

# Florida State University Libraries

---

Honors Theses

The Division of Undergraduate Studies

---

2014

## Autonomous RC Aircraft with Collision Avoidance Capabilities

Cristopher Timmons Jr.



# Abstract:

(Autonomous, ADS-B, Aircraft)

This thesis directly deals with setting up all of the electrical components to create an inexpensive automatic dependent surveillance broadcast (ADS-B) receiver that can transmit the ADS-B data received to a ground control station (GCS) using a Raspberry Pi (single chip computer), operating an autopilot system from the same GCS, and sending the telemetry data to and from the autopilot system and the GCS also using the Raspberry Pi. Furthermore, it will discuss how this setup allows the system to be capable to performing autonomous collision avoidance maneuvers. Finally, it will show how these capabilities can be demonstrated in a Hardware-in-the-Loop Simulation (HILSim). As the member in charge of performing autonomous flight for the Autonomous Aerial Vehicle senior design team last year, it was ideal to continue research in the field of autonomous flight. The original goal was to compete in the Unmanned Air Systems (UAS) Airspace Operations Challenge (AOC) NASA Centennial Challenge, however, not enough funds were acquired to enter this competition. Therefore, the focus of the thesis shifted to setting up the electrical components of an RC aircraft that would be able to meet the general requirements of the competition.

THE FLORIDA STATE UNIVERSITY  
FAMU-FSU COLLEGE OF ENGINEERING

AUTONOMOUS RC AIRCRAFT WITH COLLISION AVOIDANCE CAPABILITIES

By

CRIS TIMMONS

A Thesis submitted to the  
Department of Electrical Engineering  
in partial fulfillment of the requirements for graduation with  
Honors in the Major

Degree Awarded:

[Spring, 2014]

The members of the Defense Committee approve the thesis of Cris Timmons defended on April 25, 2014.

---

Dr. Michael P. Frank, Ph.D.

Thesis Director

---

Dr. Chiang Shih, Ph.D.

Outside Committee Member

---

Dr. Simon Y. Foo, Ph.D.

Committee Member

This thesis is dedicated to all of my friends and family. Without them I would not be where I am today. Particularly, I would like to thank my mom, dad, and brother for all of their help and support throughout the years. Also, I would like to dedicate this thesis to my both of my uncles who were pilots and inspired my interest in aeronautics. Especially, my Uncle Butch who passed away in a plane crash that could have easily been avoided. This fueled my desire to do my thesis on safe airspace operations and collision avoidance maneuvers.

## Table of Contents

Table of Figures .....	vi
Problem Statement and Background .....	1
Goals .....	1
Funding .....	2
Bess H. Ward Honors Thesis Award .....	2
Outside Funding .....	2
Competition Background .....	2
Phase 1 Challenge Focus .....	3
Phase 2 Challenge Focus .....	3
Phase 1 Rules, Regulations, and Scoring .....	3
Top Level Requirements .....	3
General Requirements .....	4
Detailed Requirements .....	4
Bonus Scoring Requirements .....	5

5	Concept of Operations .....
6	Scoring .....
6	Separation Scoring .....
8	Mission Scoring .....
	Airmanship ..... 8
8	Uncooperative Traffic Detection and Tracking (Optional) .....
9	GPS Integrity Scoring (Optional) .....

Scoring Summary .....	9
Registration .....	9
New Goals .....	10
Systems Overview .....	11
Mechanical Components .....	13
Gas/Nitro RC Plane .....	13
Storing the Electrical Components .....	13
Electrical Components .....	14
Electrical Components – Hardware .....	14
Autopilot System .....	14
GPS .....	15
Raspberry Pi and 8 GB SD Card .....	15
TTL Level Shifter .....	16
NooElec SDR USB Dongle .....	16
Huawei E303 3G USB Dongle .....	16
Belkin Wireless G USB Network Adapter .....	17
PiHub .....	17



	DX5e	DSMX	5-Channel	Transmitter/Receiver
.....	18			
18	Ground Control Station	.....		
19	Electrical Components - Powering the System	.....		
	NiMH Battery Pack	.....	19	
19	Portable Power Pack	.....		
19	Electrical Components – Software	.....		
19	Mission Planner	.....		
20	ArduPlane Firmware and HILSim Firmware	.....		
20	X-Plane	.....		
21	Plane Plotter	.....		
21	VPN Server	.....		
	Raspbian	.....	22	
	Dump1090	.....	22	
22	PPP Package, UMTSKeeper, and Sakis 3G	.....		
23	Ser2net	.....		
23	Electrical Components - Miscellaneous Components	.....		

Establishing a 3G Connection .....	23
Setting up the ADS-B Receiver .....	23
Dump1090 .....	23
Plane Plotter .....	24
Connecting the RPi, APM, and MP .....	25
Hardware Configuration .....	25
Ser2net .....	25
Mission Planner .....	26
Setting up the Receiver/Transmitter .....	26
Setting up Mission Planner .....	27
Setting up a Work Area .....	28
Setting up a Geo-Fence .....	28
Setting up a Rally Point .....	28
Hardware in the Loop Simulations .....	28
Collision Avoidance Maneuvers .....	29
Budget .....	30

Conclusion .....	32
Appendix .....	33
Code to Run Dump1090 on OS Boot .....	33
Code to Establish a 3G Connection on OS Boot .....	35
Ser2net Configuration File .....	35
Pseudo-code to Perform an Autonomous Collision Avoidance Maneuver .....	35
Sponsorship Letter .....	38
Bibliography .....	40
Special Thanks .....	43

## Table of Figures

Figure 1: UAS AOC NASA Centennial Challenge Phase 1 .....	2
Figure 2: Tracker Module .....	4
Figure 3: HiLSim Example Setup .....	6
Figure 4: Example Mission Layout .....	7
Figure 5: Example of a Collision Avoidance Maneuver to Avoid Cooperative Aircraft .....	8
Figure 6: Air Traffic Encounter Geometries .....	8
Figure 7: Challenge Separation Distances .....	8
Figure 8: Waypoints set for Mission Scoring .....	9
Figure 9: Scoring Summary .....	10
Figure 10: Top Level Design Diagram for Thesis .....	11
Figure 11: Photo of Components Used .....	12
Figure 12: Top Level Design Diagram of RC Aircraft for Competition .....	12
Figure 13: Senior Telemaster (Used in last year's senior design project) .....	13

Figure 14: ArduPilot Mega 2.5 .....	14
Figure 15: 3DR GPS uBlock LEA-6 .....	15
Figure 16: Raspberry Pi .....	15
Figure 17: 8GB SD Card w/ NOOBS .....	15
Figure 18: TTL Level Shifter .....	16
Figure 19: NooElec SDR, Antenna, and Remote .....	16
Figure 20: Huawei E303 3G USB Dongle .....	17
Figure 21: Belkin Wireless G USB Network Adapter .....	17

Figure 22: PiHub .....	18
Figure 23: Belkin F4U006 4-Port Travel USB Hub .....	18
Figure 24: DX5e DSMX 5-Channel Transmitter/Receiver .....	18
Figure 25: 4.8 V 500 mAh NiMh Battery Pack .....	19
Figure 26: Portable Power Pack for the Raspberry Pi .....	19
Figure 27: Example Mission in Mission Planner .....	20
Figure 28: Downloading Firmware to the ArduPilot Mega using Mission Planner .....	20
Figure 29: Setup of a HILSim using X-Plane .....	21
Figure 30: PlanePlotter – Chart View .....	21
Figure 31: PlanePlotter – Aircraft View .....	21
Figure 32: Dump1090 .....	22
Figure 33: Plane Plotter I/O Settings and Mode-S Receiver IP Address .....	24
Figure 34: Connection Diagram of the Raspberry Pi and the ArduPilot Mega .....	25
Figure 35: ArduPilot Mega and Receiver Connection .....	27
Figure 36: Radio Calibration in Mission Planner .....	27
Figure 37: Simulation tab of Mission Planner and HILSim using X-Plane .....	29

Figure 38: Budget List .....	31
Figure 39: Budget of Competition Pie Charts .....	31

## **Problem Statement and Background**

The purpose of this project is to construct an Unmanned Aerial System (UAS) that will fly autonomously and be capable of performing autonomous collision avoidance maneuvers. In 2013 the Aviation Safety Network recorded 101 plane crashes and 3,746 casualties for corporate jets and military transportation in 2013-2004. This number is down from 142 plane crashes and 4,596 casualties in 1994-2003, and 192 plane crashes and 6,105 casualties in 1984-1993.<sup>1</sup> As this statistic shows, we are on the right path in flight safety and autonomous collision avoidance maneuvers may be the next step in further decreasing the number of these incidents occurring. Furthermore, the Drone Crash Database shows an incomplete list of large Unmanned Aerial Vehicle (UAV) crashes that include over 120 incidents in 2007-2013. This list does not include mini-UAV because “they crash so often it would swamp the database”.<sup>2</sup> These statistics illustrate the need for further research to improve the functionality and safety of this technology.

## **Goals**

The original purpose of this project was to form a team and design a UAS to compete in the first Unmanned Air Systems (UAS) Airspace Operations Challenge (AOC) NASA Centennial Challenge. This encompassed assembling a team of Electrical, Computer, and Mechanical Engineers, raising the funds to construct the UAS and cover the registration fee, and actually constructing the UAS. The Fall semester was mainly comprised of designing a top level design for the project, trying to raise the money to construct the UAS and enter the competition, and assembling a team. As the year progressed, raising the funds to cover the registration fee proved to be a rather daunting task. The UAS itself would cost about \$1,150 to construct (\$750 for electronics and \$400 for the RC aircraft and spare parts), while early registration fee for the competition was priced at \$5,000, roughly five times more than the cost of the UAS. As the year progressed the registration fee continued to increase to \$7,000 and, finally, \$9,000. Late in the Fall semester the Bess H. Ward scholarship was received, which covered the cost of the electronics, but left the cost of the RC aircraft and the registration fee to still be covered. A sponsorship packet was sent to various companies (large corporations, local Tallahassee businesses, and local businesses from my home town) in an attempt to cover the rest of the project costs and registration fee. However, no more funding was received. Due to the lack of funding and the inability to cover the registration fee, a team was never truly formed. In the Spring semester the focus shifted to personally ordering and setting up the electronics of the UAS.

---

<sup>1</sup> "Aviation Safety Network Statistics Flight Nature Domestic Scheduled Passenger Accidents." *Aviation Safety Network Statistics Flight Nature Domestic Scheduled Passenger Accidents*. N.p., n.d. Web.

<sup>2</sup> Ibid



## **Funding**

The amount of funding needed to complete the project and enter the competition varied from \$6,150 to \$10,150, depending on when the registration fee was paid. However, the amount of funding needed to complete the project, minus the registration fee, amassed to only \$1,150. Furthermore, the cost of all of the electronics for the UAS amassed to only \$750. The funding methods for the project are discussed below.

### **Bess H. Ward Thesis Award**

This scholarship is awarded to student conducting their Honors in the Major Thesis and awards up to \$750. This scholarship was received late in the Fall semester and the funds were dispersed early February. The scholarship covered all of the components except for the RC aircraft itself. It played an essential part in funding for the project and allowed for all of the electronics to be ordered early in the Spring semester. As it turned out, no more funding was received, so this scholarship was vital in the continuation of the project.

### **Outside Funding**

Even with the Bess H. Ward scholarship of \$750, there was still \$5,400 to \$10,400 left to be raised to reach the \$6,150 to \$11,150 milestone. Therefore, outside funding would have been required to cover the rest of the cost of the aircraft and the registration fee. A sponsorship letter was drafted and sent to corporations, local Tallahassee businesses, and local businesses from my home town asking for funding for the project. However, no more funding was received. The sponsorship letter used is shown in the Appendix.

## **Competition Background**



**Figure 1: UAS AOC NASA Centennial Challenge Phase 1**

This was the first year that NASA conducted the UAS AOC Centennial Challenge. The goal of the challenge was to demonstrate a high level of operational robustness with a UAS as well as demonstrate that the UAS is capable of detecting and avoiding other air traffic in order to make integration into the National Airspace System (NAS) possible. The competition is broken into two phases. The first phase will take place in September 2014 and the second phase will be held roughly a year later. The first phase was extended from April to September due to lack of teams

registered. The first phase will award \$500,000 in prizes and the second phase is expected to allocate \$1,000,000 for prizes.<sup>3</sup> The two phases are discussed in further detail below.

### **Phase 1 Challenge Focus**

The first phase of the competition requires the aircraft to accurately fly a 4-D trajectory as well as provide safe separation from surrounding cooperative (ASD-B equipped) aircraft. The teams are encouraged to display good airmanship, operating the aircraft with competence both on ground and in the air as well as demonstrating sound judgment that results in operational safety and efficiency. The teams are also encouraged to incorporate sensors to detect uncooperative aircrafts and provide robustness to GPS failures which will be evaluated in the second phase of the challenge.<sup>4</sup>

### **Phase 2 Challenge Focus**

The main goal of the first phase of the challenge is to prepare the competitors for the second phase of the challenge which will be much more difficult. In this phase the teams must accurately fly a 4-D trajectory, communicate verbally with Air Traffic Controller under lost link conditions, provide robustness to GPS failures, and safely avoid both cooperative and non-cooperative air traffic.<sup>5</sup> The original goal was to enter in first phase of the competition so that the second phase could become a senior design project for electrical, computer, and mechanical engineers next year. However, now the goal is to setup a thorough guide to setting up the electronics for the first part of the project. If a future senior design team would like to pick up the project, I plan to provide them with a methodical and systematic guide to expedite the process of determining what equipment would need to be purchased and setting up the electronics in the RC aircraft.

## **Phase 1 Rules, Regulations, and Scoring**

### **Top Level Requirements**

Competitors in the first phase of the competition are required to demonstrate the ability to operate safely within the constraints of the designated airspace and were awarded points on how well the UAS operated based on the requirements listed below. Competitors also have the ability to earn bonus points by demonstrating the capability to perform some of the requirements of the second phase of the challenge.<sup>6</sup>

---

<sup>3</sup> "Latest News." *UAS Airspace Operations Challenge*. NASA, n.d. Web.

<sup>4</sup> Ibid

<sup>5</sup> Ibid

<sup>6</sup> Ibid

## General Requirements

1. Single aircraft operated from a ground control station (GCS)
2. Multiple 30 minute missions per day
3. Operate within defined airspace boundaries
4. Execute assigned mission
5. Execute missions on schedule
6. Ground demonstration of flight software through simulation
7. Self separation from other air traffic
8. ADS-B receiver integrated onboard
9. ADS-B traffic displayed graphically in GCS
10. Tracker module integrated onboard aircraft

## Detailed Requirements

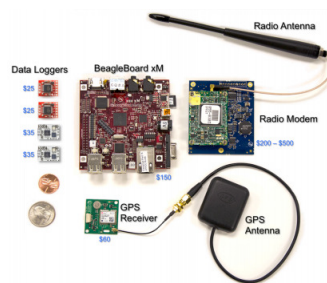
The aircraft must be equipped with an ADS-B receiver that is capable of receiving UAT broadcasts (978 MHz) and acted as an “ADS-B In”. The received data on the surrounding aircrafts location and heading must be displayed graphically in a clear and concise manner in the team’s operating environment so that it provides sufficient situational awareness for the operating crew.<sup>7</sup>

The tracker module integrated onboard the aircraft is used to provide independent, real-time telemetry of the aircraft position to the judges. The tracker also acted as an official data logging service for the competitors, which records sensor data and event records in flight that will be used in post-mission scoring. The tracker module will be given to the competitors upon completion of competition check-in. The module is configured with a mix of TTL or RS-232 serial interfaces and a Linux computer can be used to decode data streams and log specific parameters, using the competitor’s software. The module requires a 5 V power supply and the peak current draw is expected to be 1.5 A.<sup>8</sup> A picture of the tracker module’s components is shown below in Figure 2.

---

<sup>7</sup> UAS AOC

<sup>8</sup> Ibid



**Figure 2: Tracker Module**

A Hardware-in-the-Loop Simulation (HiLSim) is required to demonstrate certain safety-related flight modes of the UAS on the ground before the UAS is permitted to fly in the competition. The HiLSim can either be connected directly into the aircraft's autopilot or to an identical, uninstalled autopilot. The judges will use these simulations to verify the connection of the autopilot system to the GCS, the functionality of the tracker module, the ability to receive the ADS-B messages and accurately display them on the GCS, and that every safety-related flight mode can be triggered. The competitors are also required to walk through a simulation of the competition mission before being able to conducting the mission.<sup>9</sup> A diagram of the HiLSim setup connected directly to the aircraft's autopilot system is shown below in Figure 3.



Figure 3: HiLSim Example Setup

### Bonus Scoring Requirements

1. Jucker module integrated onboard aircraft (to interrupt or modify GPS signal)
2. Detect and avoid sensors onboard aircraft
3. Additional requirements for UAS Surrogates (N/A)

### Concept of Operations

Before being able to conduct the competition flight, the aircraft must undergo certain inspections and ground tests. This includes a physical inspection of the aircraft and the HiLSim-based tests.<sup>10</sup>

Following the ground tests the aircraft will undergo a solo Qualification Flight with no other air traffic present. This Check Flight must be completed before the aircraft would have been allowed to take part in the scored competition missions.<sup>11</sup>

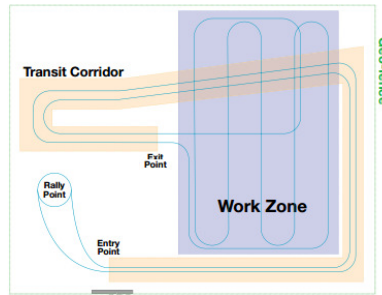
Finally, the aircraft will undergo a series of missions that were assigned by the judges. The missions will include a transit corridor, work area for the mission, and an amount of time that should be spent in the work area. Competitors are required to provide details on the aircraft such as cruise speed, stall speed, maximum speed, endurance, and maximum link range so that a flight

<sup>9</sup> UAS AOC

<sup>10</sup> Ibid

<sup>11</sup> Ibid

mission can be assigned accordingly to test each aircraft on an equal basis.<sup>12</sup> A diagram of an example mission is shown below in Figure 4.



**Figure 4: Example Mission Layout**

In order to qualify for the prize pool, competitors must successfully avoid traffic conflicts in at least 50% of their encounters, fly a minimum of four missions, and complete all missions without committing any safety violations. The first place winner will be awarded 60% of the prize pool, second place will be awarded 20%, and the remaining 20% will be divided among the remaining prize-qualifying competitors.<sup>13</sup> The winner was decided based on the score received determined by the categories listed below.

## Scoring

### Separation Scoring

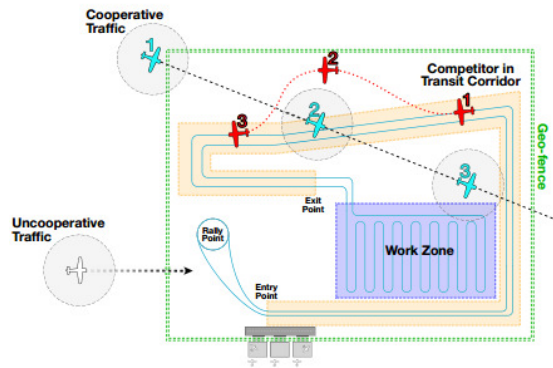
Safe separation must be maintained throughout the duration of each flight. Once a potential conflict is identified, the team must verbally declare to the judges and indicate on the GCS the aircraft involved. If an evasive maneuver is required to maintain safe separation the team must inform the judge that they plan to perform a collision avoidance maneuver and describe their planned action. Then the planned deviation route must be radioed to the range "Control". If the conflict is avoided, full points will be awarded. Each successful evasive maneuver that prevents the separation requirement from being breached will be awarded 200 points. However, no points will be awarded for unnecessary evasive maneuvers and a violation of the separation requirement will result in zero points for the entire mission. Shown at the top of the next page in Figure 5 is an example of a cooperative aircraft that will have to be avoided during the mission.<sup>14</sup>

---

<sup>12</sup> UAS AOC

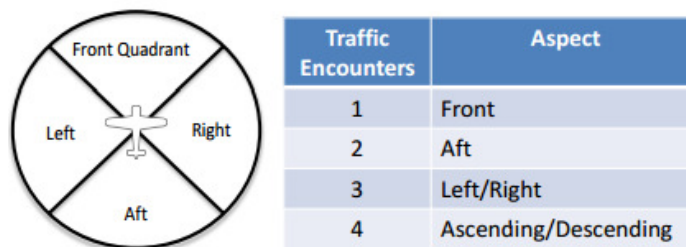
<sup>13</sup> Ibid

<sup>14</sup> Ibid



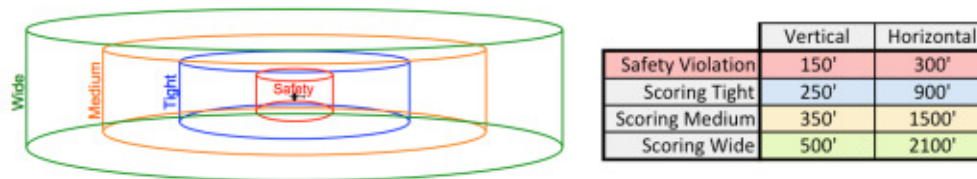
**Figure 5: Example of a Collision Avoidance Maneuver to Avoid Cooperative Aircraft**

The planned traffic encounter geometries are shown below in Figure 6. Air traffic interactions are balanced in the number and types of interactions in an attempt to maintain a fair standard for each competing team.



**Figure 6: Air Traffic Encounter Geometries**

Shown below in Figure 7 are the various separation distances. One of the separation distances will be assigned per mission and that separation must be maintained for the duration of the flight. If at any time during the mission the aircraft commits a safety violation, the team will not be eligible to receive any money from the prize pool.<sup>15</sup>



**Figure 7: Challenge Separation Distances**

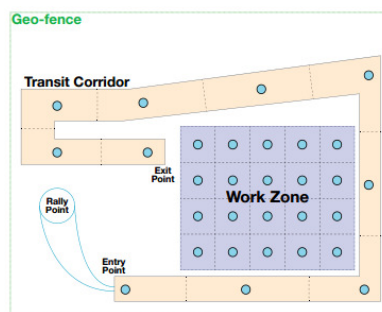
Official separation scoring and separation violations will be determined from the recorded GPS data from the Tracker Module and the broadcasted ADS-B data. Safety violations will be determined in real-time by software that was monitoring the trajectories

<sup>15</sup> UAS AOC

of the aircrafts and is subject to verification in post-flight analysis. Each team will fly 4-6, 20-30 minute, missions during the Challenge week. This gives a maximum of 600-800 points in separation points that may be awarded per mission.<sup>16</sup>

### **Mission Scoring**

Points are awarded for executing the assigned missions and will be awarded incrementally. 100 points are available during each pass through the transit corridor. (entering and exiting) and 200 points are available in the Work Zone. Therefore, a maximum of 400 points will be awarded through Mission Scoring per mission. The points are divided equally among each of the waypoints in the respective zones.<sup>17</sup> An example of the waypoints set in each zone is shown below in Figure 8.



**Figure 8: Waypoints set for Mission Scoring**

### **Airmanship**

Airmanship is described as the ability to operate an aircraft with competence and precision both on ground and in the air, exercising sound judgment that resulted in good operational safety and efficiency. Up to 50 points can be awarded for following these parameters, but these points will only be used in the event of a tie.<sup>18</sup>

### **Uncooperative Traffic Detection and Tracking (Optional)**

Bonus points will be awarded for determining the location of uncooperative aircraft during the mission. Five points are awarded for each accurate position estimate of an uncooperative aircraft per 10 seconds. However, only 1 point is awarded for accurate position estimates that occur more than 20 seconds after the uncooperative aircraft is at the estimated location. Uncooperative aircrafts will be in the vicinity of the competing aircraft up to 50% of the time. This gives a maximum of 450 bonus points that can be awarded for Uncooperative Traffic Detection per mission.<sup>19</sup>

---

<sup>16</sup> UAS AOC

<sup>17</sup> Ibid

<sup>18</sup> Ibid

<sup>19</sup> Ibid

## GPS Integrity Scoring (Optional)

Participating competitors will at least once, but no more than twice, experience a GPS loss initiated by the Jacker Module integrated in the aircraft. The score for successfully detecting inaccurate GPS data is calculated by subtracting the number of seconds that it took for the GPS loss to be detected from 220. The 3 mitigation strategies permitted are: loiter in place (600 points), return to rally (800 points), or completing the mission (1,000 points). This amasses to a total of 1,200 possible points per GPS loss instance and 2,400 bonus points for GPS Integrity Scoring per mission.<sup>20</sup>

## Scoring Summary

The table shown below in Figure 9 summarizes the range and maximum points that can be awarded for each category per mission as well as a brief description of how these points are accumulated.

Scoring Category		How Points Accumulate	Maximum Points Available Per Category
Separation		200 pts per interaction; 3-4 intruder interactions per mission; 4-6 missions per competitor	2400 – 4800 pts
Mission		100 pts available in each corridor; 200 pts available in work zone; 4-6 missions per competitor	1600 – 2400 pts
Optional	Uncooperative Detection & Tracking	5 pts per detection; 6 detections per min.; 20-30 min. per mission; 50% duty	1200 – 2700 pts
	GPS Integrity	2 GPS Loss & 2 GPS Interference per competitor (assuming interference detection at 20 sec for 200 pts)	Loiter in place: 1200 pts
			Rally Pt Return: 1600 pts Mission Complete: 2400 pts
Airmanship		50 pts per interaction, 3-4 intruder interactions per mission; 4-6 missions per competitor	600 – 1200 pts

Figure 9: Scoring Summary

## Registration

A notice of interest was filed for a team representing the FAMU-FSU College of Engineering; however, the team was never actually registered. In order to register the team must have filled out the online registration packet which included an application page, team member page, a survey, and a team agreement that must have been filled out, signed, and mailed to Development Projects Incorporated (DPI), as well as pay the registration fee. Since the registration fee was never able to be funded and a team was never actually formed, the competition could not be entered.

<sup>20</sup> UAS AOC



## **New Goals**

After it was apparent that the competition could not be entered and a set budget was established, the goal of the project shifted from forming a team and competing in the competition, to setting up the electronics to be able to satisfy the general requirements of the competition. These requirements are provided again below.

1. Single aircraft operated from a ground control station (GCS)
2. Multiple 30 minute missions per day
3. Operate within defined airspace boundaries
4. Execute assigned mission
5. Execute missions on schedule
6. Ground demonstration of flight software through simulation
7. Self separation from other air traffic
8. ADS-B receiver integrated onboard
9. ADS-B traffic displayed graphically in GCS
10. Tracker module integrated onboard aircraft

Therefore, the new goal of the project was to create an inexpensive ADS-B receiver that can be inserted into an RC aircraft that will be able transmit data to a GCS, setup an autopilot system that can be controlled from a GCS which will be able to execute an assigned mission on schedule in a defined airspace boundary, be capable of performing autonomous collision avoidance maneuvers to maintain a safe separation distance from other aircrafts transmitting ADS-B data, and be able to demonstrate these capabilities through a HILSim.

## Systems Overview

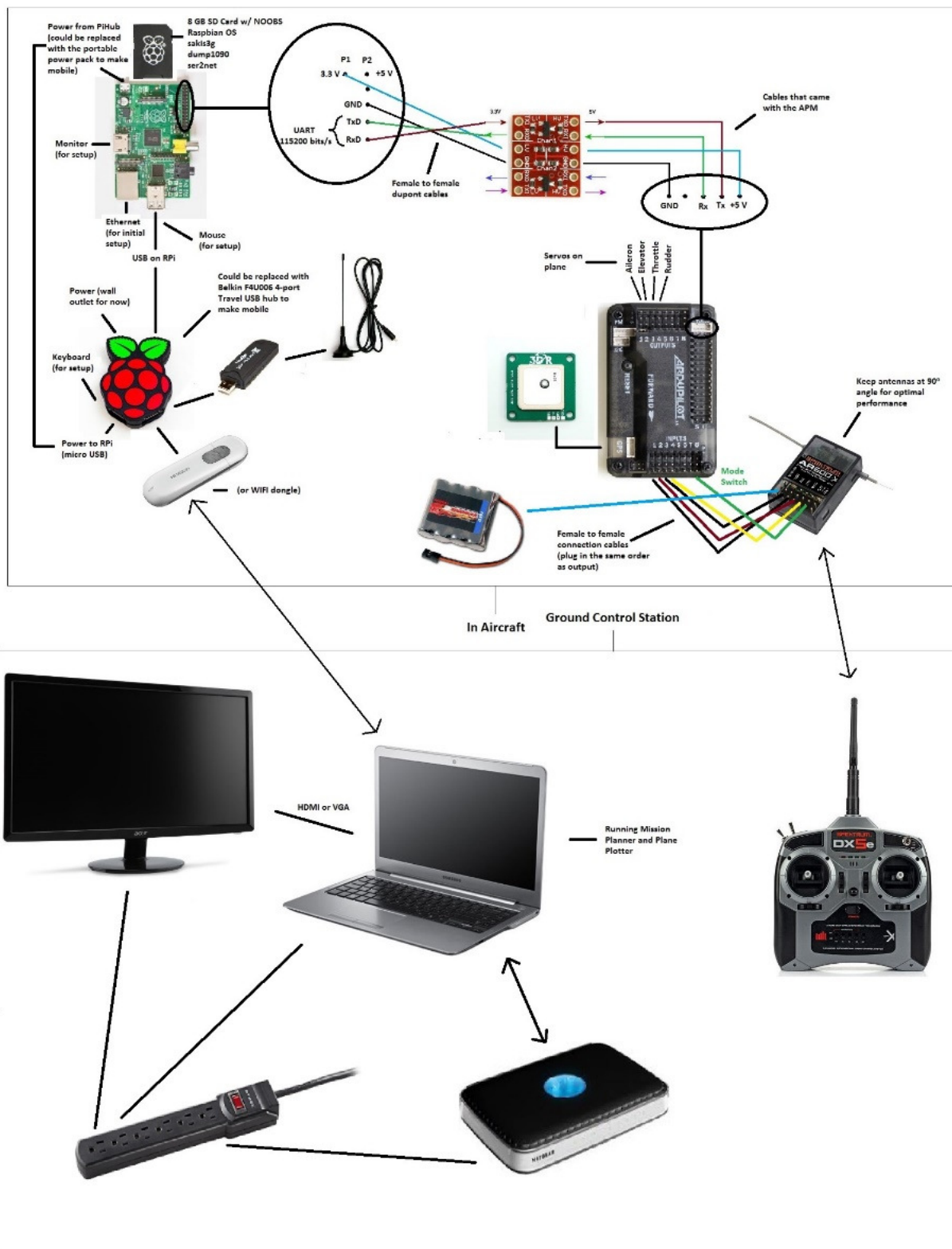
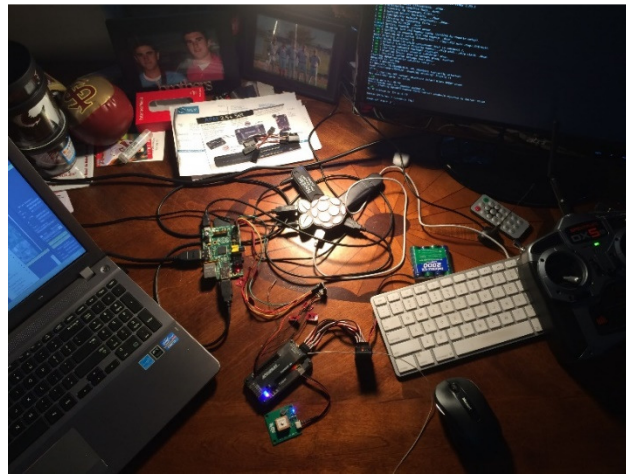


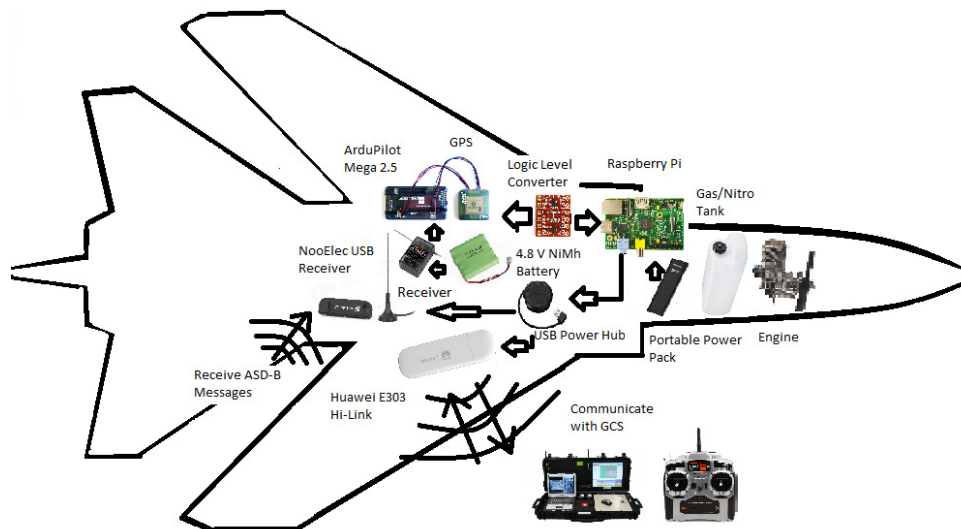
Figure 10: Top Level Design Diagram for Thesis

The top level design diagram shown in Figure 10 on the previous page is the setup that was used for the project. The Raspberry Pi (RPi) is being powered by a PiHub which is plugged into a wall outlet. It is evident that a few changes would need to be made in order to make the system portable and able to be inserted inside an RC aircraft. These changes will be further discussed in Figure 12 and the forthcoming description.



**Figure 11: Photo of Components Used**

The image above is a picture of the components that were used in the completion of the project. A WiFi dongle is being used in place of the 3G dongle because the 3G dongle was accidentally damaged in transit. The WiFi dongle would only provide a maximum range of about 200 feet,<sup>21</sup> so the 3G dongle would need to be replaced if this system was going to be inserted inside an RC aircraft.



**Figure 12: Top Level Design Diagram of RC Aircraft for Competition**

<sup>21</sup> "Belkin Wireless G USB Network Adapter User Manual." Belkin, 2004. Web.

Figure 12 on the previous page is an example of a top level design diagram that would be used for the competition. In this diagram everything is completely mobile. The PiHub has been replaced with a USB power hub that does not require an external power supply. Also, the RPi is now powered by a portable power pack. This power pack would be responsible for powering the RPi, the USB power hub, and the two dongles that are connected to the power hub.

## **Mechanical Components**

### **Gas/Nitro RC Plane**

Due to the competition requirement that multiple 20-30 minutes missions must be executed in a single day, the aircraft chosen would likely be powered by nitro or gas. It would be very difficult to have a battery powered airplane (weighted down by the electronics onboard) achieve this required flight time multiple times a day. In last year's senior design project a flight time of about 40 minutes was achieved with a 16 ounce nitro tank. An RC plane similar to the one used in last year's senior design project would be an ideal purchase due to the flight time, ease of flying, and the large fuselage for storing all of the electrical components. A photo of the RC plane used last year is shown below in Figure 13. However, this particular model is not being sold anymore at Hobby Town (the store where the RC plane was purchased 2 years earlier), but they do carry newer models that function very similarly.



**Figure 13: Senior Telemaster (Used in last year's senior design project)**

### **Storing the Electrical Components**

The position of the autopilot is vital in achieving the desired functionality of the aircraft. Optimally the RC aircraft would contain a docking station that would store the autopilot so that it would be level and facing straight ahead in the aircraft. It should also encapsulate the RPi and the other components to prevent them from being damaged by moving around in the fuselage during flight. Furthermore, a key lesson learned from last year's project was that reaching all of the components in the plane is essential. Therefore, the docking station should be designed to be easily removed so that all of the components can be accessed as easily as possible. Finally, the antenna should be located

in a position that allows for optimal signal receptance, while minimally affecting the flight of the aircraft.

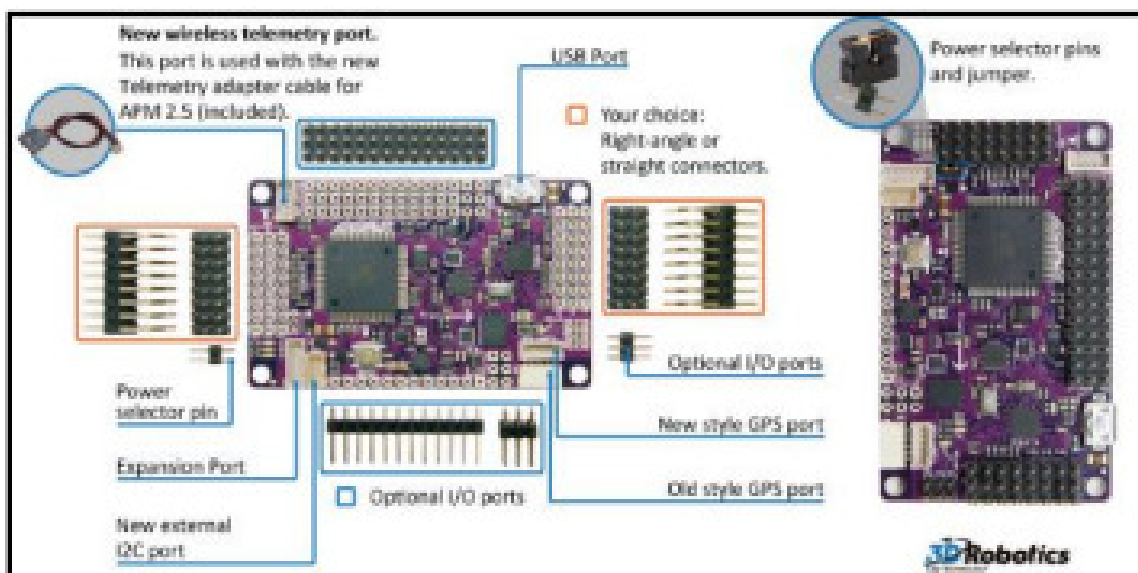
## Electrical Components

This section will cover the hardware and software used in the project, as well as why each was chosen. The following section will describe how this hardware and software was setup, connected, and used in order to achieve the desired functionality.

## Hardware

### Autopilot System

The autopilot system chosen for the project was the AutoPilot Mega 2.5 (APM 2.5). This autopilot system is plug and play, completely open source (Arduino), and is capable of achieving all of the competition requirements for autonomous flight (except for the collision avoidance maneuvers). It allows the user to set up a Geo-Fence (flight boundary), it features point and click waypoint navigation, and is capable of maneuvering through a work area. It comes equipped with various safety modes such as a loiter mode and a return to launch (RTL) mode. Also, once programmed, the autopilot system will continue flying the pre-designated mission even during a loss of GPS signal. Another reason for choosing this autopilot system is the familiarity with it because this was the autopilot system used in last year's senior design project. The familiarity with the setup, how it works, and the firmware used for a fixed wing aircraft helped immensely in the progression of this year's project. The nominal voltage requirement for the autopilot system is 5.37 V +/- 0.5 V and it has a current draw in the 2000 mA range.<sup>22</sup> The ArduPilot Mega 2.5 is shown below in Figure 14.



**Figure 14: ArduPilot Mega 2.5**

## GPS

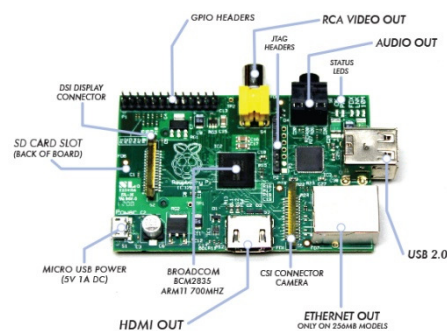
The GPS that was used in this project is the 3DR GPS uBlock LEA-6 which was ordered with the APM 2.5 and came pre-configured for APM use. This GPS boasts a 2.5 m accuracy, has a low noise 3.3 V regulator, and has a 3 V lithium backup battery.<sup>23</sup> The GPS is shown at the bottom of the next page in Figure 15.



**Figure 15: 3DR GPS uBlock LEA-6**

## Raspberry Pi and 8 GB SD Card

The RPi is a single chip computer that was used to create an inexpensive ADS-B receiver that can relay the ADS-B data to the GCS and facilitate the communication of telemetry data to and from the APM 2.5 and the GCS. The SD Card contains the operating system (OS) that the RPi runs on and stores all of the software that was downloaded in order make the RPi function as desired. The Raspberry Pi requires 5 V and draws about 700 mA.<sup>24</sup> A picture of the RPi is shown below in Figure 16 and a picture of the 8 GB SD card is shown below in Figure 17.



**Figure 16: Raspberry Pi**



**Figure 17: 8 GB SD Card**

<sup>22</sup> "APM Plane." *ArduPlane*. DIY Drones, n.d. Web.

<sup>23</sup> Ibid

<sup>24</sup> "Latest Blog Posts." *Raspberry Pi*. Raspberry Pi Foundation, n.d. Web.

### TTL Level Shifter

A logic level shifter was implemented between the RPi and the APM. It steps down the 5V signals coming out of the telemetry port of the APM to the RPi and steps up the 3.3 V signals coming out of the UART port of the RPi to the APM. This device was vital in connecting the RPi and the APM. A picture of the TTL Level shifter is shown at the bottom of the next page in Figure 18.



Figure 18: TTL Level Shifter

### NooElec SDR USB Dongle

The ADS-B receiver will be comprised of a NooElec SDR (that contains a RTL-2832U IC), which is a modified DVB-T USB dongle, combined with an antenna and tuner to provide a frequency capability of approximately 25 MHz – 1750 MHz,<sup>25</sup> which encompasses the 978 MHz UAT broadcasts. A picture of the dongle, antenna, and tuner is shown below in Figure 19.



Figure 19: NooElec SDR, Antenna, and Remote

### Huawei E303 3G USB Dongle

The Huawei E303 dongle was used to connect the GCS and the RPi through 3G. The dongle required a SIM card, which was purchased from T-Mobile. A 1GB/month plan would have been sufficient since the ADS-B data and the telemetry data being transmitted was rather small. Especially, since the 3G connection was not continuously established for exceedingly long periods of time. However, a 3GB/month plan was purchased to be cautious and because the 1GB/month plan was \$20, while the

<sup>25</sup> "NooElec NESDR Mini SDR & DVB-T USB Stick (R820T) W/ Antenna and Remote Control." *NooElec*. NooElec Inc., 2014. Web.



2GB/month plan was only \$30.<sup>26</sup> A picture of the Huawei E303 dongle is shown below in Figure 20.



**Figure 20: Huawei E303 3G USB Dongle**

### **Belkin Wireless G USB Network Adapter**

This USB network adapter was used to connect the GCS to the RPi through WiFi. The connectivity range is only about 200 feet,<sup>27</sup> so the 3G adapter would have to be used in the RC aircraft. However, this dongle played an essential role in the setup of the electrical components. It was extremely helpfully to be able to have the RPi connected wirelessly to the internet, but without using up any data on the 3G plan. It also became crucial to the continuation of the project after the 3G dongle was accidentally broken. A picture of the WiFi dongle is shown below in Figure 21.



**Figure 21: Belkin Wireless G USB Network Adapter**

### **PiHub**

This USB power hub, specifically designed for the RPi, provides a 5.2 V 3000 mA output to up to 4 USB devices.<sup>28</sup> It has a port that is specifically designed to power the RPi. This power hub was used to power the RPi, both of the dongles that were connected to the RPi, and a keyboard for the initial setup. However, this USB power hub must be plugged into a wall outlet and, therefore, in order for the system to be completely mobile it would be ideal to purchase a USB power hub that does not require an external power supply. A picture of the PiHub is shown below in Figure 22 and a picture of a possible replacement is shown below in Figure 23.

---

<sup>26</sup> "Simple Choice Plan." *Mobile Broadband Unlimited Data Plans W/ No Annual Contract*. T-Mobile, 2014. Web.

<sup>27</sup> Belkin

<sup>28</sup> "PIHUB - USB Hub for Raspberry Pi with US Power Adapter." *Adafruit Industries Blog RSS*. Adafruit, n.d. Web.





**Figure 22: PiHub**



**Figure 23: Belkin F4U006  
4-Port Travel USB Hub**

### **DX5e DSMX 5-Channel Transmitter/Receiver**

The transmitter and receiver would be used as a safety precaution in case there is a malfunction in the autopilot system, used to switch modes on the autopilot, and used in the initial setup of the autopilot in the plane. In any given situation the aircraft must be able to switch over from being autonomously controlled by the autopilot to being manually controlled by the transmitter. Also, the transmitter was be used in the radio calibration of the autopilot system to set the minimum and maximum radio values of the throttle and servos used to fly the plane. It may be desirable to purchase a transmitter that provides more switch options in its fifth channel in prder to allow for more flight modes to be assigned by the autopilot system. A picture of the transmitter and receiver used is shown at the top of the next page in Figure 24.



**Figure 24: DX5e DSMX 5-Channel Transmitter/Receiver**

### **Ground Control Station**

The GCS used was my personal laptop coupled with a separate monitor so that both APM Mission Planner and PlanePlotter software cause be accessed and displayed simultaneously. The monitor had an HDMI port which made it able to connect to the RPi and helped in the setup and download of software onto the RPi.

## **Powering the System**

### **NiMH Battery Pack**

A 4.8 V 2000 mAh NiMH battery pack will be used to power the APM 2.5. The same battery was used in last year's senior design and demonstrated a battery life of roughly 1 hour when connected to the autopilot system. A picture of the NiMH battery is shown below in Figure 25.



**Figure 25: 4.8 V 2000 mAh NiMH Battery Pack**

### **Portable Power Pack**

A recommended portable power pack could be used to power the RPi and make the system portable. This power pack will maintain the constant 5 V needed by the RPi to insure that no damage is done to the RPi due to over or under charging. The portable power pack is shown at the top of the next page in Figure 26.



**Figure 26: Portable Power Pack for the Raspberry Pi**

## **Software**

### **Mission Planner**

The Mission Planner (MP) software will be used to communicate with the APM 2.5. Along with many things, this software allows us to set up a Geo-Fence, set waypoints, conduct a work area, and set up a rally point. It will also let us alter some of the functions of the aircraft that are taken into consideration when the judges are creating the aircraft's mission such as cruise speed and maximum speed.<sup>29</sup> A screen shot of a mission in MP that includes waypoints, a work area, a rally point, and a Geo-Fence is shown below in Figure 27.

---

<sup>29</sup> APM Plane

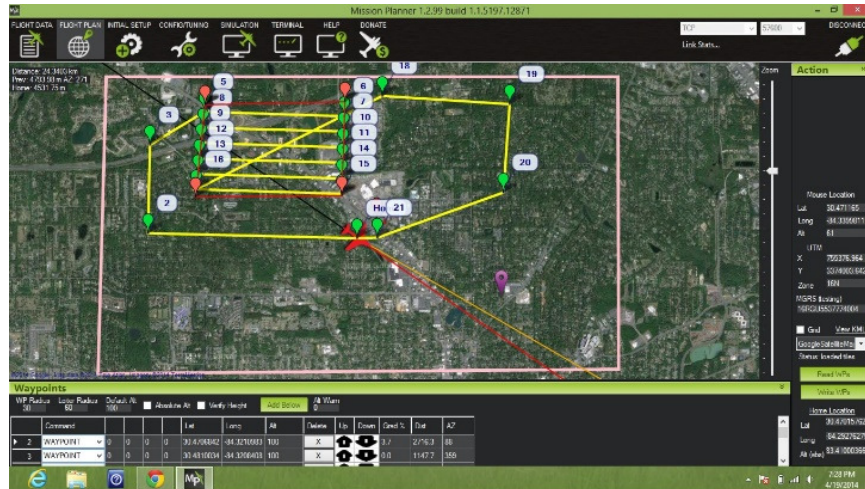


Figure 27: Example Mission in Mission Planner

## ArduPlane Firmware and HILSim Firmware

The firmware for a fixed wing aircraft can be found online and uploaded to the APM 2.5 using MP. This firmware is the software that is responsible for controlling the RC aircraft when it is in any other mode other than manual. A screen shot of how the firmware is uploaded to the APM 2.5 is shown at the top of the next page in Figure 28. Also seen below in Figure 28 is the link to install the HILSimulator firmware for the ArduPlane. This firmware is used to test the functionality of the APM and MP in an HILSim.<sup>30</sup>



Figure28: Downloading Firmware to the ArduPilot Mega using Mission Planner

## X-Plane

The X-Plane software was used in conjunction with MP to perform HIL simulations to determine how the autopilot system will function when implemented in the RC aircraft. It is extremely important to test the equipment and be confident that it is working properly before taking a test flight and risking damaging the aircraft, as well as all of the components in it. A diagram of the setup of a HILSim is shown at the top of the next page in Figure 29.

<sup>30</sup> APM Plane

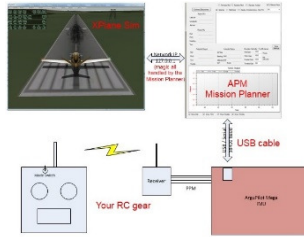


Figure 29: Setup of a HILSim using X-Plane

## Plane Plotter

The Plane Plotter (PP) software was used to receive and display the ADS-B data received from the ADS-B transmitter geographically on the GCS. The software is able to plot an aircraft's coordinate positions, altitude, and heading based upon the ADS-B messages received.<sup>31</sup> A screen shot of the Plane Plotter software chart view is shown below in Figure 30 and the aircraft view is shown at the top of the next page in Figure 31.

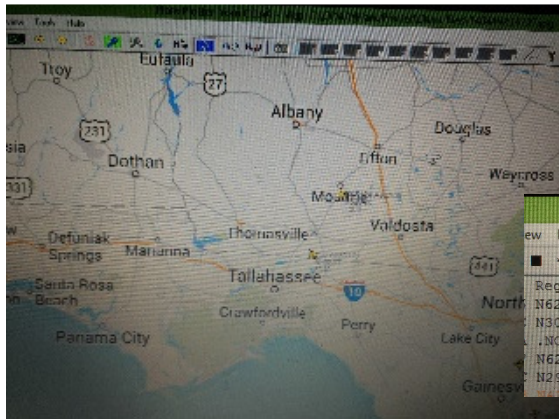


Figure 30: Plane Plotter - Chart View

Reg.	Flight	Lat.	Long.	Alt.	Course	Speed	Type	Route
N625NK	NKS124	31.21397	-83.80844	35950'	337.5°	389.6kts		
N308AT	750	30.79230	-84.04929	33975'	329.1°	377.6kts		
N625LN	SWIFT07	0.00000	0.00000	3925'	0.0°	0.0kts		
N625LN		0.00000	0.00000	600'	0.0°	0.0kts		
N295JB		0.00000	0.00000	37000'	0.0°	0.0kts		

Figure 31: Plane Plotter - Aircraft View

## VPN Server

The VPN server used was provided from LogMeIn Hamachi. It was free and very easy to setup. It provided a point to point protocol connection from the RPi to the GCS via 3G. Having a VPN allows the 3G to be routed directly to your computer, rather than to the website provided by your service provider.

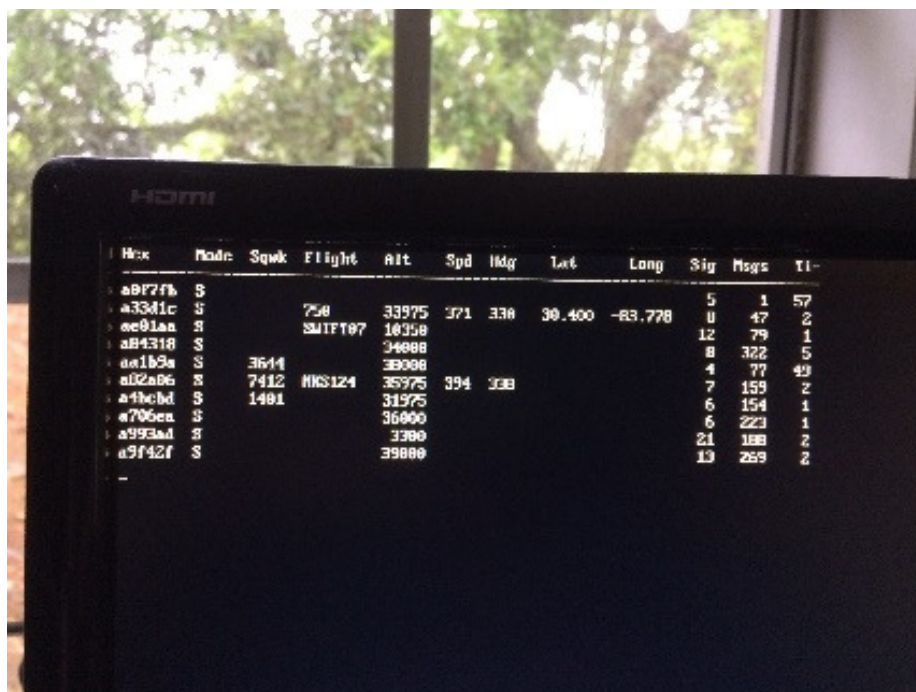
<sup>31</sup> "PlanePlotter from COAA." *PlanePlotter from COAA*. N.p., n.d. Web.

## Raspbian

The OS downloaded from the 8GB SD card w/ NOOBS was the recommended Raspbian OS. This OS is based on Debian, with uses the Linux kernel, specifically tailored to the RPi.<sup>32</sup> It was installed in the first time setup of the RPi.

## Dump1090

The dump1090 program is downloaded onto the RPi and is used to allow the RPi to receive and recode the ADS-B messages. This data can both be displayed graphically on the RPi and sent to PP to be displayed on the GCS.<sup>33</sup> A picture of the dump1090 program running on the RPi is shown below in Figure 32.



Hex	Mode	Squawk	Flight	Alt	Spd	Hdg	Lat	Long	Sig	Msgs	T1
a9f7f6	S								5	1	57
a3341c	S		750	33975	371	330	30.400	-83.778	8	47	2
ae91aa	S		SMIT107	10358					12	79	1
a84318	S			34008					8	322	5
ae1b9a	S	3614		39008					4	77	42
a02a86	S	7412	NKS124	35975	394	330			7	159	2
a1bc6d	S	1401		31975					6	154	1
a706ea	S			36000					6	223	1
a993ad	S			3300					21	100	2
a9f42f	S			39000					13	269	2

Figure 32: Dump1090

## PPP Package, UMTSKeeper, and Sakis3G

The ppp package installs the point to point protocol daemon, which is used to manage the connection between the RPi and the 3G provider. UMTSKeeper is used to automatically reconnect the 3G dongle if the connection drops. Sakis3G is used to make the 3G connection. Together, they allow the RPi to connect to the VPN server on the GCS and maintain this connection.<sup>34</sup>

<sup>32</sup> Raspberry Pi Foundation

<sup>33</sup> Taylor, David. "ADS-B Using Dump1090 for the Raspberry Pi." *ADS-B Dump1090*. Statsignal, 16 Aug. 2013. Web.

<sup>34</sup> Thomson, Andy. "Raspberry Pi as a 3g (Huawei E303) Wireless (Edimax EW-7811Un) Router." *Instructables.com*. Autodesk, Inc., n.d. Web.

## **Ser2net**

The ser2net program is a daemon that allows network connections to serial ports. It allows telnet (or raw) and tcp sessions to be established with the RPi's serial ports. It is used to forward the MAVLink data being sent from the telemetry port of the APM, which is connected to the serial port (UART) on the RPi, to MP on the GCS through a TCP connection.<sup>35</sup>

## **Miscellaneous Components**

The project required some miscellaneous components such as obviously a laptop to act as the GCS, a NiMH battery charger, a 9 V battery for the transmitter, a keyboard for the RPi setup, various cables to connect all of the components, a multimeter, etc. If this project is continued and the electrical components are made portable and inserted in an RC plane, there will undoubtedly be many miscellaneous components that come with the RC aircraft such as a starter kit, spare parts, solder, etc.

## **Establishing a 3G Connection**

Once the ppp package, UMTSKeeper, and Sakis3G are successfully installed on the RPi a 3G connection was established using Sakis3G. The Sakis3G interactive prompt asked a series of question to setup the 3G connection. At first, the 3G connection was set to connect to T-Mobile on their website, [epc.tmobile.com](http://epc.tmobile.com), but once the Hamachi VPN server was setup the 3G connection was routed to the VPN server using the following web address, [std.hamachi.logmein.com](http://std.hamachi.logmein.com), and entering the network ID created in the VPN server. The next step was to automatically establish a 3G connection when the OS booted. This was done by opening the following file, `/etc/rc.local` and entering the code to establish a 3G connection on OS boot located in the appendix.<sup>36</sup> This connection took a few seconds to establish, but was successful in connecting to the VPN server after a short delay.

## **Setting up the ADS-B Receiver**

### **Dump1090**

The first step was to install the utils required to compile the RTL-2832U IC (which is located in the Huawei E303 USB dongle) driver. Then, the RTL-2832U driver source was installed and compiled. Finally, the dump1090 application source code was downloaded which enabled the RPi to receive and decode ADS-B messages and function as an ADS-B receiver. At this point, the dump1090 program was tested and was able to successfully display the ADS-B data

---

<sup>35</sup> Minyard, Corey. "Serial Port to Network Proxy." *SourceForge*. Dice Holdings, Inc., 29 July 2013. Web.

<sup>36</sup> Thomas, Instructables



graphically on a monitor that was connected to the RPi. A picture of this program running on the RPi was shown above in Figure 32. Then a script was created that enabled the dump1090 to run automatically when the OS booted up. In order to achieve this a script labeled `/etc/init.d/dump1090.sh` was created and the code to run the dump1090 program automatically on OS boot located in the appendix was entered and saved in the file.<sup>37</sup> The next step was to setup PP to be able to receive the messages from the RPi running the dump1090 program.

## Plane Plotter

Once PP was downloaded it was time to configure PP so that it could retrieve and display the ADS-B messages from the RPi that was running the dump1090 program. This was done by altering the I/O settings of PP to look for a RTL dongle running the dump1090 program as the Mode-S/ADS-B receiver. Then the IP address and port of the RTL dongle running the dump1090 program was entered into PP. The IP address used was the IP address of the RPi and the port was 3005 as specified from (bibliography). Figure 33 below shows how the I/O settings and the IP address were setup in PP.<sup>38</sup> The pictures above in Figure 30 and 31 show PP displaying the ADS-B data received from the RPi.

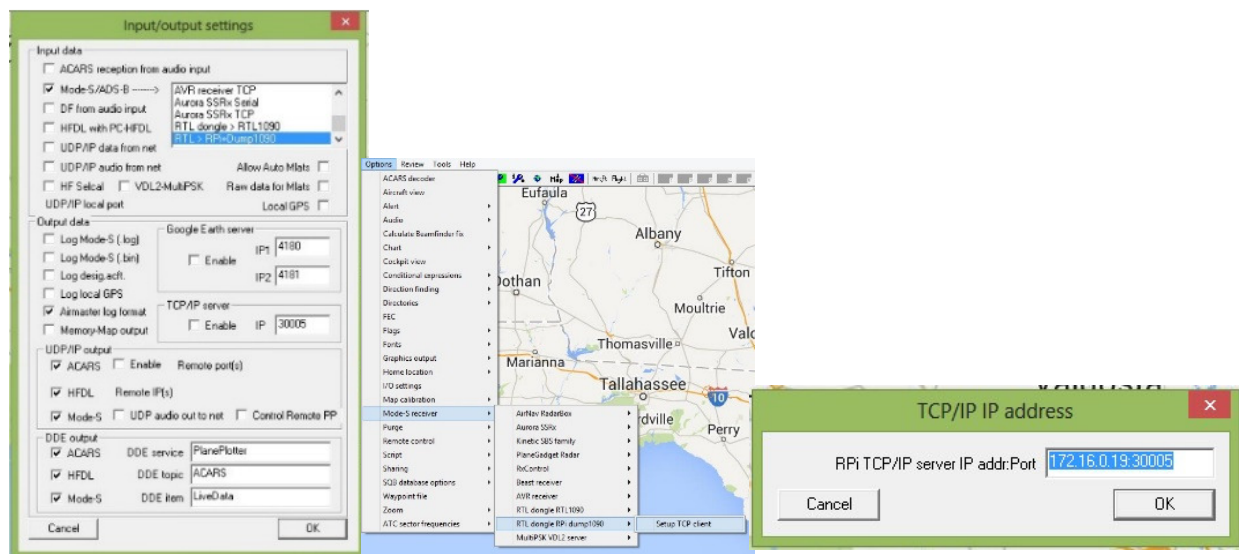


Figure 33: Plane Plotter I/O Settings and Mode-S Receiver IP Address

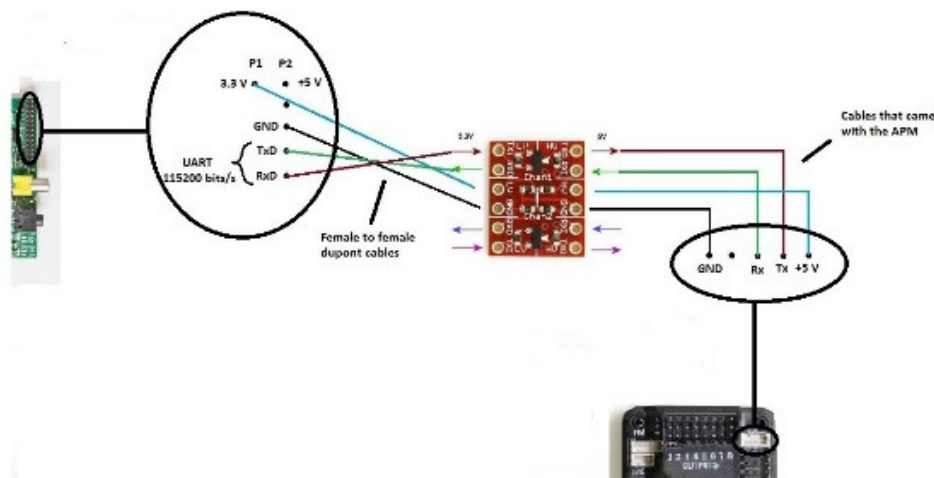
<sup>37</sup> ADS-B Dump1090

<sup>38</sup> Ibid

## Connecting the RPi, APM, and MP

### Hardware Configuration

This connection was made by inserting a TTL Level Shifter between the RPi and the APM to safely step up the 3.3 V signals from UART port of the RPi to the 5 V signals required by the telemetry port of the APM and safely step down the 5 V signals from the telemetry port of the APM to the 3.3 V signals required by the UART port of the RPi. The APM was connected to the TTL Level Shifter using the telemetry cables that came with the APM. The 5 V output of the telemetry port was connected to the high voltage pin of the TTL Level Shifter, ground was connected to ground, Rx was connected to Rx, and Tx was connected to Tx. The RPi was connected to the TTL Level Shifter using female to female dupont cables. The 3.3 V output of the UART port of the RPi was connected to the low voltage pin of the TTL Level shifter, ground was connected to ground, Rx was connected to Tx, and Tx was connected to Rx. It is very important to connect the Rx to Tx and Tx to Rx on one side of the connection. This way the Tx port of the RPi is connected to the Rx port of the APM and vice versa.<sup>39</sup> A diagram of the correct connection is shown below in Figure 34.



**Figure 34: Connection Diagram of the Raspberry Pi and the ArduPilot Mega**

### Ser2net

After ser2net was installed on the APM the manner in which MP connected to the RPi was declared in the ser2net configuration file. The configuration file was extremely informative and gave a thorough description of the format and descriptions of each part of the code contained in it. In this file the connection port was established, the baud rate was set, and specifications of the

<sup>39</sup> Liang, Oscar. "Raspberry Pi and Arduino Connected Over Serial GPIO - OscarLiang.net." *OscarLiang.net*. N.p., 21 May 2013. Web.



connection were declared. The exact code used to construct the ser2net configuration file can be found in the appendix under ser2net configuration file. Once this is done the serial port login was disabled on the RPi and the RPi was then capable of relaying the MAVLink messages to and from MP and the APM.<sup>40</sup>

## **Mission Planner**

First, the ArduPlane firmware was downloaded to MP. A picture of how to accomplish this was shown above in Figure 28. Then, the final step in configuring the RPi to act as a telemetry kit for the APM is to configure MP to connect to the RPi. This was done by selecting a TCP connection in MP, the baud rate was set to 57600 (which was the baud rate declared in the ser2net configuration file), the IP address was set to the IP address of the RPi, and the port used is the port that was specified in the ser2net configuration file.<sup>41</sup> At this point MP was able to connect to the RPi and send and receive telemetry data to and from the APM.

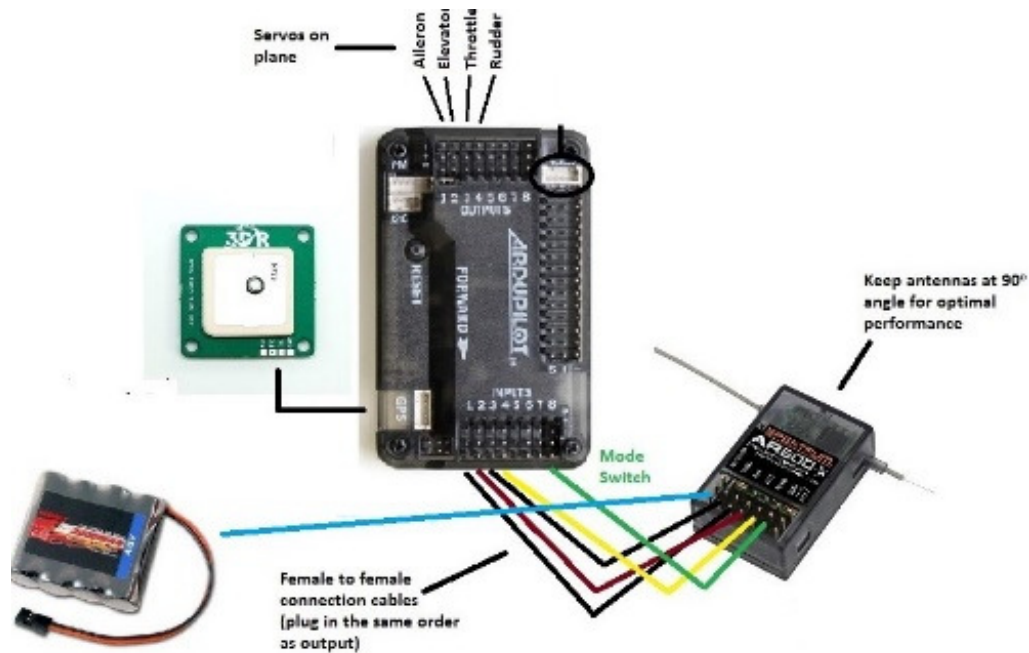
## **Setting up the Receiver/Transmitter**

First, the transmitter and receiver were setup. This was done by inserting the binding cable into the receiver, plugging in the 4.8 V battery pack with the correct polarity to match the connection to the APM (ground down), turning on the transmitter while holding the training switch, and waiting for a connection to be established. Once the communication between the receiver and transmitter was correctly established the receiver was connected to the APM using female to female connection cables for Futaba servos. Each port of the receiver was connected to match the same port on the input of the RPi to correctly control the servos. The connection diagram is shown at the top of the next page in Figure 35.

---

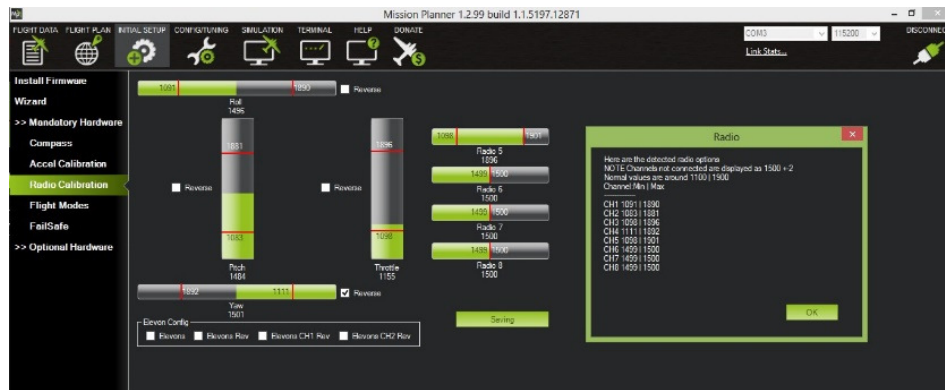
<sup>40</sup> Anderson, Chris, Simon Howroyd, Paul Mijoiu, and Tommy Larsen. "Connecting APM 2.0 and Raspberry Pi." *DIY Drones*. Ning, 1 Mar. 2013.

<sup>41</sup> Anderson, Simon, Paul, Tommy, *DIY Drones*



**Figure 35: ArduPilot Mega and Receiver Connection**

Once this connection is correctly established the radio configuration on MP was set to determine the maximum and minimum values of the servos. This is done so that the APM can accurately control the servos on the RC aircraft and fly the aircraft as desired. A picture of the radio configuration in MP is shown at the top of the next page in Figure 36.



**Figure 36: Radio Calibration in Mission Planner**

## Setting up Mission Planner

After MP was downloaded, the firmware was installed, a connection was able to be established, and the receiver/transmitter were correctly connected the first time setup wizard in MP was ran to correctly setup all of the parameters of the AM on MP. Once this was done, it was time to setup a mission on MP to fly the mission specified by the NASA UAS Challenge. A sample

mission in MP was shown above in Figure 27. In the “Flight Plan” tab of MP waypoints were added by simply pointing and clicking on the screen. The functionality of these waypoints may be changed to a variety of functions such as takeoff, land, loiter, RTL, etc<sup>42</sup>.

### **Setting up a Work Area**

A work area was created by setting up a polygon and creating a survey(grid) of waypoints to maneuver through the work area. The altitude of these waypoints, distance from each other, orientation, etc. can all be altered under the advanced options of the survey(grid)<sup>43</sup>.

### **Setting up a Geo-Fence**

A Geo-Fence was created by first setting up a return location that was easily accessible incase the aircraft was to breach the Geo-Fence. Then another polygon was setup and the create a Geo-Fence option was selected. A maximum and minimum altitude could also be set for the Geo-Fence, however, it is important to note that the Geo-Fence must not be enabled during takeoff and landing if a maximum and minimum value are set because these commands would likely breach the Geo-Fence. MP has an option to automatically enable the Geo-Fence after takeoff and disable the Geo-Fence before landing<sup>44</sup>.

### **Setting up a Rally Point**

The purpose of a rally point is to provide a closer target where the aircraft could perform a return to lauch (RTL) command without actually returning home. This could be desirable if there is a larger flight area and it would be desirable to return to a certain point, but not necessarily return all the way home. Multiple rally points can be added in MP (up to 10) and if the RTL mode is engaged, the aircraft will return to the closest rally point rather than returning all the way home. These can be added by simply selecting the create a rally point option in MP and specifying an altitude<sup>45</sup>.

## **Hardware in the Loop Simulations**

HiLSim was performed using the X-Plane software and MP. First, the HILSimulator firmware was downloaded onto the APM. Then the APM was connected to the GCS via USB. X-Plane was configured to communicate with MP by specifying the loopback IP address (127.0.0.1) and port to be 49005 for the data receiver in the network connection settings. Next the boxes for X-Plane were highlighted in the “Simulation” tab in MP and a connection was made from MP to X-Plane. Finally, the radios were reconfigured and the APM was ready to test via a HiLSim using

---

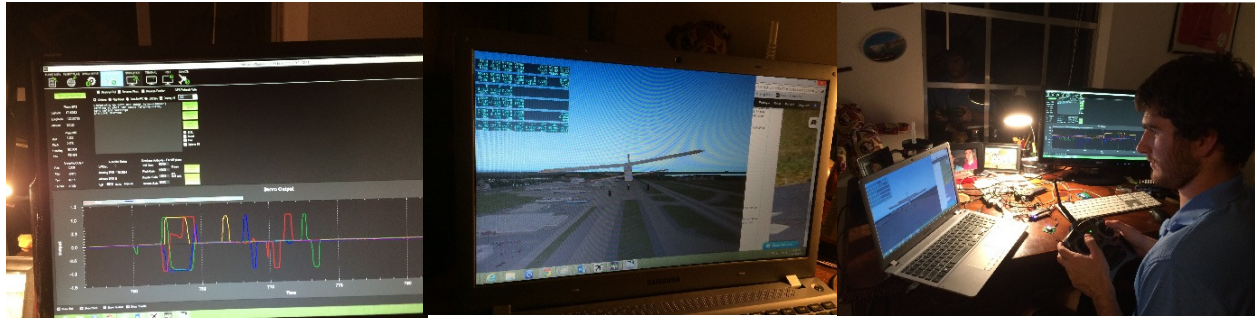
<sup>42</sup> APM Plane

<sup>43</sup> Ibid

<sup>44</sup> Ibid

<sup>45</sup> APM Plane

the X-Plane software<sup>46</sup>. A picture of the “Simulation” tab in MP showing the output of the servos and an example of a HILSim using the X-Plane software is shown below in Figure 37.



**Figure 37: Simulation tab of Mission Planner and HILSim using X-Plane**

### **Collision Avoidance Maneuvers**

At this point the aircraft can both fly autonomously using the RPi equipped with ser2net to send and receive its telemetry data to and from MP, and receive and decode ADS-B messages received by the NooElec dongle using the dump1090 program, as well as transfer this data to PP to be displayed on the GCS. An unfortunate mishap occurred where the 3G dongle was damaged, so the system is only working over WiFi, but in the future it would be ideal to have all of this working over 3G so that the connectivity range would be vastly increased. Now at this point, since the CGS can be used to both fly the aircraft autonomously and display the ADS-B data received, an aircraft transmitting ADS-B messages that is breaching a set safe separation distance around the aircraft should be able to be detected and an alert should be able to be sent to the aircraft to perform a collision avoidance maneuver.

The first step would be to set a safe separation distance to maintain. This should be able to be altered, so that a variety of safe separation distances could be set as required by the NASA UAS Centennial Challenge rules.

Next, would be to run a python script to extract the lat., long., heading, and speed of the ADS-B signals received in PP.

Now, compare the data received from the ADS-B messages in PP and compare those to the aircrafts current GPS location. If the safe separation distance is breached a trajectory to re-establish the safe separation distance required should be detected and the aircraft should perform a collision avoidance maneuver. These trajectories could be to move left or right, up or down, and speed up or slow down.

---

<sup>46</sup> Ibid

The collision avoidance maneuver could be a python script run on MP to fly in the selected trajectory to provide a re-establishment of the safe separation distance. Once the safe separation distance is re-established the mission should automatically resume as if nothing happened.

An oversimplified pseudo-code of this is shown in the appendix under pseudo-code for an autonomous collision avoidance maneuver. After this script is run the autopilot system should continue flying the mission that it was originally on. This script estimates the distance between the RC aircraft and the surrounding aircraft by using the Pythagorean equation and determines the heading of the surrounding aircraft using the Haversine formula<sup>47</sup>. Then it performs a collision avoidance maneuver based on the heading and altitude of the threatening aircraft.

### **Budget**

Figure 38 on the next page is a chart that displays the items used for the project, the cost of each item individually, and the overall cost of all of the components. It also shows a rough estimate of the cost of the project if the RC aircraft and the spare parts required for it were purchased, as well as an estimate of the complete total cost if the registration fee would have been covered and the competition was entered.

---

<sup>47</sup> Veness, Chris. "Movable Type Scripts." *Calculate Distance and Bearing between Two Latitude/Longitude Points Using Haversine Formula in JavaScript*. N.p., 2014. Web.

<b>Components</b>	<b>Price</b>
ArduPilot Mega w/ GPS	\$201.89
NooElec USB Receiver	\$21.95
PiHub	\$39.98
8 GB SD Card	\$20.89
Huawei E303 3G Dongle	\$34.96
Monitor	\$82.48
HDMI Cable	\$16.11
NiMH Battery Pack	\$22.56
Multimeter	\$17.19
5 Channel Transmitter/Receiver	\$96.74
Plane Plotter	\$35.87
T-Mobile (3GB/month)x3	\$72.00
Primal Multi-Chemistry Charger	\$37.61
40P Female to Female Dupont Cables	\$40.96
5 Female to Female Connection Cables for Futaba Servo	\$9.20
9 V Battery	\$5.49
<b>Total</b>	<b>\$755.88</b>
Raspberry Pi	\$35
<b>Total w/ RPi</b>	<b>\$790.88</b>
Avistar Elite .46 RTF	\$399.99
AMA Membership, Gas/Nitro, Spare Parts	~\$300
<b>Total w/ RC Aircraft</b>	<b>~\$1490.87</b>
Registration Fee	\$5,000-\$9,000
Travel, lodging, etc.	~\$500
<b>Total to Compete</b>	<b>~\$6,990.87 - \$10,990.87</b>

Figure 38: Budget List

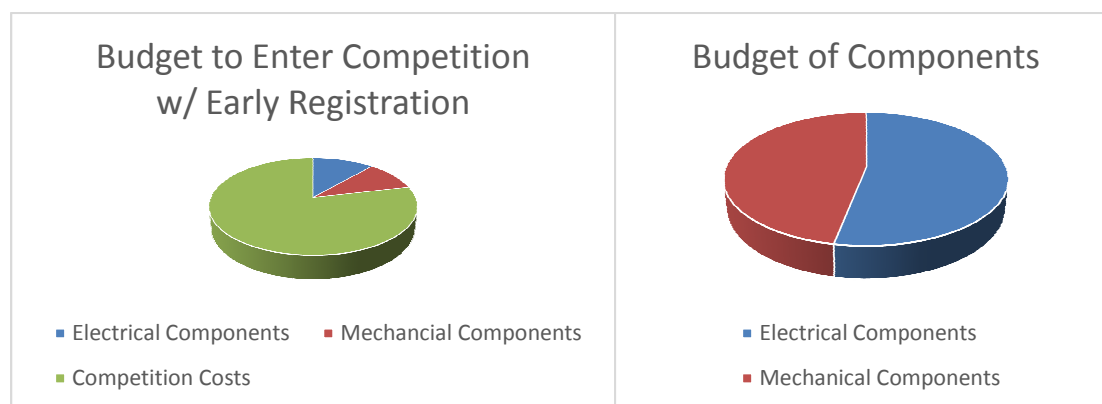


Figure 39: Budget of Competition Pie Charts

## Conclusion

An inexpensive ADS-B receiver can be made using a RPi with the dump1090 program installed and a dongle containing a RTL-2832U IC with an antenna capable of receiving 978 MHz UAT broadcasts. The ADS-B data received can then be transmitted to a GCS via 3G or WiFi and displayed geographically using the PP software. When equipped with ser2net, the RPi can also be used to facilitate communication between the APM and MP running on the GCS. Using the X-Plane software a HILSim can be conducted to verify that the APM and MP are working to control the RC airplane as desired. Since the ADS-B data can be used to plot the surrounding compliant aircrafts and the APM can be controlled from the GCS to allow for autonomous flight, collision avoidance maneuvers could be programmed on run on MP to maintain a safe separation from surrounding aircrafts broadcasting ADS-B signals.

This thesis could be used by future senior design groups to provide them with a step by step process of setting up the electrical components that can be inserted into an RC aircraft allowing the aircraft to be able to complete the general guidelines of the UAS AOC NASA Centennial Challenge or another competition similar in nature. A more detailed setup of the hardware and software can be found at my website, [cdt09d.wix.com/autonomous-aircraft](http://cdt09d.wix.com/autonomous-aircraft).

## Appendix

### Code to Run Dump1090 on OS Boot

```
#!/bin/bash
# Start dump1090 on OS startup
# Modeled after www.satsignal.eu/raspberry-pi/dump1090.html

# Declare program name, path, arguments, and pid-file
prog="dump1090"
prog_path="/home/pi/dump1090"
prog_arg="--interactive --net --net"
pidfile="/var/run/dump1090.pid"

# Declare the start command
start() {
    # Check if program is already running and, if so, exit with error
    if [ -e $pidfile ]; then
        echo "Error! $prog is already running!" 1>&2
        exit 1

    # If not open the path and run the program
    else
        cd $prog_path
        ./$prog $prog_arg 2>&1 >/var/log/$prog &
        echo "$prog is running"
        touch $pidfile
    fi
}

# Declare the stop command
stop() {
    # Check if the program is running and, if so, stop it
    if [ -e $pidfile ]; then
        killall $prog
        rm -f $pidfile
        echo "$prog stopped"

    #If not exit with error
    else
        echo "Error! $prog is not running!" 1>&2
    fi
}
```



```

exit 1
    fi
}

# Check if we are running as root
# Found at http://www.cyberciti.biz/tips/shell-root-user-check-script.html
# New way: Using EUID
if [[ $EUID -ne 0 ]]; then
    echo "This script must be run as root" 1>&2
    exit 1
fi

# Also modeled after http://tldp.org/LDP/Bash-Beginners-Guide/html/sect\_07-03.html
case "$1" in
    # Call the start command
    start)
        start
        exit 0
        ;;

    # Call the stop command
    stop)
        stop
        exit 0
        ;;

    # Upon restarting, first stop then restart the program
    reload|restart|force-reload)
        stop
        start
        exit 0
        ;;

    **)
        echo "Usage: $0 {start|stop|reload}" 1>&2
        exit 1
        ;;
esac
exit 0

```

## Code to Establish a 3G Connection on OS Boot

```
/home/pi/umtskeeper/umtskeeper --sakisoperators "USBINTERFACE = '0' OTHER =  
'USBMODEM' USBMODEM = '12d1:1506' SIM_PIN = 'Enter PIN provided by mobile  
operator' APN = 'CUSTOM_APM' CUSTOM_APN = 'std.hamachi.logmein.com' APN_USER =  
'Enter network ID created' APN_PASS = 'Enter password created here'" --sakisswitches "--sudo -  
-console" --devicename 'Huawei' --log --silent --monthstart 8 --nat 'no' &
```

## Ser2net Configuration File

```
BANNER: banner:\r\nser2net port \p device \d [\s] \r\n\r\n  
3000:raw:600:/dev/ttyAMA0:57600 8DATABITS NONE 1STOPBIT banner
```

## Pseudo-code to Perform an Autonomous Collision Avoidance Maneuver

```
from math import *  
  
# Set safe separation  
safe_sep = xxx  
# Declare and set all variables to 0...  
  
# Get ADS-B data from PP  
planes_lat = [lat1, lat2, ...] # from PP in rad  
planes_long = [long1, long2, ...] # from PP in rad  
planes_alt = [alt1, alt2, ...] # from PP in meters  
x = len(planes_lat) # determine number of lat coordinates  
y = len(planes_long) # determine number of long coordinates  
z = len(planes_alt) # determine number of alt coordinates  
  
# Get current GPS data  
gps_lat = cs.lat # in deg  
gps_long = cs.lng # in deg  
gps_lat, gps_long = map (radians, [gps_lat, gps_long]) # convert deg to rad  
gps_alt = cs.alt # in meters  
  
# Do for each plane detected  
for(i <= x)  
for(i <= y)  
for(i <= z)  
  
# Make sure we have lat, long, and alt  
if (planes_lat[i] == 0 or planes_long[i] == 0 or planes_alt[i] == 0)
```

```

i = i+1
loop

# Use pythagorean theorem to estimate distance
x1 = gps_alt * cos(gps_lat) * sin(gps_long)
y1 = gps_alt * sin(gps_lat)
z1 = gps_alt * cos(gps_lat) * cos(gps_long)
x2 = planes_alt * cos(planes_lat[i]) * sin(planes_long[i])
y2 = planes_alt * sin(planes_lat[i])
z2 = planes_alt * cos(planes_lat[i]) * cos(planes_long[i])

d = sqrt((x2-x1)**2 + (y2-y1)**2 + (z2-z1)**2)

# Determine if safe separation breached
if (d > safe_sep)
    i = i+1
    loop

else
    # Use Haversine formula to determine bearing
    dlat = planes_lat[i] - gps_lat # determin lat distance
    dlong = planes_long[i] - gps_long # determine long distance

    w = sin(dlong) * cos(planes_lat[i])
    v = cos(gps_lat) * sin(planes_lat[i]) - sin(gps_lat) * cos(planes_lat[i]) * cos(dlong)
    bearing = atan2(w,v)
    bearing = map( degrees, bearing) # convert to degrees

# Compare altitudes
alt = planes_alt - gps_alt

# Perform collision avoidance maneuver based on bearing and altitude
# Check altitude
while ( d < safe_sep)
    if (alt > 0 and cs.alt > 75)
        Script.SendRC (2,1200,True) # pitch down
        Script.SendRC (3,1300,True) # slow down
    else if (alt < 0 and cs.alt < 200)
        Script.SendRC (2,1700,True) # pitch up
        Script.SendRC (3,1700,True) # speed up

```

end if

# Check distance

x1 = cs.alt \* cos(cs.lat) \* sin(cs.long)

y1 = cs.alt \* sin(cs.lat)

z1 = cs.alt \* cos(cs.lat) \* cos(cs.long)

x2 = planes\_alt \* cos(planes\_lat[i]) \* sin(planes\_long[i]) # get new values from PP

y2 = planes\_alt \* sin(planes\_lat[i]) # make sure that its the same plane

z2 = planes\_alt \* cos(planes\_lat[i]) \* cos(planes\_long[i])

d = sqrt((x2-x1)\*\*2 + (y2-y1)\*\*2 + (z2-z1)\*\*2)

# Check bearing

if (0 < bearing < 90)

Script.SendRC(3,1300,True) # slow down

Script.SendRC(1,1300,True) # roll left

else if (90 < bearing < 180)

Script.SendRC(3,1300,True) # slow down

Script.SendRC(1,1700,True) # roll right

else if (180 < bearing < 270)

Script.SendRC(3,1700,True) # speed up

Script.SendRC(1,1700,True) # roll right

else if (270 < bearing < 360)

Script.SendRC(3,1700,True) # speed up

Script.SendRC(1,1300,True) # roll left

end if

# Check distance

x1 = cs.alt \* cos(cs.lat) \* sin(cs.long)

y1 = cs.alt \* sin(cs.lat)

z1 = cs.alt \* cos(cs.lat) \* cos(cs.long)

x2 = planes\_alt \* cos(planes\_lat[i]) \* sin(planes\_long[i]) # get new values from PP

y2 = planes\_alt \* sin(planes\_lat[i]) # make sure that its the same plane

z2 = planes\_alt \* cos(planes\_lat[i]) \* cos(planes\_long[i])

d = sqrt((x2-x1)\*\*2 + (y2-y1)\*\*2 + (z2-z1)\*\*2)

i = i + 1

loop

## Sponsorship Letter



Date

Address of  
Company

Dear Mr. / Mrs. \_\_\_\_\_

My name is Cris Timmons and I am a double major in electrical engineering and pure mathematics at the Florida State University. My goal is to compete in the NASA UAS (“Unmanned Aerial Systems”) Centennial Challenge as my honors-in-the-major thesis. This is a two-phase competition with the first phase being held spring 2014 and the second phase being held roughly a year later.

The competition entails building an autopiloted RC aircraft that must perform certain pre-specified tasks. The first phase of the competition includes safe airspace operations such as accurately and reliably flying pre-determined trajectories, being able to interact with Air Traffic Management before, during, and after unmanned aircraft operations, and separation assurance of cooperative aircraft using ADS-B (“Automatic Dependent Surveillance – Broadcast”) signals. The aircraft design is encouraged to include robustness to system failures such as a lost link or unavailable/unreliable GPS and to differentiate real surrounding aircrafts from the “ghost” aircrafts that only exist as ADS-B messages. The second phase will require the competitors to avoid both cooperative and non-cooperative aircrafts.

I plan on completing the project by incorporating an ADS-B receiver onboard the RC aircraft to receive ADS-B messages that contain information on the whereabouts of surrounding aircrafts. I plan on downloading the Flightradar24 Linux ARM sharing software to a Raspberry Pi (a single-board computer) to allow the Raspberry Pi to receive and display the ADS-B messages. Then, the Raspberry Pi will be connected to the telemetry port on an ArduPilot Mega 2.5 autopilot system. The Raspberry Pi will be equipped with a Huawei E303C (Hi-Link) Data card which will allow the serial communication between the autopilot and the Raspberry Pi to be routed via 3G to a VPN server to be accessed on the ground station. Finally, collision avoidance maneuvers must be programmed to the autopilot system to avoid the surrounding air traffic.

I want to compete in this competition not only because I am particularly interested in autonomous flight and airspace operations, but also because of the advancements in flight safety that can arise from this competition. One day commercial and civilian aircraft will be capable of completely autonomous flight while simultaneously avoiding any surrounding potential hazards. My uncle passed away in a plane crash that occurred in a routine test flight for the Golden

Knights parachute team because a civilian aircraft failed to follow the correct procedure before coming in for landing. This collision could have been easily avoided if both planes had been equipped with inexpensive ADS-B Transceivers and if either one could have performed autonomous collision avoidance maneuvers.

The aircraft, electronics, and everything else needed to complete the project will cost roughly \$1,000, but the bulk of the funding will be needed to pay the registration fee of \$5,000 (early registration).

I would be grateful if you helped in sponsoring the FAMU-FSU College of Engineering team to compete in the NASA UAS Centennial Challenge. It is the first year that the event is being held and I would like to do well in the competition so that more FAMU-FSU College of Engineering students may get involved in the future.

The amount a sponsor may contribute is flexible. The different sponsorship packages are listed below.

Bronze:

\$100 – Your name shown as a sponsor in my presentations

Silver:

\$250 – Your name and logo are shown as a sponsor in my presentations

Garnet:

\$500 – Small logo on the aircraft

Gold:

\$750 – Your name and logo are shown as a sponsor in my presentations and a small logo is shown on the aircraft

Platinum:

\$1000 - Your name and logo are shown as a sponsor in my presentations and a large logo is shown on the aircraft

If you are interested in sponsoring me, please contact me at my house (850)727-4026, on my cell (352)455-6562, or by email at [cdt09d@my.fsu.edu](mailto:cdt09d@my.fsu.edu).

Thank you for your time and consideration.

Sincerely,

Signature

Cris Timmons

## **Bibliography**

## Websites

- "Aviation Safety Network Statistics Flight Nature Domestic Scheduled Passenger Accidents." *Aviation Safety Network Statistics Flight Nature Domestic Scheduled Passenger Accidents*. N.p., n.d. Web.
- "Latest News." *UAS Airspace Operations Challenge*. NASA, n.d. Web.
- "Belkin Wireless G USB Network Adapter User Manual." Belkin, 2004. Web.
- "APM Plane." *ArduPlane*. DIY Drones, n.d. Web.
- "Latest Blog Posts." *Raspberry Pi*. Raspberry Pi Foundation, n.d. Web.
- "NooElec NESDR Mini SDR & DVB-T USB Stick (R820T) W/ Antenna and Remote Control." *NooElec*. NooElec Inc., 2014. Web.
- "Simple Choice Plan." *Mobile Broadband Unlimited Data Plans W/ No Annual Contract*. T-Mobile, 2014. Web.
- "PIHUB - USB Hub for Raspberry Pi with US Power Adapter." *Adafruit Industries Blog RSS*. Adafruit, n.d. Web.
- "PlanePlotter from COAA." *PlanePlotter from COAA*. N.p., n.d. Web.
- Taylor, David. "ADS-B Using Dump1090 for the Raspberry Pi." *ADS-B Dump1090*. Statsignal, 16 Aug. 2013. Web.
- Thomson, Andy. "Raspberry Pi as a 3g (Huawei E303) Wireless (Edimax EW-7811Un) Router." *Instructables.com*. Autodesk, Inc., n.d. Web.
- Minyard, Corey. "Serial Port to Network Proxy." *SourceForge*. Dice Holdings, Inc., 29 July 2013. Web.
- Liang, Oscar. "Raspberry Pi and Arduino Connected Over Serial GPIO - OscarLiang.net." *OscarLiang.net*. N.p., 21 May 2013. Web.
- Anderson, Chris, Simon Howroyd, Paul Mijoiu, and Tommy Larsen. "Connecting APM 2.0 and Raspberry Pi." *DIY Drones*. Ning, 1 Mar. 2013.
- Veness, Chris. "Movable Type Scripts." *Calculate Distance and Bearing between Two Latitude/Longitude Points Using Haversine Formula in JavaScript*. N.p., 2014. Web.
- Watkiss, Stewart. "Raspberry Pi Wireless - Belkin 802.11g Usb Adapter Ralink RT2750W." *PenguinTutor*. N.p., 2013. Web.
- Janovsky, Anton. "ZR6AIC." : *Setting up My Raspberry Pi as a SDR Server with RTL-2832U USB Dongle*. Blogger, 1 Feb. 2013. Web.
- Tim. "G4VXE.COM." : *Raspberry Pi Progress (RTL\_ADSB, WSPR, WSJT and Dump1090 All Compiled)*. Blogger, 6 Jan. 2013. Web.



Craft, Nix. "How To: Check the Bash Shell Script Is Being Run by Root or Not." *NixCraft Linux Tips Hacks Tutorials And Ideas In Blog Format RSS*. NixCraft, 6 Jan. 2008. Web.

Markgraf, Steve, Dimitri Stolnikov, Hoernchen, Kyle Keen, Christian Vogel, and Harald Welte. "Rtl-sdr." – *OsmoSDR*. Edgewall Software, 10 Apr. 2014. Web.

O'Hanlan, Martin. : *Raspberry Pi*. Blogger, 10 June 2012. Web.

Mintaka. "UMTSkeeper: Keep Your UMTS/GPRS/GSM Connection Alive Automatically." *UMTSkeeper Internet Keep-online*. N.p., n.d. Web.

Jämsä, Jamie. "How to Get Games to Work with Hamachi on Windows 8 | Wiretuts." *Wiretuts*. N.p., 28 Mar. 2013. Web.

"Raspberry Pi + Ser2net = Cheap NM16A (Serial Console Server)." *LesserEvil*. WordPress, 13 Apr. 2013. Web.

"Raspberry Pi and the Serial Port." *Raspberry Pi and the Serial Port*. HobbyTronics Ltd, 2014. Web.

"RPi Low-level Peripherals." *ELinux.org*. N.p., 10 Feb. 2014. Web.

"ArduPlane." *Ardupilot-mega*. DIY Drones, June 2013. Web.

Pursifull, Michael. "APM Planner / Mission Planner." - *DIY Drones*. Ning, n.d. Web.

"X-Plane 10 Global | The World's Most Advanced Flight Simulator - X-Plane." *X-Plane*. N.p., n.d. Web.

"RTL-SDR Tutorial: Cheap ADS-B Aircraft RADAR - Rtl-sdr.com." *Rtlsdrcom*. N.p., 13 Apr. 2013. Web.

Contarino, Ph.D., V. and Contarino, Ph.D., M. (2014). *USING ADS-B TO AVOID COLLISION BETWEEN UASs*. N.p., n.d., Web.

Haversine Formula in Python (Bearing and Distance between Two GPS Points)."- *Stack Overflow*. Stack Exchange Inc., n.d. Web.

## **YouTube Videos**

Barnatt, Christopher. "Setting Up a Raspberry Pi." *YouTube*. YouTube, 18 Oct. 2013. Web.

"Ardupilot Mega and Mission Planner with X-Plane Flight Simulator 10x." *YouTube*. YouTube, 06 Jan. 2014. Web.

Mijoiu, Paul. "3G Telemetry on ArduPilot 2.5 (APM) via a Raspberry PI (RPI)." *YouTube*. YouTube, 14 Apr. 2013. Web.

## **Special Thanks**

At this time, I would like to thank my advisors for all of their help and support. I would like to give a special thanks to Dr. Frank for lending me a Raspberry Pi, SD card, and a wireless USB adapter. Also, I would like to thank the Bess H. Ward scholarship foundation because without this scholarship I would most likely be unable to continue with the project due to a delay in funding. Thank you both and I hope that this thesis will help future students further their knowledge in the fields of electrical engineering, autonomous flight, communications, and computer science.