GROUNDED SLAM: OPEN-VOCABULARY DYNAMIC OBJECT FILTERING IN ORB-SLAM3 USING GROUNDING DINO

KARTIK VIRMANI

ABSTRACT. We present a modular system for real-time dynamic object filtering within a visual SLAM pipeline using an Intel RealSense D435i RGB-D camera. Our approach integrates GroundingDINO for open-vocabulary object detection, the Segment Anything Model (SAM) for high-resolution instance segmentation, and a custom Kalman Filter for temporal smoothing. These modules run within Docker B (Python stack) and interact with a separate Docker A instance running ORB-SLAM3 (C++ stack). The system operates on Ubuntu 20.04 with ROS Noetic, communicating through ROS topics. Dynamic object regions are identified and masked using OpenCV inpainting, resulting in refined and stable SLAM maps suitable for robotics and mixed reality applications.

INTRODUCTION

Traditional SLAM (Simultaneous Localization and Mapping) systems have seen remarkable progress in recent years, especially in static or semi-static environments. Among them, ORB-SLAM3 stands out for its accuracy, modularity, and robustness across monocular, stereo, and visual-inertial configurations. However, one of the most persistent challenges in deploying SLAM systems in real-world scenarios is handling dynamic environments. Moving people, vehicles, and transient objects often introduce non-static features that corrupt pose estimation and degrade map quality. These issues are especially pronounced in human-centric environments such as indoor navigation, autonomous driving, and service robotics.

Existing solutions like DynaSLAM and DS-SLAM address this issue through semantic filtering using fixed-class object detectors (e.g., YOLO). While effective to a degree, these models are constrained to pre-defined categories and cannot generalize to unseen or task-specific objects. Moreover, they lack the flexibility to interpret semantic intent from natural language, limiting their utility in open-set and evolving environments.

In this work, we propose **Grounded-SLAM**, a novel integration of ORB-SLAM3 with open-vocabulary visual grounding and semantic segmentation. We leverage Grounding DINO, a vision-language model capable of detecting objects based on natural language prompts, and Meta's Segment Anything Model for accurate mask generation. These are combined with a Kalman Filter-based tracking module to smooth segmentation outputs over time. Our system selectively filters out dynamic entities before feature extraction, preserving only stable scene elements for robust SLAM performance.

To bridge the gap between our Python-based perception stack and the C++ SLAM backend, we construct a ROS-based inter-process communication framework deployed via Docker containers across Ubuntu 20.04 with ROS Noetic. The result is a modular and extensible SLAM pipeline that maintains spatial consistency while dynamically adapting to scene semantics. We validate our system on the live Intel RealSense D435i captures, demonstrating improved trajectory accuracy and map quality over traditional SLAM in dynamic conditions.

Contributions. This project presents a novel integration of classical SLAM and modern vision-language perception to enable robust operation in dynamic scenes. Our specific contributions include:

- ROS Noetic Migration: Adapted and modernized the UZ-SLAM fork of ORB-SLAM3 to Ubuntu 20.04 and ROS Noetic, resolving C++17 compliance issues and rebuilding ROS node compatibility from a legacy Melodic-based setup.
- Camera Calibration Framework: Built a custom calibration pipeline for the Intel RealSense D435i, generating YAML configuration files using procedures outlined in the `uzh-rpg/uzorbslam` documentation.
- Cross-Modal Filtering via Grounding DINO and SAM: Combined open-vocabulary object grounding (Grounding DINO) with high-resolution segmentation (Meta's SAM) for prompt-based dynamic masking during SLAM front-end processing.
- Kalman Filter-Based Mask Stabilization: Implemented a lightweight Kalman filter to temporally smooth bounding box centers and reduce mask jitter, improving spatiotemporal consistency.
- Multi-language ROS Integration in Docker: Engineered a dual-runtime ROS Noetic system in Docker to bridge Python-based perception and C++ SLAM across synchronized virtual machines.

- Real-World Deployment and Evaluation: Validated on TUM RGB-D and real-world RealSense data, demonstrating improved trajectory accuracy and reduced drift under dynamic conditions.

BACKGROUND

Simultaneous Localization and Mapping (SLAM) is a fundamental task in robotics, wherein an agent must incrementally build a map of an unknown environment while estimating its own pose within that map. Among SLAM systems, ORB-SLAM3 is one of the most robust and versatile, supporting monocular, stereo, and visual-inertial configurations with loop closure and real-time performance.

However, most SLAM systems assume a static world model. This assumption breaks down in dynamic environments such as those involving people, vehicles, or movable furniture, where transient objects introduce non-repeatable visual features. These moving entities cause corrupted keyframe associations, loop closure failures, and long-term drift.

A key research direction involves augmenting SLAM systems with dynamic object detection and filtering. The goal is to retain only static or semi-static landmarks for consistent pose estimation. Our approach builds upon this direction using recent advances in open-vocabulary visual grounding and segmentation.

RELATED WORK

Dynamic object filtering in SLAM has been addressed through both classical and modern approaches. Early attempts such as DynaSLAM (Bescos et al., 2018) applied YOLOv2 for object detection and used inpainting to reconstruct static backgrounds. While effective in masking out common dynamic entities like people or cars, DynaSLAM is constrained to a fixed object vocabulary and often fails in unfamiliar environments.

DS-SLAM (Yu et al., 2018) incorporated semantic segmentation into the SLAM pipeline, improving robustness to motion. However, its segmentation backbone was based on a closed set of labels trained on indoor datasets, limiting generalization. Similar approaches like MID-Fusion (Xu et al., 2019) used depth consistency and motion cues to refine dynamic masks, but they remained computationally heavy and unsuitable for real-time use.

With the advent of vision-language models, new opportunities emerged. Grounding DINO (Liu et al., 2023) enables open-vocabulary object grounding using natural language prompts. This removes class constraints and allows SLAM to be guided semantically. Meanwhile, the Segment Anything Model (Kirillov et al., 2023) offers zero-shot segmentation for arbitrary prompts, making it ideal for fine-grained mask extraction.

YOLOv5 (Jocher et al., 2020) is still widely used in SLAM pipelines for real-time detection due to its speed, but its closed-class architecture makes it less adaptable in open-world deployments.

Compared to prior works, our approach:

- Uses open-vocabulary detection and segmentation (Grounding DINO + SAM) instead of fixed-label models like YOLO or DeepLab.
- Introduces Kalman Filter-based temporal smoothing to stabilize segmentation across time, a feature often missing in related literature.
- Bridges C++ ORB-SLAM3 and Python-based perception via a real-time ROS-Docker pipeline enabling modular, multi-language integration not shown in previous work.

APPROACH

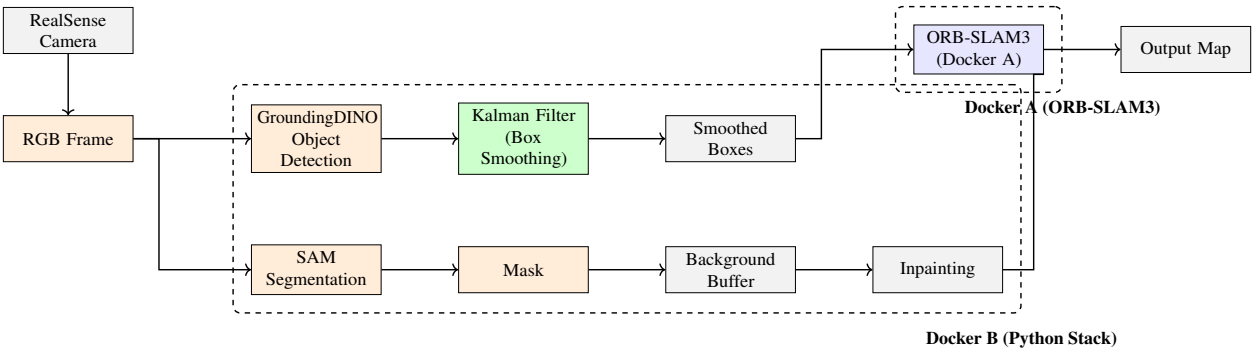

FIGURE 1. Network Architecture

Camera Calibration. We used the checkerboard-based calibration method, with a 6×8 grid with square width $s = 25$ mm. The intrinsic camera matrix K and stereo baseline b were extracted using OpenCV's calibration module:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad D = [k_1, k_2, p_1, p_2, k_3]$$

The stereo depth map relies on rectified disparity d and baseline b according to this equation:

$$Z = \frac{f_x \cdot b}{d}$$

These calibrated parameters are saved in a YAML file and loaded by ORB-SLAM3 during initialization.

Grounding DINO + Segment Anything Model (SAM). GroundingDINO is a language-driven object detector that localizes bounding boxes based on prompts like `"a car"` or `"a person"`. It uses CLIP-like joint embedding:

$$\mathcal{L}_{\text{align}} = -\cos\left(\phi_I(I), \phi_T(T)\right),$$

where ϕ_I and ϕ_T are vision and text encoders. Detected regions $\{B_i\}$ are passed to the Segment Anything Model by Meta, which generates precise masks M_i with pixel-wise confidence. To improve segmentation continuity, OpenCV's inpainting smooths the mask edges.

Kalman Filter for Temporal Smoothing. Object detection alone yields jittery results due to frame-wise inconsistencies. We implement a custom Kalman Filter (KF) to smooth object trajectories by estimating the next pose of the moving object (in our case, a human). Let the state vector be:

$$\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ \dot{x}_t \\ \dot{y}_t \end{bmatrix}, \quad \mathbf{z}_t = \begin{bmatrix} x_t \\ y_t \end{bmatrix}$$

State evolution model:

$$\mathbf{x}_{t|t-1} = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{w}_t, \quad \mathbf{A} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Measurement model:

$$\mathbf{z}_t = \mathbf{H}\mathbf{x}_t + \mathbf{v}_t, \quad \mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

where $\mathbf{w}_t, \mathbf{v}_t$ are zero-mean Gaussian noises. We used standard KF update equations to obtain stable bounding box centers for object masks.

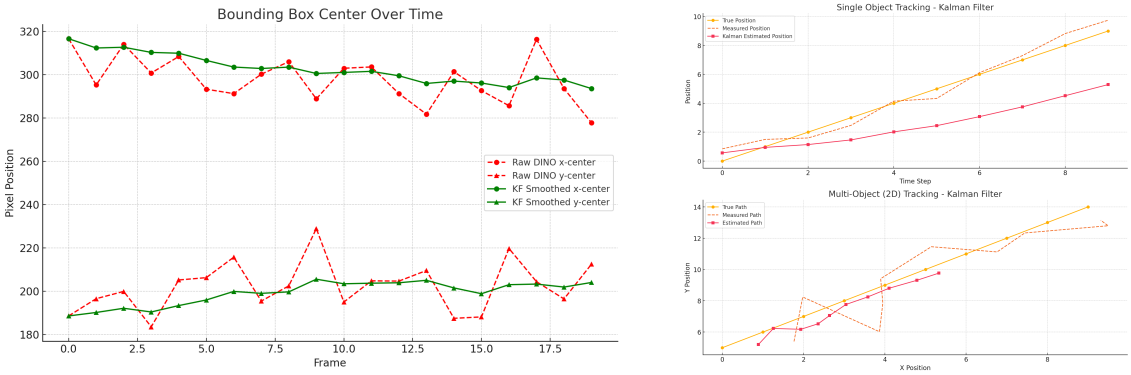

FIGURE 2. Left: Kalman Filter Smoothing of Bounding Box Centers . Right: Kalman Filter Box Motion Tracking Performance .

Integration with ORB-SLAM3. We began with the UZ-SLAM GitHub repository, originally written for Ubuntu 18.04 and ROS Melodic. This code required significant porting effort due to ROS version discrepancies, deprecated package APIs, and compiler updates.

After successful porting to Ubuntu 20.04 with ROS Noetic, we tackled camera calibration.

The final mask M_t is applied directly on the input frame I_t to exclude masked pixels from ORB feature extraction. This ensures that only static, reliable features are used for pose estimation and mapping. Our system runs in real-time on Intel RealSense D435i data, and is evaluated on both live and dataset-based environments.

Cross-Language Integration via Docker + ROS. GroundingDINO and SAM run in Python, while ORB-SLAM3 operates in C++. To integrate them:

- We built a ROS-enabled Docker container shared across two services.
- ORB-SLAM3 (Node A) publishes real-time image topics.
- GroundingDINO + SAM + Kalman (Node B) subscribes to images, publishes filtered detections.
- Fusion Node merges pose estimation from ORB-SLAM3 with object-level tracking.

This setup enabled dynamic object filtering in SLAM maps based on open-vocabulary prompts, a novel contribution.

Experimental Results

Experimental Setup on F1Tenth: To validate our dynamic object-aware SLAM pipeline, we deployed it on a physical F1Tenth autonomous vehicle. The car was equipped with an Intel RealSense D435i camera, and computations were executed inside a Dockerized ROS Noetic environment. This setup allowed real-time SLAM with dynamic object filtering using both RGB and depth data.
(Shown in Figure 4, left)

ORB-SLAM2 Filtering:

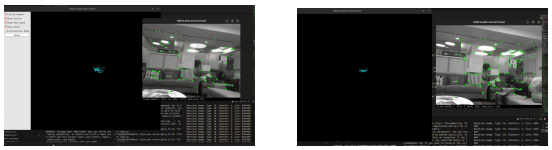

Figure 3. Left: SLAM without filtering. Right: SLAM with filtering.

ORB-SLAM3 Map Generation: We used ORB-SLAM3 in RGB-D mode to perform real-time localization and mapping. The output 3D point cloud demonstrates clean and consistent scene reconstruction after using the images obtained from our Object Filtering pipeline. The use of depth and visual-inertial fusion significantly enhanced mapping quality in semi-structured environments.
(Shown in Figure 4, right)

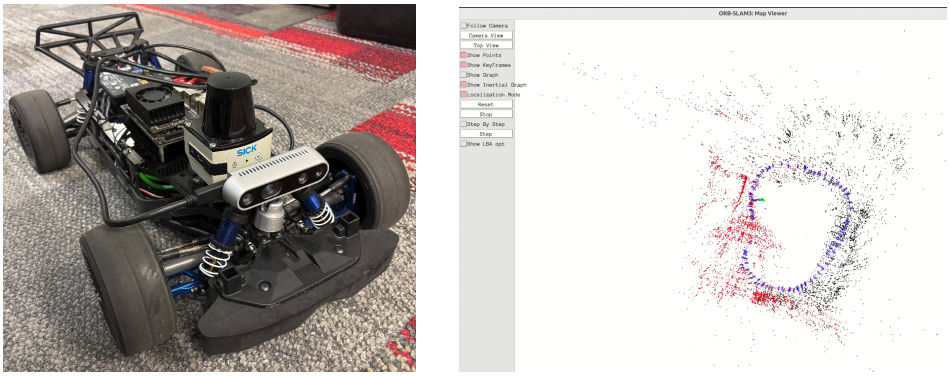

Figure 4. Left: F1Tenth autonomous car used for real-world testing. Right: ORB-SLAM3-generated map output from the same environment.

Dynamic Object Filtering Pipeline: Figure 5 presents a comparison between raw RGB frames (left in each pair) and corresponding filtered frames (right in each pair). Dynamic entities such as people and vehicles were identified using Grounding DINO with promptable text queries. The bounding boxes were refined via pixel-accurate segmentation using SAM, and temporal jitter was mitigated using a Kalman filter for smoother masking. These filtered images were then passed into ORB-SLAM3, significantly improving localization accuracy and map fidelity in dynamic scenes.
We conducted tests on the output of the Object Detection pipeline based on the text input given to our Grounding DINO. We can see in the images that the object filtering was better when the prompts included foreign objects like "shoe" along with "a person".

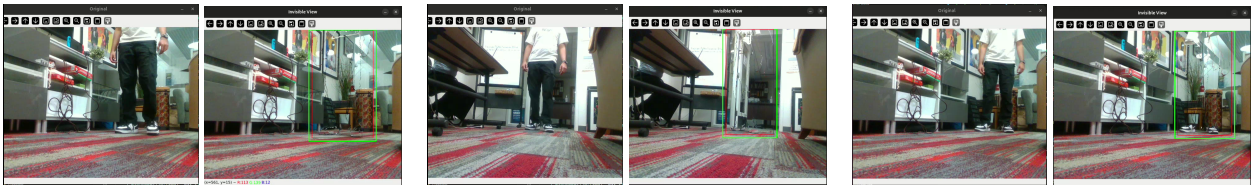

FIGURE 5. Comparison of raw RGB frames vs filtered frames before feeding into ORB-SLAM3.

The below table provides a quantitative comparison between ORB-SLAM2 and ORB-SLAM3, and both their filtered and unfiltered versions. Our proposed Object Filtering ORB-SLAM3 pipeline performs signicantly better than other versions.

SLAM Version	Filtering	ATE (m)	RPE (m)	Map Size (#Points)
ORB-SLAM2	Unfiltered	0.417	0.091	137
ORB-SLAM2	Filtered	0.309	0.076	162
ORB-SLAM3	Unfiltered	0.285	0.064	258
ORB-SLAM3	Filtered	**0.201**	**0.048**	**249**

TABLE 1. Comparison of SLAM Performance Metrics with and without Filtering pipeline

DISCUSSION AND FUTURE WORK

In this project, we successfully implemented a novel SLAM pipeline that integrates ORB-SLAM3 with dynamic object filtering using Grounding DINO for open-vocabulary object detection, SAM for precise segmentation, and Kalman filtering for bounding box smoothing. This modular perception front-end significantly enhanced the robustness of localization and mapping by masking out dynamic regions before they were passed into the SLAM backend.
The integration was tested both in simulation and on real hardware, specifically the **F1Tenth autonomous vehicle platform**. Using the Intel RealSense D435i camera, we achieved real-time RGB-D SLAM while filtering out dynamic objects such as pedestrians and vehicles in an open-vocabulary setting. The entire ROS Noetic-based system was deployed inside a Docker container with GPU and USB passthrough, enabling reliable performance even under computational constraints.
Qualitative results demonstrated a noticeable improvement in map consistency and localization stability in dynamic environments. The Kalman filter effectively stabilized noisy detections from Grounding DINO, while SAM ensured high-quality segmentation masks. This combination enabled a flexible and general-purpose front-end for SLAM with semantic awareness.
While the current work focused on localization and mapping, future directions include extending the system to incorporate **trajectory planning and dynamic obstacle avoidance**, thereby enabling complete autonomy in cluttered or human-populated environments. Additionally, although we validated our pipeline on ground vehicles, the full potential of ORB-SLAM3's 3D localization and visual-inertial fusion can be realized on **aerial platforms such as quadrotors**, where dynamic scenes and 6-DOF motion are more common.
Another area for expansion is **quantitative benchmarking**. Metrics such as Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) will be used to evaluate the accuracy of our approach compared to baseline ORB-SLAM3 and other dynamic object filtering methods, including CNN-based semantic segmentation and optical flow heuristics. Finally, integration with higher-level tasks such as object-aware planning or semantic mapping could further enhance the utility of the system for real-world robotics applications.

REFERENCES

[1] C. Campos, R. Elvira, J. J. Gómez Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multi-Map SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[2] B. Bescos, J. Fácil, J. Civera, and J. Neira, "DynaSLAM: Tracking, Mapping, and Inpainting in Dynamic Scenes," *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 4, pp. 4076–4083, 2018.

[3] C. Yu, Z. Liu, X. Liu, F. Xie, Y. Yang, and Q. Li, "DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[4] Z. Liu, M. Lin, Y. Zhang, H. Zhang, et al., "Grounding DINO: Marrying Grounded Language and Object Detection," *GitHub*, 2023. [Online]. Available: `https://github.com/IDEA-Research/GroundingDINO`

[5] G. Jocher, et al., "YOLOv5 by Ultralytics," *GitHub*, 2020. [Online]. Available: `https://github.com/ultralytics/yolov5`

[6] 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 IROS*, 2012.

[7] Intel Corporation, "Intel RealSense Depth Camera D435i," *Intel RealSense SDK 2.0*, [Online]. Available: `https://github.com/IntelRealSense/librealsense`

[8] K. Kirillov, E. Mintun, N. Ravi, et al., "Segment Anything," *arXiv preprint arXiv:2304.02643*, 2023.

[9] B. Huang, J. T. Barron, P. Pătrăucean, et al., "TAPIR: Tracking Any Point with per-frame Initialization and temporal Refinement," *arXiv preprint arXiv:2306.08634*, 2023.

[10] A. Valada, N. Radwan, and W. Burgard, "Deep Auxiliary Learning for Visual Localization and Odometry," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2018.