PE-VINS: Accurate Monocular Visual-Inertial SLAM with Point-Edge Features

Changxiang Liu[1], Hongshan Yu[1*], Panfei Cheng[1], Wei Sun[1], Javier Civera[2], Xieyuanli Chen[3]

***Abstract*—Visual-Inertial Navigation Systems (VINS) is a significant undertaking in computer vision, robotics, and autonomous driving. Currently, point-line VINS have attracted significant attention due to their increased robustness and accuracy compared to point-only VINS. However, their effectiveness relies on the existence of clear line structures within the scene. Point-line VINS may become inaccurate or fail when scenes contain scattered lines or other features like arcs. Moreover, extracting and matching line features can bring computational overheads due to complex geometric models. In order to address VINS challenges without the overheads related to lines, we propose a novel approach, denoted as PE-VINS, which adds edge features to point-based VINS. Our proposed employs edge features in scenes to establish extra correspondences between views and then enhance its accuracy and robustness. Our method identifies edge features using image gradients and selects the most informative ones in the front end. We leverage sparse optical flow to track selected edge features and triangulate them using the initial pose predicted by the Inertial Measurement Unit (IMU). In the back end, we present a novel edge feature residual formulation that differs from the traditional reprojection residual. We tightly couple the new edge residual with the reprojection and IMU preintegration residual to better refine camera poses. We test our PE-VINS on public datasets, and our results show that it outperforms existing point-line-based methods and achieves state-of-the-art VINS performance. The code will be released at https://github.com/BlueAkoasm/PE-VINS.**

***Index Terms*—SLAM, Visual-Inertial SLAM, Edge Features.**

I. Introduction

SIMULTANEOUS localization and mapping (SLAM), which enables mobile robots to perceive and map the unknown surrounding environments, has been a popular research field for several years. Compared to V-SLAM that only leverages visual information, Visual-Inertial Navigation Systems (VINS), also known as visual-inertial SLAM, have witnessed remarkable progress in recent years [1]–[5]. VINS combines visual sensors with an Inertial Measurement Unit (IMU), offering a robust and accurate solution for estimating robot poses. By leveraging IMU data, VINS recovers metric scale and hence achieves precise six-degrees-of-freedom (6DOF) pose estimation, significantly enhancing performance across a wide spectrum of applications in the field of autonomous vehicles, such as robots and self-driving cars.

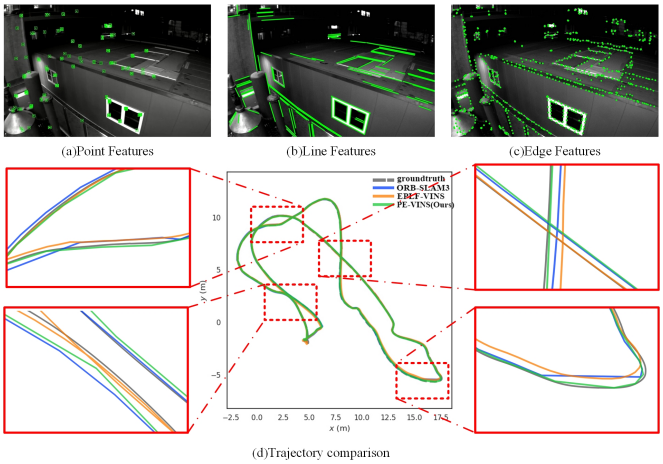


Fig. 1. ORB-SLAM3 and EPLF-VINS vs. PE-VINS (ours). (a) Point features tracked by ORB-SLAM3. (b) Line features tracked by EPLF-VINS. (c) Edge features tracked by our PE-VINS. (d) Estimated trajectory comparison between ORB-SLAM3, EPLF-VINS, and PE-VINS. By leveraging edge features, our PE-VINS improves the localization accuracy of VINS.

VINS, typically based on point features only, has made impressive advancement in many practical scenarios [6]–[9]. However, as the demand for precise positioning continues to rise, enhancing the accuracy and robustness of pose estimation remains a key area of focus in current research. In particular, challenging situations arise in scenes with very few salient point features, in which the localization accuracy of point-only VINS may significantly decrease or even lead to failure. To address these issues, considerable efforts on point-line methods [10]–[14] have been made due to the better robustness and accuracy compared to point-only methods. Point-line methods leverage the additional information provided by lines to enhance VINS performance. Basically, they extract straight lines and formulate error functions based on the distance from endpoints to the matched line. In any case, these methods are still limited in cluttered and complex scenes, as the extracted lines are typically short and clustered together. Additionally, some aspects of line tracking present further challenges, such as its frequent mismatches and high computational footprint [14].

Despite various efforts to advance the state of the art of line features, the results have often been unsatisfactory. Recently, edge features have garnered increasing attention and demonstrated considerable effectiveness. Edges are capable of

This work was supported by the National Natural Science Foundation of China under Grant (U2013203, 62373140, U21A20487, 62103137), National Key R&D Program(2023YFB4704500), the Project of Science Fund for Distinguished Young Scholars of Hunan Province (2021JJ10024); Leading Talents in Science and Technology Innovation of Hunan Province (2023RC1040), the Project of Science Fund of Hunan Province (2022JJ30024); the Project of Talent Innovation and Sharing Alliance of Quanzhou City (2021C062L); the Key Research and Development Project of Science and Technology Plan of Hunan Province (2022GK2014).

[1] Changxiang Liu, Hongshan Yu, Panfei Cheng, Wei Sun are with the Hunan University. [2] Javier Civera is with the University of Zaragoza.. [3] Xieyuanli Chen is with the National University of Defense Technology.

Changxiang Liu and Xieyuanli Chen contributed equally to this work.

(*Corresponding author: Hongshan Yu, yuhongshancn@hotmail.com)

working in most challenging scenarios, such as low texture and lighting variations, and are easy to extract [15]–[17]. Unlike straight lines in line-based methods, edges includes curves as well. However, this also means that edges do not have a complete mathematical model representation like lines. Therefore, we treat edge as a collection of numerous pixel points, and refer each pixel point as edge feature. Recent research [17], [18] has demonstrated that edge features are beneficial for localization and useful in improving the robustness and accuracy of RGB and RGB-D odometry, which inspires us to introduce edge features in VINS. It is, however, worth noting that introducing edge features naively may cause a number of problems. As one of the most critical, the tremendous number of edges that can be tracked may lead to a significant computation increase in tracking and optimization. The main research challenge is reducing the number of extracted edges, in order to track and optimize only the most informative ones, which will improve the localization results while still maintaining real-time performance.

Based on the above, we present PE-VINS, a novel visual-inertial SLAM system able to integrate edge features along with points. Unlike point-line methods, we aim to enhance the monocular system's perception of environmental structure and geometric characteristics by incorporating edge features in visual images into the point-only VINS system and thus provide reliable estimation results, as shown in Fig. 1. In the front end, we extract Canny edges and propose an edge selection to track only the most informative subset of edges due to their abundant quantity. We track the selected edge features between adjacent frames using sparse optical flow and then triangulate them leveraging poses predicted by IMU measurements. In the back-end, unlike normal reprojection residual, we formulate a new residual calculation method for edge features and tightly couple it with visual residual and IMU residual in optimization. Our main contributions are summarized as follows:

- We propose an accurate point-edge monocular VINS, coined PE-VINS, which is an evolution of ORB-SLAM3 [9]. It exploits edge features in images to complement and enhance point-only VINS. Meanwhile, it tightly couples points, edges, and IMU measurements in pose refinement, effectively improving real-time localization accuracy.
- We develop a novel edge selection method for leveraging image gradients and entropy increase to find edge features that are potentially beneficial for pose estimation.
- We present a suitable residual formulation tailored for edge features, which takes into account both the position and orientation of edge features. Moreover, it can be easily integrated into graph optimization with low computational cost.
- PE-VINS is compared with other existing VINS benchmarks, including direct, point-only, and point-line methods, on public datasets in terms of efficiency and localization accuracy. Qualitative and quantitative results show that our method has state-of-the-art (SOTA) performance. Also, the full source code for our method will be released.

II. Related Work

In this section, we first analyze the representative existing VINS methods and then explore edge-based SLAM approaches. These methods are determined based on their conceptual alignment with our technique.

A. Representative VINS methods

Existing VINS are divided into direct and indirect methods.

1) Direct VINS: This type of methods exploits the photometric data to track image pixels and solves pose estimation by directly minimizing their intensity differences between views. ROVIO [19], a direct visual-inertial odometry, employs EKF and photometric errors to update the robot state. VI-DSO [20], which is based on DSO [21] (standing for Direct Sparse Odometry), proposes a direct visual-inertial pipeline that minimizes an energy formulation composed of photometric and IMU measurement errors. DM-VIO [22] presents a delayed marginalization technique to improve scale estimation and relies solely on a single camera and an IMU, managing to surpass even stereo pipelines. Direct methods use photometric errors as residuals, while our method builds residuals based on the correspondences between features. Unlike traditional reprojection residuals, we also leverage the gradient vectors in the proposed residual.

2) Indirect VINS: Contrary to direct methods, indirect VINS extracts and tracks salient image features, such as points or lines, and estimates camera poses by minimizing the reprojection of these feature correspondences tightly coupled with the IMU measurements.

Point-only methods play a crucial role in VINS, as points are easy to extract and can provide stable correspondences. OKVIS [23] and VINS-Mono [8] are both widely accepted tightly coupled VINS based on point features. They use sliding window optimization and bundle adjustment to get accurate and robust poses. Moreover, ORB-SLAM3 [9], considered the current state of the art, is also based on point features only. DynaVINS [24] uses prior pose to reject features from dynamic objects, while Dynam-SLAM [25] constructs and optimizes landmarks based on dynamic features.

Point-line methods have gradually attracted attention due to the robustness in complex environments of line and plane features. PL-VIO [26] and PL-VINS [12] both introduce line features into VINS-Mono to enhance system robustness and estimate accurate pose in real-time. PLS-VINS [27] exploits structural constraints to fuse point and line features and obtains improvement of robustness and accuracy in illumination-changing environments. PLF-VINS [13] leverages point and parallel line fusion to improve localization accuracy. EPLF-VINS [14] modifies the line detection model to obtain high-quality line features for improving localization accuracy and efficiency. Liu et al. [28] predict line features using IMU-assisted optical flow tracking to achieve better feature matching performance in stereo VINS systems. However, line-based VINS methods only extract straight lines in the scenes, while overlooking the role of curves. Also, they typically adopt rejection strategies based on length and angle to filter line features [12] or merge nearby short lines to enhance

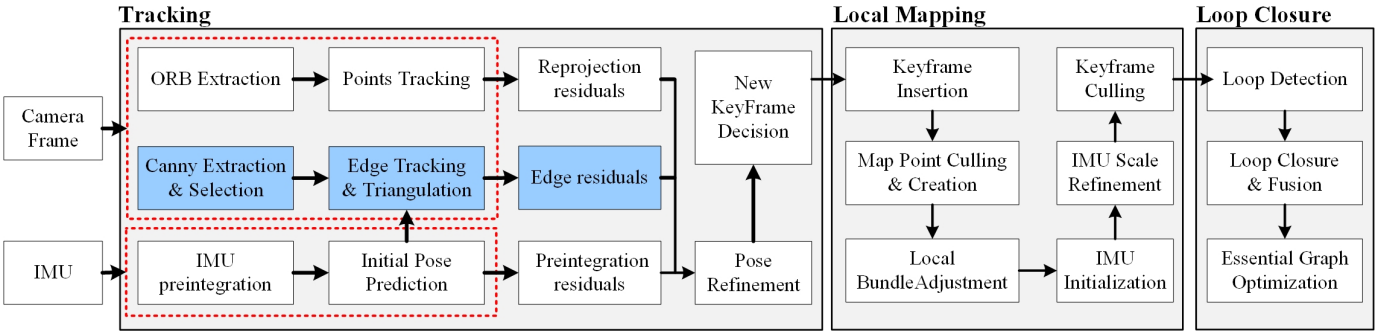


Fig. 2. PE-VINS overview. It basically consists of three threads: *Tracking*, *Local Mapping*, and *Loop Closure*. The blue box indicates the contributions of our method. Note that the front-end processing for point and edge features and the IMU preintegration are done in separate threads, but they are optimized jointly in the back end.

robustness [14]. On the contrary, the edge detection algorithm we use extracts all edges in scenes, including both straight and curved ones. We treat edges as collections of numerous pixel points and then identify potential points (subsequently referred to as edge features) that are beneficial for pose estimation. This eliminates the need to consider the length of the edges.

B. Edge-based methods

Recently, edge-based methods have attracted notable interest due to their ability to offer advantageous characteristics, such as better adaptability to low-texture and illumination-varied scenarios. Klein et al. [29] proposed a SLAM method using non-linear optimization based on edgelet, a locally straight or curved line segments. Edge SLAM [16] proposes an edge-based SLAM pipeline to handle pose estimation in less-textured environments. Tarrio et al. [30] propose to track edges by finding the closest edge using the normal direction, which consumes more computation and is prone to misalignment. Kuse et al. [31] use the distance transform (DT) method [32] to find the closest edge and add DT to the optimization energy formulation. Canny-VO [33] proposes Approximate Nearest Neighbour Fields (ANNF) and Oriented Nearest Neighbour Fields (ONNF) to replace DT in edge registration. Wang et al. [15] combine edges with direct visual odometry to avoid uncertainty of endpoints. RESLAM [34] builds a complete SLAM utilizing edges throughout all stages and achieves remarkable results. EdgeVO [17] proposes a new selection strategy that only selects a small set of edges to balance efficiency and accuracy. [35] also leverages corner and edge points in thermal-depth system to provide a strong solution for SLAM in challenging illumination environments.

Overall, edge-based methods target RGB-D cameras predominantly [17], [30], [33], [34], adopting techniques such as DT or distance field functions to handle 3D-2D edge alignment. However, it is not applicable in monocular systems since depth information of features cannot be directly obtained. Moreover, the presence of a large number of edge features in the visual data severely constrains the efficiency and accuracy of the system. In contrast, we propose a novel edge selection method to filter edges that are potentially beneficial for pose estimation. We subsequently apply sparse optical flow to track edges and leverage initial pose estimates from IMU data to triangulate them. This enables us to address 3D-2D edge alignment through reprojection.

III. System Overview

Fig. 2 shows an overview of our PE-VINS processing pipeline. Its basic structure is inspired by ORB-SLAM3 and targets monocular VINS. It consists of three different threads: *Tracking, Local Mapping, Loop Closure*. The tracking thread runs in the front end and estimates the camera pose for every frame, which is used by the other two threads. The decision to add new keyframes is also made in this tracking thread. The local mapping and loop closure threads work in the back end. The local mapping processes keyframes and refines their poses using local Bundle Adjustment (BA) optimization. Loop closure is responsible for detecting loops and then global optimization to reduce localization drift.

A. Tracking

In the tracking thread, we first preprocess camera frames, including extracting both point and edge features to establish robust correspondences between adjacent frames. In this step, we propose a new method for edge selection to improve the efficiency and robustness of PE-VINS, due to the tremendous number of edges that in general appear in images. We then pre-integrate the IMU measurements to predict the initial pose of camera frames, which is followed by tracking edge features using sparse optical flow. Note that the tracking of point and edge features are conducted simultaneously, and we preserve the original point features processing approach of ORB-SLAM3. Finally, we employ initial pose and correspondences to triangulate edge features and obtain their depth information for optimization in the local mapping thread. More details are provided in the following steps.

1) Frame preprocessing: Compared to ORB-SLAM3, when a new frame comes, we extract both ORB corner points and Canny edges. The Canny algorithm exhibits significant versatility, allowing it to adapt to a wide range of images and challenging scenes, such as low texture and varying illumination. In this process, the Canny algorithm will detect a multitude of redundant edges, potentially resulting in significant computational overhead. Therefore, we develop a

selection method to identify beneficial edge features, with further details provided in Sec. IV-B. Note that we implement the extraction of edges and point features in two separate threads to maintain efficiency.

2) IMU preintegration: IMU preintegration has been extensively discussed in many previous works. We only provide here a brief introduction for completeness. IMU data is influenced by factors including acceleration bias ($\mathbf{b}_a$), gyroscope bias ($\mathbf{b}_w$), and additive noise. The model for the gyroscope and accelerometer raw measurements $\hat{\boldsymbol{\omega}}, \hat{\mathbf{a}}$ at a time t can be written as:

$$\begin{aligned} \hat{\mathbf{a}}_t &= \mathbf{a}_t + \mathbf{b}_{a_t} + \mathbf{n}_a \\ \hat{\boldsymbol{\omega}}_t &= \boldsymbol{\omega}_t + \mathbf{b}_{\omega_t} + \mathbf{n}_\omega, \end{aligned} \tag{1}$$

where $\mathbf{a}_t$ and $\boldsymbol{\omega}_t$ represent their true values without noise and bias. We assume that the accelerometer's and gyroscope's additive noise $\mathbf{n}_a$ and $\mathbf{n}_\omega$ are both Gaussian white noise. For two consecutive frames b_k and b_{k+1}, there is several IMU measurements within a time interval $[t_k, t_{k+1}]$. Integrating this data provides the motion change from b_k to b_{k+1}:

$$\begin{aligned} \alpha_{b_{k+1}}^{b_k} &= \iint_{t\in[t_k,t_{k+1}]} R_t^{b_k}(\hat{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a)dt^2 \\ \beta_{b_{k+1}}^{b_k} &= \int_{t\in[t_k,t_{k+1}]} R_t^{b_k}(\hat{\mathbf{a}}_t - \mathbf{b}_{a_t} - \mathbf{n}_a)dt \\ \gamma_{b_{k+1}}^{b_k} &= \int_{t\in[t_k,t_{k+1}]} \frac{1}{2}\Omega(\hat{\boldsymbol{\omega}}_t - \mathbf{b}_{\omega_t} - \mathbf{n}_\omega)\gamma_t^{b_k}dt, \end{aligned} \tag{2}$$

where

$$\Omega(\boldsymbol{\omega}) = \begin{vmatrix} -[\boldsymbol{\omega}]_\times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^T & 0 \end{vmatrix}, [\boldsymbol{\omega}]_\times = \begin{vmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{vmatrix}. \tag{3}$$

$\alpha_{b_{k+1}}^{b_k}$, $\beta_{b_{k+1}}^{b_k}$, and $\gamma_{b_{k+1}}^{b_k}$ represent changes in position ($\mathbf{p}$), velocity ($\mathbf{v}$), and quaternion ($\mathbf{q}$) respectively.

3) Pose prediction: After the IMU is initialized, the state $\mathbf{p}_{b_{k+1}}^w, \mathbf{v}_{b_{k+1}}^w, \mathbf{q}_{b_{k+1}}^w$ can be propagated in the world coordinate w based on the IMU preintegration within the time interval $\Delta t_k = t_{k+1} - t_k$, as shown in the following equation:

$$\begin{aligned} \mathbf{p}_{b_{k+1}}^w &= \mathbf{p}_{b_k}^w + \mathbf{v}_{b_k}^w \Delta t_k + R_{b_k}^w \alpha_{b_{k+1}}^{b_k} \\ \mathbf{v}_{b_{k+1}}^w &= \mathbf{v}_{b_k}^w + R_{b_k}^w \beta_{b_{k+1}}^{b_k} \\ \mathbf{q}_{b_{k+1}}^w &= \mathbf{q}_{b_k}^w \otimes \gamma_{b_{k+1}}^{b_k}, \end{aligned} \tag{4}$$

where $R_{b_k}^w$ represents the rotation matrix from IMU-body to the world coordinate system, $\otimes$ denotes the quaternion product. Therefore, we use Eq. 4 to obtain an initial pose of the current frame.

4) Edge Triangulation: After selecting part of edge features (mentioned above), we use sparse optical flow to track selected edge features in adjacent frames. Illustrative results can be visualized in Sec. V-A. Once we get the correspondences of edges between the current frame and the last frame, along with the initial pose of the current frame, the same triangulation method as employed for point features can be used to obtain the depth information of edge features.

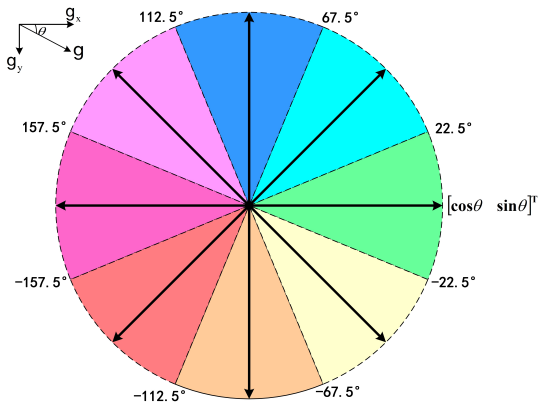


Fig. 3. Eight discretization bins for gradient direction. The gradient direction θ is defined as the angle between g_x and y g_y. Then, we use the middle angle to represent the gradient direction within each bin.

5) Pose Refinement: After obtaining the initial pose, additional correspondences are searched from the local map. Subsequently, the visual residuals established by these correspondences and IMU residuals are tightly coupled in a joint optimization to refine the camera poses. Here, we propose a new residual for edges (for more details, see Sec. IV-B), using reprojection instead of DT for 3D-2D edge alignment. A keyframe decision strategy is also employed during the tracking to determine whether the current frame should be selected as a keyframe (for the keyframe decision details, we refer the reader to the ORB-SLAM3 paper).

B. Local Mapping

The local mapping thread is activated when a new frame is selected as a keyframe. This thread maintains a map that includes all 3D points, keyframes, and their covisibility relationships. It continuously updates keyframe information, adds new map points, and removes low-quality map points. In VINS, this thread is also responsible for IMU initialization. After IMU initialization, the 3D coordinates of point and edge features are transformed into the IMU-body coordinate system. When there is no need to process keyframes, local mapping performs local bundle adjustment (BA) to optimize the poses of all keyframes observed by the current keyframe. Finally, it determines and removes redundant keyframes to control the map size.

C. Loop Closure

The loop closure thread identifies potential loops by computing the similarity between keyframes. Once a loop is successfully detected, an essential graph, including all keyframes within the loop, is established. Then, the global BA optimizes this essential graph to reduce global drift error and yield more accurate poses. We follow ORB-SLAM3 and use DBoW2 to assess the similarity between keyframes and then incorporate edges with the highest gradient, distributed across the visual images, into loop closure.

Algorithm 1 Edge Selection

Input: Edge sets within each image grid $E = \{E_1, E_2, \cdots, E_n\}$, required number of selected edges N
Output: the selected edge set S

1: Initialize $S = \phi$
2: **while** $size(S) < N$ & $E \neq \phi$ **do**
3: **for** each $E_i \in E$ **do**
4: Initialize $S_i = \phi$ $\{S_i =$ a subset of S for $E_i\}$
5: Find initial edges e_h with highest gradients
6: $S_i = S_i \cup e_h$ {Add initial edges}
7: $E_i = E_i - e_h$
8: **while** $size(S_i) < N/size(S)$ & $E_i \neq \phi$ **do**
9: $e = argmax(\rho(e, S_i)), e \in E_i$
10: $S_i = S_i \cup e$ {Select this edge}
11: $E_i = E_i - e$ {Remove this edge from E_i}
12: **end while**
13: $S = S \cup S_i$ {Merge all selected subsets}
14: **end for**
15: **end while**
16: **return** S

IV. Proposed Approach

In this section, we detail how our method works for VINS, including the proposed selection method and the new residual formulation for edges.

A. Edge Selection

The Canny algorithm detects a vast number of edges, but the majority of them are redundant [17]. Utilizing all edge features will inevitably lead to a significant increase in computational load, thus influencing system performance. Most edge-based methods are inclined to discard unnecessary edges through depth information. However, this is hard to be implemented in monocular systems. To address this challenge in a monocular VINS system, inspired by [36] and [37] we propose a novel selection method based on image gradients and entropy increase. Image gradients are calculated in both horizontal and vertical directions. Similar to Canny-VO [33], we divide the direction into several equally wide intervals, each discretization bin spanning of 45 degrees, as shown in Fig. 3. Entropy refers to the measure of information content in an image, with higher entropy indicating greater information content. The entropy for a discrete variable X can be calculated by:

$$H(X) = -\sum_{x_i \in X} p(x_i) \log_2(p(x_i)), \tag{5}$$

where, $p(x)$ represents the probability of a specific value x of the random variable X and the summation is taken over all possible values $x_i \in X$.

Specifically, our algorithm starts by partitioning an image into equally sized grids and then conducts edge selection in each separate grid. Firstly, we choose a certain number of edge features with the highest gradient magnitudes to build an initial set S due to their superior signal-to-noise ratio, which is found beneficial for motion estimation [17]. Meanwhile, we store the gradient magnitudes and directions of the selected edge features. Secondly, we iteratively compute the entropy increase before and after adding each edge feature based on gradient directions. In this process, the maximum entropy occurs when the selected features have gradients in all directions. This is because entropy, by definition, is related to the dispersion of features. If the selected edge features have various gradient directions, their differences will be relatively large, resulting in a higher entropy. So, the $p(x_i)$ in Eq. 5 represents the ratio of the number of edge features in each discretization bin to the total number of selected edge features. When computing entropy increases, we also consider that edge features with larger gradient magnitudes are more likely to be selected. As a result, we select edge features that maximize the following function:

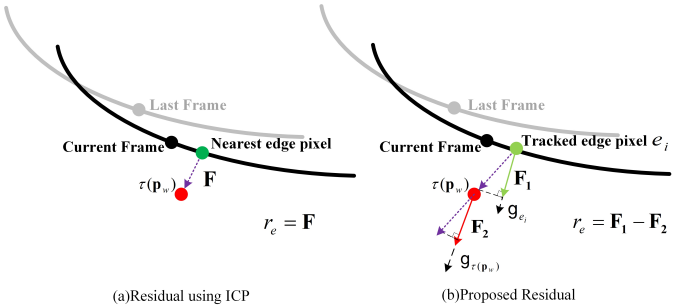


Fig. 4. (a) The ICP-based method. The residual is built by the Euclidean distance between the reprojection edge and its nearest edge feature, which is found by using DT. (b) The proposed residual in this paper. We track the edge feature (the green point) from the last frame to the current frame using KLT. Then, reproject the corresponding 3D edge to the current frame (the red point). The residual is calculated based on Eq. 8.

$$\rho(e, S) = \sigma(S \cup e)H(S \cup e) - \sigma(S)H(S), \tag{6}$$

where the function $\sigma(S)$ represents the average gradient magnitude weight of edge set S. Finally, we iteratively add edge features until reaching a certain number of selected edge features in S within each grid. The implementation of our method is summarized in Algorithm 1.

In Algorithm 1, we select edge features in each grid (Step 3) and firstly initial the selected set using three edge features with the highest gradients (Step 5,6). Then, we iteratively find the edge feature that maximizes the function in Eq 6 (Step 9). Next, we add it to the selected subset S_i (Step 10) and remove it from the original set E_i (Step 11). Finally, the algorithm repeats in the next grid if the size of S_i reaches the requirement or there is no edge in E_i.

It is noticeable that selecting edges in various directions not only facilitates edge matching but also avoids the clustering of selected edges, which is demonstrated to be detrimental to pose estimation in several previous works [8], [9], [17], [36]. Moreover, while KLT adopts the gradients in two directions to match features, edge points generally have large gradients in one direction. Our proposed selection method ensures the diversity of gradient directions in each image grid, which will help address this problem.

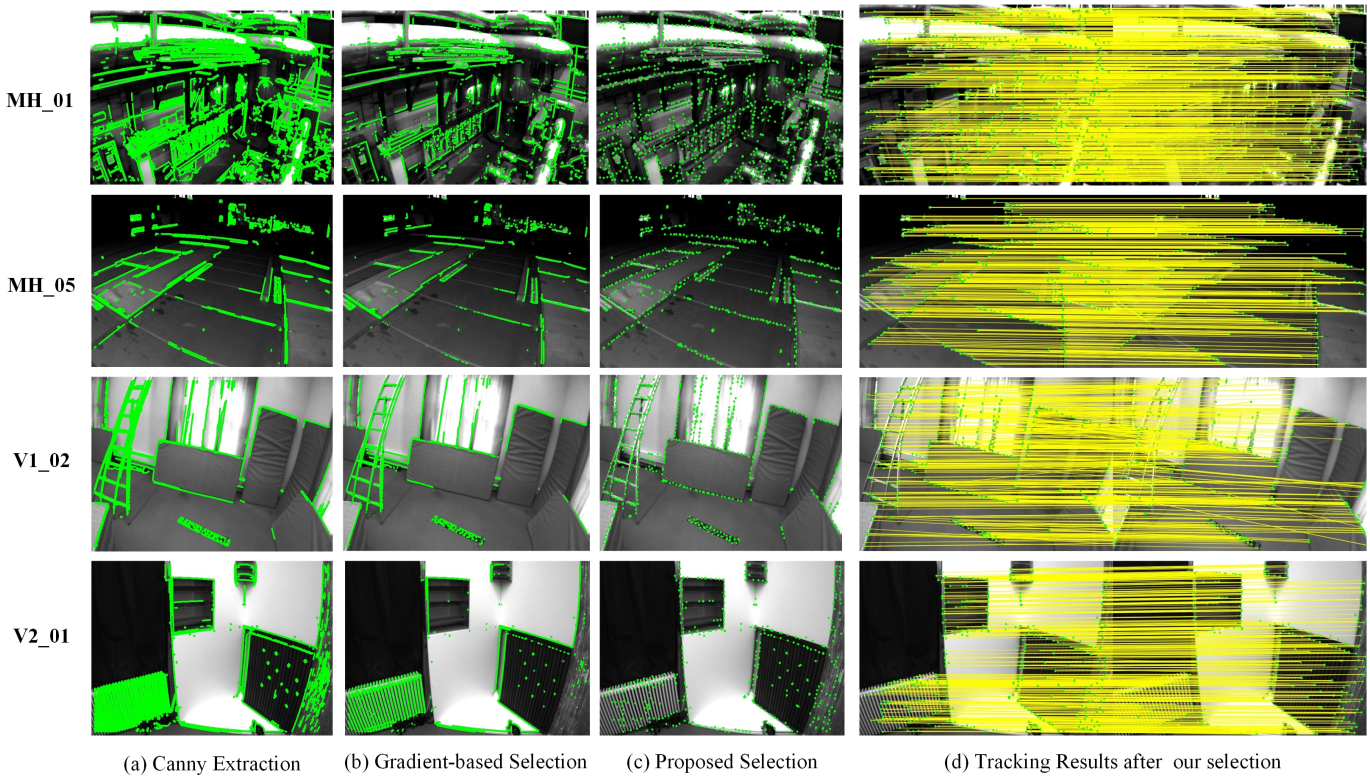


Fig. 5. The visualization and comparison of three edge selection methods on 4 sequences of EuRoC dataset. (a) is the result obtained by the Canny algorithm, (b) is the edge selection based only on image gradients, and (c) is our proposed edge selection method. (d) is the tracking results using optical flow after our proposed selection. Our proposed method avoids feature clustering and reduces the number of edges, achieving good tracking results.

B. Edge Residual

In VINS, visual projection error and IMU perintegration residual are tightly coupled to optimize initial poses. In this paper, we introduce edge features into a traditional point feature-based VINS system, so we incorporate edge residual into the optimization model to better refine poses. The total optimization model can be written as:

$$\min_{\chi}\Big(\sum_{k\in B}\|r_b\|^2_{\sum b_k b_{k+1}}+\sum_{i,j\in F}\|r_f\|^2_{\sum^{c_i}_{f_j}}+\sum_{i,j\in E}\|r_e\|^2_{\sum^{c_i}_{f_j}}\Big), \quad (7)$$

where χ represents the full system state, r_b is the IMU preintegration residual between b_k and b_{k+1}, r_f is the point feature visual reprojection residual, r_e is the edge residual and $\|\cdot\|$ refers to 2-norm.

In other edge-based methods [17], [33], edge correspondences are established by the iterative closest points-based (ICP) algorithm [38], which tends to find the nearest edge pixel through distance transform (DT) [32]. Moreover, residuals are built by computing the distance between the target edge pixel and its closest edge pixel (Fig. 4 (a)). In cases where edges are not tracked, this ICP-based approach proves effective by treating the nearest pixels in other frames as correspondences. However, the process of finding nearest neighbors consumes much time, especially with a large number of edges. Meanwhile, in continuous motion estimation, establishing correspondences between adjacent frames is readily achievable using optical flow tracking aforementioned in tracking. Hence, we propose a new formulation for edge residual using the gradient of edges, which takes into account both the edge position and orientation in the minimization. It is given by the following formulation:

$$r_e = (g_{e_i} - g_{\tau(\mathbf{p}_w)})^T(e_i - \tau(\mathbf{p}_w)), \quad (8)$$

$$\tau(\mathbf{p}_w) = \pi(T_b^c T_w^b \oplus \mathbf{p}_w), \quad (9)$$

where $\mathbf{p}_w$ represents the 3D edge in the world coordinate frame obtained by triangulation, and its observation in frame i is e_i. $g(e)$ is the gradient vector of edge feature e. In the reprojection function $\tau(\mathbf{p}_w)$, $\pi(\cdot)$ denotes the projection function for the camera model used, $T_b^c, T_w^b \in SE(3)$ is the transformation matrix from IMU-body to the camera and from world to IMU-body, respectively. $SE(3)$ is the Lie algebra and $\oplus$ indicates the $SE(3)$ operation of the transformation matrix.

The illustration of Eq. 8 is shown in Fig. 4 (b). It implies that the ideal scenario occurs when the gradient of the reprojected edge is the same as that of its observation or when their Euclidean distance is equal to zero. Note that this residual does not introduce any substantial computation load, even when dealing with a large number of edges, as the image gradient and correspondences of edges have been computed and stored in the previous steps. Furthermore, to accelerate convergence, the gradient component in Eq. 8 employs normalized vectors in eight different directions, as illustrated in Fig. 3.

C. Implement Details

During the proposed edge selection progress, we divide the image into $n \times n$ equally sized grids and select N

TABLE I
RMSE [M] OF ATE AND RUNTIME [S] COMPARISON. C USES ALL CANNY EDGES, AND G REPRESENTS THE GRADIENT-BASED SELECTION METHOD.

Dataset	RMSE [m]			Runtime (s)		
	C	G	Ours	C	G	Ours
MH_01	0.038	0.036	**0.033**	0.381	0.233	**0.122**
MH_02	0.050	0.052	**0.047**	0.383	0.229	**0.126**
MH_03	0.047	0.052	**0.040**	0.289	0.211	**0.118**
MH_04	0.057	0.108	**0.045**	0.262	0.173	**0.109**
MH_05	0.061	0.068	**0.047**	0.308	0.177	**0.111**
V1_01	0.092	0.090	0.090	0.289	0.170	**0.116**
V1_02	0.062	0.063	**0.060**	0.213	0.141	**0.100**
V1_03	0.161	0.068	**0.065**	0.157	0.114	**0.099**
V2_01	0.084	0.086	**0.073**	0.298	0.169	**0.108**
V2_02	0.060	0.058	**0.055**	0.260	0.153	**0.105**
V2_03	**0.059**	0.067	0.060	0.173	0.125	**0.098**
Average	0.07	0.068	**0.056**	0.274	0.172	**0.110**

TABLE II
OPTIMIZATION TIME AND LOCALIZATION ERRORS (RMSE) COMPARISON BETWEEN ORIGINAL REPROJECTION RESIDUAL AND PROPOSED RESIDUAL.

Dataset	RMSE[M]		Time(ms)	
	Original	Ours	Original	Ours
MH_01	0.033	**0.031**	14.3	**13.6**
MH_02	0.052	**0.045**	**12.4**	12.9
MH_03	0.048	**0.040**	12.0	**11.3**
MH_04	0.065	**0.045**	12.7	**10.6**
MH_05	0.052	**0.047**	11.3	**10.9**
V1_01	0.093	**0.090**	14.1	**13.7**
V1_02	0.061	**0.060**	9.3	**8.9**
V1_03	0.085	**0.065**	11.1	**9.7**
V2_01	**0.060**	**0.060**	**11.8**	**11.8**
V2_02	0.058	**0.055**	**10.3**	**10.3**
V2_03	0.066	**0.060**	**8.2**	8.4
Average	0.061	**0.054**	11.6	**11.1**

edge features in each grid, as defined in Algorithm 1. We set $n = 20$ and $N = 8$ in our experiments. To avoid clustering edge features, we also remove edges with Euclidean distances too close to the selected edge features. We perform triangulation on tracked edge features between adjacent frames and check the depth information of 3D edge features and their reprojection errors on the two frames. Similar to ORB-SLAM3, we then remove those 3D edge features from the map with depths below 0 or large reprojection errors. In the graph optimization, we compute the residuals in Eq. 8 between the 3D edge features observed by the current frame and the tracked correspondences. Here our proposed residual not only considers the position of edge features but also incorporates their image gradients. Then, we tightly couple the proposed residuals with point feature reprojection residuals and IMU preintegration residuals to refine camera poses.

V. EXPERIMENTS

We assess the performance of PE-VINS on public datasets, focusing on efficiency and localization accuracy. We first evaluate the effectiveness of our proposed selection method. Then, we compare the optimization time using the original reprojection residuals to our proposed edge residuals. Finally, we conduct a thorough evaluation by comparing PE-VINS with other state-of-the-art (SOTA) VINS, including direct methods, point-only methods, and point-line methods on EuRoC [39] and TUM-VI [40] datasets to demonstrate the superiority of our method. These two datasets contain visual images and synchronized IMU measurements and provide ground truth for the states. We use the Absolute Trajectory Error (ATE) to evaluate the error between the estimated trajectory and the ground truth. The Root Mean Square Error (RMSE) of ATE can be computed as

$$RMSE_{ATE} = (\frac{1}{m}\sum_{i=1}^{m}\|trans(X_i)\|^2)^{\frac{1}{2}}, \qquad (10)$$

where X_i represents the difference between the estimated trajectory and the corresponding ground truth. The operator $trans(\cdot)$ denotes that we only consider the translation errors, not rotation errors.

We implemented PE-VINS in C++ and used ROS for message transmission. The hardware platform for testing and evaluating all methods in all experiments includes an AMD Ryzen 7 @ 3.40GHz CPU and 16GB of memory. The entire pipeline runs on the Ubuntu 18.04 operating system.

A. Selection

Fig. 5 shows the results of our edge selection method, in comparison with a selection method that is based only on image gradient magnitudes [18], [33], [35]. As can be observed, only relying on gradient magnitudes for edge selection results in edge clustering, adversely affecting matching and tracking. In contrast, our method effectively spreads the selected edges over various areas in the image. Furthermore, we reduce the number of edge features without increasing the computational cost to avoid feature redundancy. For better comparison, we also provide the localization results and time costs per frame of three edge selection methods in Table I, demonstrating that our selection improves the effectiveness and achieves better performance in pose estimation. From these results, we can see that the ATE results using all Canny edges are similar to those based on image gradients, proving that the edges' effectiveness is not directly related to their gradient magnitudes. Additionally, smaller computational footprints give more time for optimizations to converge to better values. On the contrary, using a more reasonable selection method will significantly improve pose estimation performance while reducing the number of edges. Our proposed method mainly leverages the gradient directions to ensure entropy and information maximization and also calculates the distance between adjacent edges to ensure distribution discretization. Furthermore, our method can greatly reduce the time costs per frame to enhance efficiency. These results demonstrate the superiority of our method.

B. Efficiency

We evaluate the average optimization time of the original reprojection residuals and our proposed edge residuals (mentioned in Sec. IV-B). We record the optimization time and the estimation errors in all sequences of the EuRoC dataset, shown in Table II. The comparison results show that our technique not

TABLE III
RMSE[M] OF ATE FOR SEVERAL VINS BASELINES AND OUR PE-VINS ON EUROC. BOLD AND UNDERLINED RESULT RESPECTIVELY REPRESENT THE HIGHEST AND SECOND HIGHEST ACCURATE RESULT IN THE SAME SEQUENCE. PE-VINS OBTAINS THE BEST RESULTS IN 7 OF 11 SEQUENCES.

		MH_01	MH_02	MH_03	MH_04	MH_05	V1_01	V1_02	V1_03	V2_01	V2_02	V2_03	Avg.
Direct	ROVIO	0.308	0.316	0.411	0.789	1.053	0.156	0.194	0.170	0.269	0.569	0.188	0.402
	VI-DSO	0.074	0.044	0.124	0.112	0.121	0.109	0.067	0.096	0.075	0.062	0.204	0.099
Point-only	OKVIS	0.292	0.361	0.267	0.366	0.396	0.090	0.122	0.196	0.168	0.182	0.305	0.250
	VINS-Mono	0.177	0.183	0.404	0.394	0.382	0.146	0.311	0.329	0.124	0.277	0.323	0.277
	ORB-SLAM3	0.044	0.054	0.043	0.070	0.057	0.090	0.062	0.066	0.071	0.058	0.066	0.062
Point-line	PL-VINS	0.072	**0.044**	0.067	0.083	0.128	**0.043**	0.059	0.180	**0.054**	0.080	0.119	0.084
	PLF-VINS	0.056	0.052	0.071	0.090	0.101	0.045	0.047	0.119	0.060	0.078	0.178	0.082
	EPLF-VINS	0.039	0.072	0.047	0.125	0.089	0.046	**0.034**	0.123	0.096	0.058	0.149	0.080
	PLPL-VIO	0.130	0.074	0.173	0.194	0.295	0.055	0.076	0.074	0.067	0.081	0.133	0.123
	HTSR-VIO	0.111	0.079	0.202	0.208	0.242	0.062	0.076	0.164	0.093	0.133	0.169	0.140
Point-Edge	PE-VINS(Ours)	**0.031**	0.045	**0.040**	**0.045**	**0.047**	0.090	0.060	**0.065**	0.060	**0.055**	**0.060**	**0.054**

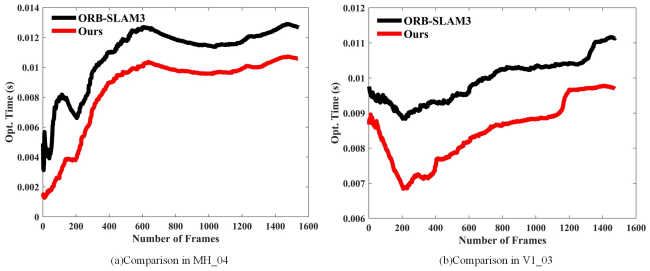


Fig. 6. Optimization time comparison in (a) MH_04 and (b) V1_03 sequences. The optimization time of our proposed method is obviously lower than that of the original method.

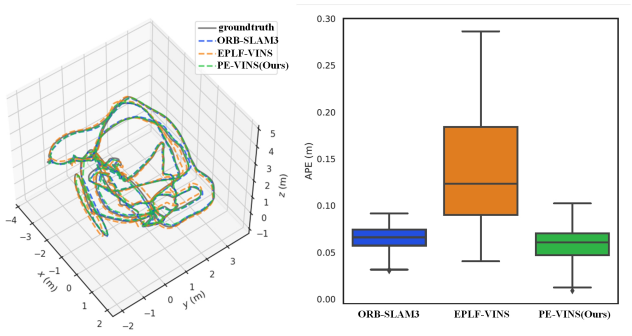


Fig. 7. Trajectory and localization accuracy comparison between ORB-SLAM3, EPLF-VINS and our PE-VINS on Euroc's sequence V2_03. PE-VINS' trajectory is closer to the ground truth and outperforms the baselines.

only improves the efficiency of optimization but also reduces the localization errors. In some sequences, such as MH_04 and V1_03, our method achieves a 16.5% and 12.9% improvement in per-frame optimization time, respectively. Moreover, we achieved an average improvement of 11.5% in ATE compared to the original method. For better visualization, we plot the specific per-frame optimization time on two sequences of EuRoC datasets, shown in Fig. 6. In both sequences, our method consistently has lower optimization times across nearly 1600 frames compared to the original method. The results demonstrate that introducing image gradients into the residual allows for reaching better values more quickly within a limited number of optimization iterations.

C. Localization Accuracy

1) Results on EuRoC dataset: In this section, we compare and evaluate PE-VINS with eight visual-inertial benchmarks in terms of localization accuracy on EuRoC datasets, including direct methods (ROVIO [19], VI-DSO [20]), point-only methods (OKVIS [23], VINS-Mono [8], ORB-SLAM3 [9]) and point-line methods (PL-VINS [12], PLF-VINS [13], EPLF-VINS [14], PLPL-VIO [41], HSTR-VIO [42]), which is shown in Table III. Note that the light changes obviously in some sequences, such as MH_04, MH_05, and V1_03. However, our method consistently achieves superior results due to the integration of edge features as additional constraints. Also, we provide the trajectory comparison for better visualization shown in Fig. 7. In the figure, we compare our PE-VINS with the point-only system ORB-SLAM3 and the point-line system EPLF-VINS. It is obvious that our trajectories are closer to the ground truth and produce lower ATE errors.

Overall, the quantitative results demonstrate that PE-VINS outperforms ORB-SLAM3 in 9 out of 11 sequences, showing an average improvement in accuracy of 12.9% over ORB-SLAM3. Furthermore, PE-VINS achieves state-of-the-art (SOTA) performance in 7 sequences, better than point-line methods, which validates the effectiveness of edge features.

2) Results on TUM dataset: We also provide results on the public TUM-VI benchmark. The data is collected by using a fish-eye camera and an aligned IMU. The ground truth of the TUM-VI dataset is only available for the start and end segments, which comprise only a small portion of the trajectory. Most sequences have considerable length, and the viewpoints change frequently, making it hard to establish loops. Therefore, similar to ORB-SLAM3, we use this ground truth to measure the accumulated drift along the whole trajectory. Quantitative results of the experiments are presented in Table IV. We compare our PE-VINS with VINS-Mono, PL-

TABLE IV
RMSE[M] OF ATE FOR THREE VINS BASELINES AND OUR PE-VINS ON TUM-VI. × STANDS FOR TRACKING FAILURE.

Dataset	VINS-Mono	PL-VINS	ORB-SLAM3	PE-VINS (ours)
corridor1	0.63	1.47	0.23	**0.11**
corridor2	0.95	8.17	0.06	**0.05**
corridor3	1.56	2.00	**0.40**	0.50
corridor4	0.25	0.59	0.20	**0.15**
corridor5	0.77	×	0.06	**0.05**
magistrale1	2.19	0.86	0.20	**0.09**
magistrale2	3.11	×	**0.46**	0.67
magistrale3	**0.40**	3.27	3.58	3.03
Average	1.23	2.73	0.65	**0.58**

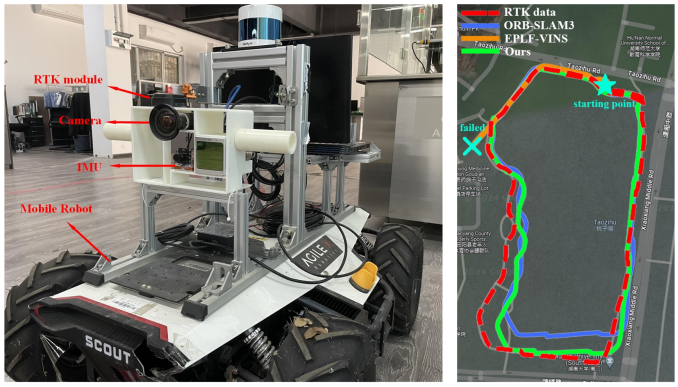
VINS and ORB-SLAM3. We also test EPLF-VINS [14] on the TUM-VI benchmark, but it only succeeds in the corridor4 sequence in our own platform. Our technique achieves the best localization results in 5 out of 8 sequences and shows an average improvement of 10.77% over ORB-SLAM3.

D. Real-world Experiment

In addition to the public dataset, we also implement experiments in a real-world environment to demonstrate the superiority of our method compared to the point-only method ORB-SLAM3 and the point-line method EPLF-VINS. We collect data using our own hardware platform, which is detailed below.

1) Mobile hardware platform: The platform, shown in Fig. 8(a), comprises a mobile robot (AglieX_SCOUT_V1), a monocular camera (Daheng MER2-502, 20Hz), an IMU (Xsens-MTi30, 200Hz) and an RTK module (used to provide ground-truth). The input image size is 612×512, and both images and IMU data are recorded through ROS. To ensure precision in evaluation, we controlled the robot to move at a normal speed, started from the starting point, and then returned to it to build a loop. We circled around the Taozihu, a lake whose total length is approximately 1300m, to collect real-world data.

2) Localization capability comparison: Our collected dataset encompasses multiple real-world challenges, including scenarios with pedestrians, trees, roads, and lakes, and various lighting conditions. We evaluate ORB-SLAM3, EPLF-VINS and our method on this dataset. The trajectory comparison results along the ground-truth are shown in Fig. 8(b). For clear visualization, we align these trajectories with Google Maps. From the figure, it can be seen that both ORB-SLAM3 and our PE-VINS successfully completed this experiment, while EPLF-VINS failed. Our trajectory is closer to the ground-truth in the long-duration test. In contrast, the drift of ORB-SLAM3 becomes larger and deviates from the actual routine. Line features underperform in large outdoor scenes due to the lack of hand-made structures. Meanwhile, outdoor line features are often short and fragmented, which makes it hard to establish robust correspondences. However, edge features are sufficient and our selection method ensures the uniform distribution of edge features across the image, which improves the localization results. Overall, this real-world experiment demonstrates that our method can maintain superior performance.



(a)Mobile Experiment Platform (b)Trajectory Comparison

Fig. 8. Real-world hardware platform and trajectory comparison results. (a) The mobile robot platform includes a monocular camera, an IMU and an RTK module. (b) The estimated trajectory comparison between ORB-SLAM3, EPLF-VINS and our method. The ground-truth is given by RTK. EPLF-VINS fails halfway, and our method is closer to the RTK result than ORB-SLAM3.

TABLE V
RUNTIME (MS) COMPARISON OF EACH IMPORTANT MODULE BETWEEN ORB-SLAM3 AND OURS

Thread	Module	ORB-SLAM3	Ours
Tracking	Feature Ext. & Sel.	13.79	16.95
	Pose Prediction	0.08	0.09
	Feature Tracking	7.79	14.66
	Total	24.47	34.59
Local Mapping	Map Point Creation	35.05	42.99
	Local BundleAdjustment	140.42	125.00
	Total	219.75	198.73

E. Runtime Analysis

We test the runtime of Tracking and Local Mapping threads in ORB-SLAM3 and our method. The results are shown in Table V. The total tracking time of ORB-SLAM3 and Ours is 24.47 ms and 34.59 ms, respectively. Although introducing thousands of extra features extends the extraction and tracking duration, our proposed method has no huge impact on the total time consumption. We remain capable of real-time operation, of which frequency exceeds the 20Hz camera rate.

VI. CONCLUSION

This paper introduces PE-VINS, an accurate monocular visual-inertial SLAM system based on point and edge features. We incorporate thousands of edge features which includes features typically overlooked by line-based methods such as curves, into the original system to enhance its perceptual capabilities. To ensure the efficiency and real-time performance of the system, we propose a novel edge selection method and an edge residual formulation. Specifically, we introduce entropy increase to select beneficial edges rather than directly based on gradient magnitudes. Also, we add image gradient parameters into the original reprojection residual to expedite convergence, thus reducing the optimization time. Through quantitative experiments, these innovations have demonstrated

their effectiveness in reducing time costs and improving localization accuracy. By leveraging edge features, our method achieves SOTA localization results compared to eight VINS baselines in the literature, including direct methods, point-only VINS methods, and point-line VINS methods on two datasets. Real-world experiments also validate the practical significance of our approach. There are also limitations in our work. For example, the 3D edge features have not been fully reused, and the sparse map including 3D edge features has not been perfectly maintained. In future work, we plan to explore additional information-based methods to further improve the reuse and maintenance of edge features in 3D maps. Also, a continuously updated information map will be developed to dynamically adjust the weights of different types of features.

REFERENCES

[1] M. Ramezani and K. Khoshelham, "Vehicle positioning in GNSS-deprived urban areas by stereo visual-inertial odometry," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 208–217, 2018.

[2] X. Ji, G. Sun, M. J. Er, and Z. Wang, "Adaptive correction of landmark for visual homing in mobile vehicles," *IEEE Transactions on Intelligent Vehicles*, 2022.

[3] C. Liu, H. Yu, Q. Fu, X. Chen, N. Akhtar, and Z.-H. Mao, "IMPS: Informative Map Point Selection for Visual-Inertial SLAM," *IEEE Transactions on Intelligent Vehicles*, 2024.

[4] P. Shi, Z. Zhu, S. Sun, Z. Rong, X. Zhao, and M. Tan, "Covariance Estimation for Pose Graph Optimization in Visual-Inertial Navigation Systems," *IEEE Transactions on Intelligent Vehicles*, 2023.

[5] C. Shu and Y. Luo, "Multi-modal feature constraint based tightly coupled monocular visual-lidar odometry and mapping," *IEEE Transactions on Intelligent Vehicles*, 2022.

[6] V. Usenko, N. Demmel, D. Schubert, J. Stückler, and D. Cremers, "Visual-inertial mapping with non-linear factor recovery," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 422–429, 2019.

[7] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 1689–1696.

[8] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[9] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[10] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "PL-SLAM: Real-time monocular visual SLAM with points and lines," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 4503–4508.

[11] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "PL-SLAM: A stereo SLAM system through the combination of points and line segments," *IEEE Transactions on Robotics*, vol. 35, no. 3, pp. 734–746, 2019.

[12] Q. Fu, J. Wang, H. Yu, I. Ali, F. Guo, Y. He, and H. Zhang, "PL-VINS: Real-time monocular visual-inertial SLAM with point and line features," *arXiv preprint arXiv:2009.07462*, 2020.

[13] J. Lee and S.-Y. Park, "PLF-VINS: Real-time monocular visual-inertial SLAM with point-line fusion and parallel-line fusion," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7033–7040, 2021.

[14] L. Xu, H. Yin, T. Shi, D. Jiang, and B. Huang, "EPLF-VINS: Real-Time Monocular Visual-Inertial SLAM With Efficient Point-Line Flow Features," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 752–759, 2022.

[15] X. Wang, W. Dong, M. Zhou, R. Li, and H. Zha, "Edge Enhanced Direct Visual Odometry," in *BMVC*, 2016.

[16] S. Maity, A. Saha, and B. Bhowmick, "Edge SLAM: Edge points based monocular visual SLAM," in *Proceedings of the IEEE international conference on computer vision workshops*, 2017, pp. 2408–2417.

[17] H. Zhao, J. Shang, K. Liu, C. Chen, and F. Gu, "EdgeVO: An Efficient and Accurate Edge-based Visual Odometry," *arXiv preprint arXiv:2302.09493*, 2023.

[18] C. Kim, P. Kim, S. Lee, and H. J. Kim, "Edge-based robust RGB-D visual odometry using 2-D edge divergence minimization," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

[19] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 2015, pp. 298–304.

[20] L. Von Stumberg, V. Usenko, and D. Cremers, "Direct sparse visual-inertial odometry using dynamic marginalization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2510–2517.

[21] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[22] L. Von Stumberg and D. Cremers, "DM-VIO: Delayed marginalization visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1408–1415, 2022.

[23] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[24] S. Song, H. Lim, A. J. Lee, and H. Myung, "Dynavins: a visual-inertial slam for dynamic environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 523–11 530, 2022.

[25] H. Yin, S. Li, Y. Tao, J. Guo, and B. Huang, "Dynam-slam: An accurate, robust stereo visual-inertial slam method in dynamic environments," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 289–308, 2022.

[26] Y. He, J. Zhao, Y. Guo, W. He, and K. Yuan, "PL-VIO: Tightly-coupled monocular visual–inertial odometry using point and line features," *Sensors*, vol. 18, no. 4, p. 1159, 2018.

[27] G. Yang, Q. Wang, P. Liu, and C. Yan, "PLS-VINS: Visual inertial state estimator with point-line features fusion and structural constraints," *IEEE Sensors Journal*, vol. 21, no. 24, pp. 27 967–27 981, 2021.

[28] X. Liu, S. Wen, and H. Zhang, "A Real-time Stereo Visual-Inertial SLAM System Based on Point-and-Line Features," *IEEE Transactions on Vehicular Technology*, 2023.

[29] G. Klein and D. Murray, "Improving the agility of keyframe-based SLAM," in *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part II 10*, 2008, pp. 802–815.

[30] J. J. Tarrio and S. Pedre, "Realtime Edge-Based Visual Odometry for a Monocular Camera," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[31] M. Kuse and S. Shen, "Robust camera motion estimation using direct edge alignment and sub-gradient method," in *2016 IEEE international conference on robotics and automation (ICRA)*, 2016, pp. 573–579.

[32] H. Breu, J. Gil, D. Kirkpatrick, and M. Werman, "Linear time euclidean distance transform algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 5, pp. 529–533, 1995.

[33] Y. Zhou, H. Li, and L. Kneip, "Canny-VO: Visual odometry with rgb-d cameras based on geometric 3-d–2-d edge alignment," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 184–199, 2018.

[34] F. Schenk and F. Fraundorfer, "RESLAM: A real-time robust edge-based SLAM system," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 154–160.

[35] X. Chen, W. Dai, J. Jiang, B. He, and Y. Zhang, "Thermal-Depth Odometry in Challenging Illumination Conditions," *IEEE Robotics and Automation Letters*, 2023.

[36] A. Fontan, J. Civera, and R. Triebel, "Information-driven direct RGB-D odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4929–4937.

[37] A. Fontan, R. Giubilato, L. Oliva, J. Civera, and R. Triebel, "SID-SLAM: Semi-Direct Information-Driven RGB-D SLAM," *IEEE Robotics and Automation Letters*, 2023.

[38] I. Dryanovski, R. G. Valenti, and J. Xiao, "Fast visual odometry and mapping from RGB-D data," in *2013 IEEE international conference on robotics and automation*, 2013, pp. 2305–2310.

[39] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[40] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The TUM VI benchmark for evaluating visual-inertial odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1680–1687.

[41] Z. Xu, H. Wei, F. Tang, Y. Zhang, Y. Wu, G. Ma, S. Wu, and X. Jin, "PLPL-VIO: a novel probabilistic line measurement model for point-line-based visual-inertial odometry," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 5211–5218.

[42] B. Zhang, Y. Wang, W. Yao, and G. Sun, "Htsr-vio: Real-time line-based visual-inertial odometry with point-line hybrid tracking and structural regularity," *IEEE Sensors Journal*, 2024.

Changxiang Liu was born in China. He is currently pursuing the Ph.D. degree with the National Engineering Laboratory for Robot Visual Perception and Control, College of Electrical and Information Engineering, Hunan University, China, under the supervision of Prof. Hongshan Yu. His research interests include mobile robot, visual SLAM and computer vision.

Hongshan Yu received the B.S., M.S., and Ph.D. degrees in control science and technology in electrical and information engineering from Hunan University, Changsha, China in 2001, 2004, and 2007, respectively. From 2011 to 2012, he was a Post-Doctoral Researcher with the Laboratory for Computational Neuroscience, University of Pittsburgh, USA. He is currently a Professor with Hunan University.

Panfei Cheng received the B.S. degree from Hunan University, Changsha, China, in 2022. He is currently pursuing the Ph.D. degree with the College of Electrical and Information Engineering, Hunan University, China, under the supervision of Prof. Hongshan Yu. His research interests include point cloud registration, SLAM and deep learning with geometry.

Wei Sun received the B.S., M.S., and Ph.D.degrees from the Department of Automation Engineering, Hunan University, Changsha, China, in 1997, 1999, and 2003, respectively. He is currently a Professor with Hunan University and the Academic Leader of the National Engineering Research Center of Robot Visual Perception and Control Technology. His research interests include computer vision, robotics, neural networks, and intelligent control, with over 80 publications in these areas.

Javier Civera is Associate Professor at the University of Zaragoza in Spain, in which he teaches master courses on Machine Learning and SLAM. He is also vice-director of the Aragón Institute or Research in his institution. His research interests are within 3D vision, having co-authored more than 100 scientific publications in this topic. Currently, he serves as Editor for IEEE Transacions on Robotics and IEEE Robotics and Automation Letters, and as Associate Editor at the International Journal of Robotics Research.

Xieyuanli Chen is now an Associate Professor at the National University of Defense Technology. He also serves as Associate Editor for IEEE RA-L, ICRA, and IROS. He received his Ph.D. degree at the Photogrammetry and Robotics Laboratory, University of Bonn. He received his Master degree in Robotics in 2017 at the National University of Defense Technology, China. He received his Bachelor degree in Electrical Engineering and Automation in 2015 at Hunan University, China.