CS 3630 Project 3 Part 1

Name: Andrew Wang
GT username: awang350
GTID: 903378837
Group #: 13

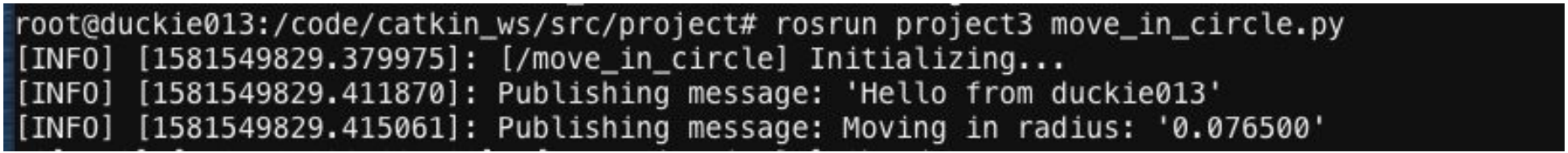
Name: Matthew Yang
GT username: myang349
GTID: 903447357

1. How did you implement calculate_radius?

We used the formula outlined in the given notes. Specifically, given l, the distance between the centers of the two wheels, Vr, the velocity of the right wheel, and Vl, the velocity of the left wheel, we can calculate both the SIGNED radius and rate of rotation around the ICC as follows:

$$R = \frac{l}{2}\frac{V_l + V_r}{V_r - V_l}; \quad \omega = \frac{V_r - V_l}{l};$$

2. Screenshot and paste the commands returned when you run command rosrun project3 move_in_circle.py



3. What are the initial velocities you used? What is the returned radius? How did the robot behave when you ran it with those configurations?

Our initial velocities were: velocity_left = 0.20, velocity_right = 1.0

The returned radius was 0.07650

The robot moved very quickly in a small, slightly uneven, circle.

4. Try making the robot move in a ‘better’ circle. (TIP: try adjusting the wheel speeds and/or the radius. Running on different surfaces may return different results.) What did you do to make the robot run in a circle of similar radius as calculated? Write the velocities and calculated radius you used.

To make the robot run in a circle of similar radius as calculated, we experimented with various velocities for the left and right wheel.

Firstly, it appears that the robot performs horribly at low velocities, regardless of the calculated radius. Additionally, the robot also performs poorly with large radiuses. In the end, we tried to balance both.

With vl = .5, vr = .8, and a calculated_radius of .2210, we were able to achieve a decent circle.

5. Why do you think the robot did not move as you thought? What can you do (Hardware and Software) to improve the results?

Firstly, the robot performs terribly on carpet, likely due to the carpet's high friction and semi-uneven grounding. Secondly, the the wires of the duckie bot would sometimes interfere with the wheels, which also caused issues. Lastly, a (likely minor) issue is that the battery pack sliding can cause uneven weight distribution.

Hardware wise, we can improve the results by picking a good surface such as a perfectly flat table. Additionally, we can also make sure to secure the wires / battery pack carefully to ensure optimal physical conditions.

Software wise, since the radius depends on the proportion of the velocities, for a given target radius, we can make sure to select appropriate velocities that are not too fast / too slow, to allow for optimal performance. We can also perform calibration to ensure the wheels spin at the correct rate as specified.