ROS: Mapping and Navigation

The objective of this lab is to learn how to use ROS navigation stack for mapping an area with the robot and then using path-planning for navigating the robot on the map.

Setting up your computer

Open terminal window.

Clone your git repository to your home folder:

```
$ git clone <URL for your personal repository <yourname>-rtech>
```

Use `ls` to confirm that ***<yourname>-rtech*** has been downloaded to your home folder.

Gazebo, RViz, Mapping, and Navigation

In this lab you are going to:

1. Set up TurtleBot3 mobile robot and its simulation.
2. Bring up TurtleBot3 in Gazebo simulation environment.
3. Learn how Gazebo and ROS work together by visualizing data from simulated world in RViz.
4. Use TurtleBot3 to map a given area.
5. Navigate the robot based on the previously saved map.

1 To demonstrate advanced robotics capabilities, we are going to use TurtleBot3 robot in a simulated world. First you need to install some packages:

```
$ sudo apt-get install ros-kinetic-joy ros-kinetic-teleop-twist-joy
ros-kinetic-teleop-twist-keyboard ros-kinetic-laser-proc
ros-kinetic-rgbd-launch ros-kinetic-depthimage-to-laserscan
ros-kinetic-rosserial-arduino ros-kinetic-rosserial-python
ros-kinetic-rosserial-server ros-kinetic-rosserial-client
ros-kinetic-rosserial-msgs ros-kinetic-amcl ros-kinetic-map-server
ros-kinetic-move-base ros-kinetic-urdf ros-kinetic-xacro
ros-kinetic-compressed-image-transport ros-kinetic-rqt-image-view
ros-kinetic-gmapping ros-kinetic-navigation ros-kinetic-interactive-markers
```

Next, move to your workspace/src and clone three TurtleBot3 repositories. **Today, we add the TurtleBot3 repositories to our own repository as submodules.** It is a way to properly link an existing git repository within your repository.

```
$ git submodule add https://github.com/ROBOTIS-GIT/turtlebot3.git
$ git submodule add https://github.com/ROBOTIS-GIT/turtlebot3_msgs.git
$ git submodule add https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git
```

Build your catkin workspace.

2 Now let us bring up Gazebo simulation environment with TurtleBot3 robot. You need to specify your TurtleBot3 model as currently there are 3 different models available: *waffle*, *waffle_pi*, and *burger*. In order to set up *waffle* as your model and subsequently bring up the Gazebo simulation, execute the following commands.

```
$ export TURTLEBOT3_MODEL=waffle
$ roslaunch turtlebot3_gazebo turtlebot3_world.launch
```

Note that Gazebo may update its model database when it is started for the first time. This may take a few minutes. After a successful launch, the opening window should look like in Figure 1.

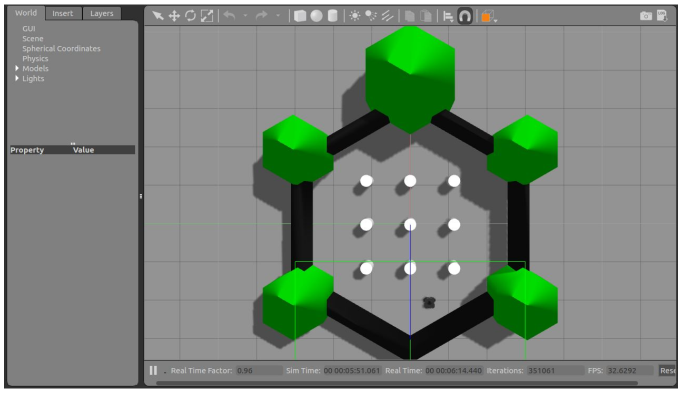


Figure 1. Gazebo window with some basic objects and TurtleBot3.

Next, let's explore the Gazebo world and see how ROS regards simulated worlds. We can use keyboard-based teleoperation for moving the TurtleBot3. Open a new terminal window.

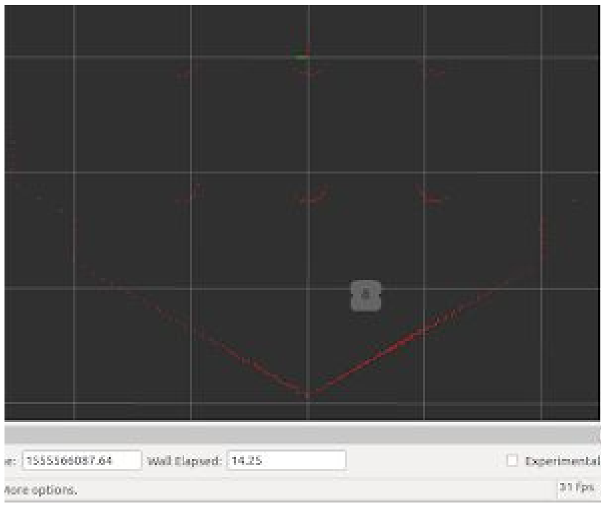
```
$ export TURTLEBOT3_MODEL=waffle
$ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch
```

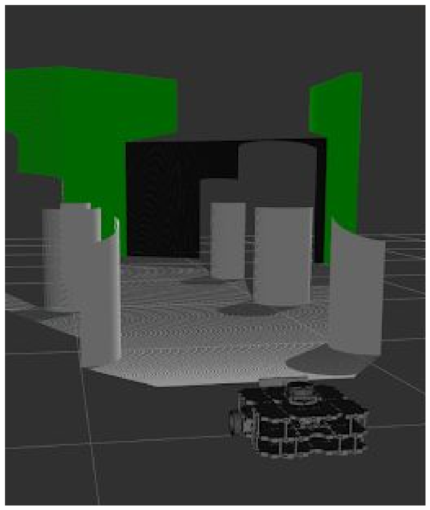
Figure out the keyboard controls for driving the TurtleBot3 and try moving the robot within its playpen.

3 Robots use cameras and laser sensors to perceive its immediate surroundings. In ROS we use RViz to visualize various sensory information, execute the following in a new terminal.

```
$ export TURTLEBOT3_MODEL=waffle
$ roslaunch turtlebot3_gazebo turtlebot3_gazebo_rviz.launch
```

Play around by enabling/disabling the visualization of different sensory information in RViz, move the TurtleBot3 around, and identify what you can see in RViz.





(a) (b)

Figure 2. LIDAR (a) and pointcloud (b) data visualization in RViz.

Shut down all the nodes.

4 Now you are ready to test out mapping and navigating a known map.

1) Launch the Gazebo world-file in order to simulate the TurtleBot3 in the environment.
2) Launch the SLAM in order to map the environment using *gmapping* algorithm:
```
$ export TURTLEBOT3_MODEL=waffle
$ roslaunch turtlebot3_slam turtlebot3_slam.launch slam_methods:=gmapping
```
3) Try to move the TurtleBot3 around the simulated world using your keyboard. Observe the map generation in RViz and create a complete map.
4) When the map appears complete, run the map server to save your map using the following command (it will save the map in your working directory under the name *mymap*):
```
$ rosrun map_server map_saver -f mymap
```

After you have saved the map, shut down all the nodes.

5 Now that you are done with mapping the environment, it is time to set some navigation goals for your TurtleBot3.

1) Launch the Gazebo world once again.
2) Launch the TurtleBot3 navigation launch-file and feed it with the map you previously generated:
```
$ export TURTLEBOT3_MODEL=waffle
$ roslaunch turtlebot3_navigation turtlebot3_navigation.launch
map_file:=$HOME/your_catkin_ws/src/mymap.yaml
```
3) Verify if, in RViz, the robot is located approximately at the same position in your map as it is in Gazebo. If not, try to put it roughly in the same position using *2D Pose Estimate* button in RViz.
4) Now, you can use the *2D Nav Goal* button in RViz to set navigation goals for your TurtleBot3 on the map.

>>> Show the final result to your lab instructor! <<<

Clean up your workspace

NB! Before you leave the lab, make sure you have pushed all the files in your catkin workspace to your git cloud service.

In terminal, `cd` to **<yourname>-rtech**

Type

```
git config user.email "youremail@example.com"
```

Type

```
git status
```

You should now see all the new and modified files in red.

Prepare the relevant files for the commit.

```
git add file_name_in_red1 file_name_in_red2
```

When you now type

```
git status
```

you should see all the added files in green. You are now ready to commit changes. Type

```
git commit –m “Insert a brief explanation”
```

Your changes have now been committed but not yet uploaded to the cloud. To upload your files, type

```
git push
```

In your web browser, **verify that all the files** have been uploaded to the **<yourname>-rtech** repository.

Delete the **<yourname>-rtech** folder and any other files you created from the lab's computer.