Reactive Robot

Andreia Rodrigues
Faculdade de Engenharia da Universidade do Porto
Porto, Portugal
up201404691@fe.up.pt

Tiago Filipe
Faculdade de Engenharia da Universidade do Porto
Porto, Portugal
up201610655@fe.up.pt

***Abstract*— The evolution of technology provides a large increase in the use of robots for various tasks. With this comes the improvement of robot's navigation defined by many different behaviours. This paper describes the implementation of a robot with wall-following behaviour using ROS and STDR simulator package. The robot is tested in two different environments. One uses the letter V and the other one the letter W. The goal is to find the wall and then follow it forever, regardless if it's the outside or interior of the wall. These experiments are reported with results and limitations.**

Keywords*— reactive robot, ROS, STDR simulator, wall-following robot

I. Introduction

These days we are experiencing an evolution of technology where there is an increasing number of robots used for different kinds of tasks, such as repetitive or dangerous tasks, where it is extremely important that robots can be autonomous and adapt to the surrounding environment.

To work autonomously, it is fundamental that the robot is capable of autonomous navigation. Lots of efforts have been devoted to the study of sub-problems, such as path-planning [1], simultaneous localization and mapping (SLAM) [2], wall-following behaviour [3] and so on. Wall-following behaviour refers to moving along walls or the edge of objects and keeping a constant distance between the robot and the wall.

The purpose of this project is to design, implement and test a reactive robot, which has a wall-following behaviour, using ROS and STDR simulator. The robot is placed in two different environments explained in the architecture chapter.

II. Architecture

A. Robot Description

This project uses ROS [4] and the STDR simulator [5] package to implement the robot and its behaviour. The robot is disk-shaped and has a radius of 0.1m (Figure 1).

It uses laser range scanner (LIDAR) to detect and measure the environment with minimum range of 0.05 m and maximum range of 1.5 m. It is composed of a total of 200 rays, forming a 200º span around the robot. The laser information is updated with a 10 Hz frequency.

The robot's movement is characterized by a fixed linear speed of 0.4 m/s.

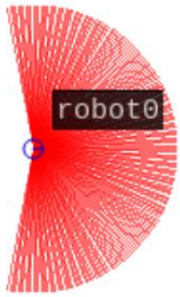

Figure 1 - Robot implemented in STDR

B. Environment

The robot is tested in two different environments, as can be seen in the following figures:

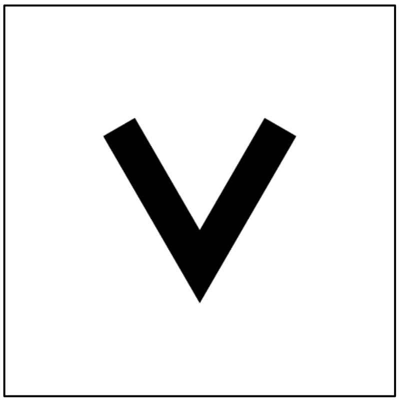

Figure 2 - V shaped map

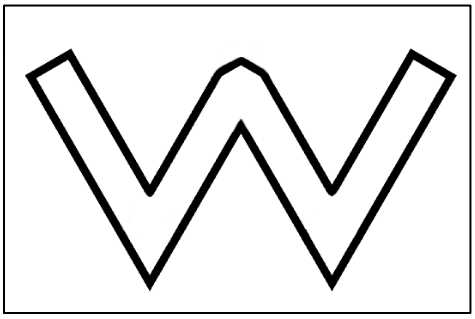

Figure 3 - W shaped map

Initially, in the V shaped map (Figure 2), the robot stands outside the letter and moves towards the wall. The V shaped map is a single filled section.

In this map the robot navigates with random behaviour until finds the wall, then follows the outside of this wall indefinitely. The dimension of this map is of 19m x 19m.

Regarding the W shaped map (Figure 3), the robot is initially positioned inside the non-filled letter.

In this map the concept is the same, but instead of following the outside of the wall, it follows the inside wall of the letter. The dimension of this map is of 36.5m x 24m.

Both maps have a resolution of 0.02m/px.

C. Algorithm

The goal of the developed robot is to follow an imaginary line at a certain distance from the wall, using a constant linear velocity throughout the path.

The angular velocity on the z axis is calculated with the following equation, present in [6]:

$$\omega = (a * (cos(\alpha) - (d - b)) * c \tag{1}$$

where a is a control parameter (set at 15 or -15, depending on the "side" variable, to be explained further on) that allows some stability on the robot's movement, α is the absolute angle set between the front of the robot and the laser ray with the lowest distance to the wall, d is the distance between the robot and the wall, b is the optimal distance to be kept by the robot in relation to the wall (set at 1.2m) and c is the robot's linear velocity on the x axis (set at 0.4 m/s).

The meaning of the expression symbols is resumed in table I.

TABLE I. EXPRESSION SYMBOLS

a	Control parameter for movement stability
α	Absolute angle between the front of the robot and the optimal laser ray
d	Distance from the wall
b	Optimal distance from the wall
c	Linear velocity

The robot laser rays determine the distance to the nearest obstacle within range. The laser was divided in two areas, "left" and "right", so we could determine what side of the robot touched the wall first and make it stick to the same side in its movement along the wall for the remaining path. Knowing that the total number of rays set for the laser was 200, and that they formed a 200º span around it, the logic to determine what side of the robot touched the wall first was simple.

The angle between each laser ray and the front of the robot can be calculated with the following equation:

$$\beta = -100 + \theta * i \tag{2}$$

where θ is the angular distance between the laser rays measurements, i the number of each ray (that goes from 0 to 200 as previously stated) and -100 is the control value so that, knowing that the angle increment is approximately of 1º between rays, β is 0º when the ray is right at the front of the robot. That being said, the "left" side is defined as where β is between -100º and 0º and the "right" side is when β is between 0º and 100º (Figure 4).

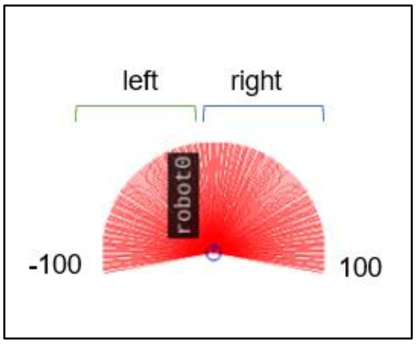

Figure 4 - Division of the laser rays

There is a global variable "side" that works like an enumeration where 0 corresponds to not having a side attributed, 1 to only using the sensors on the left and 2 to only using the sensors on the right. It is set on the first time the robot touches the wall.

This "side" global variable will not only determine what rays to have in account when comparing the distance to the wall but also what value the control parameter takes, to allow stability on the robot's movement according to what side of the laser is used to follow the wall.

Both test environments, the V and W shaped map, have two launchers each for testing the robot when it detects the wall using the "left" or the "right" side of the laser.

III. Results

The results are focused on the two goals of this project: the ability of the robot finding the wall and properly following it indefinitely and the stability of the wall-following algorithm.

Starting with the V shaped map, the robot reaches the wall after about 5 seconds, as you can see in Figure 5. When the X coordinate (pink line) begins to decrease, it means that it has reached the left part of the V and from that moment it begins to follow the wall. The X coordinate will only increase again when it goes around the corner of the V and moves towards the inner corner.

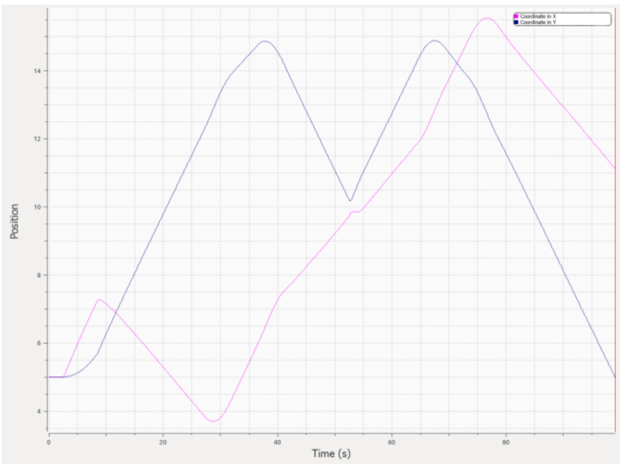

Figure 5 - Robot movement in V shaped map

About the stability of the algorithm, in Figure 6, the first 5 seconds were removed because the robot was too far from the wall and walking with a random navigation. During the time the robot was following the wall, the distance to the wall was between 0.84 and 1.5 meters, with the most predominant values being between 1.1 and 1.22. Even though the distance was not always equals to the optimal distance considered in the algorithm, the results were good and quite close to the optimal value.

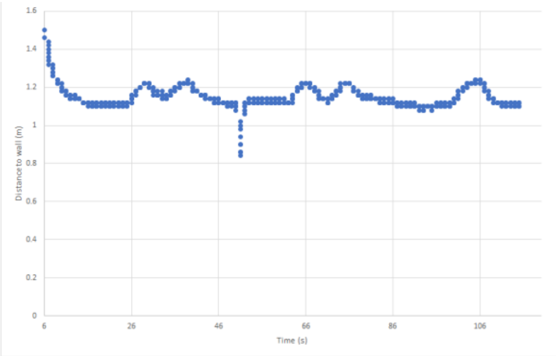

Figure 6 - Distance from robot to wall over time in V shaped map.

The Figure 7 represents the robot movement in the W shaped map for the first 100 seconds. As can be seen, in this environment it was easier for the robot to reach the wall.

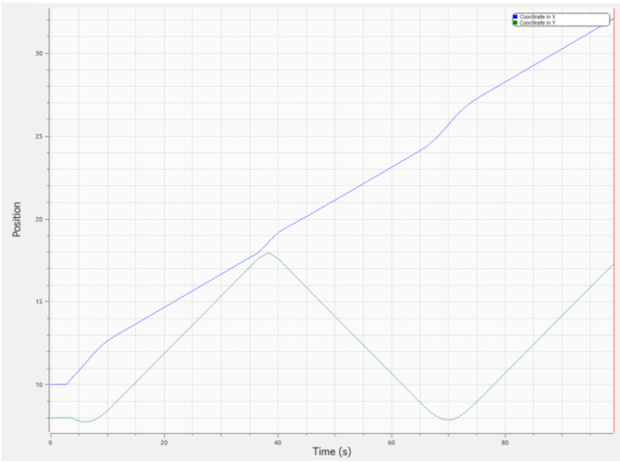

Figure 7 - Robot movement in W shaped map.

Regarding the distance to the wall, in Figure 8, the first second was removed because the distance was too far. In the scatter plot it is shown that the results were between the same range as the previous environment, but were more constant.

On the other hand the optimum value was achieved less often, because the letter of this map provides more changes of direction.

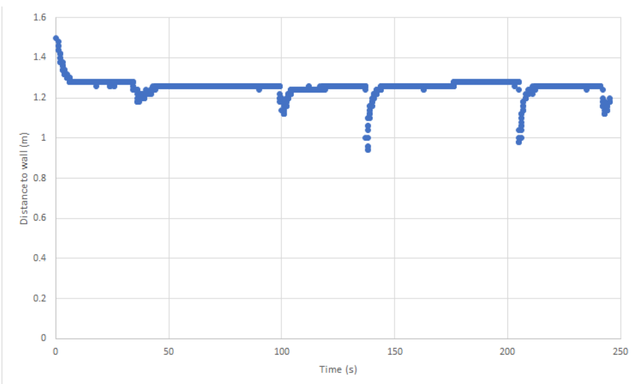

Figure 8 - Distance from robot to wall over time in W shaped map.

IV. Limitations

One of the limitations of STDR package is the robot's interaction in the same environment, because when the robot's sensor detects another robot, it approaches it as if it were an object. For the goal of this project it is not a problem as long as another robot is not attached to the environment.

Initially with the W shaped map problems were encountered when the map dimensions were not large enough. In the corners of the letter, the same side ("left" or "right" as previously described in the algorithm section) of the laser sensor of the robot would detect several obstacles at the same time and the robot would begin to move in a strange way. Due to this problem there was a need to increase the dimensions of the map.

V. Conclusions

In terms of reaching the optimal distance of the algorithm, the V shaped map had more success than the W shaped map. This means that the algorithm is not so good in environments where the shape of the walls is not constant, forcing the robot to change direction more often.

In general, the project was a success as the robot was able to reach its goals in both environments.

The algorithm developed can still be improved regarding the limitations that were referred before, although these are not critical in the context of this project.

References

[1] F. von Hundelshausen, M. Himmelsbach, F. Hecker, A. Mueller, and H.-J. Wuensche, "Driving with tentacles: Integral structures for sensing and motion," Journal of Field Robotics, vol. 25, no. 9, pp. 640-673, 2008.

[2] C. Cadena et al., "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," IEEE Transactions on Robotics, vol. 32, no. 6, pp. 1309-1332, 2016.

[3] C.-H. Hsu and C.-F. Juang, "Evolutionary Robot Wall-Following Control Using Type-2 Fuzzy Controller With Species-DE-Activated Continuous ACO," IEEE Transactions on Fuzzy Systems, vol. 21, no. 1, pp. 100-112, 2013.

[4] ROS. (2018, Oct. 29). ROS/Introduction - ROS Wiki [Online]. Available: http://wiki.ros.org/ROS/Introduction

[5] ROS. (2014, Jul. 25). stdr simulator - ROS Wiki [Online]. Available: http://wiki.ros.org/stdr_simulator

[6]] K. Bayer, "Wall Following for Autonomous Navigation," SUNFEST, University of Pennsylvania, Summer 2012.