Obstacle Avoidance project

Robot Programming

Leonardo Mongelli

2020024

Contents

1 Introduction

This report describes the Obstacle Avoidance project developed for the Robot Programming course. The entire project was done in C++ using the Robot Operating System(ROS) libraries and tools. The goal of the project is to build a node that computes a repulsive field from the local laser scans and modulates the velocity so that the robot does not clash with the wall if instructed to do so. Specifically two ROS topics have been used: **/base_scan** and **/cmd_vel**. The first one allows to read all laser scan data, while the second one is used to read and modify the robot's velocity.

2 How to compile

Before running and testing the developed code, you must create your own Catkin workspace on your machine. To do that, please follow these steps on the terminal:

```
$ mkdir my_project_workspace
$ cd my_project_workspace/
$ mkdir src
$ cd src/
$ catkin_init_workspace
$ cd ..
$ catkin build
```

After doing that, you can download my personal GitHub repository which contains the Obstacle Avoidance package with a CMakeLists.txt, a package.xml, and a /src folder in which the ROS node is located. Please, rename the package from *Obstacle-Avoidance-RP* to *my_obstacle_avoidance_package* and put it into the /src folder of your workspace. This is necessary to be coherent with the following steps. In fact, I created the ROS package and node in this way:

```
$ cd my_project_workspace/src/
$ catkin_create_pkg my_obstacle_avoidance_package roscpp sensor_msgs std_msgs
$ cd ..
$ catkin build
$ source devel/setup.bash
$ cd src/my_obstacle_avoidance_package/src/
$ emacs my_obstacle_avoidance_node.cpp
```

3 How to run and test

Once your workspace is correctly set up, you are ready to test the project. To do that, open a terminal and start the master by typing:

```
$ roscore
```

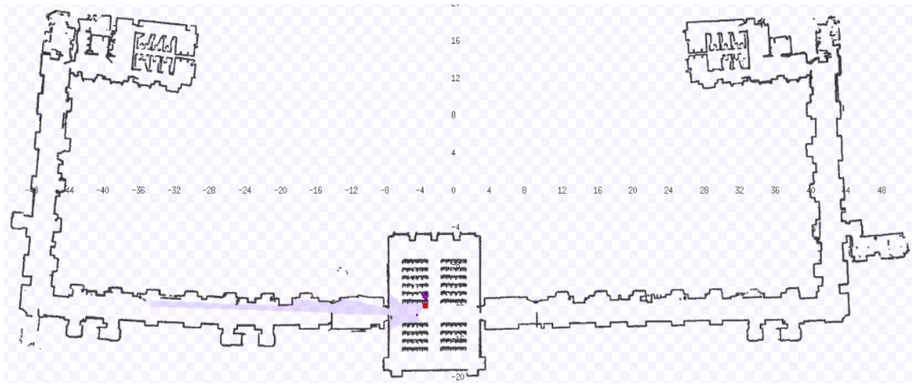
Then open another terminal and start the obstacle avoidance node:

```
$ cd my_project_workspace/
$ catkin build
$ source devel/setup.bash
$ rosrun my_obstacle_avoidance_package my_obstacle_avoidance_node
```

Again open a third terminal and start the simulation environment:

```
$ cd
$ cd Desktop/robotprogramming_2021_22-main/source/srrg2_workspace/-
  src/srrg2_configs/navigation_2d/
$ wsrp
$ rosrun stage_ros stageros\
  cappero_laser_odom_diag_obstacle_2020-05-06-16-26-03.world
```

Note that the second terminal line indicates the path where you have saved the map(.world). In this case, the simulation environment used for testing is the DIAG map provided during the Robot Programming course.

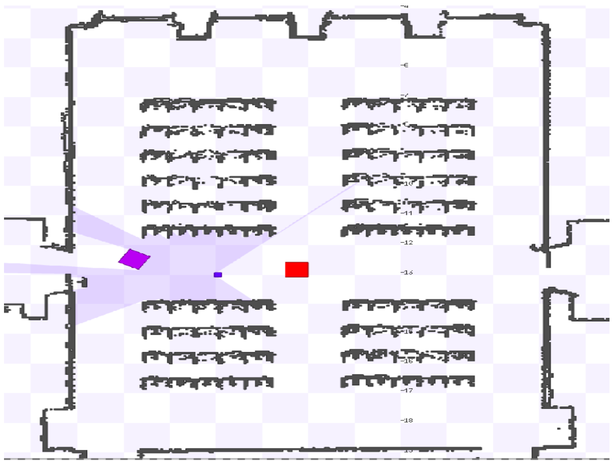


Instead, the third terminal line represents an alias in .bashrc for ”wsrp = source /Desktop/robotprogramming_2021_22-main/source/srrg2_workspace/devel/setup.bash”.

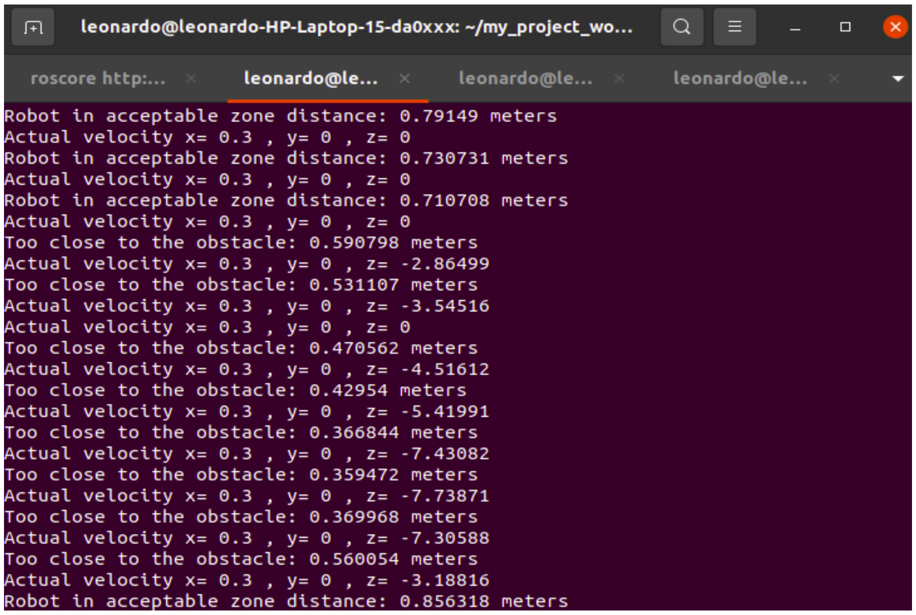
Finally, open another terminal and start the robot movement:

```
$ rostopic pub /cmd_vel geometry_msgs/Twist -r 1 -- '[0.3, 0.0, 0.0]' \
  '[0.0, 0.0, 0.0]'
```

After running this command, the robot represented by a little blue square starts moving around the map, avoiding the collision with both walls and the two obstacles represented by purple and red squares.



Simultaneously, in the second opened terminal, the one where the obstacle avoidance node was started, you can see the update of the robot's speed and the real-time distance from the closest obstacle. As you can see sometimes the closest obstacle is at an acceptable distance, so the robot's velocity remains unchanged, but other times it is too close and requires speed adjustment.



```
Robot in acceptable zone distance: 0.79149 meters
Actual velocity x= 0.3 , y= 0 , z= 0
Robot in acceptable zone distance: 0.730731 meters
Actual velocity x= 0.3 , y= 0 , z= 0
Robot in acceptable zone distance: 0.710708 meters
Actual velocity x= 0.3 , y= 0 , z= 0
Too close to the obstacle: 0.590798 meters
Actual velocity x= 0.3 , y= 0 , z= -2.86499
Too close to the obstacle: 0.531107 meters
Actual velocity x= 0.3 , y= 0 , z= -3.54516
Actual velocity x= 0.3 , y= 0 , z= 0
Too close to the obstacle: 0.470562 meters
Actual velocity x= 0.3 , y= 0 , z= -4.51612
Too close to the obstacle: 0.42954 meters
Actual velocity x= 0.3 , y= 0 , z= -5.41991
Too close to the obstacle: 0.366844 meters
Actual velocity x= 0.3 , y= 0 , z= -7.43082
Too close to the obstacle: 0.359472 meters
Actual velocity x= 0.3 , y= 0 , z= -7.73871
Too close to the obstacle: 0.369968 meters
Actual velocity x= 0.3 , y= 0 , z= -7.30588
Too close to the obstacle: 0.560054 meters
Actual velocity x= 0.3 , y= 0 , z= -3.18816
Robot in acceptable zone distance: 0.856318 meters
```