ROS2 Exercise 5: SLAM

Learning outcome

This exercise will help students to get familiar with localization, mapping and navigation for mobile robots. You will create a virtual environment and map it with a mobile robot and lidar sensor. Then, navigation in this map will be done.

Grading

- Grade 3: tasks 1 + 2 + 3, successful demo and could not answer all questions
- Grade 4: tasks 1 + 2 + 3, successful demo and could answer main questions
- Grade 5: tasks 1 + 2 + 3 + 4, successful demo and could answer main questions

Exercises:

Task 1: TurtleBot simulation

The TurtleBot package as used in Exercise 3 is the starting point. To recall:

1. Simulation of TurtleBot and tele-operation
   https://emanual.robotis.com/docs/en/platform/turtlebot3/simulation/#gazebo-simulation (switch to ROS Humble version)
2. Instead of using an existing world, build a custom world in Gazebo
   Your world needs to contain several rooms and objects within them.
   Be creative and build a challenging world to map
3. Save the world, so that it can be launched directly from command line when initializing Gazebo

Task 2: SLAM

The objective of this task is to map the world you created with turtlebot and the laser scanner, by following the explanation here: https://emanual.robotis.com/docs/en/platform/turtlebot3/slam_simulation/ (switch to ROS Humble version)

1. Run the SLAM node with chartographer as described in 6.2.2
2. Visualize all relevant topics in RVIZ while mapping. These include at least the TurtleBot, the map and the laserscan. Explain what you see.
3. Save the map and explain the contents of the created files.
4. Chartographer SLAM method has many parameters, as defined in the configuration file:

   ```
   /opt/ros/humble/share/turtlebot3_cartographer/config/turtlebot3_lds_2d.lua
   ```

5. Demonstrate that the limitations in sensor capabilities affect the mapping performance. This is possible by change the range of lidar sensor.

   Reference : https://google-cartographer-ros.readthedocs.io/en/latest/algo_walkthrough.html

Task 3: Navigation

The objective of this task is to autonomously navigate the mapped world you have created, by following the explanation here: https://emanual.robotis.com/docs/en/platform/turtlebot3/nav_simulation/
(switch to ROS Humble version)

1. Run the navigation node with your created map
   - **For issues such as map not appearing in rviz refer to fixes given in 3.10**

2. Follow the correct steps to include an initial pose estimate via RVIZ
3. Refine the localization by tele-operation of the TurtleBot
4. Demonstrate the correct navigation of the TurtleBot to challenging goals in the world.

Navigation utilizes a global planner to define a path and a local planner to avoid obstacles while following the path.

5. Find out all sensors that the TurtleBot is using to navigate the map
6. When navigation is running, several paths are projected forward of the TurtleBot, in different colors. Explain what these are.
7. The navigation stack has many parameters to change performance of navigation, based on the used robot. These parameters can be changed and are located in:

```
/opt/ros/humble/share/turtlebot3_navigation2/param/burger.yaml
```

8. How do cost values (**cost_scaling_factor**) affect the shape of the path to navigate
9. How does the obstacle radius (**inflation_radius**) affect the (local and global) navigation?

Use the configuration file (`burger2.yaml`) provided with the handout if the map does not appear in rviz2.

10. Open a fresh terminal and run the below commands to copy the burger2.yml to the respective directory. Remember to replace `{file_path}` with correct path where you downloaded the file.

```
$ sudo cp /{file_path}/burger2.yaml \
/opt/ros/humble/share/turtlebot3_navigation2/param
```

11. Launch navigation again as in the step 6.3.2 of the tutorial

```
$ export TURTLEBOT3_MODEL=burger2
$ ros2 launch turtlebot3_navigation2 navigation2.launch.py\
use_sim_time:=True map:=$HOME/map.yaml
```

References:

- https://navigation.ros.org/tuning/index.html
- https://navigation.ros.org/configuration/packages/configuring-costmaps.html
- http://kaiyuzheng.me/documents/navguide.pdf
- https://google-cartographer-ros.readthedocs.io/en/latest/index.html
- https://navigation.ros.org/configuration/packages/configuring-costmaps.html

Task 4: Automated mapping

1. This tutorial follows Cartographer SLAM method. Name a few other existing SLAM methods? Compare them with Cartographer.

2. Mapping has now been done by tele-operating the robot. Develop an automatic approach that maps the environment without tele-operation. Robot motion generation from Ex. 3 can be used to explore the environment automatically. What criteria are important to evaluate whether mapping is progressing correctly or is completed?

To arrange a demo contact: Eetu (eetu.airaksinen@tuni.fi)