IRS - Assignment 1

Guy Farjon

November 2021

1 Question 1

- Euler Angels & Quaternions - Both of these terms refer to the relation between two coordinates systems. Euler angels is a way of specifying the angular relation between the two coordinate systems, the environments coordinates and the robot's coordinates, often using $(x, y, z, roll, pitch, yaw)$. Quaternions is another form of representation in which we specify the axis and the amount of rotation by using (w, x, y, z). It is a transformation between one coordinate system (of the robot) to a fixed coordinate system (of the environment).
- Sensor Fusion - Fusing the sensed information received by multiple sensors. Each sensor has it's merits and constrains, hence fusing data from multiple sensors (of the same type or different sensor) can reduce the uncertainty regarding the environment representation.
- Deliberative Paradigm - This paradigm is all about sensing the environment, **plan** the next move, and act accordingly, in a never ending loop. In this paradigm, we need to have a prior representation of the environment and some logic on how to act according to a specific environment state.
- Reactive Paradigm - This paradigm drops the **plan** stage, and hence always sensing and acting. The sensed information here immediately transformed into actions by keeping "behaviors" in memory. These "behaviors" are reactive rules - mapping a state to an action.

2 Question 2

In Pure-Pursuit, we assume that to reach a specific point (to follow a path for example), the robot moves in a circular movement. The parameter L is the look-ahead parameter which states the direction of the robot in the next step. When L is too small (0.3), the robot might frequently change his direction, resulting with a zigzag movement. When L is too large (0.9), it will take time for the robot to get on the path (and as in our case, might hit an obstacle on

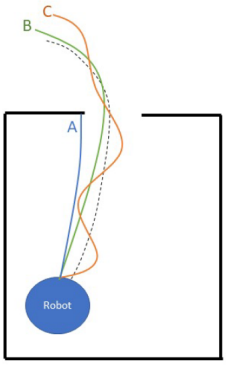

Figure 1: Results from trying to follow a given path (the dotted black line) give the three different values of L (lines A,B,C)

the way). A better value of L (0.6 here) will result on a quicker convergence to the path itself. To sum up - $A - 0.9, B - 0.6, C - 0.3$.

3 Question 3

This question was solved using Python. The script is added to the submission. In this document, I only report the final values of each parameter. For the complete calculations please see the attached Python script.

1. Data visualisation: I used two different graphs to visualise the data. In figure 2 left, the relevant area of search is shown. From the plot, we see that we have an object around the 200 degrees - meaning lower left side. In figure 2 right, I moved to a Cartesian representation. Do note that this is a rough estimation of the data since I'm not using the exact coordinate system for each of the obstacle points. Hence the visualisation is used here only as a tool for further estimations.

2. The value of D is the is smallest distance to the obstacle segment. I went over the readings and took the smallest value (which is not simple noise) - $D = 69.99$.

   The value of W is the length between the left-most and the right-most points of the obstacle. I've used the cosine sentence to calculate the exact value: $W = 17.18$

   θ is the column index of the smallest distance to the object (D) - $\theta = 199$.

3. To calculate P we'll move to Cartesian representation, $P = (-66.18, -22.79)$

4. To transfer the point P_{rcs} to P_{wcs} I've used the equations taught in the lecture. Using the given x_0, y_0, ψ, the answer is: $P_{wcs} = (-15.92, -12.83)$

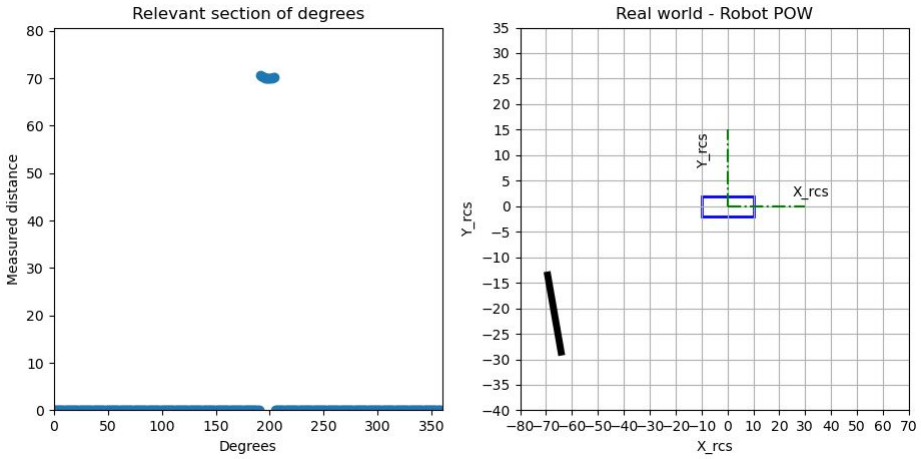

Figure 2: Left - plotting the relevant area of degrees to search the object. Right - plotting the obstacle from the robot point of view

4 Question 4

Given the environment described in figure 3:

1. The robot might accidentally reach to a point in which the repulsive filed from the obstacle and the potential field of the target cancel each other.

2. The robot might get stuck in a cyclic behavior in a corner of the obstacle.

To solve these issues, we will first add random values to each force field (to disable the possibility of getting stuck in place), and we will dynamically add repulsive field to previously visited location (to avoid getting stuck in a corner).

Figure 3: Illustration of question 4