Indoor Navigation of a Service Robot Platform Using Multiple Localization Techniques Using Sensor Fusion

Ruchik Mishra
Department of Mechanical Engineering
BITS Pilani, Hyderabad Campus
Hyderabad, India
e-mail: ruchikofficial@gmail.com

C. Vineel
Undergraduate Student, Department of Electrical and Electronics Engineering
BITS Pilani, Hyderabad Campus
Hyderabad, India
e-mail: f20160348@hyderabad.bits-pilani.ac.in

Dr. Arshad Javed
Department of Mechanical Engineering
BITS Pilani Hyderabad Campus
Hyderabad, India
e-mail: arshad@hyderabad.bits-pilani.ac.in

Abstract—**The adversities of navigating in an unknown environment are challenging for autonomous robots as they not only rely on the quality of the sensor data but also on the type of environment. The errors can cumulatively result in a catastrophic failure of the purpose. This paper proposes a way for an autonomous agent to navigate in an indoor structured environment by minimizing the errors. A 2 DoF differential drive robot is developed and the map-building process is done wirelessly in the teleoperation mode using bluetooth. In the navigation mode, various approaches are tested for localizing such as the Extended Kalman Filter, Unscented Kalman Filter and a particle filter called the Adaptive Monte Carlo Localization which uses KLD (Kullback-Leibler Distance) as the sampling method. The designed navigation system is accurate and also time saving, thereby increasing the efficiency of the robotic system. All the implementation has been done using the Robotic Operating System because of the available packages that satisfy the technical aspects of the paper and further changes have been made in them to make them suitable for use in this scenario.**

Keywords-ROS; indoor navigation; RGB-D camera; sensor fusion

I. INTRODUCTION

With the sophistication in technology and the boost of automation in the past decades, the use of robots has increased for obvious reasons. They can not only do tasks faster but also ensure precision, and access areas that may not be suitable or safe for humans. To fulfill this, the robot should have mobility i.e. it should be able to move, and any entity capable of moving (more precisely locomotion) requires the need to navigate in the environment it wants to operate in. As far as navigation is concerned, three major points come into picture: the current location of the body, the final goal point that it has to reach and the way it has to follow to get there. In case of robots, these three points can be best described as self-localization, map-building and map-interpretation, and path planning [1]. The most common form of localization that we find in case of mobile robots is, by using the wheel encoder values for odometry. But this approach may be erroneous as it may have a number of errors like friction, mechanical errors etc [2, 3]. So, a number of other approaches have been adopted. One of them is an iterative approach called Simultaneous Localization and Mapping (SLAM). In this case, a map is not needed in advance, but is generated and incremented as the robot explores new locations in its operating environment. Cummins and Newman have used an appearance based probabilistic approach to identify places and append new ones into the map taking into account the solution of perceptual aliasing with a linear complexity [4]. Kohlbrecher et al. uses an online approach of learning of grid maps using a LIDAR [5]. Dissanayake et al. has discussed a map management strategy by deleting landmarks from the map without compromising the consistency of the SLAM algorithm [6]. Chung et al aims at improving the accuracy of dead-reckoning in mobile robotics using fusion of sensor data [7]. The sensor data used were odometry and fiber optic gyroscope and fused using an indirect Kalman filter. The use of particle filters like the Monte Carlo Localization using KLD sampling approach has been used by Mishra and Javed [2], where a turtlebot model is simulated to navigate to its final goal point using amcl package of ROS. A recent approach adopted by Zhu et al. discusses a target driven visual navigation using a deep reinforcement learning approach [8]. The problem of navigation is dealt with using only visual inputs. To deal with the problems of the deep reinforcement learning related problems such as lack of capability to generalize the new goals, an actor-critic model is used whose policy is a function of both the goal and the current state. To address the second issue of data inefficiency,

978-1-7281-6139-6/20/$31.00 ©2020 IEEE

an environment with high-quality 3D scenes and a physics engine are provided by the AI2-THOR framework.

In this paper, a successful demonstration of mapping of the indoor static operating environment of the robot is presented which uses a low cost RGB-D camera to navigate the robot through the operating environment from the starting point to the goal using multiple localization techniques using sensor fusion.

The paper has been arranged as follows. Section two presents the system outline explaining both the hardware and the software framework used for the robot followed by the section three, which explains the mapping procedure followed by section four to explain the navigational part. Section five presents the results and discussions followed by the conclusion in section six.

II. System Outline

The robot used is a modest and balanced sized differential drive robot which has been equipped with reasonable hardware to serve the purpose of autonomous navigation of our target indoor environment. The robot as a whole has been shown in the Figure 1(a). It is equipped with an omni wheel at the front, two DC quadrature encoder motors (shown in Figure 1(b)) with a rated 154 rpm. Further, two microcontrollers have been used. One of them is used for fetching the wheel encoder inputs and the other is used for the motor driver and the motors. The robot is teleoperated wirelessly for the purpose of mapping and so it has also been equipped with an HC-05 Bluetooth module. The module is connected with one of the microcontrollers on board and is paired with the laptop that has been used in this project for all the computational purposes. The bluetooth communication is also used to control a servo motor that has been attached at the base of the Microsoft Kinect XBOX 360. In order to power the electronic systems on board, a Lithium ion battery of rating 12V and 5000mAh is used. The battery is connected to an LM2596 buck convertor to step down the voltage. The framework used in this paper is the Robotic Operating System, commonly known as ROS which runs parallel with the existing operating system (in our case Linux (Ubuntu 16.04 LTS)).

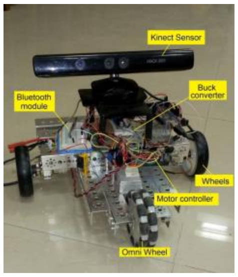


Figure 1(a)Robot

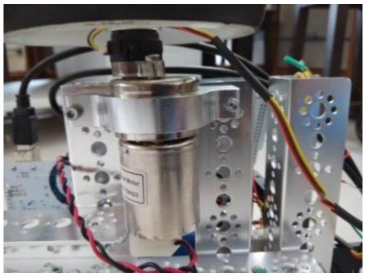


Figure 1(b). Quadrature encoder motor

III. Mapping

Acquisition of a map is a prerequisite for any mobile autonomous agent that intends to navigate in an unknown environment. Although mapping of an indoor environment is a not a new concept [9, 10], only the technology to do mapping has become cheaper over time with the arrival of the Microsoft Kinect which is capable of acquiring a 3-dimensional map of the unknown environment [11, 12].

The robot operates in two different modes. In this paper, the first is the teleoperation mode in which the robot acquires the map of its operating indoor environment with static obstacles. The second mode of operation is navigation; in which robot actively navigates in the mapped environment.

A. Teleoperation Mode

The robot uses wireless communication via Bluetooth for the purpose of teleoperation through which the map of our target environment is acquired. Here, a modified method of teleoperation is used based on an available approach [13]. A direct conversion of the analogue values of the motors of the int data type into the string data type would mean that a value like -255 which requires two bytes in the int data type would require four bytes (considering -255 separately as ‘ - ’ ,’2’ ,’5’ ,’5’). In order to increase the speed and efficiency of transmission, a fixed byte structure was adopted with the help of ctypes which is a foreign function library for python [14] .The analog values are converted into c_ubyte type of ctypes python library packed into a 6 byte string and then transmitted through bluetooth as demonstrated in Figure 2. The first byte is a confirmation bit that is needed to verify whether the data received is from the right computer or not and also makes sure that corrupted byte streams are not received.

1. Confirmation Byte
2. Motor 1 Low Byte
3. Motor 1 High Byte
4. Motor 1 Low Byte
5. Motor 1 High Byte
6. Servo Angle Byte

Figure 2. Byte transmission via Bluetooth.

Finally, the data is retrieved in a similar manner in which it was packed. The high and low bits are combined through 8 bit logical shifts and logical or operation and written to the motor and the servo.

B. Mapping Using Kinect Sensor

The OpenNI node has been used to process raw data captured from the kinect. However, this data needs to be further processed so that it can be easily used to get the laser scan and the odometry messages since in this paper navigation requires the maps as laser scans. For this reason, a ROS based approach called depthimage_to_laserscan has been used. As the name itself suggests, the package converts a depth image to 2D laser scans using a lazy subscribing which means that it will not subscribe to an image or a camera_info unless there is a subscriber for scan [15]. The conversion of the depth image taken by the Kinect into the 2D laser scan is given by the equations (1) and (2), [16]:

$$X_{i,j} = (j - \frac{width}{2}) \times (\frac{320}{width}) \times M \times Z_{i,j} \tag{1}$$

$$Y_{i,j} = (j - \frac{height}{2}) \times (\frac{320}{height}) \times M \times Z_{i,j} \tag{2}$$

Here *'i'* and *'j'* represent the row and column number of the Z array (the Z coordinate) respectively and *X, Y* are the corresponding X and Y coordinates of the corresponding Z coordinate. M is the NUI Camera Depth Image to Skeleton Multiplier Constant. The Z coordinates of the depth pixels are stored in an array equation (3) which is a 1x480 array since it takes the minimum of the Z values from each column of the Z array.

$$Z_{i,j} = \min(Z_{0,j}, Z_{1,j}, Z_{2,j}, Z_{3,j}, ...Z_{479,j}) \tag{3}$$

$X_{i,j}$ represents the X coordinate of the corresponding Z coordinate of the object in front of the camera which is stored in a 640×1 array where the corresponding Z value [16] is the nearest Z coordinate obtained from the equation (3). The geometrical relationship and the distance of the robot from the obstacle is shown in Figure 3, and represented in equation (4)[17]. The distance of the obstacle is calculated taking into consideration the design of the robot, as there is some distance between the Kinect and the omni wheel of the robot. The following equation gives the first incident of the obstacle with the robot:

$$D = Z \times \frac{\sin(90° - \theta - \phi)}{\cos(\phi)} - L \tag{4}$$

where, L is the horizontal distance between the Kinect sensor and the front end of the robot. The depth image node subscribes to the whole depth image and publishes the mean depths of 'n' pixel rows centered at the mth pixel specified by some parameters, and leaving these to the default parameters showed the appearance of false obstacles such as some parts of the ground itself as obstacle. To solve this problem, a threshold value is set according to the requirement of the robot. All the values that are below the given threshold height are neglected as obstacles. The geometrical Figure 3 and the equation (5) show the ground removal algorithm [17].

$$P_g = \{(X, Z) \mid Z \geq H \times \frac{\cos(\phi)}{\sin(\phi + \theta)} - c\} \tag{5}$$

where 'c' is the ground tolerance.

The map_server package's map saver node is run when the map of a section is obtained and then the gmapping node is reset to capture another portion of the room till the room is mapped completely and visualized on a 3D visualization tool for ROS called Rviz.

IV. Navigation

Navigation is the ability by virtue of which the robot moves from the current location of its operating environment to its desired goal point. In mobile robotics, navigation, as a research area has paced up within a decade [18] since these mobile robots are being used extensively for a variety of purposes in various fields starting from rescue operations to warehouses [12,19,20,21]. The following sections deal with the navigational approach followed by the robot, after acquisition of the map using teleoperation.

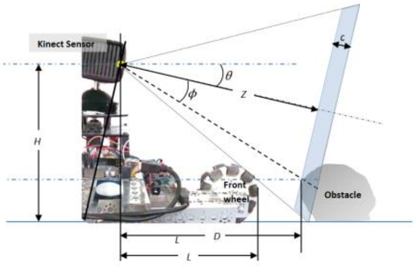


Figure 3. Estimation of the obstacle depth.

A. Kalman Filters

The most common form in which a mobile robot localizes itself and navigates in its operating environment is through the wheel encoder data that it receives. However, this data is prone to errors due to slipping of the robot, friction, etc. This problem can be solved by fusion of data from multiple sensors and then applying a filter to get the optimal solution. This is where a Kalman Filter comes into the picture. Since the problem is a non-linear one, Extended Kalman Filter (EKF) is used. The system of equations is given by (7) and (8):

$$X_k = f(X_{k-1}, u_k) + W_k \tag{7}$$

$$Y_k = g(X_k) + V_k \tag{8}$$

where, $f(X_{k-1}, u_k)$ and $g(X_k)$ are nonlinear functions, W_k is the process noise like friction or slipping of the wheel of the robot and V_k is the measurement noise. Among one of the practical difficulties to implement EKF is, its non-suitability for highly non-linear functions and it also cannot be used for non-differentiable functions. To solve this problem, the same procedure is repeated using the Unscented Kalman Filter (UKF). UKF selects a minimal set of sample points evenly distributed around the mean of the Gaussian distribution (as the state distribution) and then these samples points are passed through any given nonlinear function [22,23]. The mean and covariance of the newly transformed points of the nonlinear function gives the new state estimate by computing the empirical Gaussian distribution obtained after the non-linear transformation through UKF.

B. Adaptive Monte Carlo Localization (AMCL)

This paper uses an approach in which an adaptive sampling is used for MCL. This is done through a ROS package called amcl (Adaptive Monte Carlo Localization). It uses a sampling method called Kullback-Leibler divergence (KLD). The aim here is to bind the error that the particle filter (in our case MCL) has caused by its sample based representation. The number of the samples depends on the concentration of the density in the state space. The number of samples will be more, if more of the state space has to be covered and vice-versa [24]. The ROS package amcl takes laser scans and the output is the pose estimate of the robot [25]. The pose array which is a set of points where the robot can be probably positioned is determined by the laser scan sensor by comparing the scan message obtained with the points on the map using amcl to determine the probable positions of the robot. For initial attempts, 100 to 2000 particles are used here, to represent the position and orientation of the robot, which is shown as a polygon and the particles are shown in red in the Figure 4.

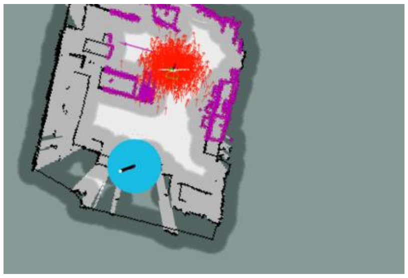


Figure 4. Use of particle filter (particles in red and the robot as a polygon).

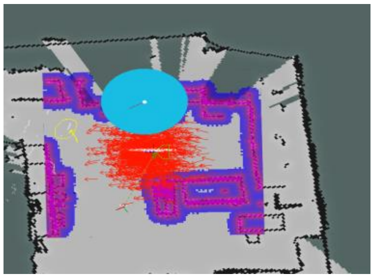


Figure 5. False obstacles marked in yellow.

The arrow helps to set the orientation of the robot through Rviz. Once the pose estimate is done, further the path has to be obtained for navigating the robot to its final goal point. For this the global and local cost maps play their role. The local cost map is used for making local path plans. It takes into account the immediate surroundings of the robot and hence can be used for immediate obstacle avoidance whereas the global cost map takes into account the static map generated at the time of mapping and locates the obstacles in the global operating environment. The areas colored in pink represent the local cost map in the Figure 4.

V. Results and Discussions

As discussed earlier, the results related to false object and localization error is presented in this Section.

A. *False Object Detection*

The 2D laserscan generated using the depthimage_to_laserscan package, without the ground removal algorithm has been shown in the Figure 5 where the false obstacle has been prominently highlighted in yellow.

Figure 6 shows that the false obstacle no longer exists on the map after the threshold for the ground tolerance has been set as discussed in the Section three.

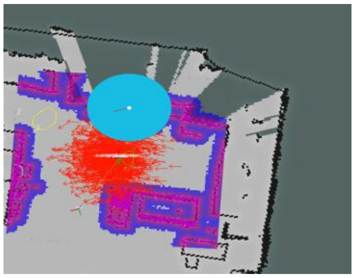


Figure 6. Corrected image with removal of false obstacles.

The map generated after the entire process of teleoperation and ground removal has been shown in Figure 7.

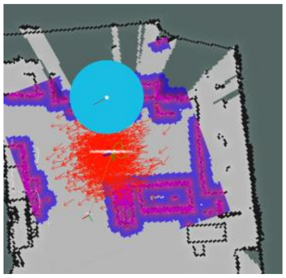


Figure 7. The map of the robot's operating environment.

B. *Navigation Error Handling*

Figure 8 shows the graphs obtained when the EKF is being used for the process of localization and navigation by fusing the visual odometry node and the encoder odometry node.

As discussed earlier, the Unscented Kalman Filter has an advantage over EKF; so Figure 9 shows the results of the filtered data that has been obtained by using the UKF.

Leagends:
🟥 (red): Visual odometry, 🟦 (blue): Encoder
data 🟩 (green):Filtered data

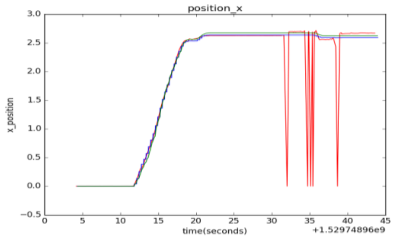


Figure 8(a). Variation of position in x-direction.

C. *Navigational Proof*

The path of the robot has been shown as visualized in rviz in the Table 1. Each step from the path starting from the initial to the final goal has been shown in four steps where actual position of the robot has also been shown corresponding to the image of the position of the robot in rviz.

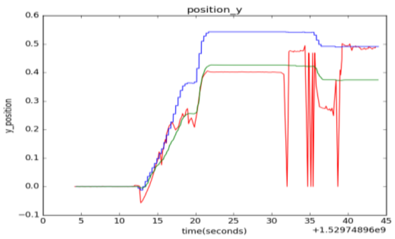


Figure 8(b). Variation of position in y-direction.

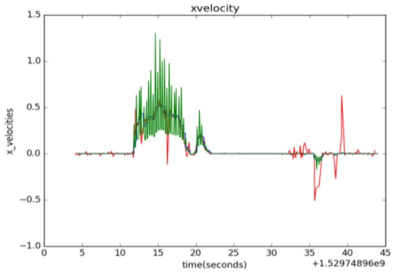


Figure 8(c). Velocity variations in x-direction with respect to time in seconds.

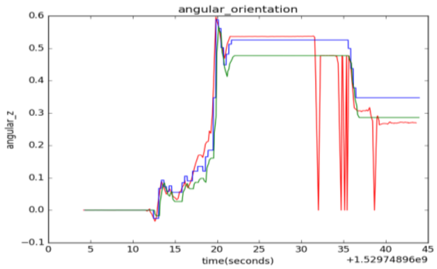


Figure 8(d). Variation of angular position.

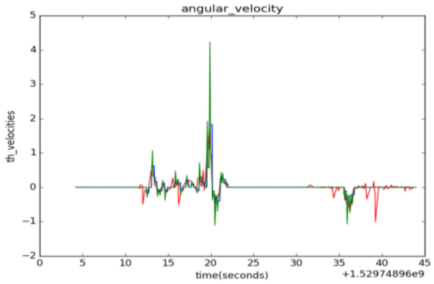


Figure (e) Variation of angular velocity.

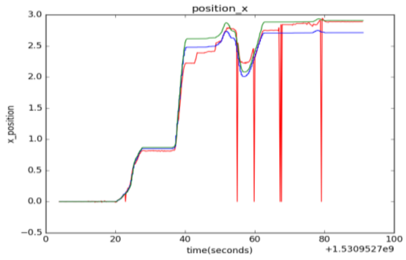


Figure 9(a). Variation of position in x-direction.

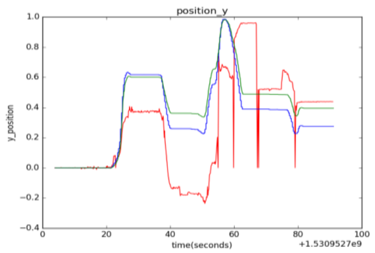


Figure 9(b). Variation of position in y-direction.

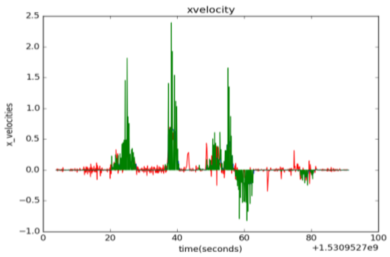


Figure 9(c). Velocity variations in x-direction with respect to time in seconds.

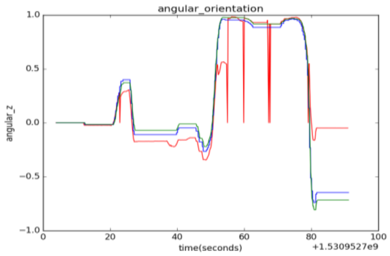


Figure 9(d). Variation of angular position.

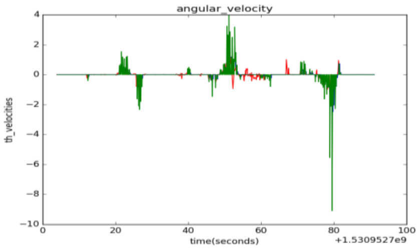
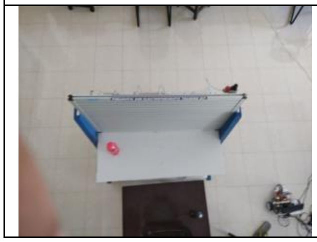
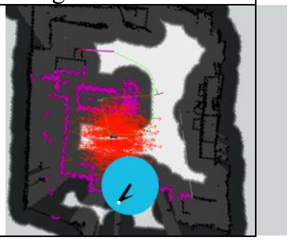
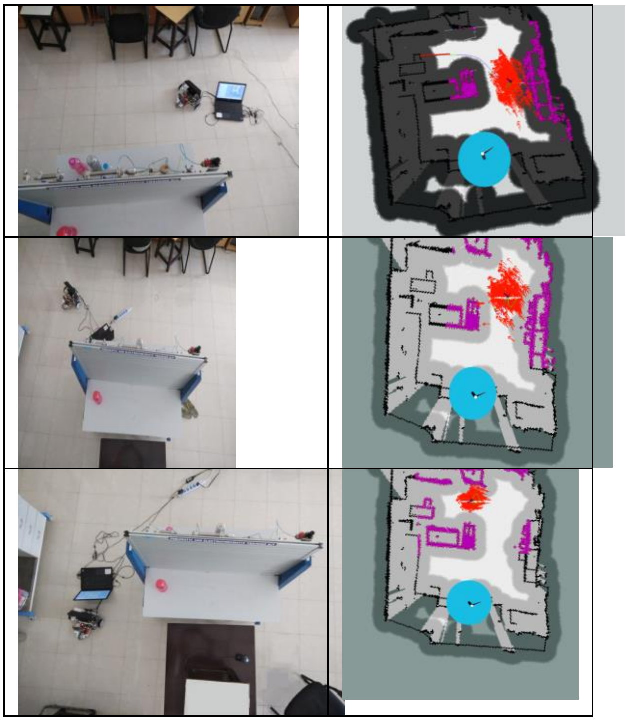


Figure 9(e). Variation of angular velocity.

TABLE 1. NAVIGATION OF THE ROBOT

Actual Navigation	Corresponding navigation in rviz
	



VI. Conclusion

In this paper we have proposed a solution for the autonomous navigation of a robot inside an indoor structured environment with static obstacles. As discussed earlier, the applications of an autonomous agent in indoor environments like warehouses, industries [26], hospitals are becoming prominent because of the advantages they have. The purpose of this research was motivated by the same and so the content of the paper has been designed keeping in mind the requirements of such an agent. The cognizance of the earlier work [2] has been a key factor in developing the current robot that we have.

References

[1] Oliver A, Kang S, Wünsche BC, MacDonald B. Using the Kinect as a navigation sensor for mobile robotics. InProceedings of the 27th conference on image and vision computing New Zealand 2012 Nov 26 (pp. 509-514). ACM.

[2] Mishra R, Javed A. ROS based service robot platform. In2018 4th International Conference on Control, Automation and Robotics (ICCAR) 2018 Apr 20 (pp. 55-59). IEEE.

[3] Eriksson, T. and Ragnerius, E., 2012. Autonomous robot navigation using the utility function method and microsoft kinect. M.S. thesis, Chalmers University of Technology, Goteborg, Sweden,2012.R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[4] Cummins M, Newman P. FAB-MAP: Probabilistic localization and mapping in the space of appearance. The International Journal of Robotics Research. 2008 Jun;27(6):647-65.

[5] Kohlbrecher S, Von Stryk O, Meyer J, Klingauf U. A flexible and scalable slam system with full 3d motion estimation. InSafety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on 2011 Nov 1 (pp. 155-160). IEEE.

[6] Dissanayake G, Durrant-Whyte H, Bailey T. A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. InRobotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on 2000 (Vol. 2, pp. 1009-1014) IEEE.

[7] Chung H, Ojeda L, Borenstein J. Sensor fusion for mobile robot dead-reckoning with a precision-calibrated fiber optic gyroscope. InRobotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on 2001 (Vol. 4, pp. 3588-3593). IEEE.

[8] Zhu Y, Mottaghi R, Kolve E, Lim JJ, Gupta A, Fei-Fei L, Farhadi A. Target-driven visual navigation in indoor scenes using deep reinforcement learning. InRobotics and Automation (ICRA), 2017 IEEE International Conference on 2017 May 29 (pp. 3357-3364). IEEE.

[9] Durrant-Whyte H, Bailey T. Simultaneous localization and mapping: part I. IEEE robotics & automation magazine. 2006 Jun;13(2):99-110.

[10] Darmanin RN, Bugeja MK. Autonomous Exploration and Mapping using a Mobile Robot Running ROS. InICINCO (2) 2016 Jul 29 (pp. 208-215).

[11] Hermans A, Floros G, Leibe B. Dense 3d semantic mapping of indoor scenes from rgb-d images. InRobotics and Automation (ICRA), 2014 IEEE International Conference on 2014 May 31 (pp. 2631-2638). IEEE.

[12] Suarez J, Murphy RR. Using the Kinect for search and rescue robotics. InSafety, Security, and Rescue Robotics (SSRR), 2012 IEEE International Symposium on 2012 Nov 5 (pp. 1-2). IEEE.

[13] ros-teleop/teleop_twist_keyboard [Internet]. GitHub. 2019 [cited 24 January 2019]. Available from: https://github.com/ros-teleop/teleop_twist_keyboard/blob/master/teleop_twist_keyboard.py

[14] ctypes — A foreign function library for Python — Python 3.7.2 documentation [Internet]. Docs.python.org. 2019 [cited 24 January 2019]. Available from: https://docs.python.org/3/library/ctypes.html

[15] depthimage_to_laserscan - ROS Wiki [Internet]. Wiki.ros.org. 2019 [cited 24 January 2019]. Available from: http://wiki.ros.org/depthimage_to_laserscan

[16] Kamarudin K, Mamduh SM, Shakaff AY, Saad SM, Zakaria A, Abdullah AH, Kamarudin LM. Method to convert Kinect's 3D depth data to a 2D map for indoor SLAM. InSignal Processing and its Applications (CSPA), 2013 IEEE 9th International Colloquium on 2013 Mar 8 (pp. 247-251). IEEE.

[17] Drwięga M, Jakubiak J. A set of depth sensor processing ROS tools for wheeled mobile robot navigation. Journal of Automation Mobile Robotics and Intelligent Systems. 2017;11.

[18] Megalingam RK, Teja CR, Sreekanth S, Raj A. ROS based Autonomous Indoor Navigation Simulation Using SLAM Algorithm.

[19] Hirose S, Fukushima EF. Development of mobile robots for rescue operations. Advanced Robotics. 2002 Jan 1;16(6):509-12.

[20] Murphy RR, Kravitz J, Stover SL, Shoureshi R. Mobile robots in mine rescue and recovery. IEEE Robotics & Automation Magazine. 2009 Jun;16(2).

[21] D'Andrea R. Guest editorial: A revolution in the warehouse: A retrospective on kiva systems and the grand challenges ahead. IEEE Transactions on Automation Science and Engineering. 2012 Oct;9(4):638-9.

[22] Understanding Kalman Filters [Internet]. In.mathworks.com. 2019 [cited 24 January 2019]. Available from: https://in.mathworks.com/videos/series/understanding-kalman-filters.html

[23] Wan EA, Van Der Merwe R. The unscented Kalman filter for nonlinear estimation. InAdaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000 2000 (pp. 153-158). IEEE.

[24] Fox D. KLD-sampling: Adaptive particle filters. InAdvances in neural information processing systems 2002 (pp. 713-720).

[25] amcl - ROS Wiki [Internet]. Wiki.ros.org. 2019 [cited 24 January 2019]. Available from: http://wiki.ros.org/amcl

[26] Hu H, Gu D. Landmark-based navigation of industrial mobile robots. Industrial Robot: An International Journal. 2000 Dec 1;27(6):458-67.