# Robust SLAM in Dynamic Environment based on Object's Mask

*Leyuan Sun (University of Tsukuba and AIST), Fumio Kanehiro (University of Tsukuba and AIST)

## 1. Introduction

Most of visual SLAM(Simultaneous Localization and Mapping), which is keypoints based uses corresponding keypoints matching to calculate camera pose. When these points are extracted onto the dynamic object, many outliers occur when the keypoints matched, which results in the inaccuracy of the transformation matrix. When the dynamic object occupies the dominant area of the picture, even the loss of camera pose tracking may occur, such as in Fig.1, the right figure is the camera pose in x/y/z translation. This is the reason that most SLAM solutions must assume that the scene is static.

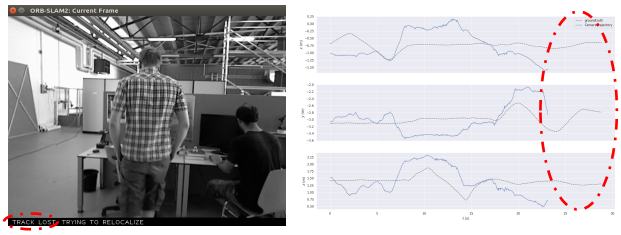


**Fig.1** Track lost in dynamic environment

In this paper, we propose a mask solution to avoid the keypoints be extracted on the dynamic object. Besides, we implement the mask function into the ORB-SLAM2[1] system, in order to make the distribution of keypoints more average, we use a small patch as the sliding window.

One of the related works is called PoseFusion[2], which uses the Elasticfusion[3] as the back-end. The difference between ORB-SLAM2 and EF is that since EF is not optimized for large scenes, it can only be reconstructed in a room-sized environment, and ORB-SLAM2 does not have this limitation.

## 2. Mask generation and implementation

### 2.1 Bool type matrix mask generation

With the rapid development of computer vision and deep learning in recent years, many methods for detecting and segmenting dynamic objects have been proposed. How to apply the results of detection and segmentation in Fig.2 to SLAM in dynamic scenes is very important. The proposed solution is to use a segmented outline to create a binary mask with white pixels inside like Fig.3.

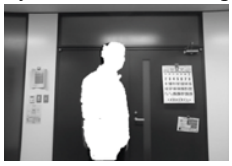
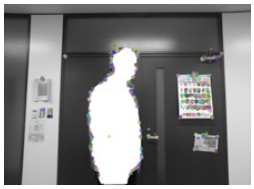


**Fig.2** Segmentation result

**Fig.3** Binary outline

If we use this kind of mask to do the keypoints extractor, due to the Oriented FAST(Feature from Accelerated Segment Test)[4] detection algorithm, the surrounding of the white boundary is detected as a keypoint in Fig.4. Under this circumstance, although the matching is made without the keypoints inside the dynamic object, the keypoints around the boundary still generate a large number of outliers in Fig.5.



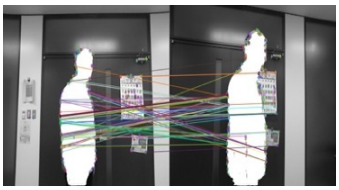


**Fig.4** Keypoints extracted on mask boundary (left)
**Fig.5** Keypoints matching (right)

In order to solve out this problem, we use the dilate function to scale up the mask in Fig.6, then reverse the black(0) and white(1). Actually, the mask is a bool type matrix, the ROI (region of interest) is '1' area in Fig.7, which means each time we do the keypoints extracting the function need to check this area whether belongs to ROI or not. After this process, the matching result is better than before in Fig.8.

**Fig.6** Dilate the mask.

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |

**Fig.7** Bool type matrix

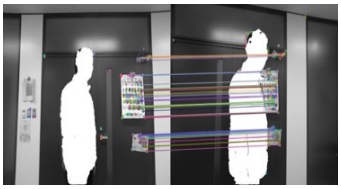


**Fig.8** Keypoints matching with mask

### 2.2 Implement in the ORB-SLAM2

In order to make the mask implementation into ORB-SLAM2 system, we need to change the Oriented FAST extractor in ORB-SLAM2, whose definition in *ORB-SLAM2/include/ORBextractor.h* expresses that mask feature is not included. Our solution is using OpenCV *detectAndCompute()* function to detect the same kind of keypoints with the mask feature to replace it.

However, as we tested, the difference between the OpenCV extractor and ORB-SLAM2 extractor is that based on the former, the extracted points will be relatively concentrated in the areas with obvious features in Fig.20, compared with the average distribution in Fig.19. If only

these areas are used, part of the information of the image is lost, resulting in a decrease in the accuracy of the camera pose in the most situation, details absolute trajectory error which is recommend by [5] for the evaluation of SLAM system, has been shown in the table.1.

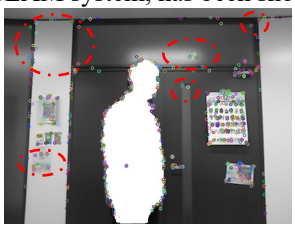


**Fig.9** ORB-SLAM2 extractor **Fig.10** OpenCV extractor

| Sequence | ORB-SLAM2 | OpenCV |
|---|---|---|
| fr1_xyz | 0.0135 | **0.0120** |
| fr1_desk | **0.0193** | 0.0219 |
| fr2_3h | 0.1128 | **0.0928** |
| fr2_desk | **0.0072** | 0.0145 |
| fr3_loh | **0.0098** | 0.0159 |
| fr3_sh | **0.0310** | 0.1958 |

**Table.1** ATE comparison between ORB-SLAM2 extractor and OpenCV extractor (freiburg1_xyz(fr1_xyz), freiburg1_desk(fr1_desk), freiburg2_360_hemisphere(fr2_3h), freiburg2_desk(fr2_desk), freiburg3_long_office_household (fr3_loh), freiburg3_sitting_halfsphere(fr3_sh))[6]

Considering the above mentioned, the objective is to keep the mask function and make the keypoints average distribution at the same time. The solution is to both make the RGB and mask image split into 30*30 pixels grid as the sliding window to extract the keypoints in each grid with the mask condition as Fig.13. The final keypoints extracted results is in Fig.14, where no points inside or around the mask and average distribution as well.

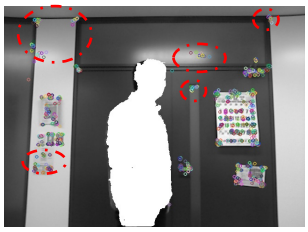



**Fig.13** Split the image and mask into 30*30 patch
**Fig.14** Keypoints extracted result, combine the mask and average distribution

## 3. Experimental results

| Dataset | SF | EF | CF | PF | Orb2 | **proposed** |
|---|---|---|---|---|---|---|
| fr3_walk_xyz | 0.35 | 0.22 | 0.37 | **0.0321** | **0.37** | **0.0207** |

**Table.2** ATE comparison between ORB-SLAM2

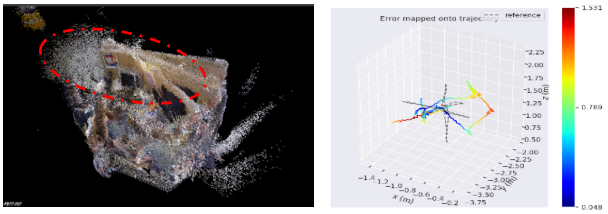


**Fig.15** Point cloud map and camera trajectory(ORB2)

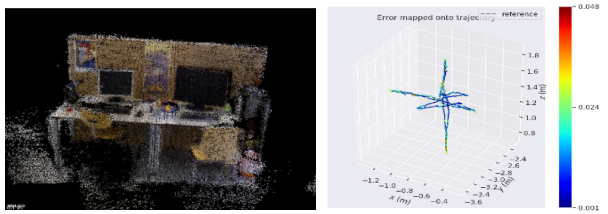


**Fig.16** Point cloud map and camera trajectory(proposed)

We use the Mask R-CNN[7] as the segmentation method to segment out human ahead. As we can see, in the left point cloud map of Fig.15, due to the inaccuracy of camera trajectory, the point clouds belonged to backboard have projected into different angles. In contrast, our proposed method has better performance both in point cloud and camera trajectory in Fig.16. Besides, the Table.2 has shown that the proposed method in ATE is smaller than the PF and other state-of-art method.

## 4. Conclusion and Future work

In this paper, we propose a scheme to use the segmentation results as the input to the ORB-SLAM2 when SLAM is in dynamic environment. The solution is using the mask of the dynamic object to let the keypoints not be extracted inside and around the dynamic object's outline. At the same time, we use the sliding window strategy to keep the advantages that keypoints average distribution in original ORB-SLAM2 extractor.

In the future, we would like to test different kinds of dynamic object segmentation methods which need to satisfy the real-time rate of SLAM in dynamic environment.

## REFERENCES


[1] R. Mur-Artal and J. D. Tardo´s, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," IEEE T-RO, 2017

[2] Tianwei Zhang, Yoshihiko Nakamura. PoseFusion: Dense RGB-D SLAM in Dynamic Human Environ- ments. 2018 International Symposium on Experimental Robotics, Nov 2018, Buenos Aires, Argentina. 2018

[3] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison, "ElasticFusion: Dense SLAM without a pose graph", Robotics: Science and Systems (RSS), 2015.

[4] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In European Conference on Computer Vision, volume 1, 2006.

[5] E. Olson and M. Kaess, "Evaluating the performance of map optimization algorithms," in RSS Workshop on Good Experimental Methodology in Robotics, 2009.

[6] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in IROS, 2012

[7] Kaiming He, Georgia Gkioxari, Piotr Dollar, Ross Girshick; The IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2961-2969