International Journal of Pure and Applied Mathematics
Volume 118 No. 7 2018, 199-205
ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

ROS based Autonomous Indoor Navigation Simulation Using SLAM Algorithm

Rajesh Kannan Megalingam, Chinta Ravi Teja, Sarath Sreekanth, Akhil Raj
Department of Electronics and Communication Engineering, Amrita Vishwa Vidaypeetham, Amritapuri, Kerala, India.

***Abstract*—In this paper, we are checking the flexibility of a SLAM based mobile robot to map and navigate in an indoor environment. It is based on the Robot Operating System (ROS) framework. The model robot is made using gazebo package and simulated in Rviz. The mapping process is done by using the GMapping algorithm, which is an open source algorithm. The aim of the paper is to evaluate the mapping, localization, and navigation of a mobile robotic model in an unknown environment.**

Keywords—Gazebo; ROS; Rviz; Gmapping; laser scan; Navigation; SLAM; Robot model; Packages.

I. INTRODUCTION

In the modern world, the need for machines are increasing due to the probability of making mistakes by the robot is less. The research and application of robotics are from healthcare to artificial intelligence. A robot can't understand the surroundings unless they are given some sensing power. We can use different sensors like LIDAR, RGB-D camera, IMU (inertial measurement units) and sonar to give the sensing power. By using sensors and mapping algorithms a robot can create a map of the surroundings and locate itself inside the map. The robot will be continuously checking the environment for the dynamic changes happening there.

Our aim was to build an autonomous navigation platform for indoor application. In this paper, we are checking the efficiency of a SLAM (Simultaneous Localization and Mapping) based robot model implemented in ROS (Robot Operating System) by measuring the travel time taken by the robot model to reach the destination. The test is done in a virtual environment created by Rviz. By placing different dynamic obstacles for different destinations in the map, the travel time is measured.

II. MOTIVATION

Working with the robots need a lot of sensors and every process needs to be handled in real time. To use the sensors and actuators which needs to be updated every 10-50 milliseconds we need a type of operating system that gives this kind of privilege. Robot Operating System (ROS) provides us with the architecture to achieve this. ROS is open source and there are a lot of codes available from good research institutes which one can readily use and implement in their own projects. Further robot's engineers earlier lacked a common platform for collaboration and communication which delayed the adoption of robotic butlers and other related developments. The robotic innovation has quickly paced up since last decade with the advent of ROS wherein the engineers can build robotic apps and programs. Robot navigation is a very wide topic which most of the researchers are concentrating in the field of robotics. For a mobile robot system to be autonomous, it has to analyze data from different sensors and perform decision making in order to navigate in an unknown environment. ROS helps us in solving different problems related to the navigation of the mobile robot and also the techniques are not restricted to a particular robot but are reusable in different development projects in the field of robotics.

III. RELATED WORKS

In the research paper [1], the Authors use ROS with agmapping algorithm to localize and navigate. Gmapping algorithm uses laser scan data from the LIDAR sensor to make the map. The map is continuously monitored by OpenCV face detection and corobot to identify human and navigate through the working environment. The authors of research paper [2] explain about 2 cooperative robots which work based on ROS, mapping, and localization. These robots are self-driving and working in unknown areas. For this project also the algorithm used is SLAM. Here the main tasks of the robots are to pick up three block pieces and to arrange them in a predetermined manner. With the help of the ROS, they made robots for this purpose. In the research paper [3], the Authors created a simulation of the manipulator and illustrated the methods to implement robot control in a short time. Using ROS and gazebo package, they build a model of pick and place robot with 7 DOF. They managed to find a robot control which takes less time. A research paper [5] compares 3 SLAM algorithms core SLAM, Gmapping, and Hector SLAM using

simulation. The best algorithm is used to test unmanned ground vehicles(UGV) in different terrains for defense missions. Using simulation experiments they compared the performance of different algorithms and made a robotic platform which performs localization and mapping. The authors of the research paper [6], made a navigation platform with the use of automated vision and navigation framework, With the use of ROS, the open source GMapping bundle was used for Simultaneous Localization and Mapping (SLAM). Using this setup with rviz, the turtlebot 2 is implemented. Using a Kinect sensor in place of laser range finder, the cost is reduced. The journal [9], deals with indoor navigation based on sensors that are found in smart phones. The smartphone is used as both a measurement platform and user interface. The Author of the journal [10] implemented a 6-degree of freedom (DOF) pose estimation (PE) method and an indoor wayfinding system for the visually impaired. The floor plane is extracted from the 3-D camera's point cloud and added as a landmark node into the graph for 6-DOF SLAM to reduce errors. roll, pitch, yaw, X, Y, and Z are the 6 axes. The user interface is through sound. Journal [11] explains why the indoor environment is difficult for an autonomous quadcopter. Since the experiment is done indoor they couldn't use GPS, they used a combination of a laser range finder, XSens IMU, and laser mirror to make 3-D map and locate itself inside it. The quadcopter is navigating using SLAM algorithm.In paper [12] the authors describe fixed path algorithm and characteristics of the wheelchair which uses this with the help of simulation techniques. The authors of paper [13] explain about an auto navigation platform made in Arduino and the use of ani2c protocol to interface components like adigital compass and a rotation encoder to calculate the distance. In the paper [14], using Fuzzy toolbox in Matlab the authors created an autonomous mobile robot and uses the robot for path planning. 24 fuzzy rules on the robot are carried out. The authors of the paper [15], creates an object level mapping of an indoor space using RFID ultra-high frequency passive tags and readers. they say the method is used to map a large indoor area in a cost-effective manner.

IV. System

A. ROS

Robotic Operating System (ROS) is a free and open-source and one of the most popular middlewares for robotics programming. ROS comes with message passing interface, tools, package management, hardware abstraction etc. It provides different libraries, packages and several integration tools for the robot applications. ROS is a message passing interface that provides inter-process communication so it is commonly referred as middleware. There are numerous facilities that are provided by ROS which helps researchers to develop robot applications. In this research work, ROS is considered as the main base because it publishes messages in the form of topics in between different nodes and has a distributed parameter system. ROS also provides Inter-platform operability, Modularity, Concurrent resource handling. ROS simplifies the whole process of a system by ensuring that the threads aren't actually trying to read and write to shared resources, but are rather just publishing and subscribing to messages. ROS also helps us to create a virtual environment, generate robot model, implement the algorithms and visualize it in the virtual world rather than implementing the whole system in the hardware itself. Therefore, the system can be improved accordingly which provides us a better result when it is finally implemented it in the hardware.

B. Gazebo

The gazebo is a robot simulator. Gazebo enables a user to create complex environments and gives the opportunity to simulate the robot in the environment created. In Gazebo the user can make the model of the robot and incorporate sensors in a three-dimensional space. In the case of the environment, the user can create a platform and assign obstacles to that. For the model of the robot, the user can use the URDF file and can give links to the robot. By giving the link we can give the degree of movement for each part of the robot. The robot model which is created for this research is a differential drive robot with two wheels, Laser, and a camera on it as shown in Fig. 1. A sample environment is created in the Gazebo for the robot to move and map accordingly. The sample map is shown in Fig. 2. In this environment, several objects were placed randomly where the map is created along with it the objects as these objects were considered as static obstacles.

C. SLAM

Autonomous robots should be capable of safely exploring their surroundings without colliding with people or slamming into objects. Simultaneous localization and mapping (SLAM) enable the robot to achieve this task by knowing how the surroundings look like (mapping) and where it stays with respect to the surrounding (localization). SLAM can be implemented using different types of 1D, 2D and 3D sensors like acoustic sensor, laser range sensor, stereo vision sensor and RGB-D sensor. ROS can be used to implement different SLAM algorithms such as Gmapping, Hector SLAM, KartoSLAM, Core SLAM, Lago SLAM.

KartoSLAM, Hector SLAM, and Gmapping are better in the group compared to others. These algorithms have a quite similar performance from map accuracy point of view but are actually conceptually different. That's, Hector SLAM is EKF based, Gmapping is based on RBPF occupancy grid mapping and KartoSLAM in based on thegraph-based mapping. Gmapping can perform well for a less processing power robot. The mapping package in ROS provides laser-based SLAM (Simultaneous Localization and Mapping), as the ROS node called slam_gmapping.

D. Rviz

Rviz is a simulator in which we can visualize the sensor data in the 3D environment, for example, if we fix a Kinect in the robot model in the gazebo, the laser scan value can be visualized in Rviz. From the laser scan data, we can build a map and it can be used for auto navigation. In Rviz we can access and graphically represent the values using camera image, laser scan etc. This information can be used to build

the point cloud and depth image. In rviz coordinates are known as frames. We can select many displays to be viewed in Rviz they are data from different sensors. By clicking on the add button we can give any data to be displayed in Rviz. Grid display will give the ground or the reference. Laser scan display will give the display from the laser scanners. Laser scan displays will be of the type sensor msgs/Laser scans. Point cloud display will display the position that is given by the program. Axes display will give the reference point.

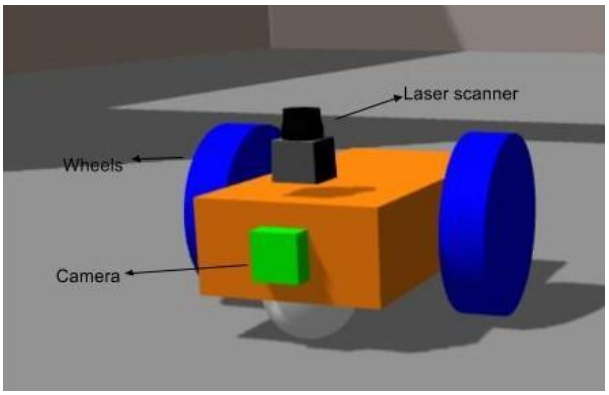

Fig. 1. Robot Model.

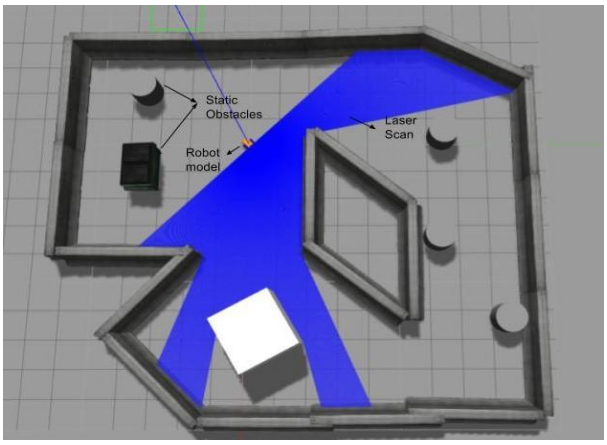

Fig. 2. Sample Gazebo Environment – 1.

V. IMPLEMENTATION

The environment for the robot model to perform the navigation is created in the gazebo and the robot model which was created is imported into the environment. The robot model consists of two wheels, two caster wheels for the ease of movement and a camera is attached to the robot model. Later the Hokuyo Laser is added to the robot and plugins were incorporated into the gazebo files. Hokuyo laser provides laser data which can be used for creating the map. Using the Gmapping packages a map is created in the Rviz by adding the different parameters that are necessary. The Fig. 3, shows the initial generation of the map when launched. Initially, the robot model is moved to every corner of the environment until a full map is created using the "teleop_key" package where the robot is controlled using the keyboard. As shown in the Fig. 4, the final generated map in the Rviz which is very much similar to the created environment in the gazebo. For visualization in Rviz, necessary topics were selected and added. The Hokuyo laser sensor which is used in this robot model publishes the laser data in the form of the topic "/scan" which is selected as a topic of laser scan in rviz. In a similar way for creating the map, "/map" topic is added. The generated map is saved using the map_server package that is available in the ROS. Once the map is generated and saved the robot is now ready for the incorporation of navigation stack packages.

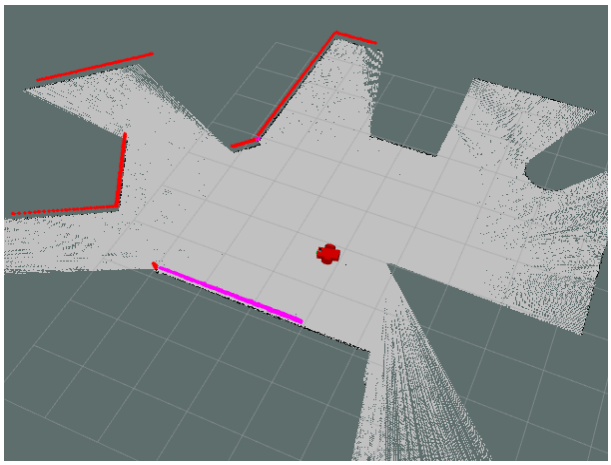

Fig. 3. Initial Scan of the robot model in Rviz.

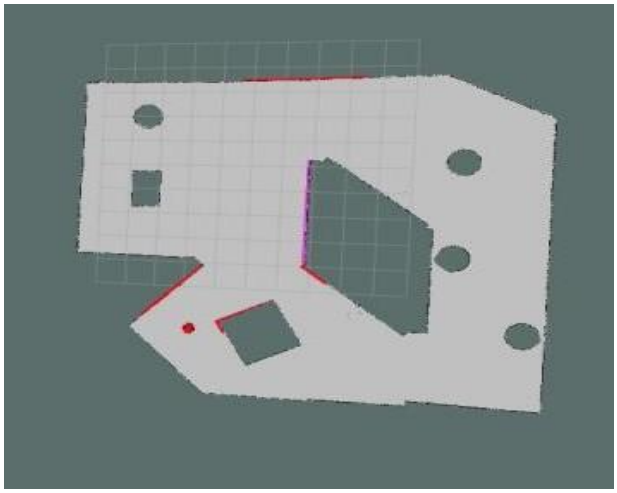

Fig. 4. The final map of the environment -1 created in Rviz.

It is very important to note that a robot cannot be navigated without feeding the map to it. Navigation stack packages by using amcl were used which provides a probabilistic localization system for a robot to move in a 2D. Now, the robot is ready to navigate anywhere in the created map. The destination for the robot can be given using the 2D nav goal option in the Rviz which basically acknowledges the robot

with a Goal. The user has to click on the desired area in the map and should also point out the orientation of the robot that it has to be in. The blue line is the actual path that the robot has to follow to reach the destination. The robot may not follow the exact path that is given to it due to some of the parameters but it always tries to follow it by rerouting itself constantly.

The node graph that is shown in the Fig. 5, indicates the different topics that are being published and subscribed to the different nodes. The /move_base node is subscribed to several topics like odometry, velocity commands, map, goal, these topics gives the necessary data for the base of the robot to navigate in the environment.

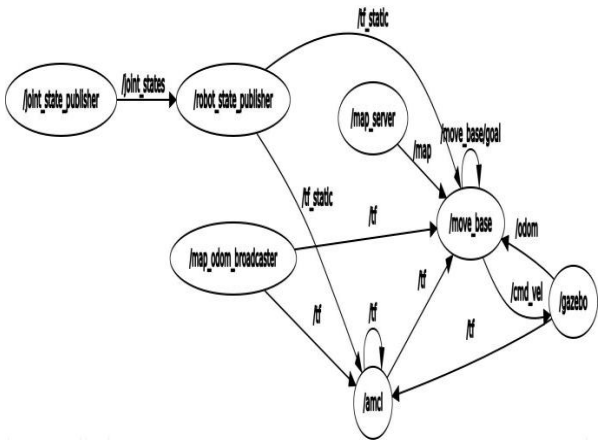

Fig. 5. Node Graph.

VI. Evaluation of the Results

In order to evaluate the performance of ROS and slam based Gmapping and navigation, specific environments were created. In each environment, different parameters like how well the SLAM generated maps represent reality, the time it took for the robot to reach the given destination. Also, the dynamic obstacles were placed in the robot's navigation path to test the amount of time that is required for the robot to reroute itself to another path.

The Fig. 6, shown is considered as environment 1. The Fig. 6, shows the several destinations that are considered for testing the algorithms. When the destination A is given as a target to the robot, the Slam finds out the shortest path according to the previously generated map but when we place dynamic obstacles in the path, as shown in Fig. 6, the laser sensor scans the map and updates it by adding the detected obstacle in the map. Once the map is updated, the Slam finds the next shortest path to reach the destination as shown in the Fig. 7.

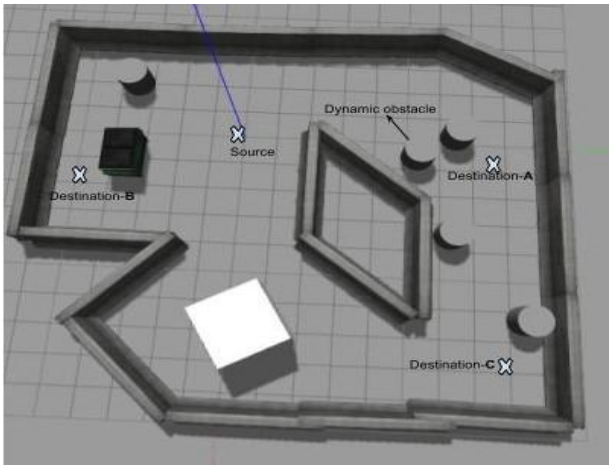

Fig. 6. Test Environment – 1.

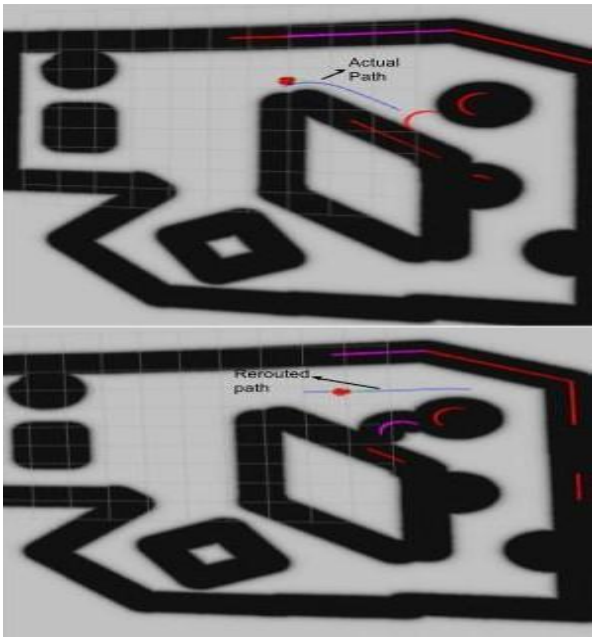

Fig. 7. Obstacle detection and path planning in theenvironment – 1.

The Table 1, indicates the different time readings that are taken in the environment – 1 in different destinations and Table 2, indicates the time readings for the same destinations with obstacles in the path.

TABLE I. Travel Time in Theenvironment – 1 Without Obstacles

Trials	Source to A (in Sec.)	Source to B (in Sec.)	Source to C (in Sec.)
1	32.66	38.55	61.64
2	29.56	39.06	66.20

3	31.35	40.82	67.83
4	31.30	42.08	60.38
5	30.02	41.36	65.32
6	29.88	39.25	61.48
7	30.40	40.25	63.77
8	31.58	43.77	64.83
9	30.12	42.92	64.36
10	30.57	41.65	67.92
Average	30.744	40.971	64.373

TABLE II. TRAVEL TIME IN THEENVIRONMENT – 1 WITH OBSTACLES

Trials	**Source to A with 1 obstacles (in Sec.)**	**Source to B with 2 obstacles (in Sec.)**	**Source to C with 3 obstacles (in Sec.)**
1	36.10	53.65	78.82
2	38.36	59.60	77.25
3	34.12	57.17	78.63
4	35.35	56.22	79.29
5	35.60	54.48	80.20
6	36.36	55.83	80.96
7	35.98	59.77	75.25
8	36.48	57.36	78.99
9	36.37	58.47	76.23
10	36.44	56.70	83.67
Average	36.116	56.925	78.929

A basic floor map is designed and used as an environment – 2 as shown in Fig. 8. The map is generated using the slam mapping packages and navigation stack for the robot to move autonomously. Fig. 9 shows the three different destinations that are considered in the environment-2 for testing the algorithms. The first test is done without adding the obstacles and in the second test different obstacles were added. The test results were shown in Table 3 and Table 4.

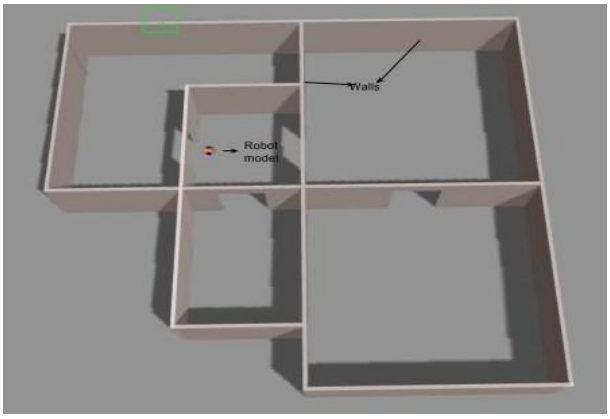

Fig. 8. Test environment – 2.

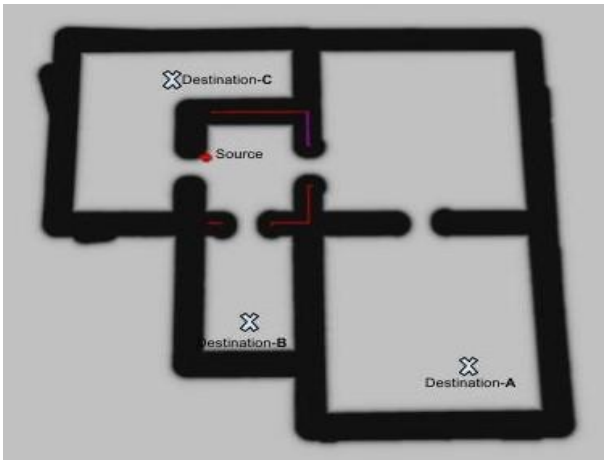

Fig. 9. Generated map of test environment – 2.

TABLE III. TRAVEL TIME IN THEENVIRONMENT – 2 WITHOUT OBSTACLES

Trials	**Source to A (in Sec.)**	**Source to B (in Sec.)**	**Source to C (in Sec.)**
1	138.28	96.83	78.56
2	147.47	85.88	77.08
3	146.42	80.10	78.22
4	145.95	79.48	80.17
5	135.89	80.61	77.23
6	139.16	81.76	77.63
7	140.60	83.48	80.41
8	142.05	43.77	64.83
9	145.64	84.37	81.57

10	145.33	83.63	82.36
Average	142.679	83.75	78.861

TABLE IV. TRAVEL TIME IN THEENVIRONMENT – 2 WITH OBSTACLES

Trials	**Source to A with 1 obstacles (in Sec.)**	**Source to B with 2 obstacles (in Sec.)**	**Source to C with 3 obstacles (in Sec.)**
1	158.68	110.55	96.87
2	162.76	104.86	96.76
3	160.31	106.42	101.59
4	156.38	108.44	94.86
5	155.51	106.15	96.63
6	150.82	109.76	99.68
7	151.05	116.27	98.33
8	158.51	112.85	97.49
9	155.09	113.91	100.92
10	155.97	114.57	95.51
Average	156.508	110.378	97.864

VII. CONCLUSION

In order to validate the performance of ROS and slam based gmapping and navigation. In this project, certain environment and the map of the same is created in Rviz simulator by driving the robot through the environment. After creating the map destination point was fixed. Then the time for the robot to reach the destination was calculated. By considering 10 trials average is obtained. A similar process was continued by changing the destination points. Also in some cases obstacles were also introduced, so that the robot will find out another path and will travel through that. The same way a second environment is created and tested. The time taken to reach the destinations are calculated.

From this research, it is observed that the robot gives a good response time and also it takes only reasonable time to cover the distance from the source to destination. As the distance increases increase time also increases. In the case of a map with obstacles, the robot will find the shortest path. And if an extra obstacle is introduced the robot will stop and recalculate the new path.

ACKNOWLEDGMENT

We thank Amrita Vishwa Vidyapeetham and HUT lab for providing us all the necessary lab facilities and support for successful completion of this research work.

REFERENCES

[1] Emil-Ioan Voisan, Bogdan Paulis, Radu-Emil Precup and Florin Dragan, "ROS-Based Robot Navigation and Human Interaction in Indoor Environment," in 10th Jubilee IEEE International Symposium on Applied Computational Intelligence and Informatics, May 21-23.

[2] SangYoung Park and GuiHyung Lee, "Mapping and Localization of Cooperative Robots by ROS and SLAM in Unknown Working Area," in Proceedings of the SICE Annual Conference 2017 September 19-22, 2017.

[3] Wei Qian, Zeyang Xia, Jing Xiong, Yangzhou Gan, Yangchao Guo, Shaokui Weng, Hao Deng, Ying Hu, Jianwei Zhang, "Manipulation Task Simulation using ROS and Gazebo," in 2014 IEEE International Conference on Robotics and Biomimetics December 5-10, 2014.

[4] Sebastian Schweigert, "gmapping," ros.wiki.org .

[5] Doris M. Turnage, "SIMULATION RESULTS FOR LOCALIZATION AND MAPPING ALGORITHMS," in 2016 Winter Simulation Conference, 2016.

[6] Hesham Ibrahim Mohamed Ahmed Omara, Khairul Salleh Mohamed Sahari, "Indoor Mapping using Kinect and ROS," in International Symposium on Agents, Multi-agent Systems and Robotics (ISAMSR), 2015.

[7] Davetcoleman, "rviz," ros.wiki.org.

[8] davetcoleman, "simulator_gazebo," ros.wiki.org.

[9] Giada Giorgi, Guglielmo Frigo, and Claudio Narduzzi, "Dead Reckoning in Structured Environments for Human Indoor Navigation," in IEEE SENSORS JOURNAL, VOL. 17, NO. 23, DECEMBER 1, 2017.

[10] He Zhang and Cang Ye, "An Indoor Wayfinding System Based on Geometric Features Aided Graph SLAM for the Visually Impaired," in 1592 IEEE TRANSACTIONS ON NEURAL SYSTEMS AND REHABILITATION ENGINEERING, VOL. 25, NO. 9, SEPTEMBER 2017.

[11] Slawomir Grzonka, Giorgio Grisetti, and Wolfram Burgard, "A Fully Autonomous Indoor Quadrotor," IEEE TRANSACTIONS ON ROBOTICS, VOL. 28, NO. 1, FEBRUARY 2012.

[12] Rajesh Kannan Megalingam, Vishnu G B, Meera Pillai, "Development of Intelligent Wheelchair Simulator for Indoor Navigation Simulation and Analysis," WIE Conference on Electrical and Computer Engineering (WIECON-ECE), 19-20, DECEMBER 2015.

[13] Rajesh Kannan Megalingam, Jeeba M Varghese, Aarsha Anil S, "Distance Estimation and Direction Finding Using I2C Protocol for an Auto-navigation Platform," International Conference on VLSI Systems, Architectures, Technology and Applications (VLSI- SATA), 10-12 JANUARY 2016.

[14] Sandeep B.S. and Supriya P, "Analysis of Fuzzy Rules for Robot Path Planning," Conference on Advances in Computing, Communications and Informatics (ICACCI), SEPTEMBER 21-24, 2016.

[15] Hemanth Malla, Preethish Purushothaman, Shivnesh V Rajan and Vidhya Balasubramanian, "Object Level Mapping of an Indoor Environment using RFID," Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), 20-21 November, 2014.