ROS and experimental robotics

Juyeon KIM
Master SAR
Student number 28611849

Theo MAGOUDI
Master SAR
Student number 28606508

***Abstract*—This is a technical report on our first ROS project on simulation and demonstration of Turtlebot 3 Burger robot. In this document, we will briefly introduce the missions of the robot, explain the ROS architecture and give an analysis of the performance of our architecture.**

I. INTRODUCTION

The main objective of this project is to control a Turtlebot 3 burger mobile robot in a simulated and real environment. The goal of this project is to develop navigation skills and test the robot's sensorimotor capabilities by making it navigate from a starting position to a goal position while solving different tasks on the path.

The project consists of three challenges that will progressively exploit the robot's sensors, including images obtained from a simulated(/camera) and a real camera (/usb_cam) to detect and follow lines, and laser scans obtained from a simulated/real LDS to detect and avoid obstacles.

After one month of work, this report aims to present the different codes, architectures used and the different successes and failures of the project.

II. PRESENTATION OF THE ROS ARCHITECTURE

A. A short overview

Our architecture mainly has one node for each challenge, topics such as /cmd_vel and /scan, and Twist, LaserScan messages.

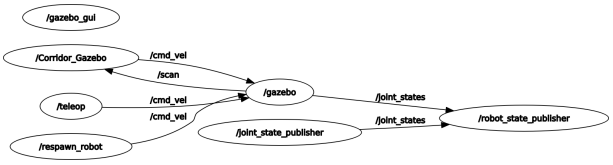

Fig. 1. Rqt graph of Corridor_gazebo node.

B. Algorithmic structure of the architecture

In this subsection, we detail the functionalities of each node. Here we used separate launch files for each challenge.

- Node Lane_following subscribed to Image topic, uses those images to compute the orientation of the robot by using the center of the two lines; a condition is applied if the robot can only see one of the two lines. If the orientation matches the center of the robot, it will send a Twist message to command a linear velocity. If the orientation does not matches
- Node Lane_following controls the robot well and brings it to the end without leaving the path, even if it slightly touches the line. For this, we need to find a middle-ground between speed and precision through optimization. We chose to have a fast linear speed in the straight lanes, and a very low linear speed in the curves, i.e. we wanted to maximize the precision, not the speed.
- Node Corridor_Gazebo subscribed to LaserScan computes the distances to the nearest obstacles from the left and right sides of the robot. The robot initially follows along the left wall and changes direction to the right once it is too close to the left wall. This node sends a Twist message to command the linear and angular velocity of the robot. At the end of the corridor where the robot does not detect anything on its right, left and front, the robot stops approximately after the red line indicating the completion of the mission.

III. PERFORMANCE CHARACTERIZATION

The lane following node controls the robot well and brings it to the end without leaving the path, even if it slightly touches the line. For this, we need to find a middle-ground between speed and precision through optimization. We chose to have a fast linear speed in the straight lanes, and a very low linear speed in the curves, i.e. we wanted to maximize the precision, not the speed.

IV. CONCLUSION AND PERSPECTIVES

To conclude, we managed to do the basic work by adding some interesting features such as respawning the robot, but several aspects of the arhitecture can be developped furthermore.

We started to implement a red line detection on our code in order to use this information in another node, so called the controller node, to move onfrom one challenge to another.

For the challenge 3, the idea is to detect the green bottles and calculate its moments, move towards these goal posts, while detecting and avoiding obstacles and not touching the blue lines.

Finally, a controller node which plays the central role could be implemented that requests information about completion of each mission and activates the following mission.