Masked ORB-SLAM3: Dynamic Element Exclusion for Autonomous Driving Scenarios Using Mask R-CNN for Increased Localization Accuracy

Aditya Om*, Aman Kushwaha†, Kyle Liebler‡, Ping-Hua Lin§ and Zhuowen Shen¶
* *Electrical and Computer Engineering, University of Michigan, Ann Arbor, MI* adityaom@umich.edu
† *Electrical and Computer Engineering, University of Michigan, Ann Arbor, MI* amankush@umich.edu
‡ *Electrical and Computer Engineering, University of Michigan, Ann Arbor, MI* liebler@umich.edu
§ *Robotics, University of Michigan, Ann Arbor, MI* pinghual@umich.edu
¶ *Computer Science and Engineering, University of Michigan, Ann Arbor, MI* mickshen@umich.edu

***Abstract*— To observe and remove the impact of dynamic objects in ORB-SLAM3 we manually selected the most dynamic sequences from the KITTI dataset and removed those dynamic objects from the image frames. We observed that ORB-SLAM3 performs better if the dynamic content from the image frames is removed during tracking. We removed dynamic objects from the image frame by performing instance segmentation to create binary masks; and then passed these masks into the ORB-SLAM3 pipeline. From our results, we saw that the removal of dynamic objects causes considerably better performance as compared to the original ORB-SLAM3 implementation.**

***Index Terms*— ORB-SLAM3, KITTI dataset, instance segmentation**

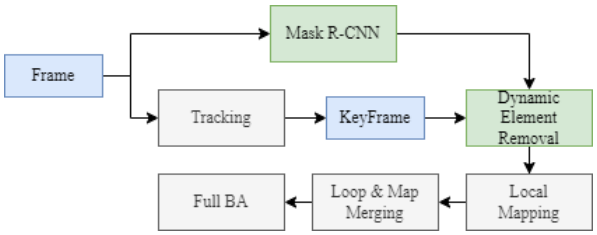


Fig. 1: Masked ORB-SLAM3 architecture

I. Introduction

SLAM (simultaneous localization and mapping) is a method to build a map and localize the object in that map at the same time. The two most popular types of SLAM are visual SLAM and LiDAR SLAM. In visual SLAM the camera image frames are used to estimate the pose of the camera. Visual SLAM can use simple cameras (wide-angle, fish-eye, and spherical cameras), compound eye cameras (stereo and multi-cameras), and RGB-D cameras (depth and ToF cameras). Visual SLAM has become quite popular in recent years.* †

A common practice in the field of visual SLAM systems is the assumption of scene rigidity. This assumption largely simplifies the problem. However, we believe that while working with visual SLAM systems it is very important to deal with dynamic objects present in the image frame. Removing dynamic objects could help to build more stable maps and decrease localization error, which would bring additional safety and efficiency to robotic systems. DynaSLAM [1] suggests a way to remove dynamic/moving objects for improving localization. DynaSLAM first detects dynamic objects, does the camera localization, and then reconstructs the occluded background of the current frame with static information from previous frames. After removing dynamic objects from the image frame, the pose of the camera is tracked using the static part of the image.

We detected objects (cars, trucks, etc.) in our data by using a pre-trained image segmentation model. The methods for this object detection are further explained in a previous work and in DynaSLAM [1].

Another previous work, ORB-SLAM3 [2], explains a SLAM pipeline in detail and its improvement over ORB-SLAM1 [3] and ORB-SLAM2 [4]. To summarize, short-term data association increases the drift/error. Long-term data association helps in a long-time tracking, remembering previously visited areas, resetting the drift, and correcting the graph using pose-graph optimization. These features in ORB-SLAM add robustness to the SLAM algorithm. The ORB-SLAM algorithm works only for the monocular images whereas ORB-SLAM3 works for monocular, stereo, and RGB-D cameras. Also, monocular SLAM suffers from scale drift. It is also worth noting that ORB-SLAM2 performs better than ORB-SLAM; and that ORB-SLAM3 performs better than ORB-SLAM only in some specific scenarios, but is similar to ORB-SLAM2 in various cases.

In the rest of the paper, we discuss related work in Section II, describe our system in Section III and IV, then present the results and its evaluation in Section V, limitations in section VI-A, and end with future work in Section VI-B.

II. Related Works

A related approach defines dynamic objects in the context of background inpainting as outliers. This way, they have applied robust constraints such as Bundle Adjustment [4], [5], [6] or RANSAC [7] to remove them. Gutiérrez-Gómez

*Our code repo: `https://gitlab.eecs.umich.edu/v_slam/orb-slam_dynamic`

†Presentation video: `https://www.youtube.com/watch?v=7ju1BXNozT4`

et al. [8] compared various robust functions that focused on the estimate-quality, while Kerl et al. [9] demonstrates robustness to the presence of tiny dynamic objects in the scene. While these solutions can effectively remove slightly dynamic (objects) feature points from a highly dynamic environment, such as in an autonomous driving scenario, they have a difficult time filtering highly dynamic feature points.

In most dynamic scenarios, if the extracted feature points belonged to a moving object, those points would violate geometric constraints [10]. Yuxiang Sun et al. [11] applied optical flow to match two frames to obtain a homograph matrix; aligning the background of the two frames, using the codebook method to segment dynamic objects.

With the advent of deep learning, convolutional neural networks (CNNs) have been applied to detect dynamic objects [12]. Object detection and semantic segmentation can show potential dynamic objects; however, their tracking will not extract feature points from these dynamic objects [1].

In another approach, object detection networks (ODNs) have been utilized for dynamic object detection. Xiao et al. [13] used a Single-Shot Multi-Box (SSD) object detector-based semantic SLAM framework to make the system suitable for a dynamic environment. Zhong et al. [14]'s robotic vision system integrates SLAM with an object detector to leverage its mutual benefits.

Furthermore, Zhang et al. [15] proposed a novel semantic SLAM framework to achieve robustness in DSs for RGB-D camera, which detected potentially dynamic objects by Mask R-CNN. Zhao et al. [16] proposed a workflow of accurate object-segmentation, wherein they mark potential dynamic-object areas on the basis of semantic information. Zhang et al. [17] introduced a CNN model that improved the accuracy of terrain-segmentation. It has been applied to autonomous robot navigation in wild environments. Ai et al. [18] proposed the dynamic deep learning (DDL)-SLAM, a robust RGB-D SLAM system for dynamic scenarios that adds dynamic object segmentation (DOS) and background inpainting. In DS-SLAM proposed by Yu et al. [19], the semantic segmentation network is combined with a moving consistency check method to obtain a reduction in the impact of dynamic objects. Thus, the accuracy of the obtained localization is enhanced for dynamic environments. Han and Xi [20] combine the visual SLAM and pyramid scene parsing network (PSPNet) in their PSPNet-SLAM system, which utilizes the semantic segmentation and optical flow to detect and eliminate dynamic points.

III. Approach

The system includes a modified version of ORB-SLAM3 and an additional Mask R-CNN component both configured to operate on monocular camera data. Data is ingested into the Mask R-CNN for instance segmentation as well as into ORB-SLAM3 for localization and mapping. The original ORB-SLAM3 architecture is displayed in Fig. 2, and the modified architecture in Fig. 1.

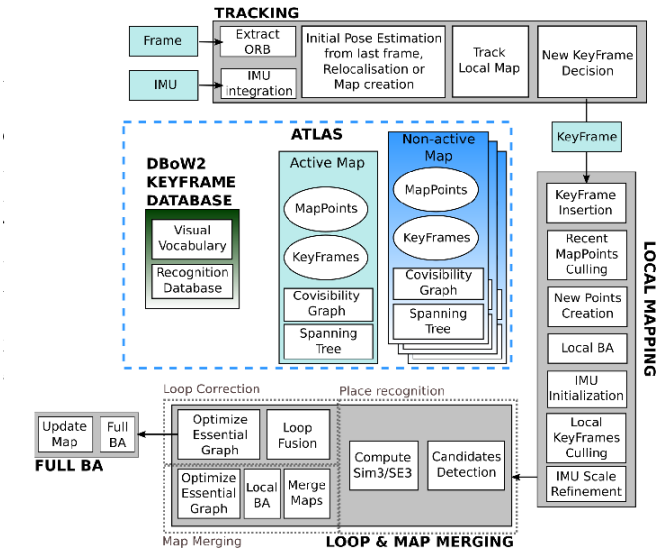


Fig. 2: ORB-SLAM3 architecture [2]

A. Dynamic Object Segmentation

A Mask R-CNN pretrained on the COCO dataset is used for the identification of dynamic objects in the scene. This model runs and performs instance segmentation on every image for a sequence of driving data. In addition, manually selected COCO object classes are categorized as dynamic; this includes cars, buses, and more. The segmentation masks of these specific objects are additively overlaid and normalized to form a singular mask. A mask, depicted in Figure 3, is saved for each frame in order to maintain synchrony with the original data.

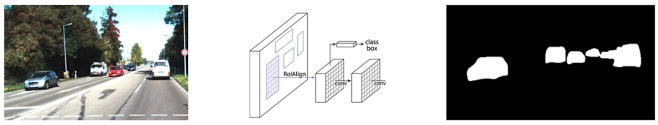


Fig. 3: Mask example

B. SLAM with Dynamic Object Removal

The general ORB-SLAM3 architecture remains the same and includes four main steps: tracking, local mapping, loop closure, and full bundle adjustment. The modifications made to this algorithm involve inserting a new step in between the tracking and local mapping phases. Tracking entails the localization of the camera for each frame by selecting significant points and frames in the data. In the original implementation, these points and frames are passed along to the local mapping phase where the points are added to the local map. With the addition of dynamic object removal, a new module intercepts these points at the output of the tracking module before they have a chance to be input to the local map. It instead removes the dynamic elements in the scene and only then passes the filtered output along to the local mapping step.

To perform dynamic object removal we first allow the tracking module to produce its key tracking points. These

features exist as two dimensional real numbered coordinates. For wider coverage and indexing purposes, we interpolate these features to the four surrounding integer coordinates. Then, by utilizing our binary masks we check for features that are overlapping with dynamic elements. These features are discarded, and the rest are preserved. The SLAM implementation then continues as usual; this happens for every frame.

Algorithm 1 Dynamic Object Removal

Require: Input array $A[] \in \mathbb{R}^{n \times 2}$
Require: Input array $M[] \in \mathbb{Z}^{dim(Frame)}$
1: **for** all elements a in A[] **do**
2: x $\leftarrow \lfloor A[0] \rfloor$
3: y $\leftarrow \lfloor A[1] \rfloor$
4: **if** M[y][x] > 0 **or** M[y+1][x] > 0 **or** M[y][x+1] > 0 **or** M[y+1][x+1] > 0 **then**
5: **Delete** a
6: **end if**
7: **end for**

IV. Experiment Setup

To test our approach we perform two experiments. The first experiment compares our masked ORB-SLAM3 using ground truth masks with the original ORB-SLAM3 implementation. To acquire ground truth masks we use the MOTS dataset [21] which is a subset of the overall KITTI dataset [22]. Unfortunately, this dataset did not contain the ground truth's trajectory; thus, the raw GPS data was selected for comparison. In the second experiment, we use KITTI data from the odometry subset [23] which does contain ground truth trajectory but does not contain ground truth masks. In this scenario we utilize our Mask R-CNN to compare our masked implementation to the original ORB-SLAM3 implementation.

For reference, Figure 4 depicts the orientation of the sensors to better aid in interpreting the results. The first experiment is with respect to the GPS sensors frame (depicted in green), while the second experiment is with respect to the left camera (depicted in red).

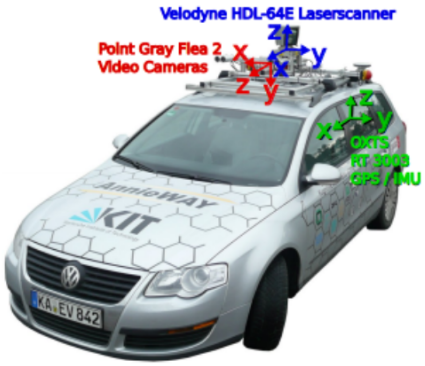


Fig. 4: Involved coordinate frames [24]

V. Results

A. Results for the First Experiment

In the original ORB-SLAM, the ORB features are generated on the dynamic objects, as shown in Fig. 5a. After the masks (in in Fig. 5b) for dynamic objects are applied, the ORB features are removed as in Fig. 5c.

We compared the trajectories from the naive ORB-SLAM and our masked ORB-SLAM against the one from GPS data, as shown in Fig. 6. The trajectories from the two ORB-SLAMs are quite similar. However, we cannot determine whether masked ORB-SLAM performs better or not since GPS trajectory cannot be taken as ground truth trajectory.

B. Results for the Second Experiment

The masks for dynamic objects in this second experiment are generated by the Mask R-CNN. A sample mask against its original frame is shown in Fig. 3.

We compared the poses from the naive ORB-SLAM and our masked ORB-SLAM against the ground truth poses, as shown in Fig. 7. For the z axis in Fig. 7a and the roll, pitch, yaw angles in Fig. 7b, the naive and masked ORB-SLAM have very similar outcomes. For the x and y axes in Fig. 7a, our masked ORB-SLAM deviates less compared to the naive one. This observation is later further supported by the statistics calculated in Fig. 8. In Fig. 8a and Table I, by calculating the Absolute Pose Error (APE) with respect to the ground truth poses frame by frame, we can see that clearly the root mean square error (RMSE) decreased significantly. Also, the mean, median and other values have also improved. In the APE distribution, Fig. 8b, our masked approach also has less error and performs better. [‡] [§]

For the APE density in Fig. 9, our masked ORB-SLAM has smaller mean and standard deviation, meaning that the errors in most frames have smaller magnitudes. Thus, our approach performs better.

TABLE I: APE Statistics: Masked vs. Naive ORB-SLAM

	Masked	Original
RMSE	**1.087**	2.123
Mean	**0.879**	1.807
Median	**0.673**	1.620
Std	**0.641**	1.159
Min	0.107	**0.066**
Max	**2.280**	5.640
SSE	**320.657**	1222.049

VI. Discussion

A. Limitations

Although the hypothesis that removing dynamic objects from scenes increases localization accuracy seems to hold true, there are some implementation specific limitations to this idea. For example, in the DynaSLAM [1] paper and in

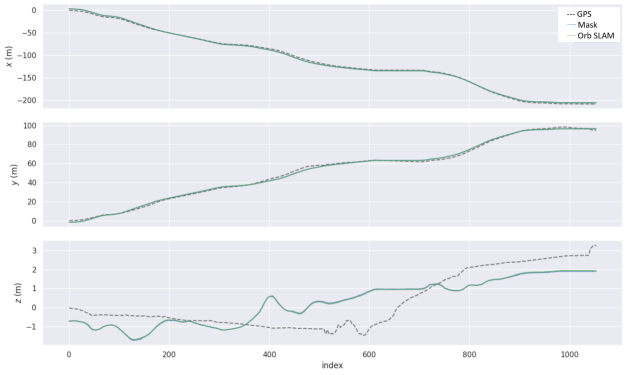
[‡]Our code repo: `https://gitlab.eecs.umich.edu/v_slam/orb-slam_dynamic`
[§]Presentation video: `https://www.youtube.com/watch?v=7ju1BXNozT4`

(a) Naive ORB-SLAM

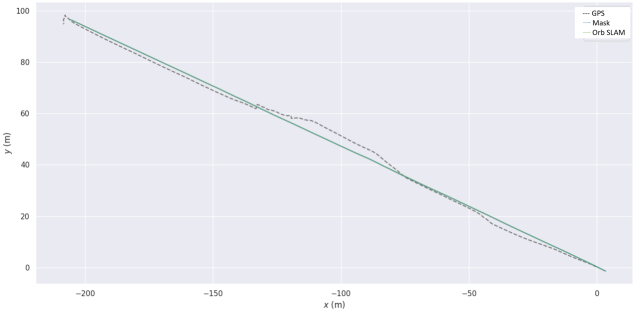
(b) Ground truth mask

(c) Our masked ORB-SLAM

Fig. 5: ORB feature extraction: masked vs. original ORB-SLAM3

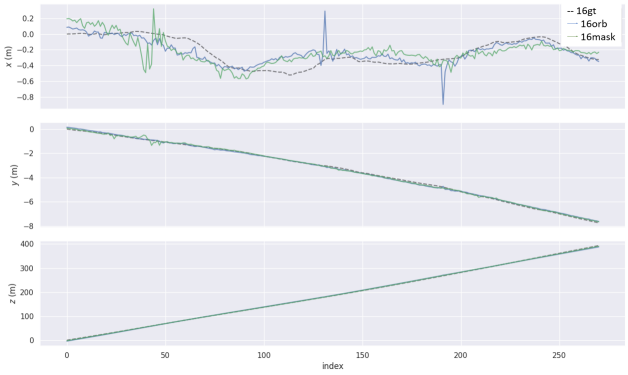


(a) x, y, and z location with respect to frame number

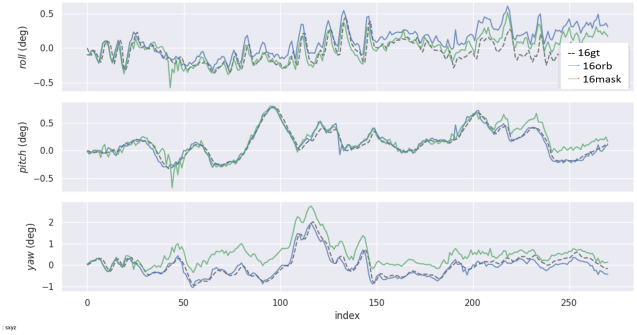


(b) Trajectories in x-y plane

Fig. 6: Location: masked vs. original ORB-SLAM vs. GPS



(a) x, y, and z location with respect to frame number

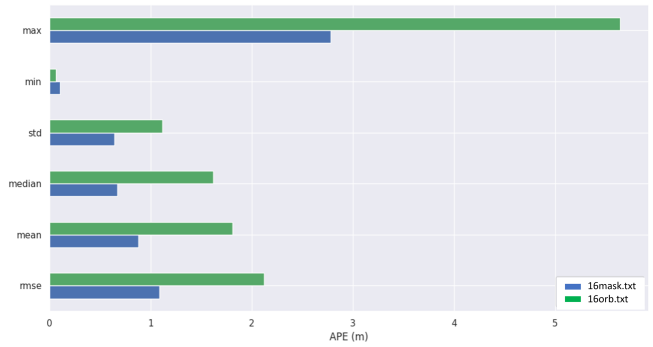


(b) Roll, pitch, and yaw with respect to frame number

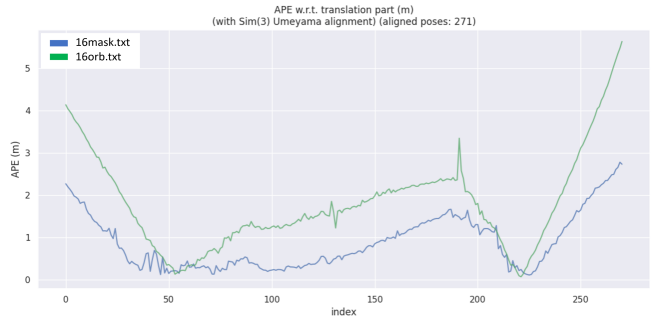
Fig. 7: Location: masked vs. original ORB-SLAM vs. ground truth

this paper, we have both neglected the possibility for dynamic object classes being in static states for the duration of the video sequence. If a car is parked along the side of the road, or a person is not moving, then they are static in the environment and could provide useful points for tracking. Further machine learning research may be able to provide solutions to this by exploiting the result of intrinsic camera qualities such as blurriness and more.

Another limitation of our approach that should be addressed is the speed of the model. In the DynaSLAM [1] model, the authors ran their implementation live and either used a lighter weight Mask R-CNN or faster computing resources. On our hardware we would not have been able to create our masks fast enough to run in a live environment, though we did not focus on this as an objective.



(a) APE statistics



(b) APE distribution

Fig. 8: Absolute Pose Error (APE) with respect to the ground truth: masked vs. original ORB-SLAM

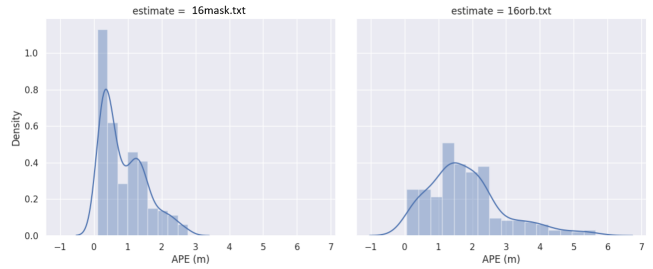


Fig. 9: Error density distribution: masked (left) vs. original (right) ORB-SLAM

B. *Future Works*

As for the future-works, there are couple portions we think we could further investigate. First, our approach only involves manually choosing object classes in the COCO dataset, i.e., car, people, bicycle, and utilizing those classes as potential dynamic objects when discarding ORB features. We believe applying an optical flow method with our mask approach could be more precise in selecting dynamic scenes and features.

Second, after proving our mask approach can come up with higher accuracy in localization, we would love to bring this method into real-time, just as DynaSLAM [1] does. Furthermore, to improve computation efficiency, Facebook AI's DETR (End-to-End Object Detection with Transformers) [25] can also perform object detection; a possible replacement for the Mask-RCNN.

Last but not least, if a new dataset with ground truth for both trajectory and semantic labels becomes available, we could further validate our results. We could also attempt to apply this methodology in LiDAR-SLAM approaches as well.

VII. Conclusion

In an autonomous driving scenario, we propose utilizing this method of removing dynamic elements from scenes in the context of solving the SLAM problem. Our results show that through using a Mask R-CNN and modifying ORB-SLAM3 we were able to increase localization accuracy over the original ORB-SLAM3 implementation.

Acknowledgment

We express our sincere thanks to our EECS 568 instructor Prof. Maani Ghaffari Jadidi for his guidance, as well as the GSIs Tzu-Yuan (Justin) Lin and Jingyu (JY) Song for all the support they provided this semester.

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. 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] R. Mur-Artal and J. D. Tardós, "Orb-slam: tracking and mapping recognizable features," in *Workshop on Multi View Geometry in Robotics (MVIGRO)-RSS*, vol. 2014, 2014, p. 2.

[4] ——, "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.

[5] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.

[6] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.

[7] F. Chhaya, D. Reddy, S. Upadhyay, V. Chari, M. Z. Zia, and K. M. Krishna, "Monocular reconstruction of vehicles: Combining slam with shape priors," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5758–5765.

[8] D. Gutiérrez-Gómez, W. Mayol-Cuevas, and J. J. Guerrero, "Inverse depth for accurate photometric and geometric error minimisation in rgb-d dense visual odometry," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 83–89.

[9] 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.

[10] A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 4306–4312.

[11] Y. Sun, M. Liu, and M. Q.-H. Meng, "Motion removal for reliable rgb-d slam in dynamic environments," *Robotics and Autonomous Systems*, vol. 108, pp. 115–128, 2018.

[12] J. Ji, S. Li, J. Xiong, P. Chen, and Q. Miao, "Semantic image segmentation with propagating deep aggregation," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 12, pp. 9732–9742, 2020.

[13] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou, "Dynamic-slam: Semantic monocular visual localization and mapping based on deep learning in dynamic environment," *Robotics and Autonomous Systems*, vol. 117, pp. 1–16, 2019.

[14] F. Zhong, S. Wang, Z. Zhang, and Y. Wang, "Detect-slam: Making object detection and slam mutually beneficial," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 1001–1010.

[15] Z. Zhang, J. Zhang, and Q. Tang, "Mask r-cnn based semantic rgb-d slam for dynamic scenes," in *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2019, pp. 1151–1156.

[16] L. Zhao, Z. Liu, J. Chen, W. Cai, W. Wang, and L. Zeng, "A compatible framework for rgb-d slam in dynamic scenes," *IEEE Access*, vol. 7, pp. 75 604–75 614, 2019.

[17] W. Zhang, Q. Chen, W. Zhang, and X. He, "Long-range terrain perception using convolutional neural networks," *Neurocomputing*, vol. 275, pp. 781–787, 2018.

[18] Y. Ai, T. Rui, M. Lu, L. Fu, S. Liu, and S. Wang, "Ddl-slam: A robust rgb-d slam in dynamic environments combined with deep learning," *IEEE Access*, vol. 8, pp. 162 335–162 342, 2020.

[19] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1168–1174.

[20] S. Han and Z. Xi, "Dynamic scene semantics slam based on semantic segmentation," *IEEE Access*, vol. 8, pp. 43 563–43 570, 2020.

[21] P. Voigtlaender, M. Krause, A. Ošep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "MOTS: Multi-object tracking and segmentation," in *CVPR*, 2019.

[22] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[23] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[24] V. Sharma, "Kitti coordinate transformations," Jul 2021. [Online]. Available: https://towardsdatascience.com/kitti-coordinate-transformations-125094cd42fb

[25] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.