2022 IEEE NIGERCON

A Modified Visual Simultaneous Localisation and Mapping (V-SLAM) Technique for Road Scene Modelling

Jibril Abdullahi Bala
Depertment of Mechatronics Engineering
Federal University of Technology, Minna
Minna, Nigeria
jibril.bala@futminna.edu.ng

Steve Adeshina
Department of Computer Engineering
Nile University of Nigeria
Abuja, Nigeria
steve.adeshina@nileuniversity.edu.ng

Abiodun Musa Aibinu
Department of Mechatronics Engineering
Federal University of Technology, Minna
Minna, Nigeria
abiodun.aibinu@futminna.edu.ng

***Abstract*— Visual Simultaneous Localization and Mapping (V-SLAM) which involves the use of cameras to map an environment and estimate agents' pose within that environment has become widely popular in the field of autonomous vehicles. Numerous V-SLAM schemes have been implemented which utilize various feature extraction methods, one of which is the use of Convolutional Neural Networks (CNN). One main shortcoming of existing approaches is that they do not focus on object detection of road sceneries which are characterized by their varying complexity, thus making them unsuitable for real time implementation. Therefore, this study presents a modified V-SLAM scheme for road scene modelling. The technique utilizes YOLOv4 for object detection, and uses the ORB features obtained from the objects to update the features in the main V-SLAM algorithm. The results showed that the modified V-SLAM technique was capable of estimating the agent's position and orientation, map the environment. The technique gave a Root Mean Square Error of 0.11621 and a point-to-point distance of 1.1726m.**

Keywords— Autonomous Vehicles, Deep Learning, Computer Vision, Visual SLAM, YOLO v4

I. INTRODUCTION

Autonomous Vehicles (AVs) are capable of navigating environments with minimal human input, perceive the surroundings using sensors, and utilise control schemes to generate paths and navigate the route [1]. Perception and localisation are major operations that determine the success of an autonomous agent. The AV uses sensors to perceive its environment, identify and avoid obstacles, and plan its trajectory from the start location to the final destination [2]. The ability of an AV to perceive its surroundings and identify its position within its environment gives it the prerequisite ability to navigate that particular surrounding [3]. Due to advancements in computer and sensor technologies, AV research and development have seen significant improvements in recent years [4]. One area that has been closely associated with AV perception is Computer Vision.

As a result of the giant strides made in Computer Vision technologies, Visual Odometry has proven to be a popular method in implementing AV perception modules. This is mainly due to the low-cost, portability, ease of hardware set-up, and versatility of cameras [5]. Visual Simultaneous Localisation and Mapping (V-SLAM) enables an autonomous system to determine its position and orientation by processing images in real time [6]. This technique, which is a method of Visual Odometry, also allows an AV to map its environment in real time. Although numerous feature extraction techniques have been implemented in V-SLAM such as FAST [7], ORB [8], SIFT [9], and BRIEF [10], deep learning has proven to be a reliable feature extraction method especially in the area of semantic segmentation and object detection [11], [12].

Despite the significant advancements in deep learning techniques utilised in V-SLAM schemes, these CNN-based methods do not focus on object detection of road sceneries of varying complexity and are not suitable for real time implementation. Therefore, this study presents a modified V-SLAM scheme based on CNN for road scene modelling. This method is expected to model the road scene based on objects, landmarks, and other features present. Additionally, the technique will estimate the autonomous agent's position and orientation within the environment.

The major contribution of this research is the utilisation of YOLO v4 to filter the keypoints in the V-SLAM technique, thus providing the conventional V-SLAM with the ability to detect major objects in the scene and incorporate the object features in the SLAM process. The rest of this paper is organised into four parts. Part II presents a review of literature while the research methodology is presented in part III. The results and relevant discussions are presented in part IV while the conclusion and direction for future research directions are presented in part V.

II. LITERATURE REVIEW

Several studies have been undertaken in the domain of Deep Learning-based V-SLAM algorithms. CNN-SLAM, a Real-time dense monocular SLAM with learned depth prediction, was presented in [13]. The research combines SLAM with depth prediction using a deep neural network. The approach also handles pure rotational movements while keeping the robustness and accuracy of direct monocular SLAM. The approach, on the other hand, was incapable of detecting objects.

In [14], a Deep Learning Approach for Moving Object Tracking using ML-RANSAC Algorithm for SLAM implementation in dynamic situations was introduced. For multi-target tracking, a unique version of the

978-1-6654-7978-3/22/$31.00 ©2022 IEEE

RANdomSAmple Consensus (RANSAC) approach known as multilevel-RANSAC (ML-RANSAC) inside the Extended Kalman Filter (EKF) framework is used. Experiments confirmed that the suggested approach effectively tracked an unknown number of randomly placed moving objects. This study makes use of both LIDAR and vision sensors, making it costly to deploy.

Additionally, [15] created an Unsupervised Deep Visual-Inertial Odometry method for RGB-D Imagery with Online Error Correction. A deep learning-based Visual Inertial Odometry system was built in this work. For RGB-D pictures, the system was capable of online error correction. The system was simulated using the KITTI dataset, although the rotational error was rather considerable.

Also, [16] developed VPS-SLAM which is a Visual Planar Semantic SLAM for Aerial Robotic Systems. The authors introduced a visual semantic SLAM approach for aerial robots that is both robust and lightweight. This method extracts semantic information from an indoor environment and maps it using YOLO v2. The system was evaluated in a static indoor setting, and the study's aerial structure suggests that the results are not generalizable to road scene modelling.

Consequently, in underground tunnel dynamic environments, [17] devised a Visual-Inertial localization approach for Unmanned Aerial Vehicles. The research provides a dynamic point detection and rejection approach based on neural network semantic segmentation. Dynamic object interference is eliminated during pose estimation as a result of this. The approach was tested using the EuRoC dataset and subsurface photos from a tunnel. As a result of the mismatch in complexity, the approach cannot be used for road scenes.

Furthermore, [18] created a CNN-based Feature-point Extraction for Real-time Visual SLAM using Embedded FPGA. In this work, a hardware-software co-design feature-point extractor based on the cutting-edge CNN-based approach, SuperPoint, is investigated. In terms of computation speed, SuperPoint was faster than SIFT and comparable to ORB. However, the approach lacked the ability to identify objects.

In addition, a CNN-based location identification solution for LIDAR SLAM was demonstrated [19]. The authors studied the performance of a CNN-based classifier with Lidar data for location identification tasks in this paper. The results of the testing reveal that the suggested model beats both the Nearest Neighbor and Random Forests techniques, with a true positive rate of 70.05 percent. Many false matches, however, arise when scenarios have extremely similar qualities and the use of LIDAR is expensive.

In [20], a dense monocular visual SLAM based on CNN for indoor mapping and autonomous exploration was created. This research combines SLAM techniques with CNN-based single picture depth estimation algorithms to densify and scale the data and create a map of the environment appropriate for exploration in real time. A UAV might utilize the technology to create a navigable 3D map of an inside space using only a monocular camera. The developed system was tested on a commercial, off-the-shelf UAV and was capable of creating a map of an unfamiliar location. However, the approach lacked object recognition skills and was unsuitable for outside settings.

In [21], a real-time depth estimation approach for SLAM systems was created utilizing Recurrent CNN with Sparse Depth Cues. The work offers a model that explores spatio-temporal information using convolutional GRU and sparse depth cues. In the trials, the proposed approaches outperformed existing methods using a real-time system. This approach, on the other hand, focuses on depth estimates rather than object identification. As a result, the model is inappropriate for object detection applications.

DLOAM: Real-time and Robust LiDAR SLAM System Based on CNN in Dynamic Urban Environments was also presented in [22]. The authors suggested a fast LiDAR-only model-free dynamic object recognition approach that employs point cloud spatial and temporal information via a convolutional neural network (CNN). This technique's accuracy and robustness were assessed, and the detection accuracy increased by 35% to 86 percent when compared to approaches that solely employ spatial information. The use of LIDAR, on the other hand, has shown to be noisy and problematic in places with severe temperatures.

According to the examined studies, one main shortcoming of existing approaches is that they do not focus on object detection of road sceneries of varying complexity and are not suitable for real time implementation. Thus, one of the most important contributions of this work is the development of a feature extraction approach for modelling the scene of a road by extracting information on the objects, landmarks, and other elements present using deep learning approaches.

III. RESEARCH METHODOLOGY

A. System Overview

The modified V-SLAM technique was built based on ORB-SLAM which is an open source visual SLAM technique that is suitable for monocular, stereo, and RGB-D cameras [23]. The choice of this process is based on its high accuracy and precision [24]. ORB-SLAM has three major components, namely: Tracking, Local Mapping and Loop Closure. The tracking component involves camera localisation, keypoint extraction and determination of when to insert a new keyframe. This is achieved through the process of extracting ORB features from the image frames. In the Local Mapping component, the system uses the keyframes to reconstruct the surroundings of the camera pose. Finally, the system searches for loops in the keyframe in the Loop Closure component. This is done to optimise the final map generated in the process.

The components of the ORB-SLAM process only handle tracking, mapping, and loop closure. Therefore, in order to achieve object detection, deep learning was utilised in this study. The deep learning technique selected for operation is YOLO v4 [25]. YOLO (you only look once) has proven to be faster and more accurate than other object detection models such as R-CNN [22], [26]. The model provides the classes of detected objects, a bounding box around the object, and confidence levels of each detected object. Additionally, the method has low computational requirements compared to other deep learning models and even has a ‘tiny’ version for deployment on embedded hardware. This integration with the V-SLAM technique is presented in Fig. 1.

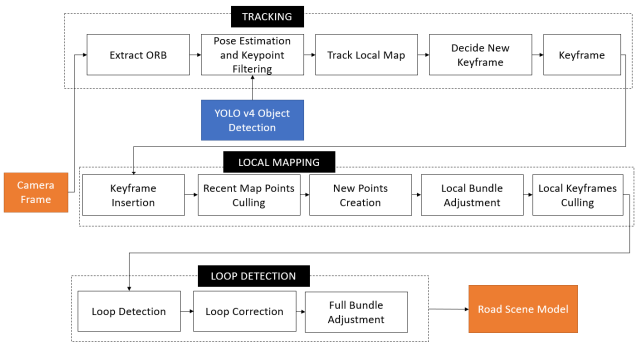

Fig. 1. Modified V-SLAM Technique

From Fig. 1, it can be observed that the YOLO v4 technique is instrumental in the keypoint selection (Tracking component). The technique detects objects in the scene, and relays the information to the keypoint filtering module of the tracking component. This in turn provides information on the detected objects in the scene.

B. Visual SLAM Process

The tracking component of the visual SLAM process involves identifying keyframes and deciding when to insert them, as well as localising the camera with every frame. This component also performs map initialisation and local map tracking. First, 1000 ORB features were extracted at 8 scale levels with a scale factor of 1.2. The matched ORB features are filtered using YOLO v4 deep learning approach. This is done to ensure that the detected objects are captured under the features extracted from the frame.

In order to determine the initial camera pose, feature matching is performed between the previous frame and the current frame. Prior to that however, the camera is calibrated according to the camera intrinsic parameters for the testing dataset. The ORB features for both the initial frame and subsequent frame are then matched using the pairwise distance between the features. The index of the matched features as well as the sum of absolute differences (equation 1) are obtained from the process.

$$SAD = \sum_{j=1}^{i} \left| I(x_i, y_i) - I(x_j, y_j) \right| \quad (1)$$

$I(x_i, y_i)$ and $I(x_j, y_j)$ represent the initial frame and subsequent frame respectively. Based on the matched features (correspondences), a Homography matrix, H_m (equation 2) and a Fundamental matrix, F_m (equation 3) are evaluated. These matrices are geometric transformation models used for map initialisation. The Homography matrix is suitable for planar scenes while the Fundamental matrix is suitable for non-planar scenes. In this study however, the choice of the matrix is based on the ratio output of equation 4.

$$x'_i F_m x_i = 0 \quad (2)$$

$$x'_i = H_m x_i \quad (3)$$

$$Matrix\ Selection = \frac{score_h}{score_h + score_f} > 0.45 \quad (4)$$

The parameters x_i and x_i' respectively represent the previous frame and current frame, while $score_h$ and $score_f$ represent the scores of H_m and F_m respectively. The relative camera location and orientation is evaluated from the geometric transformation matrix, the camera intrinsic parameters, and any inliers identified by the transformation matrix. The next step is to extract the 3D world co-ordinates from the two 2D frames. This is achieved using a process called triangulation and the map can be projected into the frame. The map point correspondences are searched for in the current frame to establish if the set of keyframes (in the current frame) match with the keyframes in the reference frame. The camera pose is subsequently optimised with the map points found in the current frame.

A frame is determined to be a keyframe if it satisfies these conditions:

- At least 20 frames have passed since the last keyframe or if the current frame tracks less than 100 keypoints.
- The map points identified (or tracked) in the current frame are less than 90% of the map points tracked in the reference keyframe.

In the mapping component, the map points of the new keyframe are used to update the existing map points identified in previous keyframes. In addition, a valid map point needs to be identified in at least three keyframes, otherwise it will be discarded or 'culled'. This 'culling' process is done to minimise the inclusion of outliers and improve the accuracy. The new map points are created by triangulation between the ORB features of the current keyframe and its connected keyframes. The pose of the current keyframe is refined via Local Bundle Adjustment which is an optimization process.

Loop detection is performed using a 'Bag of Words' (BoW) model. This technique stores a vocabulary representation of the keyframe features. If the current keyframe is visually similar to any keyframe in the BoW database, a loop connection is established. This signifies that particular scene has been visited in the past. Afterwards, a full bundle adjustment is performed to optimise the final scene model.

C. Object Detection and Keypoint Filtering

This study implements the YOLO v4 model which has been pretrained on the COCO dataset [27]. This dataset has 80 different object categories capable of detecting cars, trains, people etc. The YOLO v4 [25] architecture consists of three major sections. These are the Backbone, the Neck, and the Detection Head, as shown in Fig. 2. The backbone uses CSP-Darknet53 (Cross Stage Partial Darknet53). This model is characterised by its higher input resolution and larger receptor fields which are useful in viewing entire objects in an image and detection of small objects, respectively. The neck consists of the PANet (Path Aggregation Network) as the method of parameter aggregation from the various backbone levels. The Detection Head section predicts the bounding boxes, classification, and score. This is achieved using the YOLO v3 model.

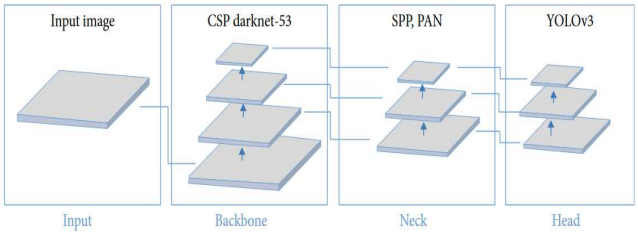

Fig. 2. YOLO v4 Architecture [28]

The object detection and keypoint filtering process using YOLO v4 is presented in Fig. 3. The process starts by using the YOLO algorithm to identify the objects in the scene and return the bounding boxes. The algorithm then proceeds to evaluate if the bounding boxes fall within the region of interest and rescale the box co-ordinates if otherwise. The bounding box co-ordinates will be used as regions of interest to extract the ORB features in the vicinity of the object. The ORB points identified in the object area will be appended to the points detected in the overall image. This process ensures that the features of the objects identified are utilised in the localisation and mapping process.

ALGORITHM 1: KEYPOINT FILTERING ALGORITHM

Input: *ORB features of main image,* F_m,
YOLO bounding box list of detected objects, BB,
Output: *Filtered Keypoints,* F_{new}

keypoint_list = 0,
$F_{all} = 0$;
for *i = 1 to size (BB)*
  resize BB(i) to fit F_m *dimension;*
  extract ORB features, F_b *from BB(i);*
  keypoint_list = keypoint_list + F_b;
end
$F_{new} = F_m$ + *keypoint_list;*

Fig. 3. Keypoint Filtering Algorithm

IV. RESULTS

A. Experimental Enviroment and Setup

The modified V-SLAM algorithm was run on an Intel core i7 processor with a speed of 2.2 GHz, a RAM of 8GB, and an NVIDIA GeForce GTX GPU with a size of 8GB. The YOLO v4 model was pretrained with on the COCO dataset which has 80 different object categories capable of detecting cars, trains, people, etc. The modified V-SLAM technique was tested on the TUM-RGB dataset. The data comprised of 2585 RGB images of the environmental setup. These images were obtained with a Microsoft Kinect camera and had a resolution of 640 x 480. The details of the dataset sequence are presented in Table I while an image sample of the dataset is presented in Fig. 4.

TABLE I. DATASET SEQUENCE DETAILS

Parameter	Value
Sequence Name	'freiburg3_long_office_household'
Duration	87.10 secs
Ground Truth Trajectory Length	21.455m
Average Translational Velocity	0.249 m/s
Average Angular Velocity	10.188 deg/s
Nuumber of Frames	2585

B. Deep Learning Based Object Detection

The model was successfully able to identify objects from the test dataset as well as extract ORB features from those object areas. Fig. 4 shows the bounding boxes around the detected objects with their appropriate labels and confidence scores, as well as the identified and extracted ORB features in the object vicinity. These extracted features are subsequently appended to the feature list used in the tracking process. It can also be observed in Fig. 4 that the ORB features on the smaller images are not identified. This is because smaller objects with a width and height lower than 64 pixels will not be subjected to feature extraction, since they play a minimal role in road scene modelling. Additionally, if the bounding box extends to outside the image height and width, it is resized to be contained within the image dimensions.

Fig. 4. ORB Features Identified on Extracted Objects

C. Modified Visual SLAM

Fig. 5 shows the reference keyframe matching process used to initialise the map. This keyframe decision is based on the matched features between the first keyframe and next keyframe that satisfies the keyframe selection conditions.

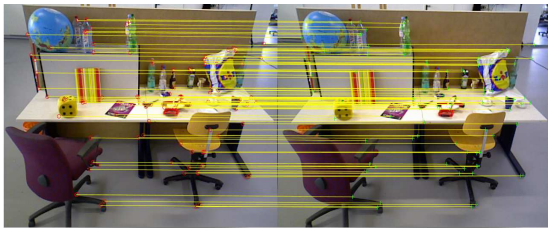

Fig. 5. ORB Feature Matching between Frames

After the map is initialised the algorithm tracks the detected ORB points on every frame to localise the agent's position and map the surroundings. After the algorithm goes through all the frames, it optimises the trajectory using Bundle Adjustment and compares the trajectory with the ground truth to evaluate the error. Fig. 6 shows the ORB features detected in Frame 17, while Fig. 7 the map points corresponding to the identified features, and the trajectory being built.

Fig. 6. ORB Features Extracted in Frame 17

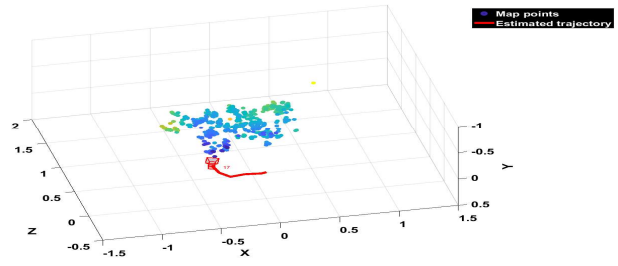

Fig. 7. Localisation and Mapping Process in Frame 17

At the conclusion of the localisation and mapping process, a full optimised map is obtained as shown in Fig. 8. The obtained trajectory is compared with the ground truth obtained from the dataset. The 3D map points represent the identified features in each of the frames. The modified V-SLAM technique was evaluated in comparison with the ground truth and the result in Fig. 8 showed that the technique was capable of localising and mapping the test environment with a Root Mean Square Error of 0.11621 and a point-to-point distance of 1.1726m.

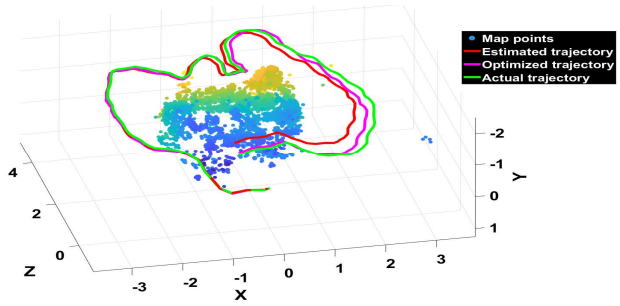

Fig. 8. Optimised Trajectory Compared with Ground Truth

V. Conclusion

In this study, a modified Visual SLAM technique was presented. The algorithm used the deep learning approach, YOLO v4 to identify objects in a scene. The ORB features from the identified objects were extracted, and using a keypoint filtering algorithm, the input image features were updated with the features from the identified objects. The results showed that the technique was able to successfully identify the objects, update the SLAM ORB feature set, and accurately track and map the environment. The trajectory obtained was compared with a ground truth and the method gave an RMSE of 0.11621 and a point-to-point distance of 1.1726m. Future works will focus on improving the robustness and accuracy of the algorithm by testing it on different datasets.

Acknowledgment

The authors wish to acknowledge the National Information Technology Development Agency (NITDA) for their support through the 2020 NITDEF scholarship scheme.

References

[1] H. Guo, W. Li, M. Nejad, and C.-C. Shen, “Proof-of-Event Recording System for Autonomous Vehicles: A Blockchain-Based Solution,” IEEE Access, vol. 8, pp. 182776–182786, 2020.

[2] A. Benterki, M. Boukhnifer, V. Judalet, and C. Maaoui, “Artificial intelligence for vehicle behavior anticipation: Hybrid approach based on maneuver classification and trajectory prediction,” IEEE Access, vol. 8, pp. 56992–57002, 2020.

[3] Y. Jeong, S. Kim, and K. Yi, “Surround Vehicle Motion Prediction Using LSTM-RNN for Motion Planning of Autonomous Vehicles at Multi-Lane Turn Intersections,” IEEE Open J. Intell. Transp. Syst., vol. 1, no. January, pp. 2–14, 2020.

[4] L. Tang, F. Yan, B. Zou, K. Wang, and C. Lv, “An improved kinematic model predictive control for high-speed path tracking of autonomous vehicles,” IEEE Access, vol. 8, pp. 51400–51413, 2020.

[5] A. Yusefi, A. Durdu, M. F. Aslan, and C. Sungur, “LSTM and filter based comparison analysis for indoor global localization in UAVs,” IEEE Access, vol. 9, pp. 10054–10069, 2021.

[6] S. Saeedi et al., “Navigating the Landscape for Real-Time Localization and Mapping for Robotics and Virtual and Augmented Reality,” Proc. IEEE, vol. 106, no. 11, pp. 2020–2039, 2018.

[7] J. Luo and S. Qin, “A fast algorithm of simultaneous localization and mapping for mobile robot based on ball particle filter,” IEEE Access, vol. 6, pp. 20412–20429, 2018.

[8] C. Shao, C. Zhang, Z. Fang, and G. Yang, “A Deep Learning-Based Semantic Filter for RANSAC-Based Fundamental Matrix Calculation and the ORB-SLAM System,” IEEE Access, vol. 8, pp. 3212–3223, 2020.

[9] C. H. Chien, C. C. J. Hsu, W. Y. Wang, and H. H. Chiang, “Indirect Visual Simultaneous Localization and Mapping Based on Linear Models,” IEEE Sens. J., vol. 20, no. 5, pp. 2738–2747, 2020.

[10] Z. Niu, X. Zhao, J. Sun, L. Tao, and B. Zhu, “A Continuous Positioning Algorithm Based on RTK and VI-SLAM with Smart phones,” IEEE Access, vol. 8, pp. 185638–185650, 2020.

[11] I. Rusli, B. R. Trilaksono, and W. Adiprawita, “RoomSLAM: Simultaneous localization and mapping with objects and indoor layout structure,” IEEE Access, vol. 8, pp. 196992–197004, 2020.

[12] X. Zhao, C. Wang, and M. H. Ang, “Real-Time Visual-Inertial Localization Using Semantic Segmentation towards Dynamic Environments,” IEEE Access, vol. 8, pp. 155047–155059, 2020.

[13] K. Tateno, F. Tombari, I. Laina, and N. Navab, “CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction,” Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, pp. 6243–6252, 2017.

[14] M. S. Bahraini, A. B. Rad, and M. Bozorg, “SLAM in Dynamic Environments : A Deep Learning,” Sensors, vol. 19, pp. 1–20, 2019.

[15] E. J. Shamwell, K. Lindgren, S. Leung, and W. D. Nothwang, “Unsupervised Deep Visual-Inertial Odometry with Online Error Correction for RGB-D Imagery,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 10, pp. 2478–2493, 2020.

[16] H. Bavle, P. De La Puente, J. P. How, and P. Campoy, “VPS-SLAM: Visual Planar Semantic SLAM for Aerial Robotic Systems,” IEEE Access, vol. 8, pp. 60704–60718, 2020.

[17] D. Li et al., “A visual-inertial localization method for unmanned aerial vehicle in underground tunnel dynamic environments,” IEEE Access, vol. 8, pp. 76809–76822, 2020.

[18] Z. Xu, J. Yu, C. Yu, H. Shen, Y. Wang, and H. Yang, “CNN-based Feature-point Extraction for Real-time Visual SLAM on Embedded FPGA,” Proc. - 28th IEEE Int. Symp. Field-Programmable Cust. Comput. Mach. FCCM 2020, pp. 33–37, 2020.

[19] Y. Yang, S. Song, and C. Toth, “CNN-based place recognition technique for LIDAR SLAM,” Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci., vol. 44, pp. 117–122, 2020.

[20] A. Steenbeek, “CNN based dense monocular visual SLAM for indoor mapping and autonomous exploration,” 2020.

[21] S. J. Lee, H. Choi, and S. S. Hwang, “Real-time Depth Estimation Using Recurrent CNN with Sparse Depth Cues for SLAM System,” Int. J. Control. Autom. Syst., vol. 18, no. 1, pp. 206–216, 2020.

[22] W. Liu, W. Sun, and Y. Liu, “DLOAM: Real-time and Robust LiDAR SLAM System Based on CNN in Dynamic Urban Environments,” IEEE Open J. Intell. Transp. Syst., 2021.

[23] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM: a versatile and accurate monocular SLAM system.,” IEEE Trans. Robot., vol. 31, no. 5, pp. 1147–1163, 2015.

[24] J. C. V. Soares, M. Gattass, and M. A. Meggiolaro, “Crowd-SLAM: Visual SLAM Towards Crowded Environments using Object Detection,” J. Intell. Robot. Syst. Theory Appl., vol. 102, no. 2, pp. 1–16, 2021.

[25] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” arXiv Prepr. arXiv2004.10934., 2020.

[26] B. Y. Suprapto, A. Wahyudin, H. Hikmarika, & S. Dwijayanti. “The Detection System of Helipad for Unmanned Aerial Vehicle Landing Using YOLO Algorithm. Jurnal Ilmiah Teknik Elektro Komputer dan Informatika, 7(2), 193-206, 2021.

[27] T. Y. Lin et al., “Microsoft COCO: Common Objects in Context,” Eur. Conf. Comput. vision. Springer, Cham., pp. 740–755.

[28] Y.-C. Fan, C. M. Yelamandala, T.-W. Chen, and C.-J. Huang, “Real-Time Object Detection for LiDAR Based on LS-R-YOLOv4 Neural Network,” J. Sensors, vol. 2021, 2021