Vision-Based SLAM for Uavs in Dynamic Environments

Pingrui Huang *

School of Electrical Automation and Information Engineering, Tianjin University, Tianjin, China, 300072

* Corresponding Author Email: hpr3474156478@163.com

Abstract. Visual SLAM (Simultaneous Localization And Mapping) is a technique mainly used for robot navigation and positioning. It uses vision sensors to map the environment and estimate its own location. In dynamic environment, they are inevitably affected by dynamic objects, which leads to the decrease of accuracy and stability of the system. In this paper, the visual odometer in SLAM system is studied, and the semi-direct visual odometer (SVO) algorithm is improved on the basis of ORB-SLAM framework. The basic principle of the algorithm is to treat pixel matching between adjacent image frames as an optimization problem, and use direct registration method to obtain camera pose for sparse feature blocks, avoiding a lot of feature extraction and matching. When a key frame appears, ORB (Oriented FAST and Rotated BRIEF) features of the image are extracted, and feature matching is used to track local maps to obtain more mapping relationships. Moreover, camera pose and three-dimensional structure of the scene are estimated by minimizing reprojection errors, thus improving positioning accuracy and map construction quality. Finally, experiments on several public data sets show that the improved semi-direct visual odometer scheme in this paper still has good robustness to moving objects in dynamic scenes. While maintaining high accuracy, it obtains better real-time performance by tracking visual feature points.

Keywords: Visual simultaneous localization and mapping, Semi-direct, Visual odometer.

1. Introduction

In recent years, visual SLAM has made remarkable progress in both theory and practice, but there are still many technical challenges [1]. Traditional vision SLAM systems are usually based on the assumption of static environment, treating the moving objects in the real environment as measurement noise and ignoring their effects on the system [2]. In fact, dynamic objects in the scene can interfere with the system and have a huge impact on the accuracy and robustness of the system [3]. The impact of dynamic environment on visual SLAM is mainly reflected in two aspects: first, the motion of dynamic objects can produce mismatching; second, the presence of dynamic objects can change the structure of the scene and lead to changes in the map. In SLAM, feature points are usually used for matching, and the motion of dynamic objects can cause the position of feature points to change, thus generating mismatching [4]. In addition, the presence of dynamic objects also changes the structure of the scene, leading to changes in the map, which can make it difficult for the SLAM algorithm to track the movement of the camera, thus affecting the accuracy of localization [5].

This paper describes the development and research status of visual SLAM and adopts a semi-direct method visual SLAM system for dynamic scenes to cope with problems such as dynamic object interference in complex environments by combining the feature point method and the direct method in response to the poor performance of visual SLAM systems for dynamic environments today [6]. The so-called semi-direct method refers to the acquisition of camera bit pose by direct matching of feature point image blocks in the image, instead of using direct matching for the whole image (filtering the points to be matched according to the gray gradient size) as in the direct matching method [7]. The direct matching method for the whole image is commonly used for RGBD sensors because RGBD sensors can acquire the depth of the whole image. If the direct extraction of feature point image blocks fails or is ineffective, the feature point method can also be combined with the extracted ORB for pose estimation, which in turn optimizes the pose using minimized reprojection error, improving the robustness and accuracy of the system. This exploits the advantages of the direct method for fast tracking and localization of the poses, while considering the advantages of the feature

point method in terms of accuracy and global optimization. The semi-direct method scheme in this paper avoids the extensive feature extraction and matching of the direct method and has better real-time performance while maintaining high accuracy [8].

2. Semi-direct method of visual mileage calculation in dynamic environment

In order to improve the applicability to dynamic scenes, this paper adopts the semi-direct method visual odometry, which only extracts ORB features in key frames for optimization and map building; in non-key frames, the direct method is used to quickly track the poses.

2.1. Visual odometer frame

Figure 1 illustrates the framework of the semi-direct method visual odometry. In monocular vision SLAM, the images need to be initialized first. Two appropriate images are selected for initialization, and the data association between the two frames is established by feature extraction and matching. The motion of the camera is estimated using a 2D-2D approach using a pairwise polar geometry. Next, triangulation is performed to obtain depth information of the initial map points. Subsequently, Bundle Adjustment (BAU) is performed to optimize the initial motion estimation and the initial map.

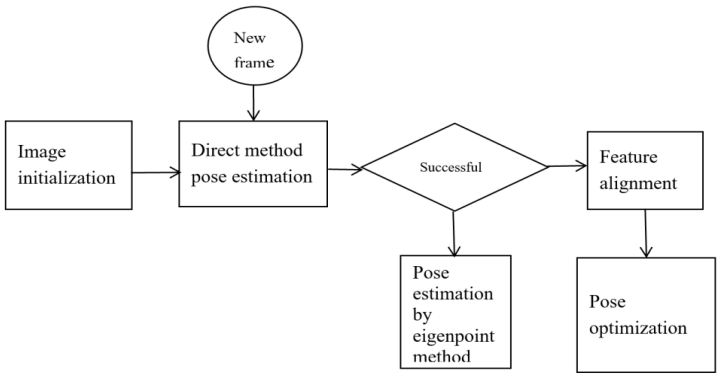

Figure 1. Semi-direct method visual odometer frame

2.2. Position estimation

2.2.1 Direct method scheme

In the semi-direct vision odometry, the direct method is first used to track and optimize the camera's poses when the previous frame is successfully tracked. The method first uses a uniform motion model to estimate the camera motion. Then, the map points that were successfully tracked in the previous frame are projected onto the current frame. As shown in Figure 2, the preliminary positional estimation is performed by minimizing the photometric error between the image blocks corresponding to the same spatial 3D points in the two frames before and after. In this paper, we use the residual template in DSO (Figure 3) to calculate the SSE (Sum of Square Error) error of the eight-point neighborhood of a point as the photometric error of the corresponding image block of that point.

$$T_{k,k-1} = \arg\min_{T_{k,k-1}} \frac{1}{2} \sum_{u_i \in \Re} \left\| \delta I\left(T_{k,k-1}, u_i\right) \right\|^2 \tag{1}$$

where $T_{k,k-1}$ is the bitwise transformation from the previous moment to the current moment, $\Re$ is the image region, u_i is the pixel position, and δI is the intensity residual. δI is defined as follows:

$$\delta I\left(T,u\right) = I_k\left(\pi\left(T \cdot P_{K-1}\right)\right) - I_{K-1}\left(u\right) \tag{2}$$

Where I represents the image intensity function, π represents the camera projection function, and P_{K-1} represents the spatial 3D point tracked in the previous frame.

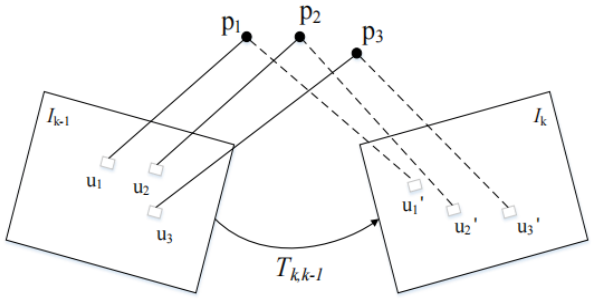

Figure 2. Schematic diagram of direct method posture estimation

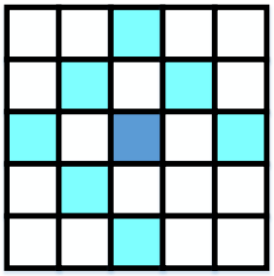

Figure 3. Residual Template

The obtained rough poses are not sufficiently accurate, so the algorithm searches the reference keyframe for co-viewing keyframes and projects the relevant map points in the local map onto the current frame using the current rough pose estimate. Even if the ORB features of the image are not extracted, the 2D point location estimate corresponding to the visible map points in the current frame can still be refined by minimizing the photometric error of the image block. Specifically, the process can be expressed as follows:

$$u_i' = \arg\min_{u_i'} \frac{1}{2} \sum_{j \in N} \left\| I_k(u_i')_j - I_r\left(\pi\left(T_{r,w} \cdot P_{wi}\right)_j\right) \right\|^2 \tag{3}$$

Where, u_i' represents the position of 2D point, I represents the pixel intensity function, I_k represents the current frame, I_r represents the co-view keyframe of the reference keyframe, π is the camera projection function, $T_{r,w}$ the bit pose estimation of the co-view keyframe in the world coordinate system, P_{wi} represents the spatial 3D point, and N is the pattern as shown in Figure 4.

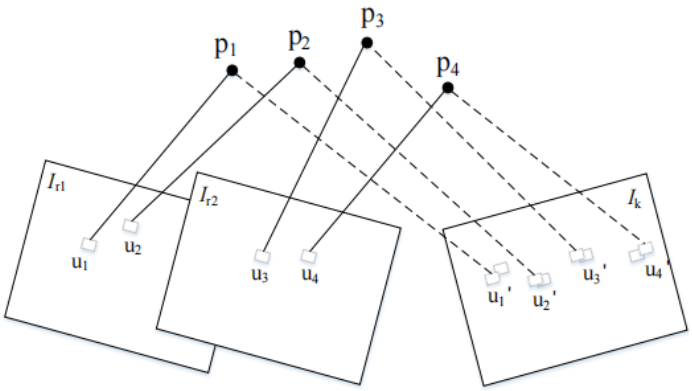

Figure 4. Feature alignment schematic

2.2.2 Feature point method scheme

When the number of 2D matched point pairs obtained from direct method pose tracking is insufficient, the direct method pose tracking is considered to fail and will turn to the feature point method to estimate the camera motion. In this paper, image ORB descriptors are used as image

features, which include two parts: key points and descriptors. The key point uses a modified FAST corner point which has the advantage of fast computation but lacks directionality. The ORB improves the FAST corner point and gives it directionality. The FAST corner point is extracted in the way shown in Figure 5. For a pixel to be a feature point, there must exist N consecutive feature pixels on a circle of radius 3 around it (these pixels differ from the pixel by a gray value above a set threshold). ORB also adds scale and rotation invariance to FAST corner points. The scale invariance is achieved by verifying the corner points at each layer during the construction of the image pyramid. the orientation of the FAST corner points is calculated by the grayscale center-of-mass method. The specific operations are as follows:

For an image block First, the moments of the image block are defined as follows:

$$m_{pq} = \sum_{x,y \in B} x^p y^q I(x,y), p,q \in \{0,1\} \tag{4}$$

To determine the center of mass of the image block:

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{5}$$

Define the direction of the eigenpoint as the direction of the vector from the geometric center O of the image block to the center of mass C:

$$O = \arctan\left(\frac{m_{01}}{m_{10}} \right) \tag{6}$$

The ORB feature descriptors form a high-dimensional vector between pixels near key points, consisting of 0 and 1. To characterize the similarity between features and to select a matching method, the BRIEF descriptor uses Hamming distance as the most commonly used distance metric. When dealing with large-scale data, the Fast Approximate Nearest Neighbor (FLANN, Fast Approximate Nearest Neighbor Search Library) algorithm is often used in order to improve the matching speed.

When the direct normal pose tracking fails, the ORB features of the current frame are extracted. The data association between the current frame and the reference keyframe is established by feature matching. Since the ORB features in the reference keyframe have been associated with the 3D map points, the poses of the current frame can be solved using PnP. If the matching with the reference keyframe fails, the global repositioning is triggered [9]. The best match is found by matching features with all keyframes and solving for the poses using PnP. Once the initial bit pose estimate is obtained, more feature matching relationships can be obtained by projecting the local map to the current frame, and bit pose optimization is performed by minimizing the reprojection error.

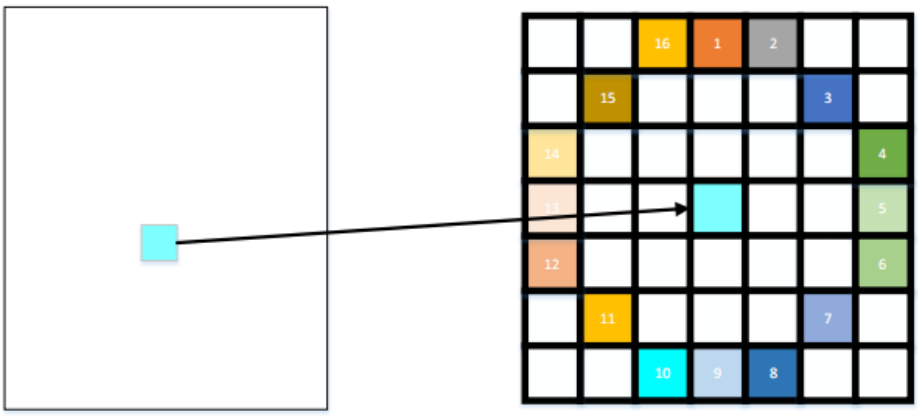

Figure 5. FAST corner point

3. Experiment and Analysis

3.1. Experimental design

In this paper, the advantages and disadvantages of feature point method and direct method as well as the principle of semi-direct method are analyzed. In order to improve the applicability of visual SLAM in dynamic environment, this chapter will use the semi-direct method SLAM to conduct experiments on the TUM public dataset. The experiments include static and dynamic experiments, and the experimental results are compared with the ORB-SALM experimental results [10].

3.1.1 Experimental environment

All experiments in this paper were run on an Intel(R) Core (TM) i5-9300H CPU running at 2.40GHz with 8G of RAM, with Ubuntu 18.04 as the operating system and ROS melodic as the environment configuration.

3.1.2 Introduction to datasets

The Technical University of Munich (TUM) is a world-renowned research university in Germany for engineering and natural sciences.TUM datasets refer to datasets created or maintained by researchers or students at the university, which can be used in various research areas and applications.The TUM RGB-D dataset consists of 39 sequences recorded in different indoor scenes using Microsoft Kinect sensors, including Testing and Debugging, Handheld SLAM, Robot SLAM, Structure vs Texture, and Dynamic Objects. Texture, Dynamic Objects, 3D Object Reconstruction, Validation Files, Calibration Files, and several task-specific datasets, each of which contains multiple data and can be used for multiple Performance testing of multiple tasks. Figure 6 shows a typical image of the TUM dataset.

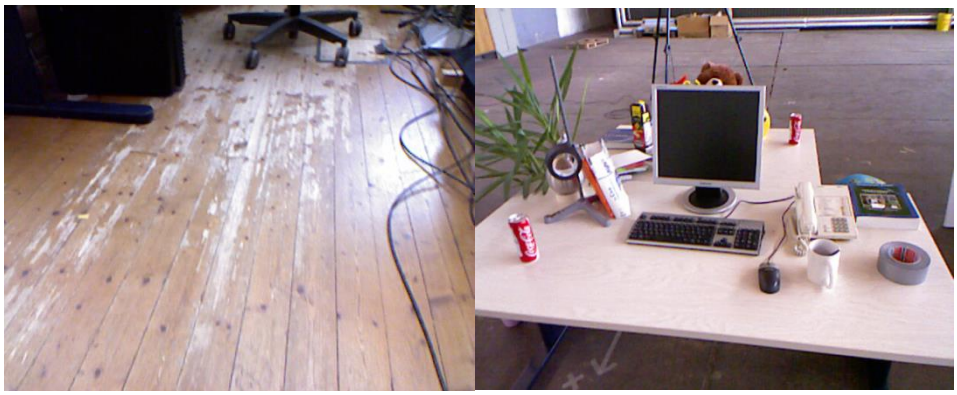

Figure 6. Typical images of the TUM dataset

3.2. Experimental Results and Analysis

3.2.1 EVO Evaluation Tool

EVO (Evolutionary Algorithm Evaluation Toolkit) is a trajectory evaluation tool for visual odometry and SLAM problems, with the core function of plotting the camera The core function is to plot the trajectory of the camera, or to evaluate the error between the trajectory and the true value. Multiple data sets of trajectory formats (TUM, KITTI, EuRoC MAV, ROS bag) are supported, as well as interconversions between these data formats. Relative Pose Error (RPE) is usually used to evaluate the robustness of the system.

Before introduction, we define the equation labeled, algorithm estimated camera pose: $P_1, P_2, \cdots, P_n \in SE(3)$, real camera pose: $Q_1, Q_2, \cdots, Q_n \in SE(3)$, where the subscript represents time t (or frame). Here we assume that the estimated pose and the true pose are aligned at each frame and the total number of frames is the same.

The relative pose error RPE describes the accuracy of the pose difference (compared to the true pose) between two frames separated by a fixed time difference Δ, which is equivalent to the error of a direct odometer measurement. Therefore, the RPE of the ith frame is defined as follows:

$$E_i = \left(Q_i^{-1} Q_{i+\Delta}\right)^{-1} \left(P_i^{-1} P_{i+\Delta}\right) \tag{7}$$

Knowing the total number and the interval, we can obtain an RPE, and then we can use the root mean square error RMSE to count this error and obtain an overall value.

$$RMSE\left(E_{1:n}, \Delta\right) = \left(\frac{1}{m}\sum_{i=1}^{m}\left\|trans\left(E_i\right)\right\|^2\right)^{\frac{1}{2}} \tag{8}$$

where $trans\left(E_i\right)$ stands for taking the translational part of the relative positional error. It should be noted that the RMSE contains two parts of the error, the rotational error and the translational error, and it is usually sufficient to use the translational error for evaluation, but if needed, the error of the rotational angle can be counted using the same method. This method is used in the experiments of this paper to calculate the root mean square error of the translational component of the positional drift in m/s.

3.2.2 Static scene experiments

In this section, we use fr1_xyz and fr2_xyz from the Category: Testing and Debugging series, fr1_floor, fr2_desk and fr2_360_hemisphere from the Category: Handheld SLAM series in the TUM dataset, for a total of five scenes. Experimentation with static scenes is carried out in sequence (Figure 7).

Category: Testing and Debugging				
fr1/xyz	30.09s	7.112m	tgz (0.47GB)	more info
fr1/rpy	27.67s	1.664m	tgz (0.42GB)	more info
fr2/xyz	122.74s	7.029m	tgz (2.39GB)	more info
fr2/rpy	109.97s	1.506m	tgz (2.13GB)	more info
Category: Handheld SLAM				
fr1/360	28.69s	5.818m	tgz (0.45GB)	more info
fr1/floor	49.87s	12.569m	tgz (0.82GB)	more info
fr1/desk	23.40s	9.263m	tgz (0.36GB)	more info
fr1/desk2	24.86s	10.161m	tgz (0.37GB)	more info
fr1/room	48.90s	15.989m	tgz (0.83GB)	more info
fr2/360_hemisphere	91.48s	14.773m	tgz (1.50GB)	more info
fr2/360_kidnap	48.04s	14.286m	tgz (0.74GB)	more info
fr2/desk	99.36s	18.880m	tgz (2.01GB)	more info
fr2/large_no_loop	112.37s	26.086m	tgz (1.92GB)	more info
fr2/large_with_loop	173.19s	39.111m	tgz (2.83GB)	more info
fr3/long_office_household	87.09s	21.455m	tgz (1.58GB)	more info

Figure 7. Sequences for static experiments

As shown in Table 1, the root mean squared error (RMSE, Root Mean Squared Error) of the relative positional error of the ORB-SLAM visual odometer and the semi-direct method visual odometer of this paper are compared. It can be seen that the relative positional errors are close to each other on the feature-dense scene sequences fr1_xyz, fr2_desk and fr2_xyz, while the drift of this paper's semi-direct method odometry is smaller on the feature-sparse scene sequences fr1_floor and fr2_360_hemisphere. On fr1_floor and fr2_360_hemisphere, the drift of this paper's algorithm is slightly smaller than that of ORB-SLAM, and the trajectory tracked by this paper's algorithm is more comprehensive on fr2_360_hemisphere (Figure 8).

Table.1. Comparison of relative pose errors of two visual odometers on scene sequences

	RMSE(m)	
Serial scenario	ORB-SLAM	SVO-SLAM
fr1_xyz	0.0121	0.0126
fr2_xyz	0.0209	0.0218
fr1_floor	0.0097	0.0088
fr2_desk	0.0703	0.0718
fr2_360_hemisphere	0.3041	0.2841

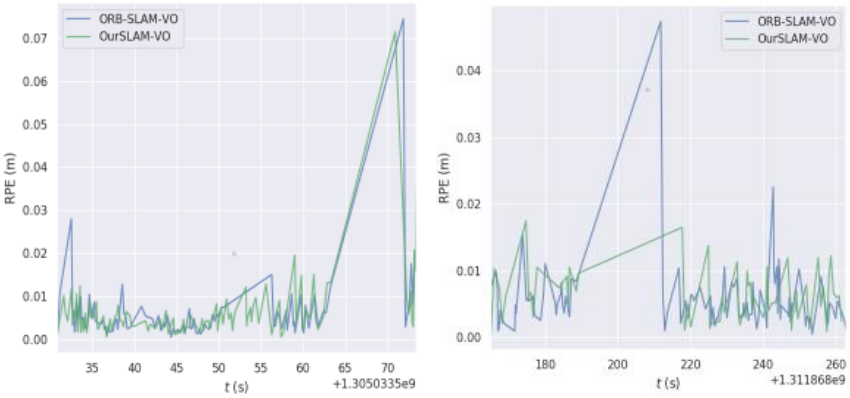

Figure 8. Comparison of two visual odometry on fr1 _ floor and fr2 _ 360 _ hemisphere

As shown in Figures 9, 10, and 11, the trajectories of ORB-SLAM are shown on the left, and the trajectories of SVO-SLAM in this paper are shown on the right. The camera trajectory estimated by ORB-SLAM visual odometry on sequence fr2_desk is better than the camera trajectory estimated by the semi-direct method visual odometry in this paper in general, and it fits the real trajectory better in the local poses. However, on the two scenes with sparse textures, fr1_floor and fr2_360_hemisphere, the camera trajectories estimated by the algorithm in this paper significantly outperform the results of ORB-SLAM. In addition, Table 2 shows the real-time performance of the two visual odometers on the TUM dataset. Obviously, the visual odometry proposed in this paper has faster tracking speed and better performance.

Table 2. Average tracking time of the two visual odometers

	ORB-SLAM	Our SLAM
Meantime(s)	0.0248	0.0225

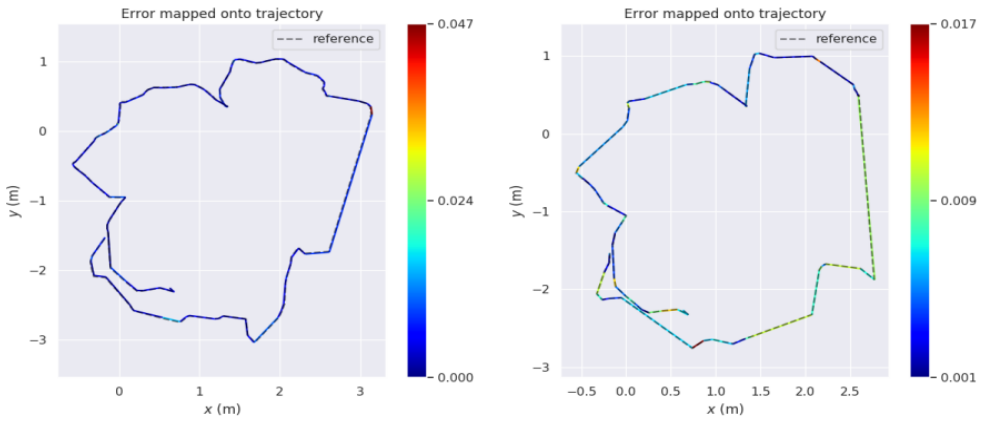

Figure 9. Comparison of estimated and real trajectories of two visual odometers on fr2_desk

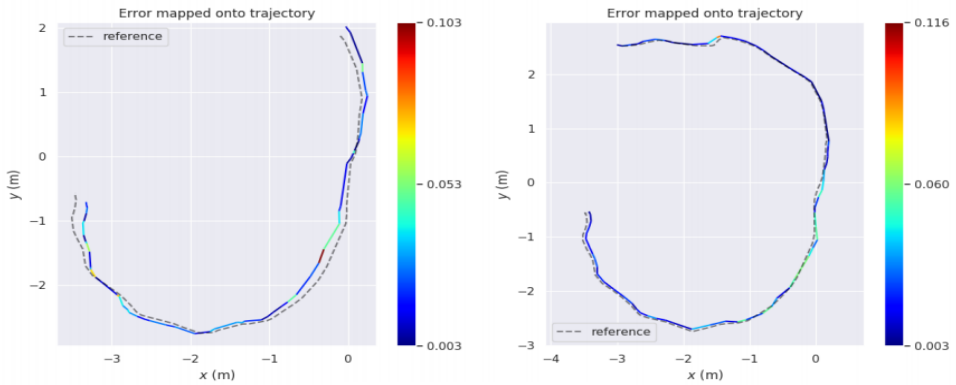

Figure 10. Comparison of estimated and real trajectories of two visual odometers on fr1--floor

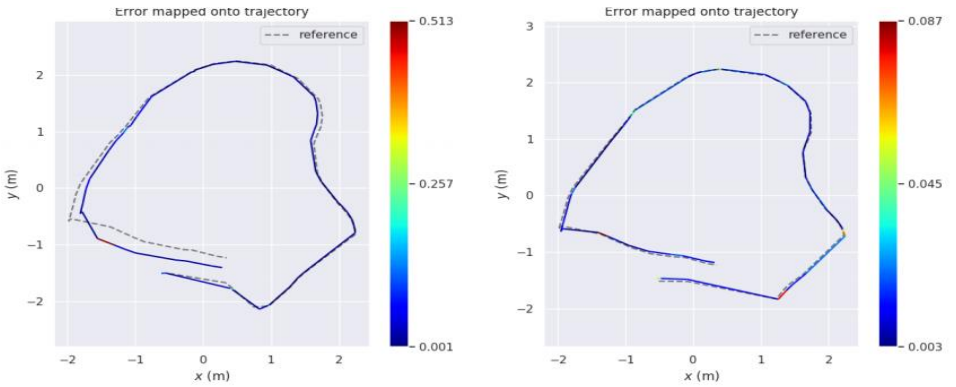

Figure 11. Comparison of estimated and real trajectories of two visual odometers on fe2_360_hemisphere

3.2.3 Dynamic scene experiment

The Dynamic Objects dataset (9) from TUM was used to study dynamic SALM, as shown in Figure 12. In this section, five dynamic scene sequences are selected for the experiment, the sequences are fr2_desk_with_person, fr3_sitting_xyz, fr3_walking_xyz, fr3_sitting _halfsphere and fr3_walking_halfsphere. in the fr2_ desk_with_person sequence a typical office scene is recorded with a person sitting at a desk. During the recording, the person moves and interacts with a number of objects (screen, phone, etc.); in the fr3_sitting_xyz and fr3_ sitting _halfsphere sequences, two people sit at a table talking and gesturing; in the fr3_walking_xyz and fr3_walking_ halfsphere sequences, two people walk through the office. Among them, fr2_desk_with_person, fr3_sitting_xyz and fr3_sitting _halfsphere belong to low dynamic scenes, where the moving objects move slowly and with small motion; fr3_walking_xyz and fr3_walking_halfsphere belong to high dynamic scene, the moving objects move fast and have a wide range of motion.

fr2/desk_with_person	142.08s	17.044m	tgz (2.71GB)	more info
fr3/sitting_static	23.63s	0.259m	tgz (0.44GB)	more info
fr3/sitting_xyz	42.50s	5.496m	tgz (0.79GB)	more info
fr3/sitting_halfsphere	37.15s	6.503m	tgz (0.66GB)	more info
fr3/sitting_rpy	27.48s	1.110m	tgz (0.49GB)	more info
fr3/walking_static	24.83s	0.282m	tgz (0.48GB)	more info
fr3/walking_xyz	28.83s	5.791m	tgz (0.54GB)	more info
fr3/walking_halfsphere	35.81s	7.686m	tgz (0.65GB)	more info
fr3/walking_rpy	30.61s	2.698m	tgz (0.54GB)	more info

Figure 12. TUM's Dynamic Objects dataset

As shown in Table 3, the performance of the ORB-SLAM visual odometer and the semi-direct method visual odometer of this paper on five sequences. Obviously, on fr3_walking_halfsphere, the trajectory error estimated by the semi-direct method visual odometer in this paper is obviously smaller than the result of ORB-SLAM.

Table 3. Comparison of the relative positional errors of the two visual odometers on five sequences

	RMSE(m)	
Sequences	ORB-SLAM	SVO_SLAM
fr2_desk_with_person	0.0681	0.0648
fr3_sitting_xyz	0.0148	0.0123
fr3_walking_xyz	0.0201	0.0163
fr3_ sitting _halfsphere	0.0142	0.0123
fr3_walking_halfsphere	0.0255	0.0169

As shown in Figures 13, 14 and 15, the trajectory maps of ORB-SLAM on the side and the SVO-SLAM trajectory maps of this paper on the right. The errors of the two algorithms in estimating the bit pose are similar on the sequences fr2_desk_with_person and fr3_ sitting _halfsphere, but the semi-direct method visual odometry in this paper tracks better in local details.

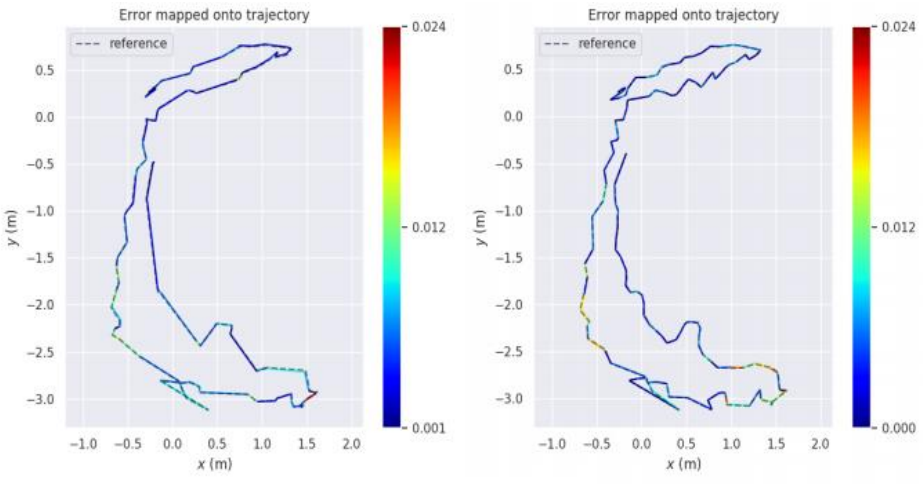

Figure 13. Comparison of estimated and real trajectories of two visual odometers on fr2_desk_with_person

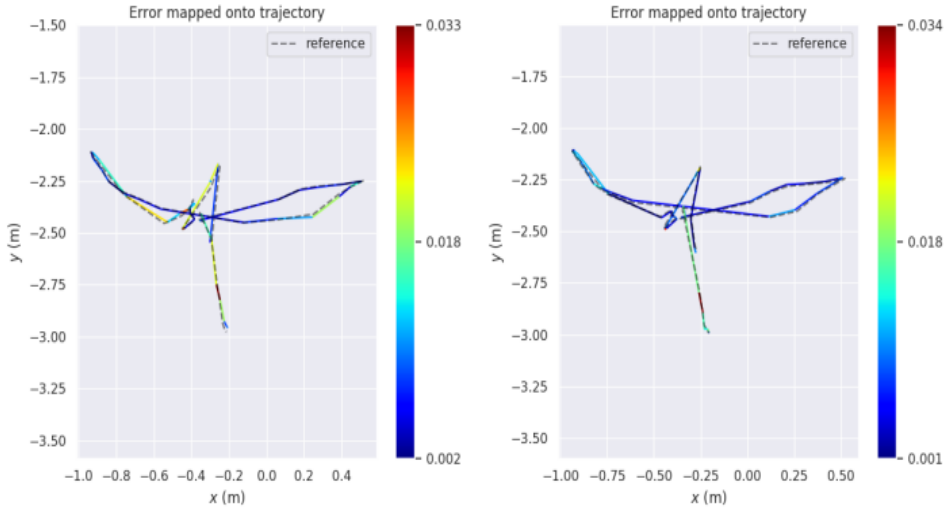

Figure 14. Comparison of estimated and real trajectories of two visual odometers on fr3_ sitting _halfsphere

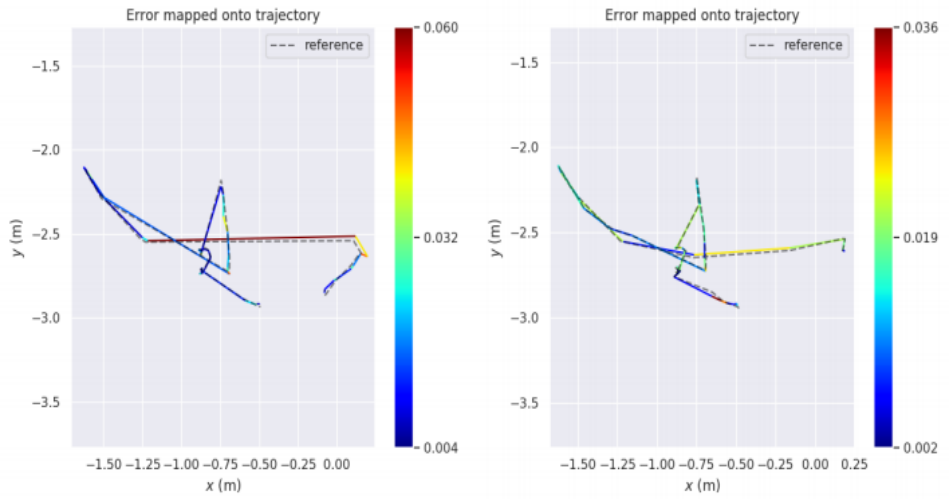

Figure 15. Comparison of estimated and real trajectories of two visual odometers on fr3_walking _halfsphere

4. Conclusions

This paper studies the visual odometer in SLAM system and improves the semi-direct visual odometer (SVO) algorithm based on the ORB-SLAM framework. The basic principle of the algorithm is to regard the pixel matching between adjacent image frames as an optimization problem. For sparse feature blocks, a direct registration method is used to obtain the camera pose, which avoids a lot of feature extraction and matching. When the key frame appears, the ORB (Oriented FAST and Rotated BRIEF) features of the image are extracted, and feature matching is used to track the local map to obtain more mapping relationships. In addition, the camera pose, and the three-dimensional structure of the scene are estimated by minimizing the reprojection error, thereby improving the positioning accuracy and map construction quality. Finally, experiments on several public datasets show that the improved semi-direct visual odometer scheme still has good robustness to moving objects in dynamic scenes. While maintaining high accuracy, it achieves better real-time performance by tracking visual feature points.

References

[1] Fischler M A, Bolles R C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography [J]. Communications of the ACM, 1981, 24(6): 381-395.

[2] Wang Y, Huang S. Towards dense moving object segmentation based robust dense RGB-D SLAM in dynamic scenarios[C]//2014 13th International Conference on Control Automation Robotics & Vision (ICARCV). IEEE, 2014: 1841-1846.

[3] Tiwari L, Ji P, Tran Q H, et al. Pseudo RGB-D for Self-Improving Monocular SLAM and Depth Prediction [J]. 2020.

[4] Hu H, Sun H, Ye P, et al. Multiple maps for the feature-based monocular SLAM system [J]. Journal of Intelligent & Robotic Systems, 2019, 94(2): 389-404.

[5] Li J, Pei L, Danping Z, et al. Attention-SLAM: A Visual Monocular SLAM Learning from Human Gaze [J]. IEEE Sensors Journal, 2020,21(05):6408-6420.

[6] Gastón C, Matías A, Nitsche, Taihú P, et al. Efficient on-board Stereo SLAM through constrained-covisibility strategies[J]. Robotics and Autonomous Systems, 2019, 116.

[7] Limeng L, Mei W. An Improved Stereo SLAM System through Combination of Points and Line Segments[J]. Computer Measurement & Control, 2019.

[8] Lee S H, Civera J. Loosely Coupled semi-direct monocular SLAM [J]. IEEE Robotics and Automation Letters, 2018, 4(2): 399-406.

[9] Li Y, Fan S, Sun Y, et al. Bundle adjustment method using sparse BFGS solution [J]. Remote Sensing Letters, 2018, 9(8):789-798.

[10] Yusefi A, Durdu A, Sungur C. ORB-SLAM-based 2D Reconstruction of Environment for Indoor Autonomous Navigation of UAVs[J]. European Journal of Science and Technology, 2020(Special):466-472.