DYG-SLAM: A Robust Visual SLAM in Dynamic Scenes using YOLOv3-tiny and Geometry Constraints

Zhenwei Cui, Tianhong Pan*, Qionghua Wang
School of Electrical Engineering and Automation, Anhui University, Hefei, Anhui 230601, China.

Abstract: Simultaneous Localization and Mapping (SLAM) has been widely applied in the intelligent robot. However, classical visual SLAM (VSLAM) systems commonly work well in static environments, and whose performance may deteriorate in dynamic environments. To solve this problem, a dynamic object removal method combining YOLOv3-tiny and geometry constraints is proposed in this study, which can accurately locate dynamic objects and retain as many static feature points in the dynamic regions as possible. First, dynamic and static feature points coarse division strategy is designed, which uses YOLOv3-tiny to divide the image frames into static and dynamic regions, and then classify feature points into corresponding regions. Moreover, the modified geometry constraints are proposed to further process the feature points in dynamic regions. Combining epipolar constraints and Kalman filter together, the presented method can accurately distinguish dynamic and static feature points, and track the dynamic feature points in the dynamic regions. The presented method is integrated into the front-end of ORB-SLAM2 system, which is a pre-processing procedure to reduce the impact of dynamic objects on the system. Experimental results on the TUM RGB-D dataset and Bonn RGB-D dataset demonstrate that the proposed method is able to significantly improve the performance of the SLAM system in a highly dynamic environment and achieve real-time performance.

Key-Words : *Visual SLAM, Dynamic environments, YOLOv3-tiny, Geometry constrains, Kalman filter*

1. Introduction

Simultaneous Localization and Mapping (SLAM) is a core technology for the intelligent robot, which can simultaneously estimate its position and build an environmental map in an unknown environment. Visual SLAM(VSLAM) which uses a camera as the perception sensor, has become an important research topic in recent years due to its low cost, high accuracy, and rich information. In the past few decades, many excellent VSLAMs have been emerged, such as ORB-SALM2[1] , RGBD-SLAM[2] , LSD-SLAM[3] .

However, classical VSLAM systems commonly work well in static environments, whose performance will deteriorate in the environment containing dynamic objects (such as people and cars). The key procedure of VSLAM in dynamic regions is to accurately locate the position of dynamic objects and remove them. At present, many VSLAM systems dealing with dynamic scenes use object detection networks [4] to locate the position of dynamic objects and eliminate them. However, many static feature points will be eliminated by mistake when the dynamic regions obtained by the object detection network are directly eliminated. As a result, the system's accuracy will be reduced, or even the system fails to track.

Therefore, a new VSLAM system named DYG-SLAM is proposed in this work for dynamic scenes based on ORB-SLAM2. The proposed method

Received: 2022/11/02, Accepted: 2022/12/23
*Corresponding author: Tianhong Pan
E-mail address: thpan@ahu.edu.cn

combines the YOLOv3-tiny[5] object detection network and a modified geometry constraints to eliminate dynamic objects. YOLOv3-tiny is used to determine the position of dynamic objects in the image, and divide the image into dynamic and static regions. The geometry constraints combine GC-RANSAC (Graph-Cut Random Sample Consensus)[7] algorithm and Kalman filter[8] to further distinguish points between the dynamic and static features in dynamic regions. The proposed method can not only accurately locate the position of dynamic objects, but also retain as many static feature points in the dynamic regions as possible. DYG-SLAM is evaluated by using multiple datasets and compared with state-of-the-art systems. Experimental results show that the presented method can improve the robustness and accuracy of SLAM system in dynamic scenes. The main contributions are summarized as follows:

(1) a complete real-time VSLAM system has been constructed for dynamic environments.

(2) a method that combines object detection and geometry constraints has been proposed to effectively eliminate dynamic objects.

(3) a method calculating the geometry probability of feature points in dynamic regions has been presented by combining GC-RANSAC and Kalman Filter.

2. Method

2.1 Overview of SLAM

The proposed method can be regarded as the front-end preprocessing stage of the SLAM system, which can effectively filter out dynamic feature points and preserve the moving probability of feature points in dynamic regions. Firstly, the object detection network is used to determine the position of dynamic objects in the image and divides the image into static and dynamic regions. Then, the modified geometry constraints combining GC-RANSAC and epipolar constraints is used to calculate the fundamental matrix of static regions and the geometry probability of feature points in the dynamic regions. Afterwards, the geometry probability of the feature points of previous frame image is considered as the prior probability. Based on Kalman Filter, the final geometry probability of the matching feature points in the dynamic regions of current frame can be obtained by fusing the prior probability and the geometry probability calculated by the geometry method. Next, the presented method expanded the probability of dynamic regions matching feature points to the unmatched feature points. Finally, the feature points with high dynamic probability (greater than 0.5) are eliminated. The presented dynamic feature points removal method will be embedded in the front-end of ORBSLAM2, which can effectively improve the accuracy of SLAM system. The framework of DYG-SLAM is shown in Figure 1. In the yellow block, the green points on the right image are the static feature points and the red feature points in the red bounding box are dynamic feature points after the first dynamic and static feature points division. In the purple block, it is mainly used to complete the task of calculating the fundamental matrix of static regions and the geometry probability of matching feature points in the dynamic regions. In the orange block, the yellow points are the static feature points and the red feature points are dynamic feature points after the probability propagation and expansion.

2.2 Object Detection

In order to determine the position of dynamic objects in the image, the presented method achieves the task using YOLOv3-tiny. YOLOv3-tiny was trained with the Crowdhuman dataset[9] , which can accurately detect the position of people in dense crowds[10] . The YOLOv3-tiny network trained on this dataset has an accuracy similar to YOLOv3 trained with MS

COCO[11] , and is more efficient than YOLOv3, with an efficiency of up to 50 FPS when running on CPU. It receives an RGB image and outputs the bounding box size, position and class. After determining the position of the dynamic objects through the position of bounding box, the presented method divides the image into static regions and dynamic regions.

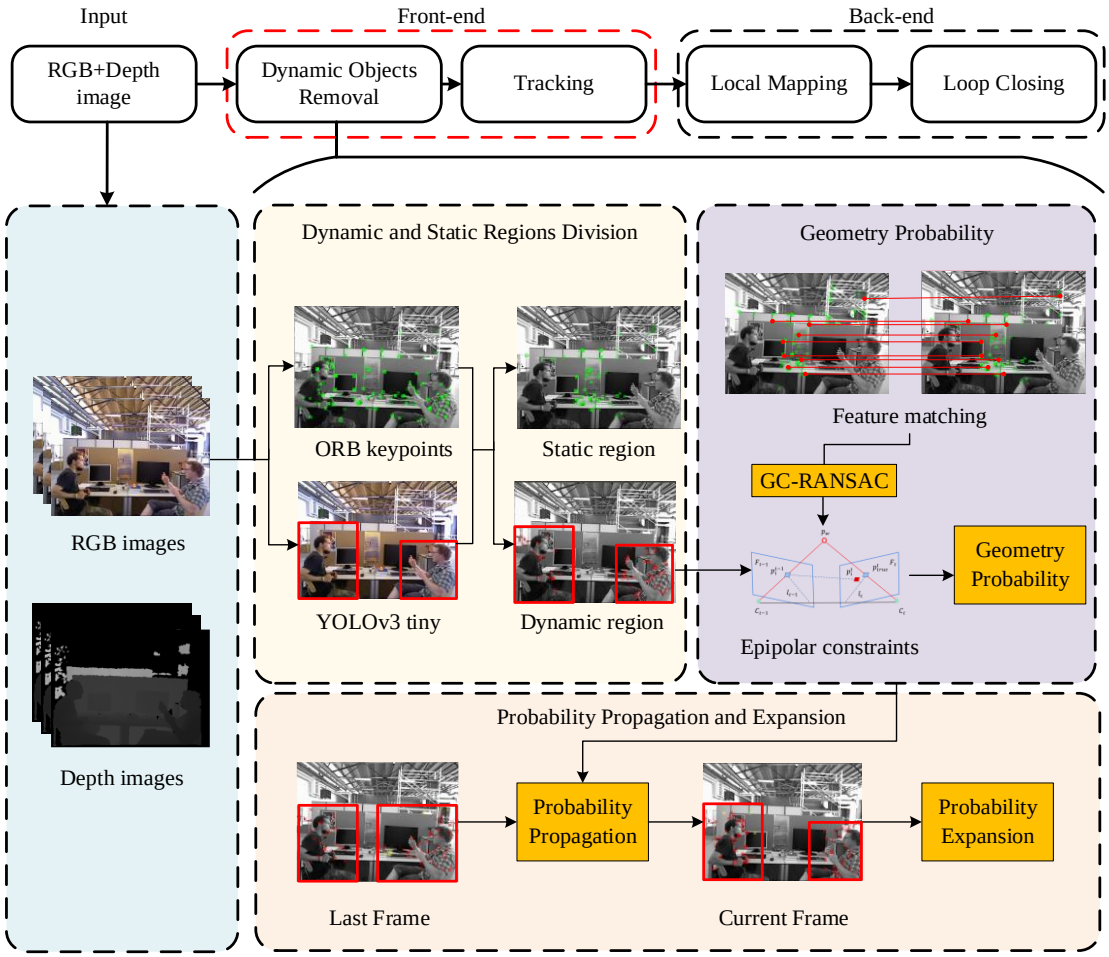


Fig.1 Framework of DYG-SLAM

2.3 Geometry Probability

The most traditional geometry methods only judge whether the feature points are static or dynamic according to the distance of geometry constraints. However, it does not divide dynamic and static objects very well. At the same time, the RANSAC (Random Sample Consensus) algorithm takes a long time to calculate the fundamental matrix *F* and the accuracy is not high. To solve these problems, a geometry constraints method using a new geometry probability is presented to divide the static and dynamic feature points.

The presented method combines the GC-RANSAC エラー! 参照元が見つかりません。 algorithm and the epipolar constraints to calculate the geometry probability of matching feature points in dynamic

regions.

2.3.1 GC-RANSAC Algorithm

The GC-RANSAC algorithm is a RANSAC-type robust estimator and is far superior to RANSAC in model estimation accuracy and computational speed.

Given a point set $M=\left\{\left(p_i^t,p_i^{t-1}\right)\middle|i=1,\ldots,n\right\}$ consisting of matching pairs of 2D and 2D points. The GC-RANSAC algorithm is used to calculate the fundamental matrix F between two consecutive frames by optimizing the energy function $E(L)=\sum_i B(L_i)+\lambda\sum_{(i,j)\in G}R(L_i,L_j)$ with $L=\left\{L_i\in\{0,1\}\middle|i=1,\ldots,n\right\}$ being a label assignment for the matching pair set M, and G being a neighbor graph. The unary term of the energy function is formulated as:

$$B(L_i)=\begin{cases}1-K\left(\phi\left(p_i^t,p_i^{t-1},\theta\right),\varepsilon\right) & if \quad L_i=1\\ K\left(\phi\left(p_i^t,p_i^{t-1},\theta\right),\varepsilon\right) & if \quad L_i=0\end{cases} \tag{1}$$

where θ is the angular parameter for fundamental matrix F, and $K(\sigma,\varepsilon)=\exp\left(-\sigma^2/\left(2\varepsilon^2\right)\right)$. $L_i=1$ stands for an outlier pair and $L_i=0$ stands for an inlier pair. $\phi\left(p_i^t,p_i^{t-1},\theta\right)$ is the distance from matching pair $\left(p_i^t,p_i^{t-1}\right)$ the fundamental matrix F, and ε is the threshold for outlier/inlier determination. The pairwise energy term is defined as:

$$R\left(L_i,L_j\right)=\begin{cases}1 & if\ L_i\neq L_j\\ \left(B(L_i)+B\left(L_j\right)\right)/2 & if\ L_i=L_j=0\\ 1-\left(B(L_i)+B\left(L_j\right)\right)/2 & if\ L_i=L_j=1\end{cases} \tag{2}$$

In this workd, $\lambda=0.14$ and $\varepsilon=0.1$. The total energy can be efficiently optimized by the graph cut algorithm[12] .

2.3.2 Geometry Probability Algorithm

Firstly, the GC-RANSAC algorithm is used to calculate the fundamental matrix F by matching feature points in two consecutive frames of static regions. The epipolar constraints for matching feature points in two consecutive frames is shown in Figure 2.

F_{t-1} and F_t represent two consecutive frames of images, p_i^t and p_i^{t-1} represent a pair of matching feature points, p_{true}^t is the true corresponding feature point to p_i^{t-1}, p_w is the world coordinate point corresponding to p_{true}^t and p_{true}^t, C_{t-1} and C_t are the camera poses at different moments.

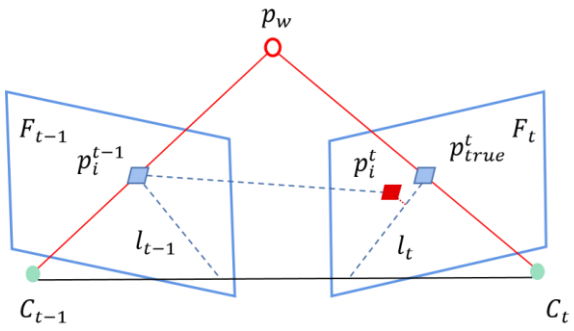


Fig.2 Epipolar constraints

Secondly, the epipolar line l_t for matching feature points in dynamic regions can be calculated from F by:

$$l_t=Fp_i^{t-1} \tag{3}$$

where $l_t=[A,B,C]^T$, $[A,B,C]^T$ is the vector representation of the epipolar line.

Then, the distances between the matching feature points in the dynamic region and the corresponding epipolar line is calculated as:

$$\phi\left(p_i^t,p_i^{t-1},\theta\right)=\frac{\left|\left(p_i^t\right)^T Fp_i^{t-1}\right|}{\sqrt{\|A\|^2+\|B\|^2}} \tag{4}$$

where p_i^t, p_i^{t-1} use the homogeneous coordinates of the feature points.

Eventually, the distances obtained by epipolar constraints can be converted to geometry probability by the following a binary sigmoid function:

$$P\left(p_i^t\right)=\frac{1}{1+C_1*\exp\left(-\left(\phi\left(p_i^t,p_i^{t-1},\theta\right)-\alpha\right)\right)} \tag{5}$$

where $P\left(p_i^t\right)$ represents the geometry probability of p_i^t, C_1 is a constant, α is the distance threshold.

2.4 Probability Propagation Algorithm

Most of the traditional geometry or deep learning-based SLAM only considers the connection of two consecutive frames, ignores the connection of nsecutive multiple frames. Therefore, we assume that e motion between two consecutive frames satisfies the Markov property[13] , and proposes a probability propagation method based on Kalman filtering.

There is an error in the feature point matching between the two frames, which can be understood as a state transition error. There is an error in the calculation of the fundamental matrix F , which can be understood as an observation error. We assume that the probability propagation between two matching feature points conforms to a Gaussian distribution. So, probability propagation method can be expressed as follows:

Suppose the variance of state transitions is Q and the variance of observations is R .

Prediction step:

$$P\left(p_i^t\right)^- = P\left(p_i^{t-1}\right)^+ \tag{6}$$

$$Q_t^- = Q_{t-1}^+ + Q \tag{7}$$

Kalman Gain:

$$K = \frac{Q_t^-}{Q_t^- + R} \tag{8}$$

Update step:

$$ATE_i = E_i^{-1} TG_i \tag{9}$$

$$Q_t^+ = (1-K) * Q_t^- \tag{10}$$

where $t-1$ and t represent two consecutive moments, the symbols + and - denote the posterior and prior probabilities. Finally, $P\left(p_i^t\right)^+$ is taken as the final geometry probability of p_i^t .

Because the feature points in the dynamic regions of the current frame are not all matched with the previous frame, it is necessary to extend the geometry probability of the matching feature points p_i^t in the dynamic regions to the unmatched feature points p_j^t . The geometry probability of updating the unmatched point according to the distance between the unmatched feature points and the matched feature points[14] . The update formula is as follows:

$$P\left(p_j^t\right) = P_{init} + \sum_{p_i^t \in D_t^m} \lambda(d)\left(P\left(p_i^t\right) - P_{init}\right) \tag{11}$$

$$\lambda(d) = \begin{cases} C_2 * \exp\left(-\frac{d}{\delta}\right) & if \quad d \le \delta \\ 0 & otherwise \end{cases} \tag{12}$$

where D_t^m represents a set of matching feature points in the dynamic regions, P_{init} is the initial value of the unmatched feature points, C_2 is a constant, d represents the Euclidean distance between matched and unmatched feature points, δ is the distance threshold.

3. Experiments and Results

In this section, the performance of DYG-SLAM towards dynamic scenes will be introduced in detail. We employ ORB-SLAM2 as the basic SLAM system. The proposed dynamic regions removal approach is integrated into the front-end of ORB-SLAM2. The presented method as a pre-processing to eliminate feature points in dynamic regions. The property of DYG-SLAM is evaluated on the public TUM dataset[15] . Furthermore, DYG-SLAM is compared with current state-of-the-art systems to verify the superiority. DYG-SLAM is implemented on a desktop PC with 3.5GHz Intel i7-7500U CPU, 8GB RAM, and a NVIDIA GeForce 940MX running Ubuntu Linux 18.04 LTS. All tests were performed five times and median values were used for evaluation. In the experiment, C_1, α, Q, R, P_{init}, C_2 and δ are set to 2, 0.4, 0.09, 1, 0.6, 1, 60 respectively.

3.1 Performance evaluation on TUM RGB-D Dataset

The TUM RGB-D dataset was proposed by the TUM Computer Vision Group in 2012, which is frequently used in the SLAM domain. It contains sequence of RGB images and depth images from a

Microsoft Kinect camera, with their corresponding ground truth trajectories. The data was recorded at 30Hz with a 640 x 480 resolution. There are mainly two types of sequences used in our experiments. In the *fr3_w* sequences, two people walking through office scene. And the *fr3_s* can be considered low-dynamic which contain two people sitting in front of a desk and moving a little bit occasionally. The feature points extraction results of ORB-SLAM2, Crowd-SLAM and DYG-SLAM on *fr3_w_xyz* sequences were shown in Figure 3. It is obviously seen that the presented method can not only label dynamic regions, but also preserve as many static feature points in dynamic regions as possible.

Estimated trajectories need to be compared with ground truth trajectories to evaluate the SLAM system. The Absolute Trajectory Error (ATE) and the Relative Pose Error (RPE) are often used as an error measure. The ATE is used to evaluate the global consistency of the estimated trajectory, while the RPE is used to evaluate the translation and rotation drift. The root mean square error (RMSE) and standard deviation error (SD) of ATE are applied as the performance indexes. The RMSE stands for the deviation of the estimated value from the true value. SD reflects the dispersion degree of the estimated camera trajectory.

The ATE at a certain moment were as follows:

$$ATE_i = E_i^{-1} T G_i \tag{13}$$

where E is the estimated trajectory, G represents the ground truth, and T is the transformation that aligns the two trajectories, Δ represents time interval. For a sequence of N poses, the RMSE of ATE can be calculated using the following formula:

$$RMSE\left(ATE_{1:N}\right) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left\|trans\left(ATE_i\right)\right\|^2} \tag{14}$$

Compared with the original ORB-SLAM2, the percentage of DYG-SLAM has been improved as follows:

$$\gamma = \left(1 - \frac{\eta}{\mu}\right) \times 100\% \tag{15}$$

where γ represents the improvement of DYG-SLAM compared with ORB-SLAM2, η represents the experiment of DYG-SLAM, μ is the result of ORB-SLAM2.

The performance improvements brought by DYG-SLAM were shown in Table 1. The predicted trajectories of the *fr3_w_static*, *xyz*, *rpy*, *halfsphere*, and *fr3_s_static* sequences were plotted in Figure 4-9. The superiority of DYG-SLAM can be clearly observed by the comparison with ORB-SLAM2 and Crowd-SLAM.

As shown in Table 1, compared with ORB-SLAM2 and Crowd-SLAM, the RMSE and SD of presented algorithm achieved 97.4% and 96.36% in high dynamic sequences and 38.27% and 59.06% in low dynamic sequences. The comparisons among Ai et al.[16] and YOLO-SLAM[17] in the form of a bar chart were shown in Figure 9. DYG-SLAM outperformed the above-mentioned methods on most sequences. Although DYG-SLAM performs well on high dynamic sequences, which does not perform well on low dynamic sequences. The reason is that the dynamic objects removal method used in the dynamic scenes with low dynamic sequences will erroneously identify dynamic feature points as static feature points and retain them.

3.2 Performance evaluation on Bonn RGB-D Dataset

The Bonn RGB-D dynamic dataset was released by Photogrammetry & Robotics Lab of Bonn University in 2019, which includes 24 highly dynamic scenes, where people perform different tasks, such as walking, manipulating boxes or playing with balloons. It was recorded with an ASUS Xtion Pro LIVE sensor, combined with an Optitrack Prime 13 motion capture system for the ground truth trajectories. In this experiment, five sequences were

selected for performance evaluation. The "*crowd*" sequences represent three people walking around the room at random. The "*synchronous*" sequences represent two people moving at the same speed and direction. These sequences are a big challenge to the traditional SLAM, but DYG-SLAM works well. Figure 10 shows the feature points extraction results of ORB-SLAM2, Crowd-SLAM and DYG-SLAM on *crowd1* sequences. The red bounding boxes are the position of dynamic regions identified by YOLOv3-tiny. Obviously, the presented method can retain a number of static feature points in the dynamic regions.

To further evaluate the performance of DYG-SLAM, the ORB-SLAM2 and Crowd-SLAM systems were chosen as the comparison. The evaluation metrics are the same as the TUM dataset. ORB-SLAM2 systems have large errors in all sequences, however, DYG-SLAM can guarantee both accuracy and robustness.

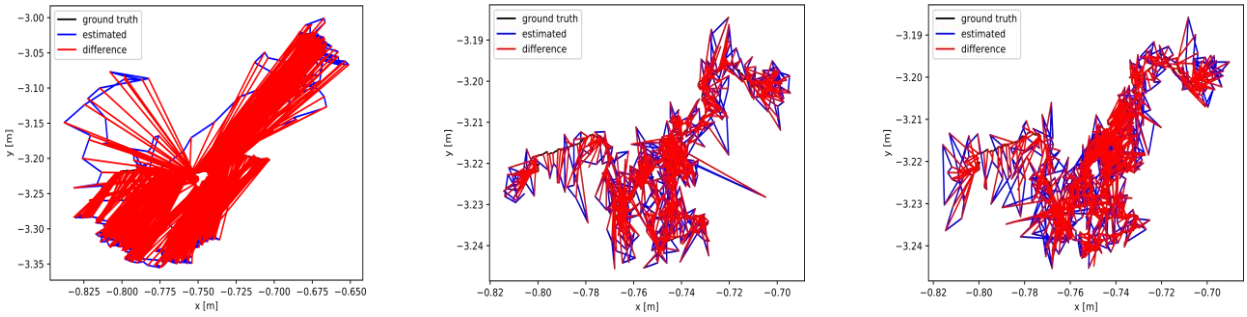
The trajectory errors of DYG-SLAM on the Bonn dataset were shown in Table 2. Figure11-14 represent the ATE plots from ORB-SLAM2, Crowd-SLAM, and DYG-SLAM for the crowd and synchronous sequences. DYG-SLAM outperforms ORB-SLAM2 on all datasets. DYG-SLAM outperforms Crowd-SLAM on crowd sequences, but it is inferior to Crowd-SLAM on synchronous sequences.

3.3 Computational time comparison

In real-world application, the real-time performance of SLAM system is a very important evaluation index. Table 3 shows the average time-comsuming (ms) of ORB-SLAM2 and DYG-SLAM run on Bonn sequences using only CPU. It is obvious from the experimental results that the time-consuming of DYG-SLAM is mainly on the Object detection and ORB feature matching in the dynamic objects removal. Time-consuming can be reduced by replacing better and faster feature matching algorithms, for instance, DynaSLAM[18] achieved a mean performance of 1.35 FPS on *crowd1* with GPU. Although DYG-SLAM is currently slower than ORB-SLAM2, it also far outperforms far outperforms those SLAM systems that require GPUs acceleration. Therefore, the presented method not only has high accuracy in high dynamic environment, but also achieved an average frame rate of 19 FPS with CPU.

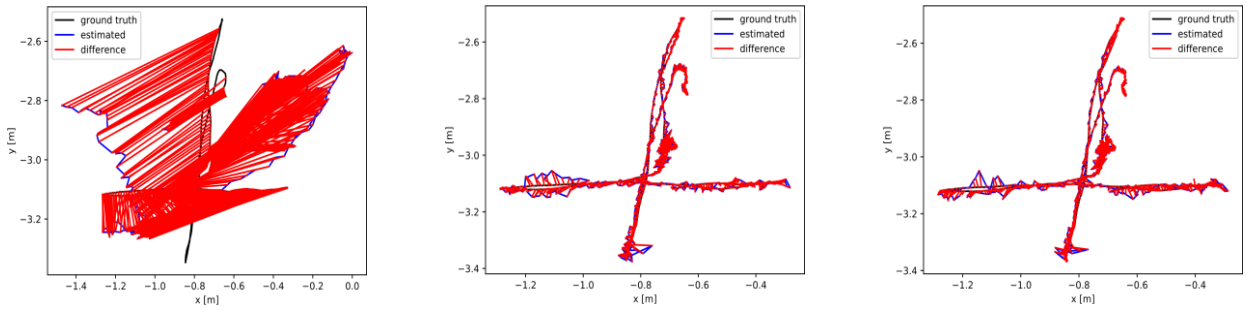
(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Fig.3 Feature points extraction using ORB-SLAM2, Crowd-SLAM and DYG-SLAM on *fr3_w_xyz*.



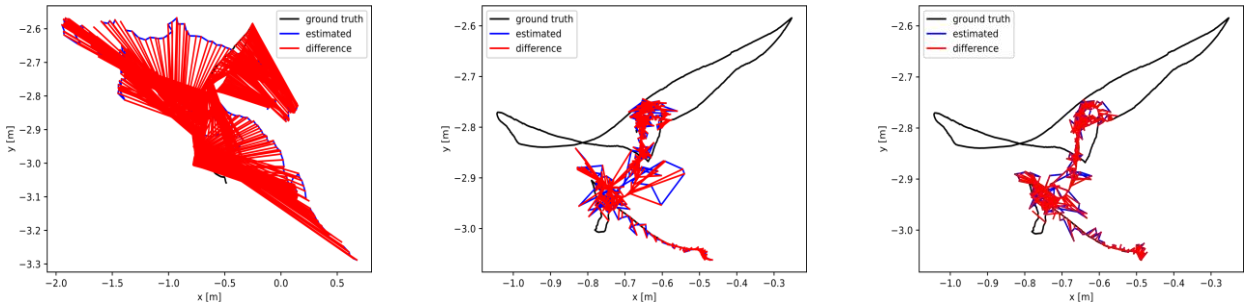
(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Fig.4 Ground truth and estimated trajectory in the sequence fr3_walking_static



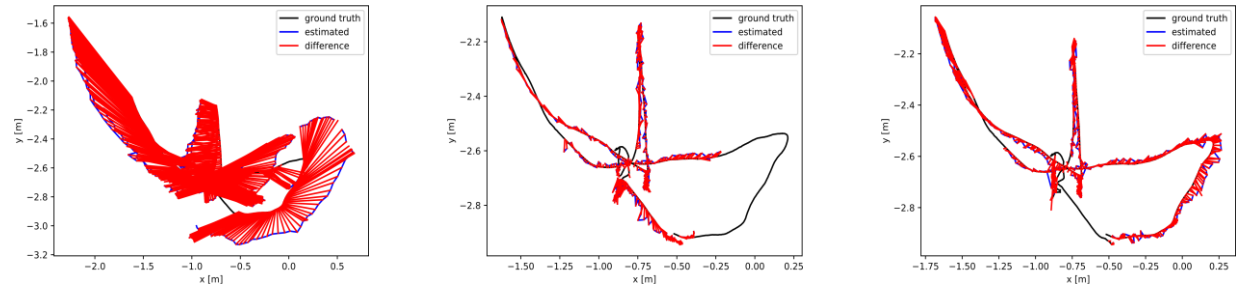
(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Fig.5 Ground truth and estimated trajectory in the sequence fr3_walking_xyz



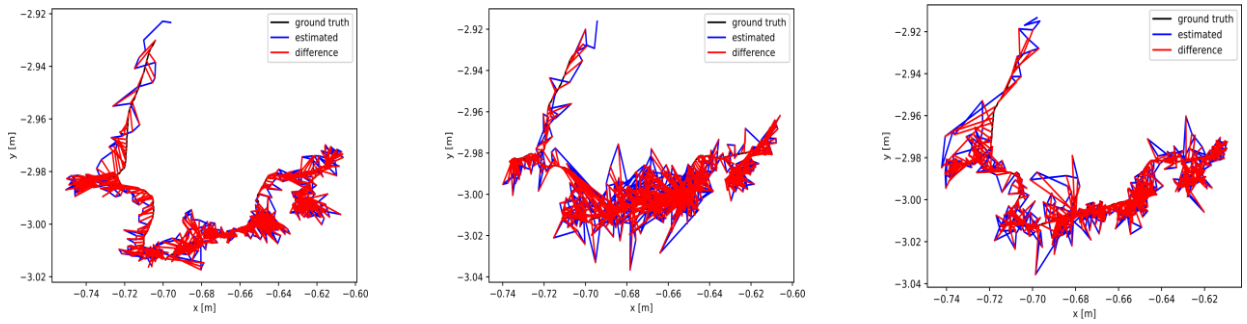
(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Fig.6 Ground truth and estimated trajectory in the sequence fr3_walking_rpy



(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Fig.7 Ground truth and estimated trajectory in the sequence fr3_walking_halfsphere



(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Fig.8 Ground truth and estimated trajectory in the sequence fr3_sitting_static

Table 1 ATE comparison using DYG-SLAM, Crowd-SLAM and ORB-SLAM2.

Sequences	ORB-SLAM2		Crowd-SLAM		DYG-SLAM		Improvements	
	RMSE	SD	RMSE	SD	RMSE	SD	RMSE	SD
fr3_walking_static	0.4111	0.1312	**0.0077**	**0.0038**	0.0107	0.0056	**97.40%**	**95.73%**
fr3_walking_xyz	0.6598	0.2558	0.0185	0.0100	**0.0172**	**0.0093**	**97.39%**	**96.36%**
fr3_walking_rpy	0.6613	0.23782	0.0403	0.132	**0.0351**	**0.0236**	**94.69%**	**90.08%**
fr3_walking_half	0.5557	0.2102	0.0275	0.0148	**0.0263**	**0.0129**	**95.27%**	**93.86%**
fr3_sitting_static	0.0084	0.0041	0.0105	0.0057	**0.0059**	**0.0027**	**29.76%**	**34.15%**
fr3_sitting_xyz	0.0087	0.0042	0.0196	0.0104	0.0154	0.0076	-77.01%	-80.95%
fr3_sitting_rpy	0.0196	0.0127	0.0168	0.0095	**0.0121**	**0.0052**	**38.27%**	**59.06%**
fr3_sitting_half	0.0175	0.0106	0.0219	0.0114	**0.0154**	**0.0078**	**12.00%**	**26.42%**

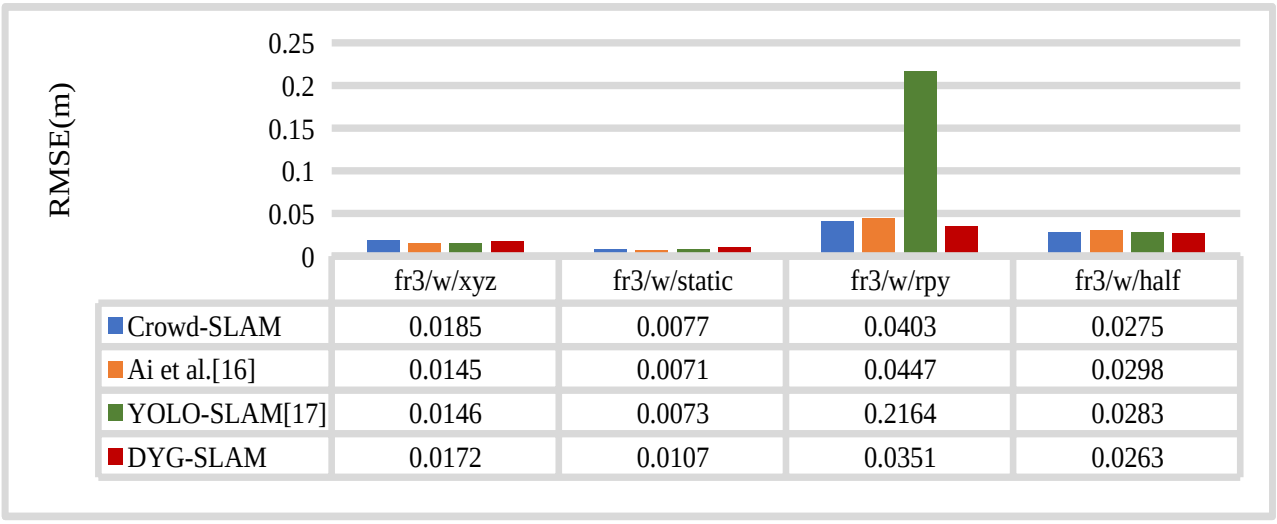


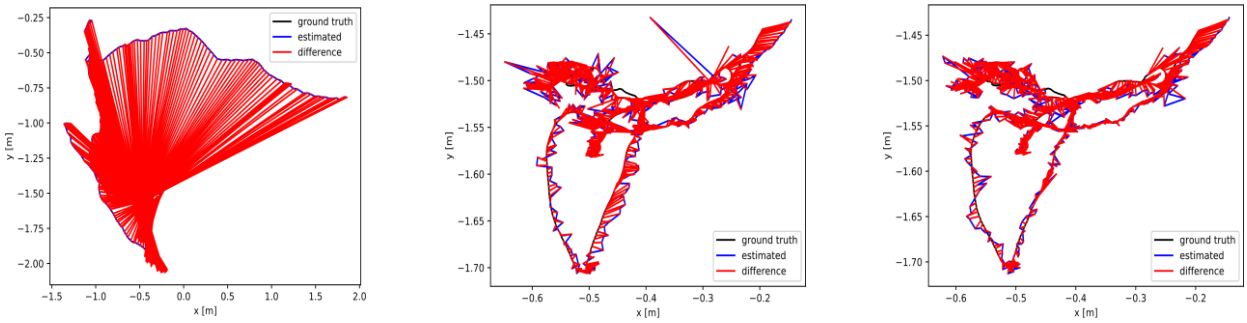
Fig.9 RMSE of ATE

(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Figure 10 Feature points extraction results of ORB-SLAM2 Crowd-SLAM and DYG-SLAM on *crowd1*.

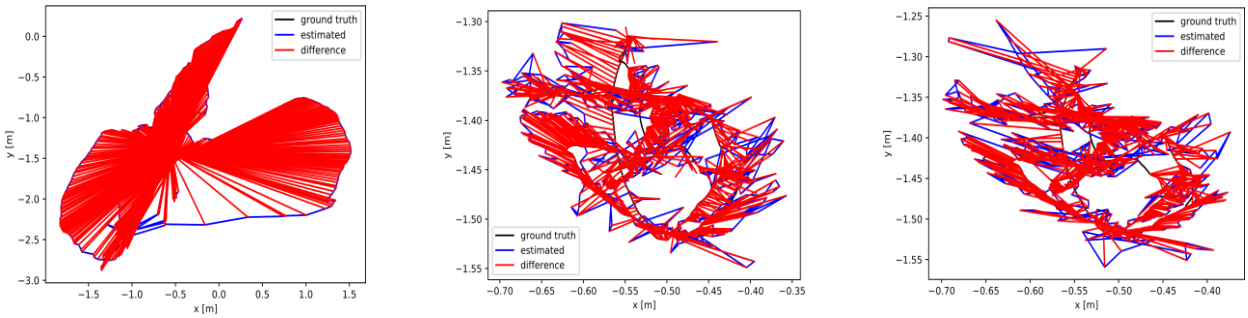
Table 2 ATE comparison using DYG-SLAM, Crowd-SLAM and ORB-SLAM2.

Sequences	ORB-SLAM2			Crowd-SLAM			DYG-SLAM			Improvements		
	RMSE	Mean	S.D.	RMSE	Mean	S.D.	RMSE	Mean	S.D.	RMSE	Mean	S.D.
crowd1	0.9195	0.737	0.5498	0.0365	0.0215	0.0295	**0.0269**	0.0204	**0.0175**	**97.07%**	97.23%	**96.82%**
crowd2	1.3606	1.2216	0.5991	0.0377	0.0319	0.0201	**0.0324**	0.0272	**0.0176**	**97.62%**	97.77%	**97.06%**
crowd3	1.1019	1.0269	0.0263	0.044	0.0351	0.0264	**0.0438**	0.0352	**0.0262**	**96.03%**	96.57%	**0.38%**
synchronous1	1.1892	1.0306	0.5934	**0.1767**	0.1077	0.1401	0.2791	0.2710	**0.0669**	**76.53%**	73.70%	**88.73%**
synchronous2	1.5847	1.4986	0.5152	**0.0189**	0.0096	**0.0162**	0.1703	0.1454	0.0888	**89.25%**	90.30%	**82.76%**



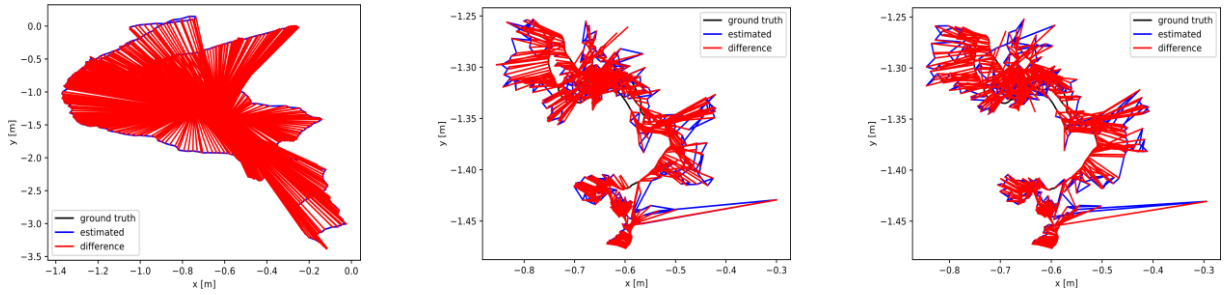
(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Figure 11 Ground truth and estimated trajectory in the sequence crowd1



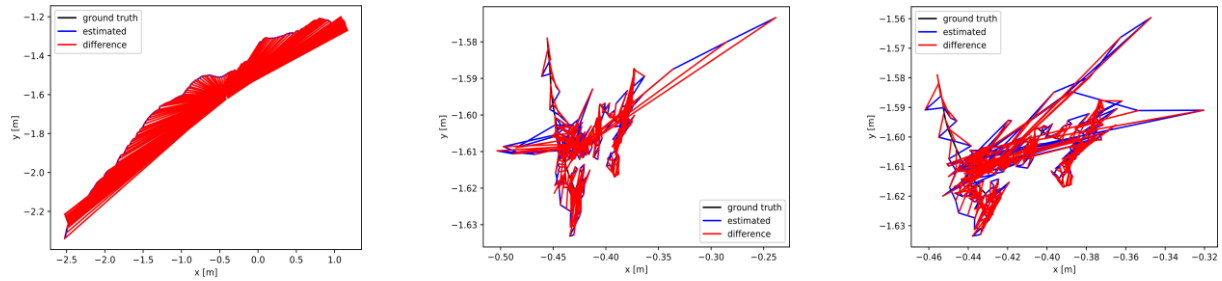
(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Figure 12 Ground truth and estimated trajectory in the sequence crowd2



(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Figure 13 Ground truth and estimated trajectory in the sequence crowd3



(a) ORB-SLAM2 (b) Crowd-SLAM (c) DYG-SLAM

Figure 14 Ground truth and estimated trajectory in the sequence synchronous2

Table 3 Comparison of the average time-consuming(ms) between the ORB-SLAM2 and DYG-SLAM

Systems	ORB feature extraction	Object detection	Dynamic objects removal		Tracking	Total time
			ORB feature matching	Update geometry probability		
ORB-SLAM2	21.36	-	-	-	16.56	**43.07**
DYG-SLAM	21.52	16.87	5.61	3.65	7.30	52.43

4. Conclusion

In this work, an efficient, accurate and robust VSLAM system was proposed, which is built on the top of ORB-SLAM2 to eliminate the influence of dynamic objects. The core of DYG-SLAM is a highly efficient dynamic feature points filtering method, including a method combining object detection and traditional geometry methods and a geometry probability calculation method combining GC-RANSAC and Kalman Filter.

DYG-SLAM was evaluated on challenging dynamic sequences of the TUM RGB-D dataset and Bonn RGB-D dataset. DYG-SLAM achieves 90% improvement over the original ORBSLAM2 on most sequences and also outperforms the state-of-the-art Crowd-SLAM. Experimental results demonstrate that DYG-SLAM can operate stably, robustly and accurately in a dynamic environment.

However, there are still some issues needed be worthy explored in DYG-SLAM. For example, DYG-SLAM only considers the environment dominated by dynamic pedestrians, and does not consider other dynamic objects, such as cars and animals, which will not completely eliminate the influence of other dynamic targets. We can train a more

complex target detection network by training, allowing it to recognize a wider variety of dynamic objects. At the same time, we can consider converting the sparse point cloud map into an octree map, and the system can be deployed on the actual robot to achieve more complex functions.

Acknowledgement

This work is in part supported by the Innovation and Entrepreneurship Training Program of Anhui under Grant 202210357114.

References:

[1] Mur-Artal, R. and Tardos, J. D., ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras, IEEE Transactions on Robotics, No.33, 2017, pp.1255-1262

[2] Endres, F. Hess, J. Sturm, J. Cremers, D. and Burgard, W, 3-D Mapping With an RGB-D Camera, IEEE Transactions on Robotics, No.30, 2014, pp.177-187

[3] Engel, J. Schöps, T. and Cremers, D, LSD-SLAM: Large-scale direct monocular SLAM, European conference on computer vision, Springer, 2014, pp. 834-849

[4] Redmon, J. Divvala, S. Girshick, R. and Farhadi, A, You only look once: Unified, real-time object detection, Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779-788

[5] Redmon J. and Farhadi, A, Yolov3: An incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018

[6] Bochkovskiy, A. Wang, C. Y. and Liao, H.Y. M, Yolov4: Optimal speed and accuracy of object detection, arXiv preprint arXiv:2004.10934, 2020

[7] Barath D, Matas J, Graph-cut RANSAC, IEEE conference on computer vision and pattern recognition, 2018, pp. 6733-6741.

[8] Welch, Greg, and Gary Bishop, An introduction to the Kalman filter, 1995, pp. 127-132

[9] Shao, S., Zhao, Z., Li, B., Xiao, T., Yu, G., Zhang, X., and Sun, J, Crowdhuman: A benchmark for detecting human in a crowd, arXiv preprint arXiv:1805.00123, 2018

[10] Soares, J. C. V. Gattass, M. and Meggiolaro, M. A, Crowd-SLAM: Visual SLAM Towards Crowded Environments using Object Detection, Journal of Intelligent & Robotic Systems, No.102, 2021, pp. 1-16

[11] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollar, P., Zitnick, C.L, Microsoft coco: Common objects in context, European Conference on Computer Vision. Springer, 2014, pp. 740-755

[12] Y. Y. Boykov and M.-P. Jolly, Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images, IEEE International Conference on Computer Vision (ICCV), N0.1, 2001, pp. 105–112

[13] Li, A. Wang, J. Xu, M. and Chen, Z, DP-SLAM: A visual SLAM with moving probability towards dynamic environments, Information Sciences, No.556, 2021, pp. 128-142

[14] Zhong, F. Wang, S. Zhang, Z. Chen, C. and Wang, Y, Detect-SLAM: Making Object Detection and SLAM Mutually Beneficial, IEEE Winter Conference on Applications of Computer Vision (WACV), 2018, pp. 1001-1010

[15] Sturm, J. Engelhard, N. Endres, F. Burgard, W. and Cremers, D, A benchmark for the evaluation of RGB-D SLAM systems, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 573-580

[16] Ai, Y. B., Rui, T., Yang, X. Q., He, J. L., Fu, L., Li, J. B., and Lu, M, Visual SLAM in dynamic environments based on object detection, Defence Technology, No.17, 2021, pp.1712-1721

[17] Wu, W. Guo, L. Gao, H. You, Z. Liu, Y. and Chen, Z, YOLO-SLAM: A semantic SLAM system towards dynamic environment with geometric constraint, Neural Computing and Applications, No.34, 2022, pp. 6011-6026

[18] Bescos, B. Campos, C. Tardós J. D.and Neira, J, DynaSLAM II: Tightly-Coupled Multi-Object Tracking and SLAM. IEEE Robotics and Automation Letters, No.34, 2021, pp. 5191-5198.