Enhancing Visual Navigation Performance by Prior Pose-Guided Active Feature Points Distribution

Liqiang Wang, Tisheng Zhang, Yan Wang, Hailiang Tang, and Xiaoji Niu

Abstract— **Feature points are the principal measurements employed in visual navigation, and their tracking qualities significantly affect the localization accuracy of visual-based navigation systems. However, the current feature extraction methods do not take into account the expected tracking quality of the features in the subsequent steps, which may result in less tracking length or limited parallax of the features. Such expected feature tracking quality can be inferred using the upcoming pose information of the carrier, which is a known factor for robots and autonomous vehicles through path planning. Based on the typical grid-based feature distribution method, this article proposes a prior pose-guided active feature points distribution (P2GD) method to redistribute the feature number in different image regions. In the proposed method, the prior poses of the carrier and the 3-D environment that represented by the sparse triangulated features are utilized to predict the future tracking of the features. The feature numbers for each image region are reassigned according to the predicted tracking parallax. The proposed method is implemented in a multistate constraint Kalman filter (MSCKF)-based visual–inertial navigation system (VINS) and evaluated with the public dataset and our private robot dataset. The experiment results indicate that the proposed method can effectively enhance the average tracking parallax of the features and improve navigation performance significantly. The robustness tests also confirm that intentionally introducing certain noises in the prior poses does not hinder the proposed method. Despite these perturbations, the proposed method still demonstrates superior navigation accuracy than the conventional grid-based method.**

Index Terms— **Feature distribution, feature parallax, tracking quality, visual feature point, visual–inertial navigation.**

I. Introduction

REAL-TIME, continuous and robust high-precision positioning, as the basis for intelligent unmanned systems, has substantial application requirements. Among many positioning sensors, cameras are widely used because of their low cost, small size, and strong applicability [1]. Therefore, visual navigation has aroused a lot of research interest, which includes direct method visual localization and feature points-based visual localization [2], [3]. The feature points-based method exhibits stronger robustness and higher accuracy compared with the direct method. As a result, the feature points-based method has been employed in many visual navigation systems, including visual odometry, visual simultaneous localization and mapping (SLAM), and visual inertial odometry systems. In these systems, the feature points, as the visual observation information, form constraints between multiple camera frames and are used to estimate the navigation state. Valid observation can accurately calculate the feature's 3-D position and constrain navigation status. Long-term observation of feature points can build more mutual constraints between frames and reduce cumulative navigation errors [4]. Therefore, valid and continuous tracking of feature points has an essential impact on the localization accuracy of visual navigation systems.

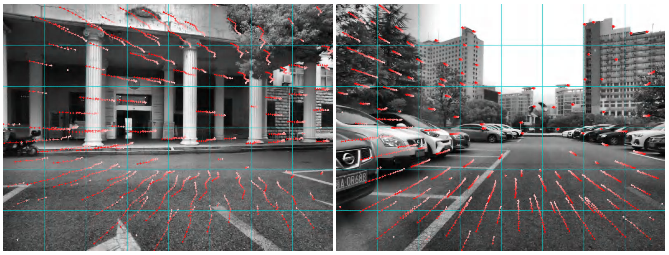

(a) (b)

Fig. 1. Feature points tracking of a moving robot. (a) When the robot turns right, the feature points in the left image edge will be lost immediately. (b) Sky region is textureless, and no points can be extracted; the faraway ahead area cannot form valid measurement because of limited parallax.

To obtain valid feature measurements and reduce redundant information, a standard method employed in visual navigation systems is to distribute feature points uniformly in the image and limit the minimum distance between features to distribute them as evenly as possible in the 3-D environment [5], [6]. This method only aims at uniformly distributing feature points in the current image plane, but it cannot guarantee the valid and continuous future tracking of the extracted feature points. Certain feature points will fail to track immediately when the carrier moves, as shown in Fig. 1(a). When the carrier turns right, the feature points at the left image edge are quickly lost, making continuous tracking and valid observations challenging. In addition, feature points cannot be extracted or tracked in the textureless image region, as shown in the textureless sky region of Fig. 1(b). In the texture image region but far away in the 3-D scene, the features' parallax is too limited to triangulate them accurately, as the features on the buildings of Fig. 1(b). As a result, these feature points cannot form valid constraints on the navigation state and have no contribution to visual localization. In some feature extraction methods based

Manuscript received 12 February 2024; revised 10 May 2024; accepted 23 May 2024. Date of publication 31 July 2024; date of current version 12 August 2024. This work was supported by the National Natural Science Foundation of China under Grant 42374034 and Grant 41974024. The Associate Editor coordinating the review process was Dr. Yulong Huang. *(Corresponding authors: Xiaoji Niu; Tisheng Zhang.)*

The authors are with the GNSS Research Center, Wuhan University, Wuhan 430079, China (e-mail: wlq@whu.edu.cn; zts@whu.edu.cn; wystephen@whu.edu.cn; thl@whu.edu.cn; xjniu@whu.edu.cn).

Digital Object Identifier 10.1109/TIM.2024.3436092

on attention mechanisms and methods that actively extract feature points [7], [8], [9], different weights are assigned to each image region based on the image texture information. Then, more feature points will be allocated in the textured region. However, many feature points still have less tracking and minimal parallax during the carrier's moving. In brief, the current feature point extraction methods do not consider the feature's following tracking quality, including tracking length and the valid parallax.

In order to obtain feature points that exhibit longer tracking length and superior valid parallax for visual localization, we propose a prior pose-guided active feature points distribution (P2GD) method, which redistributes the feature number into different image regions according to the predicted feature tracking total parallax. Specifically, the P2GD method employs the carrier's prior poses and the current 3-D environment information to predict the features' tracking information in each image region. The prior carrier poses can be obtained from the path planning module of the self-driving carrier. Notably, the 3-D environment is represented with the sparse 3-D feature points, derived from triangulating historical tracking measurements rather than relying on a prior 3-D landmark map. Then, we take the predicted tracking parallax of each image region as the distribution weight and reassign the feature number of each image region. In this way, the visual navigation system acquires more features with superior valid total tracking parallax, such that the visual navigation system demonstrates better localization accuracy. The main contributions of this article are as follows.

1) We propose and implement a P2GD method, which employs the prior carrier poses and sparse triangulated 3-D features to predict the tracking information for upcoming steps and redistribute the feature number of each image region.
2) We construct the redistributed weight matrix based on the predicted average tracking parallax of all features in each image region and design a feature distribution algorithm correspondingly.
3) We evaluate and analyze the enhancement of the proposed method on the feature tracking quality and the localization accuracy in both public and private datasets using a multistate constraint Kalman filter (MSCKF)-based visual navigation system and verify the robustness of the proposed method by adding certain noise in the prior poses intentionally.

The remainder of this article is organized as follows. The related work will be introduced in Section II. Subsequently, we will present the details of the proposed method in Section III. Then, the experiment and results will be discussed in Section IV. Finally, we will conclude the proposed feature distribution method in Section V.

II. Related Works

There are usually plenty of visual feature points extracted from an image by the feature detection algorithm, mostly clustered and poorly distributed [10]. Simultaneously, the clustered features are almost in the same direction and generally contribute duplicate and redundant information for visual localization. Also, visual features located in various directions to the camera offer different positioning constraints and enhance the visual localization accuracy. Therefore, achieving a reasonable feature point distribution becomes crucial, and the relevant researchers have studied various feature distribution methods.

A. Distance/Grid-Based Distribution

To solve the issue of clustered visual features, the nonmaximal suppression (NMS) method has been employed in visual feature extraction. Bailo et al. [11] proposed three adaptive NMS (ANMS) methods designed to efficiently extract visual features uniformly across the image. In addition, a method based on suppression via disk covering (SDC) in [12] efficiently selected the robust and good space-distributed features, demonstrating faster speed than the commonly used ANMS methods. Although these methods reduce the redundant information among the extracted visual features, they do not guarantee a reasonable feature distribution in the image.

The grid-based feature points distribution method is also commonly used in visual navigation systems [5], [6], [13]. This method distributes the features in all image planes by dividing the image into some grids/regions and assigning an average feature number to each grid. To prevent clustering, the grid-based method also enforces a minimum distance between the extracted features, making the features evenly distributed throughout every image region. However, a drawback arises if an image region is totally textureless, where the assigned feature in such region will be wasted. To address this problem, researchers have proposed a solution by dividing image regions from large to small based on a quadtree approach [14], [15]. This approach ensures that features will not be assigned to a grid if no features can be extracted, even after decreasing the threshold. This strategy makes full use of the total number of features.

In general, the distance/grid-based distribution methods only focus on distributing the features as evenly as possible in the image without considering the carrier's coming movement. As a result, these distribution methods offer no assistance to the feature's future tracking length and the tracking parallax.

B. Attention-Based Distribution

Considering that the texture of the image region determines the visual features quality, the salient region's features are generally regarded as more robust, stable, and accurate landmarks. With the deep learning matured, the identification of salient image regions has become faster and is widely employed in visual localization. SalientDSO [16] incorporated the visual saliency map into the direct sparse odometry (DSO). The authors employed the saliency map, obtained by human eye-tracking data, to predict the visual saliency and identify the informative regions. Consequently, feature distribution in the uninformative region would be downweighed. As feature points with more robustness to viewpoint and illumination changes were selected, SalientDSO demonstrated better positioning accuracy.

Frintrop and Jensfelt [17] presented a visual SLAM system that selects salient visual landmarks with an active gaze control strategy. The images' regions of interest (ROIs) were first found using the active gaze control strategy. Then, the stable landmarks were selected and tracked in the ROIs, enabling a better pose estimation for visual SLAM. Attention-SLAM [7] and SBAS [18] employed the visual saliency learned from the human gaze in the back end of their SLAM systems. The saliency prediction map served as the weight map of the feature points in the bundle adjustment of the state estimator, rather than feature point distribution in the front end.

In general, attention-based methods also distribute the features only according to the current image texture and semantics. Therefore, the features' future tracking qualities are neglected in these feature redistribution processes.

C. Active Feature Distribution

Wang et al. [19] performed active view planning for visual SLAM to address the perception failures in the featureless areas. They built an environment information map based on the Fisher information and determined the optimal informative viewpoints through horizon optimization. Then, the gimbal camera was controlled to the best viewpoint for tracking more features and acquiring more robust and accurate localization. In mobile robot navigation, motion planning and visual feature tracking were considered in [20]. Specifically, the association between visual features and the map points was taken into account in the motion planning framework. The number of associated map points in each frame was guaranteed, such that the visual SLAM system demonstrated better performance. Davison and Murray [21] proposed an automatic system using active vision for robot localization, which chooses the landmarks that can be tracked over a particular time. The localization with the active vision showed significant advantages over passive techniques. Zhao and Vela et al. [22] studied the selection of good features to track for visual SLAM. The sound features were selected according to the observability indices, and only the feature subset that contributes best to localization will continue to be tracked. Therefore, the feature association and localization accuracy will be improved. However, the feature's future tracking quality was not taken in their feature selection. Valiente et al. introduced an adaptive probability-oriented feature matching (POFM) method in [23] and an efficient version in [24], with the aim of actively and effectively identifying candidate image regions for features in a visual localization system. Utilizing the predicted robot pose obtained from a filter-motion prediction stage, they calculated the probability distribution of 3-D features in the subsequent image, which confirms robust feature matching and enhances localization accuracy. However, their POFM only focuses on robust feature matching between two consecutive image frames and does not predict the distribution of features in the upcoming images.

Some studies focus on actively adjusting path planning to achieve better feature distribution for improved visual localization. Rodrigues et al. [25] employed artificial potential fields within the image to generate control actions, guiding the vehicle toward the goal while still favoring feature-rich areas. This active localization approach yields improved feature distribution, consequently enhancing visual localization performance. Nonetheless, this method cannot be applied to the robot following fixed routes. Zhang and Scaramuzza [26], [27] constructed the Fisher information field by summarizing the scene localization information into discrete grids. By integrating the Fisher information field into the motion planning algorithm, both the visual localization rate and accuracy are enhanced. However, the method is also not suitable for robots with predetermined routes. Moreover, it necessitates a prior landmark map, thereby increasing the implementation complexity.

The active feature distribution methods generally combine motion planning with visual positioning, which considers the features' tracking length in their motion planning framework. However, these methods neglected the features' future tracking parallax, which is also crucial for the localization accuracy of the visual navigation system. Other active localization methods focus on path planning rather than feature distribution algorithms to enhance localization performance.

In summary of all the relevant researches, none of the visual feature points distribution methods comprehensively addresses the feature's future tracking quality, including the future tracking length, and the feature's tracking parallax. With sufficient guarantee of the feature's future tracking in the distribution process, the features' tracking qualities and localization accuracy of the visual navigation system can be further improved. Therefore, we propose a prior pose-guided active visual feature points distribution method to improve the features' tracking qualities and enhance the localization accuracy of the visual navigation system.

III. Methodology

The proposed P2GD method constitutes an independent step of the front end in the visual localization system. Consequently, it is adaptable to visual localization systems with different feature association methods, including optical-flow-based and descriptor-based systems. In addition, it can be integrated into systems leveraging different state estimators, including MSCKF-based visual–inertial navigation system (VINS), factor graph optimization (FGO)-based VINS, and FGO-based visual SLAM. In this article, we apply the proposed method to an optical-flow-based MSCKF-based VINS to verify its enhancement on feature tracking and system localization. In the following of this section, the overview of the proposed method will be presented first. Then, the specific steps of the proposed method will be introduced successively. Finally, we will present the implementation in an MSCKF-based VINS.

A. Overview of the Proposed Method

To keep the feature distribution method concise and efficient, we design the prior pose-guided visual feature points distribution method based on the grid-based distribution method. The flow diagram of feature points extraction with the proposed feature distribution method is illustrated in Fig. 2. The red parts in Fig. 2 represent the traditional grid-based

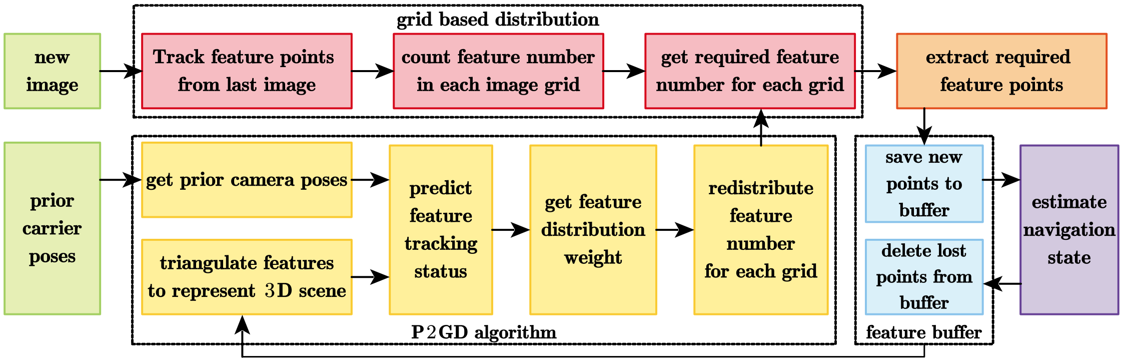

grid based distribution
new image
Track feature points from last image
count feature number in each image grid
get required feature number for each grid
extract required feature points
prior carrier poses
get prior camera poses
triangulate features to represent 3D scene
predict feature tracking status
get feature distribution weight
redistribute feature number for each grid
P2GD algorithm
save new points to buffer
delete lost points from buffer
feature buffer
estimate navigation state

Fig. 2. Feature points extraction flow diagram. The yellow part is the proposed feature points distribution method.

feature distribution method. When a new image arrives, the optimal-flow track will be first performed to obtain the associated features in the image. Then, the tracked feature in each image grid will be counted, and the required feature numbers will be calculated with the preassigned same feature amount of one grid. Then, the needed feature points will be extracted from each image grid, all new features will be employed in the VINS estimator to estimate the camera pose, and the feature points buffer will be updated.

The proposed P2GD method, as shown in the yellow parts of Fig. 2, will dynamically adjust the assigned feature number of each grid. To actively redistribute features, the proposed P2GD method needs the prior carrier poses and the 3-D scene information. The carrier's prior poses are acquired from the path planning module of a self-driving mobile robot/car, while the needed 3-D scene information is represented with the sparse 3-D feature points, which are triangulated from the saved historical feature measurements in the feature buffer. Then, the upcoming tracking status of features is predicted to redistribute features across the image plane. The predicted tracking process of P2GD is illustrated in Fig. 3. The sparse 3-D feature points, represented by the black points in Fig. 3(a), will be projected to the coming camera poses [excluding the green one in Fig. 3(a)] to determine the features' future tracking status. Subsequently, the projected feature positions on the upcoming camera planes are transformed to the current image plane, as depicted by the predicted feature tracking shown in Fig. 3(b). Then, the predicted features' parallax of each image region will be calculated as the feature distribution weight. Finally, the feature number will be reassigned to each image grid according to the weight. In the subsequently feature extraction module, the needed features will be extracted, just as that of the grid-based method.

B. Proposed P2GD Method

For the clarity of the symbols, we define the body frame and the inertial measurement unit (IMU) frame as the b frame, the camera frame as the c frame, the unified camera frame as the u frame, the pixel frame as the p frame, and the world frame as the w frame. The frame symbols with a subscript denote the corresponding frame at a specific time. In addition, we employ the rotation matrix and a vector to represent the carrier's attitude and position. In detail, $\mathbf{R}^{\mathrm{w}}_{\mathrm{b}}$ is the rotation matrix from the b frame to the w frame, and $\boldsymbol{p}^{\mathrm{w}}_{\mathrm{b}}$ is the position of the b frame in the w frame. Unless special explanation, the dimensions of the matrix and vector in the whole paper are 3-D. In this section, we introduce the specific steps of the proposed method in order.

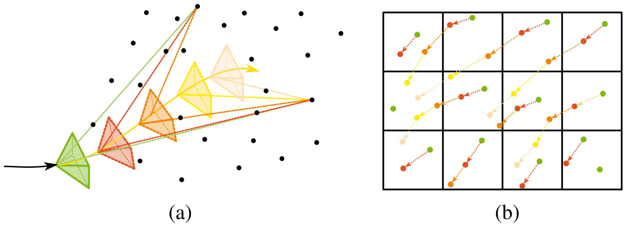

(a) (b)

Fig. 3. Diagram of the proposed method. (a) Prior camera poses and 3-D scene, where the green camera is the current pose, the other cameras are the predicted poses, and the black points are the sparse 3-D features. (b) Predicted feature trackings in the pixel plane, the green points are the 3-D points in the current pixel plane, and the other points are the expected pixel positions corresponding to the coming camera poses.

1) Prior Camera Poses and 3-D Scene: We utilize the prior carrier poses obtained from the path planning module of the self-driving vehicle as the prior poses information. The prior carrier poses are generally represented in the local body frame, i.e., $\boldsymbol{p}^{\mathrm{b}_0}_{\mathrm{b}_i}$ and $\mathbf{R}^{\mathrm{b}_0}_{\mathrm{b}_i}$, where b_0 denotes the current body frame and b_i denotes the body frame corresponding to the predicted ith pose. To get the predicted camera poses in the world frame, we perform the following projection on the prior carrier poses:

$$\begin{aligned}\mathbf{R}^{\mathrm{w}}_{\mathrm{c}_i} &= \mathbf{R}^{\mathrm{w}}_{\mathrm{b}_0}\mathbf{R}^{\mathrm{b}_0}_{\mathrm{b}_i}\mathbf{R}^{\mathrm{b}}_{\mathrm{c}}\\ \boldsymbol{p}^{\mathrm{w}}_{\mathrm{c}_i} &= \mathbf{R}^{w}_{\mathrm{b}_0}\boldsymbol{p}^{\mathrm{b}_0}_{\mathrm{b}_i} + \boldsymbol{p}^{w}_{\mathrm{b}_0} + \mathbf{R}^{w}_{\mathrm{b}_0}\mathbf{R}^{\mathrm{b}_0}_{\mathrm{b}_i}\boldsymbol{p}^{\mathrm{b}}_{\mathrm{c}}\end{aligned} \tag{1}$$

where $\{\boldsymbol{p}^{w}_{\mathrm{b}_0}, \mathbf{R}^{w}_{\mathrm{b}_0}\}$ is the current carrier pose in the world frame, $\{\boldsymbol{p}^{\mathrm{b}}_{\mathrm{c}}, \mathbf{R}^{\mathrm{b}}_{\mathrm{c}}\}$ is the camera-carrier extrinsic parameters, and $\{\boldsymbol{p}^{\mathrm{w}}_{\mathrm{c}_i}, \mathbf{R}^{\mathrm{w}}_{\mathrm{c}_i}\}$ is the predicted ith camera pose.

In feature-based visual navigation, we maintain the consecutive measurements of all feature points in the feature buffer to manage the visual measurements. Therefore, we can acquire the features' 3-D positions through their historical visual measurements and the previous camera poses to make up for the lack of depth information in the monocular camera. For a feature point f_j with m history measurements, the feature point's position $\boldsymbol{p}^{\mathrm{w}}_{f_j}$ can be formulated as follows:

$$\begin{cases}\left(\mathbf{R}^{\mathrm{w}}_{\mathrm{c}_1}\right)^T\left(\boldsymbol{p}^{\mathrm{w}}_{f_j} - \boldsymbol{p}^{\mathrm{w}}_{\mathrm{c}_1}\right) = z^{\mathrm{c}_1}_{f_j}\boldsymbol{p}^{\mathrm{u}_1}_{f_j}\\ \dots\\ \left(\mathbf{R}^{\mathrm{w}}_{\mathrm{c}_m}\right)^T\left(\boldsymbol{p}^{\mathrm{w}}_{f_j} - \boldsymbol{p}^{\mathrm{w}}_{\mathrm{c}_m}\right) = z^{\mathrm{c}_m}_{f_j}\boldsymbol{p}^{\mathrm{u}_m}_{f_j}\end{cases} \tag{2}$$

where $\{\boldsymbol{p}_{\mathrm{c}_m}^{\mathrm{w}}, \mathbf{R}_{\mathrm{c}_m}^{\mathrm{w}}\}$ is the previous mth camera pose, $z_{f_j}^{\mathrm{c}_m}$ is the feature point's depth in the c_m frame, and $\boldsymbol{p}_{f_j}^{\mathrm{u}_m}$ is the feature's unity coordinate in the c_m frame.

By multiplying the skew symmetric matrix of $\boldsymbol{p}_{f_j}^{\mathrm{u}_m}$, denoted as $[\boldsymbol{p}_{f_j}^{\mathrm{u}_m}]_\times$, at the equation's both sides, we can eliminate the feature's unknown depth $z_{f_j}^{\mathrm{c}_m}$ and obtain the following equation:

$$\sum_{i=1}^{m}\left(\left[\boldsymbol{p}_{f_j}^{\mathrm{u}_m}\right]_\times \left(\mathbf{R}_{\mathrm{c}_i}^{\mathrm{w}}\right)^T\right)\boldsymbol{p}_{f_j}^{\mathrm{w}} = \sum_{i=1}^{m}\left(\left[\boldsymbol{p}_{f_j}^{\mathrm{u}_m}\right]_\times \left(\mathbf{R}_{\mathrm{c}_i}^{\mathrm{w}}\right)^T \boldsymbol{p}_{\mathrm{c}_i}^{\mathrm{w}}\right). \tag{3}$$

As long as the feature has more than two historical pixel measurements, the feature's position $\boldsymbol{p}_{f_j}^{\mathrm{w}}$ can be solved from (3).

Once we get enough triangulated feature points, these sparse 3-D feature points are employed to represent the 3-D scene in our method. The 3-D scene, represented by the sparse 3-D features, is employed to predict the expected tracking qualities of image regions in the following several frames. That is, the average future tracking quality of the 3-D points within one grid is treated as the future tracking quality of the corresponding image region. If there are no 3-D points in one particular image region, the future tracking will be considered as nonexistent. It is reasonable for the featureless regions considering that the image region without any 3-D feature points is usually infinity or textureless.

2) Predicted Feature Tracking: We project the 3-D feature points into the predicted camera frames to acquire the features' future tracking information. Specifically, we first project one feature into the ith predict camera pose, represented as $\boldsymbol{p}_{f_j}^{\mathrm{c}_i}$, through coordinate transformation. Then, the feature's normalized coordinates in this camera frame can be calculated as $\boldsymbol{p}_{f_j}^{\mathrm{u}_i} = \boldsymbol{p}_{f_j}^{\mathrm{c}_i}/p_{f_j,z}^{\mathrm{c}_i}$, where $p_{f_j,z}^{\mathrm{c}_i}$ is the feature's depth in the ith camera frame. Before normalizing the feature's coordinate in the camera frame, we will check $p_{f_j,z}^{\mathrm{c}_i}$ to maintain the numerical stability. When $p_{f_j,z}^{\mathrm{c}_i}$ is larger than a given threshold, such as 5 cm, we take the feature's measurement in the c_i frame valid and proceed with the following projection procedure. Otherwise, the feature will be considered lost in this camera frame, its projection process will be stopped, and its predicted tracking length will be recorded as i.

Applying the distortion model and parameters, we get the distorted normalized coordinates $\left[x_{d,j}^{\mathrm{u}_i}\ \ y_{d,j}^{\mathrm{u}_i}\right]$ in the u_i frame as follows:

$$\begin{cases} x_{d,j}^{\mathrm{u}_i} = p_{f_j,x}^{\mathrm{u}_i} d_j + 2\ p_1 p_{f_j,x}^{\mathrm{u}_i} p_{f_j,y}^{\mathrm{u}_i} + p_2\left(r_j^2 + 2(p_{f_j,x}^{\mathrm{u}_i})^2\right) \\ y_{d,j}^{\mathrm{u}_i} = p_{f_j,y}^{\mathrm{u}_i} d_j + 2\ p_2 p_{f_j,x}^{\mathrm{u}_i} p_{f_j,y}^{\mathrm{u}_i} + p_1\left(r_j^2 + 2(p_{f_j,y}^{\mathrm{u}_i})^2\right) \end{cases} \tag{4}$$

where $d_j = (1 + k_1 r_j^2 + k_2 r_j^4)$; k_1, k_2, p_1, and p_2 are the camera's distortion parameters; and r_j denotes the distance between the feature and the origin of the u_i frame, which has a relationship with the normalized coordinates of $r_j^2 = (p_{f_j,x}^{\mathrm{u}_i})^2 + (p_{f_j,y}^{\mathrm{u}_i})^2$. Then, utilizing the camera projection model, we can get the feature's coordinates in the p frame, which can be represented as follows:

$$\begin{bmatrix} u_j^{\mathrm{p}_i} \\ v_j^{\mathrm{p}_i} \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{K}} \begin{bmatrix} x_{d,j}^{\mathrm{u}_i} \\ y_{d,j}^{\mathrm{u}_i} \\ 1 \end{bmatrix} \tag{5}$$

where $\boldsymbol{K}$ is the camera's intrinsic matrix and $[u_j^{\mathrm{p}_i}\ \ v_j^{\mathrm{p}_i}]$ are the feature's pixel coordinates. Then, we determine if the feature is located outside the image with its pixel coordinates. If the feature is outside the image, indicating a tracking failure, the feature's projection procedure will be interrupted. On the contrary, the predicted pixel coordinates will be recorded for the following calculation.

Afterward, we perform the same projection process on the other prior camera poses until the feature is lost or reaches the end of the prior poses. The iterative procedure is applied to all the 3-D feature points to predict their tracking information in the prior camera frames.

3) Features Distribution Weight: To reassign the feature number in the image regions according to the predicted tracking information, it is essential to define a reasonable metrics as the feature distribution weight. In two-view geometry, the parallax reflects the constraint of camera poses on the feature's position [28]. Features with more considerable parallax typically exhibit lower depth errors after triangulation. As a correspondence, the more significant parallax features contribute more constraints to the navigation state in visual localization. In addition, the longer tracking length, i.e., the more feature measurements, provides more constraints on the camera poses, resulting in a more accurate navigation result. Therefore, with considering both the two-view parallax and the tracking length, we employ the total tracking parallax as the distribution weight in the proposed method, i.e., the sum of every two-consecutive-frame parallax of the feature's all tracking frames. The total tracking parallax of the feature $\boldsymbol{p}_{f_j}$ is denoted as follows:

$$\omega_j = \sum_{i=0}^{N-1} \arccos\left(\frac{\left(\mathbf{R}_{\mathrm{c}_i}^{\mathrm{c}_{i+1}} \boldsymbol{p}_{f_j}^{\mathrm{c}_i}\right) \cdot \boldsymbol{p}_{f_j}^{\mathrm{c}_{i+1}}}{\left\|\mathbf{R}_{\mathrm{c}_i}^{\mathrm{c}_{i+1}} \boldsymbol{p}_{f_j}^{\mathrm{c}_i}\right\| \left\|\boldsymbol{p}_{f_j}^{\mathrm{c}_{i+1}}\right\|}\right) \tag{6}$$

where N is the predicted tracking length, $\mathbf{R}_{\mathrm{c}_i}^{\mathrm{c}_{i+1}}$ is the rotation matrix from the ith camera frame to the $(i+1)$th camera frame, and $\boldsymbol{p}_{f_j}^{\mathrm{c}_i}$ and $\boldsymbol{p}_{f_j}^{\mathrm{c}_{i+1}}$ are the predicted coordinates in the ith camera frame and the $(i+1)$th frame, respectively.

The current 3-D features are allocated to image grids based on their current pixel coordinates, i.e., coordinates in the zeroth camera frame. Corresponding to the 2-D p frame, there are also two indices for each image grid, including x- and y-directions. The belonged image grid index for a 3-D feature point can be obtained from

$$\begin{aligned} x &= \mathrm{floor}\left(n_{px} \cdot u_{f_j}^{\mathrm{c}_0} / I_w\right) \\ y &= \mathrm{floor}\left(n_{py} \cdot v_{f_j}^{\mathrm{c}_0} / I_h\right) \end{aligned} \tag{7}$$

where floor(·) denotes round down operation, $[I_w\ \ I_h]$ are the image width and height, and n_{px} and n_{py} are the grid numbers in the u- and v-directions, respectively. The features with the same grid indices are clustered into a set, denoted as $F_{x,y}$. Then, the distribution weight $\omega_{x,y}$ of the image region corresponding to the feature set $F_{x,y}$ is obtained from

$$\omega_{x,y} = \begin{cases} \dfrac{1}{\text{size}(F_{x,y})} \displaystyle\sum_{l \in F_{x,y}} \omega_l, & \text{size}(F_{x,y}) > 0 \\ 0, & \text{size}(F_{x,y}) = 0. \end{cases} \tag{8}$$

Then, we can get the feature number distribution weight matrix of the image

$$\boldsymbol{W}_{n_{px} \times n_{py}} = \begin{bmatrix} \omega_{0,0} & \cdots & \omega_{0,n_{py}-1} \\ \vdots & \ddots & \vdots \\ \omega_{n_{px}-1,0} & \cdots & \omega_{n_{px}-1,,n_{py}-1} \end{bmatrix}. \tag{9}$$

Considering the incorporation of feature tracking length and parallax into the distribution weight, features extracted from regions with larger weights generally exhibit improved tracking length and parallax. Consequently, by distributing a great number of features to such image regions, we can effectively enhance the overall feature tracking qualities. The feature number redistribution algorithm is detailed in Section III-B4.

4) Feature Number Redistribution: Before feature number reassignment, we should preset critical parameters, including the total feature number m, the grid numbers n_{px} and n_{py} in the u- and v-directions, and the feature's minimum distance d_{px}. The maximum feature number n in one grid is also calculated in advance using the above parameters and the image size. Then, the designed feature number redistribution algorithm is employed to reasonably redistribute the features to each image region according to the weight matrix.

As depicted in Algorithm 1, we first initialize several key parameters before proceeding with feature distribution. During this initialization, the current total weight ω_t is set to 1, the reassigned feature number $\boldsymbol{N}$ is initialized as a zero matrix, and the weight matrix $\boldsymbol{M}$ is reordered by value. Subsequently, the total features are assigned to each image grid based on their weights. Given that the feature number assigned to a single grid is finite, there might be some remaining features after distributing features to all grids with nonzero weight. In such cases, we evenly distribute the remaining features across these grids. Here, we provide the details of the two-step distribution.

In the first weighted distribution, we traverse through the grids by their weight values and assign numbers of features. We first calculate the proportion of the current grid weight in relation to the remaining total weight. Subsequently, the feature number of this grid is computed by multiplying this proportion with the remaining total feature number. Moreover, the obtained number is rounded up and constrained to the maximum number of one grid n. The final assigned feature number is saved in the feature number matrix $\boldsymbol{N}$. Notably, the dynamic proportions for feature distribution prevent the occurrence of zero features in regions with lower weights. Finally, the remaining total weight and total number of features are updated, and the current grid weight is removed from the weight matrix $\boldsymbol{M}$. This process iterates through the other grid weights in $\boldsymbol{M}$ until either the remaining total weight or the remaining feature number reaches 0.

Algorithm 1 Feature Number Redistribution Algorithm

```
Input: weight matrix $\boldsymbol{W}$, total feature number $m$, maximum
    feature number in one grid $n$.
Output: feature number matrix $\boldsymbol{N}$
  Initialize current total weight $\omega_t = 1$, set $\boldsymbol{N} = 0$.
  reorder each element of $\boldsymbol{W}$ by value.
  for $w_i \in W$ do
    if $w_i$ is 0 or $m$ is 0 then
      break;
    end if
    calculate feature number $a = ceil(m * w_i / w_t)$;
    if $a > n$ then
      $a \leftarrow n$;
    end if
    $m \leftarrow m - a$; $w_t \leftarrow w_t - w_i$; $W \leftarrow W \backslash w_i$;
    assign $a$ to $N$ by the grid index
  end for
  while $m > 0$ do
    $w_i \leftarrow$ the first data in $\boldsymbol{W}$
    calculate feature number $a = ceil(m/size(\boldsymbol{W}))$
    $m \leftarrow m - a$; $W \leftarrow W \backslash \{w_i\}$;
    assign $a$ to $\boldsymbol{N}$ by the grid index
  end while
```

The second average distribution will be executed if there are remaining features after the first distribution step. In this step, the feature number assigned to a single grid is calculated, rounded up, and then assigned to the first grid in the $\boldsymbol{M}$. Then, we update the remaining feature number n and remove this grid from the remaining weight matrix $\boldsymbol{M}$. The process continues until the remaining feature number reaches 0.

The required feature number in each grid is calculated with the assigned feature number matrix $\boldsymbol{N}$ and extracted from the new image. The pixel measurements of the new feature points are added to the feature buffer. Subsequently, the current features form pose constraints in the visual navigation system for state estimation, and the visual measurements of the lost features will be removed from the feature buffer.

C. Implementation in MSCKF-Based VINS

To verify the enhancement of the proposed method for visual navigation performance, we implement and evaluate the P2GD method in an MSCKF-based VINS.

1) MSCKF-Based VINS: In the MSCKF-based VINS, the camera poses are included in the state vector as clones rather than the feature points, which significantly reduces the state dimension and further reduces the computation burden in the state propagation and update [29]. The state vector of the MSCKF-based VINS is composed of the current IMU error state $\boldsymbol{x}_{I_k}$ and the error of cloned n historical camera poses $\delta\boldsymbol{T}^{\text{w}}_{\text{c}_1}, \ldots, \delta\boldsymbol{T}^{\text{w}}_{\text{c}_i}, \ldots, \delta\boldsymbol{T}^{\text{w}}_{\text{c}_n}$, which is denoted as follows:

$$\boldsymbol{x}_k = \left[\left(\boldsymbol{x}_{I_k}\right)^T \quad \left(\delta\boldsymbol{T}^{\text{w}}_{\text{c}_1}\right)^T \quad \cdots \quad \left(\delta\boldsymbol{T}^{\text{w}}_{\text{c}_n}\right)^T \right]^T. \tag{10}$$

Denote the covariances of the IMU state, the cloned poses, and that between the IMU state and clones poses as $\boldsymbol{P}_{I_k}$, $\boldsymbol{P}_{T_k}$, and $\boldsymbol{P}_{I_k T_k}$, respectively. With the state transition matrix

$\boldsymbol{\Phi}_k$, which is acquired from the IMU kinematic equation, the system state covariance can be propagated as follows:

$$\boldsymbol{P}_{k+1} = \begin{bmatrix} \boldsymbol{P}_{I_{k+1}} & \boldsymbol{\Phi}_k \boldsymbol{P}_{I_k T_k} \\ \boldsymbol{P}_{I_k T_k}^T \boldsymbol{\Phi}_k^T & \boldsymbol{P}_{T_k} \end{bmatrix} \tag{11}$$

where $\boldsymbol{P}_{I_{k+1}} = \boldsymbol{\Phi}_k \boldsymbol{P}_{I_k} \boldsymbol{\Phi}_k^T + \boldsymbol{Q}_k$ is the updated IMU state covariance and $\boldsymbol{Q}_k$ is the propagation noise. The augmentation or marginalization of a new camera pose or the oldest camera pose from the covariance can be referred to [24].

In MSCKF-based VINS, features lost in the current frame, or those reaching the maximum tracking length are selected to perform measurement update. The measurement equation is formulated in the p frame. The pixel coordinate of the feature $\boldsymbol{p}_{f_j}$ in the ith p frame can be calculated as follows:

$$\boldsymbol{p}_{f_j}^{\mathrm{p}_i} = \boldsymbol{h}_d\left(\boldsymbol{h}_p\left(\boldsymbol{h}_t\left(\mathbf{R}_{\mathrm{c}_i}^{\mathrm{w}}, \boldsymbol{p}_{\mathrm{c}_i}^{\mathrm{w}}, \boldsymbol{p}_{f_j}^{\mathrm{w}}\right), \boldsymbol{K}\right), \boldsymbol{\xi}\right) \tag{12}$$

where $\boldsymbol{h}_t$, $\boldsymbol{h}_p$, and $\boldsymbol{h}_d$ are the transformation, projection, and distortion functions, respectively, and $\boldsymbol{\xi}$ is the camera's distortion parameters. Performing perturbance in the projection process and integrating all measurements of the feature, we obtain the measurement equation at timestamp k

$$\boldsymbol{z}_{p_j} = \boldsymbol{H}_{x,k}\boldsymbol{x}_k + \boldsymbol{H}_f \delta \boldsymbol{p}_{f_j}^{\mathrm{w}} + \boldsymbol{n}_{p_j} \tag{13}$$

where $\boldsymbol{H}_{x,k}$ is the measurement Jacobian to the state vector, $\boldsymbol{H}_f$ is the measurement Jacobian to the feature, and $\boldsymbol{n}_{p_j}$ is the measurement noise. The null-space projection can eliminate the unknown feature's position in the measurement equation; then, the standard Kalman update can be performed.

2) Implementation: Integrating the aforementioned key components, we have developed an MSCKF-based VINS based on the open-sourced OpenVINS [6] platform. In addition, we also incorporate the proposed P2GD method into the front end of our VINS framework. In the front end of our VINS framework, optical-flow tracking is initially employed to obtain new and associated features upon the arrival of a new image. Subsequently, the required numbers of features are determined based on the preassigned count. Supplementary features are then extracted from various regions within the new image, and the required features are selected by their response values and the required numbers. The extracted features of the front end are utilized in the measurement update process of the VINS back end. While in the front end of VINS utilizing the proposed P2GD algorithm, we first query the prior camera poses and the current tracking features for the P2GD algorithm to determine the assigned numbers of features in each image region. Following this, we also perform feature tracking on the new image and identify the associated features with that in the old image. The required number of features in each image region is then calculated using the redistributed feature number provided by the P2GD algorithm. Finally, we extract the required features, employing a process similar to the aforementioned new feature extraction and selection method.

In the feature association, we utilize the RANSAC method along with bidirectional tracking inspection to validate the feature tracking results. In the measurement update process of the VINS back-end, we employ the chi-square check to efficiently eliminate measurement outliers. We also designed some strategies in our implementation to keep the robustness and lightweight of the P2GD. Once the feature's triangulated position is accurate enough, the feature's 3-D position will be reserved, and the pretriangulation for this feature will no longer be performed, which controls the computation burden. We empirically consider the feature's position accurate enough when its valid position is triangulated from more than 10 camera poses. Besides, the 3-D features that reach the maximum tracking length will remain in the feature buffer after its measurement update to compensate for the sparse features lacking reflection of the 3-D scene. Also, to reduce the feature buffer size, the 3-D features that cannot be tracked in the first predicted camera poses will be abandoned. As the feature's maximum track length in VINS corresponds to the size of the slide window, we set the size of the prior poses to the slide window's length. Besides, since every image will be taken as the keyframe in OpenVINS, the prior camera poses used in our method have the same frequency as the image data's rate. Other parameters, including the maximum feature number, the grid size, and the minimum feature distance, will be set according to the image size employed in VINS.

IV. Experiments and Results

A. Experiments and Evaluation Description

1) Experiments Description: We evaluate the enhancement of the proposed P2GD method through a VINS, and both the public dataset and our private dataset are employed in our experiments. Specifically, the public dataset is the KAIST urban dataset [30], collected from a commercial car, while the private dataset is collected from a wheeled robot. It is noted that the prior camera poses in our experiments are acquired from the truth trajectories rather than the poses from the real-time motion-planning module due to the limitation of the test conditions. Nevertheless, it still convinces enough.

The public KAIST urban dataset is a vehicle's multisensor dataset collected in a complex urban environment, where only the left camera and the industrial-grade MEMS IMU measurements are utilized in our experiments. The camera has a resolution of 1280 × 560 and a frame rate of 10 Hz, while the IMU has a data rate of 100 Hz. For this dataset, the maximum feature number in one image is set to 180, the grid size is 10 × 6, and the minimum distance between feature points is 40 pixels. Besides, the slide window's size is 20; thus, 20 prior poses are required in each prediction process. The large vehicle's speed causes a big accumulated pose error during measurement updates in VINS. Therefore, we do not employ the first estimate Jacobian (FEJ) for the KAIST dataset to avoid large linearization errors in the measurement matrix. We also initialize the VINS with given accurate initial state and IMU bias for the KAIST dataset to prevent any interference with the evaluation. Five groups of data, namely, *urban28*, *urban30*, *urban32*, *urban38*, and *urban39*, are evaluated in our experiments, of which the total time is 8321 s, and the total distance is 45 525 m.

Our private dataset is collected in the typical campus scene with a wheeled robot, as shown in Fig. 4. The left gray camera and the industrial-grade MEMS IMU are employed in our

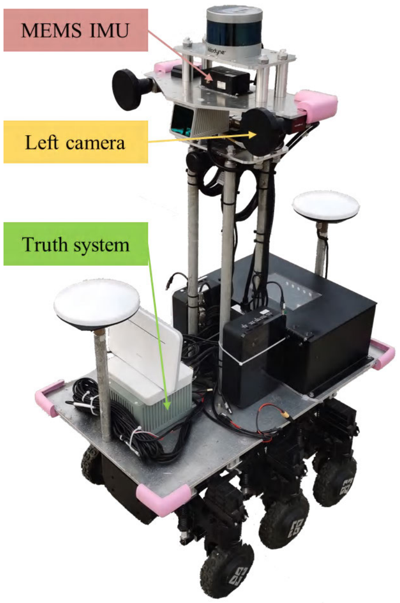


Fig. 4. Test wheeled robot of the private dataset.

experiments. The camera's resolution is 800 × 600. The data rates of the camera and IMU are 10 and 200 Hz, respectively. The truth trajectory of the private dataset is smoothed from the navigation-grade IMU and GNSS RTK positioning results. In our experiment, the maximum keypoints number in one image is 150, the image grid size is 8 × 6, the minimum keypoints distance is 30 pixels, and the size of the sliding window is 20. Our private dataset has seven groups of robot data, denoted as *Robot-A–Robot-G*, and the total time and distance are 8025 s and 10 498 m, respectively. The private dataset's trajectories are shown in Fig. 5.

2) Evaluation Method: We validate the benefits of the proposed P2GD algorithm using a VINS. To evaluate the effectiveness of the proposed active feature distribution method, we consider the widely used inactive distribution method, namely, the grid-based method, as the baseline. In the tables of this section, the VINS employing the baseline grid-based method is denoted as "baseline," while that incorporating our P2GD algorithm is denoted as "proposed method." Notably, the baseline method and our proposed method utilize the same state estimator and system parameters to ensure a fair comparison. Considering that the P2GD algorithm is implemented and evaluated in the filter-based VINS, we also compare the performance of the baseline and proposed method with the state-of-the-art MSCKF-based method, OpenVINS [6]. The system parameters of OpenVINS are optimized to the best extent for the test dataset.

Both feature tracking qualities and navigation accuracy are considered as the evaluation metrics. During VINS operation, the actual feature tracking status is recorded to calculate the tracking qualities, including the feature's tracking length, parallax between two consecutive frames, and total tracking parallax. Besides, proportions of features with different tracking lengths are also computed to analyze enhancements in the feature tracking length. For the navigation accuracy evaluation, EVO [31] is adopted to calculate the positioning error quantificationally, including the absolute error of the total test sequence [the absolute translation error (ATE) and absolute rotation error (ARE)] and the relative error of trajectory fragments with different lengths [relative translation error (RTE) and relative rotation error (RRE)]. Furthermore, the robustness of the proposed method is also evaluated by introducing certain position and attitude noises into the prior carrier poses intentionally.

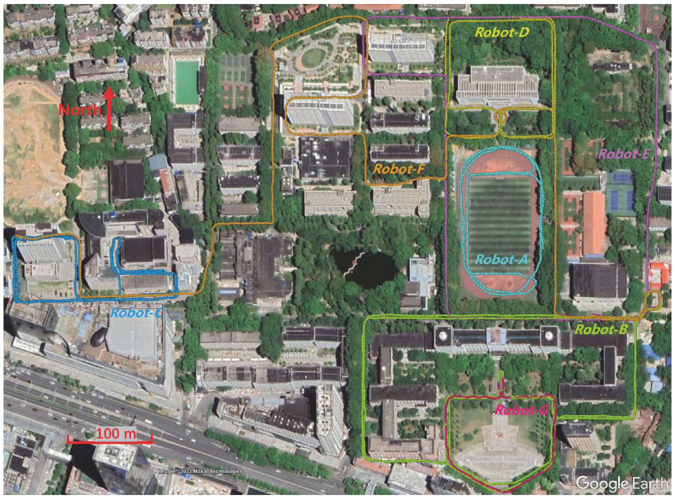


Fig. 5. Test trajectories of the private dataset.

TABLE I
AVERAGE TRACKING QUALITIES IN THE KAIST URBAN DATASET

Group	length / parallax / total parallax [frame / deg / deg]		
	Baseline	Proposed Method	Increase
urban28	4.85 / 2.47 / 11.97	4.94 / 2.93 / 14.49	0.09 / 0.47 / 2.52
urban30	6.22 / 2.20 / 13.71	6.31 / 2.54 / 16.04	0.09 / 0.34 / 2.34
urban32	4.10 / 2.40 / 9.83	4.48 / 2.95 / 13.20	0.38 / 0.55 / 3.37
urban38	5.30 / 2.77 / 14.68	5.35 / 3.21 / 17.16	0.06 / 0.43 / 2.48
urban39	5.17 / 2.79 / 14.39	5.36 / 3.36 / 18.02	0.19 / 0.58 / 3.62
RMS	**5.17 / 2.53 / 13.11**	**5.32 / 3.01 / 16.03**	**0.15 / 0.48 / 2.91**

B. Public Dataset

1) Feature Tracking Quality: Considering that the open-sourced OpenVINS also equips with a grid-based feature distribution method, the feature tracking qualities of OpenVINS and the "baseline" are similar. Therefore, in terms of the feature tracking quality, we only compare the "proposed method" with the "baseline" to evaluate the performance of the P2GD algorithm. The features' average tracking qualities with the "baseline" and the "proposed method" in the public KAIST dataset are counted in Table I, while Table II presents the proportion of features with different tracking lengths, including the minimum tracking length (one frame), maximum tracking length (20 frames), and other specific tracking lengths, such as five frames, ten frames, and 15 frames.

In Table I, the features' average tracking length, two-view parallax, and total parallax of the proposed method are more extensive than those of the baseline method across all five groups of KAIST data. Statistically, the proposed method has an improvement of 0.15 frames, 0.48°, and 2.91° in average

TABLE II

PROPORTION OF FEATURES WITH DIFFERENT TRACKING LENGTHS IN THE KAIST URBAN DATASET

Proportion [%]	Method	Tracking Length				
		1	5	10	15	20
urban28	Baseline	36.5	4.6	1.5	0.8	7.6
	Proposed	31.7	5.3	1.7	0.9	6.4
urban30	Baseline	31	4.5	1.9	1.1	12.9
	Proposed	26.8	5.2	2	1.2	11.4
urban32	Baseline	40.7	4.4	1.3	0.6	4.8
	Proposed	33.8	5.3	1.6	0.8	4.9
urban38	Baseline	33.8	4.8	1.7	0.9	9.7
	Proposed	29.4	5.4	1.9	0.9	8.1
urban39	Baseline	34.5	4.7	1.6	0.9	8.9
	Proposed	28.7	5.6	1.9	1	7.7
RMS	Baseline	35.5	4.6	1.6	0.9	9.2
	Proposed	30.2	5.4	1.8	1	8
	Increase	**-5.3**	**0.8**	**0.2**	**0.1**	**-1.2**

tracking length, two-view parallax, and total tracking parallax, respectively. From the proportion in Table II, the proportion of features that can only be tracked once is substantial, primarily due to the vehicle's high speed and the presence of many dynamic objects in the test scene. The proposed method redistributes the feature points according to the predicted total tracking parallax, assigning a few features in the region where features are quickly lost, such as the marginal region of the image. As a result, the proportion of features with once tracking in our proposed method decreases by 5.3% numerically. Our method also reduces the proportion of feature points with the maximum tracking length. It is related to the reduction of the straight-ahead long-tracking feature points. As the car almost keeps moving ahead, the straight-ahead feature points can be tracked stably. Nonetheless, the straight-ahead feature points contribute little to the localization because of low valid parallax. The proposed method assigns less weight to the straight regions, such that the feature points with maximum tracking length in our method decrease. Importantly, this reduction would not negatively affect navigation accuracy. Correspondingly, the proportions of features with all other tracking lengths, such as 5 frames, 10 frames, and 15 frames in Table II, show certain improvements in our proposed method. These features generally exhibit larger tracking parallax and provide the best contributions to navigation accuracy.

2) Navigation Accuracy: The vehicle's trajectories in the KAIST dataset are solved by the OpenVINS, "baseline," and "proposed method." To display the navigation accuracy intuitively, we present the test trajectories of *urban39* data in Fig. 6, with start points of all test trajectories aligned with the truth trajectory (note that the start point of OpenVINS is different since a slightly delayed initialization). Since a refined INS algorithm and a more suitable IMU error model [32] are utilized in both our baseline and proposed methods, their trajectories exhibit a notable improvement in smoothness compared with those of OpenVINS, especially evident in the subfigure of Fig. 6. When it comes to the overall trajectory, our baseline method also aligns better with truth trajectory compared with OpenVINS, confirming the superiority of our baseline method. From the trajectories of the proposed and baseline methods, the proposed method demonstrates lower position error and better fitness with the truth trajectory, which benefits from the superior features' tracking qualities of the proposed method.

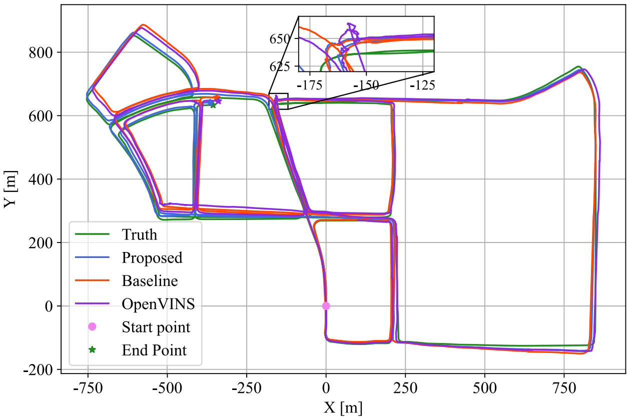


Fig. 6. Trajectories in KAIST *urban39* data.

TABLE III

ABSOLUTE POSE ERROR IN THE KAIST URBAN DATASET

Group	ARE / ATE [deg / m]			
	OpenVINS	Baseline	Proposed Method	Error Decrease*
urban28	5.13 / 34.67	2.96 / 27.25	2.54 / 22.03	0.43 / 5.22
urban30	5.32 / 20.55	2.31 / 13.41	2.25 / 11.05	0.06 / 2.36
urban32	3.02 / 25.83	1.69 / 16.10	1.99 / 11.22	-0.3 / 4.88
urban38	5.41 / 34.19	2.08 / 11.21	2.00 / 9.09	0.08 / 2.12
urban39	3.30 / 29.39	2.11 / 9.97	1.87 / 7.62	0.24 / 2.36
RMS	**4.56 / 29.41**	**2.27 / 16.77**	**2.14 / 13.22**	**0.13 / 3.55**

*Error decrease between the proposed and our baseline methods

Since there are many stationary moments in the KAIST dataset and the truth trajectories are not continuous, we only compare the ATE and ARE in the navigation performance evaluation. The absolute pose error in the KAIST dataset is counted and shown in Table III. According to the table, both our baseline and proposed methods exhibit significantly superior localization performance compared with OpenVINS, coincident with the results depicted in Fig. 6. To facilitate a more intuitive display and comparison of localization results, we exclusively evaluate our proposed method against our baseline method in subsequent tests.

Because of the extracted features with larger tracking parallax, the proposed method demonstrates superior absolute position accuracy across the five test trajectories. Most of the AREs of the proposed method are less than that of the baseline method. The statistical accuracy of the five trajectories indicates that the proposed method reduces the ATE from 16.77 to 13.22 m and the ARE from 2.27° to 2.14°. In particular, the statistical ATE of the proposed method has an improvement of 21%.

C. Private Dataset

1) Feature Tracking Quality: The features' tracking qualities in our private dataset are statistics in Tables IV and V. As the private dataset is collected in a richly textured campus with a slow-speed wheeled robot, the number of features

TABLE IV

AVERAGE TRACKING QUALITIES IN THE PRIVATE ROBOT DATASET

Group	length / parallax / total parallax [frame / deg / deg]		
	Baseline	Proposed Method	Increase
Robot-A	9.53 / 2.82 / 26.86	8.72 / 3.51 / 30.62	-0.8 / 0.69 / 3.75
Robot-B	9.39 / 3.67 / 34.42	9.45 / 4.05 / 38.29	0.06 / 0.39 / 3.88
Robot-C	9.84 / 3.34 / 32.88	9.44 / 3.83 / 36.13	-0.4 / 0.48 / 3.25
Robot-D	9.95 / 3.46 / 34.44	9.71 / 3.85 / 37.36	-0.24 / 0.39 / 2.92
Robot-E	9.58 / 3.63 / 34.83	9.38 / 4.05 / 37.97	-0.2 / 0.41 / 3.14
Robot-F	9.34 / 3.50 / 32.69	9.18 / 3.93 / 36.09	-0.16 / 0.43 / 3.4
Robot-G	9.52 / 3.50 / 33.33	9.84 / 3.89 / 38.25	0.32 / 0.39 / 4.93
RMS	**9.59 / 3.43 / 32.89**	**9.40 / 3.88 / 36.41**	**-0.2 / 0.45 / 3.52**

TABLE VI

ABSOLUTE POSE ERROR IN THE PRIVATE ROBOT DATASET

Group	ARE / ATE [deg / m]		
	Baseline	Proposed Method	Error Decrease
Robot-A	0.83 / 1.15	0.79 / 1.51	0.03 / -0.36
Robot-B	2.06 / 5.29	1.83 / 4.20	0.23 / 1.09
Robot-C	2.17 / 3.06	0.58 / 1.84	1.59 / 1.22
Robot-D	2.38 / 3.78	1.34 / 1.57	1.04 / 2.22
Robot-E	0.44 / 2.58	0.37 / 1.74	0.08 / 0.83
Robot-F	0.93 / 3.90	0.85 / 3.18	0.08 / 0.71
Robot-G	0.60 / 0.61	0.55 / 0.35	0.05 / 0.26
RMS	**1.55 / 3.28**	**1.02 / 2.36**	**0.53 / 0.91**

TABLE V

PROPORTION OF FEATURES WITH DIFFERENT TRACKING LENGTHS IN THE PRIVATE ROBOT DATASET

Proportion [%]	Method	Tracking Length				
		1	5	10	15	20
Robot-A	Baseline	21.8	4	1.9	1.1	24.6
	Proposed	21.9	4.6	2.3	1.2	19.3
Robot-B	Baseline	20.1	4.3	2.2	1.4	22.6
	Proposed	18.4	4.5	2.5	1.4	21.5
Robot-C	Baseline	20.1	4.1	2.1	1.3	26.8
	Proposed	19.1	4.5	2.4	1.4	23.2
Robot-D	Baseline	18.2	4.1	2.2	1.4	25.7
	Proposed	17.6	4.5	2.6	1.5	23.5
Robot-E	Baseline	19.3	4.2	2.3	1.4	24.3
	Proposed	18.6	4.4	2.6	1.4	22
Robot-F	Baseline	20.3	4.3	2.2	1.3	22.2
	Proposed	19.1	4.6	2.5	1.4	20
Robot-G	Baseline	20.6	4	2.2	1.3	24.3
	Proposed	18.2	4.1	2.3	1.5	24.2
RMS	Baseline	20.1	4.1	2.2	1.3	24.4
	Proposed	19	4.5	2.5	1.4	22
Increase		**-1.1**	**0.3**	**0.3**	**0.1**	**-2.4**

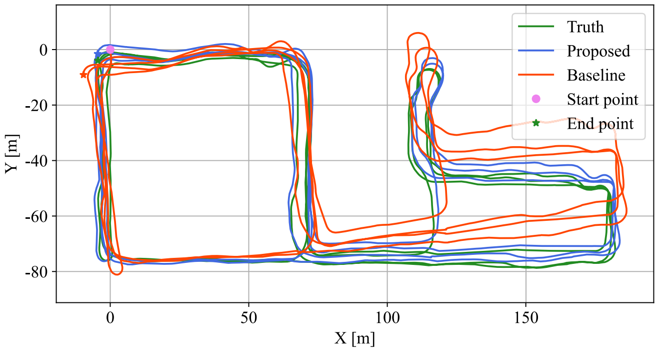


Fig. 7. Trajectories in private *Robot-C* data.

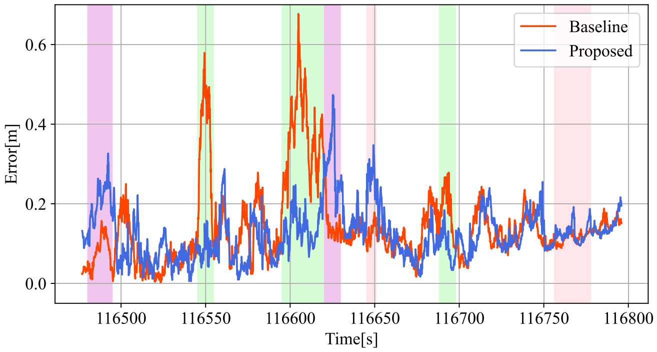


Fig. 8. 10-m RTEs of private *Robot-G* data.

tracked only once is decreased, while those reaching the maximum tracking length are increased. The proposed method assigns low weight in the straight-ahead image region, where feature points can be tracked longer. As a result, the average keypoint tracking length of the proposed method is smaller than that of the baseline method, as shown in Table IV. Nonetheless, the average two-view parallax and total parallax of the proposed method are significantly improved, with the increments of 0.45° and 3.52°, respectively. The proportion of feature points with different tracking lengths in our private dataset exhibits the same improvement as that in the KAIST dataset, which fully proves the enhancement of our method on the features' tracking qualities.

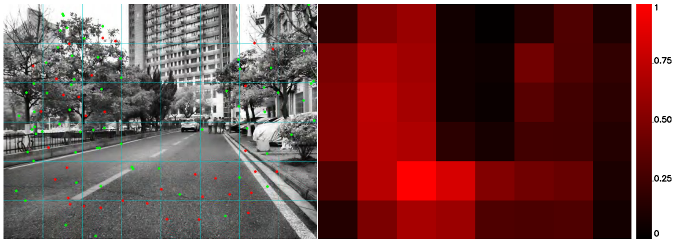
2) Absolute Navigation Accuracy: The absolute pose errors in our private dataset are first analyzed. We plot the trajectories of the *Robot-C* data in Fig. 7. The baseline trajectory diverges from the truth trajectory after the robot moves. In contrast, the trajectory of our proposed method demonstrates better consistency and fitness with the truth trajectory. Table VI statistics the ATEs and AREs of the seven groups of private data. Also benefitting from the larger features' tracking parallax, the proposed method obviously outperforms the baseline method across almost all test groups. In terms of the root-mean-square (rms) value of the seven test groups data, we observe that the proposed method reduces ATE from 3.28 to 2.36 m and reduces ARE from 1.55° to 1.02°. Statistically, the proposed method achieves the improvements of 28% in ATE and 34% in RTE, which strongly verifies the superior navigation performance of the proposed method.

3) Relative Navigation Accuracy: The truth trajectories in our private dataset are acquired from a navigation-grade IMU, which ensures the high-precision relative pose. Therefore, we also evaluate our private dataset's RTEs and RREs of different trajectory segments. We select four kinds of trajectory lengths, that is, 10, 50, 100, and 200 m, to evaluate the relative pose accuracy. Taking the *Robot-G* as an example, the 10-m RTEs of both proposed and baseline methods are plotted in Fig. 8. In this figure, we pick some typical regions and mark them with green, purple, and pink colors. In the green regions, the position error of the baseline method enlarges obviously, while the accuracy of the baseline method in the purple and pink regions slightly outperforms that of the proposed method.

We illustrate the 3-D feature's position and the feature redistribution weight at one specific position within the green regions in Fig. 9. In Fig. 9(a), there are no 3-D feature points on the straight building, as features on the building

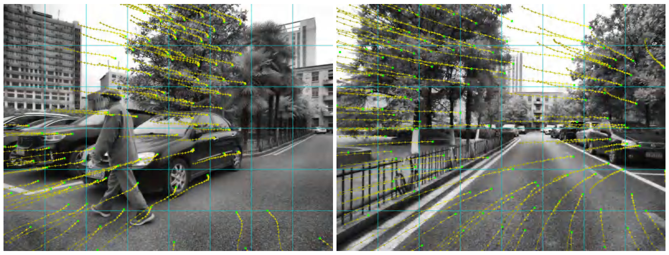
TABLE VII
Relative Pose Error in the Private Robot Dataset

RRE/RTE [deg / %]	Method	Trajectories length			
		10m	50m	100m	200m
Robot-A	Baseline	0.10 / 1.76	0.26 / 0.99	0.42 / 0.82	0.59 / 0.62
	Proposed	0.10 / 1.60	0.28 / 1.00	0.43 / 0.77	0.62 / 0.65
Robot-B	Baseline	0.14 / 2.31	0.45 / 1.78	0.80 / 1.61	1.43 / 1.61
	Proposed	0.13 / 2.13	0.39 / 1.44	0.68 / 1.24	1.15 / 1.20
Robot-C	Baseline	0.13 / 2.04	0.36 / 1.45	0.58 / 1.25	0.99 / 1.10
	Proposed	0.12 / 1.97	0.30 / 1.37	0.43 / 1.20	0.61 / 0.96
Robot-D	Baseline	0.17 / 4.57	0.57 / 3.78	0.96 / 3.30	1.57 / 2.40
	Proposed	0.16 / 2.95	0.52 / 1.98	0.87 / 1.59	1.37 / 1.10
Robot-E	Baseline	0.12 / 2.71	0.27 / 2.09	0.41 / 1.86	0.57 / 1.66
	Proposed	0.11 / 1.95	0.24 / 1.32	0.32 / 1.13	0.45 / 1.01
Robot-F	Baseline	0.12 / 3.17	0.29 / 2.48	0.44 / 2.00	0.66 / 1.55
	Proposed	0.12 / 2.44	0.29 / 1.83	0.44 / 1.58	0.69 / 1.36
Robot-G	Baseline	0.09 / 1.71	0.23 / 1.25	0.37 / 1.06	0.57 / 0.66
	Proposed	0.09 / 1.50	0.25 / 0.85	0.37 / 0.71	0.38 / 0.52
RMS	Baseline	0.12 / 2.42	0.37 / 2.16	0.61 / 1.86	1.00 / 1.49
	Proposed	0.11 / 2.34	0.34 / 1.45	0.54 / 1.22	0.82 / 1.01
Error Decrease		**0.01 / 0.08**	**0.03 / 0.71**	**0.07 / 0.64**	**0.17 / 0.48**



(a) (b)

Fig. 9. (a) Current 3-D features at one position of the green regions; the red points are triangulated in the current frame, and the green points are saved in the feature buffer. (b) Visualization of feature redistribution weight with normalized color scaling.



(a) (b)

Fig. 10. (a) Predicted feature trackings at one position of the purple regions; the green points are the current pixel coordinates, and their connected yellow points are their predicted pixel coordinates. (b) Predicted feature trackings at one position of the pink region.

lack enough parallax. Thus, the corresponding image region exhibits a very low weight, as depicted in Fig. 9(b). In contrast, the proposed method redistributes more features in image regions with enough parallax, including the ground and objects on both sides. Therefore, the contribution of features with larger valid parallax to the navigation state estimation is fully exploited by the proposed methods. As a result, the RTE in the green region of the proposed method is restricted to an average level.

The predicted feature trackings at one position in the purple region and one position in the pink position are shown in Fig. 10. The predicted features' tracking in Fig. 10(a) will be interrupted by the suddenly emerged people, such that many features redistributed by our method cannot be tracked continuously. As a result, the RTE of the proposed method increases. In the pink region, the temporary shading by trees also destroys the predicted features' tracking in Fig. 10(b) and enlarges the RTE of the proposed method slightly.

We count all the relative pose errors in our private dataset in Table VII and calculate the rms pose error of the seven data. The RTEs and RREs of the proposed method outperform that of the baseline method in almost all data and all trajectory lengths. The statistical rms errors of the proposed method are dramatically decreased than that of the baseline method. For example, the 100-m RTE is decreased from 1.86% to 1.22%.

In the relative navigation accuracy evaluation, the sudden appearance of dynamic objects and temporary obstacles disrupts the expected continuous tracking of the feature points, leading to a light degradation in the navigation accuracy of the proposed method. Consequently, the proposed P2GD cannot act out its advantages in the scenes with frequent and sudden appearance of dynamic objects. Since these dynamic objects cannot be consistently tracked and triangulated successfully, they are not taken into account in the feature distribution process. Therefore, the degradation caused by dynamic objects does not persist over an extended period. Despite these challenges, the statistical results in Table VII affirm that the proposed P2GD still significantly enhances navigation accuracy. The thorough solution for this issue will be detecting the dynamic objects and masking them from the feature point distribution process, which is one of our future works.

D. Robustness Test and Analysis

In our experiments, the truth trajectories served as the carrier's prior poses of the proposed method. However, the prior poses, which are output by the path planning module in actual applications, are not perfectly accurate. In order to evaluate the robustness of the proposed method, we add white noise into the truth trajectories of our private dataset to simulate the inaccurate prior poses. Considering that the

TABLE VIII
ABSOLUTE POSE ERROR IN THE PRIVATE ROBOT DATASET WITH DIFFERENT NOISES

Error	baseline	proposed	Position noise intensidy [m]					Yaw noise intensidy [deg]				
			0.05	0.1	0.15	0.2	0.25	0.5	1	1.5	2	2.5
ATE [m]	3.28	2.36	3.02	2.88	2.65	2.69	2.93	2.85	2.89	2.86	3.04	2.88
ARE [deg]	1.55	1.02	1.31	1.32	1.3	1.29	1.29	1.31	1.35	1.31	1.24	1.4

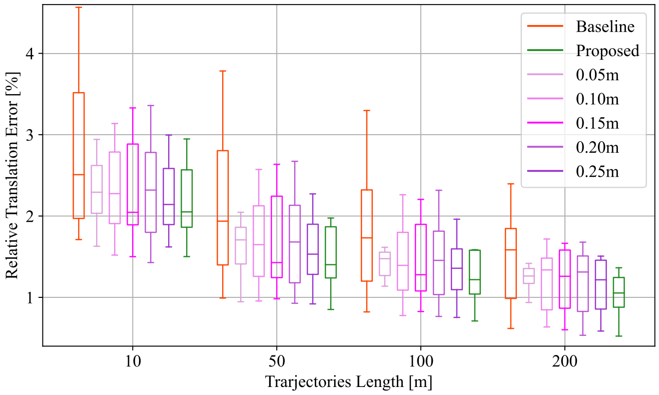


Fig. 11. Boxplots of RTEs with different position noise intensities and different trajectory lengths.

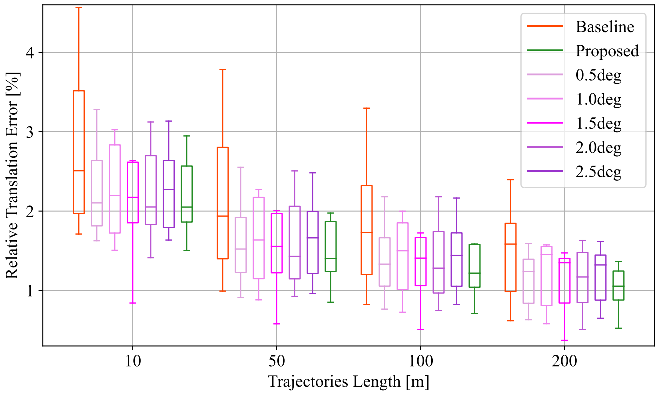


Fig. 12. Boxplots of RTEs with different heading angle noise intensities and different trajectory lengths.

wheeled vehicle has only two degrees of freedom, we add noise specifically to the horizontal position or heading angle. Multiple intensities of position and heading angle noise are tested to comprehensively evaluate the robustness. Specifically, the position noise intensities include 0.05, 0.10, 0.15, 0.20, and 0.25 m, and the heading angle noise intensities are 0.5°, 1.0°, 1.5°, 2.0°, and 2.5°. Considering the robot's maximum velocity of 1.5 m/s, the common heading angular rate of 20°/s, and the prior pose rate of 10 Hz, we determine that the maximum increments in distance and angle provided by the prior pose are 0.15 m and 2°, respectively. Consequently, the intensity of position noise at 0.25 m and heading angle noise at 2.5° is deemed sufficient for the robustness evaluation of our private dataset. To avoid the randomness of random white noise on the results, every test result is averaged from three identical tests.

We compare the navigation accuracy in our private dataset, including RTEs and ATEs, to verify the robustness of our proposed method. The RTEs are calculated with the trajectory lengths of 10, 50, 100, and 200 m. In the RTEs with different noise intensities and different trajectory lengths, the RTEs of the seven groups of data are taken as a collection to plot a boxplot. We plot and compare all the boxplots of the baseline method, the proposed method, and the proposed methods with different noise intensities. Fig. 11 is the boxplots with different horizontal position noises, and Fig. 12 is that with different heading angle noises. Compared with the proposed method, the position errors enlarge with different degrees when adding position error and heading angle error to the prior poses in Figs. 11 and 12. Even though, all RTEs of the proposed method with added certain noise are still obviously smaller than that of the baseline method. Furthermore, the RTEs' enlargement is not positively correlated with the added noise intensity, which further proves the robustness of the proposed method. The statistical results of ATEs and AREs are shown in Table VIII. Although certain strong noise is added to the prior poses of the proposed method, it still demonstrates better absolute translation and rotation accuracy than the baseline method. In general, the proposed method demonstrates superior robustness when the prior poses are inaccurate.

TABLE IX
AVERAGE RUNTIME (ms) IN DIFFERENT TEST DATA

Group	Proposed Method					Baseline
	P2GD	Track	Update	Other	Total	Total
Robot-A	0.74	3.87	2.19	0.37	7.18	6.64
Robot-B	0.86	4.17	3.44	0.38	8.85	7.77
Robot-C	0.81	4.02	2.98	0.38	8.19	7.23
Robot-D	0.91	4.02	3.25	0.39	8.56	7.54
Robot-E	0.86	3.90	3.38	0.38	8.52	7.45
Robot-F	0.85	4.17	3.21	0.38	8.61	7.26
Robot-G	0.81	4.21	3.32	0.37	8.70	7.52
RMS	**0.84**	**4.05**	**3.13**	**0.38**	**8.39**	**7.35**

E. Runtime Analysis

To analyze the real-time performance and the computation burden of the proposed method, we evaluate the runtime of the proposed method on a desktop PC (AMD R7-3700X and 32-GB RAM) using our private robot dataset. The average running times of processing one frame image in different data are summarized in Table IX. In this table, "P2GD" denotes the running time of the proposed method, "Track" is the time taken for feature extraction and tracking, "Update" represents the duration of measurement update, and "Other" includes IMU state propagation and covariance marginalization. The average runtime of the P2GD algorithm is only 0.84 ms, significantly shorter than that of feature extraction and measurement update. Compared with the total runtime of VINS, the runtime of the P2GD algorithm takes quite a small proportion, demonstrating its minimal impact on the real-time performance of

VINS. In terms of the total runtime of the proposed and baseline methods, the total runtime of our proposed method only increases by 1.04 ms, indicating comparable real-time performance in practical applications.

V. Conclusion

Taking the features' future tracking qualities into consideration, this article proposed a P2GD method. The primary objective is to improve the features' tracking parallax and further enhance the localization accuracy of the visual navigation system. Specifically, the carrier's prior poses, together with the 3-D scene represented by the sparse triangulated 3-D features, are employed to predict the features' future tracking of each image region. Then, the feature redistribution algorithm reassigns the feature number into different image regions according to the predicted tracking parallax. Finally, the corresponding number of features will be extracted in every image region and employed in the VINS estimator.

The proposed method is implemented in an MSCKF-based VINS and evaluated with both the public KAIST urban dataset and our private robot dataset. The experiment results consistently indicate that the proposed method demonstrates enhanced tracking length and total tracking parallax in features' tracking qualities and superior navigation accuracy of VINS compared with the baseline grid-based method. Moreover, even adding certain noise to the prior poses intentionally, the proposed method still outperforms the benchmark method in terms of navigation accuracy, yielding superior robustness.

The analysis of relative position error exposes that the sparse 3-D features of the proposed method cannot represent the suddenly emerged dynamic objects and occlusions in the 3-D environment. Therefore, future work will integrate a perception–prediction network into the proposed method, enabling accurate recognition and prediction of dynamic object trajectories. This integration aims to mitigate the influence of moving objects on predicted feature tracking by adjusting the feature distribution weight accordingly.

References

[1] H. Tang, X. Niu, T. Zhang, L. Wang, and J. Liu, "LE-VINS: A robust solid-state-LiDAR-enhanced visual-inertial navigation system for low-speed robots," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–13, 2023.

[2] G. Huang, "Visual-inertial navigation: A concise review," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, Montreal, QC, Canada, May 2019, pp. 9572–9582.

[3] C. Cadena et al., "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[4] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, Jul. 2016, pp. 4104–4113.

[5] T. Qin, P. Li, and S. Shen, "VINS-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[6] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A research platform for visual-inertial estimation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Paris, France, May 2020, pp. 4666–4672.

[7] J. Li et al., "Attention-SLAM: A visual monocular SLAM learning from human gaze," *IEEE Sensors J.*, vol. 21, no. 5, pp. 6408–6420, Mar. 2021.

[8] M. F. Ahmed, K. Masood, and V. Fremont, "Active SLAM: A review on last decade," *Sensors*, vol. 23, no. 18, p. 8097, 2023.

[9] Y. Chen, S. Huang, and R. Fitch, "Active SLAM for mobile robots with area coverage and obstacle avoidance," *IEEE/ASME Trans. Mechatronics*, vol. 25, no. 3, pp. 1182–1192, Jun. 2020.

[10] Y. Park and S. Bae, "Keeping less is more: Point sparsification for visual SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 7936–7943.

[11] O. Bailo, F. Rameau, K. Joo, J. Park, O. Bogdan, and I. S. Kweon, "Efficient adaptive non-maximal suppression algorithms for homogeneous spatial keypoint distribution," *Pattern Recognit. Lett.*, vol. 106, pp. 53–60, Apr. 2018.

[12] S. Gauglitz, L. Foschini, M. Turk, and T. Höllerer, "Efficiently selecting spatially distributed keypoints for visual tracking," in *Proc. 18th IEEE Int. Conf. Image Process.*, Sep. 2011, pp. 1869–1872.

[13] X. Niu, H. Tang, T. Zhang, J. Fan, and J. Liu, "IC-GVINS: A robust, real-time, INS-centric GNSS-visual-inertial navigation system," *IEEE Robot. Autom. Lett.*, vol. 8, no. 1, pp. 216–223, Jan. 2023.

[14] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[15] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021.

[16] H.-J. Liang, N. J. Sanket, C. Fermüller, and Y. Aloimonos, "SalientDSO: Bringing attention to direct sparse odometry," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1619–1626, Oct. 2019.

[17] S. Frintrop and P. Jensfelt, "Attentional landmarks and active gaze control for visual SLAM," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1054–1065, Oct. 2008.

[18] K. Wang, S. Ma, F. Ren, and J. Lu, "SBAS: Salient bundle adjustment for visual SLAM," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–9, 2021.

[19] Z. Wang, H. Chen, S. Zhang, and Y. Lou, "Active view planning for visual SLAM in outdoor environments based on continuous information modeling," *IEEE/ASME Trans. Mechatronics*, vol. 29, no. 1, pp. 237–248, Feb. 2024.

[20] X. Deng, Z. Zhang, A. Sintov, J. Huang, and T. Bretl, "Feature-constrained active visual SLAM for mobile robot navigation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 7233–7238.

[21] A. J. Davison and D. W. Murray, "Mobile robot localisation using active vision," in *Computer Vision—ECCV*, vol. 1407, G. Goos, J. Hartmanis, J. Van Leeuwen, H. Burkhardt, and B. Neumann, Eds., Berlin, Germany: Springer, 1998, pp. 809–825.

[22] Y. Zhao and P. A. Vela, "Good feature matching: Toward accurate, robust VO/VSLAM with low latency," *IEEE Trans. Robot.*, vol. 36, no. 3, pp. 657–675, Jun. 2020.

[23] D. Valiente, L. Payá, L. Jiménez, J. Sebastián, and Ó. Reinoso, "Visual information fusion through Bayesian inference for adaptive probability-oriented feature matching," *Sensors*, vol. 18, no. 7, p. 2041, Jun. 2018.

[24] M. Flores, D. Valiente, A. Gil, O. Reinoso, and L. Payá, "Efficient probability-oriented feature matching using wide field-of-view imaging," *Eng. Appl. Artif. Intell.*, vol. 107, Jan. 2022, Art. no. 104539.

[25] R. T. Rodrigues, M. Basiri, A. P. Aguiar, and P. Miraldo, "Low-level active visual navigation: Increasing robustness of vision-based localization using potential fields," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2079–2086, Jul. 2018.

[26] Z. Zhang and D. Scaramuzza, "Beyond point clouds: Fisher information field for active visual localization," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 5986–5992.

[27] Z. Zhang and D. Scaramuzza, "Fisher information field: An efficient and differentiable map for perception-aware planning," Aug. 2020, *arXiv:2008.03324*.

[28] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed., New York, NY, USA: Cambridge Univ. Press, 2003, p. 673.

[29] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Rome, Italy, Apr. 2007, pp. 3565–3572.

[30] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim, "Complex urban dataset with multi-level sensors from highly diverse urban environments," *Int. J. Robot. Res.*, vol. 38, no. 6, pp. 642–657, May 2019.

[31] M-Grupp. (2017). *Evo: Python Package for the Evaluation of Odometry and SLAM*. [Online]. Available: https://github.com/MichaelGrupp/evo

[32] H. Tang, T. Zhang, X. Niu, J. Fan, and J. Liu, "Impact of the Earth rotation compensation on MEMS-IMU preintegration of factor graph optimization," *IEEE Sensors J.*, vol. 22, no. 17, pp. 17194–17204, Sep. 2022.