Project: Map My World

Rastri Dey

Abstract—This project illustrates an application of Simultaneous Localization and Mapping algorithm leveraging ROS packages of Real Time Appearance based mapping (RTABMap) on robot explored world. A mobile robot is configured with RGBD camera and laser range finder sensors that perceives the robot's environment and implement the mapping principles of SLAM on the simulated platforms of Gazebo and Rviz. RTABMap with its memory and speed management technique to correlate visually similar frames from the sensor captured data and assembling an optimized 2D Occupancy Grid Map or 3D point cloud Octomap, is further deployed on two Gazebo built simulated environments. Moreover, the challenges on SLAM with the complexity of environmental constraints are analyzed through the results based on the case studies of Custom made and pre-supplied Gazebo worlds.

Index Terms— Mapping, Localization, Mobile robots, Occupancy Grid, Sensor data extraction

I. Introduction

Mapping on dynamically evolving environment is a paradigm shift in mobile robotics application. It's a technique to develop a robot map with identifiable features from the environment, through the sensory measurements obtained and the trajectory traversed by the robot. In technical terminology, mapping involves estimation of a posterior over the map, given the robot's poses and measurements. Mapping plays a vital role in many applications. It is particularly useful for an unexplored area, such as in the domain of space research for a newly discovered land in another planet that has not been previously known to mankind. Mapping will play a crucial role in exploring this new environment and bring it before the real world people. While in other applications there could be fair enough circumstances of a known map being present through a previously generated map by a mobile robot or a digitized map from satellite data, reality might limit it from being a viable solution, at many a times. For instance, the surroundings of a known environment can be prone to random changes and the existing map may not be up to date. Hence, even if a known map is provided, the robot must build an instantaneous map of its own to account for the changes which might have taken place on the existing environment. The external forces imposing shift in obstacles or a mere readjustment can render the existing map information to be futile. Furthermore, mapping is also required in static environments since sensor measurements are always noisy and accumulate errors, which at times can produce low accuracy maps. A good mapping algorithm will aid in producing high-accuracy maps through which the robot must simultaneously navigate and update itself. This calls for the requirement of robot localization while mapping (or SLAM) where, the robot must localize itself and define its trajectory while producing the map. In localization, the robot is having access to the map of its environment, sensory data, and actuation control with the help of which the robot calculates and updates it's poses in its trajectory. These robot poses together with measurement data serves as an input to the mapping process. Thus in a broader aspect, with slam, localization and mapping share each other's inputs and outputs.

This project explores the technology behind Simultaneous Localization and Mapping (SLAM) to work in a simulated environment. The SLAM algorithm used is Real Time Appearance Based Mapping (RTABMap), a type of GraphSLAM algorithm that uses the data collected from vision sensors to localize the robot and map its environment. A technique of loop closure is inherent to this algorithm that accounts for the correspondence problem. In the project, a simulated robot is configured with RGB-D camera and laser range finder sensors to map feature data as the robot traverses through the environment.

Furthermore, two case studies of different environments are configured for evaluating the SLAM performance on the robot deployed. The developed map is scrutinized via rtabmap-databaseViewer that stores the output map in a local database. The robot after a successful SLAM implementation, is evaluated on RTAB-Map ROS wrapper with visual representation of real time rtabmapviz. The resultant output produces a map with identifiable features from the surroundings in 3D and 2D forms.

II. Background & Formulation

Mapping a 2-dimensional space builds an information database of the environment robot is exposed to with prime focus on the available space vs obstacle present in the environment at a prescribed z-ordinate. While, mapping a 3-dimensional space generates identifiable feature rich environment in all the x-, y- and z-axes, rendering a complete information of the robot's surrounding. SLAM is an extremely

crucial feature in mobile robotics application that handles mapping in both the two and three dimensional space via various SLAM algorithms. From surveillance drones to self-driving vehicles, strategic planning and cognitive decision actions, relies on the scanned inputs of a robot environment. With SLAM, the robot estimates its pose and simultaneously build a map of its environment given noisy measurements and actuation controls. This brings in, two separate challenges of accurate Localization and appropriate Mapping into one, raising the bar to a further challenging problem of SLAM. In Localization, the robot estimates a posterior over the its pose given the measurement data, a known map and robotic controls. This can be shown as:

$$P(x_{1:t} \mid u_{1:t}, m, z_{1:t})$$

In mapping on the other hand, the robot estimates a posterior over the map of its environment using its known trajectory and measurement data. This is also referred to as the post processing phase of SLAM since localization, outputs the requirement of known poses to Mapping. Mapping in probabilistic terminology, can be represented as:

$$P(m \mid z_{1:t}, x_{1:t})$$

SLAM merges localization and mapping to develop a consolidated Map and Trajectory tracking for a mobile robot which in terms of posterior can be represented as:

$$P(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$

Here, x denotes the robot pose, m denotes the map, u denotes the actuation control and z denotes the measurement data over time instant 1(first) to t (current).

a. SLAM Challenges

In addition, to solving the localization and mapping constraints individually, there are numerous other challenges in building a robust SLAM feature in mobile robotics application. For example, handling noise in measurement data and avoiding continuous error accumulation over time that may lead to erroneous mapping and localization inaccuracies. This challenge is solely owed to sensor specification and environmental conditions that cannot be realistically controlled. Further, the robot being exposed to large environments with unknown map will need a huge hypothesis space leading to increase in computational burden. As the map expands, the SLAM must be optimized to account for map expansion and long-term data acquisition for processing. This would require a proper memory management using a suitable SLAM methodology. Moreover in a large environment the robot should account for images with similar visual appearances imposing the problem of perceptual ambiguity. Also while travelling in cycles, the robot should correlate between different objects seen at different instants of time. This is related to the correspondence problem, where the object correspondences are calculated over time to estimate the object identity from the stored database of past history of robot data with respect to the current images captured. However, the correspondence values increase exponentially over time as the robot keeps exploring the environment while capturing the images.

Adding to this, the SLAM problem also has two variants of—Full SLAM and Online SLAM. The Online SLAM problem tries to estimate the current pose and update the map given the current measurements and controls, while the Full SLAM attempts to estimate the entire trajectory and update map given, all measurements and controls. Moreover, computing the full posterior composed of the robot pose, the map and the correspondence under SLAM poses a big challenge in robotics mainly due to the continuous and discrete parameterization. The continuous parameter space involves the robot encountering various objects in the environment, mapping and keeping track of them while localizing itself as the time passes by. The discrete parameter space on the other hand accounts for the correspondence values that exponentially increases over time as the robot keeps sensing the environment and relating newly detected objects to the previously detected ones. Under known correspondences and non-noisy measurements, the SLAM problem becomes less challenging yet one needs to rely on approximations to estimate the posterior.

b. SLAM Algorithms

The stated issues in SLAM can be resolved with the algorithms of varying scopes including FastSLAM, Grid-based FastSLAM and GraphSLAM. FastSLAM solves the Full SLAM problem with the known correspondences by estimating the Trajectory through particle filter approach and estimating the Map using a low dimensional Extended Kalman Filter. The custom approach of representing the posterior with particle filter to solve the problem of mapping with known poses and EKF to solve independent features of the map modeled in local Gaussian, consolidates to build a Rao-Blackwellized particle filter approach. FastSLAM is a landmark based approach and holds the assumption of known landmark positions, that makes the FastSLAM modelling unfeasible for an arbitrary environment. The Gridbased FastSLAM is a variant of FastSLAM algorithm that combines the MCL (Monte Carlo Localization) and Occupancy Grid Mapping to estimate the robot trajectory and the map through the grid cells labeled as occupied, free and unknown by assuming known poses. The poses under MCL is estimated based on the likelihood approach of weight allocation to the particle distribution. Though Gridbased FastSLAM can resolve the limitation of having a predefined landmark position, there are room for errors in the FastSLAM model due to the finite number of particles approach, estimating the robot's most likely pose. For example, there could be chances of no particle available in the robot's most likely location, which is more significant for larger environments.

GraphSLAM solves the Full SLAM problem by working with all the data to find the optimal solution with a high accuracy through removal of particle based estimation, thus reducing significant onboard processing requirments. In a GraphSLAM approach to solving the SLAM problem, the poses of the robot's trajectory and the measurements are denoted with nodes while the estimated motions and measurement distances are denoted as links. The links are referred to as the constraints considered in the optimization process to find the system configuration that produces the smallest error. In this project, a Graph-based SLAM approach called Real Time Appearance Based Mapping or RTAB-Map is deployed on the simulated robot.

c. RTAB-Mapping

RTAB-Map uses the data acquired from vision based sensors such as Camera to localize the robot and map the environment. It is optimized for large scale and long term SLAM to allow for loop closure to be done in real -time. This technique includes the node creation and loop closure detection using bag-of-words approach of feature extraction process called Speeded Up Robust Features or SURF in an image. The camera images captured from an RGB-D sensor for instance constantly compares the current image from the stored database of previous images under a constant time and memory management framework. When a loop closure hypothesis is accepted, a new constraint is added to the map's graph and the graph optimizer minimizes the errors and propagates it to all links, correcting the map. Thus RTAB-Map algorithm maps the environment with repeated locations at a reduced constraint complexity and an efficient memory management framework enhancing the speed and processing capabilities while assembling an optimized map.

III. Scene & Robot Configuration

A simulated environment for Robot and its surroundings is built in Gazebo, underlying ROS packages. Robot and world configuration files are developed to suffice Mapping in an unique application of 'Door To Door Service'. Door to Door service is a world created for the robot to navigate and deliver parcels at a residential area by following the address corresponding to the parcel label carried by the robot to the respective house addresses. Below sections provide further insights into the configuration details.

a. Robot Configuration

A robot is configured with its urdf (Unified Robot Description Format) file that corresponds to its kinodynamic properties, visual elements and model sensors. It describes the robot's rigid links connected by joints in a chain or tree-like structure. The robot structure utilized as shown in Fig.1, is inherited from a previous project. It is having two left and right wheels, two caster wheels, three disks, six supporting legs and sensors namely camera and laser range finder (hokuyo). 'camera_joint' as mentioned in the Table.1 refers to the camera sensor and 'hokuyo_joint' refers to the laser sensor. 'camera_rgbd_joint' is a transform from the camera to the rgbd camera frame, which is the link attached to the RGB-D camera node in the .gazebo file. The measurements of different model components, their respective geometry and their position are given with respect to origin (0,0,0) in Gazebo frame. The net length towards x-, y- and z- axes refers to the net physical dimension of the robot body. The names in 'Model Component' are in reference to the urdf file used in configuring the robot design. A complete description on individual component position, dimension, sensor layout and structure is illustrated in the following table.

Model Component	Element Type	Element Dimension	Reference Origin in XYZ world frame		
			X -axis	Y-axis	Z -axis
chassis_visual	cylinder	radius=0.2 length=0.02	0	0	0.1
left_leg_visual	cylinder	radius=0.01 length=0.4	0	-0.15	0.3
right_leg_visual	cylinder	radius=0.01 length=0.4	0	0.15	0.3
front_leg_visual	cylinder	radius=0.01 length=0.4	0.15	0	0.3
back_leg_visual	cylinder	radius=0.01 length=0.4	-0.15	0	0.3
disk2_visual	cylinder	radius=0.3 length=0.02	0	0	0.5
left_leg2_visual	cylinder	radius=0.01 length=0.4	0	-0.15	0.7
right_leg2_visual	cylinder	radius=0.01 length=0.4	0	0.15	0.7
disk3_visual	cylinder	radius=0.3 length=0.02	0	0	0.9
back_caster_visual	sphere	radius=0.05	0.15	0	0.05
front_caster_visual	sphere	radius=0.05	-0.15	0	0.05
left_wheel_hinge	cylinder	length=0.05 radius=0.1	0	0.2	0.1
right_wheel_hinge	cylinder	length=0.05 radius=0.1	0	-0.2	0.1
camera_joint	box	Length=.05 Width=.05 Height=.05	0.27	0	0.54
camera_rgbd_joint	-	-	0.33	0	0.54
hokuyo_joint	box	Length=.1 Width=.1 Height=.1	0	0	0.95
Net Robot Dimensions (Length X Width X Height) =0.6 X 0.6 X 1 (Radius X Height) = 0.6 x 1			0.6 Length = 0.6	0.6 Width = 0.6	1 Height = 1

Table. 1. Design layout of robot model

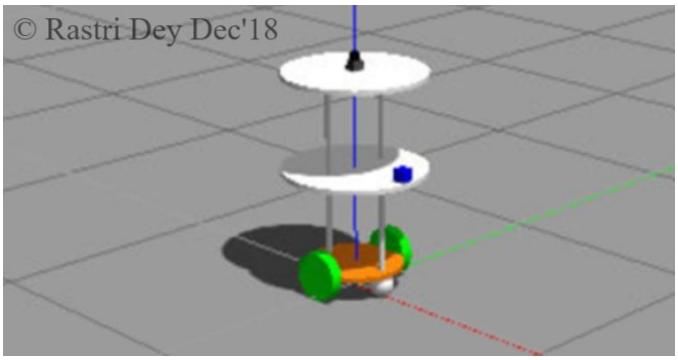


Fig. 1. Representation of designed Robot model in Gazebo

The Front View of the designed robot model in 2D diagram can be represented as below:

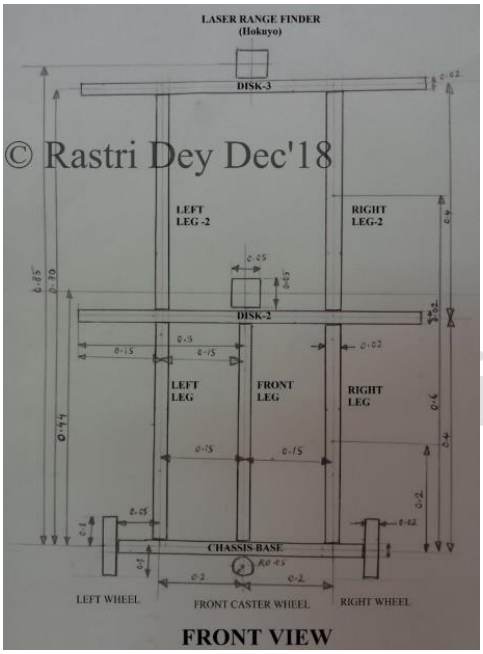


Fig. 2. FRONT VIEW of Robot

The transform tree of Robot setup can be graphically represented as:

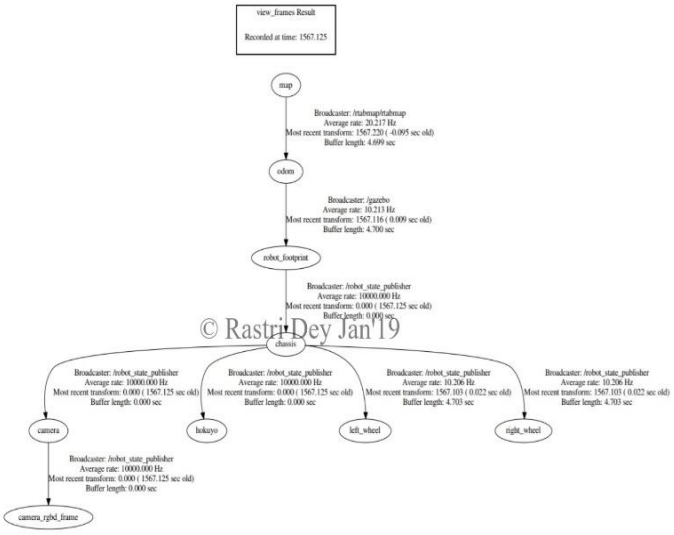


Fig. 3. Robot Transform Tree

b. Scene Configuration

A custom world is designed to facilitate the application of robot mapping in a real world city environment where the robot requires to address the fictional job of Door to Door Service of parcel delivery. A world similar to a residential area is constructed in Gazebo using the online model database. The Gazebo made world can be shown as:

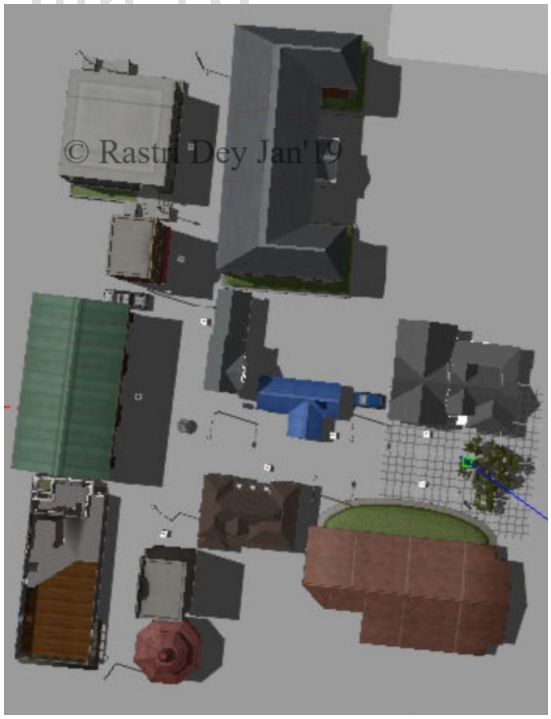


Fig. 4. Representation of custom designed world in Gazebo [TOP VIEW]

The custom designed world is comprised of 3 Lanes namely- Lane 1, Lane 2 and Lane 3. The robot's initial state rests at Lane 1.

Fig. 5. Custom world specification & Robot Initialization in Gazebo

At every Lane, various houses from Gazebo model database are elected and placed. Alongside every house, there are numbered boxes ranging from 1 to 9 to denote the individual unique address with respect to every house in the residential area. There are trees and look alike lamp posts placed at a few corners to represent a general city sight. The robot encounters general houses, post office, post box, parked vehicles, trees, lamp posts in Lane 1, school and other buildings in lane 2 and a café, salon and a few houses in lane 3. There is a center fountain at the merging point of 3 lanes.

The chief idea in building this world is that- the robot needs to travel till café in Lane 3, get the parceled food and deliver it to their respective owners staying in different lanes by linking the corresponding address of numbered boxes in the front of the houses from the numbered addresses labeled in the parcels. The application of delivery service is addressed with the robot mapping and localizing using RTAB-Map ROS package into a Gazebo built in simulated environment.

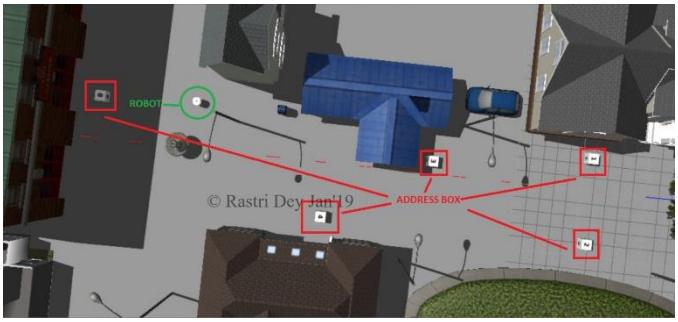


Fig. 6. Representation of 'Address Box' for Door To Door Service application

To satisfy the task, the robot must map the entire world, to localize itself and differentiate between similar looking houses or objects from one another. A small gazebo is also placed in the third lane of this world, having symmetrical dimensions at its circular 360 degree base. The robot while traversing from one lane to another always comes across a center fountain, the mapping of the fountain from all of its sides could be challenging as the robot must map accurately from different camera sensor viewing angles. The constructed Gazebo world is having similar textured long walls of houses at different lanes which evaluates the robot performance on mapping. The robot further needs to differentiate between the address boxes with similar white background. With the entire world configured, the robot is given enough space to map with its built in sensors and explore the world.

The package structure of the files developed for robot and scene configuration, looks like:

```
├── CMakeLists.txt
├── launch
│   ├── mapping.launch
│   ├── robot_description.launch
│   ├── rviz.launch
│   ├── teleop.launch
│   └── world.launch
├── config
│   └── robot_slam.rviz
├── meshes
│   └── hokuyo.dae
├── Outputs
│   ├── Transform_tree
│   ├── Robot_Configuration
│   ├── Door_To_Door_Service
│   └── Kitchen_dining
│   │   └── rtabmap.db
├── package.xml
├── rtab_run
├── teleop
├── urdf
│   ├── slam_bot.gazebo
│   └── slam_bot.xacro
└── worlds
```

```
|   ├── kitchen_dining.world
|   ├── Door_To_Door_Service.world
|   └── empty_world.world
```

Fig. 7. Schematic representation of package structure of robot and scene configuration

IV. RESULTS

Two case studies have been demonstrated in this report to illustrate the Mapping and Localization problem for two separate worlds.

a. Kitchen Dining World

The robot is allowed for exploring a kitchen_dining world under Gazebo environment using its camera RGBD and laser range finder sensors. The visual representation of robot mapping in Gazebo environment can be shown as:

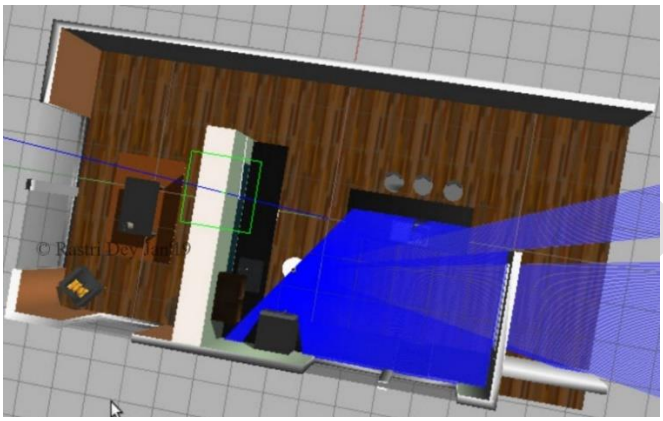


Fig. 8. Robot mapping in kitchen_dining.world in Gazebo

The blue rays are the visual laser rays hitting the objects inside the room from the robot's head sensor. The corresponding mapping in its Rviz counterpart looks like:

Fig. 9. Robot mapping in kitchen_dining.world in Rviz

The generated map of kitchen_dining.world resulted in global loop closures of 83 in number. The other parameters resulted from mapping, gives a value of 684 Neighbor and 0 for rest of the - Neighbor Merged, Local loop closure by space, Local loop closure by time, User loop closure, and Prior link. The description of this from the Graph view of RTAB-Map database viewer can be shown as below:

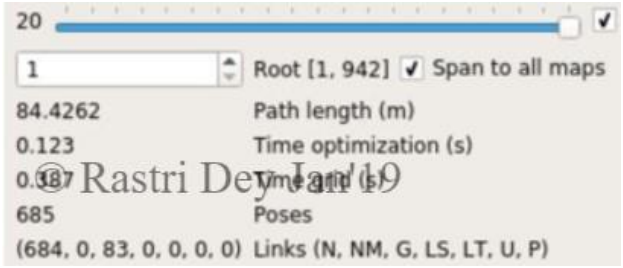


Fig. 10. Loop Closure in Graph View of RTAB-Map Visualization Tool [G=83]

The minimum visual inliers to accept the loop closures (param name="Vis/MinInliers") is set to 40 to avoid linking of similar textured objects as one. The global loop closures are obtained from the robot moving a multiple number of times around the same area. The obtained 2D map of kitchen_dining.world can be shown as below:

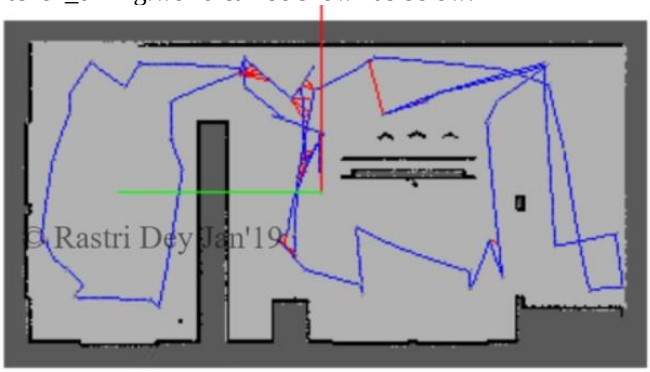


Fig. 11. 2D Map of kitchen_dining.world

The Occupancy Grid Map generated from the robot, mapping the Kitchen Dining area, can be shown as:

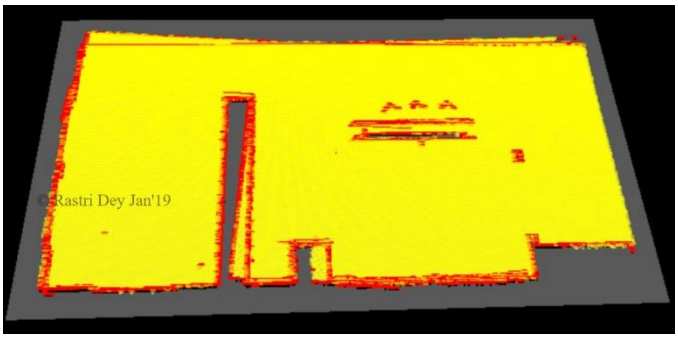
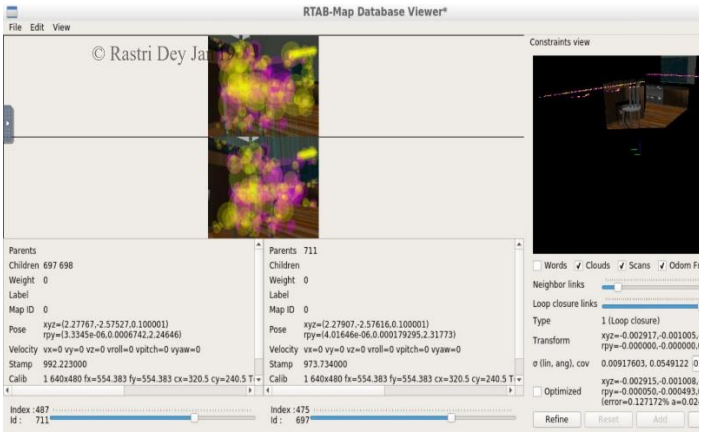


Fig. 12. Occupancy Grid Map of kitchen_dining.world

The robot while mapping in the Gazebo and Rviz simulated environments, generates a 3D map of the world, it explores. The 3D map is a point cloud data obtained from RTAB-Map databaseViewer tool. The representation of this can be found as below:

Fig. 13. 3D Map of kitchen_dining.world

The robot faces loop closures as it traverses across the same area a repeated number of times. The 3 sample loop closures detection in RTAB-Map are shown as below. It could be observed that the pink blobs indicates the common features of two different images taken at different times.



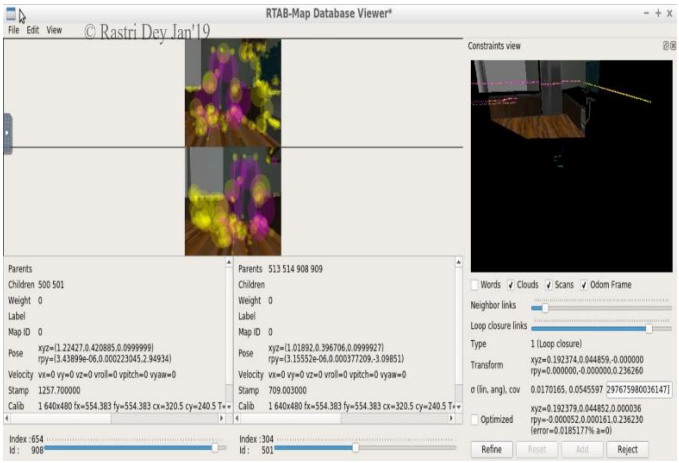


Fig. 14. 3 sample Loop Closure of kitchen_dining.world

As it could be inferred, the robot genereated a 2D map laying the boundaries of the Kitchen dining area with the three chairs, a separated wall and a cabinet, though failing at reproducing a few other articles of the room such as table, show-piece etc due to the sensor placement at a certain height on the robot body. This is mitigated in the 3D map layout, where most of the articles in the room and the boundaries of the wall can be clearly visualized. Even though owing to the robot's height and sensor elevation angles, a complete 3D recreation of the room and its articles such as the blank portion of table top has not been made feasible.

b. Custom-designed World

2D and 3D map layouts are obtained from the custom-made world of 'Door_To_Door_Service.world' through linking the package configuration of RTAB mapping with the Gazebo built-in world layout. The robot while mapping the custom world in Gazebo looks like:

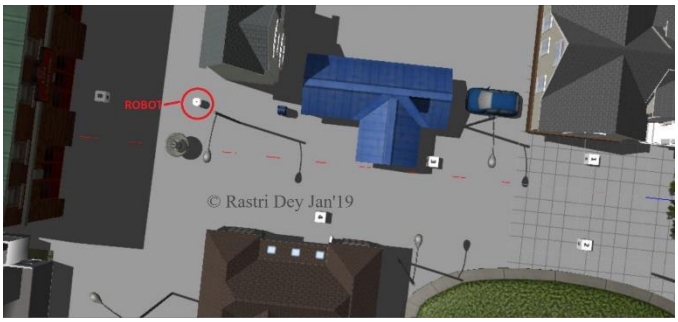


Fig. 15. Robot mapping in Door_To_Door_Service.world in Gazebo

A top view of the mapping results in Rviz environment can be represented as below:

Fig. 16. Robot mapping in Door_To_Door_Service.world in Rviz [TOP VIEW]

It should be noted that, the robot maps all the houses in the area staying indoor to the world i.e., from their front end only. This prevents the creation of a large world, thereby reducing the memory of rtabmap database while saving time and fulfilling the application it has been built in for.

The 2D map creation of the custom world looks like:

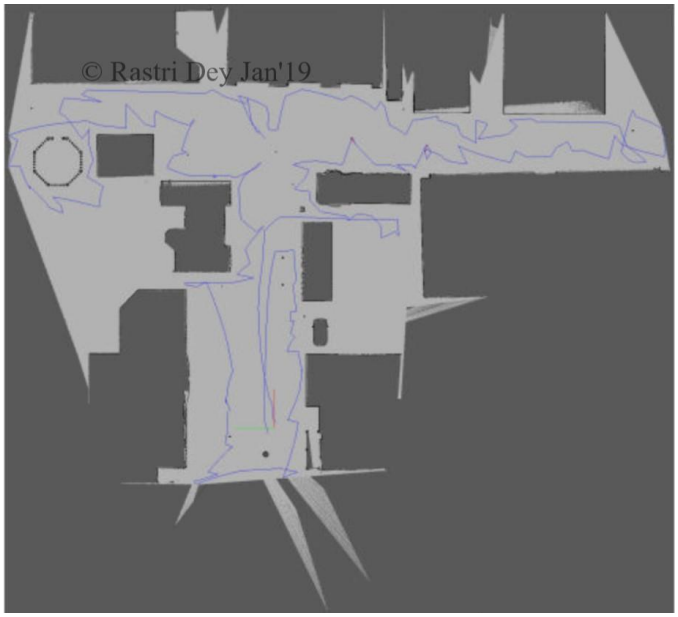


Fig. 17. 2D Map of Door_To_Door_Service.world

The obtained Occupancy Grid map of Door_To_Door_Service.world can be shown as:

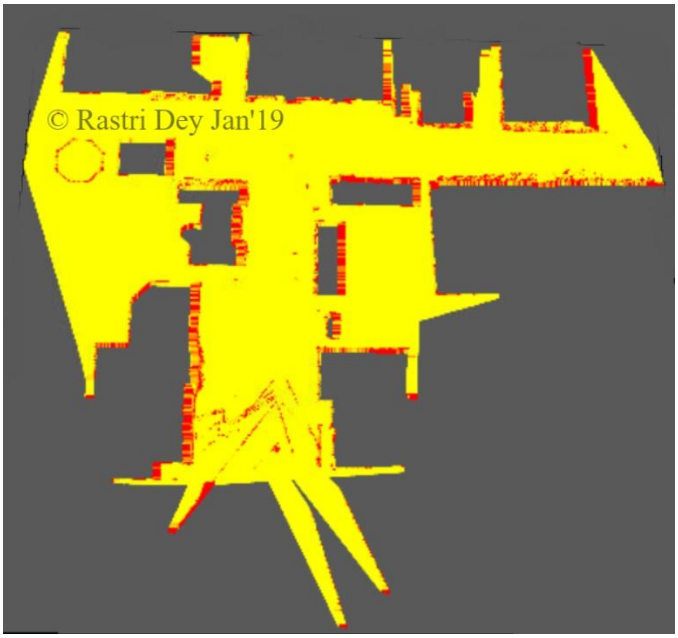
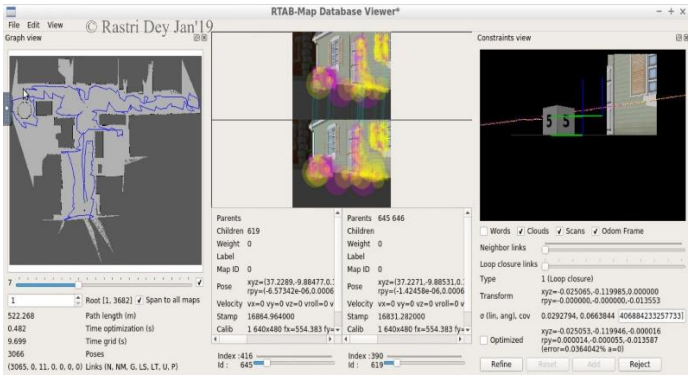
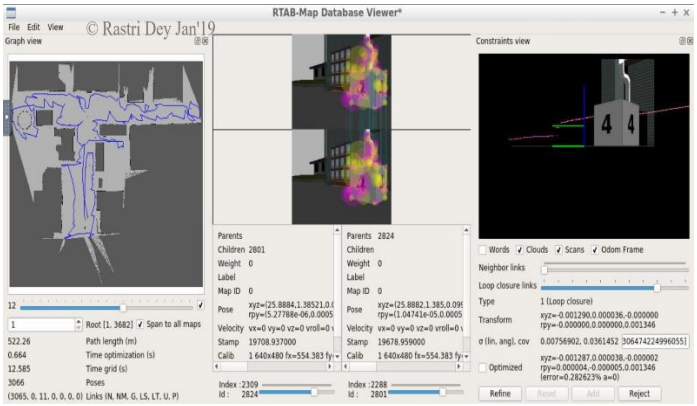


Fig. 18. Occupancy Grid Map of Door_To_Door_Service.world

The resulting loop closures and neighbor are 11 and 3065 respectively. The minimum visual inliers to accept the loop closures (param name="Vis/MinInliers") is set to a high value-80 to avoid linking of similar textured different objects as one. The global loop closures are obtained from the robot moving a multiple number of times around the same area. The sample loop closures of mapping the custom world can be illustrated from the following figures:





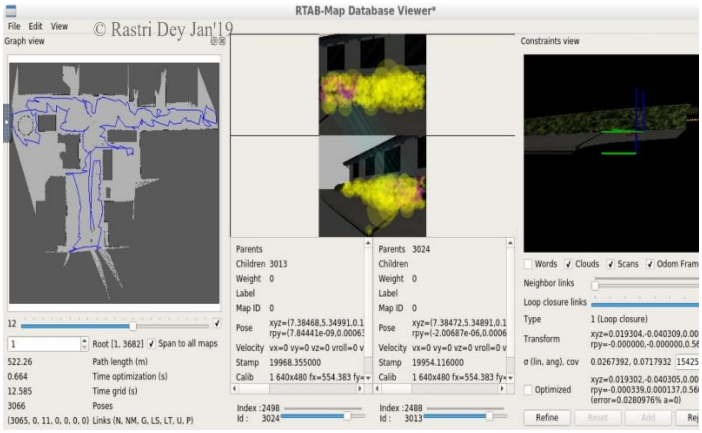


Fig. 19. 3 sample Loop Closure of Door_To_Door_Service.world [G=11]

The 3D Map representation in RTAB-Map database viewer generates a point cloud of the robot explored 3D world. This can be shown as:

Fig. 20. 3D Map of Door_To_Door_Service.world [TOP VIEW]

The voxel size while generating the dense 3D point cloud is 0.05 m. A closer look to the 3D map gives a clear picture of the individual areas that are mapped by the robot. This can be shown as:

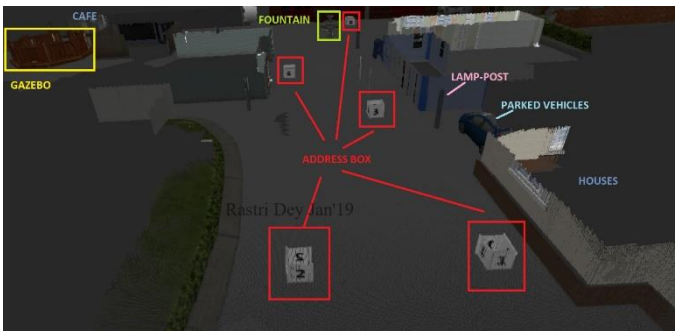


Fig. 21. 3D Map insights of Door_To_Door_Service.world

Fig. 21 and Fig. 6, can be compared for the exact depiction of simulated world objects in mapped 3D point cloud representation.

It can be inferred that the robot could make out a clear representation of the entire custom made world successfully. Though due to the sensor layout on the robotic body at a certain height and sensor elevation angles, mapping has not been made possible in the entire z-axis, leading to a truncated map development in its vertical dimension and missing out of information in 2D maps.

V. Discussion

This report discusses one of the most intriguing subject of Simultaneous Mapping and Localization through two case studies of robotic mapping under different environmental conditions. A robot has been built under urdf package with gazebo plugins for robot body and sensor layout. The robot is further integrated in a simulated world of Gazebo built-in database models and evaluated with the SLAM algorithm, RTAB-Map. The tuning of RTAB-Map parameters are done separately for both the worlds. For the first world of kitchen_dining.world, the minimum visual inliers to allow loop closure has been given a value of 40 to satisfy the proper loop closures without any misidentification of objects. While for the second world of Door_To_Door_Service, the minimum visual inliers is set to 80, after rigorous testing to obtain a successful map. A higher value helps in reducing false positives producing unusable maps. Increase in this value is although having an impact on faster loop closure with a slightly deprecated mapping performance. The loop closure for both the worlds results in a value of more than 3, 83 for kitchen_dining.world & 11 for Door_To_Door_Service.world. The custom world constructed is larger in dimension than the provided world, hence the rtabmap database produces a large amount of data for the second world. Owing to a larger environment, the time consumption is more in the mapping of second world in comparison to the first world. Yet, in a real world, a robot is never limited to a certain area and hence proper optimization for robot to map a high scale area for more span of time is relatively more useful. Moreover, the scale of a feature-rich environment is another challenge that has been covered in evaluating the mapping performance of the presented case-studies. The custom made world varies in every respect with the provided world, this enhances the robot's capabilities in wider horizons. In real world environment, this measures the robot's abilities in mapping both the indoor and outdoor worlds successfully. The sensor mounting in the robot body utilized in this project has restricted the algorithm in building a complete 3D map of either of the worlds. The 2D map and Occupancy Grid generated is restricted to the robot's sensor mounting height for both the worlds. The robot uses camera and laser sensors with the specifications of their respective azimuth and elevation profiles provided in the .gazebo files. The 3D maps obtained from the visualization tool of RTAB Map produces point clouds from the RGB-D camera at a very accurate depiction of the actual 3D world for both the

case studies. The tool further shows the trajectory of robot and loop detection as the robot explores the world.

VI. Conclusion & Future Work

3D mapping and robot localization are highly valuable in the field of autonomous robotics. A SLAM algorithm, RTAB Map is used in this project which utilizes bag of words approach to map the surrounding using vision sensors and attempts for loop closure, to identify recurring objects. The robot built in, successfully maps the constructed worlds while generating dense 3D point clouds and occupancy grid maps. An attempt to solve the door to door parcel delivery application through mapping and robot localization is though approached, not completely implemented due to limited resources and time constraints. A future work would be to implement the problem statement presented in this report and completing the entire application of a robot delivering parcels in a real world environment through SLAM based approaches. There is room for improvement in mapping as well. The maps produced in this project is restricted to a certain height, this limitation can be completely mitigated if instead of a wheeled robot on ground, a flying Drone is used for mapping the entire city environment. A drone could be programmed as per its sensor elevation and azimuth and tuned for the range required for it to elevate up and down to a get a complete 3D map image. Furthermore, RTAB map is limited to work on static environment and the robot cannot localize itself if dynamic objects are captured through the sensors at different points of time. This may engender false positives and lead to distortion in maps. A more powerful algorithm with better precision can tackle such problems and prove in mapping diverse environments even under low feature-rich surroundings. Hence a better SLAM algorithm is highly appreciable for mobile robot mapping and localizing under challenging circumstances. Finally, the involvement of Neural Networks into localization and mapping could result in learned robots facilitated with wider scope of applications. The robot's ability to localize itself in diverse dynamic environment while maintaining its stability underlying a robust algorithm is a key area of research for future enhancements of automation in robotics technology.

VII. Referneces

[1] http://wiki.ros.org/rtabmap