Group: Sean Meyer, Scott Meyer, Ameen Sassi, Andrew Bentley

Introduction: This is the Final Project Report for Sean Meyer, Scott Meyer, Ameen Sassi, and Andrew Bentley. We developed an autonomous car.

Overview of System: Our autonomous car is built using several different devices. The car itself is a simple plastic car with two wheels connected to the motors and a third stationary ‘wheel’. The motors are powered by a rechargeable battery pack. The program is run using a PCDuino and Shield mounted onto the car. It is powered by a battery pack of 4 AA batteries. Our sensory data is provided by 2 laser sensors that are mounted on the left and right side of the car and an ultrasonic sensor on the front of the car. We connected a USB WIFI router to be able to connect to the PCDuino remotely. We also used a Breadboard for connecting multiple sensors into a single slot on the PCDuino.

Challenges:

One of the main challenges was getting the sensors set up and getting them all working together. Our group has issues getting multiple laser sensors working together. The issue stemmed from the fact that only one sensor can be read from over I2C at once using the Shield and the PCDuino. We got around it by modifying the Adafruit\_VL6180X.cpp to take a I2C address. This allowed us to run our 2 laser sensors at once without having to shut one down to read from the other.

Another was not having step motors. We are not really sure what type of motors are on the car but whatever they are, they were somewhat difficult to work with. Step motors would have allowed us to better control the car in its movement compared to the motors that we were working with.

Another problem that we had was driver support and documentation, or rather, the lack of it. When trying to hook up the lasers and getting the sensor data from them, the libraries included with them were not designed or documented very well at all. This made the coding that retrieves the sensor data difficult to develop for. The libraries also forced us to build the code to drive the car in C as the libraries we needed were either not very good or nonexistent in Python.

The last issue that we ran into was making our C code and Python code to communicate. We developed the code that gathers the sensor data as well as the code to make the car drive in C code. The pathfinding algorithm that we built so that the car knows where is it and how to navigate to the end of the course was created in Python. Two different coding languages talking to each other is not impossible but it adds another level of complexity to our project. We managed to overcome this challenge by having the C code start up the Python code and having them communicate through strings. The C code provides a string to the Python which uses it to determine the next step. The Python then returns a string to the C, telling it where to go.

Proposed Algorithms:

Our system for the car was designed in two separate parts. The first is the actual code that runs the motors, gets data from the sensors, and converts the information. The second is the pathfinding algorithm that helps the car keep track of its location in the track and find the quickest path to the end. The pathfinding algorithm is built in Python. It is built on the idea that the map can be split up into 1 foot by 1 foot sections. It uses a text file that contains an adjacency map that lists each section and what is connected to it.

Evaluation and Results:

Lessons Learned: Embedded System development is both fun and difficult.

Conclusion:

Setup of Car: The setup of the car is fairly simple. The code is already loaded onto the PCDuino.

Video URL:

Division of Work:

Sean: Development of the Driving algorithm, Setting up the sensors

Scott: Development of the Driving algorithm, Setting up the sensors

Ameen: Development of the pathfinding algorithm, Initial setup of the car

Andrew: Development of the pathfinding algorithm, Create the report