Computational Visual Media
DOI 10.1007/s41095-xxx-xxxx-x Vol. x, No. x, month year, xx–xx

Research Article

Robust and Efficient Edge-based Visual Odometry

Feihu Yan[1], **Zhaoxin Li**[2], **and Zhong Zhou**[1](✉)

Abstract Visual odometry, which aims to estimate relative camera motion between sequential video frames, has been widely used in the fields of augmented reality, virtual reality, and autonomous driving. However, it is still quite challenging for state-of-the-art approaches to handle low-texture scenes. In this paper, we propose a robust and efficient visual odometry algorithm that directly utilizes edge pixels to track camera pose. In contrast to direct methods, we choose reprojection error to construct the optimization energy, which can effectively cope with illumination changes. The distance transform map built upon edge detection for each frame is used to improve tracking efficiency. A novel weighted edge alignment method together with sliding window optimization is proposed to further improve the accuracy. Experiments on public datasets show that the method is comparable to state-of-the-art methods in terms of tracking accuracy, while being faster and more robust.

Keywords visual odometry, edge structure, distance transform, low-texture.

1 Introduction

As one of the essential technologies in many emerging applications, such as robot navigation [1], augmented reality (AR) [26], and virtual reality (VR) [30], simultaneous localization and mapping (SLAM) has received widespread attention in recent years. With the help of various sensors, such as cameras, lidar, etc., SLAM can build a 3D model of the surrounding environment while tracking the position of the sensor. Specifically, SLAM using only cameras is referred to as visual SLAM [6, 8, 19, 27, 29]. Generally regarded as a component or special case of visual SLAM, visual odometry (VO) [5, 7, 10, 20, 45] mainly solves the basic problem of how to track camera pose in unknown environments.

Typical existing VO algorithms can be divided into two categories. The first comprises feature-based (indirect) methods [4, 28, 29], which extract corner points or other distinguishable feature points from images, and estimates camera motion by minimizing the reprojection error of matched features. The second category comprises direct methods [5, 6], which directly use image pixels to estimate camera pose by optimizing photometric error.

Benefitting from the invariance of feature descriptors, the feature-based method is robust for a variety of well-textured scenes, but the performance drops in low-texture areas since it relies highly on extracting sufficient features. Although the direct methods deal better with this low-texture problem by using more image information, the fundamental assumption on photo consistency makes them quite sensitive to illumination changes, which also affect their robustness in real applications.

To make VO systems more robust, researchers began to pay attention to high-level features, such as edge- [23, 34–36, 40, 45] or line- [12–15, 18, 21, 25, 33, 37, 41, 43, 46] features which are widespread in man-made scenes. Integration of edge information has been proven to improve the robustness of VO systems, but also brings extra computational overhead and has a negative effect on real-time performance.

In this paper, we propose a robust and efficient edge-based VO system, called *distance transform visual odometry* (DTVO), to mitigate the above mentioned problems. Based on the observation that edge

1 State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing 100191, China. E-mail: F. Yan, yanfeihu@buaa.edu.cn; Z. Zhou, zz@buaa.edu.cn (✉).

2 Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China. E-mail: cszli@hotmail.com.

information is abundant in man-made environments, even in homogeneous areas like wall surfaces, our method takes advantage of edge features to deal with the low-texture problem. More specifically, edge pixels are detected from each frame using the Canny algorithm [2] and utilized throughout the tracking process. In contrast to direct VO systems, we employ the geometric reprojection error instead of the photometric residual, which is more robust with respect to illumination changes. To meet the demands of real-time tracking, edges in the reference frame are projected into the distance transform (DT) map of the current frame to efficiently calculate the residuals. Moreover, a novel dynamic weighted edge registration method combined with a pyramid coarse-to-fine scheme is proposed to improve tracking accuracy. A local sliding window optimization step is also integrated to refine the depth map as well as camera pose. The proposed method is not only suitable for monocular cameras but can also be extended to RGB-D sensors. The proposed VO system is able to achieve 70 fps for 640×480 resolution images from public datasets on a CPU.

Our contributions can be summarized as:

- a 3D-2D edge alignment method that effectively tracks camera pose by leveraging local geometric information in the edge-driven DT map,
- a sliding window optimization approach to refine the propagated depth map, as well as all camera poses, within the local window, and
- using the above two modules, a novel edge-based VO system that efficiently integrates edge information to more robustly extract camera pose.

To the best of our knowledge, this is the first real-time VO pipeline for monocular cameras that utilizes DT maps. Furthermore, it can be extended to RGB-D sensors.

The remainder of this paper is organized as follows. Section 1 introduces related work. Section 3 provides the problem formulation. Tracking and local mapping based on edges are presented in Sections 4 and 5 respectively. In Section 6, we provide experimental comparisons with state-of-art algorithms. Finally, conclusions and future work are discussed in Section 7.

2 Related work

In this section, we describe related work on visual SLAM (including VO) which can be broadly divided into feature-based methods, direct methods, and edge-related approaches.

2.1 Feature-based SLAM

Feature-based methods have dominated the field of visual SLAM in the past decades. Most of them are inspired by the framework of PTAM (*parallel tracking and mapping*) [22], one of the most groundbreaking SLAM systems. PTAM divides the entire system into two independent threads, a real-time camera tracking front-end, and a mapping back-end based on optimization. One of the most representative works in recent years is ORB-SLAM [28, 29], which extracts ORB features for tracking and mapping, and innovates with a three-thread architecture by adding a loop closing component. Due to its high tracking accuracy and good scalability, ORB-SLAM has become one of the most popular state-of-the-art SLAM frameworks. However, this kind of method needs to detect reliable feature points in images, preventing their application in low-textured environments.

2.2 Direct Methods

In contrast to feature-based approaches, direct methods directly use whole image alignment based on photometric error to track camera pose. Omitting the steps of feature extraction and descriptor calculation makes the system more efficient. At the same time, more image information also means a more dense map can be recovered. These advantages have increased the popularity of the direct method in recent years. Newcombe et al. [31] presented *dense tracking and mapping* (DTAM), which achieves dense and smooth depth estimation by using a non-convex optimization process. This system needs GPU acceleration to run in real time. Engel et al. [6, 7] proposed the large scale direct monocular SLAM (LSD-SLAM) system, which performs tracking and mapping directly over image pixel intensities. It is impressive that this system is able to reconstruct a semi-dense map and operate in real time without GPU acceleration. The most influential direct method recently is *direct sparse odometry* (DSO) [5], which provides state-of-the-art performance in terms of both accuracy and robustness for monocular camera tracking.

LSD-SLAM [6] and DSO [5] are quite popular amongst direct methods, and there are many extensions [3, 8, 11, 39] to these two methods. There are also several semi-direct systems that combine the complementary strengths of direct and feature-based methods, such as SVO [10] and LCSD-SLAM [24].

The main problem of direct methods is that they rely on photometric minimization, making them sensitive to illumination changes.

2.3 Edge-related SLAM

Researchers have tried to integrate edge information into visual SLAM for a long time. Early works like [12, 37] integrated straight lines into the filter-based method, but many mismatches occur when the help of descriptors is unavailable.

Intuitively, the feature-based method is based on the extraction and matching of feature points and descriptors, which can be easily extended to edge features and descriptors. Albert et al. [33] proposed PL-SLAM, an extension of ORB-SLAM [28], which uses a *line segment detector* (LSD) [16] to extract linear features and calculates a *line band descriptor* (LBD) [44]. Zhang et al. [42] added line features detected by LSD [16] to ORB-SLAM [28], which provides long-term constraints using planes. Zuo et al. [46] fully took into account the parametric representation of spatial straight-lines, and introduced the orthogonal representation method to solve the problem of over-parameterization.

Incorporating line features into direct methods can also improve robustness. On the basis of direct visual odometry [7], Yang et al. [41] introduced LSD [16] and LBD [44] into a direct SLAM framework that can improve robustness, but the line segment detection and descriptor calculation, as well as subsequent feature matching, greatly increased computational consumption. Li et al. [25] introduced a collinear constraint into the DSO framework [5] through line segments detected by LSD [16], and reduced the computational cost by removing non-line pixels.

Ruben et al. [13] introduced LSD [16] into the SVO [10] framework to obtain a more robust system capable of dealing with untextured environments. They subsequently combined ORB features and line features to propose a VO method [14] and a visual SLAM method [15] for stereo cameras, which can produce rich geometrical maps, but their odometry method requires reduced image resolution to achieve real-time performance.

The above methods show that the introduction of edge features can effectively improve robustness, but extracting line features and matching corresponding descriptors are time-consuming, which precludes real-time application. Moreover, the over-parameterization problem is also a hindrance for optimization.

Instead of line segment features, some researchers try to estimate camera motion directly from the edge pixels. Wang et al. [40] presented a real-time RGB-D VO system that combines photometric error with edge distance error provided by the DT map. Manohar et al. [23] proposed a direct RGB-D VO system which utilizes the sub-gradient method to handle non-differentiable functions. Fabian et al. proposed DT-based RGB-D VO systems [34, 35] combining Canny [2] and machine-learned edges respectively, then later presented RESLAM [36], which is a complete edge-based SLAM pipeline for RGB-D sensors.

Although the efficient tracking brought by DT has been demonstrated in the above methods, the depth information seems to be indispensable. In this work, we proposed a novel VO pipeline that can be used for both monocular cameras and RGB-D sensors.

3 Overview

In this section, we introduce the formulation of camera motion used in our paper and give a brief overview of our edge-based VO framework.

3.1 Notation

Similar to direct methods [5, 6], we maintain a reference keyframe $I_r : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ and an inverse depth map $D_r : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$, where Ω represents the image domain, at each timestamp. More specifically, for each pixel position $\boldsymbol{p} = (x, y)^{\mathrm{T}}$ in I_r with valid inverse depth d_p, we can obtain the corresponding projected position $\boldsymbol{p}'$ given the 3D rigid motion:

$$T = \begin{pmatrix} R & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{pmatrix} \tag{1}$$

$$\boldsymbol{p}' = \Pi\left(R\Pi^{-1}\left(\boldsymbol{p}, d_p\right) + \mathbf{t}\right) \tag{2}$$

This transformation matrix comprises a rotation described by an orthogonal matrix $R \in \mathrm{SO}(3)$ and a translation described by a vector $\boldsymbol{t} \in \mathbb{R}^3$. The projection function Π and its inverse Π^{-1} convert between the 2D pixel position p and the corresponding 3D point $\boldsymbol{P} = (X, Y, Z)^{\mathrm{T}}$, which can be computed as:

$$\boldsymbol{p} = \Pi(\boldsymbol{P}) = \left(\frac{X}{Z} f_x + c_x, \frac{Y}{Z} f_y + c_y\right)^{\mathrm{T}} \tag{3}$$

$$\boldsymbol{P} = \Pi^{-1}(\boldsymbol{p}, d_p) = \left(\frac{x - c_x}{f_x d_p}, \frac{y - c_y}{f_y d_p}, \frac{1}{d_p}\right)^{\mathrm{T}} \tag{4}$$

where f_x and f_y are the focal lengths and c_x and c_y are the image coordinates of the principal point that compose the camera intrinsics c.

Typically, a minimal representation $\xi \in \mathfrak{se}(3)$ is used to represent the rigid motion, which is the Lie algebra associated with the Lie group $T \in \mathrm{SE}(3)$.

3.2 System Architecture

Our system takes monocular image sequences as input, with optional corresponding depth images, and

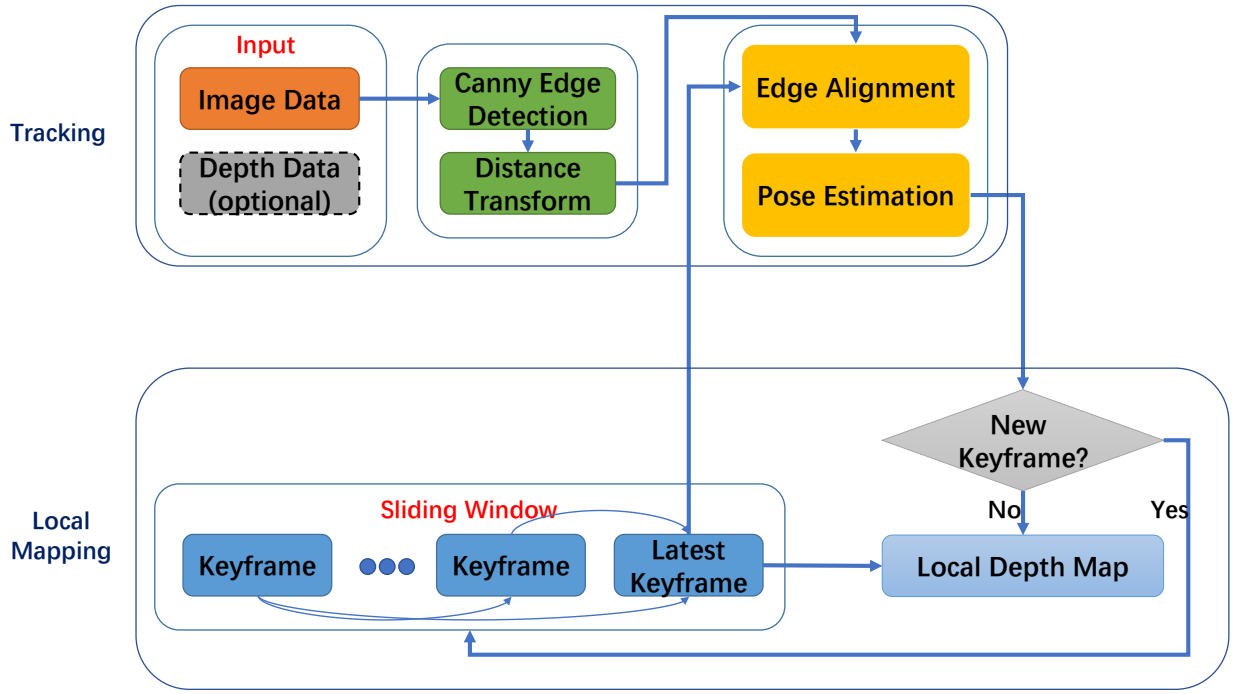


Fig. 1 System overview. Our framework comprises two main threads: tracking and local mapping. The former effectively estimates the camera pose of the current frame using 3D-2D edge alignment. The latter determines whether to add a new keyframe and optimizes it with the sliding window.

estimates the relative camera motion from successive frames based on the extracted edge information. The general structure of the proposed VO system is illustrated in Fig. 1; it comprises two main threads: tracking and local mapping. In the tracking thread, the main goal is to effectively calculate the 6-DoF camera pose for the incoming frames while being robust in challenging environments. To tackle this issue, we extract edge pixels for each frame, which is more robust than extracting point features in low-texture scenes, and leverage 3D-2D edge alignment to estimate the camera motion. In order to avoid sensitivity to illumination changes like direct methods, instead of using photometric residuals, we introduce the DT method to efficiently obtain geometric reprojection errors. In the local mapping thread, we focus on optimizing the depth of each edge pixel and the camera poses of selected keyframes within a sliding window. When the current frame is tracked successfully, the system determines whether the current frame is used to update the local depth map or create a new keyframe by propagating depth information from existing keyframes in the sliding window.

4 Tracking

4.1 Edge Detection

For each incoming frame I_c, we aim to estimate the relative camera motion ξ_{cr} between I_c and the reference frame I_r based on edge information. We first detect edge pixels using Canny edge detection [2], which works well in challenging scenes. As shown in the left two columns of Fig. 2, the Canny algorithm is reliable when dealing with low-texture scenes, since it locally finds the strongest edge pixels by non-maximum suppression of high gradient regions. Its further robustness to illumination changes is shown in the right three columns of Fig. 2.

4.2 Edge-based Reprojection Error

Typically, given the corresponding depth estimate of one edge pixel in the reference frame I_r, we can obtain its projected position (see Eq. 2) in the current frame I_c using the initial motion estimate ξ_{cr}^0.

For computational efficiency, edge pixels are directly used to formulate the error function without extracting descriptors. Unlike direct methods that employ photometric error to track camera pose, we prefer to utilize geometric error to avoid

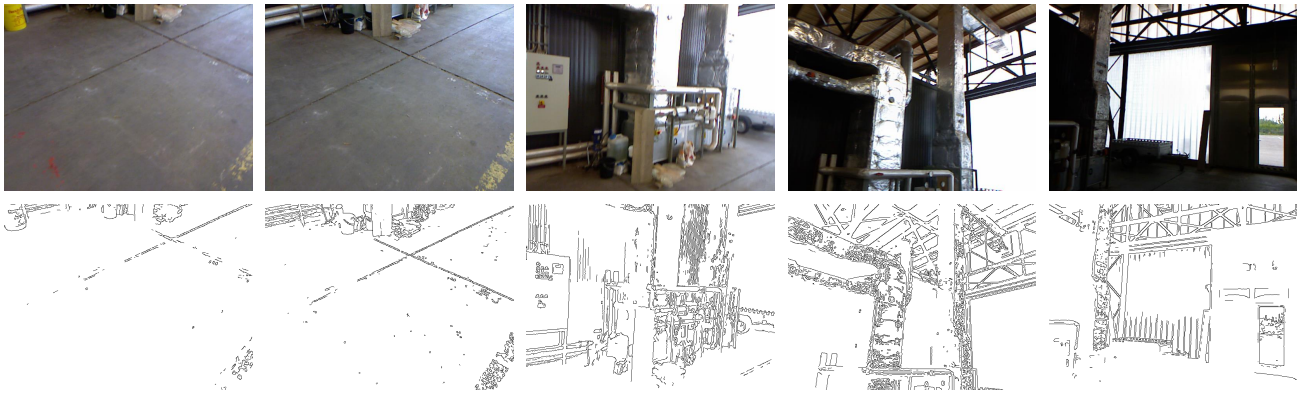


Fig. 2 Edge maps of challenging scenes. Above: original RGB images. Below: edge pixels extracted by Canny edge detection [2]. Two left columns: examples of low-texture floor. Three right columns: examples of illumination change caused by camera motion.

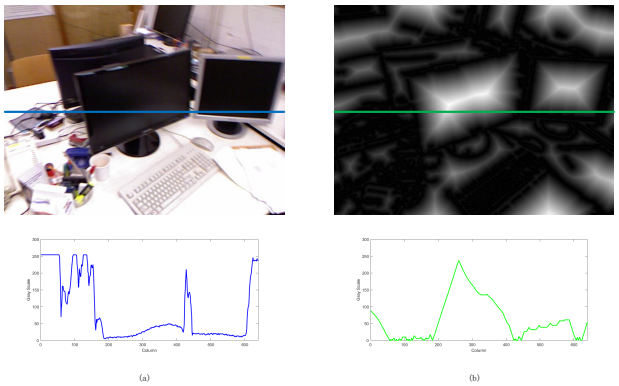


Fig. 3 Comparison of (a) photometric quantity and (b) DT value changes along a scan line. (The DT value is scaled to $[0, 255]$ for visualization.)

sensitivity to illumination changes. However, without the help of features and descriptors providing place recognition capabilities, accurate projective registration of nonparametric edges is quite challenging.

Following the iterative closest point (ICP) algorithm, we utilize the distance between the projected position and the nearest detected edge pixel in the current frame to establish 3D-2D correspondence. To improve matching efficiency, we precompute the DT map [9] $D_c : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}$ for the current frame, which calculates the Euclidean distance to the closest edge at each pixel position. This map converts calculation of the reprojection geometric error into a simple query, and since the edge only needs to be detected once, it can be reused in the subsequent iterative optimization process without the need to repeatedly calculate the distance.

Inspired by [23], there are two main motivations for choosing geometric reprojection error instead of photometric error in this work. On the one hand, the reprojection error can alleviate sensitivity to illumination changes, and on the other hand, the non-differentiability of image intensity may have a negative effect on optimization. As shown in Fig. 3, non-differentiability of the photometric quantity is significant at object boundaries, while the DT value does not change so frequently and sharply.

More specifically, given an edge pixel e_i^r in the reference frame I_r and its inverse depth $d_{e_i^r}$, the reprojection residual is computed as

$$r(e_i^r) = D_c(e_i') \tag{5}$$

where e_i' denotes the reprojection position calculated from Eq. 2.

Since $D_c(e_i') = 0$ if e_i' is an edge pixel in I_c, we can estimate the optimal relative camera motion ξ_{cr}^* by minimizing the total error function:

$$E(\xi_{cr}) = \sum_{e_i^r \in \mathcal{E}_r} \left\| r(e_i^r) \right\|_\gamma \tag{6}$$

$$\xi_{cr}^* = \arg\min_{\xi_{cr}} E(\xi_{cr}) \tag{7}$$

where $\|\cdot\|_\gamma$ is the Huber norm and $\mathcal{E}_r$ is the set of all detected edges in I_r.

During optimization of Eq. 7, it is intuitively beneficial for the entire system if the initial motion estimate ξ_{cr}^0 is more accurate. Thus, we consider four different assumptions for ξ_{cr}^0: (i) constant motion, (ii) no motion, (iii) half- or (iv) double- that of constant motion, and choose the one with the lowest energy according to Eq. 6.

4.3 Weighted Edge Alignment

One of the major challenges of directly using pixels for VO systems is that it is difficult to achieve accurate data association, especially when equipped with a

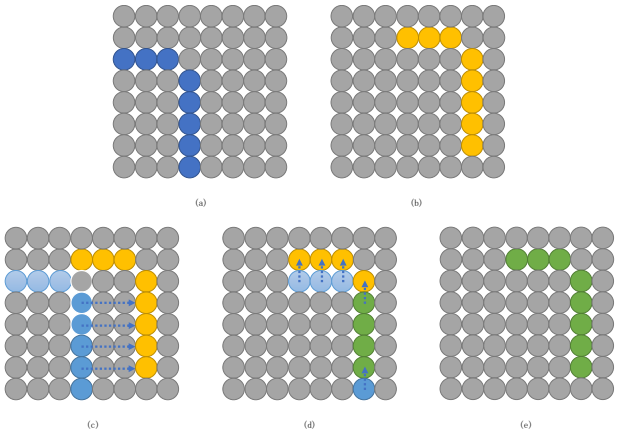


Fig. 4 Weighted edge alignment. (a) and (b) edge pixels to be registered (blue, yellow respectively). (c) projection under a no motion assumption, and a class of edge pixels with a larger number are given a higher weight (darker colors indicate greater weights). (d) after alignment of edge pixels with larger weight, the pixel sets with a smaller number are reassigned larger weights, and matching is performed again. (e) the final matched pixels (green).

monocular camera. It is worth noting that even in direct methods [5, 6], a well-designed point selection strategy is crucial to pick locally recognizable pixels with sufficient intensity gradient for matching. In our work, however, we do not need to follow the same strategy for two reasons. On the one hand, this strategy relies on local image gradient, which is the basis of edge detection; on the other hand, the selected pixels tend to maintain a uniform spatial distribution across the whole image, but the reprojection error based on the DT map (see Eq. 5) is unsuitable for non-edge pixels.

Thus, we propose a weighted edge alignment scheme, based on the observation that when we try to match two images like Fig. 4(a,b), more significant vertical regions can be aligned first, and then the horizontal edge with a smaller number of pixels can be used to fine-tune the alignment: see Fig. 4(c–e). Intuitively, we can achieve this by gradually increasing the weight of non-significant edge regions.

More specifically, we divide edge pixels of the reference frame into 5 categories according to the local spatial layout within the 3×3 neighborhood. The first 4 main types are shown in Fig. 5, and the remaining ones that do not belong to these 4 types are classified as the 5th type. Then we count the number of edge pixels of each type and combine it with the pyramid to construct a dynamic weight that changes with the pyramid level:

$$w_t^k = (\log(N_{\text{sum}}/N_t))^{(l-k)/l} \tag{8}$$

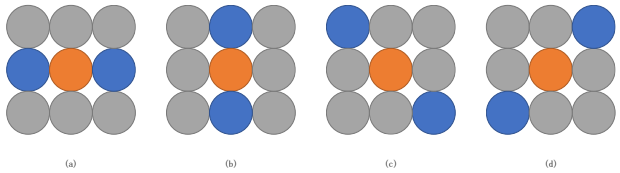


Fig. 5 The 4 main types of edge pixels, based on the spatial relationship between the target edge pixel (orange) and its neighboring edge pixels (blue). Non-edge pixels are in gray.

where $t = 0, \ldots, 4$ denotes the type, N_t is the number of edge pixels of this type, N_{sum} is the total number of edge pixels, l is the highest level of the pyramid and $k = 0, \ldots, l$ is the current level ($k = 0$ corresponds to the original image). When $k = l$, the weight of each edge pixel is 1, and as k decreases, the weight of pixel types with smaller size increase more rapidly. In practice, the weight will be within a certain range.

Now we can apply the weights to the iterative energy function calculation (Eq. 6) in the pyramidal coarse-to-fine tracking scheme:

$$E^k(\xi_{cr}) = \sum_{e_i^r \in \mathcal{E}_r} w_{t_{e_i^r}}^k \left\| r(e_i^r) \right\|_\gamma \tag{9}$$

When at coarser pyramid levels, types with more edge pixels have higher weight and dominate registration, while at finer pyramid levels, types with fewer edge pixels receive more attention and refine the relative motion estimate. This dynamic weighting method also effectively deals with the situation where the minority follow the majority.

We further illustrate the proposed weighted edge alignment in Fig. 6. A three-level (0 to 2) pyramid is used to associate the extracted edge pixels of the reference frame to the edge map of the target frame. After iteration at each pyramid level, sampled edge pixels are projected into the edge map of the current frame using the estimated relative motion. Fig. 6(c-e) shows the gradual registration of edge pixels.

4.4 Initialization for a Monocular Camera

When using a monocular camera, we need to initialize the first frame, which is also the first keyframe, to bootstrap the system. We follow the strategy of LSD-SLAM [6], which uses a random depth map for the first keyframe. Since the sparse edge representation offers a large convergence basin [35], the system will converge to a correct depth configuration quickly with the help of the weighted edge alignment method mentioned in Section 4.3.

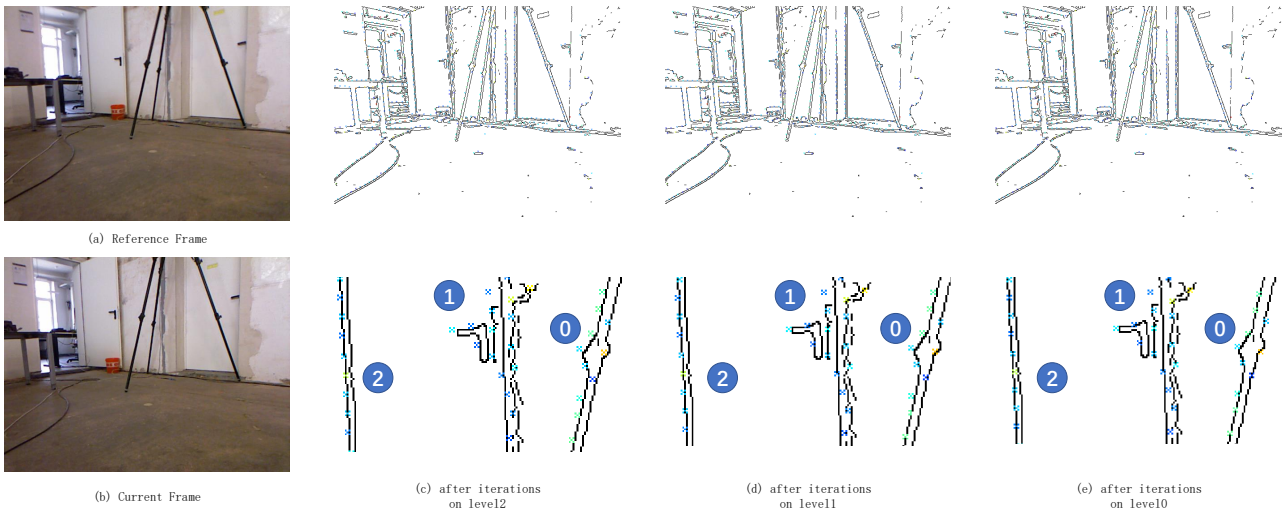


Fig. 6 Projected edge pixels after iteration, at different pyramid levels. (a): reference frame, (b): current frame, (c–e) edge pixels projected from the reference frame to the edge map of the current frame. Above: full maps. Below: close-up. Circled numbers indicate the pixel areas mainly affected by the corresponding pyramid level.

4.5 RGB-D Extension

The proposed VO method can be easily extended to RGB-D sensors. When the input contains depth images, the main difference from the monocular mode is in the tracking thread, and the rest of our system operates independently of the input. Since the depth sensor provides more reliable depth information, the initialization phase can be omitted for the RGB-D sensor. Furthermore, almost all detected edge pixels in the reference frame can be used in the tracking process, without the need to select edge pixels with valid depths as in the monocular mode.

5 Local Mapping

In the local mapping thread, when a new keyframe is created, all edge pixels with valid depth from keyframes within a sliding window are projected into it to generate the depth map. The depths of all these edge pixels, the camera pose and the camera parameters are optimized together. Meanwhile, edge pixels and keyframes far from the current frame are marginalized.

5.1 Keyframe Selection

In optimization-based VO systems, the selection of keyframes is very important to make the tracking robust to camera movement. Typically, keyframe selection ensures that a sufficient number of frames have passed since the last keyframe insertion [28], or a certain relative pose threshold has been met [6]. We also follow the same strategy and combine the edge matching method in Section 4.3 to construct three conditions as follows:

- More than 20 frames have passed since the current keyframe.
- The relative translation distance or rotation angle have met a predefined threshold.
- Among the four main edge pixel types (see Fig. 5), for at least one, the number of changes exceeds a threshold.

When one of the above conditions is met, the current frame is spawned as a new keyframe.

5.2 Sliding Window Optimization

We adopt a sliding window optimization method following [5]. More specifically, we keep a small sliding window $\mathcal{F}$ consisting of several (5–7) active keyframes, and jointly refine all valid edge depths, all camera poses, and even the camera intrinsics c. The residual of one edge pixel e_i of one keyframe $k_m \in \mathcal{F}$ is similar to Eq. 5:

$$r^{mn}(e_i^{k_m}) = D_{k_n}(e_i'(\xi_{wk_m}, \xi_{wk_n}, d_{e_i^{k_m}}, c)) \tag{10}$$

where e_i' is the projected position of $e_i^{k_m}$ in a different keyframe $k_n \in \mathcal{F}$, and ξ_{wk_m} and ξ_{wk_n} are the estimated camera poses relative to the world frame. Then we can minimize the final energy function in the sliding window:

$$E = \sum_{m\in\mathcal{F}} \sum_{e_i\in\mathcal{E}_m} \sum_{n\in\mathcal{F}\setminus\{m\}} \left\| r^{mn}(e_i^{k_m}) \right\|_\gamma \tag{11}$$

The energy can be optimized iteratively using Levenburg-Marquardt (L-M) algorithm:

$$\mathbf{H}\delta\mathbf{x} = -\mathbf{b} \tag{12}$$

Fig. 7 The sharp illumination changes in the *balloon_tracking* sequence of the Bonn RGB-D Dynamic dataset [32].

Tab. 1 Comparison of absolute trajectory RMSE (m) and relative pose RMSE (m) of DSO [5], ORB-SLAM2 [29], DLGO [25], PL-SLAM [33] and our algorithm on the TUM RGB-D dataset (× indicates algorithm failure)

Sequence	ATE					RPE				
	DSO	ORB	DLGO	PL-SLAM	Ours	DSO	ORB	DLGO	PL-SLAM	Ours
fr1/xyz	0.069	0.084	**0.054**	0.13	0.065	0.088	0.106	**0.067**	0.184	0.073
fr2/desk	1.65	1.207	1.33	**0.696**	0.87	1.81	1.266	1.64	**1.078**	1.088
fr2/xyz	**0.062**	0.119	0.065	0.1	0.063	**0.077**	0.109	0.081	0.142	0.079
fr3/office	1.18	1.233	1.168	1.12	**0.64**	1.512	1.577	1.464	1.586	**1.054**
fr3/cabinet	1.08	×	1.05	×	**0.84**	1.56	×	1.57	×	**1.354**
fr3/nstr_ntex_far	0.677	×	**0.504**	×	0.61	0.876	×	**0.74**	×	0.755
fr3/str_ntex_far	0.903	×	0.865	×	**0.69**	1.052	×	0.946	×	**0.80**

where $\mathbf{b} = \mathbf{J}^T\mathbf{W}\mathbf{r}$ consists of the Jacobian $\mathbf{J}$ and the weight matrix $\mathbf{W}$, $\mathbf{H} = \mathbf{J}^T\mathbf{W}\mathbf{J} + \lambda\mathbf{I}$ is the Hessian matrix, and $\delta\mathbf{x}$ is the optimal increment.

Due to the bounded size of the sliding window, old keyframes need to be removed before adding new ones. Following [5], we adopt a marginalization strategy using the Schur complement. Typically, the optimization can be written in a block-matrix way:

$$\begin{bmatrix} \mathbf{H}_{\alpha\alpha} & \mathbf{H}_{\alpha\beta} \\ \mathbf{H}_{\beta\alpha} & \mathbf{H}_{\beta\beta} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}_{\alpha} \\ \delta\mathbf{x}_{\beta} \end{bmatrix} = - \begin{bmatrix} \mathbf{b}_{\alpha} \\ \mathbf{b}_{\beta} \end{bmatrix} \tag{13}$$

where α and β denote the variables that we would like to keep and to marginalize, respectively. We can eliminate the coefficient of $\delta\mathbf{x}_{\beta}$ by multiplying the second line of $\mathbf{H}$ by $\mathbf{H}_{\alpha\beta}\mathbf{H}_{\beta\beta}^{-1}$ and subtracting it from the first. Moreover, following [5], when marginalizing one keyframe, all edges of this keyframe and all the edges that have not been observed in the last two keyframes in the sliding window are marginalized together to retain the sparsity of $\mathbf{H}$.

6 Results and discussion

The proposed VO system has been implemented and tested on a desktop computer with a 3.2GHz Intel i7-8700 CPU and 32GB memory. Since our system permits two input modes, monocular and RGB-D, we have tested them separately in the following subsections. Two main metrics, *absolute trajectory error* (ATE) and *relative pose error* (RPE) [38] are used to respectively evaluate the global consistency of the trajectory and the drift. Meanwhile, the root mean squared error (RMSE) of the translational component is mainly used for comparison.

6.1 Monocular VO

The monocular VO was evaluated using three public datasets, including the TUM RGB-D dataset [38], the Bonn RGB-D Dynamic dataset [32], and the ICL-NUIM dataset [17]. It is worth noting that the last two datasets use the same data format as the TUM RGB-D dataset [38] and provide ground truth trajectories, so it is easy to use the evaluation tools provided by [38] for analysis.

Tab. 1 compares ATE and RPE for DSO [5], ORB-SLAM2 [29], DLGO [25], PL-SLAM [33], and our method. The first two methods are a direct method and a feature-based method, respectively, chosen as representatives of the most commonly used frameworks at present. The next two methods, DLGO [25] and PL-SLAM [33], are extensions of the two benchmark frameworks combining line features. The results for the first three methods are from [25]; we use the same evaluation configuration to obtain results for PL-SLAM and our method.

It can be seen that the proposed method achieves improvements in robustness and accuracy compared to the other methods. In general, our method

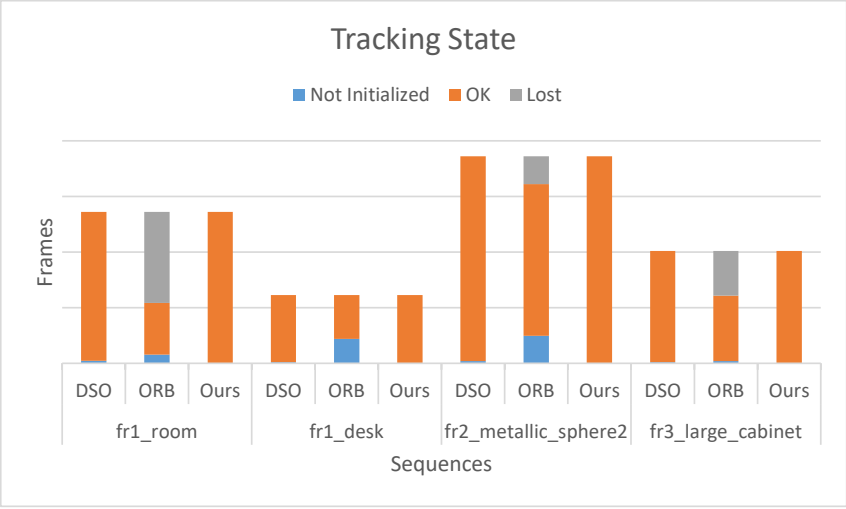


Fig. 8 Proportion of different states in the tracking process. We have tested DSO [5], ORB-SLAM2 [29], and our method on the 4 sequences of the TUM RGB-D dataset, and recorded the tracking status of each frame, including not initialized, tracking ok, and tracking lost.

and the two extensions perform better than the two benchmark frameworks, reflecting the effectiveness of introducing edge information to deal with low-texture environments. ORB-SLAM2 [29] and its extension fail for the last two low-texture scenes, while our method and DLGO achieve the best results respectively, shows that the use of additional image information can help to improve tracking robustness in low-texture scenes.

Since VO systems need to pay more attention to tracking, we performed a tracking status comparison on 4 sequences of the TUM RGB-D dataset (see Fig. 8). The tracking status of each frame was recorded when evaluating DSO [5], ORB-SLAM2 [29], and our method. It can be seen that ORB-SLAM2 [29] has careful initialization, but it omits many frames and has the most tracking loss. DSO [5] needs to select two frames with sufficient translational camera movement for initialization, so a few frames will be ignored at this stage. Our method directly initializes a random depth map for the first frame, and subsequent frames can be tracked successfully.

The Bonn RGB-D Dynamic dataset [32] contains 24 highly dynamic sequences in which people perform different tasks, such as manipulating boxes or playing with balloons, and 2 static sequences. We chose 1 static and 6 dynamic sequences to evaluate the robustness to dynamic scenarios. There are many sudden changes in brightness of the scenes caused by camera movement in this dataset (see Fig. 7), which have a great impact on the direct method.

Tab. 2 Comparison of absolute trajectory RMSE (m) of ORB-SLAM2 [29], DSO [5], PL-SLAM [33] and our algorithm on BONN-Dynamic dataset (× indicates algorithm failure)

Sequence	ORB	DSO	PL-SLAM	Ours
static	2.12	×	×	**1.91**
kidnapping_box	0.46	0.35	0.45	**0.29**
kidnapping_box2	0.36	0.35	**0.34**	0.39
balloon_tracking	0.31	×	0.27	**0.25**
balloon_tracking2	0.28	0.37	0.29	**0.27**
balloon	×	**0.18**	0.22	0.20
balloon2	×	×	×	**0.26**

Tab. 2 compares ATE for ORB-SLAM2 [29], DSO [5], PL-SLAM [33], and our method on the 7 sequences. It is worth noting that the *static* sequence contains 10,916 images, which leads to a relatively large cumulative error. From Tab. 2, it can be concluded that our method is more robust than the other three methods in dealing with dynamic scenes and illumination changes.

The ICL-NUIM dataset is captured in synthetic indoor environments. Tab. 3 compares our method with ORB-SLAM2 [29], DSO [5], and PL-SLAM [33] in terms of ATE on the 8 office sequences, in which half have have added simulated noise. It can be seen that PL-SLAM is confused, typically because it cannot detect sufficient point and line features in untextured areas such as walls, causing it to fail. The robustness of DSO and our algorithm are better than the other two

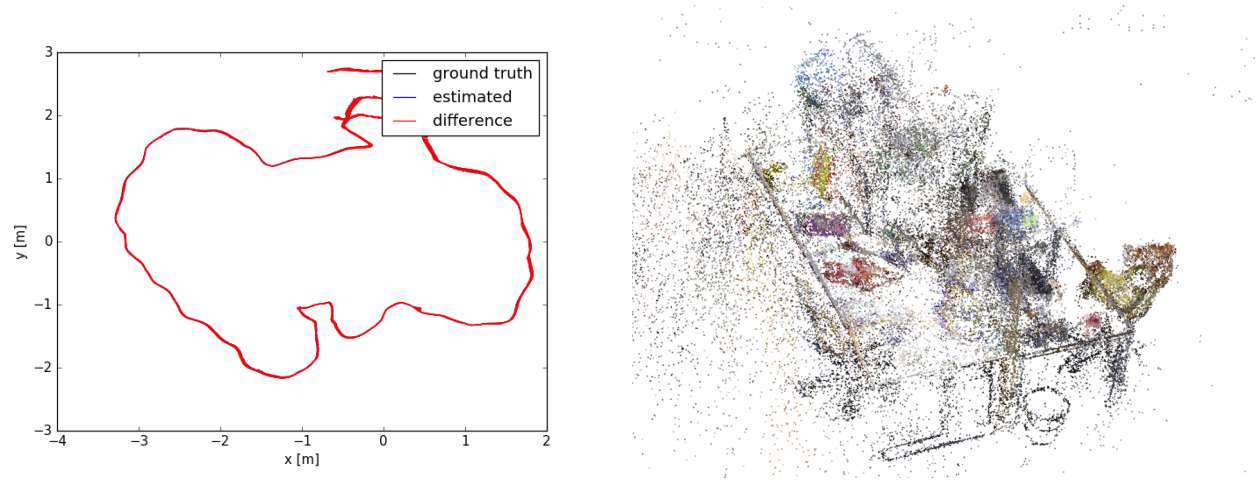


Fig. 9 Ego-motion estimation and mapping results for our proposed algorithm on the *fr3/long_office* sequence from the TUM RGB-D dataset.

Tab. 3 Comparison of absolute trajectory RMSE (m) of ORB-SLAM2 [29], DSO [5], PL-SLAM [33] and our algorithm on the ICL-NUIM dataset (× indicates algorithm failure, and * indicates tracking loss occurs but does not cause failure)

Sequence	ORB	DSO	PL-SLAM	Ours
of_kt 0	0.49*	**0.26**	×	0.49
of_kt 0 (with noise)	0.61*	**0.30**	×	0.45
of_kt 1	0.83*	0.69	×	**0.64**
of_kt 1 (with noise)	0.81*	0.68	×	**0.63**
of_kt 2	**0.73**	0.79	×	0.80
of_kt 2 (with noise)	**0.75**	0.78	×	0.80
of_kt 3	0.57*	0.62	×	**0.55**
of_kt 3 (with noise)	0.59*	0.61	×	**0.53**

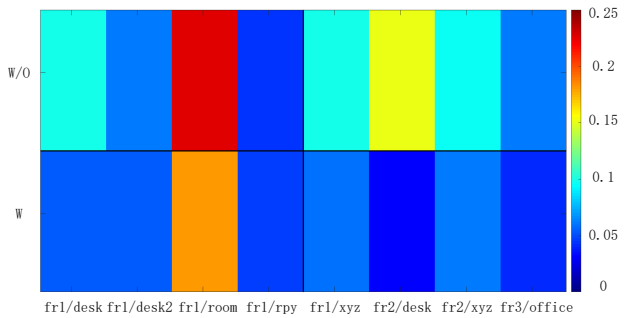


Fig. 10 Ablation study on the TUM RGB-D dataset. Each bar corresponds to the (color-coded) absolute trajectory error over the full sequence. We run each sequence (horizontal axis) without ('w/o') and with ('w') the dynamic weighted method as mentioned in Section 4.3.

methods, and our method is competitive in terms of accuracy. Note that since ORB-SLAM2 is a complete SLAM system with re-localization and loop-closure detection, tracking loss will not directly lead to system failure, while DSO and our method directly trigger system termination if tracking is lost.

6.2 RGB-D VO

The VO framework proposed in this paper has good generalisability and can be integrated with depth sensors. We compare our algorithm with state-of-the-art edge-based RGB-D VO systems, including REVO [35] and CannyVO [45]. The former uses deep learning features to favor object boundaries and omit weak edges, while the latter adopts two distance transforms, approximate nearest neighbor field (ANNF) and oriented nearest neighbor field (ONNF), to improve registration in terms of efficiency and accuracy. Quantitative evaluation results are shown in Tab. 4. It can be seen that our method is more accurate. As shown in Fig. 9, our method can estimate a consistent camera trajectory while simultaneously recovering a semi-dense point cloud for the scene.

Fig. 10 gives results of a leave-one-out ablation study using the TUM RGB-D dataset to analyse the accuracy improvements due to the proposed weighted edge alignment. We selected 8 sequences and tested each with and without the weight (see Eq. 8) in edge alignment. We can see that the use of weighted edge alignment effectively improves accuracy.

6.3 Canny Parameters

In order to compensate for the differences in scenes, we need to adapt the parameters, mainly the two thresholds $(t_{\text{high}}, t_{\text{low}})$, used by the Canny detector [2].

Tab. 4 Comparison of absolute trajectory RMSE (m) of REVO [35], CannyVO [45] and our method on the TUM RGB-D dataset

Sequence	REVO [35]	CannyVO (ANNF) [45]	CannyVO (ONEF) [45]	Ours
fr1/xyz	0.068	0.137	0.043	**0.033**
fr1/rpy	0.049	0.205	0.047	**0.042**
fr1/desk	0.061	0.212	**0.044**	0.054
fr1/desk2	0.082	0.381	0.187	**0.053**
fr1/room	0.298	0.621	0.242	**0.186**
fr2/desk	0.089	0.039	0.037	**0.032**
fr3/office	0.117	0.09	0.085	**0.042**

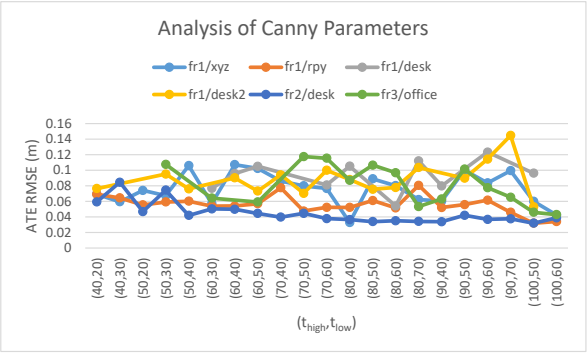


Fig. 11 Influence of different Canny parameters. We selected 6 sequences from the TUM RGB-D dataset and recorded the corresponding absolute trajectory RMSE (m) of our system in RGB-D mode using 21 different Canny parameter combinations. Results greater than 0.15 are absent.

Fig. 11 records the ATE results corresponding to different threshold selections on the 6 sequences of the TUM RGB-D dataset. It can be seen that under different scene conditions, the selection of Canny parameters has a significant impact on the results. We also give a qualitative comparison of 3D point clouds for the *fr1/desk2* sequence using two different threshold combinations, $(90, 70)$ and $(100, 50)$; the influence of the Canny parameters on the generated map is obvious. If the parameters could be adjusted adaptively according to the scene, it would bring a great improvement to the proposed edge-based system, which we will considered in our future work.

6.4 Speed

Although there are tens of thousands of edge pixels that need to be matched in each incoming frame, with the introduction of DT, our method can still achieve excellent real-time performance. Tab. 5 shows the total execution time of our method for different sequences. Note that the processing efficiency is unchanged for monocular or RGB-D sensors. Tab. 6 compares our method with state-of-the-art edge-based and line-based VO/SLAM systems in terms of the average runtime for each module in the tracking thread. It can be concluded that our method is highly efficient and has better real-time performance than state-of-the-art methods that use edge information.

Tab. 5 Execution time of our full pipeline for different sequences ('rgbd' means the system is running in RGB-D mode)

Sequence	#Images	Times (ms)	FPS
fr2/rpy	3290	45403	72.46
fr2/xyz (rgbd)	3615	49508	73.02
lr_kt1	966	12453	77.57
of_kt0	1508	17369	86.82
balloon_tracking	590	7714	76.48

7 Conclusions

Targeting low-texture scenes which challenge existing visual odometry methods, we present a novel edge-based visual odometry algorithm that estimates the relative camera pose by minimizing the geometric reprojection error of extracted edge pixels. We experimentally demonstrate that using more image information, like direct methods, can help to deal with low-texture situations. Meanwhile, the geometric reprojection residual also makes the proposed VO system insensitive to illumination changes. Combined with a novel weighted edge alignment method, the introduced DT map further improves the accuracy and efficiency of camera tracking. Moreover, the proposed VO framework has good generality making it suitable for both monocular and RGB-D cameras. A large number of experiments conducted on public datasets show that the proposed method is comparable to state-of-the-art SLAM systems in terms of tracking accuracy, and is much faster as well as more robust in challenging scenarios.

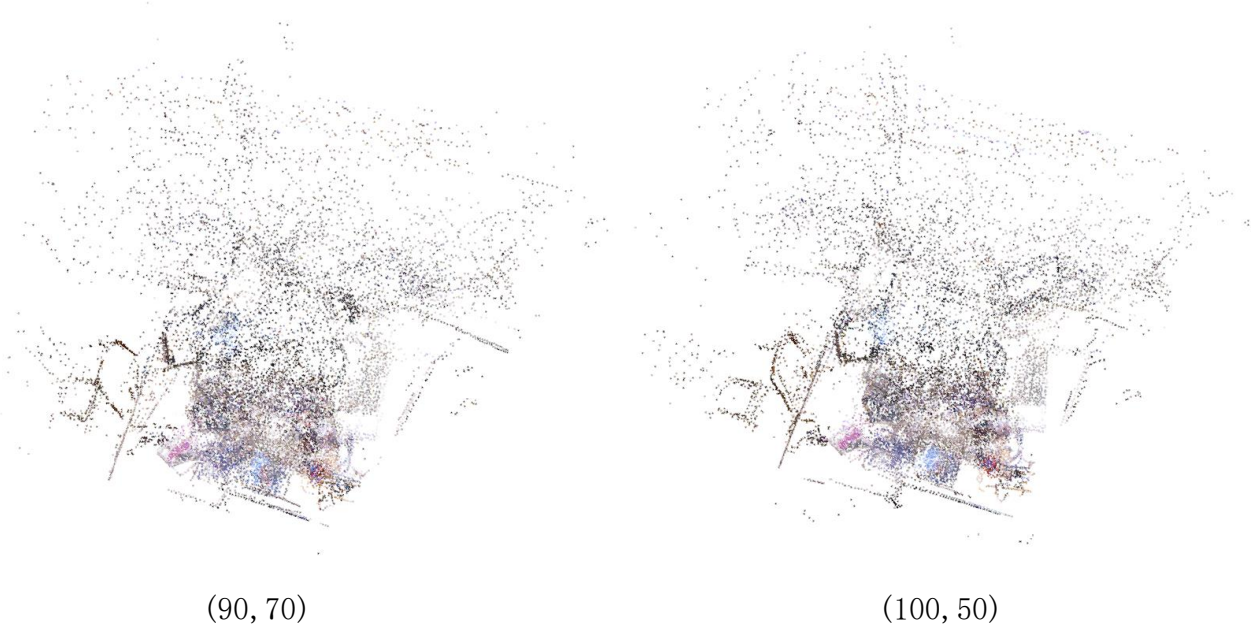


Fig. 12 Qualitative comparison of our recovered 3D maps on the *fr1/desk2* sequence of the TUM RGB-D dataset using different Canny parameters.

Tab. 6 Average execution time (ms) used by each part for per frame tracking on the TUM dataset

Operation	Yang [41]	REVO [35]	PL-SLAM [33]	Ours
Feature extraction	28.19	7.64	31.32	2.86
Initial pose estimation	4.52	2.36	7.16	1.3
Tracking	19.23	8.53	12.58	3.46
Total	51.94	18.53	51.06	7.62

Acknowledgements

This work was supported by the National Key R&D Program of China under Grant 2018YFB2100601, and the Natural Science Foundation of China under Grant (61872024, 61702482).

References

[1] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart. Topomap: Topological mapping and navigation based on visual slam maps. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3818–3825, 2018.

[2] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[3] D. Caruso, J. Engel, and D. Cremers. Large-scale direct slam for omnidirectional cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 141–148, 2015.

[4] Z.-J. Du, S.-S. Huang, T.-J. Mu, Q. Zhao, R. Martin, and K. Xu. Accurate dynamic slam using crf-based long-term consistency. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2020.

[5] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, 2018.

[6] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision (ECCV)*, September 2014.

[7] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *2013 IEEE International Conference on Computer Vision*, pages 1449–1456, 2013.

[8] J. Engel, J. Stückler, and D. Cremers. Large-scale

direct slam with stereo cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1935–1942, 2015.

[9] P. F. Felzenszwalb and D. P. Huttenlocher. Distance transforms of sampled functions. *Theory Comput.*, 8:415–428, 2012.

[10] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza. Svo: Semidirect visual odometry for monocular and multicamera systems. *IEEE Transactions on Robotics*, 33(2):249–265, 2017.

[11] X. Gao, R. Wang, N. Demmel, and D. Cremers. Ldso: Direct sparse odometry with loop closure. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2198–2204, 2018.

[12] A. P. Gee and W. Mayol-Cuevas. Real-time model-based slam using line segments. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, P. Remagnino, A. Nefian, G. Meenakshisundaram, V. Pascucci, J. Zara, J. Molineros, H. Theisel, and T. Malzbender, editors, *Advances in Visual Computing*, pages 354–363, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[13] 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)*, pages 4211–4216, 2016.

[14] R. Gomez-Ojeda and J. Gonzalez-Jimenez. Robust stereo visual odometry through a probabilistic combination of points and line segments. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2521–2526, 2016.

[15] R. Gomez-Ojeda, F.-A. Moreno, D. Zuñiga-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*, 35(3):734–746, 2019.

[16] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: a Line Segment Detector. *Image Processing On Line*, 2:35–55, 2012.

[17] 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)*, pages 1524–1531, 2014.

[18] K. Hirose and H. Saito. Fast line description for line-based slam. In *Proceedings of the British Machine Vision Conference*, pages 83.1–83.11. BMVA Press, 2012.

[19] M. Hsiao, E. Westman, G. Zhang, and M. Kaess. Keyframe-based dense planar slam. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5110–5117, 2017.

[20] J. Huang, S. Yang, T.-J. Mu, and S.-M. Hu. Clustervo: Clustering moving instances and estimating visual odometry for self and surroundings. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2165–2174, 2020.

[21] S.-S. Huang, Z.-Y. Ma, T.-J. Mu, H. Fu, and S.-M. Hu. Lidar-monocular visual odometry using point and line features. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1091–1097, 2020.

[22] 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*, pages 225–234, 2007.

[23] M. Kuse and S. Shen. Robust camera motion estimation using direct edge alignment and sub-gradient method. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 573–579, 2016.

[24] S. H. Lee and J. Civera. Loosely-coupled semi-direct monocular slam. *IEEE Robotics and Automation Letters*, 4(2):399–406, 2019.

[25] S.-J. Li, B. Ren, Y. Liu, M.-M. Cheng, D. Frost, and V. A. Prisacariu. Direct line guidance odometry. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5137–5143, 2018.

[26] F. F. Ling, C. Elvezio, J. Bullock, S. Henderson, and S. Feiner. A hybrid rtk gnss and slam outdoor augmented reality system. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 1044–1045, 2019.

[27] H. Liu, G. Zhang, and H. Bao. Robust keyframe-based monocular slam for augmented reality. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 1–10, 2016.

[28] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[29] 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*, 33(5):1255–1262, Oct 2017.

[30] T. Nescher, M. Zank, and A. Kunz. Simultaneous mapping and redirected walking for ad hoc free walking in virtual environments. In *2016 IEEE Virtual Reality (VR)*, pages 239–240, 2016.

[31] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *2011 International Conference on Computer Vision*, pages 2320–2327, Nov 2011.

[32] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss. Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7855–7862, 2019.

[33] 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)*, pages 4503–4508, 2017.

[34] F. Schenk and F. Fraundorfer. Combining edge images and depth maps for robust visual odometry. In G. B. Tae-Kyun Kim, Stefanos Zafeiriou and K. Mikolajczyk, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 149.1–149.12. BMVA Press, September 2017.

[35] F. Schenk and F. Fraundorfer. Robust edge-based visual odometry using machine-learned edges. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1297–1304, 2017.

[36] F. Schenk and F. Fraundorfer. Reslam: A real-time robust edge-based slam system. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 154–160, 2019.

[37] P. Smith, I. Reid, and A. J. Davison. Real-time monocular slam with straight lines. In *Proceedings of the British Machine Vision Conference*, pages 3.1–3.10. BMVA Press, 2006. doi:10.5244/C.20.3.

[38] 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*, pages 573–580, 2012.

[39] R. Wang, M. Schwörer, and D. Cremers. Stereo dso: Large-scale direct sparse visual odometry with stereo cameras. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3923–3931, 2017.

[40] W. Xin, D. Wei, Z. Mingcai, L. Renju, and Z. Hongbin. Edge enhanced direct visual odometry. In E. R. H. Richard C. Wilson and W. A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 35.1–35.11. BMVA Press, September 2016.

[41] S. Yang and S. Scherer. Direct monocular odometry using points and lines. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3871–3877, 2017.

[42] J. Zhang, G. Zeng, and H. Zha. Structure-aware slam with planes in man-made environment. In J. Yang, Q. Hu, M.-M. Cheng, L. Wang, Q. Liu, X. Bai, and D. Meng, editors, *Computer Vision*, pages 477–489, Singapore, 2017. Springer Singapore.

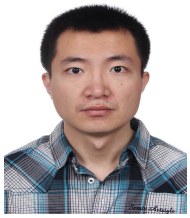
[43] L. Zhang and R. Koch. Hand-held monocular slam based on line segments. In *2011 Irish Machine Vision and Image Processing Conference*, pages 7–14, 2011.

[44] 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*, 24(7):794–805, 2013.

[45] Y. Zhou, H. Li, and L. Kneip. Canny-vo: Visual odometry with rgb-d cameras based on geometric 3-d–2-d edge alignment. *IEEE Transactions on Robotics*, 35(1):184–199, 2019.

[46] X. Zuo, X. Xie, Y. Liu, and G. Huang. Robust visual slam with point and line features. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1775–1782, 2017.

Feihu Yan is a Ph.D. candidate in computer science at the State Key Lab of Virtual Reality Technology and Systems, Beihang University, Beijing, China. He received his M.S. degree in computer science and technology from Shandong University in 2012. His research interests include 3D reconstruction and visual SLAM.



Zhaoxin Li received a Ph.D. degree in computer application technology from Harbin Institute of Technology, China, in 2016. From September 2018 to March 2019, he worked as a Postdoctoral Fellow in the Department of Computing, Hong Kong Polytechnic University. He is currently with the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include 3D computer vision and 3D data processing.

Zhong Zhou is a professor, Ph.D. adviser, at the State Key Lab of Virtual Reality Technology and Systems, Beihang University. He received his B.S. degree from Nanjing University and Ph.D. degree from Beihang University in 1999 and 2005 respectively. His main research interests include augmented virtual environments, natural phenomena simulation, distributed virtual environments, and Internet-based VR technologies. He is member of IEEE, ACM, and CCF.