GPU-Accelerated Semantic Embedded SLAM

Calvin Galagain*†, Martyna Poreba*, François Goulette†
* Université Paris-Saclay, CEA LIST, F-91120 Palaiseau, France
† U2IS, ENSTA, Institut Polytechnique de Paris, F-91120 Palaiseau, Fran
{calvin.galagain, martyna.poreba}@cea.fr, francois.goulette@ensta.fr

***Abstract*—Long-term robot deployments require SLAM systems that remain reliable under dynamic scene changes while meeting embedded runtime constraints. We address this problem by extending ORB-SLAM3 with an asynchronous semantic filtering pipeline that removes features on potentially dynamic classes before geometric tracking. To keep this semantic SLAM design compatible with embedded execution, we pair it with a CUDA ORB front-end that moves pyramid construction, FAST scoring, non-maximum suppression, orientation estimation, and descriptor extraction to the GPU. On TUM RGB-D sequences, *SemCUDA-SLAM* increases mean throughput from 33.2 FPS to 47.8 FPS (+44.1%) while reducing mean ATE from 0.17 m to 0.02 m.**

I. Introduction

Deploying visual SLAM over long periods in the wild requires embedded systems that maintain both robustness and real-time efficiency despite scene changes and moving objects. ORB-SLAM3 [2] provides a strong geometric backbone with tracking, local mapping, loop closing, and global optimization threads, but two limitations remain important in this context. First, the visual front-end is computationally heavy. For every frame, it builds a multi-scale pyramid, applies FAST detection [7], performs non-maximum suppression (NMS), estimates orientations, and computes ORB descriptors, which dominate runtime on Jetson-class hardware. Second, ORB-SLAM3 is purely geometric at the observation level and does not reject features on potentially dynamic objects, which can degrade pose estimation or trigger map resets in dynamic scenes. Existing approaches typically address these challenges independently. GPU-accelerated SLAM systems focus on improving computational efficiency, while semantic SLAM methods enhance robustness to dynamic environments by incorporating segmentation or object-level reasoning. However, combining both aspects in a way that remains compatible with embedded deployment remains non-trivial.

We propose *SemCUDA-SLAM*, a semantic extension of ORB-SLAM3 that jointly addresses these two challenges. Our approach integrates an asynchronous semantic filtering pipeline that removes features on potentially dynamic classes before geometric tracking, together with a CUDA-accelerated ORB front-end that preserves real-time performance on embedded hardware. This approach enables efficient and robust visual SLAM on embedded platforms such as the NVIDIA Jetson Orin 64GB.

II. Related Work

Geometric SLAM systems remain attractive due to their strong trade-off between accuracy, robustness, and computational efficiency. ORB-SLAM2 [5] and ORB-SLAM3 [2] demonstrated that sparse ORB-based pipelines can achieve high-quality tracking and loop closure in real time across monocular, stereo, RGB-D, and visual-inertial settings. Their modular design makes them suitable baselines for studying front-end modifications. The visual front-end is a natural target for embedded acceleration. In a ORB-SLAM pipeline, GPU acceleration is most effective for highly parallel components. ORB feature extraction and feature matching are well-suited due to their data-parallel nature. In local mapping, triangulation and parts of bundle adjustment can be partially parallelized. Loop closure also benefits from GPU acceleration in global search and large-scale matching. In contrast, graph optimization and heuristic pipeline decisions are less suitable due to irregular structures and control dependencies. Consequently, efficient designs typically adopt a hybrid CPU-GPU approach. Several embedded SLAM systems explore GPU acceleration of these stages, often combined with front-end simplifications such as reduced pyramid depth or lower input resolution to increase throughput [6], [12], [13].

Semantic SLAM approaches such as DynaSLAM [1], DS-SLAM [9], and MaskFusion [10] show that semantic cues improve SLAM robustness in dynamic scenes, but they integrate semantics in different ways and with different model classes. DynaSLAM relies on a heavy per-frame instance segmentation model (Mask R-CNN) combined with multi-view geometric checks to remove dynamic regions. DS-SLAM uses a semantic segmentation network (SegNet) within a parallel semantic thread for dynamic object filtering and semantic map construction. MaskFusion goes further by tightly coupling instance-level segmentation with dense object-aware tracking and reconstruction.

III. Method

A. Semantic SLAM Design

The main design choice is to incorporate semantic information at the observation level. Rather than replacing geometric SLAM, the objective is to exclude potentially dynamic features from the tracking pipeline at an early stage. To achieve this, a lightweight semantic model runs in a separate thread on RGB frames, and its predictions are used to filter features prior to geometric tracking. In contrast to approaches that

couple semantics with dense reconstruction, semantic mapping, or backend optimization, our method uses semantics only to regulate the visual observations entering ORB-SLAM3. This keeps the geometric backend unchanged and yields a lightweight design that remains compatible with real-time embedded execution.

B. GPU-Only Front-End

SemCUDA-SLAM is built directly on the RGB-D baseline of ORB-SLAM3. The backend is unchanged, and only the ORB observation pipeline is moved to CUDA, including pyramid construction, FAST response computation, non-maximum suppression, orientation estimation, and descriptor extraction. The resulting keypoints and descriptors are converted back into the standard ORB-SLAM3 representation, so the rest of the pipeline remains unchanged. This GPU-only variant is not numerically identical to the baseline. Small differences on the ORB observation pipeline propagate to matching, pose estimation, and map creation. The backend is therefore unchanged, but the observation process is not exactly the same as in baseline ORB-SLAM3.

C. Asynchronous Semantic Filtering

The semantic branch is executed with TensorRT (FP16) and runs asynchronously from tracking. For each RGB frame, it predicts a dense label map using TokenMask [11], from which a binary mask of potentially dynamic classes is derived. The latest mask is stored in a shared buffer and consumed by tracking without blocking. Semantic filtering is applied after ORB extraction: features falling on potentially dynamic classes are removed before geometric tracking. This preserves the geometry-first structure of ORB-SLAM3 while suppressing harmful observations. Semantic labels are also propagated to keyframes and map points for visualization.

IV. Experimental Setup and Results

We evaluate on TUM RGB-D sequences [8], reporting tracking FPS, ATE RMSE, and relative pose error (RPE) following the standard evaluation protocol. FPS measures computational efficiency, ATE RMSE evaluates global trajectory accuracy in meters, while RPE_r capture local rotational drift in degrees. Results are summarized in Table I. We compare three variants: baseline ORB-SLAM3, its GPU-accelerated version, and the final *SemCUDA-SLAM* method. The GPU-accelerated variant achieves the highest throughput, increasing FPS from 33.16 to 62.55 (+88.6%), while significantly improving accuracy with ATE reduced from 0.173 m to 0.0266 m and RPE_r from 0.3303 to 0.2842 deg. The proposed *SemCUDA-SLAM* method provides a balanced trade-off between efficiency and accuracy. It increases FPS to 47.78 (+44.1%) compared to the baseline, while achieving the lowest ATE of 0.015 m. In terms of relative pose error, it maintains low rotational drift (0.2932 deg), comparable to the GPU-accelerated variant and lower than the baseline.

TABLE I
Average performance on selected TUM RGB-D sequences.

Variant	FPS ↑	ATE RMSE [m] ↓	RPE_r [deg] ↓
ORB-SLAM3	33.16	0.1730	0.3303
ORB-SLAM3 (GPU)	62.55	0.0266	0.2842
SemCUDA-SLAM	47.78	0.0150	0.2932

V. Conclusion

We presented a semantic extension of ORB-SLAM3 for embedded RGB-D SLAM in dynamic environments. The main contribution is an asynchronous semantic filtering pipeline that removes potentially dynamic observations before geometric tracking. This design is made practical by two complementary choices: a GPU-accelerated ORB front-end that reduces the cost of visual feature extraction, and a lightweight semantic module whose asynchronous implementation allows semantic predictions to be integrated at the frame level rather than only at keyframes. The semantic variant remains substantially faster than baseline while recovering a much more stable observation stream on the most challenging sequences. For long-term embedded operation in changing environments, these results suggest that semantic filtering is the main source of robustness, while efficient front-end acceleration and lightweight semantic inference make such filtering compatible with real-time embedded deployment.

References

[1] 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.

[2] 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.

[3] M. Cordts *et al.*, "The Cityscapes dataset for semantic urban scene understanding," in *Proc. CVPR*, 2016, pp. 3213–3223.

[4] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *J. Opt. Soc. Am. A*, vol. 4, no. 4, pp. 629–642, 1987.

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

[6] A. Kumar, J. Park, and L. Behera, "High-speed stereo visual SLAM for low-powered computing devices," *arXiv preprint arXiv:2410.04090*, 2024.

[7] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. ECCV*, 2006, pp. 430–443.

[8] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IROS*, 2012, pp. 573–580.

[9] C. Yu, Z. Liu, X. J. Liu, F. Xie, Y. Yang, and Q. Wei, "DS-SLAM: A semantic visual SLAM towards dynamic environments," in *Proc. IROS*, 2018, pp. 1168–1174.

[10] M. Runz, M. Buffier, and L. Agapito, "MaskFusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *Proc. ISMAR*, 2018, pp. 10–20.

[11] C. Galagain, M. Poreba, and F. Goulette, "Token-Space Mask Prediction for Efficient Vision Transformer Segmentation," 2026.

[12] K. Khabiri, P. Hosseininejad, S. Gopinath, K. Dantu, and S. Y. Ko, "Fast-Track: GPU-Accelerated Tracking for Visual SLAM," *arXiv preprint arXiv:2509.10757*, 2025. Available: https://arxiv.org/abs/2509.10757

[13] P. Hosseininejad, K. Khabiri, S. Gopinath, S. Mohammadhashemi, K. Dantu, and S. Y. Ko, "TurboMap: GPU-Accelerated Local Mapping for Visual SLAM," *arXiv preprint arXiv:2511.02036*, 2025.