Udacity Project: Map My World

Brad Huang

July 25, 2018

1 Abstract

This project attempts to apply Simultaneous Localization and Mapping (SLAM) algorithms in ROS and Gazebo, and discusses how and why the SLAM algorithms succeed or fail to solve the problem. The SLAM algorithm used is Real Time Appearance Based Mapping (RTAB-Map), a type of GraphSLAM algorithm that uses traditional computer vision techniques to solve the correspondence problem. The algorithm works well in the first environment, where the map features are distinguishable from each other. However the mapping and localization accuracies were lower when the robot explores an environment with visually similar regions and failed to correlate different frames correctly.

2 Introduction

In the real world, there are few scenarios where we can provide the robots with fully featured maps to navigate through the environments. The robots would need to localize themselves and map the environments at the same time. The technology to solve this problem is named Simultaneous Localization and Mapping (SLAM). This project continues the previous project on localization, and explores how the Simultaneous Localization and Mapping (SLAM) methods works in a simulated environment. The robot would be using a RGB-D camera and estimates its trajectory and map feature poses as it moves through the environment. If the SLAM algorithm is successful, the robot would be able to produce a map with identifiable features in the surroundings and its trajectory.

3 Background

In localization, the robot is given a map, measurements and motions to estimate its own poses. SLAM is a much more challenging problem, since the only information the robot has is the measurements and motions, while it has to estimate both its own poses and the map feature poses. As a result, the solution space for the SLAM problem is highly dimensional, rendering many analytic solutions unsuitable for the task. In addition, SLAM algorithms

also need to identify object correspondences between images in order to resolve map features correctly. However, the correspondence values increase exponentially over time since the robot captures more images as it explores the environment.

The SLAM problem has two variations - Full SLAM and Online SLAM. The Online SLAM problem tries to estimate the current pose and the map features from the previous measurements and motions, while the Full SLAM attempts to estimate the entire trajectory and the map features at the same time. Many algorithms solve the SLAM problem on different scopes, including FastSLAM and GraphSLAM.

FastSLAM is a landmark-based algorithm that extends the Monte Carlo Localization (MCL) particle filter algorithm to estimate both the landmarks and robot poses. The Grid-based FastSLAM algorithm is a version of FastSLAM that combines the MCL and Occupancy Grid Mapping, a mapping algorithm that outputs grid cells labeled as occupied, free and unknown. However, either versions of FastSLAM have problems with arbitrary environments since they assume known landmark locations.

On the other hand, the GraphSLAM algorithm represents the SLAM problem as a graph where the poses of the trajectory and the measured locations are the nodes, and the links are the estimated motion and measurement distances represented as constraints. The algorithm solves for the best configuration of the graph to satisfy the constraints as much as possible, thus solving the Full SLAM problem.

This project uses the Real Time Appearance Based Mapping (RTAB-Map), an upgraded version of GraphSLAM that uses a bag of words approach to identify correspondences between frames using visual similarity. The algorithm detects loops in the trajectory with features such as SURF or SIFT. At every frame, the camera image would be compared against previously found map features in a constant-size working memory. This allows the algorithm to map trajectories with repeated locations by reducing the constraint complexity, and the algorithm can run in constant time with a proper memory management scheme.

4 Scene and Robot Configuration

4.1 Scene Configuration

A new scene is built from scratch in Gazebo. The models in the scene are all from the online model database. I first constructed a square room with four stone walls, and placed many robot parts close to the walls in the room. Then, I replaced three stone walls with different types of walls to avoid featureless scenes. No repeated objects except for the wall parts were present in the scene. In addition, I place different objects in front of the wooden cabinet and closet to facilitate the robot in distinguishing them from each other. The room was then left with enough room for the robot to explore.

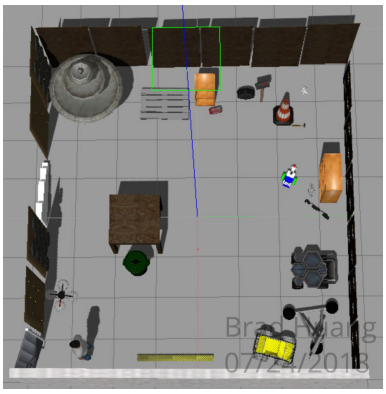

Figure 1. The robot mapping in the new environment.

4.2 Robot Configuration

The robot was inherited from the robot in the previous project. I added a transform from the camera to the rgbd camera frame, which is the link attached to the RGB-D camera node in the .gazebo file.

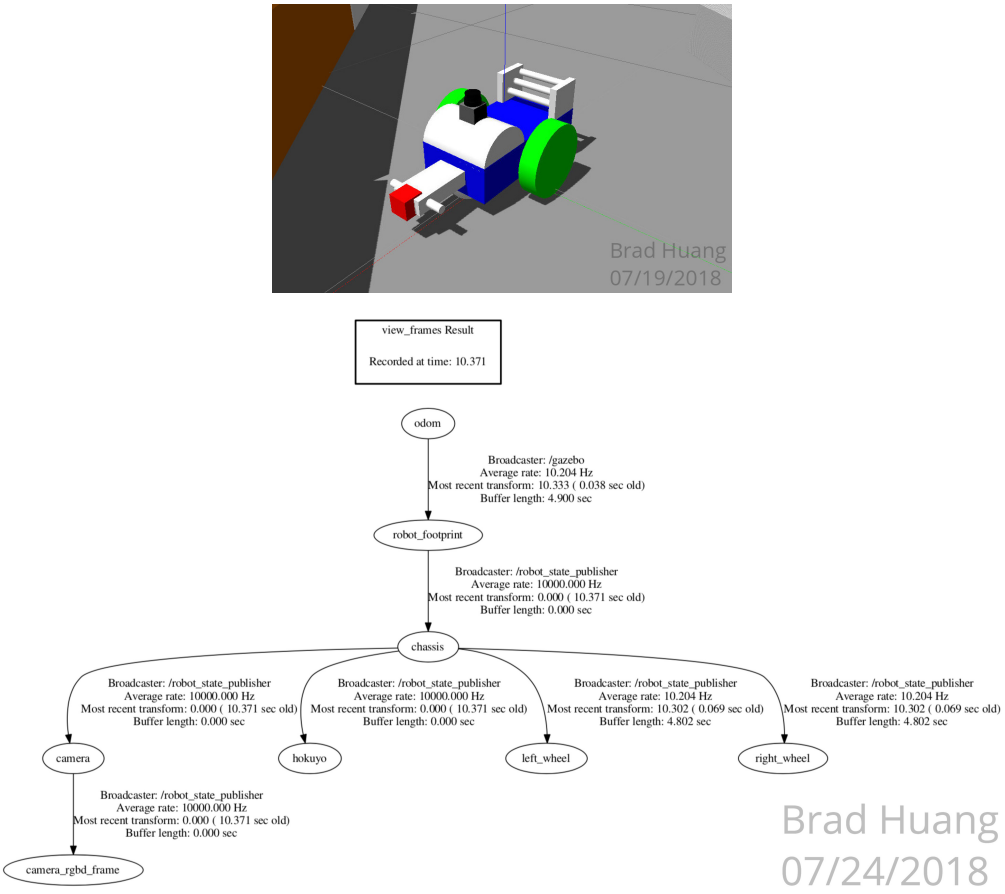

Figure 2. The robot mapping in the new environment and its transform tree.

5 Results

5.1 RTAB Map Loop Detection

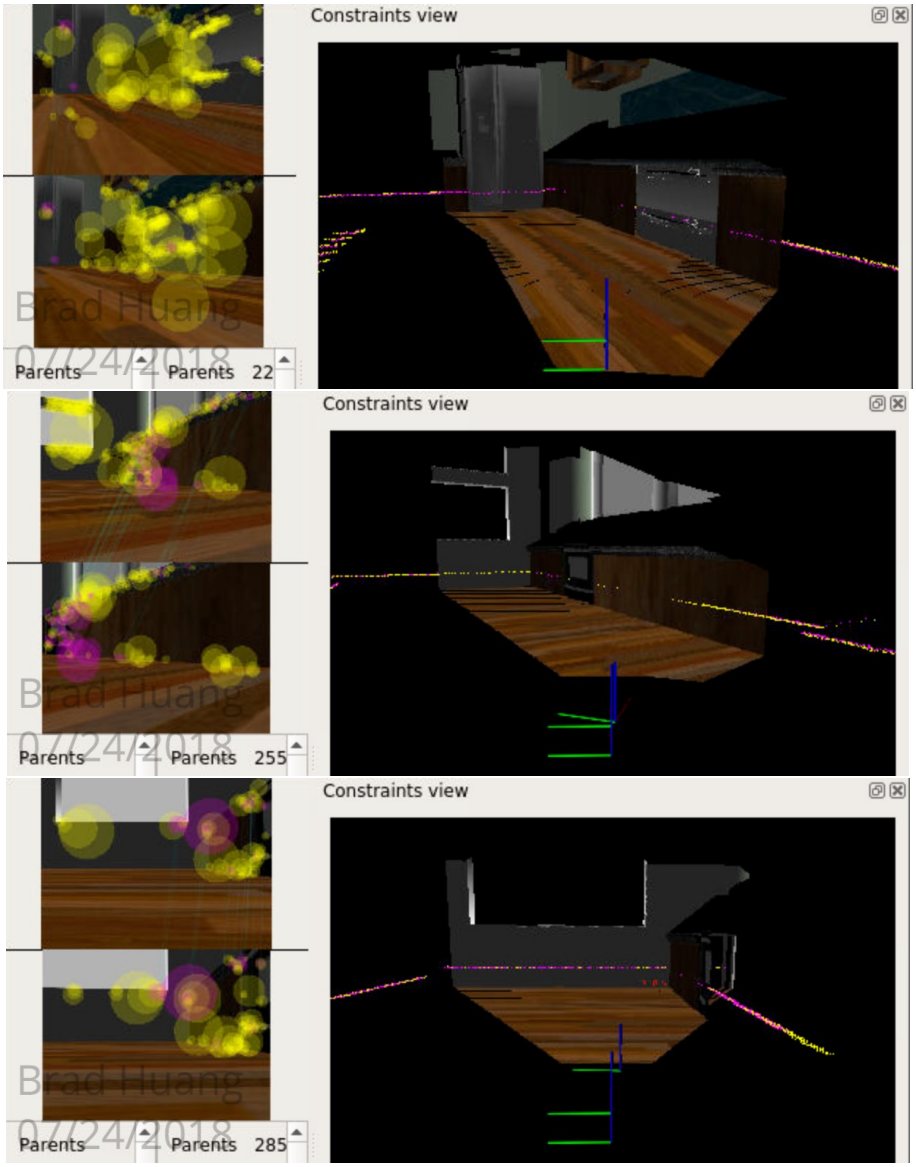

Figure 3. Example loop closures from the RTAB map database of the kitchen environment.

By inspecting the database created from the experiment run in the kitchen environment, I observed many detected loop closures. Since the robot revisited the aisle between the counter top and the sink several times, the loop closures correctly identify when the poses around there are similar.

5.2 Mapping Accuracy

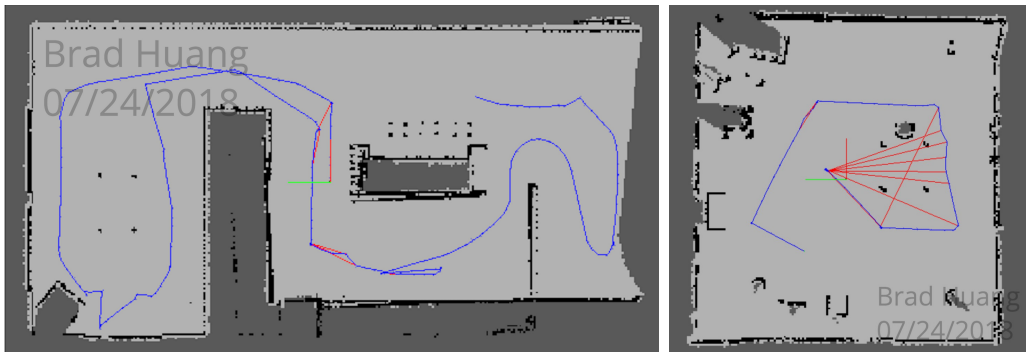

Figure 4. The 2D maps of the kitchen (left) and the garage (right) environments.

The 2D maps of both environments elicits the movable areas for the robot in the environment well. However, the 2D maps are produced from the scans of the laser rangefinder, which is mounted higher on the robot. As a result, it will only consider obstacles at the height of the laser rangefinder and omits some of the small objects in the garage environment.

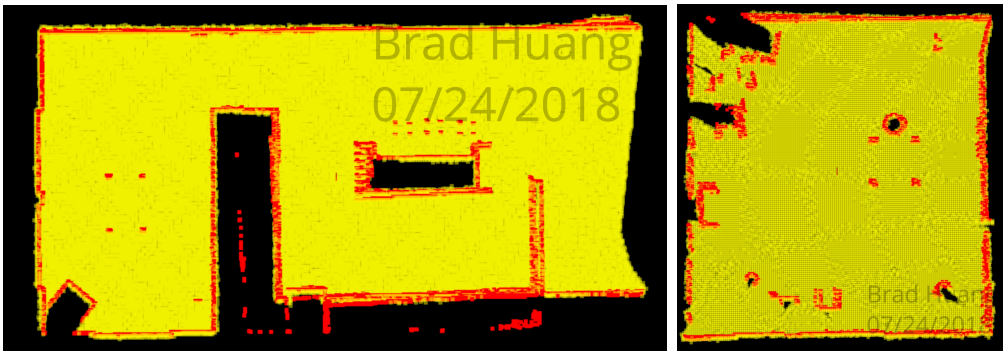

Figure 5. The occupancy grids of the kitchen (left) and garage (right) environments.

The occupancy grid maps of the environments are similar to that of the 2D maps. They are based on the laser scans and have the same drawbacks like the 2D maps - that it only shows obstacles at a certain height.

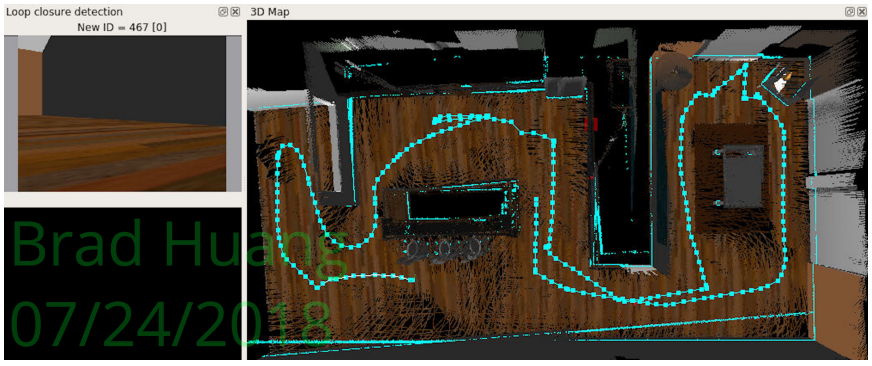

Figure 6a. The 3D map of the kitchen environment.

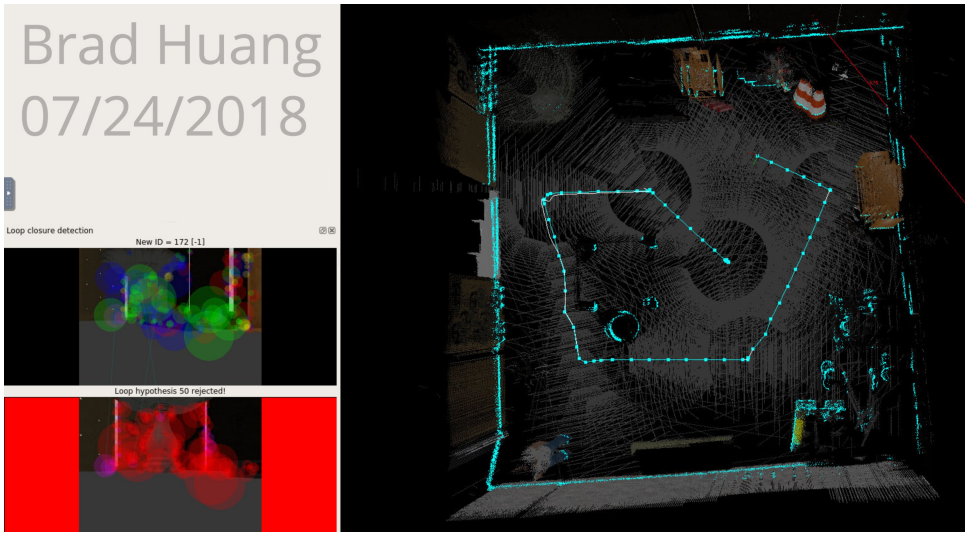

Figure 6b. The 3D map of the garage environment.

The real time mapping visualization tool of the RTAB Map can show 3D maps of the environments as the robot explores the environment. The 3D maps are visualized as point clouds from the RGB-D camera, while the tool also shows the estimated trajectory and the loop detection results at the same time. Comparing Figure 6a and Figure 6b, the 3D point clouds are not dense in the maps unless the robot traverses through the environment repeatedly. However, most of the objects in the environment can be seen in the 3D maps.

6 Discussions

Mapping the provided kitchen environment was an easier task than mapping the garage environment. The robot was able to perform the mapping task accurately enough on its first try around the room, while it took me several rounds of trial and error to make the mapping result usable in the garage environment. The kitchen environment is conducive to the success of the RTAB map because it contains feature-rich objects such as chairs and sofa. In addition, even though the wood tiles are repetitive and the walls are feature-less, the surrounding objects are dense enough for the robot to consistently map the environment even when looking towards corners of the rooms.

The robot failed a few times in the garage environment since the RTAB map algorithm produced false positives of loop closures. For example, when the camera is facing the stone wall, it detects a lot of small features on the wall. However, when the camera is facing another section of the stone wall, the algorithm would identify them as the same section, and the 3D map would be repeated while rotated to product unusable maps such as the one demonstrated in the lesson. I was able to overcome some of the problems by placing different items in front of every wall and making every wall different, but the false positives still come up from time to time.

7 Future Work

RTAB map uses the Bag of Words approach to identify reoccurring objects. However, the method needs tuning and sometimes produce false positives of loop closures. For example, in the garage scene, the robot often identifies different sections of the rock wall as the same section, or the two cylindrical shaped objects as the same one. When such false positives occur, the created map would be heavily distorted and unable to be used. Object detection and classification algorithms with better precision, such as deep neural network, might be able to reduce false positives and enable the robot to map similarly well in less feature-rich environments.

In addition, the RTAB map does not perform well when there are dynamic objects in the environment. Since the RTAB map assumes the objects to be static in the environment, when it detects an object appearing at different locations, it will be unable to produce the correct constraints and result in a low quality map. A more advanced detection algorithm would be required to distinguish a moving object from a dynamic object.

The 3D mapping is intriguing for me since I am interested in developing underwater autonomous vehicles in the future. Exploring underwater environment is difficult and often relies on robots to perform tasks in dangerous areas for a long time. Being able to perform SLAM efficiently and accurately with RTAB map is useful in such scenarios provided that the two problems mentioned above can be solved.