CSCE 274 - Section 01 - Fall 2017 - Project 4 - Khalid Salah, Logan Fisher, Julian Hong

Lab Report Project 4

**Description:**

The purpose of this project was to have the iRobot use infrared characters from the dock and the lighthouse of the robot to find the dock, and then dock the iRobot. The robot still follows the wall as in Project 3, but when it picks up a signal from the dock, it’ll drive towards it and dock itself.

We used the same dependency files from project 1 and 2, Linguist.py and Jaguar.py. Linguist.py, our serial communication interface, is used to handle raw communication between the physical iRobot and the robot action interface. Jaguar.py was created to handle the actions of the robot and serves as the action interface. Our serial communication interface was left untouched as no changes were needed. Our robot action interface did need some additional functionality added to it. We added a function that processed and returned data from the omnidirectional, left, and right sensors. The data is used to determine where the dock is.

Our main file, TheDock.py, has a thread to drive straight, a thread to listed for bumps, a thread for handling button clicks, a thread for IR sensors to follow the wall, and our newest thread that implements using the Dock’s beam configuration. We extended our IR method by including our new code in there. The iRobot follows the wall as expected, while the sensor is looking for the dock. Once it finds the dock, it breaks out the while loop and finds which side of the iRobot it’s closer to. If not straight ahead, the iRobot moves until it is sure that the dock is ahead. Once it’s within a forward motion of the iRobot, it drives at a speed where it won’t knock over the dock, docks itself, and plays a happy song.

Evaluation:

The robot works as it is expected to. When the clean button is toggled, the robot begins following the wall. It collects data from the infrared sensors to find out where the dock is compared to the iRobot. The iRobot, has commands written for the different situations it might be in. Sometimes the dock is straight ahead, sometimes it’s not. We laid out most of the options we could get from what the outputs were and it docks itself every time. A couple minor problems were encountered. First problem is if the iRobot picks up the dock signal and there’s an obstacle still in the way, it goes for the dock and completely disregards the obstacle because it’s not checking for obstacles when it’s docking. The second problem is that it doesn’t always rotate in the direction of the dock. This isn’t a huge problem because it will still rotate until its lined up with the dock. The way we tell which side the dock is on is by saying if it crosses the green buoy and then red buoy then it must be on the left side, but if it crosses the red then green it must be on the right. The problem with this solution is that the sensor read aren’t always in the correct order to cause it to turn in the right direction.

Allocation of Effort:

Khalid Salah:

* Debugged TheDock.py
* Established Buoys
* Assisted with Docking
* Testing

Logan Fisher:

* Logic Behind Docking
* Debugged TheDock.py
* Wrote Dock Method

Julian Hong:

* Wrote Report
* Debugged TheDock.py
* Assisted with Docking thread
* Testing