CSCE 274 Section 001 Fall 2019 -- Project 2 -- A. Staton, M. Ziemer, R. Carff
Date: November 18th, 2019

Description:

Project 3's goal was to implement Proportional, Derivative (PD) control on the iRobot Create2. This was done using an infrared (IR) sensor on the right side of the robot. The robot was intended to read the IR sensor value as it was driving alongside a barricade, to be able to appropriately "follow" the barricade at an appropriate distance. When a turn is sensed on the barricade (it is sensed by the previously mentioned IR sensor), the iRobot was intended to use the PD control's logic to adjust its direction.

The PD logic has two main sections: the proportional error and the derivative error. These two errors are summed to create one, total value for error that accounts for current error (the proportional error) and the rate of error (the derivative error). Each of these factors is affected by their respective gain values.

The design of Group 4's program has two parts. The first part controls the actual driving/motion of the robot. The second part updates and reads the needed sensor values from the right IR sensor and bumpers. The PD control logic is implemented with the sensor control. The group chose to consolidate the reading of sensors into one thread due to physical limitations of the iRobot Create2, which can only report accurate sensor values every 15 milliseconds. By using one thread for all sensors, Group 4 was able to mitigate any potential issues with buffer overflow.

In the driving/motion thread, the Create2's "Drive Direct" function was used to move the robot in accordance with calculations from the PD control in the opposing thread. "Drive Direct" uses a separate left and right wheel velocity to move the robot; so, the total error from the PD logic accurately added to or subtracted from these wheel velocities to "follow" the barricade.

Total error is a function of the proportional and derivative gains. To select these values, Group 4 determined a set point from the sensor readings with an ideal distance between the iRobot Create2 and the barricade/wall. This value was reported as a high value (about 20,000); so, the group identified the need for low gain values. Errors would routinely be in the range of [-10000,10000]. If the implementation subtracted/added 10000 to the velocities in "Drive Direct,"
an out-of-bounds error would have been thrown by the Python interpreter. After lots of tuning for stability with trial and error, Group 4 was able to identify the appropriate gain values.

Evaluation:

Group 4's program works; and it works well. The iRobot Create2 successfully follows a wall and adjusts for both right and left turns in the wall. Limitations arise when the barricade is suddenly moved away the iRobot. The iRobot detects a continuously high error and in response, continuously turns. Another limitation of Group 4's project is shown when the iRobot

approaches a sharp (90 degrees greater) turn. The reading of the right IR sensor alone is inadequate to be able to detect these abrupt changes in pattern. A challenge of the program was found when Group 4 attempted to read the left IR sensor of the iRobot. The infrared sensors along the bumper of the robot are not symmetrical, and the left IR sensor is more closely aligned with the front of the roomba.

Group 4 does not think that PID control would be appropriate for this assignment. Since the robot is continuously moving forward, a history of error provides minimal benefit to the robot's calculations of where is is along the wall. Also, due to the limitations of a Raspberry Pi, a prescribed size of error history would be needed. If the robot was to operate for a long period of time, the memory of the Raspberry Pi would overflow.

The group managed it's codebase with Git. Generating branches to test ideas and test functionality, while keeping a 'master' branch to maintain a working submission proved to be beneficial. It was a large factor in the group's efficacy.

Allocation of Effort:

This project's group (#4), has three members: Miles Ziemer, Robert Carff, and Austin Staton. Each member of the group made significant contributions to the development, advancement, and completion of this project. Ziemer provided many of the mathematical calculations and gave structural input to ensure the project's completion; Carff wrote many functions of the code and guaranteed their effective internal documentation; Staton also contributed to some of the programming functionality and wrote the technical documentation. This group proved to be both effective and successful in their completion of Project 1.