MDPI

Article

Robust real-time automotive Visual SLAM with dynamic object removal

Luka Sachße [1,†], Olena Horokh [1,†], Jan Fischer [1,†] and Robert Bensch [1*]

[1] ANavS GmbH, Gotthardstraße 40, 80686 München, Germany; info@anavs.de
[*] Correspondence: luka.sachsse@anavs.de (L.S.), olena.horokh@anavs.de (O.H.), jan.fischer@anavs.de (J.F.), robert.bensch@anavs.de (R.B.)
[†] These authors contributed equally to this work.

Abstract: Visual Simultaneous Localization and Mapping (SLAM) is a method that relies on visual feature tracking to estimate the camera motion while creating a map of the environment. It is crucial for autonomous navigation of robots, vehicles and drones in GNSS-denied environments (urban canyons, tunnels, indoors) and any environment with jamming / spoofing. SLAM algorithms generally assume that features in the observed environment belong to static and rigid objects. Thus, in crowded and dynamic environments such as urban traffic, the algorithm's performance in terms of camera motion estimation is heavily affected by the large amount of dynamic objects observed. To address this challenge, an innovative real-time method for the detection and exclusion of moving objects in the motion estimation stage of a Visual SLAM frontend is presented. We implement our method on a real-vehicle, evaluate it on multiple public datasets and prove that the removal of dynamic objects leads to increased accuracy and robustness of the position solution. This work was conducted under the EU-funded DREAM project.

Keywords: Visual SLAM; Semantic SLAM; dynamic environments; autonomous navigation; instance segmentation; dynamic object removal; real-time localization

1. Introduction

Visual SLAM is an extensively studied research topic and a core technique for visual navigation. While fundamental work and significant progress have been achieved in the field of Visual SLAM in recent years, numerous challenges remain, such as its application in difficult environments, e.g., texture-poor or highly dynamic environments.

Visual SLAM systems rely purely on visual feature tracking from cameras for motion estimation, usually employing classical, long-standing computer vision algorithms for feature detection, tracking, and motion estimation. In the last decades, Deep Learning techniques have transformed the field of computer vision and led to great advances in many areas such as image classification and segmentation. While Visual SLAM could benefit from incorporating feature representations or semantic information computed by Deep Learning models, this is seldom done, because of computational efficiency constraints.

A foundational premise for many Visual SLAM methods is the *static world assumption*. By assuming a static world any motion observed in the image features can be attributed solely to the camera's ego motion. In real-world scenarios, this assumption rarely holds true. Urban environments are frequented by pedestrians and various types of vehicles and pose a significant challenge to Visual SLAM systems designed under that premise, often

Received:
Revised:
Accepted:
Published:

Citation: Lastname, F.; Lastname, F.; Lastname, F. Title. *Journal Not Specified* **2025**, *1*, 0. https://doi.org/

leading to degraded performance of the VSLAM system and causing trajectory drift.

Many approaches for the handling of dynamic objects in Visual SLAM systems have been proposed. Most of these incorporate object detection, *semantic* segmentation or *instance* segmentation networks to determine and segment out the dynamic objects in a scene. DynaSLAM [1] uses Mask R-CNN [2] to obtain pixel-wise semantic segmentations of the input images. Features that belong to a pre-defined subset of potentially moving object classes are eliminated to estimate a map of the static parts of the scene. DOT [3] first segments instances of *potentially* dynamic objects (except humans) and then tracks these objects by minimizing the photometric reprojection error to propagate the instance masks, thus eliminating the requirement to segment all the frames and enabling implementation in real-time. It further proposes a metric that can be used to determine whether an object is actually moving or not. DS-SLAM [4] integrates SegNet [5] for semantic segmentation with motion consistency to improve robustness in dynamic scenes. It detects moving points by computing the fundamental matrix with RANSAC and discarding matched points with a high distance to their corresponding epipolar lines. In [6], the authors propose a bidirectional refinement framework that integrates semantic segmentation with visual SLAM in a mutually reinforcing manner. Coarse pose estimations refine semantic outputs, which in turn enhance SLAM tracking and mapping. SaD-SLAM [7] extends ORB-SLAM2 [8] by leveraging semantic masks from MASK-RCNN and depth information to identify and distinguish between static and dynamic points. It further uses epipolar constraints across multiple frames to classify points as dynamic. CFP-SLAM [9] introduces a coarse-to-fine static probability mechanism based on object detection. By combining semantic, geometric, and motion constraints, the system assigns static probabilities to keypoints and map points, using them as weights in pose optimization. OVD-SLAM [10] introduces a more efficient method for identifying dynamic points by checking their motion consistency, avoiding the heavy computation of solving the fundamental matrix. It removes points with abnormal optical flow values using a chi-square test, and assigns optimization weights to map points based on their dynamic likelihood to improve pose estimation. NGD-SLAM [11] achieves real-time accuracy while running on a CPU by introducing a mask prediction mechanism that utilizes previous segmentation results to predict the mask of dynamic objects in the current frame. It processes RGB-D input and makes use of the depth information to generate masks for objects detected using a YOLO network. In [12], the authors propose a real-time semantic RGB-D SLAM framework that applies semantic segmentation exclusively to keyframes in order to reduce computational overhead. Unknown dynamic objects are identified through depth clustering and reprojection error analysis, allowing for the removal of both known and unknown dynamic entities.

This work follows a similar strategy by incorporating an accurate instance segmentation model, a mask propagation technique and a dynamic feature recognition algorithm. In contrast to DOT and other dynamic SLAM systems that are mostly based on ORB-SLAM2, it is integrated into an efficient, sparse, keypoint-based Visual Odometry (VO) frontend named Basalt [13]. Further, it is optimized for real-time usage and evaluated on the KITTI Odometry dataset [14] featuring automotive urban outdoor scenarios.

2. Materials and Methods

This chapter describes the dynamic instance removal (DIR) algorithm and its submodules in detail. In Figure 1 a diagram of the architecture of the proposed method is depicted. Initially, instance segmentation masks are generated using a Deep Learning model (Section 2.1). The mask prediction algorithm that is introduced to compensate for the high processing time of the instance segmentation network is detailed in section 2.2. These

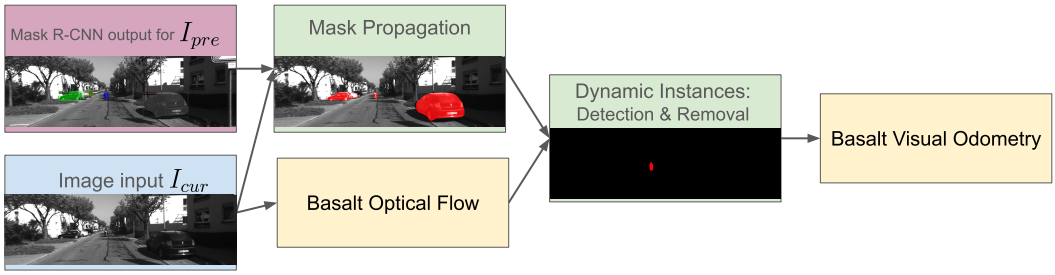
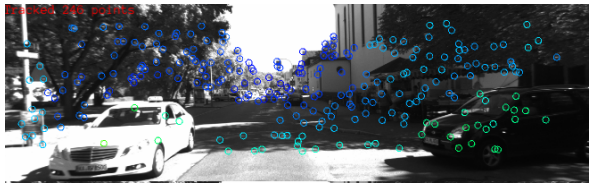
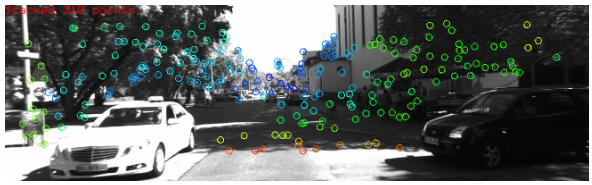


Figure 1. Architecture overview of the dyanmic instance removal method.

masks provide pixel-wise class labels for objects recognized in an image, but do not contain any information about whether an object is *in motion* or not. The algorithm distinguishing between keypoints corresponding to dynamic objects and keypoints belonging to static objects (e.g., parked cars) is defined in section 2.3. Figure 2 illustrates the features that are used for motion estimation by the Visual Odometry frontend without a moving object removal mechanism ((**a**)) and after application of the suggested approach ((**b**)).



(**a**)



(**b**)

Figure 2. Comparison between the original Visual Odometry frontend (Basalt) without dynamic object removal and the extended Basalt frontend with integrated dynamic object removal algorithm demonstrated using an image from the KITTI Odometry dataset.

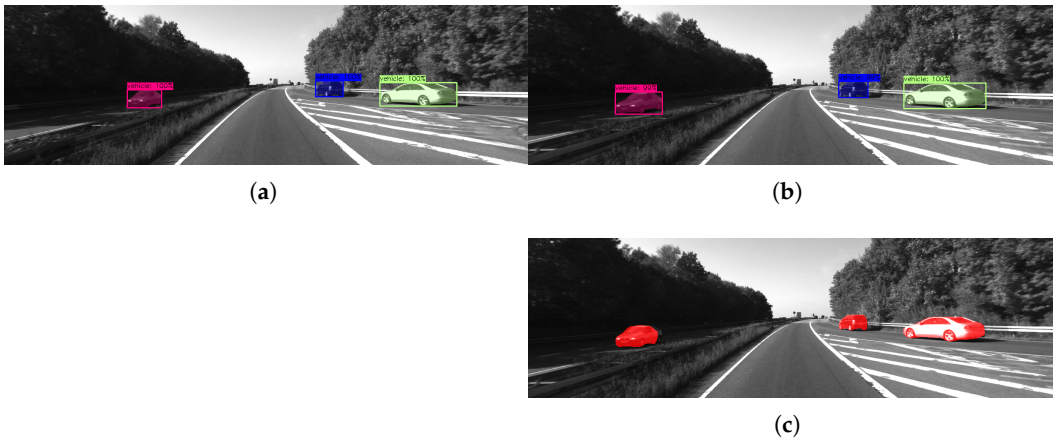
2.1. Instance Mask segmentation

Instance segmentation is performed on the left image of the stereo pair using a pre-trained Mask R-CNN model with a ResNet backbone architecture implemented in the MMDetection framework [15]. This model operates in two stages: first generating a region proposal and then performing classification and mask prediction within the proposed region. Compared to one-shot instance segmentation models like YoloAct [16], two-stage models like Mask R-CNN offer superior segmentation accuracy, but come with an increased computational load that can degrade the real-time performance of the VSLAM system. We optimize the execution of the model using TensorRT for GPU acceleration. We limit the Mask R-CNN output exclusively to potentially dynamic classes, i.e., vehicles and humans.

2.2. Mask propagation

As previously mentioned, due to the high computational cost associated with instance segmentation, it is not feasible to run this process at the full frame rate required by the SLAM system. To overcome this limitation, we introduce a mask propagation strategy that estimates the segmentation masks for intermediate frames based on previously generated

outputs. This approach allows the system to maintain high frame-rate processing while reducing computational overhead. Given the current image frame I_{cur} (whose instance segmentation masks are unknown) and the previous image frame I_{pre} with its corresponding masks, the algorithm first computes a sparse set of features using the Shi-Tomasi corner detector. Corner points that fall into a mask region are tracked to the current image frame with the incremental Lucas-Kanade Optical Flow method [17]. Given the set of matched feature correspondences $\{(\mathbf{x}_i, \mathbf{x}'_\mathbf{i})\}$ the parameters $\mathbf{p}$ of an affine 2D transformation $\mathbf{x}' = \mathbf{f}(\mathbf{x}; \mathbf{p})$ can be estimated, if there are at least three corresponding feature points available. Finally, the affine warp $\mathbf{f}$ is applied to transform an instance mask to its approximate location in I_{cur}. An exemplary usage for the mask propagation algorithm is illustrated in Figure 3.



(**a**) (**b**)

(**c**)

Figure 3. Upper row: Two consecutive images from sequence 01 of the KITTI Odometry dataset with instance mask overlays that were obtained by the Mask R-CNN instance segmentation network. The instance masks from ((**a**)) are fed to the mask propagation algorithm which predicts their location in the next image ((**c**)). Compared to the masks produced by the Deep Learning model ((**b**)) the propagated masks are slightly less accurate, as seen in the mask for the approaching car on the left.

2.3. Dynamic Instance Removal

To improve trajectory accuracy, keypoints from dynamic objects should be excluded from the SLAM processing pipeline. The dynamic status of the object instance is derived using a method that integrates epipolar geometry with instance segmentation masks. When calculating the static probability P_{static} of each potential dynamic object, we adopt the approach proposed in [4].

Given the current frame I_{cur} and the previous frame I_{pre}, we first extract FAST [18] corners $\mathbf{x}_{cur}$ from I_{cur}, and track their correspondences $\mathbf{x}_{pre}$ in I_{pre} using Lucas-Kanade Optical Flow, forming keypoint pairs $(\mathbf{x}^i_{\text{cur}}, \mathbf{x}^i_{\text{pre}})$.

Subsequently, the fundamental matrix F is estimated from the matched keypoints using the RANSAC algorithm. For each matched keypoint pair the epipolar error D_i is computed, defined as the geometric distance between a point and its corresponding epipolar line.

Since the pixel coordinates of keypoint pairs from optical flow tracking have two degrees of freedom, the chi-square distribution with $k = 2$ is used to statistically evaluate the epipolar error D_i of these matches.

Given the set of instance masks $\mathcal{S}$ obtained from the instance segmentation of I_{cur}, we evaluate the geometric consistency of keypoints within each mask. For each instance mask $S_j \in \mathcal{S}$, the epipolar errors D_i of all keypoint pairs located within the mask are sorted in ascending order. The average of the values at the $0.1M$, $0.2M$, and $0.3M$ positions in the sorted list (where M is the total number of keypoint pairs within S_j) is then computed

and assigned as the static probability P_{static} of the instance mask S_j. Instance masks with $P_{static} < 0.8$ are classified as dynamic, and all keypoints contained within these masks are subsequently discarded.

This strategy leverages both geometric motion constraints and instance segmentation to remove dynamic features efficiently, preserving only those likely to be static for accurate pose estimation.

3. Results

3.1. Metrics

The following metrics are used for the Visual Odometry trajectory accuracy evaluation: Root Mean Square (RMS) for Absolute Trajectory Error (ATE) computed on corresponding pose pairs of estimated and ground truth trajectory for global accuracy, and Relative Pose Error (RPE) computed from relative poses between two consecutive frames of estimated and ground truth trajectory to represent the local accuracy.

3.2. Dynamic Objects Ground Truth (GT) for KITTI Odometry

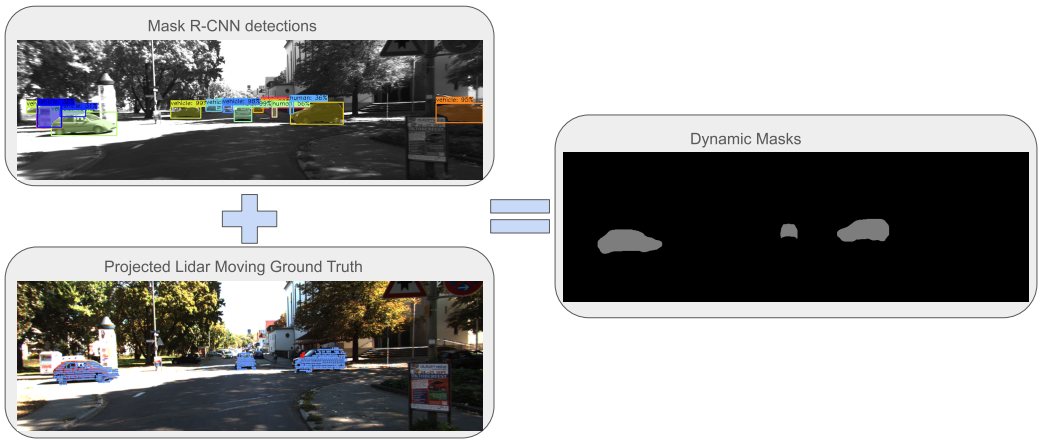


Figure 4. An example of dynamic masks generated using Mask R-CNN and Moving Object Segmentation challenge ground truth data for KITTI Odometry Seq 08, frame 2.

To analyze the KITTI Odometry dataset in terms of the presence of dynamic objects, as well as to evaluate the dynamic object detection approach, we generated pseudo-GT for dynamic objects based on Mask R-CNN instance segmentation output and Lidar ground truth data from the SemanticKITTI Moving Object Segmentation challenge [19] in the following way: To determine which of the detected object instances are truly dynamic, the labeled dynamic pointcloud is projected onto the camera frame with a predefined constant radius to form a sparse dynamic mask. The Mask R-CNN instance of a potentially dynamic semantic class is aligned with this sparse dynamic mask, and the intersection with the dynamic area of a compatible semantic class is measured and compared to an empirically adjusted threshold for each instance separately. The generated example is shown in Figure 4.

3.3. Visual SLAM accuracy

Table 1. RMSE ATE (in meters) on KITTI Odometry sequences: Basalt baseline (no masks), all masks from Mask R-CNN (Det), masks detected as dynamic (DIR), masks propagated from previous frame (Prop.), DIR on propagated masks (pDIR), pseudo-GT dynamic masks (dGT).

Seq	Basalt	Det		DIR		Prop.		pDIR		dGT	
	ATE	ATE	Δ,%	ATE	Δ,%	ATE	Δ,%	ATE	Δ,%	ATE	Δ,%
0	3,91	**3,64**	-6,81	3,72	-4,78	3,87	-1,04	3,87	-1,04	3,92	0,32
1	107,70	81,33	-24,48	**76,01**	-29,42	77,24	-28,28	80,27	-25,47	85,56	-20,55
2	10,59	**9,34**	-11,82	9,41	-11,19	9,61	-9,32	9,61	-9,31	9,72	-8,27
3	1,32	1,34	1,74	**1,30**	-1,58	1,33	0,75	1,33	0,75	1,32	0,01
4	1,30	1,31	0,77	1,31	0,76	**1,29**	-0,71	**1,29**	-0,71	1,32	1,79
5	2,94	**2,58**	-12,25	2,59	-11,76	2,77	-5,84	2,77	-5,84	2,89	-1,75
6	**2,53**	2,59	2,18	2,55	0,79	**2,53**	-0,29	**2,53**	-0,29	**2,53**	-0,26
7	1,40	1,36	-2,91	1,41	0,79	1,42	1,07	1,42	1,07	**1,31**	-6,58
8	**3,78**	3,99	5,74	4,14	9,60	3,98	5,43	3,98	5,42	3,87	2,40
9	**3,85**	3,95	2,62	3,97	3,25	3,93	2,09	3,93	2,09	3,88	0,93
10	1,12	0,98	-13,02	0,97	-13,35	**0,96**	-14,03	**0,96**	-14,03	0,97	-13,18

For the Visual Odometry solution with dynamic object removal, we first investigate the potential improvement from removing moving objects by evaluating all semantic masks from Mask R-CNN, only masks detected as dynamic applying the dynamic object method presented in Section 2.3, and as a best-case scenario, pseudo-GT masks described in Section 3.2. For better repeatability and fair comparison between different setups, we use the same pre-exported Mask R-CNN detections.

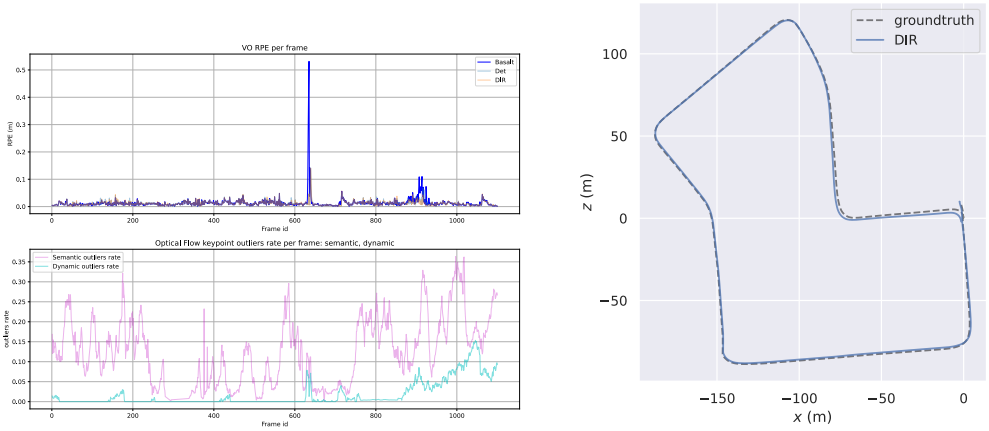
The evaluation results for the KITTI Odometry sequences are shown in Table 1. Removing keypoints from all masks produced by the instance segmentation network turns out to have a positive impact on the accuracy of the estimated trajectory, especially for the highly dynamic sequences like 01. However, a notable decrease in quality is observed for sequences with a high amount of static objects(e.g., parked cars, particularly present in sequence 08). The results on dynamic pseudo-GT masks show that keeping keypoints from those static objects, as well as filtering false detections, are critical for preventing the SLAM accuracy from decreasing. Figure 5(**a**) compares the amount of detected frame keypoints that were defined as semantic outliers (belonging to instances with semantic class of "human" or "vehicle") and the ones defined as dynamic outliers(whose instances were detected to be dynamic in pseudo-GT). The significant increase of the RPE visible in the plot is well aligned with the increase of dynamic outliers. This proves that in some cases, dynamic objects are the source of drift accumulation in a Visual Odometry trajectory.

3.4. Real-time setup

For online usage, all components of the proposed system were implemented in the ROS2 framework [20] with the aim of deployment in a real vehicle for real-time applications. The used Car PC is equipped with an Intel Core i9-13900E CPU and an NVIDIA RTX 4080 Super 16 GB GPU. For test purposes, the KITTI Odometry sequences were converted to ROS2 bag files.

The GPU-accelerated Mask R-CNN model was able to reach up to 48 fps performance on the given setup, exceeding the 10 fps data rate of the KITTI Odometry dataset and fulfilling the common real-time processing requirement of 30 fps.

The generation and processing of segmentation masks leads to a latency of 21 milliseconds in the processing pipeline, which the Visual Odometry solution has to wait. To reduce this latency we applied the mask propagation approach from Section 2.2 which reduced the latency to 14 milliseconds on average. For usage on less powerful computers



(**a**) (**b**)

Figure 5. ((a)) Upper row: RPE (y-axis) over frame id (x-axis) for pairs of successive frames. Bottom row: ratio of keypoints determined as *dynamic* to total number of frame keypoints (y-axis) over frame id (x-axis). ((b)) Estimated DIR trajectory aligned with the ground truth trajectory for KITTI sequence 07.

and embedded systems, application of the mask propagation algorithm becomes crucial, because instance segmentation networks run at less than 5 frames per second.

4. Discussion

Our analyses and experiments show that dynamic environments pose a problem to feature-based visual localization algorithms and degrade their overall accuracy. The system proposed as a solution to this problem demonstrates promising results by reducing the trajectory error (both ATE and RPE) of the position solution. Furthermore, we showed that the realization of a dynamic object removal algorithm that employs a powerful Deep Learning model in a real-time setup is feasible on a car embedded system. Nonetheless, the following limitations remain and will be addressed in future research efforts:

- A SLAM dataset incorporating dynamic object GT is essential for advancing research in moving object removal approaches for VSLAM. The proposed solution making use of Lidar data is limited to the Lidar sensor range, resulting in distant objects not being included in the pointcloud data and, although being used by SLAM, remaining unlabeled. The potential solution is to propagate the dynamic status for object observations over frames, from close ones to distant ones.
- The mask prediction algorithm struggles in certain scenarios, e.g., when applied on dynamic objects whose appearance is not cohesive over time, such as persons or bicyclists - here the stereo depth information could be employed to cluster these objects for mask refinement.
- The adopted instance segmentation network is pretrained on the large-scale image recognition dataset COCO and although generalizing well on automotive scenes, can be finetuned specifically for the desired operational environments for enhanced accuracy.
- Traditional methods based on epipolar constraints for filtering dynamic instances are outperformed by utilizing ground truth masks annotated solely for dynamic objects. This motivates training a moving object segmentation network, improving visual odometry through frontend integration.

The developed real-time capable algorithms for Visual SLAM with dynamic object removal will be integrated into the V-ROX system, which is the enhancement of the A-ROX

GNSS-INS positioning system, providing precise and robust localization in challenging environments and featuring environment detection using camera and LiDAR sensors.

Author Contributions: Conceptualization and methodology, all authors; software, L.S., O.H. and J.F.; validation, formal analysis, investigation, resources, data curation, L.S., O.H. and J.F.; writing—original draft preparation: L.S., O.H. and J.F.; writing—review and editing, L.S., O.H., J.F. and R.B.; visualization, L.S.; supervision, R.B.; project administration, L.S.; All authors have read and agreed to the published version of the manuscript.

Funding: The work is performed in the frame of the DREAM project (https://dream-project-eu.com/), funded by the EUSPA as part of the Fundamental Elements Programme (contract number: EUSPA/GRANT/03/2022).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The KITTI Vision Benchmark dataset used in this study is publicly available at https://www.cvlibs.net/datasets/kitti/. The Semantic KITTI dataset used in this study is available at https://www.semantic-kitti.org/index.html. The resources and derived data used in this study are available on request from the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ATE	Absolute Trajectory Error
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DIR	Dynamic Instance Removal
DOT	Dynamic Object Tracking
EU	European Union
FAST	Features from Accelerated Segment Test
GNSS	Global Navigation Satellite System
GPU	Graphics Processing Unit
GT	Ground Truth
RANSAC	RANdom SAmple Consensus
RMS	Root Mean Square
RMSE	Root Mean Square Error
ROS2	Robot Operating System 2
RPE	Relative Pose Error
SLAM	Simultaneous Localization and Mapping
VO	Visual Odometry
VSLAM	Visual Simultaneous Localization and Mapping
YOLO	You Only Look Once

References

1. Bescos, B.; Facil, J.M.; Civera, J.; Neira, J. DynaSLAM: Tracking, Mapping, and Inpainting in Dynamic Scenes. *IEEE Robotics and Automation Letters* **2018**, *3*, 4076–4083. https://doi.org/10.1109/lra.2018.2860039.
2. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN, 2018, [arXiv:cs.CV/1703.06870].
3. Ballester, I.; Fontan, A.; Civera, J.; Strobl, K.H.; Triebel, R. DOT: Dynamic Object Tracking for Visual SLAM, 2020, [arXiv:cs.CV/2010.00052].
4. Yu, C.; Liu, Z.; Liu, X.J.; Xie, F.; Yang, Y.; Wei, Q.; Fei, Q. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 1168–1174. https://doi.org/10.1109/IROS.2018.8593691.
5. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation, 2016, [arXiv:cs.CV/1511.00561].

6. Wang, K.; Lin, Y.; Wang, L.; Han, L.; Hua, M.; Wang, X.; Lian, S.; Huang, B. A Unified Framework for Mutual Improvement of SLAM and Semantic Segmentation. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 5224–5230. https://doi.org/10.1109/ICRA.2019.8793499.
7. Yuan, X.; Chen, S. SaD-SLAM: A Visual SLAM Based on Semantic and Depth Information. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 4930–4935. https://doi.org/10.1109/IROS45743.2020.9341180.
8. Mur-Artal, R.; Tardós, J.D. ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras. *IEEE Transactions on Robotics* **2017**, *33*, 1255–1262. https://doi.org/10.1109/TRO.2017.2705103.
9. Hu, X.; Zhang, Y.; Cao, Z.; Ma, R.; Wu, Y.; Deng, Z.; Sun, W. CFP-SLAM: A Real-time Visual SLAM Based on Coarse-to-Fine Probability in Dynamic Environments. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022, pp. 4399–4406. https://doi.org/10.1109/IROS47612.2022.9981826.
10. He, J.; Li, M.; Wang, Y.; Wang, H. OVD-SLAM: An Online Visual SLAM for Dynamic Environments. *IEEE Sensors Journal* **2023**, *23*, 13210–13219. https://doi.org/10.1109/JSEN.2023.3270534.
11. Zhang, Y.; Bujanca, M.; Luján, M. NGD-SLAM: Towards Real-Time Dynamic SLAM without GPU, 2024, [arXiv:cs.RO/2405.07392].
12. Ji, T.; Wang, C.; Xie, L. Towards Real-time Semantic RGB-D SLAM in Dynamic Environments. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 11175–11181. https://doi.org/10.1109/ICRA48506.2021.9561743.
13. Usenko, V.; Demmel, N.; Schubert, D.; Stueckler, J.; Cremers, D. Visual-Inertial Mapping with Non-Linear Factor Recovery. *IEEE Robotics and Automation Letters (RA-L) Int. Conference on Intelligent Robotics and Automation (ICRA)* **2020**, *5*, 422–429. https://doi.org/10.1109/LRA.2019.2961227.
14. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), 2012.
15. Chen, K.; Wang, J.; Pang, J.; Cao, Y.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Xu, J.; et al. MMDetection: Open MMLab Detection Toolbox and Benchmark. *arXiv preprint arXiv:1906.07155* **2019**.
16. Bolya, D.; Zhou, C.; Xiao, F.; Lee, Y.J. YOLACT: Real-time Instance Segmentation, 2019, [arXiv:cs.CV/1904.02689].
17. Lucas, B.D.; Kanade, T. An iterative image registration technique with an application to stereo vision. In Proceedings of the Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2, San Francisco, CA, USA, 1981; IJCAI'81, pp. 674–679.
18. Rosten, E.; Porter, R.; Drummond, T. Faster and Better: A Machine Learning Approach to Corner Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2010**, *32*, 105–119. https://doi.org/10.1109/TPAMI.2008.275.
19. Chen, X.; Li, S.; Mersch, B.; Wiesmann, L.; Gall, J.; Behley, J.; Stachniss, C. Moving Object Segmentation in 3D LiDAR Data: A Learning-based Approach Exploiting Sequential Data. *IEEE Robotics and Automation Letters(RA-L)* **2021**. https://doi.org/10.1109/LRA.2021.3093567.
20. Macenski, S.; Foote, T.; Gerkey, B.; Lalancette, C.; Woodall, W. Robot Operating System 2: Design, architecture, and uses in the wild. *Science Robotics* **2022**, *7*, eabm6074. https://doi.org/10.1126/scirobotics.abm6074.