SIMULTANEOUS LOCALIZATION AND MAPPING WITH RTAB-MAP

Salabson Salabiya Isa

Abstract

This paper presents Real-Time Appearance-Based Mapping (RTAB-Map) implementation of Simultaneous Localization and Mapping (SLAM) to construct both 2D occupancy grid map and 3D octomaps from the provided simulated environment as well as personal built environment using mobile robot. RTAB-Map SLAM algorithm uses data collected from vision sensor to localize the robot and map the environment and also uses a process called loop closure to determine whether the robot has seen a location before.

Key words: SLAM, Robot, RTAB-Map, Loop closure

1 INTRODUCTION

Solving SLAM problem consist of estimating the robot trajectory and the map of the environment as the robot move in it. Due to inherent noise in the sensor's measurement, a SLAM problem is usually describe by means of probabilistic problems. In SLAM, the robot acquires a map of its environment while simultaneously localizing itself relative to the map. SLAM is significantly very difficult problem, it is more difficult than localization in that map is unknown and has to be estimated along the way. It is also more difficult than mapping with known pose, since the poses are unknown and have to be estimated along the way.

In this paper, SLAM was performed on two simulated 3D worlds where robot was able to localize itself as well as mapped the respective world using RTAM-Map implementation in ROS. The supplied environment is called kitchen-dining while the personal built environment is called café.

2 BACKGROUND

Mapping is required to map both static and dynamic environments. Mapping is required in static environment since measurements are always noisy and accumulate errors and may at times produce a low accuracy map. So it is always better to map a static environment with aim to correct the noise to obtain a high accuracy map. In dynamic environment robots should perform instantaneous mapping even if map is available and update its map in order to adapt to changing environment. However, mapping with mobile robot is a challenging problem for following reasons; unknown poses, huge hypothesis space, size of the map, noise, perceptual ambiguity, and when a robot travels in a cyclic manner.

So far, you've seen key challenges of the SLAM problem. Now you will see how to overcome these challenges using various approaches algorithms discussed below.

2.1 **Occupancy Grid Map**

In SLAM, map is unknown and we also assume unknown poses to estimate the map and the poses relative to it. Estimating the map with unknown poses is challenging problem because of the large number of variables available. This can be solved using the occupancy grid mapping algorithm.

The occupancy grid mapping algorithm implements binary bayes filter to estimate occupancy value of each cell. Initially the algorithm take the previous occupancy of the cells, the poses and the measurements as parameters. The occupancy grid map then loops through all the grid cells. For each of the grid cells, the algorithm test if the cells are currently being sense by the range finder using measurement cone techniques. The cells that fall under the measurement cone are highlighted in white and black color as shown in the image below (figure 1). When looping through all the cells the algorithm will consider those black and white cells as cells fallen under the perceptual field of the measurement. Then, the algorithm will update the cells that fall under the measurement cone by computing new belief using the binary bayes filter.

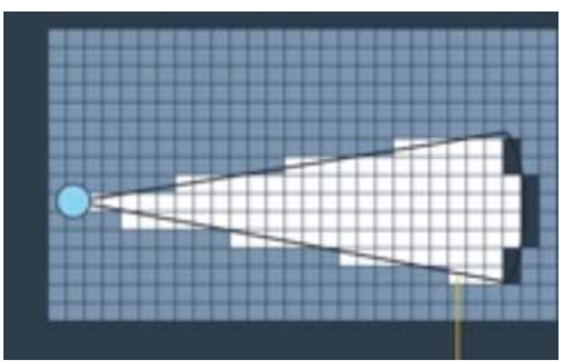

Figure 1: Measurement cone

2.2 **Grid based FastSLAM**

The FastSLAM algorithm solves the full SLAM problem with known correspondences. FastSLAM estimates a posterior over the trajectory using a particle filter approach by using a low dimensional Extended Kalman filter to solve independent features of the map which are model with Gaussian. Since FastSLAM uses a particle filter approach to solve SLAM problems, it is therefore capable of solving both the full and online SLAM problems. FastSLAM has three variances which include FastSLAM 1.0, FastSLAM 2.0 and Grid-based FastSLAM. FastSLAM 1.0 algorithm is simple and easy to implement, but this algorithm is known to be inefficient since particle filters generate sample inefficiency. To overcome this inefficiency FastSLAM 2.0 was develop which imposes different distribution that result in low number of paricles. However, both fastSLAM 1.0 and FastSLAM 2.0 algorithm uses particle filter approach to solve SLAM problem. Each particle will hold a guess of a robot trajectory, and by so doing, the SLAM problem is reduced to mapping with known poses. Therefore, this algorithm present a big disadvantage since it must always assume that there are known landmarks positions, thus FastSLAM cannot model an arbitrary environment. This is where Grid-based FastSLAM come into play. Grid-based mapping algorithm can model the environment using grid maps without predefining any landmarks positions. So by extending the FastSlAM algorithm to occupancy grid map, you can now solve the SLAM problem in an arbitrary environment.

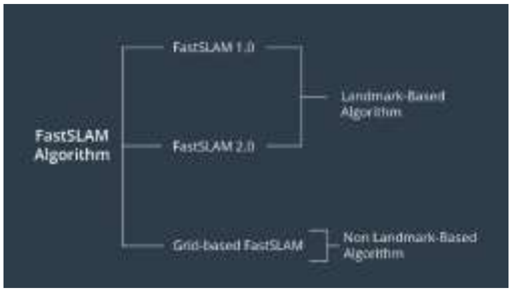

Figure 2: FastSLAM to GridSLAM Maps

2.3 Graph SLAM

Another flavor of SLAM is called GraphSLAM. GraphSLAM is a SLAM algorithm that solves the full SLAM problem. This means that the algorithm recovers the entire path and map instead of just the most recent pose and map. This difference allows it to consider dependencies between previous poses. Furthermore, GraphSLAM uses graph to represent the robot poses and the environment. A robot poses, its positions and orientations can be presented with nodes. The nodes are connected with by edges. These edges are soft spatial constraints between the robot poses. The constraint are called soft because the motion and measurement data are uncertain. The constraints will have some amount of error present. Soft constraints come in two forms. Motion constraints between two successive robot poses and measurement constraints between a robot pose and a feature in the environment.

Figure 3:Graph showing motion and measurement constraints

RTAB-Map that is used to map two simulated environments is a Graph-based SLAM approach. RTAB-Map algorithm uses data collected from vision sensors to localize the robot and map the environment. RTAB-Map has front-end and back-end structured. The front-end of RTAB-Map focuses on sensor data used to obtain the constraints that are used to for feature optimization. The back-end of RTAB-Map include graph optimization and assembly of an occupancy grid from data of the grid.

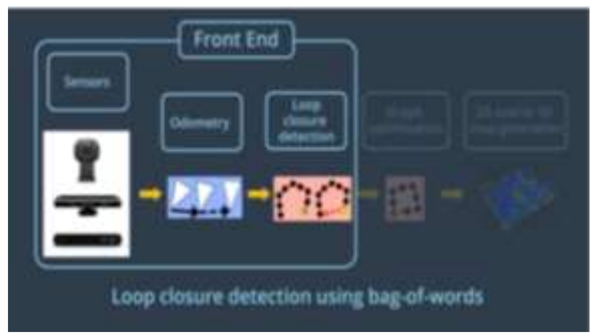

Figure 4:3D SLAM with RTAB-Map

3.0 **ROBOT MODEL CONFIGURATION**

The robot model used was based the udacity_bot created in the previous localization project. The robot model has square base with two actuators for the left and right wheel respectively. However, some adjustment were made by replacing the RGB-camera was replaced with RGB-D camera to mimic Kinect camera in order to measure depth. This is necessary for mapping with RTAP-Map. Then the new frame was added to the robot model to address for problem of RGB-D point clouds pointing up.

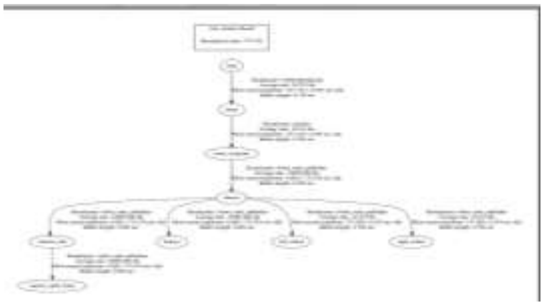

Figure 5: Robot TF tree

The new gazebo word was created from predefined model from gazebo database. In addition, more objects were added to obtain rich feature environments.

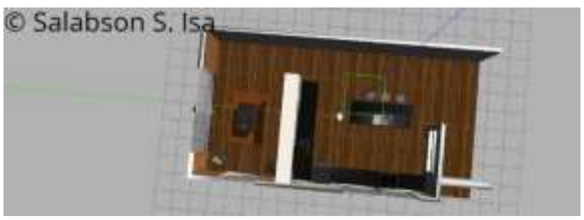

Figure 6: Kitchen_dining world

Figure 7:Cafe world

4.0 **RESULT**

Below images are the results of mapping the two worlds that is the kitchen dining and the café. Figure 8 is the 3D map of the kitchen world while figure 9 represent 2D map of the world. Furthermore, figure 10 and 11 represent 3D and 2D world of café respectively.

Figure 8:3D Map Kitchen world in Rviz

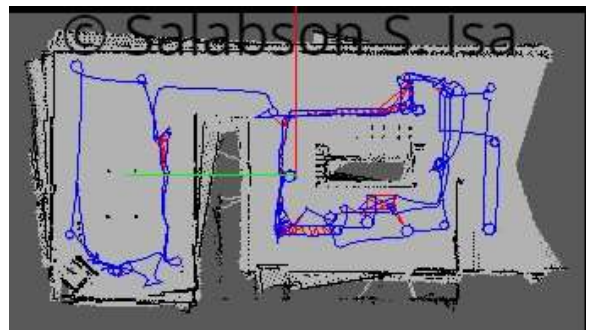

Figure 9:2D Map Kitchen world

Figure 10: 3D Cafe world

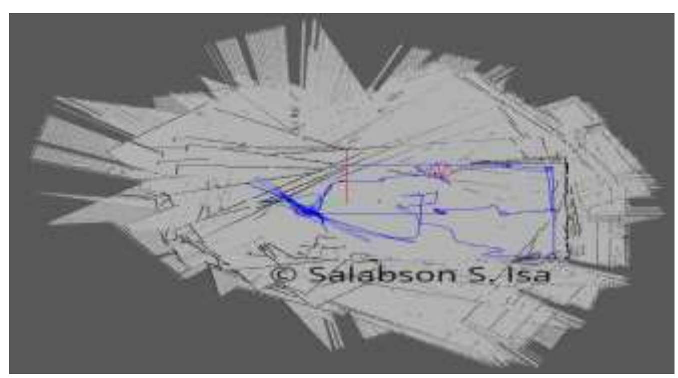

Figure 11: 2D Cafe World in Rviz

5 **DISCUSSION**

The mapping for the kitchen dining world went well. It produced better maps for both 2D and 3D.

On the other hand mapping for the custom café world do not produce good maps for both 2D and 3D as there was a lot of noise due to lack enough rich-features in the world.

6 **CONCLUSION AND FUTURE WORK**

In this project RTAP-Map SLAM mapping implementation in ROS was used to 2D and 3D maps two simulated Gazebo environments.

In future, RTAP-Map algorithm will be applied to work on hardware platform.

References

Sebastian Thrun, Michael Mentemerlo. The GraphSLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures

Dieter, Sabestian Thrun and Wolfram Burgard. 2005. Probabilistic Robotics.

www.udacity. 2018. Robotics Software engineering classroom.