2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

Outdoor Scene Dynamic Feature Point Filtering in SLAM Localization

Yimin Zhou[1], Yilun Yang[2] and Lingjian Ye[1]

Abstract— **The extraction and matching of the feature in V-SLAM is important to ensure the accuracy of the location. This paper presents a dynamic feature point filtering algorithm which combines the semantic segmentation and the geometric constraints. The algorithm performs the semantic segmentation on the preprocessed RGB images after highlight/shadow removal to initially obtain the masks of the suspected dynamic objects. Then the motion consistency detection is integrated to determine the motion states of the feature points, preserving the static features while filtering the dynamic ones, while the dealt features are subsequently used for camera motion matrix estimation. Experimental have been performed to validated on the public dataset, i.e. TUM, KITTI and custom-built dataset to demonstrate the effectiveness in V-SLAM systems.**

I. INTRODUCTION

SLAM technology is essential for autonomous robot localization, navigation and mapping in both known and unknown environments. It falls into two main categories: laser-based and visual-based SLAM. While laser SLAM offers high accuracy and strong anti-interference capability, it relies on costly lidar sensors and lacks the ability to capture environmental details such as color and texture. This limits semantic understanding and can hinder tasks like obstacle avoidance and path planning. In contrast, visual SLAM (VSLAM), which primarily uses cameras, has gained significant attention in recent years due to its lower sensor cost and powerful real-time processing capabilities.

VSLAM primarily uses cameras to extract image feature points for localization and mapping—known as the indirect approach. Meanwhile, deep learning-based methods have emerged with CNNs to directly learn mapping relationships from data for SLAM tasks.

In VSLAM, ORB-SLAM (2015) stands as one of the most notable systems with the introduction of ORB features for efficient front-end pose estimation and adoption of the multi-threaded architecture to reduce drift through loop closure detection. The system performed global and local optimization via Bundle Adjustment and the g2o framework. ORB-SLAM2 (2017) further extended this by enhancing feature extraction and keyframe selection within a thread framework, including tracking, local mapping and loop closure. ORB-SLAM3 later integrated cameras with IMUs, enabling robust real-time performance across varied environments.

Compared to feature-based indirect methods, the direct approach utilizes optical flow to compute photometric errors directly from raw images, gaining attention for its simplicity and robustness in texture-sparse or repetitive scenes. Dang et al. introduced a dynamic object handling method by fusing millimeter-wave radar and LiDAR data [5], though its performance drops in low-reflectivity or variable illumination settings. Meanwhile, Wang et al. tackled multi-sensor synchronization with OD-SLAM, which extracts features via odometry to refine pose estimation, yet its dependence on IMU data limits applicability on low-cost platforms [6].

Optical flow methods are increasingly used for detecting and tracking dynamic feature points, though they struggle with rapid motion and illumination variations. Then deep neural network was developed to predict optical flow directly from image sequences, showing higher accuracy and real-time capability in dynamic settings [7]. VINS-Mono further proposed to fuse optical flow with IMU data so as to enhance VSLAM adaptability [8]. This sensor fusion enriches environmental perception, enabling accurate dynamic feature removal and improved pose estimation in complex scenes.

Bescós et al. proposed DynaSLAM to employ Mask R-CNN to identify and remove dynamic objects in dynamic environments [9]. Furthermore, DynaSLAM2 integrates 2D instance-guided dynamic feature matching with a novel bundle adjustment scheme to jointly optimize camera poses, static points and dynamic objects [10]. While both algorithms show improved performance, their high computational demands hinder the deployment on resource-constrained devices. OVD-SLAM was developed to distinguish the foreground and background by combining semantic, depth, and optical flow information without predefined dynamic labels [11], but this method has relative large pose error.

Map construction is a key component in SLAM, such as feature maps, grid maps [12] and topological maps [13]. In VSLAM applications, point cloud maps, comprising both sparse and dense variants, are widely used. These maps aggregate spatial position parameters of objects relative to cameras. While sparse maps are generated quickly and suit scenarios requiring rapid responses [14], they capture limited geometric detail. Dense maps offer richer environmental information but demand greater computational resources [15]. The selection between sparse and dense maps directly affects VSLAM localization and navigation performance. Besides, semantic maps that integrate semantic information play an essential role in enabling robots to recognize and operate in

*This work was supported partially by National Key Research and Development Program of China Ref. 2023YFC3321600, and the Shenzhen Science and Technology Innovation Commission Project Grant Ref.JCYJ20220818101206015, Ref. JCYJ20210324101215039 and SGDX20220530111001006.

[1]Y. Zhou and L. Ye are with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences and also the University of Chinese Academy of Sciences, Shenzhen, China $\{ym.zhou, ly.ye\}$@siat.ac.cn

[2]Y. Yang is with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences and also the University of Southern Science and Technology, Shenzhen, China yl.yang@siat.ac.cn

their environments [16].

The widespread adoption of RGB-D cameras has simplified 3D map construction and spurred numerous algorithms. Kinect Fusion utilizes Kinect RGB-D cameras to acquire depth information for accurate 3D reconstruction [18]. Elastic Fusion excels in small-scale dynamic environments using an elastic fusion approach [19]. Ju et al. integrated ORB-SLAM2 with YOLOv5 to build semantic maps via voxel-based segmentation and supervoxel clustering, though accuracy requires improvement [20]. Zhao et al. combined elastic fusion with PSP-Net for semantic mapping at high computational cost but fails to filter dynamic objects [21].

This study presents an improved VSLAM algorithm based on ORB-SLAM2 to reduce the impact of dynamic objects. By integrating a YOLOv8-Seg-based segmentation network, dynamic and blurred feature points are identified and filtered so as to enhance the localization accuracy. Experiments on public road-driving datasets are performed to reduce both absolute and relative trajectory errors.

The remainder of the paper is organized as follows. Section II describes the image segmentation network based on YOLOv8-Seg for the priori information filtering, while the dynamic feature point filtering via motion consistency detection is described in Section III. Experiments are performed and result analysis are discussed in Section IV. Conclusion is given in Section V.

II. Prior Information Filtering via Semantic Segmentation

A. YOLOv8 Baseline Network

The anchor-free YOLOv8 model serves as the baseline to address the limitations of single-task networks and the poor generalization of anchor-based detectors [22]. In SLAM systems, it can accommodate real-time object detection for localization and combined detection-segmentation for integrated localization & mapping.

The YOLOv8 architecture comprises four key components: input layer, backbone network, neck and head. The backbone utilizes the CSPDarknet53 framework, while the neck employs a FAN-FPN (Path Aggregation Network-Feature Pyramid Network). The head section features two specialized output modules: a decoupled head for precise bounding box parameter prediction and a module for semantic segmentation mask generation.

The backbone builds upon YOLOv5 by replacing the C3 module with a CSP-based C2f module, incorporating the ELAN concept from YOLOv7. It utilizes CBS blocks (Convolution + Batch Normalization + SiLU) and Resx modules to maintain lightweight operation while ensuring robust gradient flow and feature preservation. The backbone concludes with an SPPF module, which employs three stacked 5×5 max-pooling layers to boost multi-scale recognition without sacrificing efficiency.

YOLOv8's neck employs a PAN-FPN module for multi-scale feature fusion, incorporating dual upsampling operations and multiple C2f modules. This culminates in a decoupled head that combines confidence prediction with bounding box regression to improve the localization accuracy. The FPN extracts multi-scale features through a bottom-up and top-down pathway, integrating high-level semantics with low-level spatial details via upsampling to enable independent predictions at each scale.

B. The Developed Bi-YOLOv8-seg Network

While YOLOv8 demonstrates robust performance in most scenarios, it exhibits limitations in accurately segmenting dynamic objects within complex dynamic scenes, particularly in distinguishing occluded or overlapped targets. Features extracted from deeper layers of the FPN often lack precise delineation of the dynamic object boundaries, as blurred edges may overlap with adjacent objects or be partially occluded by larger foreground targets, leading to localization inaccuracies.

We propose the Bi-YOLOv8-seg architecture, incorporating Bi-Backbone and Bi-Head modules. While a Bi-FPN-based feature fusion mechanism, i.e., BiYOLOv8 and Bi-Head (see Fig. 1), can preserve both shallow and deep semantics to effectively mitigate the feature degradation in dynamic scenarios. But the object localization predominantly depends on shallow features, so such hierarchical reliance would cause blurred edges to be misled by dominant sharp contours, leading to progressive information degradation until critical features are lost.

In the Bi-Head module, the top-layer feature map (20×20) is downsampled and the bottom-layer feature map (80×80) is upsampled, both integrated into the intermediate layer (40×40). This preserves the positional information of small objects from shallow layers and propagates it to larger object detectors. Furthermore, before fusion into the intermediate layer, the weighted features are directly injected into the medium-sized object detector via shortcut connections, retaining original image characteristics with minimal computational overhead. The feature fusion process in Bi-YOLOv8 can be described as,

$$\begin{cases} P_2^{td} = Conv(P_2^{in} + Resize(P_1^{in}) + Resize(P_3^{in})) \\ P_3^{out} = Conv(P_3^{in} + Resize(P_2^{td})) \\ P_2^{out} = Conv(P_2^{td} + Resize(P_3^{out}) + \omega P_2^{in}) \\ P_1^{out} = Conv(P_1^{in} + Resize(P_2^{td}) + Resize(P_2^{out})) \end{cases} \tag{1}$$

where P_x^{in} $(x = 1, 2, 3)$, P_x^{out} and P_x^{td} denote the input, output and merged component of the x^{th} layer in the feature extraction stage. P_1 is the Bottom layer (high-resolution spatial features), P_2 is the Middle layer (intermediate semantic representations), P_3 is Top layer (low-resolution semantic abstractions); ω is the weight bounded within [0,1].

It can be observed from YOLOv8 that the position of the small objects gradually diminishes during the hierarchical feature extraction process from the C2f module to the Head module. The Bi-Backbone module addresses this limitation by extracting the multi-scale features from the input images and propagating enriched contextual information to the Head module for prediction. As illustrated in Fig. 1, the proposed Bi-YOLOv8-seg backbone shares similar structure with that of the YOLOv8, comprising ten sequential modules.

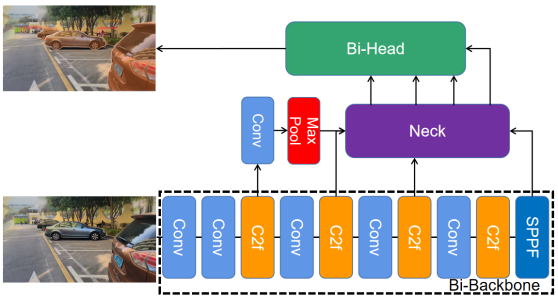

Fig. 1. The backbone of the Bi-YOLOv8-seg

The backbone network employs Convolutional layers in modules 1, 2, 4, 6, and 8, with CSP bottleneck layers in modules 3, 5, 7, and 9, followed by an SPPF layer in module 10. After the 3^{rd} layer, additional Conv and Maxpool layers resize features to 80 $\times$ 80 resolution. These processed features are then concatenated with the fifth layer's output and fed to the Neck module. Maxpool preserves essential positional information during downsampling while maintaining computational efficiency.

Through the implementation of the aforementioned dual enhancement strategies, the refined network can have higher recognition capabilities for dynamic targets in complex scenarios. Figs. 2 and 3 depict the segmentation results on the self-built dataset and TRoM dataset.

Fig. 2. The segmentation results of Bi-YOLOv8-seg on the self-built dataset

Fig. 3. The segmentation results of Bi-YOLOv8-seg on the TRoM dataset

III. Dynamic Feature Point Filtering via Motion Consistency Detection

In VSLAM systems operating in dynamic environments, the relative positional relationships among static objects remain invariant to the camera motion, while the dynamic objects introduce perturbations to the system estimation of stable spatial configurations. Hence, we propose a dynamic feature point filtering algorithm centered on motion consistency detection, called as I-VSLAM. First, it performs the in-depth analysis of feature point correspondences across consecutive frames, subsequently projecting these points into 3D space. By applying the camera motion matrix, these feature points are unified within a global coordinate system. Then the dynamic feature points inconsistent with the static environment are identified and eliminated successfully under the geometric constraints.

A. Geometric Model for Pose Estimation

The pose estimation plays a critical role used for the recovery of the camera motion trajectory via the consecutive frames. It is first to construct a geometric relationship model between frames in the video stream. Based on the theory of epipolar geometry, the fundamental matrix or essential matrix is computed, from which the camera rotation and translation transformation matrices are further derived. Epipolar geometry provides a framework independent to the scene geometry, relying instead on the camera's intrinsic parameters. This framework reveals the intrinsic projective relationship between two image frames, therefore, the camera pose estimation method based on the epipolar geometric model establishes a solid theoretical foundation for the identification and filtering of dynamic feature points.

As shown in Fig. 4, let p be a point in 3D space. The camera motion from frame t_0 to t_1 can be represented by a motion transformation matrix $[R|t]$; $\mathbf{c}_0$ and $\mathbf{c}_1$ denote the optical centers and its transformation matrix is $R_0 = I$. The plane formed by the points p, $\mathbf{c}_0$ and $\mathbf{c}_1$ is called the epipolar plane. The image plane and epipolar plane generated by the camera at t_0 and t_1 intersect at lines l_0l_1, and linec_0c_1 respectively, where the line connecting $\mathbf{c}_0$ and $\mathbf{c}_1$ intersects the respective image planes at points e_0 and e_1.

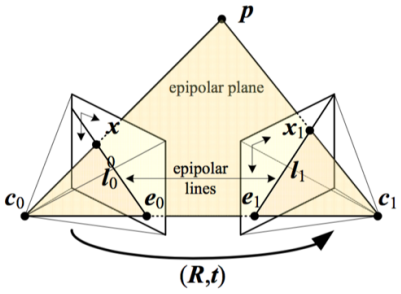

Fig. 4. The epipolar geometric model[23]

In the t_0 frame, the spatial point $p_0 = d_0\hat{x}_0$ is located at

$\hat{x}_0$ with depth d_0, whose position in t_1 frame is,

$$d_1\hat{x}_1 = p_1 = Rp_0 + t = R(d_0\hat{x}_0) + t \tag{2}$$

where $\hat{x}_j = K_j^{-1}x_j$ denote the vector at local direction. To perform the cross product of both sides with vector t, it has,

$$d_1[t]_\times\hat{x}_1 = d_0[t]_\times R\hat{x}_0 \tag{3}$$

and taking the dot product of $\hat{x}$ from both sides,

$$d_0\hat{x}_1^T([t]_\times \boldsymbol{R})\hat{x} = d_1\hat{x}_1^T[\boldsymbol{t}]_\times\hat{x}_1 = 0, \tag{4}$$

Then the basic epipolar constraints can be obtained,

$$\hat{x}_1^T E\hat{x}_0 = 0 \tag{5}$$

where $E = [t]_{\times R}$. In order to obtain the set of corresponding points between two images $\{x_i, x_{i1}\}$, it is prerequisite for calculating the essential matrix E. The set is established by setting up a homogeneous equation,

$$\begin{gathered} x_{i0}x_{i1}e_{00} + y_{i0}x_{i1}e_{01} + x_{i0}y_{i1}e_{00} + y_{i0}y_{i1}e_{11} + \\ y_{i1}e_{12} + x_{i0}e_{20} + y_{i0}e_{21} + e_{22} = 0 \end{gathered} \tag{6}$$

Under ideal conditions, the essential matrix E is singularity, i.e., $\hat{t}^T E = 0$, so it can be decomposed by SVD,

$$E = [\hat{\boldsymbol{t}}]_\times R = U\Sigma, \tag{7}$$

$$V^T = [u_0 \quad u_1 \quad t \begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} \begin{bmatrix} V_0^T \\ V_1^T \\ V_2^T \end{bmatrix}, \tag{8}$$

Since u, v are the orthogonal matrixes, σ is the singular matrix, it can be obtained,

$$\hat{t} = \pm UR_{\pm 90^\circ}\Sigma U^T \tag{9}$$

$$R = \pm U\bar{R}_{\pm 90}V^T \tag{10}$$

B. Epipolar Geometric Constraint

In the VSLAM systems, the static feature points are used for pose estimation but the dynamic feature points would degrade the precision. Theoretically, if a feature point maintains a consistent real-world position across two consecutive frames in a video stream, its projected positions in both frames should lie exactly on the corresponding epipolar lines of the current frame. In practice, however, noise from the camera and other potential disturbances could cause the projected positions of the feature points to deviate slightly within a small region around the epipolar line. Hence, a least-square optimization is formulated to address this issue,

$$s_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \boldsymbol{KT} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}, \tag{11}$$

i.e.,

$$s_i U_i = KTP_i. \tag{12}$$

We configure the least squred error item to solve the minimum value for the camera pose obtainment,

$$T^* = \arg\min_T \frac{1}{2}\sum_{i=1}^{n} \parallel u_i - \frac{1}{s_i}KTP_i \parallel_2^2 . \tag{13}$$

Another method to improve the positioning accuracy of VSLAM systems is to remove the dynamic feature points. This requires to calculate the reprojection error between the corresponding feature points in consecutive frames, which is the difference between the actual observed position of the feature point and its predicted position based on the VSLAM system pose estimation. A threshold is set to evaluate the reprojection error and filter out the feature points which do not meet the specific conditions.

Let one spatial point denoted as $P = [X, Y, Z]$, p_1, p_2 are the pixel coordinates in the image, and x_1, x_2 are the coordinates after normalization, written as:

$$x_1 = [u_1, v_1, 1]^T, \quad x_2 \quad = [u_2, v_2, 1]^T \tag{14}$$

where u_i, v_i are the pixel coordinate values. The epipolar line l_1 of the previous frame polar plan is denoted as,

$$l_1 = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \boldsymbol{F}\boldsymbol{x}_1 = \boldsymbol{F} \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} \tag{15}$$

where X, Y, Z are the directional vector of the epipolar line, so the distance from the matching point x_2 to the epipolar line l_1 is,

$$D = \frac{\mid x_2^T F x_1 \mid}{\sqrt{\parallel X \parallel^2 + \parallel Y \parallel^2}} \tag{16}$$

Based on the range of the reprojection error values, the abnormal dynamic feature points can be filtered out.

IV. Experiment and Result Analysis

The performance of the image segmentation and the VSLAM algorithm are evaluated on the public datasets (COCO 2017 and KITTI) as well as a self-built dataset.

A. Experiment of Image Segmentation Algorithm

The developed Bi-YOLOv8-seg algorithm is trained and tested on the COCO dataset with the performance comparison of YOLOv8 and YOLOv5 algorithms. After multiple iterations, it is observed that the algorithm converges after approximately 120 epochs. Based on the hardware limitation and human experience, the training parameters are set as, batchsize=8, epoch=100.

The experiments are conducted on Ubuntu 18.04 system equipped with 256 GB of RAM, NVIDIA GTX 3090 and NVIDIA GTX 3090 Ti $\times$2 GPUs, and an Intel(R) Xeon(R) Silver 4214R CPU, running with PyTorch 1.12.0+cu116, Anaconda. The evaluation criteria include mean average precision (mAP), average precision (AP), precision (P), and recall (R), where P, R are defined as,

$$P = \frac{TP}{(TP + FP)}, \quad R = \frac{TP}{(TP + FN)} \tag{17}$$

TP (True Positives) and FP (False Positives) are the correctly and erroneously segmented regions, and FN (False Negatives) is the undetected regions of the image. k is

the category number, AP measures the averaged model precision, and mAP is the mean value of AP, defined as,

$$AP = \int_0^1 p(r)dr, \quad mAP = \frac{TP}{(TP + FP)} \tag{18}$$

Ablation experiments are conducted with comparison of YOLOv5s and YOLOv8s to validate the detection efficacy of the proposed optimization method for small-scale features. For clarity and to ensure the authenticity of the experimental results, mAP@0.5 and mAP@0.5:0.95 are adopted as evaluation metrics.

The test results on the COCO 2017 dataset are summarized in Table I. It can be observed that the two modifications in Bi-YOLOv8 significantly improve the precision for small-scale features across both compact and large-scale models. Furthermore, the combined two improvements demonstrates higher recognition in small-feature targets. The enhanced feature fusion method effectively mitigates the issue where the position of small targets is overshadowed by large-scale objects during the deep learning. Meanwhile, the optimized network architecture addresses the problem of critical information loss during feature extraction caused by misleading cues from large-sized objects. Experimental results confirm that the algorithmic improvements at each stage collectively enhance the model learning capability.

TABLE I

COMPARISON OF DIFFERENT ALGORITHMS ON COCO 2017 DATASET

Algorithm	**Method**		**Structure**			
	Feature fusion	Network structure	**mAP0.5**	**mAP0.5: 0.95**	**P**	**R**
YOLOv5			56.8	49.6	62.3	67.2
YOLOv8			70.2	58.5	74.8	78.5
Bi-YOLOv8	+		72.6	59.7	76.4	80.0
Bi-YOLOv8		+	71.2	60.8	75.0	79.6
Bi-YOLOv8	+	+	74.1	62.9	78.2	82.1

To further evaluate the algorithm robustness, the trained model is tested on three datasets: COCO 2017, MPII, and PASCAL VOC 2012, with results compared and listed in Table II. The total number of the training iterations is set to 200 epochs, and both mAP@0.5 and mAP@0.5:0.95 are recorded, where Bi-YOLOv8-seg outperforms other algorithms in both standard and small-scale features.

TABLE II

COMPARISON AMONG DIFFERENT DATASETS

Dataset	**Result**	YOLOv5	YOLOv8	Bi-YOLOv8
COCO	**mAP0.5**	64	68.7	**70.5**
	mAP0.5:0.95	51.6	53.1	**54**
MPII	**mAP0.5**	42.4	50.7	**51.2**
	mAP0.5:0.95	34.2	40.3	**40.9**
VOC	**mAP0.5**	78.2	84.8	**85.3**
	mAP0.5:0.95	59.8	67.3	**68.1**

B. Experiments of the Improved VSLAM Algorithm

The performance of the I-VSLAM (Improved VSLAM) algorithm is evaluated on public datasets (TUM, KITTI) and a custom-built dataset (Dynaset). In the experiments, the datasets are divided chronologically into morning (m) and afternoon (a) subsets. To quantify the localization accuracy of the VSLAM system after dynamic feature filtering, the Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) are employed. The improvement efficacy is described by comparing the reduction rate of localization errors. The proposed algorithm is labeled as Improved, while the baseline is denoted as ORB-SLAM2. For example, $m_{Improved}$ represents the test results of the proposed algorithm on the morning dataset, and $a_{ORB-SLAM2}$ corresponds to the baseline results on the afternoon dataset, and the error reduction rate of the improved algorithm e_r is calculated as,

$$e_r = \frac{m - n}{m} \times 100\%, \tag{19}$$

where m and n are the localization errors of ORB-SLAM2 and the proposed algorithms.

Fig. 5 illustrates the feature point filtering in dynamic scenes. While numerous uniformly distributed feature points are extracted, many cluster on vehicle surfaces and highlight-shadow edges. For instance, Fig. 5(a) shows dense points on a vehicle's rear window, while Fig. 5(c) reveals similar concentrations along edges and shadows. These potentially movable features would introduce large noise into pose estimation if retained.

Fig. 5. The feature points removal in dynamic scenes

After applying the proposed highlight-shadow suppression algorithm (see [24]), the image segmentation algorithm, motion consistency constraint algorithm, the filtered feature points are shown in Fig. 5(b) and (d), demonstrating the robustness of the proposed methodology in dynamic scenes.

The SLAM trajectories are shown in Fig. 6. The ORB-SLAM2 exhibits larger discrepancies, primarily due to the presence of numerous vehicles and pedestrians along the trajectory. These variations will lead to mismatches during the feature point matching in the visual odometry process. As compared in Fig. 6(a) and (b), the improved algorithm plays a substantial role in reducing overall errors, highlighting its enhanced robustness in dynamic environments.

To thoroughly evaluate the performance of the proposed I-VSLAM algorithm in highly dynamic outdoor scenarios, the Root Mean Square Error (RMSE), Mean Error (MEAN), and Standard Deviation (STD) of the Absolute Trajectory Error

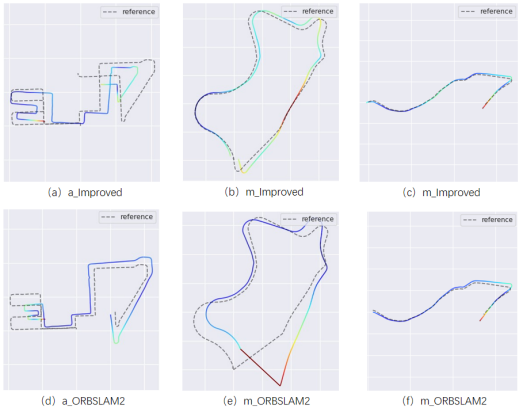

Fig. 6. The SLAM trajectory in dynamic scene

are calculated as the evaluation criteria. The comparison results are listed in Table III.

The RMSE and MEAN reflect the system robustness, while STD indicates the system stability. On the $m_dynaset$ dataset, the proposed method reduces RMSE by 77.89%, MEAN by 75.94%, and STD by 85.06%. For the a_KITTI dataset with denser dynamic objects, our I-VSLAM achieves even greater improvement: RMSE decreases by 87.25%, MEAN by 83.25%, and STD by 78.73%, demonstrating strong performance in highly dynamic environments. By removal these dynamic feature points, the localization errors generated by visual odometry can be substantially mitigated, thereby enhancing the system overall localization accuracy.

TABLE III
COMPARISON OF ORB-SLAM2 AND I-VSLAM ON DIFFERENT DATASETS

datset		m_dynaset	m_TRoM	a_dynaset	a_TRoM
ORB -SLAM2/m	ΔRMSE	0.457	0.927	0.742	0.894
	ΔMEAN	0.374	0.744	0.384	0.603
	ΔSTD	0.245	0.415	0.385	0.268
I-VSLAM/m	ΔRMSE	0.205	0.412	0.105	0.114
	ΔMEAN	0.179	0.352	0.095	0.101
	ΔSTD	0.062	0.103	0.141	0.057
e_r/%	ΔRMSE	77.89%	44.47%	85.85%	87.25%
	ΔMEAN	75.94%	8.33%	75.26%	83.25%
	ΔSTD	85.06%	73.25%	63.38%	78.73%

V. CONCLUSIONS

This paper proposes a Bi-YOLO-seg image segmentation network along with an improved I-SLAM to filter dynamic feature points with the combined semantic segmentation masks & consistency detection algorithm. By extracting the dynamic object information and analyzing the relative spatial motion of the features detected via motion consistency constraints, the dynamic features can be effectively removed, thereby improving the system stability and accuracy in dynamic environments. Experimental evaluations on the public dataset and a self-built dataset demonstrate that the proposed method can significantly enhance the localization precision in dynamic scenarios, with reduced trajectory error by an average of 87.25% compared to ORB-SLAM2 algorithm.

REFERENCES

[1] Mur-Artal R, Montiel M, TARDOS D. ORB-SLAM: a versatile and accurate monocular SLAM system, IEEE Trans. on robotics, 2015, 31(5): 1147-1163.
[2] Grisetti G, Kümmerle R, Strasdat H, et al. g2o: A general framework for (hyper) graph optimization, ICRA. 2011: 9-13.
[3] Mur-Artal R, Tardós J D. ORB-SLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras, IEEE Trans. on robotics, 2017, 33(5): 1255-1262.
[4] Campos C, Elvira R, Rodríguez G, et al. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam, IEEE Trans. on Robotics, 2021, 37(6): 1874-1890.
[5] Dang X, Liang X, et al. Moving objects elimination towards enhanced dynamic SLAM fusing LiDAR and mmW-radar, International Conference on Microwaves for Intelligent Mobility, 2020, pp. 1-4.
[6] Xu H, Yang C, Li Z. OD-SLAM: Real-time localization and mapping in dynamic environment through multi-sensor fusion, International Conference on Advanced Robotics & Mechatronics, 2020, pp. 172-177.
[7] Dosovitskiy A, Fischer P, Ilg E, et al. Flownet: Learning optical flow with convolutional networks, ICCV, 2015, pp. 2758-2766.
[8] Qin T, Li P, Shen S. Vins-mono: A robust and versatile monocular visual-inertial state estimator, IEEE Trans. on Robotics, 2018, 34(4): 1004-1020.
[9] Bescos B, Fácil J M, Civera J, et al. DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes, IEEE Robotics and Automation Letters, 2018, 3(4): 4076-4083.
[10] Bescos B, Campos C, Tardós J D, et al. DynaSLAM II: Tightly-coupled multi-object tracking and SLAM, IEEE Robotics and Automation Letters, 2021, 6(3): 5191-5198.
[11] He J, Li M, Wang Y, et al. OVD-SLAM: An online visual SLAM for dynamic environments, IEEE Sensors Journal, 2023: 13210-13219.
[12] Liu X, Ke C. Secret Sharing of Digital Raster Maps Based on Image Scrambling, International Conference on Computer Sciences & Applications, 2013, pp.18-21.
[13] Ravankar A, Ravankar A, Emaru T, et al. A hybrid topological mapping and navigation method for large area robot mapping, The 56th annual conference of Instrument and Control Engineers of Japan (SICE), 2017, pp. 1104-1107.
[14] Chen Z, Liu L. Navigable space construction from sparse noisy point clouds, IEEE Robotics & Automation Letters, 2021, 6(3): 4720-4727.
[15] Zhang F, Li Q, Wang T, et al. Dense Point Cloud Mapping Based on RGB-D Camera in Dynamic Indoor Environment, Chinese Automation Congress (CAC), 2020, pp. 2412-2417.
[16] Seichter D, Köhler M, Lewandowski B, et al. Efficient rgb-d semantic segmentation for indoor scene analysis, ICRA, 2021, pp. 13525-13531.
[17] Newcombe A, Lovegrove J, Davison J. DTAM: Dense tracking and mapping in real-time, ICCV, 2011, pp. 2320-2327.
[18] Newcombe A, Izadi S, Hilliges O, et al. Kinectfusion: Real-time dense surface mapping and tracking, 10th IEEE international symposium on mixed and augmented reality, 2011, pp. 127-136.
[19] Whelan T, Leutenegger S, Salas-Moreno R F, et al. ElasticFusion: Dense SLAM without a pose graph, Robotics: science and systems: Vol. 11. Rome, Italy, 2015: 3.
[20] Ju Q, Liu F F, Li G C, et al. Semantic map generation algorithm combined with YOLOv5, International Conference on Computer Engineering and Application (ICCEA), 2021, pp. 7-10.
[21] Lai L, Yu X, Qian X, et al. 3D semantic map construction system based on visual SLAM and CNNs, The 46th Annual Conference of the IEEE Industrial Electronics Society, 2020, pp. 4727-4732.
[22] Jocher G, Chaurasia A, Qiu J. Ultralytics YOLO[CP/OL]. 2023. https://github.com/u ltralytics/ultralytics.
[23] Oliveira P, Patete P, Baroni G, et al. Development of a bcct quantitative 3D evaluation system through low-cost solutions, The 2nd International Conference on 3D body Scanning Technologies. Citeseer, 2011, pp. 16-27.
[24] Yang Y. V-Slam Technology for Outdoor Scenes Based on Highlight Shadow Elimination and Target Segmentation, Master Thesis, Southern University of Science and Technology, 2024, in Chinese.