**To:** Dr. Berry  
**From:** Peter Heath, Matthew Schack, and Data  
**Date:** 1/7/16  
**RE:** Lab 04 Wall Following

The Purpose of this lab was to make a program that would allow Data to follow a wall. The specific goals included: finding a wall from random wander, following a wall on one side of Data, turning around inside and outside corners, and driving down a hallway following both walls. To accomplish this we used a layered architecture where layer 1 is wall following, layer 2 is following both walls, and layer 3 is random wander; the diagram of the architecture can be found in Appendix A.

To get all of the behaviors we implemented a finite state machine. We used a case based structure where each state has a different value. To change between the behaviors sensor values had to be above or below 2 times the maximum value distance Data can be from the wall. To switch between following both walls and following a wall the value had to be 3 times the maximum value. We did this so Data does not continuously switch between states but just stays in one state.

In order to get smooth wall following we used a proportional and derivative controller. When we were tuning the proportional controller Data would oscillate a lot always missing the correct distance. After lots of calibration we toned down the oscillations and the overshoot. After that we incorporated a derivative controller to further dampen the oscillations and smooth out our wall following.

Having a really high proportional controller made the oscillations really bad and ended up making Data go into the wall; however a really low proportional controller made data move basically ignoring the wall causing it to slowly crash into the wall. When the derivative controller was too high a small change in distance caused a huge change in the motor speeds making the oscillations even worse. A low derivative controller ended up doing nothing. In the end we used a proportional controller of 15 and a derivative controller of 2.

To implement the feedback control we used a differential off of a base motor speed. If the sensors read a value outside of the allowed range then the difference between the correct value and the current value would be computed and multiplied by Kp. In later iterations we also added in that the result would be combined with the difference between the current distance and the past distance multiplied by Kd. This number was added to one wheel speed and subtracted from the other wheel speed to cause the robot to make a gentile turn.

When the robot approaches an outside corner the front IR loses the wall telling Data that it is approaching a corner. This causes a special case where data will run 2 movement commands rather than one. This puts Data a little after the wall. The next time the sensors are polled the special case will activate again cause Data to do to a sharp turn clearing the corner and turning the IR sensor towards the wall. Finally, Data will continue moving relying on the IR sensor only until the sonar sensor can see the wall as well. Inside turns are much simpler where Data just turns 125° and moves forward when the front sensor sees something approaching. By over shooting the corner Data inserts a disturbance which allows Data to deal with any irregularities.

Data was pretty accurate at maintaining the correct distance from the wall when it was in a straight part. On corners it would sometimes go too far or too close to the wall, but over time it would correct itself. In addition during testing Data would occasionally lose the wall when it was turning outside corners but through calibration the problem was mostly resolved. The wall following program could be improved by writing a new function that continuously drove while polling sensors rather than stopping after each movement as well as accounting for corners better.

Random Wander

Follow Left

Follow Both

Follow Right

Move Random

Poll sensors

Movement

Appendix A:

Act

Sense

Wall following

Both Wall following

Random Wander