Map My World Robot

Yisi Zhang

Abstract—In this project, the robot navigates in areas where it has no maps for a priori. Using Real-Time Appearance-Based Mapping (RTAB-Map), a Graph-based SLAM algorithm, 2D occupancy grids and 3D octomaps from two simulated environments are created. The robot uses RGB-D camera (Kinect) to interface with the environment. The map is updated based on the visual input and loop closure is detected in real time to optimize the graph.

Index Terms—SLAM, RTAB-Map, Gazebo, ROS.

1 Introduction

When a robot navigates in a new environment, it has to figure out two unknowns: one is the map of the novel environment and the other is the location of itself. Simultaneous localization and mapping or SLAM is the name of the computational solutions for such problems.

The GraphSLAM is one of the SLAM algorithms that solves the full SLAM problem. In this project, we apply a RGB-D Graph-Based SLAM approach named RTAB-MAP (Real-Time Appearance-Based Mapping) to map out simulated environments.

2 Background

Multiple algorithms have been developed to solve the SLAM problems thus far. For example, the Occupancy Grid Mapping algorithm creates grid cells for the environment and updates the occupancy of the cells by binary codes based on measurements. Differently, the FastSLAM (1.0 and 2.0) algorithms adopt particle filter to estimate the trajectory and low-dimensional Extended Kalman Filter to estimate the map. The FastSLAM method has also been adapted to a Grid-based approach in which known landmark positions are not required and thus can be used for mapping arbitrary environment. The new popular algorithm is the Graph-based SLAM. It borrows the concept in graph theory by constructing a graph of constraints that takes all measurements and movements into account. The constraint matrix is usually sparse and thus is granted with computational advantages.

In a lot of occasions, 2D mapping gives adequate information for the robot to navigate in the environment and it is relatively computationally cheap. However, robots live in the 3D world, and by using a 2D representation for the world, a great amount of information is dropped. 3D mapping provides more accurate information of the environment and is more reliable for collision avoidance. It is especially useful for flying robots and for robots navigating in complex environment. A number of sensors can carry out 3D range measurement of the environment, including 3D lidar, RGB-D camera and stereo camera. Constructing a 3D model of the environment based on the sensory input is usually memory consuming, which is the major bottleneck in 3D mapping.

One efficient implementation of the 3D occupancy grid mapping approach is called OctoMap. It is based on octrees to represent the 3D environment. An octree is a tree data structure in which a node has eight children. It partitions a voxel into eight subdivisions (octants). The OctoMap converts point clouds into 3D occupancy maps. The occupancy is probabilistic. The trees are pruned using a threshold probability and three categories, free, occupied and unmapped areas, are represented.

3 Scene and Robot configuration

The project mapped two simulated environments: one is a provided kitchen-dining area (Fig. 1) and the other one is a created environment. The latter consists of a closed area surrounded by walls with windows and a door and furnitures scattered around Fig. 2. Distinct objects were included in the scene to reduce the chance for false detection of loop closure.

Fig. 1. Kitchen-dining world

We designed a robot using Gazebo. The robot has a box-shaped base and two wheels. To stabilize the robot, there are also two caster wheels in the front and in the back. On top of the base, there is a RGB-D camera facing the front and a laser sensor. The RGB-D camera has a kinect appearance by assigning a mesh file to it. The camera link (i.e. the kinect) has two children: a RGB frame and a depth frame (Fig. 3). The RGB frame was rotated from facing the

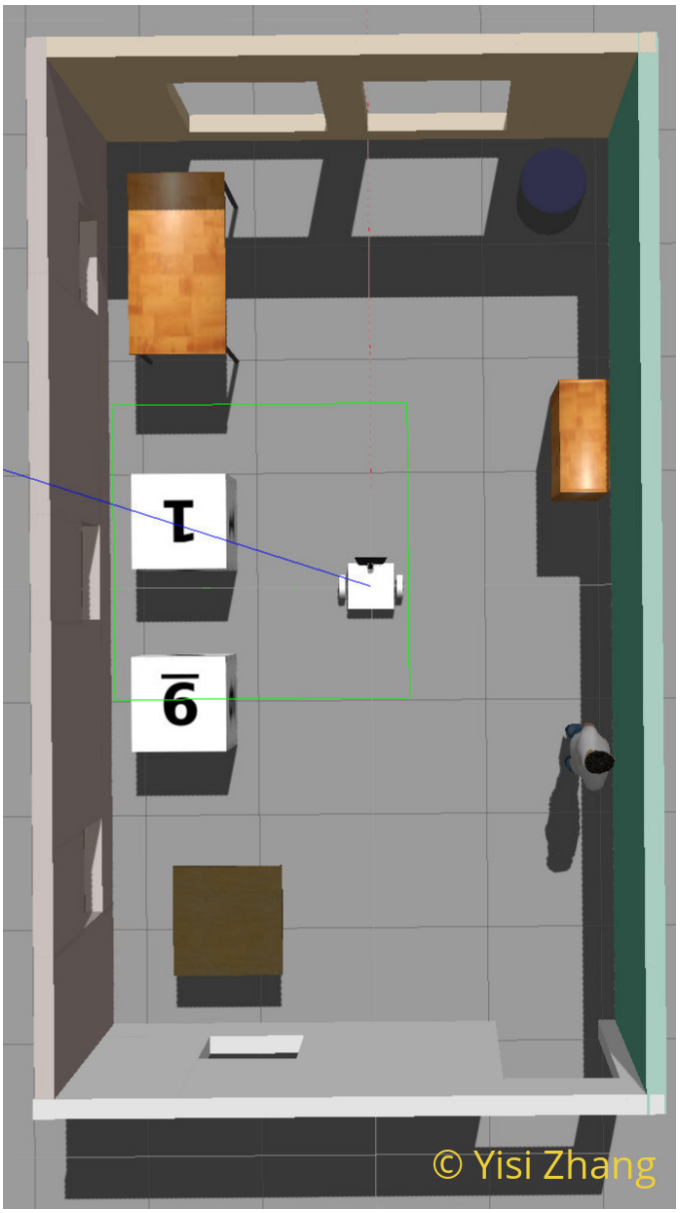

Fig. 2. Customized world

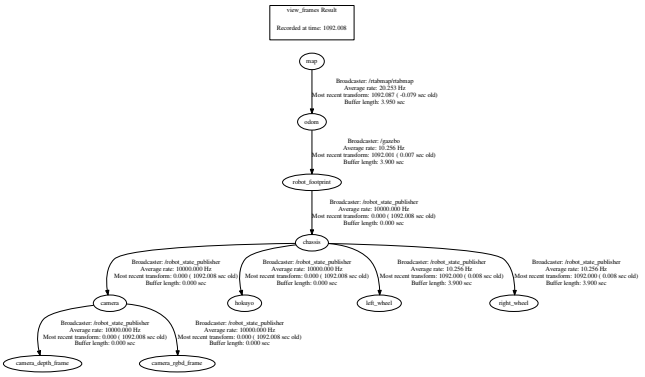

Fig. 3. Frame structure

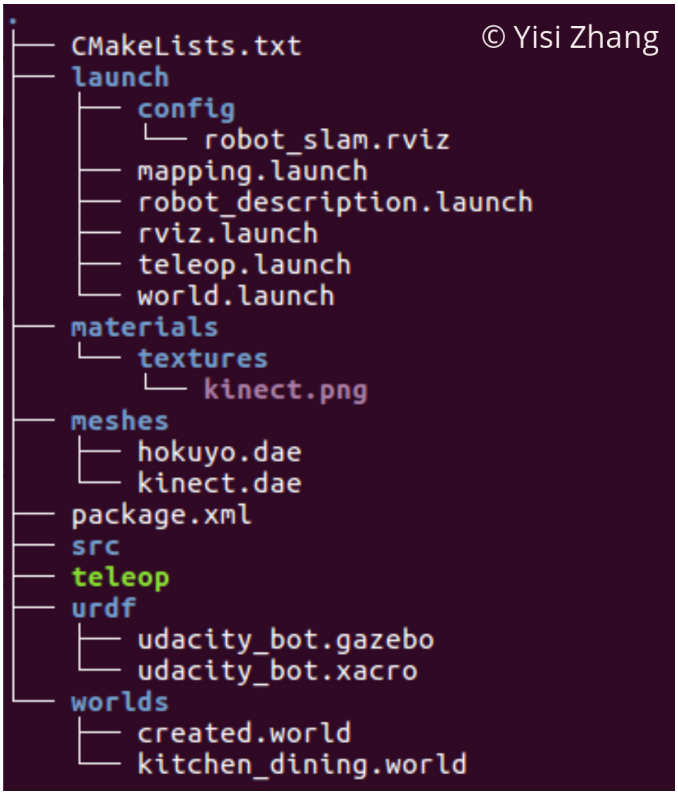

Fig. 4. Package structure

top to facing the front. The structure of the frames is shown in Figure 3. In the Gazebo file, the camera is controlled by the "libgazebo_ros_openni_kinect.so" package. The camera name is the name of the camera link and the frame name is the camera_rgbd_frame. The topic names match the mapping.launch file.

The "slam_project" package is primarily comprised of the launch files, the urdf and worlds (Fig. 4). The urdf folder contains the robot configurations (.xacro file) and the .gazebo file for the robot controllers. The worlds folder contains the .world files for the two simulated environments. The launch folder includes the .launch files. The robot and world are launched by the "world.launch" file. The "teleop.launch" file calls the teleop file in the parent folder. In the actual operation, rosrun was used instead to run the teleop file to control the robot motion. Rviz is launched by the rviz.launch file. The configuration of the rviz parameters is stored in "robot_slam.rviz".

4 Results

The mapping results of the kitchen-dining world is shown in Figure 5. The mapped world is similar to the gazebo world, with 163 occasions of loop closure detected.

The customized world was also mapped correctly, with 18 loop closure detected (Fig. 6).

5 Discussion

The robot was controlled via teleop to navigate around in the simulated world. The robot sometimes hit the obstacles as the linear and angular speed of the teleop was not optimized. As it bumped into the obstacles, the mapping quality was also affected as the angle of the RGBD images was not correct. Slower speed also improved the mapping

performance, possibly due to reduction in failures of motion update at fast speed.

The mapping result of the kitchen-dining area works relatively well. Some shifts appear in the map, but overall it resembles the original gazebo world. The kitchen area was mapped better than the dining room. This might be due to more feature variation in the kitchen than the dining room. The worst mapped location is the long corridor that has a large range of similar pattern, and this may have caused the shift of the location of the partition wall and the wall in the end.

The customized world was smaller than the provided world and thus the mapping performed slightly better. To increase feature variability of the walls, they are assigned with different colors.

6 Conclusion / Future work

In general, 3D input provides more accurate information of the environment and is more reliable for collision avoidance and loop closure detection in SLAM. Alternative 3D sensors such as the 3D lidars are usually more costly. Thus, the RGB-D camera in combination with the RTAB-Map tool is a more economical option for the SLAM task.

Because of the concrete visual information this method exploits, it can be applied to the 3D reconstruction of unknown areas where humans cannot easily get access to. The realistic scene this method reconstructed can also be applied to virtual reality environment design.

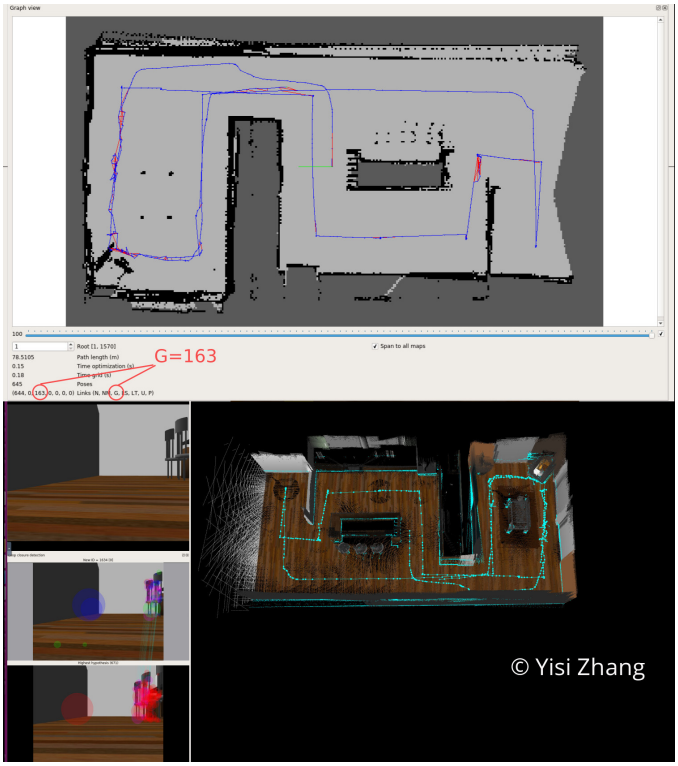

Fig. 5. kitchen-dining world - 2D and 3D maps

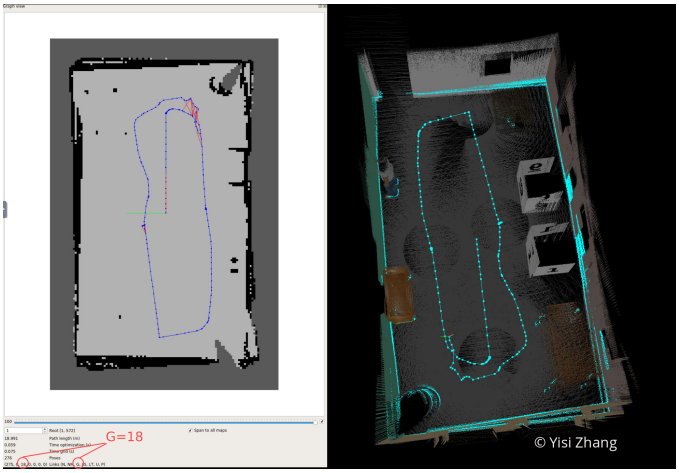

Fig. 6. Customized world - 2D and 3D maps