Map My World

Panagiotis Angelakis

Abstract—RTAB-Map is a technique under SLAM, with many advantages like speed, memory efficiency and a wide range of software tools, which make it easy to map 3D environments in real time. In this project we will use rtabmap ros package which is a ROS wrapper (API) for interacting with RTAB-Map. This package will be configured in ros to generate 2-D occupancy maps and 3D octomaps for two simulated worlds. The robot that does the necessary measurements is equiped with an RGB-D camera (kinect v1) and a laser scanner (hokuyo 2-D) providing accurate maps on the simulated environments

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

1 Introduction

RTAB-Map (Real Time Appearance-Based Mapping) is a Graph Based SLAM approach based on an incremental appearance based loop closure detector. This detector uses a bag of words (features in an image) to determine how likely a new image corresponds to a new unexplored location or a previously encountered location. When a loop closure hypothesis is accepted a new constraint is added to the maps graph, then a graph optimizer minimizes the errors in the map. This algorithm is optimized specifically for real time SLAM (memory efficient and fast). SLAM solves the localization problem and Mapping problem in an iterative cycle. In this project two 3D simulated environments are to be mapped using SLAM. One environment is provided and called Kitchen Dining and another one is constructed with Gazebo in a world format.

2 Background

Mapping is essential for mobile robots. It creates a representation of obstacles and navigable terrain in an environment based on sensor measurements. The problem becomes hard when we take into consideration noisy sensors and noisy odometry. There are many mapping algorithms in the literature that try to map small indoor environment or more complex rich feature outdoor environments. For indoor environments a 2-D occupancy map is usually enough to represent walls and obstacles especially for robots that move on a plane, for more complex environments with overhang structures or for flying robots a 3-D octomap is highly preferred. The robot needs to create these maps from scratch while keeping track of it's location, these problem is referred as Simultaneous Localization and Mapping or SLAM. SLAM algorithms need to overcome many challenges, some of them are unknown map, big size of the environment resulting in a huge hypothesis space, noisy sensors and odometry as well as perceptual ambiguity and cumulative odometry error when the robot moves in a cyclic manner without many features to assist it. Some of the most popular approaches to solve the SLAM problem are the following:

- **Extended Kalman Filter SLAM (EKF)**
- **Sparse Extendent Information Filter (SEIF)**
- **Extended Information Filter (EIF)**
- **Fast SLAM**
- **GraphSLAM**
- **Occupancy Grid**

The two most popular of these Algorithms will be further discussed

- **Fast SLAM 1 or 2** solves the full SLAM problem with known correspondences, meaning that it estimates a posterior over the trajectory path using a particle filter approach, giving an advantage to SLAM to solve the problem of mapping with known poses. It then uses a low dimensional Extended Kalman Filter to solve independent features of the map, which are modeled with local Gaussian.
- **GraphSLAM** also solves the full SLAM problem, it recovers the entire path and map which allow it to consider dependencies between multiple poses, it uses a sparse information matrix which relates different poses and landmarks. It is more computational and memory efficient from Fast Slam while also improving on the accuracy. RTAB-Map is a graph based approach to SLAM which will be used in this project, optimized for long term and large scale SLAM by holding in memory only a limited and certain number of images, allowing to perform loop closure in a constant and fast manner so that it can be known in real time.

3 Scene and robot configuration

3.1 Robot model configuration

The simulated differential drive mobile robot was created in URDF format. It is equipped with sensors that allow it to perform SLAM. An RGBD camera is used (Microsoft XBOX 360 Kinect) which provides RGB images as well as per pixel depth information, also a Hokuyo 2D laser scan was used to provide a 2-D laser scan. The hokuyo laser scanner was elevated as to not interfere with the body of the robot. The robot can be seen here.

The tf frames of the different joints of the robot can be seen below. One additional joint was added to the RGBD camera to fix the default orientation.

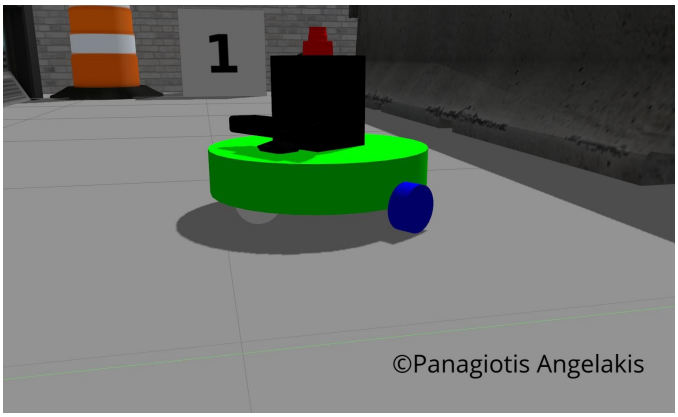


Fig. 1. Robot model

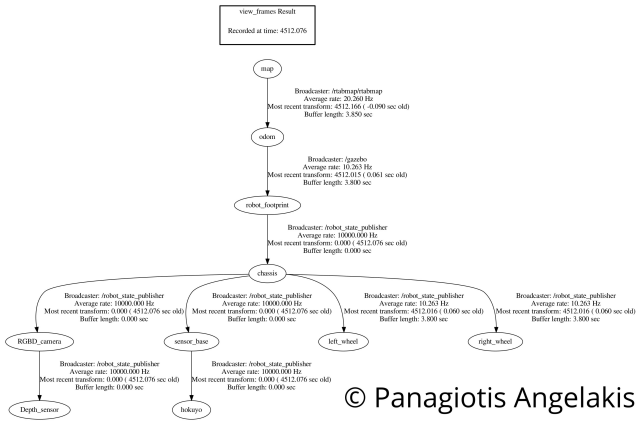


Fig. 2. TF frames

3.2 Gazebo plugins used in Robot Model

- **libgazebo ros diff drive.so** : used for driving differential drive robots with two actuators and no servos.
- **libgazebo ros openni kinect.so** : to simulate Microsoft XBOX 360 Kinect RGB-D Depth camera
- **libgazebo ros laser.so** : to simulate 2-D hokuyo laser

3.3 Ros packages used in the project

The following ROS packages were used:

- **rtabmap-ros** : ROS wrapper (API) for interacting with RTAB-map
- **Rviz** : To visualize the robot, sensor measurements, the map, pointclouds etc..
- **gazebo-ros** : A physics engine,to simulate the robot and publish robot and joint states.
- **teleop** : To manually control the robot with keyboard commands

We can see the different nodes running from the previous packages in the following graph along with the different topics and services.

Rtabmaprviz is an additional node that can be used for real time vizualization of feature mapping, loop closures and paramaters although it is not recommended to run it on a simulated robot due to the overhead, but is a nice addition

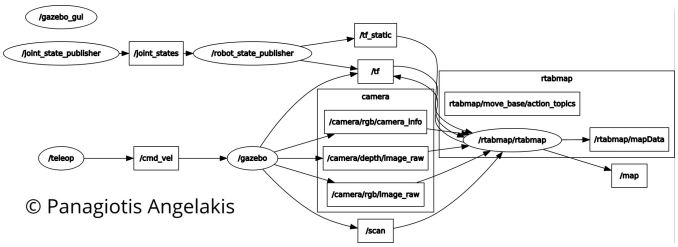
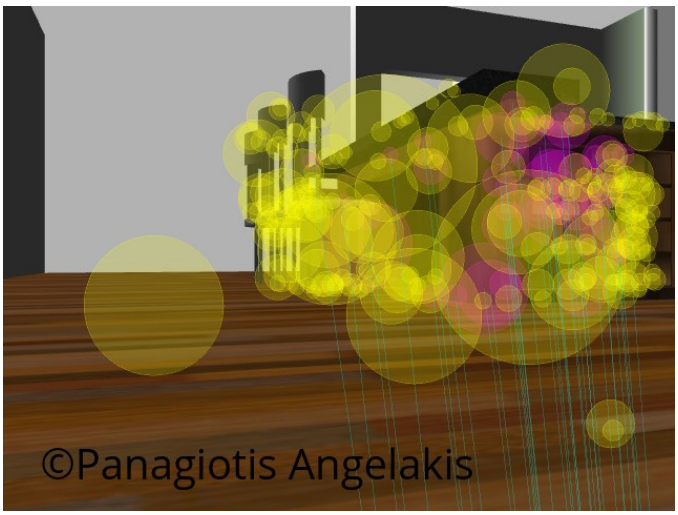
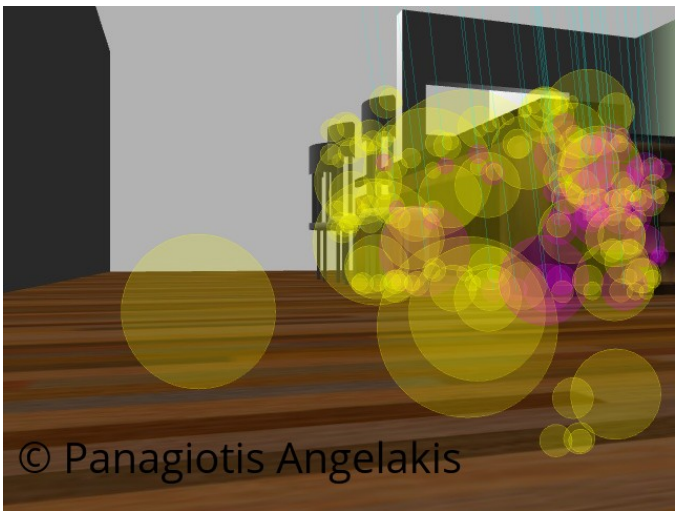


Fig. 3. ROS graph

to a real robot. An example of images used to detect loop closure are the following:

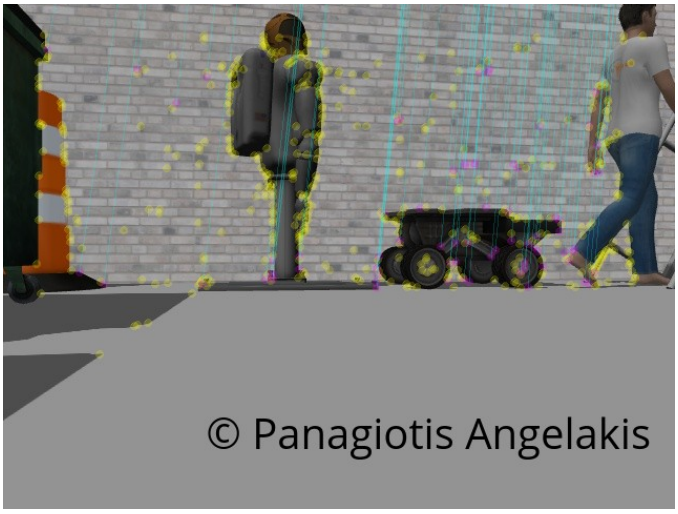




3.4 Paramaters

Many of the paramaters for the above packages and plugins we set to default values. However there are some considerations to take into acount.

- **rtabmap node** needs to subscripe to a /scan topic coming from the hokuyo 2D laser, so the mapping



needs to be correct, also the node needs to subscribe to **camera/rgb/image, camera/depth/image, rgb/camera-info** coming from the RGBD camera.

- **Loop closure technique** For the given environment (kitchen) SURF performed well, however this was not the case for the personal environment where it had difficulty closing the loops, in that case FAST/BRIEF performed better giving an accurate map of the environment.

4 Scene creation

Two environment were used to perform SLAM.

4.1 Kitchen environment

This environment was provided as a testing environmed to perform mapping. The environment represents a Kitchen Dining room, it has many features for our robot to use as landmarks, and the SURF algorithm performed well in this environment. Below we can see the world. RTAB-Map databases for both the Kitchen and the Personal world can be downloaded in the repository.

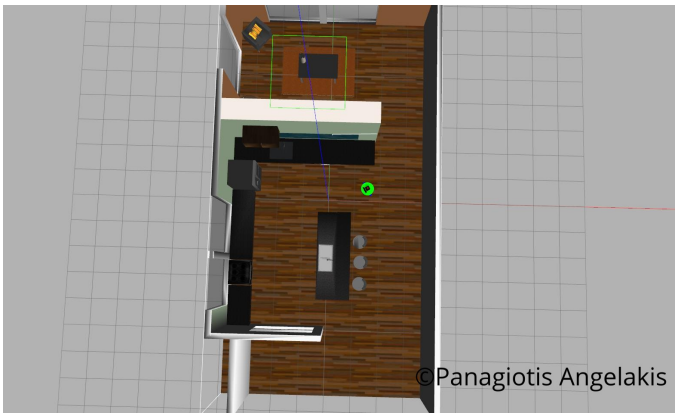


Fig. 4. Kitchen environment

4.2 Designed world

The Designed world was a little smaller in size with many objects on the sides as features. We can see the world below.

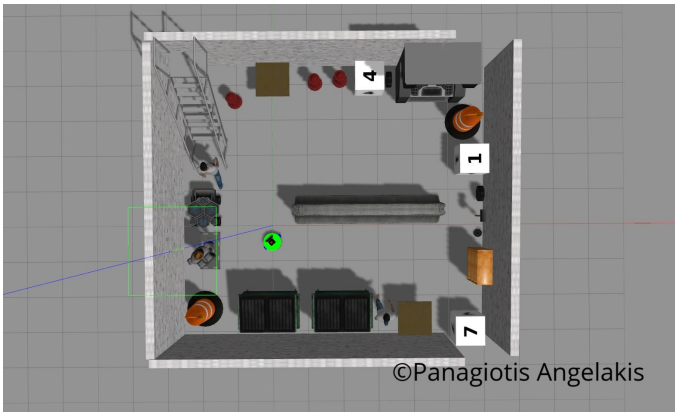


Fig. 5. Designed environment

5 Results

Kitchen world Below we can see the generated 2D occupancy and 3D map for the Kitchen environment, with database viewer we can see the frames that cause loop closure to happen among other things, here we can see 129 Global loop closures achieved.

5.1 Designed world

The generated 2-D occupancy grid map and 3-D map of the Designed world can be seen below. With the help od the database viewer we can see 111 global loop closures achieved

6 DISCUSSION

The provided kitchen world was straight forward, because it had enough features for loop detection, and the SURF loop closure algorithm worked well, with the robot being able to map the whole environment without ambiguity and same objects appearing in two different locations due to bad

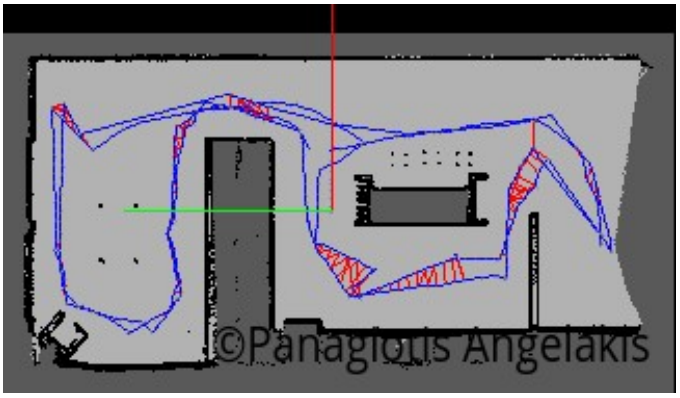


Fig. 6. 2D map of the kitchen

Fig. 7. 3D map of kitchen

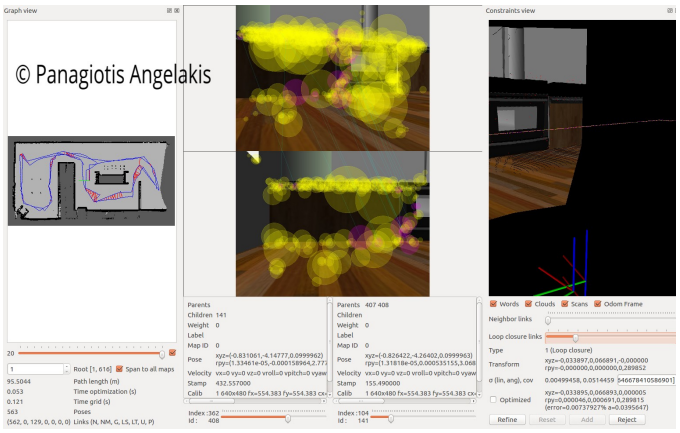


Fig. 8. Database viewer Kitchen

Fig. 9. 2D map of the kitchen

Fig. 10. 3D map of kitchen

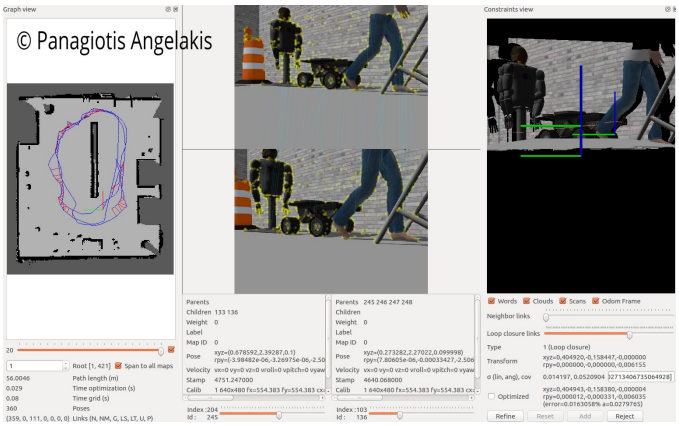


Fig. 11. Database viewer designed world

loop closing. However the first attempt to build a personal world, was designing a huge environment, there the SURF algorithm performed poorly , and the robot lost could not construct a representitive map of the environment. Here different techniques could have been tried to overcome this challenge, maybe a different loop closure algorithm different paramaters or a different SLAM algorithm altogether. Next, the world size was significantly reduced and many objects were placed in the walls. SURF loop closure algorithm performed well here however due to similar objects in the environment some times it could lose track of the position, FAST/BRIEF loop closure algorithm is a more compute efficient algorithm which overcome this problem providing a good quality 3D map of the environment. In conclusion RTAB-Map comes with a lot of flexibility allowing to provide 2D and 3D maps for many different and diverse environments, by tuning different paramaters. However it is a compute intensive algorithm that can benefit from GPU acceleration, so a mobile robot needs to have a capable gpu and compute resources to map the environment.

7 FUTURE WORK

RTAB-Map is a powerfull algorithm for robotics that has the ability to be utilized in a much wider scope of applications for mapping. Some of them are:

- RTAB Map needs a scan measurement and odometry data, if we dont have this sensors (2D laser scanner, wheel encoders or IMU) we can emulate them from an RGB-D camera with the help of the (depthimage-to-laserscan package). This is making possible to map an environment with RTAB using only hand held RGBD camera. Below we can see a map of my house

- RTAB-map can be combined with other pagkages for Navigation, allowing for the robot to map and move in the environment while avoiding obstacles detected from RTAB map. This is a capabillity that space rovers needs to have due to the long delay of human input information.
- GPS periodically corrects the error of IMU sensors in missiles, visual odometry can also improve odometry errors in flying robots.

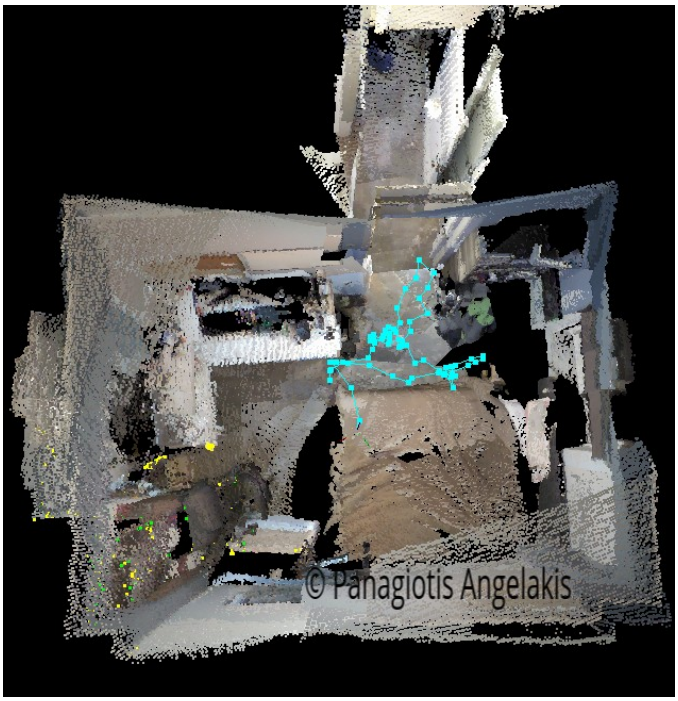


Fig. 12. My room

- Putting real time mapping and navigation planning in a mobile robot in the real world, is what will be next as my diplomatiki for my university. For this to be achieved the robot needs a lot of compute power on board, Jetson TX2 Module is a popular choice