EECE 284: Electronics Laboratory

Project Report for Fast Orange—an Electromagnetic Track Rover

John Harvey, 29240116 (25%)

Liam Hodgson, 29154119 (25%)

Scott Lawson, 34359117 (25%)

Rowan Walsh, 30817118 (25%)

March 31st, 2014

Course taught by Dr. Jesús Calviño-Fraga, P. Eng

Contents

[1.0—Introduction 3](#_Toc384499663)

[1.1—Mechanical 3](#_Toc384499664)

[1.2—Electrical 3](#_Toc384499665)

[1.2.1—Power supply 3](#_Toc384499666)

[1.2.2—Motor control 4](#_Toc384499667)

[1.2.3—Sensor array 4](#_Toc384499668)

[1.2.4—Microcontroller 4](#_Toc384499669)

[1.2.5—LCD shield 5](#_Toc384499670)

[1.3—Software 5](#_Toc384499671)

[2.0—Project analysis 6](#_Toc384499672)

[2.1—Testing 6](#_Toc384499673)

[2.2—Design analysis 6](#_Toc384499674)

[3.0—Conclusions 7](#_Toc384499675)

[References 8](#_Toc384499676)

[Bibliography 9](#_Toc384499677)

[Appendices 10](#_Toc384499678)

# 1.0—Introduction

Our project for EECE 284 was an Arduino-based rover, designed to follow a sinusoidal, 15.175 kHz electromagnetic signal. The format of the course was a time trial: each group’s robot completed a circuit of a racetrack. The runs were timed, and after each group’s entry attempted the course, the robot with the fastest time was awarded the highest mark.

The signal the robot was to follow was generated by running a 20V peak-to-peak 15.175 kHz sinusoidal voltage through a wire. The wire was embedded in the competition surface, creating the path the robots were to follow. In addition to straight track, signals were placed along the track, indicating the beginning and end of the path, as well as the direction to turn at the intersection of wires. Bridges were another feature of the track, which added an extra challenge, as robots could not vary their path too greatly while on the bridge, for risk of falling off, and as the depth of the wire from the surface was greater on the bridges.

The robot followed the racecourse by detecting the electromagnetic field generated by the wire, using inductors mounted at the front of the robot. These signals were put through peak-detection and amplification circuits, and then fed into a PID algorithm. The outputs of the PID algorithm drove MOSFET motor drivers, allowing the robot to follow the wire and racecourse with a high degree of accuracy.

## 1.1—Mechanical

The chassis, the main mechanical component of the robot, was designed to minimize mass and structurally incorporate as many essential components (microcontroller, motors, battery) as possible. The robot consisted of three sections: the main chassis, sensor array, and a piece of balsa wood which joined the two. The front chassis and sensor array were 3D printed in bright orange poly-lactic acid (PLA), to minimize mass, and in homage to the industrial-strength, orange-scented hand cleaner, Fast Orange.

The main chassis consisted of three components: a battery holder, boost-converter protector, and motor holder. The battery used (from a Nokia SOMETHING SOMETHING) fit neatly under the robot, as it was flat and thin. Four short wires extended from the battery holder to the boost-converter protector, mounted at the front of the main chassis. Wires carrying 5 and 10 volts left the boost converter, carrying power to the Arduino Uno and motors, respectively. A shield incorporating an LCD monitor, potentiometers, several screw-terminal ports, and six buttons sat atop the Arduino Uno, which was mounted on the motor holder. Unsurprisingly, the motor holder held the motors. The other components (boost-converter protector, battery holder, balsa wood) were attached to the motor holder.

The sensor array held a (NAME NAME) bearing and three inductors—two perpendicular to the direction of motion of the robot, one parallel, and all mounted close to the ground. The balsa wood was firmly inserted in the sensor array, holding it in front of the main chassis.

Various nuts, bolts, plastic spacers, and some hot glue held the components together.

## 1.2—Electrical

### 1.2.1—Power supply

The choice of batteries to power our robot was important, as a higher voltage would allow us to run the motors at a higher RPM, and a larger capacity would allow for longer testing time. The battery chosen, a rechargeable lithium polymer battery with a voltage of 3.7 volts and a capacity of 950 mAh, was taken from an old cellphone. Quick to recharge and lightweight, the battery fell well below the limit set for this competition. To provide an appropriate voltage for the Arduino and motors, a DC-DC boost converter was used, allowing us to supply 10V to the motors, and 5V to the Arduino, without surpassing the maximum battery voltage of 6V. The datasheet from the Solarbotics GM4 servos indicated that supplying the battery with 9V would result in a speed of 104 RPM, unloaded, compared to 77RPM at 6V.

[boost converter circuit, part number]

Comments on constant-current battery, selection of battery to work with boost converter

### 1.2.2—Motor control

The two Solarbotics GM4 servos provided the driving force for the robot, and were controlled via MOSFET drivers. The gate of the MOSFET was connected to the PWM output of the Arduino, and the source and drain were connected to the 10V supplied by the boost converter and the motor, respectively.

[MOSFET circuit?]

### 1.2.3—Sensor array

The robot was to follow a path defined by a buried wire carrying a 20V peak-to-peak signal at 15.175 kHz. This magnetic field produced by the wire was detected by two 100mH inductors. By placing appropriately-sized capacitors () in parallel with the inductors, the sensors could be tuned to a resonant frequency equal to that of the desired field. 1nF capacitors were selected, producing a resonant frequency of 15.9kHz, which was sufficiently close to the desired frequency to be effective.

The AC signal induced in the inductor was then converted to DC via a peak detector, and amplified. (GAIN GAIN GAIN—was it variable?) The structure of the peak detector is a BJT connected in parallel with a capacitor. A diode rectifies the input voltage, which charges the capacitor. The capacitor charges to the peak voltage of the signal, and is read by one of the digital inputs on the Arduino. After reading the capacitor’s voltage, the gate of the BJT shorts to ground, discharging the capacitor and restarting the process. A non-inverting amplifier circuit, consisting of an op-amp and a potentiometer, is used to provide variable gain.

Two inductors were mounted perpendicular to the direction of motion, to provide relative signal strength, indicating whether the robot was to the left or right of the path. One inductor was mounted parallel to the direction of motion, to detect wire perpendicular to the track, indicating the start, end, and turn warnings on the track. These inductors were mounted closer to the bulk of the robot than the other sensors—the further away the perpendicular sensors were from the main chassis, the more sensitive they would be to small changes in the angle of the robot. By placing the parallel sensor close to the main chassis, turns could be detected later, reducing the delay between detection of a turn and the turn itself.

[peak detector and amplifier circuit]

### 1.2.4—Microcontroller

An Arduino UNO microcontroller was chosen for the controller of our robot, as team members were more familiar with it than the provided microcontroller. Sufficiently powerful, reasonably inexpensive, and with a wealth of libraries, the Arduino read the sensors, controlled the motors voltage, displayed information on the LCD, and ran the PID algorithm on the sensor inputs.

[Arduino photo]

### 1.2.5—LCD shield

An LCD shield incorporating several buttons and knobs (DO THE KNOBS ACTUALLY EXIST) was used to provide an easy interface for debugging and tuning the robot.

[LCD shield photo]

## 1.3—Software

Some words about the code that makes the robot work

# 2.0—Project analysis

Some words about the things we’re going to analyze

## 2.1—Testing

Some words about tests we ran on the robot: this section doesn’t seem critical so I think we should probably ignore it. I also can’t think of any tests we ran that were important or well-documented.

## 2.2—Design analysis

While our robot performed well, several adjustments would be made if the project was undertaken again. Most importantly, the inductors would be more securely fastened. While it was easy to make fine adjustments to their positions, by bending the wires from which they hung, this created problems. The inductors were bumped around during transport of the robot, and in the occasional collision, which had dramatic effects on the robot’s ability to sense the line. Seemingly inconsequential adjustments proved to be the difference between successful and unsuccessful runs, and it was extremely difficult to quantify the adjustments made.

Using a cell phone battery was one of the better decisions made: in addition to allowing us to use a higher motor voltage, the size, ability to recharge, and weight proved advantageous. Teams, on average, went through 6-8 AA batteries. We used one battery, from an old cell phone, which was easier on the environment and pocketbook. We would likely reduce the motor voltage if the project was undertaken again: the robot ran at approximately a third of its top speed, as controlling the robot became extremely difficult after that point. A reduction in maximum speed would allow for more precise control over motor velocity.

# 3.0—Conclusions

Another summary about the design and functionality of the robot, including problems that came up and a rough time estimate

# References

If we referred to it in the body of the report (a book, website, source of information) we need to have the full citation here. The in-text citation should just be a number in square brackets, [n], and the references should look something like this:

[1] Smith, J, and F. Jones, “Designing a universal logic circuit”, Journal of Impossibly Wonderful Electronic Circuits, v.3, n.1, pp. 21-35, March, 1910.

[2] Jones, F and J. Smith, “Why universal logic circuits are impractical” , …

# Bibliography

If we didn’t refer to it in the report but its good background reading, or some literature we used but didn’t mention in the report, it goes here.

# Appendices

Any ‘extensive theoretical analyses’, part drawings, and source code should go here. The code should be properly documented (whatever that means) and indented.

Don’t attach datasheets, compiler manuals, or anything which was given in references or the bibliography.