Simulation of an Autonomous Navigation System using Quadrotor Model in Robot Operating System

Darshan KT
Post Graduate Student, Dept. of Mechanical Engg.
The National Institute of Engineering, Mysuru
Karnataka, India

Rakesh M
Dept. of Mechanical Engg.
The National Institute of Engineering, Mysuru
Karnataka, India

***Abstract*—This paper describes the simulation of autonomous navigation system using quadrotor model within a self-created environment. The Gazebo is an open source simulator software which is used for doing simulation of this entire work. Robot Operating System(ROS) is an open source software framework used for many robotics applications. It provides all necessary packages, tools required for doing autonomous navigation with quadrotor model inside a simulation environment. This entire work is done by following 3 major processes, firstly build the map of self-created environment by using Gmapping package. Then localize the quadrotor model correctly inside the built map with help of Adoptive Monte Carlo Localization(AMCL) package. Finally, provide a target pose inside the map in Rviz, then the move_base package helps to quadrotor model to autonomously navigate towards target pose by avoiding obstacles.**

Keywords—Gazebo simulator, ROS framework, quadrotor model, Gmapping, AMCL, move_base.

I. INTRODUCTION

A quadrotor, also called quad rotor-craft, is a rotor-craft that is lifted, controlled, and propelled by four rotors. The quadrotor can takeoff and land vertically. It can fly in any direction without changing its heading. With the advances in electro-mechanical systems and sensing technologies, recently quadrotors have become popular in unmanned systems research. Their small size and maneuverability make it possible to fly indoor and outdoors. Compared with other rotor-craft, quadrotors are less complicated and easier to control. They have many different applications, including border patrol, search and rescue, pipeline monitoring, wildfire monitoring, traffic monitoring, land surveys, and load transportation[1]. Similarly, quadrotor can also be used in many indoor environments or many GPS denied environment application such as warehouse inspection, rack inspection, factory inventory inspection, etc, by adopting autonomous navigation concept.

This paper objective is also developing a quadrotor which can have the capability to travel autonomously to given target pose in an indoor environment. So before going to implement this concept in real time, this paper explains the achieving same concept in indoor simulation environment with simulated quadrotor model by using suitable simulator software. The simulators have played a critical role in robotics research as tools for quick and efficient testing of new concepts, strategies, and algorithms. Among robotics simulators available in the market, in this paper chosen the Gazebo simulator which is mainly designed to accurately reproduce the dynamic environments a robot may encounter.

Gazebo also offers a rich environment to quickly develop and test systems in new and interesting ways. Gazebo uses open source Open Dynamics Engine(ODE) which is a physics engine, it helps to simulate the dynamics and kinematics associated with articulated rigid bodies and it also provides many functionalities like collision detection, mass, linear and rotational functions for joints, inertial properties for links. Gazebo allows the user to create any type of model from a simple structure such as having one or two links with joint assembly to the complex structure like multiple numbers of links and joints assembly. All simulated models created in gazebo have mass, velocity, friction and numerous other attributes that allow them to behave realistically. Gazebo supports many sensor modules and simulates its properties such as emit data and collect data when the sensor is active[2]. In this paper used the RPlidar sensor to emit data and collect data form environment.

The Robot Operating system is an open source software framework for robotics application. This software is organized into a set of packages, it also has a package called gazebo_ros which can be used to exchange the information between gazebo simulator and ROS framework. So in this work, the ROS software framework is chosen as a middle-ware to exchange information between simulation. ROS framework provides many numbers of packages required for achieving the objective of this work. Also ROS provides many useful algorithms for doing SLAM, path planning, and autonomous navigation process.

II. METHODOLOGY

A. Gazebo

Gazebo is a multi-robot 3D dynamic simulator which helps to simulate the dynamic interactions between robots in environments similar to real time. Gazebo only supports new models which were created with URDF(Unified Robot Description Format) or SDF (Simulation Description Format) file format. In this paper, the quadrotor 3D CAD design model was chosen from ‘GrabCAD’ website and imported this model into ‘SolidWorks’ design software. Here alter some design properties of quadrotor model and finally extract the URDF files from it by using ‘Export to URDF’ plugin. In the next step added the RPlidar laser sensor URDF file to extracted URDF to mount the RPlidar sensor on top of quadrotor model. This entire model has then launched in Gazebo it looks like as shown in figure 1. For doing map building and autonomous navigation application requires a simple world environment. So by using available templates, a simple indoor office environment is created which as shown in figure 2.

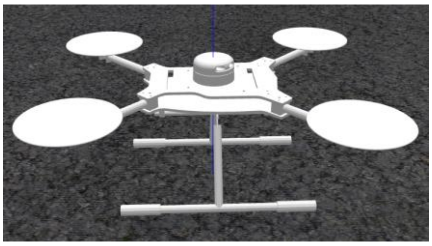


Fig. 1. Quadrotor model

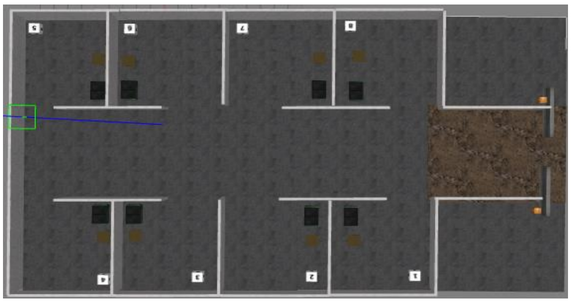


Fig. 2. Simple indoor office environment

B. ROS

Robot Operating System is a meta-operating system for robotics applications.

It provides hardware abstraction, low-level device control, and high-level functionalities like asynchronous and synchronous calls for smoothly operate a number of processes and it maintains robot configuration systems[4]. ROS acts as middle-ware or communication system, it establishes the communication between number of processes(nodes) by using publishing and subscribing messages to topics concept. It also uses ROS service concept to exchange data between two nodes. ROS can also be used as a software framework for developing many robotic applications, it provides client libraries such as rospy, roscpp, rosLisp which helps to writing, building own packages. It has many tools such as Rviz(Robot visualizer), rqt, etc, for inspection and debugging operations.

ROS software is organized into several packages, each packages contains their own configuration files, launch files, scripts, etc, required for their specific application.

C. Hector_quadrotor package

In this paper to achieve autonomous navigation application, chosen the open source hector_quadrotor package developed by Team Hector Darmstadt of Technische Universitat. This hector_quadrotor meta- package contains the following key packages.

hector_quadrotor_gazebo: This package contains launch files for starts up Gazebo simulator with spawning the quadrotor model.

hector_quadrotor_gazebo_plugins: This package contains the plugins, these are a chunk of C++ codes attached to URDF/SDF model files which are required to provide functionality and controls the simulated models. Some plugins available in this package are,

a) gazebo_quadrotor_simple_controller: which subscribes to geometry_msgs/Twist topic and calculates the required forces and torque for move quadrotor model.

b) gazebo_ros_baro: A sensor plugin simulates a barometric altitude.

c) gazebo_quadrotor_propulsion: simulates propulsion, aerodynamics and drag messages containing motor voltages and wind vector input.

hector_gazebo_plugins: This package contains generic sensor plugins not specific to UAVs such as IMU, magnetic field, GPS, and sonar data.

hector_quadrotor_teleop: This package provides a node and launches files for controlling a quadrotor using a joystick or gamepad.

hector_quadrotor_demo: This package provides sample launch files that run the Gazebo quadrotor simulation and hector_slam for indoor and outdoor scenarios[7].

D. SLAM

To do autonomous navigation by using the quadrotor model inside the gazebo world, the quadrotor model needs to know where it is, and where it wants to go to[3]. Typically, this means that it needs to have a map of the world and know where it is in a map, this combination process is called as Simultaneous Localization and Mapping(SLAM). SLAM allows the quadrotor to perform both the processes at the same time by applying its algorithms.

E. Map building

Map building is one of the SLAM process, which uses the ROS gmapping package contains slam_gmapping node that extracts the quadrotor sensors and laser scanners information to create a 2D occupancy grid map of surrounding environment. In this work, firstly takeoff the quadrotor model to a certain altitude and moves it through the simulated simple office world, at same time slam_gmapping node builds the 2D occupancy grid map of the surrounding environment wherever the quadrotor model moves. The slam_gammping node subscribes to sensor scan data produced by RPlidar laser scanner model which mounted is on quadrotor model, an odometry data which is produced by quadrotor_groundtruth_sim plugin and tf(transformation) data produced by tf package and then creates and publishes 2D map of simulated environment on /map topic as shown in figure 9. In the built map, grids filled with white color indicates free path, grids occupied with dark color indicates as obstacles and grids occupied with gray color indicate as unexplored regions as shown in figure 9.

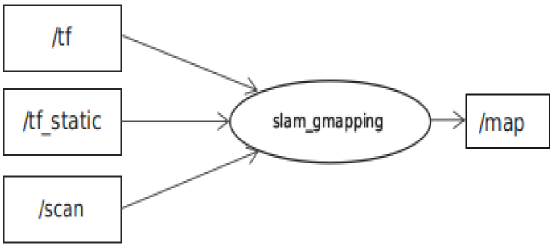


Fig 3. Working of slam_gmapping node

F. *Localization*

Localization is another SLAM process, which uses the previously built map to localize the quadrotor model. To do autonomous navigation to target pose using the map, it needs the accurate localization of the quadrotor model inside the built map which can be accomplished by using Adaptive Monte Carlo Localization(AMCL) algorithm. This ROS amcl node subscribes to scan data produced by the RPlidar model, an odometry data coming from quadrotor_groundtruth_sim plugin, saved built map from map_server and finally localizes the quadrotor model accurately which is indicated by particle filters spreading around quadrotor model inside map. Particle filters used in this are called KLD(Kullback-eibler divergence) sampling. The KLD sampling method chooses the number of samples for sampling or estimation process based on uncertainty[5]. That is if less cloud of particle filters spreading around quadrotor model indicates that the model localized correctly inside map and if more cloud of particle filters spreading around the model indicates that the model was not correctly localized.

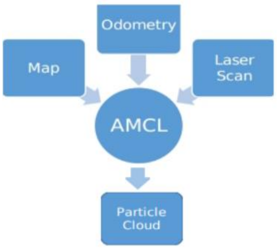


Fig 4. Working of AMCL node.

G. *Navigation*

Once the quadrotor model was correctly localized inside the built map of simulated environment, the next task is to navigate the quadrotor model autonomously from the current pose to the desired target pose. This task can be accomplished by using ROS move_base node with its configuration files. The move_base is a type of action node will attempt to reach the target goal given in the world, also it updates the quadrotor model pose like where it is in and where it wants to go. The move_base node uses costmap, planner and recovery behavior configuration files. The global_costmap and local_costmap represent places which are safe for the quadrotor, free spaces or places where the collision would happen. In the other end, global planner and local planner use D* algorithm which is amap based algorithm used to generate a least possible path for quadrotor model to move from current pose to target pose.

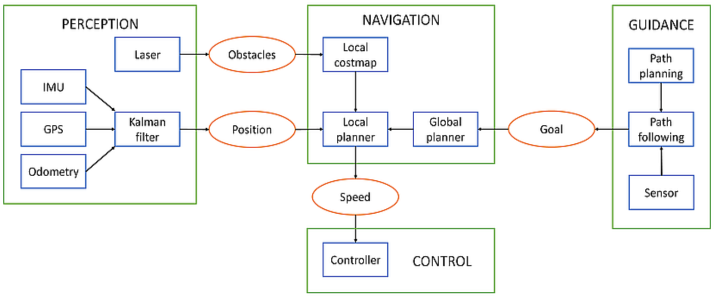


Fig 5. Working of Navigation Stack

At end move_base node utilizes this generated path given by planners for moving model on this planned path, sensors scan data from RPlidar for obstacle avoidance and finally it sends an appropriate control velocity to a model controller to move model towards target pose.

III. IMPLEMENTATION

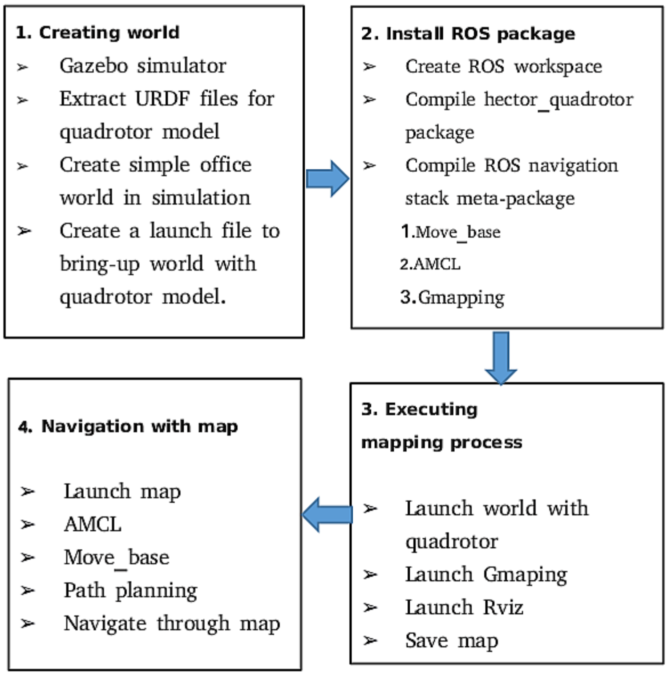

1. Creating world
➢ Gazebo simulator
➢ Extract URDF files for quadrotor model
➢ Create simple office world in simulation
➢ Create a launch file to bring-up world with quadrotor model.
2. Install ROS package
➢ Create ROS workspace
➢ Compile hector_quadrotor package
➢ Compile ROS navigation stack meta-package
1.Move_base
2.AMCL
3.Gmapping
3. Executing mapping process
➢ Launch world with quadrotor
➢ Launch Gmaping
➢ Launch Rviz
➢ Save map
4. Navigation with map
➢ Launch map
➢ AMCL
➢ Move_base
➢ Path planning
➢ Navigate through map

Fig 6. Implementation of the autonomous navigation system

To implement this work, we have followed the above steps as shown in figure 6. Firstly chosen the Gazebo simulator for doing complete work, then extracted the URDF files for quadrotor model by using the SolidWorks design software and created a simple office with eight chambers world inside the simulator. Finally launched the simulator with the quadrotor model and the world which appears in Gazebo simulator as shown in figure 7.

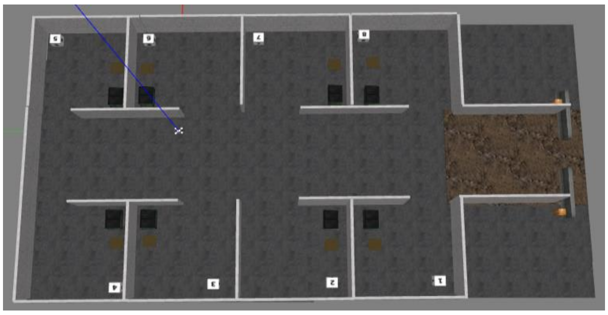
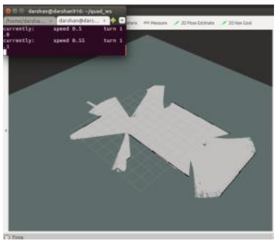


Fig 7. Simulation world with quadrotor model

A separate ROS workspace was created and compiled the required ROS packages inside a workspace. To achieve objective this work that is autonomously navigates the quadrotor model inside the created world, it requires two per-requisites processes such as building a map of environment and localization of inside the environment. To achieve the map building process, ROS gmapping algorithm was used which takes the input from laser scan data provided by Rplidar laser plugin and transformation data including odometry data provided by quadrotor_groundtruth_sim plugin in the model. Then takeoff model and starts moving the model inside the simulator. On the other end gmapping algorithm start building the map world, it is visualized in Rviz tool as shown in below figure 8&9.



```
currently:      speed 0.5       turn 1
.0
currently:      speed 0.55      turn 1
.1
```

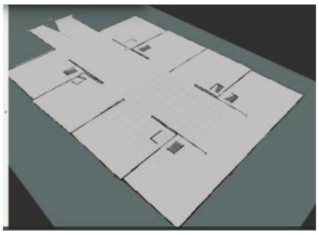


Fig 8. Starting of map building process.

Fig 9. At the end of the process.

Once built the map of the entire world by teleop the quadrotor model it can be saved using map_saver node. Then get .pgm and .yaml file of this built map as shown in figures 10&11, these can be used for further processes such as localization, planning, and navigation.

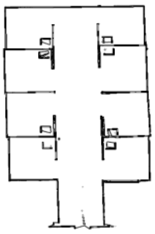
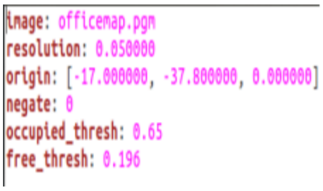


Fig 10. officemap.pgm



```
image: officemap.pgm
resolution: 0.050000
origin: [-17.000000, -37.800000, 0.000000]
negate: 0
occupied_thresh: 0.65
free_thresh: 0.196
```

Fig 11. officemap.yaml

The localization process can be accomplished by using amcl node which takes inputs as built map from map_server rosparam, RPlidar scan data, tf data with odometry and finally localize the model correctly inside map by publishing particle filters around the model as an output as shown in the figure 12.

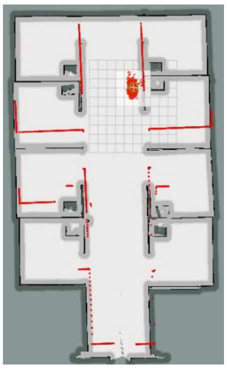
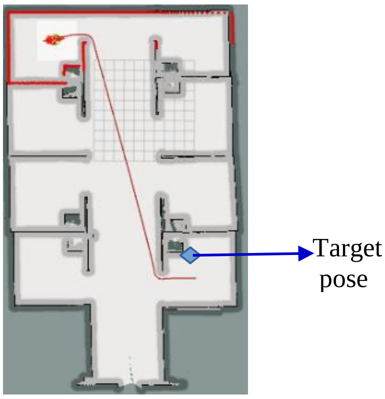


Fig 12. Localization of quadrotor model

Once localization is done correctly, then provide a target pose inside map by using Rviz "2D Nav Goal" option. The motion of the quadrotor model is carried out by move_base node. It maintains global and local cost maps for making global and local path plans. The information of obstacles are stored in cost maps. The global cost map is used of making global planning which makes a plan over the entire operating environment and the local cost map is used for short-range planning and obstacle avoidance[3]. So by using this generated path planner the move_base node moves the model by sending required commands to model. In the below figure 13 shows the path plan generated for model to move from current pose to target pose. And figure 14 shows that model is moving along the generated path towards target pose.



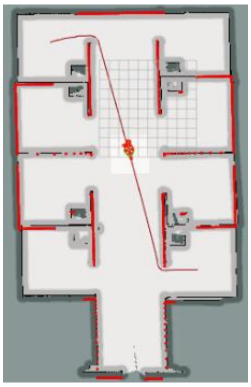


Fig 13. Generated planner Fig 14. Quadrotor moving

towards target pose towards target pose

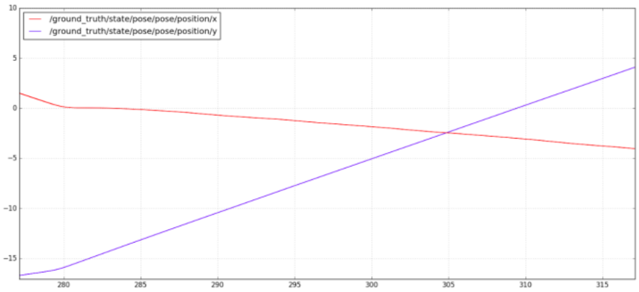


Fig 15. rqt_plot_graph for varying pose value of model.

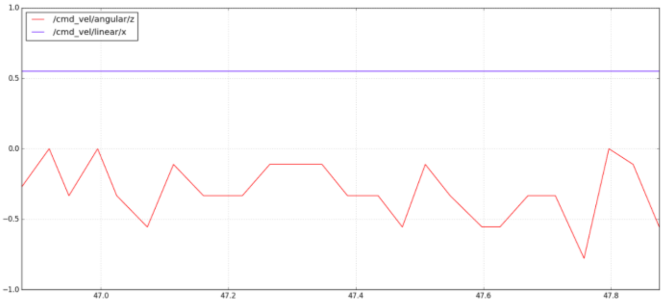


Fig 16. rqt_plot graph for velocity given to model.

Once given the target pose inside the map using Rviz tool, the move_base node takes the control of the quadrotor model controller, then it moves the model towards the target pose. While it moving it changes the its x,y co-ordinates pose values inside the map that shown in figure 15 using rqt_plot tool. Also move_base node varies the command velocity given to model model while moving depending on free path, curve path and obstacle path is as shown in rqt_plot tool in the figure 16. In that figure blue line represent linear velocity given along x-axis(max. 0.55m/s) and red line represent the angular velocity given to model along z-axis.

This angular velocity helps to model to rotation along curve paths and balances the roll, pith and yaw.

Once the model reaches target pose it will stops there and waits for another target pose and repeats the navigation process.

IV. CONCLUSION

In this paper discussed the autonomous navigation system on simulated quadrotor model within a simulated environment given by Gazebo simulator. This simulator helps us to witness the dynamic behavior of the quadrotor model with respect to the environment. The ROS framework help us to integrate the simulator with ROS packages. Also by using ROS hector_quadrotor meta-package, controls the dynamic behavior of quadrotor model inside simulator. The objective of this work was achieved by using ROS navigation stack which provides the Gmapping, AMCL and move_base packages required for doing map building, localization and autonomous navigation processes respectively. These entire operations help us to witness the efficient way to implement the autonomous navigation system on quadrotor in simulation. In the future work going to implement the same concepts on real quadrotor for doing warehouse inspection applications.

ACKNOWLEDGMENT

The authors would like to acknowledge Wipro Technologies Ltd for giving an opportunity to undergo this project and this work was jointly supported by Mr. Yateesh Kumar.S Technical consultant Wipro technologies Ltd, Mr. Ramkumar Gandhinathan Senior associate Wipro technologies Ltd Bengaluru and Dr. KR Prakash, department of mechanical engineering, The National Institute of Engineering, Mysuru.

REFERENCES

[1] Saeedi, Sajad, “Perception and navigation for an autonomous quadrotor in GPS-denied environments”, International Journal of Robotics and Automation, vol.31, 10.2316/Journal.206.2016.6.206-4434, 2016.

[2] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator,"2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Sendai, 2004, pp. 2149-2154 vol.3. 2014.

[3] Muhammad Lutfi bin Hamdan, Sami Salama Hussen Hajjaj, “Programming a Mobile Robot to Navigate a Known Area Utilizing Resources from the Cloud ”, International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 7 (2018) pp. 5348-5353, 2018

[4] Arbnor Pajaziti, “SLAM – Map Building and Navigation via ROS”, International Journal of Intelligent Systems and Applications in Engineering, ISSN:2147-6799, 2014

[5] R. Mishra and A. Javed, "ROS based service robot platform,"2018 4th International Conference on Control, Automation and Robotics (ICCAR),pp.55-59. doi: 10.1109/ICCAR2018, Auckland, 2018

[6] Q. Xu, J. Zhao, C. Zhang and F. He, "Design and implementation of an ROS based autonomous navigation system,"2015 IEEE International Conference on Mechatronics and Automation (ICMA),pp. 2220-2225, Beijing, 2015.

[7] Book - ROS Robotics By Example by Dr. Thomas L. Harman, Carol Fairchild.

[8] hector_quadrotor – ROS Wiki [Internet] Available from: http://wiki.ros.org/hector_quadrotor.

[9] navigation_stack – ROS Wiki [Internet] Available from: http://wiki.ros.org/navigation.

[10] Gazebo Tutorials - ROS Wiki [Internet] Available from: http://gazebosim.org/tutorials.