Robotics Software Engineer Nanodegree: Mapping Project

Lincoln Stein

Abstract—Mobile robot navigation is often dependent on an accurate map of the environment the robot is operating in. While in some cases, a static map may be sufficient if the environment will not change. However, most environments change periodically and so to accurately and efficiently navigate through a changing environment the robot must have the ability to create a map. There are several different methods to implement mapping, this project focuses on implementing The goal of this project is to create a ROS package that successfully implements environment mapping on a mobile robot in simulation. The robot is tested in two simulated environments

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

1 Introduction

The purpose of this project is to create a ROS package that successfully implements a robot capabale of performing Simultaneous Localization and Mapping (SLAM) using Real Time Appearance Based (RTAB) Mapping. SLAM attempts to solve two problems at once by creating a map of the environment and localizing the robot within the map as it is created. This is especially applicable to robots tasked to explore new environments or those working in environments that frequently change and require adaptation by the robot. SLAM requires successful fusion of several sensor domains and the suceful integration of sensors and information flow is a key requirement of this project.

2 Background

2.1 2D/3D Mapping

The SLAM algorithm implemented in this project provides both 2D and 3D mapping capabilities. The necessity of a 2D vs 3D map depends on the tasks assigned to a mobile robot. Mapping in 2D is generally useful for navigation, such as the development of GPS guidance systems and autonomous cars. In more complex applications, such as robots with higher degrees of freedom like a drone or a robotic arm, 3D mapping becomes necessary to enable the robot to perform successfully. In order to implement SLAM, certain requirements must be met. The robot must have a way to track its own location, detect features within the environment around it relative to its own location. The robot developed in this project utilizes wheel encoders, an RGB-D camera, and a scanning laser distance sensor. The wheel encoders are used to track movement within the environment and the RGB-D and laser scanner provide imagery and relative distances to features within the environment. There are two different methodologies used when approaching the problem of SLAM currently, online and full. Online SLAM algorithms provide an estimate of the map as well as the current pose of the robot. Full SLAM algorithms provide an estimate of the map and the entire trajectory of the robot within the environment. As with other problems in robotics, there are several solutions that present different advantages and disadvantages in solving the SLAM problem. Building up from localization algorithms, one might attempt using particle filters for estimate robot trajectory and mapping. FastSLAM is a particle based approach that is able to solve both full SLAM to estimate the full robot path as well as the Online SLAM problem as each particle estimates the robot's instantaneous pose. As in localization algorithms, a particle holds an estimate of the robot trajectory. With this trajectory estimate, the final piece of SLAM becomes mapping with known poses using occupancy grid mapping. To extend the technique to any arbitrary environment, it is necessary to remove particle filters reliance on landmarks. One method of doing so is by introducing a grid based map, creating a alternative way to relate particles. Another algorithm is GraphSLAM. As the robot travels, GraphSLAM creates a graph represented in a matrix that contains robot poses, features detected within the environment, motion constraints(pose and pose) and measurement constraints(feature and pose). This graph is then analyzed to solve the system of poses and features given the constraints. This is a way that GraphSLAM is able to improve accuracy compared to FastSLAM as it uses all the information available to find the optimal solution and not approximate guesses from lossy filtering. Real Time Appearance Based(RTAB) Mapping implemented in this work is a form of GraphSLAM that relies on appearance to determine when the robot is in the same location, known as a loop closure. Loop closure allows the algorithm to compensate for noise and drift within sensors and apply this correction throughout the map, increasing accuracy.

2.2 Robot and Scene Configuration

As discussed, the robot must have appropriate sensors to provide data to the SLAM algorithm. The robot implemented in this work consists of a mobile base with wheel encoders, a RGB-D camera located on the front, and a scanning laser range sensor on the top. The ROS package for this project is organized into several folders based on purpose. The launch folder contains all the files needed to launch the provided cafe world with robot model, the teleop node, the rtab-mapping node, and RVIZ. Within the worlds folder is where both the provided cafe world and the custom created gazebo world are stored. The urdf folder has both the .xacro and .gazebo files necessary to describe the robots configuration. The meshes folder holds any meshes used in the robot chassis, such as the hokuyo laser scanner. The src folder contains required code and scripts, such as the teleop python script. Figure 1 shows the links between the different components of the robot physical layout. The custom robot world was created by importing models into a Gazebo session, creating the layout, then saving the created environement. It is designed to simulate a robot that is staged within a lab environment and then is sent to explore the environment outside. It presents challenges to the appearance based loop closure as there is the potential for many similar images to be captured. It consists of a buildings, other robots, and various other items throughout the environment.

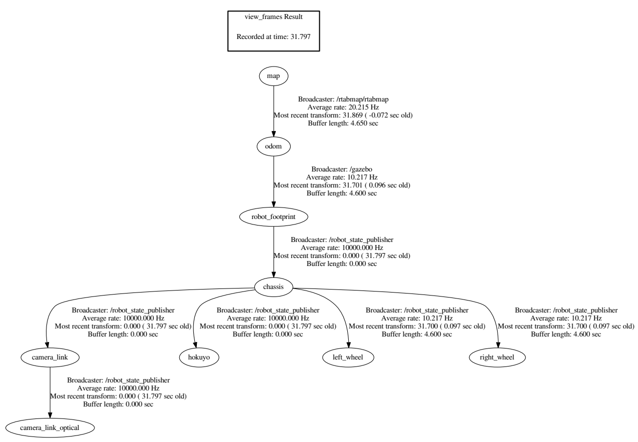

Fig. 1. Transform Tree

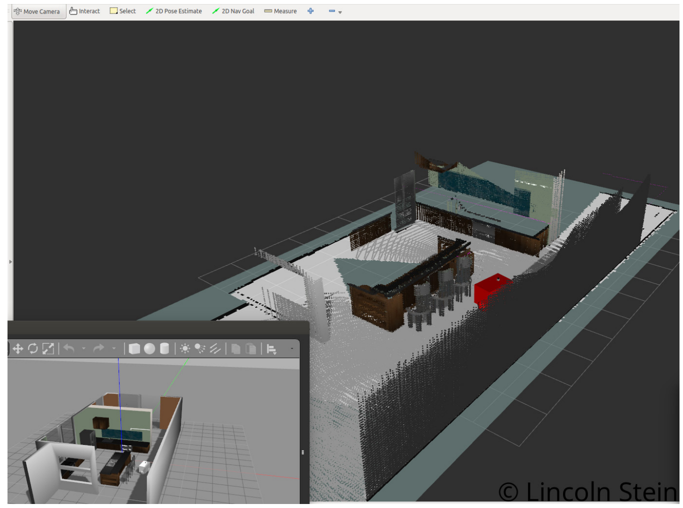

Fig. 2. Mapping The Provided Environment

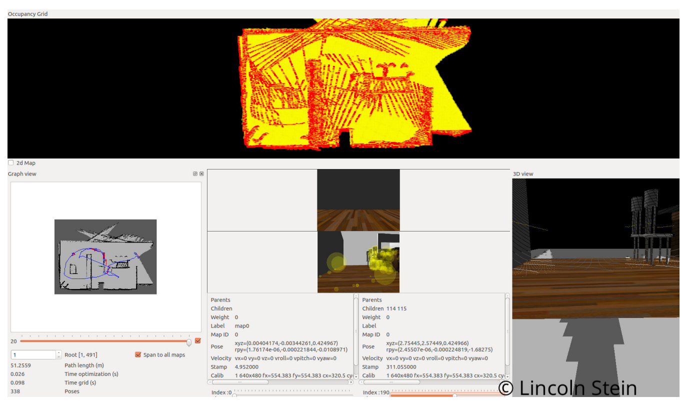

Fig. 3. Provided Environment Database Visualization

3 RESULTS

The robot performs mapping on both environments, however the current implementation struggles in the created test environment. Figure 2 shows the robot mapping the provided environment. The map shown in RVIZ is well constructed and matches the provided environment well.

Figure 3 shows the database created from the sensor data captured by the robot. The grid map can be seen to the left. The algorithm detected 85 loop closures.

Figure 4 shows the robot mapping the provided environment. The map shown in RVIZ is overlapping and doesn't match the environment as well.

Figure 5 shows the database created from the sensor data captured by the robot. The grid map can be seen to the left. The algorithm detected 258 loop closures.

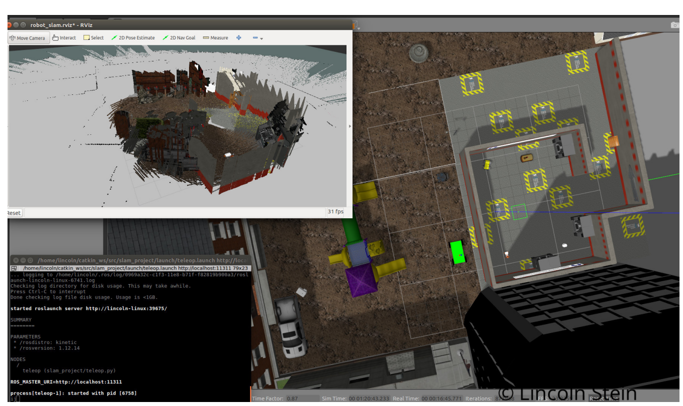

Fig. 4. Mapping The Test Environment

4 DISCUSSION

The results of mapping the provided environment display the potential of RTAB Mapping and SLAM algorithms in

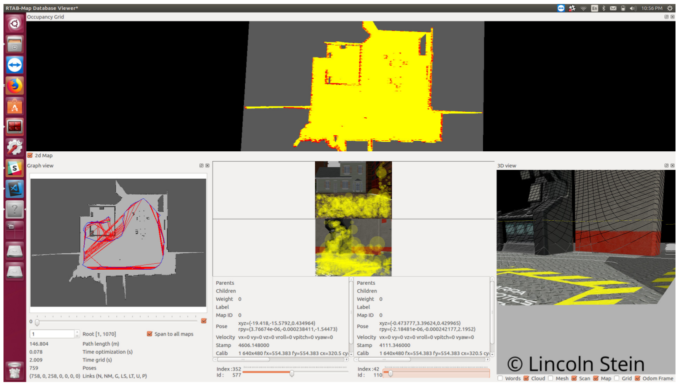

Fig. 5. Test Environment Visualization

general. The provided environment provides sufficient visual differentiators within the RGB-D camera range while traversing the environment to create unique loop closures based on appearance alone. The robot is not as effective in the much more open and visually similar test environment. Possible improvements would be to evaluate different sensors and configurations to provide more information. Other possible improvements could be gained from utilizing a different feature detector and fine tuning parameters related to loop closures.

5 CONCLUSION / FUTURE WORK

The goal of this project was creating a ROS package to implement SLAM on a mobile robot and evaluate its performance in two simulated environments. The package executes properly and implements the RTAB mapping algorithm. It creates a reasonably accurate map of the provided environment and highlights room for future improvements when mapping the test environment. The elements of this project could be applied to a wide variety of problems, from allowing a pick and place system to adapt to new parts or placements, a service robot navigating a home or office, or successfully navigating the wreckage in a search and rescue situation. Deploying this package to hardware is another goal for future work.