Map My World

Danny Fisher

Abstract—In this paper a GraphSLAM method (RTAB-Map) was utilised in a ROS package to produce 2D and 3D maps of an unknown environment. The robot was configured with a laser range scanner and an RGB-D camera and driven manually in an indoor Gazebo world in order to map the environment(2D Graph maps, Occupancy Grids and 3D Octomaps). The results showed that feature rich indoor environments were proportional to the number of loop closures and therefore crucial to generating accurate maps.

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

1 Introduction

Robot mapping is concerned with developing techniques that enable a mobile robot to construct and maintain a model of its environment based on spatial information gathered over time. [1] By mapping the robot's environment and its position, a mobile robot is better able to reach a determined target point and navigate around each obstacle.

The SLAM family of algorithms provide different solutions to the simultaneous requirement of localisation and mapping within unknown environments.

For the consideration of this report, the RTAB-Map variation is implemented. It is used within a ROS package to generate 2D (Graph & Occupancy Grid) and 3D (Octomap) maps of the environment. Its limitations and opportunities are noted and discussed.

2 Background

Localisation and Mapping are tightly coupled. For localisation to be performed independently, the map must be provided (known) and for mapping to be performed independently the robot must be localised. Therefore simultaneously localising and mapping (SLAM) methods are required when a robot is in an unknown environment. The only information the robot has access to are measurements and control inputs. If each of these data points are processed every iteration, the growth in required memory for embedded robotic systems become unmanageable. Thus it is necessary to manage the memory in order to perform SLAM.

FastSLAM is a landmark based algorithm that extends the particle filter algorithm to estimate both landmarks and robot poses. The Grid-based FastSLAM algorithm combines the particle filter with an occupancy grid to simplify the map as grid of cells either containing or not containing the robot or landmark, thus reducing complexity. However the Grid-based FastSLAM algorithm runs into issues by assuming known landmarks.

The GraphSLAM method is able to produce a map of the environment by combining measurements and representing them as nodes that are 'linked to a the position of a robot at a given time'. Fig. 1 below shows how the robots position (triangles) are constrained by the measurements (stars) in order to localise the robot in the unknown environment thus solving the full SLAM problem.

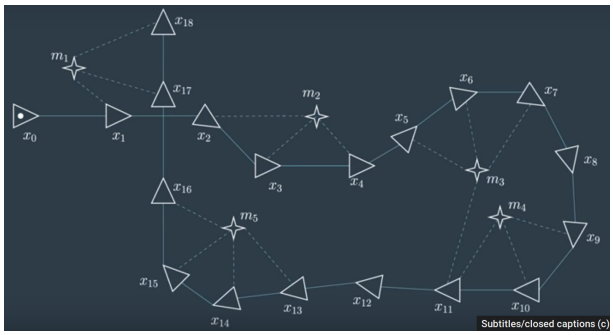

Fig. 1. GraphSLAM Mapping Constraints

The GraphSLAM method however is not able to determine whether it has been to the same location twice (correspondence). The RTAB-Map method uses current images of the environment to establish correspondence between observations instead of relying on the known landmarks. The process which performs correspondence- specifically known as a loop closure- is outlined below:

- It extracts features from each image; via the Sped Up Robust Features (SURF) algorithm,
- features are saved as unique words to save memory,
- each object/landmark in the image therefore has a bag of words associated with it,
- for every new image the robot receives a bag of words is generated and compared with the bag of words from other images,
- if the bag of words are similar i.e. when the same landmark is seen again, the algorithm is able to say that the robot has been in this location before (known as a loop closure).

3 Robot Configuration

This robot was configured in a previous project. The new addition is the RGB camera into an RGB-D camera (Kinect) (Fig. 2) The visualisation of the transform frames of the robot are also included in Fig. 3.

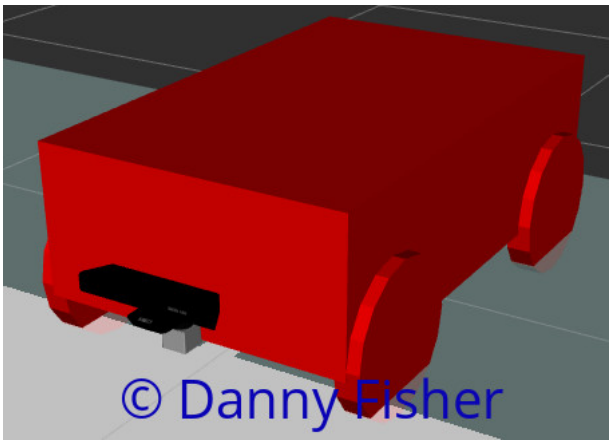

Fig. 2. Robot Model

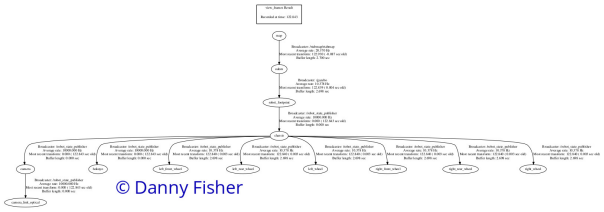

Fig. 3. tf Frames

4 Scene Configuration

For this project, two worlds were considered. The provided kitchen world (Fig. 4) and the student created world (Fig. 5).

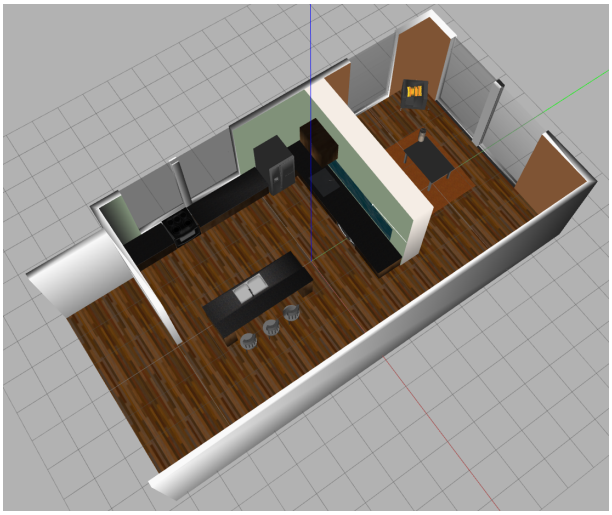

Fig. 4. Kitchen World

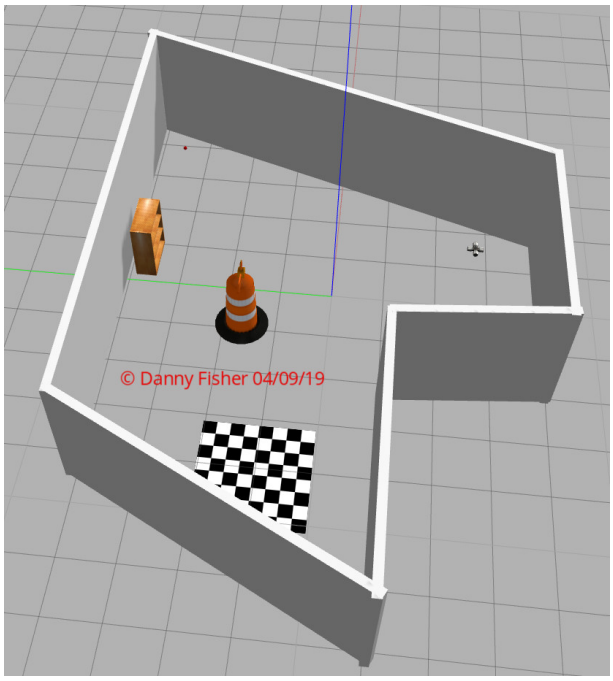

Fig. 5. Student Created World

5 Results

Multiple mapping runs were conducted on both kitchen and student created world. When the mapping algorithm was applied to the kitchen world, 12 loop closures were found, in just two complete mapping runs (Fig. 6). Images of the 2d graph, occupancy grid and 3D Octomap for the kitchen world are in Figures 7, 8 and 9 below.

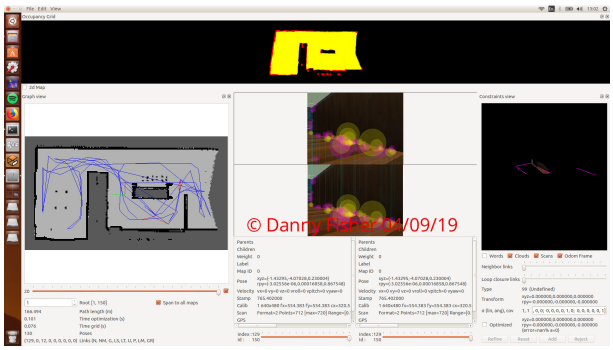

Fig. 6. Kitchen- RTAB-Visualiser

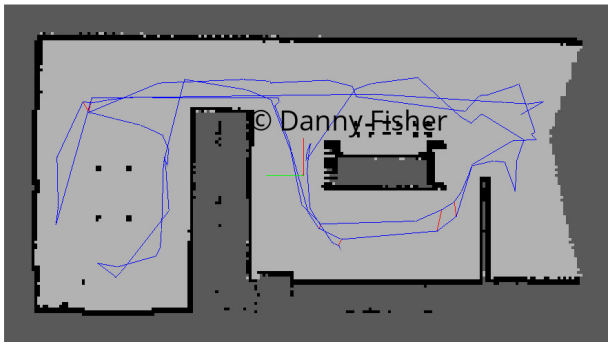

Fig. 7. Kitchen- 2D Graph

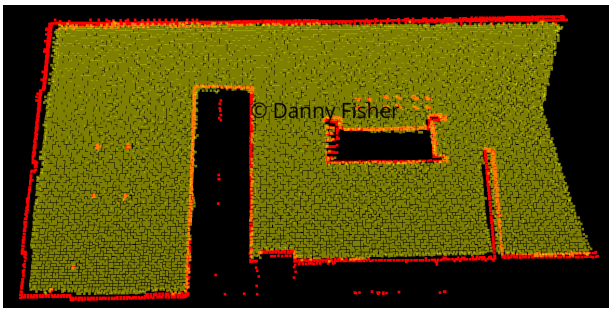

Fig. 8. Kitchen- Occupancy Grid

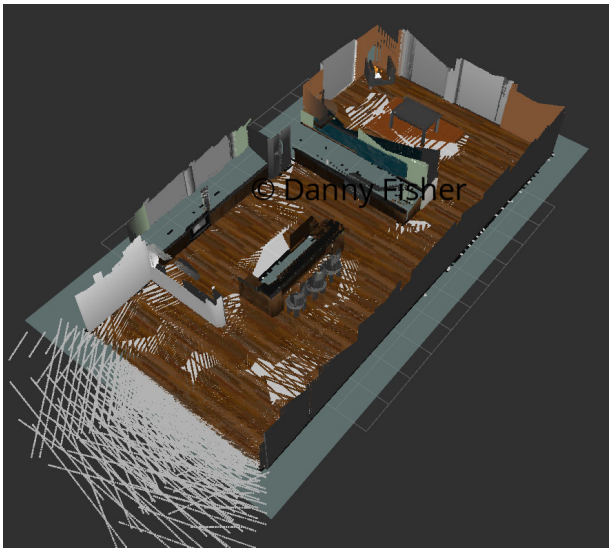

Fig. 9. Kitchen- 3D Octomap

The same mapping algorithm was applied to the student created world. After 3 mapping runs the algorithm was able to have loop closure 7 times. The results can be seen in Figures 10, 12, 11 and 13.

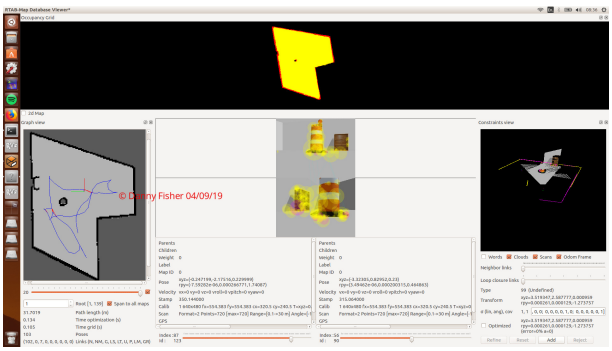

Fig. 10. Student Created World- RTAB-Visualiser

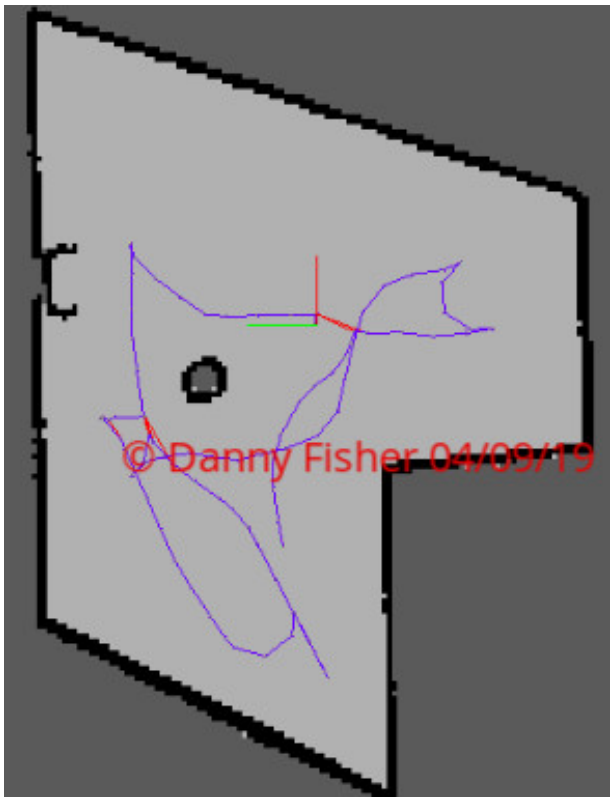

Fig. 11. Student Created World- 2D Graph

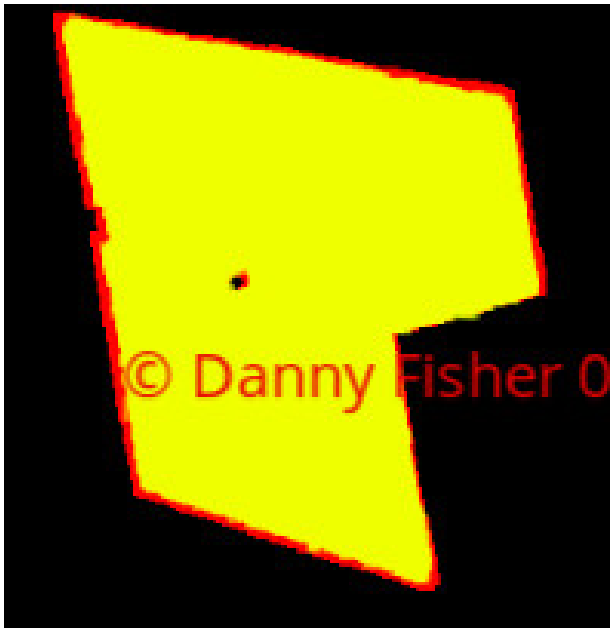

Fig. 12. Student Created World- Occupancy Grid

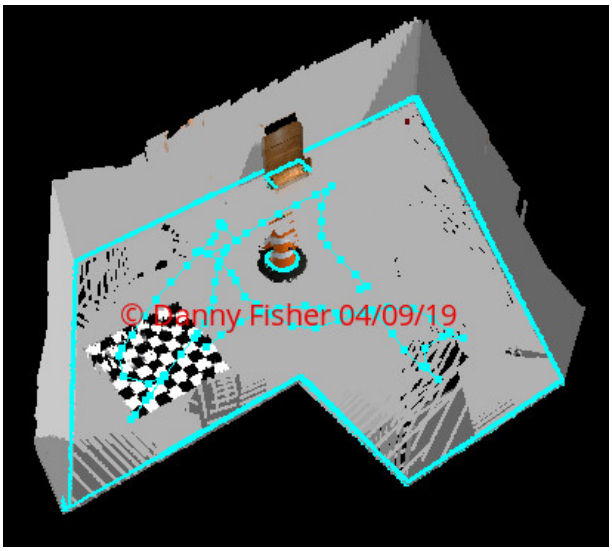

Fig. 13. Student Created World- 3D Octomap

6 Discussion

6.1 Kitchen

The results for the Kitchen world were almost complete. The robot was not able to perform loop closures well (Fig. 11 on the right hand side). Fig. 6 demonstrates the number of visual words generated for a simple wall segment in addition to the 9 loop closures. Lastly the occupancy grid was very accurate even with some subtle misalignment in the right bottom half of the map (Fig. 8).

6.2 Student Created World

The student generated world in this report was a result of multiple attempts to map a small environment successfully. It was found that the feature (cone) in the middle of the room was necessary to enable loop closures, thus provide an entire 2D map, occupancy grid and Octomap. The final results were accurate, however the octomap was missing some floor elements.

7 Future Work

There is plenty of scope for this software to improve upon. One element which warrants further work is optimising the simulation to enable to mapping process to be completed in more cluttered environments. Uploading the algorithms to hardware would highlight the challenges the whether there needs to be further adjustments as the real world is more feature rich. In addition testing the algorithms in the real world would prove the robustness (sensitivity to noisy sensors, to lighting) which is necessary for its eventual implementation.

References

[1] W. J.O., *Robot Mapping*, pp. 11–43. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.