LightLength: Measuring Distances using Electromagnetic Waves

By Daniel T. Simpson

Table of Contents

Abstract

In the field of engineering, maximizing efficiency is crucial. Most methods for measuring distances are inefficient as they vary in precision, in range, and in what they can measure. As a result, engineers can spend more time managing multiple tools than finding measurements. In addition, engineers can spend $100 to $5000 for a set of high precision tools in workshops or factories. LightLength is a new attempt at measuring distances wirelessly. Radio waves have become an increasingly popular method for measuring distance. However, prior attempts at utilizing radio waves for measurement have been imprecise with an average variation of approximately 90 meters. LightLength can find the distance between two points by miniaturizing GPS concepts and having multiple terminals track the distance between each other. Within each terminal is either a transmitter and receiver, that sends a pulse and receives a pulse respectively. When using LightLength, two terminals can be placed at the ends of an object or measurement and can find the distance in between. Currently, LightLength can measure distances with a variation of $1.421*10^{8}$ meters.

Literature Review

Introduction

When developing a new product designed to solve modern problems, engineers utilize the engineering design process. However, it is crucial to complete a product as soon as possible and maintain efficient use of time to hasten progress. Currently, one of the many issues that decreases time efficiency is the traditional method for measuring the distance between two points. When creating a prototype, it is important to keep track of the size of parts, especially when adding new features, testing certain designs, or performing a quality check on newly manufactured parts. Yet, current engineering workshops typically have multiple measurement tools design for different parts. For example, measuring an object with a concave shape requires calipers with relatively large teeth, while a long piece of plywood requires a measuring tape. These measurements both rely on the distance between two points but were found with extremely different tools. If these measurements could be done with a single product, the amount of time wasted, and the number of tools needed would decrease.

Several technologies could be used to create a single tool that can measure the distance between any two points. Wireless communication has become extremely popular over the recent decades empowering technologies such as radio stations, television broadcasting, and, most recently, Global Positioning Systems (GPS). For instance, a GPS could be miniaturized to the size of a room with several transmitters tracking the location of two receivers. These receivers would represent the two points of a measurement. To understand how such a system is possible, it is important to know the basics of waves, electrical engineering, and modern applications of wireless technologies.

Wave Basics

From sound to the rays of light, waves are everywhere. There are two main types of waves: mechanical waves and electromagnetic waves (Fullerton, D. 2012). Mechanical waves include sound and seismic activity and usually require some medium to travel across (Fullerton, D. 2012). Electromagnetic waves (EM waves) are waves that can travel in a vacuum and can travel at the speed of light (3.0×10^8 m/s) (Fullerton, D. 2012). Examples include UV light, x-rays, microwaves, and radio waves (Fullerton, D. 2012). EM waves are what drive wireless communication, and since they do not require a medium, can travel at relatively instantaneous speeds, can span large distances, and can encode data within its structure (Joseph J. Carr 2012). EM waves are also known as transverse waves, meaning they structurally resemble sine waves where a path is drawn up and down along a plane (Fullerton, D. 2012). The height from the baseline to the peak of one wave is commonly referred to as the amplitude (Fullerton, D. 2012). The rate is the frequency which is the number of peaks within a certain distance, or wavelength which can be measured as wavelength over time (λ/t), represented in Hz (Fullerton, D. 2012). Thus, if the frequency was multiplied by the time it took for a wave to travel a certain distance, the wavelength can be found. Alternatively, the time can be multiplied by the speed of light, giving the total distance traveled. Additionally, the electromagnetic wave type can be determined by the frequency. Higher frequencies greater than 7.5×10^{14} Hz make up ultraviolet light, x-rays, and gamma rays while lower frequencies less than 4.3×10^{14} Hz make up infrared light, microwaves, and radio waves (*The Electromagnetic Spectrum* (n.d.)). Radio waves are most commonly used for wireless communication (Joseph J. Carr 2012). However, radio waves take time to travel some distance commonly referred to as the Radio Frequency Time of Flight (RF ToF). In order to find the distance a radio wave traveled, the RF ToF is multiplied by the speed

of light. This means that the distance between two points can be found if each point was a node in a wireless communication system.

GPS

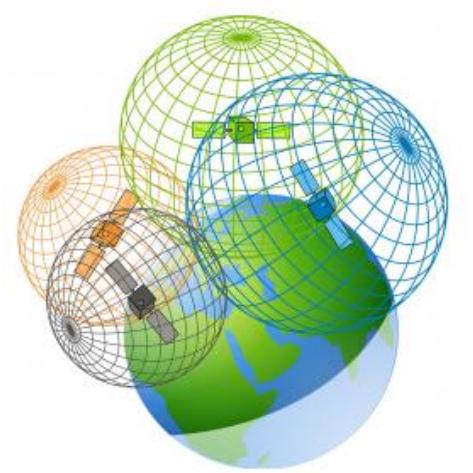


Figure 1 - Diagram of how a GPS utilizes trilateration (Trilateration vs Triangulation - How GPS Receivers Work. (2018, February 23)).

Currently, the Global Positioning System (GPS) enables many modern-day applications. It can find the location of any device that has a receiver in 3D space and with relatively high accuracy. Structurally, the GPS consists of 24 satellites orbiting around the Earth, each with a precise atomic clock, communicating with receivers on the ground (Thompson B., Richard. 1998). When a receiver is in range and requests a GPS-based service, it will receive a signal from four satellites and calculate the distance between each satellite and itself by multiplying the time it took for the radio wave to travel by the speed of light (Zahradnik (n.d.)). Typically, this RF ToF is found by subtracting the time the signal was received from the time the atomic clock initially sent the signal. When this process is repeated two or three more times with a total of three or four satellites, the exact location of the receiver can be found (*Control Segment*. (n.d.)). The process behind calculating the receiver's position is trilateration where each satellite is the center of a sphere in space whose common intersection gives the location of the receiver (Zahradnik (n.d.)). This system is easily replicable using smaller transmitters and receivers with a built-in quartz clock. More specifically, with a transceiver, a signal can be transmitted, received, and timed, similar to a GPS. This means that several transceivers can be connected in a similar fashion to a GPS and can find the location between two points wirelessly (Iforce2d). However, the quartz

clock on smaller transceiver is not as precise as an atomic clock on a satellite, thus limiting precision (Iforce2d).

Electronics Basics

Electromagnetic frequencies can be generated using electrical components. Thus, it is important to know how to design and incorporate circuits to control EM waves. Electricity, the passing of electrons through some medium, has two categories: alternating current and direct current (CodeNMore 2017). Alternating current occurs when electrons pass either direction throughout a conductive mass (CodeNMore 2017). Direct current is when electrons pass in one direction, from one point to another, across some conductive material, and it has three central measurements (CodeNMore 2017). Voltage (V) is the potential electrical energy some object has or the number of electrons available within a certain volume (CodeNMore 2017). Current (I) is the number of electrons flowing throughout a wire (CodeNMore 2017). Resistance (Ω or R) is the degree an object can resist the flow of electrons (CodeNMore 2017). Components throughout a circuit can change or control the voltage, current, or resistance as electricity flows within the circuit (CodeNMore 2017). Such components include LEDs (Light Emitting Diodes), switches, push buttons, potentiometers, capacitors, and relays (CodeNMore 2017). For wireless communication systems, there are three main communication protocols that are common in a system's circuitry. These protocols include UART (Universal Asynchronous Receiver/Transmitter), SPI (Serial Peripheral Interface), and I^2C (Inter-integrated circuit) (Arduino Communication Protocols Tutorial. (2018, June 28)). UART simply sends packages of data between two terminals that can communicate back and forth. SPI can communicate with more than two devices and utilizes clock synchronization in order have faster connections I^2C has the same abilities as SPI but is less complicated to connect to other components (Arduino Communication Protocols Tutorial. (2018, June 28)).

Past Methods for Finding RF ToF

Orthogonal Frequency Division Multiplexing (OFDM)

Researchers at Motorola Solutions Inc. in 2006 developed a way to measure distances wirelessly through multiple components within a receiver. This method only depends on one receiver having to calculate a distance rather than having two clocks and two terminals keep each other in synch as seen in a GPS (Luzzatto 2006). The method itself involves a frequency being sent across some distance to a receiver. This receiver then finds the shortest distance between the transmitter and one of its multiple components and declares that distance as the final measurement. The distance was found through various methods including Orthogonal Frequency Division Multiplexing (OFDM), which allows more data to be transferred across a 5 GHz frequency than Bluetooth Frequency Hopping (FH) Modulation (Luzzatto 2006).

Network Localization

Researchers at UC Berkeley in 2006 tested a way to measure RF ToF in environments with multiple obstructions. The team used nodes, a transmitter or receiver, to utilize what is called “mesh networking” (Lanzisera 2006). Each node would communicate a signal to each other to determine its location relative to other nodes (Lanzisera 2006). More specifically, any two nodes synchronize their clocks (similar to a GPS) and then start to communicate data about their RF ToF. This communication creates a network of the locations of all nodes (Lanzisera 2006). If a node was disrupted, its relative location to other nodes would change (Lanzisera 2006). In the end, the method was accurate from 1 to 10 meters and served better above ground than underground, where interference was higher (Lanzisera 2006).

Delaunay Triangulation

In the military, Unmanned Aerial Vehicles are often used to offer tactical views of an environment or as a replacement for humans when environments are too dangerous (Qu Y., Tian Q. 2010). These vehicles rely heavily on a GPS to map their location and when one of the

UAV's break, it is susceptible to being lost (Qu Y., Tian Q. 2010). To avoid this problem researchers thought of a way to approximate the location of a UAV with a disabled GPS through complex mathematics such as Delaunay triangulation, Voronoi diagrams, 2-D Mapping, and the Taylor Series (Qu Y., Tian Q. 2010). The researchers used data from the nearest location the vehicle was last spotted and other vehicles nearby to approximate the broken UAV's location (Qu Y., Tian Q. 2010).

Asynchronous RF ToF Measurements

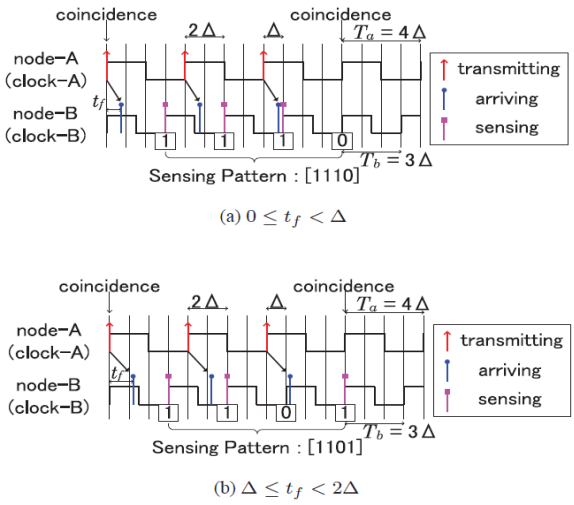


Figure 2 - The following diagram shows the synchronization between a transmitter and a receiver within a wireless communication system (Saigo Y. et al., 2012).

When timers of a transceiver are synchronized, the terminals of a wireless system can find the difference between the time when the signal was first sent and when it was first received. It is this time difference that allows for the system to find the RF ToF (Saigo Y. et al., 2012). However, the process of sending data between two terminals is complicated and consumes large amounts of power (Saigo Y. et al., 2012).There are two ways to measure RF ToF: one is two-way-ranging (where a transmitter and a receiver are in communication) without clock synchronization, and the other is one-way-ranging (where a transmitter and a receiver are not in communication) with complex clock synchronization (Saigo Y. et al., 2012). The problem is that the former gives low-precision measurements, without clock synchronization while the latter gives high-precision measurements, but with clock synchronization (Saigo Y. et al., 2012). Japanese researchers in 2012 found a way to measure RF ToF via two-way-ranging, and without clock synchronization. This new method involves tracking the rate at which signals are received

and comparing them to an internal clock. The clocks are then synchronized after the data has been sent, allowing for high precision and low power consumption (Saigo Y. et al., 2012).

Past products for Measuring Distances Wirelessly

Smartphone AR Applications

Recently, smartphones have been equipped with multiple cameras that enable phones to capture images while sensing the depth of a 3D space. In order to measure distances without the use of multiple tools, developers have made several apps such as *Air Measure*, *AR MeasureKit*, and *Tape Measure* (Matthews, M. (2018, September 19)). However, these measurements are imprecise as they can vary by about 1.5 inches in small, isolated spaces.

FARO Arm

When in an assembly line, parts are typically checked to make sure they comply with initial designs. FARO Arm is a product that involves two robotic arms that are extremely sensitive to touch (FaroArm®. (n.d.)). This means that the end of two arms can represent points of a measurement. The moment these points are touched, the arms stop and record their location relative to each other, thus finding a measurement (FaroArm®. (n.d.)). However, these arms can be expensive of up to $10,000 and require training to operate (Maintenance. (n.d.).; FARO Technologies, Inc. (2018, August 14)).

Ultrasonic and Laser Measurements

Besides radio waves, sound and light can be used to measure distances. Modern products utilize sound or light by sending a wave across some medium and having it return once reflecting off of some object (2-IN-1 50 FOOT LASER TAPE MEASURE WITH DIGITAL DISPLAY. (n.d.).; Ultrasonic Distance Meter with Laser Pointer. (n.d.)). These measurements are extremely reliable and precise. However, they are limited as sound and light cannot travel within an object. This means that ultrasonic-based and laser-based measurement tools are limited to measuring the distance from the device to some object across an open space.

Conclusion

Measuring distances wirelessly could promote efficiency in engineering workshops where parts are machined frequently. However, measuring these distances requires utilizing radio frequencies and controlling those frequencies with electrical components. Radio frequencies are crucial for their ability to travel linearly and measure distances by simply multiplying the speed of light by the time the radio frequency has traveled. Clocks, typically utilizing quartz crystals, could synchronize and send their data, measuring the time of a radio frequency. Other electrical components can control this frequency and perform the necessary calculations to measure a signal distance. With several terminals in action, and several distances measured about a specific terminal, several spheres could be modeled in a 3D space, and the intersection of these modeled spheres would determine the location of two points. The distance between these two points could then make up a single measurement with a fully operable device.

Plan

Engineering Problem Statement:

In engineering workshops, where people measure various parts, there is a need for time efficiency. Currently, efficiency decreases when utilizing multiple measurement tools that are costly, limited, and vary in precision.

Engineering Goal:

The goal of this project is to engineer a better way to quickly, efficiently, precisely, and accurately measure the distance between two points in dynamic measuring situations with a single product.

Materials

List of materials along with quantities, costs, and sources			
Title	Quantity	Cost	Source
Arduino Pro Mini 328 - 3.3V/8MHz	2	$25.98	Amazon.com
Arducam PRO Mini Atmega328 5V/16MHz 328	2	$14.98	Amazon.com
SparkFun FTDI Basic Breakout	2	$31.90	Amazon.com
Makerfire 10pcs Arduino NRF24L01+ 2.4GHz Wireless RF Transceiver Module	1	$11.98	Amazon.com
TinySine Micro USB FTDI Basic Breakout Module for Arduino 3.3V/5V	2	$19.90	Amazon.com
UCEC XY-MK-5V / XY-FST 433Mhz Rf Transmitter and Receiver Module Link Kit	1	$10.55	Amazon.com
Hook up Wire Kit (Stranded Wire Kit) 22 Gauge	1	$15.95	Amazon.com
WYCTIN Helping Hand with Magnifying Glass	1	$12.99	Amazon.com
Anbes Soldering Iron Kit	1	$29.99	Amazon.com
DAOKI 10pcs 433MHz antenna Helical Spiral Spring 5mm	1	$5.47	Amazon.com
Breadboard	1	N/A	Mr. Pawel Loven
Arduino Uno	1	N/A	Mr. Pawel Loven
Female-to-Male Jumper Wires	14	N/A	Mr. Pawel Loven
USB Mini Type B to USB Type A cable	2	N/A	Preowned
Pin Headers	40	N/A	Preowned
Personal Computer	1	N/A	Preowned
5x5x1 inch plank of wood	1	N/A	Preowned
Bandsaw	1	N/A	Preowned
High-Precision Calipers	1	N/A	Preowned
Straight-Edge Ruler	1	N/A	Preowned
Sand Paper	1	N/A	Preowned

Methodology

A staircase-like testing block was designed on a 12.7 cm x 12.7 cm x 2.54 cm piece of wood where each step decreased by 2.54 cm (Appendix A). The outline of the staircase was cut using a bandsaw and the edges of the wood were sanded using sandpaper to avoid splinters and sharp edges. While cutting and sanding the block, proper protective equipment was worn. The length of each “step” was measured from the edge itself to the back of the staircase. Using a high-precision caliper, the length of each step was found to the ten thousandths place and recorded on

the wood of the testing block using a sharpie. The first *LightLength* prototype (P-1) was made by cutting six pieces of 22 gauge hook up wire with a length of about 10 cm each. About 0.25 cm of insulation were cut from the ends of each wire. Using a WYCTIN Helping Hand, a soldering iron at a temperature of 400°C, and solder, the ends of each wire were soldered to the width of a 3.3V Arduino Pro Mini. The remaining ends of the wires were soldered to a SparkFun FTDI Breakout based off the prototype's schematic (Appendix A). Two rows of 12 pin headers were placed along the lengths of the Arduino Pro Mini. The Arduino was then connected to a breadboard with its width along rows d-h, and length across columns 1-13. Then, using male-to-female jumper wires, an NRF24L01+ transceiver was wired to the breadboard according to the prototype's schematic where pin 1 was connected to GND, pin 2 to VCC, 3 to 7, 4 to 8, 5 to 13, 6 to 11, and 7 to 12 (Appendix A). The entire process was then repeated with another Arduino, six wires, and FTDI breakout. Seven 22 gauge hook up wires were cut with a length of approximately 0.25 cm and were soldered to the second Arduino and to a second NRF24L01+ with the same connections as the first terminal. With two terminals complete, each was connected from the FTDI breakout to a personal computer through a USB Mini Type B to USB Type A cable. The Arduino IDE was then used to upload code called "Getting Started" written by GitHub user TMRh20 (Appendix B). The role was changed for one of the two files of the code to have one terminal be a transmitter and the other a receiver. The code was then uploaded to each Arduino and testing began. Testing involved placing the flat edge of each terminal of a prototype against the edges of a step. The Arduino would then record the time it took for a radio wave to travel some distance in microseconds and displayed the time (Rf ToF) in the serial monitor. The RF ToF's were then recorded for 250 data points for each step of the testing block.

Each RF ToF was then processed through the following equation to find the total distance a radio frequency traveled where *d* is the distance, *t* is the Rf ToF in seconds, and *c* is the speed of light:

$$d = tc$$

The experimental measurements were then compared to the actual measurements of each step using a one-sample t-test. A percent error was also found, and the prototype was then analyzed for improvements. A single terminal of a second prototype was made using a 5V Arduino Pro Mini, nine 10 cm long 22-gauge hook up wire, and a transmitter or receiver. Nine 10 cm long 22-gauge hook up wire and 0.25 cm of insulation from each end of each wire were cut. Using a WYCTIN Helping Hand, a soldering iron at a temperature of 400°C, and solder, the ends of each wire were soldered to the Arduino, 5V FTDI basic breakout, or to a transmitter or receiver according to the prototype's schematic (Appendix A). The process behind creating a first terminal was repeated with a remaining transmitter or receiver for a second terminal. In the end, one terminal contained a transmitter and the other a receiver. A helical spiral spring antenna was soldered to a pinout located at a single corner of each transmitter and receiver. Several iterations of code were then written and uploaded to each Arudino depending whether the Arduino was connected to a transmitter or a receiver (Appendix B. See R1x, T1&2x, & R2x). Data collection for the second prototype began with the top of each helical antenna resting against the edge of a step. After collecting >30 pieces of data, the distance between the tops of each helical antenna increased by one step. This process was repeated for each step and for each iteration of code with each iteration having a count of either 10 or 100. One iteration of code (P-2A_10 & P_2A_100) established a single start time (in milliseconds) and repeatedly calculated the net time it took for a radio wave to travel some distance (*d*) *N* times (Appendix B: "R1x") by calculating the length of time it took for a counter to reach a value of *N*, where the counter would only increase if a

radio frequency was received. A second iteration of code (P-2B_10 & P-2B_100) established a start time (in milliseconds) within a loop and repeatedly calculated the net time it took a radio wave to travel some distance (*d*) *N* times (Appendix B: "R2x") by calculating the length of time it took for a counter to reach a value of *N*, where the counter would only increase if a radio frequency was received. A final iteration (P-2CB_10 & P-2C!B_100) involved code from iteration B, but subtracting the average difference between actual and experimental lengths of prototype P-2B_100. Each set of RF ToF's were converted from milliseconds to seconds, calculated for distance, and put through a one-sample t-test. Between Arduino code and data analysis the distance between two points was calculated through the equation:

$$d = \frac{tc}{N}$$

where *t* is the time in seconds it took for a radio wave to travel across a distance, *d*, *N* times and *c* is the speed of light.

Results

For each prototype, the null hypothesis was that there was no difference between the experimental lengths, calculated by the prototype, and the actual length between two terminals of a prototype. The first prototype (P-1) was tested among distances of 0.03 m, 0.05 m, 0.08 m, 0.10 m, 0.13 m, 1.22 m, and 2.84 m with experimental averages of 5.6E+08 m, 4.5E+08 m , 3.5E+08 m, 3.3E+08 m, 3.1E+08 m, 2.3E+08 m, and 7.9E+08 m, respectively. As the total number of measurements that have been calculated increased there was an increase in the distance between two points, with slopes varying between independent variables (Appendix C). The second prototype with one iteration of code and a count of 10 (P-2A_10) was tested among 0.03 m, 0.05 m, 0.08 m, 0.10 m, 0.13 m, 1.24 m, and 1.65 m with experimental averages of 1.8E+13 m, 1.6E+13 m, 1.8E+13 m, 1.6E+13 m, 1.8E+13 m, 1.8E+13 m, and 1.4E+13 m, respectively. As the total number of measurements that have been calculated increased the distance between two points decreased decreasingly mapped by the power equation: $y = 1E+12x^{-1.537}$ (Appendix C). The second prototype with one iteration of code and a count of 100 (P-2A_100) was tested among 0.03 m, 0.05 m, 0.08 m, 0.10 m, 0.13 m, 1.22 m, and 1.68 m with experimental averages of 5.5E+08 m 2.6E+11 m, 5.5E+08 m, 2.6E+11 m, 2.6E+11 m, 2.6E+11 m, and 2.6E+11 m, respectively. As the total number of measurements that have been calculated increased the distance between two points increased decreasingly mapped by the power equation: $y = 1E+08x^{0.4359}$ (Appendix C). The second prototype with another iteration of code and a count of 10 (P-2B_10) was tested among 0.03 m, 0.05 m, 0.08 m, 0.10 m, 0.13 m, 1.22 m, and 1.45 m with experimental averages of 1.3E+09 m 1.3E+09 m, 1.3E+09 m, 1.3E+09 m, 1.3E+09 m, 1.3E+09 m, and 1.4E+09 m, respectively. As the total number of measurements that have been calculated increased the distance between two points increased decreasingly mapped by the power equation: $y = 408184x^{1.7887}$ (Appendix C). The second prototype with another iteration of

code and a count of 100 (P-2B_100) was tested among 0.03 m, 0.05 m, 0.08 m, 0.10 m, 0.13 m, 1.22 m, and 1.45 m with experimental averages of 2.2E+07 m 2.2E+07 m, 2.2E+07 m, 2.2E+07 m, 3.5E+08 m, 3.5E+08 m, and 3.5E+08 m, respectively. As the total number of measurements that have been calculated increased the distance between two points increased decreasingly mapped by the power equation: $y = 1E+07x^{1.2269}$ (Appendix C). The second prototype with another iteration of code and a count of 100 (P-2B_100) was tested among 0.03 m, 0.05 m, 0.08 m, 0.10 m, 0.13 m, 1.22 m, and 1.45 m with experimental averages of 2.2E+07 m 2.2E+07 m, 2.2E+07 m, 2.2E+07 m, 3.5E+08 m, 3.5E+08 m, and 3.5E+08 m, respectively. Across all actual distances, P-2B_100 measured an average distance of 1.6E+08 m and an average percent error of 9.37E+10%. As the total number of measurements that have been calculated increased the distance between two points increased decreasingly mapped by the power equation: $y = 1E+07x^{1.2269}$ (Appendix C). The average difference between the actual distance and the experimental distance was 1.6E+08 m. The second prototype with another iteration of code and a count of 10 (P-2CB_100) was tested among 0.03 m, 0.05 m, 0.08 m, 0.10 m, 0.13 m, 1.22 m, and 1.45 m with experimental averages of -1.4E+08 m -1.4E+08 m, -1.4E+08 m, -1.4E+08 m, 1.9E+08 m, 1.9E+08 m, and 1.9E+08 m, respectively. As the total number of measurements that have been calculated increased the distance between two points increased, mapped by the linear equation: $y = 2E+07x - 2E+08$ (Appendix C). The second prototype with another iteration of code and a count of 100 (P-2C!B_100) was tested among 0.03 m, 0.05 m, 0.08 m, 0.10 m, 0.13 m, 1.22 m, and 1.45 m with experimental averages of -1.4E+08 m -1.4E+08 m, -1.4E+08 m, -1.4E+08 m, -1.4E+08 m, -1.4E+08 m, and -1.4E+08 m, respectively. As the total number of measurements that have been calculated increased the distance between two points increased, mapped by the linear equation: $y = 2E+07x - 2E+08$ (Appendix C).

Conclusions

Data Analysis

Each prototype measured 7 different distances. Each actual distance was then compared to an experimental distance through a one-sample t-test and percent errors. Prototype P-1 measured an average distance of 4.3E+08 m compared to an average actual distance of 0.635 m. The average percent error of the first prototype was 5.839E+11% meaning Prototype P-1 was extremely inaccurate. Prototype P-2A with a count of 10 measured an average distance of 1.7E+13 m compared to an average actual distance of 0.468 m. The average percent error of Prototype P-2A with a count of 10 was 100% meaning Prototype P-2A with a count of 10 was extremely inaccurate. Prototype P-2A with a count of 100 measured an average distance of 1.8E+11 m compared to an average actual distance of 0.468 m. The average percent error of Prototype P-2A with a count of 100 was 1.44E+14% meaning Prototype P-2A with a count of 100 was extremely inaccurate. Prototype 2B with a count of 10 measured an average distance of 1.3E+09 m compared to an average actual distance of 0.435 m. The average percent error of Prototype P-2B with a count of 10 was 9.38E+10% meaning Prototype P-2B with a count of 10 was extremely inaccurate. Prototype 2B with a count of 100 measured an average distance of 1.6E+08 m compared to an average actual distance of 0.435 m. The average percent error of Prototype P-2B with a count of 100 was 9.37E+10% meaning Prototype P-2B with a count of 100 was extremely inaccurate. Prototype 2CB with a count of 100 measured an average distance of 4.4E-01 m compared to an average actual distance of 0.435 m. The average percent error of Prototype P-2CB with a count of 100 was -1.39E+11% meaning Prototype P-2CB with a count of 100 was extremely inaccurate. Prototype 2C!B with a count of 100 measured an average distance of -1.4E+08 m compared to an average actual distance of 0.435 m. The average percent error of Prototype P-2C!B with a count of 100 was -1.84E+11% meaning Prototype P-2C!B with a count

of 100 was extremely accurate. Each prototype was able to measure down to the ten-thousandths place meaning each prototype was relatively precise. The percent error between a measured distance and an actual distance showed a gradual decrease with each new iteration of P-2 marked by the equation: $y = 9E+16x-10.12$ with an $R^2 = 0.9068$, indicating the equation is strong in relation to the average percent error of each iteration (Appendix D).

Sources of Error

While measuring it is likely that other radio waves from FM radios, AM radios, WiFi, and other electronic devices interfered with the connection between two terminals. These unintended radio waves could have resonated with the radio waves of the prototype and could have caused a delay. This delay would increase the RF ToF and the distance the device measured. To limit interference in future experiments, testing will have been done at times when wireless usage was low, or in areas with a low risk of radio interference. While measuring, radio waves could have propagated in unintended directions. This would weaken the strength of a radio wave along the path of a measured distance. This weakened radio wave would increase the RF ToF and the distance the device measured. In addition, the surfaces of objects within a room could have reflected the radio waves in unintended directions causing an increase in RF ToF and the distance the device measured. To increase the strength of waves and direct radio waves along the path of a distance, metal reflectors could be mounted to each terminal of a prototype to direct the waves in a certain direction. Within an Arduino microcontroller, electrons travel long distances within a circuit causing an increase in the RF ToF and the distance the device measured. To account for the distance electrons travel within an Arduino circuit, several more experiments could be run where the average difference between actual distances and the measured distances would be treated as a constant and would be subtracted from future measurements.

Future Work

In the future, *LightLength* could be improved through hardware and software adjustments. With hardware, reflectors could be added to each terminal to direct radio waves along the distance of some measurements. Additionally, a microcontroller with a more precise clock could be used to obtain more precise measurements within a shorter amount of time. Finally, more than two terminals could be used in order to mimic a GPS and utilize trilateration to accurately find the location of two terminals and the distance in between. With software, improvements could be made to make *LightLength* more user-friendly and more precise. This would include printing the distance in the serial monitor as a dynamic average that would change with each new data point.

References

Adams, L. (2012, September 14). Competition Rising in Portable CMMs. Retrieved from https://www.qualitymag.com/articles/84339-competition-rising-in-portable-cmms

Arduino Communication Protocols Tutorial. (2018, June 28). Retrieved from https://www.deviceplus.com/how-tos/arduino-guide/arduino-communication-protocols-tutorial/

1. Ariel Luzzatto, Salomon Serfaty, inventors; Motorola Solutions Inc, assignee. Measuring distance using wireless communication. patent US7515103B2. August 21, 2006.

CodeNMore. (2017, February 09). Beginner Electronics - 1 - Introduction (updated). Retrieved from https://www.youtube.com/watch?v=r-X9coYTOV4&list=PLah6faXAgguOeMUIxS22ZU4w5nDvCl5gs

Control Segment. (n.d.). Retrieved from https://www.gps.gov/systems/gps/control/

Deshmukh A., Kulkarni S.A. 29 Dec. 2009. GPS Signal Distribution System. Trivandrum, Kerala, India. IEEE. Retrieved from https://ieeexplore-ieee.org.ezproxy.wpi.edu/document/5375894/keywords#keywords

FaroArm®. (n.d.). Retrieved from https://www.faro.com/products/factory-metrology/faroarm/

FARO Technologies, Inc. (2018, August 14). Issues That Can Degrade Accuracy with the FaroArm and Gage. Retrieved from https://knowledge.faro.com/Hardware/FaroArm_and_ScanArm/USB_FaroArm/Issues_That_Can_Degrade_Accuracy_with_the_FaroArm_and_Gage

2-IN-1 50 FOOT LASER TAPE MEASURE WITH DIGITAL DISPLAY. (n.d.). Retrieved from

https://www.generaltools.com/ltm1-2-in-1-laser-tape-measure-50-laser-distance-measure-16-tape-measure

Fullerton, D. (2012, January 17). High School Physics - Wave Basics. Retrieved from https://www.youtube.com/watch?v=zXSazsB5-jc&list=PLd2HWlWcMswsNuNfOYk8H5GtI8LLg6a3

Iforce2d. (2014, December 07). NRF24L01 range test (arduino). Retrieved from https://www.youtube.com/watch?v=lR60toEjHl8

Joseph J. Carr; George W. (Bud) Hippisley,: Practical Antenna Handbook, Fifth Edition. Radio-Wave Propagation, Chapter (McGraw-Hill Professional, 2012), AccessEngineering

Lanzisera, S., Lin, D. T., & Pister, K. S. (2006, June 30). RF Time of Flight Ranging for Wireless Sensor Network Localization. Retrieved October 7, 2018, from, https://people.eecs.berkeley.edu/~pister/publications/2006/

Maintenance. (n.d.). Retrieved from https://www.faro.com/support/maintenance/

Matthews, M. (2018, September 19). How to Measure with AR on iPhone. Retrieved from https://www.imore.com/how-do-measurements-ar-iphone

NRF24L01 RF Module Tutorial. (2017, May 04). Retrieved from https://www.deviceplus.com/how-tos/arduino-guide/nrf24l01-rf-module-tutorial/

Qu Y., Tian Q. 2010. Multi-UAV Cooperative Positioning Based on Delaunay Triangulation.Taiyuan, China. IEEE. Retrieved from https://ieeexplore-ieeeorg.ezproxy.wpi.edu/document/5636911/keywords#keywords

Saigo Y., Ko S., Takayama J., Ohyama S. 2012. Precise asynchronous RF ToF measurement based on Two-Way-Ranging using heterogeneous clocks. Akita, Japan. IEEE. Retrieved from https://ieeexplore-ieee-org.ezproxy.wpi.edu/document/6318675

Sharp, T. (2017, October 19). How Far is Earth from the Sun? Retrieved from https://www.space.com/17081-how-far-is-earth-from-the-sun.html

Sharp, T. (2017, October 27). How Far is the Moon? Retrieved from https://www.space.com/18145-how-far-is-the-moon.html

"The Electromagnetic Spectrum". (n.d.). Retrieved from http://www.pas.rochester.edu/~blackman/ast104/spectrum.html

Thompson B., Richard 1998. Global Positioning System: The Mathematics of GPS Receivers. Tucson, AZ. University of Arizona. Retrieved from https://pdfs.semanticscholar.org/60d2/c444d44932e476b80a109d90ad03472d4d5d.pdf

Trilateration vs Triangulation - How GPS Receivers Work. (2018, February 23). Retrieved from https://gisgeography.com/trilateration-triangulation-gps/

Ultrasonic Distance Meter with Laser Pointer. (n.d.). Retrieved from https://www.harborfreight.com/ultrasonic-distance-meter-with-laser-pointer-67802.html

Wall, M. (2013, September 13). Interstellar Traveler: NASA's Voyager 1 Probe On 40,000-Year Trek to Distant Star. Retrieved from https://www.space.com/22783-voyager-1-interstellar-space-star-flyby.html

Workshop, D. (2018, February 17). Using Inexpensive 433 MHz RF Modules with Arduino. Retrieved from https://www.youtube.com/watch?time_continue=1023&v=b5C9SPVlU4U

Zahradnik, F. (n.d.). Understand the Use of Trilateration in GPS. Retrieved from https://www.lifewire.com/trilateration-in-gps-1683341

Appendices

Limits and Assumptions

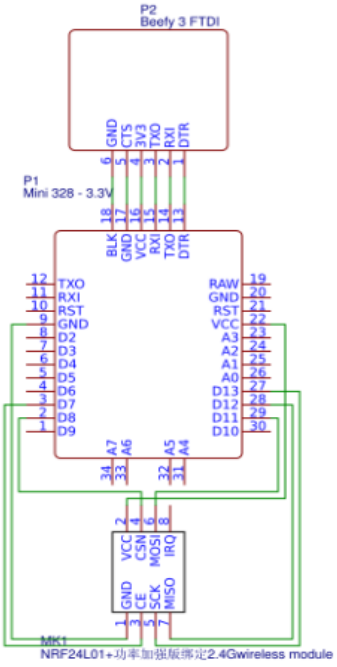
Limitations:

- Excessive & external funding was not available
- Experience in wireless communications was limited
- Experience in electronics was limited
- Project lasted approx. 5 months
- Project hours were limited due to academics
- The quality of the soldering iron, bandsaw, and electronics were at a beginner level

Assumptions:

- Participants of the engineering survey answered honestly
- Transceivers can maintain a constant connection during testing
- Individual data points were each measured independently and not influenced by prior data points
- The resonance of radio waves was not a significant factor during measurement
- The radio waves of other terminals, that were not a part of the prototype, do not affect the measurements of the prototype
- Connections between components on the prototype were secure

Appendix A – Schematics



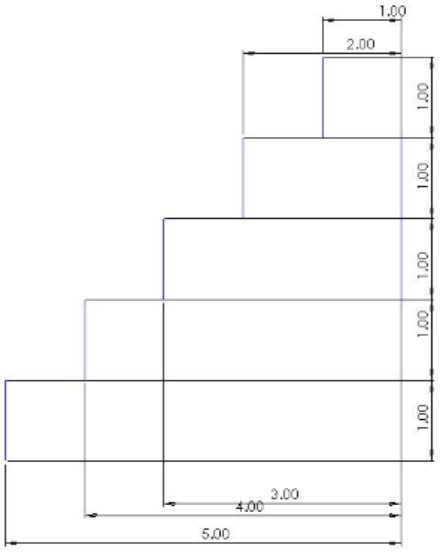


Figure 3 (Right) - A sketch of the outline of the staircase-like block used for testing

Figure 4 (Left) - An electric schematic of the first prototype (P-1)

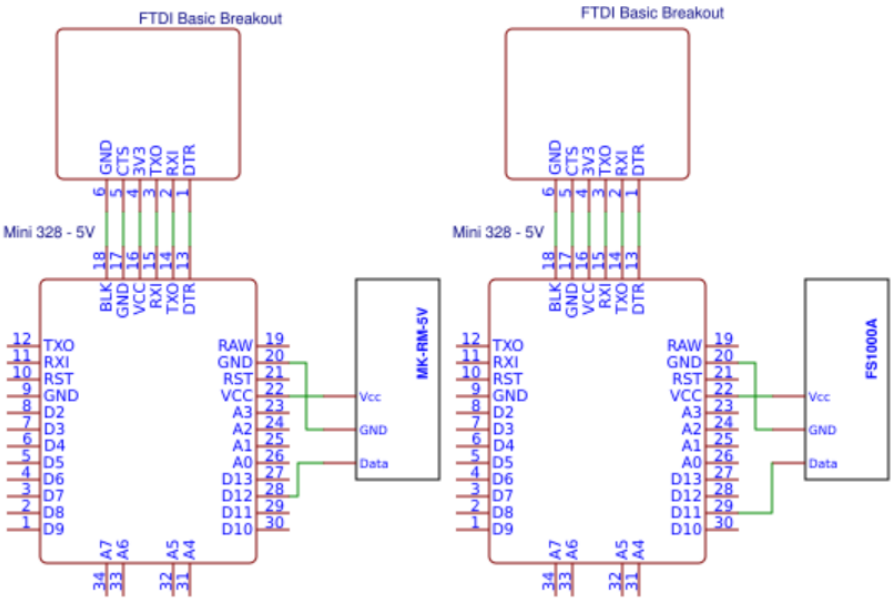


Figure 5 (Right) - An electric schematic of the receiver of the second prototype (P-2)

Figure 6 (Left) - An electric schematic of the transmitter of the second prototype (P-2)

Appendix B – Code

"Getting Started" TMRh20

```
/* Getting Started example sketch for nRF24L01+ radios

* This is a very basic example of how to send data from one node to another

* Updated: Dec 2014 by TMRh20*/

#include <SPI.h>

#include "RF24.h"

/****************** User Config ***************************/

/***      Set this radio as radio number 0 or 1 ***/

bool radioNumber = 0;

/* Hardware configuration: Set up nRF24L01 radio on SPI bus plus pins 7 & 8 */

RF24 radio(7,8);

/**********************************************************/

byte addresses[][6] = {"1Node","2Node"};

// Used to control whether this node is sending or receiving

//It is receiving

bool role = 0;

void setup() {

 Serial.begin(115200);


Serial.println(F("RF24/examples/GettingStarted"));

  Serial.println(F("*** PRESS 'T' to begin transmitting to the other node"));

  radio.begin();

  // Set the PA Level low to prevent power supply related issues since this is a
 // getting_started sketch, and the likelihood of close proximity of the devices.
RF24_PA_MAX is default.

  radio.setPALevel(RF24_PA_LOW);

  // Open a writing and reading pipe on each radio, with opposite addresses

  if(radioNumber){

    radio.openWritingPipe(addresses[1]);

    radio.openReadingPipe(1,addresses[0]);

  }else{

    radio.openWritingPipe(addresses[0]);

    radio.openReadingPipe(1,addresses[1]);

  }

  // Start the radio listening for data

  radio.startListening();

}

void loop() {

/****************** Ping Out Role ***************************/

if (role == 1)  {

    radio.stopListening();
// First, stop listening so we can talk.

    Serial.println(F("Now sending"));

    unsigned long start_time = micros(); // Take the time, and send it.  This will block until complete

     if (!radio.write( &start_time, sizeof(unsigned long) )){

       Serial.println(F("failed"));

     }
```

```
    radio.startListening();
// Now, continue listening

    unsigned long started_waiting_at = micros();
// Set up a timeout period, get the current
microseconds

    boolean timeout = false;
// Set up a variable to indicate if a response was
received or not

    while ( ! radio.available() ){
// While nothing is received

      if (micros() - started_waiting_at > 200000
){            // If waited longer than 200ms,
indicate timeout and exit while loop

          timeout = true;

          break;

      }

    }

    if ( timeout ){                                 //
Describe the results

        Serial.println(F("Failed, response timed
out."));

    }else{

        unsigned long got_time;
// Grab the response, compare, and send to
debugging spew

        radio.read( &got_time, sizeof(unsigned
long) );

        unsigned long end_time = micros();

        // Spew it

        Serial.print(F("Sent "));

        Serial.print(start_time);

        Serial.print(F(", Got response "));

        Serial.print(got_time);

        Serial.print(F(", Round-trip delay "));

        Serial.print(end_time-start_time);

        Serial.println(F(" microseconds"));

    }

    // Try again 1s later

    delay(1000);

  }

/****************** Pong Back Role
***************************/

  if ( role == 0 )  {

    unsigned long got_time;

    if( radio.available()){
// Variable for the received timestamp

      while (radio.available()) {
// While there is data ready

        radio.read( &got_time, sizeof(unsigned
long) );             // Get the payload

      }

      radio.stopListening();
// First, stop listening so we can talk

      radio.write( &got_time, sizeof(unsigned
long) );              // Send the final one back.

      radio.startListening();
// Now, resume listening so we catch the next
packets.

      Serial.print(F("Sent response "));

      Serial.println(got_time);

    }

  }

/****************** Change Roles via Serial
Commands ***************************/

  if ( Serial.available() ) {

    char c = toupper(Serial.read());

    if ( c == 'T' && role == 0 ){
```

```
    Serial.println(F("*** CHANGING TO TRANSMIT ROLE -- PRESS 'R' TO SWITCH BACK"));

    role = 1;                  // Become the primary transmitter (ping out)

  }else

   if ( c == 'R' && role == 1 ){

                        "R1x"

/* This code will output the RF ToF of a single
        char value when received *count* times
        from a transmitter

* Program was reiterated from:
        https://www.youtube.com/watch?v=b5
        C9SPVlU4U

* "Using Inexpensive 433 MHz RF Modules
        with Arduino"

* Channel: The DroneBot Workshop */

/*Initializing the libraries*/

#include <RH_ASK.h>

#include <SPI.h>

/*Initializing a connection*/

RH_ASK rf_driver;

void setup() {

  rf_driver.init();

  Serial.begin(115200);

}

/*Initializing variables for a RF ToF
        calculations*/

int counter = 0;

int count = 100;

int startTime = millis();

uint8_t buf[1];

uint8_t buflen = sizeof(buf);
```

```
    Serial.println(F("*** CHANGING TO RECEIVE ROLE -- PRESS 'T' TO SWITCH BACK"));

    role = 0;                  // Become the primary receiver (pong back)

    radio.startListening();}}} // Loop


/*Will loop the relay method for P-2*/

void loop() {

 //Will increase the counter (delay the timer)
        until it is equal to *count* and *count*
        char values have been received

  while (counter != count){

   if (rf_driver.recv(buf, &buflen)){

    counter++;

   }

  }

  //After the loop the average time it took for a
        radio wave to travel will be printed

  //This is done by taking the total time
        (millis()-startTime) and dividing it by
        the number of loops in the while loop
        above

  Serial.println((millis()-startTime)/count);\

  //Adds the current time to the startTime to
        account for the total time that has
        passed since the beginning of the
        program

  startTime += millis();

  //Resets the counter variable for another tiral

  counter = 0;

}

T1&2x by The Dronebot Workshop
```

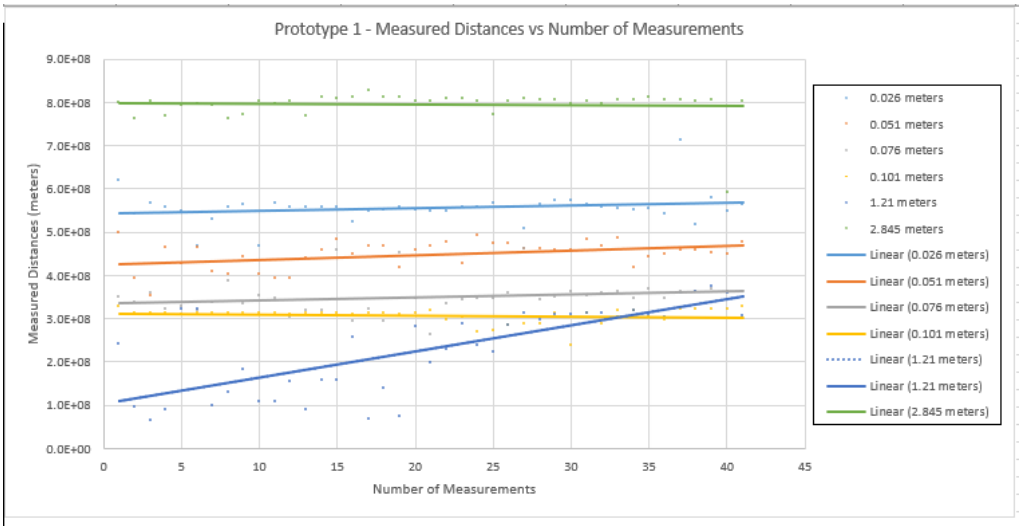
```
/* This program will initialize the RH_ASK
        library to send a single char over a
        transmitter
 * Program was taken from:
        https://www.youtube.com/watch?v=b5
        C9SPVlU4U
 * "Using Inexpensive 433 MHz RF Modules
        with Arduino"
 * Channel: The DroneBot Workshop*/
 /*Initializing the libraries */
#include <RH_ASK.h>
#include <SPI.h>
/*Initializing a transmitter*/
RH_ASK rf_driver;
void setup() {
  rf_driver.init();
}
/*Constantly sending a char over the
        transmitter*/
void loop() {
  const char *msg = "a";
  rf_driver.send((uint8_t *)msg, strlen(msg));
  rf_driver.waitPacketSent();
}
```

"R2x"

```
/ * This code will output the RF ToF of a single
        char value when received *count* times
        from a transmitter
 * Program was reiterated from:
        https://www.youtube.com/watch?v=b5
        C9SPVlU4U
 * "Using Inexpensive 433 MHz RF Modules
        with Arduino"
 * Channel: The DroneBot Workshop*/
 /*Initializing the libraries*/
#include <RH_ASK.h>
#include <SPI.h>
/*Initializing a connection*/
RH_ASK rf_driver;
void setup() {
  rf_driver.init();
  Serial.begin(115200);
}
/*Initializing variables for a RF ToF
        calculations*/
int counter = 0;
int count = 1000;
long startTime =0;
uint8_t buf[1];
uint8_t buflen = sizeof(buf);
/*Will loop the relay method for P-2*/
void loop() {
  //Sets the current time as the start time of a
        loop
  startTime = millis();
  //Will increase the counter (delay the timer)
        until it is equal to *count* and *count*
        char values have been received
   while (counter != count){
     if (rf_driver.recv(buf, &buflen)){
       counter++;
     }
   }
   //After the loop the average time it took for a
        radio wave to travel will be printed
```

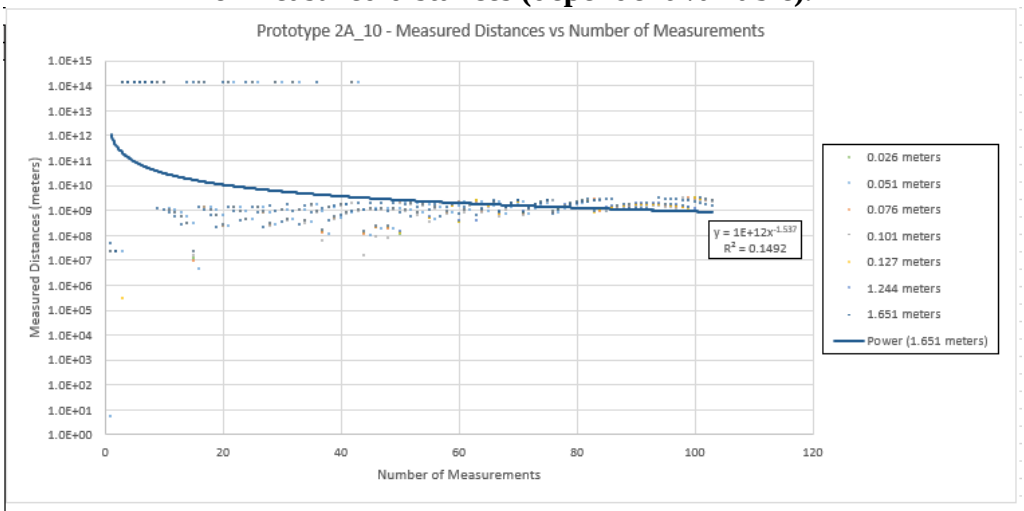
```
//This is done by taking the total time
      (millis()-startTime) and dividing it by
      the number of loops in the while loop
      above

Serial.println((millis()-startTime)/count);

//Resets the counter and startTime variables
      for another trial

counter = 0;

startTime = 0;
```

Appendix C – Data



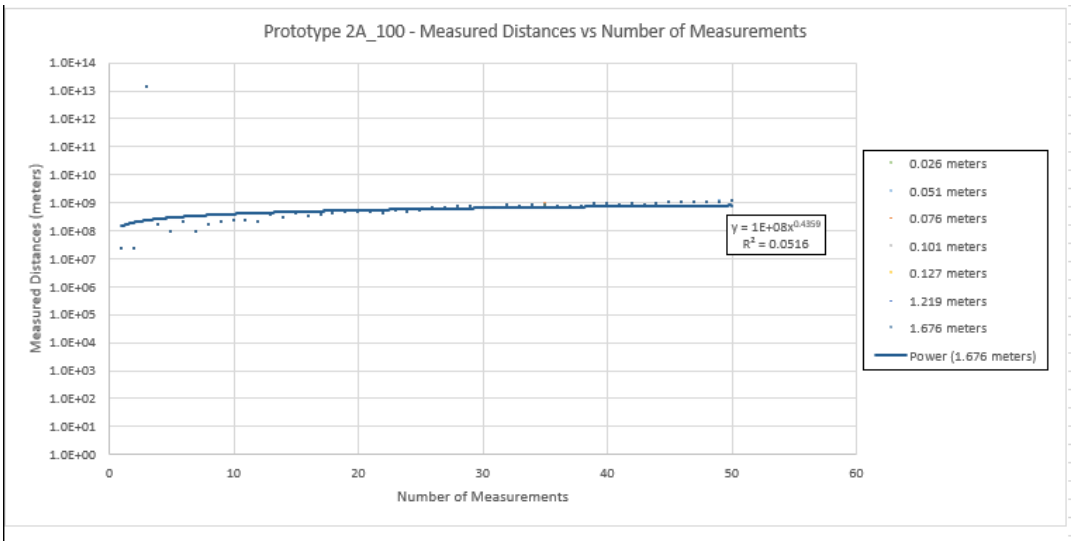
Prototype 1 - One-Sample t-Test							
Hypothetical Average (m)	0.03	0.05	0.08	0.10	0.13	1.22	2.84
Actual Average (m)	5.553E+08	4.487E+08	3.504E+08	3.320E+08	3.075E+08	2.309E+08	7.947E+08
Difference (m)	5.553E+08	4.487E+08	3.504E+08	3.320E+08	3.075E+08	2.309E+08	7.947E+08
STDEV (m)	3.606E+07	3.206E+07	3.887E+07	1.655E+07	1.742E+07	9.862E+07	3.557E+07
SE (m)	5.631E+06	5.007E+06	6.070E+06	2.584E+06	2.721E+06	1.540E+07	5.554E+06
N	41	41	41	41	41	41	41
p-Value	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001

Figure 7 - The above graph shows the trends of the distances measured by the P-1 over the number of measurements that have been taken (directly related with time). The above table shows a one-sample t-test of each actual distance (independent variable) compared to a set of measured distances (dependent variable).



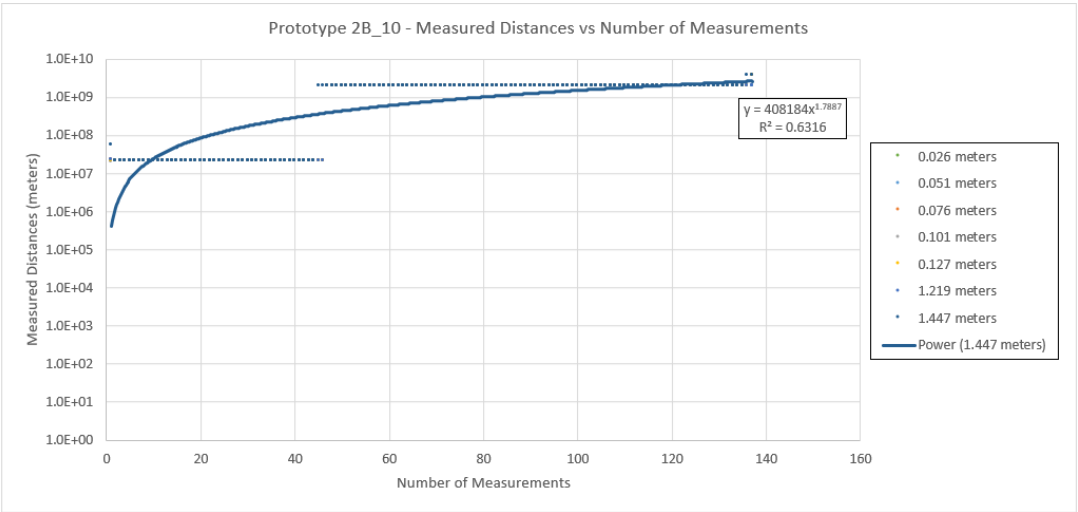
Prototype 2A; count 10 - One-Sample t-Test							
Hypothetical Average (m)	0.03	0.05	0.08	0.10	0.13	1.24	1.65
Actual Average (m)	1.751E+13	1.626E+13	1.751E+13	1.626E+13	1.751E+13	1.751E+13	1.376E+13
Difference (m)	1.751E+13	1.626E+13	1.751E+13	1.626E+13	1.751E+13	1.751E+13	1.376E+13
STDEV (m)	4.437E+13	4.300E+13	4.437E+13	4.300E+13	4.437E+13	4.437E+13	3.999E+13
SE (m)	4.372E+12	4.237E+12	4.372E+12	4.237E+12	4.372E+12	4.372E+12	3.940E+12
N	103	103	103	103	103	103	103
p-Value	0.0001	0.0002	0.0001	0.0002	0.0001	0.0001	0.0007

Figure 8 - The above graph shows the trends of the distances measured by the P-2A_10 over the number of measurements that have been taken (directly related with time). The above table shows a one-sample t-test of each actual distance (independent variable) compared to a set of measured distances (dependent variable).



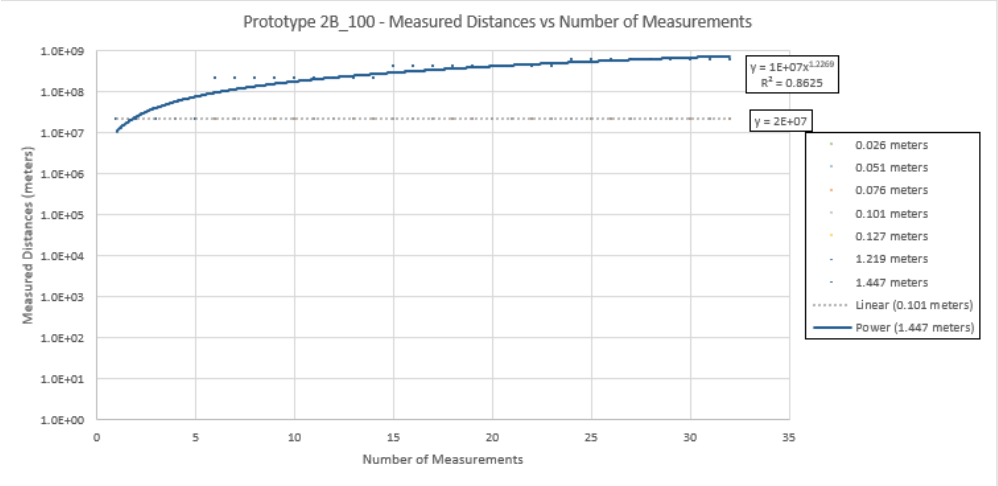
Prototype 2A; count 100 - One-Sample t-Test							
Hypothetical Average (m)	0.03	0.05	0.08	0.10	0.13	1.22	1.68
Actual Average (m)	5.501E+08	2.583E+11	5.536E+08	2.582E+11	2.583E+11	2.582E+11	2.583E+11
Difference (m)	5.501E+08	2.583E+11	5.536E+08	2.582E+11	2.583E+11	2.582E+11	2.583E+11
STDEV (m)	3.134E+08	1.822E+12	3.151E+08	1.822E+12	1.822E+12	1.822E+12	1.822E+12
SE (m)	4.432E+07	2.577E+11	4.456E+07	2.577E+11	2.577E+11	2.577E+11	2.577E+11
N	50	50	50	50	50	50	50
p-Value	<0.0001	0.3212	<0.0001	0.3212	0.3212	0.3212	0.3212

Figure 9 - The above graph shows the trends of the distances measured by the P-2A_100 over the number of measurements that have been taken (directly related with time). The above table shows a one-sample t-test of each actual distance (independent variable) compared to a set of measured distances (dependent variable).



Prototype 2B; count 10 - One-Sample t-Test							
Hypothetical Average (m)	0.03	0.05	0.08	0.10	0.13	1.22	1.45
Actual Average (m)	1.327E+09	1.327E+09	1.342E+09	1.342E+09	1.342E+09	1.327E+09	1.385E+09
Difference (m)	1.327E+09	1.327E+09	1.342E+09	1.342E+09	1.342E+09	1.327E+09	1.385E+09
STDEV (m)	9.318E+08	9.318E+08	9.569E+08	9.570E+08	9.570E+08	9.318E+08	9.698E+08
SE (m)	1.318E+08	1.318E+08	1.353E+08	1.353E+08	1.353E+08	1.318E+08	1.372E+08
N	50	50	50	50	50	50	50
p-Value	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001

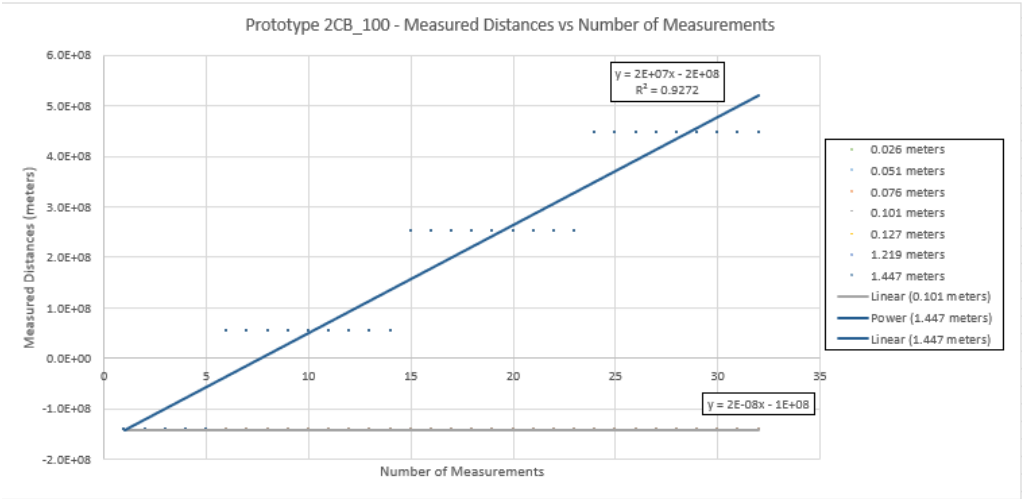
Figure 10 - The above graph shows the trends of the distances measured by the P-2B_10 over the number of measurements that have been taken (directly related with time). The above table shows a one-sample t-test of each actual distance (independent variable) compared to a set of measured distances (dependent variable).



Prototype 2B; count 100 - One-Sample t-Test							
Hypothetical Average (m)	0.03	0.05	0.08	0.10	0.13	1.22	1.45
Actual Average (m)	2.160E+07	2.160E+07	2.160E+07	2.160E+07	3.533E+08	3.533E+08	3.533E+08
Difference (m)	2.160E+07	2.160E+07	2.160E+07	2.160E+07	3.533E+08	3.533E+08	3.533E+08
STDEV (m)	0.000E+00	0.000E+00	0.000E+00	0.000E+00	2.085E+08	2.085E+08	2.085E+08
SE (m)	0.000E+00	0.000E+00	0.000E+00	0.000E+00	3.686E+07	3.686E+07	3.686E+07
N	32	32	32	32	32	32	32
p-Value	0.0000	0.0000	0.0000	0.0000	<0.0001	<0.0001	<0.0001

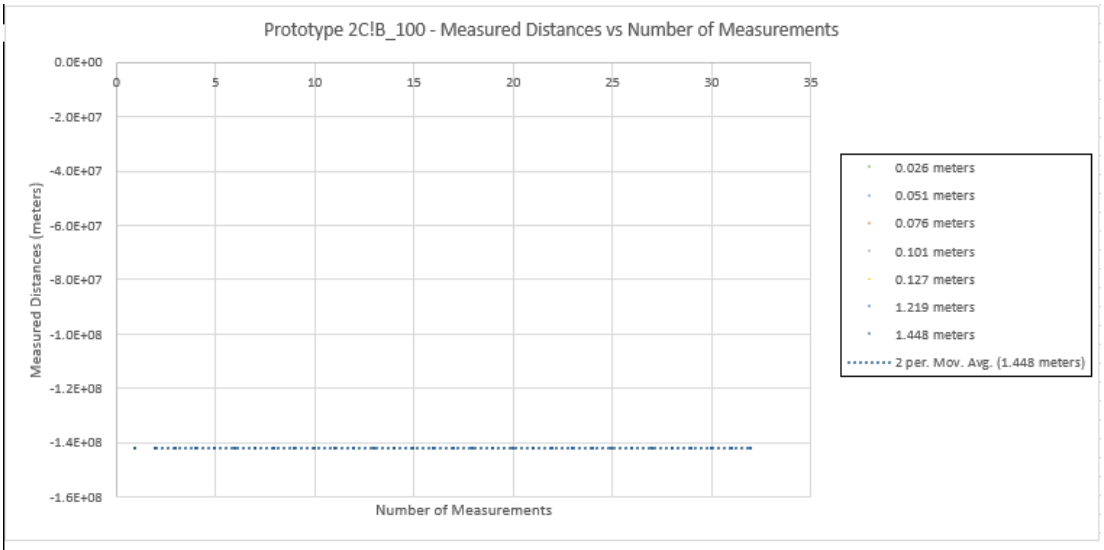
Average Difference (m)	1.6E+08

Figure 11 - The above graph shows the trends of the distances measured by the P-2B_100 over the number of measurements that have been taken (directly related with time). The above table shows a one-sample t-test of each actual distance (independent variable) compared to a set of measured distances (dependent variable).



Prototype 2CB; count 100 - One-Sample t-Test							
Hypothetical Average (m)	0.03	0.05	0.08	0.10	0.13	1.22	1.45
Actual Average (m)	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08	1.895E+08	1.895E+08	1.895E+08
Difference (m)	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08	1.895E+08	1.895E+08	1.895E+08
STDEV (m)	0.000E+00	1.211E-07	1.211E-07	1.211E-07	2.085E+08	2.085E+08	2.085E+08
SE (m)	0.000E+00	2.141E-08	2.141E-08	2.141E-08	3.686E+07	3.686E+07	3.686E+07
N	32	32	32	32	32	32	32
p-Value	0.0000	0.0000	0.0000	0.0000	<0.0001	<0.0001	<0.0001

Figure 12 - The above graph shows the trends of the distances measured by the P-2CB_100 over the number of measurements that have been taken (directly related with time). The above table shows a one-sample t-test of each actual distance (independent variable) compared to a set of measured distances (dependent variable).



Prototype 2!CB; count 100 - One-Sample t-Test							
Hypothetical Average (m)	0.03	0.05	0.08	0.10	0.13	1.22	1.45
Actual Average (m)	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08
Difference (m)	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08	-1.421E+08
STDEV (m)	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
SE (m)	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
N	32	32	32	32	32	32	32
p-Value	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Figure 13 - The above graph shows the trends of the distances measured by the P-2C!B_100 over the number of measurements that have been taken (directly related with time). The above table shows a one-sample t-test of each actual distance (independent variable) compared to a set of measured distances (dependent variable).

Heatmap of Average Measured Lengths Compared to Actual Lengths

Actual Measurements (meters)		Experimental Measurements (meters)						
		P-1	P-2A_10	P-2A_100	P-2B_10	P-2B_100	P-2CB_100	P-2C!B_100
	0.026	5.553E+08	1.751E+13	5.501E+08	1.327E+09	2.160E+07	-1.421E+08	-1.4E+08
	0.051	4.487E+08	1.626E+13	2.583E+11	1.327E+09	2.160E+07	-1.421E+08	-1.4E+08
	0.076	3.504E+08	1.751E+13	5.536E+08	1.342E+09	2.160E+07	-1.421E+08	-1.4E+08
	0.101	3.320E+08	1.626E+13	2.582E+11	1.342E+09	2.160E+07	-1.421E+08	-1.4E+08
	0.127	3.075E+08	1.751E+13	2.583E+11	1.342E+09	5.130E+08	1.895E+08	-1.4E+08
	1.219	2.309E+08		2.582E+11	1.327E+09	5.130E+08	1.895E+08	-1.4E+08
	1.245		1.751E+13					
	1.448				1.385E+09	5.130E+08	1.895E+08	-1.4E+08
	1.651		1.376E+13					
	1.676			2.583E+11				
	2.845	7.947E+08						
Average	0.951	4.313E+08	1.662E+13	1.846E+11	1.342E+09	2.322E+08	4.354E-01	-1.421E+08

Legend				
Lowest				Highest

Figure 14 - The above heat map shows the average measured distance for each actual distance per prototype. The greener a cell, the lower the value is relative to all other cells. The redder a cell, the higher the value is relative to all other cells.

Heatmap of the Percent Error of each Average Measured Length Compared to its Respective Actual Length

Actual Measurements		Experimental Measurements						
		1	2	3	4	5	6	7
(meters)		P-1	P-2A_10	P-2A_100	P-2B_10	P-2B_100	P-2CB_100	P-2C!B_100
	0.026	2.13E+10	6.71E+14	2.11E+10	5.09E+10	8.28E+08	-5.45E+09	-5.45E+09
	0.051	8.80E+09	3.19E+14	5.06E+12	2.60E+10	4.24E+08	-2.79E+09	-2.79E+09
	0.076	4.61E+09	2.31E+14	7.29E+09	1.77E+10	2.85E+08	-1.87E+09	-1.87E+09
	0.101	3.29E+09	1.61E+14	2.56E+12	1.33E+10	2.14E+08	-1.41E+09	-1.41E+09
	0.127	2.42E+09	1.38E+14	2.03E+12	1.06E+10	4.04E+09	1.49E+09	-1.12E+09
	1.219	1.89E+08		2.12E+11	1.09E+09	4.21E+08	1.55E+08	-1.17E+08
	1.245		1.41E+13					
	1.448				9.57E+08	3.54E+08	1.31E+08	-9.82E+07
	1.651		8.34E+12					
	1.676			1.54E+11				
	2.845	2.79E+08						
Average	0.951	5.84E+09	2.20E+14	1.44E+12	1.72E+10	9.38E+08	-1.39E+09	-1.84E+09
		5.8E+09	2.2E+14	1.4E+12	1.7E+10	9.4E+08	1.4E+09	1.8E+09

Legend: Lowest — Highest

Figure 15 - The above heat map shows the percent error of each average measured distance for each actual distance per prototype. The bluer a cell, the lower the percent error is relative to all other cells. The redder a cell, the higher the percent error is.

Appendix D - Projections

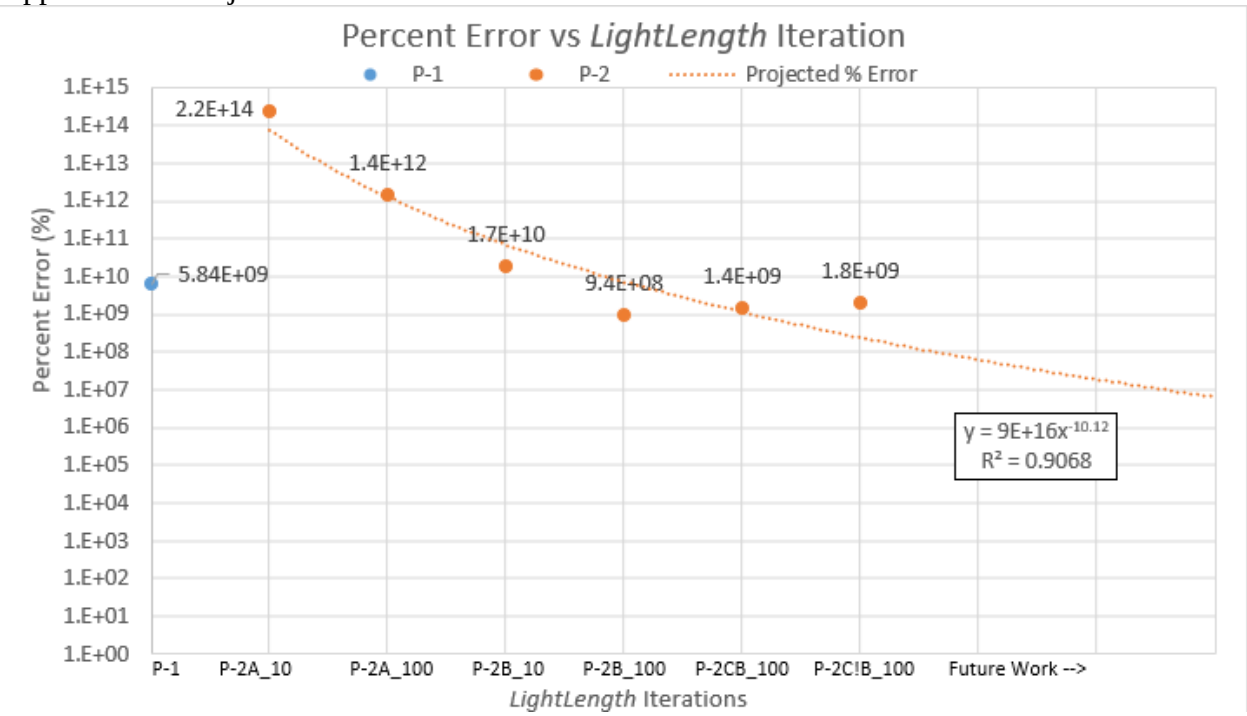


Figure 16 - The above graph shows the changes in percent error with each new iteration and a projection of percent errors over future prototypes.

Acknowledgements

Advice given by Mr. Pawel Loven, Dr. Ariel Luzzatto, Mme. Wildfong, and Ms. Curran has been a great help as each of them have provided care, and support in the development of *LightLength*. If it weren't for their efforts, *LightLength* would have never reached its current position. Additionally, a special thanks to Lucas Lazendorf for coming up with the name *LightLength*.