PA 2, CS 81

Garret Andreine

October 2020

1 Description

- Breaking down this PA I had to think of what the PD controller would be controlling. Reflecting on our previous homework I realized that the rotational velocity and how much the robot is turning towards the wall would be the value we would be manipulating with the PD controller.
- Taking this into consideration I needed a way to calculate the error and implement the PD controller. So I wrote the get_ error() method to get the current error between the robot and the wall.
- I then used the laserscan methods to find what was to the right of the robot and use those values to determine the error and what was in front of the robot.
- Using this current error and the previous error I was able to implement a PD controller with Kp and Kd variables in the PD equation
- I connected the result of this PD controller to my Rotational Velocity and then multiplied it by 2pi to amp up the speed of the turn.
- After doing this I needed to come up with a finite state machine that would switch between the robot turning how much I needed it to and adjusting when something is in front of it. So in the main function I set it up such that the robot scans for what is around it and use the distance right in front of it to determine whether the rotational velocity to fix the error should be used or whether the robot should hard turn left to avoid crashing.

2 Evaluation

- Overall I think the implementation of the program went fairly well
- I knew the value of Kp effected the turning of the robot and the Kd effected how well the robot went straight and looking at the robot it moves pretty smoothly. From a zoomed out view it clearly makes arcs which could be ironed out by adjusting the Kp and Kd. But for our purposes it is effective.
- I think the goal of following the wall .5 meters away was pretty practical but after letting the robot run for over 45 minutes I can see on a tighter spot in the maze it flipped wall sizes rather than going through the gap, which could be something the error distance didn't allow the robot to pass through.

- The robot also had to move painfully slow to avoid collision. The speed is set at .2 meters to avoid collisions.
- Overall I would say the implementation went well but there are clear areas to improve on.