Building Map

In the previous tutorial, you have learned how to create an environment in the gazebo and also learned how to launch it. In this tutorial, you will learn how to **build a map of the environment** and **autonomous navigation of robot model** inside the RViz and gazebo environment.

1. Launch the **gazebo** with your environment by the following command:

   roslaunch task_1 simple_robot_gazebo.launch

2. Run the following command to launch the **RViz** so that we can see the map process:

   roslaunch task_1 rviz_navigation.launch

3. Run "**gmapping package**" to build environment.

   Command to run gmapping node is given below:

   rosrun gmapping slam_gmapping scan:=scan1

   scan1 is a topic on which "**laser scan"** data is published by the **gazebo**. This data can be used to build the map of the world environment.

   The output in RViz is shown in Figure 1.1.

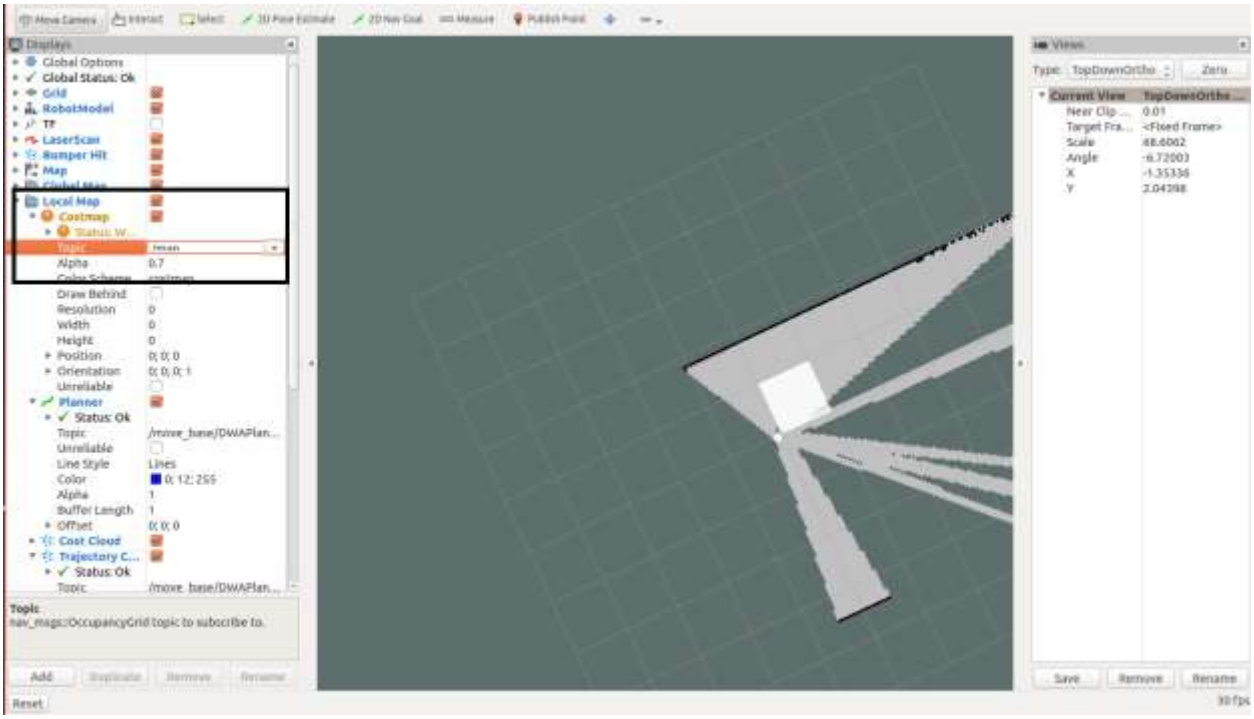

Figure 1.1 RViz map building

4. Change the option.

   Local map -> Costmap -> Topic (choose /map from drop-down list). See Figure 1.1.

5. Drive robot inside the world to build map.
   To build the map of the whole environment, you have to move robot inside the world. Launch the "robot_teleop" node by the following command to move robot inside world:

   roslaunch task_1 simple_tele_top.launch

   Now you can move the robot to build the map using the keyboard.

6. Save the map using "***map_server***"

   "map_server" is a ROS node that reads a map from disk and offers it via a ROS service. Run the following command in another terminal to save your map from RViz:

   rosrun map_server map_server –f /path_of_directory/map_file_name

   For example:

   rosrun map_server map_server –f ~/catkin_ws/src/task_1/maps/sample_map

   You can navigate to ***"maps"*** folder inside ***"task_1"*** package to check the map file. This folder contains two files with names *<map_file_name>.pgm and <map_file_name>.yaml.*

   Now that you have created and saved the map, you can interrupt (Ctrl + C) in each terminal window to close the processes.

Autonomous navigation in fixed map

Navigation stack and move_base are very important concepts. Make sure that you are thorough in these concepts.

1. **Navigation stack**

The main aim of the ROS navigation package is to move a robot from the start position to the goal position, without making any collision with the environment. The ROS Navigation package

comes with an implementation of several navigation related algorithms which can easily help im plement autonomous navigation in the mobile robots.

The user only needs to feed the goal position of the robot and the robot **odometry data** from sensors such as wheel encoders, IMU, laser scanner. The **output** of the **Navigation package are the velocity commands** which drive the robot to the given goal position.

The Navigation stack contains the implementation of the standard algorithms, such as SLAM , A *(star), Dijkstra, AMCL, and so on, which can directly be used in our application.

IMPORTANT: Visit the link here and learn the details of the navigation stack.

2. move_base

The move_base package provides an implementation of an action that, **given a goal in th e world**, will attempt to reach it with a mobile base. The **move_base** node links together **a global and local planner** to accomplish its global navigation task.

move_base is a **very important** package for navigation. To understand **move_base** visit link here.

3. Launch gazebo with an environment using following command:

roslaunch task_1 simple_robot_gazebo.launch

NOTE: Make sure you had changed the path of world file inside the "simple_robot_gazebo.launch"

In this tutorial, we are using "sample_testing.world" to explain navigation.

4. Launch RViz

roslaunch task_1 rviz_navigation.launch

5. Load map to RViz by the following command

roslaunch task_1 move_base.launch

You can navigate to launch folder in task_1 package to edit the launch files. The content of move_base.launch is shown below:

```
<launch>

  <!-- Map server -->

  <node name="map_server" pkg="map_server" type="map_server" args="$(find
task_1)/maps/sample_map.yaml" />

  <include file="$(find task_1)/launch/includes/move_base.launch.xml">
  </include>
```

```
<node pkg="tf" type="static_transform_publisher" name="odom_map_broadcaster" args="0 0 0 0 0 0 /map /odom 100" />

</launch>
```

This launch file will load the **sample_map.yaml** to RViz using the path highlighted in **box**.

IMPORTANT: Open **sample_map.yaml** file and check the path of **sample_map.png.** Make sure path given in **sample_map.yaml** file is valid.

After launching all the commands mentioned above, the RViz will look like Figure 1.2.

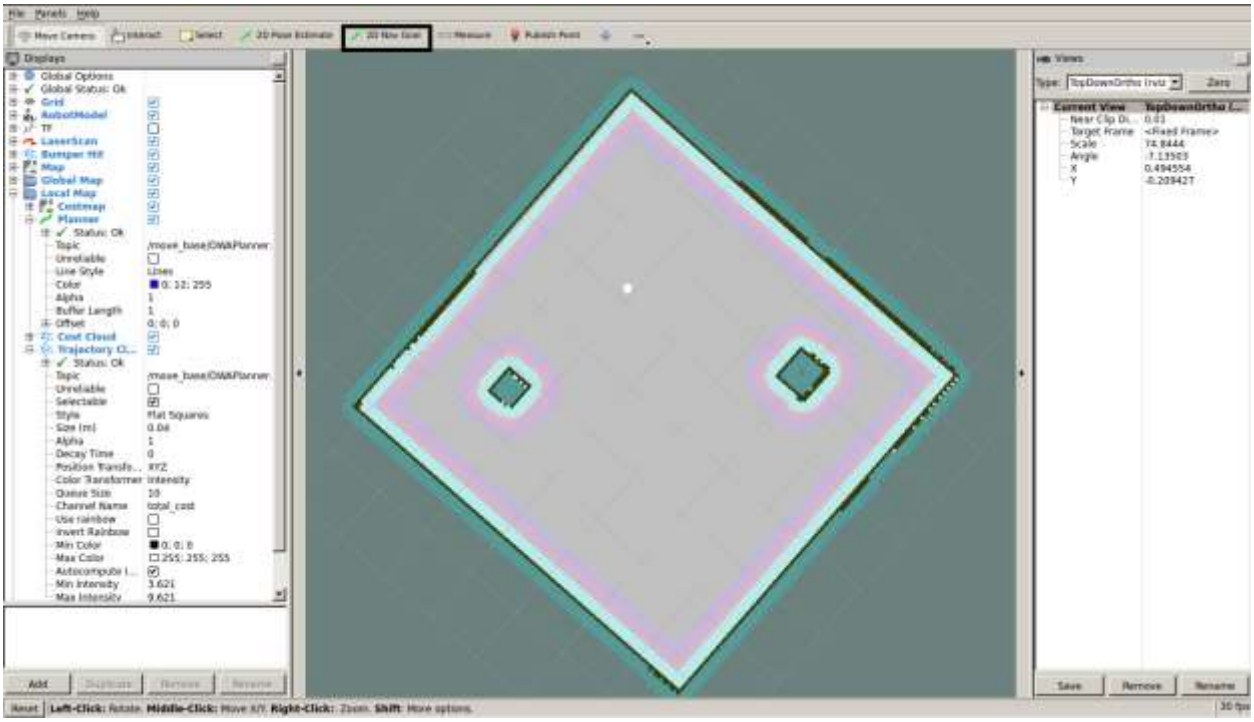

Figure 1.2 RViz with map

6. Send a navigation goal. Click the 2D Nav Goal button as shown in Figure 1.2. Click on the map where you want the robot to drive and drag in the direction the robot should be pointing at the end.

You have successfully learnt the concept of navigation using map.