2021 IEEE 6th International Conference on Computing, Communication and Automation (ICCCA)
Aurel Vlaicu University of Arad, Romania. Dec 17-19, 2021

Visual SLAM Combined with Object Detection for Autonomous Indoor Navigation using Kinect V2 and ROS

Mihir Kulkarni
Dept. of Electronics and Telecommunication Engineering
College of Engineering Pune
Pune, India
kulkarnimm18.extc@coep.ac.in

Pranay Junare
Dept. of Electronics and Telecommunication Engineering
College of Engineering Pune
Pune, India
junarepg18.extc@coep.ac.in

Mihir Deshmukh
Dept. of Electronics and Telecommunication Engineering
College of Engineering Pune
Pune, India
deshmukhmp18.extc@coep.ac.in

Priti P. Rege
Dept. of Electronics and Telecommunication Engineering
College of Engineering Pune
Pune, India
ppr.extc@coep.ac.in

2021 IEEE 6th International Conference on Computing, Communication and Automation (ICCCA) | 978-1-6654-1473-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCCA52192.2021.9666426

***Abstract*—SLAM can be defined as exploring the unknown environment while mapping the robot's surroundings alongside estimating its pose (i.e., position and orientation). It is primarily done using the sensors mounted on the robot. SLAM enables us to autonomously navigate the robot throughout the map based on given final goal coordinates or waypoints. However, SLAM algorithms alone are not capable of performing complex tasks such as autonomous payload delivery in warehouses, healthcare facilities, etc. These tasks require additional semantic information about the environment. To solve this problem, we propose a solution where the traditional Visual SLAM method is accompanied by object detection using pre-trained CNNs to enhance the robot's capabilities of navigating efficiently and performing robust 3D perception in indoor environments. RTAB-Map using the KinectV2 RGB-D Camera is selected to perform Visual SLAM while the YOLO V3 tiny model acts as the CNN detector for detecting objects of interest. Development platform used is ROS & Gazebo. The proposed solution is experimentally verified by simulating the Turtlebot in the Gazebo environment.**

Keywords— V-SLAM, Mobile Robot, YOLO, CNN's, ROS, Gazebo, Rtabmap

I. INTRODUCTION

Advances in Mobile Robotic platforms have enabled us to perform a variety of tasks using a robot, right from search and rescue operations to delivery services required in industries or hotels. In all these scenarios, the robot needs to have knowledge of the entire area in terms of a map and the precise locations of the target objects in the map. To achieve autonomous navigation, pose estimation and map building are two extremely vital tasks required for any mobile robot. Therefore, simultaneous localization and mapping (SLAM) is one of the most popular research topics and an actively studied problem in the world of robotics. Unfortunately, traditional SLAM approaches such as the creation of point occupancy grid map in LiDAR-based SLAM or low-level geometric feature extraction to build dense point cloud map in Visual SLAM (VSLAM) algorithms alone are not capable of performing complex tasks such as efficient autonomous delivery of different payloads from point location A to point location B in warehouses or handing over the meal to a specific patient in a Healthcare facility. On the other hand, developments in Machine learning, Deep learning and Computer Vision technology have provided few solutions to this challenging problem.

Our paper is organized in the following pattern: Section 2 describes work related to various VSLAM algorithms, object detection algorithms and earlier attempts to integrate the two. Section 3 enlists the Methodology that has been followed by us in order to achieve the problem statement. In Section 4 we briefly describe the experimental results that were observed during the implementation. Finally in Section 5 we have concluded our work and have discussed the future scope of this research.

II. RELATED WORK

In the last decade, various Visual-SLAM techniques have emerged such as ORB-SLAM, ORB-SLAM2, VINS Mono, RTABMap, PTAM, LSD-SLAM, DSO, etc. which have played a crucial role in the field of autonomous robot navigation [1][2]. In [3] a brief survey about the development of SLAM and compares a few common techniques like PTAM, ORB-SLAM and MonoSLAM. It further goes on to state advances in RGB-D VSLAM techniques. In [4] the perception ability of the autonomous robot navigation is enhanced by extracting object-level semantic information from the detected objects by using Mask-RCNN deep neural network and RTAB-Map Visual SLAM algorithm. In [5], an approach to solve the service robot exploration problem during the first use when opened from its package is given. SLAM algorithm is enhanced by implementing object detection using CNN network on the furniture and household dataset which in turn enhances the robot locomotion capability.

Deep learning has been applied to undertake many computer vision tasks, such as recognition, detection, and image segmentation. Object detection is divided into two major tasks: object recognition and object localization. [6] proposes an approach for indoor object detection based on a convolutional neural network (CNN). Both public Indoor Dataset and private frames of videos (FoV) dataset are used to pre-train an off-line CNN model. In [17] comparison between different single-stage and two-stage object detection models such as RCNN, Fast RCNN, Faster

Authorized licensed use limited to: COLLEGE OF ENGINEERING - Pune. Downloaded on January 11,2022 at 22:42:02 UTC from IEEE Xplore. Restrictions apply.

RCNN, You only look once (YOLO) v1, v2, v3, and SSD are covered. The result of the study showed that the YOLO v3-Tiny enhances the speed of object detection while maintaining good enough accuracy. In [16], an improved CNN based VSLAM system is proposed that replaces the single convolution layer by a parallel architecture with reduced number of parameters. While [5] provides mapping and object detection in a 2D plane, our research extends this into 3D space to allow more precise movement of the robot.

III. METHODOLOGY

In this Section the complete methodology followed by us is explained in detail. Shown below (Ref. Fig. 1) is the Block diagram of the implemented Robotic System.

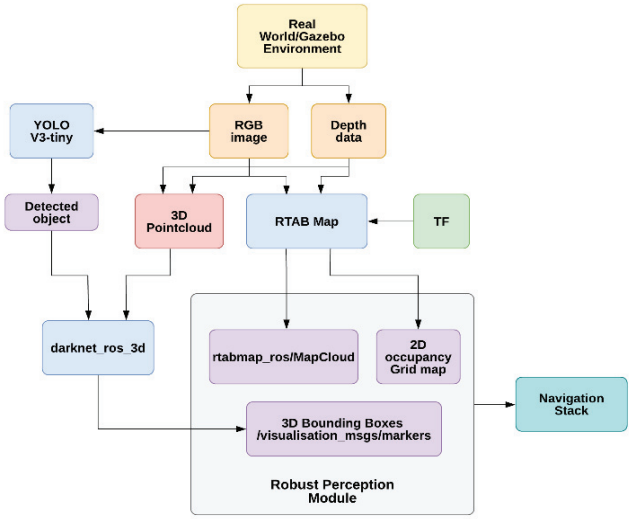

Fig. 1. Block Diagram

A. Setting Simulation environment

Currently there are many available softwares for robotics development such as Webots, UARSim, Matlab-Simulink, V-REP, Carla, etc. We choose robot operating system (ROS) and Gazebo due to their high flexibility, large community support, rich repository of packages and ability to mimic real-world scenarios. As mentioned earlier, ROS is a powerful system for robot software development that provides hardware abstraction and a low-level control interface between hardware and software. For simulation ROS melodic compatible with Ubuntu 18.04 was chosen. Gazebo is a well-designed 3D simulator that delivers dynamic simulation, advanced 3D graphics, sensors and noise, plugins as well as command line tools and offers robust simulation with its physics engine. Gazebo has emerged as one of the most popular simulators for developers in the past few years. Moreover, the compatibility of Gazebo with ROS is unmatched since it is developed and distributed by the same open-source organization, Open Robotics. On the other hand, a powerful GUI and an important tool to visualize various robot parameters, known as rviz is also used. It lays out a convenient GUI to visualize sensor information, camera images, robot model specifications, environment maps, and is an immensely useful feature that has emerged in the field of robotics which is useful for enhancing and debugging your robot controllers.

Another interesting approach to understanding Rviz and Gazebo, as rightly said by one of the original developers of ROS, Morgan Quigley: "Rviz shows you what the robot thinks is happening, while Gazebo shows you what is really happening."

1) Robot Model

TurtleBot is a ROS standard platform robot that comes with an open-source software. Derived from the Turtle robot, it was initially designed to get beginners started with ROS and robotics as well as to teach computer programming language. Because of its open-source character, detailed documentation, and easily available resources, it has become the go-to platform for ROS learners, developers and students. The Turtlebot3 comes in 3 variants (Ref. Fig. 2). Developed by Tully (now Open Robotics) and Melonee (Fetch Robotics) from Willow Garage, the Turtlebot1 has been on sale since 2011 after being developed in 2010 for ROS development. Later, the Turtlebot2 was developed by Yujin Robot in 2012 that was centered around the research robot, iClebo Kobuki. Finally, in 2017, due to the advances in ROS and the demands of the users, TurtleBot3 was developed acquainting to the lacking functions of its earlier generations [7].

Fig. 2. Turtlebot Series [10]

2) Our Simulation Environment

To test the system and verify its feasibility, we first simulated the robot and its environment in a Gazebo simulator. For the virtual environment, a custom URDF world file was written which consisted of a few test objects such as dustbin, cupboard, box, etc. (Ref. Fig. 3) and the de facto Turtlebot2 equipped with an RGB-D camera was chosen as a standard robot for testing.

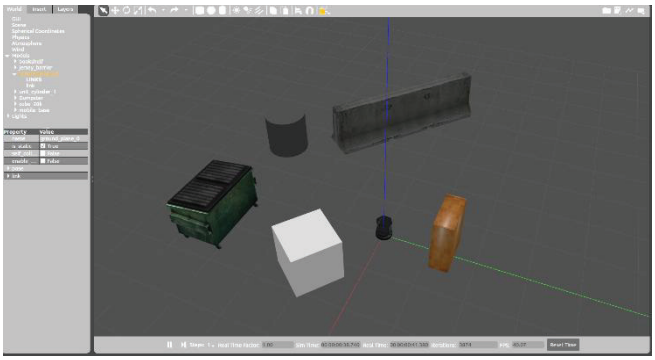

Fig. 3. Simulated Gazebo Environment

B. RTABMap implementation in Gazebo

Real-time appearance-based mapping (RTABMap) [8], is an open-source library. It implements loop closure detection with a memory management approach which limits the size of the map. The loop closure detections are always processed under a fixed time limit which helps in fulfilling online requirements for long-term and large-scale environment mapping. RTAB-Map is an RGB-D graph-based SLAM approach. It is an incremental appearance-based loop closure detector. [12] Loop closure refers to recognizing a previously visited location and updating the learnt map accordingly. The

Authorized licensed use limited to: COLLEGE OF ENGINEERING - Pune. Downloaded on January 11,2022 at 22:42:02 UTC from IEEE Xplore. Restrictions apply.

loop closure detector utilizes a bag-of-words approach to determine how likely a new image comes from a previous location or a new location. A new constraint is added to the map's graph when a loop closure hypothesis is accepted. The errors in the map are minimized by a graph optimizer. In RTABMap a memory management approach is incorporated, which limits the space of the graph required for loop closure detection such that it is capable of performing mapping in real-time and in large environments too. RTABMap which is a graph-based SLAM approach, can be divided into three parts namely: sensor measurement, frontend, and backend. In the frontend stage, the processing of sensor information is done, and geometric features from the consecutive image frames are extracted. Thus we can say that while odometry estimation is done in the frontend stage, the backend is keen on solving the detecting the drift problem. Finally, RTABMap uses the g2o algorithm for the generation of 3D maps.

For implementing RTAB-Map in Gazebo, we used the rtabmap_ros package, which is a ROS wrapper of RTAB-Map. Further, Turtlebot was equipped with an RGB-D camera in order to give ROS topics such sensor_msgs/image/rgb and sensor_msgs/image/depth. Also, the odometry data (position and orientation) ROS topic was made available to publish odometry in nav_msgs/Odometry format. The rtabmap_ros package also requires 2D Lidar data in the form of a topic sensor_msgs/LaserScan, which can be easily generated using depth image from the Kinect itself using depthimage_to_laserscan ros-pkg. Then lastly, rtabmap was started, which subscribes to all the ROS topics mentioned above in order and processes it to give us a RTAB-Map cloud on the topic rtabmap/MapData. The results of the rtabmap cloud visualized in rviz is as shown in Fig. 4.

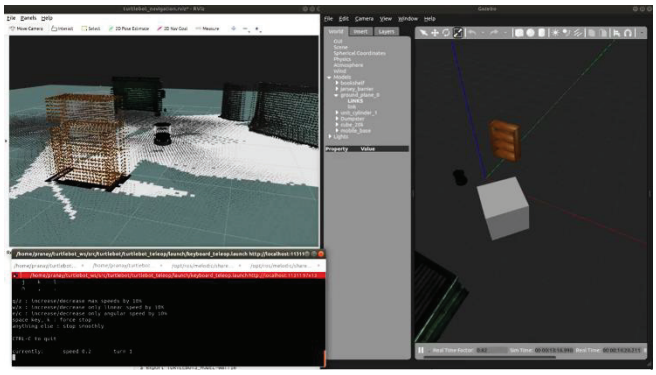

Fig. 4. RTABMap cloud of the simulated Gazebo world visualized in rviz

C. RTABMap implementation on Kinect V2

1) Interfacing the Kinect

The interfacing of the KinectV2 with Ubuntu is done using the libfreenect2 library. Libfreenect2 is basically an open-source driver for the KinectV2 and a C++ API that provides access to the KinectV2 hardware and the colour and depth image streams from the device. libfreenect2 gives us access to the Kinect hardware and its image streams. To have this data usable for ROS or to visualize the RGB-D data as a PointCloud in a 3D space, we need a bridge between libfreenect2 and ROS. This is where the iai_kinect2 package comes into picture.[13]

The iai_kinect2 package contains-

- A calibration tool for the IR sensor of the Kinect to the RGB sensor and the depth measurements.
- A library for depth registration with OpenCL support.
- Bridge between ROS and libfreenect2.
- Viewer for images and PointClouds.

It is a collection of ROS nodes on top of libfreenect2 to make the data that the Kinect produces i.e. PointCloud, available through a set of publishers. In short, a ROS driver for the Kinect. For testing the implementation of RTABMap in real-time, the KinectV2 RGB-D camera was chosen. It was interfaced with ROS by following the methodology as mentioned above.The specifications of the used Kinect V2 camera are described in Table 1:

TABLE I. KINECT V2 SPECIFICATION

Features	Specification
Resolution	RGB camera: 1920 × 1080 IR sensor: 512 × 424
Frame Rate(fps)	RGB camera: 30 IR sensor: 30
Field of view(FOV)	RGB camera: 84.1° × 53.8° IR sensor: 70.6° × 60.0°
Depth sensing Technology	Time of Flight
Depth sensor range	0.5-4.5 m
Voltage	12 V DC
Dimensions	249 × 66 × 67 [mm]
Mass	970g

Fig. 5. Indoor environment

2) Implementing RTABMap

Keeping the Kinect handheld and rotating it 360 degrees about itself so as to cover the entire room once results in the following output (Ref. Fig. 6). We observe, walls/objects that are far away from the Kinect are not recognized by it and thus, points corresponding to them are not generated while objects nearer to it are detected and marked clearly. Another point worth noting is that the 3D map will be much better and clearer if we choose the resolution as HD or QHD rather than SD. Calibrating the KinectV2 before performing the mapping can also improve the quality of the map. The indoor environment in Figure 5 was used for RTABMap implementation.

Authorized licensed use limited to: COLLEGE OF ENGINEERING - Pune. Downloaded on January 11,2022 at 22:42:02 UTC from IEEE Xplore. Restrictions apply.

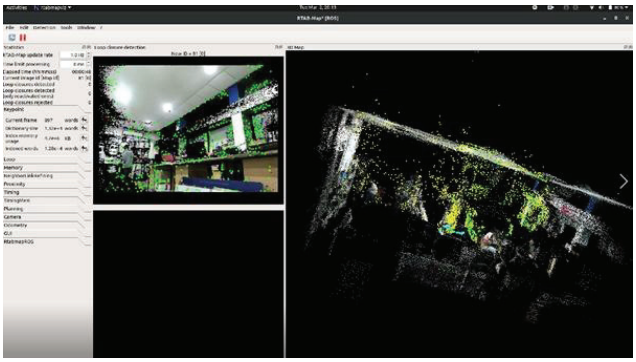

Fig. 6. RTABMap result of a room using the Kinect V2

D. Object Detection in Gazebo

In Computer Vision, object detection is the task of identifying and localizing the presence of an object in a given image or video stream. For object detection, we decided to use YOLO [14] due to its robust performance and relatively accurate results. For implementing YOLO, the darknet framework, which is an open-source neural network framework written in C and CUDA was chosen. This pre-trained model of the convolutional neural network is able to detect 80 classes from the COCO dataset. This model is applied on the following camera ROS topics:

A) camera/rgb/image_raw

B) camera/image_depth_registration/image_raw

The first topic is used for the normal YOLO detection and we apply our model on the video stream of this topic. The results can be seen in Figure 7.

3D bounding boxes are implemented using the ROS package darknet_ros_3d [15]. Once we have the 2d detection, we get the depth points corresponding to the bounding box. Then this is used for creating the 3d bounding boxes. This is published on the topic.

A) /darknet_ros_3d/markers

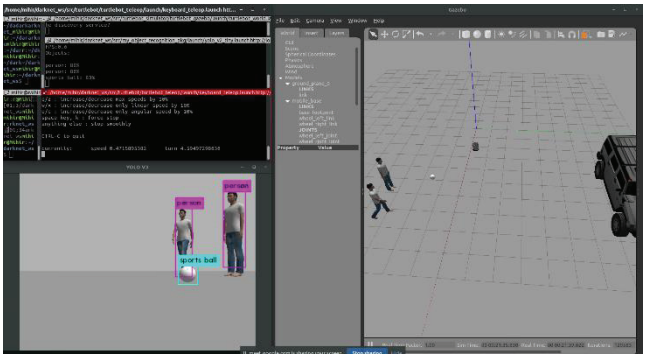

Fig. 7. 2D object detection in Gazebo using the YOLO v3 tiny object detection model

YOLO is an algorithm that makes use of Neural Networks to perform simultaneous or real-time object detection. It is a state-of-the-art algorithm based on regression that predicts classes for the whole image. The model we chose is YOLO v3 Tiny[9] as it provided a very good FPS and respectable accuracy[11]. The depth of the convolutional layer in YOLO v3-Tiny is smaller than YOLO v3. Therefore, there is a significant increase in the speed at the cost of detection. To withdraw class information, convolution layers and max-pooling layers are employed in the feedforward ordering. For final detections, it ignores the bounding boxes whose objectness score is not up to the mark.

IV. Experimental Results

We were successful in implementing the V-SLAM using the RTAB-Map ros package on simulation as well as on the hardware setup of Kinect v2. It was observed that the reconstruction of the 3D map using the hardware setup of Kinect had some noise, and at places which lacked feature descriptors, RTABMap was not able to accurately perform loop closure. On the other hand, simulation results of RTAB-Map were relatively better, helping us perceive the surrounding virtual environment accurately thereby paving the foundation for autonomous navigation. problem.

The next important result is that the pre-trained CNN detector using YOLO v3 tiny object detection model was successful in determining a few test object classes such as person, ball, car and table. In Figure 9, the left image represents the detected 2D image, whereas the one in the right visible in RVIZ is the 3D PointCloud of the corresponding image. The one at the bottom left represents the corresponding depth image.

Thus, the result showing the detected object is shown in Figure 8 and its corresponding accuracy of detection of a particular object class is shown in Figure 9.

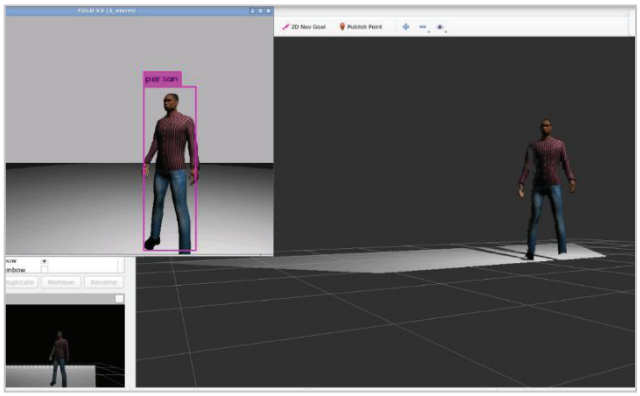

Fig. 8. Rviz visualization of object detection using the YOLO v3 tiny object detection model

Fig. 9. Accuracy of the detected object

Also, the insertion of 3D-bounding boxes on the observed class object was done in Gazebo, and the results seemed to be promising (Ref. Fig. 10) and can be further used in the Hardware implementation.

Authorized licensed use limited to: COLLEGE OF ENGINEERING - Pune. Downloaded on January 11,2022 at 22:42:02 UTC from IEEE Xplore. Restrictions apply.

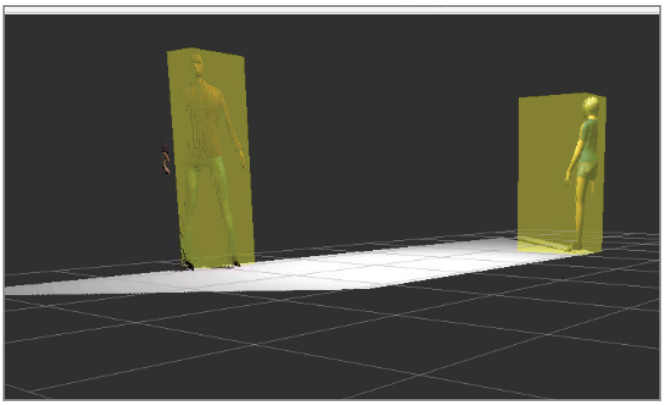

Fig. 10. 3D-bounding boxes on the observed class object

These results were achieved on our computing platform that consisted of the AMD Ryzen 5 4600H processor with NVIDIA GTX 1650 as the GPU (Graphics Processing Unit) and 16GB RAM.

V. Conclusion

We have successfully implemented RTAB-Map alongside object detection in the ROS environment using Turtlebot. In our experiments, we found that RTAB-Map has provided good 3D mapping in the simulation environment. Also, it provided satisfactory results using a Kinect V2 in the real world. This was due to the limited range of the Kinect and the depth map not being proper after about 3-4 m. The aim of implementing object detection simultaneously with the SLAM map generation was achieved, which helps in increasing the robot utility. Pretrained YOLO v3 tiny network was used as the CNN for localizing and classifying a few test objects in the household indoor environment. Training the model on our own dataset for some other specific classes can also be done separately and integrated easily with this approach. The simulation results show that the proposed approach can be used to obtain the map of the indoor environment using the RTABMap approach, Additional semantic information about the obstacles, objects of interest can also be obtained at the same time which enhances the robot locomotion capability. The ROS environment plays a pivotal role in combining both. Thus, we can see that the proposed RTABMap SLAM approach accompanied by object detection using YOLO V3-Tiny CNN model helps us to achieve robust 3D perception needed for performing complex tasks during autonomous navigation.

Although the work is sufficient for enhancing the perception capabilities during the autonomous navigation in a simulation environment, there are still many areas of optimization which can be incorporated further. Some of them are as follows: More experiments need to be conducted on the YOLO v3 tiny model in order to test its rigorousness in different complex real-time environments. Although it is widely accepted that when it comes to real-time object detection YOLO tends to outperform all other methods, we cannot ignore the possibility of other models being better for our use case. Hence models such as Faster RCNN can be further implemented and tested.

Acknowledgment

This work was supported partly by the Robotics and Automation Laboratory, COEP. We would like to thank the faculty advisor of Robotics and Automation Laboratory Dr. S.S Ohol for his constant support. We would also like to thank all members of the lab for their help and suggestions throughout the completion of the project.

References

[1] Yong-bao Ai, Ting Rui, Xiao-qiang Yang, Jia-lin He, Lei Fu, Jian-bin Li, Ming Lu, Visual SLAM in dynamic environments based on object detection, Defence Technology, 2020, ISSN 2214-9147, https://doi.org/10.1016/j.dt.2020.09.012.

[2] Z. Kong and Q. Lu, "A brief review of simultaneous localization and mapping," IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society, 2017, pp. 5517-5522, doi: 10.1109/IECON.2017.8216955.

[3] Taketomi, T., Uchiyama, H. & Ikeda, S. Visual SLAM algorithms: a survey from 2010 to 2016. IPSJ T Comput Vis Appl 9, 16 (2017). https://doi.org/10.1186/s41074-017-0027-2

[4] Kowalewski, S., Adrian Llopart Maurin, and Jens Christian Andersen. "Semantic mapping and object detection for indoor mobile robots." IOP Conference Series: Materials Science and Engineering. Vol. 517. No. 1. IOP Publishing, 2019.

[5] P. Maolanon, K. Sukvichai, N. Chayopitak, and A. Takahashi, "Indoor room identify and mapping with virtual based slam using furnitures and household objects relationship based on cnns," in 2019 10th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES). IEEE, 2019, pp. 1–6.

[6] Ding, Xintao & Luo, Yonglong & Yu, Qingying & Li, Qingde & Cheng, Yongqiang & Munnoch, Robert & Xue, Dongfei & Cai, Guorong. (2017). Indoor object recognition using pre-trained convolutional neural network. 1-6. 10.23919/IConAC.2017.8081986.

[7] ROBOTIS Emanual, https://emanual.robotis.com/docs/en/platform/

[8] M. Labbe and F. Michaud, "Appearance-based loop closure detection for online large-scale and long-term operation," IEEE Transactions on Robotics, vol. 29, no. 3, pp. 734–745, Jun. 2013. [Online]. Available: https://doi.org/10.1109/tro.2013.2242375

[9] Xiaotian Gong, "An improved method of Tiny YOLOV3" et al 2020 IOP Conf. Ser.: Earth Environ. Sci. 440 052025

[10] Turtlebot Homepage, https://www.turtlebot.com/about/

[11] X. Mengcong and M. Li, "Object Semantic Annotation Based on Visual SLAM," 2021 Asia-Pacific Conference on Communications Technology and Computer Science (ACCTCS), 2021, pp. 197-201, doi: 10.1109/ACCTCS52002.2021.00047.

[12] Labbé, M., and F. Michaud. 2019. "RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation." J. Field Rob. 36 (2): 416–446. https://doi .org/10.1002/rob.21831

[13] T. Wiedemeyer, "IAI Kinect2," https://github.com/code-iai/iai_kinect2, Institute for Artificial Intelligence, University Bremen, 2014 – 2015, accessed June 12, 2015.

[14] Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement." arXiv preprint arXiv:1804.02767 (2018).

[15] M. Bjelonic. YOLO ROS: Real-time object detection for ROS. https://github.com/leggedrobotics/darknet_ros, 2016–2018.

[16] Xuanbo Wang, "Autonomous Mobile Robot Visual SLAM Based on Improved CNN Method" 2018 IOP Conf. Ser.: Mater. Sci. Eng. 466 012114

[17] P. Adarsh, P. Rathi and M. Kumar, "YOLO v3-Tiny: Object Detection and Recognition using one stage improved model," 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), 2020, pp. 687-694, doi: 10.1109/ICACCS48705.2020.9074315.

Authorized licensed use limited to: COLLEGE OF ENGINEERING - Pune. Downloaded on January 11,2022 at 22:42:02 UTC from IEEE Xplore. Restrictions apply.