A dynamic detection method to improve SLAM performance*

GAN Yu (甘雨), **ZHANG Jianhua** (张剑华)**, **CHEN Kaiqi** (陈凯祺), **and LIU Jialing** (刘嘉玲)
College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China

(Received 1 March 2021; Revised 19 April 2021)

Simultaneous localization and mapping (SLAM) technology is a research hotspot in the field of intelligent mobile robot, and many researchers have developed many classic systems in the past few decades. However, most of the existing SLAM methods assume that the environment of the robot is static, which results in the performance of the system being greatly reduced in the dynamic environment. To solve this problem, a new dynamic object detection method based on point cloud motion analysis is proposed and incorporated into ORB-SLAM2. First, the method is regarded as a preprocessing stage, detecting moving objects in the scene, and then removing the moving objects to enhance the performance of the SLAM system. Experiments performed on a public RGB-D dataset show that the motion cancellation method proposed in this paper can effectively improve the performance of ORB-SLAM2 in a highly dynamic environment.

Document code: A **Article ID:** 1673-1905(2021)11-0693-6
DOI https://doi.org/10.1007/s11801-021-1022-5

In the field of intelligent mobile robots, simultaneous localization and mapping (SLAM) is one of the most basic and important functions, which can determine the environment perception ability of a robot. With the development of RGB-D camera and computer vision, some advanced SLAM systems have achieved satisfactory performance, such as DVO-SLAM[1] and ORB-SLAM2[2].

However, the results of many current slam algorithms are obtained under the assumption that the surrounding environment is static. But this assumption is usually incorrect in the real environment because moving objects in a real scene are inevitable. Therefore, these algorithms do not work well in real scenarios with dynamic objects. In some cases, the movement of objects in the scene will affect the quality of state estimation, and even lead to system failure. For example, in the environment, dynamic objects (pedestrians, animals) will cause inaccurate feature matching between the front and rear frames, and cause the SLAM system to record the object in the final generated map, which affects the accuracy of the generated map[3]. Therefore, how to keep the stability of the existing slam algorithm in dynamic scenarios is still a challenge.

In fact, some classic SLAM systems are robust to dynamic environments. For example, RANSAC matching[4], They can adapt to a small number of dynamic changes in the scene. However, when dynamic objects occupy a large proportion in the environment, a specific method is needed to distinguish moving objects and static scenes. In recent years, with the continuous research on slam problem, for dynamic scenes, it is roughly divided into two methods: geometric angle and deep learning methods. The most typical geometric angle is the method using optical flow method and depth data correlation[5]. And deep learning focuses on using learning methods to find out the dynamic objects that may exist in the scene and then discriminate them[6].

In this paper, in order to improve the robustness of the slam system in a dynamic environment, based on point cloud motion analysis[7], we propose a new method to detect dynamic objects by using point cloud motion vector information, focusing on detecting and filtering dynamic objects in real scenes. This method only uses three-dimensional (3D) point cloud information, so it is robust to light changes. At the same time, the method is incorporated into ORB-SLAM2, as the preprocessing stage, which is used to detect the dynamic objects that appear in the scene. Experiments on public RGB-D datasets show that our method performs well.

The main contributions of this paper are as follows:

1. A new method of dynamic object detection is proposed. This method only uses RGB-D data to detect dynamic objects that appear in the scene.

2. Based on ORB-SLAM2, a thread for detecting dynamic objects is added, and the robustness and accuracy of SLAM are improved by filtering dynamic objects in dynamic scenes by combining RGB images. Experiments

* This work has been supported by the National Natural Science Foundation of China (No.61876167), and the Natural Science Foundation of Zhejiang Province (No.LY20F030017).
** E-mail: zjh@zjut.edu.cn

are performed on the TUM RGB-D dataset, and the results prove the effectiveness of the method in improving the ORB-SLAM2 system in a dynamic environment.

The original design of the SLAM system is based on the assumption that the scene is static. This leads to the drift and even loss of attitude estimation due to the lack of effective processing of dynamic objects in a dynamic environment.

To solve this problem, dynamic object removal is required for the images captured by the camera. In the past few years, many methods for removing moving objects have been proposed. The optical flow method is a very classic algorithm for detecting dynamic objects in a scene[8]. It uses the changes in the time domain of pixels in the image sequence and the correlation between adjacent frames to find the correspondence between the previous frame and the current frame Relationship to calculate the motion information of objects between adjacent frames.

For RGB-D data, DEWAN et al[3] found matching sparse feature pairs from two frames of RGB-D data, segmented them into different motion groups, and processed outliers and dynamic scenes in this way. SCONA et al[9] proposed an RGB-D SLAM system based on Surfel, which simultaneously estimates the pose of the RGB-D camera and divides the static pixels in the current frame of the picture.

Besides, some methods are not just using motion segmentation, including CubeSLAM[10] and ClusterVO[11]. They use 3D object detection methods to extract objects from a single frame of images and provide more constraints through the BA adjustment stage of the detected objects to improve the robustness of the dynamic SLAM system.

With the development of deep learning, DynaSLAM[6] try to integrate semantic segmentation and geometric judgment methods to deal with dynamic scenes. In Ref.[6], a new one can be improved simultaneously the semantic segmentation framework, the segmentation, and visual SLAM are performed in an interweaved method and the results are used to refine each other.

Considering that there may not necessarily be a reliable static structure in complex dynamic scenes, researchers try to further improve the accuracy of camera pose estimation by tracking and estimating a series of attributes such as the pose change and speed of moving objects in the scene, such as VDO-SLAM[12] and DynaSLAM2[13].

Many of the systems mentioned above[6,9-13] use deep learning methods more or less. In some cases, the bottleneck caused by the speed of deep learning is the system cannot be implemented in real-time. Different from other dynamic SLAM with deep learning, our method only uses 3D information, we do not rely on CNN to detect specific objects (pedestrians, vehicles), we are targeting certain moving objects in 3D scenes. These moving objects can be in any category, and detecting them does not require additional training. Our method can be easily extended to outdoor scenes.

In this section, a dynamic detection method is proposed to improve the performance of ORB-SLAM2 in dynamic scenes. The overall process of our SLAM framework is illustrated in Fig.1. First of all, the RGB image and depth image captured by Kinect2 is processed in the tracking thread and the dynamic detection thread at the same time. First, the tracking thread extracts the ORB feature points of the current frame. Then, according to the segmentation result obtained by the dynamic detection thread, the tracking thread filters out the ORB feature points located in the dynamic object, thereby obtaining a more accurate attitude estimation result. Next, the method of detecting dynamic objects will be described in detail.

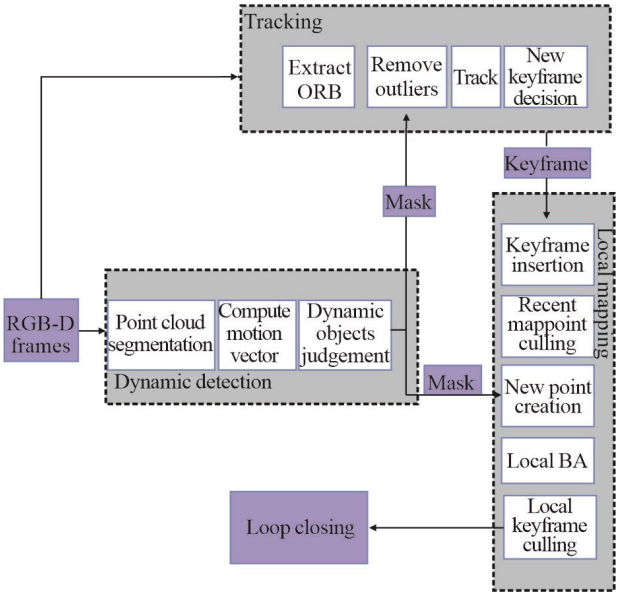


Fig.1 The overview of SLAM framework

The first step of the method requires different element segmentation based on the 3D point cloud for object detection[14]. Firstly, the obtained point cloud frame is gridded by voxel, which is to remove the outliers in the point cloud and ensure that there is no excessive noise in the point cloud. The filtered point cloud is divided into a set of point cloud clusters P_i by Euclidean clustering. For each point cloud cluster P_i, their centroids are calculated for later detection.

Due to the limitation of the camera range, the incomplete shape of some objects may form trivial and unreliable features. In order to ensure the effect of point cloud segmentation and the accuracy of detecting dynamic objects, this paper ignores clusters of less than 200 points.

For each input frame, suppose P is one of the objects in the input segmentation result, let P_1 be its most similar object in the previous frame:

$$B = \left\{ y \in P : \left\| y - P_c \right\|_2 \le r \right\}, \tag{1}$$

$$V(x) = x - \arg\min_{y \in P_1} \left\| x - y \right\|_2, \tag{2}$$

where r is determined by the distance of object from the camera. Then calculate the motion vector $\boldsymbol{W}(P)$ of P by the following equation:

$$\boldsymbol{W}(P)=\arg\max_{\partial}\sum_{x\in B}\boldsymbol{V}(x)^{\mathrm{T}}\partial\ . \tag{3}$$

Eq.(3) can be transformed into a problem of finding a singular value vector of a matrix $\boldsymbol{M}$ where each row is $\boldsymbol{V}(x)$.

Tab.1 The algorithm for detecting dynamic objects

Algorithm 1: dynamic objects detection

Input: point cloud array $X=X^1, \ldots, X^9$
Output: mask-image
  Point cloud segmentation P=Segment(X^4)
  For each $P_i \in P$
    Compute motion vector $\boldsymbol{W}_i$
    For each $X^i \in X$ do
      Compute histogram H^i
    End for
    Construct motion matrix $\boldsymbol{M}$
      Compute the line integral value δ using line detection algorithm on $\boldsymbol{M}$
    If $\delta>\tau$ then
      P_i is dynamic
    Else
      P_i is static
    End if
  End for
  Create the mask-image by the P

According to the motion vector $\boldsymbol{W}(P)$, we can get the projected histogram of P in the $\boldsymbol{W}$ direction by the following equation:

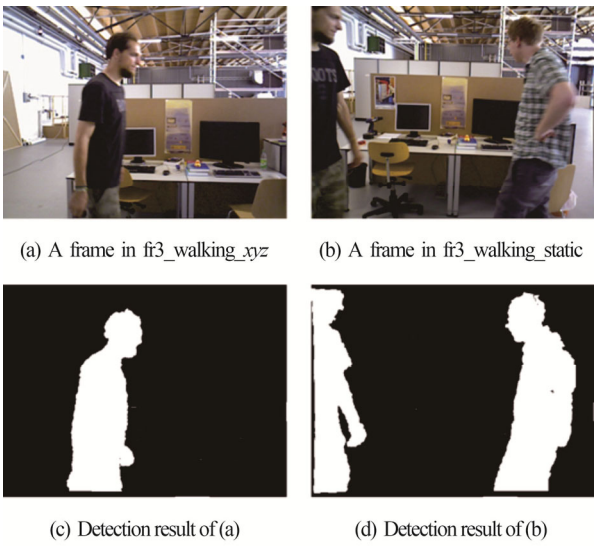
$$D(P,\boldsymbol{W})=\left\{d:d=\frac{\boldsymbol{W}(P)^{\mathrm{T}}\times x-P_c}{r},x\in P\right\}. \tag{4}$$

Considering that the structure of the local point sets is preserved, so at time t, the local point set moving in the $\boldsymbol{W}(P)$ direction is homomorphic[7]. Then through the projection distance of its point set and its motion vector $\boldsymbol{W}(P)$, the position of the object can be found recursively at each moment. Therefore, we calculate a histogram of 9 bins by superimposing a total of 9 frames of point clouds before and after. The bin in the histogram at each time is the projection distance of the object on the motion vector $\boldsymbol{W}(P)$.

To obtain the motion relationship between the objects from the histogram, we fixed each bin into n parts according to the distance, in order to obtain a $9\times n$ two-dimensional (2D) matrix. Then calculate the line integral in the 2D matrix by applying the line detection algorithm to the matrix. Because the position of a static object is almost constant, its line integral value must be very small. With this in mind, we think that if the line integral value is greater than a certain threshold τ, it is a dynamic object. After experiments, we have concluded that $\tau=0.25\times S_P$ (S_P is the total number of points in the object P) is a reasonable choice.

Each point in the point cloud can find the corresponding location in the original image. Therefore, we can find the location of the dynamic object directly in the original image. The output of the dynamic detection thread is a mask of the same size as the original image. The mask has only one channel. If a point in the point cloud belongs to a dynamic object, its pixel at the corresponding position in the image is set to 1, if it is a static object, it is set to 0.

After detecting the ORB feature points of the image, all ORB feature points belonging to the dynamic object can be filtered out before matching. In this way, the effect of dynamic objects on the matching effect can be significantly reduced. Fig.2 demonstrates some selected detection results using the dynamic objects datasets. As we can see, our approach is able to detect moving objects effectively in various challenging scenarios.



(a) A frame in fr3_walking_*xyz* (b) A frame in fr3_walking_static

(c) Detection result of (a) (d) Detection result of (b)

Fig.2 Detect results using the TUM dynamic objects datasets

In this section, experimental results will be provided to demonstrate the effectiveness of our method in a dynamic environment. We performed experiments using the public dynamic TUM RGB-D dataset and compared it with the original ORB-SLAM2 to quantify the improvement of our method in dynamic scenarios. Some images of ORB-SLAM2 at runtime are selected to illustrate the effectiveness of our method detection in real scenarios. Besides, we show the effect of using dynamic detection methods on datasets in their own generated dynamic environment. We ran each sequence ten times to avoid uncertainty in dynamic scenes. All the experiments are performed on a computer with Inter i7 CPU, 16 GB memory.

The TUM RGB-D dataset[15] provides multiple sequ-

ences in a dynamic environment. 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 full frame rate (30 Hz) and sensor resolution (640×480). In the sitting sequence, there are two sitting on a chair and speaking and gesturing, this is a low dynamic movement. In the walking sequence, two people are walking back and forth at the desk, sometimes sitting back in the chair, and sometimes walking out of the camera, the scene is highly dynamic. The walking sequence is mainly used in our experiments because in this highly dynamic environment, the robustness and accuracy of the SLAM system will be greatly affected. We used four types of camera self-motion sequences to test: static, *xyz*, *rpy*, and halfsphere.

Absolute trajectory error (ATE) and relative posture error (RPE) are two indicators used for quantitative evaluation. The metric ATE measures the global consistency, and the metric RPE measures the odometry drift.

Tabs.2—4 show the results of the quantitative comparison, where the first column of *xyz*, static, *rpy*, and half represent the four types of camera self-motion. "Without our method" means using the original ORB-SLAM2 system. The term "use our method" means that our method of removing dynamic objects is integrated into ORB-SLAM2. The values of root-mean-square error (*RMSE*), mean error, median error and standard deviation (*S.D.*) are introduced in this article, while *RMSE* and *S.D.* are more worthy of attention because they can better indicate the robustness and stability of the system. We also show the improved performance of ORB-SLAM2 after adding our method of removing dynamic objects compared to the original ORB-SLAM2. The improvement values in the table are calculated as

$$\eta = \left(\frac{\alpha - \beta}{\alpha} \right) \times 100\% , \tag{5}$$

where η represents the improvement value, α represents the value without our approach, and β represents the value with our approach.

From Tab.2, It can be seen that there is the average *RMSE* and *S.D.* high dynamic sequences have improved values of 80.18% and 77.00%, respectively. This shows that for high dynamic scenarios, our method can greatly improve SLAM performance in terms of ATE. At the same time, our method also enhances the stability of ORB-SLAM2 in dynamic scenarios. By examining *RMSE* values, we find that our method brings more improvements in static and *xyz* sequences. Tabs.3 and 4 show the performance of the visual odometer. As we can see, the results are consistent with the ATE analysis described above.

We compare our system with several of the latest RGB-D SLAM systems designed for dynamic environments:

VO-SF[8] is odometry-based methods. Its idea is similar to ours, it also uses scene flow for dynamic object detection.

Tab.2 Results of metrics ATE

	Without our method		With our method		Improvement	
	RMSE	*S.D.*	*RMSE*	*S.D.*	*RMSE*	*S.D.*
Static	0.405 0	0.148 5	**0.012 1**	**0.005 6**	97.00%	96.22%
xyz	0.588 5	0.508 2	**0.018 2**	**0.009 0**	96.90%	98.22%
rpy	0.914 8	0.452 0	**0.332 3**	**0.232 8**	63.67%	48.49%
Half	0.533 5	0.260 3	**0.167 5**	**0.054 9**	68.65%	78.89%

Tab.3 Results of metrics translational drift

	Without our method		With our method		Improvement	
	RMSE	*S.D.*	*RMSE*	*S.D.*	*RMSE*	*S.D.*
Static	0.579 5	0.403 6	**0.022 5**	**0.005 6**	96.10%	96.76%
xyz	0.836 9	0.472 3	**0.026 3**	**0.009 0**	96.85%	97.37%
rpy	1.368 4	0.779 2	**0.497 8**	**0.232 8**	63.62%	50.47%
Half	0.792 0	0.492 4	**0.284 1**	**0.054 9**	64.16%	63.41%

Tab.4 Results of metrics rotational drift

	Without our method		With our method		Improvement	
	RMSE	*S.D.*	*RMSE*	*S.D.*	*RMSE*	*S.D.*
Static	10.558 9	7.342 3	**0.524 0**	**0.306 4**	95.04%	95.83%
xyz	15.952 1	9.063 4	**0.648 0**	**0.332 7**	95.94%	96.33%
rpy	25.129 4	15.581 0	**9.257 4**	**7.443 8**	63.16%	52.23%
Half	18.497 0	10.966 0	**3.985 4**	**2.294 8**	78.45%	79.07%

MR-DVO[5] detecting the motion of moving objects based on self-motion compensation image difference.

StaticFusion[9] simultaneously estimate the motion of the camera and the probabilistic static/dynamic segmentation of the current RGB-D image pair. Then use this segmentation for weighted dense RGB-D fusion, and use the 3D model for frame-to-model alignment and static/dynamic segmentation.

Tab.5 shows the results of these methods and our method in 3 high dynamic sequences. As we can see, in the static and *xyz* data sets, our method is superior to other methods, its ATE is just around 1—2 cm. But in the fr3/half dataset, the result of our method is not very good, it's around 10 cm away from the best result.

Tab.5 Results of metric ATE

	VO-SF	StaticFusion	MR-DVO	Our method
Static	0.327	0.014	0.065	**0.012**
xyz	0.874	0.127	0.065	**0.018**
Half	0.739	0.391	**0.066**	0.167

Real-time performance is a key indicator for evaluating the SLAM system. Tab.6 shows the time required for some of the main modules in our method. The average time for dynamically judging objects is 0.373 s per frame, which is mainly spent on constructing movement Histogram through the point clouds of the adjacent frames.

Compared with some other methods of detecting dynamic objects, such as DynaSLAM[6], which takes more than 0.5 s per frame to detect dynamic objects, our method is more suitable for real scenes.

Tab.6 Time evaluation

Module	Compute motion vector	Dynamic object judgement	Remove outliers
Time (s)	0.007	0.373	0.026

Figs.3 and 4 show the ATE and RPE diagrams selected from the ORB-SLAM2 without our method and the ORB-SLAM2 with our method with a highly dynamic *xyz* sequence. As we can see, errors in the ORBSLAM2 with our method have been greatly reduced.

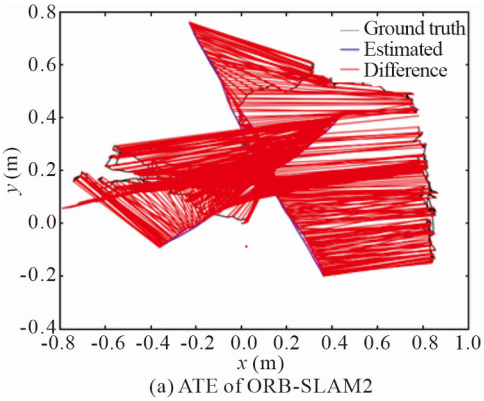
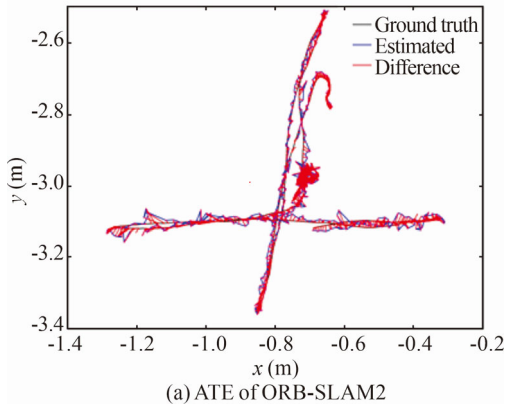


Fig.3 Results from ORB-SLAM2 without our method



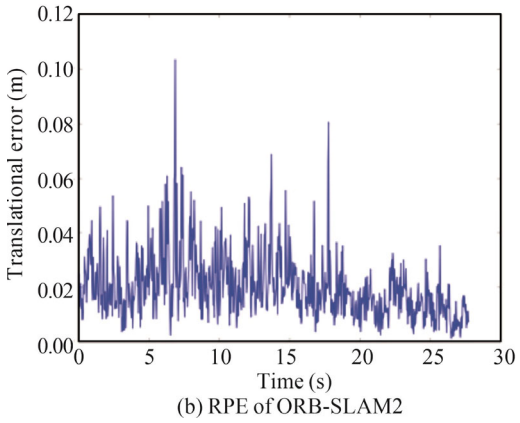


Fig.4 Results from ORB-SLAM2 with our method

In this paper, we propose a method to detect dynamic objects in the scene by using point cloud motion vector information and improve the performance of the SLAM system in a high dynamic scene by filtering out dynamic objects. The method only needs to provide RGB-D data to determine whether there are dynamic objects in the scene. The method has been paralleled into ORB-SLAM2 to act as a preprocessing phase to filter dynamic objects in the image. In the experimental section, the method is evaluated using a high dynamic sequence in the RGB-D dataset. The results show that our method can effectively improve the performance of ORBSLAM2 in various dynamic scenarios. However, our method has some limitations, when the moving object is still, it will reduce the accuracy of detection of the object. Because this method can detect and determine moving objects, but it cannot identify those stationary and possibly moving objects. In the future, we hope to incorporate semantic segmentation methods, by marking those objects that may move as candidates for tracking, if they will move in the future, then we will treat them as dynamic objects.

References

[1] KERL C, STURM J, CREMERS D. Dense visual SLAM for RGB-D cameras[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems, November 3-7, 2013, Tokyo, Japan. New York：IEEE, 2013：2100-2106.

[2] MUR-ARTAL R, TARDOS J D. ORB-SLAM2：an open-source SLAM system for monocular, stereo and RGB-D cameras[J]. IEEE transactions on robotics, 2017, 33(5)：1255-1262.

[3] DEWAN A, CASELITZ T, TIPALDI G D, et al. Motion-based detection and tracking in 3D LiDAR scans[C]//IEEE International Conference on Robotics and Automation, May 16-21, 2016, Stockholm, Sweden. New York：IEEE, 2016：4508-4513.

[4] LU Z, HU Z, UCHIMURA K. SLAM estimation in dynamic outdoor environments[J]. International journal of humanoid robotics, 2010, 7(2)：315-330.

[5] SUN Y, LIU M, MENG M Q H. Improving RGB-D

SLAM in dynamic environments：a motion removal approach[J]. Robotics and autonomous systems, 2017, 89：110-122.

[6] B BESCÓS, JM FÁCIL, CIVERA J, et al. Dynaslam：tracking, mapping and inpainting in dynamic scenes[J]. IEEE robotics and automation letters, 2018, 3(4)：4076-4083.

[7] JIANG C, PAUDEL D P, FOUGEROLLE Y, et al. Static and dynamic objects analysis as a 3D vector field[C]//IEEE International Conference on 3D Vision, October 10-12, 2017, Qingdao, China. New York：IEEE, 2017：234-243.

[8] JAIMEZ M, KERL C, GONZALEZ-JIMENEZ J, et al. Fast odometry and scene flow from RGB-D cameras based on geometric clustering[C]//IEEE International Conference on Robotics and Automation, May 29-June 3, 2017, Singapore. New York：IEEE, 2017：3992-3999.

[9] SCONA R, JAIMEZ M, PETILLOT Y R, et al. Staticfusion：background reconstruction for dense RGB-D slam in dynamic environments[C]//IEEE International Conference on Robotics and Automation, May 21-25, 2018, Brisbane, QLD, Australia. New York：IEEE, 2018：3849-3856.

[10] YANG S, SCHERER S. CubeSLAM：monocular 3D object SLAM[J]. IEEE transactions on robotics, 2019, 35(4)：925-938.

[11] HUANG J, YANG S, MU T J, et al. ClusterVO：clustering moving instances and estimating visual odometry for self and surroundings[C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 13-19, 2020, Seattle, WA, USA. New York：IEEE, 2020：2168-2177.

[12] ZHANG J, HENEIN M, MAHONY R, et al. VDO-SLAM：a visual dynamic object-aware SLAM system[EB/OL]. (2020-05-25) [2021-09-13]. https：//arxiv.org/abs/2005.11052.

[13] BESCOS B, CAMPOS C, TARDÓS J D, et al. Dynaslam II：tightly-coupled multi-object tracking and SLAM[J]. IEEE robotics and automation letters, 2021, 6(3)：5191-5198.

[14] DUBÉ R, DUGAS D, STUMM E, et al. Segmatch：segment based loop-closure for 3D point clouds[EB/OL]. (2019-01-15) [2021-09-13]. https：//arxiv.org/abs/1609.07720v2.

[15] STURM J, ENGELHARD N, ENDRES F, et al. A benchmark for the evaluation of RGB-D SLAM systems[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems, October 7-12, 2012, Vilamoura, Algarve, Portugal. New York：IEEE, 2012：573-580.