Robotic SLAM: slam_bot

Juil O (horagong@gmail.com)

Abstract—Simultaneous Localization and Mapping has been done using the robot and GRAPH SLAM algorithm. The robot has RGB-D camera and odometer and moves around the environments to collect data for generating 2D and 3D map.2D and 3D Two maps of the two different environments are generated using GRAPH SLAM called Real Time Appearance Based Mapping(RTAB-map).

Index Terms—robot, localization, particle filter

1 Introduction

In this experiment the robot is driven manually to map the environment into 2D and 3D maps. It is equipped with RGB-D camera and odometer. It uses GRAPH SLAM algorithm, especially RTAB-MAP, to make the maps from the sensor data. The following tasks should be done.

- Robot model
  - Create a mobile robot model with a RGBD camera and a laser sensor. The robot model is a mobile robot comprising of a base, at least two actuators, a RGBD camera, and a laser sensor.
  - Setup the Gazebo and Rviz environments and launch the robot inside the environment.
  - The provided transform tree has proper linkages.
- Mapping
  - Create the appropriate launch files such as: mapping.launch, teleop.launch, localization.launch
- Personal Gazebo World
  - Build a personal Gazebo world. The world could be navigated and mapped using RTAB-Map.
- Mapping Accuracy
  - When evaluating rtabmap.db, using rtabmap database viewer at least 3 loop closures are found and the occupancy grid is identifiable.
  - Overall 3D map should portray room characteristics.

2 Background

Being able to learn a map from scratch can greatly reduce the efforts involved in installing a mobile robot because some application domains do not provide the luxury of coming with an a priori map. Most buildings do not comply with the blueprints generated by their architects and even if blueprints were accurate, they would not contain furniture and other items that, from a robots perspective, determine the shape of the environment just as much as walls and doors. So mapping enables robots to adapt to changes without human supervision. [1]

3D mapping would give us the most reliable collision avoidance, and motion and path planning, especially for flying robots or mobile robots with manipulators. But 3D representations are even more costly than 2D.

Acquiring maps with mobile robots is a challenging problem. The hypothesis space of all possible maps is huge and it is a chicken-and-egg problem. Constructing a map when the robots poses are known is also relatively easy but in the absence of both an initial map and exact pose information, the robot has to do both: estimating the map and localizing itself relative to this map. So it's often referred to as the Simultaneous Localization and Mapping (SLAM) or Concurrent Mapping and Localization(CML) problem.

If the environment has cycles, it is particularly difficult to map. If a robot just goes up and down a corridor, it can correct odometry errors incrementally when coming back. Cycles make robots return via different paths, and when closing a cycle the accumulated odometric error can be huge.

Occupancy grid maps are often used after solving the SLAM problem by some other means, and taking the resulting path estimates for granted. It generates maps fit for path planning and navigation. The standard occupancy grid approach breaks down the problem of estimating the map into a collection of separate problems of estimating $p(m_i|z_{1:t}, x_{1:t})$ for all grid cell m_i, where m is the map, $z_{1:t}$ the set of all measurements up to time t, and $x_{1:t}$ is the path of the robot defined through the sequence of all poses. The controls $u_{1:t}$ play no role in occupancy grid maps, since the path is already known.

- $p(m|z_{1:t}, x_{1:t}) = \Pi_i p(\mathbf{m_i}|z_{1:t}, x_{1:t})$

This decomposition is convenient but it does not enable us to represent dependencies among neighboring cells; instead, the posterior over maps is approximated as the product of its marginals.

There are two main forms of the SLAM problem.

2.1 Online SLAM Problem

One is known as the online SLAM problem: It involves estimating the posterior over the momentary pose along with the map.

- $p(x_t, m|z_{1:t}, u_{1:t})$

Here x_t is the pose at time t. This problem is called the online SLAM problem since it only involves the estimation of variables that persist at time t. Many algorithms for the online SLAM problem are incremental: they discard past measurements and controls once they have been processed.

2.2 Full SLAM Problem

In full SLAM, a posterior over the entire path x1:t along with the map are calculated.

- $p(x_{1:t}, m|z_{1:t}, u_{1:t})$

The online SLAM problem is the result of integrating out past poses from the full SLAM problem. In online SLAM, these integrations are typically performed one-at-a-time.

The full SLAM problem with known correspondences possesses a conditional independence between any two disjoint sets of features in the map, given the robot pose. Put differently, if an oracle told us the true robot path, we could estimate the location of all features independently of each other. Dependencies in these estimates arise only through robot pose uncertainty. This structural observation will make it possible to apply a version of particle filters to SLAM known as Rao-Blackwellized particle filters. Rao-Blackwellized particle filters use particles to represent the posterior over some variables, along with Gaussians (or some other parametric PDF) to represent all other variables.

2.3 Fast SLAM

FastSLAM uses particle filters. Each particle contains a sampled robot path and a map. But here each feature in the map is represented by its own, local Gaussian and the individual map errors are conditionally independent. Hence the mapping problem can be factored into many separate problems, one for each feature in the map. FastSLAM estimates these map feature locations by EKFs, but using a separate low-dimensional EKF. The resulting representation requires space linear in the size of the map, and linear in the number of particles. FastSLAM solves both the full SLAM problem and the online SLAM problem.

2.4 Graph SLAM

GraphSLAM extracts from the data a set of soft constraints, represented by a sparse graph. It obtains the map and the robot path by resolving these constraints into a globally consistent estimate. The constraints are generally nonlinear, but in the process of resolving them they are linearized and transformed into an information matrix. Thus, GraphSLAM is essentially an information-theoretic technique.

RTAB-Map is a GraphSLAM algorithm. it uses loop closure to determine where the robot has seen a location before.Loop closure compares new images and locations to ones that are previously viewed rather than it registers them as new locations. As a robot travels to new areas in its environment, the map is expanded and the number of images that each new image compared to increase. This causes loop closure to take longer with the complexity increasing linearly. RTAB-Map is optimized for the large scale and long term SLAM by using multiple strategies to allow for the loop closure to be done in real time. The loop closure happens fast enough that the result can be obtained before the next images are acquired.

3 Scene and robot configuration

3.1 Robot Model

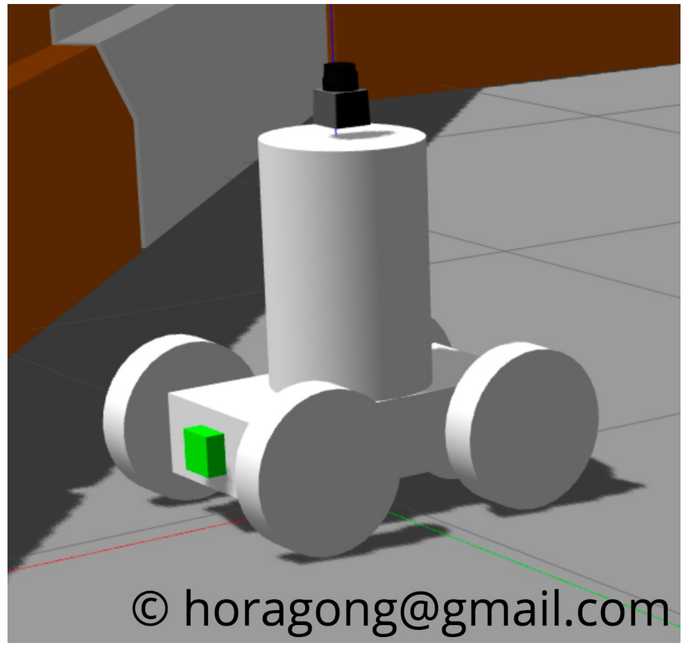

Fig. 1. map_bot

This mobile robot has a RGBD camera and a laser sensor and four actuators. RGBD camera is in front of the chassis and laser sensor is on top of the cylinder body. It has a camera link and a camera optical link for transformation of the depth image.

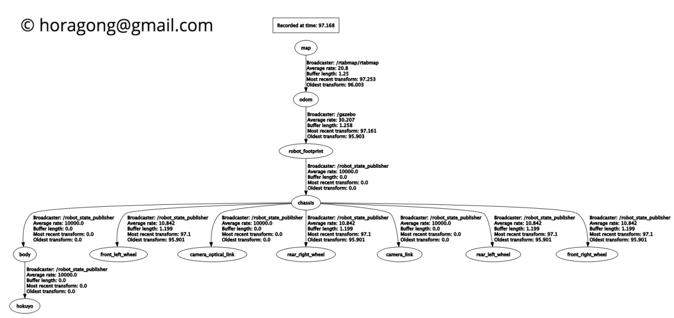

Fig. 2. linkages

3.2 Personal Gazebo World

A screenshot of robot mapping the personal Gazebo world is present. The world is created using Gazebo. The robot can navigate and map this Gazebo world using RTAB-Map. It has several tables and foundation in the cafe area.

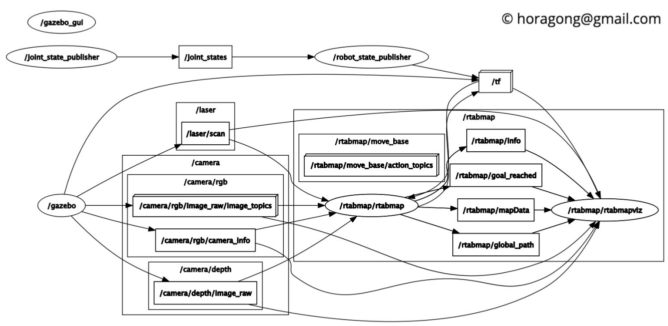

Fig. 3. transforms

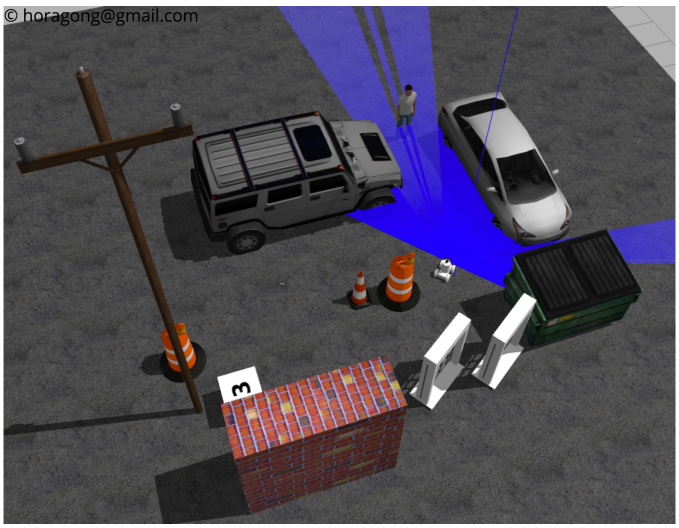

Fig. 4. map_bot

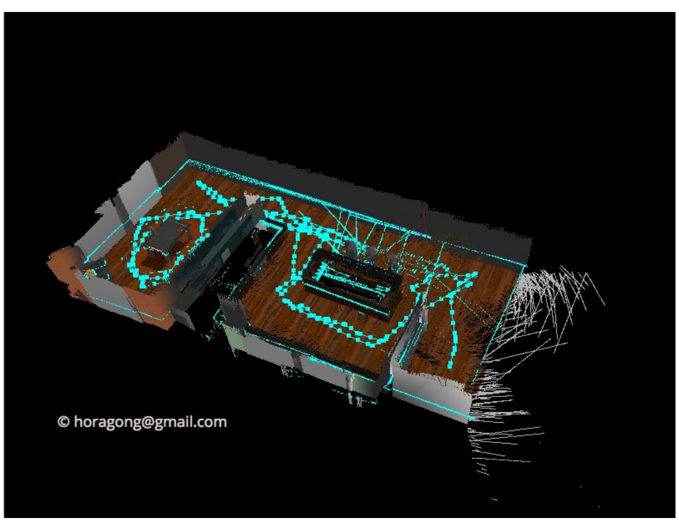

Fig. 5. kitchen 3D

4 Results

The robot is controlled by teleop during the mapping process. The runtime map image from the rtabmapviz is the following.

The 2D map image from rtabmap-databaseViewer has more than three loop closures. The result 3D map image

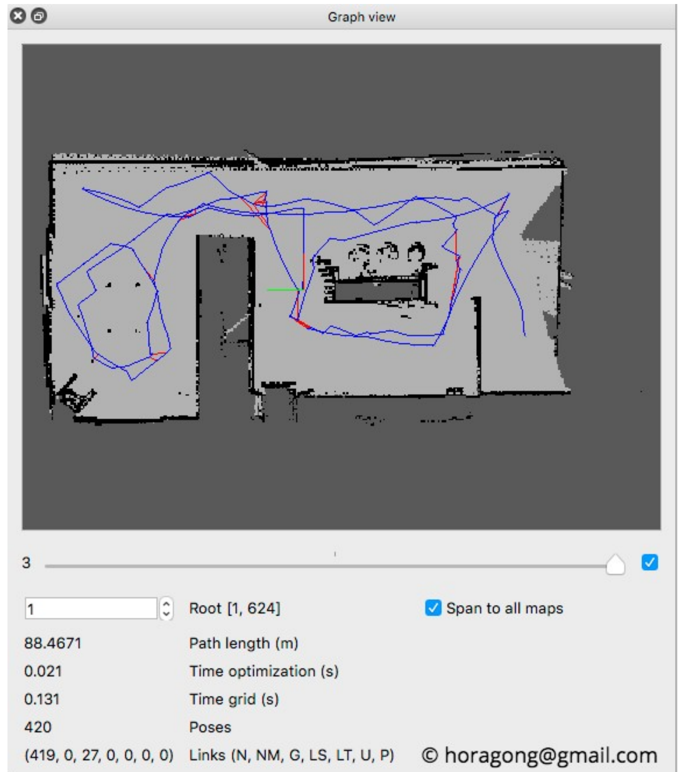

Fig. 6. kitchen 2D

from rviz shows the similar look to the provided environment.

Fig. 7. kitchen 3D

The 2D map image from rtabmap-databaseViewer has more than three loop closures. The result 3D map image

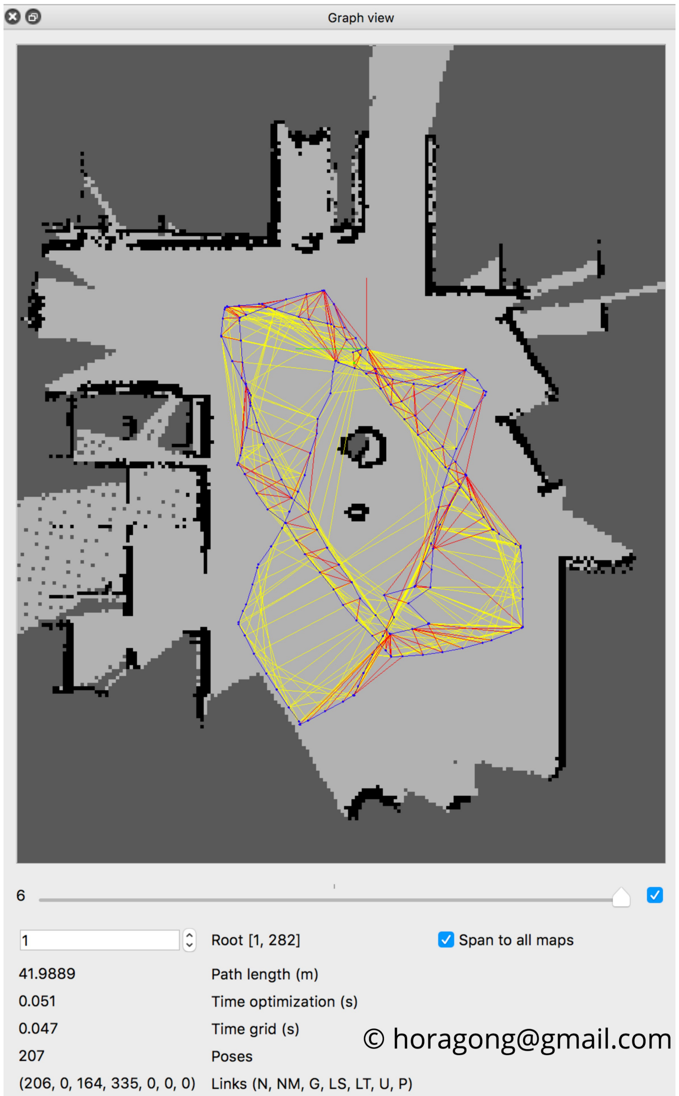

Fig. 8. my world 2D

from rviz shows the similar look to the provided environment.

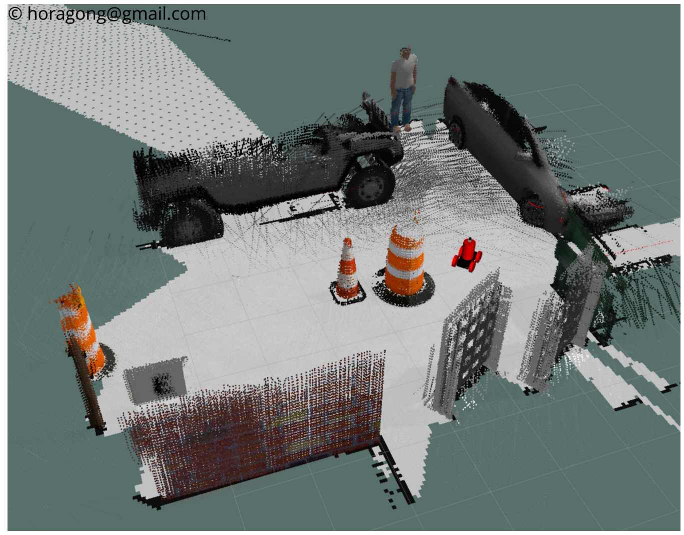

Fig. 9. my world 3D

5 Discussion

The provided kitchen environment was easy to map for the robot but the my world environment was not. At first my world has no object at the center. In that case the position of the robot sometimes changed suddenly abruptly during the movement and the map was messed up. It seems to be because the path without the identifiable object is long. I added two construction barrel at the center to lessen the width of the path. Around the barrels were made some loop closures without disrupting the map.

6 Future work

SLAM method can be used for indoor mapping. The real robot goes around shopping malls mapping the indoor shops. Those data will be used for navigation service.

References

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT press, 2006.