HO3-SLAM: Human-Object Occlusion Ordering as Add-on for Enhancing Traversability Prediction in Dynamic SLAM

Jonathan Tay Yu Liang
Department of Human and Artificial Intelligence Systems
University of Fukui
Fukui, Japan
mf228029@g.u-fukui.ac.jp

Tanaka Kanji
Department of Human and Artificial Intelligence Systems
University of Fukui
Fukui, Japan
tnkknj@u.fukui.ac.jp

***Abstract*— Detecting traversable regions from monocular vision is a fundamental problem for indoor robot navigation. In existing works, this is typically formulated as the problem of reconstructing floor regions that are directly visible to the robot. However, the assumption of floor visibility is often violated in crowded scenes such as office environments, which make the existing solutions less effective. To complement existing solutions, we propose a novel approach based on pedestrian observation, i.e., observing the behaviour of pedestrians assuming that the floor region they are standing on is traversable. In our observation, it is recently becoming realistic to implement this new approach by combining state-of-the-art of computer vision techniques, including (1) detection of dynamic objects (e.g., YOLO object detector), (2) absolute localization of stationary objects (e.g., SLAM), and (3) relative localization between dynamic and stationary objects (e.g., Occlusion ordering). A prototype system was implemented by extending state-of-the-art techniques of dynamic SLAM and object detector models, and its basic performance was demonstrated through real-world experiments. We believe that our approach can offer a valuable solution for challenging application domains such as human robot collaboration and long-horizon action planning.**

I. INTRODUCTION

Detecting walkable areas using monocular cameras is crucial for safe and efficient robot navigation. This problem has been extensively studied in computer vision, with various approaches such as obstacle detection [1], free space detection [2], and 3D reconstruction [3, 4, 5]. Many researchers have used deep learning models to predict pixel-level depth values from images [1], while others have developed vision-based algorithms that gradually expand into free space from the robot's feet [2]. Additionally, some studies have focused on reconstructing 3D geometric models using the robot's onboard camera view sequence [3, 4, 5]. However, most of the existing studies assume that the floor area is directly visible, which is not always the case in crowded scenarios as shown in Fig.1. For instance, in crowded scenes like an office where humans and robots collaborate [6], obstacles (e.g., desks) often occlude the floor areas humans are standing on. In such situations, floor detection methods become ineffective, and neither obstacle detection nor 3D reconstruction methods can provide floor information.

To address this issue, we propose an innovative approach to learning traversable areas by tracking pedestrians, i.e., observing the behaviour of pedestrians assuming that the floor region they are standing on is traversable. From our observation, it is recently becoming possible to implement this new approach by combining state-of-the-art of computer vision techniques, including (1) detection of dynamic objects (e.g., YOLO [7] object detector), (2) absolute localization of stationary objects (e.g., DynaSLAM [8]), and (3) relative localization between dynamic and stationary objects (e.g., Occlusion ordering [9]). Moreover, typical objects in human environments like desks are not tall enough to occlude a pedestrian's entire body. We conducted preliminary experiments that showed how deep learning based YOLOv7 [7] detectors could easily detect regions in human images. Although the lower body and floor contact areas of humans are often occluded, the head and upper body are detectable and not occluded. Finally, we infer the occlusion order of stationary objects and pedestrians in the same image coordinate system to localize the pedestrians in terms of their position relative to the stationary objects, whose absolute coordinates are reconstructed.

We will be using Detectron2 instance segmentation mask method [10] and bounding box method for occlusion order reasoning. After that, we make an ablation study on the combination of the methods to achieve more desirable results.

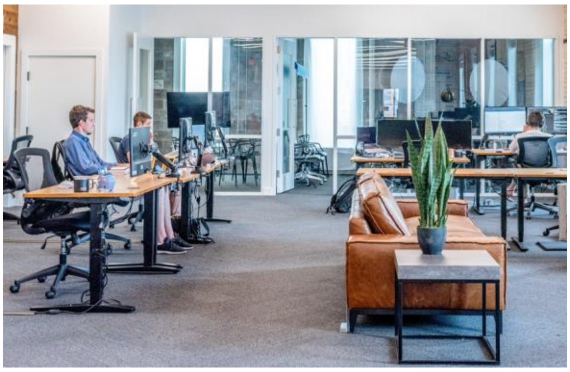


Figure 1:Crowded Environment *Photo by Austin Distel on Unsplash*

In summary, our approach to learning traversable areas by tracking pedestrians represents a significant improvement over existing methods that rely on the assumption that the floor area is directly visible. By leveraging the maturity of human detection technology and the occlusion patterns of stationary objects and pedestrians, our approach can effectively detect walkable areas in crowded scenes, making robot navigation safer and more efficient.

II. RELATED WORKS

This section presents past approaches to the traversable region detection problem and draws a relation to these previous works of the proposed method. Furthermore, we clearly state the application domain to which the proposed method contributes. In particular, we will focus on work that assumes monocular vision rather than other sensor modalities such as 3D point cloud scanners or stereo cameras.

Traversable region detection problem is closely related to visual reconstruction problems such as structure-from-motion (SfM) [4] , simultaneous localization and mapping (SLAM) [3], [11], [12] and visual odometry (VO) [5]. These approaches aim to reconstruct the robot's ego-motion and geometric environment models from the robot's view sequence. Among them, SLAM, a problem setup that

979-8-3503-3337-4/23/$31.00 ©2023 IEEE

sequentially updates environmental geometric models and maps from live view sequences, is most relevant to this work. Existing solutions of SLAM can be divided into filtering-based [13] and optimization-based methods [14], with the latter dominating the current state of the art. While most of existing optimization-based methods assume stationary or semi-stationary environments, new types of SLAM techniques have emerged in recent years that addresses a dynamic environment [12], as we focus on in this study. In particular, our implementation is inspired by one of the state-of-the-art technologies, DynaSLAM [8], which will be detailed in the subsections described later. However, all of the existing reconstruction techniques only targeted specific traversable regions that are directly visible from the robot. In contrast, our approach aims to predict traversable regions that are not directly visible by using the indirect means of pedestrian observation.

Traversable region detection technology contributes to several important applications of visual navigation such as point goal navigation [15], object goal navigation [16] and coverage path planning [17]. In these applications, representations based on traversable regions are often used for state recognition and action planning. Compared with existing traversable region detection techniques that aim to reconstruct directly visible regions, our approach extends the scope of the reconstruction target from visual regions to even invisible regions which pedestrians are standing on. This brings a number of benefits. First, from the perspective of human-robot collaboration [6], the area where the pedestrian is standing often contains more contextual and semantic information that is important for the task compared to other areas. Second, from the perspective of state recognition and action planning, considering not only the visible area but also the invisible areas enable rich state representation and long-horizon exploration planning [18].

It should be noted that there are other approaches to traversable region detection [19], such as obstacle detection and floor detection [2] and monocular depth regression [1]. These approaches again aim to recognize obstacle-free regions or traversable floor regions that are directly visible to the robot. Therefore, they and our approach are in a complementary relationship.

A. *DynaSLAM*

DynaSLAM is a state-of-the-art visual simultaneous localization and mapping (SLAM) algorithm that builds on the foundation of ORB-SLAM2 [20]. It is capable of tracking, mapping, and background inpainting in dynamic scenes, making it highly robust in a variety of configurations, including monocular, stereo, and RGB-D. One of the primary challenges in dynamic SLAM is distinguishing between camera motion and object motion within the scene. DynaSLAM overcomes this by using a probabilistic model to estimate the likelihood of an object's motion. If an object's motion is inconsistent with the predicted motion based on the camera's motion, it is identified as a dynamic object and removed from the map.

Our approach shares the design philosophy with DynaSLAM in two respects. First, our approach is also modular, with several modules such as static object modeling and dynamic object detection, each of which is connected to the ever-growing cutting-edge research field, including object detection [7], tracking [21], scene parsing [22], reconstruction [3], inpainting [23], occlusion reasoning [9], and action planning [17]. Second, the module group for stationary object recognition and the module group for dynamic object recognition are clearly separated, which allows us to focus on research and development of the latter module group without modifying the former module group. However, our approach is different from DynaSLAM for many aspects. In particular, it introduces a new type of observation that is independent of the observations used by DynaSLAM. That is, relative observations of interactions between stationary and dynamic objects are used by our new observation model, in contrast to the absolute metric observations from stationary and dynamic objects used by DynaSLAM.

III. APPROACH

Our approach leverages the strengths of ORB-SLAM3 [24] and YoloV7 [7] human detection model to address the challenge of deducing the occlusion order of humans and objects in crowded environments. ORB-SLAM3 is a leading visual SLAM algorithm that provides real-time tracking of the camera pose and map features. Meanwhile, YoloV7 is a highly accurate and efficient deep learning-based object detection model, which has superior performance in human detection.

Our framework has three sub-modules: static object modeling, moving object detection, and occlusion order inference. Given these modules, the system localizes the pedestrian region on the floor by sequentially performing static object modeling, moving object detection, and occlusion order inference. The subsections below provide details on each module.

A. *Static Object Modelling*

The generation of a 3D point cloud from a live image sequence using ORB-SLAM3 [24] marks a significant advancement in the field of vision and robotics. The problem of SLAM, which involves simultaneously navigating and mapping an unknown environment, is a critical one with numerous practical applications, including autonomous navigation and exploration.

To address the SLAM problem, existing approaches can be broadly categorized into two main types: filter-based and optimization-based methods. Filter-based methods such as Extended Kalman Filter (EKF) [25] and Particle Filter estimate the robot's pose and map incrementally by updating the estimates with each new sensor reading. In contrast, optimization-based methods such as Graph-SLAM [26] and Bundle Adjustment formulate the SLAM problem as an optimization problem, with the goal of finding the most likely map and robot pose to explain the observed sensor data.

In this paper, ORB-SLAM3 is employed as a powerful tool for SLAM applications. ORB-SLAM3, developed by the computer vision group at the University of Zaragoza, Spain, is the latest version of the ORB-SLAM [27] family of popular SLAM algorithms. It can be used with monocular, stereo, or RGB-D cameras and builds on the strengths of its predecessors, ORB-SLAM and ORB-SLAM2 [20], while also incorporating new features and improvements.

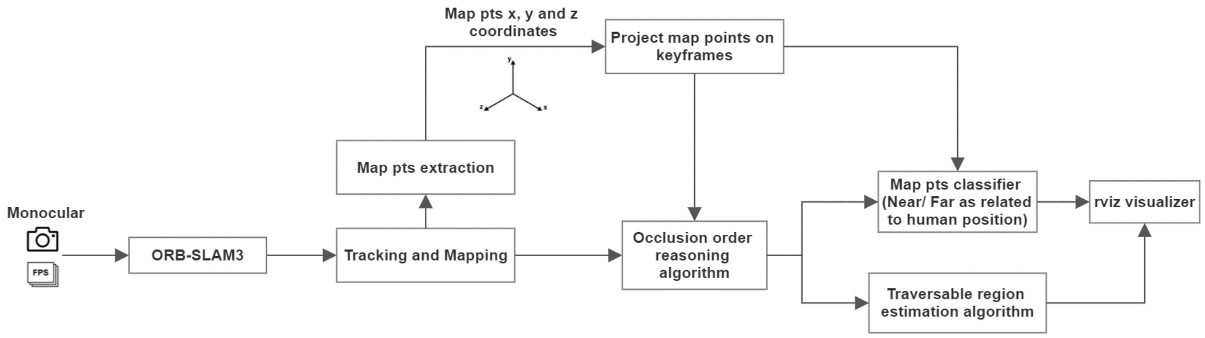


Figure 2: Block diagram of our proposed framework. A segmented video stream in rosbag format is pass into ORB-SLAM3 [24]. The feature points coordinates are extracted and projected onto the keyframes for visualization. By comparing the map pts 3D location/distance to the camera and the location of the human to the camera, we classify the map pts to two simple classes (Near and Far). We can obtain the human estimated location in the 3D map and hence estimate the traversable region. The final outcome can be visualized using Rviz [30].

One of the most significant improvements in ORB-SLAM3 is the use of a new feature descriptor called DBoW3 [28] , which enhances the system's robustness to changes in lighting, scale, and viewpoint. The output of ORB-SLAM3 is a point cloud of ORB feature points, which represents a highly accurate and detailed map of the robot's environment.

Overall, the use of ORB-SLAM3 for generating 3D point clouds from live image sequences represents a significant step forward in the field of SLAM, with numerous practical applications in the areas of robotics and computer vision.

B. Dynamic Object Detection

In our study, we utilized two different approaches for pedestrian detection: YOLO V7 [7], a representative object detector and the state-of-the-art model at the time of this writing, and Detectron2, which we used to treat pedestrian detection as a semantic segmentation problem. While YOLO V7 represents the human region as a bounding box, Detectron2 [10], represents it as a free-form region, allowing for a more accurate approximation of the human domain. Conversely, YOLO V7's bounding box representation offers a high frame rate, making it an attractive option in scenarios where speed is critical.

To compare the performance of these two methods, we conducted experiments to evaluate their accuracy and computational speed in the common field of real-time processing. Our results, which are reported in the experimental section, shed light on the relative strengths and weaknesses of the two approaches. Ultimately, our study provides valuable insights for researchers and practitioners seeking to optimize pedestrian detection performance for real-time applications.

C. Occlusion Order Inference

Pedestrian tracking is a fundamental task in computer vision that aims to locate and follow people's movement in visual scenes. However, occlusions caused by other humans or objects can pose significant challenges to accurate pedestrian localization, particularly in crowded environments. To address this issue, we have developed a technique called occlusion-ordering-based relative-depth estimation which aims to analyze depth values to determine the occlusion order of humans and objects.

By leveraging occlusion-ordering information, it becomes possible to discern which person or object appears to be in front of another. Specifically, when one person's occlusion order is consistently greater than that of another person or object, it suggests that the latter is occluded by the former. This method is particularly useful in crowded environments where people and objects are likely to overlap, making it difficult to accurately track individual pedestrians.

The implementation of occlusion-ordering-based relative-depth estimation has practical applications for various fields, including indoor robot navigation, where accurate pedestrian localization is necessary for detecting traversable regions. By improving the accuracy of pedestrian tracking, this approach can enhance the ability of robots to navigate crowded indoor environments and operate safely and efficiently in these settings.

D. Implementation Details

We developed a prototype of the proposed system by adding and modifying the public code of ORB-SLAM3 [24]. In particular, we modified some of the original code by adding some functionalities with some built-in ORB-SLAM3 APIs. From Fig. 2 above, we can see the flowchart block diagram of the framework. We integrated our developed occlusion ordering algorithm with the Detectron2 mask to acquire this framework. The implementation is discussed in detail in this subsection.

Feature points and keyframes extractions:

We first input our video stream into ORB-SLAM3 to obtain a map with sparse feature points. In ORB-SLAM3, the feature points are obtained using the Oriented FAST and Rotated BRIEF (ORB) feature detector and descriptor. The ORB feature detector identifies key points in the image based on their high intensity gradients and corner-like structures. Once the key points are detected, the ORB descriptor is computed for each key point. These key points serve as potential feature points. In this context, the feature points are shown as green feature points in our map. We concurrently extract the feature points and project it onto the keyframes. To achieve this, we modified the original ros_mono.cc, System.cc and System.h files. In ros_mono.cc, a new function is added to save the keyframe trajectory and feature points coordinates. A logic is added to check if a new keyframe is added and save it as images in the image grabber function.

Two functions are added to System.cc and System.h:

GetLastKeyFrameID()

SaveMapAndTracjectory()

Once we have the keyframes and feature points ready, we use python code to perform the projection. Initially, the keyframe pose is a quaternion, so we need to convert it into a 3x3 matrix before passing it to OpenCV [29] that calculates it.

Projection of feature points onto keyframes:

We make use of the keyframes, and feature points extracted from the map earlier to make the projection.

We can visualize the projection with Rviz [30] visualizer as shown in Fig.3 below. The trajectory and feature points can be seen clearly in this visualizer. The green points are the feature points that are being projected onto the current keyframe, and the red points are the feature points that have been built.

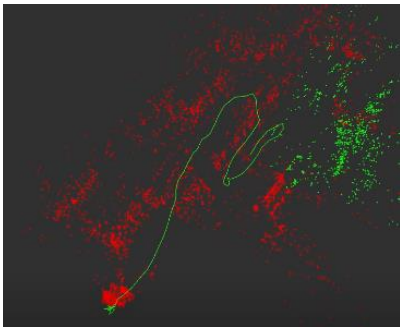


Figure 3:Projected feature points as shown in Rviz visualizer

Occlusion ordering algorithm:

The occlusion ordering algorithm makes use exclusively of the human area range and the feature points coordinates in 2D space. This is simply a 2-class classification problem where we classify the feature points as in front of or behind the human with higher probability.

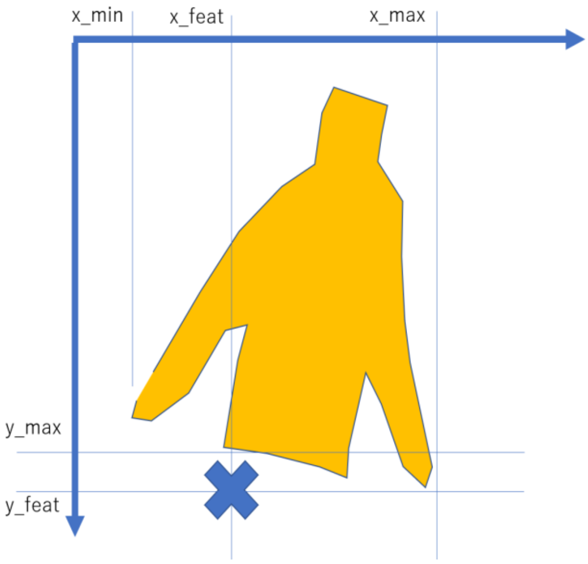


Figure 4:Occlusion ordering algorithm

As shown in Fig.4 above, the "X" represents the feature points with coordinates (x_feat, y_feat). (x_min, x_max, y_min, y_max) represents the human area range. A simple if-else statement loop can determine and estimate the position of the feature points with higher probability. The simplicity of this algorithm can provide a robust system which requires less computation power to operate an on-line process. It could efficiently be integrated into SLAM systems and be used in environments with greatly degraded visual information.

We label the feature points in our output video for a better visualization of the algorithm. The cases for the feature points labels are as follows:

d = '-1' when the feature points are in front of the human

d = '1' when the feature points are behind of the human

d = '0' when the feature points are not available/ not in range of the human area.

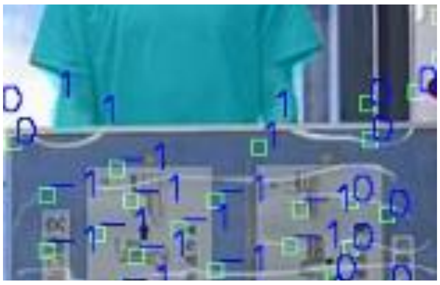


Figure 5:Feature points occlusion reasoning value annotations

As shown in Fig. 5 above, the feature points which are in range of the human area are correctly classified as '-1' which means that the feature points are in front of the human. On the other hand, the feature points that are in the x range and out of y range can be classified as '1' which means the feature points are behind the human with a higher probability. In a '0' case, it means that the feature points are out of the human area range and are not available to be classified.

We execute the occlusion ordering algorithm and then projects the feature points onto keyframes to find the estimated location of the human in 3D space. Once again, the outcome can be visualized with rviz visualizer with colored feature points.

Figure 6: Feature points annotations with occlusion ordering algorithm

As shown in Fig. 6 above, the board that is behind the human has a lot of feature points built by ORB-SLAM3 [24]. The feature points 3D coordinates are used to be compared with the human location and hence are classified as '1' which means that is behind the human. We can ultimately visualize this with Rviz [30] visualizer. For the scene in Fig.6, as shown in Fig. 7 below, we can observe that there are points that are colored as Orange and Purple.

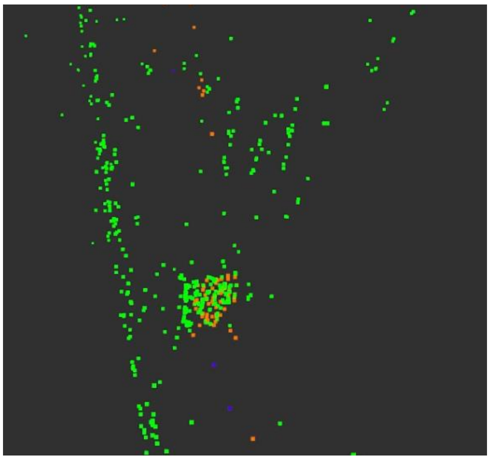


Figure 7:Feature points colored for visualization purpose

The orange cluster is the board which is behind the human at this scene. All the points that are labeled as one and observed at the current frame are colored as orange and classified as behind the human. On the other hand, feature points that are classified as purple are the points that are in front of the human.

IV. Experimental Results

A. Experimental environment setup:

We first set up the test environment in a wide area to build a map with ORB-SLAM3 [24] as shown in Fig.8 below. A large object, such as a board, is positioned somewhere in the middle to serve as the occlusion object.

Figure 8:Experimental Environment setup

The video stream is utilized to track and map the environment , aiming to generate a map with a sufficient number of sparse feature points using ORB-SLAM3. Consequently, the environment needs to be spacious enough to ensure the acquisition of a high-quality map with sparse feature points.

Subsequently, a second video stream featuring a human subject is employed for the occlusion ordering algorithm, as shown in Fig.9 above. The human subject is required to move around and act as a dynamic object.

Figure 9:Human in occluded scenario

B. Preparing with data obtained:

The video stream is first input in the form of rosbag for ORB-SLAM3 [24] to run. After that, a map with sparse feature points is acquired and observed as shown in Fig. 10 below.

Figure 10: ORB-SLAM3 map

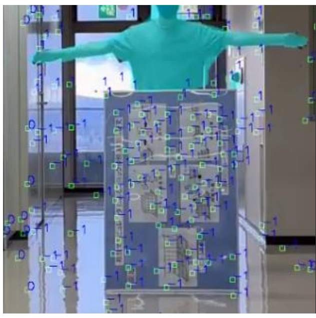
We have now obtained a map with all the coordinate information included. We save the extracted keyframe poses and feature point coordinates, which serve as the outputs ready for projection. Once the projection is completed, the keyframe images and projected keyframes are saved as images. We can visualize the projection using the rviz visualizer [30] , as showed in Fig.3. The resulting output of the mapping process is the visualize.rviz file. At this stage, we have obtained all the necessary data for the occlusion ordering algorithm. With the following input files:

- Feature points coordinates file
- visualize.rviz
- <Video> (Any form of video steam input)

We can proceed with the occlusion ordering process which identifies the occlusion order of the feature points relative to the human.

C. Study with annotated ground truth:

We randomly sampled 50 green feature points and manually annotated their correct classifications (Behind or in front). By comparing the ground truth with our system's runs, we investigated the algorithm's performance. We randomly selected 5 frames and calculated the average percentage of correctly labeled feature points.



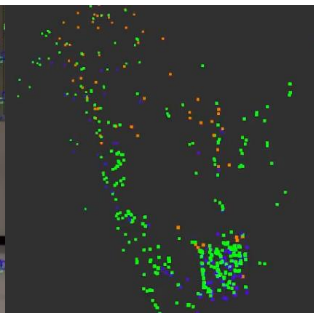


Figure 11: Ground truth study *Figure 12: Colored feature points*

All the feature points are annotated relative to the human position, as shown in Fig. 11 above. Observation by eyes and manual annotation is required to obtain an accurate ground truth. The framework classifies the feature points with different colors in rviz [30] for visualization purposes. As shown in Table 1 below, an outstanding correctness rate of 96.71% has been achieved.

The results are recorded in the Table 1 below:

No	Labeled feature points				
	GT_Front	***GT_Behind***	***R_Front***	***R_Behind***	***%***
1	2	48	2	48	100.00
2	4	46	4	46	100.00
3	7	43	7	43	100.00
4	35	15	32	18	91.43
5	38	12	35	15	92.11
					Average = 96.71

Table 1:Ground truth vs Actual run performance (Abbreviations: GT – Ground Truth, R – Run)

For ground truth assessment, we randomly selected 5 frames and manually annotated the green feature points. We compared the accuracy of our framework with the ground truth annotations to evaluate its performance. Based on our experiments, the algorithm demonstrates exceptional performance when the human is positioned in front of the occlusion. Our proposed framework exhibits robustness and is deemed to perform well.

D. Ablation Study:

Table 2 below presents the results of a series of ablation studies that combine the modules in the proposed framework. These studies utilize human-object occlusion ordering datasets to evaluate the performance of the methods employed. The occlusion ordering algorithm is executed using the YOLO v7 [7] human detection model. Traditionally, bounding boxes are applied around the detected objects when using such human detection models. In our case, we can directly use the bounding box edges (x_min, x_max, y_min, y_max) to determine the position of the feature points using the algorithm. A detailed explanation is provided in Figure 4 above. Alternatively, we can incorporate the Detectron2 instance segmentation mask for this purpose. It is worth noting that during the ORB-SLAM3 [24] process, unwanted map can occasionally be generated on dynamic objects, which in our case, is the human. To address this issue, we can apply a filter using the same Detectron2 mask on the human area . This filtering method is essentially what we have employed in our proposed framework.

We processed 3 different datasets and evaluated their performances using different methods. The correctness ratio represents the percentage of correctly labeled feature points compared to manually annotated ground truths. The method used is the same as shown in Table 1. We randomly sampled 50 feature points from random frames and annotated them through observation. The limitations of the bounding box method in obtaining an accurate contour of the human pose present an obvious flaw in this study. For example, when the human is positioned behind an object, the bounding box may include points that are located at the side of the human contour, leading to a larger inaccuracy in the estimation.On the other hand, utilizing Detectron2 to acquire the contour of the human shape offers a significantly more reliable and accurate method. This approach allows us to filter out unwanted feature points associated with the human and obtain a precise boundary that corresponds to the actual human pose.

	1st	***2nd***	***3rd***
Mask with Mask filter (Proposed)	1.0	0.96	0.98
B.Box with Mask filter	0.87	0.79	0.78
B.Box	0.75	0.71	0.73

Table 2:Ablation studies with different module

E. Segmentation Methods Study:

We tested various segmentation methods for our purpose, including instance segmentation, semantic segmentation, panoptic segmentation and image segmentation. Interestingly, we found that all performance results were consistently similar across all methods.This finding suggests that the proposed method is not significantly influenced by the choice of segmentation approach.

V. CONCLUSIONS

In conclusion, our research introduces a simple yet robust human-object occlusion ordering framework for estimating traversable regions in crowded environments using SLAM systems. By leveraging the position of pedestrians, we can accurately predict walkable areas based on the occlusion order between humans and objects. The framework's simplicity makes it easy to implement and integrate into existing SLAM systems, while still demonstrating remarkable robustness in handling occlusions in complex and dense environments. This has significant implications for various applications, such as robot navigation and autonomous vehicles, where accurately predicting walkable areas is crucial for safe and efficient movement.

VI. REFERENCES

[1] A. Bhoi, "Monocular Depth Estimation: A Survey," *arXiv:1901.09402*, p. 8, 2019.

[2] Y. Xu and P. Ghamisi, "Consistency-Regularized Region-Growing Network for Semantic Segmentation of Urban Scenes with Point-Level Annotations," *IEEE Trans. Image Process*, vol. 31, pp. 5038-5051, 2022.

[3] C. Cadena , L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid and J. J. Leonard, "Past, Present, and Future of Simultaneous Localization And Mapping: Towards the Robust-Perception Age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309-1332, 2017.

[4] D. Moran, H. Koslowky, Y. Kasten, H. Maron, M. Galun and R. Basri, "Deep Permutation Equivariant Structure From Motion," *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5976-5986, October 2021.

[5] S. Wang, R. Clark, H. Wen and N. Trigoni, "DeepVO: Towards End-to-End Visual Odometry with Deep Recurrent Convolutional Neural Networks," *In 2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2043-2050, 2017.

[6] M. Selvaggio, M. Cognetti, S. Nikolaidis, S. Ivaldi and B. Siciliano, "Autonomy in Physical Human-Robot Interaction: A Brief Survey," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7989-7996, 2021.

[7] A. Bochkovskiy, H.-Y. M. Liao and C.-Y. Wang, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv,* 2022.

[8] 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.

[9] J. Tighe, M. Niethammer and S. Lazebnik, "Scene Parsing with Object Instances and Occlusion Ordering," *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* June 2014.

[10] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo and R. Girshick, "Detectron2," Facebook, 2019. [Online]. Available: https://github.com/facebookresearch/detectron2. [Accessed 19 03 2023].

[11] A. J. Davison, I. D. Reid, N. D. Molton and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 29, no. 6, pp. 1052-1067, 2007.

[12] L. Zhao, B. Wei, L. Li and X. Li, "A Review of Visual SLAM for Dynamic Objects," *2022 IEEE 17th Conference on Industrial Electronics and Applications (ICIEA）,* pp. 1080-1085, 2022.

[13] M. Lin, C. Yang and D. Li, "An Improved Transformed Unscented FastSLAM With Adaptive Genetic Resampling," *IEEE Transactions on Industrial Electronics,* vol. 66, no. 5, pp. 3583-3594, 2019.

[14] H. Zhou, Z. Hu, S. Liu and S. Khan, "Efficient 2d graph slam for sparse sensing," *iros,* pp. 6404-6411, 2022.

[15] X. Zhao, H. Agrawal, D. Batra and A. G. Schwing, "The surprising effectiveness of visual odometry techniques for embodied pointgoal navigation," *In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV),* pp. 16127-16136, October 2021.

[16] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," *In 2017 IEEE International Conference on Robotics and Automation (ICRA),* pp. 3357-3364, 2017.

[17] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta and R. Salakhutdinov, "Learning to explore using active neural slam," 2020.

[18] C. Hoang, S. Sohn, J. Choi, W. Carvalho and H. Lee, "Successor feature landmarks for long-horizon goal-conditioned reinforcement learning," *Advances in Neural Information Processing Systems,* vol. 34, pp. 269633-26975, 2021.

[19] D. Stadler and J. Beyerer, "Improving multiple pedestrian tracking by track management and occlusion handling," *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 10958-10967, June 2021.

[20] R. Mur-Artal and J. D. Tardos, "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.

[21] X. Li, C. Ma, B. Wu, Z. He and M.-H. Yang, "Target-aware deep tracking," *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* June 2019.

[22] R. Singh, P. Gupta, P. Shenoy and R. Sarvadevabhatla, "Float: Factorized learning of object attributes for improved multi-object multi-part scene parsing," *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 1445-1455, June 2022.

[23] D. Kim, S. Woo, J.-Y. Lee and I. S. Kweon, "Deep video inpainting," *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* June 2019.

[24] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel and J. D. Tardos, "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.

[25] R. Agramonte and C. Urrea, "Kalman Filter: Historical Overview and Review of Its Use in Robotics 60 Years after Its Creation," vol. 2021, 2021.

[26] S. Thrun and M. Montemerlo, "The GraphSLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures," 2006.

[27] R. Mur-Artal, J. M. M. Montiel and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics,* vol. 31, no. 5, pp. 1147-1163, October 2015.

[28] R. Muñoz-Salinas, "DBoW3," 17 02 2017. [Online]. Available: https://github.com/rmsalinas/DBow3. [Accessed 19 03 2023].

[29] I. Culjak, D. Abram, H. Dzapo, T. Pribanic and M. Cifrek, "A brief introduction to OpenCV," *2012 Proceedings of the 35th International Convention MIPRO, Opatija, Croatia,* pp. 1725-1730, 2012.

[30] W. Woodal, "rviz," ROS, 16 05 2018. [Online]. Available: http://wiki.ros.org/rviz. [Accessed 19 03 2023].