TÉCNICO LISBOA

INTRODUÇÃO À ROBÓTICA / INTRODUCTION TO ROBOTICS

2024/2025

Mini-Project 2

Hand-out: October 2024

Objective

The objective of this assignment is to provide students with the opportunity to get familiarized with the practical aspects of mobile robot path planning and path following. For this purpose, students must get acquainted with some of the available ROS packages for navigation (particularly the navigation stack), learn how to use them, and demonstrate the ability to explain formally their working principle. The project is focused on the use of Rapidly Exploring Random Trees (RRT) algorithms for path planning.

Procedure

The work will be implemented both in simulation and on a TurtleBot3 Waffle Pi[1] mobile robot. The robot has an onboard laser scanner, used to acquire the environment map and to self-localize (done in Mini-Project 1), as well as to detect unexpected obstacles, and a Raspberry Pi processor where all ROS drivers are running.

The followings steps 1 to 3 involve using the TurtleBot3 Gazebo simulation environment. Steps 4 and 5 involve running experiments with a real robot, using an external computer that communicates with the onboard ROS master using WiFi.

1. Run a path planning/path following simulation in Gazebo, following the instructions in [2] and using rviz to set the estimate for the current robot position and the goal position. Explore the launch files `turtlebot3_navigation.launch` and `move_base.launch` and get acquainted with the `move_base` package and the parameter files (`.yaml`) to understand the components that are responsible for planning the path (global planner) and generating velocity commands (local planner) and their interplay with other nodes already used in the previous project (`amcl` and `map_server`)[3].

[1] http://www.robotis.us/turtlebot-3-waffle-pi/

[2] https://emanual.robotis.com/docs/en/platform/turtlebot3/nav_simulation/

[3] http://wiki.ros.org/navigation/Tutorials/SendingSimpleGoals

2. Substitute the default global path planner by a new one that implements an RRT algorithm (write a global path planner as a plugin in ROS)[4]. Skeleton code for this plugin is provided in the repository https://github.com/irob-labs-ist/rrt_planner. To detect possible collisions with obstacles when planning the path, resort to the `costmap_2d` ROS package [5].
3. Run several simulations to fine tune the parameters of `move_base` and improve the performance of the system.
4. Considering now the real TurtleBot3 Waffle Pi Robot and the map obtained in the previous assignment, write a simple program that interacts with `move_base` to plan and follow a path that visits four waypoints similar to those represented in Figure 1[6].

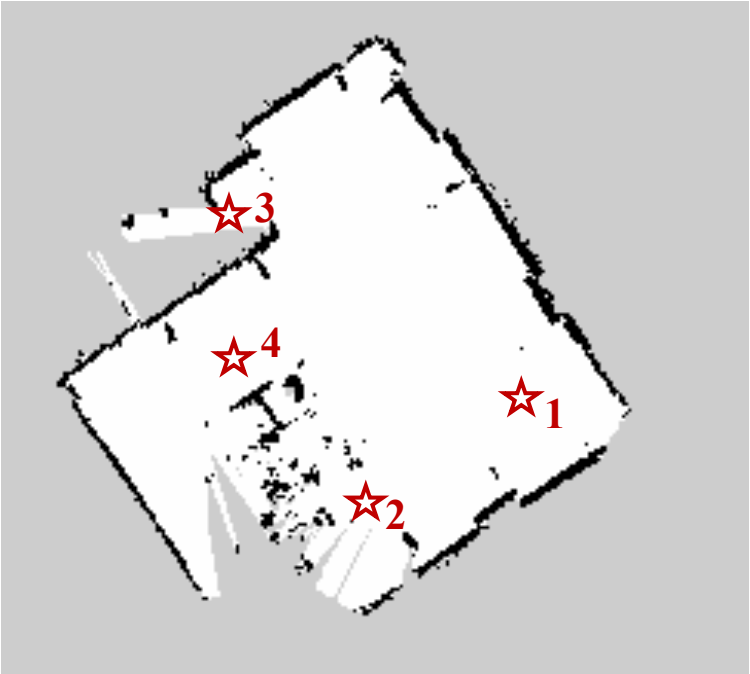

Figure 1. – Map of the LSCD4 lab and the set of four waypoints that the robot has to visit.

5. Repeat 4. but add unexpected (unmapped) obstacles to the path to be traversed by the robot. Modify/configure `move_base` to adequately avoid those obstacles, using the laser scanner to detect them.

Expected results

The following list represents the minimal set of results to be reported:

- planned paths – in RVIZ;
- actual robot path (estimated by the robot) – in RVIZ;
- comparative analysis of the planned paths based on a set of relevant metrics.

---

[4] http://wiki.ros.org/navigation/Tutorials/Writing%20A%20Global%20Path%20Planner%20As%20Plugin%20in%20ROS

[5] http://wiki.ros.org/costmap_2d

[6] http://wiki.ros.org/navigation/Tutorials/SendingSimpleGoals