PLPF-VSLAM: An Indoor Visual SLAM with Adaptive Fusion of Point-Line-Plane Features

Jinjin Yan, Youbing Zheng*, Jinquan Yang, Lyudmila Mihaylova, Weijie Yuan, Fuqiang Gu

***Abstract*—Simultaneous Localization and Mapping (SLAM) is required in many areas and especially visual-based SLAM (VSLAM) due to the low cost and strong scene recognition capabilities Conventional VSLAM relies primarily on features of scenarios, such as point features, which can make mapping challenging in scenarios with sparse texture. For instance, in environments with limited (low- even non-) textures, such as certain indoors, conventional VSLAM may fail due to a lack of sufficient features. To address this issue, this paper proposes a VSLAM system called visual SLAM that can adaptively fuse Point-Line-Plane features (PLPF-VSLAM). As the name implies, it can adaptively employ different fusion strategies on the point-line-plane features for tracking and mapping. In particular, in rich-textured scenes, it utilizes point features, while in non-/low-textured scenarios, it automatically selects the fusion of point, line, and/or plane features. PLPF-VSLAM is evaluated on two RGB-D benchmarks, namely the TUM datasets and the ICL_NUIM datasets. The results demonstrate the superiority of PLPF-VSLAM compared to other commonly used VSLAM systems. When compared to ORB-SLAM2, PLPFVSLAM achieves an improvement in accuracy of approximately 11.29%. The processing speed of PLPF-VSLAM outperforms PL(P)-VSLAM by approximately 21.57%.**

***Index Terms*—Visual SLAM; Tracking; Mapping; Non-/Low-textured Scenarios.**

I. Introduction

SIMULTANEOUS Localization and Mapping (SLAM) is the process of constructing or updating a model (map) of an environment without prior knowledge while locating itself in it simultaneously [1; 2; 3]. In other words, SLAM can produce two results, environment models (maps), and locations of agents. Agents of SLAM can be people, robots, or drones, which are equipped with external sensors, such as LiDAR, or cameras. With the wide application of robots, SLAM has become a key technology for the autonomous navigation of robots. According to the main external sensor for localization and mapping, SLAM systems are generally categorized into three types: LiDAR-based SLAM, visual-based SLAM (VSLAM), and multi-sensor fusion SLAM. While LiDAR is commonly used for SLAM, cameras (including monocular, stereo, and RGB-D cameras) offer advantages such as a simple structure, low cost, strong scene recognition ability, and the ability to capture rich texture information. Consequently, VSLAM has gained significant attention in both academic and industrial fields [4].

SLAM has been widely used in various fields such as indoor autonomous navigation [5], virtual reality (VR) [6], Augmented Reality (AR) [7]. Conventional VSLAM typically relies on point features to track the movements of agents and build maps, as this approach is simple and effective. However, because images of non-textured or low-textured environments (Figure 1) lack sufficient point features, conventional VSLAM may suffer from some issues, such as tracking loss [8], failure in loop detection stage [9]. To address these challenges, researchers have been exploring alternative approaches. For instance, to deal with tracking loss issues, they attempted to develop point-line-plane-based VSLAM systems that can combine line and plane features [13]. Additionally, some researchers tried to employ deep learning-based techniques for loop detection [10; 11; 12]. This paper only focuses on the tracking loss issue, because loop detection is not always a necessary step in all scenarios. Considering the current attempts still face limitations in low-/non-textured indoor environments. Consequently, further investigation and development of VSLAM systems that can effectively operate in such scenarios remain essential.

Inspired by that the conventional point-based VSLAM can handle scenes with rich textures, and the structures of indoor space (such as walls are perpendicular to the floors and ceilings) can be used as an effective supplement a feature in non-/low-textured areas, we propose a VSLAM system with adaptive fusion of point-line-plane features (PLPF-VSLAM). It is able to adaptively select proper feature fusion strategies for localization and mapping, according to the texture richness of scenes. For scenes with rich texture features, the system will fuse point and line for tracking, and store such features in the map for mapping. For scenes lacking texture features, it will empty the point-line-plane fusion.

This paper is organized as follows. Section 2 provides an overview of the current research on VSLAM. Section 3 presents the PLPF-VSLAM. Section 4 evaluates the PLPF-VSLAM by using two RGB-D benchmarks, TUM dataset and ICL_NUIM dataset. Upon the results, conclusions and future work are drawn in the final section.

II. RELATED WORK

In recent years, many different VSLAM systems have been presented. According to the type of feature utilized, VSLAM

J. Yan, Y. Zheng and J. Yang are with Qingdao Innovation and Development Center, Harbin Engineering University, Qingdao 266400, China; (jinjin.yan, bing_164, yjq980314)@hrbeu.edu.cn

L. Mihaylova is with the Department of Automatic Control and Systems Engineering, The University of Shefeld, Shefeld S1 3JD, U.K. l.s.mihaylova@sheffield.ac.uk

W. Yuan is with the Department of Electrical and Electronic Engineering, Southern University of Science and Technology, Shenzhen 518055, China; yuanwj@sustech.edu.cn

F. Gu is with the College of Computer Science, Chongqing University, 400044, Chongqing, China; gufq@cqu.edu.cn

Manuscript received January XX, 2023; revised XX XX, 2023.

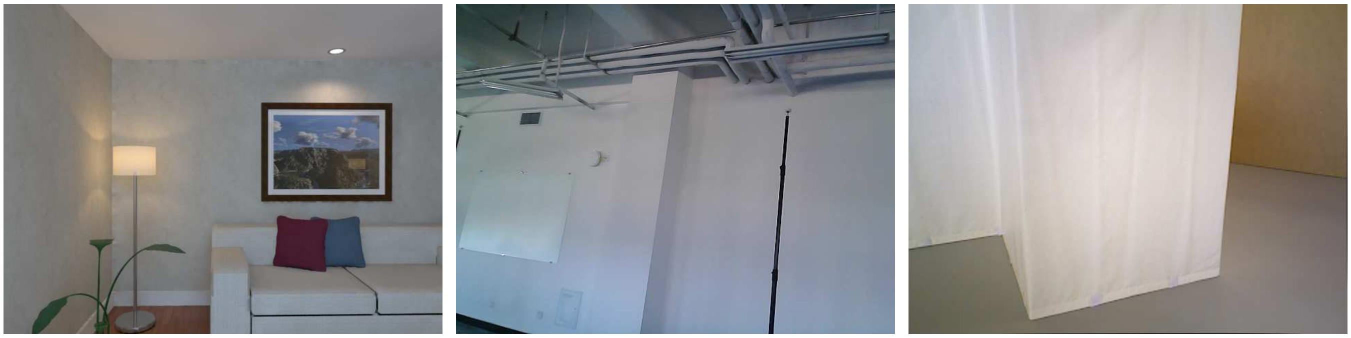

(a) Rich-textured scene (b) Low-textured scene (c) Non-textured scene

Fig. 1. Three indoor scenes with different levels of texture richness.

can be categorized into three types: (i) point-feature-based VSLAM, P-VSLAM, (ii) line feature-based VSLAM, L-VSLAM and (iii) VSLAM based on fusion of point, line, and/or plane features, PL(P)-VSLAM. It should be noted that it is difficult for a SLAM system to fulfill the accuracy requirements of mapping and tracking solely based on plane features. Thus, plane features are rarely used alone but are usually employed together with the other two types of features.

A. P-VSLAM

P-VSLAM systems primarily rely on point features for tracking and mapping. The commonly used point features in P-VSLAM include Scale-Invariant Feature Transform (SIFT) [14], Speeded Up Robust Features (SURF) [15], and Oriented FAST and Rotated Brief (ORB) [16]. The processing methods of point features are highly developed, which makes P-VSLAM becomes the current mainstream of VSLAM. There are several classic P-VSLAM systems, such as PTAM system [17], MonoSLAM [18], SVO (Semi-direct Visual Odometry) [19]. In general, the PTAM system is regarded as the prototype of P-VSLAM. This system brought three major innovations to SLAM systems: it (i) replaces the traditional Kalman filtering with nonlinear optimization; (ii) employs a keyframe mechanism. That is, the system only needs to process the most representative image, rather than each frame of an image, which greatly improves the efficiency of the calculation; (iii) to meet real-time requirements, separates the tracking and mapping process by using a multi-threading mechanism. However, because without considering the global loop closing, the PTAM system is only applicable to small scenarios, and its tracking process is easy to fail. Afterward, an open-source VSLAM based on point feature named ORB-SLAM was released, [20]. It employs the ORB feature, loop closing detection mechanism, and the BOW model, which forms a very complete framework of point-feature-based VSLAM. Because ORB-SLAM is prone to tracking loss when the camera rotates violently, on the basis of this version, the authors released ORB-SLAM2 after two years [21]. This system can support monocular, stereo, and RGB-D cameras. It realizes real-time localization and mapping, in which the accuracy of localization is at a centimeter-level. Hence, it is the most typical P-VSLAM system. But it should be mentioned that this system is very sensitive to dynamic objects and is easy to have tracking loss in dynamic scenes.

B. L-VSLAM

In response to the limitations of P-VSLAM in non-/low-textured scenarios, researchers began studying L-VSLAM. Such a VSLAM utilizes line features as the primary source of information for tracking and mapping. For instance, [22] applied the line feature in the SLAM system, in which a line is represented by two endpoints. Yet, this system is only suitable for small scenes where entire line segment can be fully captured. To address this limitation, [23] applied infinitely long line segments to large scenes. This practice effectively expands the applicable scenarios and further makes the process of matching line segments between frames easier. However, the initialization of line segments in space may fail in scenarios with a large landmark space. Other than that, [24] proposed a 3D line-based stereo VSLAM system, which employs two different representations to parameterize 3D lines to obtain a better result. Inevitably, this system also has shortcomings. In particular, it is time-consuming in the straight line tracking process as it is based on the optical flow method.

C. PL(P)-VSLAM

PL(P)-VSLAM incorporates a fusion of point, line, and/or plane features to enhance tracking and mapping accuracy. During the extraction and matching of line features, several challenges may arise, such as unclear endpoint positions and weak set constraints, leading to a high number of mismatches. As a result, researchers shifted their focus towards fusion-based VSLAM, which typically includes three types: point-line (PL), point-plane (PP), and point-line-plane (PLP).

There are several PL-based VSLAM systems, such as LSD-SLAM [25], monocular-based PL-SLAM [26; 27], Point-Line Fusion (PLF)-SLAM [28], PLI-VINS [29]. The LSD-SLAM [25] applied the direct method to semi-dense monocular in SLAM, which achieved semi-dense scene reconstruction on the CPU. But, this system is prone to tracking loss when the camera moves quickly. The monocular-based PL-SLAM proposed by [27] utilized fusion of point and line features to the entire SLAM process. This system addresses tracking and matching problems of specific line segments by removing outliers based on the comparisons of length and orientation of line features. Similarly, [30] proposed a low-drift monocular SLAM method for indoor scenes. In this system, the estimation of rotation and translation are decoupled to reduce long-term drift in indoor scenarios. In particular, it estimates a drift-free rotation between cameras by using spherical mean-shift clustering and a weak Manhattan world hypothesis [31]. And then, the translation between the cameras is calculated based on the features of points and lines.

VSLAM that uses plane features generally include PP-based [32; 33; 34] and PLP-based VSLAM [35; 36; 37]. One of the typical PP-based VSLAM systems was proposed by [33], which takes the data from RGB-D camera as the input to do localization and mapping in a low-textured scenario. This system improves its accuracy and robustness by employing structural imformation in the whole process. However, the system assumes that plane edges should be intersections of vertical planes, limiting its applicability in scenarios with inclined planes. By fusion features of point, line, and plane, the SLAM system presented by [36] decouples rotation and translation, and then obtains the rotation of object drift by constructing a Manhattan world. This practice further improves the accuracy of the system. Meanwhile, on the basis of an instance-based meshing strategy, this system constructed dense maps by dividing plane instances independently. However, the initialization of building a Manhattan world needs three pairs of perpendicular planes or lines. Thence, users need to consider whether the scenario meets such a specific condition before using it. PLP-SLAM [38] tightly incorporates the semantic and geometric features (point, line and plane features) to boost both frontend pose tracking and backend map optimization. However, this method does not perform well in low-textured environments. UPLP-SLAM [39] designed a mutual association scheme for data association of point, line and plane features, which not only considers the correspondence of homogeneous features (i.e., point-point, line-line and plane-plane pairs), but also includes the association of heterogeneous features (i.e., point- line, point-plane and line-plane pairs). By considering these cross-feature associations, UPLP-SLAM aims to improve the accuracy of the SLAM system, even in low-textured environments.

III. PLPF-VSLAM: VSLAM WITH ADAPTIVE FUSION OF POINT-LINE-PLANE FEATURES

PLPF-VSLAM can adaptively select fusion strategies according to the different characteristics of the scenarios. As shown in Figure 2, PLPF-VSLAM has the same framework as the conventional VSLAM [21], which includes three threads: tracking, local mapping, and loop closing. Compared with conventional VSLAM, the improvements happen in the tracking threads. In particular, taking the numbers of matched features as a reference. Four new modes (Point Tracking Mode, Point-Line Tracking Mode, Point-Plane Tracking Mode and Point-Line-Plane Tracking Mode) are adaptively selected for tracking process.

A. Tracking

In the PLPF-VSLAM system, RGB and depth images are utilized as inputs. The tracking process involves estimating the pose transformation between two frames of images. Initially, point and line features are extracted from the RGB images, while plane features are extracted from the depth images. Meanwhile, incorrect feature matches are eliminated to ensure accuracy. Once the feature extraction is completed, the system constructs various projection error functions based on the matching results and pre-defined thresholds. These projection error functions capture the differences between the projected and the actual observed features. By optimizing these projection errors, the system obtains the pose estimation results, which represent the transformation between the two frames of images.

1) Feature Extraction and Matching: ORB features are commonly used in VSLAM systems due to their desirable characteristics, such as invariance to rotation and scale, fast extraction, and efficient matching. These characteristics contribute to improved efficiency and performance in many scenarios. However, in non-textured or low-textured environments, the effectiveness of ORB features may be limited because they struggle to extract sufficient point features for accurate pose estimation. With that in mind, the PLPF-VSLAM adds line feature extraction based on Line-Segment-Detector (LSD) approach [40], and uses Line Band Descriptor (LBD) [41] to describe the feature of line segments.

Indoor scenarios have a large number of non/low-textured planes, but they show many structural features (e.g., vertical, parallel). Such features also can be employed to improve the stability of a VSLAM system. The approach presented in [42] is employed to extract plane features (figure 3), which includes three steps: (i) divide the point cloud in the image into N nodes and remove the nodes with missing or discontinuous depth information, (ii) cluster the eigenvalues of each pixel in the image, and cluster the continuous blocks with feature differences within the threshold range into the same segmentation block, (iii) carry out iterative optimization for each pixel to output parameters of plane features.

In this paper, a plane is described in the Hessian form, i.e., $\pi = (n^{\mathrm{T}}, d)^{\mathrm{T}}$. In this equation, $n = (n_x, n_y, n_z)^{\mathrm{T}}$, which is the normal vector of the plane; d is the distance from the camera's optical center to the plane. In the process of matching two plane features, two values are needed: one is the angle of the normal vector between them, and another is the d of them. Two conditions are used to determined if two planes can be matched. One is if the angle between normal vectors of the two plane is less than a threshold (i.e., $|\theta_1 - \theta_2| \leq t_\theta$), and another is if the distance between the two planes is less than a threshold (i.e., $|d_1 - d_2| \leq t_d$).

2) Pose Estimation: During the pose estimation process, the detected 3D points, lines, and planes from the previous frame are projected onto the current frame. This projection is done using the estimated pose transformation between the two frames. By projecting the features, their positions in the current frame can be estimated. To evaluate the accuracy of the pose estimation, a re-projection error is calculated by comparing the projected features with the corresponding features that

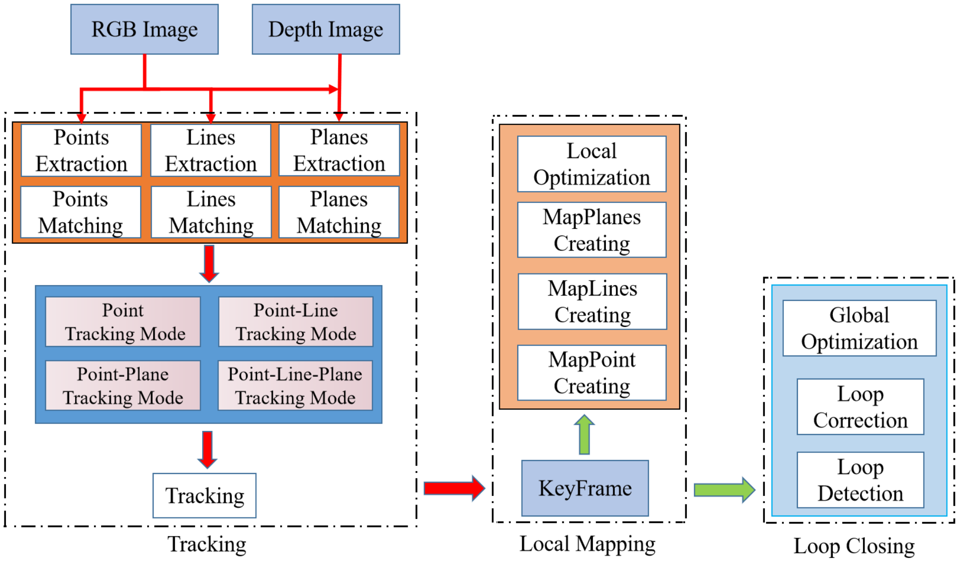
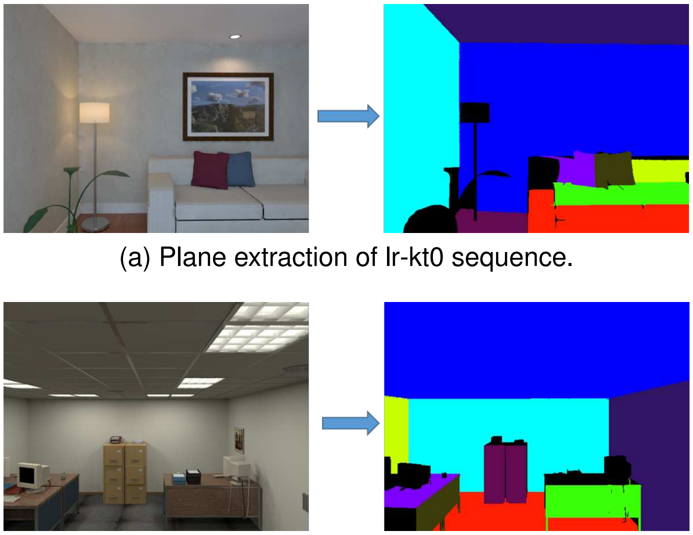


Fig. 2. The framework of the PLPF-VSLAM.



(a) Plane extraction of lr-kt0 sequence.

(b) Plane extraction of of-kt0 sequence.

Fig. 3. Plane extraction of two sequences in the ICL_NUIM dataset.

detected directly in the current frame. This re-projection error is used to construct an error function, which represents the differences between the projected and observed features. This error will be further minimized during the optimization process to obtain the optimal pose estimation.

For point features, the re-projection error function is Equation (1):

$$e_p = u_i - \frac{1}{s_i} K T_{cw} P_i \tag{1}$$

where u_i is the feature point corresponding to the 3D point in the current frame; P_i represents the 3D point in the world coordinate system; K indicates the camera internal parameters, and T_{cw} denotes the transformation matrix from the world coordinate system to that of the camera.

The Jacobian matrix of Equation (1) on T_{cw} is Equation (2), while that on P_i is Equation (3):

$$\begin{aligned} \frac{\partial e_p}{\partial \delta\xi} &= \frac{\partial e_p}{\partial P'} \frac{\partial P^{'}}{\partial \delta\xi} \\ &= \begin{bmatrix} \frac{-f_x}{Z} & 0 & \frac{f_x X}{Z^2} \\ 0 & \frac{-f_y}{Z} & \frac{f_y Y}{Z^2} \end{bmatrix} \begin{bmatrix} I & -P'^{\wedge} \end{bmatrix} \\ &= \begin{bmatrix} \frac{-f_x}{Z} & 0 & \frac{f_x X}{Z^2} & \frac{f_x XY}{Z^2} & \frac{-f_x Z^2 - f_x X^2}{Z^2} & \frac{f_x Y}{Z} \\ 0 & \frac{-f_y}{Z} & \frac{f_y Y}{Z^2} & \frac{f_y Z^2 + f_y Y^2}{Z^2} & \frac{-f_y XY}{Z^2} & \frac{-f_y X}{Z} \end{bmatrix} \end{aligned} \tag{2}$$

$$\begin{aligned} \frac{\partial e_p}{\partial P_i} &= \frac{\partial e_p}{\partial P'} \frac{\partial P^{'}}{\partial P_i} \\ &= -\begin{bmatrix} \frac{f_x}{Z} & 0 & -\frac{f_x X}{Z^2} \\ 0 & \frac{f_y}{Z} & -\frac{f_y Y}{Z^2} \end{bmatrix} R \end{aligned} \tag{3}$$

where $P^{'}$ represents the coordinate of P_i in the camera coordinate system, and R denotes the rotation matrix from the world coordinate system to that of the camera.

For line features, we formulate the re-projection error function based on the point-to-line distance between l and two endpoints of projected line from the matched 3D line in the key-frame. For each endpoint P, the re-projection error can be noted as Equation (4):

$$e_L = l^{\mathrm{T}} K T_{cw} P \tag{4}$$

where K is the internal parameters of the camera; T_{cw} represents the transformation matrix from the world coordinate system to that of camera; P denotes the endpoint of the 3D line segment; l is the coefficients of the 2D line equation.

The normalized line of a line feature is Equation (5):

$$l = \frac{l_s \times l_e}{|l_s \times l_e|} = (l_a, l_b, l_c) \tag{5}$$

The Jacobian matrix of Equation (4) on T_{cw} is Equation (6):

$$
\begin{aligned}
\frac{\partial e_L}{\partial \delta \xi} &= \frac{\partial e_L}{\partial P'} \frac{\partial P^{'}}{\partial \delta \xi} \\
&= \begin{bmatrix} \frac{l_a f x}{z} & \frac{l_b f y}{z} & -\frac{l_a x f x + l_b f y y}{z^2} \end{bmatrix} \begin{bmatrix} I & -P'^{\wedge} \end{bmatrix} \\
&= \begin{bmatrix} \frac{l_a f x}{z} & \frac{l_b f y}{z} \\ -\frac{l_a f x x + l_b f y y}{z^2} & -\frac{l_b f y z^2 + l_a f x x y + l_b f y y^2}{z^2} \\ -\frac{l_a f x z^2 + l_a f x x^2 + l_b f y x y}{z^2} & \frac{-l_a f x y + l_b f y x}{z} \end{bmatrix}_{1 \times 6}
\end{aligned}
\tag{6}
$$

The Jacobian with respect to P is Equation (7):

$$
\begin{aligned}
\frac{\partial e_L}{\partial P} &= \frac{\partial e_L}{\partial P'} \frac{\partial P^{'}}{\partial P} \\
&= \begin{bmatrix} \frac{l_a f x}{z} & \frac{l_b f y}{z} & -\frac{l_a x f x + l_b f y y}{z^2} \end{bmatrix} R
\end{aligned}
\tag{7}
$$

where $P^{'}$ is the coordinate of P in the camera coordinate system.

A plane in the Hessian form has four parameters, but that in 3D space only has three degrees of freedom. Thus, to address this over-parameterization gap, we denote the unit normal vector by ϕ and ψ to change its representation, where ϕ and ψ are the azimuth and elevation angles of the normal. Then, a plane is represented as a minimized parametric form with only three parameters, i.e., it can be represented as Equation (8) [33]:

$$
\tau = q(\pi) = (\phi = \arctan \frac{n_y}{n_x}, \psi = \arcsin n_z, d)^{\mathrm{T}}
\tag{8}
$$

The re-projection error is Equation (9):

$$
e_\pi = q(\pi_m) - q(T_{cw}^{-T} \pi_w)
\tag{9}
$$

where π_m is the observed value of the corresponding plane in the current frame; π_w is the 3D plane in the world coordinate system; T_{cw} is the transformation matrix from the world coordinate system to that of camera.

The Jacobian matrix of the re-projection error (Equation (9)) on T_{cw} is Equation (10) [33]:

$$
\begin{aligned}
\frac{\partial e_\pi}{\partial \delta \xi} &= \frac{\partial e_\pi}{\partial \pi_c} \frac{\partial \pi_c}{\partial \delta \xi} \\
&= \begin{bmatrix} \frac{n_{cy}}{n_{cx}^2 + n_{cy}^2} & \frac{-n_{cx}}{n_{cx}^2 + n_{cy}^2} & 0 & 0 \\ 0 & 0 & \frac{-1}{\sqrt{1 - n_{cz}^2}} & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 0 & -(n_c)^{\wedge} \\ -(n_c)^{\mathrm{T}} & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 & 0 & \frac{n_{cx} n_{cz}}{n_{cx}^2 + n_{cy}^2} & \frac{n_{cy} n_{cz}}{n_{cx}^2 + n_{cy}^2} & -1 \\ 0 & 0 & 0 & -\frac{n_{cy}}{\sqrt{1 - n_{cz}^2}} & -\frac{n_{cx}}{\sqrt{1 - n_{cz}^2}} & 0 \\ n_{cx} & n_{cy} & n_{cz} & 0 & 0 & 0 \end{bmatrix}
\end{aligned}
\tag{10}
$$

The Jacobian with respect to π_w is Equation (11):

$$
\begin{aligned}
\frac{\partial e_\pi}{\partial \pi_w} &= \frac{\partial e_\pi}{\partial \pi_c} \frac{\partial \pi_c}{\partial \pi_w} \\
&= \begin{bmatrix} \frac{n_{cy}}{n_{cx}^2 + n_{cy}^2} & \frac{-n_{cx}}{n_{cx}^2 + n_{cy}^2} & 0 & 0 \\ 0 & 0 & \frac{-1}{\sqrt{1 - n_{cz}^2}} & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} R^{\mathrm{T}} & 0 \\ -t^{\mathrm{T}} R & 1 \end{bmatrix}
\end{aligned}
\tag{11}
$$

where $\pi_c = (n_{cx}, n_{cy}, n_{cz}, d)^{\mathrm{T}}$ is the plane in the coordinate system of camera.

After obtaining the re-projection error of each feature, we start to construct the optimization objective function based on the least squares. The construction of different objective functions for different scenarios is based on their richness of features. For scenarios with rich textures, we choose the P-VSLAM. For other scenes with insufficient features, we use the fusion of four modes: P-VSLAM, PL-VSLAM, PP-VSLAM, and PLP-VSLAM according to the number of features in the scenario. The criteria for distinguishing if a scenario is non-/low-textured or rich of textures is the number of matched point-line-plane features (n_p, n_l, n_π). The objective function is Equation (12):

$$
T_{cw} = \begin{cases}
argmin(F1), & n_p \epsilon [\alpha_1, \alpha_2], n_l \epsilon [\beta_1, \beta_2], n_\pi \epsilon [\gamma_1, \gamma_2] \\
 & \cup n_p \epsilon [\alpha_2, \alpha_3], n_l \epsilon [\beta_1, \beta_3], n_\pi \epsilon [\gamma_1, \gamma_2] \\
argmin(F1 + F2), & n_p \epsilon [\alpha_1, \alpha_2], n_l \epsilon [\beta_2, \beta_4], n_\pi \epsilon [\gamma_1, \gamma_2] \\
 & \cup n_p \epsilon [\alpha_2, \alpha_3], n_l \epsilon [\beta_3, \beta_4], n_\pi \epsilon [\gamma_1, \gamma_2] \\
argmin(F1 + F3), & n_p \epsilon [\alpha_4, \alpha_1], n_l \epsilon [\beta_1, \beta_3], n_\pi \epsilon [\gamma_1, \gamma_2] \\
 & \cup n_p \epsilon [\alpha_4, \alpha_1], n_l \epsilon [\beta_3, \beta_4], n_\pi \epsilon [\gamma_3, \gamma_2] \\
argmin(F1 + F2 + F3), & n_p \epsilon [\alpha_5, \alpha_4], n_l \epsilon [\beta_1, \beta_4], n_\pi \epsilon [\gamma_1, \gamma_2] \\
 & \cup n_p \epsilon [\alpha_4, \alpha_1], n_l \epsilon [\beta_3, \beta_4], n_\pi \epsilon [\gamma_1, \gamma_3]
\end{cases}
\tag{12}
$$

where α_i, β_i, γ_i are the numbers of matched point, line, and plane features; $F1$, $F2$, and $F3$ represents the objective function of point, line and plane features, respectively. $F1$, $F2$, and $F3$ are expressed as Equation (13):

$$
\begin{cases}
F1 = \sum H_p e_{p_i}^T \Gamma_p^{-1} e_{p_i}, \\
F2 = \sum H_l e_{L_j}^T \Gamma_l^{-1} e_{L_j}, \\
F3 = \sum H_\pi e_{\pi_k}^T \Gamma_\pi^{-1} e_{\pi_k},
\end{cases}
\tag{13}
$$

where H_p, H_l and H_π are Huber functions of point, line and plane, respectively; Γ_p, Γ_l and Γ_π are the covariance matrix associated to the scale at which the key points, line endpoints, and planes were detected, respectively.

IV. Local Mapping

The local mapping thread plays a role in the construction of the local map, leveraging the keyframes generated within the tracking thread to estimate the precise pose of each keyframe, along with the associated map points, lines, and planes. These features are subsequently assimilated into the local map. In the course of processing a keyframe, the Bundle Adjustment (BA) algorithm is employed with the aim of mitigating the local pose error. BA optimizes the poses and positions of the map features by minimizing the re-projection error, ultimately yielding a local map of heightened accuracy and consistency.

A local map in PLPF-VSLAM primarily consists of keyframes and their associated map points, lines, and plane features. The construction of the local map involves fusing different types of features based on their richness in the given scenarios. Initially, the keyframe generated by the tracking thread is added to the local map. Subsequently, a selection process ensues to ascertain the inclusion of specific point, line, and plane features within the map. If a feature can be reliably tracked across no fewer than three keyframes, it will be considered stable and thus included in the local map. Conversely, if a feature cannot be tracked consistently, it will be removed from the map. Once the keyframes and their corresponding map features are added to the local map, optimization is carried out using the BA algorithm. This optimization practice serves to refine the camera poses and the positions of the point, line, and plane features in the local map, with the overarching objective of minimizing the reprojection error. By optimizing the local map, the accuracy and consistency of the map representation are improved, leading to heightened reliability of localization and mapping outcomes.

V. Loop Closing

In the field of VSLAM, relying solely on the pose transformation calculation between two adjacent keyframes leads to an inevitable accumulation of errors. This accumulation, in turn, renders the system unreliable over extended duration of operation. Therefore, it is critical to eliminate the accumulated error by performing pose optimization in loop closing. The loop closing of PLPF-VSLAM is based on the approach presented in [21]. This process mainly includes two components: loop detection and loop correction.

The loop detection is to detect the loop keyframe by using a BOW model [43]. To determine whether the current keyframe can be used as a loop keyframe, we need to calculate the similarity transformation from the current keyframe to the loop keyframe; on the basis of similarity transformation, obtain the translation and rotation between the current and the loop keyframe; and perform projection and matching according to the translation and rotation to detect the reliability of the current loop.

The loop correction starts by adjusting all camera poses based on a known similarity transformation. Then, the adjusted pose is employed to update the map points that correspond to the connected keyframes. Meanwhile, it fuses the map point of the loop keyframe with that of the current keyframe. Afterward, these fused map points are further re-projected and re-matched to establish new matching relationships, and according to the new relationships, the poses of all cameras are optimized based on the pose graph. Finally, loop correction is finished after using the full BA algorithm.

VI. Experiments

To evaluate the performances of PLPF-VSLAM, we utilized two commonly used RGB-D benchmark datasets, Technical University of Munich (TUM) dataset [44] and the Imperial College London and National University of Ireland Maynooth (ICL_NUIM) dataset [45]. The former includes a large set of data sequences containing both RGB-D data from the Microsoft Kinect and ground truth pose estimates from the motion capture system. Notably, the accuracy of the ground truth measurements attains a millimeter-level precision. The latter collects the image sequences in synthetic indoor spaces, mainly including living room and office. This dataset can provide RGB images, depth images, and ground truth of camera poses. TUM dataset not only shows rich-textured scenes, but also low-/non-textured scenarios. More important, it has real trajectories during data collection. Therefore, TUM dataset is employed to determine parameters for PLPF-VSLAM, while the ICL_NUIM dataset for testing.

PLPF-VSLAM is compared with five other VSLAM systems, including ORB-SLAM2 [21], PL-SLAM [27], LSD-SLAM [25], Planar-SLAM [36], L-SLAM [46], in which the first one is based on point features, the second and third are based on fusion of point and line features, and the last two are based on fusion of point-line-plane features. It should be noted that the performances of the five VSLAM systems come from the literature. The computer for this experiment is equipped with Intel Core i7-7500U (3.5GHz) and 12G memory.

The performance of the four modes (P-VSLAM, PL-VSLAM, PP-VSLAM, and PLP-VSLAM) is evaluated by the Root Mean Square Errors (RMSE) of the absolute trajectory error (ATE) (Equation (14)).

$$RMSE_{ATE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\|trans(\hat{x}_i) - trans(x_i)\|^2} \tag{14}$$

where $\hat{x}_i$ represents the keyframe trajectory estimated by a VSLAM, and x_i denotes the real trajectory of the camera.

Other than the RMSE, we propose another criterion to evaluate the overall performance (O_P) of the four modes. The O_P is determined by the mean tracking time of each frame, and the RMSE. For a given scenario, the mode that has the minimum of O_P will be selected. O_P can be calculated by Equation (15):

$$O_P = \eta \times \frac{tm_i}{\sum_{i=1}^{4} tm_i} + \lambda \times \frac{RMSE_i}{\sum_{i=1}^{4} RMSE_i} \tag{15}$$

where tm_i denotes the mean tracking time of each frame of the ith mode; $RMSE_i$ represents the RMSE of the ith mode. i = 1, 2, 3, and 4 corresponds to the four modes P, PL, PP and PLP; and η and λ are the weights of mean tracking time of each frame, and RMSE, respectively.

A. Determination of Parameters

In this part, we attempted to determine the parameters of PLPF-VSLAM by processing the TUM dataset. The thresholds for judging if two planes are matched are set on the basis of the research [33; 36]. In particular, the threshold of the angle between normal vectors of the two plane is set as 10^o and the threshold the distance between the two planes is set as $0.1m$ (i.e., $t_\theta = 10^o$, $t_d = 0.1m$). Furthermore, we leverage parallel and perpendicular relationships of the map planes as additional constraints during the tracking process.

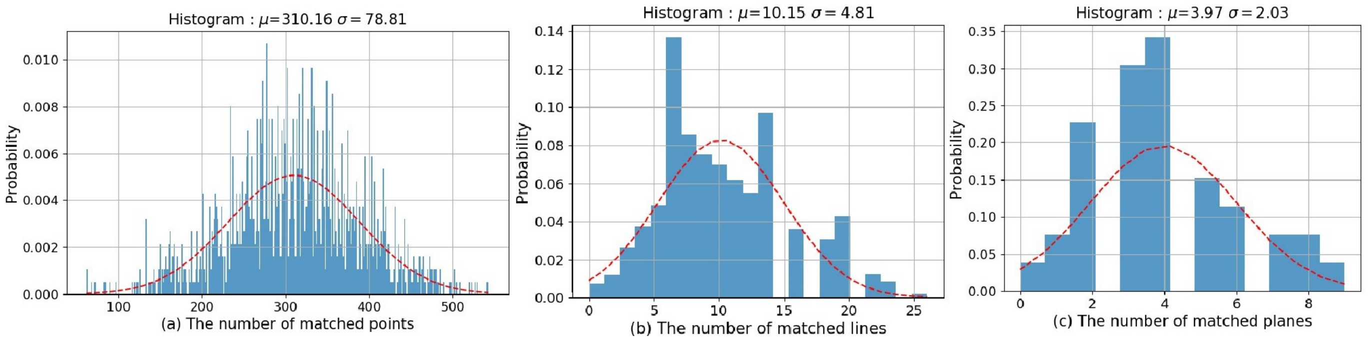
The parameters of the four modes (P-VSLAM, PL-VSLAM, PP-VSLAM, and PLP-VSLAM) on the ten selected sequences from the TUM dataset are shown in Table I. These sequences were chosen specifically because they contain plane features, allowing for a comprehensive evaluation of the different modes. For each sequence, three parameters were computed for each mode: the mean tracking time per frame, the RMSE, and the O_P value. These parameters provide insights into the performance of each mode in terms of computational efficiency and accuracy.

1) Weight Calculation: To achieve a balance between processing speed and accuracy in PLPF-VSLAM, it is crucial to carefully select suitable values for the parameters η and λ. These two parameters play a significant role in determining the overall performance metric O_P. The parameter η represents the weight assigned to the mean tracking time per frame, while the parameter λ represents the weight assigned to the RMSE. Finding the right balance between these two parameters is essential for optimizing the overall performance of the system.

TABLE I

PARAMETERS OF THE FOUR MODES ON THE TEN SELECTED SEQUENCES FROM THE TUM DATASET. X_{point}, X_{line} AND X_{plane} ARE THE NORMAL DISTRIBUTION MODELS THAT THE MATCHING NUMBERS OF POINT, LINE, AND PLANE FEATURES. "×" MEANS THAT TRACKING LOSS HAPPENED OR A SIGNIFICANT PORTION OF THE SEQUENCE IS NOT PROCESSED. THE η AND λ ARE 0.47 AND 0.53, RESPECTIVELY, WHEN COMPUTING O_P.

Sequences	Distribution	Time	P-VSLAM	PL-VSLAM	PP-VSLAM	PLP-VSLAM
fr1/room	$X_{point} \sim N(310.16, 78.81^2)$	Mean tracking time(s)	0.104	0.126	0.147	0.153
	$X_{line} \sim N(10.15, 4.81^2)$	RMSE	0.082	0.150	0.075	0.059
	$X_{plane} \sim N(3.97, 2.03^2)$	O_P	**0.211**	0.329	0.239	0.221
fr1/desk	$X_{point} \sim N(252.55, 75.36^2)$	Mean tracking time(s)	0.121	0.138	0.169	0.178
	$X_{line} \sim N(12.47, 2.26^2)$	RMSE	0.066	0.082	0.188	0.100
	$X_{plane} \sim N(2.91, 1.94^2)$	O_P	**0.174**	0.207	0.360	0.259
fr1/xyz	$X_{point} \sim N(280.78, 75.91^2)$	Mean tracking time(s)	0.113	0.121	0.151	0.156
	$X_{line} \sim N(14.30, 1.86^2)$	RMSE	0.008	0.014	0.011	0.010
	$X_{plane} \sim N(3.17, 1.96^2)$	O_P	**0.197**	0.278	0.267	0.259
fr3/nstr_ntex_far	$X_{point} \sim N(65.06, 44.85^2)$	Mean tracking time(s)	×	×	×	0.092
	$X_{line} \sim N(2.04, 1.14^2)$	RMSE	×	×	×	0.036
	$X_{plane} \sim N(0.90, 0.30^2)$	O_P	×	×	×	**1.000**
fr3/nstr_tex_far	$X_{point} \sim N(351.59, 111.41^2)$	Mean tracking time(s)	0.085	0.087	0.121	0.126
	$X_{line} \sim N(12.05, 2.79^2)$	RMSE	0.151	0.149	0.213	0.051
	$X_{plane} \sim N(0.87, 0.34^2)$	O_P	0.237	0.238	0.336	**0.189**
fr3/nstr_tex_near	$X_{point} \sim N(440.51, 52.16^2)$	Mean tracking time(s)	0.105	0.110	0.139	0.147
	$X_{line} \sim N(14.43, 5.24^2)$	RMSE	0.020	0.017	0.021	0.022
	$X_{plane} \sim N(0.86, 0.35^2)$	O_P	0.231	**0.216**	0.270	0.283
fr3/str_ntex_far	$X_{point} \sim N(180.49, 54.89^2)$	Mean tracking time(s)	0.052	0.069	0.089	0.100
	$X_{line} \sim N(5.64, 3.10^2)$	RMSE	0.028	0.021	0.016	0.015
	$X_{plane} \sim N(3.58, 0.83^2)$	O_P	0.264	0.244	**0.241**	0.251
fr3/str_ntex_near	$X_{point} \sim N(177.55, 76.08^2)$	Mean tracking time(s)	0.043	0.071	0.096	0.091
	$X_{line} \sim N(3.72, 1.62^2)$	RMSE	0.060	0.051	0.028	0.019
	$X_{plane} \sim N(2.92, 0.86^2)$	O_P	0.269	0.282	0.243	**0.206**
fr3/str_tex_far	$X_{point} \sim N(423.26, 64.07^2)$	Mean tracking time(s)	0.108	0.109	0.137	0.131
	$X_{line} \sim N(11.41, 4.09^2)$	RMSE	0.011	0.008	0.013	0.015
	$X_{plane} \sim N(3.28, 1.28^2)$	O_P	0.229	**0.196**	0.279	0.296
fr3/str_tex_near	$X_{point} \sim N(427.36, 57.93^2)$	Mean tracking time(s)	0.108	0.109	0.137	0.131
	$X_{line} \sim N(16.95, 3.88^2)$	RMSE	0.099	0.104	0.139	0.141
	$X_{plane} \sim N(2.53, 0.5^2)$	O_P	**0.213**	0.220	0.285	0.282



(a) The number of matched points

(b) The number of matched lines

(c) The number of matched planes

Fig. 4. The Gaussian fitting result of the number of matched features in the fr1/room sequence.

First of all, according to the mean tracking time and RMSE in the last four columns of Table I, the v_1 and v_2 for each of the four modes in each sequence are computed by using Equation (16). It should be added that the sequence named

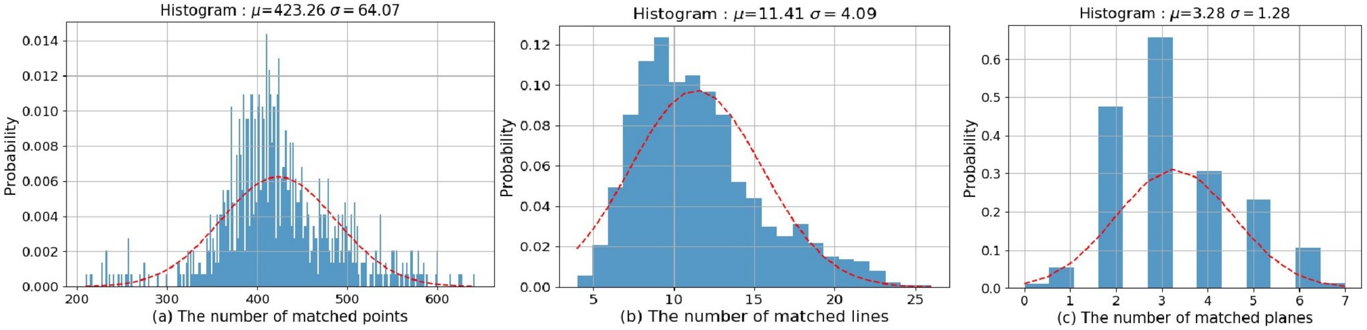


Fig. 5. The Gaussian fitting result of the number of matched features in the str_tex_far sequence.

fr3/nstr_ntex_far is not considered in the calculation process, because it only can work in the PLP-SLAM mode.

$$\begin{cases} v_1 = \frac{tm_i}{\sum_{i=1}^{4} tm_i} \\ v_2 = \frac{RMSE_i}{\sum_{i=1}^{4} RMSE_i} \end{cases} \tag{16}$$

Then, we calculated the v_1 and v_2 for the four modes of P-VSLAM, PL-VSLAM, PP-VSLAM, and PLP-VSLAM in the nine sequences, and the average ratio (ρ) based on v_1 and v_2 is Equation (17).

$$\rho = \frac{1}{36} \sum_{i=1}^{36} \frac{v_1}{v_2} \tag{17}$$

Having ρ, to balance the impact of the mean tracking time of each frame and RMSE on O_P, we set the relationships between η and λ as Equation (18):

$$\begin{cases} \eta = \rho \cdot \lambda \\ \eta + \lambda = 1 \end{cases} \tag{18}$$

Finally, we can obtain the η and λ, where η = 0.47 and λ = 0.53. Then, the O_P is computed (Table I).

2) Data Processing: For each sequence, after counting the number of matched point, line, and plane features are counted, we found that they follow the Gaussian distribution (Figure 4 and 5). The two figures shows the Gaussian fitting result of the number of matched features in the fr1/room and str_tex_far sequences, respectively.

3) Data Fusion: According to Table I, the four fusion modes (P-VSLAM, PL-VSLAM, PP-VSLAM, PLP-VSLAM) perform the best in multiple sequences. Therefore, we need to fuse the distributions of numbers of matched point-line-plane features corresponding to multiple sequences into a Gaussian distribution to represent the number of matched features threshold interval used by each mode. For example, on the basis of O_{P1}, P-VSLAM performs the best in the four sequences of fr1/room, fr1/desk, fr1/xyz, and fr3/str_tex_near. Therefore, we fused the distributions of numbers of matched point, line, and plane features in each of these four sequences to obtain an optimal distribution.

In order to combine multiple distributions into a single one, we aim to minimize the variance of the resulting distribution. For instance, suppose we have two Gaussian distributions, $X_1 \sim N(\mu_1, \sigma_1^2)$ and $X_2 \sim N(\mu_2, \sigma_2^2)$, and we want to fuse them into a new distribution $X \sim N(\mu, \sigma^2)$, To achieve this, we can follow the specific method outlined in Equation (19):

$$\begin{cases} X = a \cdot X_1 + b \cdot X_2 \\ a + b = 1 \end{cases} \tag{19}$$

Equation 19 further can be simplified as Equation (20), and then the variance of the fused distribution based on it becomes Equation (21):

$$X = X_1 + k \cdot (X_2 - X_1) \tag{20}$$

$$\begin{aligned} \sigma^2 &= var[X_1 + k \cdot (X_2 - X_1)] \\ &= var[(1-k) \cdot X_1 + k \cdot X_2] \\ &= (1-k)^2 \cdot var(X_1) + k^2 \cdot var(X_2) \\ &= (1-k)^2 \cdot \sigma_1^2 + k^2 \cdot \sigma_2^2 \end{aligned} \tag{21}$$

To minimize the variance of the fused distribution, we can take the derivative of Equation (21) with respect to k. This yields the following result (Equation 22):

$$\frac{\alpha \sigma^2}{\alpha k} = -2 \cdot (1-k) \cdot \sigma_1^2 + 2 \cdot k \cdot \sigma_2^2 \tag{22}$$

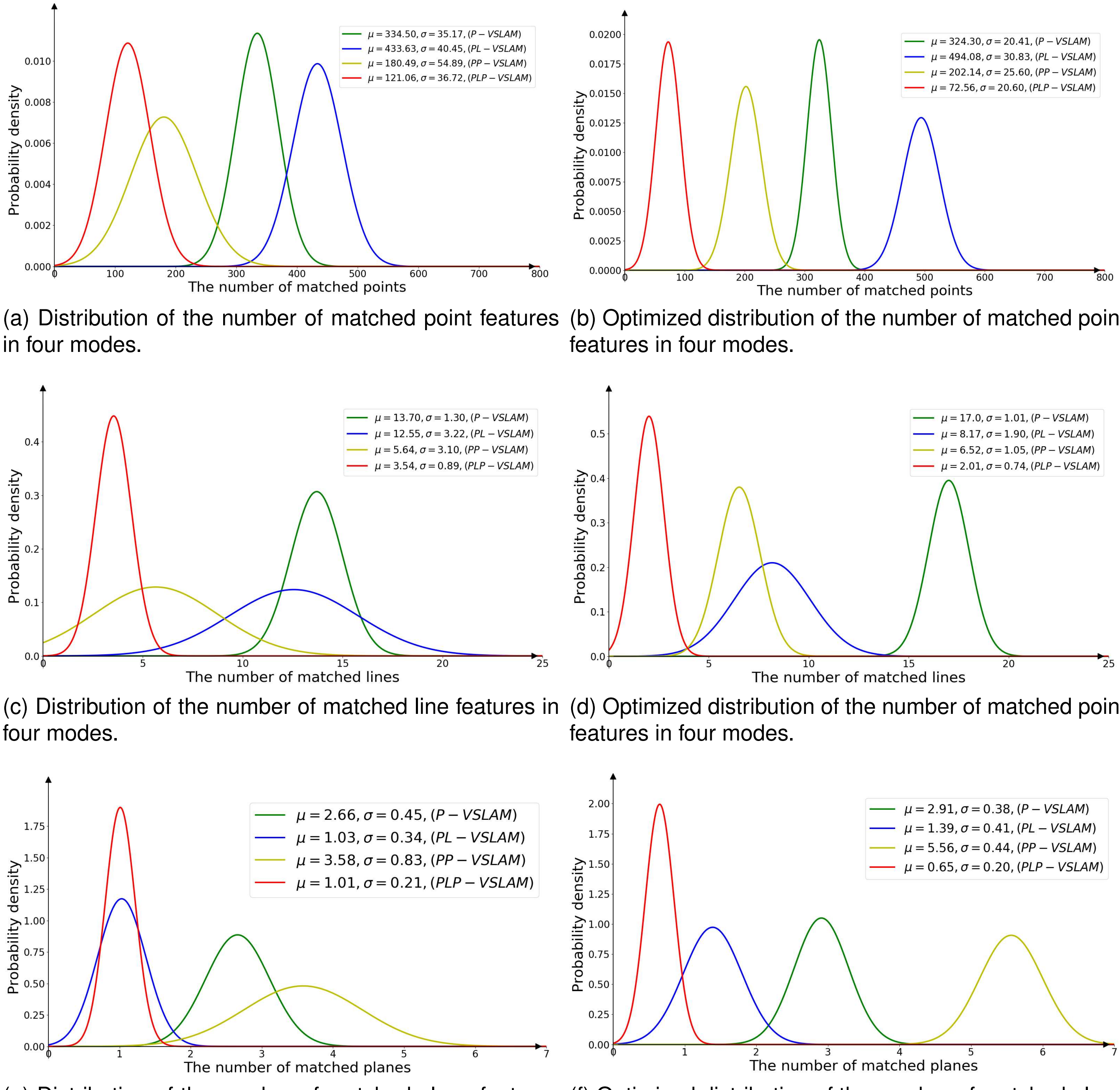
On the basis of the k (Equation (23)) that minimizes the variance, we can compute the final fused distribution. After fusing the distributions of the four modes, we finally get the feature matching distribution of them (Figure 6a, 6c, and 6e).

$$k = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \tag{23}$$

4) Optimization: By observing the Figure 6a, 6c, 6e, we noticed that there were many overlapping areas between the four Gaussian distributions. These overlaps pose a challenge in accurately determining the boundaries of the appropriate mode. To address this issue, we propose an optimized method that reduces these overlaps.

When we encounter two Gaussian distributions with overlaps, we firstly identify the corresponding interval ranges of them. For the overlapping parts of the interval, we assign them to the distribution with the maximum probability density to obtain the interval range corresponding to each distribution. After obtaining the interval range, we recalculate these two Gaussian distributions using the data corresponding to the interval range and repeat the above process until the results converge, thus separating the two relatively overlapping distributions better.

Furthermore, during the optimization process, we also limit the value of μ (e.g., $\mu_{PP-VSLAM}$ of the PP-VSLAM



(a) Distribution of the number of matched point features in four modes.

(b) Optimized distribution of the number of matched point features in four modes.

(c) Distribution of the number of matched line features in four modes.

(d) Optimized distribution of the number of matched point features in four modes.

(e) Distribution of the number of matched plane features in four modes.

(f) Optimized distribution of the number of matched plane features in four modes.

Fig. 6. Feature matching distribution of the four modes .

distribution in Figure 6a should meet the requirement: $\mu_{PLP-VSLAM} < \mu_{PP-VSLAM} < \mu_{P-VSLAM}$) to avoid distorting the results. The specific process is detailed in Algorithm 1. The optimized result is shown in Figure 6b, 6d, 6f.

After processing and optimizing the data, we summarized Figures 6b, 6d, and 6f to obtain the thresholds for each mode (Equation (24)). However, it should be noted that overlaps can occur between different modes. In such cases, we consider the μ of the Gaussian distributions corresponding to the number of matched point, line, and plane features for different modes, and choose the mode that has a closer μ to the overlapping region as the best choice.

$$PLPF-VSLAM = \begin{cases} P, \quad n_p\epsilon[270,390], n_l\epsilon[0,21], n_\pi\epsilon[0,+\infty] \\ \quad \cup\, n_p\epsilon[390,+\infty], n_l\epsilon[0,8], n_\pi\epsilon[0,+\infty] \\ PL, \quad n_p\epsilon[270,390], n_l\epsilon[21,+\infty], n_\pi\epsilon[0,+\infty] \\ \quad \cup\, n_p\epsilon[390,+\infty], n_l\epsilon[8,+\infty], n_\pi\epsilon[0,+\infty] \\ PP, \quad n_p\epsilon[130,270], n_l\epsilon[0,8], n_\pi\epsilon[0,+\infty] \\ \quad \cup\, n_p\epsilon[130,270], n_l\epsilon[8,+\infty], n_\pi\epsilon[2,+\infty] \\ PLP, \quad n_p\epsilon[0,130], n_l\epsilon[0,+\infty], n_\pi\epsilon[0,+\infty] \\ \quad \cup\, n_p\epsilon[130,270], n_l\epsilon[8,+\infty], n_\pi\epsilon[0,2] \end{cases} \tag{24}$$

Algorithm 1: Optimization

```
Input: $X_1 \sim N(\mu_1, \sigma_1^2)$, $X_2 \sim N(\mu_2, \sigma_2^2)$
Output: $X_1 \sim N(\mu_1, \sigma_1^2)$, $X_2 \sim N(\mu_2, \sigma_2^2)$
$th_\mu$: The threshold condition that $\mu$ should satisfy during the loop.
$th_\sigma$: The threshold condition that $\sigma$ should satisfy during the loop.
$\mu_{pre}$: $\mu$ of the previous Gaussian distribution.
$\mu_{next}$: $\mu$ of the next Gaussian distribution.
while True do
    /* N is the optimized interval range. */
    for $i \leftarrow 0$ to $|N|$ do
        /* k is the step size. */
        $p_1 \leftarrow \frac{1}{\sqrt{2\pi}\sigma_1} \cdot e^{-\frac{(i \cdot k - \mu_1)^2}{2\sigma_1^2}}$
        $p_2 \leftarrow \frac{1}{\sqrt{2\pi}\sigma_2} \cdot e^{-\frac{(i \cdot k - \mu_2)^2}{2\sigma_2^2}}$
        if $p_1 > p_2$ then
            $\mu_{1n} \leftarrow \mu_{1n} + p_1 \cdot i \cdot k$
            $M_1 \leftarrow M_1 + p_1$
        else
            $\mu_{2n} \leftarrow \mu_{2n} + p_2 \cdot i \cdot k$
            $M_2 \leftarrow M_2 + p_2$
    $\mu_1 \leftarrow \frac{\mu_{1n}}{M_1}$, $\mu_2 \leftarrow \frac{\mu_{2n}}{M_2}$
    for $i \leftarrow 0$ to $|N|$ do
        if $p_1 > p_2$ then
            $\sigma_{1n}^2 \leftarrow \sigma_{1n}^2 + p_1 \cdot (i \cdot k - \mu_1)^2$
        else
            $\sigma_{2n}^2 \leftarrow \sigma_{2n}^2 + p_2 \cdot (i \cdot k - \mu_2)^2$
    $\sigma_1^2 = \frac{\sigma_{1n}^2}{M_1}$, $\sigma_2^2 = \frac{\sigma_{2n}^2}{M_2}$
    if $|\mu_1 - \mu_{1pre}| < th_\mu$ and $|\sigma_1 - \sigma_{1pre}| < th_\sigma$ and $|\mu_2 - \mu_{2pre}| < th_\mu$ and $|\sigma_2 - \sigma_{2pre}| < th_\sigma$ then
        break
    if $\mu_1 < \mu_{pre}$ or $\mu_2 > \mu_{next}$ then
        break
    $\mu_{1pre} \leftarrow \mu_1, \sigma_{1pre} \leftarrow \sigma_1$
    $\mu_{2pre} \leftarrow \mu_2, \sigma_{2pre} \leftarrow \sigma_2$
```

where P, PL, PP, and PLP represent P-VSLAM, PL-VSLAM, PP-VSLAM, and PLP-VSLAM, respectively. The n_p, n_l, and n_π mean the numbers of matched point, line, and plane features, respectively. The two weights are assigned as $\eta = 0.47$ and $\lambda = 0.53$.

Based on the μ of the distributions corresponding to the number of matched point, line, and plane features, the applicable conditions of the four modes are shown in Figure 7. In the figures, the four modes, P-VSLAM, PL-VSLAM, PP-VSLAM, and PLP-VSLAM are colored green, blue, yellow, and red, respectively. Figure 7(a) is the overview, while (b) and (c) are side-views. The Figure 7(a) can be populated as Figure 7(d).

B. TUM Dataset

In this experiment, we conducted the experiments by using the results shown in Equation (24). With this PLPF-VSLAM, we then evaluated the performance of the six VSLAM on the ten sequences from the TUM dataset (Table II). In order to more intuitively show the performance of different VSLAM systems, we make Figure 8 based on the Table II. As for the cases where tracking loss or data is missing, the maximum RMSE in the sequence is used.

The trajectories and reconstruction of maps in PLPF-VSLAM are visually depicted in Figure 9 and 10 (here, only the most representative scenes are displayed). This figure provides a comprehensive visualization of the paths followed by the camera and the resulting map reconstruction. In sequences with rich texture features (fr1/xyz, fr1/desk, fr3/nstr_tex_near, fr3/str_tex_far, fr3/str_tex_near), the PLPF-VSLAM automatically selects point or point-line features for tracking and mapping. In the sequences with sparse features (fr1/room, fr3/nstr_ntex_far, fr3/nstr_tex_far, fr3/str_ntex_far, fr3/str_ntex_near), it adds plane features to the tracking and mapping to ensure the accuracy. In terms of mapping results, after adding plane features, the maps have a better structural characteristics of the scene, and clearer outlines.

In addition, considering that real-time performance is also an important indicator of a VSLAM, we compared our system to ORB-SLAM2 and Planar-SLAM on five sequences from the total time and mean tracking time (Table III and Figure 11). The processing times show that ORB-SLAM2 has the best performance in scenarios with rich features (fr1/room, fr3/str_tex_far, fr3/nstr_tex_near), which is much faster than the systems based on the fusion of point-line-plane. Compared with the Planar-SLAM system that also uses point-line-plane feature fusion, PLPF-VSLAM has a better performance on all sequences. It can be explained by that not all the scenarios are non-/low-textured, thus our system adaptively selects the fusion of point, line, and plane features, which makes the processing time shorter. However, compared to ORB-SLAM2, our method is slower. On the one hand, adaptive fusion is based on the number of matched features. The extraction and matching of line and plane features increases the processing time. On the other hand, on a sequence, our system not only uses point features but also line and plane features in some places for mapping and tracking, which also increases the processing time.

To sum up, PLPF-VSLAM demonstrates its versatility in handling both rich-textured and non-/low-textured indoor scenarios. As for the processing time, it is slightly longer than ORB-SLAM2, but is faster than Planar-SLAM which is also based on the fusion of point, line, and plane. The reason why our system takes longer time is that it aims to deal with non-/low-textured indoor scenarios. Thus, the threshold of feature extraction is not as strict as that of ORB-SLAM2, which is time-consuming when more features are involved in the computation. Moreover, adding the additional line and plane features to the map also takes more time. Therefore, overall, the PLPF-VSLAM system is the best when taking all aspects into account.

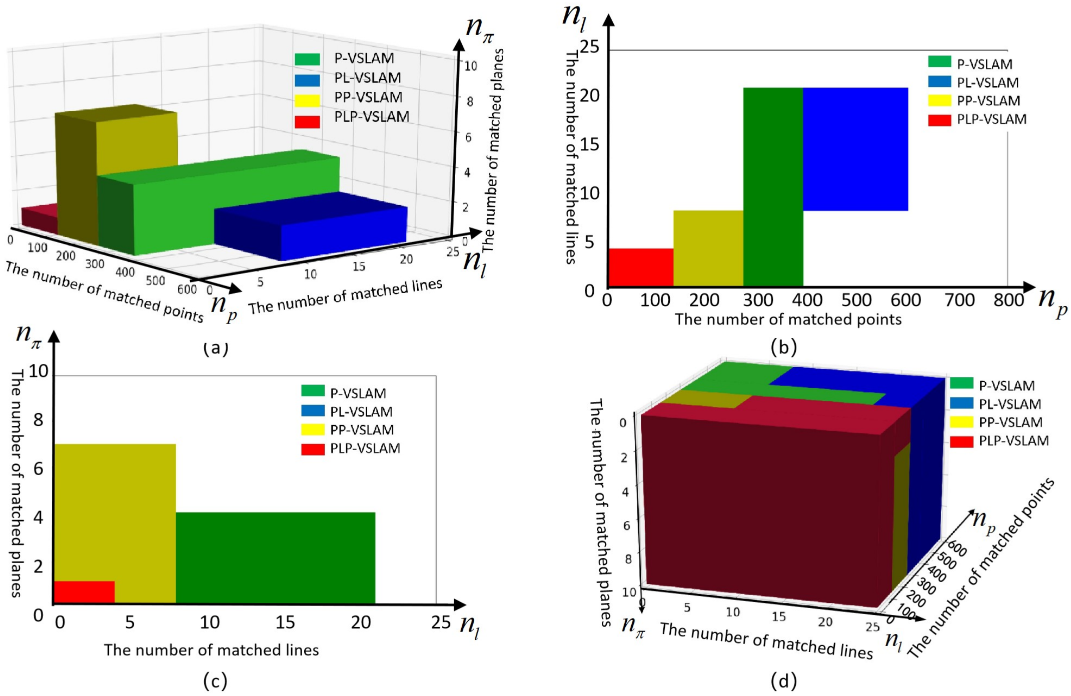


Fig. 7. The applicable conditions of four modes, where $\eta = 0.47$ and $\lambda = 0.53$. P-VSLAM, PL-VSLAM, PP-VSLAM, and PLP-VSLAM are colored green, blue, yellow, and red, respectively. Figure (a) is the overview, while (b) and (c) are side-views. Figure (d) is the expansion of Figure (a).

TABLE II
RMSE OF DIFFERENT VSLAM ON THE TUM DATASET (UNIT: M). "×" MEANS THAT THE TRACKING IS LOST AT SOME POINT OR A SIGNIFICANT PORTION OF THE SEQUENCE IS NOT PROCESSED; "-" INDICATES THAT THE DATA IS NOT PROVIDED IN LITERATURE.

Sequence	PLPF-VSLAM (Ours)	ORB-SLAM2	PL-SLAM	LSD-SLAM	L-SLAM	Planar-SLAM
fr1/xyz	0.011	**0.009**	0.012	0.090	-	×
fr1/desk	0.073	**0.020**	-	0.107	-	×
fr1/room	**0.019**	0.052	-	-	-	×
fr3/nstr_ntex_far	**0.020**	×	-	-	-	0.027
fr3/nstr_tex_far	**0.076**	0.098	×	0.183	-	×
fr3/nstr_tex_near	0.022	**0.014**	0.021	0.075	0.066	0.029
fr3/str_ntex_far	**0.009**	×	-	-	0.141	0.017
fr3/str_ntex_near	**0.019**	×	-	-	0.066	0.030
fr3/str_tex_far	0.015	0.010	**0.009**	0.079	0.212	0.047
fr3/str_tex_near	**0.011**	0.016	0.013	-	0.156	0.062

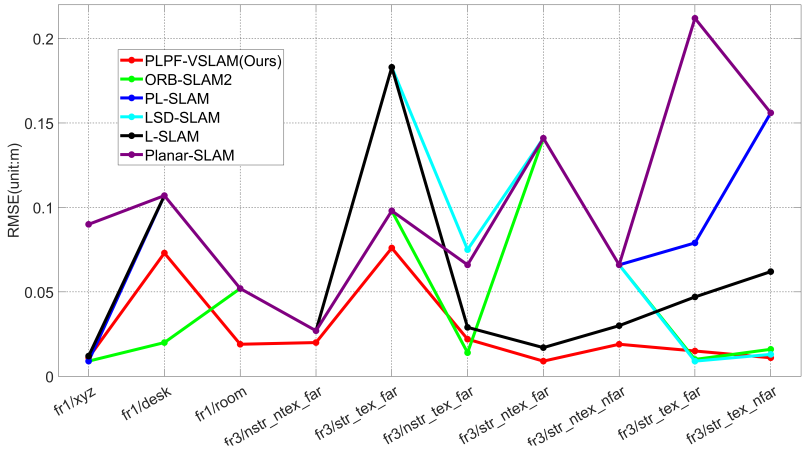
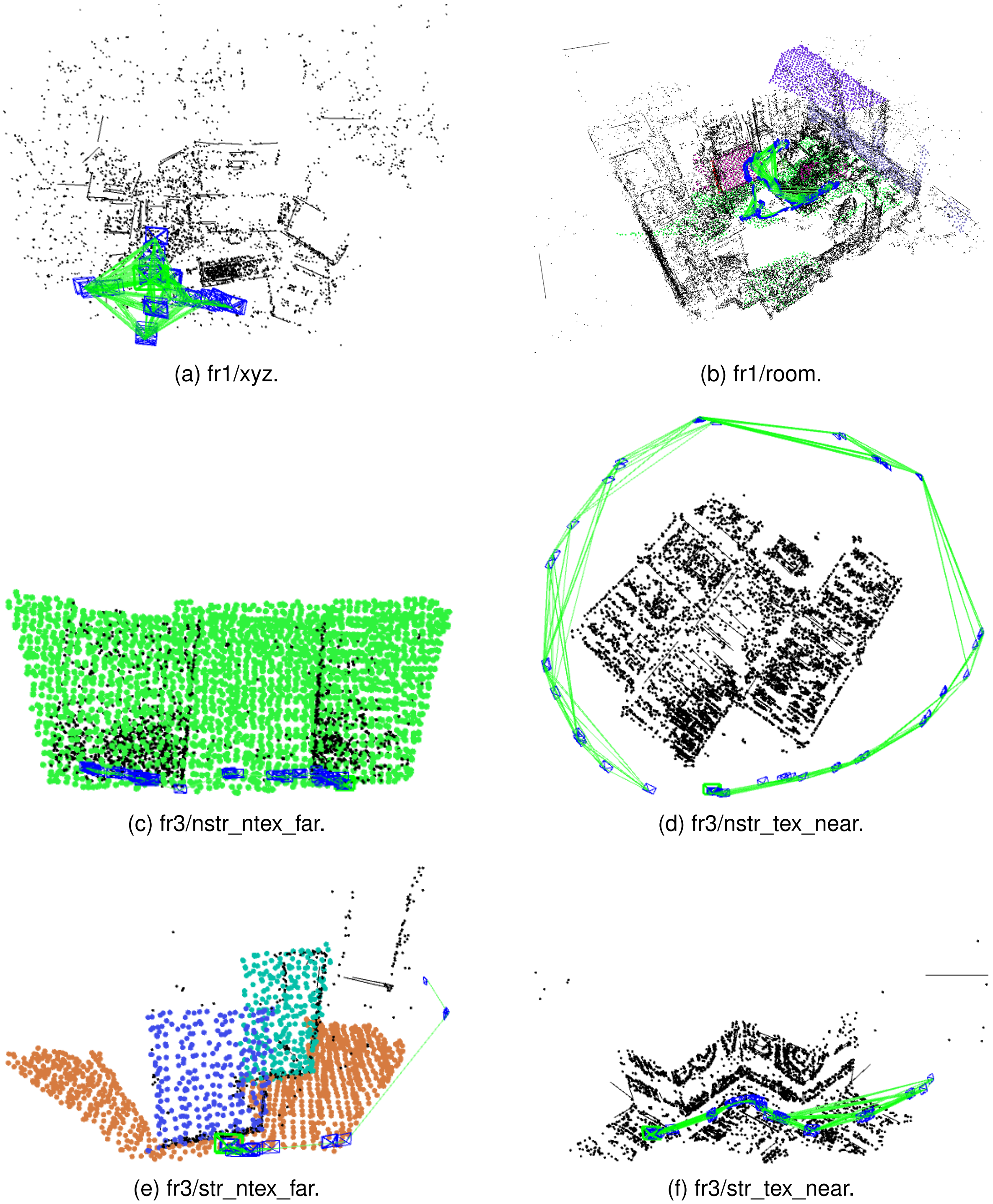


Fig. 8. Comparison of the RMSE of different VSLAM on the TUM dataset.



(a) fr1/xyz.

(b) fr1/room.

(c) fr3/nstr_ntex_far.

(d) fr3/nstr_tex_near.

(e) fr3/str_ntex_far.

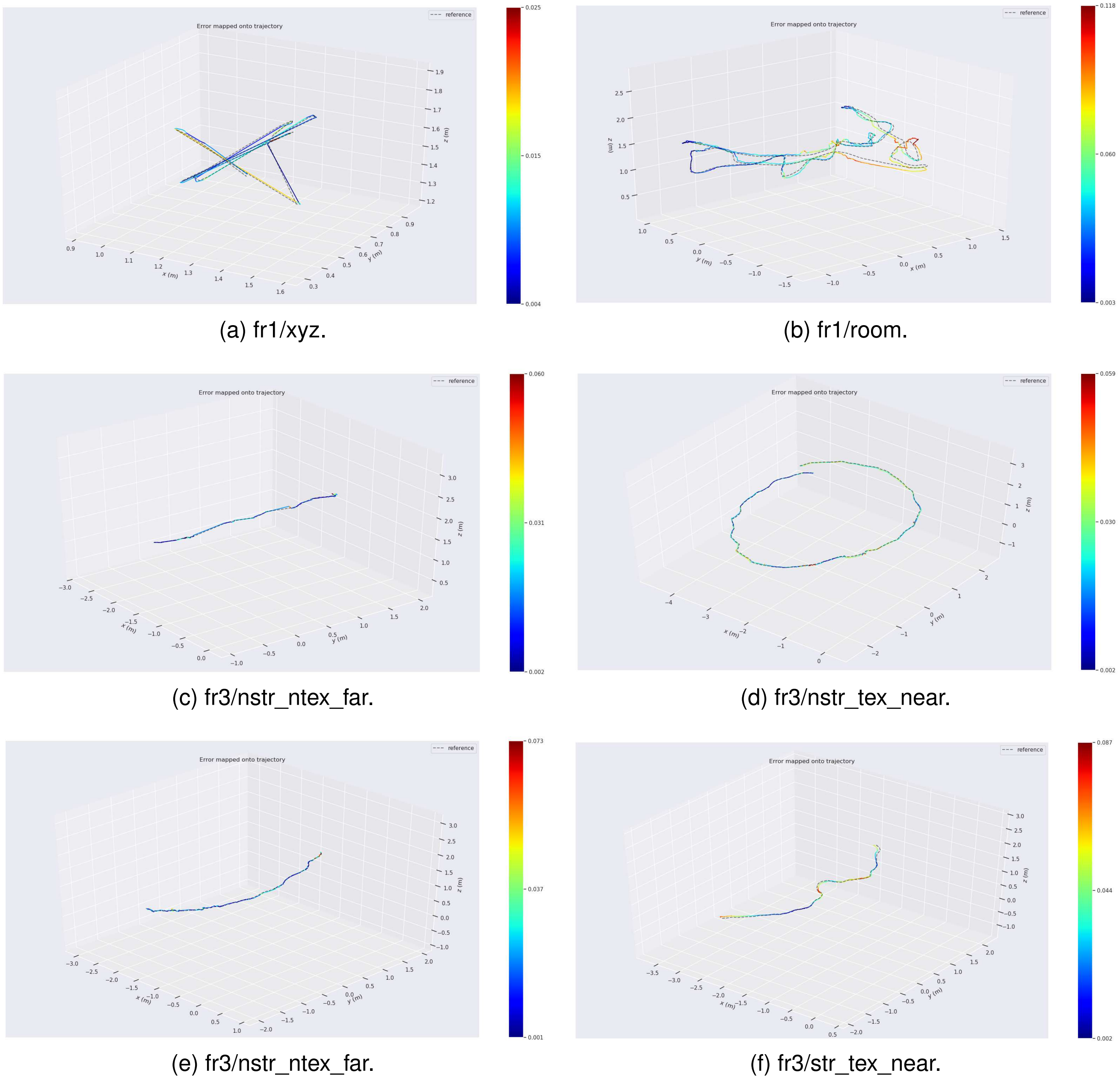
(f) fr3/str_tex_near.

Fig. 9. Reconstructed maps of the six most representative sequences from the TUM dataset.

C. ICL_NUIM Dataset

The sequences from living room and office in ICL_NUIM dataset are also used to evaluate the accuracy of the PLPF-VSLAM. Base on RMSE of ATE, we compare our system with ORB-SLAM2, L-SLAM, and Planar-SLAM. The performances of different systems are shown in Table IV and Figure 12. As no non-/low-textured scenarios are involved, four systems stably finish localization and mapping.

In the ICL_NUIM dataset, our system, along with L-SLAM and Planar-SLAM, achieves favorable results. This can be attributed to the abundance of structural features present in the dataset, which offer ample line and plane features. These additional features serve to provide valuable constraints for pose estimation, thereby enhancing the overall accuracy of the system. Therefore, in such scenarios, the VSLAM based on point, line and plane have a better performance than ORB-SLAM2 based on point features.



(a) fr1/xyz.

(b) fr1/room.

(c) fr3/nstr_ntex_far.

(d) fr3/nstr_tex_near.

(e) fr3/nstr_ntex_far.

(f) fr3/str_tex_near.

Fig. 10. Trajectories of the six most representative sequences from the TUM dataset

TABLE III
THE PROCESSING TIME OF EACH VSLAM SYSTEM ON THE TUM DATASET (UNIT: S). ”×” MEANS THAT THE TRACKING IS LOST AT SOME POINT OR A SIGNIFICANT PORTION OF THE SEQUENCE IS NOT PROCESSED.

Sequences	Time	PLPF-VSLAM (Ours)	ORB-SLAM2	Planar-SLAM
fr1/room	Total time(s)	195.654	**70.727**	×
	Mean tracking time(s)	0.113	**0.035**	×
fr3/nstr_ntex_far	Total time(s)	**55.993**	×	75.459
	Mean tracking time(s)	**0.092**	×	0.118
fr3/str_ntex_far	Total time(s)	**83.032**	×	91.847
	Mean tracking time(s)	**0.072**	×	0.098
fr3/str_tex_far	Total time(s)	186.457	**42.509**	211.259
	Mean tracking time(s)	0.109	**0.027**	0.142
fr3/nstr_tex_near	Total time(s)	298.821	**162.148**	369.843
	Mean tracking time(s)	0.115	**0.033**	0.152

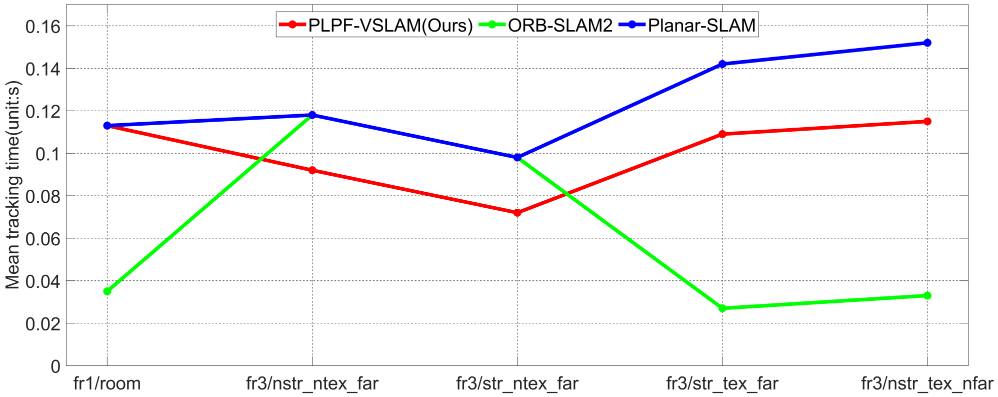


Fig. 11. Comparison of mean tracking time of different VSLAM on the TUM dataset.

TABLE IV
RMSE OF DIFFERENT VSLAM ON ICL_NUIM DATASET (UNIT: M).

Sequence	PLPF-VSLAM (Ours)	ORB-SLAM2	L-SLAM	Planar-SLAM
lr-kt0	0.007	0.009	0.012	**0.006**
lr-kt1	**0.013**	0.201	0.027	0.015
lr-kt2	**0.017**	0.033	0.053	0.020
lr-kt3	0.058	0.017	0.143	**0.012**
of-kt0	0.046	0.073	**0.020**	0.041
of-kt1	**0.012**	0.085	0.015	0.020
of-kt2	0.026	0.023	0.026	**0.011**
of-kt3	0.032	0.034	**0.011**	0.014

!h

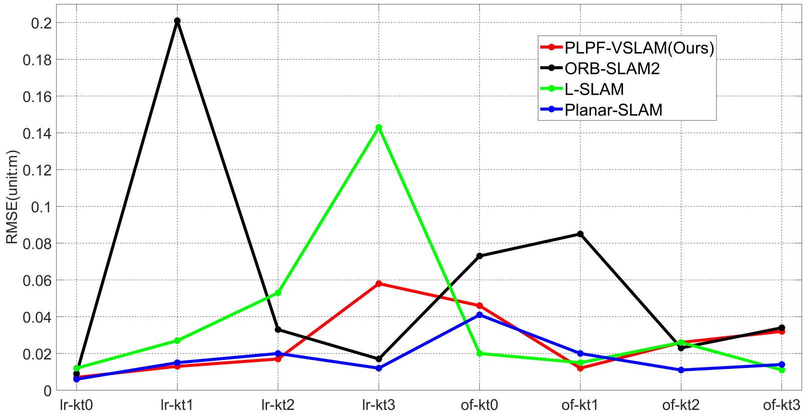


Fig. 12. Comparison of RMSE of different VSLAM on ICL_NUIM dataset.

VII. Discussion

The experimental results of PLPF-VSLAM demonstrate its successful performance across various scenarios, yielding satisfactory outcomes. Particularly noteworthy is its robustness in non-textured or low-textured scenes. However, it is important to address four specific aspects in further discussion.

First of all, for the time being, the PLPF-VSLAM is more suitable for indoor scenarios, because it takes the RGB-D camera as the sensor for data collection. The RGB-D camera is not suitable for the outdoors.

Secondly, in this experiment, the weights for O_P were set to $\eta = 0.47$ and $\lambda = 0.53$, based on our experience. However, it is important to emphasize that these weights serve as an example and are adjustable. Researchers can further determine the appropriate weights based on their specific requirements, as long as the condition $\eta + \lambda = 1$ is maintained. This allows for customization and adaptation of the weighting scheme according to the particular needs and characteristics of the SLAM system being developed.

Thirdly, in the experimentation process with PLPF-VSLAM, the main parameters (n_p, n_l, and n_π) were determined based on the TUM Dataset. However, it is important to acknowledge that the performance and accuracy of the system can be further improved by leveraging larger and more diverse datasets. Employing a larger dataset facilitates a better understanding of the distribution of feature matching numbers and aids in obtaining more precise parameters. Furthermore, a larger dataset offers a wider range of scenarios and variations, thereby enhancing the robustness and generalization capabilities of the system. Therefore, it is highly recommended to consider utilizing a comprehensive dataset when determining parameters, as it enables the achievement of more accurate and reliable results.

Last but not least, the biggest advantage of PLPF-VSLAM is that it can be applied to scenes with a different richness of texture, especially the low- even non-textured scenarios. However, it is important to acknowledge that PLPF-VSLAM may exhibit slower processing times compared to ORB-SLAM2. This can be attributed to the additional time-consuming processes involved, such as feature extraction and matching for point, line, and plane features. To further enhance the performance of PLPF-VSLAM, efforts can be directed towards optimizing and improving these aspects.

VIII. Conclusion and Future Work

This paper presents a VSLAM named PLPF-VSLAM, which leverages the fusion of point-line-plane features. This system is applicable to all scenes regardless of texture richness, including the low- and non-textured scenarios. A key point of PLPF-VSLAM is its adaptive selection of tracking and mapping modes, enabling transformation into P-VSLAM, PL-VSLAM, PP-VSLAM, or PLP-VSLAM based on the texture richness of the scene. PLPF-VSLAM is evaluated on the TUM and ICL_NUIM datasets. In comparison to other similar VSLAM systems, it has an overall better performance in terms of both accuracy and processing speed, particularly in non-/low-textured scenarios. In particular, when compared to ORB-SLAM2, it achieves an accuracy improvement of approximately 11.29%. In terms of processing speed, it outperforms PL(P)-VSLAM by approximately 21.57%. Therefore, we conclude that PLPF-VSLAM is a good attempt of VSLAM, which is able to provide an effective idea for the subsequent solution of VSLAM in indoor scenarios.

Future work will concentrate on further elaboration and testing of the current work from three aspects: (i) Extend this system to outdoor. Specifically, the plan is to use stereo cameras to make the system not only can deal with indoor scenarios, but also outdoors; (ii) Improve the accuracy of PLPF-VSLAM by including more diverse set of sequences (data sets) during the parameter determination stage; (iii) Accelerate the processing speed by optimizing the calculation process of parameters during mode selection.

Acknowledgments

This work was supported by the Shandong Province Natural Science Foundation Youth Branch [No. ZR2022QD121], the Fundamental Research Funds for the Central Universities [No. 3072022FSC0401].

References

[1] B. Huang, J. Zhao, and J. Liu, “A Survey of Simultaneous Localization and Mapping,” *arXiv preprint arXiv:1909.05214*, 2019.

[2] A. Singandhupe and H. M. La, “A Review of SLAM Techniques and Security in Autonomous Driving,” in *2019 Third IEEE International Conference on Robotic Computing (IRC).* IEEE, 2019, pp. 602–607.

[3] A. R. Khairuddin, M. S. Talib, and H. Haron, “Review on Simultaneous Localization and Mapping (SLAM),” in *2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE).* IEEE, 2015, pp. 85–90.

[4] K. Di, W. Wan, H. Zhao, Z. Liu, R. Wang, and F. Zhang, “Progress and Applications of Visual SLAM,” *Journal of Geodesy and Geoinformation Science*, vol. 2, no. 2, p. 38, 2019.

[5] S. Wen, X. Lv, H. K. Lam, S. Fan, X. Yuan, and M. Chen, “Probability Dueling DQN Active Visual SLAM for Autonomous Navigation in Indoor Environment,” *Industrial Robot: the International Journal of Robotics Research and Application*, 2021.

[6] X. Jiang, L. Zhu, J. Liu, and A. Song, “A SLAM-based 6DoF Controller with Smooth Auto-calibration for Virtual Reality,” *The Visual Computer*, pp. 1–14, 2022.

[7] M. Juan, M. Mendez-Lopez, C. Fidalgo, R. Molla, R. Vivo, D. Paramo *et al.*, “A SLAM-based Augmented Reality APP for the Assessment of Spatial Short-term Memory Using Visual and Auditory Stimuli,” *Journal on Multimodal User Interfaces*, vol. 16, no. 3, pp. 319–333, 2022.

[8] J. Guo, R. Ni, and Y. Zhao, “DeblurSLAM: A Novel Visual SLAM System Robust in Blurring Scene,” in *2021 IEEE 7th International Conference on Virtual Reality (ICVR).* IEEE, 2021, pp. 62–68.

[9] K. A. Tsintotas, L. Bampis, and A. Gasteratos, “The Revisiting Problem in Simultaneous Localization and Mapping: A Survey on Visual Loop Closure Detection,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 11, pp. 19 929–19 953, 2022.

[10] S. Arshad and G.-W. Kim, “Role of Deep Learning in Loop Closure Detection for Visual and Lidar Slam: A Survey,” *Sensors*, vol. 21, no. 4, p. 1243, 2021.

[11] L. Lu, J. Hou, S. Yuan, X. Yao, Y. Li, and J. Zhu, “Deep Learning-assisted Real-time Defect Detection and Closed-loop Adjustment for Additive Manufacturing of Continuous Fiber-reinforced Polymer Composites,” *Robotics and Computer-Integrated Manufacturing*, vol. 79, p. 102431, 2023.

[12] S. An, H. Zhu, D. Wei, K. A. Tsintotas, and A. Gasteratos, “Fast and Incremental Loop Closure Detection with Deep Features and Proximity Graphs,” *Journal of Field Robotics*, vol. 39, no. 4, pp. 473–493, 2022.

[13] H. Li, Z. Li, and X. Chen, “PLP-SLAM: Visual SLAM Method based on Point, Line, and Plane Feature Fusion,” *Robot*, vol. 39, no. 2, pp. 214–220, 2017.

[14] D. G. Lowe, “Distinctive Image Features from Scale-invariant Keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[15] H. Bay, T. Tuytelaars, and L. V. Gool, “SURF: Speeded up Robust Features,” in *European Conference on Computer Vision.* Springer, 2006, pp. 404–417.

[16] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An Efficient Alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 2564–2571.

[17] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," in *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE, 2007, pp. 225–234.

[18] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[19] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast Semi-direct Monocular Visual Odometry," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 15–22.

[20] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[21] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[22] P. Smith, I. Reid, and A. J. Davison, "Real-time Monocular SLAM with Straight Lines," in *BMVC*, vol. 6, 2006, pp. 17–26.

[23] T. Lemaire and S. Lacroix, "Monocular-vision based SLAM using Line Segments," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 2791–2796.

[24] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh, "Building a 3D Line-based Map using Stereo SLAM," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1364–1377, 2015.

[25] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale Direct Monocular SLAM," in *European Conference on Computer Vision*. Springer, 2014, pp. 834–849.

[26] G. Yang, Q. Wang, P. Liu, and H. Zhang, "An Improved Monocular PL-SlAM Method with Point-line Feature Fusion under Low-texture Environment," in *2021 4th International Conference on Control and Computer Vision*, 2021, pp. 119–125.

[27] 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.

[28] B. Fang and Z. Zhan, "A Visual SLAM Method based on Point-line Fusion in Weak-matching Scene," *International Journal of Advanced Robotic Systems*, vol. 17, no. 2, p. 1729881420904193, 2020.

[29] Z. Zhao, T. Song, B. Xing, Y. Lei, and Z. Wang, "PLI-VINS: Visual-inertial SLAM based on Point-line Feature Fusion in Indoor Environment," *Sensors*, vol. 22, no. 14, p. 5457, 2022.

[30] Y. Li, N. Brasch, Y. Wang, N. Navab, and F. Tombari, "Structure-SLAM: Low-drift Monocular SLAM in Indoor Environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6583–6590, 2020.

[31] Y. Zhou, L. Kneip, C. Rodriguez, and H. Li, "Divide and Conquer: Efficient Density-based Tracking of 3D Sensors in Manhattan Worlds," in *Asian Conference on Computer Vision*. Springer, 2016, pp. 3–19.

[32] R. Guo, K. Peng, W. Fan, Y. Zhai, and Y. Liu, "RGB-D SLAM Using Point-plane Constraints for Indoor Environments," *Sensors*, vol. 19, no. 12, p. 2721, 2019.

[33] X. Zhang, W. Wang, X. Qi, Z. Liao, and R. Wei, "Point-plane SLAM using Supposed Planes for Indoor Environments," *Sensors*, vol. 19, no. 17, p. 3795, 2019.

[34] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng, "Point-plane SLAM for Hand-held 3D Sensors," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 5182–5189.

[35] L. Xia, J. Cui, X. Li, D. Zhang, J. Zhang, and L. Yi, "A Point-line-plane Primitives Fused Localization and Object-oriented Semantic Mapping in Structural Indoor Scenes," *Measurement Science and Technology*, vol. 33, no. 9, p. 095017, 2022.

[36] Y. Li, R. Yunus, N. Brasch, N. Navab, and F. Tombari, "RGB-D SLAM with Structural Regularities," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 581–11 587.

[37] H. Li, Z. Hu, and X. Chen, "PLP-SLAM: a Visual SLAM Method based on Point-line-plane Feature Fusion," *Jiqiren*, vol. 39, no. 2, pp. 214–220, 2017.

[38] F. Shu, J. Wang, A. Pagani, and D. Stricker, "Structure PLP-SLAM: Efficient Sparse Mapping and Localization using Point, Line and Plane for Monocular, RGB-D and Stereo Cameras," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2105–2112.

[39] H. Yang, J. Yuan, Y. Gao, X. Sun, and X. Zhang, "UPLP-SLAM: Unified Point-line-plane Feature Fusion for RGB-D Visual SLAM," *Information Fusion*, vol. 96, pp. 51–65, 2023.

[40] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: A Fast Line Segment Detector with a False Detection Control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, 2008.

[41] L. Zhang and R. Koch, "An Efficient and Robust Line Segment Matching Approach based on LBD Descriptor and Pairwise Geometric Consistency," *Journal of Visual Communication and Image Representation*, vol. 24, no. 7, pp. 794–805, 2013.

[42] C. Feng, Y. Taguchi, and V. R. Kamat, "Fast Plane Extraction in Organized Point Clouds using Agglomerative Hierarchical Clustering," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6218–6225.

[43] D. Gálvez-López and J. D. Tardos, "Bags of Binary Words for Fast Place Recognition in Image Sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[44] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and

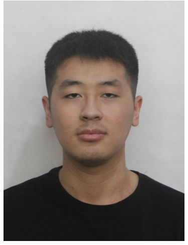
D. Cremers, “A Benchmark for the Evaluation of RGB-D SLAM Systems,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 573–580.

[45] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, “A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1524–1531.

[46] P. Kim, B. Coltin, and H. J. Kim, “Linear RGB-D SLAM for Planar Environments,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 333–348.

Jinjin Yan is an associate professor at the Qingdao Innovation and Development Center, Harbin Engineering University. He obtained his Ph.D. degree from the University of New South Wales, Sydney, Australia, in 2020. His started his Ph.D. at TU Delft in 2016 and later transferred to UNSW in 2018. His research interests include space modeling for navigation, navigation path planning, indoor and outdoor seamless navigation, Simultaneous Localization and Mapping (SLAM), Mobility as a Service (MaaS), cooperative path planning and task assignment for unmanned system swarms, and full-coverage path planning.

Youbing Zheng is currently a postgraduate student pursuing a Master’s degree at Qingdao Innovation and Development Center, Harbin Engineering University. His major is Control Science and Engineering. In 2021, he received Bachelor’s degree in Measuring & Control Technology and Instrumentation from Changsha University of Science & Technology, Changsha, China. His research interests include space modeling, visual-based SLAM, positioning and navigation of autonomous systems.



Jinquan Yang is currently a postgraduate student pursuing a master’s degree at Qingdao Innovation and Development Center, Harbin Engineering University. His major is Control Science and Engineering. He received the Bachelor’s degree in Measuring & Control Technology and Instrumentation from Qilu University of Technology, jinan, China, in 2021. His research interests include indoor positioning and navigation, optimal estimation, path planning, machine learning.

LYUDMILA MIHAYLOVA (M’98-SM’2008) is Professor of Signal Processing and Control at the Department of Automatic Control and Systems Engineering at the University of Sheffield, United Kingdom. She received MEng degree in Systems and Control Engineering, M.Sc. degree in Applied Mathematics and Informatics and her Ph.D. degree is in Systems and Control Engineering, all from the Technical University of Sofia, Bulgaria. Her research is in the areas of autonomous systems and machine learning with various applications such as navigation, surveillance, and sensor networks. She is Associate Editor-in-Chief for the IEEE Transactions on Aerospace and Electronic Systems since 2021 and a Subject Area Editor for the Elsevier Signal Processing Journal since 2022. She was the president of the International Society of Information Fusion (ISIF) from 2016 to 2018. She is on the Board of Directors of ISIF. She has been serving for organizing conferences – as the general vice chair of UKCI’2022, a program chair for the International Conference on Information Fusion, Fusion 2022, technical chair for Fusion 2021, publicity chair for IEEE MFI’ 2021 and UKCI 2021. She was the general vice-chair for the International Conference on Information Fusion 2018 (Cambridge, UK), of the IET Data Fusion & Target Tracking 2014 and 2012 Conferences, publications chair for ICASSP 2019 (Brighton, UK) and others.

Weijie Yuan Weijie Yuan (Member, IEEE) received the B.E. degree from the Beijing Institute of Technology, China, in 2013, and the Ph.D. degree from the University of Technology Sydney, Australia, in 2019. In 2016, he was a Visiting Ph.D. Student with the Institute of Telecommunications, Vienna University of Technology, Austria. He was a Research Assistant with the University of Sydney, a Visiting Associate Fellow with the University of Wollongong, and a Visiting Fellow with the University of Southampton, from 2017 to 2019. From 2019 to 2021, he was a Research Associate with the University of New South Wales. He is currently an Assistant Professor with the Department of Electrical and Electronic Engineering, Southern University of Science and Technology, Shenzhen, China. He currently serves as an Associate Editor for the IEEE Communications Letters, an Associate Editor as well as an Award Committee Member for the EURASIP Journal on Advances in Signal Processing. He has led the guest editorial teams for three special issues in IEEE Communications Magazine, IEEE Transactions on Green Communications and Networking, and China Communications. He was an Organizer/the Chair of several workshops, special sessions, and tutorials on orthogonal time frequency space (OTFS) and integrated sensing and communication (ISAC) in flagship IEEE and ACM conferences, including IEEE ICC, IEEE/CIC ICCC, IEEE SPAWC, IEEE VTC, IEEE WCNC, IEEE ICASSP, and ACM MobiCom. He is the Founding Chair of the IEEE ComSoc Special Interest Group on Orthogonal Time Frequency Space (OTFS-SIG). He was listed in the World’s Top 2Stanford University for citation impact in 2022. He was a recipient of the Best Ph.D. Thesis Award from the Chinese Institute of Electronics, an Exemplary Reviewer Award from IEEE Transactions on Communications and IEEE Wireless Communications Letters, and a Best Editor Award from IEEE Communications Letters.

Fuqiang Gu is currently a Professor in the College of Computer Science at the Chongqing University. He obtained his Ph.D degree from the University of Melbourne in 2018. Before joining the Chongqing University, he has subsequently worked at the RWTH-Aachen University, University of Toronto, and National University of Singapore. His research interests include positioning and navigation, autonomous systems, SLAM, brain-inspired computing, and deep learning. He is a member of IEEE and ACM.