Fall 12-20-2017

Locating Unmanned Aerial Vehicles (UAVs)

Ali Ghubaish
Washington University in St. Louis

WASHINGTON UNIVERSITY IN ST. LOUIS

School of Engineering and Applied Science

Department of Computer Science and Engineering

Thesis Examination Committee:
Raj Jain, Chair
Roger Chamberlain
Ben Moseley

Locating Unmanned Aerial Vehicles (UAVs)

by
Ali Hussain A Ghubaish

A thesis presented to the School of Engineering
of Washington University in St. Louis in partial fulfillment of the
requirements for the degree of
Master of Science

December 2017
St. Louis, Missouri

Contents

List of Figures

List of Tables

Acknowledgments

First, I would like to thank Professor Raj Jain for his endless guidelines and support to complete this thesis and my Master Degree. He is the ultimate professor with whom every student wishes to work and I am glad that I worked under his supervision.

Second, I would like to thank the committee, Professor Chamberlain and Moseley for their great comments and the help to improve the overall write-up.

Third, I would like to thank my intelligent colleagues, Tara Salman, Guillaume Valentis, Xipeng Wang, Arghya Datta, Marcio Teixeira, Maede Zolanvari, and Ria Das for their help in this thesis.

Finally, I would like to thank my parents and siblings for the endless support they gave me during my master degree.

Ali Hussain A Ghubaish

Washington University in St. Louis

December 2017

Dedicated to my parents and siblings.

ABSTRACT

Locating Unmanned Aerial Vehicles (UAVs)

by

Ali Hussain A Ghubaish

Master of Science in Computer Engineering

Washington University in St. Louis, December 2017

Research Advisor: Professor Raj Jain

Despite the popularity and usefulness of Unmanned Aerial Vehicles (UAVs) or drones, they are not allowed to fly in some areas without prior permission from the Federal Aviation Administration (FAA). However, many incidents of UAVs breaching such restrictions have been reported. A UAV location system can help the law enforcement to be alerted and can prevent UAVs breaching any restricted area without permission. This master thesis proposes a UAV location system where each UAV has a unique identification tag. The method consists of two stages: distance and location estimation. We compared distance estimation using three different methods: Time of Arrival (ToA), counter, and Received Signal Strength Indication (RSSI). Long Range Wide Area Network (LoRaWAN) protocol is utilized in the system. Initial results have shown that RSSI is the most accurate among the three methods and also has a minimal cost. Therefore, RSSI was used to estimate the distance between the UAV and each of the ground stations. Location of the UAV can be determined using four ground stations coordinates and their estimated distance from the UAV. Several factors that may affect the measured RSSI are also discussed. These include different environments, different heights, antenna directions, and different message lengths.

Chapter 1

Introduction

According to the Federal Aviation Administration (FAA), around seven million Unmanned Aerial Vehicles (UAVs) will be sold in the United States by 2020 [1]. UAVs have many potentials in both civilian and military applications. However, they can also hinder public safety and privacy when flying in an unauthorized area. They can disturb businesses and governments that control some sensitive areas to provide safety and privacy to the public. Therefore, several areas are restricted where UAVs are not allowed. Such places include airports, borders, and many others. In 2016 alone, 1,800 incidents were reported of UAVs sighting, including UAVs coming too close to airplane and resulting in dangerous situations [2-4]. This number has increased by more than one third compared to 2015. Although no accident has happened, it is important to find a solution to reduce such incidents.

A number of solutions have been proposed for UAV surveillance such as the mandatory registration in the FAA registry, geolocation system, drone guns, signal jammers, and human vision [5-9]. The FAA started a UAV registry in 2015 to locate the owners of UAVs violating any rules [10]. UAV's manufactures have started using the Global Positioning System (GPS) information to prevent their UAVs from flying in restricted areas. Two drone guns, "Dronegun" and "DroneDefender," have been offered by two different companies that can take over the control of any UAV [7, 8]. These guns are used to jam the signal between the UAV and its remote control then the UAV will be controlled by the guns. However, drone guns require UAVs to be in Line of Sight (LoS) with the gun and a human to use it. Signal jammers have been used to prevent UAVs from being controlled by their owners, which will make the UAVs go back to its home point. This solution will jam all the wireless devices that use the same frequency band as the UAV. This includes the WiFi access points (APs) that make the jamming idea not convenient in most places. Humans' vision, cameras and proper monitoring can help enforce security in the restricted areas. This solution is costly and difficult to maintain in the long run.

Finding a wireless technology that can reach long distances and consumes low energy may help UAVs location problem. Long Range Wide Area Network (LoRaWAN) is a potential candidate because it is a long range, low power, and low cost wireless technology protocol [11]. This technology can reach from 15 to 30 kilometers in the optimal cases [12]. Thus, deploying a system that uses LoRaWAN can help track the UAVs.

This work proposes a UAV Location system using LoRaWAN and experimental results using a prototype. The system can be an add-on to any UAV or it can be integrated with the UAV system by the manufacturers. The system consists of two main components: the UAV and a set of ground stations (GSs). Both of them use LoRaWAN boards. The UAV node will broadcast a unique id message that is received by the GSs. Then, every GS estimates the distance between itself and the UAV. A total of four GSs is required to estimate the location of the UAV in three dimensions. This system can be added to existing or future UAVs as a required component like car plates. It will help law enforcement to be alerted when a UAVs illegally flies in a restricted area.

The rest of the thesis is organized as follows: Chapter 2 provides a background for LoRaWAN protocol, distance estimation, and location estimation; Chapter 3 presents related work; Chapter

4 discusses the system components and architecture of the main components; Chapter 5 explains the design of main components; Chapter 6 shows the experimental implementation and results in detail. A brief discussion about each of the methods is presented in Chapter 7. Chapter 8 describes ideas for the future work. Finally, a conclusion of this work is presented.

Chapter 2

Background

In this chapter, we discuss the characteristics of the LoRaWAN wireless protocol technology and the methods used to estimate the distance between the UAV and the GSs. In addition, a brief illustration of how to estimate the location of the UAV is presented.

2.1 LoRaWAN

LoRaWAN is a low power, low cost, and long-range wireless technology. It was first released to the public and originally developed by Cyclos, France in 2015. It is a Media Access Control (MAC) protocol and it was standardized by LoRa Alliance [13, 14]. The LoRa physical layer is what enables the technology to reach long ranges with low power consumption by using the chirp spread spectrum modulation. This modulation is resistant to Doppler shift and multi-path fading. Chirp spread spectrum has been used in military and space communications for decades, but LoRa was the first to use it in a low cost commercial product [11]. The frequency bands used in LoRa are different based on the rules of each country. However, they lie in the 433-915 MHz range, which helps it to reach long ranges as it is a low frequency band.

There are three classes to be supported in the devices with this technology: Class A, B, and C. Class A is the mandatory class for all LoRaWAN devices because it provides the basic rules between the nodes in the network, while the other classes provide extra features to be used as needed by different applications.

2.2 Distance Estimation Methods

To estimate the distance between two nodes, three different methods can be used. In this section, these three methods are explained. These methods include ToA, counter, and RSSI.

1. Time of Arrival (ToA)

ToA can be used to measure the distance by estimating the time elapsed between sending and receiving. GPS uses the ToA and the location of one satellite to measure the distance between the satellites and the client node [15]. It requires a highly accurate clock that is synchronized between the nodes if a one-way function is used. By sending a message that contains Time of Departure (ToD) to another node, the other node can determine its distance from the sender, provided the two nodes have synchronized clocks.

2. Counter

The counter value can be used to measure how long an event takes to finish. In our work, the counter value is based on a round-trip message. The counter is triggered when the event starts and the counter is stopped when the event is completed. This method has been used in aircrafts since 1950 [16]. In a two-node system consisting of a GS and a client, the node that needs to measure the distance is the GS. Therefore, GS should be the node the starts and ends that counter. In addition, multiple GSs will be communicating with the UAV, so the load on the UAV should be low for the UAV node to be able to quickly respond to other GSs.

3. Received Signal Strength Indication (RSSI)

RSSI is the measure of the power level in the received radio signal [17]. It is generally used to estimate the quality of the link. RSSI can also be used to estimate the distance between two nodes under certain conditions. This method is measured in decibel (dB) and it is generally a negative value for most wireless datalinks.

2.3 Location Estimation

Estimating the location of any node requires knowing two pieces of information. First, the location of at least three GSs for 2-dimensions (2D) or four GSs for 3-dimensions (3D) coordinates is required. Second, the distance between them and the target node must be known. By estimating the distance using any of previously stated methods, the location of the targeted node can be estimated. For example, the time method is being used in GPS. The GPS system consists of around 31 satellites [15]. Each satellite broadcasts its location and time. By knowing how far a client is from the satellite, the client knows its distance from this satellite and knows that it is located in a sphere with the estimated distance as a radius. Adding at least two more satellites information can help the client estimate its location in 2D by finding the points where the three satellites' spheres intersect. Further, adding more satellites' information to the equation can pin point the client location with an error of a few meters.

Chapter 3

Related Work

Most prior work in UAV location estimation use GPS information to estimate distance and location. Such estimations are needed to locate the UAV while they are in the air, which is beneficial for several applications. For example, UAVs systems are currently used for many applications in life such as medical supplies and products deliveries [18, 19]. Most of these applications need to know the location of the UAV. Therefore, they utilize available GPS information. Our goal was to find an alternative technology. Therefore, alternative solutions have been proposed for specific applications.

In [20], the authors have designed GuideLoc system that helps rescue people from a natural disasters using the UAV. This system uses RSSI and the angle of arrival (AoA) of the trapped person to find the location of that person. The system changes the direction based on the strength of the signal. Once, the UAV gets over that person, it uses GPS data of that location as the trapped person location. Our system differs from this system by relying on the RSSI value to estimate the UAV location and not the nodes location.

In [21], the authors have designed a HiQuadLoc system that helps the UAVs in locating themselves indoor by using WiFi access points (APs) RSSI value in two phases: offline and online phases. They divide the indoor area into cubes in the offline phase, which later on will help the UAV to detect its location correctly in the online phase. They have used 20 APs in an area of 1100 m^2. They were able to achieve an average error of 1.64 m with indoor UAV 3D location estimation. The UAV speed was tested between various UAV speeds up to 3 m/s. They concluded that the location error increases with the increase in the UAV speed. In this thesis, our system also uses RSSI for distance estimation; however, we target outdoor environments rather than indoors.

In [22], the authors have used a tag reader that is connected to the UAV. This tag reader is used to localize radio-frequency identification (RFID) tags in an outdoor environment based on the RSSI value that is being read by the tag reader. When the tag reader reads any RFID tag, the UAV can determine its location because the RFID tags' locations are known.

In [23], they used the UAV's center-modem to detect network APs using the RSSI value broadcasted by the APs in the network. The UAV uses RSSI value to estimate its location based on the APs known locations. They claim that their system will identify the limitations of the network they have been investigating, which is invaluable information in hazard scenarios. They found out that even with noise in the RSSI value recorded by the UAV, they were able to analysis the network while it is operating.

In [24], to solve the issue of locating the UAV in Non-line of sight (NLOS) environments, they used RSSI value to identify the propagation conditions and the particle swarm optimization (PSO). To correctly detect the UAV location, they used the maximum joint probability algorithm. Their work was in indoor environments. Since our system is going to be used outdoors with high altitude, it is not required to solve the NLOS issue at least at this point.

In [25], the work examined the possibility of using UAVs in emergency situations, where the communication infrastructure is not available or difficult to reach. It studied the effect of different factors to get the best setup for the UAVs altitude and antenna type. After conducting two experiments with different settings, they found that the directional antenna is more suitable for wireless mesh networks because it can increase the throughput with higher altitude. In addition, they showed the possibility of connecting two devices by using Voice over IP (VoIP) via two UAVs that work as cell towers.

In [26], the authors have implemented a system that can mimic the handover between the cell towers in the Cellular networks to be used in wireless networks with WiFi APs. These APs can be mounted to UAVs. The WiFi stations were connected via the Wireless Distribution System (WDS) mode, which is available in most open source routers. They were able to achieve a stable and an alive VoIP connection between two devices while it was being handed over from one WiFi station to another one.

Chapter 4

System Components and Architecture

In this chapter, the system components and architecture of the three main methods will be discussed. These methods have been used to estimate the distance between the nodes in the system and the UAV. One of them has been used to estimate the 3D location of the UAV. Further, we show our architecture for both estimations and how those components have been utilized.

4.1 System Components

The system consists of four main components: Microprocessor/Microcontroller boards, LoRaWAN modules, computers, and a battery. In addition, secondary components that were needed to be used, including connection bridge, antennas, and cables. In the following, we list and discuss all the components:

- Microprocessor/Microcontroller boards: Raspberry pi 3 (RPi3), Arduino MEGA, and ESP8266 SMT Module (ESP-12e) are the boards used in this work. The RPi3 is a microprocessor that is used to control Cooking Hacks LoRaWAN module and to report the results back to a computer to which RPi3 is connected. The Arduino MEGA is another microcontroller that can be used to control the Cooking Hacks LoRaWAN module. It has the ability to provide better real-time response as a controller compared to RPi3.

- LoRaWAN module: Libelium LoRaWAN, Moteino LoRa, and Seeeduino LoRaWAN are the LoRaWAN modules that have been used in this work. Both Libelium and Seeeduino modules use 433/868MHz frequency bands, while Moteino module uses 915MHz. Libelium uses the Libelium LoRaWAN module (henceforth ’Cooking Hacks LoRaWAN’).
- Connection bridge: This bridge is used to connect Cooking Hack LoRaWAN module to a microprocessor/microcontroller. There are two types of the connection bridge: Raspberry Pi to Arduino and Multiprotocol shield for Arduino. The first bridge is used with the RPi3, while the other one is for Arduino MEGA. Moteino and Seeeduino have their own boards soldered together with the LoRaWAN module, but they both use the same type of microcontroller as well as ESP-12e which is Arduino microcontroller.
- Computer: a regular computer is connected to either RPi3, Arduino MEGA, or any LoRaWAN enabled boards to program the LoRaWAN module or record the data.
- Battery: any power bank is sufficient to power the LoRaWAN module that is connected to the UAV, but 5000 mAh is recommended.
- Antenna: Both Cooking Hacks module and Moteino LoRa board require a separate directional antenna to work, while the Seeeduino LoRaWAN board has a built-in wire antenna.
- Cables: RPi3, ESP-12e, Moteino, and Seeduino boards use Micro USB cables, while Arduino MEGA uses USB type B to be powered-on and controlled.

4.2 System Architecture

The architecture of our system is divided into two main stages: the distance and location estimation. For distance estimation, three main methods are to be used based on their order in this work and they all share similar architecture which will be discussed first.

4.2.1 Distance Estimation

To estimate the distance between two nodes such as the UAV and one of the GSs, one of the methods discussed earlier should be used. In this work, three methods have been explained based on Figure 4.1. Each method has different components and connections as explained in the following list:

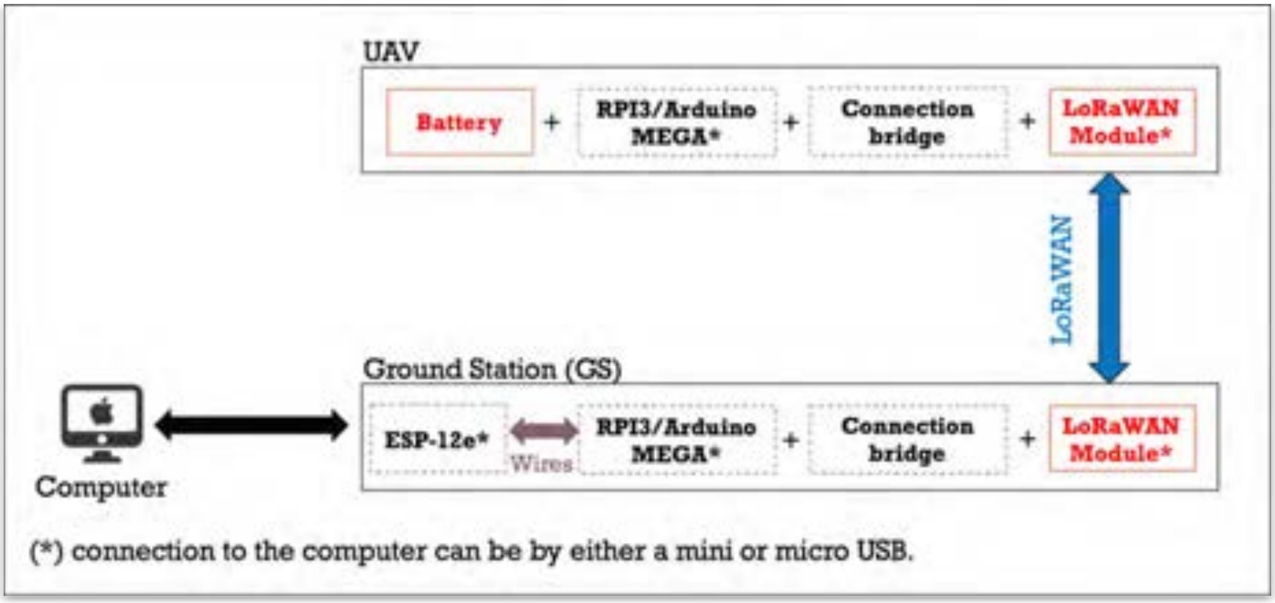

Figure 4.1: The System Architecture for the Distance Estimation

- **Time of Arrival (ToA)**

The system uses wave propagation time to estimate the distance between the UAV and one of the GSs. Two different microprocessors/microcontrollers have been used in this work: RPi3 and Arduino MEGA. Both boards use Cooking Hacks LoRaWAN module. This module, as explained in the system components, requires a connection bridge to work with either one of the microprocessors/microcontrollers.

1. Cooking Hacks with RPi3: A Raspberry to Arduino connection bridge is required to be connected first to the RPi3. Then the module, which uses XBee socket, will be connected to that shield. Two of these boards will be used. One is attached to the UAV and powered by a battery. The other will be connected to a computer to record the data and to control it.
2. Cooking Hacks with Arduino MEGA: A multiprotocol shield for Arduino is required to be connected first to the Arduino MEGA with some changes in the hardware connection between the two, which is explained in the Appendix. Because the Arduino MEGA is not supported by that shield, the previous step is important to do. The reset is similar to the RPi3 connection.

- **Counter**

The system can use a counter to estimate the round-trip propagation time and hence the distance between the UAV and one of the GSs. Two different microprocessors/microcontrollers have been used in this work: Arduino MEGA and ESP-12e. The connection between the Arduino MEGA and the Cooking Hacks board is the same as in the time method. In addition, the ESP-12e is the counter, which is connected to the Arduino MEGA. The connection between the Arduino MEGA and the ESP-12e is based on the three wires: two of them for the serial connection and one for the signal connection. The serial connection is used to send commands from the Arduino MEGA to ask for the recorded counter from the ESP-12e and to receive that counter value. The signal wire is used by the Arduino MEGA to ESP-12e to start/stop the counter in the ESP-12e. The “digitalWrite” function was not consistent, so the registers of the Arduino MEGA were used to trigger the digital pin in the Arduino board to send the signal to the ESP-12e board. The speed of the signal was stable because the registers in the Arduino MEGA were used to send the command.

- **RSSI**

  The RSSI value can also be used to estimate the distance between the UAV and one of the GSs. Two different LoRaWAN boards have been used in this work: Moteino LoRa and Seeeduino LoRaWAN. Two of each one of the two boards were used. One was attached to the UAV and powered by a battery. The other was connected to a computer to record the data and to control it. RSSI values are collected on the computer and a model was built for distance estimation based on the received value. This will be discussed in detail in the next chapter. The only difference between the two LoRaWAN boards is that Moteino LoRa uses the 915 MHz frequency band and an external directed antenna, while Seeeduino LoRaWAN uses 433/868MHz and a wire antenna soldered to the board.

4.2.2 Location Estimation

To estimate the location of the UAV, any one of the three methods discussed above can be used. In this work, RSSI is used. The architecture for location estimation is illustrated in Figure 4.2. Seeeduino LoRaWAN boards are used in this stage over Moteino boards because they showed the possibility to be used for location estimation as explained in Chapter 6. The system consists of four GSs and one UAV. Seeeduino LoRaWAN board has been attached to each of the GSs and the UAV; in addition to a battery for the board connected to the UAV. Each GS is connected to a computer to record the RSSI value received from the module connected to it. In this part, the data are collected from all the four GSs manually by transferring all the data from all the 4 GSs' computers to one computer. On that computer, the data are processed based on the known distances between the four GSs and the RSSI values received from the messages sent by the Seeeduino LoRaWAN board attached to the UAV to estimate its location.

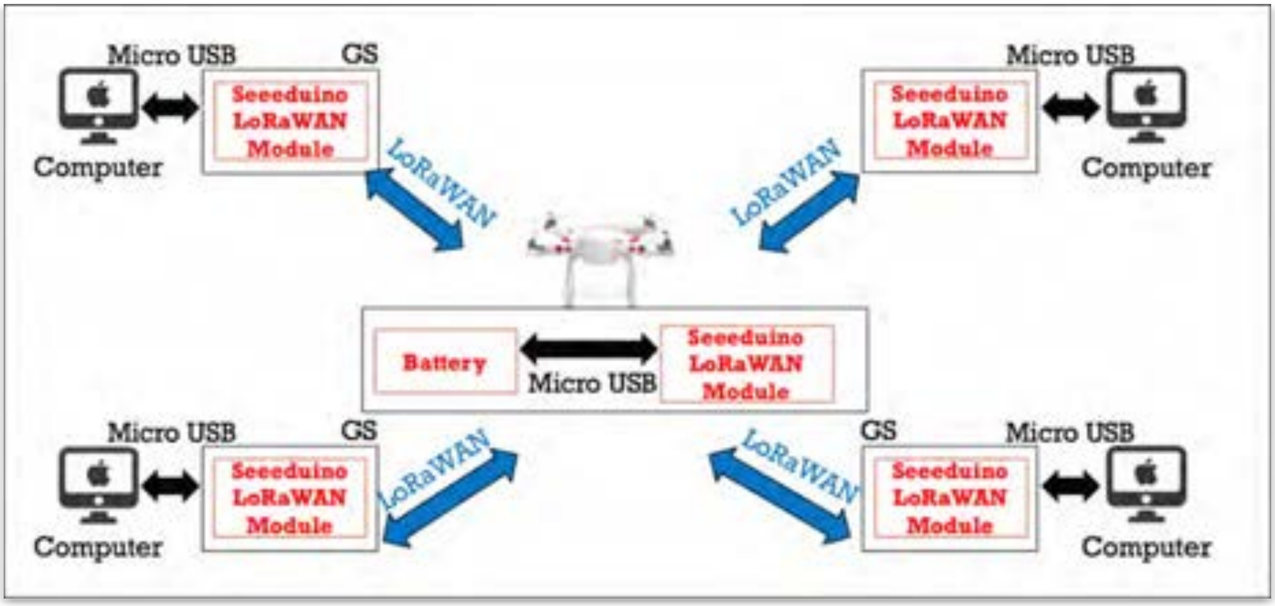

Figure 4.2: The System Architecture for the Location Estimation

Chapter 5

Design

In this chapter, the design for the three main methods will be explained. This includes the functions used to estimate the distance for each method with graphs that illustrate their functions. For location estimation, RSSI method will be used to estimate the distance which is later used to estimate the location. RSSI was selected as it showed better results in the distance estimation stage compared to the other two methods.

5.1 Distance Estimation Using Time of Arrival (ToA)

Two functions have been used to estimate the distance between the two nodes: one-way and Two-way ToA functions [27]. The one-way ToA function uses the Flight Time (FT) of one message sent from the UAV to one of the GSs as shown in Figure 5.1.

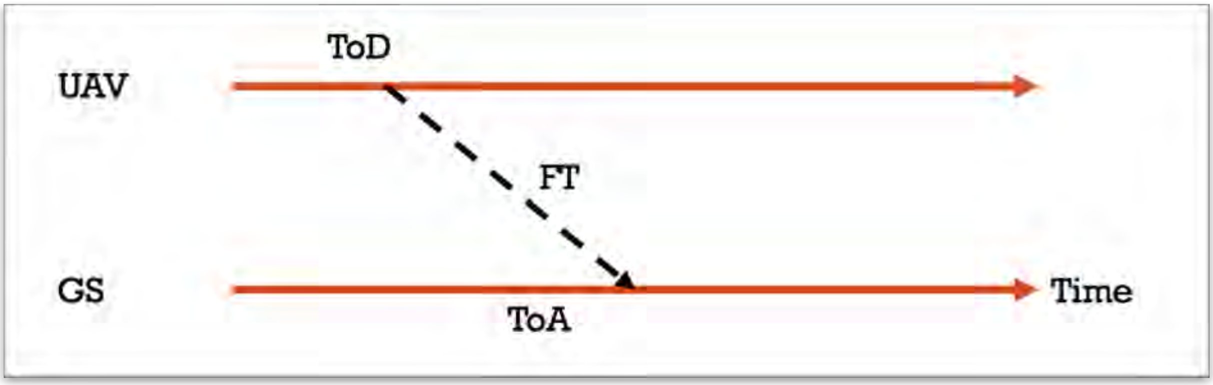

Fig. 5.1: One-Way ToA Function.

A message containing the ToD is sent from the UAV to the GS. The FT is the time between the ToD from the UAV and the ToA at the GS. The FT is multiplied by the speed of light, which is ~300 m/s. From that, the distance between the two nodes can be estimated.

$$Distance = (ToA - ToD) \times Speed\ of\ light \tag{5.1}$$

On the other hand, the two-way ToA function uses the FT round-trip message. This time is the time between the ToD of a round-trip message from one of the GSs to the ToA of that message back from the UAV as shown in Figure 5.2.

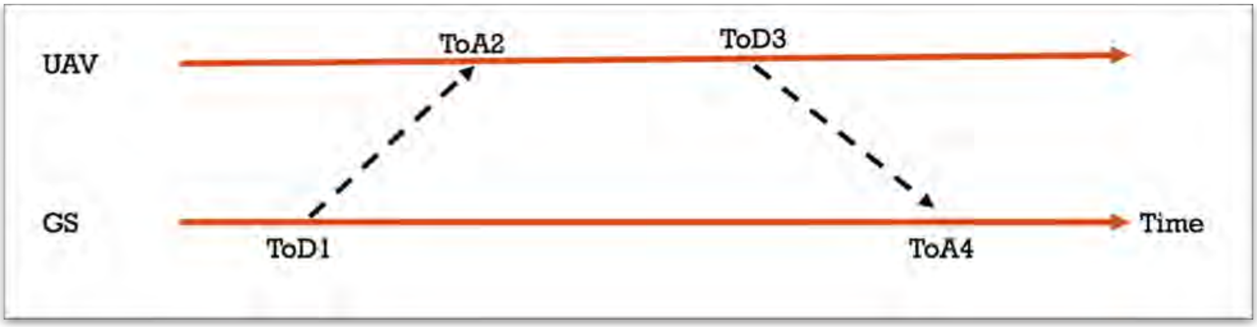

Fig 5.2: Two-Way ToA Function

A message containing the ToD as (ToD1) is sent from the GS to the UAV. A second message containing the ToA (ToA2) and ToD (ToD3) at the UAV node is sent back to the GS. The GS records the time the second message arrived from the UAV as (ToA4). To find the overall FT, the delta time between ToA and ToD in the UAV and the GS side must be calculated as shown in Equation 5.2.

$$Distance = \frac{(ToA4 - ToD1) - (ToD3 - ToA2)}{2} \times Speed\ of\ light \tag{5.2}$$

The UAV side delta time is the difference between ToD3 and ToA2, while the GS side delta time is the difference between ToA4 and ToD1. The overall FT is divided by two because it is a round-trip message. Multiplying the overall FT by the speed of light gives the distance between the two nodes. To differentiate between different UAVs, we can add the UAV ID to the second message coming from the UAV.

5.2 Distance Estimation Using A Counter

The round-trip FT of sending a message to the UAV and back can be measured by using a counter at the GSs as shown in Figure 5.3.

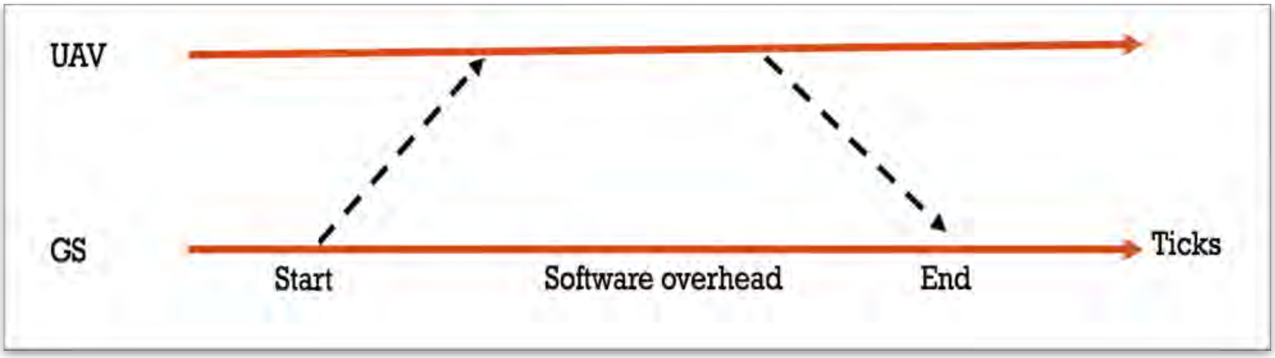

Figure 5.3: Counter Function.

A message containing the GS ID is sent to the UAV. When this message is sent, the GS sends a signal via the digital pin in the Arduino MEGA board to the ESP-12e's digital pin to start the counter (Start). The sending signal process, which triggers the counter, increases the counter value by one count compared to one microsecond counter time inside the ESP-12e. The UAV sends the same message back to the GS. When the message comes back from the UAV to the GS, the GS sends a signal to the ESP-12e to stop the counter (Stop). Then the GS gets the data back from the ESP-12e via the serial port. Also, the software overhead can be calculated in terms of the counter values as shown in Equation 5.3.

$$Distance = \left(\frac{End - Start - \ software\ overhead}{2} \times Counter\ Tick\ Time\right) \times speed\ of\ light \qquad (5.3)$$

The counter values multiplied by the counter tick time is used to estimate the distance between the UAV and the GS, which is similar to the distance measuring equipment (DME) system in aircrafts [16]. In addition, to differentiate between different UAVs, we can add the UAV ID to the second message, which comes from the UAV.

5.3 Distance Estimation Using Received Signal Strength Indication (RSSI)

RSSI value is the third method to be used for distance estimation between the UAV and one of the GSs. The distance estimation uses the RSSI value of one message from the UAV to the GS. The UAV constantly broadcasts a message that has its ID. The message frequency was two seconds because this is the minimum time interval for our LoRaWAN board to avoid losing messages. The message length and its effect are explained in Chapter 6.

The GS works as a listener, so it waits till it gets a message from any UAV. Once the GS gets a message, it records the UAV ID and RSSI. The mean of 5 RSSI values has arbitrarily been chosen as a tradeoff between the time and the variability of the RSSI values. This value is used in Equation 5.4 as (meanRSSI) to get the corresponding distance [28]. Also, meanRSSI is rounded up to the nearest integer.

$$Distance = 10^{-(\frac{\lceil meanRSSI \rceil - C}{10L})} \qquad (5.4)$$

Here, L is the path loss exponent, and C is a constant value. Both C and L were pre-calculated using the RSSI data from known distances. The model has been built based on Equation 5.5 and is explained in detail in Chapter 6.

$$RSSI = -10 * L * log_{10}(d) - C \tag{5.5}$$

Since it was difficult to use the laser meter to measure the distance between the UAV and one of the GSs after 200 meters, Equation 5.6 was used to compute the slant distance (SD) [29].

$$SD = \sqrt{GD^2 + H^2 - (2 * GD * H * co\,s(\beta))} \tag{5.6}$$

This equation accurately measures the distance between the two nodes. It requires the height (H) of the UAV, which was set to 50 meters, the distance between the ground point (GP) under the UAV and the GS is defined as (GD), and (β) is the angle between the GP and the GS as shown in Figure 5.4.

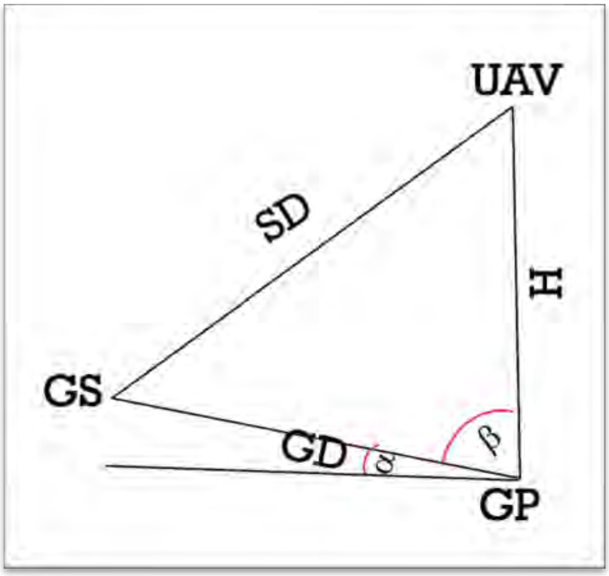

Figure 5.4: Training Data Distance Setup.

Since the GS not aligned with GP, we assumed that the UAV has a 90-degree angle with the GP. The angle (α) between the ground point and the GS was measured with the laser meter that provides the distance between the GP and the GS and the angle (α) between them. Then β can be calculated by subtracting α from 90 degrees as shown in Equation 5.7.

$$\beta = 90 - \alpha \tag{5.7}$$

5.4 Location Estimation Using RSSI

The location estimation uses the slant distance (SD) between the UAV and four GSs. We used LoRaWAN board that requires at least two seconds as the interval time between successive messages. To satisfy this requirement, the UAV was flying while it was stationary in one spot. Trilateration technique is used to determine the location of the UAV [27, 30]. This technique allows determining the exact location of any object in 3-dimension of its distance from at least four points with their known locations. In our case, the UAV is the object whose location and height need to be determined, while the four GSs are the points with known locations as shown in Figure 5.5.

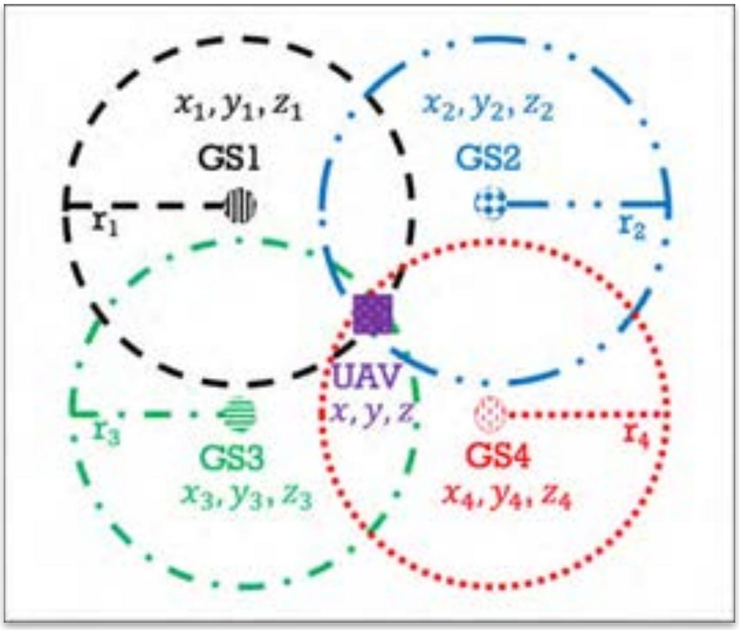

Figure 5.5: Trilateration System Architecture.

The UAV is on the surface of a sphere with radius r_i centered at GS_i. The r_i is equal to SD_i for each GS. The location of the UAV is a 3-element vector X= {x, y, z}. It can be computed as the intersection of the 4 spheres as shown in Equation 5.9. Each sphere consists of the 3D coordinates of each GS and the radius value (SD) between itself and the UAV.

$$\begin{aligned} r_1^2 &= (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 \\ r_2^2 &= (x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 \\ r_3^2 &= (x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 \\ r_4^2 &= (x - x_4)^2 + (y - y_4)^2 + (z - z_4)^2 \end{aligned} \tag{5.9}$$

We can expand out the squares in each one as shown in Equation 5.10.

$$\begin{aligned} r_1^2 &= x^2 - 2x_1x + x_1^2 + y^2 - 2y_1y + y_1^2 \\ r_2^2 &= x^2 - 2x_2x + x_2^2 + y^2 - 2y_2y + y_2^2 \\ r_3^2 &= x^2 - 2x_3x + x_3^2 + y^2 - 2y_3y + y_3^2 \\ r_4^2 &= x^2 - 2x_4x + x_4^2 + y^2 - 2y_4y + y_4^2 \end{aligned} \tag{5.10}$$

By subtracting the fourth equation, r$_4$, from all the first three equations, we get Equation 5.11.

$$\begin{aligned}
2(x_4 - x_1)\mathrm{x} + 2(y_4 - y_1)y + 2(z_4 - z_1)\mathrm{z} &= r_1^2 - r_4^2 - x_1^2 - y_1^2 - z_1^2 + x_4^2 + y_4^2 + z_4^2 \\
2(x_4 - x_2)\mathrm{x} + 2(y_4 - y_2)y + 2(z_4 - z_2)\mathrm{z} &= r_2^2 - r_4^2 - x_2^2 - y_2^2 - z_2^2 + x_4^2 + y_4^2 + z_4^2 \\
2(x_4 - x_3)\mathrm{x} + 2(y_4 - y_3)y + 2(z_4 - z_3)\mathrm{z} &= r_3^2 - r_4^2 - x_3^2 - y_3^2 - z_3^2 + x_4^2 + y_4^2 + z_4^2
\end{aligned} \tag{5.11}$$

From Equation 5.11, we can write the following matrix in Equation 5.12.

$$AX = b \tag{5.12}$$

Here, A is the coefficient matrix, X is the 3 element vector of the UAV location, and b is the right-side vector as shown in Equation 5.13.

$$\begin{bmatrix} 2(x_4 - x_1) & 2(y_4 - y_1) & 2(z_4 - z_1) \\ 2(x_4 - x_2) & 2(y_4 - y_2) & 2(z_4 - z_2) \\ 2(x_4 - x_3) & 2(y_4 - y_3) & 2(z_4 - z_3) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r_1^2 - r_4^2 - x_1^2 - y_1^2 - z_1^2 + x_4^2 + y_4^2 + z_4^2 \\ r_2^2 - r_4^2 - x_2^2 - y_2^2 - z_2^2 + x_4^2 + y_4^2 + z_4^2 \\ r_3^2 - r_4^2 - x_3^2 - y_3^2 - z_3^2 + x_4^2 + y_4^2 + z_4^2 \end{bmatrix} \qquad 5.13$$

A X = b

Here, to find X, the least squares method is used to minimize the error from the previous equations, as shown in Equation 5.14.

$$X = (A^T A)^{-1} A^T b \qquad 5.14$$

If the height for all the GSs is the same, the last column of matrix A is all zeros and matrix becomes not invertible. This can be taken care of by deleting the last column of matrix A, computing only x and y values from the above equations, and z is computed separately as in Equation 5.15.

$$z = \sqrt{r_i^2 - (x - x_i)^2 - (y - y_i)^2} + z_i \tag{5.15}$$

Chapter 6

Experimental Implementation and Results

In this chapter, the experimental implementation and results for the three main methods are explained. The steps to prepare the experiments in the software and hardware sides are presented. In addition, the results for each method are shown.

6.1 Distance Estimation Using ToA

In this initial experiment, the UAV was not involved and the distance was estimated between two LoRaWAN boards to check for feasibility. As explained in Chapter 3, two microprocessors/microcontrollers boards were used. Both boards used Cooking Hacks LoRaWAN module with 433 MHz frequency band and peer-to-peer (P2P) mode. Two connections were done: RPI3 and Arduino MEGA with Cooking Hacks LoRaWAN.

For RPI3 with Cooking Hacks LoRaWAN module connection, Cooking Hacks LoRaWAN was used. Libelium, the LoRaWAN chip on Cooking Hacks board, provides a library and a list of examples to use the LoRaWAN module to work easily with Cooking Hacks connection bridge. The connection between the LoRaWAN module and RPI3 is as follows: connect the connection bridge to the RPI3 board, then connect the LoRaWAN module to the connection bridge by placing it in the XBee socket slot as shown in Figure 6.1. The two LoRaWAN boards connected to both RPI3 boards need to be synchronized, so the radio setup is required. The radio setup, which is explained in the Appendix, is to set the output power, frequency, spreading factor, coding rate, and bandwidth to be the same on both nodes. Also, the CRC mode is enabled.

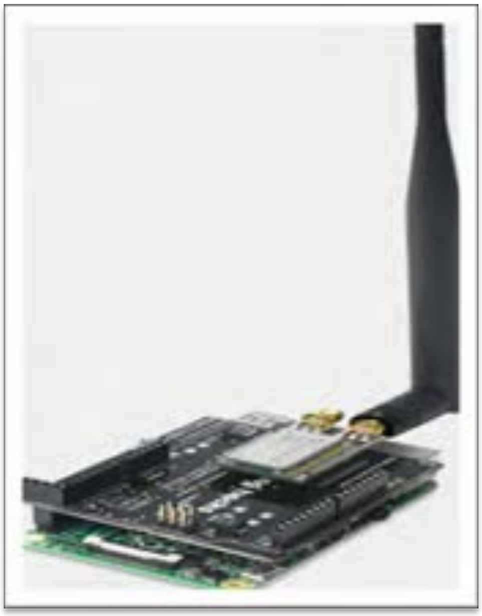

Figure 6.1: RPI3, Connection Bridge, & Cooking Hacks LoRaWAN Module Setup

After the setup, one GS gets the ToD as a message from the other ground station. The radio setup and the message transmission process are based on the Libelium examples. However, the codes in both nodes only support hexadecimal string; thus, a conversion code was used in both nodes. This code converts from ASCII string to hexadecimal string in the UAV node, then back to ASCII string in the receiving GS node. The configuration for both nodes can be found in the Appendix. The One-way ToA and Two-Way ToA functions used in the ToA measurements use the same implementation with slightly different handling of the messages.

For Arduino MEGA with Cooking Hacks LoRaWAN module, the setup of this part is similar to RPI3, but with some changes as shown in Figure 6.2. Libelium library only supports Arduino

UNO. Since Arduino UNO is not good to be used because of the dynamic memory deficiency in it, the Arduino MEGA board was used. The changes are in both the library and the Cooking hacks bridge for the Arduino boards. Also, Arduino MEGA requires Arduino IDE and the Libelium library to be downloaded in the Arduino folder. The changes can be found in the Appendix. The XBee slot to be used by default in the connection bridge is SOCKET1.

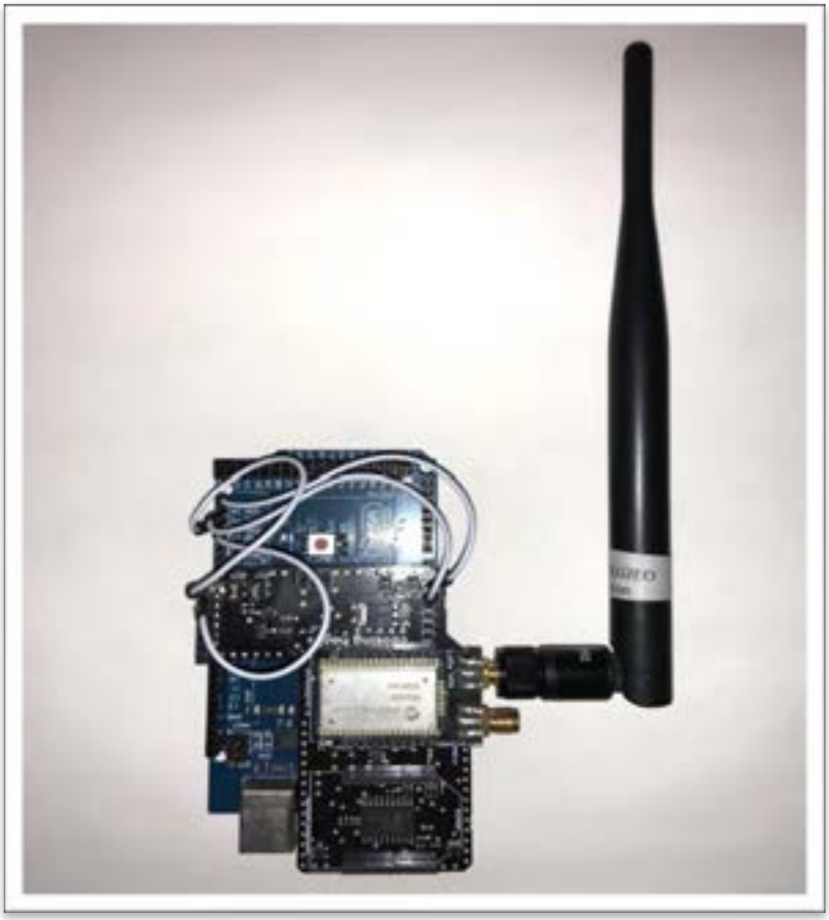

Figure 6.2: Arduino MEGA, Connection Bridge, & Cooking Hacks LoRaWAN Module Setup

Since both boards, RPI3 and Arduino MEGA, have similar implementations, the RPI3 results for both functions have only been shown. The One-Way ToA function output consists of 100 messages as shown in Figure 6.3.

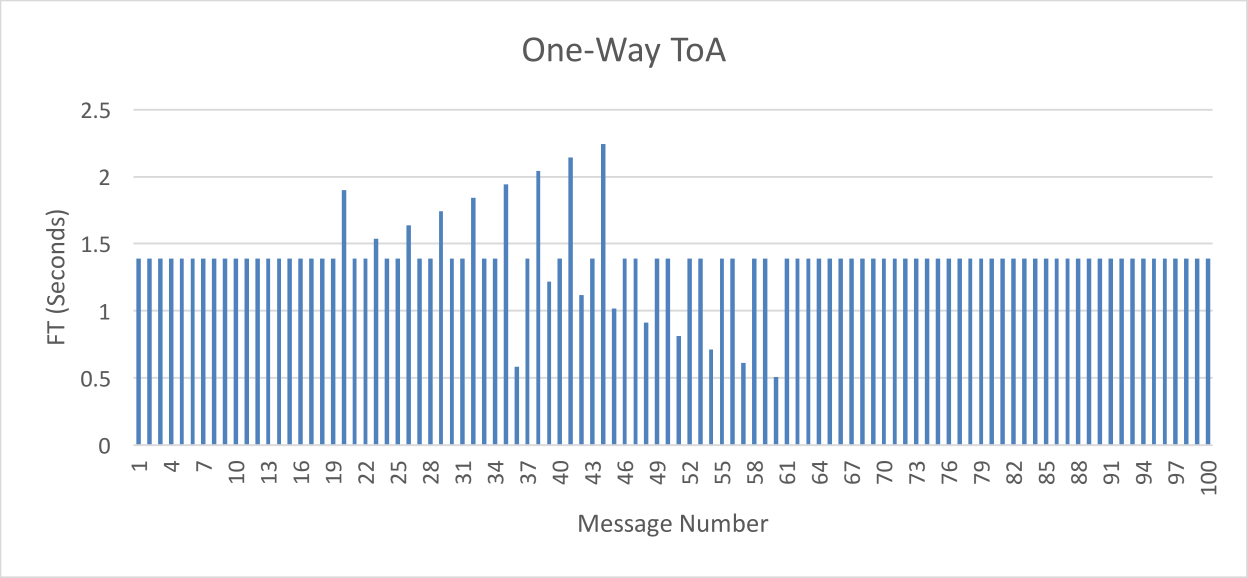

Figure 6.3: One-Way ToA Function Output

This part was to measure the FT for a message by placing the two nodes next to each other. Owing to the inconsistency of 40 percent observations, we tried two different ways to analysis the data: FT and FT excluding the observations in which FT values exceeded 2 seconds (FT-(>2)) as shown in Table 6.1.

Table 6.1: One-Way ToA Function Statistics (Seconds)

	FT	FT-(>2)
Mean	1.385557351	1.36281421
STDEV	0.246153444	0.206018831
STD_ERR	0.024615344	0.020601883

However, one second error in the FT can cause at around 300,000 kilometers error in the distance between the nodes. As seen from Table 6.1, the mean values for FT and FT-(>2) were more than 1s which can result in a large error in the FT. Furthermore, the variation between the samples was high, so the FT error was high, which will result in a large distance error.

Next, we tried the Two-Way ToA function. In this experiment similar to the one-way function, the FT and FT-(>2) were calculated as shown in Table 6.3. Since we are interested in the distance between the two nodes, the values in Table 6.3 have been divided by two since it is a round-trip message.

Table 6.2: Two-Way ToA Function Statistics (Seconds)

	FT	FT-(>2)
Mean	1.605085357	1.60169605
STDEV	0.119347272	0.11310448
STD_ERR	0.005337372	0.005058186

The Two-Way ToA function output consists of 500 packets as shown in Figure 6.4. This part was to measure the FT for a message by placing the two nodes next to each other. The mean values for the FT and FT-(>2) are more than 1s which results in a large error in the distance calculations after multiplying the FT with the speed of light. Consequently, this method failed again to be used to estimate the distance between the UAV and one of the GSs.

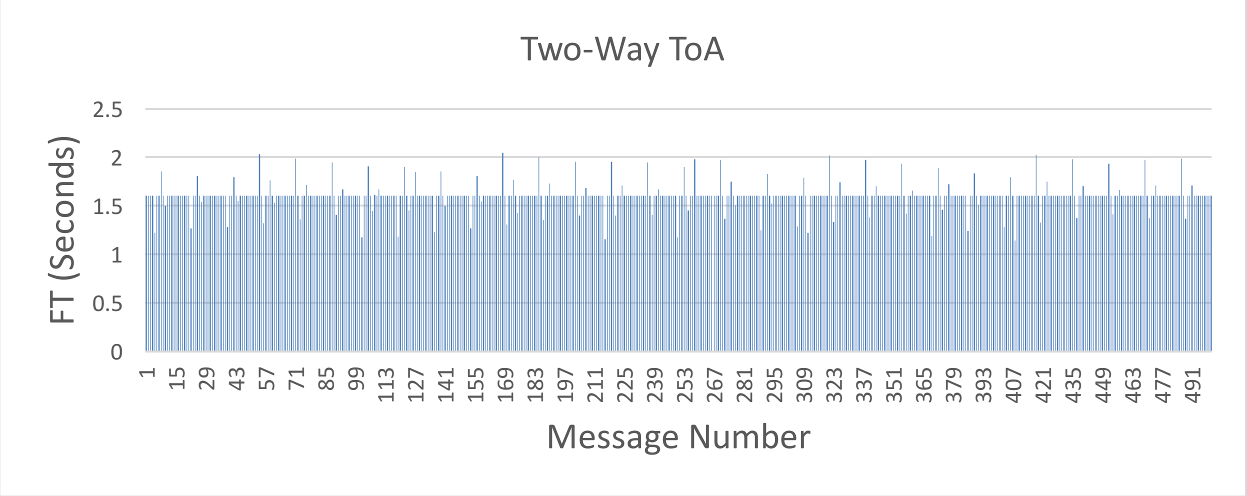

Figure 6.4: Two-Way ToA Function Output

6.2 Distance Estimation Using Counter

Since the ToA method did not work, the counter method was used. Again, the distance was estimated between two LoRaWAN boards to check for feasibility of this method. Due to the fact that RPI3 uses operating system to operate, time resolution with nanosecond accuracy was not possible. Even with Real-Time Operating System (RTOS), the scheduler works as soft real time and not hard real time. As a result, getting that accuracy with RPI3 was not possible unless the code is uploaded directly into the microprocessor without an Operating System (OS). For that reason, Arduino boards were chosen because they run as close to hard real-time system. Arduino MEGA was used to test this method. As in the ToA method, Cooking Hacks LoRaWAN module and connection bridge were used with the Arduino MEGA. Both nodes had the same configuration, which can be found in the Appendix. This includes the modifications in software and hardware previously mentioned in the ToA method.

To get accurate counter value for as small as one nanosecond, the ESP-12e board was used. This board has microcontroller with 160 MHz clock speed which gives as low as 6.25 ns time resolution compared to the fastest Arduino Due board that only gives 23.81 ns [31]. ESP-12e can provide a distance accuracy of 1.8 meters between the two nodes. Arduino IDE requires a setup to work with ESP-12e. This setup between the Arduino MEGA and the ESP-12e can be found in the Appendix, while the connection is shown in Figure 6.5. The ESP-12e uses the Micro USB cable to power-up.

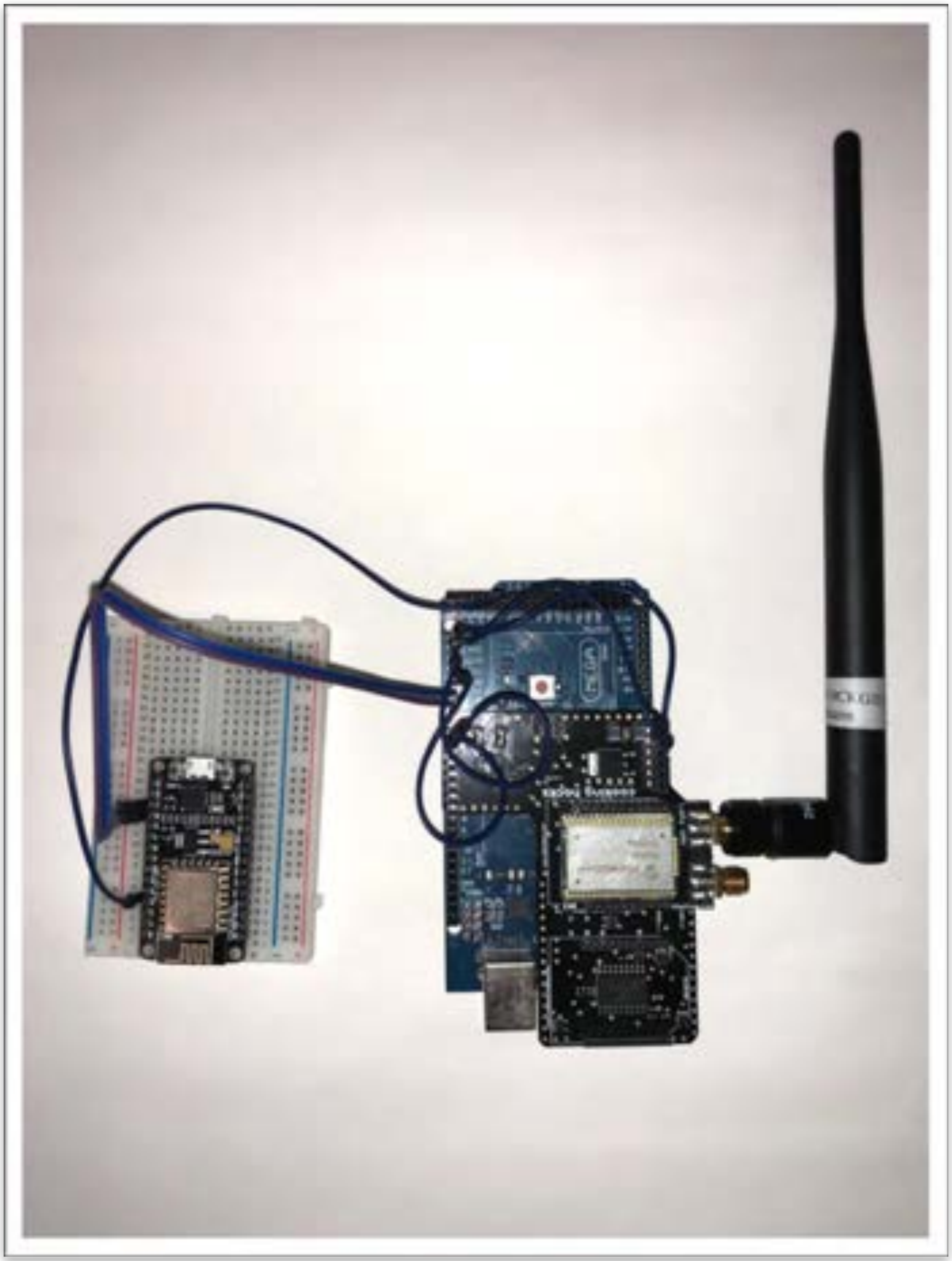

Figure 6.5: Arduino MEGA, Connection Bridge, Cooking Hacks LoRaWAN Module, and ESP-12e Setup

For the LoRaWAN module code, the same edited code in the Arduino MEGA used in the ToA method was used with the addition of the ESP-12e connection. The code in the ESP-12e was an edited version of the code in [31]. The serial pins were used to send the counter data from the

ESP-12e to the GS, while the digital pin in both nodes was used to let the GS control the starting and ending of the counter in the ESP-12e. One GS sends a message to the other node and at the same time, it triggers the counter in the ESP-12e. Then when the second board sends the message back, the GS sends a signal to stop the counter in the ESP-12e. The ESP-12e sends the counter value back to the GS. No results have been shown for this method because of the high variability in the counter values even though we used the same message at the same distance. This method failed to be used to estimate the distance between the UAV and one of the GSs.

6.3 Distance Estimation Using RSSI

Finally, the RSSI value was used to estimate the distance between the UAV and one of the GS since the counter method failed to achieve that. The boards in this experiment were attached in the UAV and the four GSs. Also, it was done in two different environments, both of them were outdoors. All the nodes in this experiment used Seeeduino LoRaWAN board that is based on Arduino Zero bootloader with LoRaWAN protocol embedded in it; thus, no extra board was needed [32]. All nodes had the same configuration, which can be found in the Appendix. Arduino IDE requires a setup to work with Seeeduino LoRaWAN board, which can be found in the Appendix. Seeeduino provides a library and examples to use their board. The codes in the UAV and the GSs are based on the examples provided from Seeeduino with as few lines as possible, and replacing most of the functions in the code with the actual commands to be sent directly to the LoRaWAN module.

The GS gets the UAV ID, RSSI, and signal-to-noise ratio (SNR) values from the messages sent from the UAV. Three messages with different lengths and format were tested as shown in Table 6.3.

Table 6.3: Different Message lengths and Formats List

#	Message	# of Bytes	Format
1	FF 31	2	Hexadecimal
2	FF1	3	String
3	FF1 is the UAV ID number that is being used to identify this UAV	66	String

Initial test was based on two nodes mounted on two tripods and not attached to the UAV, which explains the high RSSI values, with distances ranging from 100 to 500 meters. From Table 6.4, we see that the longer the message the higher the RSSI value and the longer distance it can reach. This is shown in Figure 6.6. Thus, the longest message, M3, was used to complete the rest of this experiment.

Table 6.4: RSSI Values for Different Message Lengths

Distance	100m	200m	300m	400m	500m
Message 1	-122.46	-119.90	-120.32	-121.73	-123
Message 2	-120.48	-119.11	-120.78	-124.23	-125.08
Message 3	-111.18	-108.74	-112.54	-114.77	-115.55

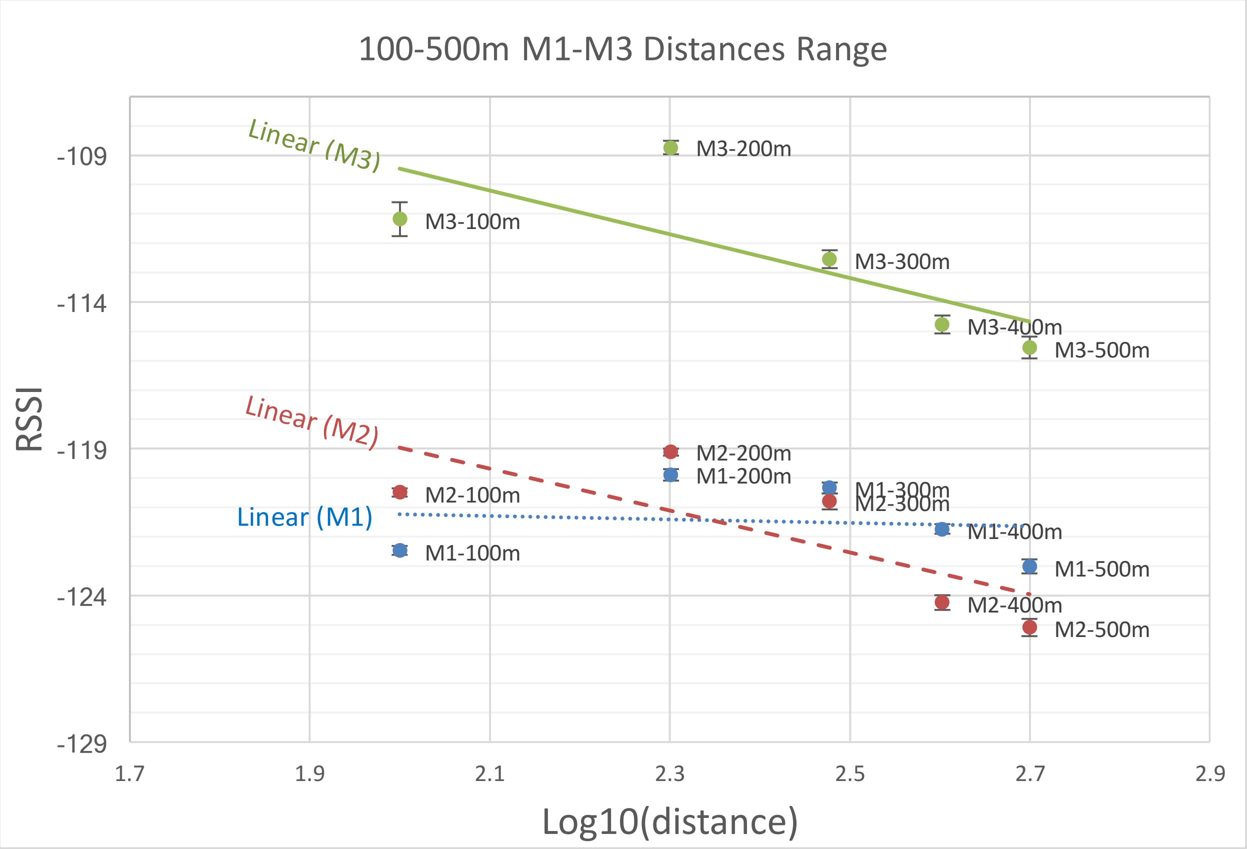

Fig 6.6: RSSI Values for Different Message lengths

The measurements for the model part consist of 6 mean RSSI values; each one of them consists of 125 samples in 6 different distance ranges: 100 to 600 meters. Initially, the height for the UAV was fixed to 50 meters to make the model less complicated. Furthermore, the RSSI values were the same for other heights up to 100 meters as long as the slant distance (SD) was same. The calculated SDs between the UAV and the GSs were really close to the actual distances as shown in Table 6.5. The UAV used to collect these RSSI values was DJI Phantom 2.

Table 6.5: RSSI Calculated Slant Distances

Nominal Distance	Ground Distance (GD)	Height (H)	UAV-GS angle (β)	GP-GS angle (α)	Slant Distance (SD)
100m	100m	50m	79.1°	10.9°	102.97m
200m	200.2m	50m	81.7°	8.3°	199.19m
300m	299.8m	50m	83.3°	6.7°	298.19m
400m	400.3m	50m	84°	6°	398.15m
500m	500.5m	50m	84.7°	5.3°	498.34m
600m	600.7m	50m	84.9°	5.1°	598.29m

Table 6.6 shows the notations for Table 6.7. The analyses of the collected data are shown in Table 6.7. The equations used to find each value in that table are also shown [33].

Table 6.6: Notations

Simple	Meaning
x	Individual RSSI mean from the 6 mean values
y	Individual $log_{10}(distance)$ from the 6 distances ranges
$\bar{x}$	The mean value for all the 6 mean values
$\bar{y}$	The mean value for all the 6 $log_{10}(distance)$
$\hat{y}$	The estimated value of every y after subtracting $\bar{y}$ from every y value
n	The number of samples (6 means in this case)
SSE	The sum of square errors
SST	The total sum of squares
SSR	The sum of squares explained by the regression
R^2	The coefficient of Determination

Table 6.7: Seeeduino LoRaWAN Board Collected Data Analysis (RSSI)

Nominal Distance	100m	200m	300m	400m	500m	600m
Sample variance (Var) $= (\frac{\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} x_i^2 /_n}{n-1})$	4.78	2.34	2.62	1.46	1.30	1.22
Sample Standard Deviation $(STDEV) = \sqrt{Var}$	2.19	1.53	1.62	1.21	1.14	1.10
Sample Standard Error $(STD\_ERR) = \frac{STDEV}{\sqrt{n}}$	0.20	0.14	0.14	0.11	0.10	0.10
Sample Mean $(\bar{x}) = \frac{\sum_{i=1}^{n} x_i}{n}$	-79.41	-82.94	-85.81	-85.58	-87.93	-88.32
95% Confidence Interval (CI) $= \bar{x} \mp \frac{1.96 * STDEV}{\sqrt{n}}$	(-79.79, -79.03)	(-83.21, -82.68)	(-86.09, -85.52)	(-85.79, -85.37)	(-88.13, -87.73)	(-88.52, -88.13)
b1 = $\frac{\sum xy - n\bar{x}\bar{y}}{\sum x^2 - nx^{-2}}$ = -11.654						
b0 = $\bar{y} - b_1\bar{x}$ = -56.134						
L = $\frac{b1}{-10}$ = 1.165						
C =$b0$ = -56.134						
Total Error $(\sum_{i=1}^{n} e_i) = \sum_{i=1}^{n} y_i - \hat{y} =$ 0.000						
SSE =$\sum_{i=1}^{n} e_i^2$ = 1.616						
SST =$\sum_{i=1}^{n}(y_i - \bar{y})^2$= 56.059						
SSR = $SST - SSE$= 54.444						
R^2 = $\frac{SSR}{SST}$ = 0.971						

The linear regression model of the RSSI mean values for all the distances is shown in Figure 6.7. From Figure 6.7, we can see that the model provides a good fit as showed in the high R^2 value. The model was trained with distances starting from 100 meters because of the high variability in the distances below that. After calculating the confidence interval (CI), we found that the CIs for 300 and 400 meter values overlapped. This means that the mean RSSI values for these two distances were not statistically different, which may cause an error of 100 meters in the distance calculations. At this point, we decided to check the RSSI value with other LoRaWAN boards.

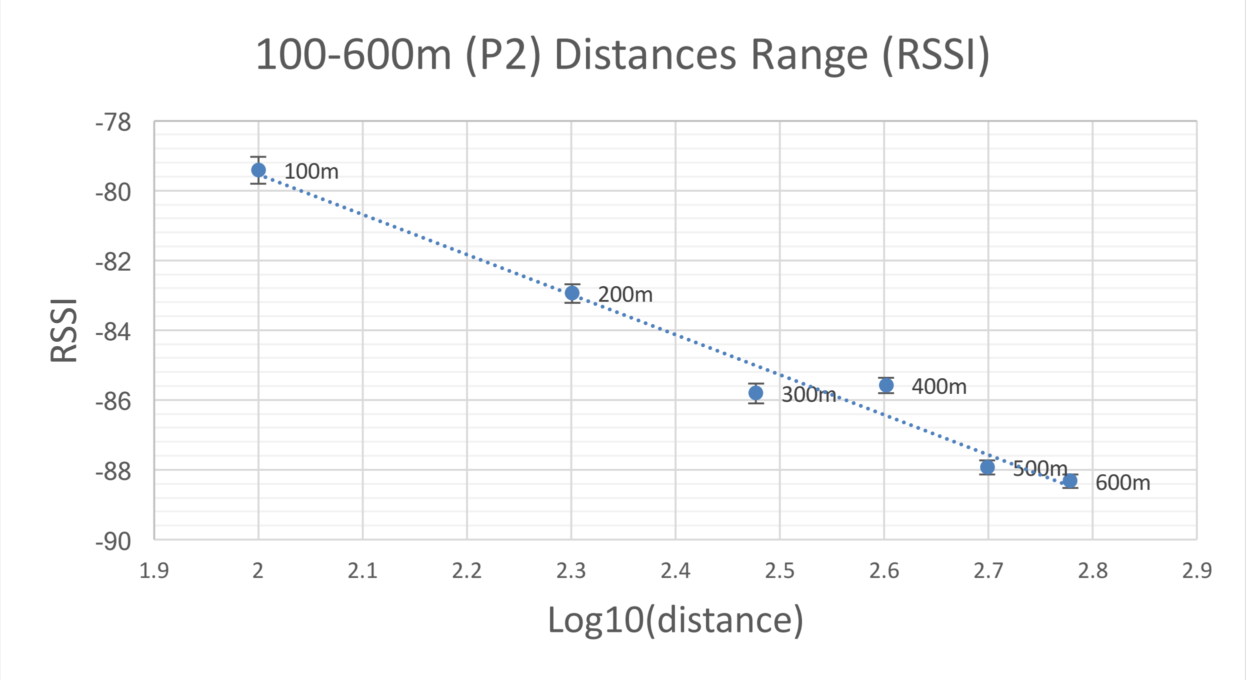

Figure 6.7: Linear Regression Model for measurements using Seeeduino LoRaWAN board

Seeeduino board was chosen over Moteino board because its RSSI values were almost consistent for different distance ranges, from 100 to 800 meters, as shown in Table 6.8. Furthermore, the perfect length of the message in Moteino board was message number 2. This is because for longer messages, message 3, the messages were transmitted in several pieces and would result in repeated RSSI values.

Table 6.8: Moteino LoRaWAN Board Collected Data Analysis (RSSI)

Distance	100m	200m	300m	400m	500m	600m	700m	800m
Sample Mean	-104.48	-103.68	-103.51	-104.97	-105.06	-104.86	-104.62	-104.65
Confidence Interval	(-104.61, -104.36)	(-103.81, -103.55)	(-103.61, -103.40)	(-105.03, -104.90)	(-105.11, -105.00)	(-104.93, -104.80)	(-104.72, -104.52)	(-104.75, -104.56)

6.4 Location Estimation Using RSSI

Location estimation stage consists of 4 GSs with one UAV. The UAV used in this stage was DJI Phantom 4. The GSs were 200 meters away from each other. All the GSs antennas' directions were up. The UAV antenna had a spring shape facing down as shown in Figure 6.8. The battery that was used to power-up the LoRaWAN board attached on the UAV was under the board itself as shown in Figure 6.8.

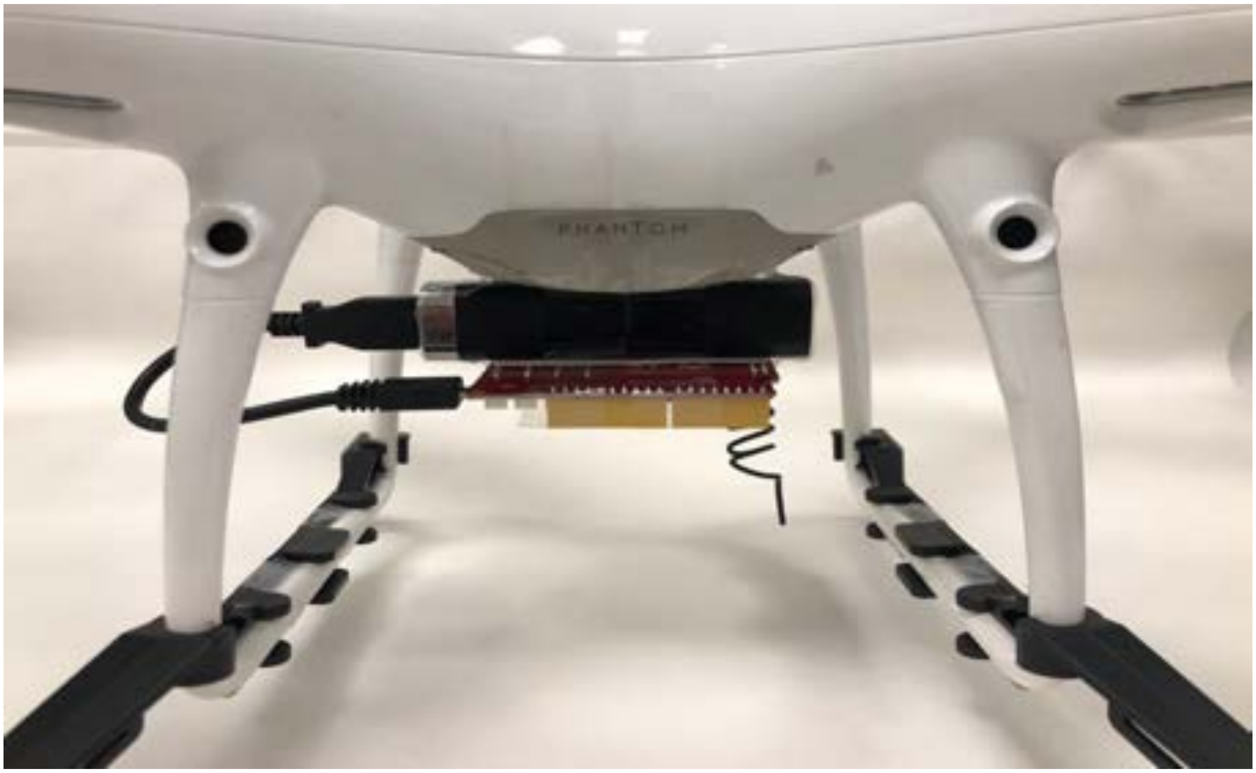

Figure 6.8: Seeeduino LoRaWAN board with the Battery attached to the UAV

The UAV height was set to 50 meters. The measured and estimated SD from the model is shown in Table 6.9. The real SD and H were measured using GPS, while the estimated SD and H were based on the RSSI value received from the UAV and the equations in Chapter 5. Since the height for all the GSs in this work was the same, 1.2 meters, the z value calculation was after getting 2D coordinates of the UAV as discussed in Chapter 5.

Table 6.9: Seeeduino LoRaWAN Board Location Data Analysis

	GS1	GS2	GS2 (Est.)	GS3	GS4
MeanRSSI	-80	-86	-80	-79	-81
Est. SD	112	367	112	92	136
Real SD	145	161	161	139	155
Mean % Error	24.8%				
Est. Location	(83.3; 93.5; #)				
Real Location	(88.1; 104.6; 50)				
% Error	(5.5%, 10.6%, #)				

Each meanRSSI value from each GS consists of the rounded-up mean of 5 RSSI values. Five meanRSSI values were collected from each GS. From these 5 meanRSSI values in each GS, the round-up mean is calculated. GS2 showed higher distance value than expected because that GS was in the other direction of the LoRaWAN board antenna that attached to the UAV. Because the meanRSSI values for GS1, GS3, and GS4 were reasonable and located inside the range, their mean value was used to replace the GS2 meanRSSI value. The mean percentage error for all the GSs estimated values was ~25 percent, which is expected by this method. Since all GSs estimated radius values to estimate the UAV height were negative under the square root, no estimated UAV height was shown. The estimated height from all GSs showed imaginary values because there was no real intersection between their spheres. MATLAB was used to write the equations for the Trilateration technique to determine the coordination from the information in Table 6.9. The results from the MATLAB showed the location of the UAV was very close to the real location as shown in Figure 6.9. The bold lines showed the real SD values and point to the real UAV location, while the dashed lines showed the estimated SDs and point to the estimated location.

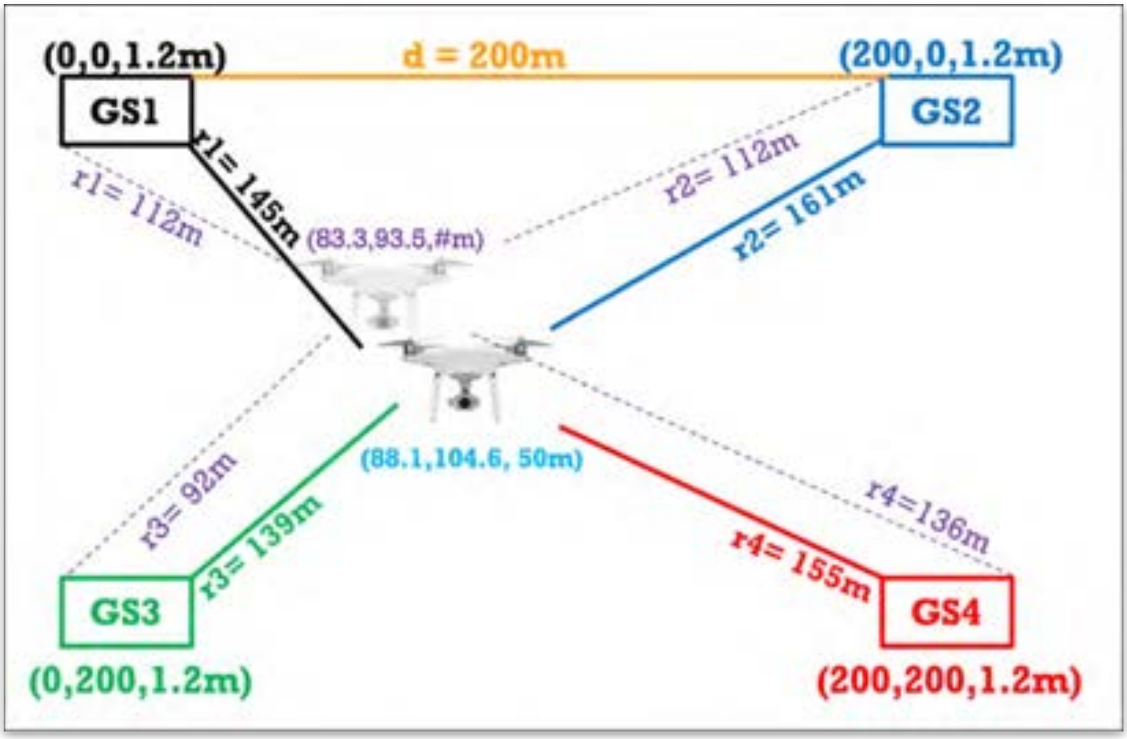

Figure 6.9: Location Estimation

Chapter 7

Discussion

In this work, three different methods were presented. These methods include: ToA, counter, and RSSI. A brief discussion about the results of each method has been presented.

The ToA method did not work because of the hardware boards, software functions, and time synchronization issue. The hardware boards used in this experiment were RPI3 and Arduino MEGA. RPI3 could not provide nanosecond time resolution because it uses non-real-time OS, while Arduino MEGA could provide as low as microsecond time resolution. The issue in Arduino was that it uses functions that give error of four μs and that is not acceptable for distance estimation because of the large error.

The counter was not good because of the counter board and the LoRaWAN module. The counter board used in this work was ESP-12e, which can count up to one millisecond before restarting the counter and losing the consistency of the counter value. Since LoRaWAN modules in the market use around 100 μs to complete only one command from a set of commands, it was difficult to use the counter with this protocol. In addition, the time for that command fluctuated by ± 4 μs. The whole code took around 2 seconds to complete one round-trip message between the two nodes. To the best of my knowledge, there is no cheap hardware that can keep counting from 1 ns to 2 s with high resolution and without losing the consistency of the counter value for the same message.

Since we were able to get the SNR value in the RSSI using Seeeduino LoRaWAN board, the noise value is easy to calculate. We found out that the noise value increases with the increase of distance between the two nodes. RSSI showed better results with some restrictions which are summarized as follows:

- Model accuracy: An accurate model needs to be built to use the method.
- Height: A specific height need to be taken into account while building the model to make the number of changing factors as low as possible, which makes it less complicated.
- Battery capacity: Different battery capacitates to run the LoRaWAN board attached to the UAV can cause different RSSI values for short ranges, below 300 meters.
- Antenna Direction: The antenna direction and position affect the RSSI value.
- Seeeduino LoRaWAN Board Power Cable: The cable used to provide the power to the board attached to the UAV should be in the opposite direction of the antenna to balance the power in all directions. This is because it acts as a second antenna for the Seeeduino LoRaWAN board.
- Battery Location: The battery used with the LoRaWAN board attached to the UAV needs to be under the board; otherwise, the RSSI value will be higher from the battery direction.
- Environments: Different environments affect the RSSI value because the model was based on a specific environment. As a result, the effect on the RSSI value somewhat equals the difference between the two environments.
- Modelling Range: The RSSI value for shorter distances (less than 100 meters) is not useable because of the high fluctuation in the RSSI values.

- Movement: The UAV should be in stationary mode for at least 10 seconds. This is because the LoRaWAN board that we used requires at least two seconds as the time interval between messages.

Location estimation was possible with the RSSI. However, with the constraints mentioned above, one of the GSs RSSI value showed a value that was out of the range in which the UAV should have been. This problem was solved temporarily by using the mean of the other three GSs RSSI values only if their values were acceptable. Also, the estimated UAV height was not possible to get since all the estimated SDs we got from the GSs were underestimated compared to the real values.

In summary, ToA and counter methods were not suitable to estimate the distance between the UAV and one of the GSs, while RSSI showed better results. RSSI was used to estimate the UAV location with some constraints.

Chapter 8

Future Work

The RSSI method looks promising for estimating the location of the UAV. It requires five LoRaWAN boards attached to the UAV and four GSs. The computed location was close to the real location of the UAV except that there was one totally wrong value for one of the GSs. This is because that GS location was in the other direction of the board's antenna. To solve this issue in the future, we plan to try LoPy boards. These boards allow using an external antenna, which supports longer distances and better signal power in different directions if installed in the middle of the UAV (or different antenna shape can be used). Thus, longer distances can be added to the model. Also, it supports the use of the LoRa physical layer and runs on ESP-32 board. This makes the board able to do better ToA calculations without the need to any additional board. The system now only works in the offline phase because the data are collected manually from the four GSs' computers. By adding additional LoRaWAN board to each GS or connect them by wires, we can make the system to estimate the location in real time. The system works only if the UAV is stationary in one spot for at least 10 seconds. To overcome this issue, different LoRaWAN boards will be tested to find the board that allows microsecond interval time between messages.

Chapter 9

Conclusion

Due to the increased popularity of UAVs in the United States, the need for a system to locate them is needed. Several solutions are available today, but most of them lack the ability to locate the non-line of sight UAVs. This thesis proposed an alternative to GPS in case it is not available or difficult to use. Three methods were used to estimate the distance, including ToA, counter, and RSSI. They have been used based on the order where the next method was used if the previous one failed. Only the RSSI method was found usable. After collecting data from different distances, ranging from 100 to 600 meters, the system was able to estimate the distance with ~75% accuracy if the antennas were correctly aligned. Four GSs were used to estimate the location of the UAV with some restrictions. Once the distances from the four GSs to the UAV are known, the location of the UAV including its height can be estimated. However, because the estimated distances were underestimated, there was no real intersection between the GSs spheres, resulting in not showing the estimated UAV height. The location estimation uses Trilateration technique to find the 3D coordinates of the UAV. The results showed almost 90% correct location estimation if the UAV is stationary for 10 seconds. This system can be used to alert law enforcement and locate any UAV if it enters a restricted area without permission.

References

[1] F. A. Administration. (2016). *FAA Releases 2016 to 2036 Aerospace Forecast*. Available: https://www.faa.gov/news/updates/?newsId=85227

[2] W. Bellamy, III. (2017). *Drones Came Too Close to Airplanes 1,800 Times in 2016*. Available: http://www.aviationtoday.com/2017/03/17/drones-came-close-airplanes-1800-times-2016/

[3] M. Zanona. (2017). *Drone sightings near airplanes on the rise*. Available: http://thehill.com/policy/transportation/362556-drone-sightings-near-airplanes-on-the-rise

[4] C. Shine. (2017). *FAA playing high-tech hide and seek at DFW to stop drones from colliding with flights*. Available: https://www.dallasnews.com/business/dfw-airport/2017/04/28/1800-drone-sightings-last-year-faa-testing-systems-prevent-airport-collisions

[5] F. A. Administration. *DRONE REGISTRATION*. Available: http://federaldroneregistration.com/?gclid=CjwKCAiAxarQBRAmEiwA6YcGKHV8CMGVSgIsDn0cSsvI41X-n7Z41MB3aFl9iIXuRf8Bx2p6Co7QyRoCYBcQAvD_BwE

[6] DJI. (2015). *DJI Introduces New Geofencing System for its Drones*. Available: https://www.dji.com/newsroom/news/dji-fly-safe-system

[7] K. D. Atherton. (2015). *This Device Turns Any Gun Into An Anti-Drone Ray*. Available: https://www.popsci.com/dronedefender-is-an-anti-drone-rifle-attachment

[8] K. D. Atherton. (2016). *This drone gun knocks drones out of the sky gently, with radio waves*. Available: https://www.popsci.com/drone-gun-downs-drones-with-radio-waves

[9] J. Lee. (2016). *8 Ways to Prevent Drones Infringing on Your Privacy*. Available: http://www.makeuseof.com/tag/8-ways-prevent-drones-infringing-privacy/

[10] Wikipedia. *Regulation of UAVs in the United States*. Available: https://en.wikipedia.org/wiki/Regulation_of_UAVs_in_the_United_States

[11] L. Alliance. (2015). *A technical overview of LoRa® and LoRaWAN™*. Available: https://docs.wixstatic.com/ugd/eccc1a_ed71ea1cd969417493c74e4a13c55685.pdf

[12] Joseph Rafferty, Jonathan Synnott, Andrew EnnisIan, ClelandChris, and a. N. Little, "A Secure, Out of Band, Mechanism to Manage IoT Devices," presented at the Ubiquitous Computing and Ambient Intelligence: 11th International Conference, PA, USA,, 2017.

[13] R. Jain. (2016). *Low Power WAN Protocols for IoT: IEEE 802.11ah, LoRaWAN*. Available: http://www.cse.wustl.edu/~jain/cse574-16/ftp/j_14ahl.pdf

[14] htdvisser. (2017). *LoRaWAN*. Available: https://www.thethingsnetwork.org/wiki/LoRaWAN/Home

[15] Wikipedia. *Global Positioning System*. Available: https://en.wikipedia.org/wiki/Global_Positioning_System

[16] Wikipedia. (2017). *Distance measuring equipment*. Available: https://en.wikipedia.org/wiki/Distance_measuring_equipment

[17] Wikipedia. (2009). *Received signal strength indication*. Available: https://en.wikipedia.org/wiki/Received_signal_strength_indication

[18] J. W. Rosen. (2017). *Zipline Expands Drone Delivery of Medical Supplies*. Available: https://www.technologyreview.com/s/608696/zipline-expands-drone-delivery-of-medical-supplies/

[19] Amazon. *Amazon Prime Air*. Available: https://www.amazon.com/Amazon-Prime-Air/b?node=8037720011

[20] Anwen Wang, Xiang Ji, Dajun Wu, Xuedong Bai, Nana Ding, S. C. Jing Pang*, et al.*, "GuideLoc: UAV-Assisted Multitarget Localization System for Disaster Rescue," *Mobile Information Systems,* vol. 2017, pp. 1-13, 2017.

[21] Xiaohua Tian, Zhenyu Song, Binyao Jiang, Yang Zhang, Tuo Yu, and a. X. Wang, "HiQuadLoc: A RSS Fingerprinting Based Indoor Localization System for Quadrotors," *IEEE Transactions on Mobile Computing,* vol. 16, pp. 2545 –2559, 2017.

[22] G. Greco, C. Lucianaz, S. Bertoldo, and a. M. Allegretti, "Localization of RFID tags for environmental monitoring using UAV " *2015 IEEE 1st International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI),* pp. 480-483, 2015.

[23] Sergio Ferreira, Guilherme Carvalho, Filipe Ferreira, and a. J. Sousa, "Assessing the capacity of man-portable UAVs for network access point localization, using RSSI link data," *2014 International Conference on Unmanned Aircraft Systems (ICUAS),* pp. 355-364, 2014.

[24] L. Cheng, C. Wu, Y. Zhang, and a. Y. Wang, "An Indoor Localization Strategy for a Mini-UAV in the Presence of Obstacles," *International Journal of Advanced Robotic Systems,* vol. 9, 2017.

[25] D. Ismail, "Experiment Design and Analysis of a Mobile Aerial Wireless Mesh Network for Emergencies," Master Degree MS Project, Department of Computer Science and Engineering, Washington University in Saint Louis.

[26] Y. Li, "Seamless Handover among WiFi Access Points on Raspberry Pi," Master Degree MS Thesis, Department of Computer Science and Engineering, Washington University in Saint Louis, 2015.

[27] W. Dargie and a. C. Poellabauer, *Fundamentals of Wireless Sensor Networks*: John Wiley & Sons, Ltd, 2010.

[28] T. Salman, "Parameters Estimation to Achieve Distance-Based Security Breaching," Master thesis, Computer Science and Engineering, Qatar University (Doha-Qatar), Spring 2015.

[29] *The Law of Cosines*. Available: https://www.mathsisfun.com/algebra/trig-cosine-law.html

[30] Wikipedia. *Trilateration*. Available: https://en.wikipedia.org/wiki/Trilateration

[31] R. Tyagi. (2017). *6.25 Nanosecond Resolution Timer for Any Microcontroller!* Available: http://www.instructables.com/id/625-Nanosecond-Resolution-Timer-for-Any-Microcontr/

[32] (2016). *Seeeduino LoRaWAN*. Available: https://www.seeedstudio.com/Seeeduino-LoRaWAN-p-2780.html

[33] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. New York, NY: Wiley-Interscience, April 1991.

Acronyms

AoA	Angel of Arrival
APs	Access Points
CI	Confidence Interval
DME	Distance Measuring Equipment
DT	Delta Time
ESP-12e	ESP8266 SMT Module
FAA	Federal Aviation Administration
FT	Flight Time
GD	Ground Distance
GPS	Global Positioning System
GS(s)	Ground Station(s)
LoRaWAN	Long Range Wide Area Network
LoS	Line-of-Sight
MAC	Media Access Link
meanRSSI	RSSI mean value
NLOS	Non-Line-of-Sight
PSO	Particle Swarm Optimization
RFID	Radio-Frequency Identification
RPi3	Raspberry Pi 3
RSSI	Received Signal Strength Indication
SD	Slant Distance
SNR	Signal-to-Noise Ratio
STDEV	Standard Deviation

STD_ERR	Standard of Error
ToA	Time of Arrival
ToD	Time of Departure
UAV(s)	Unmanned Aerial Vehicle(s)
VoIP	Voice over IP
WDS	Wireless Distribution System

Appendix

A. Configuration:

- Cooking Hacks LoRaWAN:

Parameter Name	Value
Output Power	15
Frequency	433375000
Spreading Factor	sf12
Coding rate	4/5
Bandwidth	125
CRC mode	on

- Seeeduino LoRaWAN:

Parameter Name	Value
Frequency	433
Spreading Factor	sf12
Bandwidth	BW125
txPreample	8
rxPreample	8
Output Power	20

- Moteino LoRaWAN:

Parameter Name	Value
Output Power	13
Frequency	915
Spreading Factor	4096
Coding rate	4/5
Bandwidth	125
CRC mode	on

B. Wire Connection:

- Cooking Hacks LoRaWAN with Arduino MEGA:

  1. Software:
     - The best way to make it work without any issue is by changing all “serial0” to “serial1” at the end of “arduinoUART.cpp” code in the library.

  2. Hardware:
     - Bend the connection bridge pins that connected to pins number “0 and 1” in the Arduino MEGA.
     - To use Serial1 for the LoRaWAN module and Serial0 for the Arduino MEGA output, do the following modification in the connection bridge. Use Male-to-Male wires to connect those pins to pins number “18 and 19” in the Arduino MEGA.
     - Use Male-to-Male wires to connect the connection bridge pins that connected to pins number “A4 and A5” in the Arduino MEGA to pins number “20 and 21” in the Arduino MEGA. Because i2c pins in Arduino MEGA are different than the ones in Arduino UNO.

- ESP-12e with Arduino Mega:

  - Use Male-to-Male wire to connect D0 pin in the ESP-12e to any digital pin in Arduino MEGA such as a pin number “31”. Use two Male-to-Male wires to connect the “Tx and Rx” pins in the ESP-12e to pins number “16 and 17” in the Arduino MEGA.

C. Arduino IDE Setup:

- ESP-12e:

  - Add this link to the additional boards manager URLs in the Preferences settings: http://arduino.esp8266.com/stable/package_esp8266com_index.json

  Add the “esp8266” boards from the board manager in the tools section.

- Seeeduino LoRaWAN:

  i. Add this link to the additional boards manager URLs in the Preferences settings: https://raw.githubusercontent.com/Seeed-Studio/Seeed_Platform/master/package_seeeduino_boards_index.json

  ii. Add the “Seeeduino SAMD” boards from the board manager in the tools section.

- Moteino LoRaWAN:

  - Add this link to the additional boards manager URLs in the Preferences settings: https://lowpowerlab.github.io/MoteinoCore/package_LowPowerLab_index.json

  - Add the “Moteino & MightyHat” boards from the board manager in the tools section.