Map My World Robot

Shreyas Chandra Sekhar

Abstract—Robot model with RDBG camera and a laser sensor was considered to perform SLAM using rtabmap ros package on a custom created world. The topics from the RDBD camera was used as RGB image inputs and the depth image was converted to laserscan. Sensor fusion result of these sensors was used as inputs to the robot. A teleop script was created and remapped to provide velocity command to the robot. The robot was navigated through the custom created map until 3 loop closures and the occupancy grid was identified.

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

1 Introduction

AUTONOMOUSrobots have many real time applications. Some of them are Self driving cars, Vacuum cleaners, mining rovers and so on. There are two main tasks in autonomous robots, which are real-time mapping and localization. The robots would have to map the environment real time, which are limited to the on-board system resources.

As the environment of the robot is constantly changing, the robot would have to build an acceptable estimate of the map using the sensory data, trajectory and poses. This is termed as mapping.

Given the map and sensory data, the robot finds the path for the source to destination. This is termed as localization.

The combined problem is called SLAM.

2 Background

SLAM is complicated challenge to solve as map is needed for localization and the Robot poses is need for mapping. However, SLAM is the basic need of a mobile robot to function in an unknown environment.

There are multiple challenges in solving mapping. The map and poses are both unknown to the robot. The possible maps which is termed as hypothesis space in mapping is huge. This further grows exponentially in the continues mapping space. As the on-board resources are very limited, the algorithm would have to be memory and CPU efficient. The inaccuracy of the sensors adds up to this challenge. Having identical objects and places in the map complicates the situation as well as its becomes challenging to distinguish between them. This is known as Perceptual Ambiguity. The noise and in accuracy in the maps amplifies as the robot moves in cycles.

Before getting into the solutions of SLAM problem here is what SLAM attempts to solve. Given measurements z, controls u, solve posterior map m along with poses x. [1]

There are two types of SLAM problems.

previous controls and estimates are not considered in computing the new pose and map. Online SLAM solves instantaneous poses independently from previous measurements and controls.

Full SLAM computes the estimates over the entire path along the map given all the measurements and controls.

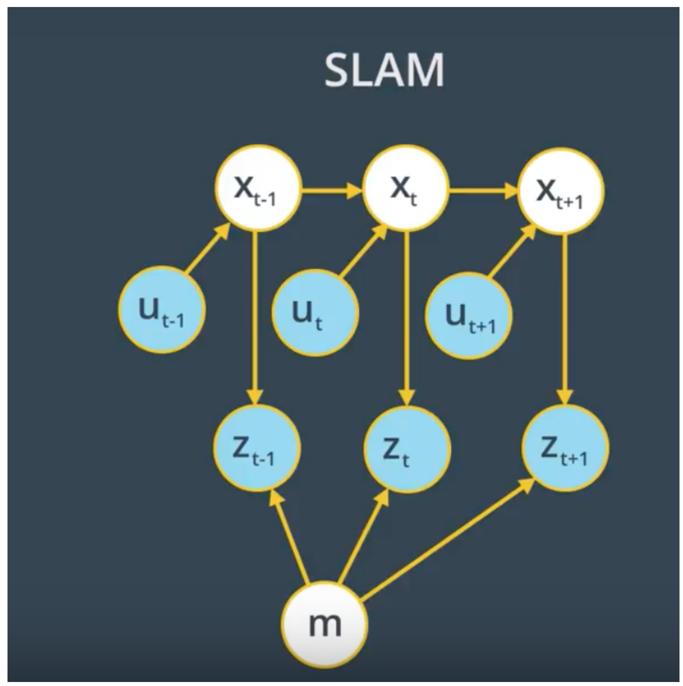

Fig. 1. SLAM

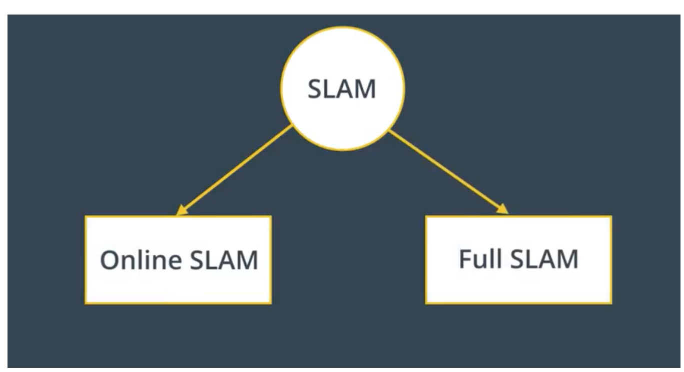

Fig. 2. SLAM Problems

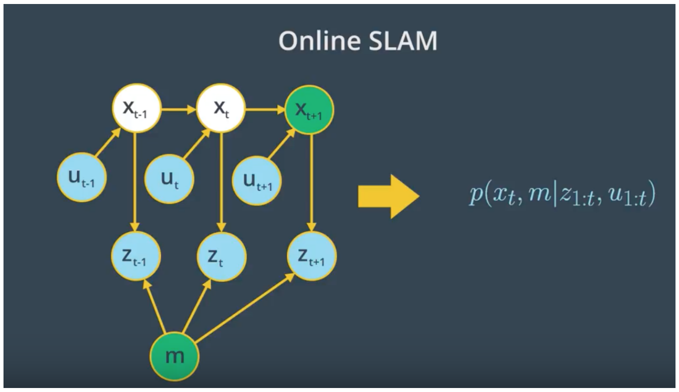

Fig. 3. Online SLAM Problems

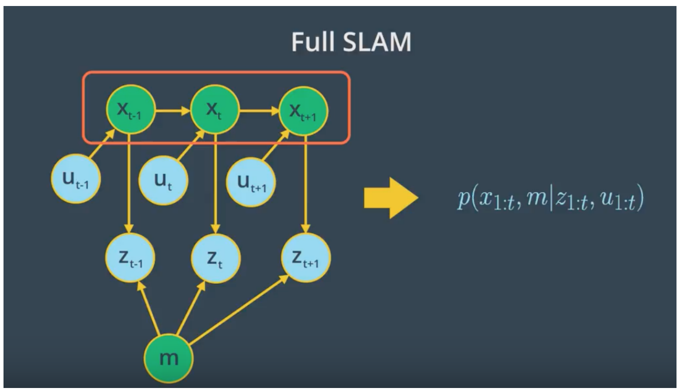

Fig. 4. Full SLAM Problems

[1]

As seen SLAM comes in two forms, Online SLAM and Full SLAM. SLAM feature can also be defined in teams of its nature.

Nature-Continuous indicates the continuous collection of sensory information like odometry to estimate robot poses and continuous sense of environment to estimate the location of the object and map.

Nature-Discrete indicates the estimation of relation between the detected objects during map navigation. This helps the robot to understand if its been in the position earlier. The answer of this would be discrete, that is either 'true' or 'false'. This discrete relation between objects is known by correspondence.

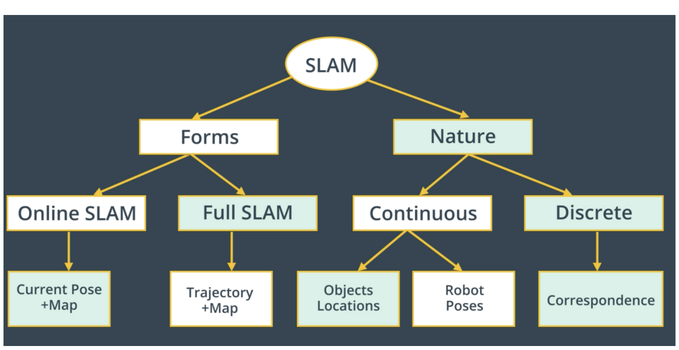

Fig. 5. Nature of SLAM

After adding the correspondence to the Online-SLAM and Full-SLAM the robot can get a better understanding its location by establishing a relation between the objects. [1]

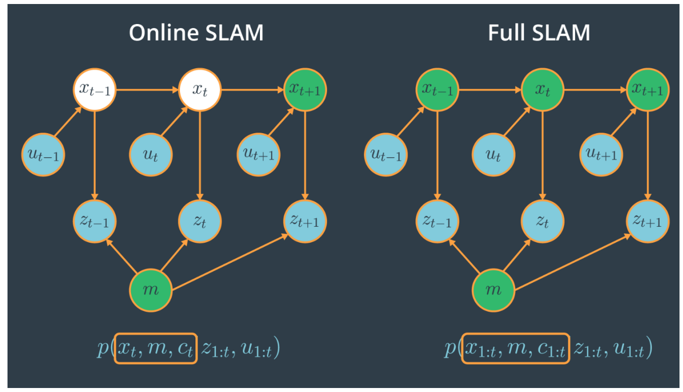

Fig. 6. SLAM with Correspondence

2.1 Occupation Grid Mapping algorithm

The process of generating the map with known poses and no noise sensors is called Mapping with Known Poses.Mapping with Known Poses is represented by the below graph. Where, X is represents poses, Z the measurements and m the map.

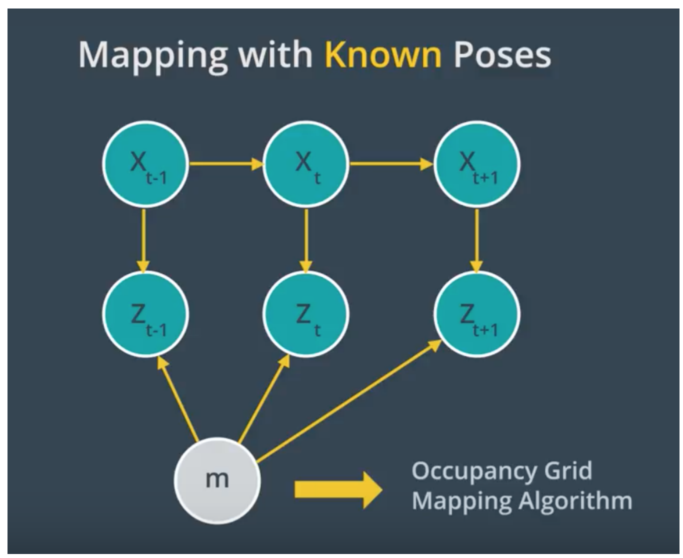

Fig. 7. Occupancy Grid Mapping Algorithm

Occupancy Grid Mapping Algorithm can estimate the map with noisy measurements and known poses. In most real time applications optometry measurements are noisy and mapping happens after SLAM.

In SLAM the problem is mapping with unknown poses rather than known poses. During SLAM, the robot build map of the environment and localizes itself relative to it. After SLAM the Occupancy Grid mapping uses these robot posses. With the known poses from SLAM and noisy measurements, mapping algorithm generates path planning and navigation.

As mentioned the the map estimation becomes expensive with dimensions and it becomes challenging to compute with limited resources on the robot. To overcome the

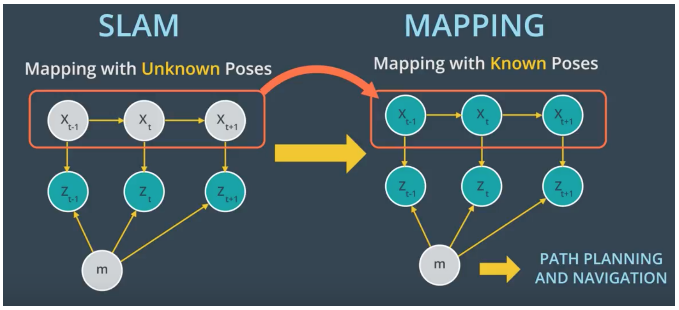

Fig. 8. mapping and SLAM

huge computation memory the posterior map is commuted by relating cells.

$$\prod_i P(m_i|z_{1:t}, x_{1:t})$$

Fig. 9. Mapping relating cells

2.2 Binary Bayes Filter

Binary Bayes Filter solves static mapping problem by taking log's odd ratio of the belief. Depending on the measurement value the mapping grid cells are updated. This state is called Inverse Measurement Model.

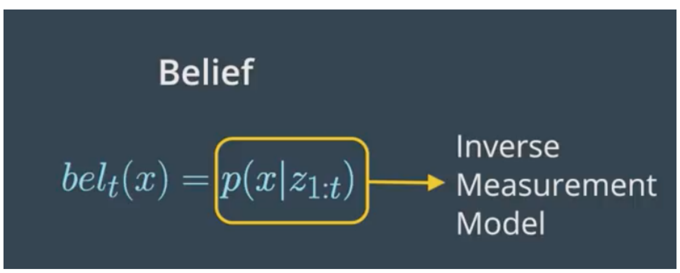

Fig. 10. belief

2.3 SLAM Solutions

There are many SLAM solutions, here are brief of some of them.

2.3.1 *Extended Kalman Filter SLAM (EKF)*

Kalman filters was used for trajectory estimation of the Apollo program by NASA. It has a wide range of applications in controls engineering with high amount of noise. KF works well only under linear Motion, measurement models and when the State prediction is a unimodal Gaussian distribution. EKF is used to over this limitations by considering the small linear motion in the non-linearity. EKF are however too resource intensive and CPU bound.

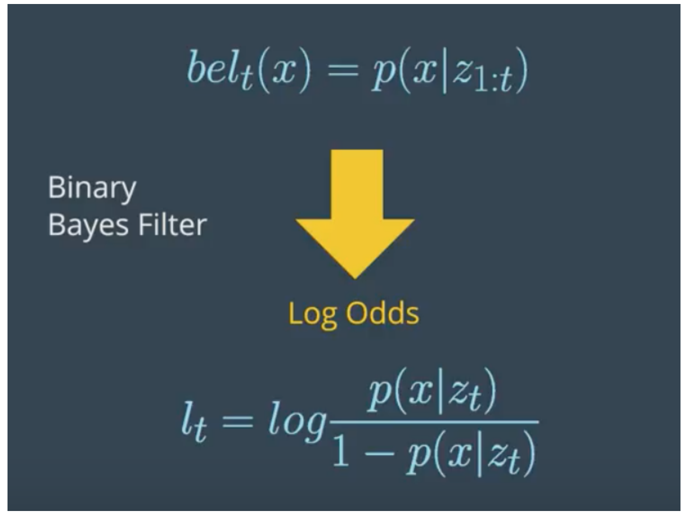

Fig. 11. Log Odds ratio

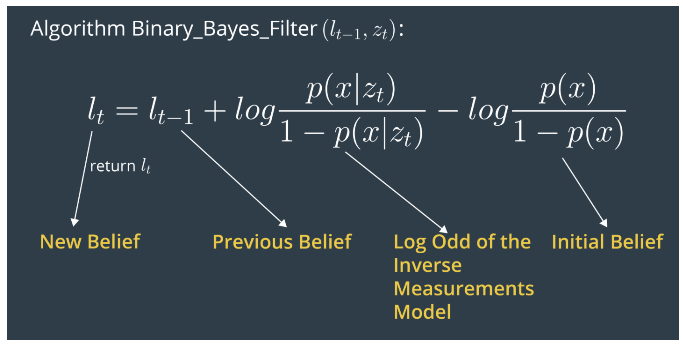

Fig. 12. Binary Bayes Filter

2.3.2 *Particle Filters*

Monte Carlo Localization (MCL) is also known as particle filters as it uses particles for localization. Each particle is a virtual element which resembles the robot. Each particle has its pose in the given map. These particles are depicted from the on board sensors about the likelihood of its current location.

2.3.3 *Fast SLAM*

Fast SLAM users custom particle approach to solve the Full SLAM problem with known correspondences. It estimates the posterior over the robot path along with the map using particles. Each particle provide the estimate of the robot poses. Apart from trajectory, each particle holds a map represented as Gaussian. A low dimensional EKF is used to solve the mini-problem with each sub-map.

Fast SLAM solves Full SLAM problem and each particle solves online-SLAM problem. An extension of of Fast SLAM, called Grid-base FastSLAM can solve SLAM with Non Landmark-Based Algorithm.

Grid-based FastSLAM uses both MCL and Occupancy Grid Mapping Algorithms

This algorithm involves three techniques.

- Sampling Motion estimates the current pose given the previous pose and current control.

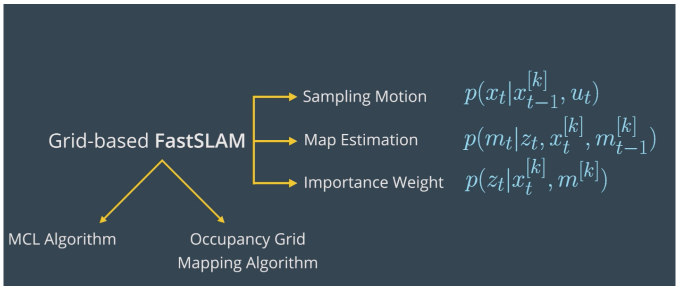

Fig. 13. Grid based Fast-SLAM

- Map Estimation estimates the current map given the current measurements, current and previous pose of the particle.
- Importance Weight estimates the current likelihood of the measurement given the current particle pose and particle map.

There are few limitations of Fast SLAM. As this technique uses particle filters, its quite likely that a particles may not available at the most likely location. This impacts the accuracy of this algorithm in large environment.

2.3.4 Graph SLAM

Graph SLAM solves the Full SLAM problem. It considers the entire path trajectory, map to form a relation between current and previous poses. Graph SLAM needs a reduced amount of on-board processing resources.

Graph SLAM is also much accurate than Fast SLAM.

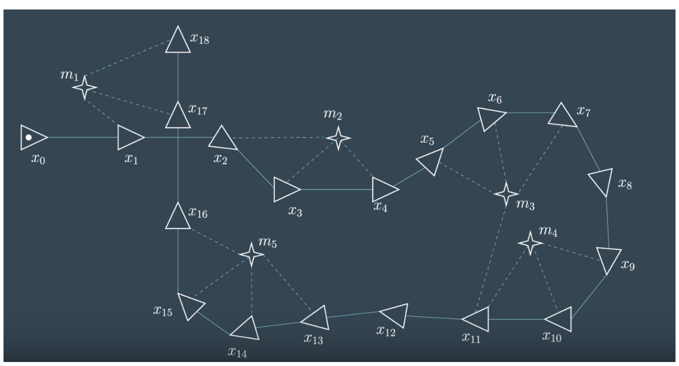

Fig. 14. Graph SLAM representation

- Poses are represented by triangles as they move
- Features of the environment like corners are represented by stars
- Motion constraints represented in solid lines tie together two poses
- Measurement constraints in dashed line tie together a feature and pose

After a graph of all the robot poses are formed with map features, Graph SLAM estimates the most likely robot path and map of the environment considering the constrains.

This task is divided into Frontend and Backend.

- Frontend Frontend constructs the graph using odometry and sensory measurements collected by the robots. They add nodes to the graph as the robot traverse through the environment. It further deals with Perceptual Ambiguity.
  Frontend differs with applications depending upon the map and sensors.
- Backend Backend takes the completed graph with all constraints and comes up with most probable robot poses and map.

2.3.5 RTAB Map

Real-Time Appearance-Based Mapping (RTAB Map) is a grap based SLAM approach. RTAB map uses data collected from vision sensors as the robot moves through the environment to localize the robot and map the environment. Loop-closure is used to determine if the robot has seen the environment before.

As the robot moves through the environment, the number of pose images to compare increases as well. As a result, Loop closures takes longer.

Front-End of RTAB map uses sensor data used for feature optimization. RTAB map uses just Odometry and Loop closure constrains are used and not landmark constrains for feature optimization. The Odometry constrains can come from IMU, lidar or visual odomtry. RTAB map is appearance based with no metric distance information. RTAB map can use RDGB camera to create geometric constrains of a loop closure. Front-end further involves graph management. This includes node creation and loop closure detection using bag-of-words.

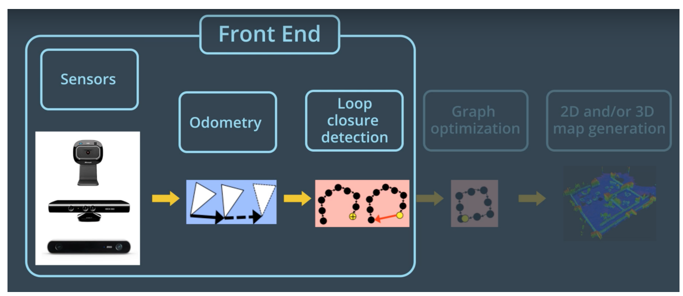

Fig. 15. RTAB Map Front-end

Bag-of-words are used in Vision based mapping. The speed it up robot features are extracted from an image. Each feature has an descriptors associated with it called Feature descriptors. The Feature descriptors are unique representation of the pixel. The point of interest where the feature is located is split into smaller sub regions. The pixel intensity of regularly spaced points are calculated and compared in the smaller sub-regions. The difference in the sample points are used to categorize the subregion of the image. Similar features or synonyms grouped together. The collection of these clusters represents vocabulary. When a Feature descriptor is matched to one in the Vocabulary, the feature is refereed to a word and can be refereed as Visual Word.

When all features are quantized the image is represented as bag of words. Each word associates to a feature in the image. This makes image retrieval efficient over larger data set.

To compare an image with all previous images, a matching score is given to all images containing the same word.

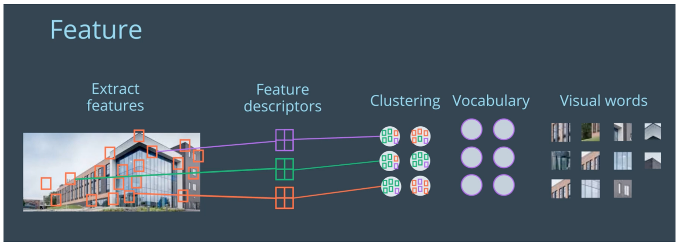

Fig. 16. RTAB Bag of Words feature

Each word keeps track of which image it seen in so that similar images can be found. This is called inverted Index. If a word is seen in an image, its score increase. If the image shares many visual words with the query image, it score higher.

Bayesian filter is used to evaluate the score which is hypothesis that the image has been seen before. When the hypothesis reaches an predefined threshold a loop closure is detected.

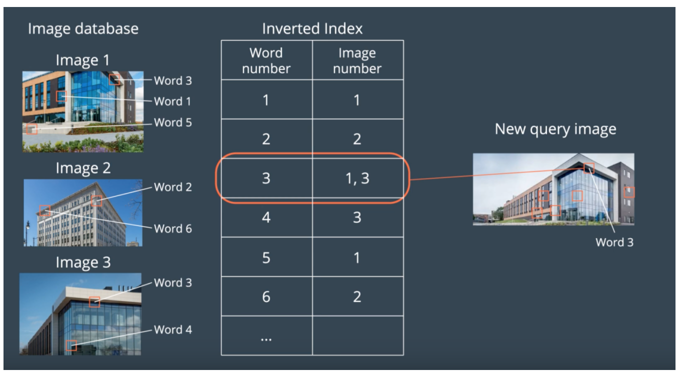

Fig. 17. RTAB Inverted Index

Bank-end of RTAB map involves Graph optimization and assembly of occupancy grid from the graph. There are two types of Loop Closures in RTAB map, namely Local and Global.

- local loop closure Loop closures are detected for a portion of the map region. This approach fails if the estimation position of the robot is in-correct.
- global loop closure Robot location is compared with all the previous visited locations. If no match found the new location is added to memory. This linearly increases the size of memory and time to identify loop closures as the robot moves to new locations.

RTAB map uses global loop closure to detect loop closure in real time.

RTAB map uses Memory Management to techniques to limit the number of images considered as candidates of loop closure. RTAB map keeps the most recent and frequently observed locations in the robot's Working Memory(WM) and moves rest to Long-Term Memory(LTM).

When a new image is acquired, a new node is created in Short Term Memory(STM). On creating a node features are extracted and compared to the vocabulary to create bag-of-words for the node. Nodes are assigned a weight in the STM based on how long the robot spent in the location. When

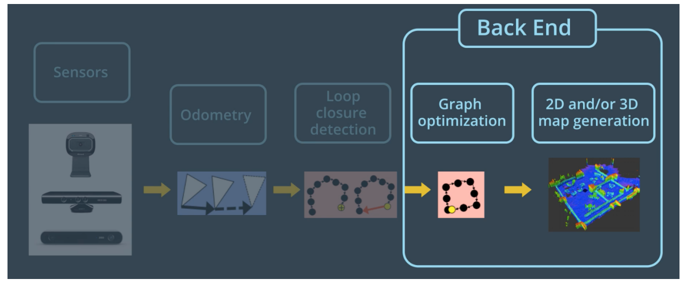

Fig. 18. RTAB Map Back-end

STM reaches the predefined nodes, the older nodes are moved to WM to be considered for loop closure detection. Loop closure happens in WM. WM size depend on a fixed time. When the time required to process new data reaches the time threshold, some nodes are transfered from WM to LTM. So the memory size of WM is close to constant. Oldest of the less weighted nodes in WM are transferred to LTM first, so WM has nodes seen for longer periods of time. LTM is not used for loop closure detection and graph optimization. If loop closure is detected, neighbors in LTM of an old node are transfered back to the WM.

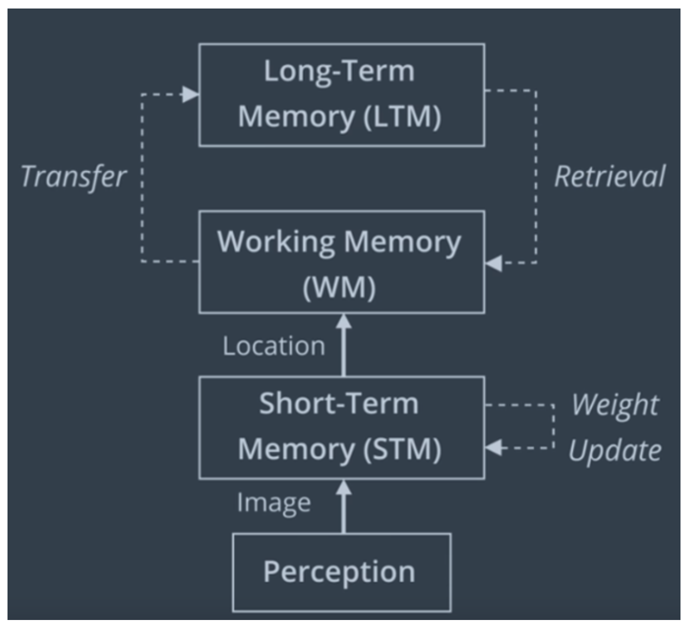

Fig. 19. RTAB Map Memory Management

RTAB map has three graph optimizations.

- Tree-based network optimizer (TORO)
- General Graph Optimization (G2O)
- Smoothing and Mapping (GTSAM)

These optimization uses node poses and link transformation as constraints. The maps are corrected by the errors introduced by odometry on a loop closure detection.

2.4 Scene and robot configuration

A new Gazebo environment was created using a house layout model . This model has multiple rooms with tables and other furnitures.

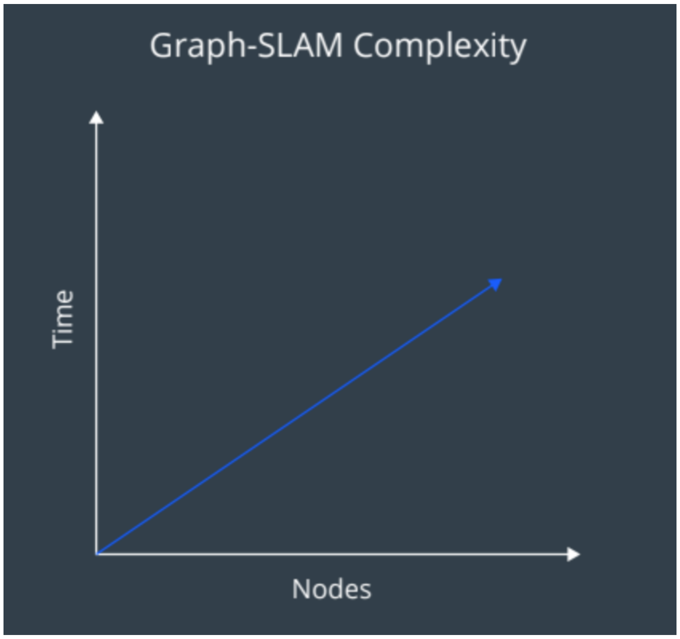

Fig. 20. RTAB map complexity increases linearly with number of nodes

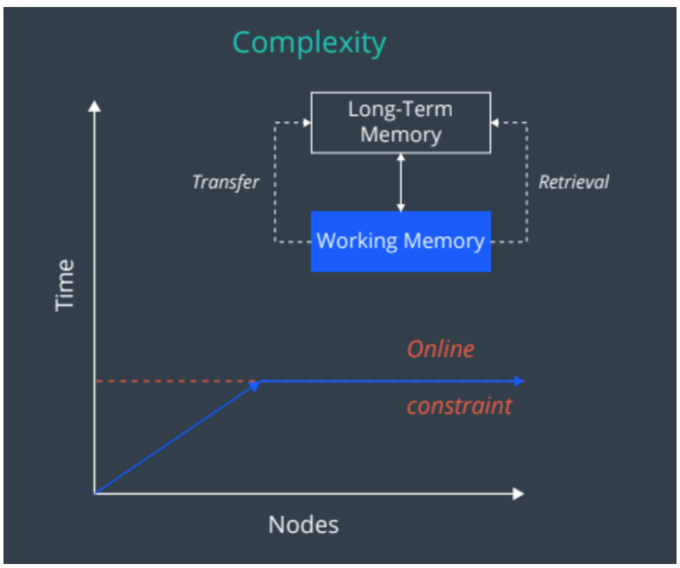

Fig. 21. RTAB map Time complexity constant by controlling nodes considered for loop closure

The robot was built with RDBG camera on the topmost point of its front frame. The RDGB camera was connected to a fake frame to correct the rotation as it was inverted. RDBG camera was uses for both RGB image sensor input as well as layer input. DepthImageToLaserScanNodelet ROS package was used convert RDBG inputs to laser input. Script provided for teleop was configured as robot inputs to control its velocity.

3 Results

The robot could successfully navigate through the map. The 3D map could be build and 137 global loop closures were identified.

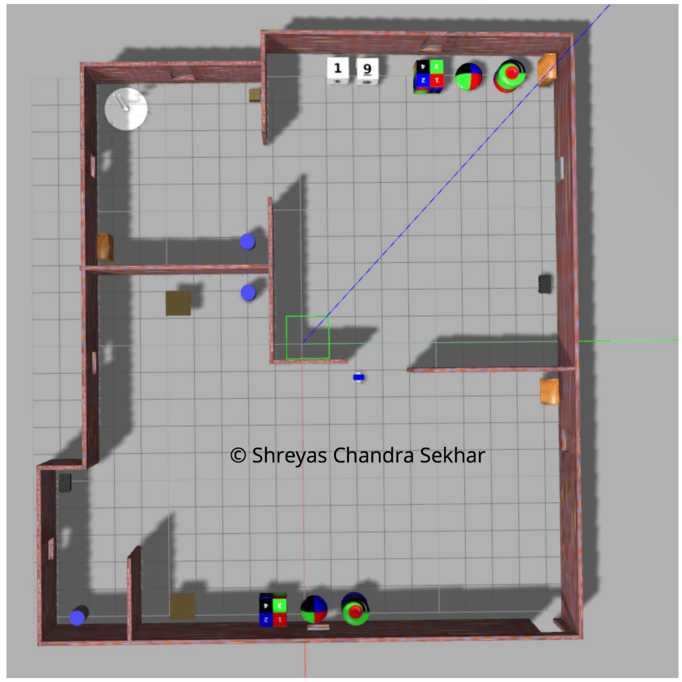

Fig. 22. Gazebo World Model

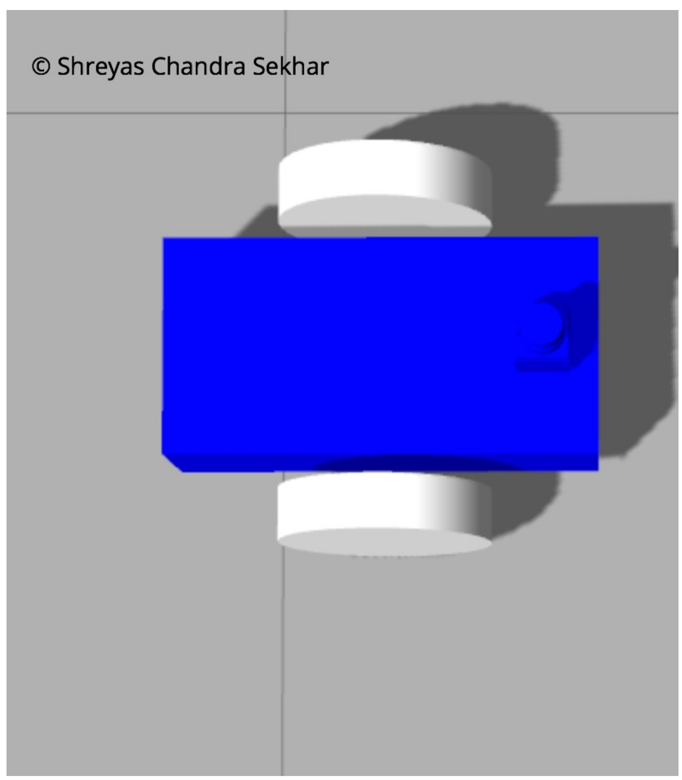

Fig. 23. Robot Model

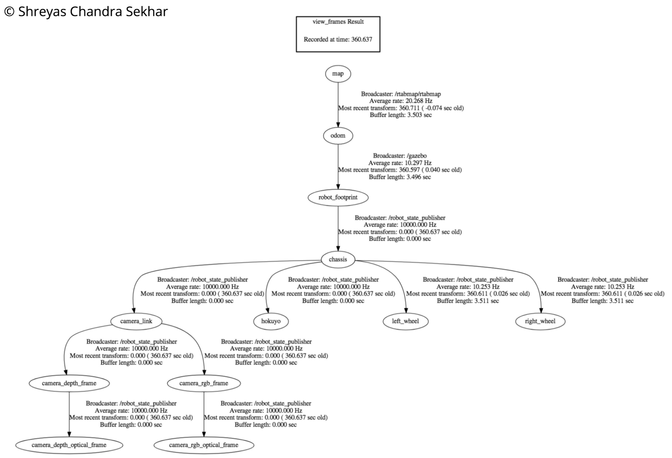

Fig. 24. Robot Frames

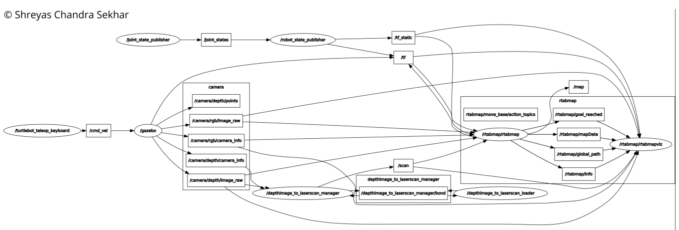

Fig. 25. Robot Ros Graph

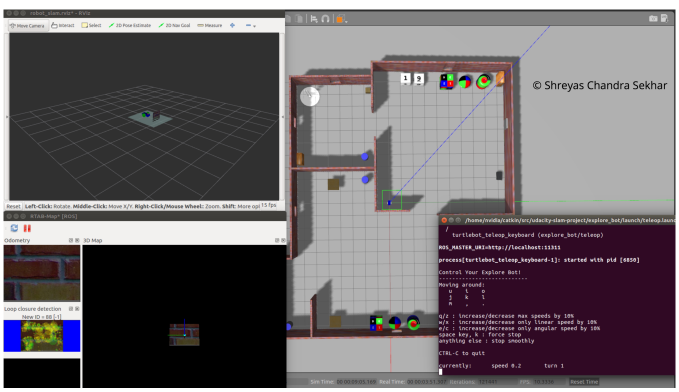

Fig. 26. SLAM

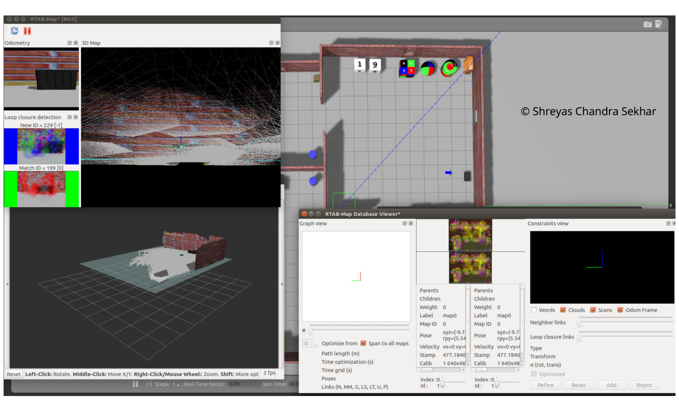

Fig. 27. SLAM

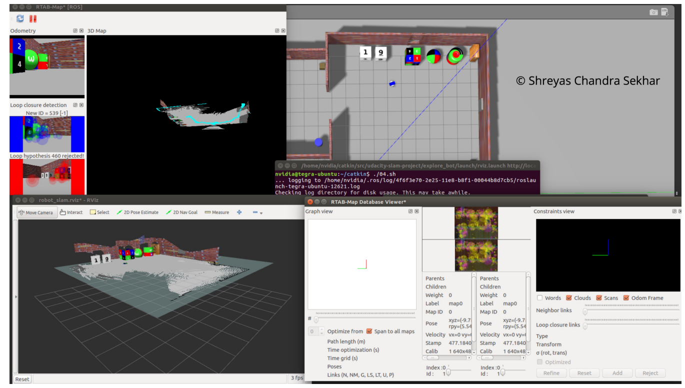

Fig. 28. SLAM

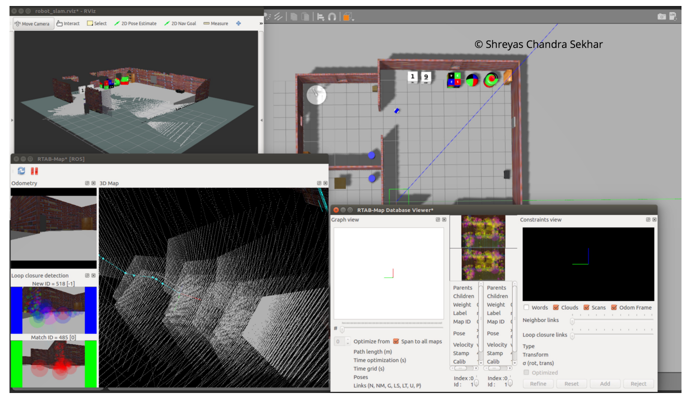

Fig. 29. SLAM

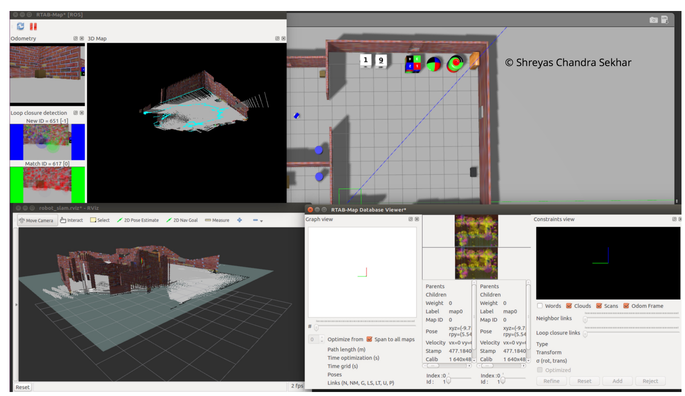

Fig. 30. SLAM

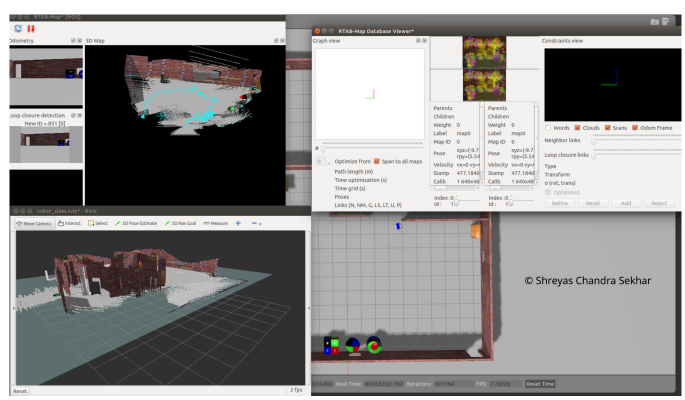

Fig. 31. SLAM

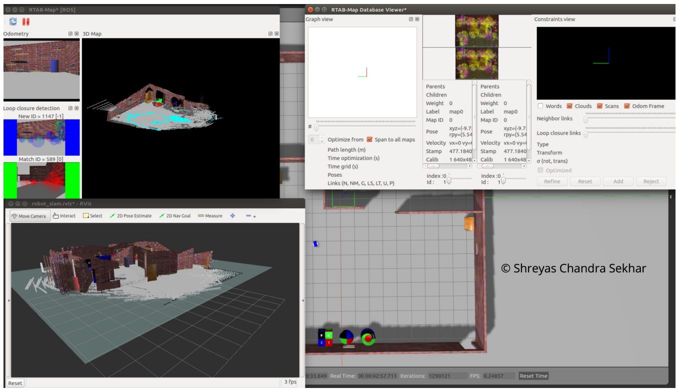

Fig. 32. SLAM

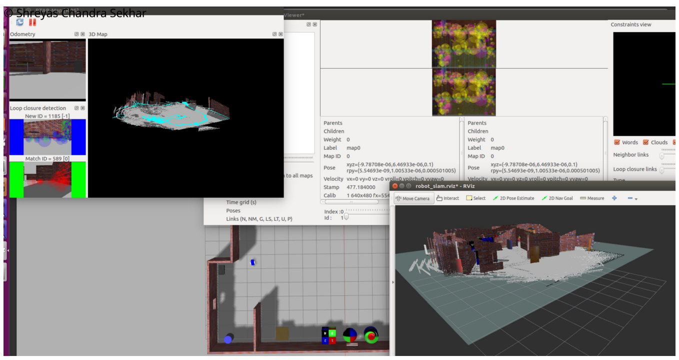

Fig. 33. SLAM

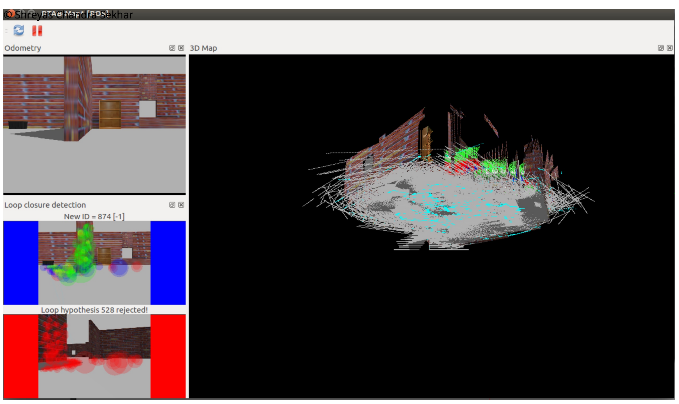

Fig. 34. SLAM completed

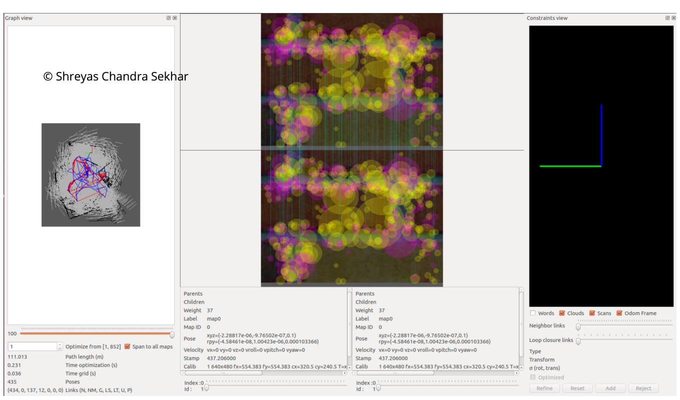

Fig. 35. SLAM completed

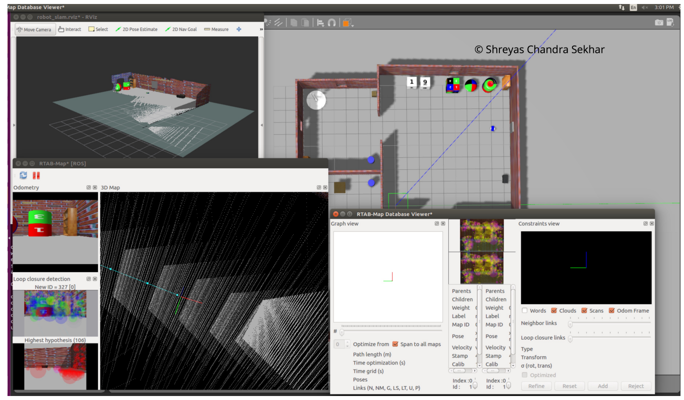

Fig. 36. SLAM completed

4 Discussion

SLAM problem could be solved using RTAB map on TX2. Most of the time not more 6FPS could be achieved and rviz crashed many times. The robot had to navigated multiple times over the map. Choosing a smaller map could have helpful in handling this.

5 Conclusion / Future work

The 3D map could be constructed with more than 130 global detection using RTAB map. In future localization will also be implemented on a smaller environment. This project could be implemented in real hardware

rtab database file is located in result folder of the repository

References

[1] W. B. Sebastian Thrun, Dieter Fox and F. Dellaert, *LATEX: Robust Monte Carlo Localization for Mobile Robots*. Elsevier Preprint, 2001.