Real-Time Visual SLAM for Dynamic Environments using Hybrid Segmentation and Optical Flow

Yung-Ching Sun, Peng-Chen Chen, Chi Zhang, Hao Yin
{ycs, pcchenm, zhc, haoyin}@umich.edu

***Abstract*—In dynamic environments, traditional SLAM systems struggle to maintain accurate localization and mapping due to the presence of moving objects that violate the static-world assumption. To address this challenge, we propose a robust and modular dynamic SLAM framework that enhances ORB-SLAM3 by integrating real-time dynamic region segmentation and optical flow-based motion analysis. Our method leverages FastSAM and YOLO11n-seg to detect potentially dynamic regions, which are further refined using dense optical flow to identify true motion. These dynamic regions are masked to exclude moving region feature points before SLAM processing, enabling improved camera trajectory tracking. Experimental results on the TUM RGB-D and Bonn RGB-D datasets demonstrate significant improvements in localization accuracy and runtime efficiency, achieving real-time performance without requiring prior knowledge of object classes. Our project page can be found here: https://github.com/ycsun2113/HybridDyn_VSLAM.git.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) has emerged as a cornerstone in robotics, augmented reality, and autonomous navigation. Traditional SLAM systems, such as ORB-SLAM [1], perform well in static environments. However, their accuracy significantly deteriorates in dynamic scenes due to moving objects that violate the static-world assumption. Overcoming this limitation is essential for deploying SLAM-based systems in real-world, unstructured environments.

A common strategy to mitigate this issue involves detecting and masking dynamic regions. Semantic segmentation is often used to identify potentially dynamic areas. However, this approach has notable shortcomings: it may fail to detect unknown dynamic objects and often produces masks that include both static and dynamic elements, thus reducing detection accuracy.

In this work, we propose a dynamic-aware SLAM pipeline that improves the robustness of ORB-SLAM3 [2] in dynamic environments by integrating a real-time dynamic object detection and masking module. Specifically, we leverage FastSAM [3] and YOLO11n-seg [4], two recent state-of-the-art real-time segmentation models, to segment potentially dynamic regions within the scene. Optical flow is then employed to generate dynamic masks within these regions, enabling precise identification of truly dynamic areas in the scene. The resulting masks are passed into ORB-SLAM3 [2] to filter out ORB feature points in dynamic regions for estimation and tracking.

Our system is modular and efficient. We validate our approach through extensive experiments. Results show that our method significantly improves localization accuracy in dynamic scenes compared to baseline ORB-SLAM3 [2], while maintaining real-time performance.

II. Related Works

Visual SLAM (V-SLAM) has been extensively studied in recent years, primarily due to the availability of low-cost sensors and the potential for semantic information extension. V-SLAM approaches can generally be categorized into direct-based and feature-based approaches. Dense-based V-SLAM methods, such as DVO-SLAM [5], leverage dense information from RGB-D sensors, minimizing the photometric and the depth error to estimate camera motion and reconstruct the environment with high accuracy. Feature-based V-SLAM approaches, such as ORB-SLAM [1] and its variants [6, 2], rely on keypoint-based feature extraction and matching techniques (e.g., ORB features) to estimate camera trajectories and build maps of the environment. These systems are compatible with various sensor configurations, including monocular, stereo, RGB-D, or visual-inertial setups. While both dense-based and feature-based V-SLAM approaches have demonstrated high accuracy and robustness across a wide range of static environments, their performance often degrades in dynamic environments, where moving objects can lead to inaccuracies in both mapping and localization.

To address the limitations of traditional SLAM systems in dynamic environments, recent research has focused on developing SLAM systems that are robust to dynamic scenes. Many approaches detect and mask dynamic objects, such as DynaSLAM [7] enhance ORB-SLAM2 [6] by integrating object segmentation and multi-view geometry to detect and mask dynamic objects. While DynaSLAM [7] achieves significantly improved camera tracking and mapping performance in dynamic environments, this kind of methods relies heavily on pre-trained segmentation networks such as Mask R-CNN [8] and often runs offline due to the high computational demands of segmentation. Additionally, if a moving object is not recognized by the segmentation

model, it may introduce errors in SLAM estimation. Panoptic-SLAM [9] was proposed to address this issue. By using panoptic segmentation, it filters dynamic points without relying solely on labeled objects. However, its heavy computational load prevents real-time operation. OVD-SLAM [10] utilizes optical flow and object-level segmentation to jointly optimize camera poses and dynamic object trajectories, enabling more accurate scene understanding in highly dynamic settings. However, it requires reliable detection and could become unreliable or computationally intensive in complex environments. However, existing methods often suffer from high computational cost, dependence on labeled data, or limited real-time performance. These limitations motivate the need for a more efficient and generalizable approach to dynamic scene handling in SLAM.

III. Methodology

To achieve robust SLAM performance in dynamic environments containing labeled and unlabeled moving objects, we designed a SLAM system that combines object segmentation with optical flow analysis to detect dynamic regions and remove dynamic objects' keypoints. Our approach utilizes RGB-D input and consists of dynamic region masking and tracking. The overall workflow is shown as Figure 1.

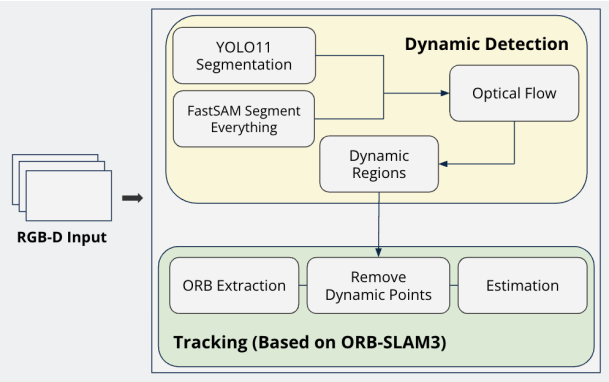

Fig. 1. System overview of our proposed method.

A. Dynamic Region Masks Generation

To improve the tracking performance of ORB-SLAM3 [2] in dynamic environments, we developed a dynamic mask generation pipeline that detects and masks highly dynamic regions from RGB images. Our method combines semantic segmentation using YOLO11n [4] segmentation model, instance segmentation using FastSAM [3] segment everything model, and motion analysis via optical flow. This hybrid approach leverages semantic and motion information to identify dynamic regions and generate masks accurately overall algorithm is shown in Algorithm 1.

1) Dataset Preparation (lines 1-2): The RGB image sequences are loaded from the defined path. All images are sorted by their timestamps.

2) Semantic and Instance Segmentation (lines 5-6): We employ two segmentation models and compare their effectiveness for dynamic object masking: (1) FastSAM [3] segment everything model and (2) YOLO11n-seg [4]. FastSAM [3] is a fast and efficient variant of Segment Anything Model (SAM) [11] trained on 2% of SA-1B with 50x faster inference. We initially chose FastSAM due to its ability to segment everything in the scene, including unlabeled objects. This characteristic is useful when there exist undetectable and unlabeled moving objects in the environment, which could lead to tracking inaccuracies. YOLO11-seg [4], on the other hand, is a lightweight segmentation variant of the YOLO11 [4] architecture. While it only segments objects with known labels, it excels at detecting dynamic object classes. We defined a list of highly dynamic object categories, such as people, cars, and animals, and prioritized them during dynamic region checking.

Through experiments, we observed that FastSAM sometimes produces fragmented segmentation and masks. For example, a moving person may only be partially segmented, which leads to inaccuracy in masking. To address this issue, we combined YOLO11n-seg with FastSAM to enhance our dynamic object segmentation. By combining two segmentation methods, we achieved a more robust approach for dynamic region masking. FastSAM allows us to capture unlabeled dynamic objects, while YOLO11n-seg provides reliable segmentation for key dynamic categories, see comparisons in Figure 2. A comparison of the results from different segmentation method combinations can be found in Section IV.

3) Optical Flow Estimation (lines 7-11): Using segmentation is insufficient to determine whether an object is moving, as not all dynamic objects belong to predefined highly dynamic classes, and FastSAM segments everything model segment unlabeled objects. To improve the accuracy, we apply optical flow estimation after segmentation to verify whether each segmented region is dynamic.

To estimate the motion between each frame, we convert two consecutive RGB frames I_i and I_{i+1} to grayscale and compute dense optical flow using the Gunnar Farneback method, which we implemented by using $cv2.calcOpticalFlowFarneback()$ function in OpenCV. Gunnar Farneback method is a method to calculate dense optical flow, which looks all points in the two frames and detects the pixel intensity change. For each pixel p, it computes the flow field:

$$F(p) = \vec{v}(p) = (u(p), v(p)) \tag{1}$$

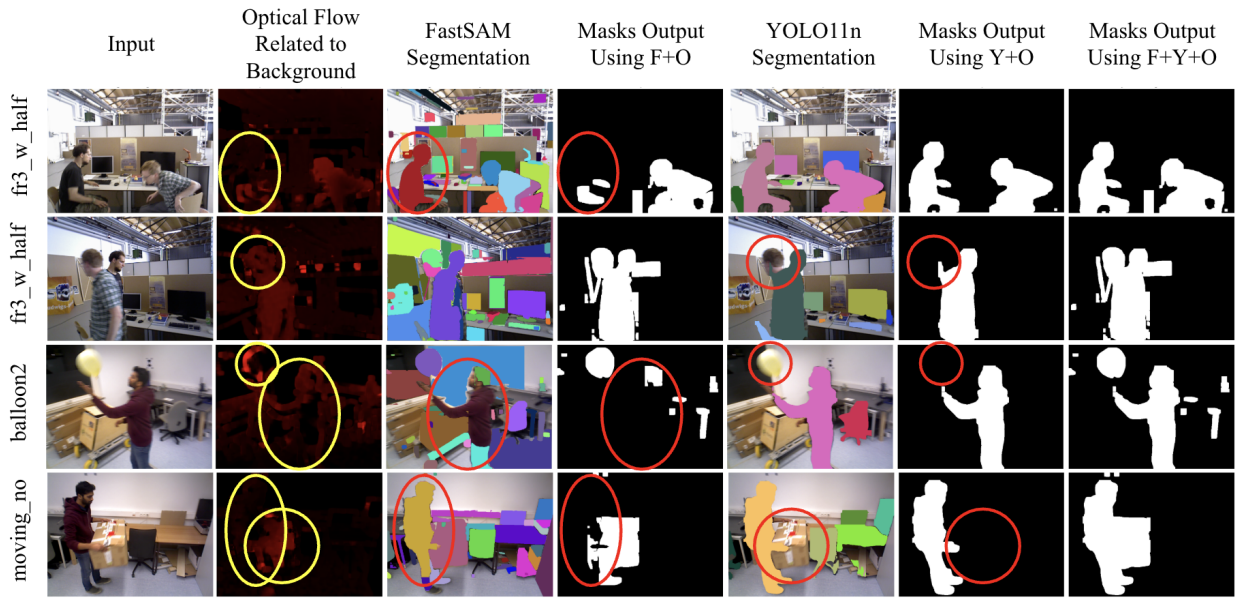

Fig. 2. Reliability comparison of dynamic region detection using FastSAM and YOLOn-seg with optical flow. Yellow circles show highly dynamic objects; red circles indicate segmentation failures. Fusing both methods leads to more robust dynamic masking.

where $u(p)$ and $v(p)$ represent the horizontal and vertical motion, respectively. Then, the flow magnitude can be calculated by:

$$M(p) = \sqrt{(u(p))^2 + (v(p))^2} \tag{2}$$

This gives a dense magnitude map that reflects pixel-wise motion between two frames.

4) Dynamic Regions Classification (lines 12-23): Due to camera movement, the entire image typically exhibits optical flow, even for static regions. Therefore, we compare the mean flow magnitude of each segmented region against that of a reference background region. If the segmented region's flow magnitude significantly exceeds the background, we classify it as a dynamic region.

We tested two methods to define the background region: (1) Depth-based: We incorporate the depth images for each frame, regions with greater depth (e.g., walls) are selected. However, this often misclassified static floor that closer to the camera as dynamic. (2) YOLO11n-seg mask exclusion: Areas outside the YOLO11n-seg masks are treated as background. This method is more reliable, as YOLO11n-seg typically does not segment static surfaces like walls and floors, while FastSAM tends to include these static regions. Then, we compute the ratio r between the masked region and the background flow:

$$r = \frac{mean\ flow\ magnitude\ in\ masked\ region}{mean\ flow\ magnitude\ in\ background\ region} \tag{3}$$

If the ratio of YOLO11-seg's mask $r_{yolo} > \epsilon_{dyna,yolo}$ or the ratio of FastSAM's mask $r_{fs} > \epsilon_{dyna,fs}$, the corresponding masked region is treated as a dynamic region. We set different threshold values for YOLO11n-seg masks ($\epsilon_{dyna,yolo}$) and FastSAM masks ($\epsilon_{dyna,fs}$) due to their segmentation characteristics. YOLO11n-seg tends to produce larger segmented masks that cover the whole object areas, resulting in lower average flow magnitudes for each region. In contrast, FastSAM segments objects more finely and more fragmentally, which can lead to higher localized flow magnitudes. If we use the same threshold for both, YOLO11n-seg masks may incorrectly classify a moving human as static, while FastSAM masks may classify many small surrounding fragments as dynamic. We found that setting $\epsilon_{dyna,yolo}$ as 1.15 and $\epsilon_{dyna,fs}$ as 1.7 yields reliable dynamic region masks.

5) Generate Binary Masks (lines 24-27): All dynamic region masks from both YOLO11n-seg and FastSAM are aggregated into a single binary mask after the above dynamic checking. To ensure full coverage that can remove the dynamic key points in ORB-SLAM3, each mask is dilated using a 9*9 morphological kernel. Then, the mask can be used by ORB-SLAM3 to remove key points in dynamic regions during tracking.

B. Dynamic Keypoints Removing and Tracking

In this section, we detail our tracking pipeline, which is built upon the fundation of ORB-SLAM3 [2]. Traditional methods often suffer from instable tracking in the dynamic environment since the moving objects will affect the pose estimation. To address this issue, we

Algorithm 1 Dynamic Region Masks Generator

Require: RGB images path img_path, YOLO11n Segmentation model M_{yolo}, FastSAM Everything model M_{fs}, A list of highly dynamic object's classes dy_obj_class;

$img \leftarrow$ ReadAndSortImages(img_path);
$N_{img} \leftarrow$ number of images;
for i = 0 : N_{img} - 1 **do**
  $I_1, I_2 \leftarrow img[i], img[i+1]$;
  $Seg_{yolo}, Seg_{fs} \leftarrow M_{yolo}(I_1), M_{fs}(I_1)$;
  $Mask_{yolo}, Mask_{fs} \leftarrow Seg_{yolo}, Seg_{fs}$ masks;
  F = calcOpticalFlowFarneback(I_1, I_2);
  $all\_masks\_yolo \leftarrow \cup$ of all Msk_{yolo} in I_1;
  $background = \neg all\_masks\_yolo$
  $F_{bg} \leftarrow$ optical flow in $background$;
  $\mu_{F_{bg}} \leftarrow$ mean optical flow in $background$;
  $dyna\_obj\_masks \leftarrow \emptyset$
  for each mask m_{yolo}, m_{fs} in $Mask_{yolo}, Mask_{fs}$ **do**
    $\mu_{F_{m,yolo}} \leftarrow$ mean optical flow in m_{yolo};
    $\mu_{F_{m,fs}} \leftarrow$ mean optical flow in m_{fs};
    $r_{yolo} = \mu_{F_{m,yolo}} / \mu_{F_{bg}}$;
    $r_{fs} = \mu_{F_{m,fs}} / \mu_{F_{bg}}$;
    if $m_{yolo}.class$ in dy_obj_class or $r_{yolo} > \epsilon_{dyna,yolo}$ **then**
      $dyna\_obj\_masks.append(m_{yolo})$;
    end if
    if $r_{fs} > \epsilon_{dyna,fs}$ **then**
      $dyna\_obj\_masks.append(m_{fs})$;
    end if
  end for
  $DyMask = zeros\_like(I_1)$;
  for each $dymask$ in $dyna\_obj\_masks$ **do**
    $DyMask[dymask] = 255$;
  end for
end for

incorporate a dynamic region mask to filter out the feature points on potentially moving objects. The key steps of our tracking thread are list below.

1) ORB Feature extraction: We begin the pose tracking by extracting ORB (Oriented FAST and Rotated BRIEF) [12] feature, which are descriptors designed for efficient and robust keypoint detection. In our system, we first extract ORB features from the raw input images. Then, these features are used to establish correspondences between the current frame and previous frames, which allow us to estimate the the camera pose.

2) Remove Dynamic Feature Points: Although we successfully obtain correspondences between the current and previous frames, feature points located on dynamic objects can significantly decrease localization accuracy. To deal with this, we use a dynamic region mask to exclude these features, which means that ORB keypoints that fall within the dynamic mask will be removed during the process. This filtering reduces the chance of including outlier correspondences caused by moving objects. Through this method, it ensures that only static scene elements are used for motion estimation, leading to more stable and accurate camera tracking in dynamic environments. In our implementation, we modify *Frame.cc* in the ORB-SLAM3 [2] source code to filter out the ORB features within the dynamic regions.

3) Estimation and Localization: After filtering out dynamic features, the remaining ORB features from static objects are used to estimate the camera trajectory and pose. Static keypoints are matched across frames, and the camera pose is computed using the Perspective-n-Point (PnP) algorithm with RANSAC to reject outliers. This results in a robust estimation of the transformation between frames. By relying solely on static features, the system achieves more accurate and reliable localization in dynamic environments.

IV. Results

A. Implementation

In our implementation, we modified the open source code from *ORB-SLAM3* to integrate our dynamic region masks for dynamic features filtering and build the docker container for ORB-SLAM3 using: *ORB-SLAM3 Docker*. Our code is available at: https://github.com/ycsun2113/HybridDyn_VSLAM.git.

B. Experimental Setup

To evaluate the performance of our dynamic-aware SLAM system, we conduct experiments on two widely used dynamic RGB-D datasets and selected three sequences from each: `fr3_w_half`, `fr3_w_xyz`, and `fr3_w_static` from TUM RGB-D [13] and `moving_no_box`, `placing_no_box`, and `balloon2` from Bonn RGB-D [14]. We compare the tracking performance of our approach against ORB-SLAM3 [2] and DynaSLAM [7], using Absolute Trajectory Error (ATE) as the primary metric. We also analyze the runtime efficiency and segmentation effectiveness using different dynamic region masking methods.

Our framework integrates YOLO11n-seg [4], FastSAM [3], and Optical Flow. We use three different dynamic region mask methods for the evaluation, which are YOLO11n-seg [4] with optical flow ($Y + O$), FastSAM [3] with optical flow ($F + O$), and combining these three together ($Y + F + O$). We then compared the localization results of our methods with ORB-SLAM3 [2] and DynaSLAM [7].

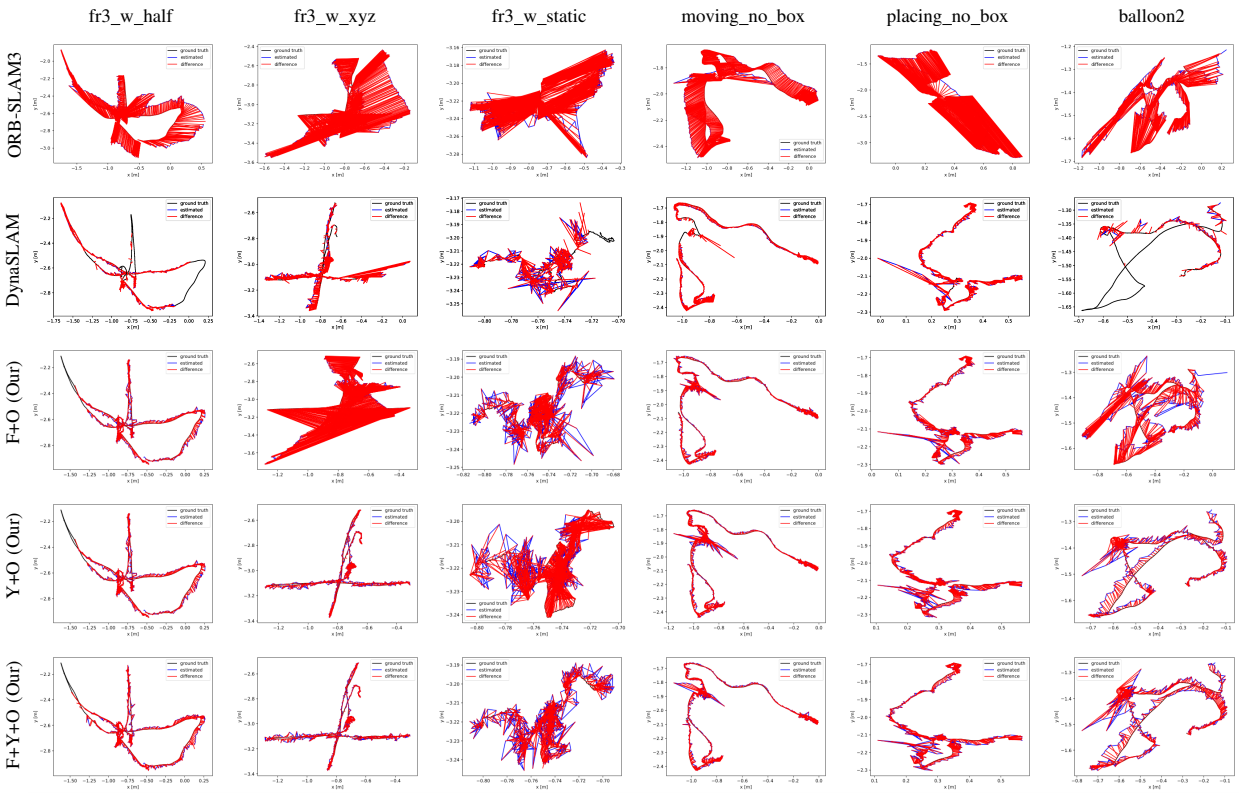

Fig. 3. Trajectory comparison across different methods. The black curve represents the ground-truth camera trajectory, the blue curve shows the estimated trajectory, and the red curves indicate the deviation between the ground-truth and estimated trajectories.

TABLE I
RESULTS OF ATE AND IMPROVEMENT RATE COMPARED WITH ORB-SLAM3 ON TUM [13] AND BONN [14] RGB-D DATASETS.

Dataset	**ORB-SLAM3** [2]	**DynaSLAM** [7]	**F+O (Our)**		**Y+O (Our)**		**Y+F+O (Our)**	
	ATE [m]	ATE [m]	ATE [m]	Improvement [%]	ATE [m]	Improvement [%]	ATE [m]	Improvement [%]
fr3_w_half	0.332	**0.025**	0.043	86.99%	0.031	90.66%	0.031	90.66%
fr3_w_xyz	0.427	0.134	0.378	11.41%	0.018	95.82%	**0.017**	**95.94%**
fr3_w_static	0.291	**0.008**	0.011	96.10%	0.015	94.82%	**0.008**	**97.34%**
moving_no_box	0.179	0.026	0.030	83.24%	0.049	72.76%	**0.025**	**86.12%**
placing_no_box	0.772	0.028	0.032	95.85%	0.031	96.04%	**0.027**	**96.49%**
balloon2	0.227	**0.028**	0.114	49.70%	0.032	85.70%	0.037	83.48%

C. Tracking Accuracy

Table I presents the Absolute Trajectory Error (ATE) results from experiments conducted on six dynamic scenes from the TUM RGB-D [13] and Bonn RGB-D [14] datasets using various methods.

As shown, ORB-SLAM3 [2] struggles with dynamic environments, it absolute trajectory errors and trajectory difference show great inaccuracies in each dynamic sequences. Our three methods achieve significant improvements compared to ORB-SLAM3, an improvement rate of over 80% was achieved in most of the sequences, demonstrating that the proposed pipeline effectively addresses ORB-SLAM's limitations in dynamic environments.

Although DynaSLAM [7] shows well ATE performance in several cases, it often suffers from tracking loss issues. As shown in Figure 3, which visualizes the ground truth and the estimated trajectories of different methods, some results for DynaSLAM [7] display only the black line (ground truth). This indicates that in certain scenarios, the method fails to maintain tracking.

In our approach, the F+O configuration achieves consistent improvements over the ORB-SLAM3 baseline in most sequences. These results highlight the capability of FastSAM in capturing dense object boundaries and handling unlabeled dynamic objects. However, performance degrades in high-dynamic scenes. Due to motion blur from rapid scene movement, FastSAM may seg-

ment objects incorrectly. Its tendency to over-segment also makes the results highly sensitive to the dynamic threshold, affecting both masking and tracking accuracy.

The Y+O configuration consistently outperforms the ORB-SLAM3 baseline, demonstrating YOLO11n's strength in identifying dynamic objects, with optical flow reducing false positives. However, its reliance on predefined classes limits its ability to detect unlabeled objects. This leads to poorer performance in `moving_no_box` and `placing_no_box`, where YOLO11n-seg [4] fails to detect the moving, non-obstructive box.

Among all the methods, the Y+F+O configuration achieves the highest tracking accuracy in four out of six sequences. As previously discussed, the motivation behind combining the two segmentation models was to leverage their complementary strengths, a strategy that has been validated by our results. By integrating object-level segmentation with YOLO11n-seg [4] and fine-grained instance segmentation with FastSAM [3], further refined through motion filtering with optical flow estimation, Y+F+O consistently outperforms other configurations. Its robustness across diverse scenarios makes it the most promising solution for dynamic SLAM.

D. Runtime Analysis

To achieve real-time applications, time efficiency is a crucial metric for evaluation. Therefore, we separately evaluate the inference time of the dynamic region mask and the tracking time. Table II presents the runtime of each method along with the hardware specifications used for testing.

TABLE II
COMPARISON OF INFERENCE AND MEAN TRACKING TIME OF DIFFERENT METHODS.

	ORB-SLAM3 [2]	**Dyna-SLAM [7]**	**F+O (Our)**	**Y+O (Our)**	**Y+F+O (Our)**
Platform (CPU)	i9	i9	i9	i9	i9
Platform (GPU)	RTX4060	Tesla M40	RTX4060	RTX4060	RTX4060
Inference Time [ms]	-	195	15.09	11.213	28.05
Tracking Time [ms]	10-16	333.68	11-18	11-18	11-18

As you can see, DynaSLAM [7] spends nearly 0.2 seconds on dynamic mask generation due to its use of Mask R-CNN, which is a relatively inefficient semantic segmentation method. In contrast, our proposed methods significantly reduce the mask inference time while maintaining competitive tracking performance. For tracking time, our approaches also show better efficiency, achieving faster performance compared to DynaSLAM [7] while preserving or even improving tracking accuracy.

In conclusion, our approach demonstrates high tracking performance with high running efficiency, making it well-suited for real-time dynamic SLAM applications.

The presentation video of our project can be found here: https://youtu.be/QZCojxtFrqo.

V. CONCLUSIONS

In this work, we propose a lightweight and effective dynamic SLAM pipeline that combines YOLO11n-seg [4], FastSAM [3], and optical flow for dynamic region masking. By filtering out dynamic objects before tracking, our system achieves robust camera pose estimation even in challenging environments with motion and occlusion. The use of YOLOv11n-seg [4] with optical flow allows for effective segmentation of dynamic objects in the scene. Also, the integration of FastSAM [3] with optical flow allows us to identify unlabeled dynamic regions that may be missed by YOLO-based methods. This combination of these two segmentation methods helps us to obtain more reliable dynamic region masks, leading to a more comprehensive and robust strategy for dynamic SLAM.

In addition, experimental results demonstrate that our method achieves the lowest or second-lowest Absolute Trajectory Error (ATE) across most sequences, showing its robust and reliable tracking performance in dynamic environments. Furthermore, the proposed approach maintains efficient inference and tracking times without experiencing tracking failures, making it well-suited for real-time applications.

VI. FUTURE WORK

While our current approach demonstrates strong performance in dynamic environments with real-time capability, several directions remain for future exploration:

- **Real-world deployment:** We aim to apply and evaluate our method in real-world robotic to assess its performance in real-time.
- **Scene mapping and understanding:** We plan to render scene map and integrate with semantic segmentation, which will allow the system to generate richer scene representations and improve high-level understanding.
- **Depth integration:** Incorporating depth information, either from stereo cameras or depth sensors, can further enhance the accuracy of dynamic object detection. This will improve the SLAM performances in more complex dynamic environments.

These improvements aim to make the system more versatile and robust for deployment in complex, dynamic real-world environments.

REFERENCES

[1] 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.

[2] 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.

[3] X. Zhao, W. Ding, Y. An, Y. Du, T. Yu, M. Li, M. Tang, and J. Wang, "Fast segment anything," 2023.

[4] G. Jocher and J. Qiu, "Ultralytics YOLO11," 2024.

[5] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," in *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2013, pp. 2100–2106.

[6] 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.

[7] 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.

[8] 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.

[9] G. F. Abati, J. C. V. Soares, V. S. Medeiros, M. A. Meggiolaro, and C. Semini, "Panoptic-slam: Visual slam in dynamic environments using panoptic segmentation," in *2024 21st International Conference on Ubiquitous Robots (UR)*. IEEE, 2024, pp. 01–08.

[10] J. He, M. Li, Y. Wang, and H. Wang, "Ovd-slam: An online visual slam for dynamic environments," *IEEE Sensors Journal*, vol. 23, no. 12, pp. 13 210–13 219, 2023.

[11] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," *arXiv:2304.02643*, 2023.

[12] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571.

[13] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[14] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss, "ReFusion: 3D Reconstruction in Dynamic Environments for RGB-D Cameras Exploiting Residuals," 2019.

[15] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for rgb-d cameras," in *2013 IEEE international conference on robotics and automation*. IEEE, 2013, pp. 3748–3754.