ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.33, No-1-2, 2025

Occlusion SLAM: Improving Visual SLAM by Leveraging Occluded Points

Sujan Dhali
Indian Institute of Technology Kanpur
Kalyanpur, Kanpur 208016, Kanpur, Uttar Pradesh, India
sujankd20@iitk.ac.in

Bhaskar Dasgupta
Indian Institute of Technology Kanpur
Kalyanpur, Kanpur 208016, Kanpur, Uttar Pradesh, India
dasgupta@iitk.ac.in

ABSTRACT

This paper presents a visual SLAM (simultaneous localization and mapping) system, called Occlusion SLAM, that uses occluded points—map points that are sometimes visible and sometimes blocked by objects in the foreground—and demonstrates a noticeable improvement in camera motion estimation. Occlusion SLAM is built upon ORB SLAM2, with an additional segmentation module based on YOLOv8 and the proposed occluded point detection module. The segmentation module creates a binary mask image of the predefined dynamic objects, which the occlusion detection module uses to identify occluded points, as they are intermittently revealed by the motion of an object.

Occlusion SLAM was evaluated on highly dynamic datasets, such as the "walking" and "crowd" sequences from the TUM and Bonn datasets, respectively. Results indicate that the occluded point detection module, combined with segmentation, achieves significant improvements in absolute trajectory estimation with minimal computational requirements compared to ORB SLAM2. The method showed its full potential when at least one-fifth of the map points in a keyframe were identified as occluded points. The experiments also show competitive performance against other well known methods, such as DS SLAM, Dyna SLAM, RDS SLAM, Crowd SLAM and SG SLAM.

Keywords

Simultaneous localization and mapping (SLAM), Pose estimation, Projection transformation, Occlusion, Semantic Segmentation, Bundle adjustment.

1 INTRODUCTION

Simultaneous localization and mapping is a problem where a robot simultaneously localizes itself and creates a surrounding map of the environment using a single or multiple sensors. Visual SLAM is a subdomain of this problem, which uses visual sensors such as cameras and camera-like sensors. Methods such as PTAM SLAM [KM07], the ORB SLAM family [MAMT15, MAT17, CER+21], and RGB-D SLAM [EHE+12] are some well-known approaches in visual SLAM systems that, throughout the years, have gained much popularity. Though these algorithms work very well in static environments, they fail when there is a small or significant dynamic presence in the environment.

To mitigate the effect of dynamic objects while estimating the trajectory, the literature is broadly categorized into two parts. One is a deep learning based method that uses deep neural network models to detect predefined dynamic objects [ZWZ+18, LLC+20, YM21, SGM21]. The second is a geometric approach that detects outliers by studying the motion of pixels [TLD+13, KKS09, LW10]. There are other popular methods that use both of the above approaches such as DS SLAM, DYNA SLAM, YOLOSLAM, SEG SLAM, etc.

The DS SLAM [YLL+18] and SDF SLAM [CM20] used SegNet [BKC17] to detect predefined dynamic objects and applied the epipolar constraint to detect dynamic features missed by segmentation. Dyna SLAM [BFCN18] applied MASK R-CNN [HGDG17] to perform pixel-wise segmentation and further computed the difference between actual and projected depth to identify anomalous pixels. YOLOSLAM [WGG+22] used YOLOv4 [BWL20] for segmentation and a depth-based RANSAC to eliminate outliers. MISD SLAM [YWC+22] used YOLACT++ [BZXL22] and calculated the reprojection

http://www.doi.org/10.24132/JWSCG.2025-8

error to detect the other dynamic features. In the work by Jin *et al* [JJY+23], they used SparseInt [CWC+22] to detect dynamic objects and used epipolar constraint as a geometric model. Crowd SLAM [SGM21] used a segmentation model YOLOv3 [RF18] and was specifically trained on human datasets such as Crowdhuman dataset [SZL+18] so that it could efficiently detect crowds of people. SG SLAM [CSZZ23] used NCNN framework based SSD model for segmentation and further added epipolar constraint to detect the dynamic features. DYGS SLAM [ZZC+25] used prompt based segmentation model called and EfficientSAM [YX23] and used depth projection error for detecting the dynamic points. Similarly DN SLAM [RZZH24] used SAM [KMR+23], which is a prompt based segmentation, for detecting dynamic object.

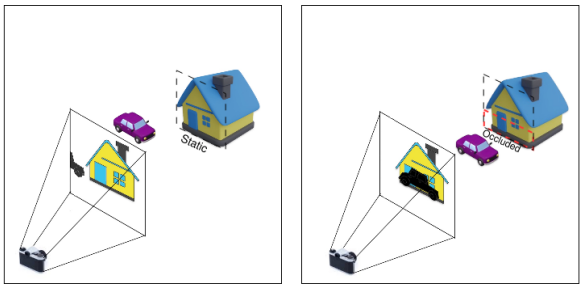


Figure 1: Illustration of the occlusion property. The left image shows a static scene with all objects visible, while the right image demonstrates occlusion, where a moving object (car) partially obstructs the house, altering the perceived scene from the viewpoint of the camera.

Most of the work discussed so far as well as other literature have used segmentation models to detect dynamic objects, and features on the detected objects have been removed. Next they apply a geometric approach to detect dynamic features that are missed through the segmentation. Now, the question arises: can segmentation alone provide deeper insights into the presumably static features?
In this work, we propose an approach, where the main goal is to assign a higher quality to certain selected points using the results from the segmentation. One such type of points are occluded points. For example, as shown in Figure 1, when a car (a dynamic object) passes in front of a house (a static object), some portions of the house are occluded by the car. This indicates that the occluded region remains stationary, unaffected by the presence of the dynamic object. In this article, we show that without using motion consistency module, successful inclusion of such occluded portions alone can help enhance camera motion estimation without significantly affecting computation time.

2 METHODOLOGY

2.1 System Description

Occlusion SLAM is built upon the framework of ORB SLAM2. Figure 2 shows the modified framework of the proposed system. With the structure of ORB SLAM2, two additional modules are introduced, namely the segmentation module and the occlusion point detection module. First the image from a video sequence is fed into both feature detection module and segmentation module. The segmentation module, which uses the YOLOv8 model, creates a mask the input image for predefined dynamic objects such as cars and people.

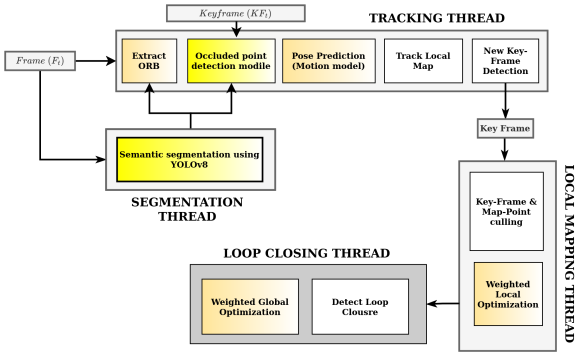


Figure 2: Proposed framework of Occlusion SLAM. Yellow modules represent newly introduced components, orange modules indicate modified components and white modules remain unchanged from ORB-SLAM2.

Then, the mask, along with the current image, is used to extract ORB features from the current image. The segmentation mask ensures that feature points, located on the detected dynamic objects are removed. The remaining features are further used to create a local map for each keyframe. This local map and the segmentation mask are used in the occlusion point detection module to detect the occluded points. After detecting occluded map points, those points are assigned higher weight compared to other map points. This weight enriches the information of those points, ensuring their grater importance in any bundle adjustment processes.

2.2 Semantic Segmentation

For detecting dynamic objects, the segmentation module uses the YOLOv8 [JCQ23] model. With its shorter inference time and better handling of motion blur, YOLOv8 is a good choice for real-world applications. YOLOv8 is trained on the MS COCO dataset [LMB+14], which includes 80 object classes. This segmentation module detects predefined dynamic objects and creates a mask image for them, which is then merged into a single mask. To ensure that feature points are not detected at the edges of objects, an additional morphological operation is performed

to expand the boundary of the detected objects. The features outside the segmented area are further used to create a local map. Additionally, the created map points are labeled as “STATIC".

2.3 Occlusion point detected module

To check whether a map point of a keyframe, which was labeled as "STATIC", is occluded or not, a map point is first projected onto the current frame. If its corresponding feature in the current frame is not visible or, more precisely, blocked by a segmented mask and later evident in future frames, then it is considered an occluded point. Figure 3 shows the detection of occluded point.

Algorithm 1: Pseudocode for detecting occluded points.

Input : Current Frame (F_t),Key Frame (KF_t)
$Img = F_t \rightarrow getImage()$
$segImg = F_t \rightarrow getSegmentation()$
$k = 0$
while $k < (KF_t \rightarrow Mappoints.size())$ **do**
  $pMP = KF_t \rightarrow Mappoints[k]$
  if pMP is valid **then**
    $State = pMP \rightarrow state()$
    if $State ==$ “$OCCLUDED$” **then**
      $k = k+1$
      continue
    end if
    $X^w = pMP \rightarrow getWorldPose()$
    $x = calculateProjection(X^w)$
    $Ib = InBound(x, Img)$
    $Is = InSegmentation(x, segImg)$
    if NOT Ib **then**
      $k = k+1$
      continue
    end if
    if $State ==$ “$STATIC$” and (Is) **then**
      $pMP \rightarrow state() =$ “$BLOCKED$”
    else if $State ==$ “$BLOCKED$” and NOT (Is) **then**
      $pMP \rightarrow weight() = w_1$
      $pMP \rightarrow state() =$ “$OCCLUDED$”
    end if
  end if
  $k = k+1$
end while

Suppose a map point associated with a keyframe KF_t is P_t^i. To check whether the map point belongs to the segmented region or not, first the map point ${P_t}^i$, with position X^w in the world coordinate, is projected onto the current frame. The required projection transformation is calculated by estimating approximate position T_w^c of the current frame. In our method, we use Iterative PnP algorithm [MUS16], which uses matched map points between keyframe KF_t and current frame F_t, and estimate the position of the current frame. Equation 1 shows the projection transformation of a map point onto the current frame, where K is the calibration matrix of current frame and x_i is the projected keypoint on the current frame represented in homogeneous coordinate system.

$$x_i = K * (T_w^c * X_i^w)_{1:3} \ . \tag{1}$$

With a good approximation of the current frame position, all the map points of the keyframe are projected onto the current frame. Initially, the status of the point P_t^i is changed to “BLOCKED", if its projection falls within any segmented mask. This means the corresponding feature of the map point in the current frame, which was suppose to be visible, is blocked by the segmented objects. Later, in any future frame when that “BLOCKED" map point is projected onto the current frame and lie within image boundary but outside the mask region, then it is declared “OCCLUDED". When a map point of a keyframe is declared as a occluded point, it will be considered as an occluded point throughout the runtime of the Occlusion SLAM. Algorithm 1 describes the method of detecting the occluded points. The algorithm takes the Current Frame object and Keyframe object as input and modifies the weight of the map points.

2.4 Weighted bundle adjustment

When a map point is visible in a frame at a particular key point, the difference between the key point's position and the actual projection produces re-projection error. Bundle adjustment is a process of minimizing the collective re-projection errors to determine the best position of map points as well as camera frames. To represent which re-projection error is more important than others, our method introduces weights of each observation. When a higher weight, w of a map point with the position of X_j^w is multiplied with information matrix of all re-projection errors associated with that point, it gives more importance to that point in the bundle adjustment process. Equation 2 shows the total re-projection error.

$$E = \sum_j \sum_i \left[e_{ij}^T \cdot (w * \Omega_{i,j}) \cdot e_{ij} \right] \ , \tag{2}$$

where

$$e_{ij} = x_{ij} - \pi \left(T_i^w, X_j^w \right) \ .$$

Here, e_{ij} represents re-projection error produced by point P_j when projected onto the frame F_i with the position T_i^w, and x_{ij} is the key point on the frame F_i, where the point P_j is observed; $\Omega_{i,j}$ is the information matrix of the observation. In our method, initial weight of a map point is $w_0 = 1.0$. After it is detected as an occluded point, a higher weight is assigned to it; in our case it is $w_1 = 3.0$ a

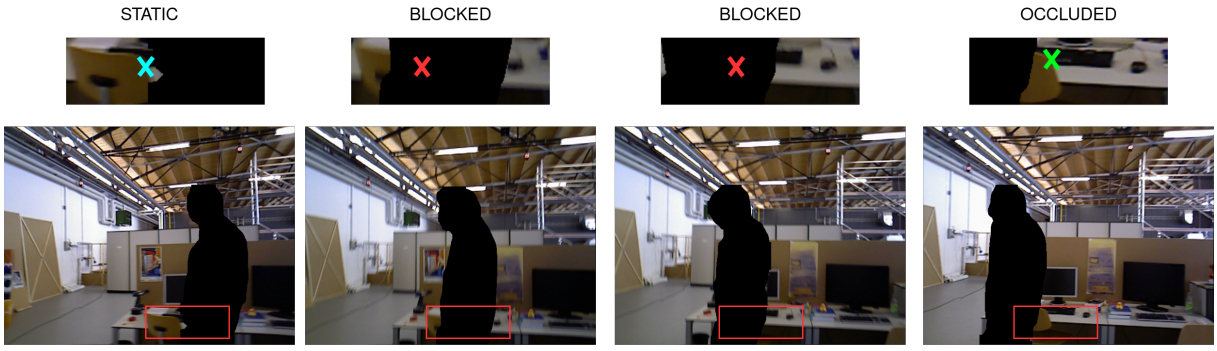


Figure 3: Illustration of the working principle of occluded point detection module. Initially, the key frame creates map points from a static background. As a dynamic object (person) moves, a certain map point, which is static (cyan) become blocked (red) due to segmentation. When the (dynamic) object moves further, previously blocked points may reappear, marking them as occluded (green).

3 RESULTS AND DISCUSSION

Table 1: Comparison of RMSE of absolute trajectory for ORB SLAM2 and Occlusion SLAM on TUM dataset and Bonn dataset.

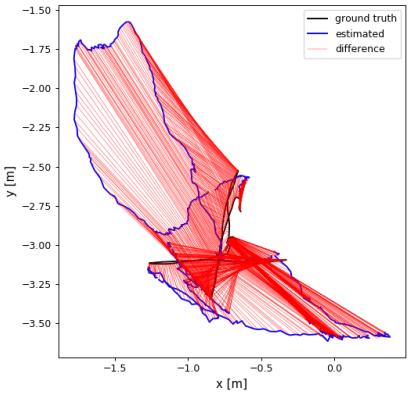
Dataset	ORB SLAM2	Occlusion SLAM	Imp.(%)
walking_xyz	0.8665	**0.0152**	**98.16**
walking_static	0.4702	**0.0065**	**98.60**
walking_half	0.4310	**0.0286**	**93.35**
Crowd	0.8780	**0.0177**	**97.98**
Crowd2	1.4830	**0.0459**	**96.90**
Crowd3	1.1830	**0.0332**	**97.18**

We evaluate Occlusion SLAM with two types of datasets: the TUM public dataset [SEE+12] and the BONN RGB-D dataset [PBL+19]. Both datasets have multiple sequences; in our experiment, we mostly focused on the datasets with highly dynamic presence, such as “walking" in the case of TUM and “crowd" in the BONN datasets, respectively. Initially, we differentiate our method with the original system, which is ORB SLAM2 [MAT17]. Later we did a detailed comparison of our method with segmentation (without detecting occluded points). Finally, we compare our method with several other popular algorithms which was discussed earlier in section 1. We have also analyzed computational performance of our method with and without the occluded point detection module. The hardware used to evaluate all the methods includes an AMD Ryzen 7 7700X CPU, an RTX 4080 GPU, and 32 GB RAM.

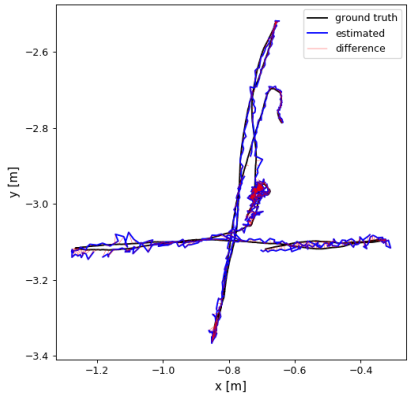
The RMSE of the absolute trajectory of both ORB SLAM2 and Occlusion SLAM is shown in Table 1. The first three rows show the sequences of TUM dataset and the latter half shows the comparison for the BONN dataset. As seen in the table, our method not only surpasses ORB SLAM2 in every dataset; in some cases, our method improved accuracy by more than 98%. Figure 4a & 4b show the estimated trajectory and its error in the “walking_xyz" dataset for both ORB SLAM2 and Occlusion SLAM.

Table 2 shows the improvement achieved by using occluded points over segmentation. In the case of the TUM datasets Occlusion SLAM did not show any significant enhancement in results; however, for BONN dataset, results demonstrate considerable amount of improvement. It has been observed that even though TUM datasets contains highly dynamic elements such as walking people, the dynamic subjects in that dataset do not cover the substantial portion of the environment. This results in a lower number of occluded points being generated. On the other hand, the BONN dataset (particularly in the crowd sequences), the number of people is significantly higher, leading to a greater number of points occluded in some keyframes. This also explains the observed improvement in BONN dataset.

It has been also discovered that pose estimation can be significantly affected, if the features are extracted from moving objects. These objects may include dynamic objects missed by the segmentation module or moving objects that are not predefined as dynamic. However, if an estimation incorporates a sufficient number of occluded points, it can mitigate the impact of these falsely classified static points. This effect is particularly evident in the “crowd2" sequence. As a quantitative measurement, we have observed that if the local map of a keyframe contains more than 20% occluded points, the effect is considerably noticeable. In this case method showed 63% reduction of RMSE



(a)



(b)

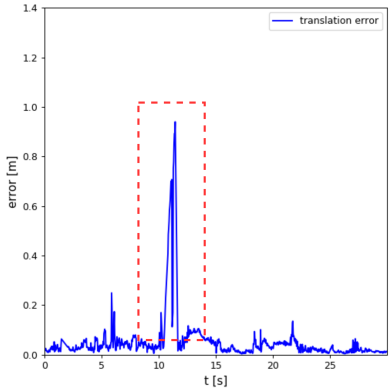
Figure 4: Comparison of estimated trajectories on the walking_xyz dataset. (a) ORB SLAM2 shows significant deviation from the ground truth. (b) Occlusion SLAM demonstrates a closer alignment with the ground truth.

value compare to segmentation. Figure 5a and 5b illustrate the translation error observed in segmentation as well as Occlusion SLAM for the “crowd2" sequence.

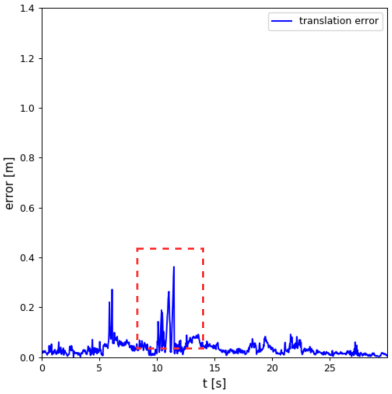
Table 3 presents a comparison between DS SLAM [YLL+18], DYNA SLAM [BFCN18], RDS SLAM [YM21], CROWD SLAM [SGM21] and SG SLAM [CSZZ23].This comparison is made to check the effectiveness of occluded point detection against various geometric approaches as those. The results indicate that, despite our method not yet utilizing any geometric approach, on which DS SLAM, DYNA SLAM and SG SLAM rely, to filter out the dynamic points, it achieves comparatively good results. Our method outperforms DS SLAM and RDS SLAM in every sequence. When compared to the DYNA SLAM, our method is competitive across most of the datasets except for the “crowd2" sequence. Regarding CROWD SLAM which used a carefully trained segmentation model to remove the people from the environment, our method produces comparable results. It is important to note that the objective of this comparison is not

Table 2: Comparison of RMSE of absolute trajectory for segmentation only and Occlusion SLAM on TUM dataset and Bonn dataset.

Dataset	Seg (w/o occlusion)	Seg (w/ occlusion)
walking_xyz	0.0153	0.0152
walking_static	0.0067	0.0065
walking_half	0.0276	0.0286
Crowd	0.0213	**0.0177**
Crowd2	0.1243	**0.0459**
Crowd3	0.0405	**0.0332**



(a)



(b)

Figure 5: Comparison between translation error generated in crowd2 dataset using (a) only segmentation (without detecting occluded points) to estimate trajectory and (b) using occluded point detection along with the segmentation for estimating trajectory.

Table 3: Comparison of Root Mean Square of ATE for DS SLAM, DYNA SLAM, RDS SLAM, CROWD SLAM, SG SLAM Occlusion SLAM.

Dataset	DS SLAM	DYNA SLAM	RDS SLAM	CROWD SLAM	SG SLAM	Occlusion SLAM
walking_xyz	0.0487	0.0155	0.0671	0.0221	0.0176	**0.0151**
walking_static	0.0066	0.0065	0.0102	0.0073	0.0082	**0.0065**
walking_half	0.0329	0.0265	0.0124	0.0352	0.0349	**0.0286**
Crowd	0.1255	0.0204	0.0414	0.0184	0.0264	**0.0177**
Crowd2	0.1065	0.0393	0.0437	0.0300	0.0808	**0.0459**
Crowd3	0.0450	0.0414	0.0432	0.0360	0.1060	**0.0332**

to demonstrate the superiority of occlusion point detection module compared to other methods that use motion consistency modules. Instead the goal is to highlight the importance of including the occluded points in bundle adjustment process. We expect that the proposed module can serve as an anchor for a motion consistency module. In cases where a motion consistency module fails to detect a dynamic point, this method can help reduce the impact of false static points and provide greater robustness in the trajectory estimation process.

The average computation time for each frame is shown in Table 4. These results illustrate total computation time after applying segmentation, without and with occluded point detection module. It is evident from the results that the only bottleneck of our method is the segmentation module, as occluded point detection module requires only 2-5 ms per frame across datasets. The accuracy of the proposed module can be further improved by increasing the number of iterations in the *IterativePNP* algorithm; however, doing so would also raise the computational cost of the occlusion detection module.

However, Occlusion SLAM may face challenges in certain situations. First, as Occlusion SLAM depends only on the segmented mask and starts by assuming every feature outside the segmented region as static, it will certainly fail if features are extracted from the objects which are not identified as dynamic by segmentation. For example, Figure 6 shows the extracted features from a moving box. In this case, trajectory estimation gets badly affected by the features on the moving box. Furthermore if a dynamic point is classified as an occluded point, it can produce a negative impact on the estimated trajectory.

Secondly, as mentioned above, if the number of occluded points is insufficient, then they will not contribute significantly towards an improvement of the estimation. Finally, if an object that is not predefined as dynamic (such as a box) remains static for a long duration and does contribute to generating occluded points, it can disrupt the estimation if the object later becomes dynamic.

Table 4: Comparison of time (ms) performance between ORB SLAM2 , segmentation with occluded point detection module and segmentation with occluded point detection module.

Dataset	ORB SLAM2	Seg (without occlusion)	Seg (with occlusion)
walking_xyz	15.74	23.58	**28.02**
walking_static	18.12	23.15	**28.48**
walking_half	15.83	24.09	**26.67**
Crowd	19.92	19.05	**25.61**
Crowd2	13.03	18.62	**21.25**
Crowd3	15.13	21.06	**24.49**

4 CONCLUSION

In this article we propose a method for detecting an occluded point and utilizing it for better estimation in visual SLAM. This method, named Occlusion SLAM, is based upon the framework of the ORB SLAM2, with the additional segmentation module and occluded point detection module. As a segmentation model, we use a faster and more accurate version of YOLO, called YOLOv8. The mask created by the segmentation module is used in the occluded point detection module.

An evaluation over TUM and Bonn datasets has shown that our choice of segmentation model and preferential treatment of the occluded points produces more than 98% improvement in trajectory estimation over ORB

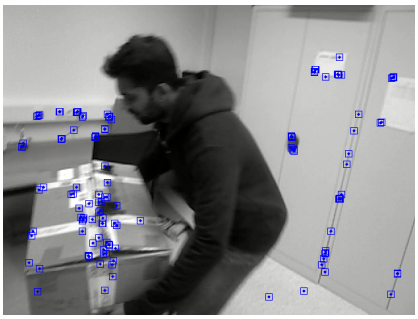


Figure 6: Extracted features on a moving box not identified as dynamic, leading to degraded trajectory estimation in Occlusion SLAM.

SLAM2. Results also reveal, that if a keyframe has enough occluded points, it can significantly reduce the overall error in both motion only bundle adjustment, and local bundle adjustment. Our method also shows competitive results, when compared with various other substitute methods such as DS SLAM, DYNA SLAM, RDS SLAM, CROWD SLAM and SG SLAM. Furthermore, the computational analysis revealed a required processing time of only 2-5ms per frame.
As future extension, this method could be further enhanced by integrating various geometric modules along with our proposed occlusion module, to ensure that occlusion properties are applied exclusively to genuine static points. This is expected to remedy a disruptive limitation that may arise through any false occlusion interpretation as mentioned earlier. Additionally, during experiments, we observed that no existing dataset provides an effective scope of occlusion detection. In a parallel work, we intend to create a dataset sequence specifically designed to give such scope in ample measure.

5 REFERENCES

[BFCN18] Berta Bescos, Jose M. Facil, Javier Civera, and Jose Neira. Dynaslam: Tracking, mapping, and inpainting in dynamic scenes. *IEEE Robotics and Automation Letters*, 3(4):4076–4083, October 2018.

[BKC17] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

[BWL20] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *ArXiv*, abs/2004.10934, 2020.

[BZXL22] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact++ better real-time instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(2):1108–1121, February 2022.

[CER+21] Carlos Campos, Richard Elvira, Juan J. Gomez Rodriguez, Jose M. M. Montiel, and Juan D. Tardos. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, December 2021.

[CM20] Linyan Cui and Chaowei Ma. Sdf-slam: Semantic depth filter slam for dynamic environments. *IEEE Access*, 8:95301–95311, 2020.

[CSZZ23] Shuhong Cheng, Changhe Sun, Shijun Zhang, and Dianfan Zhang. Sg-slam: A real-time rgb-d visual slam toward dynamic scenes with semantic and geometric information. *IEEE Transactions on Instrumentation and Measurement*, 72:1–12, 2023.

[CWC+22] Tianheng Cheng, Xinggang Wang, Shaoyu Chen, Wenqiang Zhang, Qian Zhang, Chang Huang, Zhaoxiang Zhang, and Wenyu Liu. Sparse Instance Activation for Real-Time Instance Segmentation . In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4423–4432, Los Alamitos, CA, USA, Jun 2022. IEEE Computer Society.

[EHE+12] Felix Endres, Jürgen Hess, Nikolas Engelhard, Jürgen Sturm, Daniel Cremers, and Wolfram Burgard. An evaluation of the rgb-d slam system. In *2012 IEEE International Conference on Robotics and Automation*, pages 1691–1696, 2012.

[HGDG17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[JCQ23] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023.

[JJY+23] Jing Jin, Xufeng Jiang, Chenhui Yu, Lingna Zhao, and Zhen Tang. Dynamic visual simultaneous localization and mapping based on semantic segmentation module. *Applied Intelligence*, 53:1–15, 03 2023.

[KKS09] Abhijit Kundu, K Madhava Krishna, and Jayanthi 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*, pages 4306–4312, 2009.

[KM07] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, 2007.

[KMR+23] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[LLC+20] Feng Li, Weijun Li, Wenfeng Chen, Weifeng Xu, Lingqing Huang, Dan Li, Shuting Cai, Ming Yang, Xiaoming Xiong, and Yuan Liu. A mobile robot visual slam system with enhanced semantics segmentation. *IEEE Access*, PP:1–1, 01 2020.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[LW10] Kuen-Han Lin and Chieh-Chih Wang. Stereo-based simultaneous localization, mapping and moving object tracking. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3975–3980, 2010.

[MAMT15] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardos. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, October 2015.

[MAT17] Raul Mur-Artal and Juan D. Tardos. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, October 2017.

[MUS16] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: A hands-on survey. *IEEE Transactions on Visualization and Computer Graphics*, 22(12):2633–2651, 2016.

[PBL+19] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss. ReFusion: 3D Reconstruction in Dynamic Environments for RGB-D Cameras Exploiting Residuals. In *International Conference on Intelligent Robot Systems (IROS)*, 2019.

[RF18] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[RZZH24] Chenyu Ruan, Qiuyu Zang, Kehua Zhang, and Kai Huang. Dn-slam: A visual slam with orb features and nerf mapping in dynamic environments. *IEEE Sensors Journal*, 24(4):5279–5287, 2024.

[SEE+12] 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)*, pages 573–580, Oct. 2012.

[SGM21] João Soares, Marcelo Gattass, and Marco Meggiolaro. Crowd-slam: Visual slam towards crowded environments using object detection. *Journal of Intelligent & Robotic Systems*, 102, 06 2021.

[SZL+18] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018.

[TLD+13] Wei Tan, Haomin Liu, Zilong Dong, Guofeng Zhang, and Hujun Bao. Robust monocular slam in dynamic environments. In *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 209–218, 2013.

[WGG+22] Wenxin Wu, Liang Guo, Hongli Gao, Zhichao You, Yuekai Liu, and Zhiqiang Chen. Yolo-slam: A semantic slam system towards dynamic environment with geometric constraint. *Neural Computing and Applications*, 34:1–16, 04 2022.

[YLL+18] Chao Yu, Zuxin Liu, Xin-Jun Liu, Fugui Xie, Yi Yang, Qi Wei, and Qiao Fei. Ds-slam: A semantic visual slam towards dynamic environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1168–1174, 2018.

[YM21] Liu Yubao and Jun Miura. Rds-slam: Real-time dynamic slam using semantic segmentation methods. *IEEE Access*, PP:1–1, 01 2021.

[YWC+22] Yingxuan You, Peng Wei, Jialun Cai, Weibo Huang, Risheng Kang, and Hong Liu. Misd-slam: Multimodal semantic slam for dynamic environments. *Wireless Communications and Mobile Computing*,

2022:1–13, 04 2022.

[YX23] Lemeng Wu Yunyang Xiong, Bala Varadarajan. Efficientsam: Leveraged masked image pretraining for efficient segment anything. *arXiv:2312.00863*, 2023.

[ZWZ+18] Fangwei Zhong, Sheng Wang, Ziqi Zhang, China Chen, and Yizhou Wang. Detect-slam: Making object detection and slam mutually beneficial. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1001–1010, 2018.

[ZZC+25] Fan Zhu, Yifan Zhao, Ziyu Chen, Chunmao Jiang, Hui Zhu, and Xiaoxi Hu. Dygs-slam: Realistic map reconstruction in dynamic scenes based on double-constrained visual slam. *Remote Sensing*, 17(4), 2025.