Visual SLAM for Dynamic Environments based on Static Key-points Detection

Liangtao Zhong
School of Information Engineering
Southwest University of Science and Technology
Mianyang, China
zhongliangtao@mails.swust.edu.cn

Xiaoqiang Zhang*
School of Information Engineering
Southwest University of Science and Technology
Mianyang, China
xqzhang@swust.edu.cn

Lingyan Ran
School of Computer Science
Northwestern Polytechnical University
Xi'an, China
lran@nwpu.edu.cn

Yamin Han
College of Information Engineering
Northwest A&F University
Xianyang, China
yaminhan@nwafu.edu.cn

Hongyu Chu
School of Information Engineering
Southwest University of Science and Technology
Mianyang, China
chuhongyu@swust.edu.cn

***Abstract*—Recently, the technique of virtual reality has drawn great attention from both academics and industry fields. In many VR applications, the time-varying orientations and the positions of the VR headset are needed, where the technique of simultaneous localization and mapping (SLAM) can be applied. While great progress has been made in vision-based SLAM algorithms and systems in the recent two decades, the problem of visual SLAM in dynamic environments with moving objects is still unaddressed. In this paper, a static key-point detection approach is proposed based on optical flow feature clustering and potential moving region detection and is utilized for RGB-D SLAM in a dynamic environment. Firstly, considering the fact that there could be multiple moving objects in the scene, an optical flow feature is designed to cluster the multiple motion patterns in the scene. Secondly, an ultra-lightweight object detection network is utilized to detect potential moving regions in the scene, without using GPU resources. Finally, by utilizing the key-point clustering and potential moving regions detection, the static key-points are detected and inputted into the keypoint-based visual SLAM methods. The proposed approach is easy to insert as a pre-processing module in various key-point based visual SLAM systems. Extensive experimental results with qualitative and quantitative evaluations have demonstrated the performance with satisfactory accuracy of the proposed approach in various challenging dynamic scene scenarios.**

***Index Terms*—Simultaneous localization and mapping (SLAM), static key-points detection, object detection, optical flow.**

I. Introduction

In the past few years, virtual reality (VR) has been extensively investigated, because VR would change the way humans work and have fun. An increasing number of companies are using the in-outside tracking method to achieve VR headset position tracking instead of the outside-in method [1], [2]. The main reason is that outside-in tracking requires an external device to capture the location of the VR headset, which limits the VR device to a small space, while in-outside tracking only requires the camera to observe the environment and use SLAM to localise the device. In-outside tracking can improve the convenience of VR headset, but need to solve the problem of SLAM tracking accuracy in complex environments.

Visual SLAM has been developed significantly in recent years and can achieve good localization accuracy in most structured or static scenes [3]–[6], but there are still many challenges for complex scenes, especially those containing dynamic objects. Most of the current visual SLAM assumes that the environment is static, all key-points detected in the scene would be used for pose estimation. However, dynamic objects, such as moving people, are inevitable in the real world, and the use of key-points associated with dynamic objects will result in serious pose estimation errors and poor SLAM performances.

To address the above mentioned issue, it is necessary to identify the dynamic objects from the static parts and discard them before pose estimation. In the literature [7]–[10], semantic segmentation is used by many researchers as a pre-processing stage to filter out data related to dynamic objects. However, the high computational power, for example, high end GPUs, is usually required for semantic segmentation and is not supported by mobile devices such as VR headset.

In this paper, a static key-point detection approach based on optical flow feature clustering and potential moving region detection is proposed, which is integrated into the front-end of the ORB-SLAM2 [4] to build a complete SLAM system that is able to obtain better pose accuracy in dynamic scenes.

The main contributions of this paper include:

- A static key-point detection algorithm based on optical flow feature clustering and potential moving region detection is proposed in this paper, which can be integrated into most SLAM system that uses sparse key-points in the front-end.
- An ultra-lightweight object detection algorithm is utilized to filter potential motion objects, and no GPU is required

*Xiaoqiang Zhang is the corresponding author.

in the proposed method, which reduces computational cost in practical applications.
- The proposed algorithm is inserted into the front-end of ORB-SLAM2 [4] and is evaluated on the TUM RGB-D dataset. Experimental results with quantitative evaluation demonstrate that the proposed method can significantly improve the localization accuracy of the SLAM in dynamic environment.

The rest of this article is organized as follows. Section II is an overview of the development of SLAM and SLAM in dynamic environment in recent years. Then the framework and details of the proposed algorithm, including key-point optical flow feature clustering and static key-point detection, are given in section III. Subsequently, section IV provides qualitative and quantitative results of performance of the proposed method both on TUM RGB-D dataset. Finally, a brief conclusion is given in section V.

II. Related work

A. Visual SLAM

Visual SLAM is widely investigated, because of the convenience and low cost of the cameras. In early works [11] of pure visual information based SLAM approaches, salient image patches are served as the feature regions in the front end. An EKF framework is build for camera pose estimation in MonoSLAM [11]. For a large environment, since number of image feature and the state vector size in the EKF is large, the real time performance are affected. In order to deal with the computational load problem and with the help of multi-thread programming, the two main problems in SLAM, which are the motion tracking and map building, are simultaneously performed in PTAM [12]. The separated multi-threading framework are then widely used in modern visual SLAM approaches. Based on the multi-threading framework of PTAM, the real time and robustness performance of SLAM is highly improved in ORB-SLAM [3]. In the tracking thread or the front-end, the rotation invariant binary ORB feature [13] are used, which reduce the computational load in feature extraction and matching. In ORB-SLAM2 [4], the depth information of each feature points can be directly triangulated (stereo camera) or obtained (RGB-D camera), which increases the robustness in SLAM performance. The ORB-SLAM family get a important improvement in [5], which incorporates the input of inertial sensors for a more robust SLAM approach.

B. Visual SLAM in dynamic environments

The methods mentioned in subsection II-A assume that the environment is stationary, which would work well in most static scenarios. The method based on scenario static assumption needs to be combined with a new algorithm to work properly in dynamic scenarios.

Kundu et al. [14] detect dynamic key-points by defining a epipolar geometry constraint that states that the matching point in the subsequent view should lie on the corresponding pole line, and if the tracked feature is too far from the pole line, it is likely to be a dynamic key-point. Dai et al. [15] present a priori based on static object geometry in a feature-based SLAM framework, which uses the connectivity of map points to separate moving objects from the static background in order to reduce the influence of moving objects on the pose estimation. With the success of deep learning in computer vision, researchers are combining deep learning with SLAM to enhance SLAM performance. Wu et al. [8] and Xiao et al. [16], respectively, constructed a dynamic environment SLAM system in which a object detection algorithm is used to simply mask possible dynamic objects, such as people, cars. These methods reduce the number of key-points associated with dynamic objects and also reduce the number of static key-points in the bounding-box. If there are more dynamic objects in the scene, so that the bounding-box covers most of the area of the image, and the pose is not correctly estimated because of the small number of static key-points. Literature [7], [10], [17] use semantic segmentation to filter dynamic objects. Although it retains more static key-points in the scene to improve accuracy of the pose estimation, it cannot be performed in real time when no GPU is available.

III. The proposed approach

In this section, the proposed method will be described in detail. The framework of our method is presented in subsection III-A. In subsection III-B, key-point optical flow feature clustering is presented in detail. The static key-point detection algorithm utilizing key-point clustering and potential moving region detection is described in subsection III-C.

A. Framework

For precise localization in dynamic scenes, a straightforward idea is to discard dynamic key-points and use as many static key-points as possible in the scene for pose estimation. Fig. 1 illustrates the architecture of the proposed approach based on the idea.

The raw RGB image is fed into the proposed algorithm, which is quickly executed and then returns the correct static key-points to the SLAM system, as shown by the blue text and arrows in Fig. 1. The proposed algorithm mainly consists of two modules. In the first module, ORB key-points are tracked using optical flow. After that we model a Gaussian mixture model (GMM) [18] for clustering multiple motion patterns using self-designed optical flow feature. In the second module, potential moving regions are obtained through an ultra-lightweight object detection network without GPU resources. Then the static key-point detection algorithm, which combines the clustering and potential moving region results, outputs the correct static key-points to the visual odometer.

The proposed approach could be plugged into typical key-frame based sparse visual SLAM frameworks, for example, ORB-SLAM2 [4], while improving the localization accuracy in dynamic scenes. Fig. 2 shows the implementation details. The proposed method, which is shown as a green box, can be inserted into the front-end of visual SLAM. Since potential moving detection requires more computational resources, we

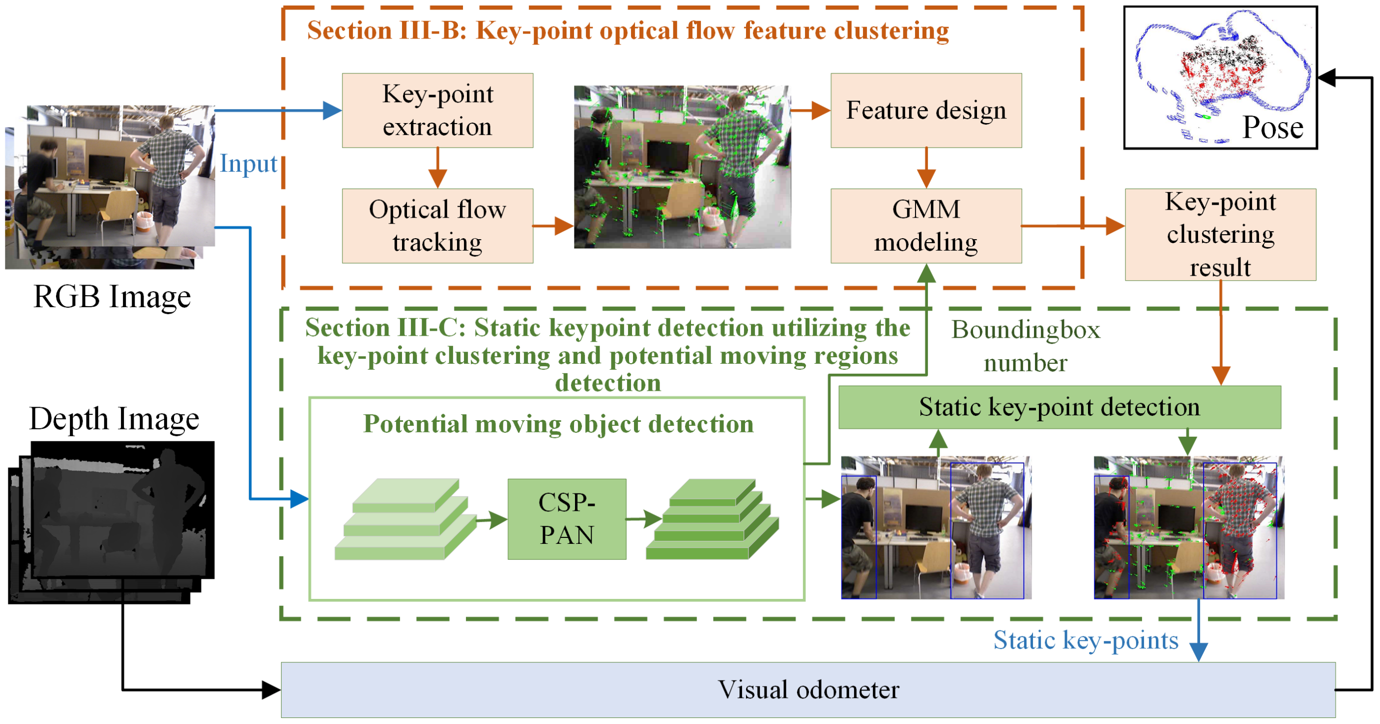

Section III-B: Key-point optical flow feature clustering
Key-point extraction
Optical flow tracking
Feature design
GMM modeling
Pose
Input
RGB Image
Key-point clustering result
Section III-C: Static keypoint detection utilizing the key-point clustering and potential moving regions detection
Boundingbox number
Depth Image
Potential moving object detection
Static key-point detection
CSP-PAN
Static key-points
Visual odometer

Fig. 1. The pipeline of the proposed method. In the key-point optical flow feature clustering module, the key-points of the raw RGB image are tracked using optical flow, then the self-designed optical flow features are built into Gaussian mixture models for key-point clustering (See Section III-B). In the static key-point detection module, static key-points are detected using the algorithm that combines key-point clustering results and potential moving object detection (See Section III-C).

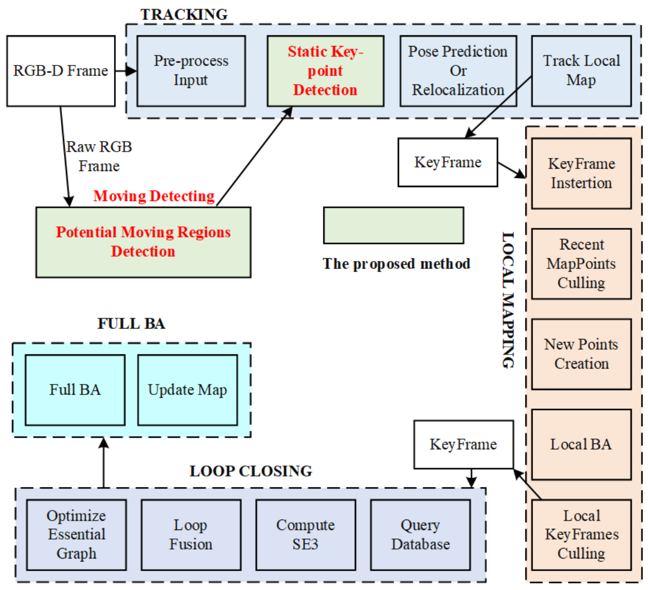


Fig. 2. The structure of the Enhanced ORB-SLAM2 that combines the proposed approach.

put this module into a independent thread for execution. The proposed algorithm in the *tracking* thread combines the potential moving regions output by the *moving detecting* thread to detect the static key-points, which are then fed to the subsequent module of the tracking thread.

B. Key-point optical flow feature clustering

It is a known fact that optical flow can represent the motion of objects in scene. Therefore, the idea of using optical flow to distinguish key-points associated with dynamic and static objects is straightforward.

We directly track the key-points needed for the front-end of the SLAM system using optical flow [19], for example in this paper we track the ORB features [13]. When tracking key-points using optical flow, we construct a first-in-first-out queue (FIFO) consisting of sequential n frame key-points from RGB image, and we always track the key-points at the head and tail of the queue, in order to get more significant optical flow features. Fig. 3 shows where the images that we will track come from. In order to cluster multiple motion patterns in the scene, the optical flow feature is designed and is represented by optical flow direction θ and norm l, denoted by $\boldsymbol{X} = [l\ \theta]$. The optical flow norm is defined as:

$$l = \|\overrightarrow{p_1 p_2}\|_2 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \tag{1}$$

The direction of the optical flow is denoted as:

$$\theta = \begin{cases} \arccos\left(\frac{x_2 - x_1}{l}\right) & y_2 - y_1 \geqslant 0 \\ -\arccos\left(\frac{x_2 - x_1}{l}\right) & y_2 - y_1 < 0 \end{cases}, \theta \in (-\pi, \pi]. \tag{2}$$

In all the above equations, $p_1 = (x_1, y_1)$ denotes the pixel coordinates of the key-point of the current frame, and $p_2 = (x_2, y_2)$ the pixel coordinates of key-point before n frames.

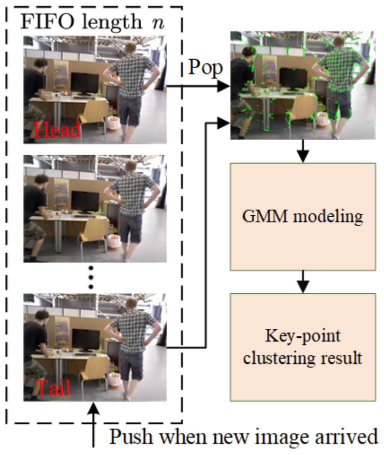


Fig. 3. Optical flow tracking block diagram. Optical flow tracks two images from the head and tail of the FIFO.

To obtained the ground truth labels of the motion patterns in the dynamic scene, an unsupervised clustering method is needed to distinguish different motion patterns in the scene. In this paper, we cluster different motion patterns using a Gaussian mixture model (GMM) [18]. We assume that there are b moving objects in the scene, the number of motion patterns is calculated by

$$N = 1 + \varphi + b, \tag{3}$$

where number 1 donates static motion pattern. Hyper-parameter φ is introduced to adapt to different scenarios where the moving objects in the scene may not be rigid bodies, meaning that a single moving object may yield multiple movement patterns, such as a person's lower body static while waving their hands. To obtain the number of moving objects in the scene, namely, b, a potential moving region detection algorithm is designed. The number of the detected bounding-boxes is regarded as the number of moving objects. Please see subsection III-C.

The optical flow feature of each motion pattern can be described by a Gaussian distribution. Then, the GMM is obtained by weighted sum of the feature of N motion patterns. The probability density function of the model is:

$$p\left(\boldsymbol{X}\right) = \sum_{i=1}^{N} \alpha_i N\left(\boldsymbol{X}; \boldsymbol{\mu}_i, \mathit{\Sigma}_i\right), \tag{4}$$

where $\boldsymbol{X}$ is the two-dimensional optical flow feature, $\boldsymbol{\mu}_i$ the mean vector of each Gaussian component, $\mathit{\Sigma}_i \in \mathbb{R}^{2\times2}$ the covariance matrix of each Gaussian component and α_i the mixing coefficient of each component.

The parameters of the GMM are iteratively estimated by the expectation maximization (GM) algorithm [20]. Up to this point, we have clustered the optical flow features into different motion patterns. However, the moving status of each clusters, for example, whether it is static or in motion, is still unknown. In the next subsection, the moving status of clusters is determined.

C. Static key-point detection utilizing the key-point clustering and potential moving regions detection

Potential moving regions detection. In this paper, we consider the outer rectangle of the human body as potential moving regions. And an ultra-lightweight object detection algorithm PP-PicoDet [21] is utilized to detect the regions, to balance efficiency and accuracy. The network backbone of PP-PicoDet was redesigned based on ShuffleNetV2, namely Enhanced ShuffleNet, to be more robust on devices with limited computing power resources. In the neck, the PAN structure is used to obtain multi-level feature maps and the CSP structure is used for feature concatenation and fusion between the adjacent feature maps. In the detector head, 5×5 convolution is used to expand the receptive field. Besides, depthwise convolution and pointwise convolution is widely used to reduce the parameter size. For classification, PP-PicoDet use Varifocal loss [22] to couple classification prediction and quality prediction. For regression, GIoU loss [23] and Distribution Focal Loss [24] are utilitied in PP-PicoDet. The total loss is as follows:

$$L = L_{vfl} + 0.25 L_{dfl} + 2 L_{giou}, \tag{5}$$

where L_{vfl} is the Varifocal Loss, L_{giou} the GIoU loss, L_{dfl} the Distribution Focal Loss.

The PP-PicoDet leverages the advantages of multiple SOTA algorithms to achieve better trade-offs between accuracy and latency compared to other popular models. The PP-PicoDet with only 0.99M parameters achieves 30.6% mAP in the Microsoft COCO dataset [25]. We use the NCNN [26] inference library to deploy the algorithm on an AMD Ryzen7 4800H CPU with only latency of 18ms. The potential motion region detection algorithm is designed with the input being the raw RGB image and the output being the location, class, and the number of bounding-boxes, which will be used in subsection III-B.

Static key-point detection. To detect static key-points, we consider that most key-points outside the potential moving region are static. Therefore, if a key-point inside the bounding-box is static, it is labeled the same as most key-points outside the region. Now, this problem changes to counting the number of various key-points outside the potential moving region, and then labeling the static key-points with the highest number. The static key-point detection algorithm is show in Algorithm 1. At this point, the filtered static key-points are added to the visual odometer for pose estimation.

IV. Experimental Results

In this section, experimental results would be presented to demonstrate the effectiveness of our method. Firstly, we introduce the dataset used in this paper. Then, we qualitatively evaluate the static key-point detection algorithm. We finally performed a performance comparison with multiple advanced SLAM systems using the TUM RGB-D dataset [27]. All the experiments are performed on a computer with AMD Ryzen7

Algorithm 1 Static Key-point Detection

```
Input: All key-point set $\mathcal{G} \in \{g_i | i \in \mathbb{N}\}$, Bounding-boxes set $b$, the number of motion patterns $N$, Key-point label set $K \in \{k_i | k_i \in \mathbb{N} \wedge k_i < N, i \in \mathbb{N}\}$
Output: Static key-point label $c$
1: Initialize an array of size $N$: classCount
2: for each $g_i \in \mathcal{G}$ do
3:     if $g_i.position \notin b$ then
4:         classCount[$k_i$] ← classCount[$k_i$] + 1
5:     end if
6: end for
7: $c$ ← MaxElementIndex(classCount)
8: return $c$
```

4800H and 32GB memory. The FIFO length n and hyper-parameters φ mentioned in subsection III-B are set to 4 and 2, respectively.

A. Dataset

The TUM RGB-D dataset [27] contains RGB-D data and ground-truth data in order to establish a novel benchmark for the evaluation of visual odometry and visual SLAM systems. The dataset contains the color and depth images of a Microsoft Kinect sensor along the ground-truth trajectory of the sensor. The data was recorded at 30 Hz and sensor 640×480 resolution. The ground-truth trajectory was obtained from a high-accuracy motion-capture system. In the dataset, sequences *f3_walking_** and *f3_sitting_** are high dynamic scene and low dynamic scene respectively. To measure the quality of the estimated camera trajectory of visual SLAM systems, TUM dataset presents different evaluation metrics. The root mean squared error (RMSE) of absolute trajectory error (ATE) is defined as follows:

$$r_{\text{ate}} = \min_{\mathbf{T}\in\mathrm{SE}(3)} \sqrt{\frac{1}{|I_{\text{gt}}|} \sum_{i\in I_{\text{gt}}} \|\mathbf{T}\mathbf{p}_i - \hat{\mathbf{p}}_i\|^2}, \tag{6}$$

where $\hat{\mathbf{p}}_i$ is ground-truth 3D positions, $\mathbf{p}_i$ tracked positions, $\mathbf{T}$ an optimal pose $\mathrm{SE}(3)$ aligned $\hat{\mathbf{p}}_i$ and $\mathbf{p}_i$. And all tracked poses where ground truth is available are used, which corresponds to indices I_{gt}.

The RMSE of the relative pose error (RPE) is defined as:

$$\begin{aligned} r_{\text{rpe}} &= \sqrt{\frac{1}{|I_{\text{gt},\Delta}|} \sum_{i\in I_{gt},\Delta} \|\text{trans}(\mathbf{E}_i)\|^2} \\ \mathbf{E}_i &= \left(\hat{\mathbf{T}}_i^{-1}\hat{\mathbf{T}}_{i+\Delta}\right)^{-1}\left(\mathbf{T}_i^{-1}\mathbf{T}_{i+\Delta}\right), \end{aligned} \tag{7}$$

where $\text{trans}(\cdot)$ takes the 3D translational component of a pose. This error measures how accurate pose changes are in a small time interval Δ. The set of frame indices $I_{\text{gt},\Delta}$ is the same as I_{gt}, but we have to take out Δ poses at the end of each tracked segment.

The ATE stands for global consistency of trajectory, while RPE measures the translational and rotational drift. In addition, to further evaluate the effectiveness of the proposed algorithm, we introduced the mean error, standard deviation, and median of ATE and RPE respectively in our experiments.

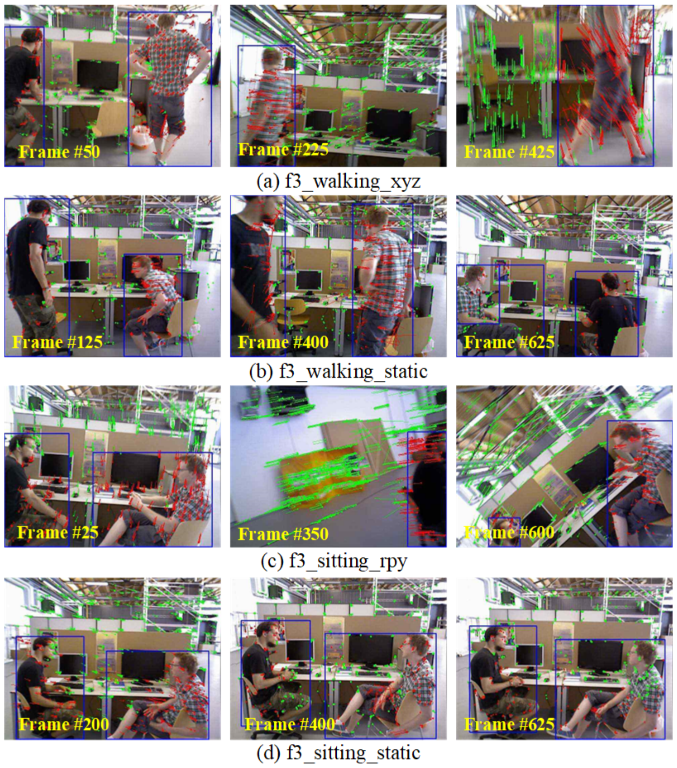


Fig. 4. Part of the results of static key-points detection in the TUM RGB-D dataset.

B. Qualitative evaluation of static key-point detection

Fig. 4 show part of the results of static key-point detection on the TUM RGB-D dataset. The blue box, red line and green line donate the potential moving regions detected by PP-PicoDet [21] , dynamic key-points and static key-points, respectively. The sub-figures and yellow words indicate where the image comes from. As we can see in Fig. 4 , the proposed method is able to effectively detect potential moving objects (person) in dynamic scene, which is robust to body partial occlusion and image rotation. The static key-points can be correctly detected by the proposed method. Since the hyper-parameter φ is introduced in this paper, static key-points in static regions of the body can also be detected normally when the human body is partially moving, as in frame #400 and frame #625 of *f3_sitting_static* sequence.

C. Quantitative evaluation using TUM RGB-D dataset

To further evaluate the performance of the proposed method, we compare it with the vanilla ORB-SLAM2 [4] and DS-SLAM [17]. To more visually demonstrate the performance of the proposed method, the results of multiple SLAM system runs on the *f3_walking_xyz* sequence(high dynamic scene) of TUM RGB-D dataset are plotted, as Fig. 5. It can be seen that the proposed method has significantly smaller ATE and RPE in high dynamic scenes compared to the remaining two algorithms.

TABLE I

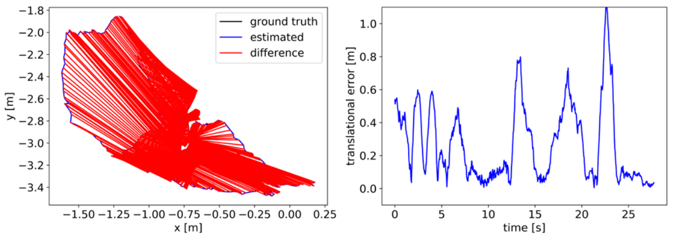
Results Of Metrics Absolute Trajectory Error (ATE)

Sequences	**ORB-SLAM2 [4]**				**DS-SLAM [17]**				**Ours**				**Improvement compared to ORB-SLAM2**			
	rmse	**mean**	**median**	**std**	**rmse**	**mean**	**median**	**std**	**rmse**	**mean**	**median**	**std**	**rmse**	**mean**	**median**	**std**
f3_sitting_rpy	0.0222	0.0172	0.0130	0.0139	0.0197	0.0143	**0.0101**	0.0136	**0.0153**	**0.0123**	0.0102	**0.0091**	30.79%	28.59%	21.55%	34.31%
f3_sitting_static	0.0082	0.0072	0.0065	0.0038	**0.0066**	**0.0057**	**0.0050**	0.0033	0.0070	0.0063	0.0058	**0.0031**	13.86%	12.74%	10.88%	18.02%
f3_walking_hs	0.5847	0.5028	0.4300	0.2986	0.0328	0.0279	0.0226	0.0174	**0.0295**	**0.0253**	**0.0212**	**0.0151**	94.95%	94.96%	95.08%	94.94%
f3_walking_rpy	0.7093	0.5777	0.3910	0.4116	0.3669	0.3242	0.2628	0.1719	**0.0284**	**0.0210**	**0.0163**	**0.0190**	96.00%	96.36%	95.84%	95.38%
f3_walking_static	0.1641	0.1505	0.1738	0.0654	**0.0069**	**0.0062**	0.0062	**0.0029**	0.0076	0.0063	**0.0057**	0.0042	95.37%	95.80%	96.70%	93.52%
f3_walking_xyz	0.7283	0.6297	0.5476	0.3658	0.2288	0.2139	0.2188	0.0814	**0.0157**	**0.0135**	**0.0119**	**0.0079**	97.85%	97.85%	97.82%	97.84%

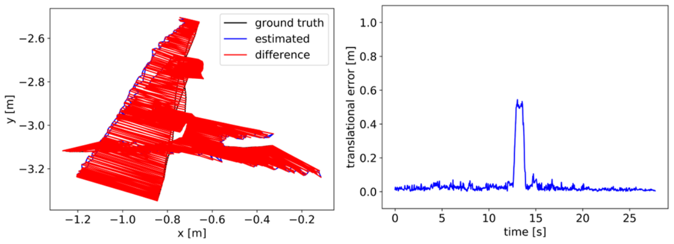
TABLE II

Results Of Metrics Relative Position Error (RPE)

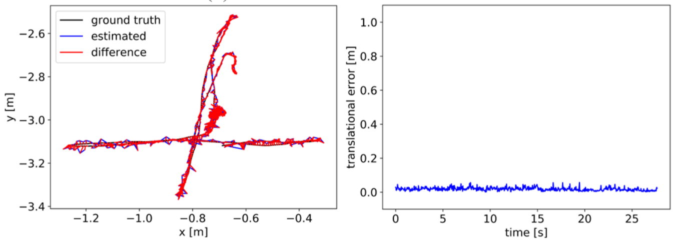
Sequence	**ORB-SLAM2 [4]**				**DS-SLAM [17]**				**Ours**				**Improvement compared to ORB-SLAM2**			
	rmse	**mean**	**median**	**std**	**rmse**	**mean**	**median**	**std**	**rmse**	**mean**	**median**	**std**	**rmse**	**mean**	**median**	**std**
f3_sitting_rpy	**0.0136**	0.0100	0.0081	**0.0091**	0.0140	0.0106	0.0079	0.0092	0.0139	**0.0100**	**0.0077**	0.0097	-2.63%	0.83%	4.89%	-6.66%
f3_sitting_static	0.0049	0.0042	0.0037	**0.0025**	0.0063	0.0055	0.0050	0.0030	**0.0046**	**0.0038**	**0.0031**	0.0026	6.09%	9.64%	16.77%	-3.20%
f3_walking_hs	0.0463	0.0182	0.0131	0.0426	0.0148	0.0116	0.0094	0.0093	**0.0139**	**0.0107**	**0.0085**	**0.0088**	70.02%	41.11%	34.99%	79.28%
f3_walking_rpy	0.0317	0.0233	0.0173	0.0214	0.0252	**0.0163**	**0.0114**	0.0193	**0.0236**	0.0171	0.0129	**0.0163**	25.38%	26.74%	25.32%	23.80%
f3_walking_static	0.0170	0.0111	0.0067	0.0129	0.0071	0.0057	0.0049	0.0043	**0.0052**	**0.0044**	**0.0039**	**0.0027**	69.59%	60.13%	42.08%	79.16%
f3_walking_xyz	0.0247	0.0205	0.0170	0.0139	0.0151	0.0114	0.0085	0.0099	**0.0127**	**0.0104**	**0.0084**	**0.0073**	48.68%	49.41%	50.90%	47.13%



(a) The ATE and RPE of ORB-SLAM2



(b) The ATE and RPE of DS-SLAM



(c) The ATE and RPE of the proposed method

Fig. 5. The plots of ATE and RPE from multiple visual SLAM.

We evaluate the proposed method on six of the representative sequences of the TUM RGB-D dataset, namely sequence *f3_sitting_rpy*, sequence *f3_sitting_static*, sequence *f3_walking_halfsphere*, sequence *f3_walking_rpy*, sequence *f3_walking_static* and sequence *f3_walking_xyz*. More quantitative comparison results are shown in Table I and Table II.

The **bolded** font in the table indicates the best items and **Improvement compared to ORB-SLAM2** columns in the table refers to the improvement of the method in this paper relative to the vanilla ORB-SLAM2. The improvement values in the tables are calculated as follows:

$$\eta = (1 - \frac{c}{m}) \times 100\%, \tag{8}$$

where η is the value of improvement, m the value of vanilla ORB-SLAM2 and c the value of the proposed method.

As we can see from Table I and Table II, the proposed method gets a significant improvement with ATE and RPE in high dynamic scenes, for example, *f3_walking_** sequences, compared to the vanilla ORB-SLAM2. In terms of ATE, all metrics in high dynamic scenes are improved by more than 93% over the vanilla ORB-SLAM2. In a few sequences, DS-SLAM [17] achieved little better results than ORB-SLAM2 [4] and the proposed method. In *f3_sitting_rpy* and *f3_sitting_static* sequences, vanilla ORB-SLAM2 has performed better than the other two algorithms in part metrics, which we believe is mainly due to the fact that ORB-SLAM2 utilizes the RANSAC [28] to eliminate outliers in low dynamic scenes, making it robust to small movements.

V. Conclusions

In this paper, a static key-point detection approach is proposed based on optical flow feature clustering and potential moving region detection. The proposed approach is inserted as a pre-processing module to key-point based visual SLAM system, ORB-SLAM2. We qualitatively evaluate the static key-point detection approach in this paper, and quantitatively compare the Enhanced ORB-SLAM2 with the vanilla ORB-SLAM2 and DS-SLAM. The experimental results demonstrate that the proposed approach can significantly improve the performance of the key-point based visual SLAM system in dynamic scenes.

However, there are still some ongoing works in the proposed method. First we can redesign the deep object detection network for SLAM system to further improve the performance. Further, we can also extend the system to multiple non-overlapping cameras. We can use multiple non-overlapping cameras to observe different fields of view after solving for the external parameters of the multiple non-overlapping cameras using the method of literature [29], to improve the robustness of the SLAM system.

Acknowledgment

This work is supported in part by the Natural Science Foundation of China under Grant 62201479, in part by the Natural Science Foundation of Sichuan Province under Grant 2023NSFSC1388, in part by the Doctoral Fund of Southwest University of Science and Technology under Grant 19zx7123, in part by the Open Fund of Key Laboratory of Civil Aircraft Airworthiness Technology under Grant SH2020112706.

References

[1] "Pico," https://www.picoxr.com/.

[2] "Meta Oculus Quest 2," https://www.meta.com/quest/.

[3] 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.

[4] 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.

[5] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[6] T. Qin, P. Li, and S. Shen, "VINS-mono: A robust and versatile monocular visual-inertial state estimator," vol. 34, no. 4, pp. 1004–1020, conference Name: IEEE Transactions on Robotics.

[7] Y. Ai, T. Rui, M. Lu, L. Fu, S. Liu, and S. Wang, "Ddl-slam: A robust rgb-d slam in dynamic environments combined with deep learning," *IEEE Access*, vol. 8, pp. 162 335–162 342, 2020.

[8] W. Wu, L. Guo, H. Gao, Z. You, Y. Liu, and Z. Chen, "Yolo-slam: A semantic slam system towards dynamic environment with geometric constraint," *Neural Computing and Applications*, vol. 34, no. 8, pp. 6011–6026, 2022.

[9] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1168–1174.

[10] I. Ballester, A. Fontán, J. Civera, K. H. Strobl, and R. Triebel, "Dot: Dynamic object tracking for visual slam," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 705–11 711.

[11] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[12] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.

[13] 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.

[14] A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 4306–4312.

[15] W. Dai, Y. Zhang, P. Li, Z. Fang, and S. Scherer, "Rgb-d slam in dynamic environments using point correlations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 373–389, 2020.

[16] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou, "Dynamic-SLAM: Semantic monocular visual localization and mapping based on deep learning in dynamic environment," vol. 117, pp. 1–16, publisher: Elsevier.

[17] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1168–1174.

[18] D. A. Reynolds, "Gaussian mixture models." *Encyclopedia of biometrics*, vol. 741, no. 659-663, 2009.

[19] B. D. Lucas, T. Kanade *et al.*, *An iterative image registration technique with an application to stereo vision*. Vancouver, 1981, vol. 81.

[20] N. Sammaknejad, Y. Zhao, and B. Huang, "A review of the expectation maximization algorithm in data-driven process identification," *Journal of process control*, vol. 73, pp. 123–136, 2019.

[21] G. Yu, Q. Chang, W. Lv, C. Xu, C. Cui, W. Ji, Q. Dang, K. Deng, G. Wang, Y. Du *et al.*, "Pp-picodet: A better real-time object detector on mobile devices," *arXiv preprint arXiv:2111.00902*, 2021.

[22] H. Zhang, Y. Wang, F. Dayoub, and N. Sunderhauf, "Varifocalnet: An iou-aware dense object detector," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8514–8523.

[23] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 658–666.

[24] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, and J. Yang, "Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 002–21 012, 2020.

[25] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[26] "NCNN," https://github.com/Tencent/ncnn.

[27] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1680–1687.

[28] J. M. Martínez-Otzeta, I. Rodríguez-Moreno, I. Mendialdua, and B. Sierra, "Ransac for robotic applications: A survey," *Sensors*, vol. 23, no. 1, p. 327, 2022.

[29] X. Zhang, L. Zhong, C. Liang, H. Chu, Y. Shao, and L. Ran, "Metric calibration of aerial on-board multiple non-overlapping cameras based on visual and inertial measurement data," in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*. Springer, 2021, pp. 16–28.