Pseudo Depth Maps for RGB-D SLAM

Yue Zhang
School of Advanced Technology
Xi'an Jiaotong-Liverpool University
Suzhou 215123, China
Yue.Zhang2103@student.xjtlu.edu.cn

Taoyu Wu
School of Advanced Technology
Xi'an Jiaotong-Liverpool University
Suzhou 215123, China
Taoyu.Wu21@student.xjtlu.edu.cn

Haocheng Zhao
Artificial Intelligence Department,
IDPT, JITRI
Wuxi 214000, China
Haocheng.Zhao@liverpool.ac.uk

Shuang Du
Research and Development Department
Suzhou Inteleizhen Intelligent
Technology Co., Ltd
Suzhou 215123, China
dushuang@irobotop.com

Limin Yu*
Department of Communications and
Networking
Xi'an Jiaotong-Liverpool University
Suzhou 215123, China
limin.yu@xjtlu.edu.cn

Xinheng Wang*
Department of Mechatronics and
Robotics
Xi'an Jiaotong-Liverpool University
Suzhou 215123, China
xinheng.wang@xjtlu.edu.cn

2022 International Conference on High Performance Big Data and Intelligent Systems (HDIS) | 978-1-6654-9144-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/HDIS56859.2022.9991394

***Abstract*—In visual SLAM, RGB-D cameras can actively obtain pixel depth information but are expensive and unsuitable for outdoor use. In contrast, monocular depth estimation is relatively inexpensive and more readily available. There have been some systems that fuse monocular depth estimation with SLAM algorithms, but no method that can directly generate depth maps has been tested with experiments. We propose that using monocular depth estimation results can generate 16-bit pseudo depth maps, which can be combined with monocular images as pseudo RGB-D. The relevant camera parameters can be configured according to the general RGB-D camera SLAM requirements. Experiments with Monodepth2 and ORB-SLAM3 on the KITTI dataset demonstrate that pseudo RGB-D can achieve ~~as~~ satisfactory results in SLAM as in stereo computation. The research paves the way for researchers can quickly examine monocular depth estimation results in more SLAM frameworks, reducing the cost of testing new depth monocular estimation frameworks.**

Keywords*—pseudo depth maps; pseudo RGB-D; SLAM; depth estimation

I. Introduction

Simultaneous Localization and Mapping (SLAM) is a research technique for localization and mapping that performs an important role in the navigation and localization of mobile robots. Visual SLAM consists of four parts: Visual Odometry (VO), Optimization, Loop Closing, and mapping [1]. In VO, the estimation of camera pose is mainly accomplished, while monocular VO suffers from scale ambiguity (scale ambiguity) due to the lack of depth information in the input [2], [3]. The problem is usually solved by combining the Inertial Measurement Unit (IMU). Still, with the gradual maturation of deep learning frameworks [4], unsupervised monocular depth estimation methods have gradually replaced traditional monocular depth estimation methods [5], [6].

In recent years, many SLAM frameworks that combine monocular depth estimation have emerged, such as UnDeepVO and D3VO [7], [8]. However, their standard practice is to integrate monocular depth frameworks directly with VO, which can lead to the inability to quickly and efficiently exploit the depth information generated by new monocular depth estimation frameworks in the SLAM module when they become available [8]. In deep learning, the general approach uses Pytorch for model training and libtorch for model deployment. However, not all operations and methods are supported by TorchScript, although tools such as tracing scripting are officially provided by torch for conversion [9], [10]. So it wastes conversion and deployment time when combined with the slam framework to test monocular depth estimation. To address this issue, we implement a transformation of the output of the deep learning framework into a pseudo depth map and input the transformation results into the SLAM framework.

In this paper, we propose a method to produce pseudo RGB- D dataset that can reduce the cost of testing novel monocular depth estimation frameworks and use the Monodepth2 [11] and ORB-SLAM3 [12] models and the KITTI dataset [13] to demonstrate the feasibility of pseudo RGB-D. Our main contributions and innovations are as follows:

1. The depth prediction result is turned into a 16-bit pseudo depth map by the conversion algorithm, which then saves the depth information of the image.

2. Pseudo RGB-D using a combination of generated pseudo depth maps and monocular maps can be run directly in slam without needing a python to C++ pre-trained model port.

3. A scheme for modification of scale factors and camera baseline parameters for RGB-D cameras is provided for modification in experiments.

The paper is organized as follows: Section II contains related work. Section III describes our method of generating the depth map and the setting of the relevant camera parameters. Section IV presents our experiment and results. Section V presents limitations and future work to our approach.

II. Related Work

A. Vision sensors

The mainstream vision sensors are Monocular, Stereo, and RGB-D. Monocular cameras have a scale uncertainty quality that prevents a correct measurement of the scale of the object in a single image, so depth information cannot be obtained in a single image by monocular cameras [2], [14]. In contrast, Stereo cameras benefit from the imaging of two cameras and can estimate depth information based on the parallax principle, which can improve VO results to a large extent [15]. However, Stereo requires high hardware arithmetic power (GPU or FPGA). To reduce the need for arithmetic power, RGB-D cameras can be used to accomplish the acquisition of depth information. However, the imaging principle of RGB-D cameras, which can be affected by daylight or the low resolution of depth images, has limitations in application scenarios [16].

* Corresponding author.

B. Monocular Depth Estimation

With the continuous development of deep learning in recent years, the accuracy, computational efficiency, and output image resolution of monocular depth estimation have been significantly improved [4], [17]. Two CNN-based networks (monocular depth estimation and multi-view pose estimation) are used for unsupervised training and end-to-end learning methods, and ultimately this framework achieves estimation of depth and camera pose [18]. The advantage of this framework is that no stereo camera is required, and the pose estimation results are comparable to ORB-SLAM2 based on the evaluation of KITTI.

In self-supervised monocular depth estimation, PackNet uses 3d convolution without a pooling layer to address the resolution degradation caused by the traditional coding process. Although the dual network structure achieves better depth estimation with lower computational effort [19], Monodepth2 uses auto-mask loss to significantly improve the accuracy of depth estimation [11]. After comparing the above-mentioned self-supervised monocular estimation methods, we chose Monodepth2, which improves the minimum projection error, full-resolution multi-scale sampling, and auto-masking loss to construct a self-supervised monocular depth estimation scheme, Monodepth2 [11], [18]. Several loss functions proposed by Monoedepth2 have been borrowed by almost all subsequent methods that employ video sequences [20], [21].

Monodepth2 was evaluated under the KITTI based on Eigen et al. data after segmentation, and the Monodepth2 monocular method outperformed all existing self-supervised monocular estimation methods for SOTA [22].

C. SLAM Framework

The first monocular SLAM approach was MonoSLAM [23], which used the extended Kalman filter (EKF) to track Shi-Tomasi corner points of an image through correlation-guided search [2]. Parallel Tracking and Mapping (PTAM) [24] then proposed and implemented parallelization of tracking and map building and proposed the keyframes mechanism. PTAM pioneered nonlinear optimization as a back-end solution instead of a filter back-end solution.

Based on these ideas, ORB-SLAM [12], [25], [26] was computed using ORB features whose descriptors provide short- and medium-term data associations, thus establishing the complexity of co-view constraint tracking and graph building. The DBoW2 [27] bag-of-words approach was used to establish long-term data associations for relocation. In addition, ORB-SLAM3 has rich support for sensors, monocular, stereo, and RGB-D cameras can be used in this system. Due to the multi-map system in ORB-SLAM3, its accuracy and robustness are greatly improved. Based on these features of ORB-SLAM3, this framework was chosen for testing pseudo depth maps in this paper.

III. METHOD

This paper uses Monodepth2 for monocular depth estimation operations. Monodepth2 uses a combination of depth estimation and pose estimation networks to predict the depth in each image frame.

The training part of the whole algorithm does not need to label the dataset in advance. It predicts the depth results of monocular images directly by training an architecture built on a self-supervised loss function on a sequence of moving images (both monocular and stereo). The camera parameters are learned directly during the learning process, and no further input parameters are required for the subsequent conversion to depth-valued output.

A. Producing 16-bit Pseudo Depth Maps

Prediction results for monocular depth estimation are saved as a matrix according to the size of the image, with the predicted depth value stored at the location of each pixel point. The more bits in the image, the more colors are available leading to a higher color representation accuracy. The depth maps obtained from the RGB-D cameras currently in use on the market are generally 16-bit and 16-bit depth maps can express depth results for integer values from 0 to $2^{16}-1$. To ensure accurate transmission of the values, the maximum value is set at 65.5m, which is the maximum depth (d_{max}) that can be expressed in 16 bits, and the predicted exceeded results are also defined as a maximum value.

The depth values of the image are first normalized between 0 and 1 in the manner shown in Equation 1 where d_{ij} is the predicted depth value for each image pixel and D_{ij} is the value for each pixel after normalization.

$$D_{ij} = \begin{cases} \dfrac{d_{ij}}{d_{max}}, & d_{ij} < d_{max} \\ 1, & d_{max} \le d_{ij} \end{cases} \quad (1)$$

The data values are then mapped to the interval range and converted into three dimensions using the image's custom color bar to store the location and depth information of the pixel points separately and saved as a matrix pattern of uint16. Equation 2 shows the specific calculation and D_{ij}^{16bit} is the new matrix obtained after the transformation. This allows us to convert the predicted floating point numbers in meters to discrete integers for display in the depth image.

$$D_{ij}^{16bit} = \left| D_{ij} \times \left(2^{16}-1\right) \right| \quad (2)$$

The first three results for the first plot of sequence 07 in KITTI, the matrix output predicted by Monodepth2 (raw output), and the matrix results after processing by Equations 1 and 2 (16-bit output) are shown in Table I. The original output matrix is converted to an integer matrix (8-bit output) that can be saved for image visualization. The original matrix is in meters, and in the 8-bit result, we can see that the second and third data, which differ by 11 cm, are nevertheless expressed in the same way.

TABLE I. COMPARE OUTPUT ARRAYS

Matrix type	***Value 1***	***Value 2***	***Value 3***
original output	8.155626	8.230361	8.339033
8-bit output	31	32	32
16-bit output	8159	8234	8343

The two depth map plots shown in Figure 1 visualize the 8-bit output and 16-bit output matrices, respectively. The depth map obtained from the raw output resulted in some obvious missing depth values, marked using the red boxes.

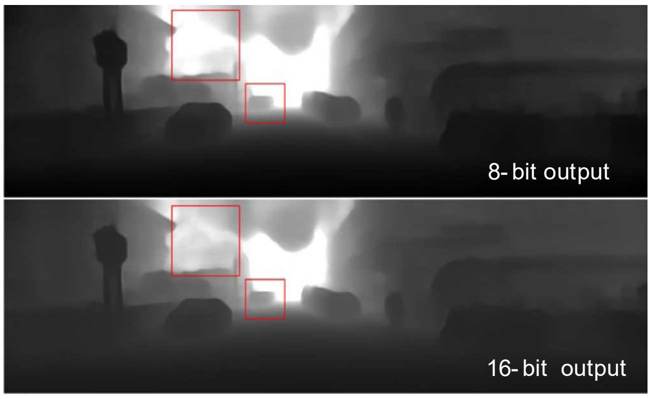

Fig. 1. original and 16-bit pseudo depth maps

The dataset consisting of the 16-bit pseudo depth maps and the images obtained by the monocular camera is subsequently called pseudo RGB-D.

B. Derivation of Related parameters

In the original RGB-D camera, the RGB images and the depth maps are not necessarily read in the same period, so a text needed to be created for each RGB image and depth map to match. Similarly, matching text is required for pseudo RGB-D to allow the SLAM algorithm to successfully achieve the two image information. An index can be used instead for datasets that do not have a time frame.

The depth and color cameras in the original RGB-D camera need to transform the position of the pixel values that would exist between the actual 3D coordinate data and the projected image. The rotation matrix (R) and translation matrix (T) between the two cameras are shown in Equations 3 and 4, with the subscript rgb being the color camera parameters and ir being the depth camera parameters.

$$R = R_{rgb} R_{ir}^{-1} \tag{3}$$

$$T = T_{rgb} - R_{rgb} R_{ir}^{-1} T_{ir} = T_{rgb} - RT_{ir} \tag{4}$$

For the pseudo RGB-D depth map we have generated, the two cameras are approximately coincident. Therefore, the inner parameters of the cameras can be shared and the distance between the two cameras set to a smaller value.

In addition to the depth value projected on the image, another critical parameter is the depth factor. For example, a DepthMapFactor of 5000 in TUM's RGB-D dataset[28] would represent 5000 units of 1 meter in the depth map. The depth factor (DF) for pseudo RGB-D could be obtained directly using Equation 5.

$$DF = \frac{2^{16} - 1}{d_{max}} \tag{5}$$

One more parameter can be adjusted in ORB-SLAM3, which is ThDepth, together with stereo.b determines the effective depth (ED) of depth points, calculated as shown in Equation 6.

$$ED = Stereo.b \times ThDepth \tag{6}$$

The depth map and associated camera parameters needed for the RGB-D camera in SLAM are set up and can be taken directly into SLAM for experimentation.

IV. EXPERIMENT

A. Setting of the Experiments

KITTI's sequences 07 and 10 are used to evaluate our solution, and Ubuntu 20.04 is the experimental system. Python3.8.3, pytorch1.11, and the CPU i7-9750H are also included in our configuration and no GPU environment.

As is customary in this field, Relative Position Error (RPE) and Absolute Trajectory Error (ATE) are used as accuracy measures.

The RPE calculates the change in attitude between the same two timestamps, so in Equation 7~~,~~ Δ is a time interval, Q is the ground truth pose, and P is the estimated pose. They are a 4*4 matrix including rotation error and translation error. The difference between the real and estimated poses is calculated at the same time interval after alignment with the timestamp. Then the difference is made to obtain the relative positional error, which can be used to estimate the drift of the system.

$$E_i := \left(Q_i^{-1} Q_{i+\Delta} \right)^{-1} \left(P_i^{-1} P_{i+\Delta} \right) \tag{7}$$

ATE can directly calculate the difference between the camera ground truth (GT) value and the SLAM system's estimated value. The absolute trajectory error is the direct difference between the estimated and true poses and supplies an intuitive representation of the trajectory's algorithm accuracy and global consistency. In Equation 8 of the ATE, the same Q and P represent the true and estimated ground poses, respectively. S is the transformation matrix from the estimated to the true pose, where the transformation matrix is $S \in SE(3)$ based on the stereo and RGB-D scale consistency. The similar transformation matrix $S \in Sim(3)$ is used for monocular due to scale uncertainty.

$$F_i := Q_i^{-1} S P_i \tag{8}$$

The Root Mean Square Error (RMSE) is usually evaluated on all-time indices for both RPE and ATE for the calculation of the translational component in the assessment~~,~~ so that there is less impact on the outliers.

As there are no time frames in the original KITTI, they are converted to tum format, and the output of all camera KITTI is set to be in tum format as well. The estimated trajectories are then aligned to the ground truth using time frames.

B. Image Processing

The matrix results obtained using Monodepth2 predictions are normalized using equations before being transferred to a 16-bit depth map matrix, and pseudo depth maps are drawn. Monodepth2 was pre-trained using KITTI 00-08 as the training set, and the results are shown here using 07 of these and 10 as the test set. The effect of the original image and the pseudo depth maps are shown in Fig.2 and Fig.3, respectively.

Fig. 2. Raw RGB image

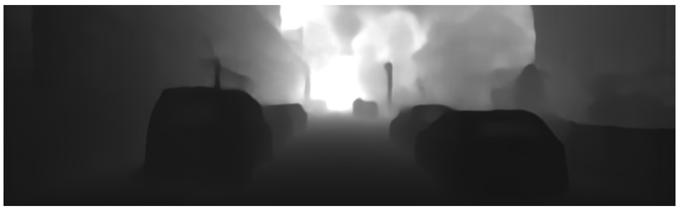

Fig. 3. Pseudo depth map

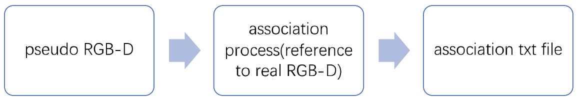

Fig. 4. process of setting association text

To read from the RGB-D camera in ORB-SLAM3, it is also necessary to write the RGB's and depth maps' paths and timestamps using the association's text. Thus, another associate's text is generated simultaneously. Here, the process is according to the original TUM format, as shown in Figure 4. And modify the RGB-D camera-related parameters, here we set DepthMapFactor to 1000 and Thdepth to 35.

Finally, the processed data and the text of the associated parameters are placed according to the general RGB-D camera format to facilitate direct operation in the SLAM algorithm.

C. Results

Based on the information about the cameras in the KITTI, the localization and building of the three cameras in ORBSLAM3, monocular, stereo, and the pseudo RGB-D, are tested. The RPE results are shown in Table II, where the pseudo RGB-D can be achieved the same accuracy as stereo and performs better. It is demonstrated that the results of the pseudo RGB-D in SLAM do not show large drifts.

TABLE II. RPE W.R.T. TRANSLATION PART (M)

Methods	**Relative Position Error (RPE)**	
	Sequence 07	***Sequence 10***
	RMSE(m)	***RMSE(m)***
ORB-SLAM3 Mono	0.099254	0.158076
ORB-SLAM3 Stereo	0.026493	0.032889
ORB-SLAM3 pseudo RGB-D(ours)	**0.019553**	**0.028656**

The ATE results are shown in Table III, where the RMSE results between stereo and pseudo RGB-D in Sequence 07 and 10 are both less than 3 meters, which is better than the monocular purpose results. It is demonstrated that the pseudo RGB-D results on ORB-SLAM3 also do not have large translation errors.

TABLE III. ATE W.R.T. TRANSLATION PART (M)

Methods	**Absolute Trajectory Error (ATE)**	
	Sequence 07	***Sequence 10***
	RMSE(m)	***RMSE(m)***
ORB-SLAM3 Mono	2.345489	8.102078
ORB-SLAM3 Stereo	**0.538103**	**1.927677**
ORB-SLAM3 pseudo RGB-D(ours)	0.691126	2.202083

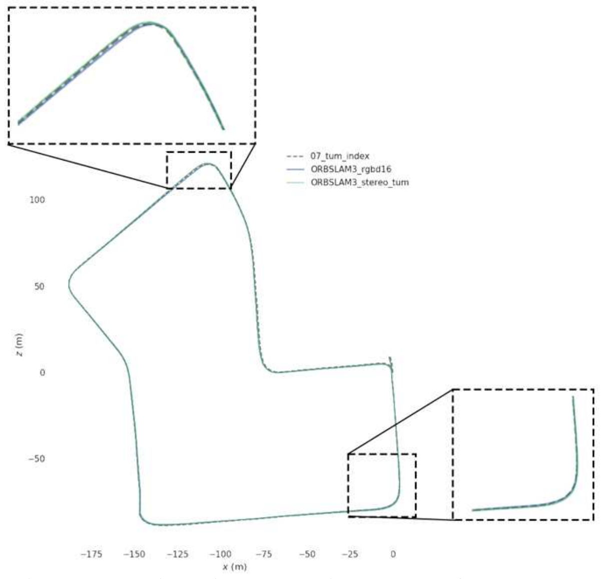

Fig. 5. stereo and pseudo RGB-D trajectory maps of Sequence 07

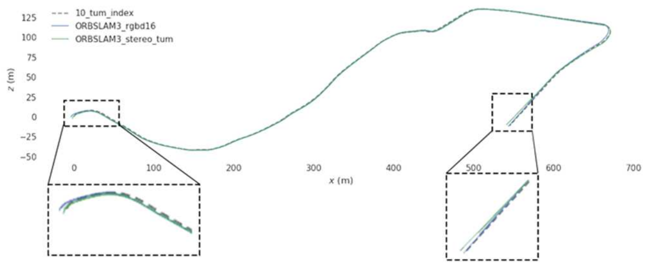

Fig. 6. stereo and pseudo RGB-D trajectory maps of Sequence 10

Figure 5 compares the trajectory results obtained from the stereo and pseudo RGB-D 16-bit depth images. The dashed parts of both plots are the GT of the original camera trajectories from the KITTI sequence07, the green line is the estimated trajectory from the stereo camera, and the blue line is the estimated trajectory from the pseudo RGB-D. They both fit the GT well, but in some corners, the pseudo RGB-D performs closer to the true value.

Figure 6 shows KITTI sequence10 in the same content format as Figure 5, it can be seen that the part where the paths turn is where the two results are most likely to deviate compared to GT and that the pseudo RGB-D could fit the true values better.

V. CONCLUSION

This paper proposes to use only monocular cameras and generate 16-bit pseudo depth maps using monocular depth estimation results and monocular camera data as pseudo RGB-D. We also offer a way to set up the relevant camera parameters. After experiments using KITTI, it is found that better relative bit pose results can be achieved in SLAM than with stereo cameras. This is an efficient way to generate depth maps using only monocular cameras, allowing researchers to reuse the pseudo depth maps in SLAM before fusing monocular estimation and SLAM algorithms, thus increasing the efficiency of the experiments.

However, when using indoor data, it was found that the indoor dataset had limitations in its performance on Monodepth2, with trajectory drift, which will require separate retraining of the model for monocular estimation in the next step. In the future, we also hope to experiment with pseudo depth mapping schemes that can be applied to dense buildings and navigation systems and develop more complete SLAM systems.

Acknowledgement

We are grateful to Tongpo Zhang and Sihao Li for providing guidance and assistance in academic writing and the combined efforts of Huajun Yuan, Hanlin Li, and Yuhan Zhang from our joint project team.

The authors would like to thank Suzhou Inteleizhen Intelligent Technology Co. Ltd and Xi'an Jiaotong-Liverpool University for their financial support to conduct this research by Key Program Special Fund in XJTLU under project KSF-E-64, XJTLU Research Development Fund under projects RDF-19-01-14 and RDF-20-01-15, and the National Natural Science Foundation of China (NSFC) under grant 52175030.

References

[1] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual SLAM algorithms: A survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, 2017, doi: 10.1186/s41074-017-0027-2.

[2] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans Pattern Anal Mach Intell*, vol. 29, no. 6, 2007, doi: 10.1109/TPAMI.2007.1049.

[3] B. Huang, J. Zhao, and J. Liu, "A Survey of Simultaneous Localization and Mapping with an Envision in 6G Wireless Networks," 2020.

[4] F. Tosi, F. Aleotti, M. Poggi, and S. Mattoccia, "Learning monocular depth estimation infusing traditional stereo knowledge," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019, vol. 2019-June. doi: 10.1109/CVPR.2019.01003.

[5] R. Mahjourian, M. Wicke, and A. Angelova, "Unsupervised Learning of Depth and Ego-Motion from Monocular Video Using 3D Geometric Constraints," 2018. doi: 10.1109/CVPR.2018.00594.

[6] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey, "Learning Depth from Monocular Videos Using Direct Methods," 2018. doi: 10.1109/CVPR.2018.00216.

[7] R. Li, S. Wang, Z. Long, and D. Gu, "UnDeepVO: Monocular Visual Odometry Through Unsupervised Deep Learning," 2018. doi: 10.1109/ICRA.2018.8461251.

[8] N. Yang, R. Wang, J. Stückler, and D. Cremers, "Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 11212 LNCS. doi: 10.1007/978-3-030-01237-3_50.

[9] J. Lee, S. Lee, and J. G. Ko, "Mobile YOLACT: Toward Lightweight Instance Segmentation for Mobile Devices," in *International Conference on ICT Convergence*, 2021, vol. 2021-October. doi: 10.1109/ICTC52510.2021.9621125.

[10] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, Dec. 2017, vol. 2017-October, pp. 2980–2988. doi: 10.1109/ICCV.2017.322.

[11] C. Godard, O. mac Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, vol. 2019-October. doi: 10.1109/ICCV.2019.00393.

[12] 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 Multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021, doi: 10.1109/TRO.2021.3075644.

[13] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *International Journal of Robotics Research*, vol. 32, no. 11, 2013, doi: 10.1177/0278364913491297.

[14] H. Liu, G. Zhang, and H. Bao, "A survey of monocular simultaneous localization and mapping," *Jisuanji Fuzhu Sheji Yu Tuxingxue Xuebao/Journal of Computer-Aided Design and Computer Graphics*, vol. 28, no. 6, 2016.

[15] D. D. Nguyen, A. Elouardi, S. A. Rodriguez Florez, and S. Bouaziz, "HOOFR SLAM System: An Embedded Vision SLAM Algorithm and Its Hardware-Software Mapping-Based Intelligent Vehicles Applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 11, 2019, doi: 10.1109/TITS.2018.2881556.

[16] S. Zhang, L. Zheng, and W. Tao, "Survey and Evaluation of RGB-D SLAM," *IEEE Access*, vol. 9, 2021, doi: 10.1109/ACCESS.2021.3053188.

[17] C. Q. Zhao, Q. Y. Sun, C. Z. Zhang, Y. Tang, and F. Qian, "Monocular depth estimation based on deep learning: An overview," *Science China Technological Sciences*, vol. 63, no. 9. 2020. doi: 10.1007/s11431-020-1582-8.

[18] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-January. doi: 10.1109/CVPR.2017.700.

[19] V. G. Rares, Ambrus, S. Pillai, A. Raventos, and A. Gaidon, "3D packing for self-supervised monocular depth estimation," 2020. doi: 10.1109/CVPR42600.2020.00256.

[20] L. Wang, J. Zhang, O. Wang, Z. Lin, and H. Lu, "SDC-Depth: Semantic Divide-And-Conquer Network for Monocular Depth Estimation," 2020. doi: 10.1109/CVPR42600.2020.00062.

[21] S. Pillai, R. Ambruş, and A. Gaidon, "SuperDepth: Self-supervised, super-resolved monocular depth estimation," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2019, vol. 2019-May. doi: 10.1109/ICRA.2019.8793621.

[22] S. B. Busam, "Predicting depth , surface normals and semantic labels with a common multi-scale convolutional architecture (arXiv 2014) Seminar Recent Trends in 3D Computer Vision," *ICCV*, no. arXiv, 2015.

[23] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proceedings of the IEEE International Conference on Computer Vision*, 2003, vol. 2. doi: 10.1109/iccv.2003.1238654.

[24] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," 2007. doi: 10.1109/ISMAR.2007.4538852.

[25] 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, 2015, doi: 10.1109/TRO.2015.2463671.

[26] 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, doi: 10.1109/TRO.2017.2705103.

[27] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, 2012, doi: 10.1109/TRO.2012.2197158.

[28] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, *A Benchmark for the Evaluation of RGB-D SLAM Systems*. 2012. doi: 10.1109/IROS.2012.6385773.