Map My World

Seyfi Gozubuyuk

Abstract—In this project, the aim is to utilize ROS packages to create map of two different environments. A mobile robot will navigate in the Gazebo and RViz simulation environments to obtain data. The overview of the tasks can are as follows, creating map for the given environment, create a new environment and build the map of the new environment using RTAB-Map ROS package.

Index Terms—Robot, IEEEtran, Udacity, LaTeX, SLAM, RTAB-Map, ROS, Robot Model.

1 Introduction

The problem is to create the map of an environment while the robot is navigating. The robot model, xbot, was built in the previous project, Where Am I [1]. In the localization problem, the robot knows the map, and the sensor measurements help the robot to localize itself. On the contrary, the map is not available in the mapping or the SLAM problem. SLAM stands for Simultaneous Localization and Mapping [2], and it needs to obtain the map to localize the robot and to localize the robot to create the map. The most common algorithms are Fast SLAM and Graph SLAM. This project test the RTAB-MAP (Real-Time Appearance-Based Mapping) [3]. There are two different worlds to test the algorithm. The first one is the kitchen dining world provided in the lessons. The second test world is xworld, which was developed as a part of the project. The project environment contains Gazebo and RViz on ROS Kinetic, which was running on Ubuntu 16.04. The ROS packages required to run this project are RTABMap. In addition, the provided scripts teleop and rtab_enable the project to run.

2 Background

The problem is to create a map of the environment, as mentioned earlier. Furthermore, localization of the robot is a requirement to complete the task. Since the sensors are not precise, there is noise to affect the measurements coming from the sensors, and the controls are not perfect to move the vehicle to the desired location. Therefore, the algorithm should handle the noise.

2.1 Occupancy Grid Mapping

Occupancy Grid Mapping is an algorithm that assumes the location of the robot is available when constructing the map [4]. During SLAM operation, then the algorithm builds the map of the environment and localize the robot relative to it. After, the algorithm uses the exact robot poses filtered from SLAM. Then, using the known poses from SLAM and noisy measurements from the sensors generates a fit for path planning and navigation.

2.2 Grid-based FastSLAM

FastSLAM estimates a posterior over trajectory using particle filters. It solves the Full SLAM problem. Full SLAM looks for the entire path up to time t, using all of the controls and measurements. It has the advantage of mapping with known poses and utilizes low dimensional EKF to solve independent features of the map.
Grid-based FastSLAM uses Monte Carlo Localization (MCL) instead of EKF. It first estimates the trajectory, then the map by assuming known poses and applying the occupancy grid mapping algorithm.
In the Grid-based FastSLAM technique, there are three steps. First one is sampling motion. In this step, the current location of a given particle is calculated using the previous one and the current controls. In the second step, which is map estimation, with the inputs of current measurements, the particles current location and the previous map, the next map is estimated. The last step updates the weights of the particles.

2.3 GraphSLAM

GraphSLAM uses a graph representation to the SLAM problem. It constructs graphs and then matrices. They help to determine different observations showing the same landmark. [5]. The main feature of GraphSLAM is graph optimization with maximum likelihood estimation (MLE). The procedure is as follows

1) Remove inconsequential constants.
2) Convert the equation from likelihood estimation to negative lo likelihood estimation
3) Calculate the first derivative of the function and set it to zero to find extrema.

The properties of the information matrix and vector are as follows

- A motion constraint ties together two poses.
- A measurement constraint ties together one feature and one pose.
- Each operation updates four cells in the matrix and two cells in the vector.
- All other cells are zero, therefore the matrix is sparse. Sparsity helps solving equations on limited hardware.

3 SIMULATIONS

The simulation environment consists of ROS (Kinetic), Gazebo and RViz. The main ROS package is the RTAB-Map package. There are two different environments to test on the simulation. The robot model in the tests is the xbot from the Where Am I project. The robot has been upgraded with an RGBD camera. It still has the hokuyo laser sensor. The inputs to the RTAP-Map package are the laser scan data and the raw image data from the RGBD camera.

3.1 The Kitchen Dining

The xbot has created the map of the Kitchen Dining world. Figure 1 shows the world and xbot on Gazebo. After the mapping completed, the visualization of the database can be seen from the Figure 2.

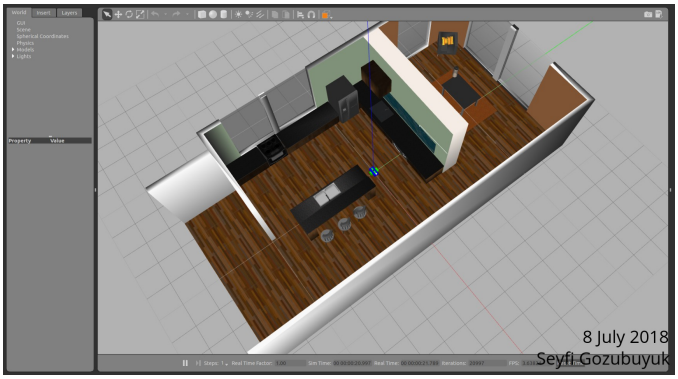

Fig. 1. Kitchen Dining in Gazebo

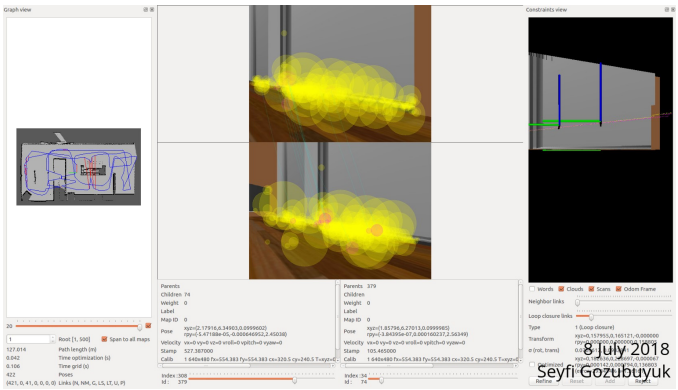

Fig. 2. Kitchen Dining in RTABMap DB Viewer

3.2 The X World

The X World was built as the custom environment. To increase mapping accuracy, different types of objects were distributed to the map. Figure 3 shows the custom world in Gazebo. The map from the mapping process is shown in Figure 4. The screenshot from the RTAPMap database visualization is in the Figure 5.

3.3 Achievements

The algorithm was able to create maps for both of the environments, however, the maps were not perfect. There were little rotations and noises on the maps. Figure 6 shows the frame, and Figure 7 shows the topics and the ROS graph.

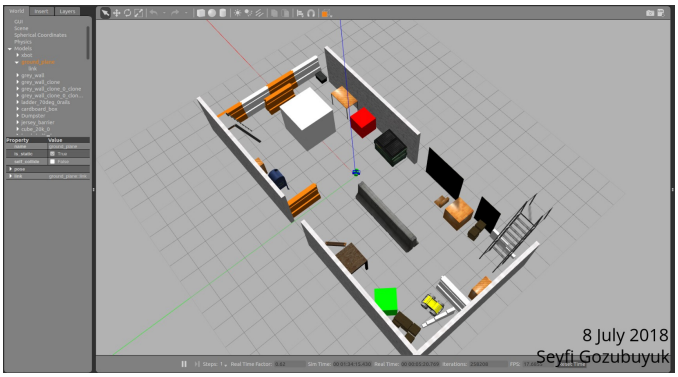

Fig. 3. X World in Gazebo

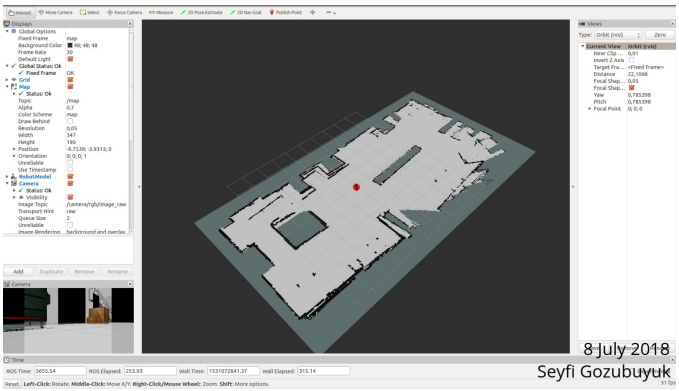

Fig. 4. X World in RViz

4 RESULTS

4.1 Mapping Results

4.1.1 *The Kitchen Dining*

The map as an image file is shown in Figure 8. The image file does not show an image as successful as the RTAPMap viewer. There exist rotations in the map.

4.1.2 *xbot*

Figure 9 shows the image file saved with map_server map_saver. Again, the saved map is not as successful as the image on the RTAPMap viewer. The map is skewed from the horizontal middle line.

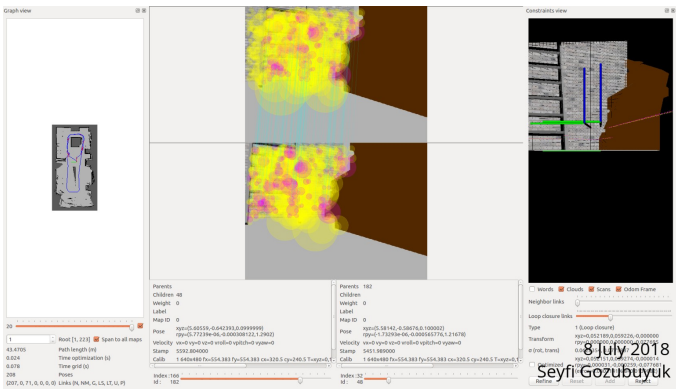

Fig. 5. X World in RTABMap DB Viewer

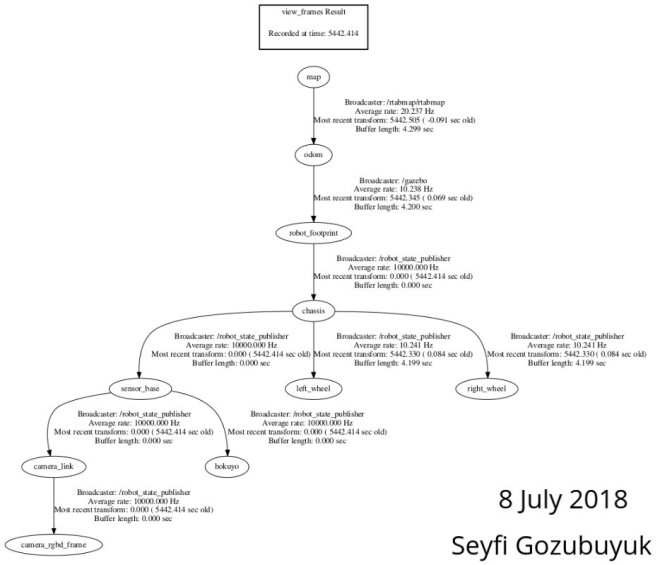

Fig. 6. TF Frames

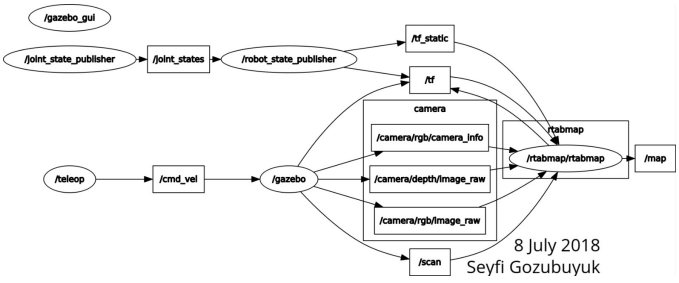

Fig. 7. ROS Graph and Topics

4.2 Technical Comparison

The first world is a more realistic world. It has objects to serve as landmarks which lead to a better mapping for the algorithm. There are several objects in the custom map, but still, it is harder to create the map of the x world.

5 Discussion

The algorithm was able to create a map for each environment. The robot navigated for almost two laps in both of the worlds. Both of the map images look warped. The parameters for the algorithm should be optimized, but due to time limitations, the tunning operation would be future work.

6 Conclusion / Future work

It is required to tune the parameters for mapping. Besides, a more complex environment should be created for further tests.
Testing the algorithm on the Jetson TX2 and comparing the results with the current results is a waiting task for the future work. Upon receiving successful results, the algorithms in the project can be used on a real robot such a vacuum cleaner or an autonomous forklift.

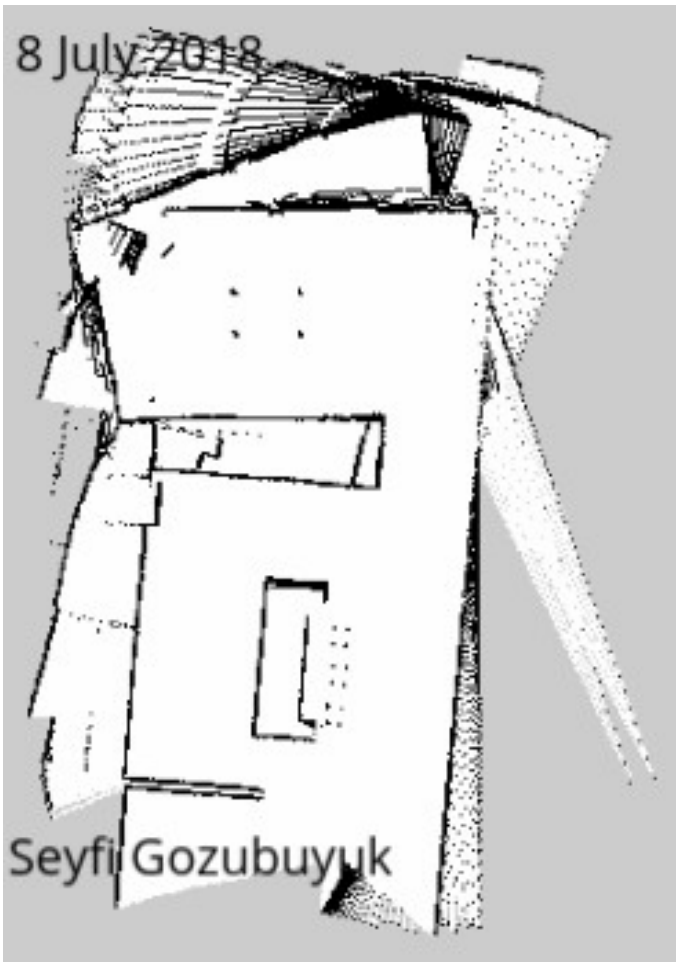

Fig. 8. Kitchen Dining Map

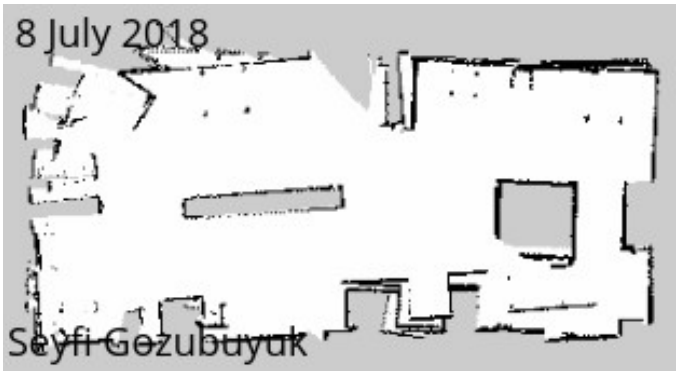

Fig. 9. X World Map

6.1 Hardware Deployment

The project was deployed in the simulation environment. The machine has an I7 CPU, 8GB RAM, and 650M GPU. The configuration was able to run the project, but the algorithm and simulation become slower for complex worlds. With more powerful CPU and GPU, it will be possible to increase the complexity of the environment.
More powerful hardware will enable to test on different and larger custom worlds. Using a Jetson TX2 would increase the performance of this project.

REFERENCES

[1] Seyfi Gozubuyuk, "Where am i localization project," 2018. [Online; accessed 07-July-2018].
[2] SLAM, "Simultaneous localization and mapping — Wikipedia, the free encyclopedia," 2018. [Online; accessed 07-July-2018].
[3] IntRoLab, "Real-time appearance-based mapping," 2017. [Online; accessed 07-July-2018].
[4] Occupancy Grid Mapping, "Occupancy grid mapping — Wikipedia, the free encyclopedia," 2016. [Online; accessed 07-July-2018].
[5] Graph SLAM, "Graph slam — Wikipedia, the free encyclopedia," 2016. [Online; accessed 07-July-2018].