DCS-SLAM: A Semantic SLAM with Moving Cluster towards Dynamic Environments

Jiacheng Zhang[1,2], Mingyu Gao[1,2*], Zhiwei He[1,2], Yuxiang Yang[1,2*]
[1] School of Electronics and Information, Hangzhou Dianzi University, Hangzhou, China
[2] Zhejiang Provincial Key Lab of Equipment Electronics, Hangzhou, China

***Abstract*— Semantic SLAM, which can greatly improve tracking performance in highly dynamic scenes and construct dense maps with semantic information, has gained a lot of attention recently. However, it remains a challenge to obtain rich semantic information while ensuring real-time tracking. In this paper, we propose a new semantic SLAM framework, called DCS-SLAM, which uses the Moving Cluster association and MapPoint's motion probability propagation to solve this problem. Specifically, a novel data communication method is devised to obtain the Moving Cluster for each frame in a low-cost way by fully integrating geometric information and asynchronous semantic information. Besides, the Bayesian probability transfer model of MapPoint is applied to further remove the motion points. Experimental results will demonstrate that our framework has better computational efficiency while guaranteeing absolute trajectory accuracy. Moreover, our DCS-SLAM can achieve dense static map construction with rich semantic information by effectively removing the dynamic objects.**

I. INTRODUCTION

Simultaneous localization and mapping(SLAM)[1] is an important link for mobile robots to realize scene perception, which can provide support for subsequent robot planning and decision-making. Visual SLAM(vSLAM)[2] uses a camera as a sensor, which can obtain richer scene information than the LiDAR sensor, receiving extensive attention from researchers at home and abroad. The current mainstream vSLAM frameworks, LSD-SLAM[3], ORB-SLAM Series[4], [5], [6] etc. all use static assumptions for tracking. They can use numerical methods such as RANSAC[5], robust kernel functions[7], etc. to reject small motion points. However, if the motion target is too large, it will greatly affect the tracking or even directly lead to tracking failure. How to use vSLAM in highly dynamic scenes has been a problem that attracts attention.

How to remove the feature points of moving objects is the key to improving the SLAM effect in dynamic scenes. [8], [9], [10], [11] use geometry-based methods, which reduce the impact of high dynamics on tracking to some extent and can run quickly, but they are lacking in robustness and stability. In addition, they can't remove the influence of temporarily stationary movable objects - the feature points on this part are often unstable. Therefore, how to recognize dynamic objects in the environment and realize robust SLAM in a dynamic environment are still a difficult problems and a hot spot of research to be solved.

*Corresponding author

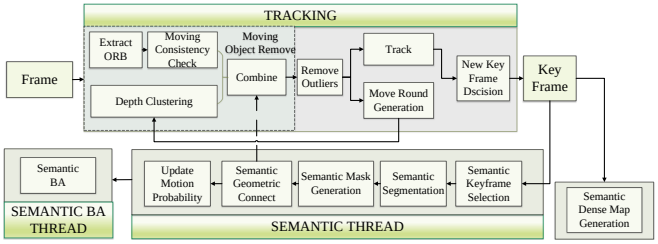

Fig. 1: Program framework diagram: In our framework, the SEMANTIC thread runs asynchronously to the TRACKING thread, so no semantic blocking occurs. In the TRACKING thread, we parallelize the modules to obtain real-time performance.

In recent years, researchers have used semantic analysis to provide recognition of dynamic objects. [12], [13], [14], [15], [16], [17] use semantic segmentation networks[18], [19], [20], [21] to recognize specified targets, together with some geometric methods. Although the effect of dynamic objects is initially addressed, these methods all use a synchronous mode with the visual odometry when using the semantic segmentation network. As a result, some methods[13], [14], [15] give up the real-time nature of SLAM, and others[12] can only use semantic segmentation networks with less time consumption but poorer performance. To reduce time consumption, only a limited variety of semantic networks can be used, which in turn affects the semantic performance. This is because the detection range of the semantic segmentation network needs to be extended as much as possible to minimize the possibility of dynamic objects not being detected by the network, thus improving the scene adaptation of semantic SLAM and serving subsequent advanced tasks.

To reduce the time consumption of the synchronization mode, [16], [17] allow not to wait for the semantic segmentation results between frames, and eliminate the effect of semantic processing time consumption on the frame rate of pose output. However, all of these frameworks lack a real-time dynamic point removal module during the visual odometry stage. When the semantic delay is too long, the lagged semantic information will lead to the current tracking error and then generate a chain of tracking errors, which also affect the actual operation of these frameworks, so they do not support semantic segmentation networks with a long delay. Therefore, it is a challenging problem to ensure that SLAM can accomplish real-time tracking in highly dynamic scenes while acquiring as much semantic information as

possible for subsequent advanced tasks.

To resolve the appeal contradiction, we design a SLAM framework combining semantic and geometric information to achieve real-time tracking in highly dynamic scenes. This framework is dominated by moving cluster association for front-end filtering and the Bayesian probability transfer model of MapPoint for back-end optimization. We also carry out further optimization of the semantic results for dense map construction. Our main contributions are as follows:

- We propose a new framework for cooperating vSLAM and semantic segmentation networks: We perform data association after combining semantic and geometric information, to get Moving Cluster in a low-cost way at the visual odometer stage. Moreover, we use MapPoint's motion probability propagation to further remove the motion points. This new framework ensures accurate tracking in real-time while allowing longer semantic delays to capture richer semantic information.
- After extending the original semantic mask in-depth, judging the movable objects, and adding noise removal to build a clear static dense map, we build a semantic dense map with richer semantic information to serve subsequent advanced tasks.
- We combine our motion removal method with the ORB-SLAM3 system. The TUM RGB-D benchmark dataset is used for evaluation and method comparison. The experimental results demonstrate that the method can greatly improve the robustness and accuracy of the ORB-SLAM3 system.

II. The Proposed Method

This section will introduce DCS-SLAM in detail through three aspects. Firstly, the framework structure diagram of DCS-SLAM is given. Then, the way of semantic and geometric cooperation to remove moving objects is described in detail, including obtaining geometric and semantic information and combining them into a Moving Cluster, as well as the way to further optimize in semantic BA. Finally, we introduce the way of semantic dense map construction.

ORB-SLAM3 has good performance in most practical situations, so DCS-SLAM takes ORB-SLAM3 as the base framework. Fig. 1 shows the framework of our program, and we do not show the framework repeated with ORB-SLAM3.

A. The acquisition method of Geometric Cluster

After the frame input, we perform Depth Clustering and Moving Consistency Check in parallel to combine into Geometric Cluster.

We perform Depth Clustering on the depth image. Specifically, we obtain the 3D coordinates of each pixel point by back-projecting the depth image according to the pinhole camera model. Then we perform Kmeans clustering on the 3D coordinates of each pixel point to obtain Clusters. To meet the real-time requirements and establish the data association of Cluster stably, we make the following improvements to Kmeans clustering.

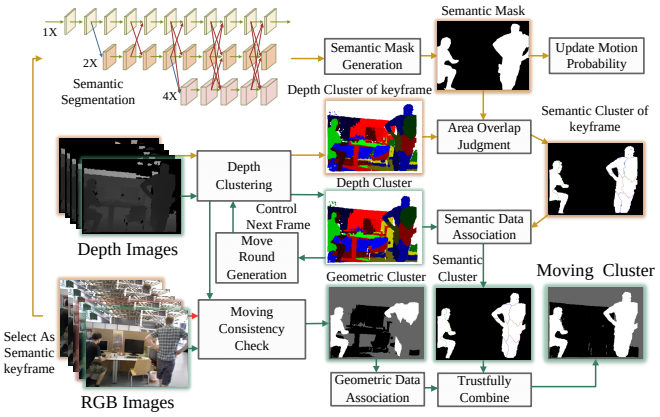

Fig. 2: Moving Cluster acquisition method. The Moving Cluster of the current frame (in green) is acquired with the previous frame (in Red) RGB information and the semantic keyframes' (in yellow) queue corresponding information.

Based on the continuity of the images, we only fine-grained clustering at initialization and subsequently fine-tuned the Clusters only. We generate the previous frame's Move Round in parallel with the Track mode and use it to control the fine-tuning of the current frame. Specifically, we obtain the edges of each Cluster in the previous frame and adjust the expansion operator to expand according to the recent movement speed. We consider that a pixel can complete transferring the Cluster to which it belongs only if it is in the edge region of both the Cluster it is in and the Cluster it wants to transfer; Otherwise, it can only inherit the Cluster from the previous frame.

During clustering iterations, if a Cluster loses all owned pixels and needs to randomly transfer its cores, we disable the cluster core randomization to the previous frame's Moving Cluster and Move Round to prevent the possibility of reclassifying the moving part as stationary during the semantic delay.

With the above improvements, we can quickly obtain Clusters that can represent the same object stably over a longer period and call their set a Depth Cluster.

We propose the concept of core transfer to indicate that the Cluster transforms the represented object. We judge a Cluster as having undergone a core transfer if it satisfies the following two conditions:

Condition 1: we calculate the difference among the three dimensions coordinates of each Cluster core in the current frame and the previous frame, and when it exceeds the threshold θ_t, we consider that a core transfer has occurred. Clusters that are at the edge of the frame have the lower θ_t because we think objects usually disappear from there.

$$\left\| core(c_i^t) - core(c_i^{t-1}) \right\|_2^2 \geq \theta_t \tag{1}$$

Condition 2: when a Cluster loses all pixels and thus randomizes the core in three dimensions, we consider that a core transfer has occurred.

In the Moving Consistency Check, we calculate the distance $d(p_t^i, lp_{t-1}^i)$ from the polar line lp_{t-1}^i for each key

point p_t^i. and count the static values $Ps(p_t^i)$ and dynamic values $Pd(p_t^i)$ of static and dynamic key points. Notice that we will not use the key points located in the Moving Cluster of the previous frame when calculating the basis matrix.

$$Ps(p_t^i) = \begin{cases} (0.1 - d(p_t^i, lp_{t-1}^i))\tau_1 & if(d(p_t^i, lp_{t-1}^i) < 0.1) \end{cases} \tag{2}$$

$$Pd(p_t^i) = \begin{cases} d(p_t^i, lp_{t-1}^i)\tau_2 & if(d(p_t^i, lp_{t-1}^i) > 0.7) \end{cases} \tag{3}$$

where τ_1,τ_2 are the constants used to scale the static and dynamic values to the same magnitude, respectively.

Then, we count the static $Ps(c_t^i)$ and dynamic value $Pd(c_t^i)$ in Cluster to obtain the probability of the Cluster's geometric motion probability m_i^t at the current frame.

$$Ps(c_t^i) = \sum_{j \in c_t^i} Ps(p_t^j) \tag{4}$$

$$Pd(c_t^i) = \sum_{j \in c_t^i} Pd(p_t^j) \tag{5}$$

$$m_i^t = 0.5 + Pd(c_t^i) - Ps(c_t^i) \tag{6}$$

where geometric motion m_i^t is a normalized number. The closer its value is to 1, the more likely it is to be dynamic; otherwise, the more likely it is to be static. We refer to the set of m_i^t as the current frame's Geometric Cluster.

B. The acquisition method of Semantic Cluster

The SEMANTIC thread always processes the latest semantic keyframe in the queue asynchronously. When the segmentation result of Semantic keyframes is obtained through Mseg network, we generate a Semantic Mask based on the previous moving objects and perform an Area Overlap Judgment of the Semantic Mask and its corresponding Depth Cluster to determine which Cluster should belong to the Mask. Specifically, we count the area occupied Mc_t^i of each Cluster c_t^i, and calculate the area relationship with each connected component M_h of Semantic Mask.

$$Area_{and} = Mc_t^i \& M_h \tag{7}$$

$$Area_{or} = Mc_t^i | M_h \tag{8}$$

$$Area_{not} = Mc_t^i \& (\neg M_h) \tag{9}$$

We use the IOU condition to determine the case where the Cluster overlaps with M_h, the Cluster is considered to overlaps with M_h when the ratio of $Area_{and}$ to $Area_{or}$ is larger than λ_1; The Include condition is used to determine the case where the Cluster belongs to M_h, the Cluster is considered to belongs to M_h when the ratio of $Area_{not}$ to $Area_{and}$ is less than λ_2. λ_1,λ_2 are set to 0.5 and 0.1 respectively in the experiment. We define the semantic motion probability S_i^t and call the set of S_i^t Semantic Cluster of Semantic keyframes.

$$IOU = \frac{Area_{and}}{Area_{or}} > \lambda_1 \tag{10}$$

$$Include = \frac{Area_{not}}{Area_{and}} < \lambda_2 \tag{11}$$

$$S_i^t = \begin{cases} 1 & IOU|Inlcude \\ 0 & others \end{cases} \tag{12}$$

C. The acquisition method of Moving Cluster

We want the Moving Cluster to be acquired before the formal Track to prevent chaining errors of tracking. Therefore, in the Combine module of the TRACKING thread, we combine Geometric Cluster and Semantic Cluster in a trustful way after data association to obtain the Moving Cluster in a real-time and stable manner. The overall flowchart for our acquisition of the Moving Cluster is shown in Fig. 2.

Traditional geometric and semantic SLAM filter dynamic points in frames. However, it is believed that keeping track of dynamic objects greatly improves the pose estimation performance, some recent semantic SLAMs have proposed the use of MapPoint-based movement probability propagation algorithms. Similar to it, we propose a Cluster-based algorithm: each Cluster carries motion probability $D_t(c_i)$, until core transfer occurred. The Cluster-based association not only achieves the temporal association of data but also better accomplishes the spatial association.

We set a Boolean value $D_t^S(c_i)$ for each Cluster in the current frame which characterizes the degree of semantic dynamic. We search the history of semantic keyframes from back to front corresponding to Cluster's semantic motion probability S_i^t, and statistics dynamic times until core transfer happened. We set the $D_t^S(c_i)$ of the Cluster that satisfies the dynamic time's condition to 1. Then we can get the current frame's Semantic Cluster.

Similar to the semantic data association, we also establish a geometric data association, where we set the geometric motion probability $D_t^G(c_i)$ for each Cluster, using sliding average to obtain:

$$D_t^G(c_i) = (1 - \beta_n) \times D_{t-1}^G(c_i) + \beta_n \times m_i^t \tag{13}$$

where β_n denotes the confidence level of the geometric result of a new frame, which is normally 0.6; when a Cluster is judged as static or dynamic for more than 5 consecutive frames, we consider that this judgment has been formed and set it to 0.2.

Also, due to the instability of geometric judgment, we introduce error control. We find that the degree of motion of the object is gradual under normal conditions. When the geometric judgment judges the sudden change of the scene as highly dynamic, we consider that there is an error in the geometric judgment, and do not accept the update of geometric motion probability $D_t^G(c_i)$.

After that, we combine $D_t^S(c_i)$ and $D_t^G(c_i)$ to obtain the motion probability $D_t(c_i)$ for each Cluster in a trustful way. Our trust judgment is derived from the following analysis:

Geometric judgments are real-time but not stable; Semantic judgments are credible but have a semantic delay.

In most cases, dynamic objects exist continuously in the scene, so we can expand the semantic information obtained from the interval to all frames based on the continuity of the

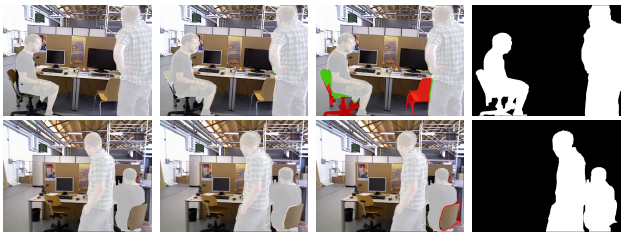

Fig. 3: Mask expansion for dense map construction: (a) Semantic mask, (b) Depth correction of the moving object mask, (c) Moveable object mask modification, (d) The final mask.

image. At this time, the semantic information $D_t^S(c_i)$ is the main basis for judging the dynamics.

There is a situation where a moving object enters the scene during the semantic delay, and the semantic information can't detect the presence of the new moving object, and we trust the geometric information $D_t^G(c_i)$ completely at this time.

Specifically, when a Cluster has undergone a core transfer after the most recent semantic result, it is considered that the Cluster has changed the described object and the original semantic information can no longer represent the Cluster. We define the motion probability at time t for each Cluster c_i as $D_t(c_i)$. $D_t(c_i)$ is calculated as follows:

$$D_t(c_i) = \beta_s \times D_t^S(c_i) + (1-\beta_s) \times D_t^G(c_i) \tag{14}$$

When a Cluster has undergone the core transfer after the latest semantic information, then $\beta_s = 0$, otherwise $\beta_s = 0.8$.

Whether a Cluster is dynamic or static is judged using predefined probability thresholds θ_d and θ_s. They are set to 0.6 and 0.4 respectively in the experiment. We call the set of dynamic Clusters a Moving Cluster.

$$Status(c_i) = \begin{cases} dynamic & D_t(c_i) > \theta_d \\ static & D_t(c_i) < \theta_s \\ unknown & others \end{cases} \tag{15}$$

In the subsequent module Remove Outliers, we calculate the descriptors of key points except those located in the Moving Cluster of the image pyramid. The final feature points are obtained and used for the formal Track.

D. Further filtering of semantic BA

Numerical Area Overlap Judgment will always be caught in the dilemma of judging too many static Clusters as dynamic, or not judging dynamic Clusters. Judging too many static Clusters as dynamic can also greatly affect tracking and even lead to failure. Therefore, we tighten the criteria for judging Cluster as dynamic. For the missing parts, we make up for them on Semantic BA Thread. In this thread, we update the motion probabilities of all MapPoint's corresponding to the undeleted key points on the Semantic Mask by the Bayesian probabilistic transfer model. After that, we perform Semantic BA, i.e., we use motion probabilities as weights for back-end optimization. In this way, we can compensate to some extent for the missing dynamic part of the front end.

E. Semantic Dense Map Construction

One of the main objectives of our framework is to relax the semantic delay requirement to use a more effective semantic segmentation network. The network we use, MSeg, takes longer but has more semantic classes and higher detection rates than the currently used networks, Mask-RCNN and SegNet. These advantages will play an important role in the construction of semantic dense maps.

Before constructing the semantic map, we need to construct the static dense map, and we modify the original semantic mask as follows.

Depth correction of the moving object mask: According to Semantic Mask, we perform depth expansion of the moving object mask. We distinguish moving object masks into individuals based on the connected component analysis. Then, we calculate the depth range of each individual and consider the pixels that are within the depth range and in the vicinity of the individual as the depth extension of that individual, As shown in Fig. 3(b).

Movable object mask modification: Movable objects have a relatively large impact on the construction of dense maps. For example, in a dynamic scene, a character moving a chair will generate multiple chairs in the scene. We first get the movable object mask based on the result of semantic segmentation, then we use the absolute area condition to determine the side observation (shown on the top of Fig. 3(c)) and the contour surround condition to determine the back observation (shown in the bottom of Fig. 3(c)). We can remove large chunks of moving and motionable objects as shown in Fig. 3(d).

Spatial residual part removal: there still have residual redundant pixels. Considering that the residual information mostly appears at the edges of moving objects, for each pixel that is about to appear in the point cloud, we obtain its projected position in the previous semantic keyframe and remove the pixels with too large a depth difference.

After getting the static dense map, we replace the RGB values of the point cloud with the pseudo-color values of the semantic segmentation result to construct the semantic dense map.

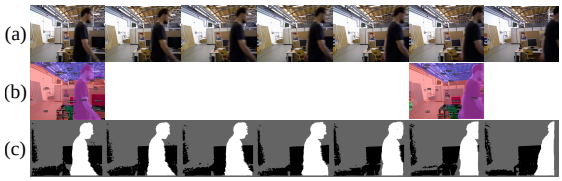

Fig. 4: Continuous Moving Cluster: (a) RGB sequences of highly dynamic scenes, (b) semantic segmentation results of semantic keyframes, (c) Moving Cluster.

III. Experimental Result

In this section, experimental results are presented to demonstrate the effectiveness of DCS-SLAM. Using the public TUM RGB-D dataset[22], we evaluate its tracking performance and real-time performance by comparing it with state-of-the-art SLAM methods.

TABLE I: Absolute Trajectory Error(ATE)(M).

Seq	ORB-SLAM3		DS-SLAM (65ms/f)		DM-SLAM (>250ms/f)		DP-SLAM (>250ms/f)		DynaSLAM (500ms/f)		RDS-SLAM (58ms/f)		DetectSLAM (>100ms/f)		DCS-SLAM (80ms/f)	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
w/half	0.5362	0.1953	0.0303	0.0129	0.0274	0.0137	0.0254	0.0129	0.0296	0.0157	0.0259	0.0141	0.0514	-	0.0256	0.0126
w/rpy	0.8397	0.4006	0.4442	0.2350	0.0328	0.0194	0.0356	0.0218	0.0354	0.0190	0.1468	0.1051	0.2959	-	0.0398	0.0253
w/static	0.4188	0.1600	0.0081	0.0033	0.0079	0.0040	0.0079	0.0037	0.0068	0.0032	0.0815	0.0224	-	-	0.0083	0.0039
w/xyz	0.7491	0.3862	0.0247	0.0161	0.0148	0.0072	0.0141	0.0073	0.0164	0.0086	0.0213	0.0127	0.0241	-	0.0155	0.0074
s/static	0.0079	0.0038	0.0065	0.0033	0.0063	0.0032	0.0059	0.0029	0.0108	0.0056	0.0088	0.0043	-	-	0.0068	0.0034

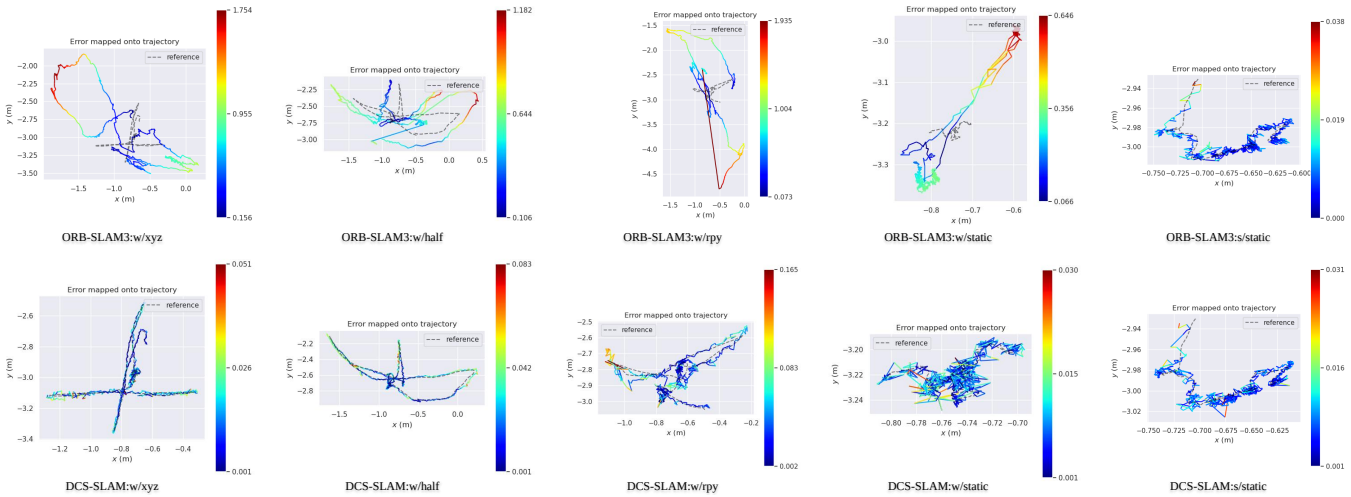

Fig. 5: In comparisons of ORB-SLAM3 and DCS-SLAM absolute trajectory error (ATE), we use colors to indicate the degree of deviation, the colored solid line indicates the result of the corresponding frame calculation, and the gray dashed line indicates the ground truth.

A. Moving Cluster Acquisition

As stated in the System Introduction, obtaining a Moving Cluster, is the focus and difficulty of our work. It is also the most intuitive manifestation of the usefulness of our framework. Here we show the Moving Cluster results in Fig. 4, although the semantic keyframe interval is large, we can obtain a continuous and stable Moving Cluster.

B. Tracking performance evaluation

In TABLE I, We present the comparisons of DCS-SLAM with ORB-SLAM3[6], SOTA synchronous methods: DS-SLAM[12], DynaSLAM[13], DM-SLAM[14], and DP-SLAM[15], SOTA asynchronous methods: Detect-SLAM[16] and RDS-SLAM[17]. Specifically, low dynamic scene "s/static" and high dynamic scenes "w/half", "w/rpy", "w/static", "w/xyz" of TUM RGB-D dataset[22] are used in the experiments. We evaluate the trajectory using the root mean square error(RMSE) and standard deviation(S.D.) of the absolute trajectory error(ATE). For all sequences, we test 10 times and take the middle value.

We can see that DCS-SLAM has a good balance of accuracy and speed. DCS-SLAM outperforms ORB-SLAM3 in each scenario. Compared with the synchronous method with fast segmentation network SegNet[19], such as DS-SLAM, we have higher accuracy. Compared with the synchronous method with fine segmentation network MaskRCNN[20], such as DM-SLAM, DynaSLAM, and DP-SLAM, we have better real-time performance with only a slight decrease in accuracy. Compared with other asynchronous methods, such as Detect-SLAM, and RDS-SLAM, we have improved accuracy in all sequences and do not lose track in any case. It is because we trustfully combine the geometric and semantic information with data association to obtain Moving Cluster, thus segmenting the moving objects before the formal track and getting rid of the possibility of causing chain errors. In addition, Mseg[21] is used as our semantic segmentation network, which can obtain richer semantic information than MaskRCNN and SSD[18]. Due to our novel asynchronous architecture, the 550ms processing time of Mseg does not affect the visual odometer process. Hence, our DCS-SLAM can achieve a tracking speed of 80ms/frame. We also plot the absolute trajectory error of DCS-SLAM and ORB-SLAM3 in Fig. 5. We use TUM's trajectory evaluation tool to test the data in TABLE I. We use the standard evaluation tool EVO to plot the data in Fig. 5. A video demo of the testing is also provided[1].

C. Semantic Dense Map Construction

We select the high dynamic scene w/xyz and low dynamic scene s/static for dense map construction and compare them with ORB-SLAM3. As shown in Fig. 6(a)(b), it can be seen that in the high dynamic scene, ORB-SLAM3 has difficulty in constructing a dense map due to the failure at tracking, while in the low dynamic scene, ORB-SLAM3 can build the overall environment, but the dynamic objects stay in the environment forever due to the lack of dynamic object removal. Our framework solves these problems.

[1]https://github.com/boyong-csy/DCS-SLAM

Fig. 6: Dense map construction: (a) Dense map construction of ORB-SLAM3 (b) Dense map construction of DCS-SLAM (c) Semantic dense map construction for DCS-SLAM.

In addition, we show the semantic dense map in Fig. 6(c). It can be seen that our semantic dense map has rich semantic information. Because Mseg[21] is applied as our semantic segmentation network, which has 194 classes. This is also attributed to our novel asynchronous structure, the 550ms processing time of Mseg does not affect the speed of the tracking process. Hence, we can use a time-consuming but precise semantic segmentation network like MSeg in our SLAM framework to construct a high-quality semantic dense map.

IV. CONCLUSIONS

In this paper, we propose a new way to communicate semantic segmentation and vSLAM to solve the problem of vSLAM in dynamic environments. We have further relaxed the semantic segmentation processing time limit. It enables SLAM to use more time-consuming semantic segmentation networks while ensuring real-time tracking. We believe that the framework of semantic SLAM should be: fine-grained semantic segmentation and a low-cost complementary framework in conjunction. At present, semantic segmentation is developing rapidly, but it is still difficult to meet the real-time requirements of SLAM while having good segmentation performance. Hence, it is necessary to apply semantic segmentation into SLAM framework in an asynchronous way as our DCS-SLAM.

Acknowledgement This work was supported in part by the National Natural Science Foundation of China under Grant 61873077, the Hangzhou Key Research and Development Project of China under Grant 2022AIZD0009, and the Zhejiang Provincial Key Lab of Equipment Electronics.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual slam algorithms: A survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, pp. 1–11, 2017.

[3] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.

[4] 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.

[5] 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.

[6] 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.

[7] T. Pire, T. Fischer, G. Castro, P. De Cristóforis, J. Civera, and J. J. Berlles, "S-ptam: Stereo parallel tracking and mapping," *Robotics and Autonomous Systems*, vol. 93, pp. 27–42, 2017.

[8] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving rgb-d slam in dynamic environments: A motion removal approach," *Robotics and Autonomous Systems*, vol. 89, pp. 110–122, 2017.

[9] S. Li and D. Lee, "Rgb-d slam in dynamic environments using static point weighting," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2263–2270, 2017.

[10] 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.

[11] T. Zhang, H. Zhang, Y. Li, Y. Nakamura, and L. Zhang, "Flowfusion: Dynamic dense rgb-d slam based on optical flow," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7322–7328.

[12] 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.

[13] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4076–4083, 2018.

[14] J. Cheng, Z. Wang, H. Zhou, L. Li, and J. Yao, "Dm-slam: A feature-based slam system for rigid dynamic scenes," *ISPRS International Journal of Geo-Information*, vol. 9, no. 4, p. 202, 2020.

[15] A. Li, J. Wang, M. Xu, and Z. Chen, "Dp-slam: A visual slam with moving probability towards dynamic environments," *Information Sciences*, vol. 556, pp. 128–142, 2021.

[16] F. Zhong, S. Wang, Z. Zhang, and Y. Wang, "Detect-slam: Making object detection and slam mutually beneficial," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 1001–1010.

[17] Y. Liu and J. Miura, "Rds-slam: real-time dynamic slam using semantic segmentation methods," *IEEE Access*, vol. 9, pp. 23 772–23 785, 2021.

[18] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[19] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[20] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[21] J. Lambert, Z. Liu, O. Sener, J. Hays, and V. Koltun, "Mseg: A composite dataset for multi-domain semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2879–2888.

[22] 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.