# Smart Parking Infrastructure (SPI)

Edwin Munguia, EE, Michael Skaza, EE, Alexander Donadio, EE, and Daniel Perry, EE

***Abstract*— Every day we come across the issue of having to drive to work, but the biggest problem of them all has always been parking. Worrying about where you must park? The purpose of SPI is to provide a low-cost interface that allow you to view live availability of parking spaces near your location. Growing traffic volumes, increasing air pollution, rising costs: A mobility solution cannot solve all your infrastructure challenges on its own. But it will always contribute to making your city or municipality even more modern, more attractive, and more environmentally friendly.**

## I. Introduction

Daily drivers run into the daily problem when driving to a public place, PARKING. Finding parking, especially remotely is very difficult. Time spent wasted trying to find an open spot, increased carbon dioxide emissions, and overall personal stress are all main drawbacks of not having a smart parking infrastructure in place. Our product is a sensor network that offers an application with real time analysis of open and occupied parking spaces. The system will allow you to see the real time availability of the spots in the designated destination and provide a timer/notification if you are running low on your parking meter time. Our product, SPI, will be a fully scalable solution with a robust infrastructure behind it, offering a much-needed solution to parking services.

### *A. Significance*

Wireless sensing networks have made great strides in recent years in improving and refining current systems in place, while also being incredibly diverse in their application. These systems can include numerous sensing nodes to receive data, and then subsequent actions can be acted upon that data with accurate and efficient results. Increased population density in growing communities also coincides with increased levels of road vehicle congestion, reduced mobility, business access, housing and leisure activities, as well as on-street parking accessibility. According to a recent poll, motorists spend an average of 17 hours a year searching for parking spots. This adds up to around $345 per driver in wasted time, fuel, and emissions [9]. With the use of this product, that congestion and trouble can be curtailed with precise monitoring and by allowing users to access this sensor information to better traverse a parking lot. In fact, this data that these sensor nodes are gathering can have more applications in evaluating a variety of things such as distribution of cars, reserving parking spots, billing, and more all for the benefit and ease of customer and administrative use.

### *B. Context and Existing Products*

There are already several solutions to this issue of enabling smart parking using sensor technologies, such as the Siemens Intelligent Parking Solution [5]. This technology utilizes in-pavement and overhead detectors to monitor parking spaces. The sensor uses an algorithm to calculate whether an object is present and the size of the object. Compared to our design, this design uses radar which can be expensive to build nodes out of. Also, radar can over generate information that can prove difficult to parse what is needed, especially when it is transmitted wirelessly, and with are multiple signals vying for the same space. All that data can become questionable with little wave space, too much data, and so many signals.

Next, looking at the Inrix Ultrasonic Parking Availability Technology, we see the use of ultrasonic sensors to detect vehicle presence [4]. This technology is significantly more expensive than other technologies, and the data itself is more susceptible to error, since it requires a flush surface to bounce off of and bounce back to, and cars so often are not parked well at all.

This paper will propose a method using magnetometers, and wirelessly transmitting that data in a power efficient and cost-effective manner, with real world results demonstrated by testing done in our very own university campus in real time. With such little information needed to transmit, merely the coordinates of the magnetic field surrounding each magnetometer, there will be significantly less chances for error, and an ease in transmitting unparalleled. Reducing our sensor nodes to as little current being used as possible, we will ensure a lengthy battery life to make it easy on the administrative side of things. The goal here is to supply real time information to users of an Android application of whether an individual parking spot is open, to remotely access these spots ahead of time.

### *C. Societal Impacts*

The main group of people we would target as customers would be universities, large corporate offices, or any owner where parking is a main concern. The customers/people who visit these places would benefit the most as they would know, in real time where to park. Also, the owners themselves would benefit because of increased traffic flow. Parking services might see a decrease in employment as we aim to automate parking services with our design. We made our design to be easy to implement and portable. The owner should be able to implement the system and then have the users reap the benefits. While we aim to have a 100% accuracy rating, we may

A.D. Author from Boston, MA (e-mail: emunguia@umass.edu).
E.M. Author from Providence, RI (e-mail: adonadio@umass.edu).
D.P. Author, Jr., from Tyngsborough, Ma (e-mail: dperry@umass.edu).
M.S. Author from Franklin, Ma (e-mail: mskaza@umass.edu).

experience glitches in the system at certain times. A main consequence if our system experiences one of these glitches would be the false information given to the user. This in turn may lead the user not to use the application, thus making the application useless.

### *D. Requirements Analysis and Specifications*

The requirements of our design can be summed up into two categories: Accessibility and Accuracy. A crucial component to our product is its ease of use and easy installation. We are basing our design on a simple set up and practical user interface. The second requirement, accuracy, pertains to the level of precision that we are striving for when collecting data in order to make the design feel reliable and accurate. Table 1 below will provide our specifications for our design.

| Requirement | Specification | Value |
|---|---|---|
| **Accuracy** | Presence of vehicle | > 90% |
| **Battery** | Life expectancy | At least 6 months |
| **Portable** | Application show parking status | None |
| **Weather** | Operable in all-weather conditions | -20 F to 140 F |
| **Responsive** | Latency | Every 1 min |

*Table 1: Requirements and Specifications*

## II. Design

### *A. Overview*

Now that we have explored the goals of our system, we can examine the system that we are developing to deliver these features. We will work from a high level down to explore each piece that composes our system. Our system is comprised of four main components: the enclosure sensors, the gateway, the cloud computing, and the application. A visual representation of these subsystems and their interactions involving inputs and outputs can be viewed in our block diagram in Figure 1.

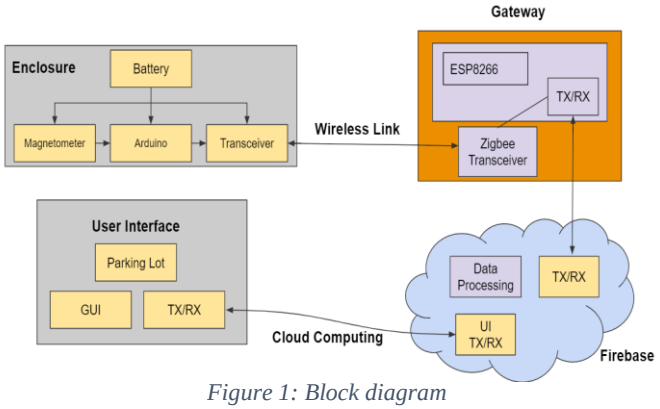


*Figure 1: Block diagram*

After converging on the idea for this system, the next step was to discuss its implementation. The question was asked: "How can we create a low-powered and low-cost device?" After conducting some research, we discovered a device called a magnetometer [10]. A magnetometer measures the strength of the magnetic field at a position in space. An example of a magnetometer is a compass, which measures the direction of Earth's magnetic field. In our project, we will utilize the magnetometer to sample coordinates of Earth's magnetic field in the X, Y, and Z direction. We needed to think of a multitude of scenarios of which our device can fail and make sure that our system would work under all these scenarios.

We were really worried about was temperature and what type of enclosure it should be in. If you refer to system specification, we want it to work under all weather conditions. We had to research an enclosure that allowed this while not interfering with our electronic system it was contained in. After some extensive research we found an IP67 rating enclosure which would not interfere with our electronics and can withstand the all sorts of weather in temperatures rating from -20 F to 140 F.

Our next concern was how are we going to detect a car accurately every time it parks over the parking spot. Our first proposed concept was to use an ultrasonic sensor [6] which uses basic principle of sound propagation and reflection by material in the ultrasonic frequency range. By applying this principle, ultrasonic sensors can still function well in conditions where light intensity is low or dark. This was considered because it was such a great sensor that was low powered and could provide great reliability. However, the downside to using such sensor is that if we park the car at an angle the wave will not be able to reflect of the car and go into the receiver. Refer to Figure 2 for a visual. This led us to find the magnetometer, a sensor which is reliable and detects the magnetic field around it allowing us to detect a vehicle presence.

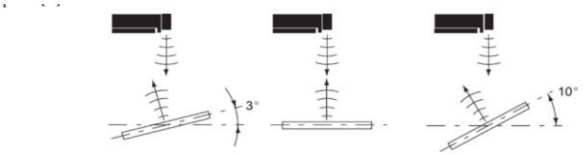


*Figure 2:Reflection of Objects using Ultrasonic Sensor*

### *B. Sensor*

In order to detect the magnetic field of the earth, we utilize the MAG3110 [1], which is a magnetometer that detects the X, Y, and Z coordinates of the earth's magnetic field. The sensor is ultra-small having dimensions of 2 mm x 2 mm x 0.85 mm, 0.4 mm pitch, in a 10-pin package [1]. It has sensitivities down to 0.10 microTesla, meaning we will able to detect earth's magnetic field (which is usually around 50 microTesla) [2]. A car can be modeled as a large, concentrated piece of metal, which disrupts the magnetic field. Once a vehicle drives over the sensor, the magnetometer senses a disruption in the magnetic field, and can perform the required action. Throughout the semester, many tests were performed to test the

magnetometer. We performed tests in open fields and parking lots to get accurate readings of earth's magnetic field when no vehicle was present. Also, tests were performed in parking lots with vehicles surrounding an empty spot.

### C. *Control and Transmission*

For our sensor network to be effective and efficient, we need it to last for a long time. In order to conserve power, we decided to only transmit our sensor data (which is the most power hungry) every time a drastic change in the magnetic field is sensed. In order to do this, we utilize deep sleep algorithms embedded in the Arduino and Xbee Series 3 transceiver [3]. This allows the network to only transmit a few times a day, because only a few changes in parking lot status happen every day. The Xbee Series 3 utilizes the Zigbee protocol which operates on the IEEE 802.15. 4 physical radio specification and operates in unlicensed bands including 2.4 GHz, 900 MHz and 868 MHz [3]. The Arduino awakes the MAG3110 everyone minute, samples the magnetic field, and if a drastic change (which is programmed to be a difference greater than 2 standard deviations of testing data) is present, then the Xbee transmits the data to the gateway.

### D. *Gateway*

The gateway consists of an ESP8266 [7] and another Xbee Series 3 transceiver. The gateway does no computing, it rather serves as a path to receive data from the sensor network to the cloud. Once again, low power modes are implemented on both the ESP8266 and the Xbee transceiver. The gateway is plugged in to an outdoor power outlet.

### E. *Cloud Computing*

Once we receive the raw XYZ measurements from the magnetometer [1] data relayed from the wireless gateway, it is then the responsibility of our cloud application to determine if the spot is currently occupied, where the server will then update the database accordingly. To see if a sensor has a car above it, our currently implementation is to see if the Z direction of the magnetic field is above a certain threshold, we obtained from average car sensor readings. Our rationale for doing this calculation on the cloud rather than the physical device was so the car detection algorithm would be easier to change in this future. These changes would affect the entire system independent of updating each individual parking sensor's firmware, which would be very difficult given a large number of sensors. Once we determine if there is a car present, we then update our database hosted on Google's Firebase with a 1 to indicate the presence of a car, and a 0 to indicate there is no car which would then be visible to the android application.

### F. *User Interface*

The final subsystem is consisting of an android application that will provide real time information about the availability of the parking spaces. Our plan was to create an easy to use application that would make finding parking more efficient and save time. Smart Parking Infrastructure was programmed in Android Studio using the programming language Java. This platform provides a lot of tools/resources to approach Android Application Development. It provides the SDK tools which are resources being provided by Android Studio to enable specific features and tools needed to make the application user friendly. Our current minimum SDK version is Android 4.1 and our target was the latest version of Android 9.0 Pie.

For out project, we used the following SDK tools:

- Android SDK Build-Tools: Used to debug, build, run, and test an Android application.
- Android Emulator: Allows us to emulate an Android device to test our program without having to upload an APK file onto an Android device. We used a phone for testing instead of an emulator.
- Android SDK Platform-Tools: Used to support the features for the current android platform
- Android SDK Tools: Downloadable component for the Android SDK that includes the complete set of development and debugging tools for the Android SDK
- Firebase: Used to establish connection to our cloud platform.

All the tools were taking into consideration when buying our phone for testing and interaction. We selected the LG Phoenix while taking into consideration the budget and our goals and decided it was the best for our purposes.

While programming the application, we ran into two major issues. How are we going to able to connect to our cloud platform and how are we going to display the parking spaces available in a user-friendly manner. In order to solve our first main issue, we discovered that Android Studio had a downloadable SDK that will allow for an easy to use implementation when coding in Android Studio. With the use of this SDK the android application will always have a secure connection to the database server if the device is connected to the internet. Figure X shows how connection to Firebase is established. The app engine in our case is Android and it is synchronized through all of our devices that are constantly updating data from our database.

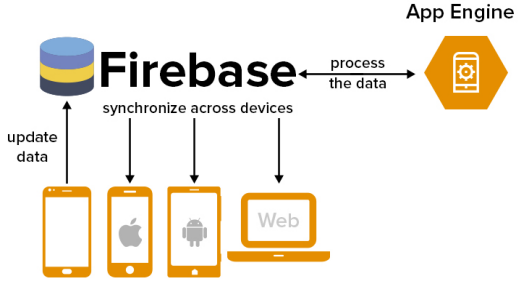


*Figure 3: Connection to cloud*

This allows us to constantly be providing the user the most accurate and recent data as possible.

The next issue is how are going to create a physical map of the parking spots available in order to show the user a top view layout of the parking lot while also showing them live data. We decided to hand create our top layout of the parking spaces using a software called AutoCAD [8]. This allowed us to create the parking lots by tracing all of them. This allowed us to easily implement how the data we receive from the cloud will change the parking spot availability. Figure 4 shows the basic user interface that we have developed so far. It is a

dynamic list that updates based of our cloud server which displays the status of the parking spots.

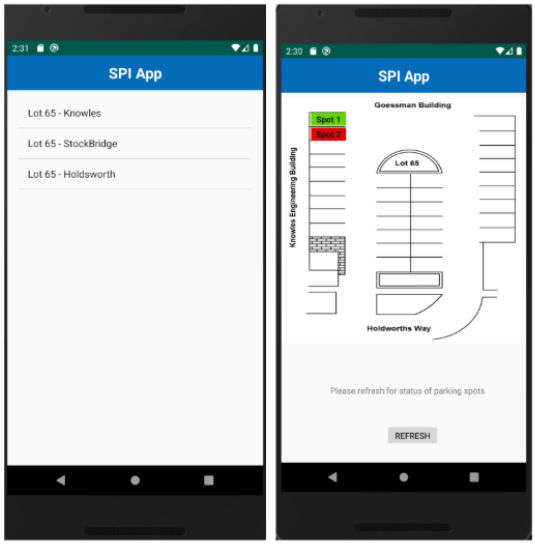


*Figure 4: User Interface displaying parking spot (Red = Taken, Green = Available)*

## III. Project Management

Progress on the project has overall been consistent. All our proposed MDR deliverables can be viewed within Table 2, as well of the completion status. All deliverable status was met to satisfaction.

| MDR Deliverables | |
|---|---|
| **Deliverable** | **Status** |
| We will show the ability to detect vehicle presence with use of a magnetometer at an accuracy rating of 90% | Completed |
| We will be able to send XYZ coordinates of the magnetometer's readings via Zigbee protocol to the receiver and ESP8266 | Completed |
| Graphical User Interface will be in basic stages, showing green as open and red as taken | Completed |
| Show basic map/layout of parking area and show the appropriate status of the parking spot as read by the magnetometer. | Completed |
| Receive relayed Sensor Data from ESP8266 over internet | Completed |
| Update Database in accordance with received sensor magnetometer reading | Completed |
| Push out requested database information to Android Application | Completed |

*Table 2: MDR Deliverables*

There is still plenty of work needed to be completed for us to be satisfied that out project is complete. Our primary goal in the software side is to continue to add additional functionalities to the cloud server and application in order to make it a true application and have a great user interface. Through the hardware side we will be creating the PCB and make the enclosure encasement that will hold it. We plan on adding more sensors so we can see the functionality of everything working together. Please look at the Gannt Chart below to look at the additional functionalities that we are plan on doing.

Team member contribution levels have not been concerning thus far. All members have contributed in some way to a key aspect of the project. Having a team composed of both Electrical Engineering with a wide knowledge of software and hardware has proved beneficial, as this project requires aptitude in facets from both fields. Communication between all group members has been consistent and meetings have been scheduled and attended by all members regularly.

## IV. The Product

### A. Product Overview

Seen below in our product sketch in Figure 5 and actual pictures of our network in Figure 6, we had the 2 nodes in the network up and running.

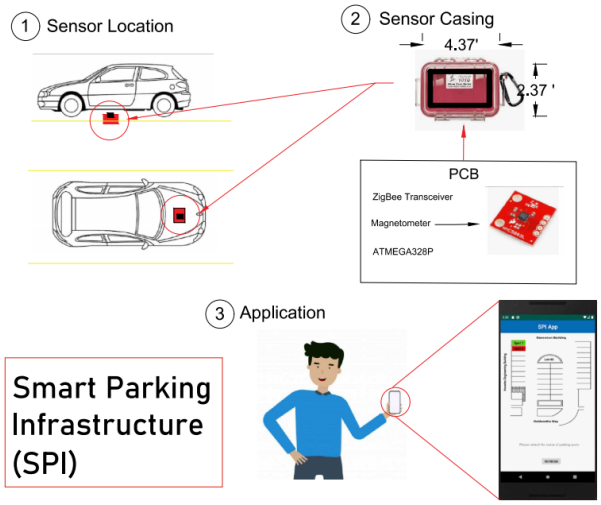


*Figure 5: Product Sketch*

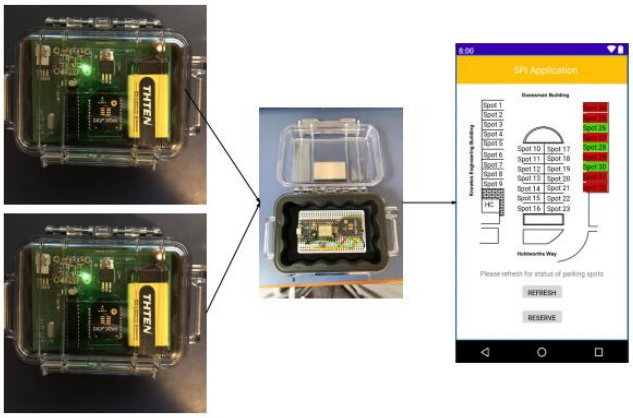


*Figure 6: Photos of Sensor Network and Application*

### *B. Electronic Hardware Component*

As shown above in Figure 6, we had completed our PCB and it was fully populated and functional. We designed it in Altium, and it consisted of many components. First, we had a lithium ion battery to power the board. This ran into some power electronics consisting of a diode, capacitors, and voltage regulators. Next, we had a the ATmega 328P microcontroller handling the magnetometer. Also, that was connected to the Xbee chip which communicated the data to the gateway. In total, we had two of these PCB's working functionally, each identical with different channel numbers. Lastly, we had the gateway hand soldered. This consisted of an ESP8266 and another Xbee transceiver. This was plugged directly into an outlet. All these sensor boards were enclosed in a Pelican 1010 Micro case, which is rated at IP67.

### *C. Product Functionality*

Because our PCB was the main hurdle we faced in the second semester, and we had it functional at CDR, our project was mainly completed. We had 2 sensor nodes running properly, both connected to the gateway, and able to update our Android application effectively. For CDR, we tested 100 times and 94% of the tests were accurate. For FPR we really wanted to get this to as close to 100% as possible and that was going to be our focus for the remaining of the semester.

### *D. Product Performance*

As mentioned above, we met our specification of at least 90% accuracy by testing at a 94% accuracy rating. The next specification was for the battery to last at least 6 months. We calculated a 7-month battery time by utilizing 1200mAh Lithium ion batteries and only transmitting data when there was a change in vehicle presence. Next, we wanted to show the application showing parking lot status, which we did through the Android application. Next, it had to be operable in all weather conditions, which we satisfied by utilizing the Pelican 1010 cases. Lastly, we wanted to update the map every 1 minute, which we satisfied through our programming of the ATmega chip.

## V. Conclusion

The prototype of the SPI device is working satisfactorily. The sensor network can reasonably detect the presence of a vehicle, and then successfully relay that through Xbee chips to an ESP Gateway. From the ESP, which is connected wirelessly to IoT, the information of the magnetometer is relayed to Google's Firebase, our 'Cloud' server. In this data base, the computation to determine whether there is a vehicle present or not is decided, and the Android application is then updated, with the display showing green for a spot that lacks a vehicle, or red if it is taken. Putting the sensor circuit together, with the magnetometer and a voltage regulator attached to the Arduino on a breadboard and connecting the ESP and the Arduino with the Xbee chips enables all this to happen. The server had to be set up and configured on Firebase, and the application was encoded, ensuring the two-way communication necessary, and with proper graphics.

For FPR we really wanted to try and increase our accuracy rating and clean up our enclosures. We wanted to make the enclosures more robust and add mounting plates.

## Acknowledgment

We would first like to thank our advisor, Professor Amir Arbabi for his advice and guidance throughout the semester. Also, a special thanks to Professor Hollot, Professor Soules, Fran Caron, and Chuck Malloch for their help with the course.

## References

[1] Freescale, Xtrinsic MAG3110 Three-Axis, Digital Magnetometer, MAG3110 datasheet, p. 1
[2] S. User, "Earth's Magnetic Field," *Earth's Magnetic Field*.
[3] Digi, Digi Xbee 3 Hardware Reference Manual, p. 13
[4] M. Braibanti, "Our Newest Innovation: Ultrasonic Sensor Parking Availability Technology," *Inrix*. Available: https://inrix.com/blog/2017/12/ultrasonic-sensor-parking-availability-technology/.
[5] "Intelligent Parking Solutions." *Siemens.com Global Website*, https://new.siemens.com/global/en/products/mobility/road-solutions/parking-solutions/intelligent-parking-solutions.html.
[6] Burnett, Roderick. "Understanding How Ultrasonic Sensors Work." *MaxBotix Inc.*, MaxBotix Inc, www.maxbotix.com/articles/how-ultrasonic-sensors-work.html.
[7] "ESP8266 Overview: Espressif Systems." *ESP8266 Overview | Espressif Systems*
[8] "Uses of AutoCAD: Basic Concepts About AutoCAD." *EDUCBA*, 12 Dec. 2019, www.educba.com/uses-of-autocad/.html
[9] K. McCoy, "Drivers spend an average of 17 hours a year searching for parking spots," *USA Today*, 13-Jul-2017.

[Online]. Available: https://www.usatoday.com/story/money/2017/07/12/parking-pain-causes-financial-and-personal-strain/467637001/.
[10] "Magnetometer," *Wikipedia*, 22-Jan-2020. [Online]. Available: https://en.wikipedia.org/wiki/Magnetometer.
[11] Microchip, ATmega48A/PA/88A/PA/168A/PA/328/P, megaAVR Data Sheet

## APPENDIX

### A. *Design Alternatives*

Originally, the design we determined called for a microcontroller connected wirelessly to a Pi, which in turn was connected to a cloud service, Amazon Web Services provider, or AWS. For the prototype, we decided that the microcontroller should be an Arduino for ease of use, and to program and work with easily. As we move forward, the Arduino will be swapped out with just an Atmega328P chip to ensure maximum power savings [11]. We decided on using Xbee chips using Zigbee protocol rather than normal Wi-Fi since it draws less current, resulting in a lower power consumption. In a similar vein, we also decided on using an ESP chip rather than a Pi since they effectively would do the same thing, act as a conduit between the nodes and the Cloud, but the ESP draws significantly less current. AWS proved to be more difficult to deal with than Google Firebase, and it allowed better and easier connection to use Firebase with an Android application, as they align more than with Amazon.

For the sensor, we considered using an ultrasonic sensor, rather than a magnetometer. Ultrasonic sensors however are significantly more expensive, require more power, and can potentially result in more error due to requiring bouncing off flush surfaces. Magnetometers are much cheaper to make and for the purposes of detecting vehicles, much more reliable. The magnetometer we chose also requires a very low amount of current and can idle at even lower amounts. Cheap, power efficient, and reliable are all valued in our project.

### B. *Technical Standards*

In our project, the main standard we followed was the ZigBee/IEEE 802.15. 4 protocol. This is a specification that is used for wireless networking to create low power digital radios for low power needs. Next, we meet IP67 rating by enclosing our product in the Pelican 1010 Micro case. IP67 rating refers to the Ingress Protection Code, IEC standard 60529, means that it is both dust tight, and can be immersed in water, up to 1 meter for 30 minutes. This standard was important to meet because we wanted our product to be suited for all weather conditions. Lastly, we utilized the Google Firebase platform and followed the JSON protocol which is a data interchange format. This was compatible between our transceivers and the cloud communication. Overall, we followed many standards and protocols to create a universal, functional product.

### C. *Testing Methods*

For the sensor network, many tests were performed. The first tests performed was simple X, Y, and Z magnetometer readings outside, with no vehicles present. We tested over 100 of these scenarios to determine an accurate threshold value for an empty parking spot. Below is data we gathered from one of those days of testing. Also, it is important to note that we varied time of day, and weather conditions when performing this threshold testing.

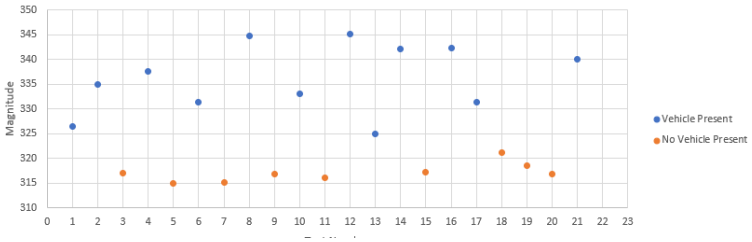


*Figure 7: Testing data of vacant and taken parking spots*

We decided to make our threshold value 317 microTesla, which was calculated by taking the standard deviation of all our data when a vehicle was present. In order to account for unwanted noise, we added a buffer of +/- 2% of this standard deviation. This means that if the magnetometer read 317 microTesla +/- 2%, then no vehicle is present. After multiple tests, this algorithm satisfied our design specifications of over a 90% accuracy rating, so we have met our requirements.

Next, we needed to test the Xbee underneath a vehicle, and the distance it was able to transmit. Because we have these sensors underneath a vehicle, there was concern if there would be too much interference during transmission. To test this, we simply setup our gateway Xbee by the parking lot and placed another Xbee with an Arduino underneath a vehicle. The Xbee underneath a vehicle simply sent a command to light up an LED on the gateway. We tested this under many vehicles and were able to do so successfully at ranges over 200 feet. Lastly, we needed to implement and the entire system. First, we used Google Firebase to see updates that were sent by the sensor. Then, once the app was finished, we tested the whole system 110 times, under different conditions. Things like time of day, temperature, location of parking spot, and vehicle driven over sensor all varied. We were able to get an accuracy of 92%, which met out initial specifications.

### D. *Team Organization*

Mike and Alex were responsible for the hardware portion of the project. This includes creating the sensor network, doing the threshold and magnetic field testing, and creating and coding the gateway. Mike has been mainly the group leader, sending out emails, and organizing the reviews. Mike and Alex have been communicating effectively and efficiently and need to work in tandem with Dan. Dan is responsible for the cloud computing side of things and also responsible for creating the website. Dan works with Mike and Alex as he receives data from the gateway and make computations on that data. Dan serves as the backend for Edwin, who created the Android application. Dan and Edwin must work very closely as Dan's information is sent directly to Edwin, where he must update the application accordingly. The team has not really had any breakdowns in communication as things are going smoothly so far.

*E. Beyond the Classroom*

For the hardware portion of this project, testing and debugging have been some of the most important skills gained so far. Because our project consists of something that needed to be tested to gain baseline values, we needed to get enough data to be statistically significant. Working with the Xbee's has proved to be challenging but having many videos and tutorials online to help has been quite helpful. Although most videos will not give you the exact answer, there is great documentation to use and build off. Our team had one experience where our network suddenly stopped working. We were able to use our debugging skills and quickly pinpoint the problem to the fact that we weren't transmitting information. Classes like ECE 353 and ECE 324 really helped us in debugging and testing strategies. In the cloud portion of the we managed to learn the basics and some complicated features of managing the application and the data coming from all the sensors. We learned how to create administrator features and user features to make the application be more user friendly for both parties. While developing the cloud on firebase we managed to use a lot resources like Google's cloud book that allowed us to do a lot of features included the cloud that were beneficial to the application and the data being transmitted from the sensors.

For the application part we learned how to develop an application from scratch. We also learned how majority of applications communicate to the cloud servers and can send and retrieve the data. We got to apply some of the technical skills we got from school and apply them with skills we had to learn in order to complete the application. We managed to use skills learned from Intro to Programming and Data Structures to help us code the entire application. We had a multitude of useful resources like the Internet and the use of multiple professors that have experience in creating applications. This allowed to make connections that will be useful in our life and future careers as professionals.