Homework 1

Miroljub Mihailovic, 2021445
Luigi Sarrocco, 2020634
Ghedin Roberto 2015135

December 2021

1 Introduction

The first goal of this homework is to make Tiago able to explore the world around him. The environment consists of two rooms with non-static obstacles, so it is possible to move them in Gazebo. To do this, an Action/Client system has been developed, able to read the request and give a response. The second goal is to detect obstacles around the room (based on the position of Tiago), exploiting the laser scan of the robot. During these weeks, we have tied different approaches for detecting the circle/cylindrical objects, among with Hough transform and Circle Hough transform. However, they did not work well, therefore we have opted for our implementation that we are going to explain in the following.

2 First part: Moving toward point B

2.1 Action Client

The Action client is the part that will make the request to the action server for a specific location in the map. This position is defined by the x, y coordinates (the origin (0,0) is on the robot starting position) and by the rotation. The communication between the two parts takes place via a specific file called *navigation.action*. Therefore, the client will be constantly notified by the server feedback so that it is continuously informed about the progress of the robot.

2.2 Action Server

The Action server receives the request made by the Action client ($Pose2D$) and try to send a *move base action* to the robot. While the robot is moving, the client is constantly informed about the progress using the feedback implementation. Once it has arrived at the destination, it starts to scan the environment in order to detect objects. In the following subsection, we are going to explain in detail our implementation.

3 Second part: Obstacles detection

3.1 Cloud point and transformation from base link to map

The laser scan topic is pretty useful to detect the obstacles surrounding Tiago.The dark blue lines represent the distance between the robot and the first obstacle following that direction. $CBScan$ function is applied in such a way to transform the laser data into cloud points. Since the points are collected exploiting the base link perspective, we need to transform back to the map frame the whole points found previously, and this is done using the function $listPoints$.

3.2 Removing static obstacles

The idea of removing static obstacles (e.g. walls) consist of erasing all the straight lines as it could be seen in figure 2. A first step cancel out only the vertical and horizontal lines. Subsequently, we erase also the points that belong to the oblique walls that are present in the room, exploiting their angular coefficient. The final list of points will be the list of positions of the non-static obstacles. Every computation has been done using some empirical value as threshold. Because of this, sometimes points of the same cluster are subdivided in 2 different clusters. A fast check is made and, if points of different cluster have a distance under a certain threshold, the two clusters are merged. Below are reported two plots: the first (figure 1) is a set of cloud point detected, and the second one (figure 2) is the result after applying the algorithm for removing lines where each dot represents the centre of the object detected.

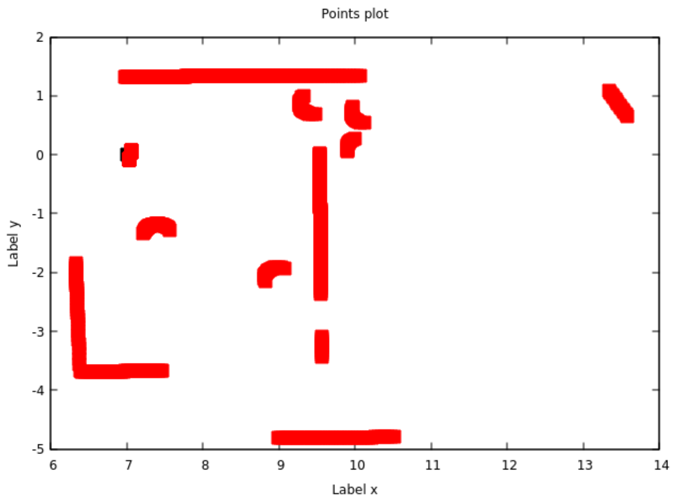

Figure 1: Cloud points from the robot view, the position of the robot is (7,0)

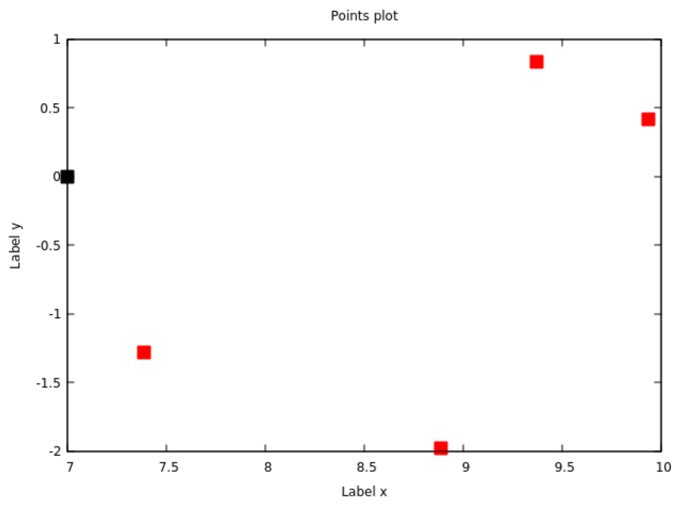

Figure 2: The red dots represent the centre of the objects' positions while the black one the current robot position

4 Extra points: new navigation routine

The navigation stack seems inefficient in narrow passages of the room. So we have implemented a new routine that works better in this context. The main idea to consist of finding only vertical lines, therefore we define a middle point and assign this as a goal. Once the first point is reached, it means that we are successfully passed the narrow zone, then we reassign the initial goal in order to reach the final position.

Starting from the initial point (when Gazebo is opened), our routine seems working good, but in the other cases it presents some errors that we were not able to fix in time.

5 Conclusion

Without considering the new navigation routine, all the tasks of the robot are computer correctly. The robot is able to detect the obstacles around the room and reach (when is possible) the goal location send by the action client.