 4. Absolute trajectory error is used to evaluate the accuracy of the algorithm, that is, the root-mean-square error of the Euclidean distance between the estimated pose and the real pose. The Evaluation of Odometry and SLAM (EVO) tool was used for data alignment and error calculation when compared with the real trajectory. The ideal scale factor $|\hat{S}|$ is measured by aligning the estimated keyframe trajectory to the best fit with the ground truth. The scale error can be calculated as $|S^* - \hat{S}|/|\hat{S}| \times 100\%$.

Among the 11 sub-sequences, the proposed algorithm outperforms the other two algorithms in 8 datasets. The accuracy improvement of motion estimation due to more effective linear feature constraints in the EuRoC dataset is also evident from the error performance analysis. Our method outperforms PL-VIO with wireless segment length suppression significantly in sequences like MH_01easy, MH_02_easy, and MH_03_medium that contain abundant linear features. While redundant linear feature constraints enhance tracking robustness, excessive feature tracking can hamper real-time performance and nonlinear optimization may compromise overall system accuracy. In low contrast and low texture sequences such as V1_02_medium, V1_03_difficult, V2_02_medium, and V2_03_diffcult, the inclusion of linear features improves our method's accuracy compared to VI ORB-SLAM. Notably, VI ORB-SLAM fails to track the entire dataset in the challenging sequence of

Table 4. Motion estimation errors (cm) of each algorithm running under the EuRoC dataset. Each result from ours was calculated as the average of over 5 executions

EuRoC Sequence	PL-VIO	VI ORB-SLAM		Ours	
	point + line + inertial	point + inertial		point + line + inertial	
	RMSE, cm	RMSE, cm	Scale error, %	RMSE, cm	Scale error, %
V1_01_easy	6.5	**2.7**	0.9	4.4	1.1
V1_02_medium	9.3	2.8	0.8	**2.3**	0.7
V1_03_difficult	14.1	–	–	**5.6**	2.1
V2_01_easy	5.8	**3.2**	0.2	4.1	0.7
V2_02_medium	9.4	4.1	1.4	**3.5**	0.8
V2_03_difficult	26.0	7.4	0.7	**6.9**	0.4
MH_01_easy	8.4	7.5	0.5	**6.7**	1.0
MH_02_easy	7.5	8.4	0.8	**5.8**	0.9
MH_03_medium	11.4	8.7	1.5	**7.2**	1.1
MH_04_difficult	**12.2**	21.7	3.4	16.9	2.1
MH_05_difficult	14.7	8.2	0.5	**8.0**	0.4
AVG	11.39	7.47	1.07	**6.49**	**1.03**

Scale: The estimated keyframe trajectory is scaled to the best fit with the ground-truth trajectory.
"–" means that the tracking is lost at some point and a significant portion of the sequence is not processed by the system.

V1_03_diffcult with illumination changes and motion blur; this further highlights the effectiveness of linear feature constraints on motion estimation. Comparing the scale errors between VI ORB-SLAM and the proposed algorithm with line feature integration, in the MH factory environment, VI ORB-SLAM shows lower scale errors due to its full bundle adjustment optimization and loop closure detection. However, in environments like V1 and V2 with a large number of low texture regions, the proposed algorithm with line feature constraints exhibits a significant reduction in scale error compared to VI ORB-SLAM, achieving better accuracy performance (Figs. 8, 9, 10).

The running time of the tracking thread for the proposed algorithm, ORB-SLAM3 [26], and PL-SLAM, as well as the improved EDlines line feature extraction algorithm used in this study, are compared in Table 5. It can be observed that the proposed algorithm benefits from the inclusion of IMU pre-integration, resulting in a shorter time for initial pose estimation, map tracking, and keyframe decision compared to PL-SLAM. Moreover, the proposed algorithm reduces the line feature extraction time by approxi-

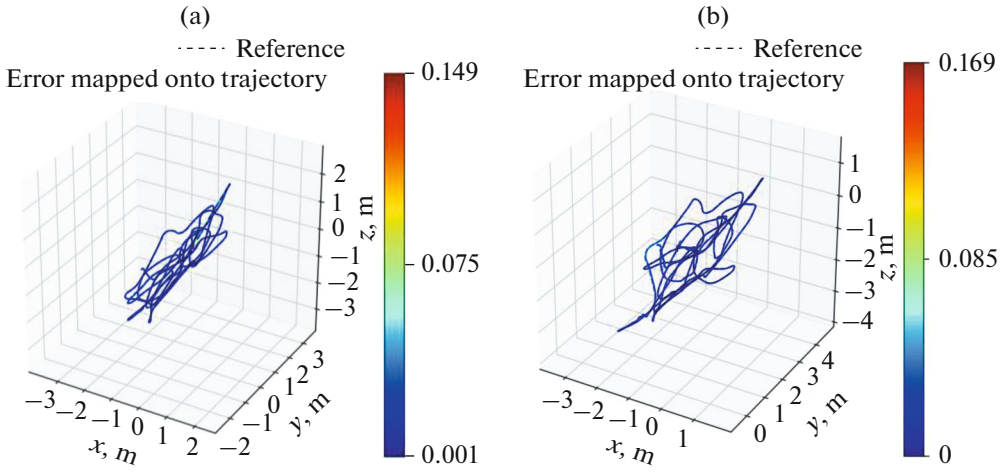

Fig. 8. Comparison of the performance in the V1_02 dataset, The left side of the picture shows the error reference bar, from blue to red or from bottom to top indicating the increase of the ATE. (a) Ours, and (b) VI ORB-SLAM.

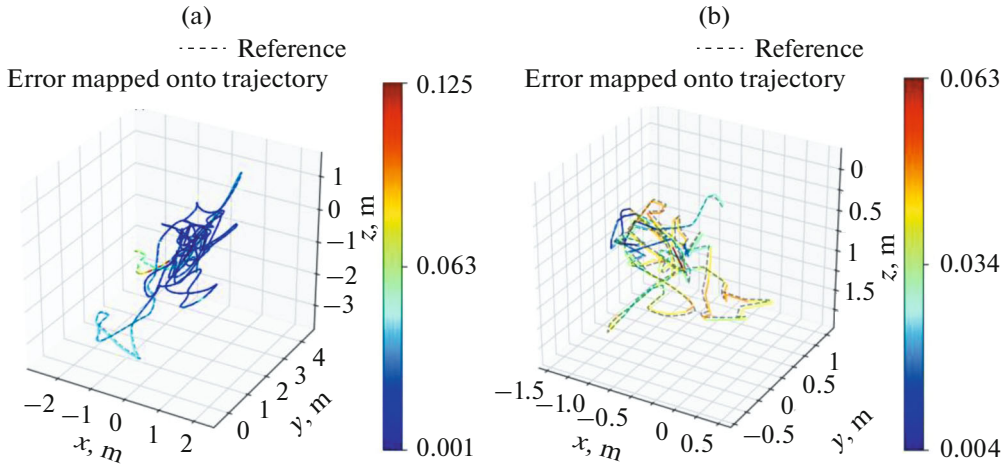

Fig. 9. Comparison of the performance in the V103 dataset, The left side of the picture shows the error reference bar, from blue to red or from bottom to top indicating the increase of the ATE. (a) Ours, and (b) VI ORB-SLAM.

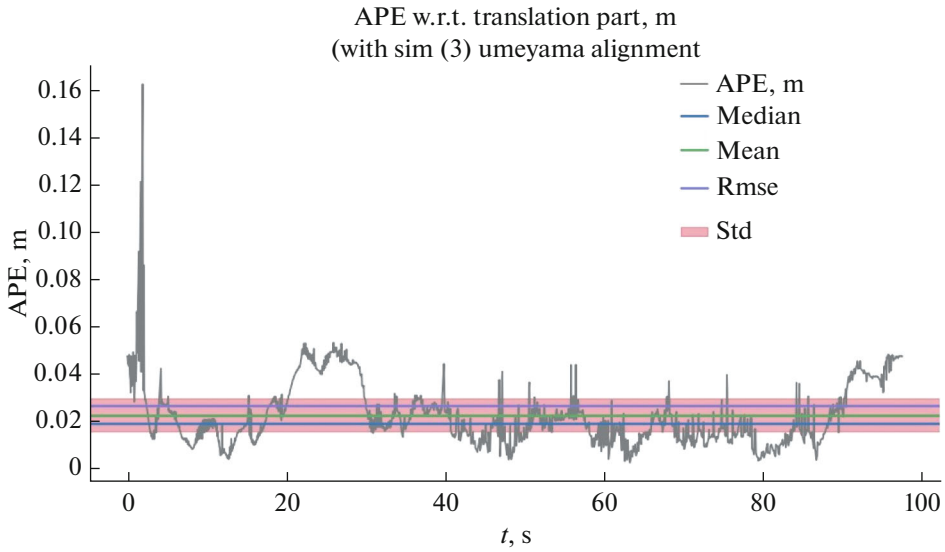

Fig. 10. Evaluation results of the proposed algorithm in the V1_02 dataset.

mately 50% through the use of the improved EDlines algorithm. However, the tracking thread of the proposed algorithm takes slightly more time than the state-of-the-art ORB-SLAM3, while the inclusion of line feature constraints leads to slightly lower time consumption in local map tracking compared to ORB-SLAM3. The time consumption of the tracking part for each algorithm indicates that the proposed visual-inertial algorithm based on the combination of improved line features and point features has significant advantages in terms of running speed compared to PL-SLAM. Although the proposed algorithm slightly lags behind ORB-SLAM3 in terms of real-time performance, it still achieves a suboptimal performance.

4. CONCLUSIONS AND FUTURE WORK

In response to the issue of point feature tracking failures in indoor environments with low texture, this paper proposes a visual-inertial odometry algorithm based on the fusion of point and line features. The proposed improved EDlines line feature extraction algorithm effectively reduces the extraction of low-quality line segments and decreases the mismatch rate of line features, providing significant and effective constraints for odometry estimation. By incorporating point-line reprojection errors, IMU errors, and prior information into a nonlinear optimization framework, the robustness of point feature tracking and the accuracy of localization is enhanced, particularly in weak texture environments. The proposed improved

Table 5. Comparison of the time (ms) consumption of the three algorithms for tracking threads

Tracking	ORB-SLAM3	PL-SLAM	Ours
Feature extraction of point	11.98	10.76	12.49
Feature extraction of line	–	28.56	14.31
IMU pre-integration	0.29	–	0.32
Initial pose estimation	0.52	7.16	2.63
Local map tracking	10.74	12.58	10.22
Chose new keyframe	0.08	0.32	0.17
Total, ms	23.6	59.9	40.6
Rate, Hz	**20**	5	**10**

Symbol "–" means that the algorithm does not have this process for tracking.
The run rate of not less than 10hz can be considered to meet the real-time performance.

EDlines line feature extraction algorithm and odometry algorithm are tested on the EuRoC dataset and TUM RGB-D dataset, demonstrating higher localization accuracy, robustness, and real-time performance compared to existing approaches.

For future work, the limitations of the proposed system are evident in large-scale scenes, prompting us to consider incorporating full bundle adjustment optimization and loop closure detection into the VIO system. Furthermore, we aim to explore the promising avenue of semantic SLAM and expand our approach from monocular vision to stereo vision, leveraging planar features as well.

FUNDING

This work was supported by ongoing institutional funding. No additional grants to carry out or direct this particular research were obtained.

DATA AVAILABILITY

Public datasets EuRoC and TUM RGB-D were used.

CONFLICT OF INTEREST

The authors of this work declare that they have no conflicts of interest.

REFERENCES

1. Liu, Y.J., Zhang, Y.Z., Rong, L., Jiang, H., and Deng, Y., Visual odometry based on the direct method and the inertial measurement unit, *Robot*, 2019, vol. 41, no. 5, pp. 683–689. https://doi.org/10.13973/j.cnki.robot.180601
2. Pan, L.H., Tian, F.Q., Ying, W.J., Liang, W.G., and She, B., VI-SLAM algorithm with camera-IMU extrinsic automatic calibration and online estimation, *Chinese Journal of Scientific Instrument*, 2019, vol. 40, no. 6, pp. 56–67. https://doi.org/10.19650/j.cnki.cjsi.J1904954
3. Mourikis, A.I., and Roumeliotis, S.I., A multi-state constraint Kalman filter for vision-aided inertial navigation, *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2007, pp. 3565–3572. https://doi.org/10.1109/ROBOT.2007.364024
4. Mur-Artal, R., Montiel, J.M.M., and Tardós, J.D., ORB-SLAM: A versatile and accurate monocular SLAM system, *IEEE Transactions on Robotics*, 2015, vol. 31, no. 5, pp. 1147–1163. https://doi.org/10.1109/TRO.2015.2463671
5. Mur-Artal, R., and Tardós, J.D., Visual-inertial monocular SLAM with map reuse, *IEEE Robotics and Automation Letters*, 2017, vol. 2, no. 2, pp. 796–803. https://doi.org/10.1109/LRA.2017.2653359
6. Qin, T., Li, P., and Shen, S., VINS-Mono: A robust and versatile monocular visual-inertial state estimator, *IEEE Transactions on Robotics*, 2018, vol. 34, no. 4, pp. 1004–1020. https://doi.org/10.1109/TRO.2018.2853729
7. Pumarola, A., Vakhitov, A., Agudo, A., Sanfeliu, A., and Moreno-Noguer, F., PL-SLAM: Real-time monocular visual SLAM with points and lines, *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 4503–4508. https://doi.org/10.1109/ICRA.2017.7989522
8. Gioi, R., Jakubowicz, J., Morel. J.-M., and Randall, G., LSD: A fast line segment detector with a false detection control, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008, vol. 32, no. 4, pp. 722–732. https://doi.org/10.1109/TPAMI.2008.300
9. Gomez-Ojeda, R., Briales, J., and Gonzalez-Jimenez, J., PL-SVO: Semi-direct monocular visual odometry by combining points and line segments, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4211–4216. https://doi.org/10.1109/IROS.2016.7759620
10. He, Y.J., Zhao, J., Gao, Y., He, W., and Yuan, K., PL-VIO: Tightly-coupled monocular visual–inertial odometry using point and line features, *Sensors*, 2018, vol. 18, no. 4, pp. 1159–1184. https://doi.org/10.3390/s18041159
11. Fu, Q., Wang, J.L., Yu, H.H., Islam, A., Guo, F., and Zhang, H., PL-VINS: Real-time monocular visual-inertial SLAM with point and line, *arXiv,* 2020, pre-

print arXiv:2009.07462.
https://doi.org/10.48550/arXiv.2009.07462

12. Shu, F.W., Wang, J.X., and Pagani, A., Structure PLP-SLAM: Efficient sparse mapping and localization using point, line, and plane for monocular, RGB-D and stereo cameras, *ArXiv*, 2022, preprint ArXiv 2207 06058, https://doi.org/10.48550/arXiv.2207.06058
13. Yoon, S., and Kim, A., Line as a visual sentence: Context-aware line descriptor for visual localization, *IEEE Robotics and Automation Letters*, 2021, vol. 6, no. 4, pp. 8726–8733. https://doi.org/10.1109/LRA.2021.3111760
14. Yunus, R., Li, Y., and Tombari, F., Manhattan SLAM: Robust planar tracking and mapping leveraging mixture of Manhattan frames, *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 6687–6693. https://doi.org/10.1109/ICRA48506.2021.9562030
15. Wu, T.-H., and Chen, K.-W., LGC Net: Feature enhancement and consistency learning based on local and global coherence network for correspondence selection, *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 6182–6188. https://doi.org/10.1109/ICRA48891.2023.10160290
16. Akinlar, C., and Topal, C., EDlines: Real-time line segment detection by edge drawing, *Proc. 18th IEEE International Conference on Image Processing (ICIP)*, 2011, pp. 2837–2840. https://doi.org/10.1016/j.patrec.2011.06.001
17. Lupton, T., and Sukkarieh, S., Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions, *IEEE Transactions on Robotics*, 2012, vol. 28, no. 1, pp. 61–76. https://doi.org/10.1109/TRO.2011.2170332
18. Forster, C., Carlone, L., Dellaert, F., and Scaramuzza, D., On-manifold pre-integration for real-time visual-inertial odometry, *IEEE Transactions on Robotics*, 2017, vol. 33, no. 1, pp. 1–21. https://doi.org/10.1109/TRO.2016.2597321
19. Mur-Artal, R., and Tardós, J.D., ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras, *IEEE Transactions on Robotics*, 2017, vol. 33, no. 5, pp. 1255–1262. https://doi.org/10.1109/TRO.2017.2705103
20. Kschischang, F.R., Frey, B.J., and Loeliger, H.-A., Factor graphs and the sum-product algorithm, *IEEE Transactions on Information Theory*, 2001, vol. 47, no. 2, pp. 498–519. https://doi.org/10.1109/18.910572
21. Dellaert, F., and Kaess, M., Square root SAM: Simultaneous localization and mapping via square root information smoothing, *International Journal of Robotics Research*, 2006, vol. 25, no. 12, pp. 1181–1203. https://doi.org/10.1177/0278364906072768
22. Kümmerle, R., Grisetti, G., Strasdat, H.M., Konolige, K., and Burgard, W., *G2o: A general framework for graph optimization, Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3607–3613. http://doi.org/10.1109%2FICRA.2011.5979949
23. Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., and Furgale, P., Keyframe-based visual–inertial odometry using nonlinear optimization, *International Journal of Robotics Research*, 2015, vol. 34, no. 3, pp. 314–334. https://doi.org/10.1177/0278364914554813
24. Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M.W., and Siegwart, R., The EuRoC micro aerial vehicle datasets, *International Journal of Robotics Research*, 2016, vol. 40, no. 6, pp. 1157–1163. https://doi.org/10.1177/0278364915620033
25. Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D., A benchmark for the evaluation of RGB-D SLAM systems, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 573–580. https://doi.org/10.1109/IROS.2012.6385773
26. Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M.M., and Tardós, J.D., ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM, *IEEE Transactions on Robotics*, 2021, vol. 37, no. 6, pp. 1874–1890. https://doi.org/10.1109/TRO.2021.3075644