**Initial Experiment To Test Magnetometer Readings**

**Introduction**

This report discusses an experiment to look at and understand magnetometer readings from the Canary, a 1x4” sized board with multiple sensors. This initial test was used to see how the magnetometer readings were affected as a box of screws was moved in different motions near the Canary. It was important to know how the readings would react to real data and how they would react to noise. The experiment was conducted by Furqaan Afzal in a lab setting.

**Experimental Details**

The hardware consisted of a Canary for the magnetometer readings, a CC debugger to flash the Canary with the appropriate firmware, an FTDI cable to send data over UART, a laptop computer to collect the outputted data, a box of screws, and a power supply to power the Canary. Figure 1 shows a snapshot of this setup, and Figure 2 shows the box of screws used.



**Figure 2**3 x 2 x 2 box filled with screws

**Procedure**

In this experiment, a 3 x 2 x 2 box of screws was held approximately five inches away from the Canary for each motion. The Canary sampled a new reading every 40 milliseconds or at a frequency of 25 hertz. The experiment lasted a total of 30 seconds and collected a total of 750 sample readings during that time.

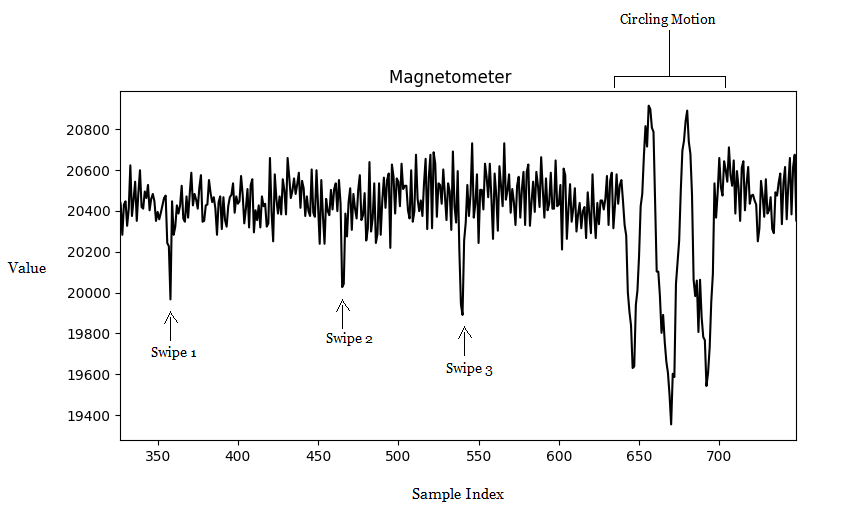
The first 14.4 seconds were a waiting period to collect 360 sample readings at an idle state. After the waiting period, the box of screws was moved in a swift swiping motion around five inches away from where the magnetometer sits on the Canary. After the first motion, there was another waiting period of 4.28 seconds or 107 sample readings. After the second waiting period, there was a second swiping motion in the opposite direction. After the second swipe, a third waiting period ensued for a total of three seconds or 75 sample readings. After this waiting period, there was a third and final swiping motion. After all the swipes, there was a waiting period of four seconds or 100 sample readings. Once that was over, there was a circling motion that lasted a total of 2.4 seconds or 60 sample readings. To finish off the experiment, there was a final waiting period of 2.2 seconds or 55 sample readings.

**Results And Discussions**

This section analyzes the results of the experiment. Each sampling held the magnetometer x, y, and z values for that sampling. The vector of the magnetometer data is plotted on a line graph in Figure 3 on the next page. The image is zoomed in to better show the swipes and circling motion. Because the image is zoomed in, it omits the initial waiting period. However, the waiting period in-between motions and the final waiting period can still be seen.

Looking at Figure 3, each individual motion is clearly shown in the line graph. Although the noise shown during each waiting period is also apparent, it does not hide the real magnetic data. The noise can range from 20,200 to 20,700. In other words, any real data within that range will be cancelled out and will not be distinguished from the noise. Since the data of the reading for each motion was either above or below that range, we can clearly see each motion depicted in the graph.

The equation we used to calculate the vector can be found in the Appendix section of this report.



**Figure 3**Line graph of the vector of the magnetometer x, y, and z values

**Conclusion**

Overall, the test went as expected and gave us the information we needed. There is some noise with the magnetometer which gives us better information for the next set of data we collect. We may run new tests in the future to see how close or far in inches, we must be in order to get good readings. Nonetheless, we learned that if the magnetic data values are not in the range of 20,200 and 20,700, then we can safely conclude that we are seeing real magnetic data and not just noise.

**Appendix**

Vector Equation: V^2 = magX^2 + magY^2 + mag^Z^2

The data is stored in a mysql database file called readSerial45.sql.

The script readSerial.py was used to collect the data.

The script plotData.py was used to create a line graph of the data.

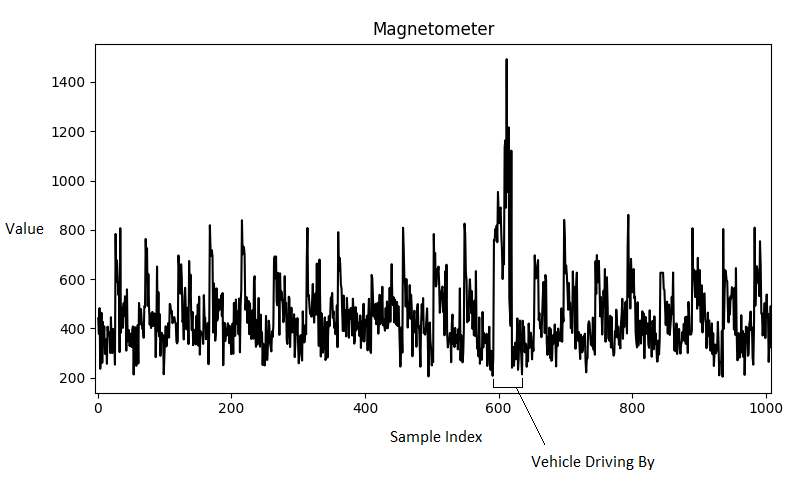
The Canary is running the uart\_demo firmware.

The source code for uart\_demo and python scripts can be found using the following git url: https://github.com/PureEngineering/contiki/tree/master/examples/canary

**How To Differentiate Between A Small And Large Vehicle**

**Objective**

Our goal is to differentiate between a small vehicle and a large vehicle driving by. The current method to detect a vehicle driving by is to use the Magnetometer on the Canary. Looking at the pulse width in Figure 1, it is possible to detect when a vehicle drives by. This is a good method for detecting a vehicle; however, this method alone cannot detect the size of the vehicle, because a large vehicle driving by at a high speed will be indistinguishable from a small vehicle driving by at a slow speed. Therefore, In order to differentiate between the two vehicles, we must also know the speed of the vehicle.

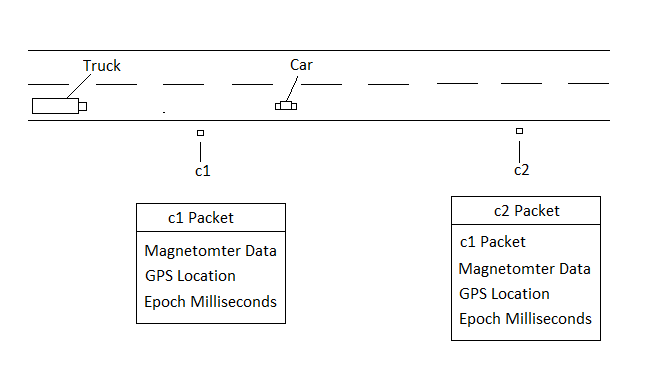


**Figure 1**Magnetometer data at four feet away

**Proposed Solution**

The proposed method to detect the speed of the vehicle will introduce the mesh and use two Canaries which we will refer to as c1 and c2. Both Canaries, c1 and c2, will be placed an arbitrary distance away from each other along the same road. Take a look at Figure 2 to see an image of the proposed setup. When c1 detects a vehicle, it will issue an event which will send a packet to c2. The data packet will include the magnetometer data, GPS location, and the epoch time in milliseconds. When c2 receives this packet, it will look for the same vehicle and once it detects the vehicle, it will send the data for both c1 and c2 to the gateway. The two data sets will then be compared. We can get the speed by looking at the time it took a vehicle to get from the location of c1 to the location of c2. Measuring both the pulse width and the speed should give us a better understanding of the size of the vehicle that drove by.

There are of course caveats to this solution, there must be a way to confirm the two vehicles were in fact the same vehicles. If for some reason c2 does not detect the vehicle and a different vehicle drives by, the data will be incorrect. Or, if it detects a vehicle driving on the other side of the road, the data will again be incorrect.



**Figure 2**Image of the proposed setup