Moving object removal for robust visual SLAM in dynamic environment

Jit Chatterjee, Michèle Rombaut, Bruce Canovas

Moving object removal for robust visual SLAM in dynamic environment

Jit Chatterjee
GIPSA Lab and University of Grenoble Alpes
Grenoble, France
jit.chatterjee@grenoble-inp.org

Supervised by: Michele Rombaut and Bruce Canovas

I understand what plagiarism entails and I declare that this report is my own, original work.
Name, date and signature: Jit Chatterjee, 12th June 2020

Abstract

Visual 3D Simultaneous Localization And Mapping (SLAM) is an important technique to reconstruct a 3D space which helps in the navigation of mobile robots. The classical SLAM systems assume that the environment is static. In dynamic environment, these SLAM systems work in a random manner and affects the SLAM system which degrades the 3D reconstruction as the camera motion estimation gets distorted due to dynamic objects. Due to this reason, in real-time scenario, a mobile robot will have difficulties to navigate in dynamic environments. In this paper, we have proposed a more robust visual SLAM in dynamic environment by eliminating the dynamic objects using Convolutional Neural Network (CNN) and optical flow methods. Our proposed method can generate Sparse 3D maps of dynamic environments by removing the dynamic objects with a robust and more accurate manner.

1 Introduction

Simultaneous Localization And Mapping (SLAM) is the main step for navigation of mobile robots. These SLAM systems are mainly designed keeping in mind for static environment. Dynamic feature points disrupts the SLAM system fully generating wrongly estimated 3D maps. Thus, in recent years with the increase in use of intelligent mobile robots, navigation becomes difficult in dynamic environments. There are many existing 3D SLAM techniques like Monocular SLAM system [Mur-Artal and Tardós, 2015] such as ORB-SLAM and Stereo or RGB-D SLAM systems [Mur-Artal and Tardós, 2017] such as ORB-SLAM2, based on ORB feature points. But these SLAM systems mainly rely on static feature points for the 3D map reconstruction.

In the dynamic environment, the feature points on the dynamic object disrupts the process and should be removed while estimating the camera trajectory path and the 3D map reconstruction, so that they have less impact on the SLAM system for better accuracy. Some traditional methods for moving object detection get rid of dynamic elements by make use of optical flow [Cheng *et al.*, 2019] or probabilistic methods [Sheikh and Shah, 2005] that put weights on feature points or RANSAC [Shao-Wen Yang and Wang, 2009] to filter motion that are not dominant. Usually these methods are fast and lightweight but they are more prone to noise and assume that the static part of the environment is dominant over the dynamic part.

More recent methods rely on Deep Learning and Semantic segmentation [Zhang *et al.*, 2018] methods like DynaSLAM [Bescos and Neira, 2018] which uses Mask R-CNN [He *et al.*, 2017] and Multi-View Geometry to detect the dynamic objects. These methods are robust but as they are based on Mask R-CNN which is a two step detector and needs larger computations for segmentation which are computationally too expensive and also they are hard to get real-time performance in onboard platform and sometimes fail to generalize well, depending on the observed environment. There are other mixed methods where dynamic objects are removed in RGB-D SLAM [Sun *et al.*, 2018] using segmentation and optical flow or in Visual SLAM [Liu *et al.*, 2019] using YOLOv3 [Redmon and Farhadi, 2018] and optical flow. All these methods are not that efficient in real-time scenarios as there are huge number of computations and need larger GPU power to execute efficiently. They are too costly, really expensive with regard to memory and computing power and they don't suit well for embedded systems.

To solve this problem, we propose a much simpler and robust method of detecting the dynamic objects and eliminating them from the RGB-D SLAM system to get more accurate 3D map. Our solution consists of potential moving object detection, generating masks on the depth images and compute an initial camera-trajectory estimation without considering the feature points inside the mask of potential dynamic objects. Then using optical flow and the initial camera-trajectory estimation, the dynamic objects gets detected and the final camera- trajectory path gets generated removing the dynamic feature points. Finally, the 3D map gets reconstructed using the SLAM system. Our proposed method is sort of hybrid combining low cost deep learning system and more traditional methods, making it robust in most cases, so that it can run on a mobile robot in dynamic indoor environment.

2 Global Architecture

We propose a robust RGB-D SLAM system in dynamic environment by eliminating the dynamic feature points which makes the SLAM system stable. First, a Convolutional Neural Network with small model size and fast inference speed detects the potential dynamic objects (human beings for instance) from the RGB images and provides bounding boxes. Now, the bounding boxes of these potential dynamic objects is fed into a Background-Foreground segmentation using the depth images and potential dynamic masks gets generated. An initial camera motion gets estimated without the feature points inside the potential dynamic masks. Then, using optical flow motion and the initial camera motion estimation, true dynamic objects gets detected and finally the camera trajectory gets estimated with the sparse 3D reconstruction using SLAM system.

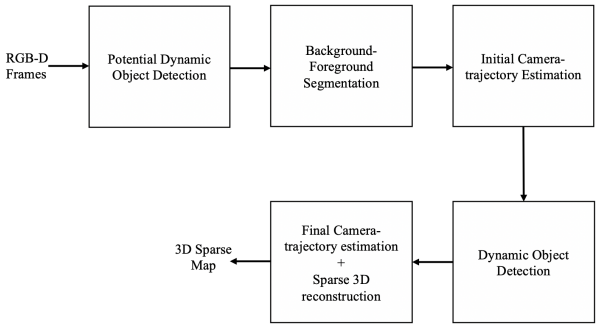

Figure 1: Block Diagram of the proposed method

3 Potential Dynamic Object Segmentation

We have further divided this step into two sub steps using object detection and background-foreground segmentation. For object detection we are using a light-weight CNN for faster computation in real-time environments. For background-foreground segmentation, we are using a more traditional method that is adaptive Gaussian Mixture Model clustering.

3.1 Object Detection

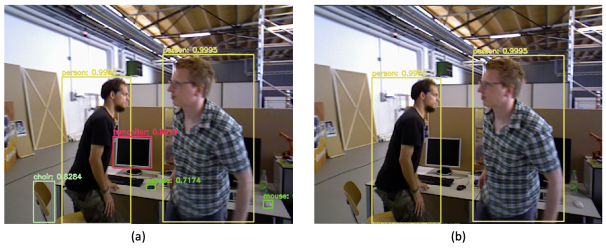

Figure 2: (a) Object detection using YOLO. (b) Potential dynamic object detection.

Object Detection is detecting objects in a frame using Machine Learning or Deep Learning Methods. Various objects like car, human, cats, dogs, chair, desk can be detected using simple methods like Hog Detector [Dalal and Triggs, 2005] or other complex methods using Convolutional neural networks (CNN) like Mask R-CNN [He *et al.*, 2017], Faster R-CNN [Ren *et al.*, 2017]. These are two step detectors and includes high computations. There are also various one step detectors like SSD [Liu *et al.*, 2016], RetinaNet [Lin *et al.*, 2017] and YOLO. These one step detectors are fast and can be used in real-time applications. In our solution, we use YOLOv3-tiny [Redmon and Farhadi, 2018] as it is more robust in terms of accuracy and speed. We have trained YOLOv3-tiny on the COCO dataset [Lin *et al.*, 2014] which comprises of 80 classes containing both static objects (chair, TV-monitor) and dynamic objects (like human beings). We are using RGB images as input to the YOLOv3-tiny which is the neural network used here as it is light-weight, fast even on embedded devices and it detects potential dynamic objects like human beings in the frame and it provides a bounding box around the detected object. YOLO provides multiple bounding boxes around the same object, using non-maximum suppression we can get the bounding box with highest confidence and calculate the Intersection Over Union (IOU) with other boxes of the same object and eliminate other boxes with lower confidence than the threshold. In our solution, we have kept the threshold of 0.3 and IOU of 0.45 for YOLOv3-tiny, as we got optimal results in these values.

3.2 Background-Foreground Segmentation

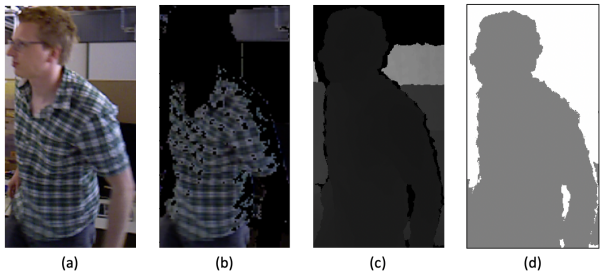

Figure 3: (a) RGB Image of YOLO bounding box. (b) GMM clustering on the RGB image. (c) Depth image of the corresponding RGB image. (d) Background-Foreground Segmentation using adaptive GMM clustering on the depth image.

As represented in Fig 3(a), the bounding boxes of the potential dynamic objects still contains some static part. To remove the static part in the bounding boxes we have implemented a Background-Foreground [Stauffer and Grimson, 1999] Segmentation [Bouwmans *et al.*, 2008] on the corresponding depth images based on adaptive Gaussian Mixture Model (GMM) [Dar-Shyang Lee, 2005] clustering. The GMM is based on the expectation-maximization (EM) [Dempster *et al.*, 1977] algorithm and estimates the background-foreground segmentation with the help of a statistical model of intensity for each pixel in the image frame. We have implemented the adaptive GMM clustering [J *et al.*, 2019] on the depth images for faster and efficient computation and also depth image provides better information for background-foreground segmentation. We have used the al-

gorithm from the open source openCV library. Using the adaptive GMM clustering, masks of the potential dynamic objects are generated.

4 Initial Camera-trajectory estimation

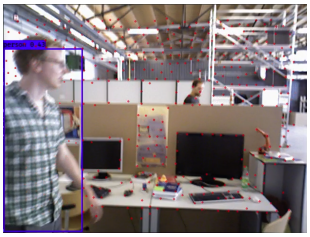

Figure 4: Eliminating feature points from the potential dynamic objects estimated by YOLO and GMM Clustering.

After creating the mask of the potential dynamic objects, an initial camera-trajectory estimation is done using the RGBD-PTAM [Pire *et al.*, 2017] which is a RGB-D SLAM system able to compute the camera trajectory in real-time and to build a sparse 3D map. RGBD-PTAM uses Good Features to Track (GFTT detector of openCV) [Jianbo Shi and Tomasi, 1994] which is based on the Shi-Tomasi method to get the feature points. Before the initial camera motion estimation, the feature points inside the potential dynamic masks are eliminated and not used for the estimation. Only the static feature points are used for the camera motion estimation, so that the error gets decreased.

5 Dynamic Object Detection

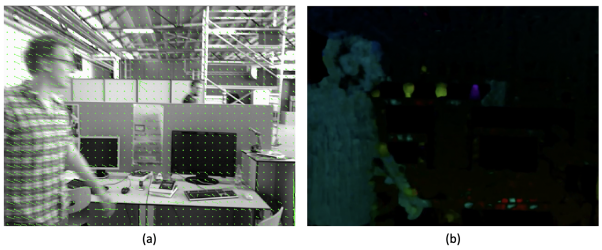

Figure 5: (a) Optical Flow. (b) Dense Optical Flow.

For dynamic object detection we have used the initial camera pose estimation with the optical flow method. Optical flow describes the apparent motion of objects between two frames which is caused due to the dynamic objects or camera movement. Optical flow can be generated using Lucas Kanade method [Lucas and Kanade, 1981] which is the sparse optical flow and Gunnar Farneback method [Farnebäck, 2003] which is the dense optical flow. In our proposed method, we have used the dense optical flow using Gunnar Farneback method for the dynamic object detection. Dense optical flow detects all the pixel intensity change between the two frames and computes the flow vectors (as represented in Fig 5(a)) of the highlighted pixels. For our solution, we have used the dense optical flow as its more accurate and robust than the sparse optical flow.

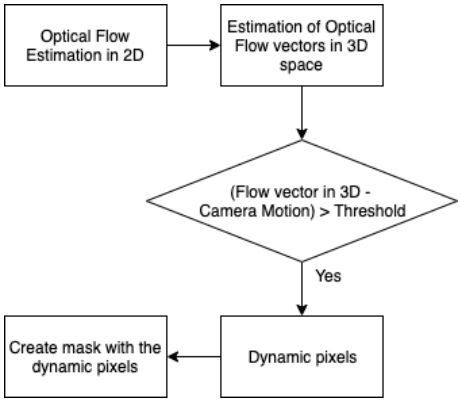

Figure 6: Dynamic Object detection using Optical Flow and Initial Camera motion estimation.

After generating the optical flow in 2D, we have estimated set of flow vectors (p_x, p_y) in 2D space. Using the camera intrinsic parameters (c_x, c_y, f_x, f_y) and the depth values from RGB-D, we have estimated the flow vectors (P_X, P_Y, P_Z) in 3D space. We have converted the 2D frames into 3D point cloud. For 2D pixels p(u,v) and depth d to be converted into 3D pixels P(x,y,z) we use the following equations:

$$x = (u - c_x) * d/f_x$$
$$y = (v - c_y) * d/f_y$$
$$z = d$$

Using the initial camera motion estimation, we can get the rotational and the translational motion from the rotational matrix and the translational values (t_x, t_y, t_z). The optical flow we have computed is from current frame to previous frame and the camera motion estimation is from previous frame to current frame. Using the equation below we have computed the Camera Motion in 3D space.

$$P_M = (R)^T * P - (R)^T * t$$

where, P_M is the position of a 3D point in previous frame obtained by applying the inverted camera motion to the position of a 3D point in current frame, R is the rotational matrix, P is 3D pixels P(x,y,z) and t is the translational value (t_x, t_y, t_z).

$$P_F = (P_X, P_Y, P_Z)$$

where, P_F is a pixel position in previous frame obtained by adding the flow displacement associated to a pixel in current frame and (P_X, P_Y, P_Z) are the flow vectors in 3D space.

In theory, the difference between the pixel position P_F and P_M in 3D space for static pixels should be zero. However, because of noisy depth measurements and due to inaccuracies in the calculated optical flow, the difference in value is not equal to zero. So, we have kept an optimal threshold to determine whether an pixel is dynamic or static. If the difference is greater than the threshold then the pixels are dynamic or else

static. Using these dynamic and static pixels we have created a mask to determine the dynamic objects in the RGB-D frame as presented in Fig 7(b).

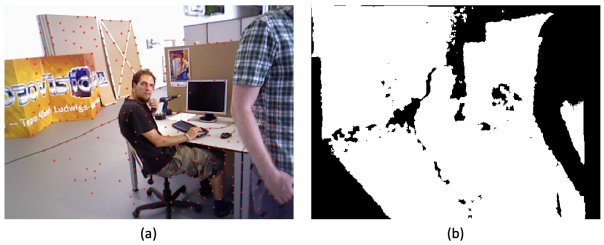

Figure 7: (a) Static features points after eliminating the dynamic feature points. (b) Mask generated using Optical Flow and Initial Camera motion estimation which depicts the dynamic objects in black and static objects in white.

6 Sparse 3D reconstruction

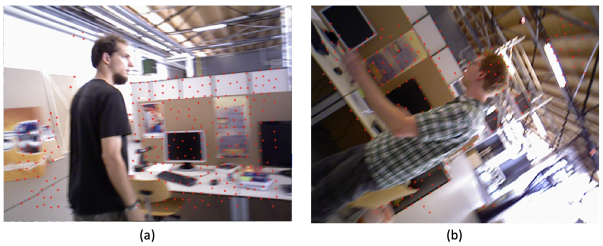

Figure 8: Static features points after eliminating the dynamic feature points

After eliminating the dynamic feature points as presented in Fig 8, we compute the final camera motion estimation using the RGBD-PTAM SLAM system and generate the sparse 3D map reconstruction as presented in Fig 9.

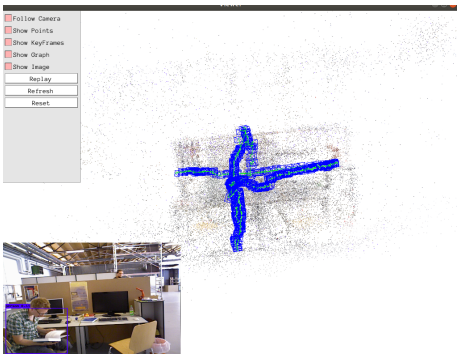

Figure 9: Final Camera-trajectory estimation and Sparse 3D map reconstruction using RGBD-PTAM.

7 Results

We have used the TUM RGB-D [Sturm *et al.*, 2012] dataset to test our proposed RGB-D SLAM system. The TUM RGB-D dataset consists of many sequences but we have tested our model on the dynamic indoor sequences as our robot is an indoor mobile robot. The sequence freiburg3 walking half-sphere is an indoor office high dynamic scene where two persons walk and the RGB-D sensor has been moved on a small half sphere of approximately one meter diameter. This sequence is intended to evaluate the robustness of RGB-D SLAM and odometry algorithms to quickly moving dynamic objects in large parts of the visible scene. We have tested our model on a laptop (with 12 GB RAM and i7 8th generation processor CPU power only).

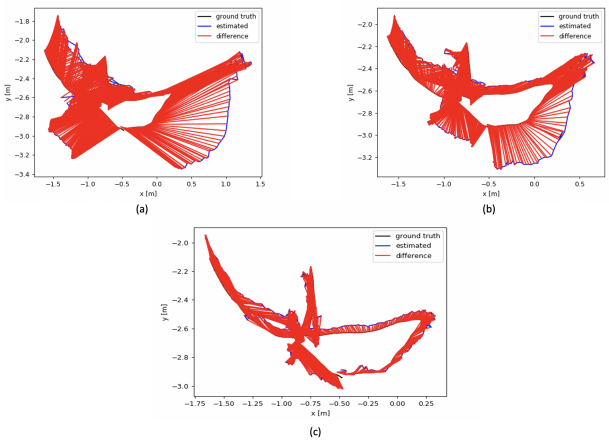

Figure 10: Absolute Trajectory Error (ATE): (a) With normal SLAM System. (b) With potential dynamic object elimination. (c) Our SLAM system with dynamic object elimination.

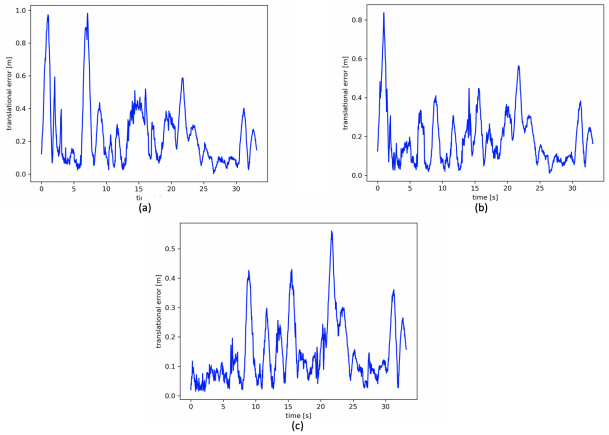

Figure 11: Relative Pose Error (RPE): (a) With normal SLAM System. (b) With potential dynamic object elimination. (c) Our SLAM system with dynamic object elimination.

For Visual SLAM, there are two main evaluation criteria: Absolute Trajectory Error (ATE) and Relative Pose Error (RPE). ATE estimates the difference between the real coordinates and the estimated coordinates as presented in Fig 10 and RPE represents the local accuracy of the measured trajec-

Table 1: COMPARISON OF ABSOLUTE TRAJECTORY ERRORS (RMSE)

TUM Dataset RGB-D sequences	Normal SLAM	With YOLO	With our method
fr3 walking half-sphere	0.76m	0.33m	0.17m

Table 2: COMPARISON OF RELATIVE POSE ERROR (RMSE)

TUM Dataset RGB-D sequences	Normal SLAM	With YOLO	With our method
fr3 walking half-sphere	25.85 deg	25.33 deg	25.21 deg

tory within a certain time interval as presented in Fig 11. The translational error can be estimated using ATE but the to estimate the rotational error we need to take RPE into account.

When we have tested our SLAM system on the freiburg3 walking halfsphere sequence of TUM Dataset, we got an Absolute Translational Error (RMSE) of around 0.17m and Relative Pose Error of 25.21 deg. Our result depicts that our method can deal with the high-dynamic scenarios very effectively and the accuracy of our RGB-D SLAM system is increased by 60 %. We have implemented our solution in python using openCV modules (Gunnar-Farneback dense optical flow, GFTT Detector, GMM), pyTorch (YOLOv3-tiny) and RGBD-PTAM.

Moreover, our solution is very robust and efficient as we have used light weight Neural Networks (YOLOv3 tiny), GMM on depth image which runs fast and give accurate results and the dense optical flow which is a bit slower than the sparse optical flow but gives high accuracy in dynamic object detection. Over all, our solution runs quite good on a CPU but will work much faster on GPU (like the Nvidia Jetson Board on our mobile robot). Hence our solution is robust and lightweight and is perfect for mobile robot navigation in dynamic indoor environment.

8 Conclusions

We have proposed a robust method to distinguish and eliminate dynamic feature points in this paper which makes the visual SLAM system robust and more accurate, so that, it can be implemented on mobile robots for navigation in dynamic indoor environments. The proposed method can be divided into two main parts, one being the initial camera motion estimation by eliminating features points from potential dynamic objects and other being the final camera pose estimation using optical flow method and the initial camera motion estimation. We have tested our SLAM system on TUM dataset and we got nice results. The SLAM system is robust and our method is light-weight which is why it is perfect to run on embedded boards. However, there are some limitations which we need to take care especially in terms of accuracy in highly dynamic environments.

9 Future Works

Here, we have suggested some future works regarding our SLAM system to make it more efficient.

- Adaptive threshold in optical flow which will help to eliminate dynamic feature points more efficiently and avoid eliminating static feature points.
- Due to the pandemic situation, our proposed solution couldn't be tested on our mobile robot (consisting of Intel RealSense D435 as the RGB-D camera and Nvidia Jetson board as GPU) which is at the GIPSA Labs. In future, we want to test our solution on the mobile robot in dynamic indoor environment.
- Lastly, in our solution we have used Dense optical flow, instead we could have used sparse optical flow to make it much faster as it will reduce the computation time which will speed up the system and we can actually compare between the accuracy and robustness.

Acknowledgments

This paper and the research behind it would not have been possible without the exceptional support of my supervisors, Michele Rombaut and Bruce Canovas. I would also like to thank Denis Pellerin, Serge Olympieff and Amaury Nègre for their constant support.

References

[Bescos and Neira, 2018] Fácil JM. Civera Javier Bescos, Berta and José Neira. DynaSLAM: Tracking, mapping and inpainting in dynamic environments. *IEEE RA-L*, 2018.

[Bouwmans *et al.*, 2008] Thierry Bouwmans, Fida El Baf, and Bertrand Vachon. Background Modeling using Mixture of Gaussians for Foreground Detection - A Survey. *Recent Patents on Computer Science*, 1(3):219–237, November 2008.

[Cheng *et al.*, 2019] Jiyu Cheng, Yuxiang Sun, and Max Q.-H. Meng. Improving monocular visual slam in dynamic environments: an optical-flow-based approach. *Advanced Robotics*, 33(12):576–589, 2019.

[Dalal and Triggs, 2005] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[Dar-Shyang Lee, 2005] Dar-Shyang Lee. Effective gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):827–832, 2005.

[Dempster *et al.*, 1977] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.

[Farnebäck, 2003] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In Josef Bigun and Tomas Gustavsson, editors, *Image Analysis*, pages

363–370, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[He *et al.*, 2017] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[J *et al.*, 2019] Cho J, Jung Y, Kim DS, Lee S, and Jung Y. Moving object detection based on optical flow estimation and a gaussian mixture model for advanced driver assistance systems. *Sensors (Basel)*, 2019.

[Jianbo Shi and Tomasi, 1994] Jianbo Shi and Tomasi. Good features to track. In *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[Lin *et al.*, 2014] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[Lin *et al.*, 2017] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2017.

[Liu *et al.*, 2016] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. *Lecture Notes in Computer Science*, page 21–37, 2016.

[Liu *et al.*, 2019] H. Liu, G. Liu, G. Tian, S. Xin, and Z. Ji. Visual slam based on dynamic object removal. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 596–601, 2019.

[Lucas and Kanade, 1981] Bruce Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision (ijcai). volume 81, 04 1981.

[Mur-Artal and Tardós, 2015] Montiel J. M. M. Mur-Artal, Raúl and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[Mur-Artal and Tardós, 2017] Raúl Mur-Artal and Juan D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017.

[Pire *et al.*, 2017] Taihú Pire, Thomas Fischer, Gastón Castro, Pablo De Cristóforis, Javier Civera, and Julio Jacobo Berlles. S-ptam: Stereo parallel tracking and mapping. *Robotics and Autonomous Systems*, 93:27–42, 2017.

[Redmon and Farhadi, 2018] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. https://arxiv.org/abs/1804.02767, 2018.

[Ren *et al.*, 2017] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[Shao-Wen Yang and Wang, 2009] Shao-Wen Yang and C. Wang. Multiple-model ransac for ego-motion estimation in highly dynamic environments. In *2009 IEEE International Conference on Robotics and Automation*, pages 3531–3538, 2009.

[Sheikh and Shah, 2005] Y. Sheikh and M. Shah. Bayesian modeling of dynamic scenes for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1778–1792, 2005.

[Stauffer and Grimson, 1999] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 2, pages 246–252 Vol. 2, 1999.

[Sturm *et al.*, 2012] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580, 2012.

[Sun *et al.*, 2018] Yuxiang Sun, Ming Liu, and Max Q.-H. Meng. Motion removal for reliable rgb-d slam in dynamic environments. *Robotics and Autonomous Systems*, 108:115 – 128, 2018.

[Zhang *et al.*, 2018] L. Zhang, L. Wei, P. Shen, W. Wei, G. Zhu, and J. Song. Semantic slam based on object detection and improved octomap. *IEEE Access*, 6:75545–75559, 2018.