Map My World : Performing SLAM with ROS

Vijayasri Iyer

Abstract—This paper presents an attempt to solve the robot navigation problem by using the popular method known as SLAM (Simultaneous Localization And Mapping). SLAM is the method of performing robot navigation in an unknown environment. The experiment presented in this paper aims to solve the above problem in two different simulated Gazebo environment using ROS (Robot Operating System), with a custom differential drive robot model. The Graph SLAM method, along with its implementation as a ROS package and challenges to setup as well as usage of the package are discussed. The results of the experiment will fuel further research in autonomous robot navigation.

Index Terms—Robot, IEEEtran, Udacity, LaTeX, SLAM.

1 Introduction

In robotic systems the navigation problem consists of three important sub-problems; localization, mapping and path-planning. Localization is the process of a robot estimating its location in an environment, based on a combination of its prior knowledge of the environment and sensor measurements, while mapping is that branch which deals with the study and application of ability of a robot to localize itself in a map / plan and sometimes to construct the map or floor plan. Both of these problems are dependent on each other, for accurate results. The SLAM (Simultaneous Localization And Mapping) is an approach to solving both these problems in simultaneously, in real-time. It is the computational problem of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it. The SLAM problem can be solved using a number of approaches, using a number of algorithms in each. In this paper, the Graph SLAM approach is used to solve the Full SLAM problem, thus allowing the robot to estimate its entire navigation path.

2 Background

The SLAM problem can be divided into two distinct problems, the Online and the Offline SLAM problem. The nature of the SLAM problem can be continuous or discreet depending upon the environment. The Online problem deals with estimating the robot path, based on the only the current state i.e, the robot's current state depends only on its immediately previous state. The Offline or Full SLAM problem involves estimating the entire path of the robot given its previous and current states. There are various algorithms that solve each of these approaches. The FastSLAM approach is a popular approach used to solve the full SLAM problem. This approach has 3 variants namely FastSLAM 1.0 2.0 which assume known landmarks and Grid-Based FastSLAM, which is non-landmark based algorithm. An advantage of this approach is that it uses a particle filter or an occupancy grid map algorithm to estimate it's position. Another approach to solving the Full SLAM problem is Graph SLAM. It is by far, one of the most complicated SLAM algorithms since it can require a lot of computation. This approach uses four types of data namely, Poses, Features, Motion constraints and Measurement constraints. The work done by the algorithm is split into two parts, the Front-End and Back-End. The Front-End deals with constructing the graph using the sensor and odometry measurements collected by the robot. It also performs the task of solving the data association problem i.e checking whether a location has been visited before or not. The Back-End deals with graph optimization and is responsible for producing the output containing the most probable poses of the robot. This approach, although effective, is a 2D SLAM approach. In this paper, a 3D Graph based SLAM approach called RTABMAP is used. This method is explained in detail in the upcoming section.

2.1 RTABMAP

Real-Time Appearance Based Mapping, also known as RTABMAP is a 3D graph SLAM approach that uses visual odometry and sensor measurements to approximate it's map. The Front-End of the algorithm uses the sensor and odometry measurements to perform loop closure detection, whereas the Back-End is used for Graph Optimization and 2D/3D Map Generation. Loop closure is a method to solve the data association problem is mentioned above. In RTABMAP, loop closure is done using a method called SURF or Speeded Up Robust Features. Here, each feature has a descriptor associated with it which is a representation of the pixels that make up a feature and the point where the feature is located, is split into smaller square sub regions. From these sub-regions the regions of regularly spaced sample points are calculated and compared. In order to compare the large amount of features, they are clustered into groups of similar features or synonyms. Collection of these clusters represents the vocab. when a feature descriptor is matched to one in the vocab it is called quantization. now the feature that is linked to a word, can be referred to as a visual word. When all features in an image are quantized the image is now a bag of words each word keeps track of the images that it is associated with, for efficient retrieval of data. A matching score is given to all images containing the same words. this is managed in the form of a table called the

inverted index. A bayesian filter with the hypothesis that the image is seen before is used to evaluate the scores. once the score crosses a certain threshold then a loop closure is detected. An illustration of visual bag of words approach can be seen in fig 1.

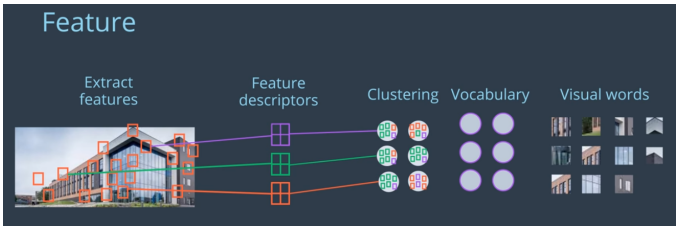

Fig. 1. Visual Bag of Words

The primary advantage of using the RTABMAP approach includes Loop Closure Detection, efficient Memory Management and Graph Optimization techniques such as Tree-based network optimizer, General Graph Optimization etc. An illustration of the memory management in RTABMAP algorithm is given below in fig 2. This task uses rtabmap-ros, which is a readily available ROS package. The documentation for this package is available in the ROS wiki pages.

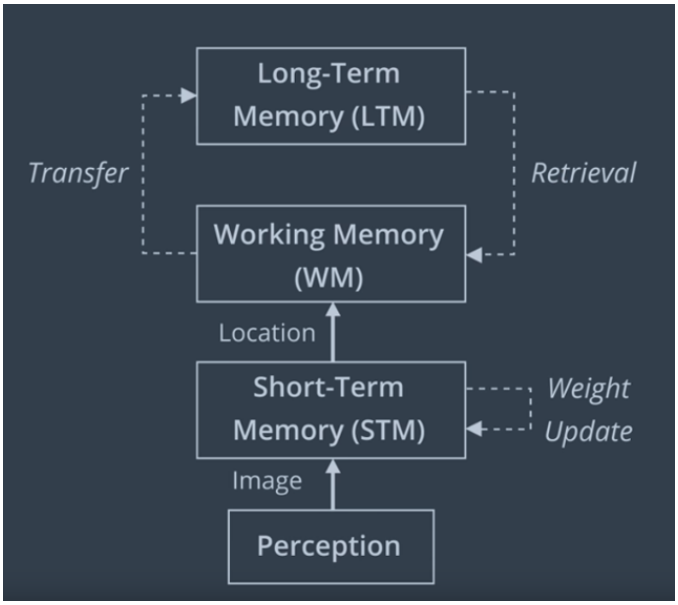

Fig. 2. Memory management

The reason RTABMAP is used for this task, is because of the ability of the algorithm to leverage visual data for generating the map. This makes it easier to perform SLAM using a relatively cheap sensor such as a stereo camera or a kinect, instead of an expensive LIDAR sensor. This is quite an advantage when it comes to building a physical prototype.

3 Scene and Robot Configuration

For this experiment, the robot was tested in two worlds; the gazebo kitchen world and a custom world.

3.1 Robot Design

The robot used in this task is a very simple differential drive robot, with two regular wheels and two castor wheels for support. The robot uses two sensors a kinect and a hokuyo lidar. An illustration of the robot design can be seen below in fig 4.

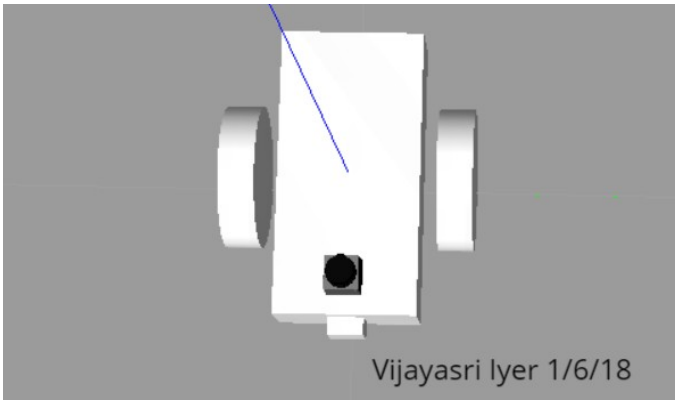

Fig. 3. Robot model

The transform tree of the robot is given below in fig 4. A transform tree is a structure that helps understand the linkages of the robot and their significance as well as function. This transform tree was generated using the tf view_frames package in ROS.

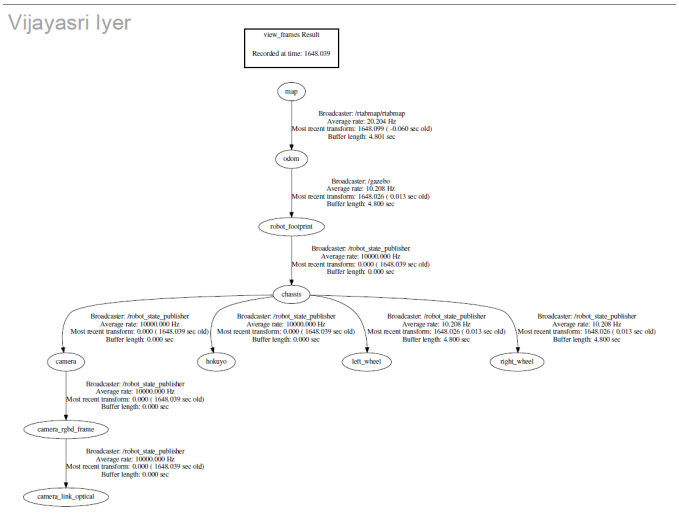

Fig. 4. Transform tree

3.2 Personal World

The custom world was a simple 3x5 room with a lot of furniture consisting of a cabinet, bookshelf, cafe table and a cardboard box among others. All of the models were taken from the gazebo model database, and the room was constructed using the gazebo building editor. An illustration of the custom world can be seen below in fig 5.

An image of the kitchen dining gazebo world is also given below in fig 6.

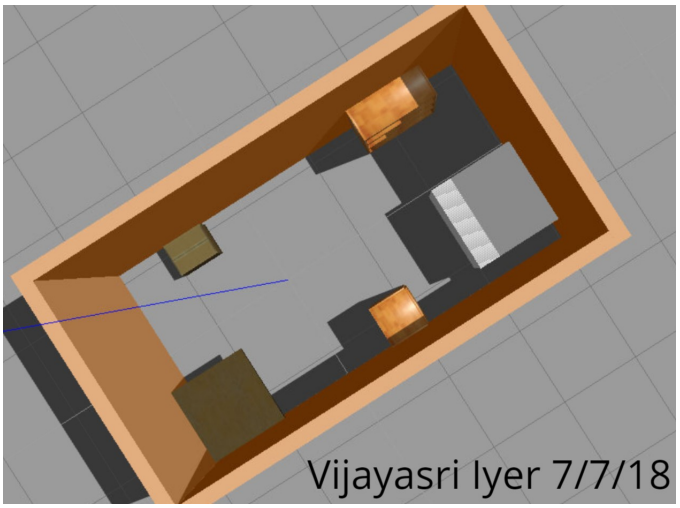

Fig. 5. Custom World

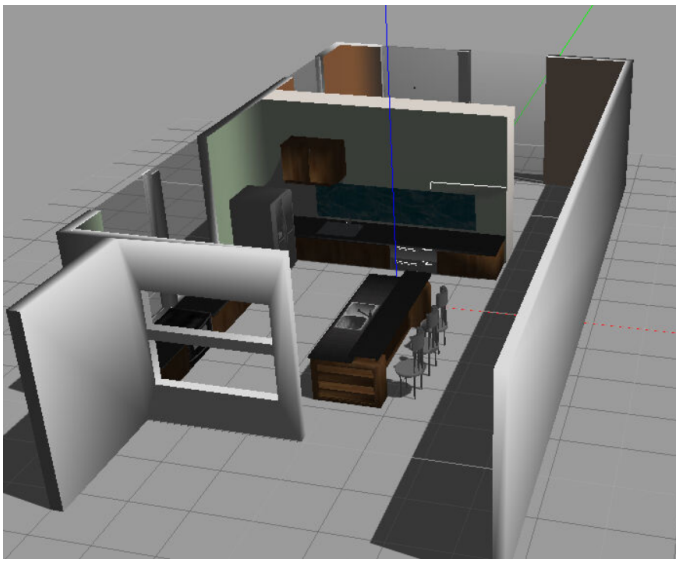

Fig. 6. Kitchen World

4 Results

The result 2D and 3D maps for both the world are given in fig 7, 8, 9, 10. The robot was able to successfully generate maps of both the worlds. The mapping of both the robots took some time to reach a comprehensive map that resembled the environment (15 mins for the kitchen world and 10 mins for the custom world). The number of global loop closures made are 51 for the kitchen world and 11 for the custom world. This information along with the maps was extracted from the rtabmap database viewer tool.

5 Discussion

The maps of both the worlds generated by the robot was comprehensive, but time consuming. This could be due to the complexity of the environment or the way in which obstacles are placed in the world. Even in case of the custom world, which almost half the size of the kitchen world, it was difficult to generate map in little time, due to the positioning of the obstacles. This indicates that the robot is unable to generate high accuracy maps in complex environments, hence it is best suited for indoor SLAM. This method of mapping is also computationally expensive, hence, it is best performed with a GPU accelerated compute platform such as the Jetson TX2. Overall, this algorithm is ideal for home service robots, working in confined and safe environments with high compute power.

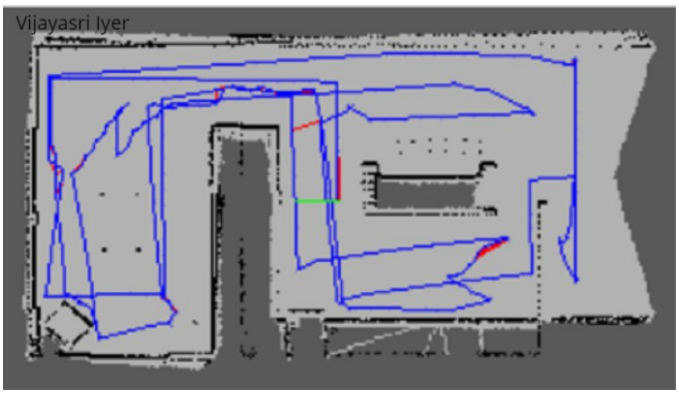

Fig. 7. Kitchen World 2D

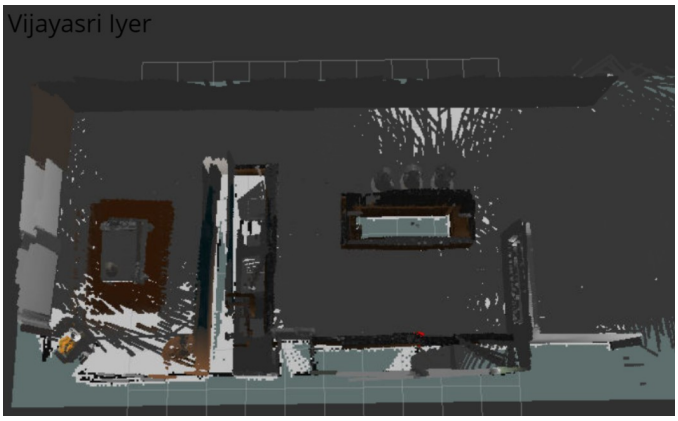

Fig. 8. Kitchen World 3D

5.1 Topics

- In which world did robot performed better?
  The robot performed better in the kitchen world.
- Why it performed better? (opinion)
  Since it was given a lot of time and had a better mapping trajectory to perform loop closures which affects the overall quality of the generated map.
- What types of scenario could SLAM be performed?
  As mentioned in the discussion section, this type of 3D SLAM is ideal for indoor environments.
- Where would you use SLAM in an industry domain?
  SLAM can be used in industry warehouses for navigation.

6 Conclusion / Future work

The results of this project are viable to create a basic home service robot which can work in dynamic environments. The future versions of this project could have features such as

Fig. 9. 3D

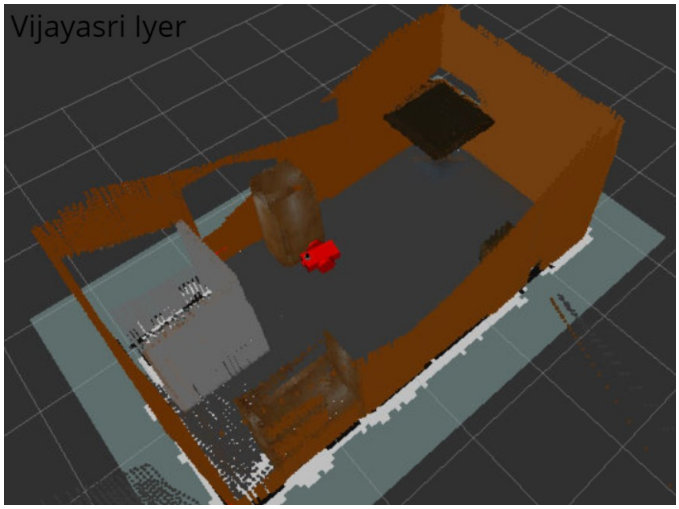

Fig. 10. Kitchen World 3D

object recognition and voice recognition with AI augmentation for the robot and deploying it in a more complex home environment. This will allow it to perform pick and place, detect the objects/people in a area/room or take voice commands from the user for specific tasks thus adding to its functionality.

6.1 Modifications for Improvement

- Use a different base configuration i.e larger, circular/rectangular base.
- Adding a separate camera for object detection.

6.2 Hardware Deployment

1) What would need to be done?
   The Jetson TX2 can be used for the SLAM task, and the setup instructions for the dependencies are provided by NVIDIA itself.
2) Computation time/resource considerations?
   Since the Jetson is a GPU accelerated platform, computation will be relatively easy and hence, less time consuming.