Mapping Project: Map My World Robot

Jianfeng Yang

Abstract—In this project, attempts were made to map two simulated worlds in the Gazebo and RViz simulation environments, utilizing ROS packages. Two simulated worlds were used: the first one was provided by Udacity, and is a simulated environment containing a kitchen and a dining room; the second one was created from scratch inside Gazebo. Several aspects of robotics were learned and discussed, with a focus on mapping with the ROS package RTAB-map. A ROS package was created, which launches a custom robot model in a Gazebo world and utilizes packages like RTAB-map and the keyboard-teleop. A significant amount of time was devoted to the actual mapping since the movement of the robot is purely manual. The robot has successfully mapped the two worlds and generated two databases.

Index Terms—Robots, Mapping, RTAB-map, ROS, Keyboard-teleop, Mobile Robots, Gazebo, RViz, IEEEtran, Udacity.

1 Introduction

Mapping is the term used to describe the techniques in producing the map of the environment a robot is in. It goes without saying that for autonomous robots, to have an accurate map to refer its poses against is the prerequisites for navigation, and is extremely important. Unlike localization in which the environment is known, and the poses are to be estimated, mapping usually assumes a known path, and generates the environment. It is usually more complicated than localization since the map is generated in a continuous space and contains an infinite number of variables. Numerous techniques are used for mapping, and one particular is focused here namely the occupancy grid mapping. Together with localization, mapping has become a vital part of the modern technologies called SLAM. For the past few decades, SLAM has made revolutionary progress and enabled drones, self-driving cars, and other autonomous robots. [1] A particular type of SLAM algorithm called GraphSLAM is getting popularized nowadays, utilizing the latest visualization technology. More specifically, the Real-Time Appearance-Based Mapping, or RTAB-map for short, is used in this project as the 3D SLAM approach. Using loop closure detection and other methods, RTAB-map is capable of solving the full SLAM problem, meaning that it can recover the entire path and the map.

2 Background

2.1 Algorithms

Generally, mapping can be categorized by the dimension of the environments, i.e., two-dimensional mapping or three-dimensional mapping. In reality, a two-dimensional mapping is usually for a mobile robot that's traveling on a flat surface. For aerial robots like drones and others that move in a three-dimensional space, mapping can also be achieved through the occupancy grid algorithm, but the computational cost would be much larger. There are many algorithms used in mapping, and one particular will be used here: the occupancy grid mapping.

2.2 Importance of Mapping

Theoretically, mapping is performed when the environment is unknown. In practice, however, a mapping process is vital even if a map is provided since it can be highly dynamic and changing constantly. If the robot moves according to the provided map without performing instantaneous mapping of its environment continuously, it could quickly get stuck or even lost.

2.3 Challenges and Difficulties

There exist a significant number of challenges and difficulties in a mapping process. First of all, unlike the situation in localization, in mapping, neither the map nor the poses of the robot are known. Secondly, because of the continuous nature of the map, the hypothesis space is much larger, especially when the robot is deployed to an open space where there might be an infinite number of unknown objects. Last but not the least, the characteristics of the environment also poses difficulties on the job. For example, a large environment usually contains a large set of data, which might be a challenge to the onboard processor of the robot. Besides, noisy measurement and cyclic movement of the robot also creates inaccuracy in mapping. Finally, there might be scenes and objects that are similar inside the environment, cause a difficulty named perceptual ambiguity. It confuses the robot, making it think that it has been to a traversed area, which is unexplored.

2.4 Occupancy Grid Mapping

The basic idea of occupancy grid mapping is to uniformly divide the dimension space into a finite number of grids, and mark them with binary values based on the measurement of the sensors. Integrating the status of each cell will generate an estimated map of the environment.

3 Simulations

In this project, two simulated gazebo worlds were mapped. First one is a provided kitchen and dining room environment; the second is a customized world that contains many different objects and will be discussed in detail later.

3.1 Achievements

The robot has successfully mapped both worlds.

3.2 Configuration

3.2.1 Robot Extension

As instructed in the lessons, the robot is an upgrade from the customized roborock robot used in the localization project. More specifically, the RGB camera was upgraded with an RGBD one.

3.2.2 Package Structure

The package structure is the same with the localization project.

3.3 Result

3.3.1 Provided World

Initially, the robot is placed at the center.

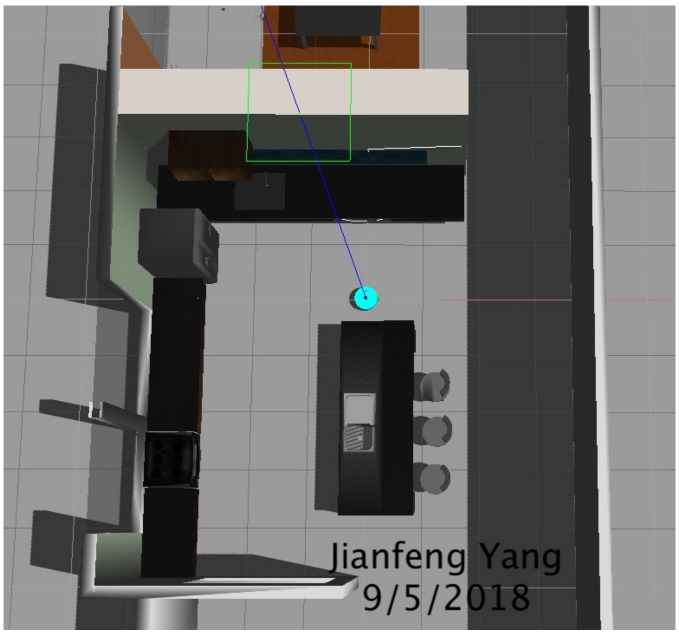

Fig. 1. Kitchen and Dining Initially Loaded

As the robot traverse the environment, navigated by keyboard, an occupancy grid and 3D map were generated. This is the map generated when the robot is roughly half way through:

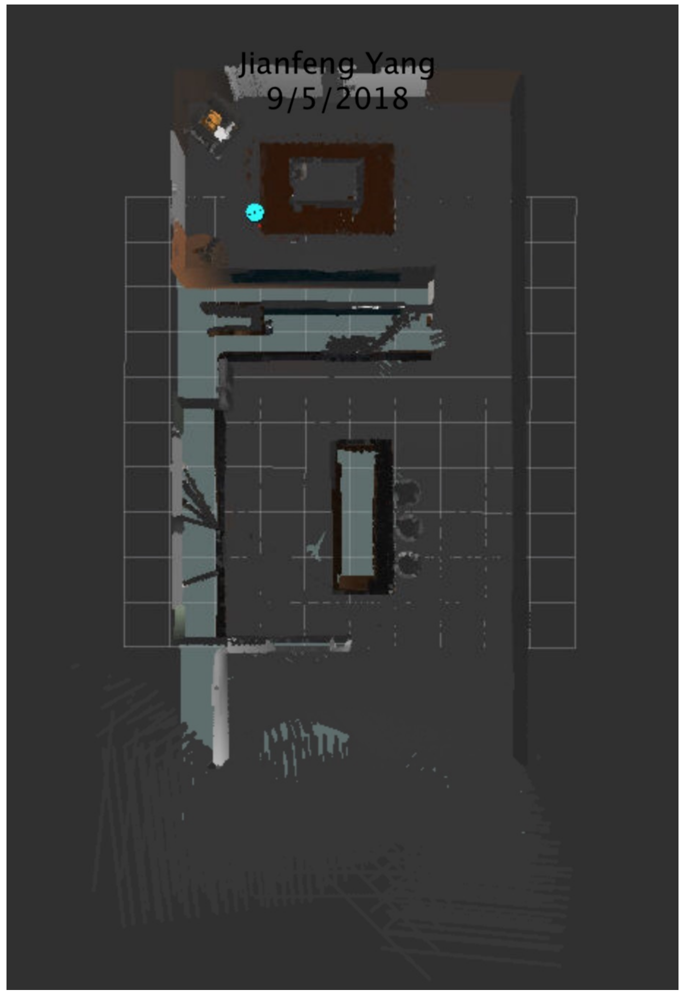

Fig. 2. Kitchen and Dining, Half Way Mapped, Top View

From another angle:

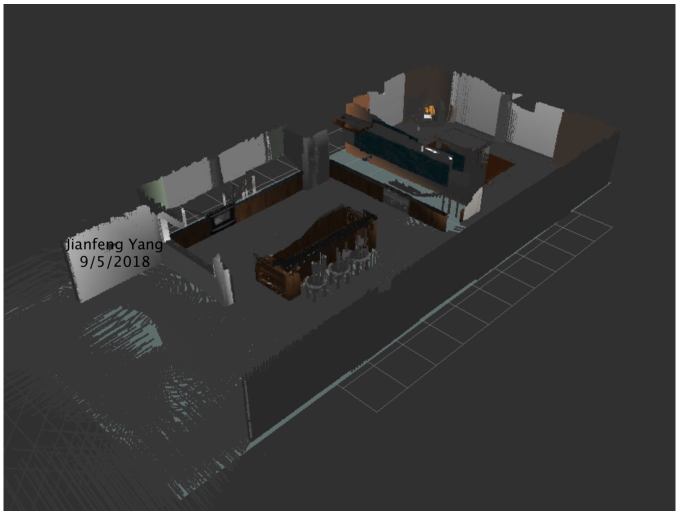

Fig. 3. Kitchen and Dining, Half Way Mapped, Perspective

And this is the whole environment mapped:

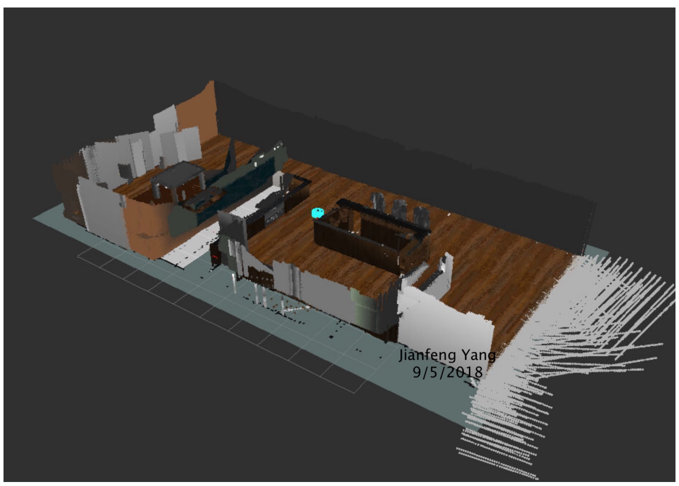

Fig. 4. Kitchen and Dining, Fully Mapped

As the following result shows, the environment is feature rich, and there are many features in common between two corresponding shots.

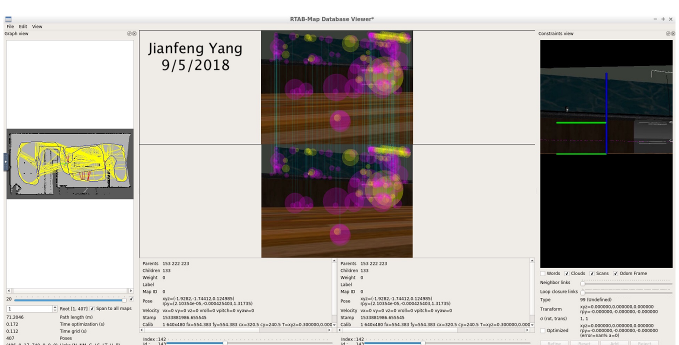

Fig. 5. Kitchen and Dining, Database Viewer

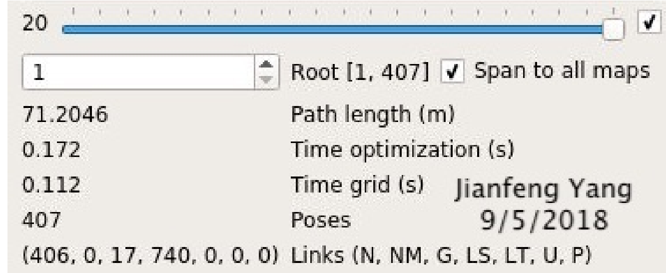

Fig. 6. Kitchen and Dining, Links and Loop Closures

3.3.2 *Personal World*

As shown in the top-view screenshot, this world is filled with all kinds of objects. The world is a squared space, surrounded uniformly by concrete walls. There are four openings at the center of each side of the square, leaving the robot a path to move in and out of the world. One of the openings is partially blocked by a post, and another is surrounded by two other objects; the rest of them are left clear. All of these differentiations are meant to create a diversified environment for the robot.

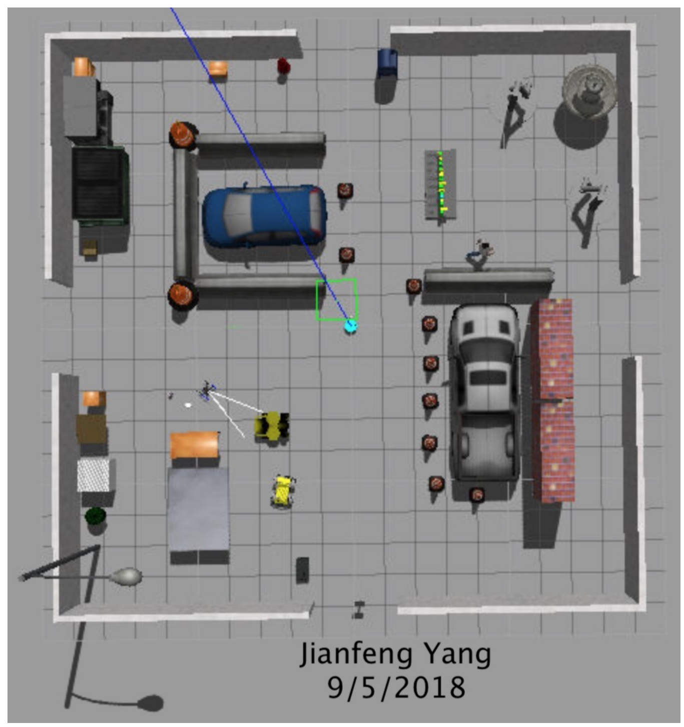

Fig. 7. Personal World, Top View

Inside the world, space is roughly divided into four sections. Each of them is filled with objects that are of various sizes and shapes, and of different materials. Some of them, like the truck, all the tables in the bottom left section, offers space underneath them where the robot can travel through. Moreover, in the top right section, several dynamic objects are always moving. These different kinds of objects create more variance for the mapping.

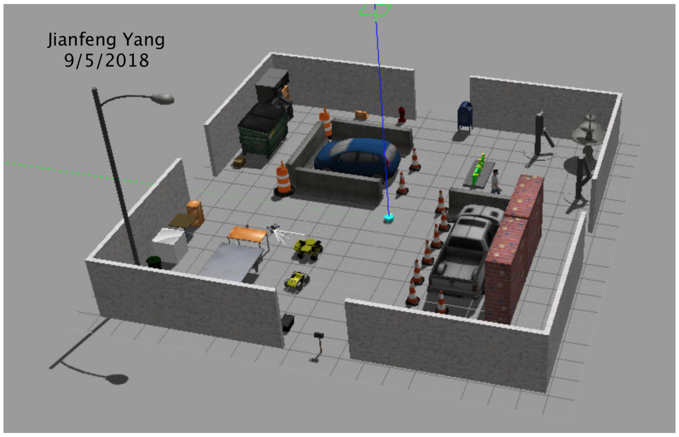

Fig. 8. Personal World, Perspective

The robot has gone the same process to map the personal world. And here is the result:

Fig. 9. Personal World, Fully Mapped

And here shows the database viewer. As the result shows, the environment is feature and loop closure rich.

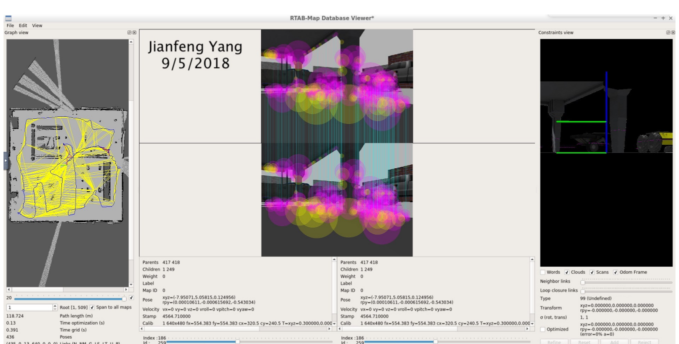

Fig. 10. Personal World, Database Viewer

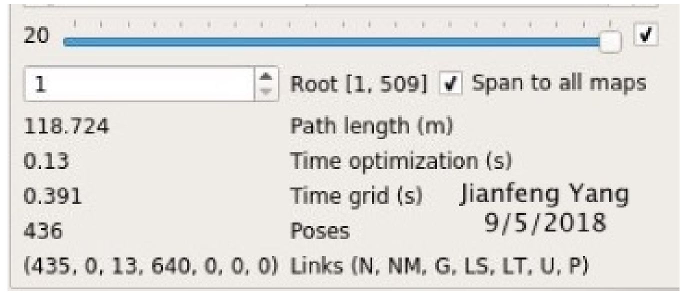

Fig. 11. Personal World, Links and Loop Closures

4 Discussion

4.1 Difficulties and Problems

Overall, this project is straightforward and can be completed by following the instructions. However, two significant obstacles halted the progress for a painfully long time.

4.1.1 Wrong Direction of Optical Frame

As mentioned in the common question section of the lecture, if the same configuration for the robot from the localization project was used, it is highly possible that the direction of the optical frame is wrong. More specifically, the point cloud for the 3D map will appear upwards in RViz, which should be horizontal.

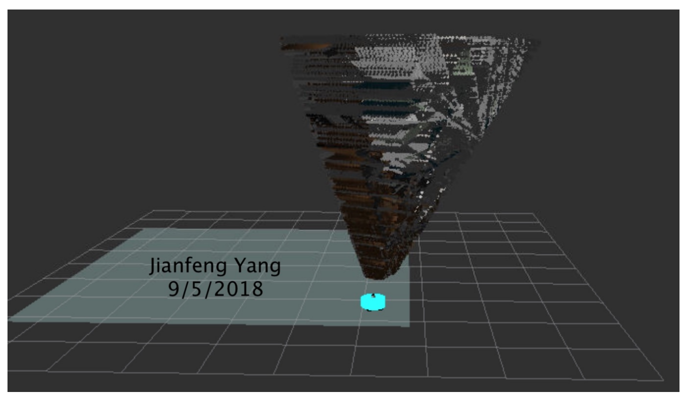

Fig. 12. Point Cloud Projecting Upwards

Though the way to fix this was readily presented in the common question section, it was not an easy fix. The correct way was to add an extra dummy frame between the joint and the camera, but I misunderstood the instruction and kept trying to modify the camera frame. It went without saying that I failed, miserably, until after a long time when I finally realized that the key was the extra frame.

4.1.2 Missing Physics for the Given World

Another major setback was the missing physics for the given world. The kitchen and dining world that was initially stored in the Udacity workspace was not correctly configured and lacking the physics definition; thus all the objects in the worlds have no collisions and other physical characteristics. The result is that the mapping robot can roam freely in the world, moving through walls and objects like a ghost.

This was not only visually frustrating but also severely crippling the mapping result. Since there were no collisions, the laser beam from the robot has no reflection, and always went to infinitely far. This essentially made the laser sensor useless, and no 2D grid map was ever generated. It took me countless time and effort to figure out the cause and solution eventually. Once the physics was in effect, the mapping process works flawlessly.

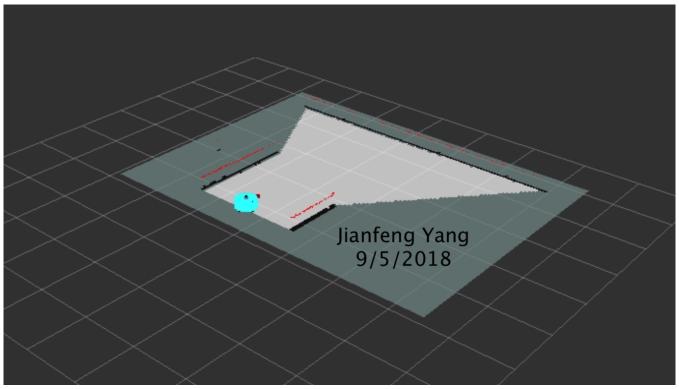

Fig. 13. Correct Display of the Laser Reflection and Grid Map

5 Conclusion / Future work

In this project, two simulated worlds were mapped using a manually navigated robot. First the given world, then a customized one that was built from scratch. The robot has successfully mapped both worlds with a lot of links and loop closures being found.

5.1 Modifications for Improvement

Due to time constraints, there are still a lot more improvements remaining unexplored.

- Add Navigation Stack
  As mentioned previously, during both mapping process, the robot was navigated via keyboard control. For a more advanced simulation, the navigation stack can be added, making it a true SLAM experience.
- Mapping Package Parameters
  It is worth noting that no parameter tweaking was involved in this project, and the robot has to go a long way to generate a full map. Given more time, a carefully tuned mapping stack should produce a more accurate result in less effort.

References

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. Leonard, "Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, p. 13091332, 2016.