2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

A Low-Texture Robust Hybrid Feature Based Visual Odometry

He Wang[1], Qi Zhang[2], Zhiwen Zheng[1], Xiaoli Li[3], *Fellow, IEEE* , Hongye Tan[1], Ru Li[1]

***Abstract*— In low-texture scenes, Visual Odometry (VO) algorithms often encounter challenges stemming from sparse feature sets and reduced accuracy in feature matching. To overcome this, integrating plane features and vanishing point characteristics can provide additional constraints for refining camera poses. Optical flow-based tracking methods may also offer improved matching precision compared to traditional feature-based approaches. Motivated by these challenges, we present a robust Visual Odometry system tailored for low-texture environments. Our system combines a vanishing point-based approach for camera pose optimization with a Manhattan-aided algorithm for matching line segments using optical flow. By incorporating planes and vanishing points as supplementary features for pose estimation, we enhance overall accuracy without significant time overhead. We utilize detected line features to compute vanishing points, improving accuracy without compromising efficiency. In addition, our Manhattan-aided optical flow technique supplements and refines the results of line feature matching, further enhancing the accuracy of vanishing points. Evaluation on various public datasets demonstrates the superior accuracy and robustness of our system compared to state-of-the-art Simultaneous Localization And Mapping (SLAM) and VO methods. Notably, our method effectively addresses issues of failure in low-texture scenes and improves the accuracy of line feature matching compared to baseline methods. We will release our source code upon paper acceptance.**

I. INTRODUCTION

Visual Odometry (VO) has attracted widespread attention in recent years, due to its delicate balance between accuracy and computational complexity. As the cornerstone of Simultaneous Localization and Mapping (SLAM) systems, VO's accuracy profoundly impacts the overall accuracy of SLAM [1]. While traditional VO methods excel in various scenarios, they face significant hurdles in low-texture environments where visual tracking becomes notably challenging, consequently impairing the fidelity of the trajectory estimation [2].

In low-texture scenarios, many VO systems rely solely on point features [3], which, given their limited quantity, struggle to maintain continuous camera tracking. To address this limitation, some systems have integrated line features, commonly found in indoor settings [4], [5]. However, the Line Band Descriptor (LBD) [6], used for this purpose, is susceptible to mismatches in areas with similar textures. In such environments, where feature information is sparse, accurately matching line features becomes challenging for LBD [6]. Due to the factors mentioned above, these approaches have fallen short of anticipated results. [7] and [8] aim to enhance pose estimation accuracy by integrating common indoor planes and facet features into SLAM systems. However, since the number of planes in a scene is typically much lower than that of point or line features, the improvement in pose estimation accuracy remains relatively modest. Another well-known strategy is to adopt the Manhattan world (MW) assumption [9] to reduce the rotation drift. [10] and [11] using this assumption achieve more accurate rotation estimation by decoupling poses in man-made environments. However, it is important to note that not all environments adhere to this assumption.

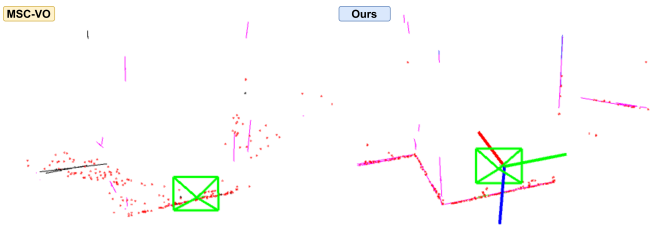


Fig. 1. The mapping performance of MSC-VO (left) is compared to our proposed method (right) in low-texture sequences. Upon tracking failure in MSC-VO where both mapping threads halt, it becomes evident that our approach not only estimates the Manhattan world in the environment but also produces more accurate maps.

To tackle the challenge of limited features in low-texture environments, this study integrates planes and often neglected vanishing points into the system. Notably, our approach computes vanishing points based on extracted line features without adding extra time for feature extraction. Additionally, in order to address the situation of low accuracy in line feature matching in low-texture environments, we introduce a line feature matching technique using Manhattan-assisted KLT, this technique serves as a supplement to descriptor matching for line segments. By utilizing the Manhattan assumption [9] , the camera rotation matrix obtained as an initial value for optical flow is more accurate compared to using the identity matrix. Moreover, omitting the translation vector may lead to optical flow getting trapped in local minima. To address this issue, we introduce two additional constraint terms. Manhattan-assisted KLT enhancing matching accuracy and facilitating vanishing point convergence, which relies on line features. In summary, our contributions include:

*This work was supported in part by the Science and Technology Cooperation and Exchange Special Project of ShanXi Province under Grant 202204041101016, in part by the 1331 Engineering Project of Shanxi Province under Grant 215541015.

[1]He Wang, Zhiwen Zheng, Hongye Tan and Ru Li are with School of Computer and Information Technology, ShanXi University, ShanXi 030006, China wh_sxu@foxmail.com, zzw_cd@foxmail.com, tanhongye@sxu.edu.cn, liru@sxu.edu.cn

[2]Qi Zhang is with the University of Bath, Bath BA2 7AY, UK qz727@bath.ac.uk

[3]XiaoLi Li is with the Institute for Infocomm Research, *A**Star, 1 Fusionopolis Wy, #21-01 Connexis, Singapore 138632 xlli@i2r.a-star.edu.sg

- A simple cost function is proposed for the vanishing point alignment problem in backend optimization, seamlessly integratable into any line feature-based VO or SLAM system to enhance pose estimation accuracy.
- A refined line feature matching strategy is proposed, utilizing an initial rotation derived from Manhattan to furnish a robust initial estimate for the KLT algorithm. It mitigates local extremes by leveraging both the common vanishing point and collinearity constraint.
- We evaluate the proposed method on both the ICL-NUIM and TUM RGB-D datasets, demonstrating its effectiveness, and conduct thorough comparisons with other state-of-the-art VO and SLAM techniques.

II. RELATED WORK

A. Visual SLAM for Low-Texture Scene

Low-texture scenes pose a significant challenge for visual SLAM systems. For instance, Engle et al. introduced DSO and LSD-SLAM, leveraging raw pixel intensities and minimizing photometric error [12], [13]. Gomez-Ojeda et al. incorporated lines into the semi-direct method SVO [14], [15]. Zhou extended this by introducing collinear constraints into DSO's photometric error and presenting a new 3D line representation [12], [16]. Wu et al. exploited deep neural networks for planar segmentation and performed tightly coupled optimization of poses, points, and planes [17]. While direct methods excel in texture-less environments without extra computation for feature extraction, they often encounter challenges with large-scale optimization problems [18].

Indirect methods are prevalent in real-world applications due to their established reliability. A prominent example is ORB-SLAM2 [3], renowned for its capability to achieve localization and mapping in well-textured sequences, leveraging Oriented FAST and Rotated BRIEF (ORB) features for high-performance position recognition [19], [20]. Pumarola et al. extend ORB-SLAM's formulation [21] to handle both point and line correspondences simultaneously [4]. While demonstrating impressive performance in well-textured scenarios, their efficacy diminishes in low-textured environments.

B. Vanishing Point Based SLAM

The Vanishing Point (VP) represents the intersection of parallel lines in the image plane within three-dimensional space, holding valuable geometric properties applicable across various domains. In their work, [22] integrated VP into the optimization framework of line-based SLAM, projecting it onto the unit Gaussian sphere to mitigate non-convergence issues stemming from error accumulation. Furthermore, Lim et al. [23], [24] leveraged VP to not only circumvent the degeneracy problem of line segments but also significantly enhance the precision of camera pose estimation.

C. Line Segment Matching

The LBD (Line Band Descriptor) matching method, as proposed by Zhang [6], is widely utilized for line features due to its impressive effectiveness in many scenarios. However, it may encounter challenges in areas with similar or weak textures, leading to confusion. To address these issues, Gomez-Ojeda introduced a purely geometrical approach for robust line segment matching [25]. Additionally, other methods [26]–[28] enhanced matching accuracy by extending the sparse optical flow method to line segments.

Inspired by [26], [28], we integrate the aforementioned approaches. Notably, we introduce a novel enhancement by incorporating the rotation matrix derived from the Manhattan world into the sparse optical flow method, thereby elevating the accuracy of optical flow prediction.

III. METHODOLOGY

A. System Overview

Our proposed method builds upon MSC-VO [29], as illustrated in Figure 2. Point features are detected and characterized using an ORB detector [30], similar to MSC-VO. Line features are extracted via the Line Segment Detector (LSD) [31], with Manhattan-assisted optical flow employed to enhance the matching accuracy of line features. Additionally, we integrate vanishing points and planes as observation models into our system. Finally, we use g2o [32] to minimize the reprojection error, facilitating camera pose estimation.

B. Observational Models

As sole reliance on point and line features often fails to yield robust position estimation in low-texture environments, integrating vanishing points computed from extracted line features becomes crucial. Moreover, given the prevalence of untextured planar regions in indoor environments, considering vanishing points and planar surfaces becomes necessary.

Points: Points are detected and described using ORB [30]. Each 3D point is represented as $\mathbf{P} = (X, Y, Z)$, with its 2D observation denoted as $\mathbf{p}_{obs} = (u, v)$. Matches are established by projecting 3D points onto the image and identifying the closest observation based on the Hamming distance between their respective descriptors.

Lines: To detect and match line features in each frame, we employ a robust LSD detector [31] and an improved LBD descriptor [6]. We represent 3D lines and their 2D observations with their endpoints (S^l, E^l) and (s^l, e^l), respectively. For a 2D line segment $\mathbf{l}_k$, its direction vector is denoted as:

$$\mathbf{l}_k = \frac{\mathbf{e}_k^l - \mathbf{s}_k^l}{\left\|\mathbf{e}_k^l - \mathbf{s}_k^l\right\|} \tag{1}$$

Planes: To detect planes, we utilize the AHC method [33] to extract them from the downsampled 3D point cloud. Subsequently, for matching the detected planes with those in the map, we adopt the approach outlined in [7]. The extracted planes are parameterized using the Hessian form as:

$$\pi = (\mathbf{n}^\top, \mathbf{d})^\top \tag{2}$$

Here, $\mathbf{n} = (n_x, n_y, n_z)^\top$ is the normal vector to the plane, while d is the distance of the plane from the origin [34].

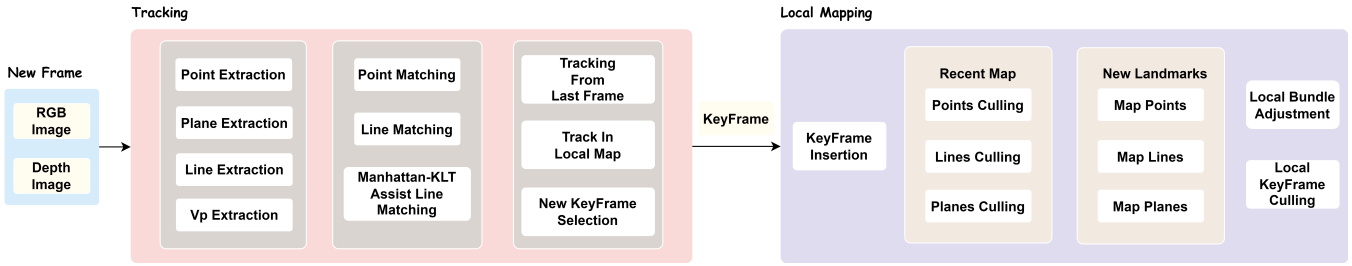


Fig. 2. The framework of the proposed method. When receiving a new image input, the point, line, vanishing point, and plane features are extracted first, then the segment matching is optimized in the feature matching stage, followed by the estimation of the camera pose, and the camera pose and landmarks are optimized in the local map optimization.

Vanishing Points: We detect vanishing points in images as per [34], with the vanishing point denoted as $\mathbf{v} = (v_x, v_y, 1)$.

Let's denote the points on the 3D line passing through point $\mathbf{A}$ with direction $\mathbf{D} = \left(\mathbf{d}^\top, \mathbf{0}\right)^\top$ as $X(\lambda) = \mathbf{A} + \lambda\mathbf{D}$. Under the influence of the camera intrinsic parameters $\mathbf{P} = \mathbf{K}[\mathbf{I}|\mathbf{0}]$, the point $X(\lambda)$ is projected onto the image as:

$$x(\lambda) = \mathbf{P}X(\lambda) = \mathbf{PA} + \lambda\mathbf{PD} = \mathbf{a} + \lambda\mathbf{Kd} \tag{3}$$

where $\mathbf{a}$ denotes the image of point $\mathbf{A}$. Consequently, the vanishing point $\mathbf{v}$ of this line can be determined by taking the limit in the following manner:

$$\mathbf{v} = \lim_{\lambda\to\infty} x(\lambda) = \lim_{\lambda\to\infty} (\mathbf{a} + \lambda\mathbf{Kd}) = \mathbf{Kd} \tag{4}$$

C. Manhattan-KLT

Considering the stability and accuracy of line feature matching, we propose to use drift-free rotations obtained by Manhattan-aided KLT to predict the positions of sampled points along the line segments, which is built on top of the methods in [26]. As a result, we can derive the following:

$$\begin{cases} \eta_2 K^{-1} u_2 = \eta_1 R K^{-1} u_1 + t \\ u_2 = K R K^{-1} u_1 \\ R_{C_i,w} = R_{C_i,m} R_{m,w} \\ R_{i,i-1} = R_{i,w} R_{i-1,w}^T \end{cases} \tag{5}$$

In considering two successive camera frames $\mathbf{C}_1$ and $\mathbf{C}_2$, let $\mathbf{u}_1$ and $\mathbf{u}_2$ represent the corresponding pixel points on these frames, and denote their depths as η_1 and η_2, respectively. The first equation in (5) can then be deduced.

Here, $\mathbf{R}$ and $\mathbf{t}$ represent the rotation matrix and translation vector respectively, between the two frames $\mathbf{C}_1$ and $\mathbf{C}_2$. $\mathbf{K}^{-1}$ denotes the inverse intrinsic matrix of the camera. Given that the depths of corresponding points between the two frames are approximately equal and the translation vectors are negligibly small, we can simplify the first equation in (5) to yield the second equation in (5).

Utilizing the Manhattan assumption, we can derive a rotation $\mathbf{R}_{C_i,m}$ from the Manhattan world to the current camera frame $\mathbf{C}_i$. Using this, we can obtain $\mathbf{R}_{C_i,w}$ from the third equation in (5).

Given that the camera rotation $\mathbf{R}_{C_{i-1},w}$ of the $\mathbf{C}_{i-1}$ frame is known, obtaining the rotation matrix from frame $\mathbf{C}_{i-1}$ to the current frame $\mathbf{C}_i$ is straightforward. This can be represented as the fourth equation in (5).

For all line segments $L = \{l_1, l_2, ..., l_n\}$ extracted by the LSD [31] in each frame $\mathbf{C}_i$, we aim to ensure the accuracy of line feature prediction. To achieve this, we align the line segments based on their lengths for uniform sampling and filter out sampling points lacking depth information.

When deriving the camera rotation from the previous frame to the current frame, leveraging the second equation in (5), we determine the position of the sampling point $\mathbf{u}_{i-1}$ in the previous frame at the prediction point $\mathbf{u}_i$ in the current frame. Once $\mathbf{u}_i$ in the current frame is obtained, we utilize the KLT sparse optical flow to track $\mathbf{u}_i$ and obtain the final prediction point $\mathbf{u}_i'$. Subsequently, the RANSAC algorithm is employed to robustly estimate line segments by fitting the predicted sampled points. To achieve this, we employ an extended Lucas-Kanade optical flow algorithm with a multi-level pyramid to track the sample points, building on the assumptions of grayscale invariance and neighborhood motion consistency. This process yields:

$$\begin{cases} \epsilon_1 = \sum_{j=1}^{k} (C_1(x_1^j, y_1^j) - C_2(x_1^j + t_{x1}, y_1^j + t_{y1})) \\ \dots \\ \epsilon_n = \sum_{j=1}^{k} (C_1(x_n^j, y_n^j) - C_2(x_n^j + t_{xn}, y_n^j + t_{yn})) \end{cases} \tag{6}$$

Here, each line segment is represented by sampled points in the form of $(x_1, y_1), ..., (x_n, y_n)$, where the translation vectors of the sampling points are defined as $(t_{x1}, t_{y1}), ..., (t_{xn}, t_{yn})$. $C_i(x, y)$ denotes the pixel value of point (x, y) in the i-th frame, and k represents the number of all pixels in the neighborhood of these sampled points.

Since the second equation in (5) omits the translation vector of the sampling points, and considering that lines are more prone to depth discontinuities and occlusions compared to points, this positioning method presents challenges in directly locating corresponding line features. In light of this, drawing inspiration from [28], we introduce additional structural constraints to address this limitation.

Co-VP constraint. Across two successive frames, the vanishing point of the same line segment is typically similar. Hence, the Euclidean distance between the vanishing points

of the predicted line segment and the original line segment in the pixel plane should ideally be 0. First, we compute the vanishing points $\mathbf{v}_1$ and $\mathbf{v}_2$ of the original and predicted line segments respectively using (4), where θ_1 denotes the weight value. Thus, we define the co-vanishing point constraint as follows:

$$\mathbf{E}^{vp} = \left\| \sqrt{(v_1 - v_2)^2} \right\|^2 \cdot \theta_1 \tag{7}$$

Collinearity constraint. We maintain adherence to the collinearity constraint outlined in [28]. Across two successive camera frames, the lengths between the two endpoints and the midpoint of a line segment are approximately conserved. The collinearity constraint is expressed as follows:

$$\mathbf{E}^{c} = \| \triangle u_s + \triangle u_e - 2 \triangle u_m \|^2 \cdot \theta_2 \tag{8}$$

where θ_2 is the weight value. $\triangle u_s$, $\triangle u_e$ and $\triangle u_m$ represent the distances between the starting point, ending point, and midpoint, respectively.

By incorporating the two constraints described above into (6), we obtain:

$$\varpi = \underset{\varpi}{argmin} \left[\sum_{i=1}^{n} \rho(\epsilon_i) + \rho(\mathbf{E}^{vp}) + \rho(\mathbf{E}^{c}) \right] \tag{9}$$

Here, $\varpi = \{\triangle u_1, \triangle u_2, ..., \triangle u_n\}$ represents the translation vector of the sample points before and after applying optical flow. We employ the Gauss-Newton method to solve this optimization problem. Introducing additional constraints enhances the accuracy of optical flow tracking for sample point positions, mitigating potential local extrema arising from rapid camera motion or repetitive texture scenes. Moreover, these constraints aid in reducing suboptimal sample point predictions based on prior Manhattan information.

D. Pose Estimation

After achieving accurate feature matching, the optimization process proceeds to obtain the camera pose $\mathbf{T} \in SE(3)$. This involves minimizing the cost function by addressing reprojection error:

$$\begin{aligned} \mathbf{T} = \underset{T}{argmin} & \sum_{i \in P} \rho(E_i^p) + \sum_{j \in L} \rho(E_j^l) \\ & + \sum_{k \in S} \rho(E_k^s) + \sum_{n \in V} \rho(E_n^v) \end{aligned} \tag{10}$$

where ρ represents the robust Huber cost function. P, L, S, and V denote the sets of all point, line, plane, and vanishing point matches, respectively. Finally, we employ the Levenberg-Marquardt algorithm to estimate the optimal camera pose. The definitions of each error term in (10) are outlined as follows:

Point Residue:

$$\mathbf{E}_i^p = \| p_i - \Pi \left(R_{cw} P_i + t_{cw} \right) \|_{\Sigma^p}^2 \tag{11}$$

Here, p_i denotes the pixel coordinates of a 2D feature point, and P_i represents its corresponding 3D point in the map. The function Π is responsible for projecting 3D points from space onto the 2D pixel plane. $R_{cw} \in SO(3)$ and $t_{cw} \in \mathbb{R}^3$ denote the current camera rotation and translation, respectively, Σ is the corresponding covariance matrix.

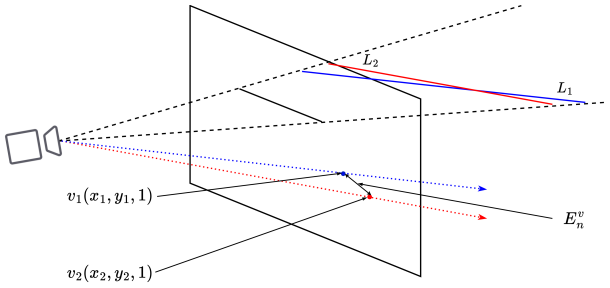


Fig. 3. Illustration of the vanishing point residual.The blue line segment and the red line segment depict the observation and estimation, respectively, of the 3D line corresponding to the line feature. In estimating the vanishing point, we utilize the direction vector of the straight line. The residual is defined as the difference between the observed vanishing point from the observed line and the estimated vanishing point from the 3D line.

Line Residue:

$$\mathbf{E}_j^l = \left\| [d(x_s^j, l^j), d(x_e^j, l^j)] \right\|_{\Sigma^l}^2 \tag{12}$$

where x_s^j and x_e^j denote the two endpoints of the 3D line segment, which, after being projected through Π, are represented as points on the 2D pixel coordinates. l^j represents the corresponding line feature on the 2D pixel plane. $d(x, l)$ denotes the distance between a point and a line.

Plane Residue: For the representation of residuals for a plane, we follow [35]:

$$\mathbf{E}_k^s = \| q(\pi_k) - q(T_{cw} \pi_x) \|_{\Sigma^s}^2 \tag{13}$$

Here π_k represents the observed plane in the frame, while π_x denotes the corresponding map plane. In addition, q denotes:

$$q(\pi) = (\arctan(\frac{n_y}{n_x}), \arcsin(n_z), d) \tag{14}$$

Vanishing Point Residue: As shown in Figure 6, the vanishing point residual is as follows:

$$\mathbf{E}_n^v = \left\| d\left(v_n, \hat{v_n}\right) \right\|_{\Sigma^v}^2 \tag{15}$$

where v_n and $\hat{v_n}$ denote the vanishing point coordinates of the corresponding line feature in the camera coordinate system, given in the pixel coordinate system and the world coordinate system, respectively. The error is computed as the distance between them.

IV. Experimental Results

In this section, we showcase our experimental results to demonstrate the performance of our proposed method. Subsequently, we evaluate the accuracy of line feature matching, presenting results on the ICL-NUIM and TUM RGB-D datasets, conducting an ablation study, and scrutinizing the system's time overhead. The experiments were conducted using an Intel Core i7-7700 CPU with 32GB of memory, and were specifically performed on two benchmark datasets: the ICL-NUIM dataset and the TUM RGB-D dataset.

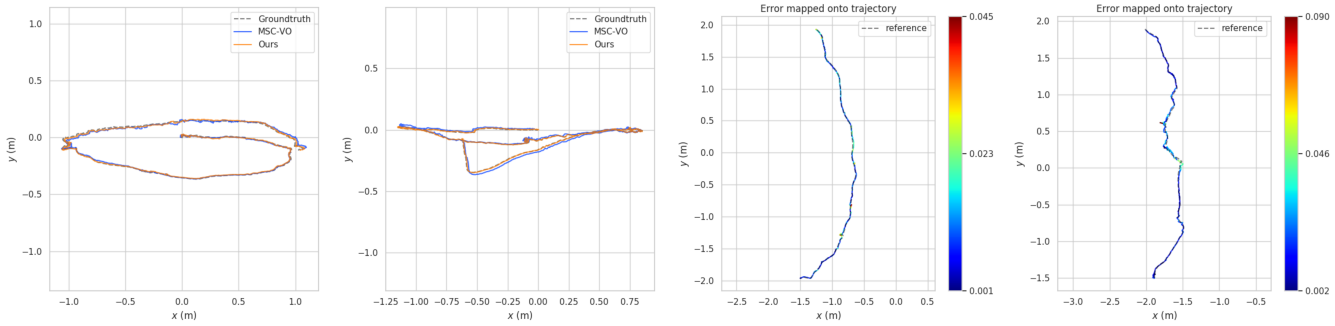


Fig. 4. (a) and (b) compare the trajectory plots of MSC-VO and our method on two sequences from the ICL-NUIM dataset. As can be seen from the figure, our method has a higher degree of alignment with the true trajectory compared to the baseline. (c) and (d) depict the trajectory error plots of our method on two low-texture sequences from the TUM RGB-D dataset, please note that the baseline method fails to track in these two sequences.

A. Evaluation of Line Feature Matching

To evaluate the performance of line tracking, we compare our line tracking algorithm with the popular descriptor-based approach LBD [6]. We use the LSD [31] algorithm to detect line features. To prevent many redundant lines, we fix the maximum number of detected lines per frame to 40. As shown in Table I, we report the average number of correct matches and the inlier ratio of sequences from the ICL-NUIM dataset.

TABLE I

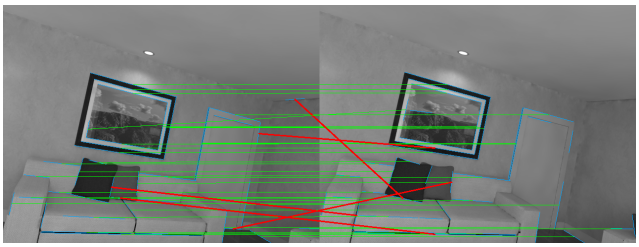
THE PERFORMANCE OF TRACKING LINE FEATURES USING THE ICL-NUIM DATASET WAS EVALUATED.

Sequences	LSD + LBD (%)	Proposed (%)
lr-kt0	83.80	**93.80**
lr-kt1	86.66	**96.50**
lr-kt2	83.17	**93.50**
lr-kt3	81.13	**94.35**
of-kt0	89.27	**96.22**
of-kt1	82.28	**93.97**
of-kt2	85.16	**94.85**
of-kt3	86.26	**95.72**
Average	84.72	**94.86**

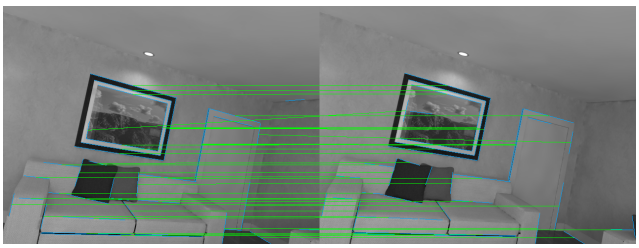
In Table I, we present the accuracy of line feature matching on the ICL-NUIM dataset. While the LBD descriptor matching [6] has demonstrated commendable results, we observe that a slight increase in runtime enables us to enhance accuracy by ten percent. We deem this improvement to be highly significant for enhancing system performance. Visually, Figure 5 highlights the disparities between our approach and the LBD descriptor matching method [6]. Our method complements the LBD [6] approach by rectifying erroneous feature line matches between frames and augmenting the number of successful matches.

B. Evaluation of VO

To validate the efficacy of our proposed method, we conducted comparisons with state-of-the-art VO and visual SLAM systems, including MSC-VO [29], RGB-D SLAM [35], and PS-SLAM [7]. MSC-VO [29] introduces a robust RGB-D VO framework tailored for low-textured environments, aiming to enhance pose accuracy in scenes featuring structural regularities and MA alignment. However, its performance is compromised in low-textured scenes due to fewer detected features and lower accuracy in feature matching. PS-SLAM [7] mitigates drift errors in indoor environments by incorporating perpendicular and parallel constraints for plane features. Although it offers advantages in this regard, its source code is unfortunately not available. RGB-D SLAM [35] achieves superior performance by decoupling the rotation and translation of the camera. Additionally, we included ORB-SLAM2 [3], LPVO [10], DVO [36], and ManhSLAM [37] in our baselines for comparison. To ensure fairness in our experiments, we ran the source code of these projects on our machine to maintain a consistent environment. Our method demonstrates robust operation on low-



(a) LSD [31] + LBD [6]



(b) LSD [31] + Ours

Fig. 5. Line tracking results obtained using our method and the LBD [6] approach, employing the LSD [31] line detector. The image pairs are sourced from the ICL-NUIM dataset. False matches identified by the LBD [6]-based method are highlighted in red lines.

TABLE II

TRANSLATIONAL ERROR (RMSE) AND BEST COUNT FOR ICL-NUIM AND TUM RGB-D SEQUENCES (UNIT: M). 'X' INDICATES THE METHOD LOST TRACK DURING THE TRACKING PROCESS, WHILE 'W/O' SIGNIFIES THAT RESULTS ARE NOT AVAILABLE.

Sequence	ORB-SLAM2	PS-SLAM	RGB-D SLAM	ManhSLAM	LPVO	DVO	MSC-VO	Ours
lr-kt0	0.025	0.016	**0.006**	0.007	0.015	0.108	0.011	**0.006**
lr-kt1	**0.008**	0.018	0.015	0.011	0.039	0.059	0.010	**0.008**
lr-kt2	0.023	0.017	0.020	0.015	0.034	0.375	0.012	**0.007**
lr-kt3	0.021	0.025	0.012	**0.011**	0.102	0.433	0.041	0.027
of-kt0	0.037	0.032	0.041	0.025	0.061	0.244	0.028	**0.022**
of-kt1	0.029	0.019	0.020	**0.013**	0.052	0.178	0.021	0.015
of-kt2	0.039	0.026	0.011	0.015	0.039	0.099	0.023	**0.007**
of-kt3	0.065	**0.012**	0.014	0.013	0.030	0.079	**0.012**	0.014
fr1-xyz	0.010	0.010	w/o	0.010	w/o	0.026	0.010	**0.009**
fr1-desk	0.022	0.026	w/o	0.027	w/o	0.036	**0.019**	0.020
fr2-xyz	0.009	0.009	w/o	0.008	w/o	w/o	**0.005**	**0.005**
fr2-desk	0.040	0.025	w/o	0.037	w/o	w/o	0.025	**0.021**
cabinet	0.075	0.067	0.035	**0.023**	0.520	0.690	x	0.055
large-cabinet	0.124	0.079	**0.071**	0.083	0.279	0.979	0.120	0.089
snot-far	x	0.020	0.022	0.040	0.075	0.213	x	**0.012**
snot-near	x	**0.013**	0.025	0.023	0.080	0.076	x	0.020
Best-Count	1	2	2	3	0	0	3	**9**

texture sequences and delivers strong performance across other sequences as well. Figure 4 provides a comparison of trajectory errors between MSC-VO [29] and our proposed method, along with the trajectory error performance of our method specifically in low-texture sequences.

Table II highlights the performance of the proposed method across 16 sequences, where it achieves the best performance in 9 sequences and the second-best in 3, based on translational RMSE (ATE). The two most accurate results are underscored for emphasis. Notably, our method surpasses the baseline MSC-VO [29] in 14 out of the 16 sequences, demonstrating robust operation even in low-texture environments. While MSC-VO [29] excels in sequences like of-kt3, fr1-desk, and fr2-xyz, it encounters challenges in some low-texture sequences, resulting in tracking loss. In scenes with well-defined structures, methods leveraging Manhattan world assumptions for decoupling camera pose, such as ManhSLAM [37] and RGB-D SLAM [35], exhibit significant accuracy. Notable performance is observed in challenging scenes like lr-kt3 and cabinet for ManhSLAM [37]. In the lr-kt3 sequence, suboptimal results arise as the camera approaches a textureless wall, while rapid motion around a rectangular box in the cabinet sequence accentuates the advantages of methods like ManhSLAM [37] and RGB-D SLAM [35]. Although our method's performance degrades in such scenarios, it still outperforms PS-SLAM [7]. In low-texture sequences like large-cabinet and snot, our method achieves comparable or superior performance to ManhSLAM [37] by leveraging accurate vanishing points and line feature matching. Our approach not only demonstrates robustness in low-texture sequences but also achieves higher accuracy than MSC-VO [29] in most other sequences. On average, our method achieves a reduction of 30.52% and 5.82% in errors on the ICL-NUIM and TUM RGB-D datasets, respectively. As scenes where MSC-VO [29] tracking failed were excluded, the improvement on the TUM RGB-D dataset is less significant compared to the ICL-NUIM dataset. Additionally, Figure 6 illustrates the absolute rotation pose error for ORB-SLAM2 [3], MSC-VO [29], and our method, showcasing the superior performance of our method in terms of both translation and rotation compared to the other two methods.

TABLE III

TRANSLATIONAL ERROR(RMSE) AND BEST COUNT FOR ICL-NUIM AND TUM RGB-D SEQUENCES (UNIT: M).

Sequence	MSC-VO	VP	VP+Plane	E	Ours
lr-kt0	0.011	0.007	**0.006**	0.007	**0.006**
lr-kt1	0.010	0.009	0.009	0.011	**0.008**
lr-kt2	0.012	0.008	0.008	0.008	**0.007**
lr-kt3	0.041	0.038	0.031	0.029	**0.027**
of-kt0	0.028	0.026	0.024	0.023	**0.022**
of-kt1	0.021	0.022	0.021	0.017	**0.015**
of-kt2	0.023	0.015	0.009	**0.007**	**0.007**
of-kt3	**0.012**	**0.012**	0.015	0.016	0.014
cabinet	x	x	x	0.062	**0.055**
large-cabinet	0.120	**0.083**	0.090	0.116	0.089
snot-far	x	x	x	0.014	**0.012**
snot-near	x	x	x	**0.016**	0.020

C. Ablation Study

We propose the use of vanishing points and Manhattan-assisted optical flow to enhance system performance in low-texture scenes. In this section, we conduct an ablation study to evaluate the effectiveness of each method. For the ablation study of Manhattan-assisted optical flow, we follow the approach outlined in [26], where the identity matrix is used to assist optical flow, denoted as E. Subsequently, we remove the optical flow assistance and denote it as VP+Plane, and further eliminate the Plane component to obtain VP.

Table III shows that incorporating vanishing points as features into VO or SLAM systems improves the accuracy of camera poses. Moreover, utilizing Manhattan-assisted optical

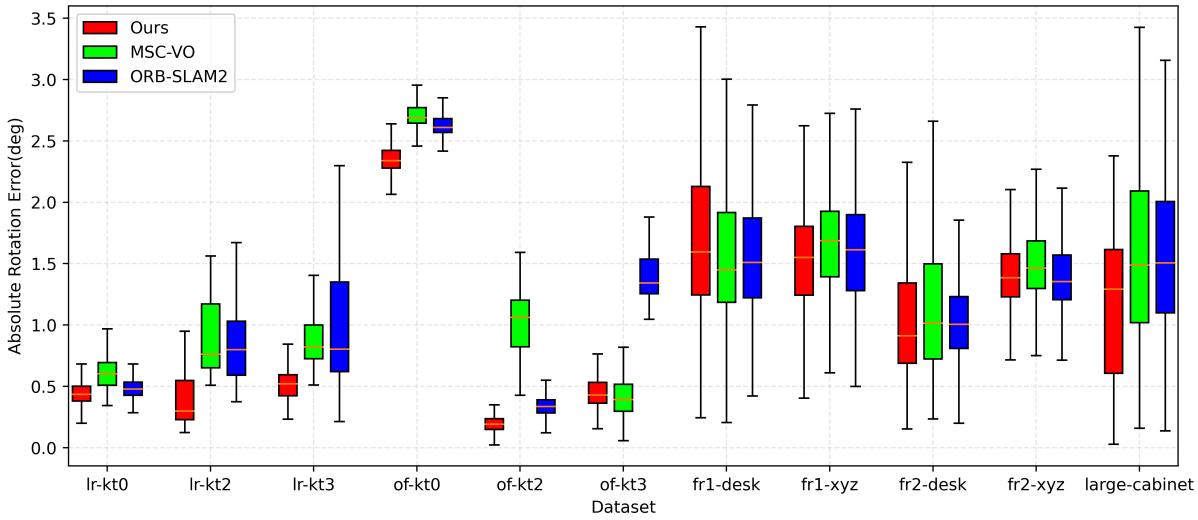


Fig. 6. Box plot of Absolute Rotation Error according to some sequences for ORB-SLAM2 [3], MSC-VO [29] and our method.

flow outperformed the baseline in 11 out of 12 sequences, affirming the effectiveness of our proposed method.

D. Runtime Analysis

This section presents the running time analysis of the proposed system. Table IV compares the average runtime per frame for each module of MSC-VO [29] with our method. The modules include PE for point extraction, LE for line extraction, PLE for plane extraction, CPE for camera pose estimation, and LBA for local bundle adjustment. We assess the runtime of both methods on four sequences from the ICL-NUIM dataset, respectively. It's important to note that our method integrates the extraction of line features with vanishing point features, where the detection of vanishing points incurs minimal additional computational overhead. Notably, our method demonstrates higher runtime in camera pose estimation and local map optimization modules compared to MSC-VO [29]. This can be attributed to the detection, tracking, and optimization of more stable features in our system. Despite the increased computational costs, our method effectively mitigates tracking failures in low-texture scenarios. Furthermore, these modules operate in parallel, facilitating efficient processing, and thus the overall system runtime is not simply the sum of individual module runtimes.

TABLE IV

THE AVERAGE RUNNING TIME COMPARISON OF PRINCIPAL COMPONENTS WITH MSC-VO. - DENOTES ABSENCE OF SUCH MODULE IN THIS METHOD.

	PE	LE	PLE	CPE	LBA
MSC-VO	9.63 ms	28.28 ms	-	28.10 ms	345.58 ms
Ours	9.34 ms	25.87 ms	97.48 ms	79.67 ms	554.18 ms

V. CONCLUSIONS

In this paper, we proposed a feature-based visual odometry (VO) system designed to navigate low-texture environments with robustness. Our approach integrates vanishing points and planes as supplementary features and resolves the challenge of inaccurate line feature matching in such scenarios. We achieve this by employing a Manhattan-assisted KLT algorithm to complement and refine the results of line feature matching. Experimental results affirm the superior performance of our proposed method in low-texture environments. Looking ahead, we see potential in exploring the integration of traditional visual SLAM with 3D Gaussian Splatting to generate more intuitive maps, offering promising avenues for future research.

REFERENCES

[1] N. Yang, R. Wang, and D. Cremers, "Feature-based or direct: An evaluation of monocular visual odometry," *arXiv preprint arXiv:1705.04300*, pp. 1–12, 2017.

[2] A. Tourani, H. Bavle, J. L. Sanchez-Lopez, and H. Voos, "Visual slam: what are the current trends and what to expect?" *Sensors*, vol. 22, no. 23, p. 9297, 2022.

[3] 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.

[4] 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.

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

[6] 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.

[7] 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.

[8] H. M. Cho, H. Jo, and E. Kim, "Sp-slam: Surfel-point simultaneous localization and mapping," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 5, pp. 2568–2579, 2021.

[9] J. M. Coughlan and A. L. Yuille, "Manhattan world: Compass direction from a single image by bayesian inference," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. IEEE, 1999, pp. 941–947.

[10] P. Kim, B. Coltin, and H. J. Kim, "Low-drift visual odometry in structured environments by decoupling rotational and translational motion," in *2018 IEEE international conference on Robotics and automation (ICRA)*. IEEE, 2018, pp. 7247–7253.

[11] ——, "Linear rgb-d slam for planar environments," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 333–348.

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

[13] 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.

[14] R. Gomez-Ojeda, J. Briales, and J. Gonzalez-Jimenez, "Pl-svo: Semi-direct monocular visual odometry by combining points and line segments," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4211–4216.

[15] 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.

[16] L. Zhou, S. Wang, and M. Kaess, "Dplvo: Direct point-line monocular visual odometry," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7113–7120, 2021.

[17] F. Wu and G. Beltrame, "Direct sparse odometry with planes," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 557–564, 2021.

[18] M. Outahar, G. Moreau, and J.-M. Normand, "Direct and indirect vslam fusion for augmented reality," *Journal of Imaging*, vol. 7, no. 8, p. 141, 2021.

[19] D. G. Viswanathan, "Features from accelerated segment test (fast)," in *Proceedings of the 10th workshop on image analysis for multimedia interactive services, London, UK*, 2009, pp. 6–8.

[20] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*. Springer, 2010, pp. 778–792.

[21] 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.

[22] J. Ma, X. Wang, Y. He, X. Mei, and J. Zhao, "Line-based stereo slam by junction matching and vanishing point alignment," *IEEE Access*, vol. 7, pp. 181 800–181 811, 2019.

[23] H. Lim, Y. Kim, K. Jung, S. Hu, and H. Myung, "Avoiding degeneracy for monocular visual slam with point and line features," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 675–11 681.

[24] H. Lim, J. Jeon, and H. Myung, "Uv-slam: Unconstrained line-based slam using vanishing points for structural mapping," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1518–1525, 2022.

[25] R. Gomez-Ojeda and J. Gonzalez-Jimenez, "Geometric-based line segment tracking for hdr stereo sequences," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 69–74.

[26] H. Wei, F. Tang, C. Zhang, and Y. Wu, "Highly efficient line segment tracking with an imu-klt prediction and a convex geometric distance minimization," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3999–4005.

[27] L. Xu, H. Yin, T. Shi, D. Jiang, and B. Huang, "Eplf-vins: Real-time monocular visual-inertial slam with efficient point-line flow features," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 752–759, 2022.

[28] D. Yan, H. Jiang, T. Li, and C. Shi, "Efficient vanishing point estimation for accurate camera rotation estimation in indoor environments," *IEEE Robotics and Automation Letters*, 2023.

[29] J. P. Company-Corcoles, E. Garcia-Fidalgo, and A. Ortiz, "Msc-vo: Exploiting manhattan and structural constraints for visual odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2803–2810, 2022.

[30] 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.

[31] 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.

[32] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3607–3613.

[33] 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.

[34] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[35] 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.

[36] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 2100–2106.

[37] R. Yunus, Y. Li, and F. Tombari, "Manhattanslam: Robust planar tracking and mapping leveraging mixture of manhattan frames," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6687–6693.