USD-SLAM: A Universal Visual SLAM Based on Large Segmentation Model in Dynamic Environments

Jingwei Wang, Yizhang Ren, Zhiwei Li, Xiaoming Xie, Zilong Chen, Tianyu Shen, *Member, IEEE*, Huaping Liu, *Senior Member, IEEE*, and Kunfeng Wang, *Senior Member, IEEE*

***Abstract*—Visual Simultaneous Localization and Mapping (SLAM) has been widely adopted in autonomous driving and robotics. While most SLAM systems operate effectively in static or low-dynamic environments, achieving precise pose estimation in diverse unknown dynamic environments continues to pose a significant challenge. This letter introduces an advanced universal visual SLAM system (USD-SLAM) that combines a universal large segmentation model with a 3D spatial motion state constraint module to accurately handle any dynamic objects present in the environment. Our system first employs a large segmentation model guided by precise prompts to identify movable regions accurately. Based on the identified movable object regions, 3D spatial motion state constraints are exploited to remove the moving object regions. Finally, the moving object regions are excluded for subsequent tracking, localization, and mapping, ensuring stable and high-precision pose estimation. Experimental results demonstrate that our method can robustly operate in various dynamic and static environments without additional training, providing higher localization accuracy compared to other advanced dynamic SLAM systems.**

***Index Terms*—SLAM, localization, object detection, segmentation and categorization.**

I. Introduction

VISUAL Simultaneous Localization and Mapping (V-SLAM) technology can estimate camera poses and simultaneously construct environmental maps using various types of cameras as sensors. Visual SLAM typically employs monocular, stereo, and RGB-D cameras as sensors [1]. These cameras offer advantages such as low cost, strong portability, and the ability to provide rich image information. Consequently, visual SLAM technology has been widely applied in fields such as robotics [2], autonomous driving [3], and virtual/augmented reality [4]. However, some conventional algorithms [5], [6] rely on the assumption that the surrounding environment is static or contains only a limited number of dynamic elements. This assumption does not hold in many real-world scenarios, such as pedestrians and vehicles on the road in autonomous driving, which limits the application of SLAM technology in practical scenarios.

Subsequent research has aimed to make SLAM technology compatible with dynamic environments. Early methods typically used geometry-based or optical flow techniques to remove dynamic objects, but these methods were computationally complex and had low accuracy [7]. With advancements in computer vision, more algorithms have combined geometric or optical flow techniques with segmentation detection models to more stably improve localization accuracy [8], [9]. However, current segmentation models typically perform well only on trained datasets. The gap between these datasets and real-world environments leads to significant performance drops in long-tail distributed or in-the-wild samples, limiting the performance of dynamic SLAM.

To address the aforementioned issues and tackle the challenges posed by highly dynamic indoor and outdoor scenes, we propose a universal visual SLAM system. This system consists of three components: a large segmentation model module, a 3D spatial motion state constraint module, and the final tracking and mapping module. First, we use the large segmentation model SAM [10] to pre-obtain high-precision masks of movable objects. These masks, combined with the original images, form the precise prompts for SegGPT [11], which accurately segments movable objects and delineates movable regions. Next, by incorporating 3D spatial motion state constraints, we compare the 3D spatial motion state information of the movable object regions with that of the static regions to determine the overall motion state of the movable object regions. This process removes dynamic areas within the regions, preserves the static areas at the current time, and extracts static feature points. These static feature points are then input into the tracking and mapping module for subsequent tracking and pose estimation, thereby achieving stable and high-precision camera pose estimation. Compared to other dynamic scene SLAM methods, our algorithm demonstrates superior performance and stronger generalization capabilities. It achieves better localization results across different scenes, particularly in out-of-domain scenarios.

Received 30 July 2024; accepted 31 October 2024. Date of publication 14 November 2024; date of current version 25 November 2024. This article was recommended for publication by Associate Editor Reza Sabzevari and Editor Javier Civera upon evaluation of the reviewers' comments. This work was supported in part by the Science and Technology on Metrology and Calibration Laboratory under Grant JLKG2023001B004, in part by the National Natural Science Foundation of China under Grant 62302047 and Grant U22A2061, and in part by the National Key Laboratory of Hybrid Human-Machine Augmented Intelligence, Xi'an Jiaotong University under Grant HMHAI-202416. *(Jingwei Wang and Yizhang Ren contributed equally to this work.) (Corresponding authors: Xiaoming Xie; Kunfeng Wang.)*

Jingwei Wang, Zhiwei Li, Xiaoming Xie, Tianyu Shen, and Kunfeng Wang are with the College of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, China (e-mail: xmxie@mail.buct.edu.cn; wangkf@mail.buct.edu.cn).

Yizhang Ren is with the College of Instrumental Science and Optoelectronic Engineering, Beihang University, Beijing 100191, China.

Zilong Chen and Huaping Liu are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China.

Digital Object Identifier 10.1109/LRA.2024.3498781

In summary, our contributions are summarized as follows:

- A universal visual SLAM algorithm named USD-SLAM that stably operates in any dynamic environment and obtains high-precision poses.
- In the dynamic SLAM algorithm, the semantic module for the first time entirely adopts a large segmentation model, which can robustly handle movable objects in any environment without additional training.
- We use 3D spatial motion state constraints to process the movable regions segmented by the large model, maximizing the retention of all static regions.
- We compare our proposed method with the latest existing methods on multiple datasets (such as KITTI and TUM), demonstrating the superior performance of our approach.

II. Related Work

In this section, we will introduce some classic semantic-based dynamic SLAM methods, divided into two categories.

Methods based solely on semantic information: Ji et al. [12] segmented dynamic objects using a network to identify any movable objects in the scene, removing all highly movable or potentially dynamic objects while retaining static ones, thereby improving position estimation accuracy. Wang et al. [13] proposed a step-by-step semantic fine extraction approach, combining target detection and contour extraction for more efficient dynamic object extraction using semantic information. Qiao et al. [14] performed monocular depth estimation to obtain depth information, combined it with video panorama segmentation to compute instance labels for spatial locations, semantic categories, and temporal consistency of the object's 3D points, and localized dynamic objects to eliminate their influence.

Methods based on the combination of geometric and semantic information: Yu et al. [15] used SegNet for pixel-wise semantic labeling and applied pairwise geometric constraints for motion consistency checking. Dynamic objects were identified and removed if they exhibited both semantic and geometric motion features. WF-SLAM [16] used weighted features based on semantic and geometric information to identify and reject dynamic regions. Cheng et al. [17] retained static points and removed dynamic ones to improve tracking accuracy and localization precision. DynaSLAM [8], based on ORB-SLAM2 [5], introduced dynamic object detection and background rendering, showing robust performance in dynamic scenes across monocular, stereo, and RGB-D datasets by combining multi-view geometry and semantic segmentation models. DynaSLAM II [18] further integrated multi-target tracking but was limited to rigid objects. Chang et al. [19] achieved real-time operation through segmentation threads and performed well in indoor dynamic scenes. DN-SLAM [20] improved localization and mapping by integrating semantic segmentation, optical flow, and SAM [10], enhancing segmentation accuracy, extracting static features, and generating dense maps using NeRF. Wen et al. [21] used semantic segmentation and 3D scene flow to determine real object motion in autonomous driving and remove dynamic object interference. DynaMoN [22] improved localization accuracy by integrating motion and segmentation masks into dense BA, removing dynamic interference. Additionally, it combined a 4D scene representation with NeRF for 3D reconstruction [23] to generate new views in dynamic scenes.

III. System Description

In this section, we introduce the complete universal semantic SLAM system. Designed based on the core principle of universal generalization, this system can stably acquire camera poses in various unknown dynamic environments without the need for additional training. Our system inputs include stereo images and RGB-D images, and it outputs the camera pose for each frame. The complete system framework is shown in Fig. 1, and the specific modules are described in the other parts of this section.

A. Segmentation Thread

In current research, segmentation-based dynamic SLAM systems typically require retraining the network to reliably segment movable objects within a scene. Moreover, these segmentation algorithms perform poorly on unknown objects and small target objects, affecting the localization results of dynamic SLAM systems. To accommodate the versatility of traditional SLAM and the diversity of unknown scenarios, we employ the universal large segmentation model SegGPT as the segmentation module. We use precise masks obtained from the large segmentation model SAM as prompts to guide SegGPT in the targeted and precise segmentation of movable objects.

1) Interactive Precise Guidance Prompts: This module serves as the preprocessing module of the entire system, which does not directly process unknown scenes but guides the segmentation model to handle unknown scenes through prompts. To enhance the large model's segmentation control of movable objects and reduce the additional resources consumed by segmenting non-essential objects, we propose interactive precise guiding prompts to direct the large model in the targeted segmentation of movable objects in the scene. Specifically, the composition of the precise guidance prompts is shown in Fig. 2. The main framework is based on the large segmentation model SAM, which utilizes prompt information associated with points and boxes to segment movable objects in the example scene images and obtain the corresponding masks. These masks, together with the example scene images, form the precise guidance prompts for SegGPT. The example images are sourced from the internet, open-source datasets, etc., and include different categories of movable objects. Therefore, as long as the prompt contains a certain object category, the mask of the same category can be obtained in unknown scenes. Therefore, the more movable object mask categories included in the prompt, the stronger the segmentation generalization in unknown environments, but at the cost of higher computational consumption. We only select object categories with a higher degree of dynamism to achieve a dynamic balance between generalization and computational resource consumption. As shown in Fig. 3(d), although our prompt does not include content from the Bonn dataset, we can still achieve good segmentation results directly on the Bonn dataset.

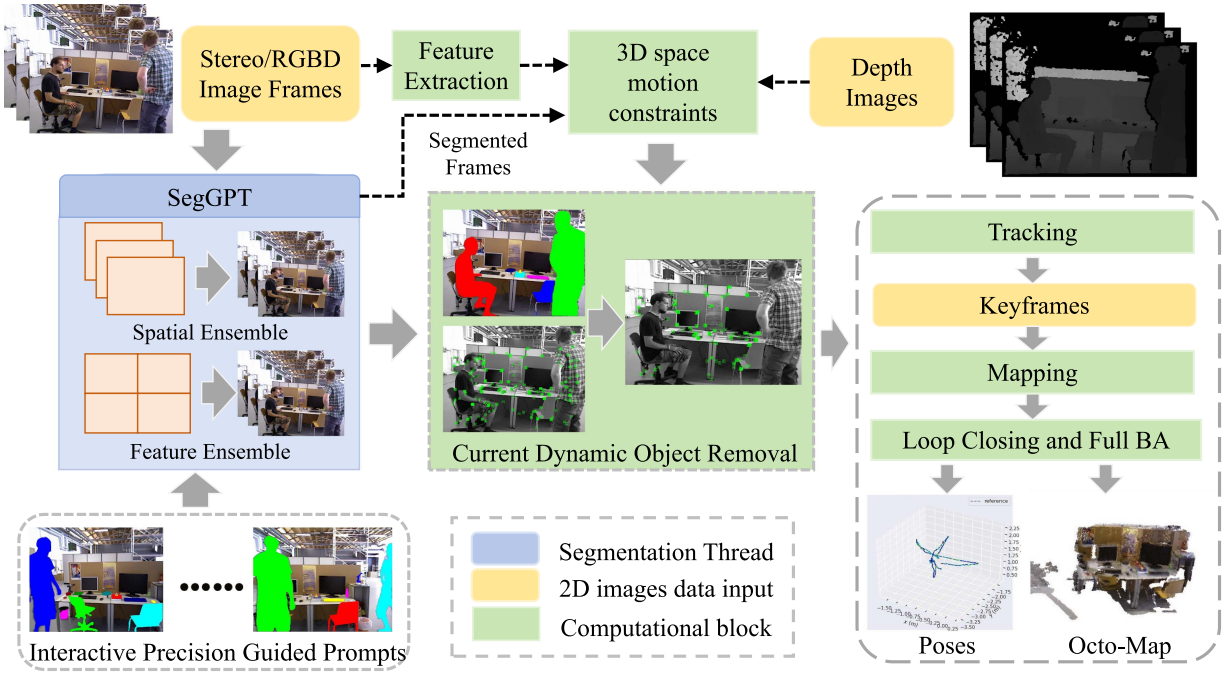

Fig. 1. System Overview: The system consists of three main components: 1) Segmentation Module, which uses a large segmentation model guided by precise prompts to accurately segment movable objects in the scene; 2) 3D Motion State Constraints Module, which removes dynamic object regions by utilizing the 3D spatial motion intensity of matched pairs; 3) Tracking and Mapping Module, which uses the preserved static regions for tracking to obtain high-precision pose estimation and mapping.

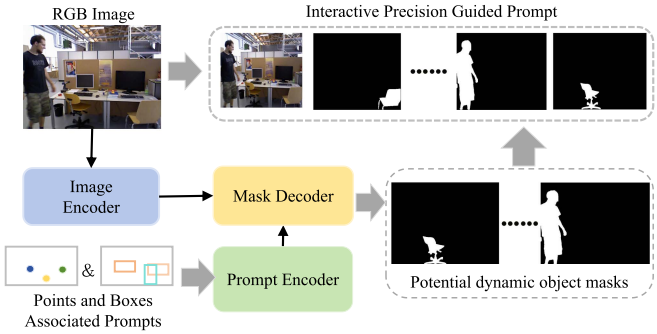

Fig. 2. Composition of precise guidance prompts: open example images and the masks of movable objects within the images.

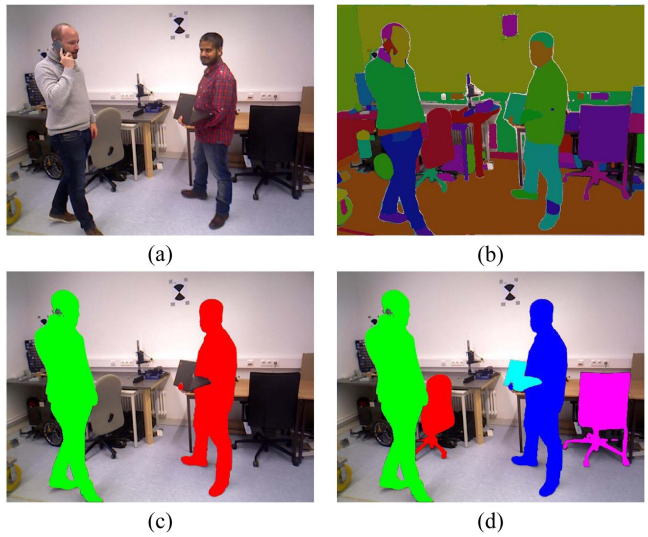

Fig. 3. Results obtained by different segmentation methods. (a) Original image. (b) Results after SAM segmentation. (c) Results after instance segmentation. (d) Segmentation results of our large segmentation model.

The point and box-associated interactive prompts allow for fine-tuning of the segmentation within the box area by adding or removing parts of the region. By combining the advantages of these two prompts, segmentation errors can be reduced, and object categories can be more precisely obtained. As a result, object category extraction from the example images can be flexibly and quickly modified and optimized by humans, reducing the subsequent workload of mask optimization.

2) Universal Large Segmentation Model: SegGPT is a universal large segmentation model capable of segmenting various content within a contextual framework. By unifying different segmentation tasks, the model converts multiple data types into a uniform image format to perform tasks without relying on semantic labels, enhancing adaptability and flexibility while reducing data preparation costs. Based on example prompts, SegGPT can perform contextual reasoning, using feature integration and spatial integration to enhance the segmentation prediction capabilities and robustness of the current frame. This ensures that when processing scene sequence objects, the model can reference segmentation results from multiple examples. Feature integration involves extracting and averaging features across multiple examples to improve the model's stability and robustness, ensuring high-precision segmentation results. Spatial integration stitches together multiple reference images and masks, utilizing global information to further enhance segmentation

performance. By leveraging these contextual cues from example images, the model can maintain segmentation consistency even when input images are affected by environmental changes such as lighting or fog, effectively reducing segmentation errors.

After the image sequence is transmitted, SegGPT segments the movable objects based on the precise guiding prompts, dividing the scene into two parts: movable object areas and static areas. Subsequent 3D spatial motion constraints only need to assess the overall movement of the movable object areas, retaining regions that are judged to be static. As shown in Fig. 3, other segmentation methods can only segment fixed classes or all objects, whereas our method can directionally segment different classes of movable objects.

B. 3D Spatial Motion State Constraints

After the segmentation thread, the scene is divided into movable object regions and static regions. However, not all movable objects are dynamic in the current frame, such as temporarily parked cars or people sitting by the roadside. We assess the overall motion state of the movable object regions to reduce computational cost while maximizing the retention of static regions, thereby improving tracking and localization accuracy. Based on this, we use 3D spatial motion constraints to determine the current motion state of the objects. Initialization tracking preliminarily obtains the transformation relationship T between frames. This transformation is used to calculate the 3D spatial motion intensity for each pair of matched feature points. By comparing the motion intensities of static and movable regions, we can determine the overall motion state of the current movable region. Even if movable objects are incorrectly segmented due to lighting or foggy conditions, the motion intensity between frames will be calculated for the segmented region, thereby correcting the segmentation errors. This effectively reduces segmentation errors and maintains robust and efficient performance in any environment.

The 3D spatial position information (X_i, Y_i, Z_i) of the special points in the 3D spatial motion state information is obtained based on the combination of the 2D point pixel coordinate values (x_i, y_i) as well as the depth information d_i recovery. where the depth information is obtained by the RGB-D camera directly or by the recovery calculation of the stereo camera. The depth information of the stereo camerad_i

$$d_i = \frac{fb}{X_L - X_R} = \frac{fb}{d} \tag{1}$$

where f is the focal length, obtained from the internal reference matrix, b is the baseline between the cameras, which is the spatial distance between the cameras, obtained from the external reference matrix between the two cameras, and $d = X_L - X_R$ is the parallax between the left camera and the right one. Thus the spatial coordinates of the relevant points on the frame $n-1$ image can be expressed as:

$$\begin{pmatrix} X_{n-1}^i \\ Y_{n-1}^i \\ Z_{n-1}^i \end{pmatrix} = d_{n-1}^i K^{-1} \begin{pmatrix} x_{n-1}^i \\ y_{n-1}^i \\ 1 \end{pmatrix} \tag{2}$$

The spatial coordinates of the relevant points on the image of the next n frames are:

$$\begin{pmatrix} X_n^i \\ Y_n^i \\ Z_n^i \end{pmatrix} = d_n^i K^{-1} \begin{pmatrix} x_n^i \\ y_n^i \\ 1 \end{pmatrix} \tag{3}$$

The spatial position relationship between the two frames is:

$$\begin{pmatrix} X_n^i \\ Y_n^i \\ Z_n^i \end{pmatrix} = R \begin{pmatrix} X_{n-1}^i \\ Y_{n-1}^i \\ Z_{n-1}^i \end{pmatrix} + t \tag{4}$$

The transformation matrix $T = (R|t)$, R is the rotation matrix, t is the translation matrix, x, y, d are the pixel coordinates of the feature points as well as the depth information, and K is the camera internal reference matrix.

Thus we can obtain the 3D spatial motion vector V and the vector intensity $\|V\|$ between the two frames at this point as:

$$V = \begin{pmatrix} X_n^i - X_{n-1}^i \\ Y_n^i - Y_{n-1}^i \\ Z_n^i - Z_{n-1}^i \end{pmatrix} \tag{5}$$

$$\|V\| = \sqrt{(X_n^i - X_{n-1}^i)^2 + (Y_n^i - Y_{n-1}^i)^2 + (Z_n^i - Z_{n-1}^i)^2} \tag{6}$$

We recover the spatial coordinates of the feature points and calculate the corresponding 3D spatial motion information intensity. The 3D spatial motion intensity of the static feature points should be consistent; however, due to noise and errors that often exist in real scenes, the flow intensity of the static scene will change. Therefore, it is necessary to calculate the intensity of multiple feature points within the region to comprehensively evaluate and reduce errors. We set a threshold θ for judgment. Only when the proportion of spatial points within the region with 3D spatial intensity higher than the threshold is considered static at the current moment, otherwise, it is considered dynamic, thus filtering out the moving objects at the current moment. The threshold is taken as the upper control limit θ of the scene flow vector intensity of Nstatic points, excluding the dynamic region, $\theta = \mu + k\sigma$, where k is set to 3 to ensure that the threshold is the controllable upper limit of the error of the static value. The mean and error are derived from the spatial motion intensity within the static region.

$$\mu = \frac{1}{N} \sum_{n=1}^{N} \|v_n^s\| \tag{7}$$

$$\sigma = \sqrt{\frac{1}{N-1} \sum_{n=1}^{N} (v_n^s - \mu)^2} \tag{8}$$

where v_n^s is the spatial motion model of a single feature point in the static region, then v_n^d is the spatial motion model of a single point in the dynamic region

$$\begin{cases} \left\|v_n^d\right\| \geq \|\theta\|, & v_n^d \in dynamic\ point \\ \left\|v_n^d\right\| \leq \|\theta\|, & v_n^d \in static\ point \end{cases} \tag{9}$$

Based on the above equation (9), the real motion state of each spatial point is determined. If 70 percent of the feature points

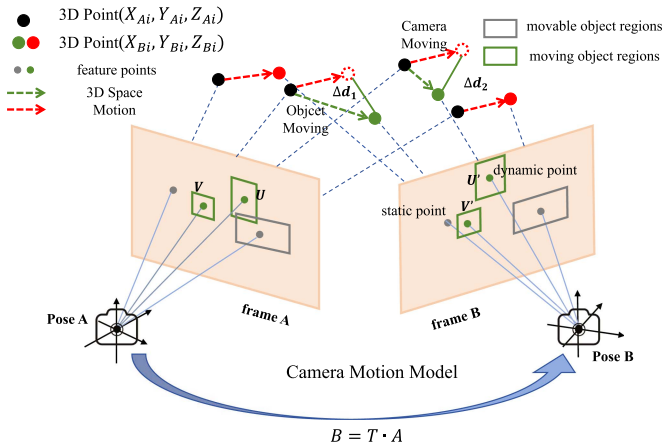

Fig. 4. A schematic diagram of the 3D spatial motion state information of feature points between frames. The true motion state of objects is determined based on the different motion intensities of the static region and the movable object region.

Fig. 5. Extraction of static object feature points combined with 3D spatial motion state information. (a) Feature point extraction under segmentation alone. (b) Feature point extraction after segmentation combined with 3D spatial motion state constraints.

in the region of the movable object have a 3D spatial motion vector strength less than the threshold, the region is considered static at the current moment; otherwise, the region is classified as dynamic. Additionally, if the movable region occupies more than half of the current frame, the density of feature points is increased by extracting more points from the same region to ensure robustness in matching and localization.

As depicted in Fig. 4, the black, green, and red points represent the 3D spatial positions of matching pairs between two frames. The red dashed arrows indicate the spatial displacement vectors of feature points in the static region between frames, while the green dashed arrows represent the spatial displacement vectors of feature points in the actual dynamic region. The green solid lines illustrate the differences in spatial motion intensity of the actual moving objects. The gray box denotes the movable object region, and the green box indicates the current moving object region. The motion trends of static spatial points remain largely consistent, with the same 3D spatial motion intensity. In contrast, the motion trends of dynamic spatial points differ significantly from those of static points. Based on this, we can evaluate and correct the segmented mask regions by assessing the overall motion state of the regions, ensuring that true static feature points are retained even in the event of segmentation errors, thus optimizing the transformation relationship T between the two frames.

We conducted tests on the KITTI dataset, and the results are shown in Fig. 5. When only segmentation is applied, the parked vehicle in the red-boxed area in the figure is recognized as a dynamic region and removed. When combined with 3D spatial information, the vehicle is recognized as a static region, and the corresponding feature points are maintained. This preliminary validation demonstrates the feasibility of our method.

C. Tracking and Mapping

The inputs for this part include RGB images, depth information of the images, dynamic segmentation masks, and all static feature points after removing the dynamic feature points. The entire tracking thread and map-building process are implemented based on the ORB-SLAM2 scheme, with adjustments made to the keyframe selection strategy: 1) increasing the lower bound of the keyframe selection interval, with a dynamic selection of the interval between frames; 2) selecting keyframes immediately when the number of feature points in the tracking thread is too low; 3) selecting frames with a low proportion of movable object segmentation areas as keyframes; 4) selecting keyframes only when the 3D spatial motion intensity between frames exceeds a certain threshold.

The overall tracking part consists of two sections: initialization tracking and local tracking optimization. Initialization tracking occurs after the segmentation thread is completed, where feature points in the absolutely static region of the unsegmented area are matched and tracked to achieve the initialization tracking effect. This initially obtains partial pose estimates and transformation matrices between frames. Local tracking optimization continuously optimizes and determines the static feature points of the current frame through the segmentation thread and the 3D spatial motion information judgment thread. The updated static matching pairs are added to the initial matching pairs for constraints, allowing for local adjustments and optimization of the tracking thread to achieve a more accurate localization effect. By incorporating Octomap, a dense map that is easy to update and can be used for navigation is constructed.

IV. Experimental Results

In this section, we compare our proposed algorithm with SOTA algorithms. The datasets include the indoor dynamic datasets TUM [24] and Bonn [25], as well as the outdoor dynamic dataset KITTI [26]. Note that our algorithms are run directly on the datasets without any additional training.

A. Datasets

Indoor TUM and Bonn RGB-D datasets: These indoor datasets are widely used for evaluating SLAM and visual odometry methods. The data include both dynamic and static sequences, with each sequence containing color (RGB) images, depth images, and ground truth trajectories of camera motion.

Outdoor KITTI stereo dataset: This outdoor dataset is extensively used in autonomous driving research. The data cover a variety of highly dynamic real-world environments. Among them, 11 sequences (00-10) from the KITTI odometry dataset provide ground truth trajectories, which are used to test and evaluate our SLAM algorithm.

TABLE I
ATE COMPARISON RESULTS FOR TUM AND BONN PARTIAL SEQUENCES OF THE INDOOR HIGH DYNAMIC RGB-D DATASETS

	RGB-D Sequence	ATE RMSE[m]					
		ORB-SLAM2 [5]	DynaSLAM [8]	LC-CRF-SLAM [27]	DN-SLAM [20]	**USD-SLAM(S)**	**USD-SLAM(A)**
Indoor High Dynamic Datasets	Tum_walking_xyz	0.459	**0.015**	0.024	0.015	0.037	0.035
	Tum_walking_half	0.351	0.030	0.041	0.026	0.021	**0.020**
	Tum_walking_rpy	0.662	0.037	0.055	**0.032**	0.058	0.035
	Bonn_balloon	0.065	0.029	0.033	0.030	0.029	**0.028**
	Bonn_balloon2	0.230	0.032	0.026	**0.025**	0.030	0.029
	Bonn_crowd2	0.603	0.029	0.038	0.028	0.028	**0.028**
	Bonn_crowd3	0.524	0.037	0.035	0.026	0.029	**0.026**
	Bonn_person_tracking	0.796	0.039	0.044	0.038	0.029	**0.028**
	Bonn_person_tracking2	1.068	0.126	0.045	0.042	0.046	**0.041**
	Bonn_moving_ob_box2	0.622	0.262	0.295	**0.120**	0.173	0.178

TABLE II
ATE COMPARISON RESULTS FOR KITTI PARTIAL SEQUENCES OF THE OUTDOOR HIGH DYNAMIC STEREO DATASETS

	Stereo Sequence	ATE RMSE[m]					
		ORB-SLAM2 [5]	DynaSLAM [8]	DynaSLAM2 [18]	Dynamic SLAM [21]	**USD-SLAM(S)**	**USD-SLAM(A)**
Outdoor High Dynamic Dataset	KITTI_01	10.40	9.40	10.05	8.78	5.87	**4.54**
	KITTI_03	0.60	0.60	0.92	0.77	0.25	**0.23**
	KITTI_04	0.20	0.20	0.18	0.21	0.17	**0.16**
	KITTI_05	0.80	0.80	0.89	0.80	0.40	**0.39**
	KITTI_06	0.80	0.80	0.72	0.79	**0.53**	0.69
	KITTI_07	0.50	0.50	**0.47**	0.52	**0.47**	0.50
	KITTI_09	3.20	**1.60**	2.12	2.89	3.73	2.95

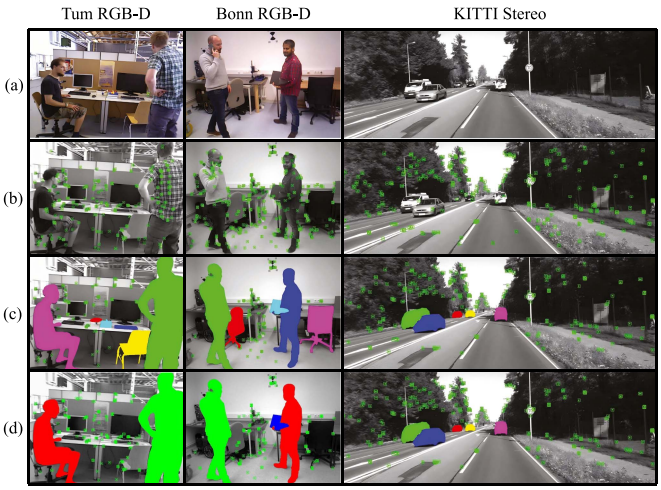

Fig. 6. Partial qualitative results. (a) Original image. (b) Extracted original ORB feature points. (c) Static feature points extracted by USD-SLAM(S). (d) Static feature points extracted by USD-SLAM(A).

B. Qualitative Analyses

Fig. 6 presents some qualitative results for three dynamic datasets, with no additional training during the experiments. From left to right are example dynamic sequence frames from the TUM, Bonn, and KITTI datasets. USD-SLAM(S) represents the algorithm that only incorporates the large segmentation model, while USD-SLAM(A) refers to the overall algorithm that integrates both the large segmentation model and 3D spatial motion constraints. The results from (b) to (c) show that the segmentation algorithm can accurately capture movable objects in the scene. The results from (c) to (d) demonstrate that the 3D spatial motion constraints can identify and remove the masks of stationary movable objects, while the masks of dynamic regions remain unchanged, as seen in the KITTI sequence. This method can globally correct the movable regions, maximizing the preservation of static regions. For example, it removes regions of walking people and moving cars while preserving regions of parked cars and stationary chairs. It can also correct segmentation errors, as shown in the TUM sequence, where 3D spatial motion constraints corrected the static regions of objects like books and keyboards that were incorrectly segmented.

C. Quantitative Analyses

This subsection provides a quantitative analysis of our method, evaluating the accuracy of our camera poses. Tables I and II present the comparison of our proposed method with other state-of-the-art SLAM methods, using the RMSE (Root Mean Square Error) of ATE (Absolute Trajectory Error) as the evaluation metric. To comprehensively validate the generalization and effectiveness of the proposed method, we conducted tests on indoor and outdoor dynamic datasets. Notably, our method operates stably without requiring additional training. Furthermore, to ensure consistency and comparability, some algorithm metrics were cited from referenced papers.

Table I presents a comparative analysis of USD-SLAM (A and S) with ORB-SLAM2 [5], DynaSLAM [8], DN-SLAM [20], and LC-CRF-SLAM [27]. The evaluation was conducted using ten highly dynamic sequences from the TUM and Bonn indoor RGB-D datasets. The results in the table show that our algorithm's localization accuracy outperforms other algorithms in most sequences. Even in the few cases with lower performance, the differences are measured only in millimeters. The lower

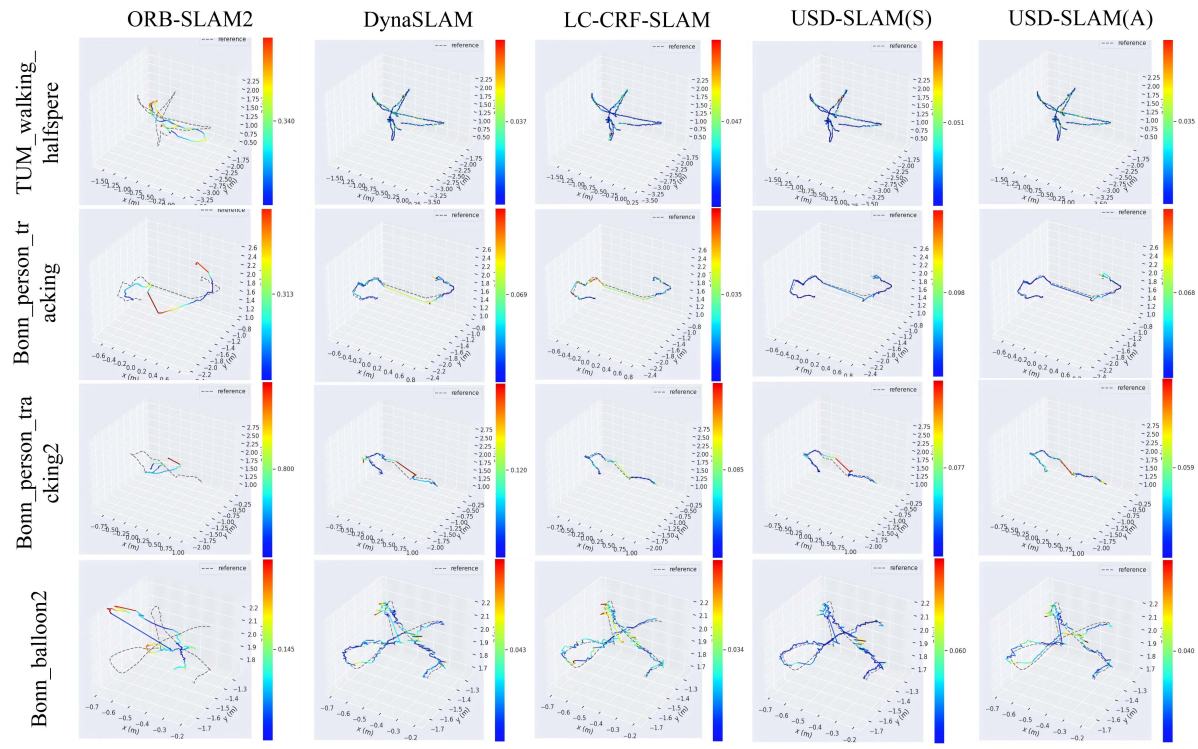

Fig. 7. In the indoor TUM and Bonn dataset sequences, EVO was used to plot the estimated camera trajectories of ORB-SLAM2, DynaSLAM, LC-CRF-SLAM, USD-SLAM(S), and USD-SLAM(A), along with their differences from the ground truth.

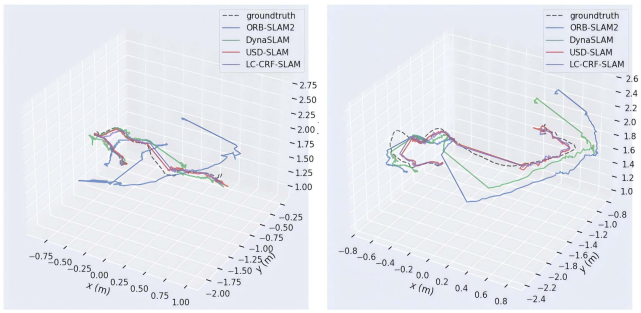

Fig. 8. Comparison of camera trajectories for ORB-SLAM2, LC-CRF-SLAM, DynaSLAM, USD-SLAM, and ground truth on the same TUM dynamic sequence.

performance in some sequences is due to the high proportion of dynamic objects in the images, resulting in insufficient static feature points after segmentation. However, even in these challenging scenarios, USD-SLAM still outperforms the ORB-SLAM2 algorithm. As shown in the trajectory comparison results in Figs. 7 and 8 plotted by EVO, our method compares favorably with other advanced algorithms, with results very close to the ground truth. Our method can stably achieve high-precision pose estimation in dynamic datasets, outperforming most current advanced algorithms.

Table II presents the localization accuracy results of our method compared with DynaSLAM2 [18] and Dynamic SLAM [21] on the high-dynamics stereo sequences of the KITTI dataset. The results in the table show that if most of the moving objects are stationary, DynaSLAM exhibits significant trajectory errors, while ORB-SLAM2 performs better. DynaSLAM2, by tracking dynamic objects, can accurately estimate their velocity even when they are temporarily stationary, providing effective pose estimation information, and making it superior to DynaSLAM. Compared to these algorithms, the localization accuracy of our system improved by 73.20%, 44.50%, 59.50%, and 53.90%, respectively. In the more complex dynamic scenarios of the outdoor KITTI dataset, our method can still accurately eliminate dynamic interference. In all dynamic sequences, our system surpasses the original ORB-SLAM2, significantly improving trajectory estimation accuracy in dynamic environments. Based on semantic information, our algorithm uses sparse 3D spatial motion intensity for comparative analysis to identify static regions at the current time for each frame. It achieves a dynamic balance between resource consumption and localization accuracy, and can still efficiently obtain high localization accuracy in unknown environments.

Ablation Experiments: We conducted ablation experiments on the dataset to validate the effectiveness of our proposed method. Under the same conditions, we compared and evaluated the ORB-SLAM2, USD-SLAM(S), and USD-SLAM(A) algorithms. The results, as shown in the last two columns of Tables I and II, demonstrate that the comparison between USD-SLAM(S) and ORB-SLAM2 confirms the effectiveness of the large segmentation model, while the comparison between USD-SLAM(S) and USD-SLAM(A) verifies the feasibility of the 3D spatial motion state constraints. Fig. 7 also shows the EVO-generated trajectory results corresponding to these experiments.

TABLE III
RUNTIME ANALYSIS UNDER THE SAME HARDWARE CONDITIONS USING THE SAME TUM DATASET SEQUENCE

Systems	Average Processing Time Per Frame (ms)
ORB-SLAM2	31.34
DynaSLAM	499.39
LC-CRF-SLAM	47.68
NO-USD-SLAM(S)	91.38
NO-USD-SLAM(A)	93.99
USD-SLAM(S)	84.95
USD-SLAM(A)	86.90

Timing Analysis: We tested the average time taken by the system to process each frame under the same conditions and compared it with other systems, as shown in Table III. In NO-USD-SLAM, "NO" indicates that the keyframe selection strategy has not been modified. After modification, the processing time was reduced by approximately 10 ms, confirming the effectiveness of the strategy adjustment. Both our algorithm and DynaSLAM utilize semantic segmentation networks, which result in significant resource consumption and relatively slow runtime. However, by employing multithreading, data preprocessing, and keyframe optimization strategies, we improved frame processing efficiency, achieving a runtime that is significantly faster than DynaSLAM. Although real-time operation has not yet been achieved, we believe that with better hardware and lightweight models, real-time processing can be realized.

V. CONCLUSION

This letter proposes USD-SLAM, a universal visual SLAM framework based on a large segmentation model and 3D spatial motion constraints, supporting stereo and depth cameras. Experiments demonstrate that our algorithm outperforms other algorithms in highly dynamic environments and can stably operate in dynamic environments. In the future, we will attempt to stably integrate monocular cameras. We also plan to make the large segmentation model lightweight and tightly coupled to improve running speed. Additionally, we aim to address the robustness issues caused by excessive dynamic objects through dynamic object tracking.

REFERENCES

[1] C. Campos, R. Elvira, J. J. G. Rodr'iguez, J. M. M. Montiel, and J. D. Tardós, "Orb-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021.

[2] V. Niculescu, T. Polonelli, M. Magno, and L. Benini, "NanoSLAM: Enabling fully onboard SLAM for tiny robots," *IEEE Internet Things J.*, vol. 11, no. 8, pp. 13584–13607, Apr. 2024.

[3] Q. Zou, Q. Sun, L. Chen, B. Nie, and Q. Li, "A comparative analysis of LiDAR SLAM-based indoor navigation for autonomous vehicles," *IEEE Trans. Intell. Transport. Syst.*, vol. 23, no. 7, pp. 6907–6921, Jul. 2022.

[4] J. Song, J. Wang, L. Zhao, S. Huang, and G. Dissanayake, "Mis-SLAM: Real-time large-scale dense deformable SLAM system in minimal invasive surgery based on heterogeneous computing," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 4068–4075, Oct. 2018.

[5] R. Mur-Artal and J. D. Tardós, "Orb-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[6] P. Kaveti, S. N. Vaidyanathan, A. T. Chelvan, and H. Singh, "Design and evaluation of a generic visual SLAM framework for multi camera systems," *IEEE Robot. Automat. Lett.*, vol. 8, no. 11, pp. 7368–7375, Nov. 2023.

[7] H. Zhan, C. S. Weerasekera, J. Bian, and I. D. Reid, "Visual odometry revisited: What should be learnt?," in *Proc. 2020 IEEE Int. Conf. Robot. Automat.*, 2010, pp. 4203–4210.

[8] B. Bescós, J. M. Fácil, J. Civera, and J. Neira, "DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 4076–4083, Oct. 2018.

[9] L. Yang and L. Wang, "A semantic SLAM-based dense mapping approach for large-scale dynamic outdoor environment," *Measurement*, vol. 204, 2022, Art. no. 112001.

[10] A. Kirillov et al., "Segment anything," in *Proc. 2023 IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 3992–4003.

[11] X. Wang, X. Zhang, Y. Cao, W. Wang, C. Shen, and T. Huang, "SegGPT: Towards segmenting everything in context," in *Proc. 2023 IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2023, pp. 1130–1140.

[12] T. Ji, C. Wang, and L. Xie, "Towards real-time semantic RGB-D SLAM in dynamic environments," in *Proc. 2021 IEEE Int. Conf. Robot. Automat.*, 2021, pp. 11175–11181.

[13] Z. shen Wang, Q. Zhang, J. Li, S. Zhang, and J. Liu, "A computationally efficient semantic SLAM solution for dynamic scenes," *Remote. Sens.*, vol. 11, 2019, Art. no. 1363.

[14] S. Qiao, Y. Zhu, H. Adam, A. L. Yuille, and L.-C. Chen, "Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 3996–4007.

[15] C. Yu et al., "DS-SLAM: A semantic visual SLAM towards dynamic environments," in *Proc. 2018 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1168–1174.

[16] Y. Zhong, S. Hu, G. Huang, L. Bai, and Q. Li, "WF-SLAM: A robust vSLAM for dynamic scenarios via weighted features," *IEEE Sensors J.*, vol. 22, no. 11, pp. 10818–10827, Jun. 2022.

[17] S. Cheng, C. Sun, S. Zhang, and D. Zhang, "SG-SLAM: A real-time RGB-D visual SLAM toward dynamic scenes with semantic and geometric information," *IEEE Trans. Instrum. Meas.*, vol. 72, 2023, Art. no. 7501012.

[18] B. Bescós, C. Campos, J. D. Tardós, and J. Neira, "DynaSLAM ii: Tightly-coupled multi-object tracking and SLAM," *IEEE Robot. Automat. Lett.*, vol. 6, pp. 5191–5198, Jul. 2021.

[19] J. Chang, N. Dong, and D. Li, "A real-time dynamic object segmentation framework for SLAM system in dynamic scenes," *IEEE Trans. Instrum. Meas.*, vol. 70, 2021, Art. no. 2513709.

[20] C. Ruan, Q. Zang, K. Zhang, and K. Huang, "DN-SLAM: A visual SLAM with ORB features and NERF mapping in dynamic environments," *IEEE Sensors J.*, vol. 24, no. 4, pp. 5279–5287, Feb. 2024.

[21] S. Wen et al., "Dynamic SLAM: A visual SLAM in outdoor dynamic scenes," *IEEE Trans. Instrum. Meas.*, vol. 72, 2023, Art. no. 5028911.

[22] M. A. Karaoglu et al., "Dynamon: Motion-aware fast and robust camera localization for dynamic neural radiance fields," 2023, *arXiv:2309.08927*.

[23] A. Cao and J. Johnson, "Hexplane: A fast representation for dynamic scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 130–141.

[24] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 573–580.

[25] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss, "Refusion: 3D reconstruction in dynamic environments for RGB-D cameras exploiting residuals," in *Proc. 2019 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 7855–7862.

[26] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The kitti vision benchmark suite," in *Proc. 2012 IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[27] Z.-J. Du, S.-S. Huang, T.-J. Mu, Q. Zhao, R. R. Martin, and K. Xu, "Accurate dynamic SLAM using CRF-based long-term consistency," *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 4, pp. 1745–1757, Apr. 2022.