Autonomous Target Scoring Drone

EE 175 Final Report

Department of Electrical Engineering, UC Riverside

Project Team Member(s)	Jiaming Zhou & Gerardo Hernandez
Date Submitted	19, March, 2018
Section Professor	Dr. Liang Ping
Revision	Final
Permanent Emails of all team members	Jzhou010@ucr.edu Ghern027@ucr.edu

Revisions

Version	Description of Version	Author(s)	Date Completed	Approval
Final	This is the final version of the report. We did not submit any previous draft due to the limited around of time we had.	Jiaming Zhou & Gerardo Hernandez	03/19/2018	

Table of Contents

1 * Executive Summary

Our senior design consists of making an autonomous target scoring drone. The objective of the project came from the NAVY. The NAVY asked our senior design lab to design a system that enables a drone to fly and score targets located at predetermined GPS coordinates. The overall goals were to have a base station where the user would be able to select the number of targets, enter in coordinates to the destinations, and always have the ability to manually take control of the drone. Another set of goals was to have the drone hover over the targets, take images of the targets, and score each target. The design objectives consisted of hacking into an off the shelf drone controller and sending it commands so that it could take the actions necessary to complete the objective. The key features consist of having LED indicators on the base station alert the user if in autonomous or manual mode, alert the availability of GPS, sonar, and compass data, and alert if RF communication was lost. In addition, the base station constantly displays whether the drone is taking off, calibrating, going to a destination, or if going back home. Our testing results showed that we can keep communication with the drone at a distance of approximately 200m, fly for approximately 17 minutes at an altitude of approximately 20ft, and have a 2m radius accuracy in reaching GPS locations. Our achievements consist of completing all of our objectives even though one of the team members left the group 3 weeks into the 20 week project duration.

Our project's intended use is to assist NAVY personnel in scoring targets. Navy personnel would no longer have to go into the battlefield and see if a target has been hit. This project has the potential to save NAVY personnel lives and decrease the time spent scoring targets. If a few sensors were to be added to our design, we would have the ability to measure RF signals, collect crop field information, deliver items, and assist in search and rescue.

2 * Introduction

2.1 * Design Objectives and System Overview

The concept of our project is to design a system which can autonomously score a target from the sky. The application of our design pertains to aiding NAVY personnel in target scoring. We're designing a system which gives on off the shelf drone the ability to fly autonomously while scoring targets. The project is meaningful because NAVY personnel no longer have to risk their lives by checking whether a target has been hit. The NAVY intends on using our project as a way to score targets for field testing. The NAVY wants to be able to take our project anywhere and have it function autonomously. The project is related with electrical engineering because power distribution, wireless communication, embedded systems, circuits, computer vision, and coding are all needed in order to accomplish our objective. The overall goal of the project is to control an Autel X-Star Premium drone so that after entering GPS coordinates into a base station, the drone can autonomously take images of multiple targets and score them.

A high-level description of our system can be broken down into 2 parts. The first part consists of data acquisition. Data acquisition is composed from the components mounted on the drone. The components are a sonar sensor, GPS module, triple-axis accelerometer/magnetometer, 5MP camera, raspberry pi, Arduino Uno, Arduino Nano, and a transceiver. The second part consist of the base station. The base station is the user interface and control center of the drone. The base station takes in GPS coordinates, controls and sends commands to the drone. The base station is composed of a 4x4 keypad, a 16x2 LCD screen, 6 LEDs, an Arduino Mega, 2 Dual OpAmps, a voltage regulator, and a transceiver.

The integrated development environment (IDE) we started off using was Atmel Studio. We used Atmel Studio to program the Atmega1284 microcontroller but eventually realized that we were making our work harder by working with the limited 8-bit microcontroller. The integrated development environment we ended up using is the Arduino IDE named Sketch.

Quantitative technical design objectives:
- Communication with drone at a distance of approximately 200m,
- 17 minutes of flight time at an altitude of approximately 20ft
- 2m radius accuracy in reaching its GPS location

Shared Responsibilities:
- Researching: Components and Algorithms
- System Assembly: Wiring, Soldering, Power Management
- Documentation: Final Report & Note Taking
- Testing and Debugging

Jiaming:
- On-board system:
  - Acquiring sensor information
  - Designed safety protocols
- Base-Station:
  - Auxiliary codes: provide specific functionalities.

Gerardo:

- Base-Station
  - State Machine
- 3D Design & 3D Printing
- Computer Vision & Image Processing

2.2 * Backgrounds and Prior Art

The literature which allowed us take control of the drone:
http://jireren.github.io/blog/2016/11/25/how-to-hack-a-drone-controller/
This website hacked into the Autel-X Star Premium controller with the use of 2 dual op amps and an Arduino Uno microcontroller. We started our design in a similar fashion except that we chose to use an 8-bit microcontroller name the Atmega1284, 5 multiplexers, 4 adjustable voltage regulators, and 2 voltage dividers. The rest of the design was created by us.

2.3 * Development Environment and Tools

Hardware & development environment:

- Raspberry pi with MATLAB
- Arduino Uno with Sketch
- Arduino Mega with Sketch
- Arduino Nano with Sketch

Testing equipment:

- NO.APS-274K Power Supply
- FTIKE A830L Multimeter
- BK Precision 40Mhz Oscilloscope Model: 1541A

2.4 * Definitions and Acronyms

Base station: Used to enter coordinates, manually control the drone, indicate the drone's actions, and indicated if GPS, compass, and RF antenna communication is maintained. The base station contains the Arduino Mega, the microcontroller which has the state machine code written to it.

3 * Design Considerations

3.1 * Realistic Constraints

One of the earliest constraints we had to deal with is weight constraint. The drone we are working with has a maximum payload weight of 400 grams. This means that we have to design our onboard system whose weight is within the limit of 400 grams. This limits the selection of battery that we could choose from. We have to make sure that the battery we chose has large enough capacity to support the whole system for a considerable amount of time and is not too heavy. As a result of the limited capacity of the battery, the range of components that we could choose from is also limited. The transceiver, which is used to transmit sensor information back to the base station, for example could not consume too much power. Even with the low power transceiver we have chosen, it could still consume up to 250 mA per hour. Along with other small components of the whole system, the battery could get drained at rate that is too fast for the whole system to be any useful.

Budget constraint is also a factor that has a significant impact on our design. We are not told at the beginning how much funding that we are going to be given. Even though we are told that we might be reimbursed afterward, we have to relied on our own funding during the whole design process. Since we are students with limited amount of capital resources, we could not purchase expensive devices even though it could have allowed us to make better designs. One of the examples is that the drone has a problem going to the right direction when there is a lot of wind. The reason for this phenomenon is that when there is a lot turbulence, it causes the compass to produce incorrect data. The compass we implemented is simply a magnetometer with little support on tilt compensation. This means that whenever the compass is tiled in any direction, the reading are rendered useless. The constraint on budget has limited us to design our system with inexpensive consumer electronic devices.

Another constraint that we have is the amount of accessibility in regard to controlling the drone. Since the drone used is a consumer product which is designed for basic uses such as flying over scenes to take images, there is not a lot of support that allows users to define their own functionalities. We are limited to only the higher-level movement commands, which we gain access to by hacking into the transmitter.

Lastly, we also have the problem of processing power. One of the design objectives is to score targets using computer vision. However, even simple image processing could be very computationally expensive. To design a standalone system that could be mounted onto the drone is a challenge. The device we have chosen to do the image processing is the raspberry pi, which is significantly more powerful than the microcontroller we used to gather information. Yet by using the raspberry pi, it could still be a challenge to do the target scoring in real time. Eventually we have decided to do the image processing on the pc and use the pi as an image capturing device only. This is done to also limit power consumption.

3.2 * System Environment and External Interfaces

As it has been mentioned in the previous section, we are limited to only the higher level movement commands of the drone. To control the device using custom hardware, we must send commands, voltages in our case, through the original transmitter. The way we accomplished this is by reconnecting the wires that originally transmit commands from the joysticks, which control by the user, to the corresponding outputs of our custom control system. The system imitates the same voltages that prompt the drone to perform certain actions depending on what actions are required. This is the only limitation we have in regards of what we must operate in.

3.3 * Industry Standards

All the communication protocols used in our system, such as SPI and I2C, are integrated capability of the modules that we are using. Please refer to their documentations for compliance of industry standards. However, the design of our state machine is designed with the standard taught in the course EE 120B at UCR. The structure and means by which we implement the algorithms follow strictly the guideline that was taught to us.

3.4 * Knowledge and Skills

The skills needed for this project are mainly coding skills, embedded system, and critical thinking. Some of the knowledge used in implementing the objective are learned from courses, but a lot of other knowledge is learned through researching.

Knowledge that is required for this project includes:

1. Embedded system (EE 120B)
2. Image process (EE 152)
3. Computer Vision (EE 146)

Skills or Knowledge that we have to learn:

1. Microcontroller programming
2. Knowledge in how to use peripheral modules
3. RF Communication
4. 3D Designing

3.5 * Budget and Cost Analysis

Num	Item Name:	Cost per unit ($)
1	Adafruit Ultimate GPS Breakout	39.01
2	Elegoo Arduino Uno kit	17.99
3	Multicolored jumper wires	6.99
4	MakerFocus Wireless Transceivers	10.29

5	LSM303 Accelerometer + Magnetometer	14.93
6	Transceiver voltage regulator	9.99
7	Atmega2560 board	13.99
8	Floureon 1000mV Lipo Battery	10.99
9	Elegoo Nano boards	11.86
10	PLA 3D printer Filament	22.99

3.6 * Safety

A safety concern that we have to take into consideration is the drone losing control or performing any unexpected behaviors. Due to the uncertainties of the control system, miscalculation due to errors from sensors, there is always a possibility of the drone performing some sort of unexpected behaviors. Flying off to unspecified locations or direction, for example, can be considered as one of these behaviors. Errors caused by sensors are hard to avoid. For example, the gps used in calculating the direction in which the drone flies has chances of acquiring coordinates that are a few meters off, depending on the number of satellites it could communicate with.

A safety objective that we were trying to achieve is to have the drone behave correctly the majority of the time. The desired rate is ninety-nine percent. To achieve this objective, we have developed algorithms that check the status of all the sensors. This ensures that when any of the sensors malfunctions or loses connection, the user is alerted to prevent accidents from happening. Codes have also been included to ensure the gps information being sent back to the base station is as accurate as possible. The procedure checks to see if the gps information being sent back is valid. If the information is invalid for a certain duration of time, the station will be notified that the gps is not working properly.

Another safety objective that we are trying to achieve is that the user of the system would always be able to take control over the drone, despite the mode the drone is in, whenever he or she desires. This allows the users to avoid disastrous outcomes. This is accomplished by incorporate codes into the state machine that allows user to revert the state back to manual at anytime. To do that, the code is set to have the highest priority.

Lastly to ensure the safety of users when handling the electronics on the drone, the power sources are properly regulated through a power management system. This system involves voltage regulator and buck converter, which limit both the voltages and currents being sent to the circuits and modules.

3.7 * Performance, Security, Quality, Reliability, Aesthetics etc.

To ensure meeting the minimum performance requirement, we have listed some of the design considerations. These considerations affect the process in which we create our designs. Below is a design consideration list that we have created:

Performance - How accurate does the autonomous need to be? What is it being used for?
Security - How safe does the system have to be? How could the system cause injuries?
Serviceability - how could it be maintained?
Aesthetics - What kind of product appearance do the users expect?
Reliability - How to ensure circuits work most of the time?

3.8 * Documentation

To ensure consistency and accuracy, we documented any changes that we made on a regular basis. Changes made on either the software or hardware were written down on notebooks and documents. The way by which we prevented accidents from setting the progress back was by making new version of the code whenever we put new updates. This ensured that we could also return to the working version if the new implemented codes did not work properly or as expected.

3.9 Risks and Volatile Areas

In terms of the on-board system, an area that we consider to be volatile is the program that is used to retrieve sensor information from the modules. In total there are three modules that are constantly sending information to the microcontroller, and the microcontroller itself is constantly writing data to its port for transmission. Any change to the code might affect the functionality of the other modules. In the early development phase, we had instances of the sonar sensor conflicting with the transceiver. The transmission of data had caused the microcontroller to have pause moments where it stopped receiving sonar sensor data. This was not acceptable for the reason that the sonar sensor was responsible for the safety of the drone during the landing phase.

4 * Experiment Design and Feasibility Study

The first experiment we conducted was to see the accuracy of our GPS module. The GPS module we started off with is named Cirocomm 595k. The module would update its coordinate location after moving a distance of 200ft. The resources all indicated that the GPS coordinates should have been updating continuously without the need to travel 200ft in order to see a change in coordinates. After 2 weeks of zero progress, we ended up purchasing Adafruit Ultimate GPS. The new GPS immediately gave us GPS coordinates which updated at 10Hz.

The second experiment conducted consisted of measuring the joystick output voltages and then apply the same voltages back into the drone's controller. The reason why this experiment was necessary, was to make sure that we could control the drone's actions. The method we first used was to have the Atmega1284 send out 8 signals to 4 multiplexers, each multiplexer taking in 2 signals. 2 signals were needed in order to represent 4 values (0-15). With 2 signals per multiplexer, we were able to select 4 voltage sources as inputs per multiplexer. Each multiplexer output had a designated joystick voltage input. After being able to select the correct voltages per multiplexer, we removed the propellers from the drone, turned it on, and began to test each of the voltage configurations. The drone's motors rotated accordingly to our commands. We then proceeded by adding the propellers and holding the drone from its legs to further test the commands. As expected, the drone would attempt to rise when we sent it the voltage necessary to rise and attempt to descend when we sent it the voltage necessary. We proceeded by taking the drone out to a park, attaching a rope to its legs, and sending it commands from the Atmega1284 microcontroller. The drone behaved as expected. We were able to take full control of the drone by mimicking the voltages that the joystick would have been outputting.

The third experiment took place after we realized that the Atmega1284 was going to limit us in the amount of precision we would want when entering GPS coordinates. Seeing how the Atmega1284 is only an 8-bit microcontroller, collecting GPS coordinates and compass readings was going to be a problem. With 8 bits, we are only able to represent values from 0-255. We needed values which could represent in the millionths for GPS coordinates and a value from 0-360 for the compass. We decided that switching to the Arduino Mega would be a good choice because it could represent values using 32 bits (3.4028235E+38 and as low as -3.4028235E+38). Another reason why we chose to use the Arduino Mega is because we would now have the ability to use PWM signals mimic the different voltages required to control the drone. The experiment required us to make a non-inverting voltage follower in order to have a constant voltage going to the drone's controller opposed to have a PWM signal. After constructing the circuit, we tested it by sending the drone commands and seeing if the drone's controller responded as expected. Everything worked out and we were then able to eliminate the Atmega1284, 4 multiplexers, 4 adjustable voltage regulators, and 2 voltage dividers and instead use the Arduino Mega.

The fourth experiment consisted of getting 7.4V to step down to 5V and having the ability to withdraw at least 700mA from the voltage source. The LM7805 voltage regulator we were using was only able to putout 500mA. The alternative was to use the LM2576, a buck converter which has the ability to output a range of 1.23V-37V and provide a current of up to 3A. After connecting the circuit we saw that the hardware was in full operation because now we had met the amp expectations for the raspberry pi.

The fifth experiment we conducted was in regard to seeing the range of the RF antenna communication. The reason why the RF antenna was implemented into the design was mainly because this method would allow us to reach a large range with minimal financial costs.

The sixth experiment we conducted was in regards to seeing how accurate the sonar sensor would read distance. We needed an accuracy of approximately 3 inches. The reason we used a sonar sensor was to have the drone SHUTOFF after descending and being situated 15 inches above the ground. If we were 12 to 18 inches above the ground and had the drone go into its SHUTOFF state we would not encounter any problems. If the drone were to go into its SHUTOFF state too high or too low from the ground, then we would encounter dramatic problems, problems which might terminate the drone physically.

The seventh experiment was to see the accuracy of the compass. The reason why a compass was implemented into the design was to ensure that the drone faces its destination and flys towards it, thus taking the shortest path to the destination. We could have used strictly used the GPS module to tell us where the drone was located. For example, we could travel down the latitude until our latitude difference is 0 and the same could have been done with longitude. This would in the long run take too much time but would also complete the objective.

4.1 * Experiment Design

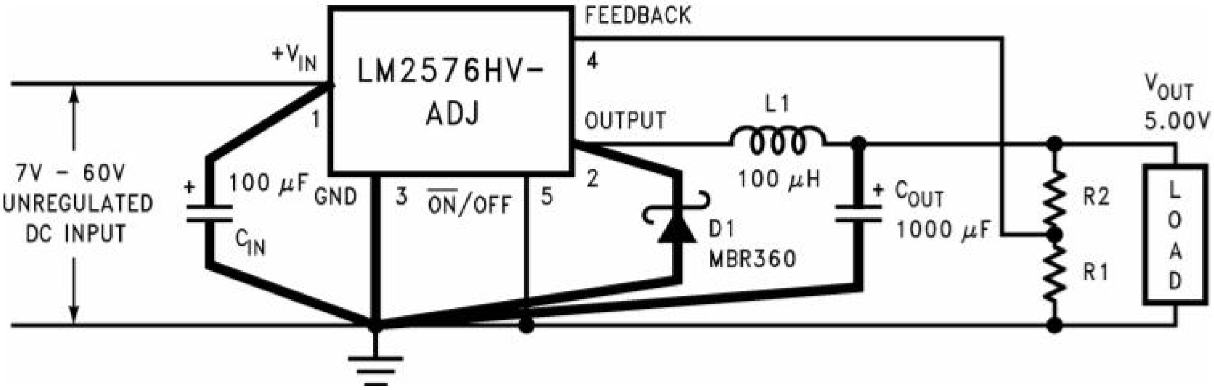
GPS experiment:

The objective was to read the GPS coordinates of our current location. The setup involved the whole team. We connected the GPS module to an Arduino Uno and connected the Arduino Uno to our laptop, via USB. We were then able to read the coordinates of the GPS module on our laptop screen. We expected the GPS coordinates to update a few times per second but instead only saw updates after travelling 200ft.
After the GPS experiment Emmanuel Martinez left the group for the rest of the senior design.

Measuring Joystick Output Voltages:
The objective was to measure the voltages coming from the joysticks of the drone’s controller. Jiaming and Gerardo opened the drone controller and discussed which of the wires were expected to have the varying output voltage. Each joystick had a pair of black, red, and yellow wires. Jiaming and Gerardo decided to cut the yellow wires and take their voltages, assuming that the black wire was ground, red wire was high, and yellow wire was the varying output voltage.

Gerardo set up the Atmega1284 with the 4 multiplexers in order to have varying voltage sources directed towards the drone’s controllers. Jiaming set up the adjustable voltage regulators so that the same voltages that the drone’s controller outputted would be outputted by the adjustable voltage regulators.

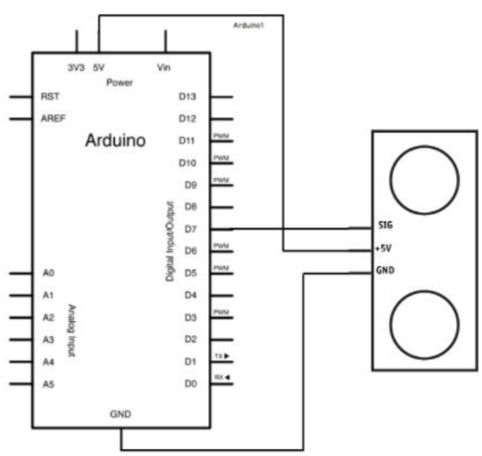
Gerardo set up the buck converter circuit by following the following schematic.



R1 = 1000Ω
R2 = 3065Ω

After Jiaming connected the RF antenna, Arduino Uno, and the GPS module, Gerardo took the components out into the street while Jiaming stayed in a stationary position to see if he was still receiving data. We expected to receive data at a minimum range of 150m.

The objective of the sonar sensor is to have the ability to measure the distance in front of the sonar sensor. In our case the sonar sensor is placed underneath the drone such that if the drone were to be placed on the ground, the sonar sensor would be 1 inch off of the ground. After reading the sonar sensors specs we knew that the distance range for the was between 1.5inches to 118 inches. Jiaming set up the sonar sensor with the Arduino Uno as follows:



The objective of using the compass is to know the exact direction in which the drone is facing. The compass was connected to the Arduino Uno using I2C. The procedure before reading data from the compass involved rotating the compass horizontally in order to calibrate the compass. We expected to turn the compass by 90,180,270, and 360 degree and get the same readings on the display.

4.2 * Experiment Results, Data Analysis and Feasibility

The whole team conducted the GPS experiment. The results of our first GPS module turned out to be insufficient because the GPS coordinates would only update after having traveled 200ft. After the team replaced the 595k GPS module with the new Adafruit's Ultimate GPS with the same setup, we were able to notice a major difference in the coordinate update frequency which was 10Hz.

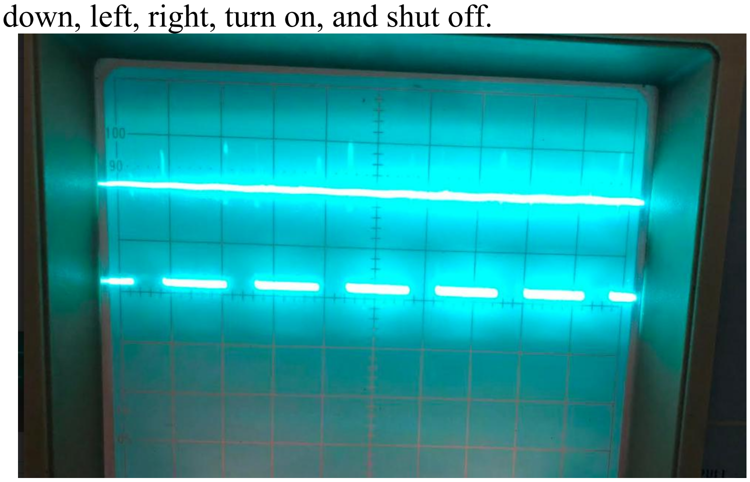
Jiaming measured the varying output voltages to be:
Left Vertical:
- Up 2.65V
- Middle 1.51V
- Down 0.48V

Left Horizontal:
- Spin Left 0.47V
- Middle 1.63V
- Spin Right 2.69V

Right Vertical:
- Forward 0.46V
- Middle 1.54V
- Backward 2.71V

Right Horizontal:
- Left 2.63V
- Middle 1.63V
- Right 0.44V

Gerardo set up the non-inverting voltage followers. The following image shows the PWM signal from the Arduino Uno on the bottom and the constant voltage from the output of the non-inverting voltage follower. We were able to mimic the varying voltage from the joysticks. At this point we had the ability to give the drone high level commands such as forward, backward, spin right, spin left, up, down, left, right, turn on, and shut off.



This figure shows the input (Arduino PWM signal) on the bottom and the output (constant voltage from the dual op-amp)

The RF antenna was able to send data at a distance of approximately 200m. Jiaming would receive data at about 2Hz. The results from this experiment gave us an indication as to how far we could fly the drone without losing communication. The RF antenna was kept in a low power state in order to conserve energy. We could have made the range longer but decided to keep it this way due to our power constraints.

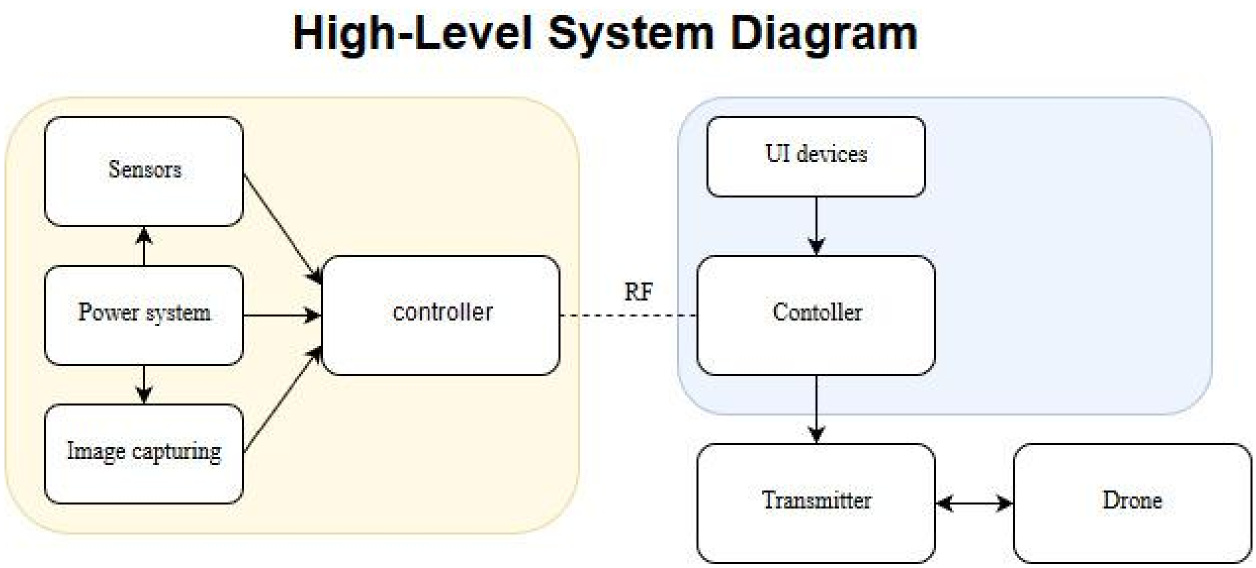
The sonar sensor experiment was conducted by Jiaming and Gerardo. We got a ruler, placed a flat solid object in front of the sonar sensor at multiple distances, and compared the ruler's distance to the readings seen on the serial monitor of Arduino's IDE (Sketch). The distance resulted to be accurate to within 1cm.

We placed the compass on a turntable and rotated the table in 90 degree intervals. The compass displayed its accuracy to within 2 degrees. The drawback in using the compass is that when tilted towards its pitch of roll directions, the compass jumps to random values. This makes the compass very unstable when flying on the drone. An algorithm was implemented in the state machine to compensate for incorrect compass readings.

5 * Architecture and High Level Design

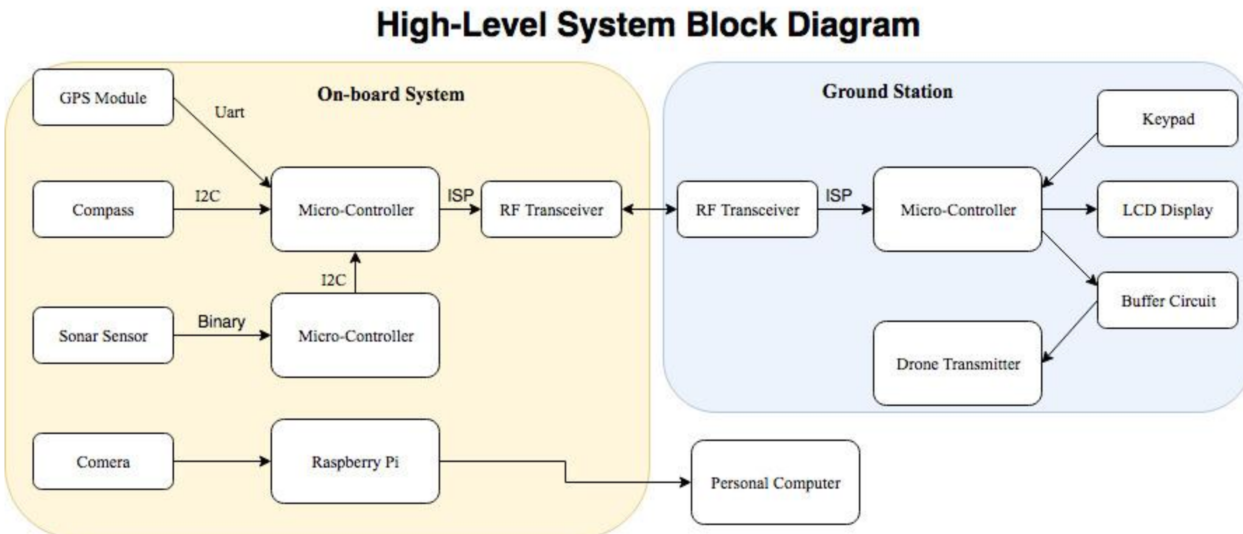
5.1 * System Architecture and Design

Top-Level System Block Diagram:



The whole system consists of two main parts. The on-board system, which is made up of sensors and two microcontrollers which send back the necessary information. This information is needed in order for the base station to figure out the course by which the drone navigate to the destination. The second main part, as has been mentioned, is the base station. The base station is made up of a microcontroller and other peripheral devices. The microcontroller serves as the brain of the whole system. It uses the information sent back from the drone to determine which commands or actions should be send to the transmitter. The peripheral devices are used to form the main user interface, which is used by the users to enter the necessary parameters.

5.2 * Hardware Architecture

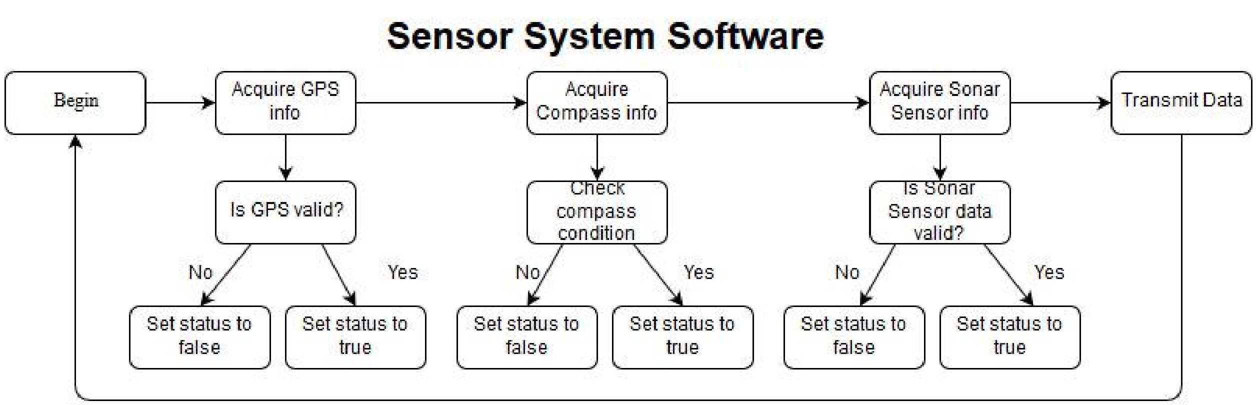


Responsibility:

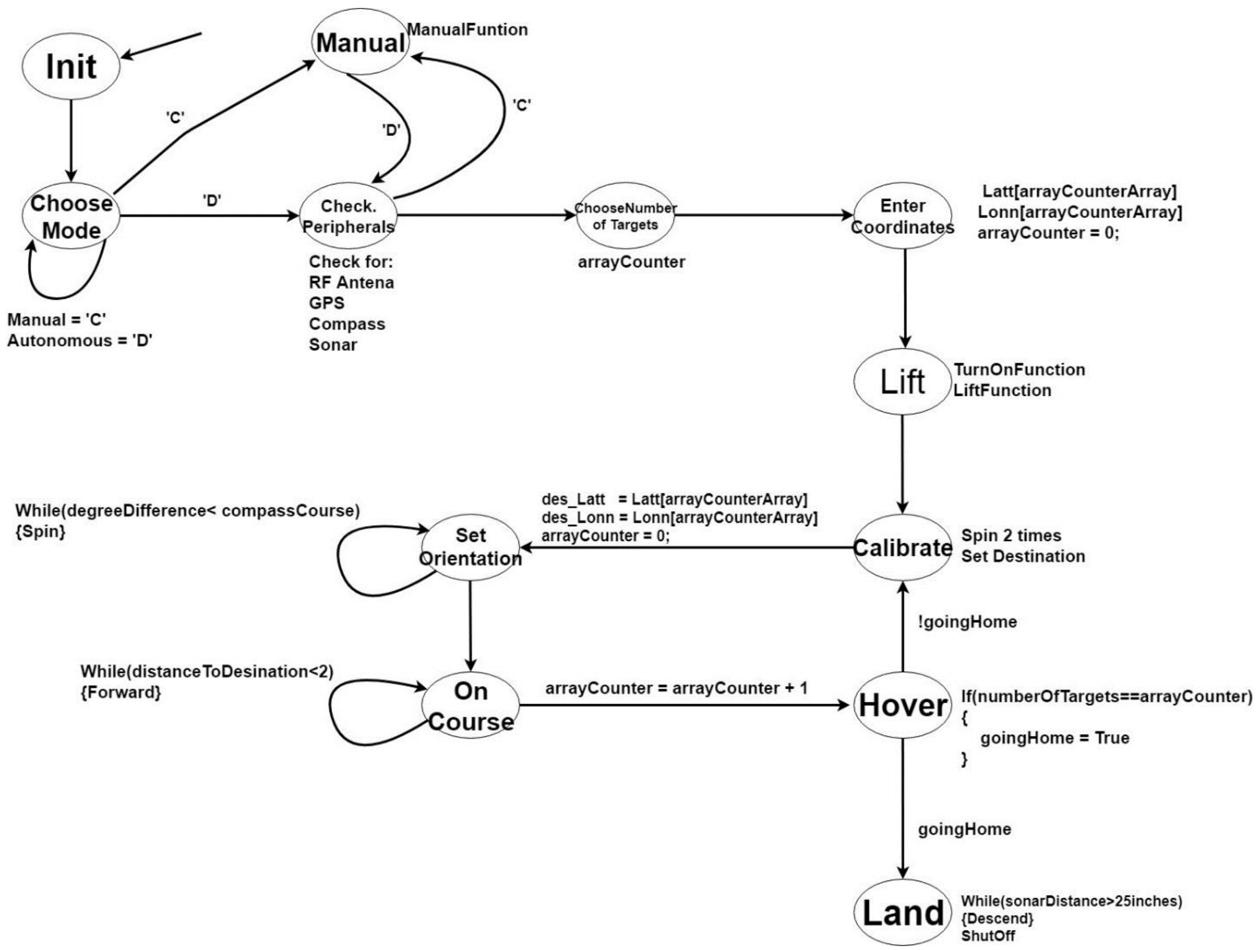
The responsibility of this project is evenly divided and distributed among both members. Jiaming Zhou is mainly responsible for handling the on-board system, which includes integrating the sensors, transceivers, and microcontrollers. Gerardo Hernandez is responsible for putting together the base station. The base station consists of a microcontroller and multiple user interface related peripheral devices. The work is shared equally by both of us.

5.3 * Software Architecture (only required if your design includes software)

On-board System (Jiaming Zhou):



Base station State Machine (Gerardo Hernandez):



5.4 * Rationale and Alternatives

Current approach:

The approach that we have adopted to implement our project objective is determined after we have considered all the factors. These factors include constraints and expertise of the members. One of the biggest constraint that we have to work around while designing our architecture is the drone. The drone is an off the shelf product whose main use is for entertainment purposes. It is used for consumers to fly around and take pictures or video of scene. It lacks the accessibility for both hobbyist and professionals to do any sort of modification to suit their needs. Hence, our only option to get access to the control of the drone is to imitate the highest level commands which the original transmitter uses to control the drone. This has determined a significant portion of our hardware architecture in terms of how to control the movement of the drone.

The expertise or skill possessed by the member of the project also play a significant role in determining the overall architecture of the project. Since we have gained a substantial amount of knowledge pertaining to state machine through our courses taken at UCR, we have decided to implement a state machine to function as the main processor of the system. This enables us to make a system that is more efficient and effective, and it also refrains us from having to learn another totally new method, which is not time efficient given that we only have two quarters to work on the project along with other responsibilities from classes.

Another constraint which determines the approach we have taken is the amount of resources we have. The budget we get allocated with is very limited, hence the whole system has to be built with components that are reasonably priced or already owned. There are many devices that could provide better accuracy or even functionalities than the ones we have decided to use; however, the cost will get substantially higher.

Alternative approach:

An alternative approach is to use a more customizable drone, which allows us more freedom to adjust the controlling hardware to fulfill the need of the project objectives. This way we could gain access to the drone's internal IMU system and its telemetry system. By adopting this approach, it could improve the accuracy of the navigation system and avoid some of the technical difficulties we faced during the process.

6 * Low Level Design

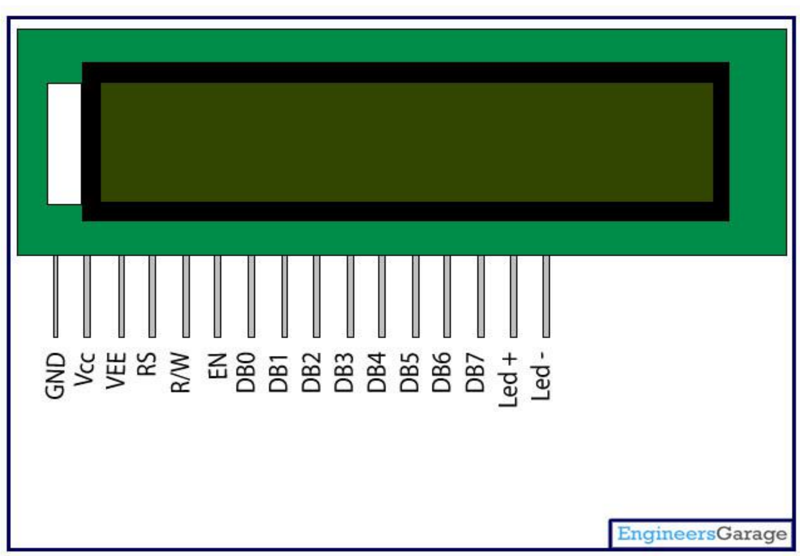
6.1 * Base Station's *16x2 LCD Display*

Gerardo is responsible for this module

The LCD display is connected to the Arduino Mega inside of the Base Station. The LCD display serves as the Base Station's state machine output. This is where the user sees the coordinates entered, drone states, and actions the drone undergoes. The connections from the Arduino Mega to the LCD display are as follows:

From left to right

LCD Display	Arduino Mega
GND	ground.
Vcc	5V.
Vee	10k potentiometer.
RS	pin 38.
R/W	ground.
EN	pin 36.
DB4	pin 28.
DB5	pin 30.
DB6	pin 32.
DB7	pin34.
LED+	5V in series with a 330Ω resistor.
LED- is	ground.



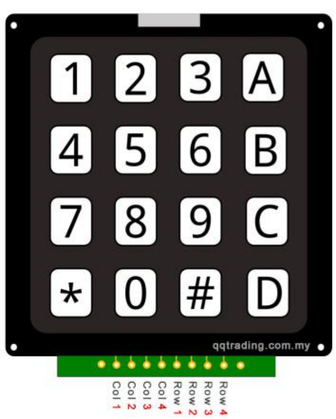
The potentiometer connected to Vee is used to change the contrast on LCD screen.

6.2 * Base Station's *4x4 keypad*

Gerardo is responsible for this module.
The 4x4 keypad is where the user can select autonomous or manual mode, enter in coordinates, and control the drone's actions.

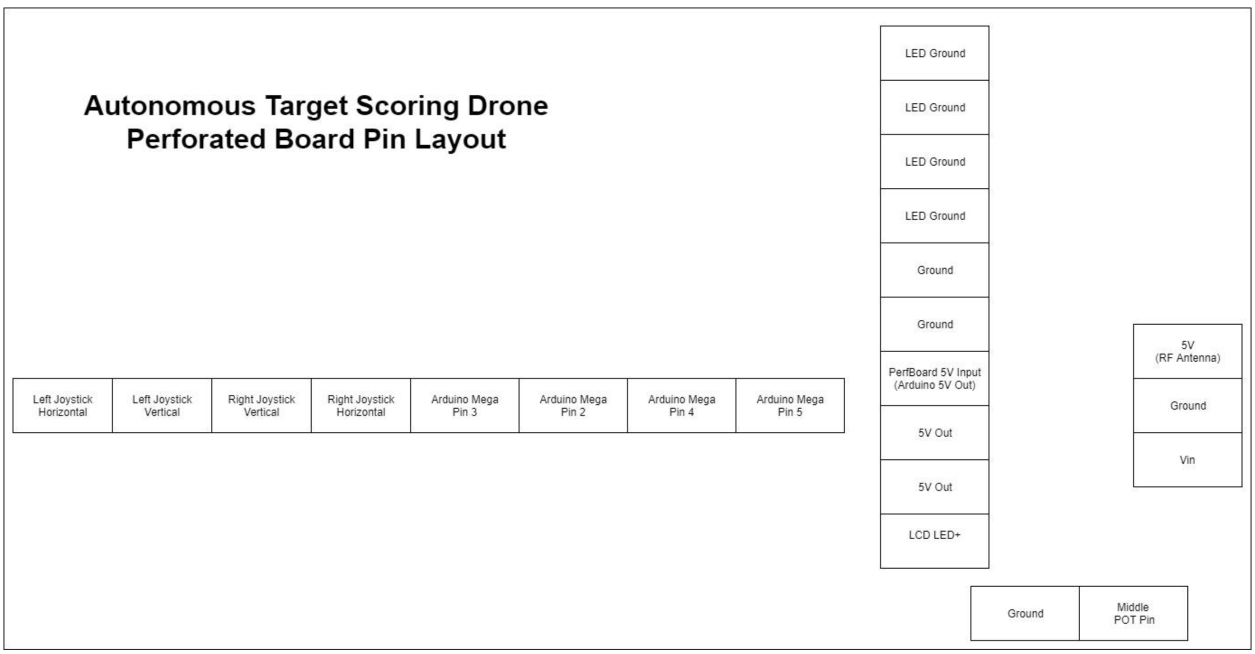
Keypad Functions:
1 - Spin Left
2 - Forward
3 - Spin Right
4 - Left
5 - Back
6 - Right
7 - Down
8 - Stationary
9 - Up
A - Turn On
B - Turn Off
C - Manual Mode
D - Autonomous Mode
* - Decimal Point
# - Enter
The connections from the Arduino Mega to the 4x4 Keypad are as follows:

Keypad	Arduino Mega
Row1	A3
Row2	A2
Row3	A1
Row4	A0
Col1	A7
Col2	A6
Col3	A5
Col4	A4



6.3 **Perforated Base Station Board*

Gerardo is responsible for this module.
The perforated is where the Joysticks Potentiometers Inputs, Arduino Mega PWM Input, 5V Arduino Mega Output, LED Grounds, Voltage Regulator, and LCD Contrast Potentiometer Pin are all located. This board has 2 LM358N Low Power Dual-Operational Amplifiers which have outputs connected to all of the Joystick Outputs. Arduino Mega Pins 2,3,4, & 5 are the PWM signals the Op-Amp uses as inputs. The drone's controller potentiometers are connected to the Op-Amp outputs. The way in which the base station is set up is such that the user only has to connect power via USB (5V minimum) to the Arduino Mega and connect the 5 wires which are stationed on the clip of the Base Station's lid.
Black wire - Drone Controllers Ground
Green wire - Drone Controllers Left Joystick Horizontal
Yellow wire - Drone Controller Left Joystick Vertical
Yellow wire with black tape - Drone Controller Right Joystick Horizontal
Green wire with Black tape - Drone Controller Right Joystick Horizontal



6.4 * Base Station's Arduino Mega*:*

Gerardo is responsible for this module.
The Arduino Mega is the microcontroller used for the Base Station. The purpose of the Arduino Mega is to run the state machine, turn on/off LED indicators, connect the RF antenna, connect the 4x4 Keypad, connect the 2x16 LCD keypad, and connect the drone's potentiometer inputs. The connections from the Arduino Mega to the rest of the components are as follows:

Keypad | Arduino Mega
Row1 A3

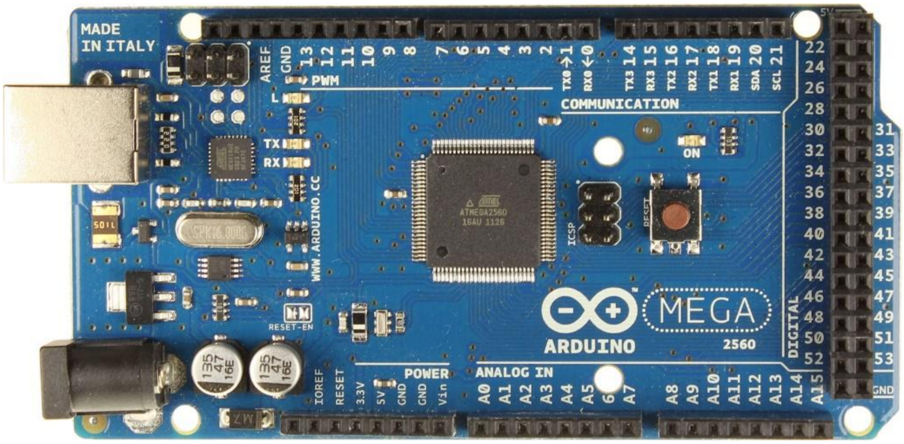
Row2	A2
Row3	A1
Row4	A0
Col1	A7
Col2	A6
Col3	A5
Col4	A4

2x16 LCD Display	Arduino Mega
GND	ground.
Vcc	5V.
RS	pin 38.
R/W	ground.
EN	pin 36.
DB4	pin 28.
DB5	pin 30.
DB6	pin 32.
DB7	pin 34.
LED-	ground.

Indicator LED's	Arduino Mega
GPS	pin22
Compass	pin23
RF Antenna	pin24
Autonomous	pin25
Manual	pin26
User Command	pin27

PWM Signals to Perforated Board	Arduino Mega
LeftJoyStickVertical	pin2
LeftJoyStickSpin	pin3
RightJoyStickBackForth	pin4
RightJoyStickLeftRight	pin5

Antenna Connections	Arduino Mega
CE	pin15
CSN	pin16
MISO	pin50
MOSI	pin51
SCK	pin52



6.5 * *Raspberry Pi B+ Plus 5MP Camera:*

Gerardo Hernandez is responsible for this module.
Connect the Raspberry Pi to the 5MP camera module.
Install the computer vision toolbox on Matlab.
Make sure to place the Raspberry pi and the computer collecting the score on the same network.
After both computer running Matlab and Raspberry pi are connected to the same network, run the following:

```
clear rpi
rpi = raspi();
cam = cameraboard(rpi,'Resolution','640x480');
d = imdistline;
```

Measure the radius of your target and hit. Enter those radii in to the Matlab script imfindcircles functions.
Run the script while the drone is hovering over the target to collect score.

6.6 * 3D Design and Printing*:*

Gerardo Hernandez is responsible for this module.
The Base Station and on board components are all situated on PLA material and design using SolidWorks2017. The LED's on the base station are held with friction. The 3D print was printed at 200micron resolution running at 150mm/s.

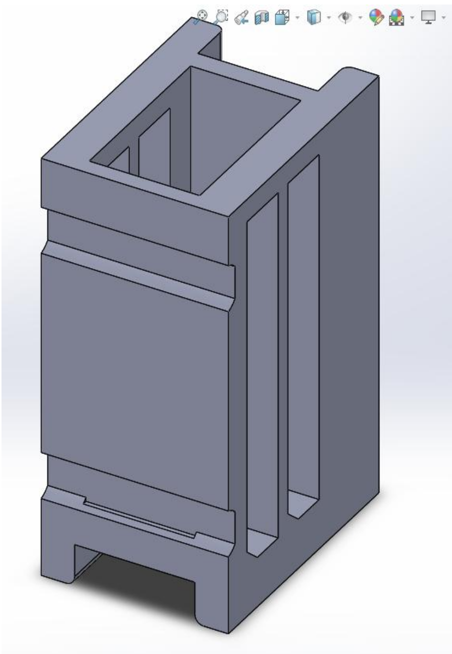
Screw Sizes:
Base Station:
Diameter 3mm
Length 6mm
Battery Holder:
Diameter 3mm

Length 24mm

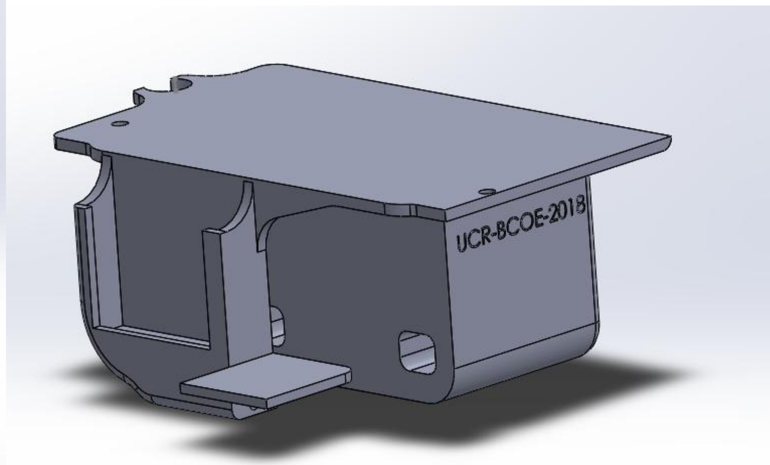
Leg Attachment *to* Navigation Component Holder:

Diameter 4mm

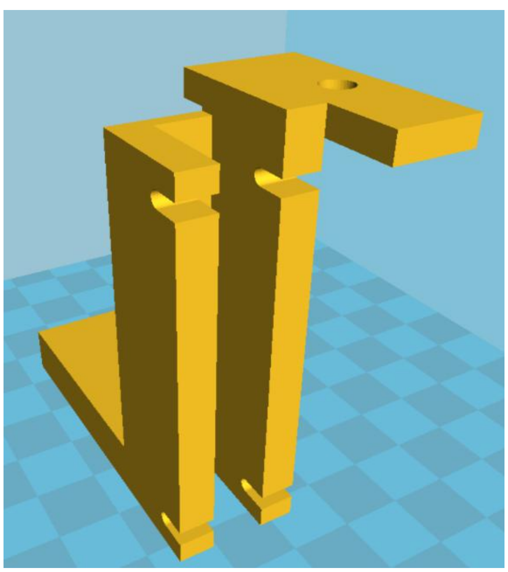
Length 17mm



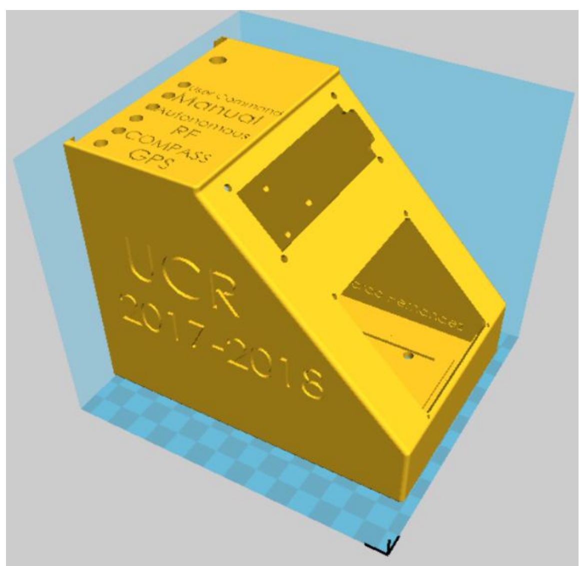
Battery Holder



Navigation Component Holder



Drone Leg Attachment

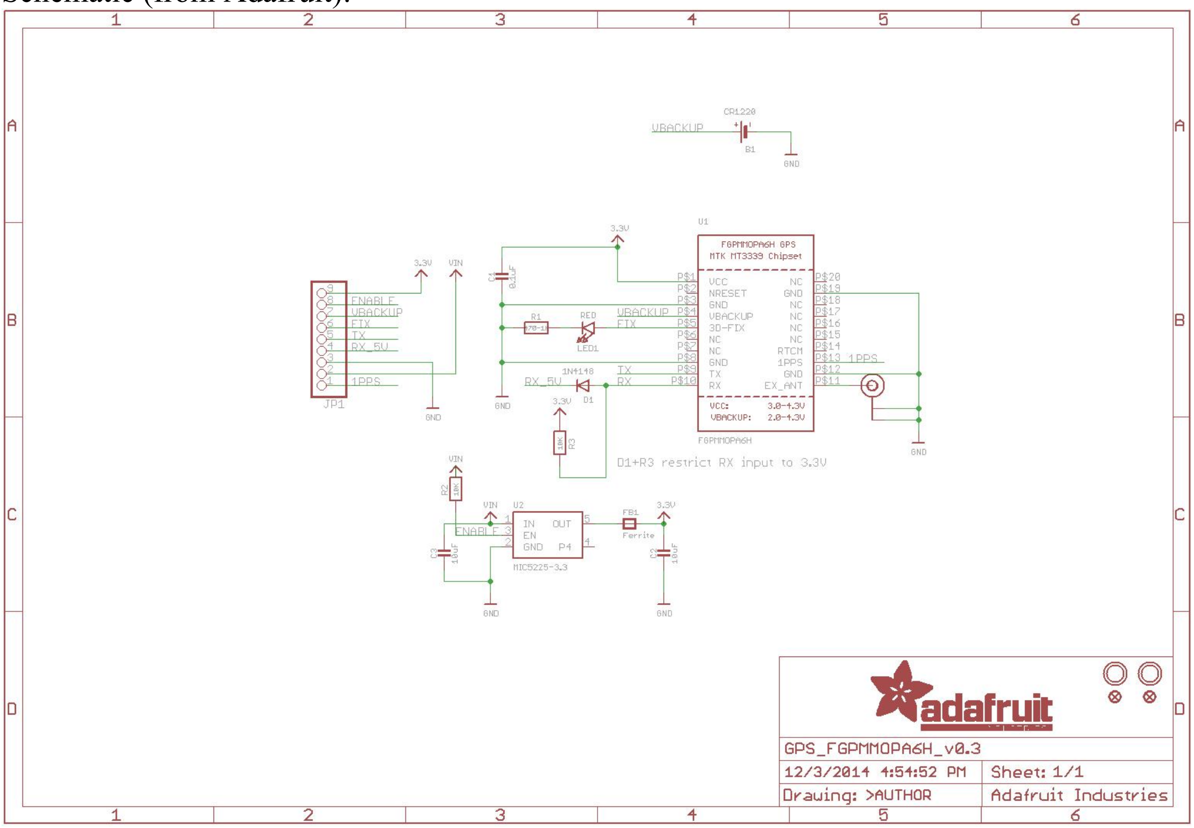


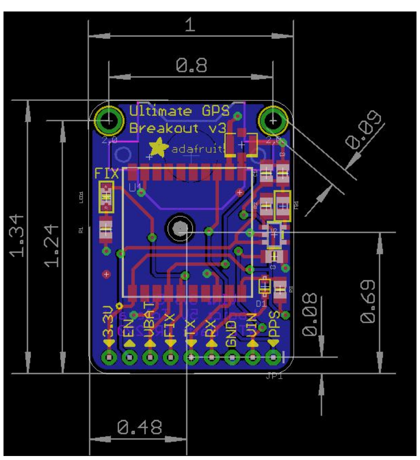
Base Station

6.7 ** Adafruit Ultimate breakdown GPS module:*

Jiaming Zhou is responsible for this module.
Reference:
https://learn.adafruit.com/adafruit-ultimate-gps/downloads
https://www.adafruit.com/product/746

Schematic (from Adafruit):





This module is built using the MTK3339 chipset. It could track up to 22 satellites on 66 channels. The sensitivity of the receive is -165dB. It refreshes its location ten times per a second. It can be powered off a 3.3 voltage source, but a 5-volt source is preferred. During navigation, it consumes only 20mA, which makes it ideal for our project due to power limitation. The interfacing protocol is uart, which utilizes RX and TX ports.

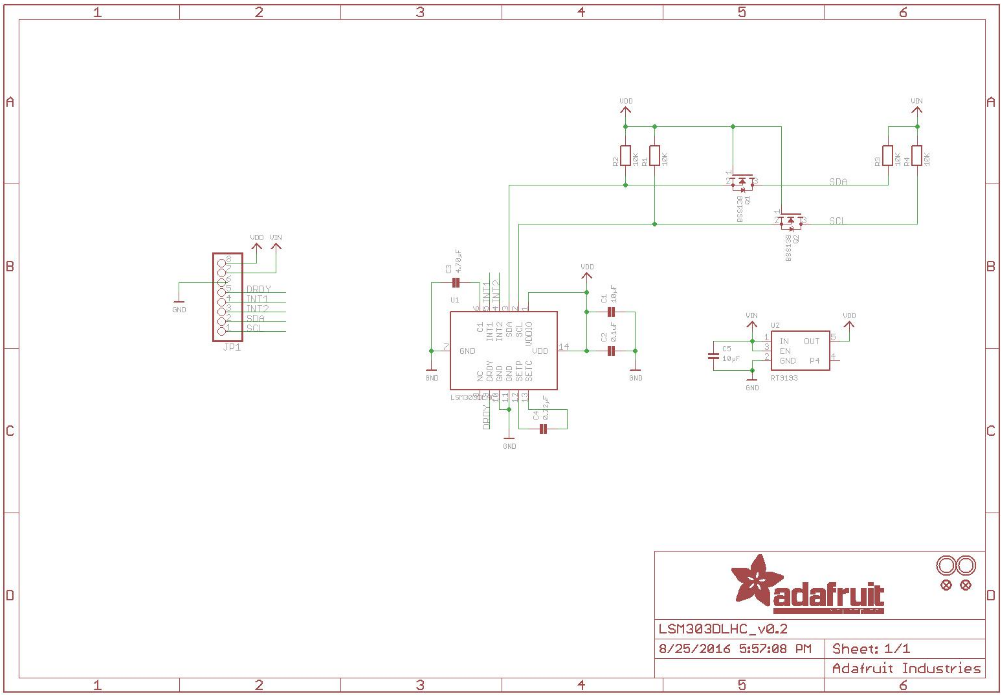
Technical Details (Source: Adafruit):

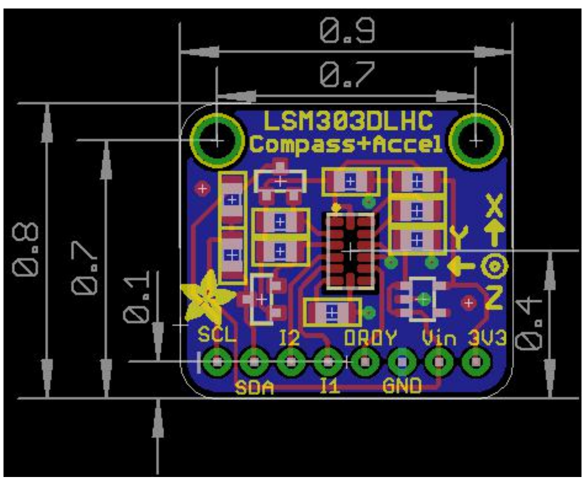
- Satellites: 22 tracking, 66 searching
- Patch Antenna Size: 15mm x 15mm x 4mm
- Update rate: 1 to 10 Hz
- Position Accuracy: < 3 meters (all GPS technology has about 3m accuracy)
- Velocity Accuracy: 0.1 meters/s
- Warm/cold start: 34 seconds
- Acquisition sensitivity: -145 dBm
- Tracking sensitivity: -165 dBm
- Maximum Velocity: 515m/s
- Vin range: 3.0-5.5VDC
- MTK3339 Operating current: 25mA tracking, 20 mA current draw during navigation
- Output: NMEA 0183, 9600 baud default, 3V logic level out, 5V-safe input
- DGPS/WAAS/EGNOS supported
- FCC E911 compliance and AGPS support (Offline mode: EPO valid up to 14 days)
- Up to 210 PRN channels
- Jammer detection and reduction
- Multi-path detection and compensation

6.8 * *Adafruit LSM303 Magnetometer and accelerometer:*

Jiaming Zhou is responsible for this module.

Reference:
https://cdn-shop.adafruit.com/datasheets/LSM303DLHC.PDF





This is a module that combines a magnetometer with a triple-axis accelerometer that can be used as an IMU unit. It can be powered off either a 3.3 V or a 5 V source. The module has been factory calibrated, but it is not accurate enough for applications that require high accuracy. In our project, we incorporate codes that could do calibration for further accuracy due to the fact that it is being used for navigation.

Technical Specifications (Source: Adafruit):

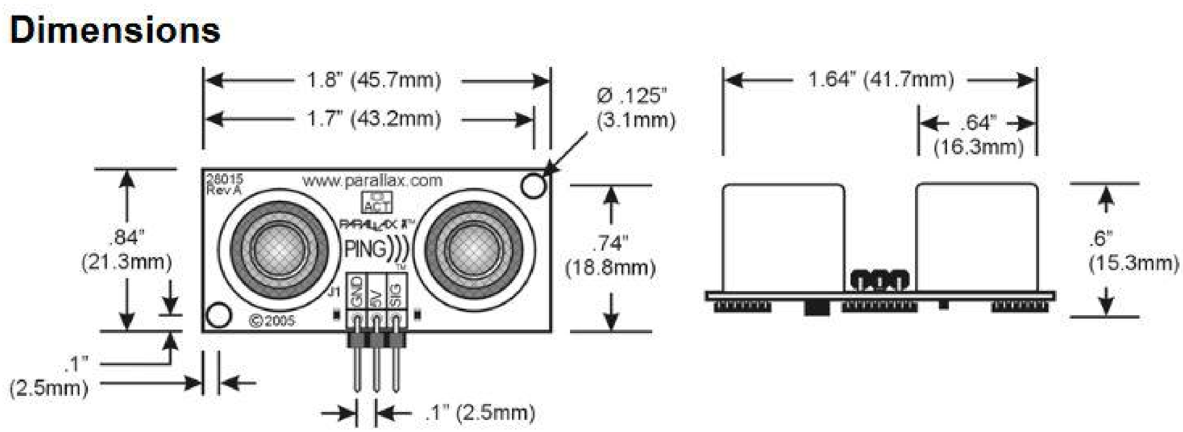
- 3 magnetic field channels and 3 acceleration channels
- From ±1.3 to ±8.1 gauss magnetic field full-scale
- ±2g/±4g/±8g/±16g selectable full-scale
- 16 bit data output
- I2C serial interface

- Analog supply voltage 2.16 V to 3.6 V
- Power-down mode/low-power mode
- 2 independent programmable interrupt generators for free-all and motion detection
- Embedded temperature sensor
- Embedded FIFO
- 6D/4D orientation detection
- ECOPACK RoHS and "Green" compliant

6.9 * *Adafruit LSM303 Magnetometer and accelerometer:*

Jiaming Zhou is responsible for this module.

Reference: https://www.parallax.com/sites/default/files/downloads/28015-PING-Sensor-Product-Guide-v2.0.pdf



This is an ultrasonic distance sensor that could accurately measure distance between 2cm to 3 meters. The sensor measures distance by sending an ultrasonic pulse and recording the time it takes for the pulse to return. By using the speed of sound and the time the pulse taken to reach back, the distance can then be calculated.

Technical Specifications (Source: parallax incorporation)

- Range: 2 cm to 3m (0.8 in to 3.3 yd)
- Burst indicator LED shows sensor activity
- Bidirectional TTL pulse interface on a single I/O pin can communicate with 5 V TTL or 3.3 V CMOS microcontrollers
- Input trigger: positive TTL pulse, 2 us min, 5 us typ.
- Echo pulse: positive TTL pulse, 115 us minimum to 18.5 ms maximum.
- RoHS Compliant
- Supply voltage: +5V DC
- Supply current: 30 mA typ; 35 mA max

- Communication: Positive TTL pulse
- Package: 3-pin SIP, 0.1” spacing (ground, power, signal)
- Operating temperature: 0-70 degree C
- Size: 22mm H x 46 mm W x 16 mm D (0.84 in x 1.8 in 0.6 in)
- Weight: 9g (0.32 oz)

6.10 * Elegoo Arduino Uno(Atmega328p):

Jiaming Zhou is responsible for this module.

Reference:
https://store.arduino.cc/usa/arduino-uno-rev3
https://www.arduino.cc/en/uploads/Main/arduino-uno-schematic.pdf

This is an all purpose microcontroller that has been widely used by both students and professionals. This is great for proving a concept or prototyping a system. Its development kit allows for easy coding and debugging. The chipset, atmega328p, is stronger enough for many applications. It has 14 programmable input and output pins. It can be powered by both USB connection and a 7-12 V source.

Technical Specifications:

- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Input Voltage 7-12 V (6 voltages minimum and up to 20 voltages)
- Digital I/O pins: 14
- PWM Digital I/O pins: 6
- Analog Input Pins:6
- DC Current per I/O pin: 20 mA
- DC Current for 3.3V pin: 50 mA
- Flash Memory: 32 KB (0.5 KB is used by bootloader)
- SRAM: 2 KB
- EEPROM: 1KB
- Clock Speed: 16 MHz
- LED_BUILTIN: 13
- Length: 68.6 mm
- Width: 53.4 mm
- Weight: 25 grams

7 * Technical Problem Solving

Document all the technical difficulties you encountered

7.1 Transceiver Power Consumption problem:

Identified by Jiaming Zhou

We had a problem supplying enough current to the transceiver for it to be working properly. At the early stage of the project, we powered the transceivers using the 3.3 voltage pin from the Arduino Uno board. However, we encountered a problem with it only being able to transmit at a very limited distance, and the transmitting rate was incredibly slow. After some researching and reading, we discovered that the module could consume up to 250 mA during operation, depending on the power setting. The 3.3-volt pin from the Arduino could only supply up to 20 mA.

7.1.2 Solving the transceiver power consumption problem:

The solution that we came up with was to use an external power source with a voltage regulator. The regulator made sure that the voltage being supplied to the module was within the proper range. By using an external power source, we solved the problem of limited current. This had taught us knowledge about power management.

7.2 Atmega 1284 floating problem:

Identified by Gerardo Hernandez

Initially we were using a custom circuit to supply four voltages to control the drone using an atmega1284 as the controller. Although it was able to fulfill the requirement, we later discovered that it was extremely difficult to enter floating values and get data from the RF Antenna using the atmega1284. The atmega1284 is an 8 bit microcontroller, it lacks sufficient support libraries. To enter floating points through a keypad connected to the microcontroller, it would have required us to program a custom library to enable this functionality. This could consume a lot of time, and it was not part of our project’s objective. Additionally, an eight bit microcontroller would introduce significant precision errors.

7.2.2 Solving the Atmega 1284 floating problem:

To solve this problem, we decided to use the Arduino mega as the main controller instead. The Arduino Mega board has libraries that allow us to incorporate the keypad for the user to enter floating point inputs, which are required for gps coordinates. This solution had also alleviated the precision error problem to a considerate degree. Although we still had trouble getting the seventh decimal place right, due to the limitation of the microcontroller, the error was small enough to be negligible. Through solving this, we had learned a substantial amount about how processors handle floating values.

7.3 Compass inaccurate output due to tilting:

Identified by Jiaming Zhou

A problem that we had discovered during testing was the compass would produce inaccurate values when the drone tilted, caused by mainly turbulence. The compass was basically a magnetometer, which relied heavily on the orientation and the magnitude of the magnetic field of the earth. To magnetometer was created to produce accurate values only when the module is parallel to the ground. By having this problem, any wind could cause the drone to fly to a direction that is incorrect. Although we had incorporated code that would redirect the drone when it realized that it was flying to a wrong direction, it was not very power efficient.

7.3.2 Solving compass inaccurate output due to tilting:

We were unable to alleviate the problem due to that fact that it was a limitation of the module. We had attempted to incorporate compensating algorithm using the accelerometer, but we were unsuccessful. The result was the algorithm seemed to have little effect on the outputs the module produced. The output values seemed to be the same regardless of whether or not the algorithm had been incorporated. This problem could be fixed in the future by purchasing a more expensive module that has support for tilt compensation.

7.4 Raspberry Pi connection problem:

Identified by Gerardo Hernandez

Part of project's objectives was to take images of targets and score them based on the position of the marks. A problem that we ran into was connecting the raspberry pi to a network through which we could send the image back for processing. Due to the fact that the connection it used was WIFI connection, having a mobile connection was an obstacle we had to solve. We initially thought that we could use the school's network, but we discovered that the process by which the Raspberry pi uses is only applicable to standard routers. The school's network has its own custom login system, which is not compatible with the Raspberry Pi.

7.4.2 Solving raspberry pi connection problem:

This problem was solved by using the cellular data network we had on our phone. We set up our phone as personal hotspot, and then we connected both the Pi and our computers to it. We were worried at the beginning that this solution would not work due to limited range. After some experiment, we discovered that Pi was able to send pictures even though the phone was at some substantial distance away. This problem had taught us networking knowledge.

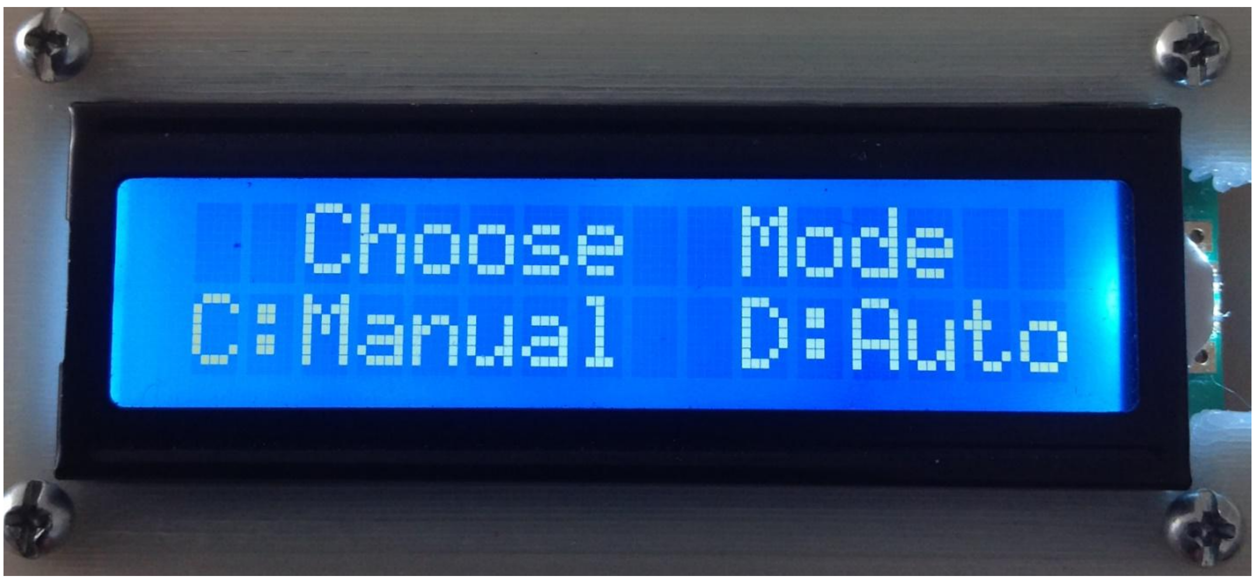
8 User Interface Design

The user interface is the Base Station. The base Station has 6 LED indicators, a 2x16 LCD display, and a 4x4 Keypad. The user interface is designed such that the user can choose manual mode or autonomous. If the drone is in autonomous mode, the user can go into manual mode at any moment and take full control of the drone.

8.1 Application Control

The application of the user interface if manual mode is chosen is such that the 4x4 Keypad mimics the actions that the drone's controller has. If the user chooses automatic mode a series of displays will first ask how many targets the user wants to visit, what the latitude and longitude of each target is, and whether the user wants to start autonomous mode.

8.2 User Interface Screens



This screen asks the user whether Manual or Autonomous mode will be selected.

This screen is the default screen displayed in Manual Mode

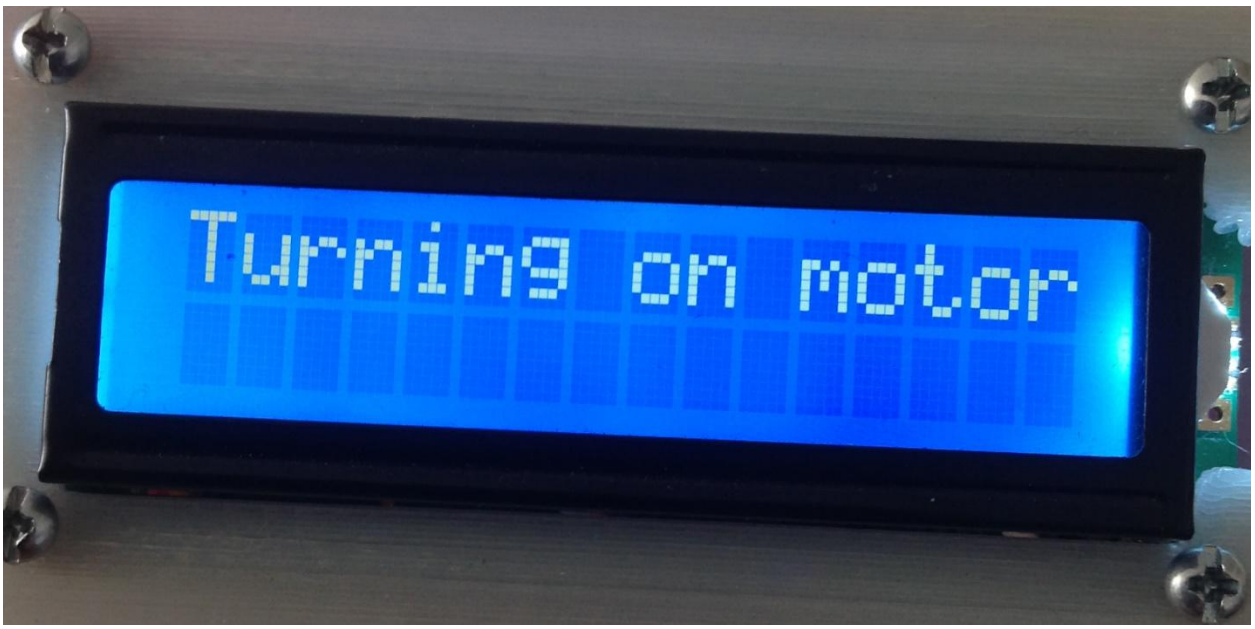
This screen is where the user enters the number of targets to visit.

This screen instructs the user on the functionality of entering coordinates

This is the screen where the user enters the latitude

This is the screen where the user enters the longitude

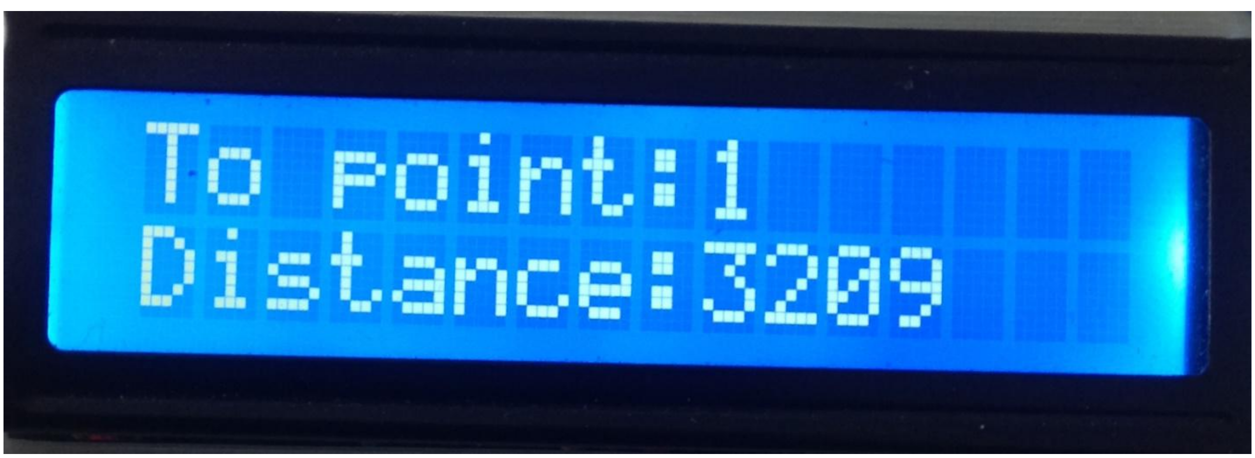
This is the screen where the user chooses to begin autonomous mode



This is the screen which alerts the user that the drone is turning on the motors

This is the screen which alerts the user that the drone ascending

This screen alerts the user that the drone is calibrating its compass



This screen shows the distance remaining (in meters) to get to the target
(This was a dummy coordinate. The system doesn't have the ability to travel more than 150 meters in one run due to limitation of the telemetry system.

9 * Test Plan

9.1 * Test Design

Test 1: Buffer circuit: (Experimenter: Gerardo Hernandez)

Objective:

This test is designed to check the functionality of the buffer circuit we built. The purpose of the buffer circuit is to produce four constant voltage outputs, which are created to imitate the output of the joysticks from the transmitter. It measures the accuracy and the stability of the circuit.

Setup:

1. Buffer circuit
2. Microcontroller
3. Power supply (5V)
4. Oscilloscope
5. Laptop

Procedure:

1. Connect the inputs of the buffer circuit to four PWM ports of a microcontroller
2. Connect the power supply to the corresponding input of the buffer circuit (op-Amp)
3. Attach the oscilloscope to the output of the buffer circuit. (repeat it for the other three outputs)
4. Record the reading from the oscilloscope and compare it to the desired output
5. Check to see if there is any ripples. (preferably none).
6. Repeat it 4 more times.

Expected Result:

The result of the experiment should show that the buffer circuit outputs four constant voltages. The actual voltages depend on the input. Because of the design, there should not be any spike or surges of voltage.

Test 2: Design of the power manage system (Experimenter: Jiaming Zhou)

Objective:

The purpose of this test is to determine whether or not the power management system is working properly. The power management system is created to supply proper power to each module and circuit from a fix source(battery).

Setup:

1. Battery
2. Power management circuits (voltage regulator and buck converter)

3. Multimeter

Procedure:
1. Connect the battery to the power management system.
2. Attach the sensor of the multimeter to each critical node
3. Check the voltages of those nodes.
4. Repeat it 4 more times.

Expected Result:

The expected output of the power system should be voltages and currents that fulfill the operating power rating of the modules being used.

Test 3: Testing the autonomous flight (Jiaming & Gerardo)

This test is performed to test the autonomous capability of the project. It tests to see whether or not the programmed written is able to take a predefined coordinate and direct the drone to the locations. Note: This test is to be performed multiple times at different dates.

Setup:
1. Drone
2. Base station
3. Charged batteries

Procedures:
1. Properly set up both the on-board system and the base station
2. Follow the user interface instructions to enter the coordinates
3. Through observation, check to see if the drone is flying to its destinations and how accurate is the landing.

Expected Result:

For the first few initial tests, we expect bugs and behaviors that will need corrective actions. However, no damaging bugs or error should be happening. After improvements are made, we expect the performance should fulfill the project objective. We might still need to finetune the system to achieve higher accuracy.

Test 4: Testing the target scoring system (Jiaming & Gerardo)

This test is aimed to test the accuracy and the reliability of the image processing system. It tests how often does it produce a valid score under varying conditions. These conditions could be tilting due to wind and target offset caused by inaccurate gps positioning.

Setup:
1. Ready both the drone and its on-board system

2. Base station
3. Raspberry Pi (ready to take and send back images)

Procedure:
1. Lift the drone to the height where the drone normally operates at during operation
2. Navigate the drone to the target, so that it is hovering above the target.
3. Take and send back an image of the target
4. Process it through the program.

Expected result:

Depending on how accurate the drone is able to navigate to the targets, the camera may or may not be able to capture an image with the target in it. But this is not an issue of the image processing program, but more of a navigation problem. If an image is captured, the program will produce a score.

9.2 * Bug Tracking

Bugs and errors were tracked and logged as they were being discovered during testing. Depending on where the bugs occurred, the corrective task was assigned to the member who was handling the part.

9.3 * Quality Control

We made sure all the required tests were performed to test the functionalities of our system. Results were recorded to post-analysis and to make improvement if any bugs were spotted. To ensure consistent results, we conducted most of our tests in control settings, where bugs caused by unexpected factors was reduced to minimal. Corrective actions made to alleviate or rectify certain problems were documented to keep track of changes that had been made.

9.4 * Identification of critical components

The battery of the on-board system needs to be paid with special attention. There were numerous occasions where the drone would drop suddenly. Later we discovered that it was caused by the battery dropping below a certain voltage. Before every flight, the experimenter needs to make sure the battery is fully charged to avoid accidents.

9.5 * Items Not Tested by the Experiments

Every module that is used on the system had been tested.

10 * Test Report

Test 1: Buffer circuit

Experimenter: Gerardo Hernandez
Iterations: 5 times

1. Test results:

•

Trials #	Output 1 (volts)	Output 2 (volts)	Output 3 (volts)	Output 4 (volts)
1	1.51	1.51	1.61	1.61
2	1.52	1.51	1.60	1.62
3	1.51	1.52	1.61	1.61
4	1.52	1.52	1.61	1.61
5	1.51	1.51	1.61	1.61

•

2. Comparison with expected results:
   - The results we have gotten are similar to our expectation. The voltages produced are fairly accurate and stable. The voltages change as supposed to when the PWM sources change.
3. Analysis of Test results:
   - The results of each iteration are similar. We did not observe much variation from the outputs.
4. Corrective actions taken:
   - No corrective actions have been taken due to the results being fairly accurate.

Test 2 Design of the power manage system:

Experimenter: Jiaming Zhou

1. Test results:

•

Trail	Output 1: (Volts)	Output 2: (Volts)
1	4.89	8.12

2	4.86	8.10
3	4.88	8.10
4	4.89	8.10
5	4.89	8.10

2. Comparison with expected results:
   - Although the voltages for output are not 5V, due to voltage drop inside the regulator, the results are pretty close to what we expected.

3. Analysis of Test results:
   - The test result that we have gotten is pretty good. As it has been mentioned, the output voltages for output 1 are not fully 5 V, but it should not affect the operation of the whole system.

4. Corrective actions taken:
   - No corrective actions could be taken due to the voltage drop inside the circuit is inevitable.

Test3: Testing the autonomous fly:

Experimenter: Jiaming Zhou & Gerardo Hernandez

1. Test results:
   - Day 1(Feb 21, 2018): The drone was able to autonomously fly to a single target. Returning home and landing autonomously were still having trouble.

   - Day 2 (Feb 24, 2018): The drone was again able to autonomously fly to a single target. We were able to get better results with returning home and landing autonomously.

   - Day 3 (Feb 28, 2018): Landing autonomously was nearly flawless. We made multiple attempts, and the drone was able to land flawless every time. Today the drone was able to fly autonomously to two locations in a single flight. However, sometimes the drone would miss either one or both targets and needed to reorient. Additionally, the drone did not hover over the target one-hundred percent of the time.

   - Day 4 (March 3 2018): We were able to achieve three targets this time. However, missing target would still occur from time to time. We discovered that it was due to the compass. Tilting due to turbulence could cause the compass to produce inaccurate values. Hovering 2-3 meters away from the targets was still a problem.

2. Comparison with expected results:

The results we had gotten were expected. We expected bugs that needed to be fixed. However, some of the bugs were not expected such as missing the target due to incorrect values produced by the compass caused by turbulence. The process also took longer than we had originally planned. Having to go out into the field to test our system had limited us the number of times we could test reasonably.

3. Analysis of Test results:

The results provided some constructive feedback by which we used to improve the system. It shows us problems that we did not fully anticipate at the beginning.

4. Corrective actions taken:

Numerous actions were taken to correct the errors and bugs we encountered during testing. An example was that we had to manipulate the speed at which the drone landed slower. Landing safely was difficult due to its internal control system. If the drone was to sense any abnormal behaviors during the landing phase, it could cause it to correct itself and tilt, causing the drone to crash or break its propellers. We had to make it land slower to ensure stability. We had also adjusted the distance where the drone would go from landing to shutting off. Depending on the weather of the day, the drone might not stabilize at the same distance every time. WE had to choose a distance that gives the drone the most buffer.

The problem of the compass producing inaccurate values due to tilting was hard to alleviate. We had attempted to incorporate codes that could compensate for the tilt using accelerometer data. However, the implementation was unsuccessful. The result was that the algorithm did not seem to have any effect on the outputs. Regardless of whether or not the codes were incorporated, the outputs were the same. We were not sure whether it was caused by the algorithm itself or the equations that were used to formulate the algorithm. This problem remained unsolved.

Test4: Testing the target scoring system:

Experimenter: Jiaming Zhou & Gerardo Hernandez

1. Test results:
   - Day 1 (March 10, 2018): The result was not very desired. Due to the inaccuracy of the GPS (precision < 3 meters), sometimes the camera was not able to capture any images with the target in it. The images were also suffering from severe distortion.

   - Day 2 (March 11, 2018): The images captured was free from distortion after we decided to lift the drone higher, and the number of times the camera was able to capture images with the targets in them was also increased. However, for some reason the problem was not able to detect the circles of the target.

- Day (March 14, 2018): We were to able capture and score the targets at a decent rate.

- Day (Match 16 2018): We were able to increase even further the number of times images with targets in them being captured.

2. Comparison with expected results:
    - As been anticipated, the accuracy of the navigation system played a significant role in determining whether or not images with targets in them were able to be captured. We however did not expect the wind generated from the drone would cause distortion in the images.
4. Corrective actions taken:
    - To increase the chance of the camera being able to capture images with targets in them, we increased the height at which the drone operates. This allowed the camera to have a wider capturing frame. We had also made adjustment to the code to solve the problem of the problem not being able to detect the circles of the target. What we discovered was that the parameters which determine what circles to look for was set incorrectly.

11 * Conclusion and Future Work

11.1 * Conclusion

Jiaming Zhou:

Through hard work, we have achieved a significant portion of the design objectives. The project can be broken down into two main objectives: autonomous navigation and target scoring. While the navigation system is not perfect, it can safely navigate the drone to its destinations autonomously. There are a few areas where it can be improved. The IMU system, for example, can be improved to increase the reliability and the accuracy of the system. However, in order to achieve this, either more time or funding is required. The software can also be improved to increase computational efficient.

I have learned a lot through the course of this project. In terms of technical knowledge, I have gained better coding skills and many physical experiences. Courses that I have taken contributed to my theoretical knowledge, yet very few have contributed any hands-on experiences. Additionally, it has also improved my engineering skills in the sense that I could approach and tackle engineering problem much better than before. I have gained better understanding of how to read datasheets and solve technical problems. In terms of personal growth, I learned that time management is crucial to complete a project with quality. Senior design has consumed a large portion of my time in both quarters. It was hard to balance other class works, especially when they were challenging, and senior design. By having better time management, I could have gotten my work done in a more effective and efficient manner.

This project has also helped me learn how to work in a team better. In the past, I had spent most of my time working alone. I enjoyed the freedom to make every decision and be able to take action without having to first consult with any teammates. For small project, this is an effective method for it could progress faster. However, for bigger projects like senior design, having teammates is very helpful in the sense that they could contribute knowledge and skills that I do not necessarily have. Working with another person has contributed to my communication skills. Not only did I have to speak more often, but I also needed to make sure the message that I was trying to convey was as clear as possible. In the process, we had also had a lot of fun and grew new knowledge together.

Overall, I consider this to be a positive experience where I was able to increase my knowledge and personal skills. I wish we had more time, resource, and an extra member. Even though we had three people initially, one left at the beginning of the first quarter without any notice. This had definitely made the project a lot tougher since we had to cover the responsibility that was originally his. On the bright side, this had prompted us to do better and work under pressure instead of within our conform zone.

Gerardo Hernandez:

The conclusion of our project is such that we are able to autonomously have a drone safely fly to multiple destinations. Due to the limitations on the GPS and compass modules, the drone travels to its destination within 3 meters. When the drone is 1 meter or less away from its destination we are able to take an image of the target and score it correctly. We have built a system that is user friendly, portable, and most importantly built a system that is SAFE. Safety was a major concern in this project. We made sure that at any moment then user could take over the autonomy of the drone and fly it manually. Jiaming is mostly responsible for the completion of everything onboard the drone and I am mostly responsible for the base station, and completely responsible for all of the 3D design/printing and computer vision.

I learned how to be systematic about completing objectives. I learned how to work around the limitation of my 5”x5”x6” 3d printer. I 3d design the parts such that every cm had a purpose and did not occupy space just for aesthetics. I had to make sure that certain components were as far apart from each other, facing in specific directions, and easily interchangeable. I learned how to make use of my personal components in order to save time and money. I learned how a buck converter works and why it’s preferred over other step down voltage sources. I learned how to use Matlab’s computer vision toolbox. I learned how to make a set of voltage sources from very basic components such as multiplexers, adjustable voltage regulators, voltage dividers, and the Atmega1284. Building the state machine for this project was one of the most consuming tasks I underwent. I learned that no matter how complex your system may be, having the ability to create a state machine is crucial to allowing ideas turn into reality.

The skills that I gained throughout this project consist of improving my soldering skills. I learned different knots to implement when soldering. One of the knots is actually used in NASA.

Professionally I learned that every choice I made for the team and even in my personal life would indirectly affect the team. I learned that having good team members makes the biggest difference in the results one sees. As mentioned prior, Emmanuel Martinez left the team without notice 3 weeks after the Winter quarter. This heavily affected the team and it made me realize how important it is to have a reliable team to work with. Luckily Jiaming is an extremely hard working individual. If it weren't for Jiaming’s hard work and dedication, this project would have not been completed. Professionally, one must take accountability, one of the core values UCR holds.

This project has given me more confidence in knowing that I could accomplish any task that’s handed to me. With limited time and budget, I feel that the prototype we produced was accomplished with professional expectations. I feel that I’ve matured professionally in giving presentations, creating content based on our project, and feeling passionate towards a project. Having the ability to spending countless hours with someone from a different background and enjoying the time spent with that person even though the conditions were stressful, make me realize that I have what it takes to work in industry with all walks of life. Personally I’ve learned that humor is man’s best friend.

11.2 Future Work

The system that we have developed could be improved in many areas. One of them is to integrate a better IMU system. The current IMU(compass) is lacking in flexibility in terms of the conditions it could operate in. It does not have tilt compensation. Whenever there is turbulence, the reading from the compass become inaccurate, rendering the navigation to be faulty.

The system needs to have a close loop embedded into it. We need to make the drone adjust itself on its way towards the destination and not only after it has passed its destination.

The system needs to take images such that the raspberry pi runs in a standalone fashion. One solution is to have the raspberry pi use 4 of its GPIO pins as outputs and have the Arduino Uno take those outputs as flags. A value from 0-15 could be sent by the raspberry pi to the Arduino Uno to specify the score. The score should then be sent out by the onboard transceiver to the base station. The base station should have an array of variables to store each of the scores, per destination.

Physical improvements would pertain to weight on the drone. Having a lighter drone would allow the drone to reach more destinations, fly for longer durations, and have a larger field of view.

11.3 * Acknowledgement

We would like to acknowledge Dr. Liang Ping, Dr. Maaref, and Ms. Kimia for their support and guidance. We would also like to thank Dr. Kassas for lending us the drone, and Dr. Roman, Chomko for his support. We want to give our special thanks to our families and friends.

12 * References

Raspberry pi camera
https://www.adafruit.com/product/3099?gclid=EAIaIQobChMIkuGDzPf_1glVlIF-Ch2_zA5oEAYYAyABEgKQtvD_BwE

Install the MATLAB Support Package for Raspberry Pi
https://www.youtube.com/watch?v=Z9V4Xcxlc9Q

MATLAB
https://www.mathworks.com/videos/using-matlab-with-a-raspberry-pi-camera-board-94192.html

Files used to print 3D housing for the raspberry pi can be found here:
https://www.thingiverse.com/thing:396858

Working with Raspberry Pi Hardware
https://www.mathworks.com/help/supportpkg/raspberrypiio/examples/working-with-raspberry-pi-hardware.html

Tutorial: Reading GPS data via Arduino
https://www.youtube.com/watch?v=v74E01ESdUA

Drone Controller Hacking Using Arduino
http://jireren.github.io/blog/2016/11/25/how-to-hack-a-drone-controller/

GPS - NMEA sentence information
http://aprs.gids.nl/nmea/

Image Category Classification Using Deep Learning
https://www.mathworks.com/examples/matlab-computer-vision/mw/vision-ex77068225-image-category-classification-using-deep-learning

Calculating the score using matlab
https://stackoverflow.com/questions/33709384/calculating-the-scores-using-matlab

Detecting rectangles in matlab
https://www.mathworks.com/matlabcentral/answers/35243-detecting-rectangle-shape-in-an-image

Edge Detection with MATLAB
https://www.mathworks.com/videos/edge-detection-with-matlab-119353.html

MATLAB Coder Target for Raspberry Pi
https://www.mathworks.com/videos/matlab-coder-target-for-raspberry-pi-1493068888613.html

Collect and Analyze Data Using MATLAB and Raspberry Pi

https://www.mathworks.com/videos/collect-and-analyze-data-using-matlab-and-raspberry-pi-92959.html

Image Category Classification Using Deep Learning
https://www.mathworks.com/examples/matlab-computer-vision/mw/vision-ex77068225-image-category-classification-using-deep-learning

Image detection
https://www.mathworks.com/help/images/examples/detect-and-measure-circular-objects-in-an-image.html

13 * Appendices

*** Appendix A: Hardware**

Microcontrollers:
Arduino Uno
Arduino Mega
Raspberry Pi B+ Plus 5MP Camera Module
Interface Components:
2x16 LCD Display
4x4 Keypad
Sensors:
Ping Sonar Sensor
Ultimate Breakout GPS
LSM303 Accelerometer + Magnetometer
Communication:
NRF24L01 Wireless Transceiver Mod*ule*
Materials
PLA at 1.75mm diameter

*** Appendix B:** Equipment List

- NO.APS-274K Power Supply
- FTIKE A830L Multimeter
- BK Precision 40Mhz Oscilloscope Model: 1541A
- Bukito 3D Printer

*** Appendix C:** Software List

1. Arduino Sketch:
https://www.arduino.cc/en/Tutorial/Sketch
2. Solid Works:
http://www.solidworks.com/
3. MATLAB:
https://www.mathworks.com/products/matlab.html

Appendix E: User's Manual:

1. Connect the drone's potentiometer inputs and ground to the perforated board.
2. Connect the 5V USB cable to the input of the Arduino Mega.
3. Connect the 7.4V battery to the voltage regulator on the drone.
4. Turn on the drone.
5. Turn on the drone controller.
6. Calibrate the drone.
7. Choose whether the drone will be operated in Manual or Autonomous mode.
8. If Manual Mode is chosen, turn on the drone's motors by pressing 'A'

9. If Manual Mode is chosen, follow the direction on the LCD display.
10. After Landing the drone, disconnected all batteries.