RTAB-Map SLAM Project

Pritesh Gudge

Abstract—The third project in term 2 of the Udacity Robotics Nano Degree program requires students to use ROS and Gazebo along with RTAB-Map, to create a 2D occupancy grid and a 3D octomap of two environments(worlds) - one supplied as a part of the project and the other custom generated. The project aims at performing an application of SLAM techniques in a simulated environment. The objective is to extend a previous robot creation to upgrade sensors to supply the necessary sensor messages for RTAB-Map. This leverages the laser scanner, IMU/Wheel Encoder but replaces the camera with a RGB-D camera (ie kinect). Further the ROS project is created with all links connected with appropriate naming and mapping. The robot is launched and teleoped around the environments to generate a map of the environment. After successfully mapping the supplied environment, a custom generated environment is created and mapped using the same technique.

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

1 Introduction

The robot model in this project uses a Simultaneous Localisation and Mapping (SLAM) technique called RTAB-Map (Real-Time Appearance-Based Mapping). It is a RGB-D Graph Based SLAM approach that uses incremental appearance based loop closure detection [1]. The RTAB-Map ROS wrapper [2] is leveraged with visual representation in real time via rtabmapviz. The resultant map is stored in local database that be later interrogated via rtabmap-databaseViewer [3].

2 Background

When a robot encounters a new environment where there is no supplied map, it needs to be able to create this map and localise its pose using it. This combined localisation and mapping process is referred to as SLAM (Simultaneous Localisation and Mapping).

The main mapping algorithms are Occupancy Grid Mapping, Grid-based FastSLAM, Graph-SLAM and RTAB-Map.

2.1 Occupancy Grid Mapping

The Occupancy Grid Mapping [4] is a 2D algorithm where each grid cell is identified as Unknown/Undiscovered Zone, Free Zone or Occupied. This represents a slice of the 3D world.

2.2 Grid-Based FastSLAM

The Grid-Based FastSLAM [5] approach combines SLAM (Synchronised Location and Mapping) using a MCL (Monte Carlo Localisation) Algorithm and an Occupancy Grid Mapping. The main advantage of is the MCL particle filter approach but it always assumes there are known landmark positions. Thus it is unable to model an arbitrary environment.

2.3 Graph SLAM

Graph-SLAM [6] uses a graph based approach to represent poses, features from the environment, motion constraints (between two poses) and measurement constraints (ties together a feature and a pose). It solves the full SLAM problem, it covers the entire path and map and not the most recent pose.

2.4 RTAB-Map

This project uses RTAB-Map [2], which is a Graph-SLAM approach that uses loop closure with Visual Bag-of-Words [7] for optimisation. The loop closure detection occurs against working memory to constrain the number of images interrogated. Working memory can be transferred and retrieved from long term memory to reduce complexity. The algorithm used for loop closure detection is SURF (Speeded Up Robust Features) [8].

The possible outputs of RTAB-Map are 2D occupancy grid map, 3D octomap or a 3D point cloud.

Robots are of varying dimensions inclusive of height. Whilst mapping a 2d environment may show where fixed walls etc are it does not take into account height. A robot, that is propelled on the floor, may be able to navigate under some obstacles but not others eg a chair vs a large table. Hence the need to understand the environment from a 3D perspective.

However building a 3D map is more costly then a 2D map. This is not only in terms of Compute & Data costs but also in the cost of the sensors required. However, simple sensors such as a single camera may be cheaper but the algorithms required can be more complex.

3 Simulations

3.1 Robot Model Configuration

The robot model used was based on the udacity_bot created in the previous project as the robot model (which had a rectangular base with differential drive controller for the left and right wheels). The camera was removed and replaced with a kinect leveraging the *openni_camera* ros package with the gazebo controller *Openni Kinect*. No changes were made to the hokuyo laser range finder. An additional joint was added to rotate the kinect data 180 degrees. It was positioned on the front of the robot so as to not interfere with the laser range finder. The bot configuration files can be found under the urdf directory. Visualization of the frames follows is show in the figure 1.

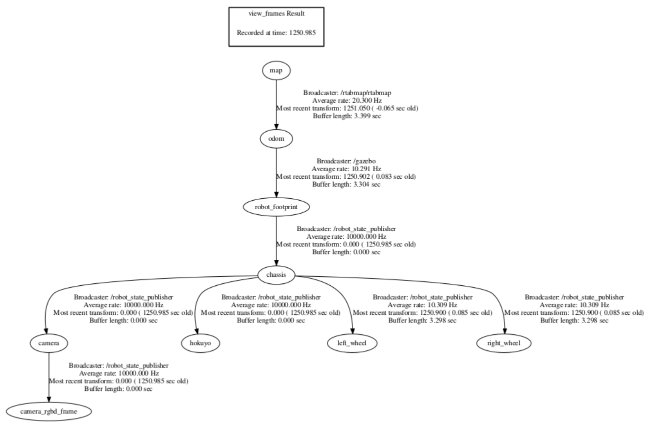

view_frames Result
Recorded at time: 1250.985
map
Broadcaster: /rtabmap/rtabmap
Average rate: 20.300 Hz
Most recent transform: 1251.050 (-0.065 sec old)
Buffer length: 3.399 sec
odom
Broadcaster: /gazebo
Average rate: 10.291 Hz
Most recent transform: 1250.902 (0.083 sec old)
Buffer length: 3.304 sec
robot_footprint
Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (1250.985 sec old)
Buffer length: 0.000 sec
chassis
Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (1250.985 sec old)
Buffer length: 0.000 sec
Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (1250.985 sec old)
Buffer length: 0.000 sec
Broadcaster: /robot_state_publisher
Average rate: 10.309 Hz
Most recent transform: 1250.900 (0.085 sec old)
Buffer length: 3.298 sec
Broadcaster: /robot_state_publisher
Average rate: 10.309 Hz
Most recent transform: 1250.900 (0.085 sec old)
Buffer length: 3.298 sec
camera
hokuyo
left_wheel
right_wheel
Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (1250.985 sec old)
Buffer length: 0.000 sec
camera_rgbd_frame

Fig. 1. TF Frame

3.2 Worlds

Two worlds were created in gazebo - one supplied as kitchen_dining.world (Fig. 2)and the other customised cafekitchen.world (Fig. 3)

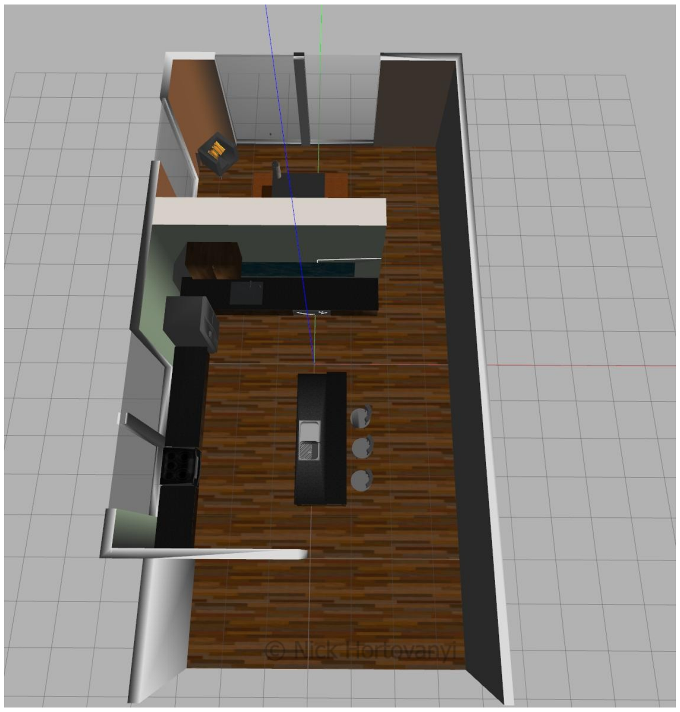


Fig. 2. Kitchen World

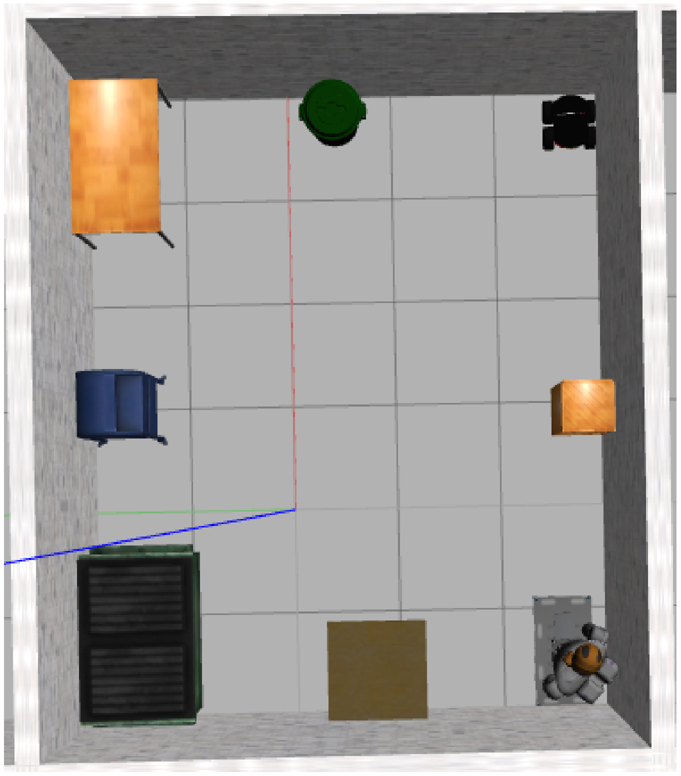


Fig. 3. Custom World

3.2.1 Parameters

The Kitchen World model was mapped with RtabMap features ″Kp/MaxFeatures″ 400 and ″Vis/MinInliers″ at 15, and ″Reg/Strategy″ 0

The Custom World model was mapped with RtabMap features ″Kp/MaxFeatures″ 400 and ″Vis/MinInliers″ at 20, and ″Reg/Strategy″ 1

4 Results

4.1 Kitchen World

The kitchen world 2D and 3D maps are shown in figure4 and figure 5 respectively. In the 2D Map and the 3D map the general layout of the kitchen and the furniture structures are visible. A total of 198 loop closures were detected(Fig. 6).

4.2 Custom World

The kitchen world 2D and 3D maps are shown in figure7 and figure 8 respectively. In the 2D Map and the 3D map the general layout of the floor and the furniture structures are visible. A total of 237 loop closures were found(Fig. 9).

4.3 Technical Comparison

The kitchen_dining model performed significantly better then the student created cafe model. This was due to the richer and more complex features of the kitchen_dining model.

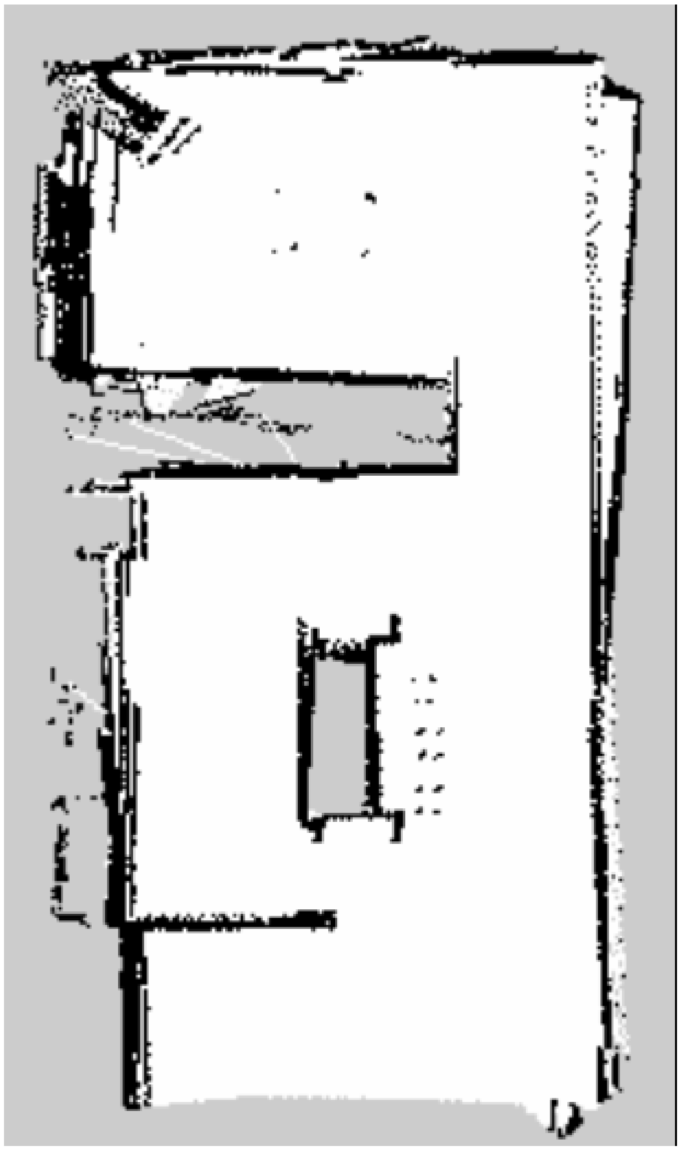


Fig. 4. Kitchen World 2D map

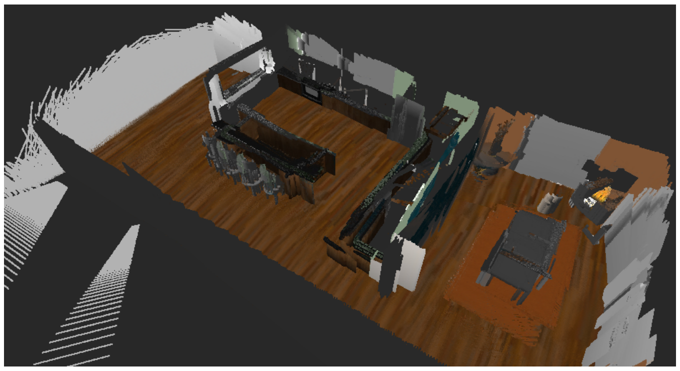
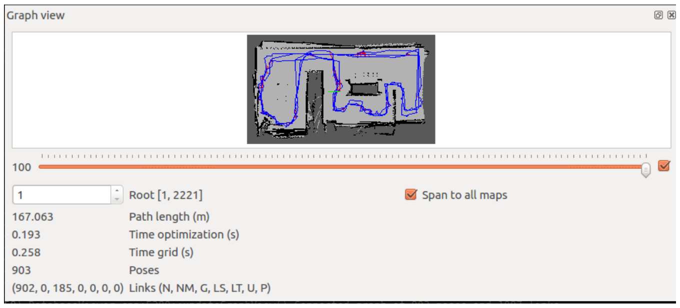


Fig. 5. Kitchen World 3D Map



```
100
1                  Root [1, 2221]                Span to all maps
167.063            Path length (m)
0.193              Time optimization (s)
0.258              Time grid (s)
903                Poses
(902, 0, 185, 0, 0, 0, 0) Links (N, NM, G, LS, LT, U, P)
```

Fig. 6. Kitchen World Loop Closures

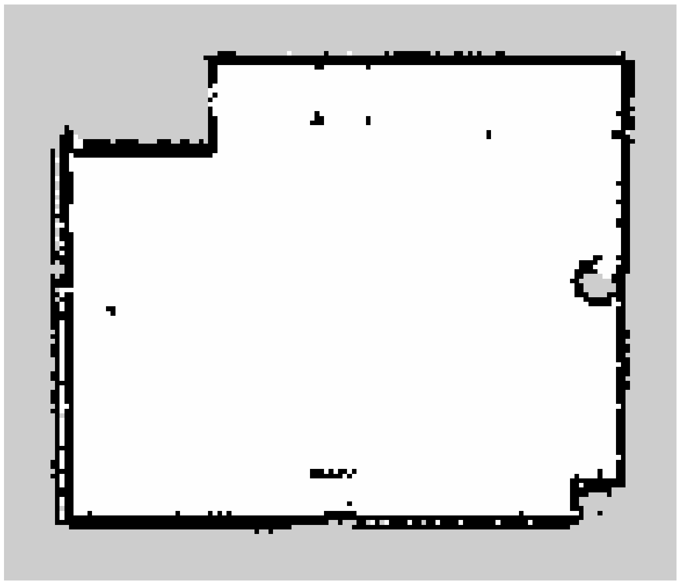


Fig. 7. Custom World 2D map

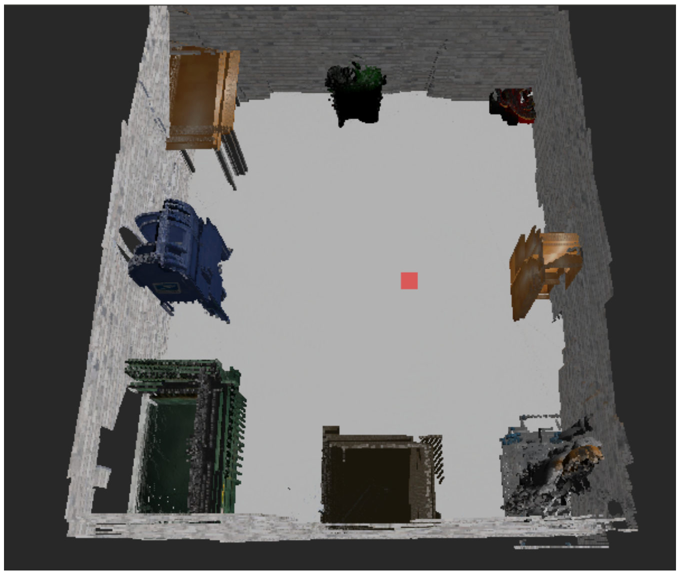
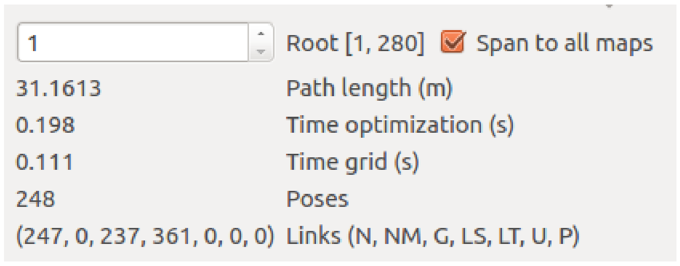


Fig. 8. Custom World 3D Map



1	Root [1, 280] ✓ Span to all maps
31.1613	Path length (m)
0.198	Time optimization (s)
0.111	Time grid (s)
248	Poses
(247, 0, 237, 361, 0, 0, 0)	Links (N, NM, G, LS, LT, U, P)

Fig. 9. Custom World Stats

5 Discussion

The robot was teleoped (navigated via the keyboard) around the room. At some points the robot did not move forward. This appeared to be when it started to perform loop closure. The parameters were not changed from the default provided as satisfactory result was obtained with the same. However the 3D map quickly started to resemble the physical kitchen dining gazebo model. To improve loop detection rates some, on the spot circles were performed. Of particular note were the features in the main kitchen area. More SURF features were identified there as there was more variation in the surfaces.

For the custom layout Vis/MinInliers was increased from 15 to 20 and Reg/Strategy was changed to 1(Including ICP). As the size of the environment was small and to prevent repeated and incorrect loop closures the above values of parameters were chosen.

The custom layout gazebo model wall surfaces were tiled, repeatable pattern with lack of other discerning features sometimes caused the loop closure detection to map to an incorrect previous image. This then distorted the map. Additional features were added to achieve a successful map.

6 Conclusion / Future work

Mapping is important to help understand the world. There are a plethora of sensors and of interest is the about to arrive solid state lidars. As the price point of these sensors continues to drop it will open up opportunities to create richer and more realistic 3D maps at a cheaper price point. Being able to map an environment cost effectively to create a replicated virtual world will increasingly be important to allow for the training of deep learning models. We are actively looking to do this and then supplant the trained model back into a robot so it can navigate in the original environment that was mapped.

6.1 Modifications for Improvement

Examples:

- Adding more sensors to collect data e.g. 3D Lidar
- Modifying the sensor location and layout. e.g. Adding RGBD camera's to the sides of the robot, to collect data about the lateral surroundings.

6.2 Hardware Deployment

1) To be deployed on hardware, a computation unit and appropriate sensors will have to be installed on the robot chassis and the wheels. Proper calibration and verification needs to be done to limit the errors.
2) High CPU and RAM hardware is required to perform SLAM data collection from the multiple sensors, especially to detect loop closures online.

References

[1] M. Labbe, "Loop closure detection." https://github.com/introlab/rtabmap/wiki/Loop-closure-detection.

[2] M. Labbe, "Rtab-map's ros-pkg." http://wiki.ros.org/rtabmap_ros.

[3] M. Labbe, "Database viewer." https://github.com/introlab/rtabmap/wiki/Tools#database-viewer.

[4] Multiple, "Occupancy grid mapping." https://en.wikipedia.org/wiki/Occupancy_grid_mapping.

[5] C. Stachniss, "Grid slam." http://ais.informatik.uni-freiburg.de/teaching/ws12/mapping/pdf/slam13-gridfastslam.pdf.

[6] Multiple, "Graph slam." https://en.wikipedia.org/wiki/GraphSLAM.

[7] Multiple, "Graph slam." https://en.wikipedia.org/wiki/Bag-of-words_model_in_computer_vision.

[8] A. M. . A. K., "Introduction to surf." http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html.