

An experimental comparison of visual SLAM systems

Aayushi Gautam
*Dept. of Computer Science and Engg.
M.B.M Engineering College
Jodhpur, India
aayushi363@gmail.com*

Suraj Mahangade
*Electronics And Telecommunication Engg
Fr. Conceicao Rodrigues Institute of Technology
Mumbai, India
surajmahangade99@gmail.com*

Vishal Indal Gupta
*Dept. of Computer Science and Engg.
Indian Institute of Technology Bombay
Mumbai, India
vishal.gupta_1997@iitb.ac.in*

Rishikesh Madan
*Dept. of Computer Science and Engg.
Indian Institute of Technology Bombay
Mumbai, India
rishi.madan@iitb.ac.in*

Kavi Arya
*Dept. of Computer Science and Engg
Indian Institute of Technology Bombay
Mumbai, India
kavi@cse.iitb.ac.in*

Abstract—Monocular vSLAM (Visual Simultaneous Localisation and Mapping) is a navigation technique in which autonomous robots use visual information from a single camera to model a map of unknown environments while simultaneously localising themselves on that map. This paper explores prevalent algorithms' performance in a highly dynamic use case on a quadcopter where other sensors' data (viz. IMU, optical flow data) is not accessible. Three monocular vSLAM methods prevalent in the research community, Parallel Tracking and Mapping, ORB-SLAM and its enhancement ORB-SLAM2 (with 4585, 3936 and 2538 citations respectively) performed poorly in our use case. Better results were obtained using a Convolutional Neural Network based approach designed for ground-based vehicles. This paper presents an experimental comparison of these algorithms on metrics such as point cloud generation and localisation. Experiments were conducted physically and in the Gazebo simulator on the DJI Tello, a nano-quadcopter with an onboard camera, where we chose not to use the IMU data provided by the API.

Index Terms—Simultaneous Localisation and Mapping; AUV navigation; Perception robotics; AI and ML in robotics

I. INTRODUCTION

Visual SLAM (vSLAM) has been a topic of vigorous research in the Computer Vision and Robotics communities. SLAM techniques are used to build maps of unknown environments and localise the sensors on the map. As compared with radar, sonar and other devices that are used to find ranges, visual sensors such as laser range finders and cameras have good information acquisition. Hence, Visual SLAM systems are widely studied and used [1]. This technique has provided solutions to many real world problems and has applications such as Augmented Reality based visualisation, rover localisation and navigation in deep space exploration, indoor positioning and navigation, autonomous navigation in large scenes, computer vision based 3D modelling and

self-driving cars [2]. With the increase in popularity and wide applications of drones, these vSLAM techniques are implemented on them for application in real-world search and rescue scenarios. For this application on a drone, monocular cameras are used as a state estimation sensor because of their low mechanical complexity, high image acquisition speed, and low power consumption.

Feature extraction and Bundle Adjustment (BA) are the most used techniques in various Visual SLAM packages. Bundle Adjustment is the refining of 3-D coordinates which depict the points of a given set of images from various viewpoints, simultaneously with the geometry of the scene, parameters of relative motions and optical characteristics of the camera. To obtain reliable outcomes from real-time SLAM algorithms, the following key points are taken into account, along with Bundle Adjustment [3]:

- 1) Selection of corresponding observations of map points from the subset of key-frames, to avoid redundancy.
- 2) An initial estimation of key-frame point location for nonlinear optimisation and the ability to perform fast globalisation to close loops.

The very first implementation of BA in SLAM was by Klein and Murray, called Parallel Tracking and Mapping (PTAM) [4]. It is a simple method which effectively selects key-frames, matches them with feature point triangulation, localises the camera and re-localises it upon tracking failure. Although PTAM was a robust it lacks large loop closing, to solve such issues a new system was developed called ORB-SLAM by Ral Mur-Artal and Juan D. Tardós [1]. This system is based on the basic idea of PTAM with various other contributions, such as using ORB features for tracking, mapping and re-localisation. Oriented FAST and rotated BRIEF (ORB) features are robust features that can be used in computer vision tasks such as object recognition and 3D reconstruction briefly explained in [5]. Notably, as stated in

This work was funded by the Ministry of Education, Government of India, as part of the project e-Yantra (RD/0113-MHRD001-021).

[1] camera localisation accuracy is achieved by this approach better than other approaches which directly use pixel intensities instead of feature re-projection errors. This approach expands the versatility of PTAM to previously untraceable environments. Further, for more accuracy in mapping, this system was updated and called ORB-SLAM2. The system has an embedded place recognition module for re-localisation of the camera in case of tracking failure. It uses the same ORB features and pre-processing as the input to extract features at important key-point locations that show good steadiness to camera auto-gain and illumination changes. This system's main contribution was that it uses close and far stereo points for more accurate results. Along with three main threads for tracking, mapping, and loop closing, it is composed of an additional thread to perform full BA after a loop closure for more accurate mapping. Apart from Bundle Adjustment, which provides a good estimation of camera localisation, Vision-based depth reconstruction using Convolutional Neural Networks (CNNs) is also a widely used method. Andrey Bokovoy in [6] demonstrates that these systems are more accurate than other conventional methods that are used in the pipeline of monocular vSLAM and are fast enough to work in real time. For CNN inference, a Robot Operating System (ROS) node is implemented. For localisation and mapping, RTAB-Map is used [7].

In recent years, vSLAM research has seen many advancements in terms of precision and efficiency. In this paper we undertake a systematic comparison of the top four such methods, concentrating on their ability to generate the perfect map of the unknown environment. Furthermore, this paper discusses the working pipeline of PTAM, ORB-SLAM, ORB-SLAM2 and Vision Based Depth Reconstruction in sections II, III, IV and V respectively. Section VI, briefly explains the experiment and experimental conditions along with the major observations. Finally, section VII presents the conclusion based on our observations.

These results are useful in determining the relative strengths and limitations of the methods. These experiments were motivated by the lack of experimental confirmation of the performance of various methods. Many approaches did not give the expected results claimed by the developers, which makes it difficult to determine whether a method is successful if implemented under various conditions, mainly on drones, and which method might give the best result. While experimenting, we tried to keep the conditions constant.

II. PARALLEL TRACKING AND MAPPING (PTAM)

One of the earliest approaches developed for implementing visual SLAM was demonstrated by Klein and Murray in [4] with excellent feature tracking and key-point extraction methodology. This method can be summarised by the following points in the context of SLAM:

(i) Two parallel threads for tracking and mapping. Doing so makes the tracking probabilistically independent of the



Fig. 1. Map Initialization done in PTAM using 5 point stereo initialisation with the help of user.

map-making procedure and helps the mapping process reduce redundant video frames.

(ii) Bundle Adjustment, which is a key-frame based system, is used for Mapping. Though it is computationally expensive compared to other incremental mapping methods, it is fast and accurate, which makes it more reliable for real-time mapping.

(iii) The map is initialised from stereo pairs (5-Point Algorithm), which requires human intervention, i.e map initialisation is not automatic.

(iv) Require a large number of points, i.e key-frames, to model a map.

A. Tracking

The tracking thread receives the images from the monocular camera and maintains a real-time estimate of the camera's position relative to the modelled map. For every frame, the tracking system performs the following:

1) **Image procurement:** Images are captured by a DJI Tello Drone camera. The camera delivers 2592 x 1936 pixel frames at 30Hz. The tracking system constructs a four-level image pyramid and for each pyramid, a FAST-10 which is use to extract feature points and map objects as stated in [8] and a corner detector is run.

2) **Camera Pose, Patch Search and Pose update:** A decaying velocity model is used to estimate the camera pose.

To find a single map point in the current frame, a fixed range patch search is carried out around the point's predicted image location.

3) **Two stage coarse-to-fine tracking and failure recovery:** Patch search and pose update are done twice in order to increase the tracking system's durability against rapid camera accelerations. Coarse-to fine is a method which that divides the problem into stages and then solve them from coarse to fine [9]. Using same approach in tracking, in the first stage of coarse-to fine tracking, an initial search over 50 map points which appear at the highest level of the frame's image pyramid is done. A new measurement is done on this basis for further process. In the second stage, 1000 remaining visible image patches are re-projected into the image and again patch search is performed. After these two stages, the final frame pose is calculated.

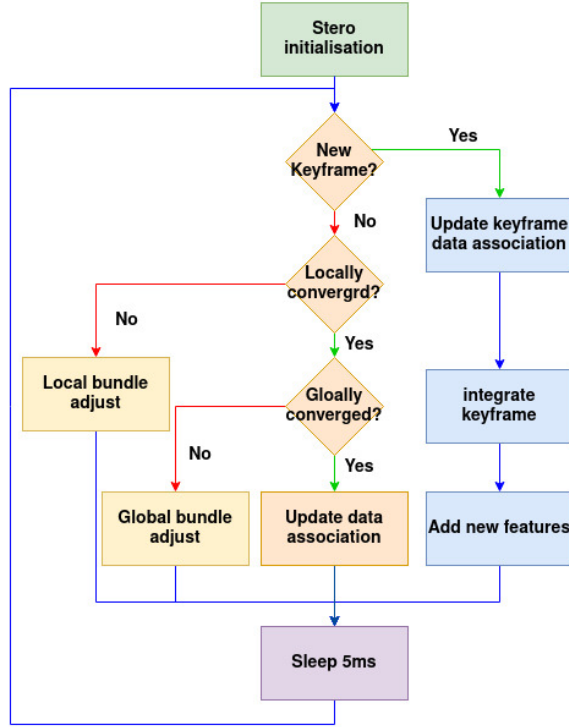


Fig. 2. Mapping thread flowchart. This thread runs in an infinite loop after initialisation

Although the tracking system is robust, then too, there is the possibility of tracking failure if the fractions of feature fall below a certain threshold value, so to avoid this tracking failure, a recovery system is employed along with the recovery method.

B. Mapping

There are two stages in map building (i) Creating an initial map by stereo initialisation technique as described in [10]. (ii) Expanding the initial map by addition of new key-frames by tracking system. Complete flow chart of the mapping thread is shown below:

1) **Map Initialisation:** The very first stage of the mapping thread is initialising the map. For this, a five-point-stereo initialisation method is employed which is briefly explained in [11]. This requires user intervention. The first user places the camera above the workspace that is to be tracked and presses a key. At this stage, the first key-frame of the system is stored on the map and 100 2D patch tracks are initialised. In the second stage, the user translates the camera slightly and again presses the key (Fig.1). Here, the 2D patches initialised in the previous stage are tracked and a second key-frame is derived.

2) **Key-frame insertion and epipolar search:** After map initialisation, we have only two frames, so a new key-frame is added every time the following conditions are fulfilled: (i)The tracking quality is good. (ii)If time exceeds twenty frames from last added key-frame. (iii) The distance between

the camera and the nearest keypoint already in map should be minimum.

A new map point that is to be added requires depth information that can not be derived from a single key-frame and that is why triangulation from another view is required. The closest key-frames from the already existing map are selected as the second view. Correspondence between the two views is established. If they both match, then a new map point is triangulated and inserted into the map.

III. ORB-SLAM AND ORB-SLAM2

ORB-SLAM demonstrated by Raúl Mur-Artal and Juan D. Tardós in [1] is based on the basic idea of PTAM. Along with the threads present in PTAM, it has one additional thread for loop closing. This system can be summarised as :

(i) Tracking Thread, whose main aim is to localise the camera i.e. initial pose estimation and map initialisation, decide when to insert a new frame.

(ii) Local Mapping Thread, processes the new frame inserted. Remove the redundant frames and apply local bundle adjustments.

(iii) Loop Closing Thread is in charge of detecting a loop and computes a similarity transformation that calculate the drift accumulated in the loop.

Although it is a very good development in the field of visual SLAM, it has some limitations and its accuracy can be improved. On further enhancement of ORB-SLAM a new system was developed called ORB-SLAM2.

A. Tracking

This section describes the various steps involve in the thread responsible for tracking tracking performed with every frame from the camera.

1) **ORB acquisition:** Same as image procurement in PTAM, here FAST corners are extracted. To ensure equal distribution each scale level is divided into grid to extract minimum 5 corners per cell. Here in place of path search, ORB descriptor which is a fast local feature detector, first presented by Rublee in [5], is used to find the correctly detected corners.

2) **Map Initialisation:** In ORB-SLAM, map initialisation is automatic. ORB features are extracted from the current frame and matched with the reference frame. If no matches are found, then the reference frame is reset. Then two models that are homography and fundamental matrix (FM) are computed in parallel. Simultaneous estimation of FM and homographies are done by employing compatibility constraint between them and briefly demonstrated by Chen and Suter in [12]. Models are selected on the basis of the type of scene. For a planar projection with low parallax, a homography model is computed. For a non-planar projection with enough parallax, a fundamental matrix is computed. Finally, perform a full Bundle Adjustment to refine the initial reconstruction [13].

3) **Initial Pose estimation:** A constant velocity motion model is used to predict the camera pose and perform the guided search over the map points that are observed in the last frame. If the camera pose has enough inliers, a guided search is again employed for more matches with the map points of the candidate frame [14].

4) **Track Local Map:** After the initial camera pose is estimated, the map can be forecasted into the frame and more map points can be searched for correspondence. Only the local maps are projected in order to reduce the complexity of large maps. It has a collection of key-frames K_1 , which share key-frames with current frame. And a collection of K_2 frames which are neighbours to K_1 in covisibility graph. In the current frame, both K_1 and K_2 are searched [4]. Following the search, the camera pose is optimised using all of the map points contained in the frame. Finally the redundant frames are culled and try to insert new key-frames as rapid as possible.

B. Local Mapping

Mapping thread is independent of tracking thread and hence running parallel with it. Following are the steps performed in local mapping every time with a new key-frame K_i .

1) **Key-frame insertion and Map point culling:** In order to retain the map points in the map, they must be traced in greater than 25% of the frames in which they are visible. Or if more than one key-frames have passed the map-points then it must appear in at least three key-frames. These conditions reduce the number of outlier in the map.

2) **New map point creation:** By triangulating ORB from the connected key-frames in the covisibility graph new map points are created. The parallax, reprojection error, and scale consistency are all checked in order to accept the new point positive depth.

3) **Local Bundle Adjustment and key-frame culling:** The currently processed key-frame is optimised by local BA. At the middle and end of the optimisation, outlier points marked are discarded. It is also try to detect redundant frames and delete them.

C. Loop Closing

In ORB-SLAM system, there is a separate thread added for detecting and closing loops. It searches for loops within every new keyframe and compute similarity transformations [1]. This thread again consists of several steps they are:

1) **Loop Detection:** To accept a candidate to be a loop, it should be detected at least three time consistently. If there are many places with similar appearances, there could be many loop candidates.

2) **Similarity transformation and loop fusion:** Computing transformation of loop help us to find out the error accumulated in the loop and also serve as geometrical validation of the loop. After the similarity transformation, next major step is loop correction. It involves the fusion of duplicate map points and new edges in the co-visibility graph to link the loop closure as described in [15].

D. Full Bundle Adjustment

This is the additional thread which is added in ORB-SLAM2 for more accuracy. Local Bundle Adjustment optimizes a set of covisible keyframes and all points seen in those frames. Full Bundle Adjustment is the special case of local BA where all keyframes and points in the map are optimized, except the origin keyframe [16]. Main contribution of this thread is to perform full BA after the pose-graph optimisation, so that the system is able to compute the optimal structure and motion solution.

IV. VISION-BASED DEPTH RECONSTRUCTION (TX2 _ FCNN)

Depth estimation is one of the important part of map modelling and path planning [17]. This method uses convolution neural networks to determine the depth reconstruction and further map modelling steps, which are briefly described in [18].

A. Architecture

Based on the limited computational resources, Fully-convolution Neural Network for depth reconstruction have different architectures.

Generally CNN model consists of an encoder and a decoder. Function of encoder is to extract the high level features and function of decoder is to generate depth maps from these features. Apart from these two, this model have improved some blocks of the network to make it more accurate and fast.

1) **Encoder:** This implementation chose ResNet50 [19], a versatile feature extractor, as an encoder. Although it is a deep layer network with many network blocks, it is fast enough for real-time depth reconstruction. It is denoted as a **Basic encoder** and the cropped version of this encoder is termed as **Lite Basic**.

2) **Decoder:** In this system, three decoders are employed in order to generate depth map from the features extracted by the encoders.

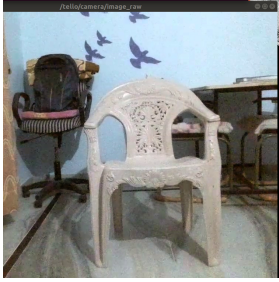


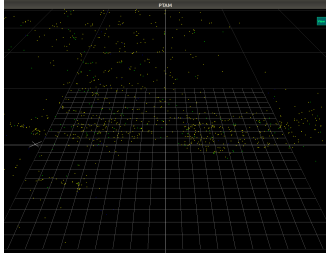

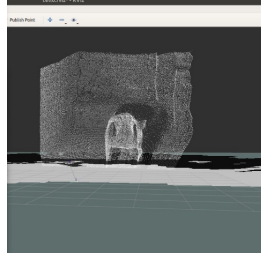
(i) Base line decoder, with 5 deconvolution blocks. Each block consists of deconvolution + batch normalisation + activation layer, which is explained by Romera, E., Alvarez, J. M., Bergasa, L. M. and Arroyo, R. in [20]. This decoder is denoted as **ReLU**.

(ii) A decoder that composed of the blocks that use upsampling followed by non-bottleneck convolution and deconvolution + batch normalisation + activation layer. This denoted as **Upsampling + nonbt**

(iii) Lastly, we use up-convolution decoder. Each block of this decoder composed of unpooling + convolution + batch normalisation + ReLU + Dropout. It is denoted as **interl**.

For projections from encoder layer to decoder layer we use **skip connections** denoted as **SC**. It produces highly sharpened depths on the object's edges but do not improve the model's accuracy.

TABLE I
POINT CLOUD DATA COMPARISON. THIS THE OUR OBSERVATION BASED ON THE EXPERIMENT CONDUCTED. THIS TABLE CLEARLY SHOW US THE PERFORMANCE OF DIFFERENT ALGORITHM FOR GENERATING A POINT CLOUD.

Algorithm	PTAM	ORB-SLAM2	CNN
Original Image			
Point Cloud			

B. Visual SLAM implementation

This model have been implemented over many different data sets like NYU Dataset v2, demonstrated in [21]. But we are interested in its implementation with a ROS node, as our main task is to perform tracking and mapping in real time using a robot.

For localisation and mapping RTAB-Map is used. Both CNN inference and RTAB-Map is run on NVidia Jetson TX2. The model of CNN used here composed of **Basic + SC** as encoder and **upsampling + nonbt** as decoder [6]. This combination runs the NVidia Jetson GPU with 16 FPS, and the vSLAM (RTAB-Map) operate the CPU. As these two nodes are independent there is not interference between the working of these two algorithms.

V. EXPERIMENTAL ANALYSIS AND OUTCOME

To compare all these four vSLAM systems, we performed an indoor experiment using a **Drone** in simulation as well as in real world, which is equipped with a Vision Positioning system and an onboard camera.

(i) **Experimental Setup:** A room with a cluttered environment and a chair is chosen for performing the experiment. The main reason behind choosing such an environment is that 2 of the techniques compared, i.e. PTAM and ORB-SLAM2, use keypoints for mapping and a cluttered environment will provide objects that will act as keypoints.

(ii) In the experiment basically two nodes are run. One is **Tello Driver**, to drive the Tello drone and other is the **vSLAM** node, for localisation and mapping.

(iii) Point cloud generated by each method is re-scaled using to find the measurement of an object in the image frame using Point Cloud Library.

A. Observations

After performing these experiments, we outlined some observations (Table I), for each algorithm used.

1) **PTAM:** (i) As PTAM is based on key frames tracking so it gives results in the environment which have many obstacles or objects.

(ii) The map is densely initialised from a stereo pair and for that we require a clear image but if the image stream from hardware is very pixelated leading to no stereo-initialisation.

(iii) The point cloud generated by PTAM is very sparse hence not able to get the exact object.

(iv) Due to the local minimum problem in BA, getting the global optimal of the map and camera poses in large environments is difficult.

2) **ORB-SLAM:** (i) Difficult to initialise in Gazebo simulator environment.

(ii) Points with sufficient parallax are not included, this leads to insufficient information about camera rotation.

(iii) Often, erroneous points situated extremely close to the camera appear in the point cloud. These points affected the cost map generation, eventually affecting the path planning, giving collision errors in move-base.

3) **ORB-SLAM2:** (i) Same as ORB-SLAM. Just a new thread added for full BA.

(ii) Much faster than ORB-SLAM and points with sufficient parallax are taken into consideration and hence giving information about orientation of the camera.

As PTAM ORB-SLAM and ORB-SLAM2, all are based on single idea of feature tracking using Key frames. We observe that PTAM keeps on inserting key frames hence leading to redundancy. While ORB-Slam and ORB-Slam 2 delete the duplicate key frames leading to less redundant data. PTAM

ORB-SLAM and ORB-SLAM2 struggled with bent edges and pure yaw movement.

4) **CNN depth reconstruction:** (i) Very good result for indoor navigation. Easy to train over indoor environments.

(ii) Point cloud generated from this algorithm is dense and more accurate.

Table II shows the comparison of the extracted height and width of a know object from the point cloud data of the 3 different algorithms. The predicted height has the least error for the CNN based approach as compared to the other two.

TABLE II
OBJECT SIZE COMPARISON

	Height(meters)	Width(meters)
Actual	0.8	0.5
PTAM	0.21	0.26
ORB-SLAM2	0.562	0.351
CNN	0.668	0.532

VI. CONCLUSION

The aim of this paper is to give a comparative analysis of some available vSLAM systems. On implementing each of these, we realise that the systems implemented earlier were not up to the mark. The study concludes the following:

(i) By the observations made on the basis of the readings by the Point Cloud Library we conclude that the point cloud generated by Vision-based Depth Reconstruction is perfect as the measurements are close to the real world measurements (Table II).

(ii) If there is no constraint over hardware restriction Vision-based Depth Reconstruction (TX2 _ FCNN) is the best method as compared to the other 3 for implementing Simultaneous Localisation and mapping for indoor environments. The depth map output can be directly used for RGBD SLAM and it performs admirably.

(iii) Amongst all methods that use Bundle Adjustment i.e PTAM, ORB-SLAM and ORB-SLAM2. Performance of ORB-SLAM2 is comparatively better as it is based on the basic principal of PTAM but has some enhancement such as a separate thread for loop closure and detection. To improve the quality of map generated a separate thread for full Bundle Adjustment is also involved.

ACKNOWLEDGMENT

We would like to express deep sense of gratitude to Markus Achtelik, Stephan Weiss and Simon Lynen, developers of PTAM-ROS package, Raul Mur-Artal developer of ORB-SLAM and ORB-SLAM 2 package, Bokovoy, A. and Muravyev, K. and Yakovlev, K. developer of Vision-based Depth Reconstruction (TX2 _ FCNN) package. We would like to thank our colleague Soofiyan Atar for his support in developing some scripts for implementation. Lastly, the we'd like to thank the communities surrounding these FOSS packages for the discussions and support.

REFERENCES

- [1] 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.
- [2] K. Di, W. Wan, H. Zhao, Z. Liu, R. WANG, and F. Zhang, "Progress and applications of visual slam," *Journal of Geodesy*, vol. 2, 2019.
- [3] J. Shi *et al.*, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*. IEEE, 1994, pp. 593–600.
- [4] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.
- [5] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.
- [6] A. Bokovoy, K. Muravyev, and K. Yakovlev, "Real-time vision-based depth reconstruction with nvidia jetson," in *2019 European Conference on Mobile Robots (ECMR)*. IEEE, 2019, pp. 1–6.
- [7] M. Labbé and F. Michaud, "Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.
- [8] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*. Springer, 2006, pp. 430–443.
- [9] W. Guan, Q. Liu, G. Han, B. Wang, and S. Li, "An improved coarse-to-fine method for solving generation tasks," in *Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association*, 2019, pp. 178–185.
- [10] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, and J. J. Berles, "Stereo parallel tracking and mapping for robot localization," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 1373–1378.
- [11] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 1. Ieee, 2004, pp. 1–1.
- [12] P. Chen and D. Suter, "Simultaneously estimating the fundamental matrix and homographies," *IEEE transactions on robotics*, vol. 25, no. 6, pp. 1425–1431, 2009.
- [13] Y. Chen, Y. Chen, and G. Wang, "Bundle adjustment revisited," *arXiv preprint arXiv:1912.03858*, 2019.
- [14] G. Younes, D. Asmar, E. Shammass, and J. Zelek, "Keyframe-based monocular slam: design, survey, and future directions," *Robotics and Autonomous Systems*, vol. 98, pp. 67–88, 2017.
- [15] H. C. Longuet-Higgins, "The reconstruction of a plane surface from two perspective projections," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 227, no. 1249, pp. 399–410, 1986.
- [16] R. Mur-Artal and J. D. Tardós, "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.
- [17] B. Li, C. Shen, Y. Dai, A. Van Den Hengel, and M. He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1119–1127.
- [18] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. L. Yuille, "Towards unified depth and semantic prediction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2800–2809.
- [19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [20] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 263–272, 2017.
- [21] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgb-d images," in *European conference on computer vision*. Springer, 2012, pp. 746–760.