Dead Reckoning 2.0
Inertia

Thomas Piacentino, Dylan Ross, Sagar Patel, Kevin Lane, Navdeep Sekhon

Executive Summary:

Team Inertia's overall objective was to design and construct a Dead Reckoning system that is able to accurately track user movement when GPS signals are degraded. Our Dead Reckoning system consists of two different subsystems, or units, known as the "Stationary Computing Unit," and "Mobile Unit," that collaborate in order to achieve our desired goal of mapping user position. The Mobile Unit utilizes data collected by various sensors, including an accelerometer, magnetometer, and infrared proximity sensor, which provide the necessary information, such as step count, stride length, and angle of direction, to determine user position in relation to the location in which GPS became ineffective. Once this data is collected, it is transmitted to our Stationary Computing Unit, via XBee radios, where it is processed and ultimately plotted in order to provide a visual of the user's path of travel, and current position.

Team Inertia is hoping our product is an improvement, and evolution of the previous attempts at Dead Reckoning systems due to its automated stride measurement system. This stride measurement system relies on an accelerometer-assisted pedometer that triggers an infrared proximity sensor mounted on the user's foot to take a measurement of the distance between the user's front and back foot. This measurement is used to determine the magnitude of the stride length by adding the distance between the user's feet and the length of the user's shoe. In addition to the stride measurement system, we have implemented a magnetometer-based compass to increase heading accuracy. Ultimately, we are hoping that our innovative ideas have lead to a Dead Reckoning system that is dependable, reliable, and, after a bit of additional work, ready for the field.

Problem Statement:

During WWII, some of the first INS (Inertial Navigation Systems) were developed to help guide rockets. Seventy years later, the combination of gyroscopes, accelerometers, and magnetometers are commonly used in the field of Dead Reckoning to help locate a person, or an object in the absence of GPS signals. These stand-alone systems are crucial for the safety and operation of many forms of transportation, in particular: airplanes, submarines, and ships. In the off-chance that a plane were to lose communication with ground, for example, an on-board INS would assist the pilots in roughly determining the location of the plane by taking real time linear and angular acceleration values, and using computational models to update the planes location on an on-board mapping system.

Foot navigation in dense outdoor or underground environments poses the issue of degraded GPS signal, rendering most personal tracking units useless. Some example causes of degraded GPS signals are as follows:

- Delay due to the attenuation of the signal as it passes through the atmosphere. Averaging the delays is implemented as an error correction.
- Signal reflection caused by objects such as tall buildings or large rocky areas prior to reaching the receiver. Buildings, terrain, or other dense outdoor environments can cause signal reflection prior to reaching the receiver, these can also completely block reception leading to position error or possibly no reading at all (seen indoors, underwater, and underground).
- Inaccuracy of the receiver internal clock in comparison to the satellite's atomic clock.
- Inaccuracy of the satellites reported location along with the number of satellites the receiver can "see."

Inertial navigation allows self-contained, hybrid systems to be developed for instantaneous position and velocity with accurate azimuth and altitude vector calculation. Today, technological advancements have allowed INS to be scaled down in size for personal use without financially burdening the customer. With this in mind, the main motivation for our project was to develop a small, lightweight system of sensors that can be attached to the uniforms of soldiers, and can monitor user position in the event of lost GPS signal. With that being said, the United States Armed Forces are our primary customers. We aim to make fighting overseas safer for our soldiers by providing them with a system that can accurately determine their displacement, and assist them in returning to their home base.

System Requirements and Design Constraints:

- System (most importantly) must provide accurate displacement and heading angle measurements
- System must be small, and lightweight
    - Geared toward the military, don't want to burden to soldiers with an additional set of large, bulky equipment
- System must be low power
    - Must be run by battery so that the system is portable, and again, small and lightweight

- System needs to be able to be seamlessly integrated into the soldiers' uniforms
- System must be low-cost
    - Less than $200
- System must utilize wireless transfer of data
    - Soldier's cannot be wired into a laptop while they are on a mission
- System must be reliable
    - This device may be used to ensure the safety of the men and women in the Armed Forces. With this in mind, the system needs to be dependable

Goals:

Listed below are the Fall and Spring goals that we set for Team Inertia at the beginning of the academic year:

Fall

- Research and understand previous Dead Reckoning group's project
- Research and understand equipment that we are implementing in our Dead Reckoning system
- Configure and test sensors separately, including accelerometer, gyroscope, infrared proximity sensor, and magnetometer
    - Make adjustments to achieve minimal error in measurements
- Create program that will map user movement
- Combine sensors, and begin testing Dead Reckoning system using known walking paths
- Plot test results, and see if they agree with our expectations
    - Make adjustments to achieve minimal error in system
- Send data to laptop via XBee radios

Spring

- Combine sensors with more accurate measurement and continue testing Dead Reckoning system
- Adjust individual sensors for minimal error and measurement variability
- Incorporate various environments and paths into testing
- Package mobile device in durable, portable, water-resistant enclosure
- Finalize mapping code - make it interface with mapping application such as Google Maps
- Look to make Dead Reckoning system work in real-time
- If time allows, look into powering mobile device using alternate source (other than battery)

Schedule:

Below is a rough schedule of our work throughout the academic year:

Project Schedule		
Week	**Dates**	**Tasks**
1	**9/8 - 9/14**	• **Draft research proposal, and submit to Cotton on 9/13 by the end of the day**
3	**9/22 - 9/28**	• **Create general overview of project, block diagrams detailing operation of our Dead Reckoning device** • **Compile and finalize parts list needed for prototyping our device** • **Determine direction of the project (magnets, IR proximity sensor?)** • **Previous Dead Reckoning group code does not work -> write our own**
4	**9/29 - 10/5**	• **Connect various sensors to Arduino to figure out any problems** • **Create and work on Team Website**
5	**10/6 - 10/12**	• **Begin prototyping stride measurement system (using either magnets or IR)** • **Continue developing code to run sensors in MATLAB** • **Finish Magnet research**
6	**10/13 - 10/19**	• **Troubleshoot problems encountered when running different sensors through MATLAB** • **Test IR Sensor Accuracy** • **Calibrate Gyroscope and Accelerometer**
7	**10/20 - 10/26**	• **Create "wearable" device, work on code to detect steps** • **Prepare mid-term report, presentation**
8	**10/27 - 11/2**	• **Generate and parse data from sensors independently**

9	11/3 - 11/9	• Begin testing Magnetometer • Write code to link all sensors together including code to fire IR sensor every time a step is detected. • First full stride measurement system test in Evans Hall, begin code to plot position
10	11/10 - 11/16	• Stride measurement system improvements/troubleshooting (continued), • Continue working on code to connect all sensors, plot a walking path. • Test stride measurement system (continued)
11	11/17 - 11/23	• Prepare for end of semester presentation, continue testing, generate new maps plotting position on it
12	11/24 - 11/30	• Presentations/work on final report
13	12/1 - 12/7	• Presentations/work on final report
14	12/8 - 12/14	• Final report due to Cotton by 12/10
15	2/2 - 2/8	• Continue magnetometer testing to ensure accurate data in various environments • Pick up where we left off with testing and plotting the data from all the sensors
16	2/9 - 2/15	• Improve code to make it more efficient at detecting footsteps and obtaining accurate stride length and angle direction from magnetometer
17	2/16 - 2/22	• Incorporate magnetometer data into full system • Conduct more tests • Determine when system in inaccurate • Improve all code and wearable system
18	2/23 - 3/1	• Continue testing full system and plotting data • Research overlaying mapped data on Google maps
19	3/2 - 3/8	• Continue testing full system and plotting data • Improve tracking accuracy by adjusting code • Improve “wearable device”
20	3/9 - 3/15	• Continue testing full system and plotting data • Improve tracking accuracy by adjusting code • Improve “wearable device”
21	3/16 - 3/22	• Continue testing full system and plotting data • Improve tracking accuracy by adjusting code

		• **Improve "wearable device"**
22	**3/23 - 3/29**	• **Continue testing full system and plotting data** • **Improve tracking accuracy by adjusting code** • **Improve "wearable device"**
23	**3/30 - 4/5**	• **Continue testing full system and plotting data** • **Improve tracking accuracy by adjusting code** • **Improve "wearable device"**
24	**4/6 - 4/12**	• **Continue testing full system and plotting data** • **Improve tracking accuracy by adjusting code** • **Improve "wearable device"**
25	**4/13 - 4/19**	• **Continue testing full system and plotting data** • **Improve tracking accuracy by adjusting code** • **Improve "wearable device"**
26	**4/20 - 4/26**	• **Continue testing full system and plotting data** • **Improve tracking accuracy by adjusting code** • **Improve "wearable device"**
27	**4/27 - 5/3**	• **Continue testing full system and plotting data** • **Improve tracking accuracy by adjusting code** • **Improve "wearable device"**
28	**5/4 - 5/10**	• **Continue testing full system and plotting data** • **Improve tracking accuracy by adjusting code** • **Improve "wearable device"**
29	**5/11 - 5/17**	• **Complete final paper**
30	**5/18 - 5/24**	• **Presentation**

Approach:

The overall approach for our project was similar to the system development life-cycle (SDLC) in that it combined several different phases of system design in order to ensure that our final product was successful, reliable, and implementable in real-world scenarios. These phases included, but were not limited to: planning, implementation, testing, documentation, deployment, and maintenance. Below is an outline of our overall approach that utilized the phases mentioned above:

- Understand previous Dead Reckoning group's project approach and results by researching their documents (hardware data sheets, test data, mapping code, filter code, etc.)
    - Is their code is useful to us?
    - What mistakes did they make that we can learn from?
    - How can we take their overall project idea and make it better?

- Brainstorm overall design for Dead Reckoning device utilizing an Arduino, infrared proximity sensor, gyroscope, accelerometer, and magnetometer
    - Automated stride measurement system
        - Accelerometer acts as pedometer - triggers stride measurement
        - Infrared proximity sensor measures stride
        - Gyroscope provides relative angle of direction
    - Magnetometer-based compass
        - Provides heading
- Research infrared proximity sensor, accelerometer, gyroscope, and magnetometer to understand configuration, and raw data output of the sensors
- Wire sensors to Arduino board, and implement code to calibrate sensors and receive measurements
- Link Arduino board to MATLAB via USB for data processing
- Write program to sense and count steps from accelerometer output
    - Test code by walking with sensor, determine if the number of steps taken is the same as the program's output
    - Adjust code accordingly, retest
- Write program to determine absolute angle of direction from gyroscope data
    - Test by turning sensor and making sure angle outputs agree
    - Adjust code accordingly, retest
- Begin writing code for magnetometer
- Understand the calibration of the magnetometer code used by last year's group (SkyNet)
- Write program to measure distance using infrared proximity sensor
    - Test by comparing values output by sensor to pre-measured distances
    - Adjust code accordingly, retest
- Design stride measurement system
    - Program that triggers the infrared proximity sensor to take a measurement when a step is sensed by accelerometer
- Combine stride measurement system, gyroscope, magnetometer-based compass
    - Test using pre-determined paths
    - Do our results make sense? What do we need to do to improve?
    - Adjust hardware, and software accordingly
- Implement wireless data transfer using XBees
- Write mapping code
- Document hardware data sheets, code, test results
- Test, and continue to improve system to achieve accurate results

Challenges:

The majority of the challenges we faced were a result of not having enough experience with the equipment that we utilized in our Dead Reckoning system. However, since we had group members who were well versed in working with different types of hardware and software, we were able to overcome these problems. Documented below is a cumulative list of the challenges that we faced, or planned on facing during system design:

- Stride Measurement Design

    - Stride measurement system hasn't been attempted by any other group, the previous group relied on manual measurements of stride
    - The stride measurement itself posed a challenge for our group given that the infrared proximity sensor had to be pointed directly at the opposite foot when a step was taken in order to get an accurate measurement. With this in mind, we made sure that the device was oriented in a way that provided consistent measurements.
    - Possible lag between accelerometer step sensor, and triggering of infrared proximity sensor had the potential to cause errors in the collected data
- Combining all of our sensors
- Wireless data transfer
- Mapping our data on an application, such as Google Maps
- Implementing magnetometer-based Dead Reckoning
    - Magnetic declination
    - Tilt error
    - Calibration step
- Creating a wearable device – still a challenge!
    - Device needs to be adequately protected from collisions, moisture, sand/dirt/dust, and other environmental factors
    - Device needs to be compact, and comfortable to wear
    - Device needs to remain as still as possible despite excessive user movement

Design:

Our system design is organized into two different subsystems. The first subsystem is our "Mobile Unit" which is comprised of our Arduino Uno (powered by a 9V battery), our MPU-6050 chip, which contains our accelerometer and gyroscope, our HMC5883L chip, which contains our magnetometer, and finally, our infrared proximity sensor. The Mobile Unit is worn by the user, and is responsible for collecting data regarding the user's stride lengths, stride count, and angle of direction. The previously mentioned data is then wirelessly communicated to our second subsystem, the "Stationary Computing Unit." The Stationary Computing Unit receives and processes the data coming in from the Mobile Unit, which is used to plot the path of the user on a mapping application, such as Google Maps. Below is a block diagram of our overall system, and the equipment involved within each unit:

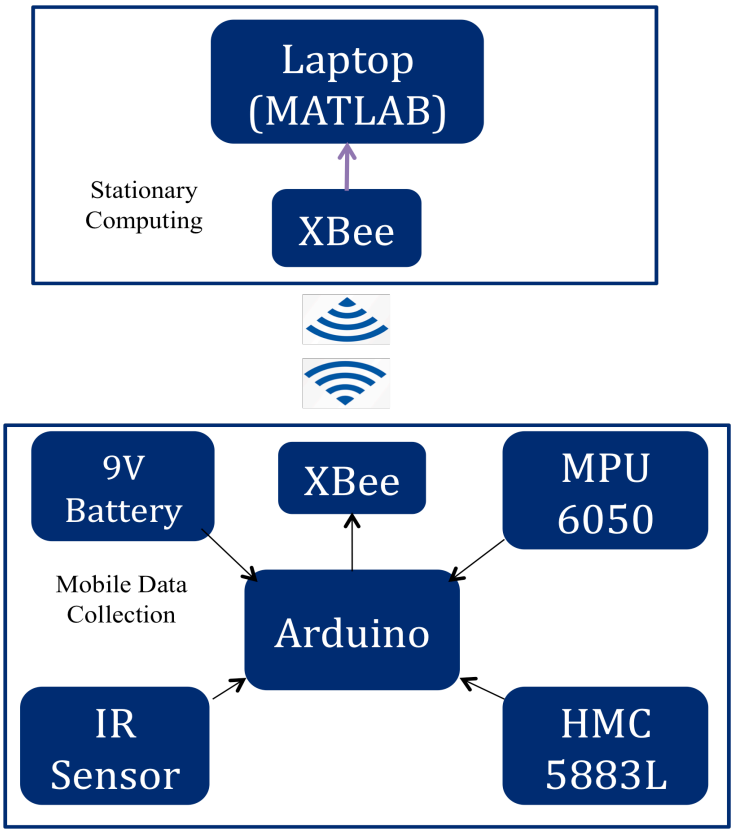

Figure 1: Overall Block Diagram of the Dead Reckoning System

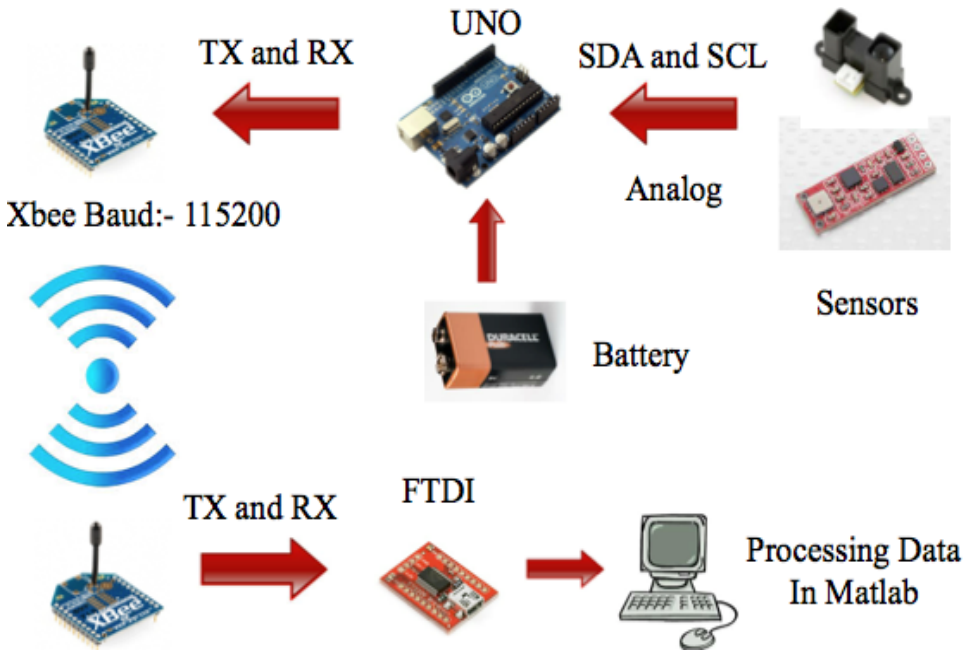

Figure 2: Setup of Hardware - Software Link

Mobile Unit:

- **Hardware**
  - 9V Battery
    - Powers Arduino Uno
  - XBee Radio
    - Wireless Data Transfer
  - MPU-6050 – Accelerometer, Gyroscope
    - Step sensor, angle of direction
  - HMC5883L – Magnetometer
    - Magnetometer-based compass
  - IR Sensor
    - Stride length measurement

Stationary Computing Unit:

- **Hardware**
  - Laptop
  - XBee Radio
    - Wireless data transfer
- **Software**
  - MATLAB
    - Data processing, and mapping

Safety Issues:

No known safety issues to date.

Project Results:

- **Mobile Components Working Separately with Stationary Computing Unit Results**
  - MATLAB is able to process accelerometer data, accurately sense, and count the amount of steps taken by the user, as demonstrated below:

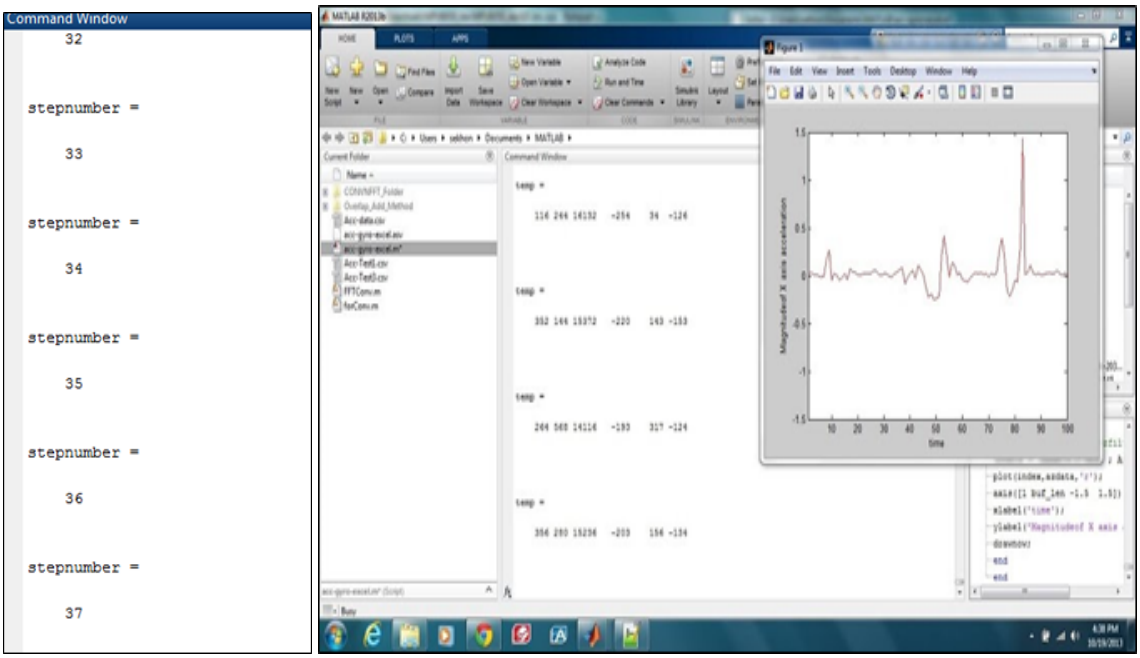

Figure 3: Step Counter, and Real Time Plot of Acceleration Data

- Note that these steps have been determined to be accurate during a "normal step" at a reasonable walking speed. If the leg that the accelerometer is attached to does not move swiftly, the step will not be detected due to a lack of spike in the acceleration data sent to MATLAB. The threshold value for step detection must be adjusted accordingly if there is an issue.

- Infrared proximity sensor provides accurate and reliable measurements within acceptable ranges, as shown in our test data below:

Set Distance (cm)	Avg. IR (cm)	
10	27.42159	10
15	22.3077	15
20	24.31524	20
25	27.55247	25
30	31.92165	30
35	36.30563	35
40	42.67238	40
45	48.44651	45
50	54.41151	50
55	60.80722	55
60	65.33413	60
65	71.98548	65
70	76.30367	70
75	81.55724	75
80	88.04238	80
85	94.91254	85
90	100.2492	90
95	104.8971	95
100	110.7472	100

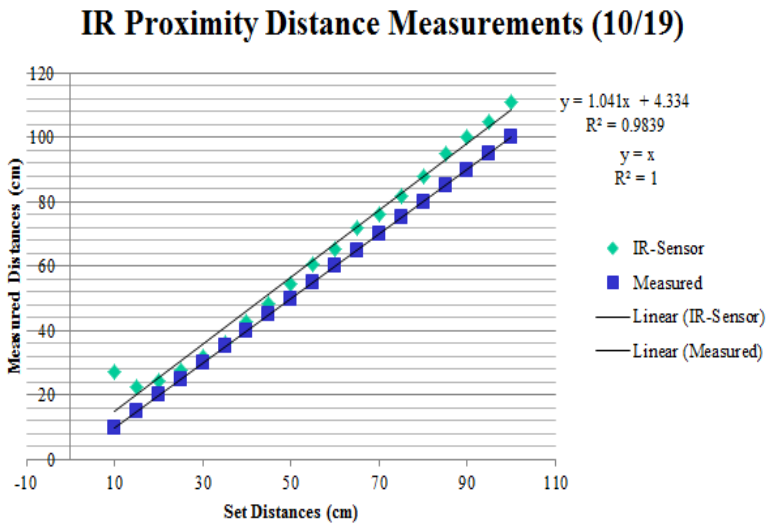

Figure 4: Infrared Proximity Sensor Test Data, and Plot

        - A new IR proximity sensor with a smaller range was purchased in the spring to accommodate the short stride lengths associated with walking. Focusing on detecting walking steps became the main priority for our system, rather than incorporating a variety of movements including side steps, backwards steps, running, etc. Tracking a variety of movements does not seem unreasonable based on the success of the current IR proximity sensor.
    - Gyroscope outputs angular velocity, which can be manipulated in MATLAB to determine absolute angle measurements.
        - The gyroscope has not been implemented into the final design due to the accuracy that has been gained through the use of the magnetometer. If one is concerned about the influence of magnetic forces in a particular area, this method would be a reliable alternative.
    - Beginning to implement the magnetometer-based compass
        - Magnetometer data allows us to output absolute angle measurements in degrees, and radians.
        - To increase the reliability of our magnetometer, and avoid the effects of magnetic distortion, a calibration step was needed. The calibration step that we used is similar to the calibration used by last year's SkyNet group. The calibration step must be completed before every use of the system, and merely consists of simple clock-wise and counter-clockwise rotations of the magnetometer. These rotations provide us with the offset values that help us account for magnetic distortion. This calibration provides us with more accurate heading measurements.
        - For the final design, fabrication techniques became an alternative to the original tilt compensation algorithm. Holding the magnetometer parallel to the surface not only eliminates the need for tilt compensation, but it allows us to minimize our heading errors. The issues associated with implementing the tilt compensation code drove us to make changes in our fabrication that would eliminate the tilt of the magnetometer. With more time, the tilt compensation code could be fixed and implemented into our program, which would allow us more freedom in placing the magnetometer, since we wouldn't have to worry about tilt. Below are the necessary calculations for the heading based on the desired tilt compensation. This method uses an accelerometer to detect the pitch and roll of the sensor, which is then utilized in the tilt compensation algorithm.

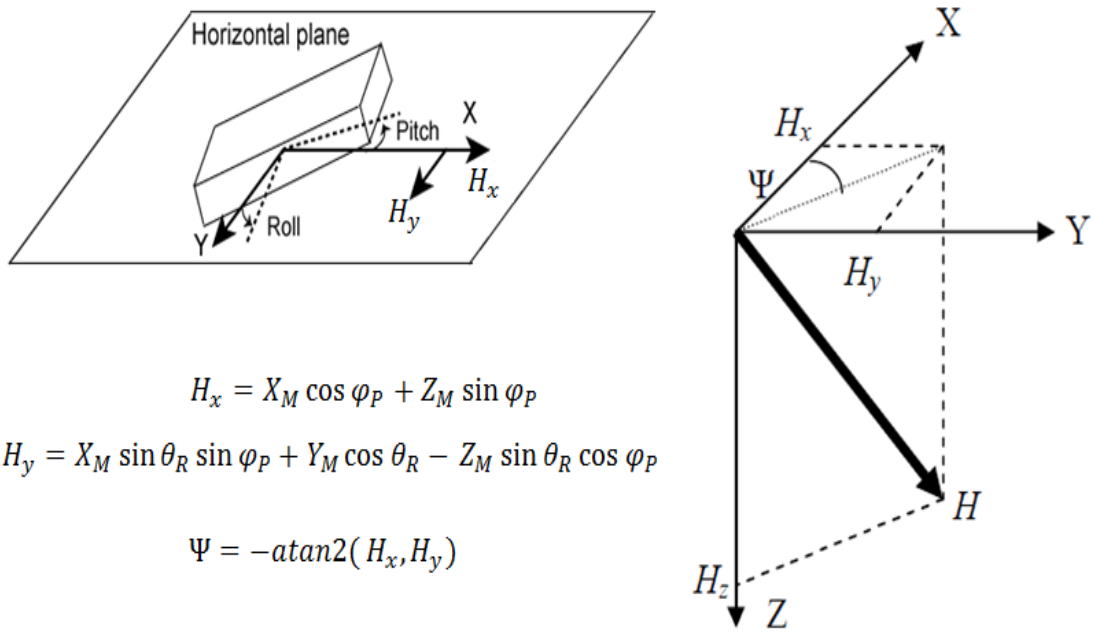

Figure 5: Magnetometer-Based Dead Reckoning System Heading Calculations

- **Overall Fall Results**

  Originally, we configured and tested each component of the Mobile Unit separately with the Stationary Computing Unit in order to ensure that we were able to take data from the accelerometer, magnetometer, and infrared proximity sensor, and process it accordingly. At this point, we began combining all of the different components of our Dead Reckoning system together, including the accelerometer-assisted pedometer, stride length measurer, and magnetometer-based compass.

  We performed our first full test in the hallways of Evans Hall. We measured a path of travel that formed an "L" shape, and had our test-subject walk this known path wearing our Dead Reckoning device. The test-subject, following the path of travel, took several steps forward, made a 90 degree turn to the right, took several steps forward, turned around, and headed back to the original starting destination. Listed below is the theoretical plot of our test-subject's path of travel, which we generated the data for ourselves, and our experimental plot, which was generated using the data we received from our sensors during the test run.

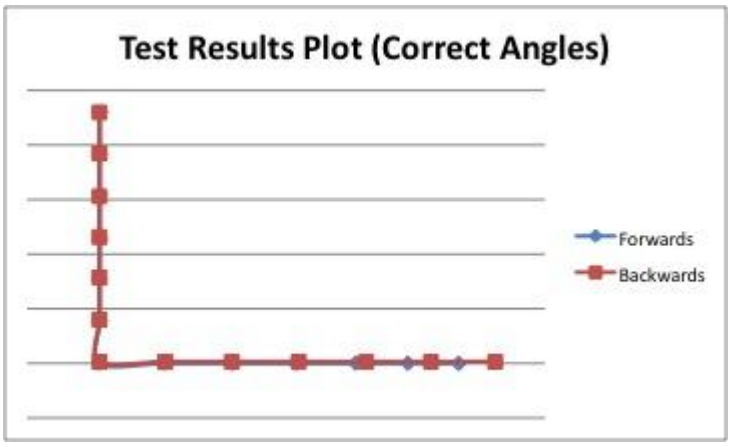

Figure 6: Theoretical Plot of Test-Subject's "L" Path of Travel

Figure 7: Experimental Plot of Test-Subject's "L" Path of Travel

In looking at the plot above, we see that there are "L" shaped paths of forwards and backwards travel, similar to the theoretical plot; however, it is obvious that absolute-angle measurement inaccuracies are the source of the discrepancies between the theoretical, and experimental data.

Overall, our first full test with all of the components working together was successful. Our Dead Reckoning system was able to provide stride length measurements within a few centimeters of the values we were expecting, and we were able to generate a plot that shows our forward and backward path of travel clearly; however, the absolute angle values from our magnetometer were not as accurate as we would have liked them to be. With this in mind, we began to adjust our device fabrication to eliminate the tilt of the magnetometer, and implement SkyNet's calibration code (Spring). Both of these changes would help to improve the accuracy of our magnetometer-based compass tremendously. Additionally, we purchased a new infrared proximity sensor with a shorter range of operation that would provide us with more accurate stride measurements in the range necessary.

- **Overall Spring Results**

  Following the test results we received in the Fall semester, we decided to take a closer look at our magnetometer before running another full system test. The majority of our problems, as seen in Figure 7, were a result of inaccurate sensor measurements, with the main source of error being our magnetometer-based compass, which provided insufficient absolute heading angle measurements. When we first implemented the magnetometer, we ran several tests that gave us the impression that it was working correctly with a set magnetic declination value. However, once we began testing our full system in Evans Hall, it became apparent that the angle measurements we were receiving were drifting with the sensor not even rotating. This was an issue that we faced even after implementing the calibration code for the sensor, which will be discussed later in this section.

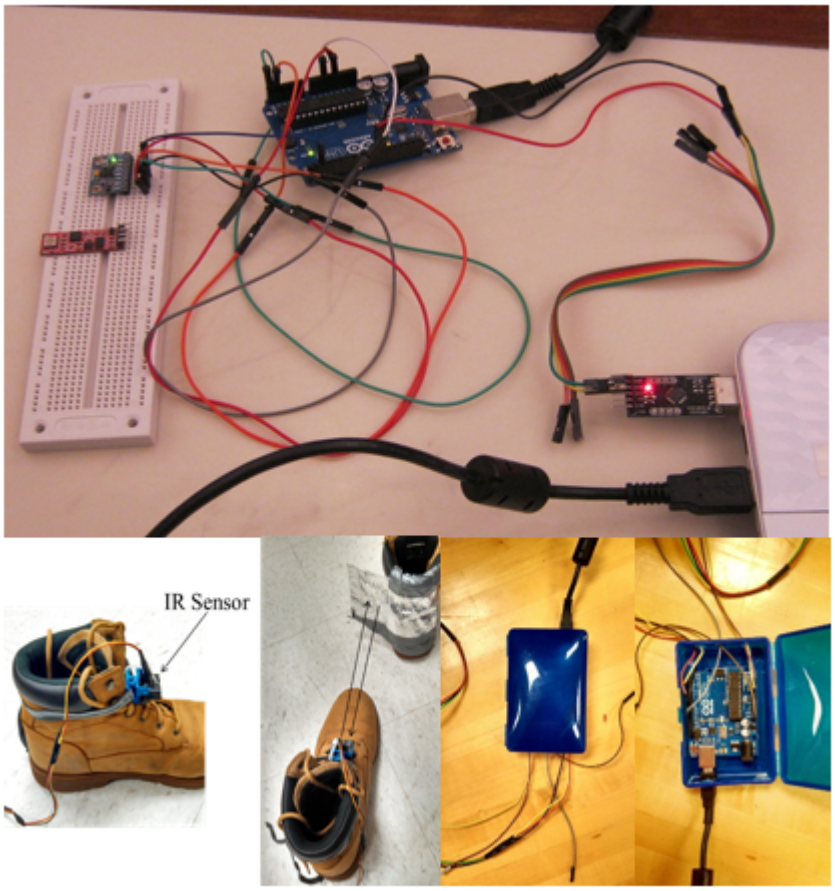

Figure 8: Initial Full System Fabrication

In addition to improving the magnetometer-based compass, improving our system fabrication also became a top priority. The system was working, and all of the components were communicating as needed, however, we needed to make some adjustments to the fabrication of our device so that it would be convenient (in terms of wearability), reliable (in terms of wiring connections), sturdy, and cleaner (in terms of presentation). Our initial system fabrication, which can be seen in Figure 8, consisted of wires of varying length branching out from the Arduino in several directions and

connecting to the necessary sensors. This initial fabrication was not very sturdy and hindered our ability to walk in a normal fashion. With this in mind, we aimed to fabricate our system in a way that made it durable, and easy to wear. The final system fabrication can be seen in Figure 9. The sensors now have the ability to be placed on each leg and/or hip with no risk of the wires disconnecting. Additionally, each sensor has their wires bundled together, which allows the user to hide wires within their pant legs. For wireless transfer capability, the XBee adapter is connected directly to the Arduino – making the switch from wired to wireless communication seamless.

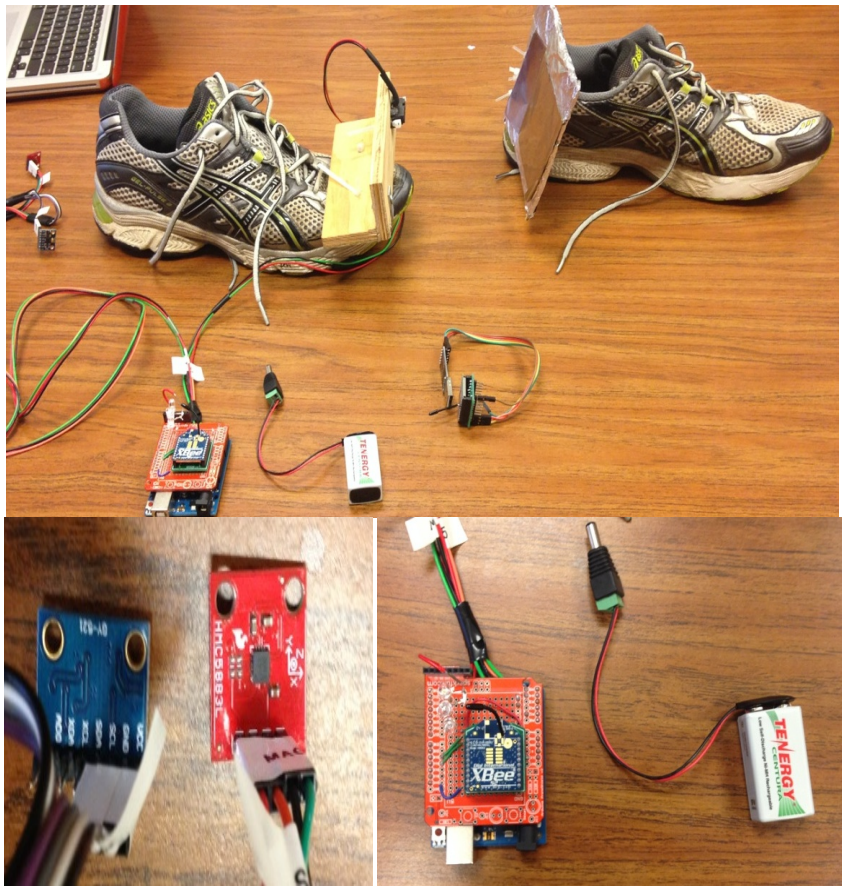

Figure 9: Final Full System Fabrication

After revamping our system fabrication, we continued to make adjustments to the magnetometer to correct for its inaccuracies. For our next test, we moved our testing outside with hopes of avoiding as much magnetic field interference as possible. Instead of testing our full system, we focused on using only the accelerometer and magnetometer, and substituted a constant step value for the IR proximity sensor data. We used the triangular path in front of Evans Hall as our test path, assuming that our test would be considered a success if we were able to accurately plot a triangle that began and ended in the exact same spot using a constant step value. However, even with the adjustments to our fabrication, which stabilized the magnetometer to keep it from tilting, our tests were still unsuccessful. Figures 10 and 11 display the first full test results from the walk around the triangle.

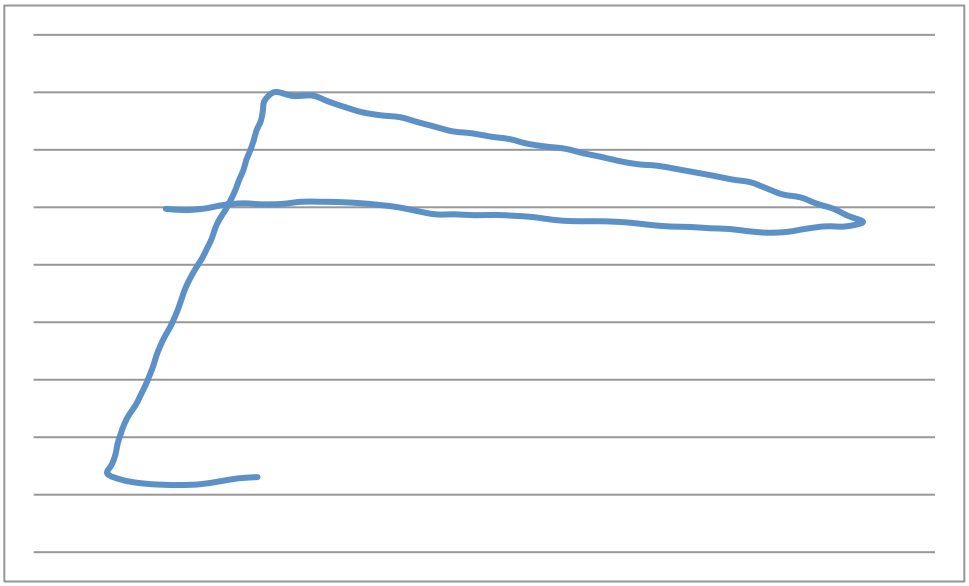

Figure 10: Experimental Plot of Test-Subject's Triangular Path of Travel

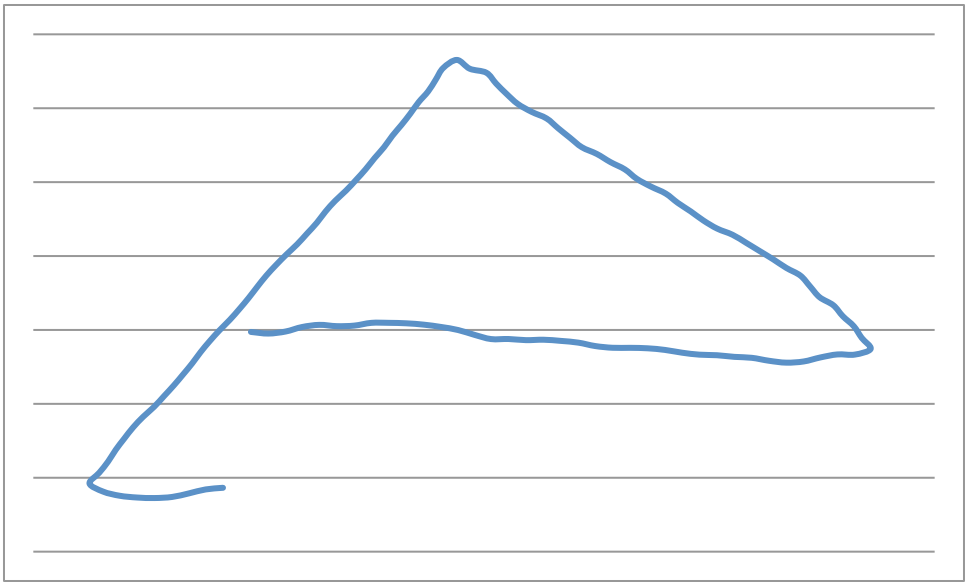

Figure 11: Altered Experimental Plot of Test-Subject's Triangular Path of Travel

As seen in Figure 10, the path recorded was far less than the ideal equilateral path traveled. This was due to both inadequate step-detection, and most importantly, insufficient magnetometer measurements. Using these results, our group wanted to see how the plot would change towards the desired result if we made our own mathematical adjustments to the data. The angles for the turns we took while walking the triangle should have been 120, 240, and 360 degrees, respectively. However, the measurements that we received were multiples of 140 degrees. With this in mind, we decided to subtract 20 degrees from every value measured by the magnetometer to bring them closer to the values we were expecting. The plot in Figure 11 contains the altered heading values to correct for magnetometer error – once again, the plot is far less than ideal, but the results

are much better than the plot in Figure 10. Based on this conclusion, we decided that our sensor was still suffering from magnetic distortion, so we began researching the calibration code that was successfully developed by one of the senior design groups from last year (SkyNet).

Once we were able to implement the calibration code into the Arduino and apply the offset values to the raw data produced by the magnetometer, the heading values seemed to be much more accurate (<5% error). However, upon performing another full system test, we discovered that we were beginning to have read/write errors due to discrepancies in the way we were sending data to our computing unit from the Arduino. The issue was resolved by making some minor adjustments to our program, which resulted in a fully functioning heading angle sensor, and overall system.

Again, rather than testing the entire system at once, we decided it would be best to run a test with just the accelerometer-assisted pedometer and the new magnetometer-based compass with calibration. Prior to the test, we improved our accelerometer-assisted pedometer by tweaking the threshold for detecting a step, and using time stamps to eliminate any steps that occurred within one second of each other (essentially eliminating the detection of multiple steps). Before implementing the infrared proximity sensor, we walked the triangular path in front of Evans Hall again, as well as the previous Dead Reckoning group's path, which started at Gore, proceeded to Memorial, and ended at Evans Hall. Using our magnetometer and accelerometer data, and a constant stride length we were able to plot a near-perfect path around the triangle, and from Gore to Memorial to Evans Hall. Initially, the stride length measurement was the main goal of our project, but with the infrared proximity sensor being our most reliable device, we knew that if we could improve our pedometer and heading angle measurements, we would be on our way to a fully functioning Dead Reckoning system. The results from the accelerometer-assisted pedometer and magnetometer-based compass system can be seen in Figure 12 and 13.

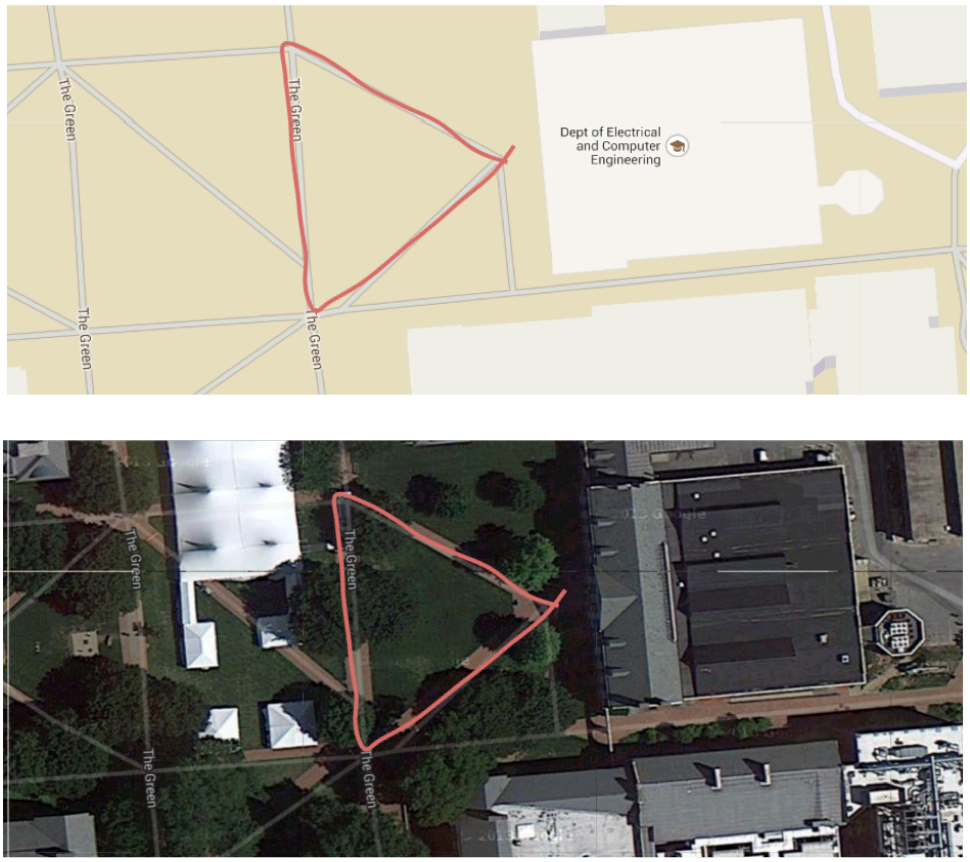

Figure 12: Accelerometer-based Pedometer & Magnetometer-based Compass – Triangle Plot

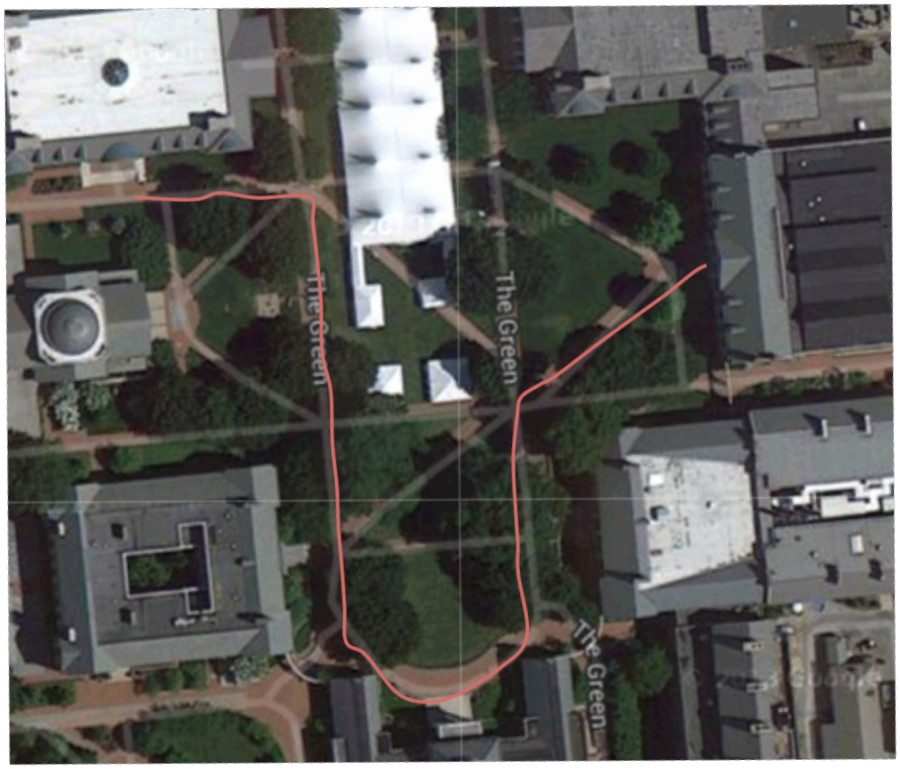

Figure 13: Accelerometer-based Pedometer & Magnetometer-based Compass – Gore Plot

Based on the results of our accelerometer-based pedometer, and magnetometer-based compass tests, we knew that we were finally ready to implement the infrared proximity sensor back into the Dead Reckoning system. In order to improve the accuracy of or displacement measurements, we placed the sensor on a mount that rested upon the toe of the shoe. The sensor was slightly offset to the left, while a reflector offset to the right was placed on the opposite foot. This setup was preferable due to the fact that it would ensure that we would yield accurate measurements consistently, and it would give us "straight-line displacement," rather than a diagonal measurement that required geometry to solve for the displacement. Below are the plots from our final full system test. Our plots were scaled and laid over the maps – there is some error in the total distance traveled determined by our system due to the fact that our system only measures the step of one foot, and doubles this measurement to add to the total displacement. In our final test, we happened to be testing during the changing of classes, resulting in a non-constant walking pattern. Therefore, if we took a step of 15 cm, and then a step of 30 cm, it would be counted as 30 cm total with our estimation, and not 45 cm. Despite this error, the measurements we received are rather accurate, as seen in Figures 14 and 15.

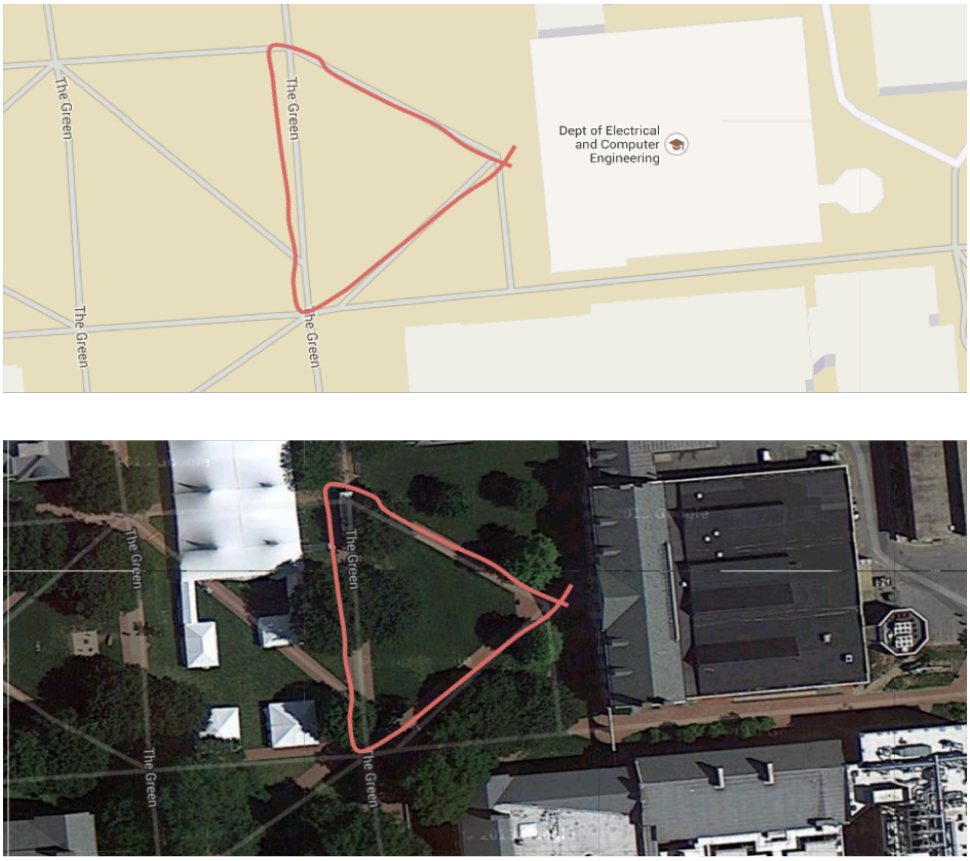

Figure 14: Final Full System Testing – Triangle Plot

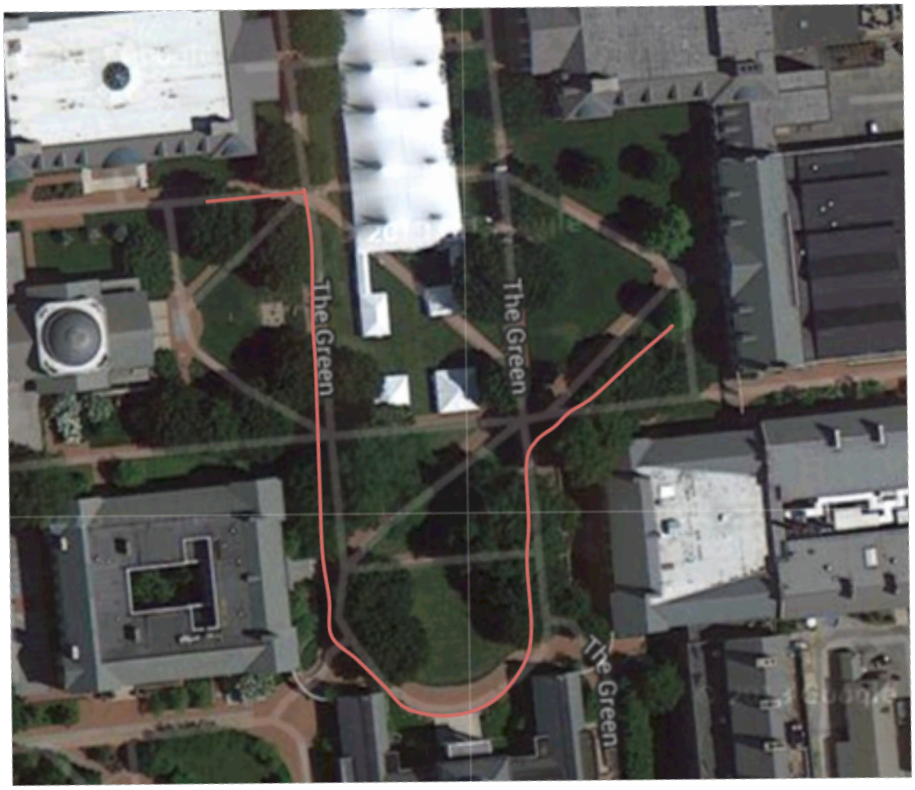

Figure 15: Final Full System Testing – Gore Plot

- **Lessons Learned**
  - None of us have had experience with Arduino boards, or any of the sensors we are using in our dead reckoning device. Through our project, we have learned how to configure, program, and integrate these components into a full system.
  - Testing is the most important aspect of system design – it reveals significant improvements that need to be made within the system.
  - Programming, and implementing XBee radios.
  - The many factors that distort heading measurement, and how to account for each of them.
  - How to analyze, and process data from our sensors using MATLAB.
  - How to break a project into smaller tasks, and then integrate them together to create a final product.
  - Overall – problem solving!

- **Recommended Future Work With More Time**
  - Implement gyroscope
  - Implement Kalman Filter to mitigate gyroscope drift
  - Replace existing IR sensor with a more accurate one

        - IR sensor we have now is good, but there are probably better ones on the market, just might be a bit expensive!
        - Gather large amount of data from IR sensor with known stride length
            - Analyze this data to determine the accuracy of the sensor, look for any consistent offsets
            - Write code to account for these known offsets, thus providing a more accurate stride length
    - Implement a tilt compensation algorithm for the magnetometer-based compass
        - Tilt is our only source of error for our heading measurements
    - Implement an IR/MPU-6050 on each foot
        - Errors in total displacement result from our estimates using 1 step
    - Continue testing, and adapting system in order to reduce error
    - Begin writing mapping code, so that we are able to take our data and plot it on a map application, such as Google Maps, in real time
    - Product to prototype
        - Create a wearable enclosure for our Mobile Unit
            - Attaches to belt
        - Integrate wiring into pants, shirt for convenience

Engineering Standards:

No engineering standards used for this project.

References:

- Kalman Filter:
    - http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it/
- Infrared Proximity Sensor:
    - http://www.instructables.com/id/Simple-IR-proximity-sensor-with-Arduino/
    - http://luckylarry.co.uk/arduino-projects/arduino-using-a-sharp-ir-sensor-for-distance-calculation/attachment/sharp-gp2y0a02-circuit/
- Accelerometer and Gyroscope:
    - http://www.instructables.com/id/Guide-to-gyro-and-accelerometer-with-Arduino-inclu/
    - http://www.youtube.com/watch?v=jfptwZSG5WA
- Mapping (Polar vs. Cartesian):
    - http://www.youtube.com/watch?v=-QNFl6l5qcs
- History of Dead Reckoning:
    - http://en.wikipedia.org/wiki/Dead_reckoning
- Magnetometer:
    - http://www.loveelectronics.co.uk/Tutorials/13/tilt-compensated-compass-arduino-tutorial
    - http://www-personal.umich.edu/~johannb/Papers/paper162.pdf
    - https://sites.google.com/site/udelskynet/home
- GPS:
    - http://www.pocketgpsworld.com/howgpsworks.php
- Overall Sensor Code

- I2cdevlib - https://github.com/jrowberg/i2cdevlib/tree/master/Arduino
    - Main source of code and help with programming sensors

Appendix:

- **Equipment Needed/Used**
    - Laptop with MATLAB
    - Arduino UNO
    - Infrared Proximity Sensor
    - MPU-6050
    - HMC5883L
    - XBees
    - 9V Battery
    - FTDI Chip

- **Budget Details**

Equipment	**Quantity**	**Cost**
Arduino UNO	1	$28.49
MPU-6050	1	$18.80
HMC5883L	1	$6.95
9V Battery	1	$3.48
XBee	2	$42.00
Infrared Proximity Sensor	1	$18.00
FTDI Chip	1	$14.95
	Cost:	$132.67