Autonomous Search & Rescue Drone

UCF Senior Design Group #40

A collaboration between *Moises Cortes Lugo*, *Lisa Harrison*, *Christian Lozano*, *Joseph Manalo*, *Joshua Mutugi* & *Nicholas Norman*.

Table of Contents

1. Executive Summary

1.1. Report Purpose

This document describes, in detail, the initial design requirements, specifications, and milestones for our project. As a disclaimer, we are still in the beginning stages of planning and development; as such, there is detail contained herein that we may change in the future.

1.2. Relevant Background Information

1.2.1. Problem to be Addressed

With an extensive history in civilian and military applications, Combat Search and Rescue missions have been critical in locating military as well as civilian personnel. These rescue missions can occur on land and at sea, including various levels of complexities that can limit their effectiveness. Some of these complexities include hostile insurgents, environmental factors and human limitations in searching, which can drastically decrease the chances of finding the victim in a reasonable amount of time.

Unmanned Vehicles (UVs) can greatly help reduce these inhibitions with a more robust, deployable solution. Aerial drones, for example, can cover more land than a human team, implementing better optics and reducing human interaction with potential threats. UVs can carry cargo and access difficult areas that humans cannot safely access. All of these, combined with the facts that drones remove human lives from being put at risk when locating victims. After a victim is located, first aid can be directed to the location with an extraction team and provide necessary aid.

1.2.2. Common Unmanned Vehicle (UV) Trade Study

Many Search and Rescue organizations are implementing autonomous vehicles to aid in locating lost victims. Several of these Unmanned Vehicles (UVs) will be considered, as well as their advantages and disadvantages. A final choice will be made based on mission requirements, with the UV that is best equipped for the mission being selected. The four UVs that will be considered are: Tracked vehicles, Four-wheel-drive vehicles, multirotor aircraft and fixed-wing aircraft. The most important criteria to be considered are: (1) time to complete the mission, (2) visibility of target, (3) robustness (likelihood of malfunction during mission), (4) payload and (5) navigation capabilities.

Figure 1.2.2.A: Dragon Runner Tracked Vehicle Drone

Tracked Vehicle

Starting with tracked vehicles, these are the most popular options for a variety of reasons. Used frequently in bomb disposal missions, these robots are robust and can travel across virtually any ground terrain. They are recognized as having the simplest mechanical system, consisting of two independently controlled motors driving two or more wheels that pull a continuous band of treads. This design typically incorporates a rotating turret mounted on top, with a high caliber gun, as well as other missiles and machine guns. Drone versions, such as the Dragon Runner IED detector pictured above, utilize this turret space to mount cameras and sensors on a gimbal, with extendable arms that can achieve a better view of the target, or use a hand-like robotic device to grasp an object, in this case a bomb. These designs are controlled by handheld controllers and require active user input to control the turret.

The benefits of this design are improved weight distribution than wheeled vehicles on the ground, increased traction in soft ground, and better maneuverability with ability to turn 360 degrees in place. Tracked vehicles rarely are prohibited by terrain obstacles and perform well in muddy or otherwise wet conditions. Another thing to consider in this type of UV design is its large payload capacity. Due to its land navigation design and improved weight distribution, it can carry the largest amount of onboard weight out of all the four UV options. The cons to this design are having the lowest top speed (51mph, FV101 Scorpion), in full-sized versions, as well as inability to move if a tank tread is damaged. Another con is controllability. If a search and rescue tank with movable turret is to be implemented autonomously, the levels of complexity drastically increase and this paired with the speed of the drone, may prove to be infeasible in searching larger areas.

4 Wheel-Drive Vehicle

Figure 1.2.2.B: Honda Self-Driving Four-Wheeler

Wheeled vehicles on the other hand, namely 4WD variations, use four driven wheels with independent suspension to navigate various terrains. These wheels generally are equipped with large, knobby tires to increase traction in rural areas. Similar to the tank system, these vehicles employ a motor system to the wheels to drive the vehicle forwards and in reverse. An increase in level of complexity with these UVs is a steering system, which has a limited turn radius and more components to account for.

Benefits of this design are that they are simple to repair and more fuel efficient and quieter than tracked vehicles. Their top speed is also much higher, giving them more capability to cover larger areas for search and rescue. However, they do not navigate terrain as well as a tracked vehicle, and if one is stuck, the whole vehicle is unable to free itself and escape.

Fixed-Wing UAV

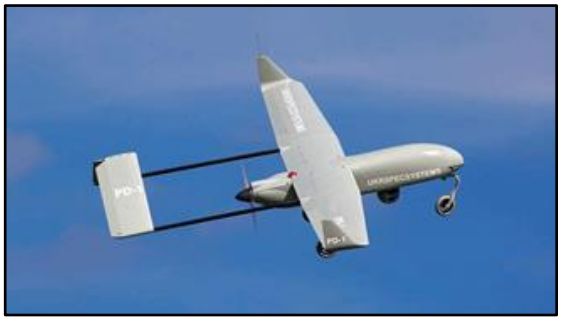


Figure 1.2.2.C: Nigerian Air Force PD-1 UAV

Fixed wing aircraft are the first aerial option to be considered, and offer a simple, robust system to search from victims in the air. This design consists of an aircraft with fixed wings and a separate propulsion system, either a turbine (jets), or a motor and propeller, which is most common. These drones need extra space to take off and land, and rely on forward propulsion in order to maintain stability in the air.

Benefits of this design is the field of vision, which is superior to ground-based units. Also, very few obstacles can re-route a search path in a given search area. This is because these drones typically fly high above the landscape, providing the widest field of vision. The range on these drones is the best out of all four options, with handheld radio connectivity for miles, and if connected with satellites, virtually endless connectivity and ability to cover the most area. Their flight endurance can be upwards of an hour of continuous flight, although they must continue moving forward.

Drawbacks to this design start with visualization in a dense landscape. As these UVs fly over the landscape, trees and other natural vegetation can easily block the field of view. These drones also need a higher speed to stay in motion, and unlike the other three designs, cannot maintain a steady visual on the target, but can only make multiple passes in the same area. Overall, they provide superior, but limited target visualization and navigation capabilities.

Multirotor UAV

Figure 1.2.2.D: Magni Military Quadcopter by Elbit

Multirotor drones are a more versatile aerial option, and use multiple motors with propellers to vertically take off and hover, maneuvering through aircraft pitch and varying speeds in each motor. This design uses PID controllers to constantly monitor and adjust motor speeds to maintain stable orientation of the aircraft. This is the most complex design, using a flight controller with advanced stabilization technology to control equilibrium as well as six degrees of motion.

Advantages of this design are superior mobility and visualization. This UAV is not inhibited by any landscape, and is able to fly between trees, in caves, over marshes and in deserts. These drones are adept for maneuvering in small spaces, implement vertical take-offs and landings, as well as hover in place. The power of these drones is comparatively high, as some are capable of reaching up to 163mph (DRL RacerX). As with the fixed-wing, this UAV provides a superior viewing angle as well to ground based vehicles.

Drawbacks in this design are that these drones are incredibly powerful, which in the event of a crash, large forces will be present, and a smaller likelihood that the drone will survive as compared to the other designs (FPV quadcopters are known for exploding on impact). Another drawback is the relatively short flight time. Typical hobby-level quadcopters can only remain airborne for 13 minutes, but larger, military-grade multirotor UAVs can fly up to 70 minutes with a payload (Lockheed Martin Indago 3).

Comparison

Here is a ranking of the four UV vehicles, compared with the five respective criteria. In each criterion, each vehicle will be assigned a value from 1 – 4, with 1 meaning that this vehicle is least adequate for the criteria, and 4 meaning that it is the best equipped for the given criteria. The vehicles will be compared amongst each other, so no value will be repeated in a given criterion. Here are the results:

Criterion	Tracked	Four-Wheeled	Fixed Wing	Multi-Rotor
(1) Speed	1	2	4	3
(2) Target Visualization	2	1	3	4
(3) Robustness	3	2	4	1
(4) Payload	4	3	2	1
(5) Navigation	2	1	3	4

Table 1.2.2.A: The results show that for the majority of the criteria, aerial drones are preferred

Speed: Fixed wing UAVs can cover the largest amount of area in the shortest amount of time with a higher top speed than any land-based vehicle.

Visualization: UAVs also offer the best visualization of the target with an aerial view, with the multi-rotor being the top choice due to its ability to hover in place.

Robustness: The robustness criteria, measured primarily by the UVs ability to resist failure in a mission, primarily goes to the fixed wing which rarely contacts its environment, flying above it. This vehicle is least likely to undergo failure in a mission. In a stricter sense, the tank is the most robust design, armored to be able to defend against attacks and an adverse environment.

Payload: This criterion greatly favors land-based vehicles, as fuel economy is not as strongly affected by an increase in payload, as is the case with aerial vehicles. This primarily goes to the tracked vehicle, which distributes its weight more efficiently. Aerial vehicles are greatly inhibited by carrying extra loads, as they continuously fight a gravitational force while in the air

Navigation: Finally, the navigation criterion favors an aerial design, which travels over the land, not hindered by water, rocky terrain, cliffs, or any other landscape that would prove otherwise impossible for land-based vehicles. The multirotor has an advantage over the fixed wing, which can only fly above these landscapes. Unlike the fixed wing, the multi-rotor vehicle can fly through them, such as in forests or caves, still being able to hover in place where a land-based vehicle cannot reach and a fixed wing cannot attain visualization.

Conclusion

Considering these five criteria and the nature of this mission (location of a victim in an area less than 22,000 feet squared), multi-rotor UVs are the most equipped. Their speed and agility, as well as superior mechanical design allow them to navigate any terrain at varying speeds and heights. They can navigate through smaller, less accessible places, and take off and land vertically in the chance of an emergency landing. The most important factor, target visualization, is optimized in this design that can provide an aerial view of a landscape and hover in place, offering more time for object detection.

1.3. Solution & Approach

Our project aims to reduce the human manpower and exposure necessary for SAR by automating the process of human detection and location, which – in SAR – is half the battle. To this end, we propose building an autonomous aerial drone (specifically, a quadcopter) that flies within a search area of 22,000 ft^2 at a height of 17 ft. above the ground, and can – with real-time camera footage – locate a human that is standing, sitting, or lying down, and report that human's location.

To build the Autonomous SAR Drone, our team is composed of six members, each with a unique skill set and area of work, as listed below:

- **Moises Cortes Lugo:** Integration/Telemetry
- **Lisa Harrison:** GPS Tracking/Location
- **Christian Lozano:** Computer Vision/User Interface
- **Joseph Manalo:** Mechanical/Thrust/Power System Design
- **Joshua Mutugi:** Computer Vision
- **Nicholas Norman:** Mechanical Hardware Design

Acting as guides and consultants are Dr. Kurt Stresau – a prominent UCF faculty member and coordinator of its Mechanical Engineering senior design projects – and Chase Burchett – a Sr. level engineer at Lockheed Martin whose primary role is product design and development.

Our mission is to provide emergency and military personnel an autonomous solution to solving a big problem within SAR (human detection and location), and our hope is that it helps to reduce the human risk and danger that exists within this field.

2. Project Overview

2.1. Project Goals, Objectives, & Functions

The end result of this project is a drone that can be given a search perimeter via a user interface, and then autonomously locate and search the given perimeter to find a human located there. Upon discovery of the human, the drone will then land and transmit its GPS coordinates back to the user interface to facilitate rescue. The initial concept for the user interface is a simple web-based input structure that takes in the coordinates of the corners of the area to be searched, which are then sent to the drone for development of a route to the search area from the launch area, and then the execution of the search pattern within the perimeter.

In order to make this happen, the drone will use GPS and path planning algorithms to locate the search perimeter and enact its gridded search function within the perimeter, and will use a trained computer vision algorithm to interpret video footage gathered from its onboard camera, making sure to detect any and all humans within the frame. The user interface is currently envisioned as a simple web-based input system, which relies on the user to know the exact coordinates of each corner of the search perimeter. This can potentially be made more robust by the introduction of a more map-based interface, which would allow the user to highlight an area on the map to use as the search perimeter, rather than having to input specific coordinates.

The function of this drone is to facilitate first responder and military operations. These organizations are frequently called upon to conduct search and rescue operations in difficult and hostile environments, whether because of terrain, weather, or enemy action. The ability to locate a victim without having to put boots on the ground in these situations makes the likelihood of finding the victim without putting responders at risk far more feasible than those responders were forced to manually search the area. Furthermore, given the modular nature of the drone, it is possible that each organization will be able to fit their drone with the best sensors and cameras for their needs, and still rely on the same navigation algorithms developed for the original drone project.

2.2. Broader Implications

Once a drone is developed that can effectively execute the original mission (locate a human using computer vision), a lot of that technology can be applied to related applications. In addition to using computer vision, new models could use thermal or infrared imaging to detect heat-emitting objects in less-than-ideal visual conditions, such as inclement weather, or when part or all of the victim is obscured by foliage, debris or other environmental complications. Furthermore, in military applications especially, the victim being sought may be missing one or

more limbs, or otherwise not conform to the algorithm that has been taught to the computer vision program to identify a human. In these cases, having additional sensors can increase the chance of a successful rescue.

Again, the same technology that is used to find a person can also be used to deliver necessary items to the victim, in the event that help cannot get there immediately. By increasing the payload of the drone, it could be used to deliver water, medicine, or emergency blankets and other supplies to sustain the victim until help can arrive.

The combination of an automated drone with computer vision can be adapted to apply to a much broader range of objectives and tasks as well. A drone can be used for inspection, to be flown alongside buildings or structures to find weaknesses or points of interest on the structure using computer vision. Likewise, the drone can fly along railways, powerlines, or roads to find damage. A few of these drones can be deployed for surveillance purposes, to patrol borders and report intruders. The drone could be used to find and follow a target for action filming or sports. Delivery is also an option, especially when there is no set delivery site and the drone must make assessments on where to land or drop off the payload. This system can also be used for agricultural purposes to assess crops or track livestock.

2.3. Legal, Ethical, & Privacy Issues

2.3.1. Rules & Registrations

Currently these are the rules we must abide by for this project are listed below:

1. The drone must be registered by the FAA, and properly labeled.
2. Fly below 400 feet above the ground.
3. Obtain authorization if in controlled airspace.
4. Keep the drone within visual line of sight of the pilot or of someone who is in direct communication with the pilot.
5. Do not fly at night unless the drone is properly lit to show its location and orientation.
6. Do not interfere with any manned aircraft.
7. Never fly over people or moving vehicles
8. Do not interfere with emergency response activities such as law enforcement or accident response teams.
9. Never fly under the influence of drugs or alcohol including some over the counter medications
10. Do not operate your drone in a careless or reckless manner.

Violating these rules could lead to criminal and or civil penalties.

As part of our project, we should not have any issues with any of the rules stated. Once we have a prototype built, we will be registering the drone. Drone registration requires an FAA account and 5 dollars per UAS registered. We will most likely use Joseph Manalo's FAA account and split the fee. If we want to test the drone at night, we will need to add in LEDs which should not be a major setback. We do not plan on flying in high traffic areas, or anywhere above 100 feet or far enough away to be outside of visual line of sight.

Joseph Manalo has experience with drone operation and the laws surrounding flying them, so they will make sure all laws will be followed. They also have AMA certification which includes an insurance plan, in case there is any unexpected damage done to property. Testing will be done at certified AMA fields, or at remote areas on UCF campus with proper permission acquired.

2.3.2. Legal Restrictions for Further Development

If this project is to be developed further, there will be many new issues that would need to be solved. The operator would have to obtain a part 107, especially if this is to be used commercially or by government/military organizations. To obtain the certification, the operator must:

1. Be at least 16 years old.
2. Be able to read, write, speak, and understand English.
3. Be in a physical and mental condition to safely fly a UAS.
4. Study for the Knowledge Test
5. Obtain an FAA Tracking Number by creating an Integrated Airman Certification and Rating Application.
6. Take and pass the Knowledge Test at an FAA approved Knowledge Testing Center
7. Complete FAA Form 8710-13 for a remote pilot certificate

After the pilot is certified, they may have to then apply for waivers that will allow them to bypass rules that are not included under the Part 107 certification. The following rules may need to be waived during development:

- Operation from a moving vehicle or aircraft (In cases where the operational team will be travelling by vehicle while the drone is operating)
- Operation outside of visual line of sight (In cases where the drone will be developed to travel further distances)

- Operation of multiple small drone systems (In the case that the project is developed to allow many drones to be operated by a single operator)
- Operation in controlled airspace (In cases where the mission for the drone is in controlled airspace)

In some special cases, the waiver process can be expedited. Organizations that are considered for this expedited process include:

- Fire Fighting
- Search and Rescue
- Law Enforcement
- Utility or Other Critical Infrastructure Restoration
- Damage Assessments Supporting Disaster Recovery Related Insurance Claims
- Media Coverage Providing Crucial Information to the Public

The team working on the project in the future will be able to take advantage of the expedited process and obtain these waivers in a timelier manner. The team just needs to submit an Emergency Operation Request Form, which is located on the FAA website.

In addition to the already established laws, there are new laws involving Remote ID, that are planned to be enforced in the future. These new laws state that all drones must have a Remote ID broadcast module. This module will broadcast the drones ID, location, altitude, velocity, control station location, time mark, and Emergency status. There will be some specified areas called FAA-recognized identification areas, where a Remote ID broadcast module will not be required. These areas can be set up by community-based organizations or educational institutions. However, this means that in most cases, further development of this drone will need to abide by the new requirements and account for the addition of the broadcast module.

2.3.3. Ethical Issues

The ethical issues that this project may face in the future is concerning the same ethical discourse of automation and the robotization of many jobs across the globe. There is a distinct lack of trust that some people have with automated systems. People in the past have found simple automated systems such as red-light cameras to be an issue. They do not know whether they can trust a red-light camera to accurately trigger in the event of someone running a red light. They are also worried about how the courts would handle false positives, as there is not another human party to

testify against them. There is also the issue of automated systems taking jobs away from people who relied on them.

The main issues that an automated search and rescue drone could run into is the possibility that people may take offense that the search and rescue teams are not using people to go out and search themselves. They may feel like the teams are not actually trying to search and do not care about the predicament being handled. This of course is not true because a well-made autonomous search and rescue drone or drones will be able to locate a target much more effectively and cheaply as it may replace manned vehicles and aircraft used in the search. Using a drone to do the footwork will also ensure that no more human lives are being put at risk during the search, in the cases that the search location may be hazardous. Because of this, some cases that may not have been considered before because of danger may be able to be resolved with these drones.

Another issue that may arise is if this technology is to be misused. We have already seen multiple drone assassination attempts made against political figures. These attempts were using manually piloted drones with small explosives strapped onto drones. The pilot would be in a remote location where they could not be traced back to the drone. Someone could possibly take and adapt a search and rescue drone to instead locate and fly at a target in the attempt to do harm, which if done on a large scale could be devastating. Of course, there are military applications for this but there is definitely a conversation on ethics surrounding the entire subject to be had before development should be taken in that direction. Some may say it can save soldiers from having to put their own lives at risk, while others may see it as just the dangerous weapon that it is. Even though these issues stray away from our initial point of development, they should still be kept in mind by the developing team.

2.3.4. Privacy Issues

Some people find that drones with cameras on them are in direct violation of their own privacy, especially in cases where the drones are flying near or above their houses. In our time with this project, we should not run into these issues as we will do most of our testing in remote locations away from any housing or people in general.

However, if teams in the future are to use this drone close to people or near residential areas, this may become an issue. Some people who feel like their privacy is being breached by drones with cameras on them may call the authorities or may attempt to disable and/or steal the drone even though doing so is considered a federal offense. If this project is to be used by official search and rescue teams in the future, they will need some way to let people know their objective, and the consequences of interfering with the drone. Hopefully in the coming years, the general public

will become better educated on drones and their uses, and any negative stigma around them will settle.

3. Project Requirements & Specifications

The requirements and specifications particular to this project are divided up according to specific areas of work, and listed below.

3.1. Requirements

3.1.1. Unmanned Quadcopter

- Flies within a field of at most 22,000 ft.^{2} – maximum – autonomously (without human intervention)
- Flies at 17ft. in the air with a minimum velocity of 15 mph. in all four directions (forward, backward, left, right)
- Follows a predetermined flight pattern – based on the search area
- Drone payload does not exceed 1 lb.

- **(Stretch Goal)** Use multiple autonomous drones of similar design to operate in swarm pattern

3.1.2. Computer Vision Model

- Takes in real-time video footage from drone camera
- Detects all humans within an input frame
- Model accuracy is at least 80%
- Written in Python, and saved via Pickle or Joblib
- Returns a flag (1 = *Found*) if human is detected within input video frame

- **(Stretch Goal)** Utilizes facial recognition to determine identify of found individual
- **(Stretch Goal)** Detects humans as based on thermal signature
- **(Stretch Goal)** Detects humans in hilly terrain, rocky territory, etc.

3.1.3. Real-time Video Input Footage

- Minimum dimensions must be 320x240p
- Video footage itself must be stable – not be unstable or shaky, in any way

3.1.4. GPS Tracking/Location

- Takes in drone's position within the field
- Accepts flag returned by CV model if human is detected
- Returns drone position within the field after human is detected

3.2. Realistic Design Constraints

As every project possesses varying degrees of realistic design constraints - restrictions on our ability to perform this project as we define it during this and next semester - so ours too has such restrictions.

Budget

Like any project, ours requires money, which is then used to purchase parts, components, testing equipment, desktops, etc. The more money we have, the more likely we can spend it on items that can help meet and exceed our goals.

Our budget is dependent on the Mechanical Engineering Senior Design department through Dr. Stresau, but we have to cap our purchases at $500. Access to more would lead to our ability to incorporate components that could even be used for stretch goals - such as thermal and infrared sensors for monitoring human activity in a field.

Time

Complementing the budget is time. Two semesters are what it will take to complete this project - one of which is spent on writing what the project consists of, and why it is being completed.

More time - even four to six months - means that we can meet our stretch goals, which include:

- Identifying objects within the field that indicate human activity
- Having multiple autonomous drones work in harmony with each other in a swarm pattern
- Detect humans based on thermal signature or infrared sensors

Legal Restrictions and Training

Currently, only Joseph Manalo has experience dealing with the legal cases and ramifications of drone flying. Flying a drone is not a difficult task to learn, but knowing the legality and ethical rules and regulations takes time and effort to learn, thus limiting full-team involvement in piloting the drone.

Access to Manufacturing Tools/Commercial Simulator Software

As we are using carbon fiber to create the drone frame, we will need access to manufacturing tools. Between resources available in the UCF engineering lab and what Chase Burchett and Dr. Stresau can provide us, we should be able to acquire these tools.

4. Division of Labor & Statements of Motivation

4.1. Division of Labor

Below we list the different members on the team, as well as their roles and responsibilities.

4.1.1 Lisa Harrison

- GPS: get the location, feed the search algorithm, send location coordinates to UI
    - Sending telemetry data (sending GPS coordinates at specific interval)
        - MAVLink, translating GPS data, NEMA protocol transfer
- Track planning
    - Giving the drone a path to follow (sent to Moises)

4.1.2. Joseph Manalo

- Component selection for core design of drone
    - To select the following specific components
        - Motor
        - Prop
        - ESC
        - Vtx
        - Camera
- Coordinate with other members as needed to make their component decisions
- Head of drone construction, flight testing, and thrust/power system
- Head of budget planning and legal counsel

4.1.3. Moises Cortes Lugo

- Embedded Systems (controlling the drone's motors and general movement)
    - Programming flight controller
        - Controls drone's movement based off of given input (path to follow)
        - Put drone in "cruise control"
        - 2 modes: manual mode / UAV mode (follow flight script)
        - Not low-level programming

4.1.4. Christian Lozano

- Head of Computer UI/UX
    - Make user interface to interact with the drone
    - Display image and location coordinates of human (if found)

- Supporting Computer Vision
    - Image acquisition and model selection

4.1.5. Nicholas Norman

- Head of Mechanical Design
    - Coordinates with members to optimize mechanical design of frame and subordinate assemblies supporting component integration

- Frame Design
    - 3D modeling and manufacturing of an optimized frame, guided by structural integrity, weight, cost and camera stability as noted in design requirements

4.1.6. Joshua Mutugi

- Head of Computer Vision
    - Acquire live footage from video receiver
    - Select most suitable CV model to use
    - Process computer vision model to detect humans in frame
    - Alerts UI with notification if successful

4.2. Statements of Motivation

Below are the statements of motivation from each of the team members.

4.2.1. Lisa Harrison

My motivation for this project stems from the opportunity to apply my previous experiences to my academic (and soon to be professional) career. Before I started college, I was a paramedic and search and rescue technician for several fire departments and SAR teams. That same mentality propelled me into the US Navy, and now it is with almost a decade of incident response/preparation under my belt that I find myself on this project.

This project grabbed my attention from the first day of presentations, as I saw in it the perfect blend of what I used to do, and where I am going with my career. I am a mechanical engineering student, but minoring in computer science and intelligent robotics. The deeper into the CS and IRS fields that I have gone, the more I have realized that robotics software is where my passion is, so the fact that this project allows me as an ME to work on a project that has such a heavy CS focus is a blessing in and of itself, and then to have the goal of the project be the facilitation of a job that was my first passion is above and beyond what I could have hoped for from senior design.

Having seen first-hand the difficulties of executing search and rescue operations in challenging terrain, I have a unique perspective on the necessity of this type of technology. The development of a drone that can locate a search perimeter, and then intelligently search that area for a human figure, is the beginning of a much larger potential application, and I am excited to see where we can take it. I am especially intrigued by the fact that we will be using computer vision as opposed to thermal imaging to conduct the search. Thermal imaging can be affected by a number of environmental factors, and ultimately is only useful for objects emitting a heat signature. Computer vision, on the other hand, can be used to identify any number of distinct shapes, and the algorithms that we will develop on this project can be tailored to fit a variety of different applications, not just human search and rescue. If we can reach a point where we can combine computer vision and thermal imaging, we will have reached a point where we have a single piece of equipment capable of both rescue and recovery missions, and that is an invaluable tool, especially in a field where transporting large or heavy quantities of equipment is simply not feasible.

4.2.2. Joseph Manalo

My personal interest in the project stems from my overall interest in drones and other unmanned aircraft. Back in 2018, I bought my first hobby grade drone and put in countless hours trying to learn how to fly it. This drone was a first-person view quadcopter that was manually controlled and sent a live video feed back to a set of goggles I would wear. The sense of freedom and power I received from flying through the air at speeds over a hundred miles per hour got me hooked on the hobby. Since then, I've joined an FPV drone club on campus, learned to build my own quadcopters, and began my drone racing career. I quickly surpassed the other students in the club

and became the race team captain and coach. We have competed in dozens of races, achieving many podium positions on the collegiate scale worldwide, and in my personal career ranking top 50 in the world individually. As part of the club, we have run STEM events to get kids interested in STEM fields by showing them our drones and how they are built and flown.

I think these experiences over the past few years have built up my interest in working with drones, not just in the hobby and sports areas, but in a professional sense. When I joined senior design my first question was if there were any drone related projects that I could take part of. My overall goal in joining this team is to learn how to apply what I already know about drones and expand my knowledge to be able to help solve real world problems with them. I think I have always wanted to be part of something that could change lives, and if we can accomplish our goal, this project could help me take what was a hobby of mine, changing the world for the better.

4.2.3. Moises Cortes Lugo

When I heard Joshua Mutugi's project pitch for the search and rescue drone I was immediately interested. There are various reasons why I am drawn to this project. One of the biggest reasons was being able to work with a robotic system. I have been drawn to robots for some time and was allured by the thought of working with the embedded systems of this machine. I enjoyed taking EEL 4742C (embedded systems) with professor Zakhia Abichar. I learned how these little computers are attached to many devices, such as cars, stoves, microwaves, clocks, etc., to accomplish specific computational tasks. I also learned how to program a Texas Instruments' MSP430FR6989. Having to program the drone seems like an excellent learning opportunity; and a challenging one.

Another motivation for this project is the challenge of designing and building a drone with integrated machine learning and computer visions algorithms. This project will not be simple. It will take a lot of work from all of us to get this accomplished. As a team we will have to do much research in many areas like construction materials, flight controllers, drone software stack, web development stack, GPS functionality, computer vision and machine learning algorithms, telemetry, and search algorithms for the drone. I enjoy the challenge; it is the culmination of our road to our bachelor's degree. Where we have the opportunity of applying all our knowledge from previous classes and problem-solving skills into this project. It will also be a great learning experience.

My third motivation for pursuing this project is the knowledge gained from this experience. As I mentioned before there are many things we do not know how to solve currently. But we will do research and learn very much to be able to solve the problems that could halt our progress. I know I will learn very much in this project and that I must research how to program the drone's

flight path search algorithm. I will learn how to send and receive GPS coordinates with the drone, program the drone, learn of search algorithms for a specified perimeter and many things that I currently do not know since we are in the initial phases of the project. I will gain the experience of working in a big project with a multidisciplinary team; something that occurs in the workforce. I am sure this project is to our benefit and perhaps others.

Lastly, this project can possibly serve others by saving lives. For instance, a drone could be very helpful in a natural disaster when searching for victims. There are areas where people cannot go because of the dangerous terrain, or maybe there is no path to walk. On occasions search and rescue teams cannot find someone because of limited sight from the ground. A drone can solve this, it can fly high and be our eyes to search for any victims in need of rescuing. The drone could use machine learning and computer vision to find people in need of rescuing and send the GPS location to search and rescue teams. Drones can also help speed up the search process when limited in the number of people that can look for someone. This reason, and the previously mentioned ones are my motivations for working on this project.

4.2.4. Christian Lozano

My personal interest in the project comes from my interest in trying to make a difference in the world while also broadening my skills. One of the first things about the project that caught my attention was that it was interdisciplinary. I really liked this because what engineers do has always fascinated me because, as a computer scientist, almost everything I do is digital. So, usually I don't deal with the physical aspect of things. I feel like while working with them I actually have the chance to be able to see firsthand what goes into the planning when constructing a physical device that will actually go up into the air. Not only did this interest me, but I have always been fascinated with the idea of Artificial Intelligence and how much we can facilitate our lives with technology especially since every day Artificial Intelligence is more and more integrated into our everyday life. Thus, this seemed like the perfect chance for me to get some hands-on experience with developing an autonomous device.

As I mentioned earlier, I also had an interest in making a contribution to the world and trying to leave my mark on it. I thought this project was the perfect chance for many different reasons. My biggest one being that thousands of people a year get lost in forests and deserts, and as of right now we have no effective way besides going over as much area as we can manually to try and find the person. Unfortunately, even though those who go and try to search for hours and maybe even days usually don't end up finding the person because of just how vast the search area can be. But this project would automate this whole process with swarm technology and potentially we'd be able to cover terrains that would not have been even imaginable to cover in person and

much less in the amount of time the drones would complete them in, allowing us to create what seems to me an invaluable technology to be used for future generations.

4.2.5. Nicholas Norman

My personal motivation for this project stems from a life-long love of robotic systems and their use in the armed forces. From reverse engineering R/C vehicles to integrating on-board video systems for my UCF *laptop-controlled spy tank* project, the idea of remotely controlling a vehicle has always fascinated me.

Another source of motivation was seeing the incredible engineering on display in war movies as a child, from B-2 bombers and Panzer tanks to P-52 Mustang planes throughout our nation's history. The impact these war machines have made is remarkable, as well as the design complexity that went into them, due to the needs that they were meeting for our country.

Aside from the personal aspect of this project, the vast array of drone applications is incredible. I'm excited to work with drones (even quadcopters) that can go over 100 miles per hour and have the most athletic abilities of any vehicle I have ever seen. We are just beginning to understand what these things are capable of, and I look forward to seeing how else we can utilize them in order to best accomplish our mission.

4.2.6. Joshua Mutugi

I've always been interested in Artificial Intelligence and its applications, as its ability to classify and predict information allows for multiple uses, both publicly and privately.

Search and Rescue (SAR) operations are universal, and very hazardous for humans in many situations. The more tasks a drone or similar machinery can do, the less danger humans have to be exposed to, and the faster the mission can be completed. Computer Vision (CV) is at the heart of many of these efforts, because being able to detect and locate a human in a disaster zone is half the battle in SAR.

This project – when Nick Norman and I first discussed it – had problems relating to CV that were intriguing – such as video detection of humans and integration of a completed model with an autonomous drone. These problems have a real-world application, accomplish a tangible goal, and pertain not only to this academic project, but also to other applications in the SAR and defense industry, and thus I became very interested in solving them.

Furthermore, since CV is a field that I have relatively little knowledge on, I want to gain experience so as to broaden my horizons in learning about AI, so that that knowledge can translate over to more solutions for other problems.

5. Technical Investigation

Below is the research done for each drone aspect, as well as conclusions reached for the project.

5.1. Drone Frame

5.1.1. UAV Frame Requirements

The airframe of a UAV is its most basic structure, as well as its most vital component. Its function is to withstand all aerodynamic forces imposed on it during flight, as well as stresses due to the weight of onboard components as well as a payload (if equipped). While structural integrity is a key component, the frame also needs to be lightweight, promoting a higher overall fuel efficiency by reducing unnecessary structures.

The key drivers of a UAV frame are overall purpose, size, and material. Our purpose is SAR, which only necessitates flight components and a small (1lb) payload. The size of the drone frame will be small - medium scale, as driven by the propeller choice, which itself is driven by the overall weight of critical flight components. The drone frame form will be designed around the overall 'footprint' of these on-board components. The material we are choosing is carbon fiber, due to the high strength-to-weight ratio (2457 vs. 63.1 kN*m/kg in stainless steel).

The specific requirements for this vehicle are the following:

- Hold all necessary components
- Provide adequate CV footage
- Carry payload of 1lb

The following section will describe current quadcopter frames and their corresponding uses.

5.1.2. Common Quadcopter Frames

Several types of consumer and commercial-grade quadcopters exist, ranging from smaller, household drones to larger, more capable drones in commercial / MIL-SPACE applications. The following applications and their corresponding designs will be reviewed – FPV, Freestyle, Cinematic and Military quadcopters.
Consumer drones typically employ a 3–5-inch propeller with a 6–10-inch (diagonal) frame and are less than $500, while commercial drones tend to incorporate a larger frame, up to 15 feet long (Boeing CAV), and can cost hundreds of thousands, if not millions of dollars.

5.1.2.1 FPV Racing

FPV drones are the most popular in the consumer industry. These drones are designed around the "first person view" experience, where a live-stream video feed is fed to a pair of goggles worn by the user, as compared to other remote controlled aircraft experiences, where the pilot controls from their line of sight on the ground. Racing drones are designed to be lightweight, nimble and incredibly fast. As seen in the figure below, these drones carry only the essential equipment in order to decrease drag and lower their moment of inertia. They also use higher speed motors, triple blade propellers and powerful 6s LiPo batteries to increase thrust and responsiveness.

Their frame design contributes to this function as well. Racing drone frames are typically lightweight "True X" frames, meaning that all four of their arms converge in a central point, giving the most direct transfer of force from the motor to the quadcopter body. In order to maintain a concentrated center of gravity, all of the core components (ESC, flight controller, VTX transponder and radio receiver) are all situated in the center of the drone. In order to support this, the frame incorporates either a unibody design, with an integrated platform for the electronics, or a bolt on platform. Either of these options suffice, but in the event of a crash, it is easier to replace a piece of the frame, rather than the whole system. The battery is mounted directly underneath the drone, maintaining the center of gravity, which is critical for cornering in races [2]. Also, note the thinner arms on the example below, which reduce weight and are more aerodynamic.

Figure 5.1.2.1.A: Sample FPV Racing Drone

5.1.2.2 FPV Freestyle

FPV Freestyle drones are similar to their racing drone counterparts, in that they use onboard live-stream footage used to aid in piloting the UAV during operation. They are also capable of incredible speeds and agile maneuvers, but the focus of these drones is on their ability to perform physics-defying stunts and tricks, often recording flights with strapped-on GoPro cameras.

The frames used in these types of drones are typically "modified X" frames, in which the arms intersect, but not at the center point. There is also a center platform that runs along the middle of

the drone. The front two arms intersect in front of the center of the platform, while the rear two arms intersect behind it. This is done for several reasons. First, having a longer distance from the motor to the center of gravity increases the moment of inertia, which helps perform acrobatic maneuvers that require high inertial forces. Second, this arrangement provides more control by having the arms spaced out along the central platform, along where the center of gravity is distributed (and front heavy if equipped with an additional camera). As these drones tend to be flung around at high speeds (upwards of 100 mph), their frames are also thicker and more rigid, in order to be impact resistant when crashing.

Figure 5.1.2.2.A: FPV Freestyle Drone

5.1.2.3 Cinematic Drones

Another style of FPV quadcopter, cinematic drones are the last major option for consumer use. (Other, higher-grade, commercial drones used in cinematic production, although hex copters and octocopters are preferred). These drones are designed for capturing expansive landscapes and breathtaking scenery, as well as shoot footage for drone photography, as the name implies. The most important factor in this drone design is stability, as jerking motions can affect video quality and potentially cause the quadcopter, equipped with expensive cameras, to crash. Flight time is also a concern, as some shots can take the drone far away from its starting point.

Key features in the consumer-grade cinematic drone design are a central platform to house all electronics, an optional H-frame for added stability, and ducted propellers. Using an H-Frame, compared to the previous two designs, the moment of inertia is even greater, allowing for a smoother flight experience and easier control. The ducted propellers also help optimize thrust by reducing losses due to the tip vortex (as seen on the top portion of the diagram below), as well as increase lift by following Bernoulli's principle [4]. An added feature to ducted propellers is the protection given to the drone as well as its environment.

Figure 5.1.2.3.A: H-Frame Drone

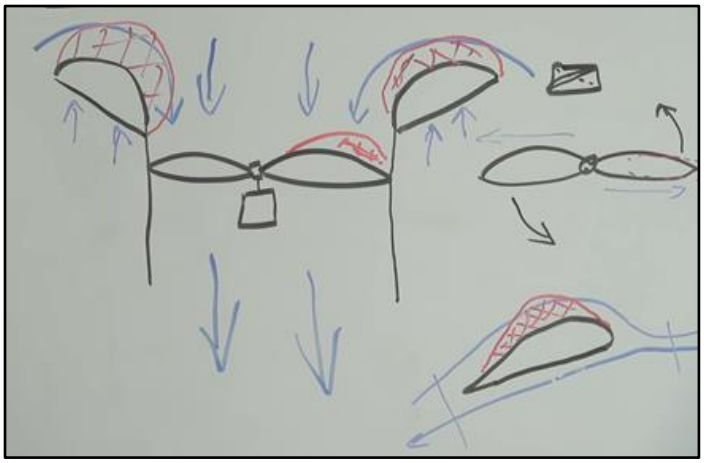


Figure 5.1.2.3.B: Illustration of Bernoulli's Principle

5.1.2.4 Commercial / Military Drones

In commercial industries, quadcopters take on a more complex role. Unlike consumer drones, these drones employ high-tech systems, which military drones use on tactical missions. In this category of quadcopters, many tactical systems are incorporated, chiefly munition payloads as well as onboard computers with thermal-imaging systems for surveillance missions.

The PC-1, for example, is a four-armed multi-rotor helicopter used in the Armed Forces of Ukraine. This drone is 21 x 19 inches and has remote track path planning, 38 minutes of flight time (over three times that of consumer-grade drones), up to 5kg of payload, detection and tracking of ground targets, among other features [5]. The Lockheed Martin Indago 3 is another high-tech military grade quadcopter equipped with night and thermal vision cameras for real-time surveillance and has long-range capabilities. This drone, for example, is 32 x 32 inches and can work up to 30 KM away from a ground station and has a 50–70-minute battery life. It can

carry up to a 5lb payload, has a 360 degree 6-axis gimbal that rotates a HD camera with 32x digital zoom and integrated laser for high accuracy targeting, all with a price of $25,000 [6].

Overall, the size, complexity and power capabilities that usually limit most consumer drones are not present with commercial and military-grade drones. These types of drones showcase advanced quadcopter technology, some of which will be shown in this project.

Figure 5.1.2.4.A: PC-1 Drone Used by the Ukrainian Military

5.1.3. Important Frame Design Considerations

5.1.3.1 Hardware

In designing a quadcopter frame, several important considerations must be accounted for. These include, but are not limited to: all hardware onboard the drone, drone mission (surveillance, cargo transportation, high-speed tracking), as well as existing boundary conditions for each on-board component. Together, these constraints guide the design of the drone frame, as well as subordinate components which support its functionality.

In this project, the drone will be performing a search-and-rescue mission. The requirements for this mission are a flight speed of 15mph, provide platform for a payload of 1lb (besides necessary components), and provide adequate CV footage. The drone will conduct this mission inside a max area of 22,000ft. Other hardware considerations, given the finalized component selection (see sections 5.1.5, 5.1.8), are using 8-inch propellers, two 1300mAh 6s LiPo batteries, and an onboard telemetry module, as well as other critical flight components. The only other technical boundary condition is maintaining 5cm distance from the frame to the GPS module in all directions to avoid signal interference.

Accommodating all of these hardware components, the two most-common drone frame shapes that will be mentioned in Section 5.1.3.2 must be evaluated for feasibility. All of the previously

mentioned components must fit on a standard sized frame for an 8-inch propeller (350mm [3] in either an X or H frame configuration). Displayed below is a concept X-frame with flight hardware directly installed (excluding batteries and on-board telemetry module):

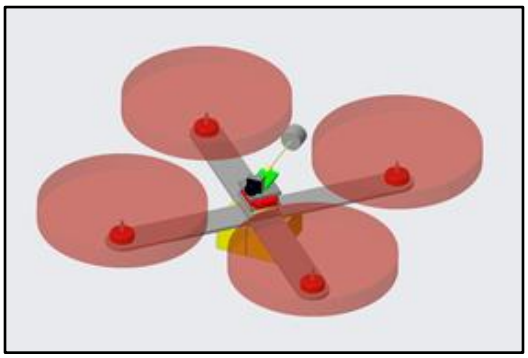


Figure 5.1.3.1.A: X-frame: Oblique View

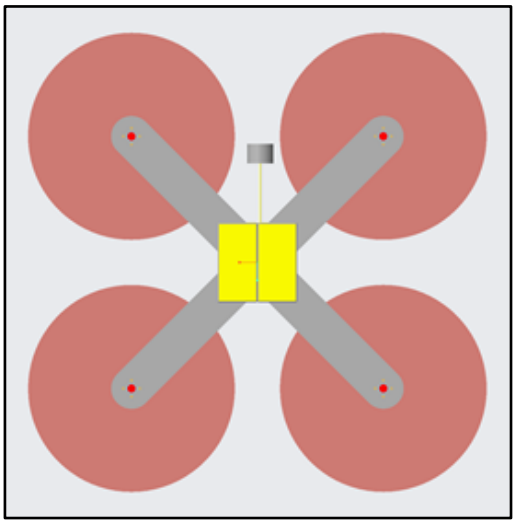


Figure 5.1.3.1.B: X-frame: Bottom View

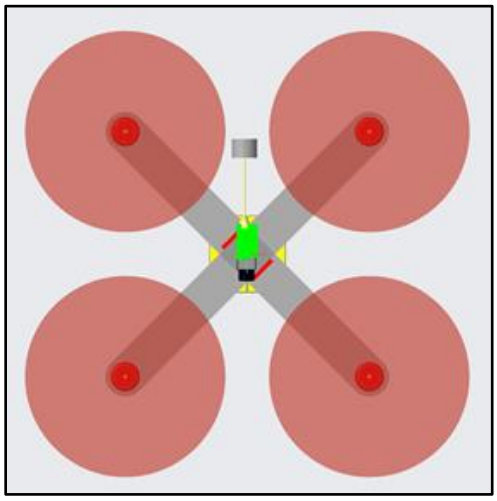


Figure 5.1.3.1.C: X-frame: Top View

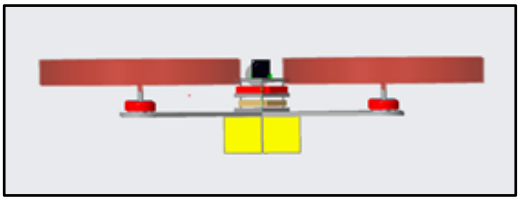


Figure 5.1.3.1.D: X-frame: Front View

As clearly shown in the above images, the X-frame allows for perfect symmetry, with all the critical components mounted on center intersecting portions on the frame. With a basic setup with only critical components and one battery, the frame by itself would be sufficient. Major components, such as the ESC and flight controller, can be adequately contained in what is referred to as a 'flight stack,' demonstrated in Figure 5.1.3.1.D.

However, two batteries are present, as well as a telemetry module and GPS, which need further retention sub-assemblies. Other considerations would be the visible camera angle, which would be blocked by the spinning propellers during flight, and would also need a sub-component to attach to the frame. As is, it would greatly be inhibited in its ability to provide adequate footage for the CV model.

Therefore, the X-frame platform is theoretically sufficient, but further revisions are necessary in order to incorporate the additional flight components in to the vehicle that are necessary to carry out the intended mission.This leads to the consideration of a cargo transportation system for these components which are unable to be mounted to the flight stack.

5.1.3.2 Preliminary Research with Common Frames

As mentioned above, the two common quadcopter frame shapes are X-frames and H-frames. X-frames are generally preferred, due to simplicity of design, symmetry and reduced moment of inertia (force required to change inertia on a body). Having low inertial moments are especially critical when operating at high speeds and performing tight maneuvers on a track. Also, an X-frame (or modified X-frame) helps maintain absolute control of the aircraft while inducing high inertial loads in acrobatic stunts in a free-style competition.

An H-frame, on the other hand, is typically employed with entry-level to mid-level cinematic drone photography. These quadcopters require smooth, stable footage and will not be performing extensive acrobatic flight maneuvers as their FPV counterparts. The principle behind an H-frame is increased storage and space for electronic components as well as increased rigidity and stability in flight, due to increased moments of inertia. In order to compare the stability of flight between these two frame shapes, consider the following calculations below:

Here is a sample calculation of inertia for an X-frame of diagonal length 250mm, made of Cyanate Ester Carbon Composite (D = 1640 kg/m^3). The mass of all components excluding the frame is 723g. Distance r = distance from rotor to center of frame.

$$I = m * r^2 = 0.8771kg * \left(\frac{250*10^{-3}}{2} m\right)^2 = 13.705 * 10^{-3} kg * m^2$$

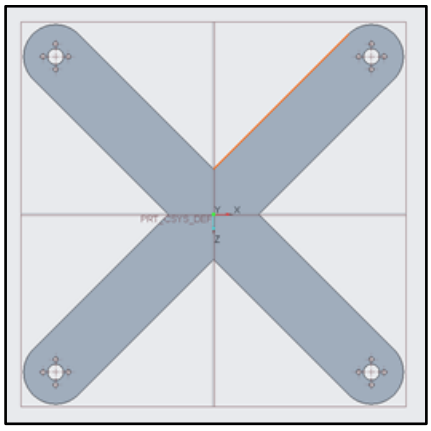


Figure 5.1.3.2.A: 250mm X-Frame
(Mass of Frame = 0.15411kg)

For an H-frame of similar diagonal length and material (Cyanate Ester Carbon Composite), with *r = L1 + L2* (Distance along drone frame to center point), the moment of inertia becomes:

$$I = m * r^2 = 0.91385kg * \left(\left(\frac{250}{2}\cos(45°) + \frac{250}{2}\sin(45°)\right) * 10^{-3}m\right)^2$$
$$= 28.558 * 10^{-3kg} * m^2$$

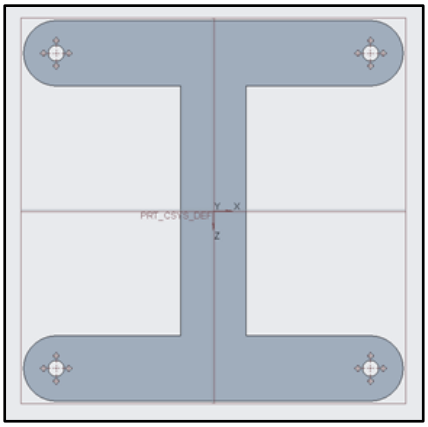


Figure 5.1.3.2B: 250mm H-Frame
(Mass of Frame: 0.19085kg)

As seen above, on a similar-sized drone (250mm diagonal), having an H-frame increases the moment of inertia by 108% as well as the weight by 23.8%, neither of which are desirable for our application. An increased moment of inertia can be helpful when rapid, sudden moves are not desired by the user, and allows for smoother footage to be taken, but the H-frame is not as symmetrical as the X-frame, which will have a more efficient hover and flight stability.

Consequently, while on a smaller scale this might not be as noticeable, in larger drones with complex missions and tight tolerances for flight stability, this data can play a critical role in mission success. Other factors will determine which frame type is selected, namely payload, propeller and motor size, as well as the previously mentioned requirements (listed above).

Another important factor to note is the inverse pattern in which the propellers spin. Similar to helicopters, vertically propelled aircraft must account for inertial rotation experienced due to the centripetal force caused by the spinning propeller. While this is not a dynamic design factor, it is important to note the way in which these forces are accounted for. Helicopters typically include a tail rotor to compensate for this spinning effect on the aircraft. As seen below, most multirotor aircraft incorporate symmetric opposing propeller directions to cancel out the centripetal effect on the rotors. As long as they are equally compensated (two counter-clockwise and two clockwise), the placement of these propellers is irrelevant to the design. See below:

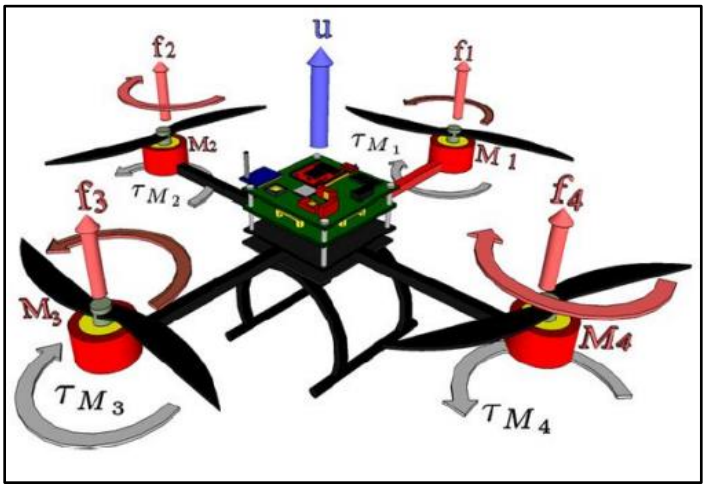


Figure 5.1.3.2.C: Counteracting Centripetal Forces Induced by Propellers

5.1.3.3 Preliminary Tests with Common Frames

Several tests were conducted on the most common frame, the X-frame (as seen in the figures above) made in carbon fiber. Finite Element Analysis was used in Creo Simulate to determine the properties and effects of the maximum load of the motors on the frame. This would be used in order to decide which would be the most adequate for our applications. Several high-impact test instances will be considered, and the sum of all of them displayed below (in test number 4). The worst-case scenario is loss of control of the motors, in which they achieve full power, or cut off completely. The following instances will be considered:

1. Inertial loading from max throttle on rear two motors (pitching the aircraft forward) while at hover
2. Inertial loading from max throttle on left two motors (rolling the aircraft to the left) while at hover
3. Max throttle given to left front motor at hover
4. Max throttle given to all four motors at once

For tests 1-4, a force of F will be applied to the models in the corresponding motor location. This force value is the max thrust force applied by the drone per motor, with given battery, motor and propeller selection. The 250mm X and H-frames being studied have the same masses as mentioned above, four 1900KV brushless motors, each swinging a 7 inch, 4.5inch pitch propeller, and a 1300mAh 22.2 6s LiPo battery. The equations that lead to this maximum thrust value are the following:

$$F_T = P * d^3 * RPM^2 * 10^{-10},$$

Where:

- P = pitch of rotor blade,
- d = diameter of rotor blade
- RPM = max rotations per minute (at full power)

$$F_T(ounces) = 4.5 * 7^3 * (22.2V * 1900KV) * 10^{-10} = 274.612 \text{ Ounces per motor}$$

$$F_T(N) = Ounces * 0.278014 = 274.612 * 0.278 = 76.346 \text{ Newtons per motor}$$

The material itself was carbon fiber, and material properties were calculated using the average of three values found from different resources. The average density value was 1.58, and the Poisson's ratio was 0.33. Additionally, the Young's Modulus was 100 GPa, and these three values were entered into the simulator for the X - frame in the third test (full throttle given to all four motors). The Von Mises Stress, Displacement and Principal Stresses were measured:

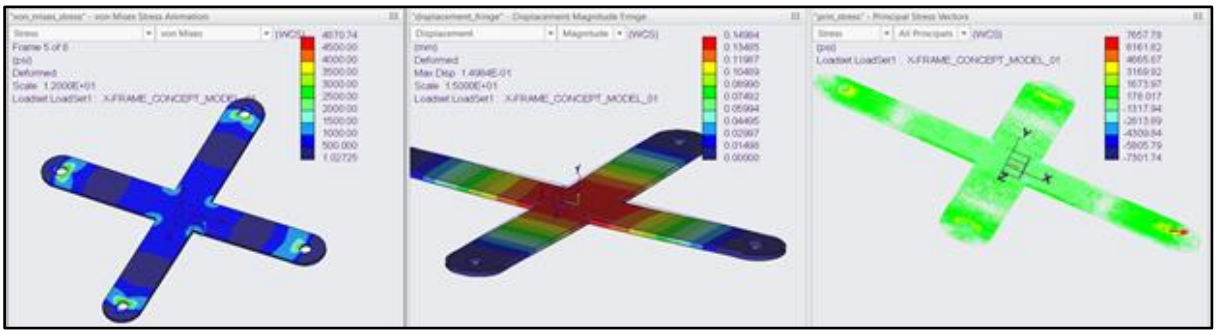


Figure 5.1.3.3.A: Test Results On X-Frame Design

Aside from max displacement in the middle of the frame and major stress concentrations in each of the corner sections where the arms meet the center (expected), the results revealed that no critical damage to the frame was experienced. Through all of these tests, the carbon fiber frame retained structural integrity. Further subordinate design changes will be made to eliminate these minor concerns above.

5.1.3.4. Cargo Rail System Concept

Figure 5.1.3.4.A: Drone Cargo Rail

One area for revision is a modular cargo rack, which can be bolted to the underside of the X-frame. This allows for the development of attachments that can hold and protect additional flight components, easily mounted to the frame. This would also maintain a central location of the center of gravity, as well as lower it, further increasing stability. . These rails will serve as the connection points for the design. With this setup, the major limiting factor would be thrust from the motors, in order to carry the extra weight. This design is not feasible on an H-frame due to lack of symmetry.

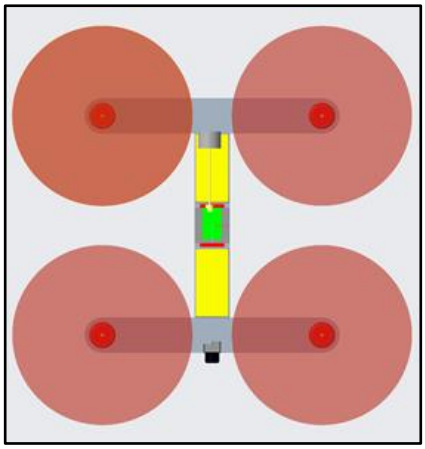


Figure 5.1.3.4.B: H-frame: Top View

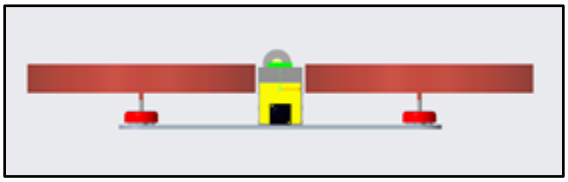


Figure 5.1.3.4.C: H-frame: Front View

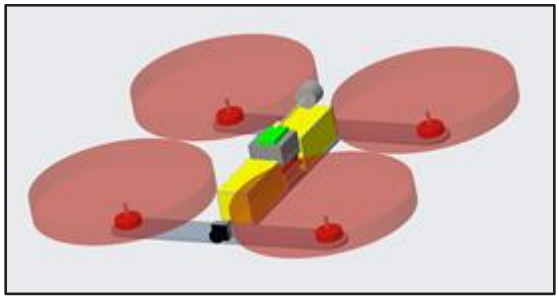


Figure 5.1.3.4.D: H-frame: Oblique View

Conversely, the H-frame holds all the components on its frame without modification. Both batteries, all components, and the missing onboard computer will have plenty of room for fastening, as well as offer further cargo space if needed. As it stands, it is symmetric about the y-axis and has lower moments of inertia than its x-frame counterpart. The negatives about this design are that it is not symmetrical about both the x and y axes, meaning that it has a higher moment of inertia in the y-axis than in the x-axis, due to the battery weight. The main concerns would be stability during flight and viewing angle for the CV algorithm. Possible issues could also arise due to the proximity of the batteries to the flight controller and VX transmitter.

For either model, they will employ a livestream camera, providing real-time footage to a CV algorithm, which will in turn search for potential victims in the mission. Providing adequate footage for the CV model is critical. The camera angle on the drone will determine its field of view, and if in fixed position on the drone, the flight speed of the aircraft. As FPV drones are typically equipped with a higher angle view, they render the aircraft heavily leaning forward, thus increasing speed and forward inertial forces. In this project, the max velocity is 15mph, which is far away from the 100+ mph top speeds of most racing drones, therefore, a smaller angle of attack, or differently mounted camera is needed.

5.1.3.5. Material Selection and Manufacturing

The materials used for this drone build will principally be carbon fiber, as well as other strong 3D filaments for supporting frame members. Typically, FPV frames are made from carbon fiber sheets, which are cut into form and bolted into place. This allows for a simple frame to be mass produced and easily assembled by inexperienced hobbyists [2]. These frames are stiff and regarded for their durability, withstanding impacts from multiple crashes and protecting valuable electronics.

Most commercial drones have a custom 3D frame, engineered for the intended purpose with careful design considerations. These 3D frames are often 3D printed, using a carbon composite nylon filament, which is durable, but more flexible than pure carbon fiber. These 3D filaments require a special printer, and have limited accessibility. Other nuts, bolts, and mounting hardware are used, and may be implemented to reinforce the frame or supporting structures.

Our design will account for both types of drone frames. Due to simplicity, either the frame and supporting pieces can be carbon fiber or 3D printed, or a mix of the two. Some components need to be sturdy and might benefit from being cut out of a pre-formed carbon fiber sheet. Others, namely non critical members, do not have as tight design restrictions and can be 3D printed using a variety of filaments.

5.1.4. SAR Frame Development

5.1.4.1. Quadcopter X-Frame Design

Figure 5.1.4.1.A: Printed Quadcopter X-Frame Design

Overview

The first and most critical component to be designed was the actual frame for the SAR drone. As mentioned above, the vehicle chosen was a multi-piece X-frame quadcopter (noted by equidistant rotor arms in a symmetrical pattern), and was cut from a 400 x 300mm sheet of 4mm thick 3K plain weave carbon fiber. It features a three-hole 3mm arm hole mount with a centered 30.5 x 30.5mm flight stack and four 3mm holes in a 16 mm diameter for the motor mount. Additionally, the propeller directions are as follows: the two front rotors are CW, and the two rear rotors are CCW.

Carbon Fiber Frame

The thickness of the carbon fiber was determined by over-engineering common quadcopter frames. Typical frames are 3mm, with 4mm frames being much stiffer, adding further protection against crashes and potential design flaws (such as stress concentrations and other forces unaccounted for during design phase).

3K Plain Weave carbon was chosen due to the simple use case and structure of the SAR vehicle. The “K” factor refers to the amount of carbon fibers in the pattern. Common values are 3K, 6K, 12K and 15K. There is no added benefit for having an increased K value, and since we are driven by budget, 3K is plenty sufficient for our purposes (specific strength = 529 Kpsi or 3650 mPa). These 3000 carbon fibers are woven into patterns, some of which are called the *plain weave, twill weave,* and *harness satin weave*.

The plain weave shown below has short spaces between interlaces, which give it a high level of stability, and incredible strength and rigidity in the XY plane. This weave is strongly recommended for flat structures.

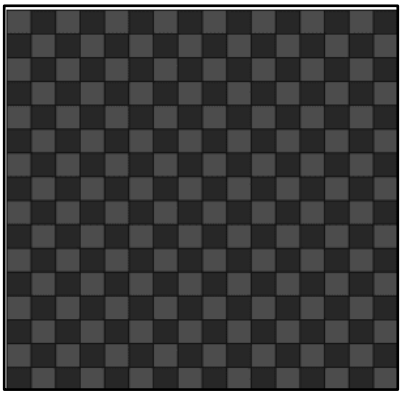


Figure 5.1.4.1.B: Plain Weave Design

The twill weave has good pliability and can form to complex contours, and is better at maintaining stability than a harness satin weave, but is not as stable as a plain weave. Having a longer distance between two interlaces in this weave leads to fewer crimps compared to the plain weave and less stress concentrations.

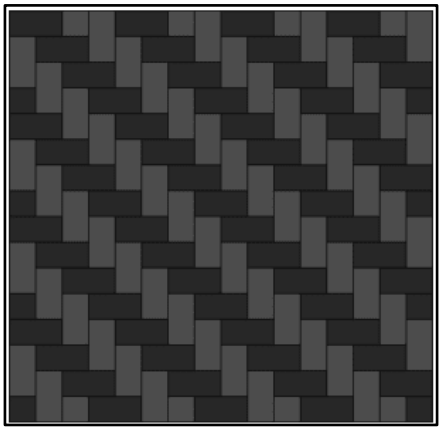


Figure 5.1.4.1.C: Twill Weave Design

The Harness Satin Weave is commonly related to fabric and drapery applications. As imagined, it has excellent draping qualities and improved appearance, forming around complex contours with ease. However, for our applications, its low stability is not preferred.

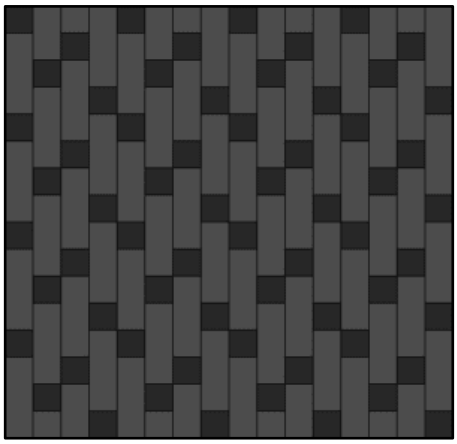


Figure 5.1.4.1.D: Harness Satin Weave Design

Overall, the plain weave best suits our purposes as it provides the best stability for a flat surface (X-frame) and high strength in the result of a crash or a strong force in the z-direction (such as a motor force). Any of the other weaves are best suited for aesthetics and curved surfaces. The 3K carbon fiber count is also our best choice, given its high strength and affordability in the plane weave variant.

Drone Frame Form

An important factor to consider in design development is manufacturability. While the preliminary testing was performed on a unibody X-frame, this design was unfavorable due to several reasons.

First, if the design failed, or an arm broke, the entire frame would need to be re-manufactured. This would apply in the fabrication process, where one wrong cut would render the entire frame useless as well as in the instance of a crash. Second, the carbon fiber sheet needed to cut the frame from would need to be larger than the actual drone design, which would increase costs and wasted material (see image below). However, in the assembly, all of these issues are eliminated. The frame can be cut in such a way as to minimize the unused carbon fiber material and maximize the usability of the overall drone components.

With the 300 x 400 x 4mm 3K plain weave Carbon Fiber sheet, two 6 x 6 inch center plates and five 10 x 2 inch arms were fabricated for under $60, allowing for a spare arm in case of a quality control issue. This would have been impossible with a unibody design.

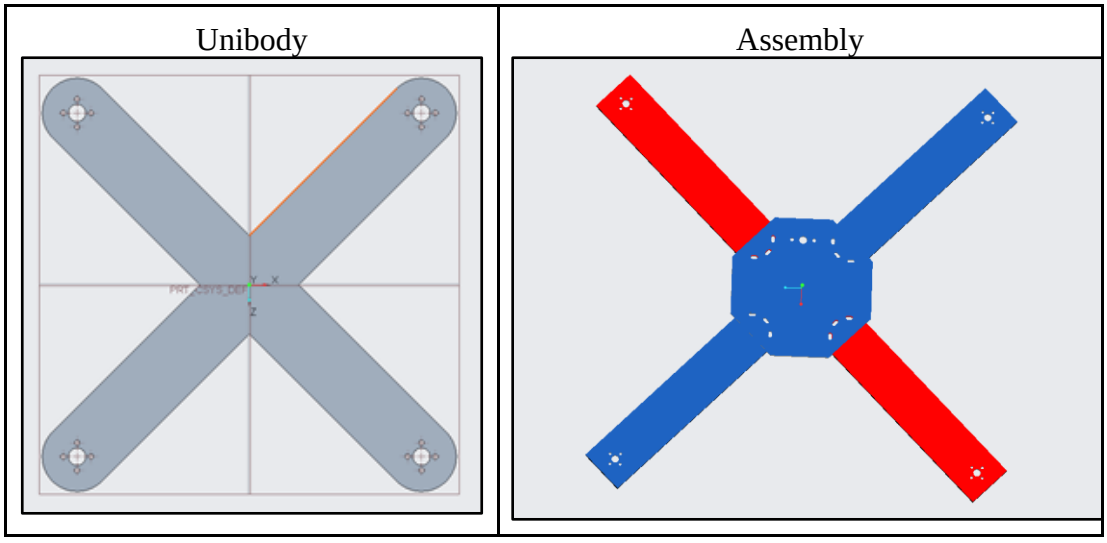


Figure 5.1.4.1.E: Unibody vs Assembly Design

Another reason to include an assembly-style design for the X-frame is the ability to increase the available surface area for mounting purposes. Not only would this extra area be useful for direct mounting (such as bolting and clamping), but also allows for rail components to be more securely mounted. Below, the arms and center plate are going to be considered.

Center Plate

The center plate was designed around its primary boundary conditions, which necessitated space for each of the 10 x 2 inch arms to mount as well as to clear space for the center flight stack (flight controller and ESC, both mounted on a 32 x 32 mm square base). A lesser boundary condition was in place, which was to provide room for side-by-side mounting for both batteries and the flight stack, as it would also provide frontal clearance for the camera assembly.

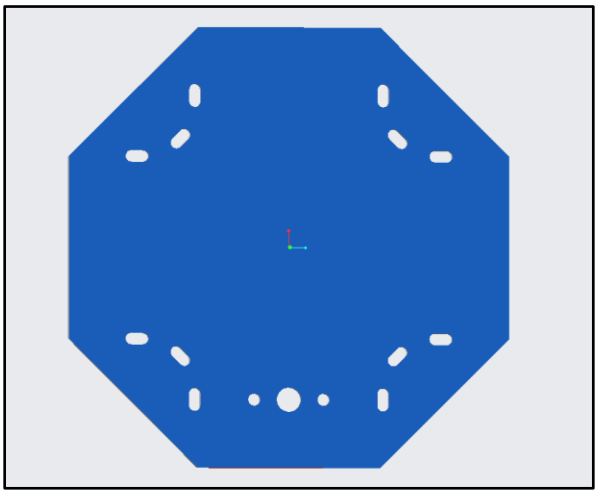


Figure 5.1.4.1.F: Dimensions: edge-to-edge (Diameter = 6 Inches)

The footprint of all on-board components was taken into consideration, and placing the two on-board batteries on either side of the flight stack gave a value of just under 6 inches.This value became the driving dimension for the center plate, which is 6 inches in ‘diameter,’ from top to bottom and side to side. This allows for a conservative amount of space between each arm and the flight stack.

Another design consideration independent of the footprint of the other components was the GPS mounting location. Since the requirement was 5cm of clearance in all directions, a separate bracket must be developed and attached to the main frame to provide this distance. This system was to be developed and mounted close enough to the center stack that the included cable could connect to the flight controller.

Drone Rotor Arms

The rotor arms were fabricated from the 300 x 400mm 3K carbon fiber sheet with three design purposes in mind: to maximize overall rotor length from the center plate, maximizing the moment of inertia and stability in the first iteration of the overall drone frame, to allow for a spare arm to be fabricated, and to provide enough space for further drone arm modifications in

subsequent iterations, while providing an adequate, functioning drone arm. Additionally, a 10lb motor force was tested on each frame design in Creo Simulate, with fixed mounting hole constraints.

The above guiding design considerations lead to the following rectangular form. It features two attachment points, on the left, motor mounts, and on the right, attachment points to the center plate.

The motor attachments were based off of the universal four 3mm holes in a 16mm radial pattern. This pattern is common in FPV quadcopter applications, with the other mounting patterns being 12 x 16mm and 12 x 12mm. The motors used in this drone are 2407, which feature a 16 x 16mm pattern, although 19 x 16mm is commonly used as well. A typical pattern is pictured below:

Figure 5.1.4.1.G: Motor Mounts for 220X-240X Size Motors

The other mount is a three-bolt 3mm batten as well, although designed with elliptical shapes for quality control purposes. Its implementation is visible below:

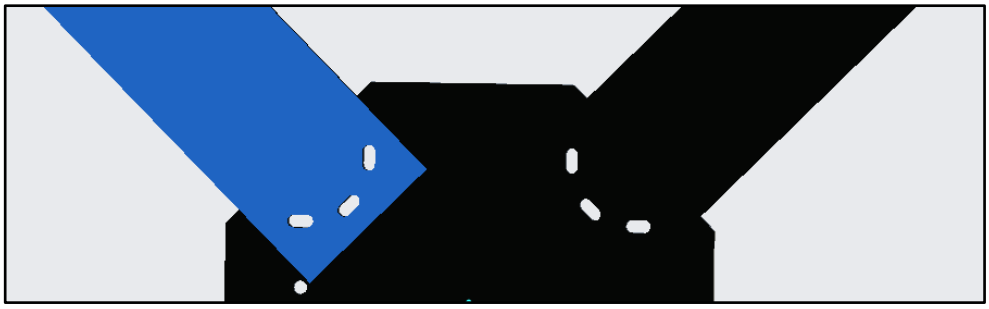
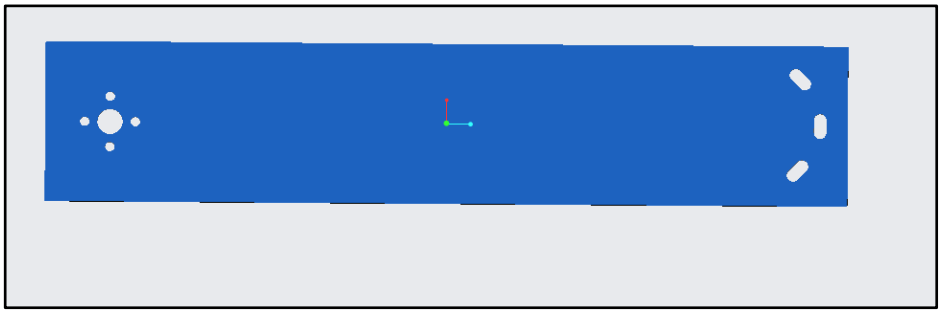
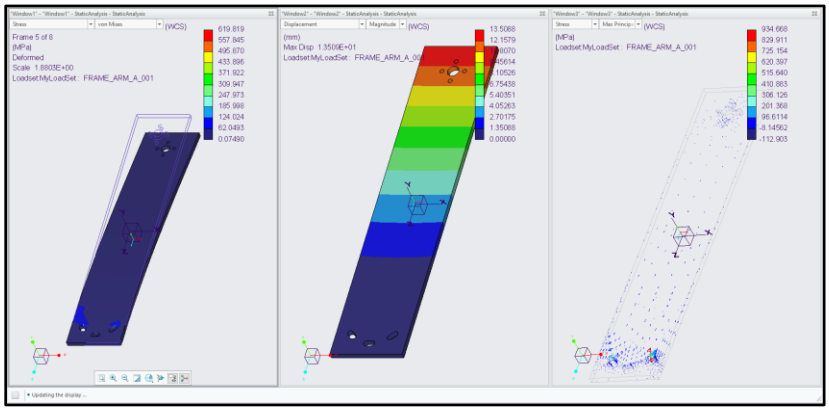


Figure 5.1.4.1.H: Connection Points for Drone Arms

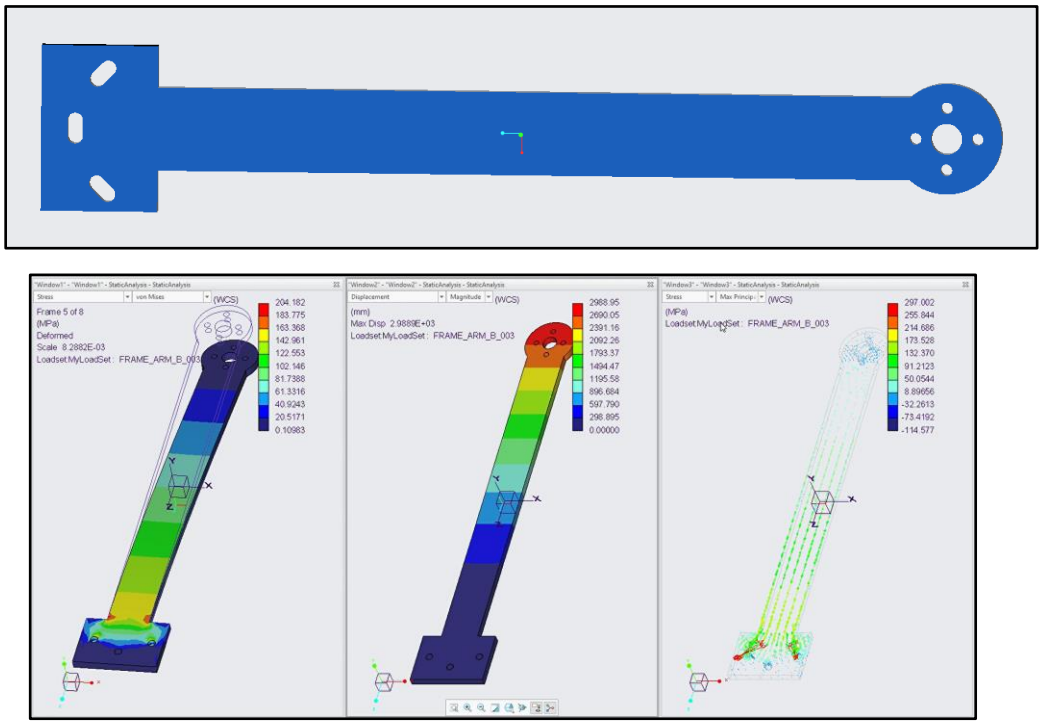
Note: Although neither the actual arm nor the base plate feature the elliptical pattern in the actual design, this pattern has been designed in the Creo CAD file for two reasons. First, while assembling the brackets onto the frame, a quality control issue was noted in the uneven spacing and positioning of the arm and center plate holes. This caused a misalignment of the rail brackets and rails on the frame assembly, and prevented adequate integration. A design improvement was necessitated, and the elliptical holes visible above functioned to eliminate this alignment issue, allowing the brackets a certain degree of freedom, given the imperfectly drilled mounting holes on the frame and arms. Second, Creo allowed for a more cohesive connection parameter with this shape, given the adjusted mounting holes for the rail frame brackets.

As stated above, this was the first iteration of the drone arm. A large, yet unrealistic value for the Von Mises stress is noticed, at 619.819 MPa. Although this number is large, it is not expected to cause any material failure in the structure, given the adjusted test boundary conditions. It does, however, show an imbalance in an overly rigid structure, which is reflected in the design changes of iterations two and three. After this rigid structure was created, the second design goal was weight reduction which is shown in the following iterations below:





Figures 5.1.4.1.I & 5.1.4.1.J: Iteration 1
Dimensions: 2” x 10”
Weight: 71g
Max Von Mises Stress: 619.819 MPa



Figures 5.1.4.1.K & 5.1.4.1.L: Iteration 2 Design
Weight: 32.14g
Max Von Mises Stress: 204.182 MPa

In iteration 2, a much lower maximum Von Mises stress of 204.182MPa was experienced, and only in the stress concentration area of the sharp corner experienced this. This equals a 67% reduction of anticipated stress in the model, while achieving a 55% weight reduction. The design changes are reflected in narrowing the overall width of the drone arm, while maintaining the frame mount thickness.

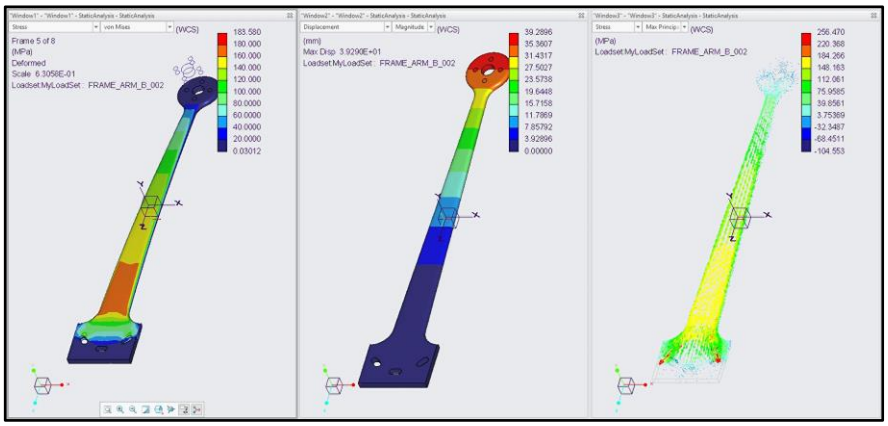
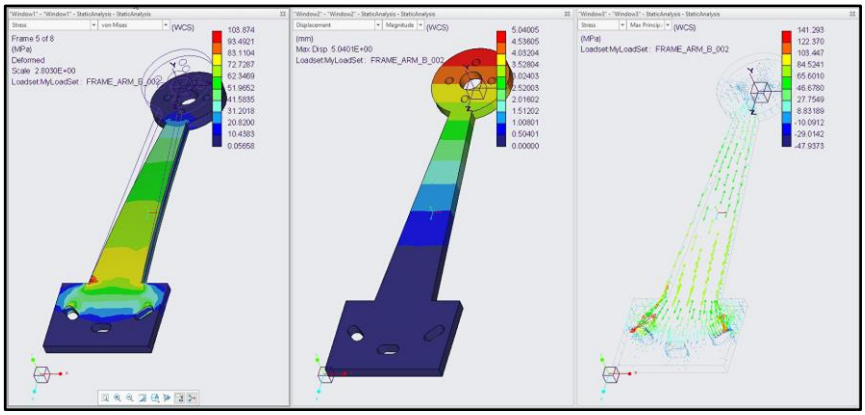


Figure 5.1.4.1.M: Iteration 3 Design,
Weight: 24.38g
Max Von Mises Stress: 183.58 MPa

In iteration 3, further weight reduction and stress reduction design intentions were realized. Principally, a trend of lowering stresses towards the motor mount led the following iteration to narrow from the plate mount to the motor mount. Also, rounded edges along the entire arm were incorporated in order to minimize stress concentrations. These design changes reduced the weight another by another 30.3% per arm, and reduced the Von Mises stress experienced by 10%.



Figures 5.1.4.1.N: Iteration 4 Design,
Weight: 18.4g
Max Von Mises Stress: 103.874 MPa

In Iteration 4, a major design change was implemented, which was cutting the arm length from 10 inches overall to 6 inches. This was done after further research on PID controllers, as drone stability was a concern up until this point. Having the longer drone arms increased the moment of inertia by 40% as well as the *gain*, which is the effect of making a PID controller change on the overall motor's directional control. Reducing these values requires more force to achieve the same angular acceleration, in other words, a change of direction requires more power.

Given these drawbacks to reducing the arm length, the advantages still outweigh the drawbacks, given the mission requirements. Since this drone is not doing acrobatic movements nor traveling over 15 mph, its flight path is considered stable in all use cases. Thus, this design change is justified as it further reduces weight.

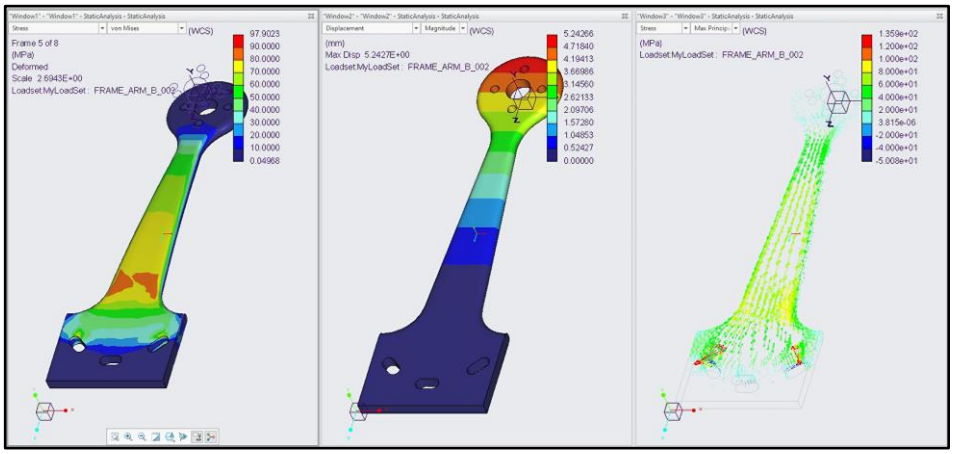


Figure 5.1.4.1.O: Iteration 5 Design,
Weight: 16.1773g
Max Von Mises Stress: 97.90 MPa

Iteration 5 was the last iteration on the drone frame arm, which incorporated rounded edges to reduce the stress concentrations present in the sharps corners of iteration 4. Aside from this minor change, a direct reduction of 12.08% and 5.75% in the weight and Von Mises stresses was achieved from the last iteration.

Comparing iteration 5 to iteration 1, the efficiencies were greatly improved. A total reduction of weight and Von Mises stress was 84% and 77.22%, respectively as the drone arm was reduced to weighing 16.18g from 71g, and four of these arms reduced the overall weight of the drone arms by 215.2g from 284g to 64.8g, while maintaining 60% of the leverage and efficiency that these 6” employ compared to the previous 10” arms.

The overall frame appearance is shown below, with the updated arm design.

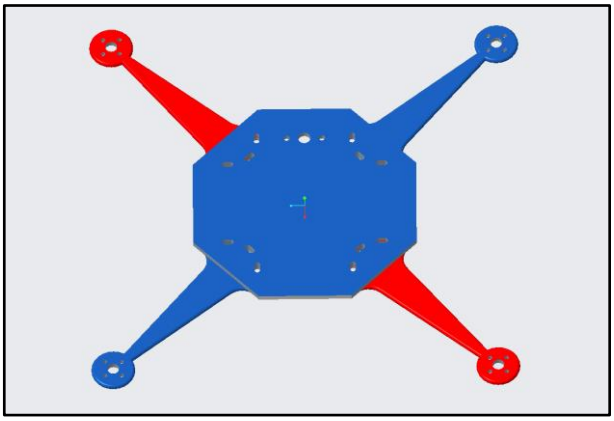


Figure 5.1.4.1.P: Overall Drone Frame with Updated Arms

Weight: 168.2g

5.1.4.2 Tactical Rail System (TRS) and Component Design

This section refers primarily to CAD designed parts, which were 3D printed using general PLA filament. This filament is typically used for geometrical prototypes, and is easy to reproduce, but not recommended for heavy duty applications. Due to its simplicity of fabrication in 3D printing, it was used to make almost all of the designed components and the TRS brackets, printed on a consumer-grade Ender 3 3D printer.

Other supplemental materials were used, such as aluminum tubing and ABS plastic for more complex parts, such as the rails for the TRS and the BLG battery landing gear, which featured complex geometries difficult to produce on an Ender 3 with PLA filament. The forces experienced by the majority of these tertiary parts were low enough to not require FEA finite element analysis on Creo Software, but were test fit to ensure functionality, and updated with any relevant design changes.

Tactical Rail System (TRS)

The following system to be developed was the Cargo Rail System (CRS). This subsystem was to be the vehicle by which further components would be attached to the frame, and would lend itself to a universal mounting system for future attachments (IR camera, packages to be delivered to found persons, additional onboard computers, etc.).

Inspired by tubular designs in which the components would attach to and be designed around a round ‘snap-on’ or insertion connection, the following design was created:

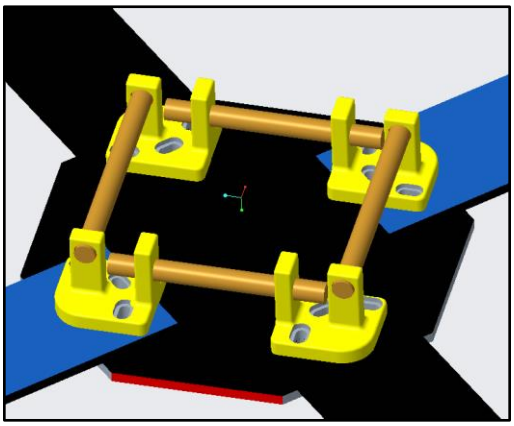


Figure 5.1.4.2.A: Tactical Rail System (TRS)
Weight: 43.6g

Although simplistic in nature, this design is particularly helpful in its SAR functions for several reasons. First, the aluminum tubes (pictured in gold) serve attachment points with just under 400mm of usable space. There are four pieces in total, two that are 80mm long and two that are 120mm long. Combined, they offer frontal and side mounting options as well as complete protection for the centrally mounted flight stack (flight controller and ESC). The rectangular shape of the mounting rails allows for parallel mounting for components such as batteries, which come in pairs of two, and need to be mounted on either side of the flight stack to maintain a center of gravity.

TRS Rails

The rails were originally brass, due to availability and ease of fabrication. Brass rails did not experience deformation when being cut with a handsaw, and maintained form through repeat installations on the brackets. However, brass has a high density, at 8.73 g/cm^3 and the strength to weight ratio is low at 67.8. Aluminum, however, has a density of 2.7 g/cm^3 and a strength to weight ratio of 115, being lighter and still maintaining the rigidity needed to support any attached components on a 8inch propeller SAR quadcopter drone.

Changing the rails to aluminum saved 24.4g as the 4 brass rails above weighed 36g, while the four aluminum rails weighed 11.6g. Overall, this change caused a 67.8% weight reduction.

TRS Brackets

The second component in the TRS is the rail bracket, which serves the purpose of connecting the tubular rail attachment point to the drone frame via the arm connection points. Two iterations were developed, with the focus being on providing two attachment points (one for the frame, another for the 8mm diameter aluminum rail).

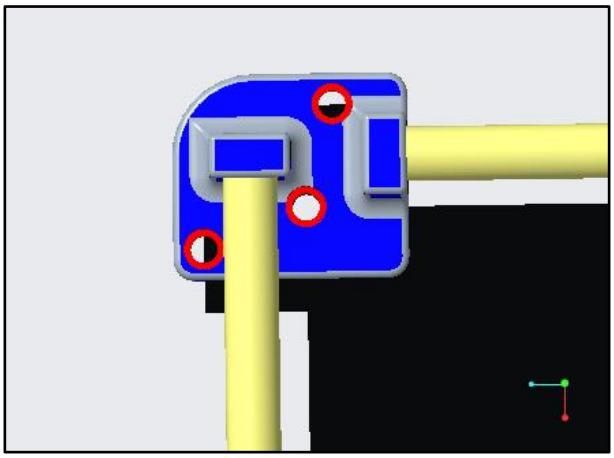


Figure 5.1.4.2.B: Iteration 1
Volume: 9.0763 cm^3
Weight: 7.846 g

Iteration 1 was a simplistic design, based on the two requirements for this bracket - connect to the frame and 8mm diameter tubes. The design features no thru holes, but an attachment where the rails slide in snugly. Also, circular holes are included, which attach onto the frame, with eyelets for the bolts to attain a tighter fit.

This first design was problematic because of limiting the tolerances for the rail lengths themselves, having an 8mm tolerance on the length to be cut, as each of the blind holes are 4mm deep. This also meant that each rail had to be installed on all brackets before being bolted onto the frame, instead of inserting them after the bracket is mounted, which complicates the assembly process. Additionally, the blind holes proved to limit the amount of rail real estate for mounting brackets, shortening the rail space by approximately 60mm.

The most important limiting factor, as alluded to above, is the connectivity of the 3mm circular holes which attach to the frame plate. Since these attachment points offer no flexibility in tolerances for the frame holes, which are not 100% accurate, any deviation in their positioning rendered the iteration 1 brackets unusable.

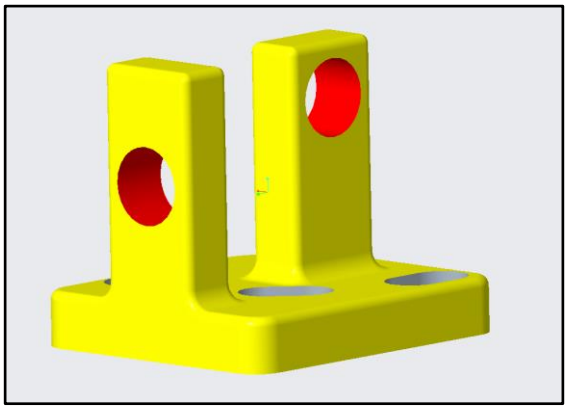


Figure 5.1.4.2.C: Iteration 2, Oblique View
Weight: 8g

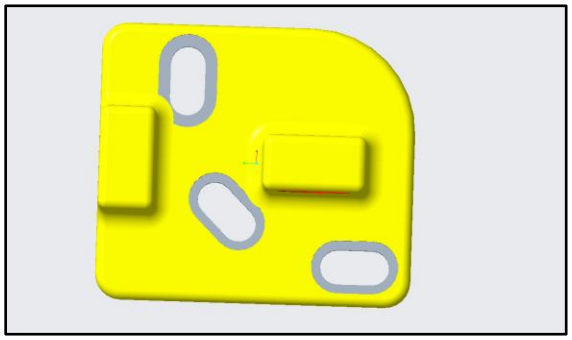


Figure 5.1.4.2.D: Iteration 2, Top View
Weight: 8g

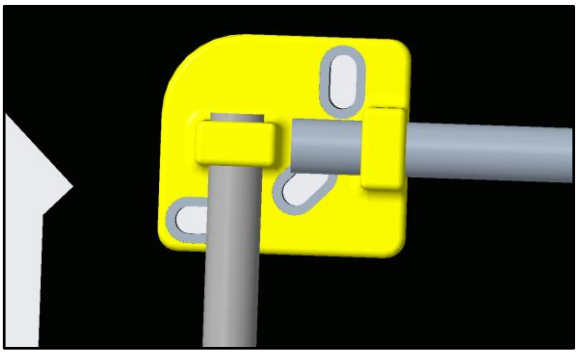


Figure 5.1.4.2.E: Iteration 2, with Rails Installed
Volume: 9.255 cm^3
Weight: 8g

The iteration two brackets featured two major updates over the first iteration, thru rail mounting holes and elliptical mounting slots. The purpose of the thru mounting holes was to allow for a lower tolerance for the rail length, only requiring that the rail be at least between the two mounting posts, and allowed for perpendicular mounting between the two rails, as utilized in the landing gear.

Adding in the elliptical mounting slots was also a large improvement as the center of each slot was the theoretical location of the corresponding frame mount hole, but allowed for a 1.5mm deviance in each direction (horizontal for the left hole, oblique 45 degree translation for the middle hole and vertical translation for the rightmost hole). These elliptical holes still featured eyelets for the bolt head to attach snugly against the bracket, and allowed for the adequate alignment necessary to install the four perpendicular rails, even with slight deviations in the frame hole alignments.

Tactical Camera Mount (TCM)

The tactical camera mount, pictured below, served two primary purposes based on the stable viewing angle requirement in the requirements section. The first was to provide a secure attachment point to the drone, and the second was to allow for an adjustable, yet firm viewing angle. See the TCM below:

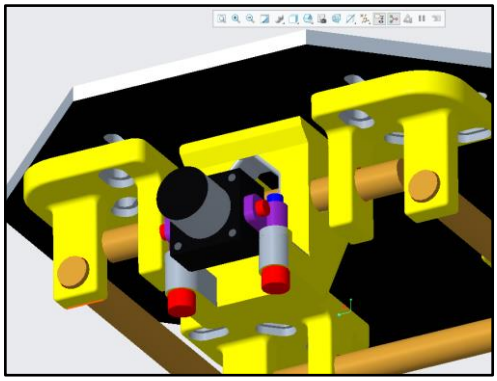


Figure 5.1.4.2.F: TCM Assembly, Iteration 5

The TCM was primarily designed with two connection points in mind, the included camera mounts (see below) as well as the tactical rail, which would secure the component to the frame. The included side mounts featured an 8mm hole on either side of the camera, which were threaded through tightly with an 8mm bolt.

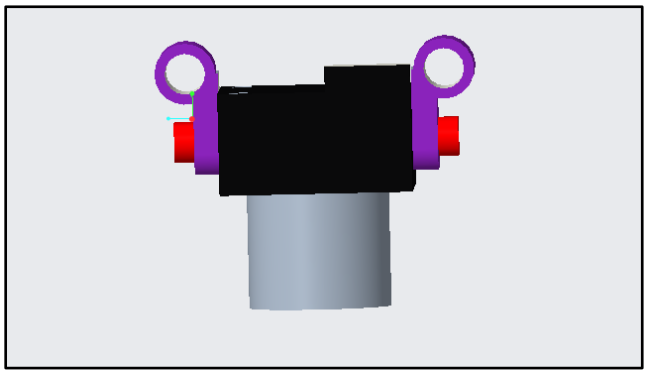


Figure 5.1.4.2.G: Camera with Side Mounts, Top View

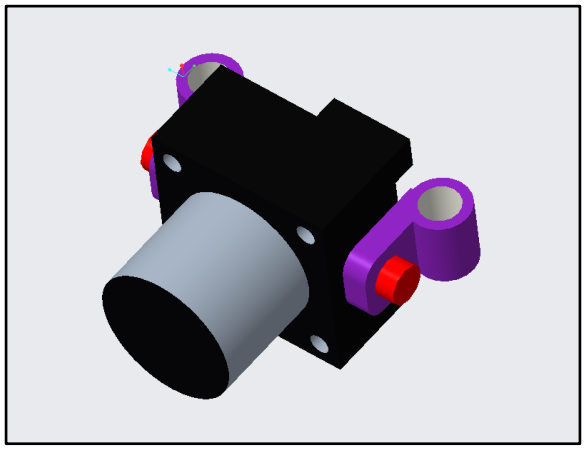


Figure 5.1.4.2.H: Camera with Side Mounts, Oblique View

As visible in the preceding images, two adjustable, purple side mounts came with the drone camera and offered an excellent platform for connection and camera viewing angle adjustment. All that needed to be designed was a fixation point to the purple side mounts, while giving enough space for the camera to be tilted to the desired viewing angle. This led to the first iteration, visible below.

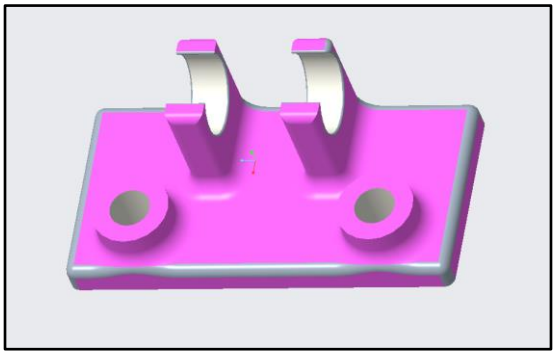


Figure 5.1.4.2.I: TCM Camera Mount, Iteration 1
Volume = 3.236 cm^3
Weight = 2.80g

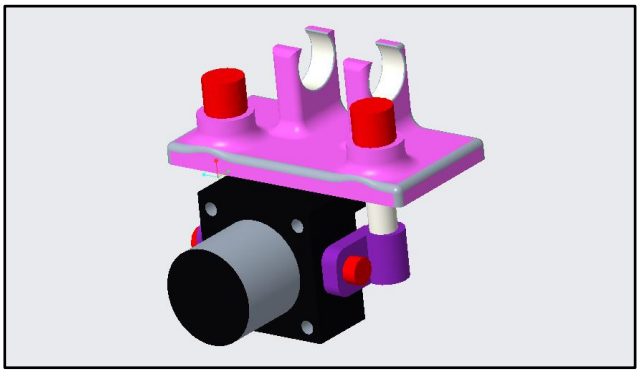


Figure 5.1.4.2.J: TCM, Iteration 1

With the main concern being weight reduction and minimalism, this design met the attachment requirements. Using the least material possible, this design theoretically would clip onto the frame rail. However, upon first installation, the rail clips snapped, and a further revision was necessitated.

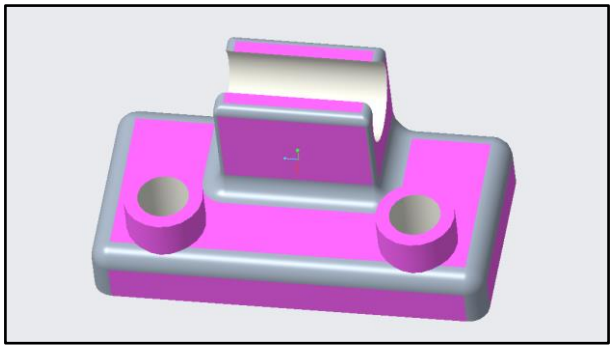


Figure 5.1.4.2.K: TCM, Iteration 2
Volume: 5.868 cm^3
Weight: 5.074g

This iteration received two major updates - one was a full-length snap-on connection point to the rail system, and the other was an increased overall thickness to the bracket for increased rigidity. This piece also snapped upon installation on the rail, and necessitated a design overhaul. A more secure, durable piece was needed to retain the camera against the frame.

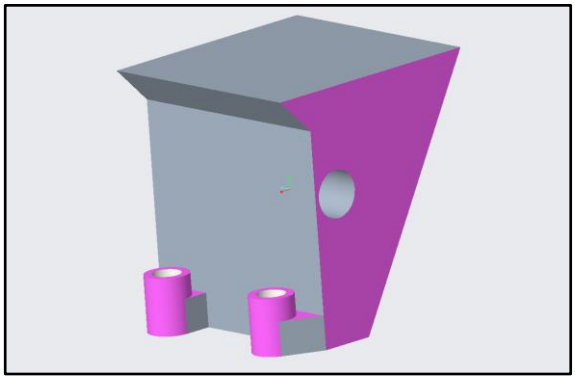


Figure 5.1.4.2.L: TCM, Iteration 3
Volume: 40.154 cm^3
Weight: 34.7131g

Providing secure retention on the camera mount, the rail, and additionally a secure mounting against the frame, which reduced instability as the mount rotated around the rail, this iteration provided the foundation for the final camera mount design.

This came at a drastic weight increase, however, as the new part weighs 34.7g vs the original 2.8g on the first iteration. While the design met the basic requirements providing camera adjustment and a firm connection, another redesign was necessary in order to cut down on unnecessary weight.

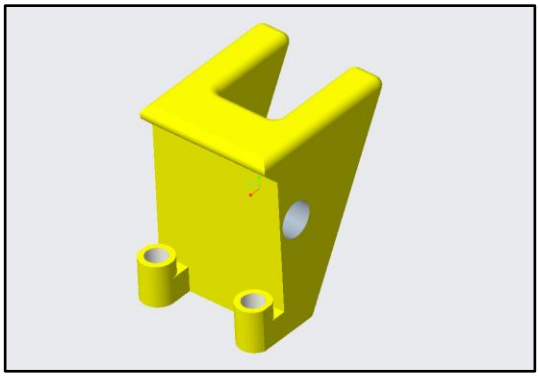


Figure 5.1.4.2.M: TCM, Iteration 4
Volume: 23.577 cm^3
Weight: 20.38g

The updated design cut out on the middle section as well as round off corners to reduce stress concentrations. The overall weight savings in this iteration vs the previous is 41.3%, while still providing adequate camera mounting and control. Even still, a further redesign was necessary to remove more unnecessary material and reduce weight.

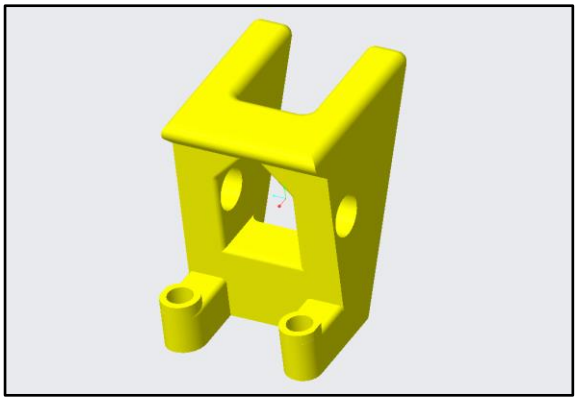


Figure 5.1.4.2.N: TCM, Iteration 5

Volume: 22.01 cm^3
Weight: 19.03g

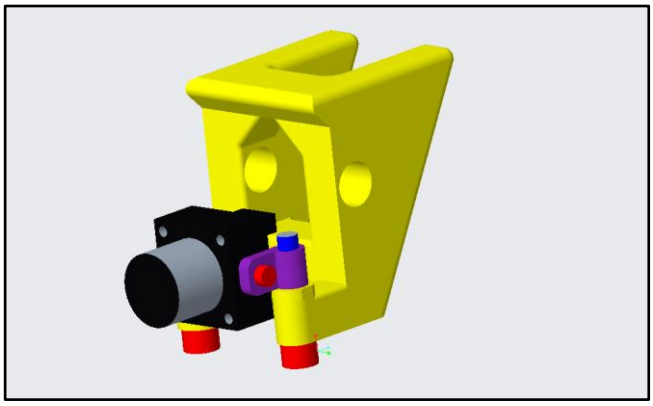


Figure 5.1.4.2.O: TCM with Camera and Side Mounts

The final iteration of the TCM provided two new design features, while slightly reducing the overall weight of the part. The first is allowing for even more adjustability of the camera lens with extended and further extruded camera side mounts. This update removed the camera body itself from colliding with the TCM during adjustment. The other design change, by removing unnecessary material in the center of the part, the cables running from the camera to the flight controller could be routed through the central passage in the part. This was the final design for the TCM and met the requirements of a firm mounting platform with an adjustable viewing angle.

Tactical GPS Mount (TGM)

The Tactical GPS Mount was the only electronic device that had a design requirement which directly altered its functionality. Due to interference and connectability issues from satellites, these devices must have at least 5cm from any solid body or emitting electronic component in order to avoid jamming the signal.

Consequently, the only requirement for this component was extending it from the body of the drone by 5cm. The TGM is simplistic in design and consists of three parts, two of which are 3D printed using PLA, and the other which is cut out of aluminum.

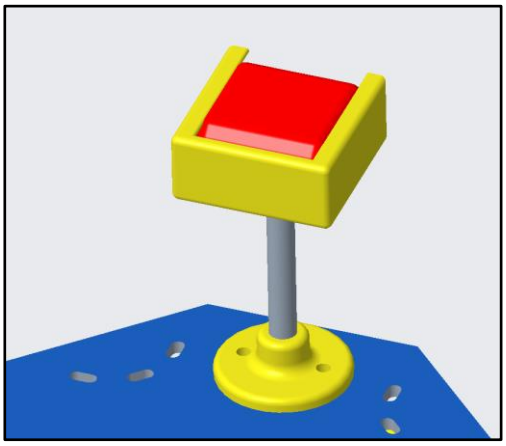


Figure 5.1.4.2.P: TGM, Iteration 1
Total Weight: 33g

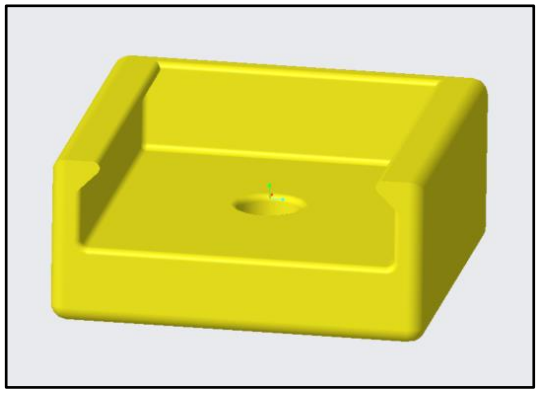


Figure 5.1.4.2.Q: TGM Retaining Block, Iteration 1

This design was simplistic in nature. The retaining block functioned with a friction fission, which was contoured around the top sloping edge of the GPS module (pictured in red above). An aluminum rod was inserted into both the top and bottom holes, successfully separating the GPS module from the drone frame by 6cm.

Battery Holder Landing Gear (BLG)

The Battery Holders and Landing Gear were the most complicated subsystems to design, and eventually merged into one overall design, incorporating the heaviest individual components (at 200g a piece) into the landing gear. This improved stability by lowering the center of gravity, and using these rigid structures (batteries are commonly inflicted with high impact forces during FPV race crashes) as a damped landing for takeoff and landing. The only risk to the battery is from punctures from the underside.

This subsystem consists of three components: the top portion, which must firmly connect to the rail system and the top side of the battery, the lower portion, which must provide a secure

mountain surface to the battery and protect against punctures, and an elastic strap, which connects all three pieces to the battery, providing a firm, yet elastic fit.

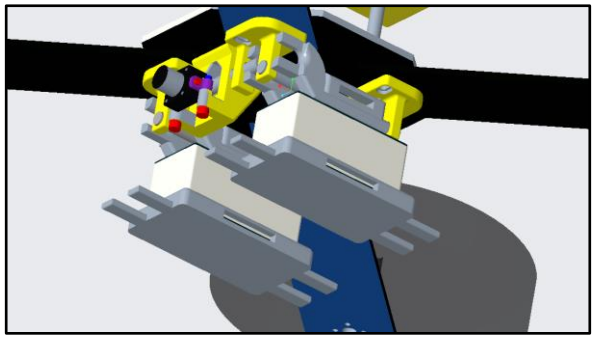


Figure 5.1.4.2.R: BLG Iteration 2 Mounted on Drone

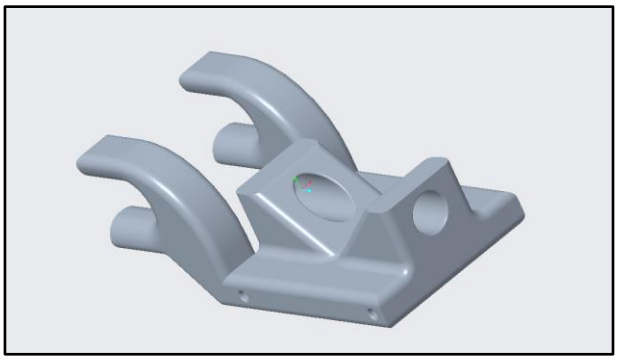


Figure 5.1.4.2.S: BLG Upper Mount, Iteration 1
Volume: 27.985 cm^3
Weight: 24.2g

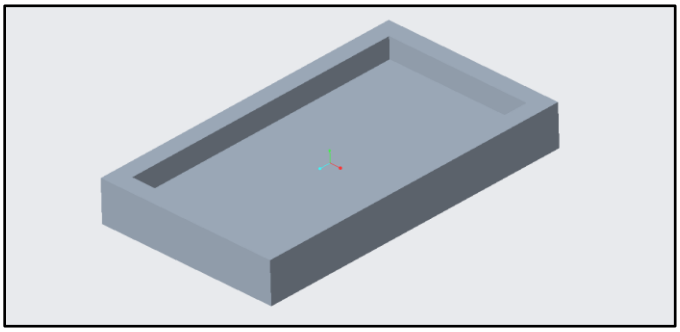


Figure 5.1.4.2.T: BLG Lower Mount, Iteration 1
Volume: 37.356 cm^3
Weight: 32.3g

The first iteration design did not adequately mount to the drone battery and frame. Although the upper mount was inserted into the rail, the front ‘claw’ portions were too wide to correctly fit into the perpendicular rail, as visible in the Overall BLG Mount Figure above.

Additionally, the top portion could not connect to the battery via strap, and would not secure the assembly adequately. The lower portion also needed revision as the battery strap would not snug around the bottom side of the mount. Furthermore, the bottom portion was not stable, and the drone was unbalanced and toppled over with this mount.

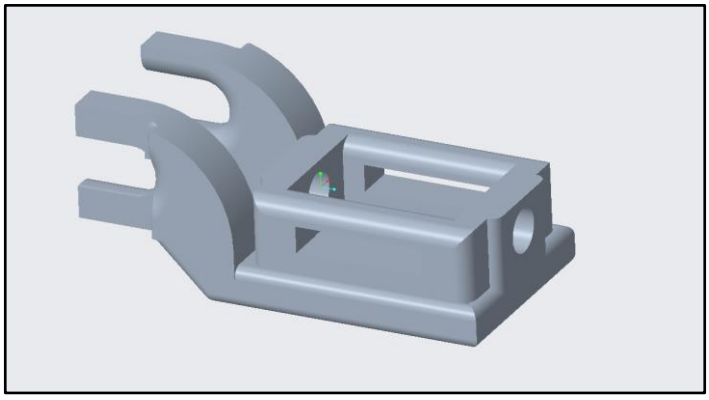


Figure 5.1.4.2.U: BLG Upper Mount, Iteration 2, Oblique View

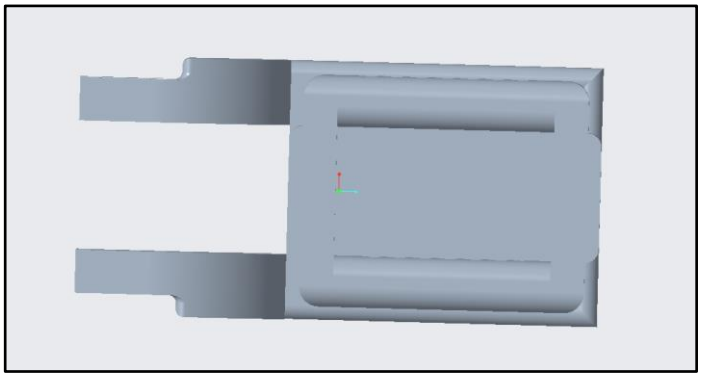


Figure 5.1.4.2.V: BLG Upper Mount, Iteration 2, Top View
Volume: 30.413 cm^3
Weight: 26.3g

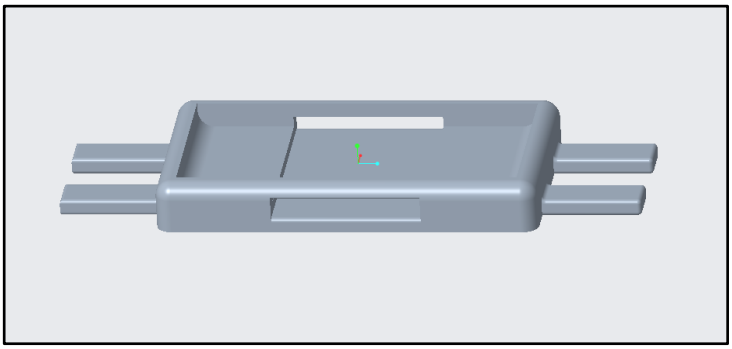


Figure 5.1.4.2.W: BLG Lower Mount, Iteration 2
Volume: 34.584 cm^3
Weight: 29.9g

With the second and final iteration, the critical design criteria of the BLG mounts were realized. These criteria were creating a retention system for the batteries as well as a platform for take offs and landings, which the included slots in both the top and bottom mounts included, as well as the extruded 'feet' on the lower mounts. The overall weight reduction from iteration 1 to 2 was 0.6g for both mounts, while improving the design.

5.1.4.3. Final Drone Design and Weight Map

Having developed the TRS (Tactical Rail System), TCM (Tactical Camera Mount), TGM (Tactical GPS Mount) and BLG (Battery Landing Gear), the physical SAR vehicle is now complete. A complete model is displayed below, with the 8 inch propellers as the shaded regions above the motors.

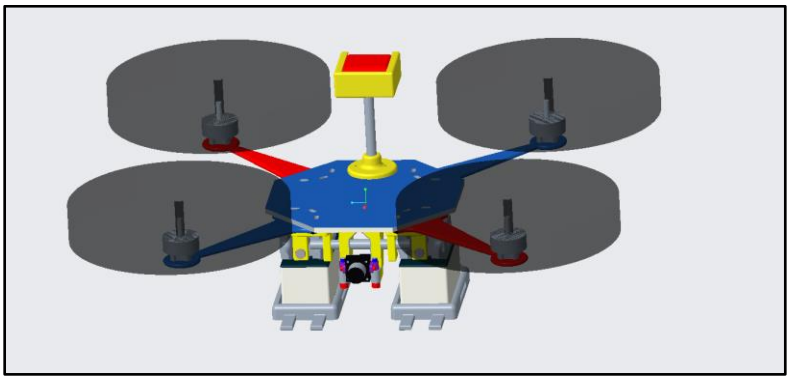


Figure 5.1.4.3.A: Final Drone Frame
Weight: 1234 g

Component	Initial Weight (g)	Final Weight (g)	Weight Reduction (g)
Carbon Fiber Frame	387.377	168.177	219.2
TRS	67.4	43.6	23.8
TCM	43.3	27.6	15.7
BLG	520.6	520.6	0
TGM	33	33	0
FC	11	11	N/A
ESC	15	15	N/A
Motors	158.8	158.8	N/A
RF transmitter	15	15	N/A
VTX transmitter	6.5	6.5	N/A
propeller	9	9	N/A
Misc	225.7	225.7	
Total:	1,492.7 g	1234 g	258.7 g

Table 5.1.4.3.B: Weight Map

5.1.5. Conclusion

Overall, the preliminary research and development for this project was extensive. From the initial frame design evaluation for X and H-frames to additional cargo systems capable of carrying critical mission components and additional payloads for various missions, much was explored in this section of the project.

This drone frame shape and size needed to incorporate the extra space for their corresponding components, and not interfere with three different signals being transmitted at the same time (RF telemetry, VTX, and GPS). With 8-inch propellers on a 350mm frame and an estimated flight time of 15-20 minutes with current components, size is not a limiting factor. In the potential setup of the H-frame (as seen above), having all the electronic components in the middle between two batteries, interference can be an issue. In the X-frame, this is not an issue, as the batteries will be mounted below the aircraft, and if a modular cargo system is implemented, space can be freely optimized for drone performance.

Another critical requirement was drone stability. This refers to not only stable flight, but providing adequate footage for the CV model used in the mission to find the lost victim. While the H-frame might have higher moments of inertia, the overall stability is best in the x-frame,

which is perfectly centered and restricted from acrobatic movement. In the development stage, this symmetry was exploited and led to a weight-saving effort to shorten the arms without affecting mission capability.

Finally, given additional cargo, the design must be modular and able to adapt to the mission. If a 1lb payload is added to the drone for supplies for the victim, or an additional device in order to track their location while a secondary team comes in to rescue them, the drone should be able to adapt. The X-frame that was developed incorporated a TRS Tactical Rail System, which expanded its modularity and mission potential with a plethora of attachment points for additional hardware and cargo.

Overall, the X-frame provides the most adequate platform for our design. The size, stability and modularity gave way to make the necessary design changes in order to maximize our drone's mission readiness. Initially with an overall extra weight of 420.21 grams onboard the frame, this number was reduced to 185.91 grams, and each subcomponent was tailored to more efficiently meet its design requirements. This is an overall weight savings of 234.3 grams, which equals 55.8% through updated designs and unnecessary weight reduction efforts.

The manufacturing methods were primarily 3D printing and carbon fiber material removal using hand tools. The reason for this was due to availability and ease of use. 3D printed parts can be replicated inexpensively, using standard PLA filament, and carving carbon fiber by hand allows for more precision in making design changes. Overall, it is preferred to purchase a pre-made carbon fiber frame, as all hardware mounting holes incorporate better quality control and are more precise, as well as generally cheaper price than buying carbon fiber sheets. Time savings and health in carbon fiber working is a major advantage as well to purchase pre-made parts.

Future considerations would be weight reduction on center plate, as well as reducing the infill on 3D prints (the density of each print, ranging from 10 - 100% solidity). These efforts could further increase flight time efficiency, as well as remove unnecessary structures in each component. One method is to implement Generative Design Topology, which is a software designed exactly to automatically perform structural analysis on each part, and given a minimum volume, construct a structure that meets the design and test requirements. This software is computer generated and controlled, and requires zero user input to generate the most efficient design.

Another consideration in the future of this project is integrating the hardware so that flight obstacle avoidance could be utilized. Range detecting sensors can and have been mounted underside the drone as well as below the motor mounts on each arm, and offer incredible feedback and control through difficult terrains. While the frame is currently small enough to maneuver in a moderately wooded area, having range detecting sensors would greatly help the agility and usefulness of this design.

A final consideration for future development of this project is the SWARM capability. The TRS rail system is designed to incorporate further hardware add-ons, and could easily integrate an onboard computer or electronic component that communicates with a 'Motherboard' style master computer in a tethered fashion to other drones. This platform, under $500, is adequately suited for future development in this area.

5.2. Location Tracking

5.2.1. Overview

GNSS has been used by military and first response organizations for nearly 30 years, and is therefore the logical choice for navigation on this project. We briefly considered using a simultaneous location and mapping (SLAM) algorithm for navigation, but we quickly realized that asking the drone to simultaneously orient itself and run the computer vision algorithms for victim detection was not feasible given our onboard computing and power consumption restraints. Furthermore, GNSS is already the industry standard, and our mission is to locate a victim and relay their location back to incident command. We are operating on the assumption that IC will already have at least rough maps of the area that is being searched, making the mapping function of a SLAM algorithm redundant and the entire thing unnecessarily complicated.

We decided to use an onboard GNSS (global navigation satellite system) because it is the most efficient means of determining and controlling the location of the drone and any items of interest that the drone locates (ideally the victim). In the event of catastrophic failure, the GNSS will provide the coordinates of the downed drone, facilitating retrieval of the technology. It allows the drone to be controlled via a series of waypoints that are generated based on the user-defined search area, and to fly directly back to its base station at the conclusion of its mission or in the event of emergency. Additionally, having onboard GNSS allows the drone to hover steadily in one location, which could be particularly useful if the computer vision algorithm requires additional data from a particular part of the flight path.

In general, a GNSS unit is actually nothing more than a fancy receiver. Radio signals are constantly broadcast from a global system of orbiting satellites, and the GNSS unit detects these signals and uses them to determine its location. This link with the satellite system is called a "fix", and provides positioning, navigation, and timing data to the drone. A reliable fix requires a connection to at least satellites for triangulation purposes, and a fourth is preferable for accurate timing data. The more satellites the GNSS unit can connect to simultaneously, the more accurate the fixed data.

Several countries maintain satellite constellations (sets of satellites that orbit independently to provide global coverage), and some GNSS units are limited to using satellites from one or another constellation. The Global Positioning System (GPS) is ubiquitous in the United States and is maintained by the federal government. Many GNSS units sold in the United States use GPS either exclusively or in addition to one of the other constellations. Other common constellations are the BeiDou Navigation Satellite System (BDS), run by the People's Republic of China, the Galileo GNSS, operated by the European Union, and GLONASS (Globalnaya Navigazionnaya Sputnikovaya Sistema), the GNSS of the Russian Federation. India and Japan also maintain their own constellations, but they are less widely used outside of those countries.

Each portion of the fixed data is used by the drone in various ways. An accurate position is the beginning of every command the drone requires – without knowing where it is initially, it is impossible for the drone to find a path to where it is supposed to be, or to know when it has arrived. Navigation is achieved via waypoints that are generated when the optimal route between the current location and the goal location is determined. Once the drone arrives at the intended search area, waypoints allow the drone to search the area in an efficient manner.

Timing data is used in this application to send timestamps along with the video footage. The video footage is being processed on a ground computer, and it will take time both for the footage to reach the computer and for the computer to process it. Without accurately timestamped footage, the computer vision algorithm would find a victim and trigger its alert, but the drone would have already moved on from the location where the footage was shot, so the coordinates provided to the user would be out of date. Depending on how long the computer takes to process the data and how fast the drone is moving, this discrepancy could be dozens of meters, and in difficult terrain it could jeopardize the ability of ground parties to find the victim. By timestamping the video footage with the GNSS data, the software will be able to determine the time at which the footage that triggered the alert was shot, and work backwards to determine where exactly the drone was when it took the footage, greatly increasing the accuracy of the victim position data returned to the user.

The data from the GPS is displayed as NMEA sentences, which consist of a set of values, separated by commas. The sentence is started with a "$" and terminated by a checksum, as shown in Figure 3. NMEA stands for National Marine Electronics Association and is the standard for GPS data transmission. The use of a marine-based language stems from the first use of GPS, when it was developed by the military in 1973 as a means to accurately determine the position of its submarines so it could accurately aim the long-range missiles that were fired from them. Most navigation and marine-use electronics communicate in NMEA sentences to ensure compatibility across platforms.

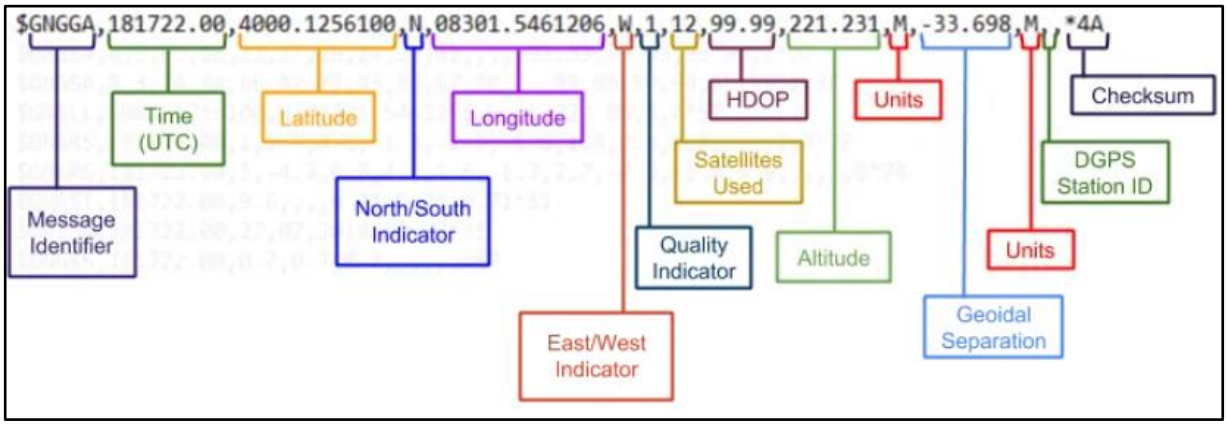


Figure 5.2.1.A: NEMA Sentence Structure

The first word in a message is always preceded by the "$" character, and is referred to as the message identifier. This word denotes which satellite constellation is providing the fix**.**

The next word in the message is the time the message was received, and is in the form hhmmss.ss, so 113543.15 would be 11:35:45.44 a.m. Time is always reported in Universal Time Coordinated by the satellites and must be converted into a useful time zone if the current time is to be displayed. For most purposes utilizing the time data, it is sufficient to leave the data in UTC, as we are more concerned with time elapsed than absolute time.

The next four words concern location – latitude, north/south indicator, longitude, and east/west indicator. Latitude and longitude are given in DDMM.MMMMM, where D stands for degree and M stands for minutes. The north/south indicator is a single letter denoting whether the position given for latitude is north or south of the Equator (N or S), and the east/west indicator is a single letter denoting whether the position given for longitude is east or west of the Prime Meridian (E or W).

The next three words communicate the quality of the data, beginning with the quality indicator, a quantitative representation of how good the data is. This value takes into account the geometry of the satellites at the time of the fix, and how long ago the data received from the fix was collected. It then weighs that information against how likely those factors are to affect the accuracy of the data, and the result is the quality indicator. The values vary between 0 and 5, with higher numbers indicating higher quality data.

The next number indicates how many satellites were used in the fix, with more satellites yielding more accurate information. Next is a value for the horizontal dilution of position, which is an indication of how the orientation of the satellites used in the fix may affect the accuracy of the

data. Similarly, to the quality indicator, these values range from 0 to 5, which higher numbers indicate higher reliability.

The next four words give information about the altitude of the receiver. The altitude is given as a number, and the next word indicates the units used (usually meters or feet). The next word is the geoidal separation, which is the difference between the Earth ellipsoid (a smoothed model of the world that GNSS systems use to approximate the shape of the Earth) and the actual height above sea level at a given location (altitude). The letter immediately following the geoidal separation is another unit indicator.

The next two words are the age of any corrections made to the data and the station ID of the station that made the corrections, if any were made. If no corrections were made, these words are omitted.

The final word in the NMEA sentence is the checksum, which is a hexadecimal representation of the sum of all the characters in the sentence. Each sentence starts with $ and ends with * (before checksum), so the checksum is the sum of all characters between these two.

5.2.2. Specifics of the Adafruit GPS

For this project, we decided to go with the Adafruit Ultimate V3 GPS because it has a number of features that make it the best choice for this application. First of all, it is inexpensive – only about $40. It can be easily mounted on a breadboard for ease of testing and connects easily to the Arduino unit I am using for now (until we have all of the components selected for the rest of the project). It can be wired directly to the onboard power supply and only draws 20mA while in operation, so its power consumption is minimal compared to some of the other components. It also has the option to use a backup battery, so if power is lost for any reason the data it has stored does not get lost as well. It has a decent onboard antenna and also features u. FL connector for attaching an active external antenna, so if we encounter a configuration issue where the GPS needs to be placed in a location in the drone where its ability to get a solid fix, we will be able to add an external antenna to improve the accuracy of the GPS. Finally, the Adafruit GPS can be connected via breadboard and breakout pins without the need for soldering, which simplifies our build process.

Figure 5.2.2.A: Adafruit Ultimate V3 GNSS Unit

The Adafruit GPS comes with a number of breakout pins that can be used to wire the unit in specific configurations depending on the needs of the overall design. These pins are located on the left side of the unit, as depicted in Figure 1. The top pin is a clean 3.3V output, which is useful in applications where the GPS unit needs to act as a conduit and allow power to transfer to other components, as will most likely be necessary on our drone (pending final wiring diagrams). The EN pin is the enable pin, and can be used to enable or disable the entire GPS unit. Unless this pin is used in conjunction with the VBAT (battery backup) pin, disabling the GPS will result in a loss of fix.

The RX and TX pins are for data input and output, respectively, and both communicate with 9600 baud data by default, which is easy to configure with the Arduino. Another convenient

feature of the Adafruit that is not standard on some other GPS units is the FIX pin. This pin both provides data to other components about the status of the current fix, and also toggles an onboard LED light to visually signal whether the GPS currently has a fix. While the feature won't be useful after the GPS is installed on the drone, at this point in the process being able to glance at the unit and know whether it has a fix is incredibly helpful.

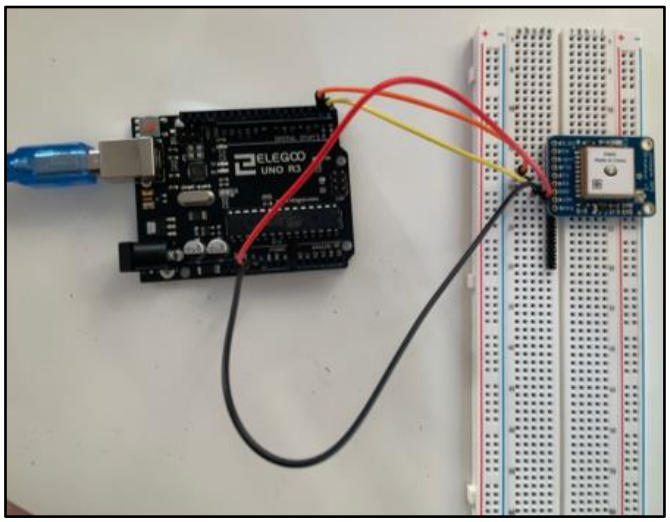


Figure 5.2.2.B: GNSS/Arduino/Computer Connection

The pins and the included breadboard connector piece make the Adafruit very easy to wire to an Arduino unit for communication with the computer, as shown in Figure 2. The four wires shown

here connect to the power in, ground, input, and output pins to their respective ports on the Arduino. When the unit is switched on, data from the GPS unit is displayed on the screen.

Alternative Modules

Some alternatives besides the Adafruit GPS Module are listed below with their accompanying specifications:

Matek M8Q-5883 GPS Module

- Constellation: GPS, GLONASS, SBAS, QZSS
- Dimensions: 20x20x10mm
- Weight: 7g
- Connections: Soldered
- Accuracy: Horizontal Position – 2.5m; Velocity – 0.05m/s

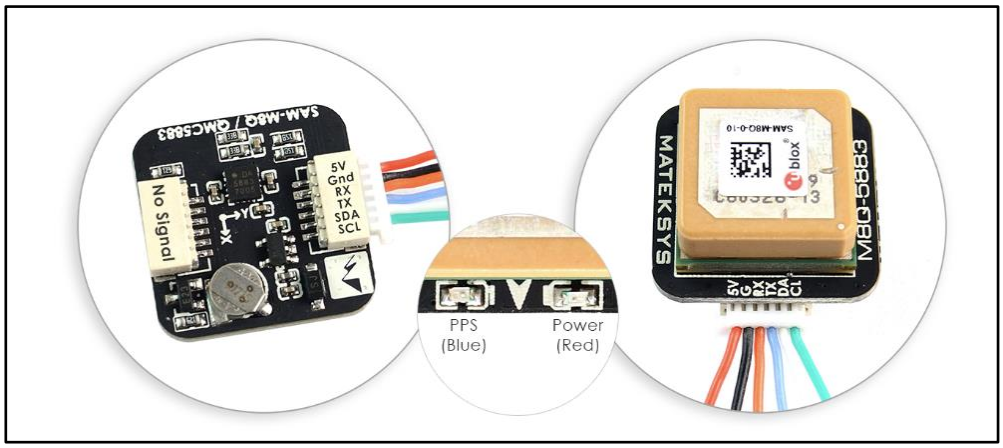


Figure 5.2.2.C: Matex M8Q-5883 GPS Module

Adafruit Ultimate V3 GNSS

- Constellation: GPS
- Power Draw: 20mA during navigation
- Dimensions: 25.5x35x6.5mm
- Weight:8.5g
- Connections: Pins
- Accuracy: Horizontal position - <3 m; Velocity – 0.1m/s

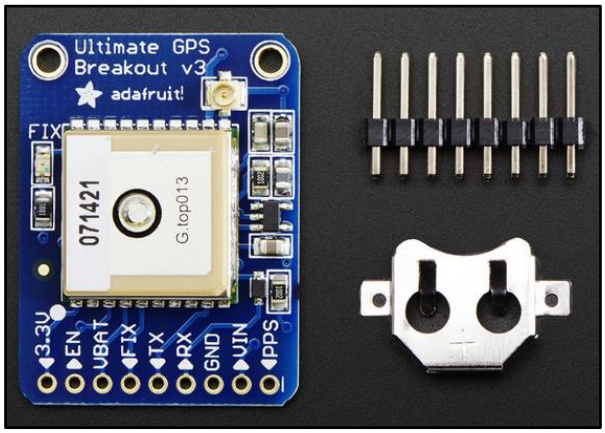


Figure 5.2.2.D: Adafruit Ultimate GPS v3

NEO-6M GPS Module

- Constellation: GPS
- Power Draw: 45A
- Dimensions: 27.6x26.6mm
- Weight: 3.46oz
- Connections: 4 pins
- Accuracy: Horizontal Position – 2.5m

5.2.3. Search Algorithms

Human search and rescue teams use a variety of search methods to scour an area for a potential victim, but the choice of which search algorithm to use is guided by the evidence at hand. In the opening stages of a mission, the team might have the last known position and intended destination of the victim, making it likely that the team will find the victim along that route. When such information is available, a simple hasty search along the route is usually the most efficient for finding the victim. Without this information, however, more in-depth search algorithms are required, involving deliberate search patterns to ensure that nothing is missed in the area being searched. Within these two extremes lie several other search patterns, all of which are outlined below [13].

Rapid Search Algorithms

Rapid Search algorithms check the immediate area, trails, roads, buildings, campsites, and specific areas of high probability.

1. **Linear Features Search:** Determine the most likely route missing person would have taken and cover this route (Hasty search)
    a. Useful for hikers, walkers

b. Usually performed by first responders
c. Follow travel aids (trails, signs, etc.)
d. Clue awareness is critical, especially at decision points

2. **Points of Interest Search:** Thoroughly cover specific area where the subject is likely to be found
    a. Check scenic overlooks, bathrooms, playgrounds, swim pools, bodies of water, other areas of interest

Segment (Area) Search Algorithms

Segment Search algorithms check an area defined by easily identifiable boundaries (topological features such as rivers, gullies, or field edges, or manmade features such as walls, fences, or roads), in which there is a reasonable probability the victim will be found based on intended route and last known position.

1. **Route (Area) Search:** A systematic search where teammates follow tracks parallel to a side boundary and maintain and predetermined separation
    a. Depending on available manpower, more than one pass may be required
    b. Passes are done in straight lines, maintaining uniform separation and providing uniform coverage of the area
        i. Tighter spacing improves coverage but costs more in terms of manpower
        ii. Spacing may have limitations due to density of vegetation, terrain complexity, or other environmental factors.

    c. Boundaries should be easy to recognize and follow whether natural or man-made
        i. Boundaries might be set up by the search team using cones or flagging tape

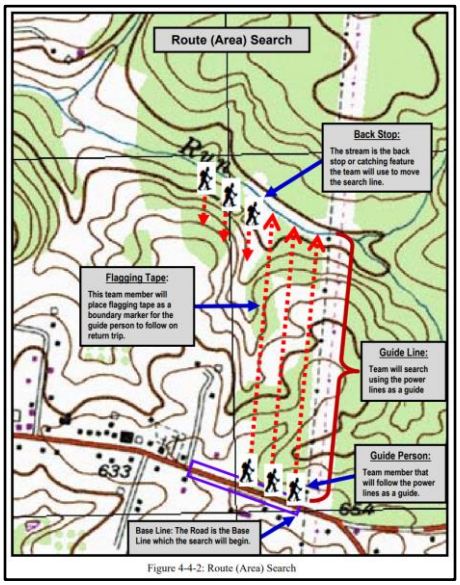


Figure 5.2.3.A: Route search along a straight, man-made boundary line

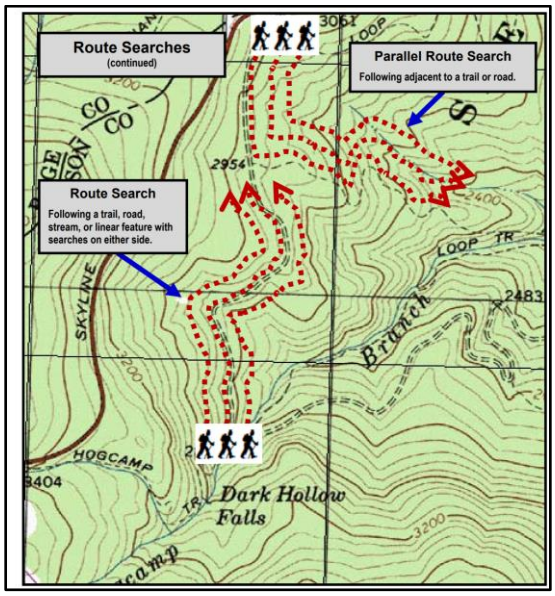


Figure 5.2.3.B: Route search along a varying natural boundary

2. **Area (Grid) Search:** More deliberate search used to increase the probability of detection for a victim or other evidence
    a. Similar to a route search but with tighter spacing
        i. Often used to locate evidence at a crime scene or an unresponsive subject
        ii. Searchers may follow a compass bearing instead of an area boundary
        iii. Searchers must look in all six directions – forward, backward, left, right, up, and down

b. May include “purposeful wandering” – leaving the grid in pursuit of a likely find

c. “Tight grid searching” is even more meticulous than the grid search, and used only as a last resort
   i. Involves more manpower for more intense search
   ii. Usually reserved for evidence collection in pursuit of things like shell casings and other small pieces

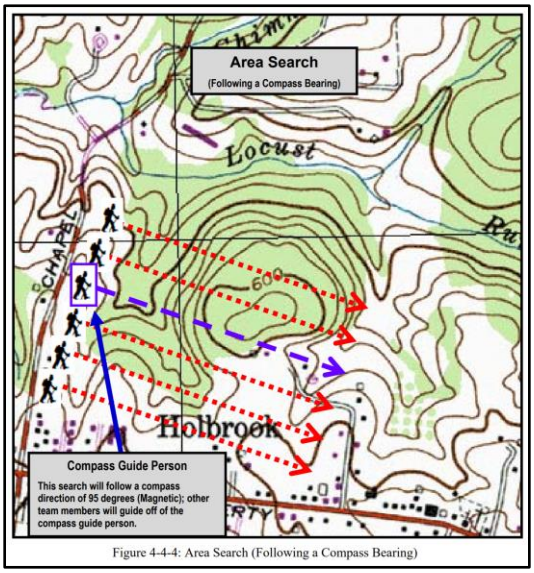


Figure 5.2.3.C: Area Search (Following a Compass Bearing)

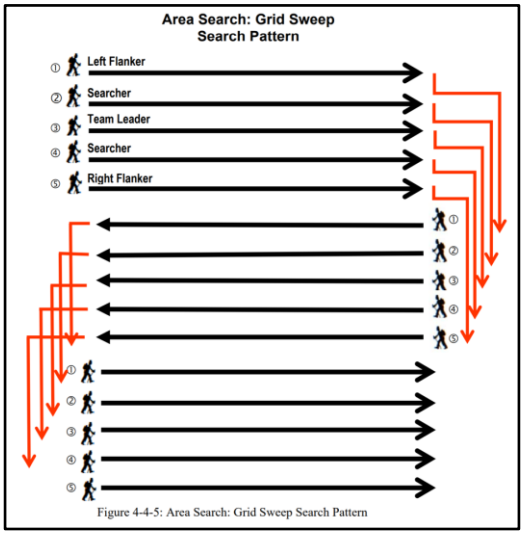


Figure 5.2.3.D: Area Search: Grid Sweep Search Pattern

3. **Sound Sweep:** Searchers alternate between moving, calling out for the victim, and listening for a response
    a. Covers a large area relatively quickly, but requires the victim to be responsive
    b. Must be carefully coordinated, as all searchers must move, call, and listen at the same time to avoid obstructing a possible response from the victim by their own movements or calls
    c. Particularly useful in environments where visualization of the victim is impaired, such as heavy vegetation or irregular terrain
    d. May involves whistles or other devices for noise making, on either the part of the victim or the searchers

4. **Expanding Circle Search:** Searchers begin at a logical point (last known position), and move outward in an expanding spiral
    a. Especially useful when the route of the victim is not known, or when a trail is lost and needs to be found again
    b. Only useful in small areas, but can be done with one or only a few people
    c. Searchers must be careful not to destroy tracks or evidence
    d. May be used following the contour of a hill, with the searchers beginning at the top and working downhill

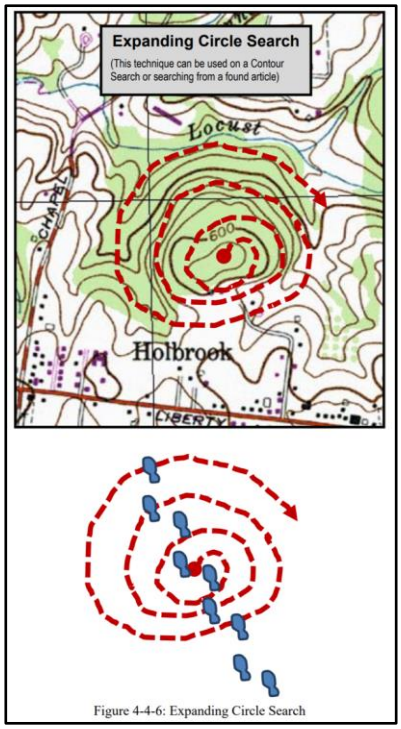


Figure 4-4-6: Expanding Circle Search

Figure 5.2.3.E: Expanding Circle Search

5. **Search Aids:**
    a. Dogs – commonly used as in conjunction with human searchers to provide scent tracking
        i. Usually start with expanding circle search, then move to a route search when the dog catches the scent

    b. Horses – allow human searchers to cover ground more quickly and from a higher vantage point
        i. Can be trained to alert to certain stimuli, such as the smell of blood
            1. Not trackers, but can help find a piece of evidence

        ii. Are more versatile in difficult terrain than ATVs or other land vehicles

    c. ATVs – can follow a trail quickly and carry additional gear
        i. Must be used judiciously, as they can obscure noises made by the victim or damage trails or evidence left behind

Ground-based search algorithms carried out by humans are all based on the evidence at hand. The facts of the situation at the beginning of the mission determine which algorithm to use, and evidence located during the search affects the choices made by incident management teams regarding how to proceed in the search. For our drone, it will be capable only of detecting the actual victim – it can't change its search plan because it sees the wreckage of a plane nearby, it will simply continue in its programmed pattern.

For this reason, it is important that we choose a search algorithm that is both efficient and complete. Fortunately, for human-powered SAR, completeness often comes at the cost of time and manpower, but the use of a drone eliminates many of the hardships faced by human searchers, allowing one drone to cover drastically more area than a human searcher. Furthermore, our drone will be looking not with human eyes, which need to focus on and interpret everything that is seen, but with a camera and a computer program capable of rapidly identifying the victim while weeding out everything else.

5.2.4. Explore the Nature of Air-based SAR Missions

In addition to the land-based SAR algorithms, we also looked at some air-based algorithms, used with planes or helicopters.

Air-based SAR missions typically involve a plane or other aircraft as the main SAR tool. Here are several ways that air-based SAR missions are conducted:

Rapid (Hasty) Search

1. **Route Search**: Search along intended route
    a. Follow intended route (of subject) from point of departure to intended destination (or alternative roads/paths)
    b. Can follow multiple passes (at different times of day, different directions
    c. These are usually expanded to "trackline pattern"

2. **Trackline Search**: Rapid search with multiple passes of intended route
    a. Start with route search, and two patterns possible:
        i. Single, non-return: searches either side of the track line
        ii. Single, unit-return: searches trackline in both directions combined with searching both sides of the trackline

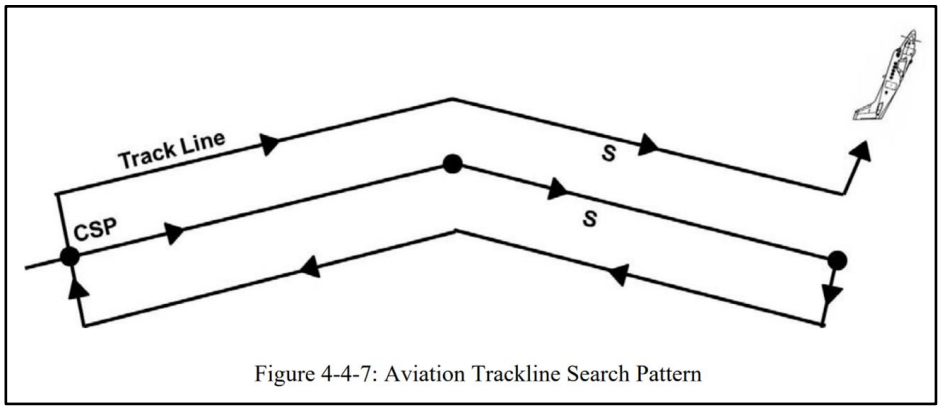


Figure 5.2.4.A: Aviation Trackline Search Pattern

3. **Electronic**: Rapid search with electronic direction-finding equipment able to detect distress beacons.
    a. Route search with DF equipment
    b. Range of detection varies with search unit altitude and / or terrain
    c. Specific coordinate search for activated distress beacon

4. **Contour**: Rapid search of a vertical area, following the contours of the environment.

a. Can be conducted with fixed wing, rotary wing, or UAV craft.
b. Starts at the base of an environmental feature (hill or mountain) and works up and around the feature.
c. Can follow a potential route up the feature more quickly than a ground team.

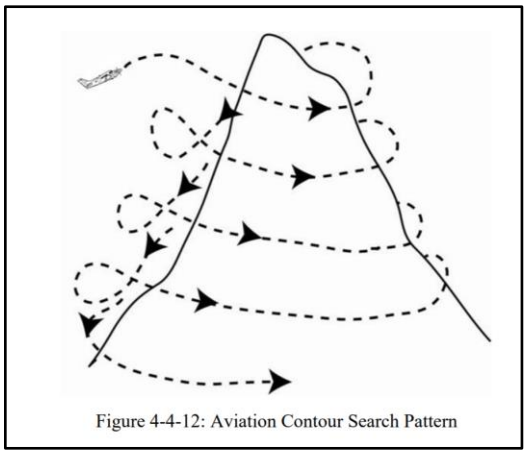


Figure 5.2.4.B: Aviation Contour Search Pattern

Area Search (Aviation)

1. **Parallel Search pattern**: Systematic search of a rectangular area with multiple passes
   a. "legs" of the search path (each longitudinal "sweep") are usually aligned to cardinal headings parallel to long axis of search area
   b. Pattern typically used for large, level search areas when uniform coverage desired (**our project…**)

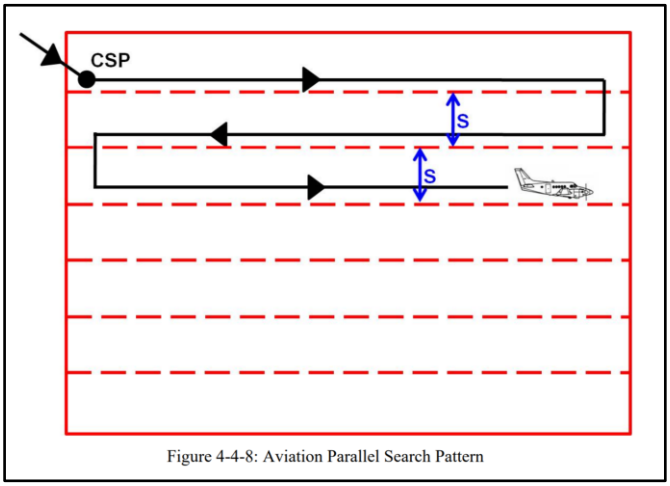


Figure 5.2.4.C: Aviation Parallel Search Pattern

2. **Creeping Line Grid**: Systematic search of a rectangular area that allows search at one end or adjustments for sun angle.
    a. Specialized type of parallel pattern where direction of creep is along major x-axis
    b. Used to cover one end of an area first
    c. Able to change search direction (due to glare, bad line of sight, etc.)
    d. Used to provoke a signal if subject has signaling devices (i.e., flare gun)

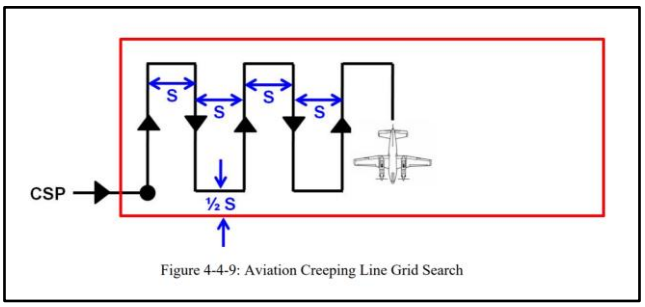


Figure 5.2.4.D: Aviation Creeping Line Grid Search

3. **Sector Search**: Extensive coverage of a center point at different angles and with multiple passes.
    a. Similar to expanding circle and square and used when search area is not extensive
    b. Concentrated over central point, with radius, r < 20 miles
    c. Both rotary and fixed-wing capable

4. **Expanding Square**: Thorough check around center point with rectangular legs.
    a. Starting point similar to expanding circle
    b. Pattern uses straight "legs" with turns only at corners of each square
    c. May require continuous reprogramming of navigation units
    d. Leg spacing 0.25 – 0.5 NM in missing person searches

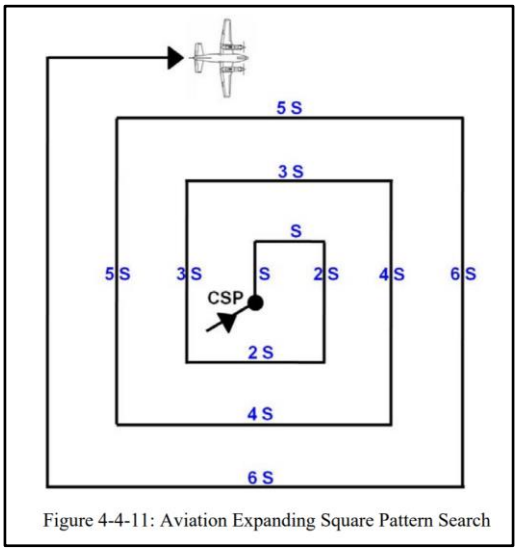


Figure 5.2.4.E: Aviation Expanding Square Pattern Search

Our particular use case involves searching an area approximately 150ft by 150ft, free of tall obstructions or significant environmental features. We chose to use the parallel search pattern because it allows the camera to see every inch of the search area while relying on minimal computing power to determine the specifics of the search route. It also allows for some variation in the shape of the search area chosen by the user, because even if the search area is more of a rectangle than a square, the same algorithm that determines the search route for a square can still be used to determine the route for a rectangle, and the parallel search pattern simply becomes a hybrid creeping line pattern with no further inputs.

5.3. Computer Application

5.3.1. Flask Framework: Advantages & Disadvantages

For the computer application we have decided to use Flask. When making this decision, we faced the following pros and cons:

Pros	Cons
Python-based framework (easy to create)	Complicated process to scale up

Flexible (when first created)	Not secure
Easy to run and complete tests	

Table 5.3.1.A: Pros and Cons of Selecting Flask Framework for UI

5.3.2. Flask Framework Conclusion

To go more into detail with the pros and cons, one of our greatest reasons for using the Flask framework is that it's written in python. This is a huge bonus because our computer vision models are all made on python so the compatibility between technologies would not be an issue anymore. Which leads me onto the next pro, Python is known to be one of the simpler languages to learn and execute, due to it being very near pseudo-code while also still being a very powerful language with many libraries to include and use [23].

Which leads into one of the cons which was that the front-end developer was not familiar with python at the start. This was an issue at first but since Python is a simple language to learn it ended up becoming a very easy problem to solve. Going back to the framework, it is also very efficient when it comes to testing because of its integrated support for unit testing.

A big concern regarding the cons of picking Flask as our framework was its scalability since it is a micro framework, meaning that it could possibly cause issues down the line if we wanted to work on this beyond senior design. Following this there is also the issue of security since the process is no longer between the web framework and the developer, because of the involvement of other third-party modules that could possibly be malicious.

5.3.3. Google Maps API

For the UI/UX we have decided to use the google maps API. This is to have some indication of where the coordinate points have been placed, in order to make sure that the drone is going over the correct area that the user wants it to go. The specific version of the google maps API we are using is the one that runs from JavaScript called "Maps JavaScript API" by Google maps platform.[24] The Google maps API has many features that are very easy to integrate into the application such as:

- Marker placement

- Terrain view
- Satellite view
- Elevation
- Map Localization

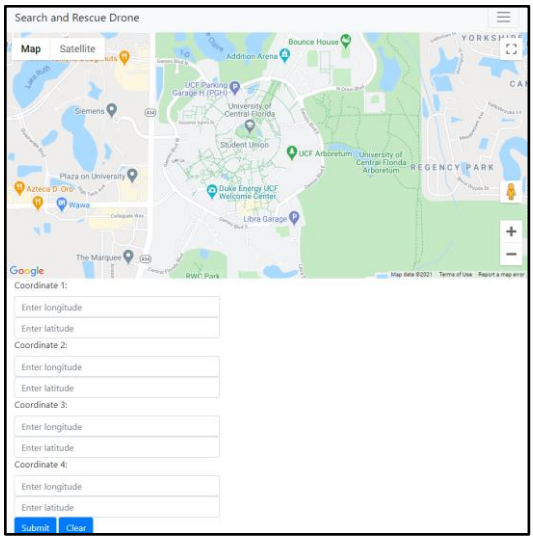


Figure 5.3.3.A: Google Maps API with Terrain View Enabled

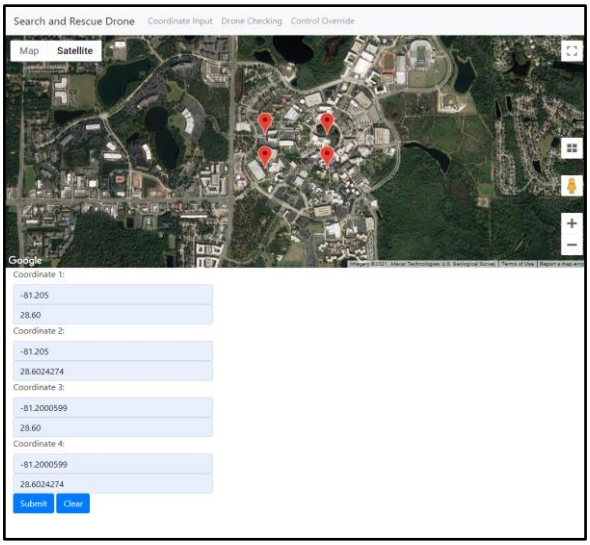


Figure 5.3.3.B: Google Maps API with Satellite View Enabled and Four Coordinates Implemented

In the above image (Fig. 5.3.3.B), the drone will search for a human within the space of the four coordinates placed on the map).

5.3.4. Waypoint Generation

After the user selects the coordinates within the UI, the program takes those coordinates and converts them into an exact rectangle to simplify calculation of the search pattern. To do this, the program takes the most extreme longitude and latitude coordinates and uses those to generate the search box, ensuring that the generated rectangle is the smallest possible shape that includes every inch of the user-designated area.

The algorithm then generates a parallel search pattern based on the altitude of the drone, the field of view of the camera, and the portion of the video feed that is most effective at successfully locating a human being. The first waypoint is the first corner of the search box. The drone then flies down the first side of the rectangle, before turning 90° and moving along the box edge a calculated distance, so that when the drone again turns 90° and flies back up the search box, the camera is seeing new terrain for the CV algorithm to analyze, with enough overlap to ensure the victim is not missed between passes. This pattern continues until the entire box has been searched or the victim is located.

5.3.5. User Interface (UI) Prototypes

Several UI web page prototypes for different aspects of the drone have been created and are showcased below.

Coordinate Input Page

Figure 5.3.5.A: Initial UI Prototype Web Page for User-Defined Coordinate Input

Function for the coordinate input page are as follows:

- Display a UI to the user so they know the area the drone will search
- Give users a place to input the four coordinates to search in between

- Gives users an option to select which drone to give instructions to (in case of future drone swarm usage)
- Allows access to other pages through the menu bar
- Clear option to reset all fields and start over

Drone Statistics Page

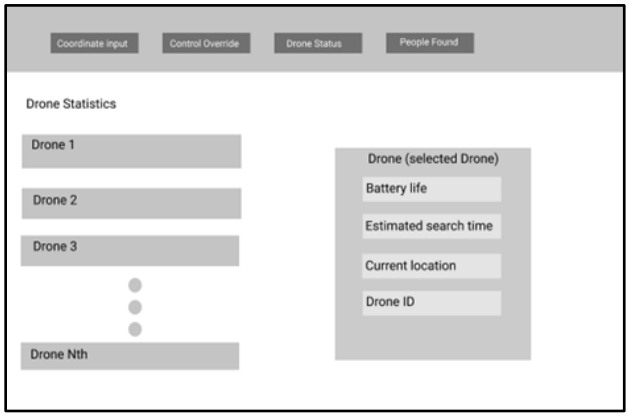


Figure 5.3.5.B: Initial UI Prototype Web Page for Drone Statistics

The functionality of the web page will be to select what drone statistics the user wants to pull up on the left. Then, on the right, the page will display the drone battery life, estimated time left in search, current location and drone ID to ensure mission success.

Individual Found Page

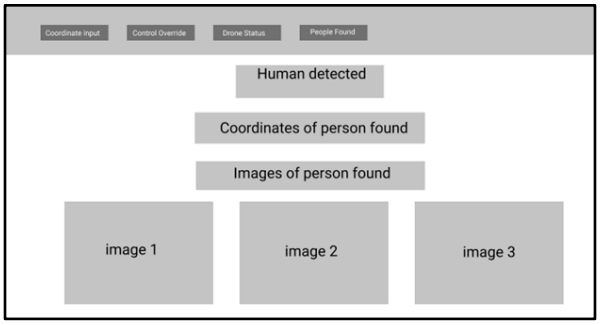


Figure 5.3.5.C: Initial UI Prototype Web Page for the Location of the Individual

The functionality of this page will be to display the coordinates of the detected persons, as well as images for human verification.

Drone Override Page

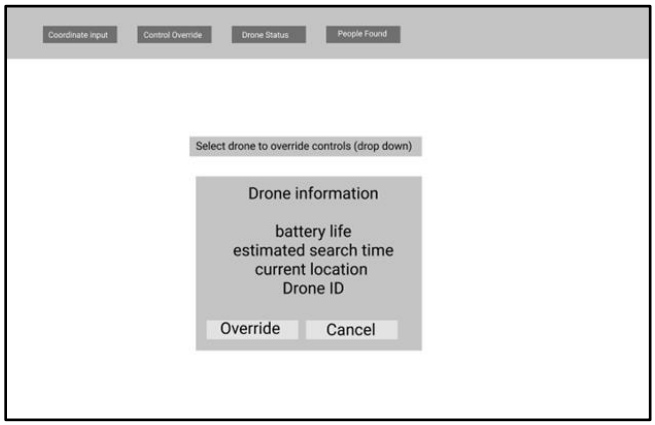


Figure 5.3.5.D: Initial UI Prototype Web Page for the User Override of the Drone

This page will ask for the specific drone whose commands are being overridden, and then it will show a confirmation box to ensure that current plans are to be overridden or to cancel the action, if necessary. The override commands that can be performed are as follows:

- Emergency land
- Retreat

5.3.6. User Interface (UI) Final Design

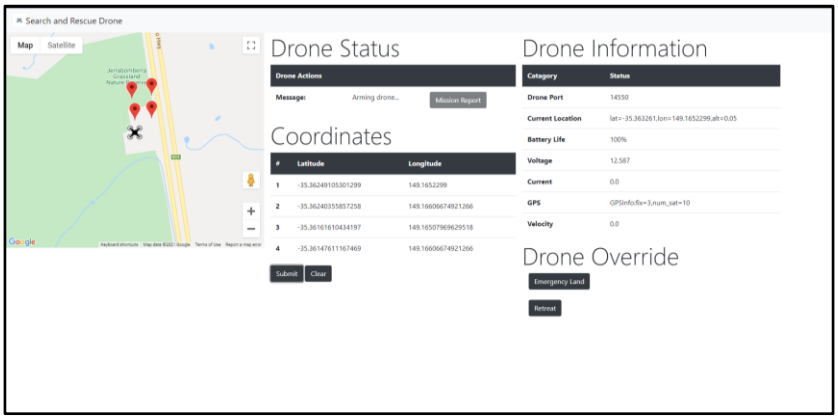


Figure 5.3.6.A: Final UI Design (Map View)

For the Final Design of Computer application we decided to go with a single page command center. With the new design we will only need the internet on the initial load up to see the Google maps API and then from there we can just click on the map itself to define the search area.

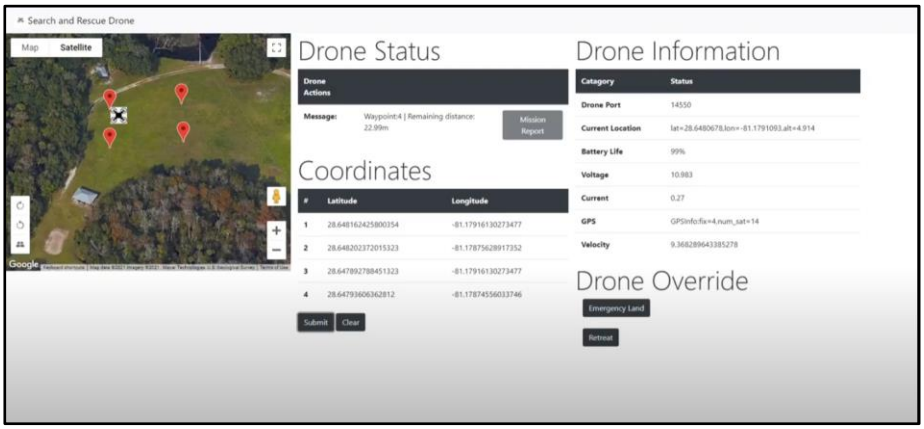


Figure 5.3.6.B: Final UI Design (Satellite View)

Along with that there was an added feature of terrain mode to have a better understanding of the layout of the land and where we want to search. Also there is not a drone icon that updates in real time with the location of the drone.

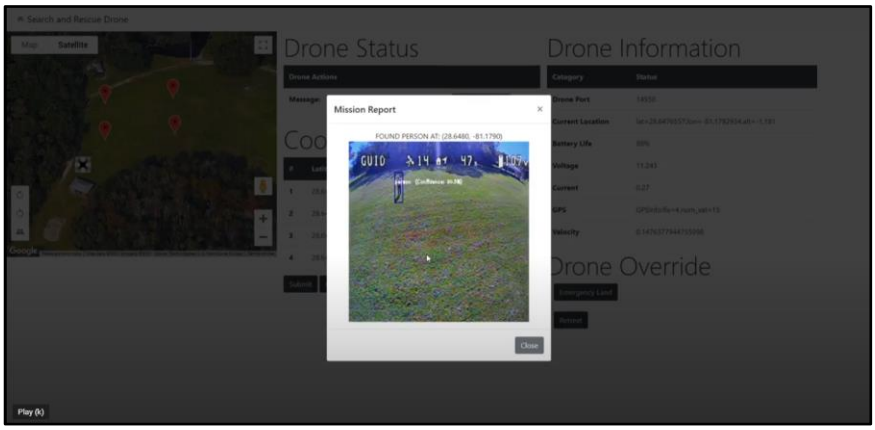


Figure 5.3.6.C: Mission Report Example

When the Mission is complete the drone will notify the user, and then show the mission report which would include the frame and coordinates of where the person was found. If no one was found there will just be a return to launch with no mission report since it was not successful.

5.4. Flight Controller System

A UAV (Unmanned Aerial Vehicle) is a complex device with many different components integrated into it. All these components must properly interface for the drone to fly flawlessly and to use all the peripherals connected to it. This section discusses the different parts selected for the project's flight controller system, hardware (computer systems) and software(programming), and makes comparisons with other part options. It will also discuss the functionality of the software and protocol used for the drone's operation. Below you will find a general list of the topics covered in this section, including:

- Sensors
- The Selected Flight Controller
- Companion Computer
- Software Stack

5.4.1. Sensors and Peripherals

Many of the peripherals connected to a drone are sensors. Sensors are needed in the system to be able to interpret the drone's surroundings. The information gathered from these sensors allows the drone to make appropriate decisions and interact with the world around it. This section discusses the different sensors and peripherals used in the Search and Rescue drone, as well as possible options to the selected parts. The types of sensors discussed herein are:

- Height sensors
- Range finders
- Accelerometer
- Gyroscope

5.4.2. Height Sensors

For this project the drone needs to be able to detect altitude. This is used to determine how close the drone is to the ground for landing as well as to maintain a steady height during flight. different kinds of sensor can be used to measure height such as:

- Barometer
- GPS
- Altimeter

This section will discuss the benefits of each part and will compare them.

Barometer

A barometer is a device that is used to detect air pressure. This is very useful since by determining the air pressure. It is used for weather forecasting by detecting changes in air pressure. This ability of detecting change in air pressure can be leveraged to detect height. As height increases, air pressure decreases. The inverse of this is: as height decreases air pressure increases, as Fig. 5.4.2.A below showcases.

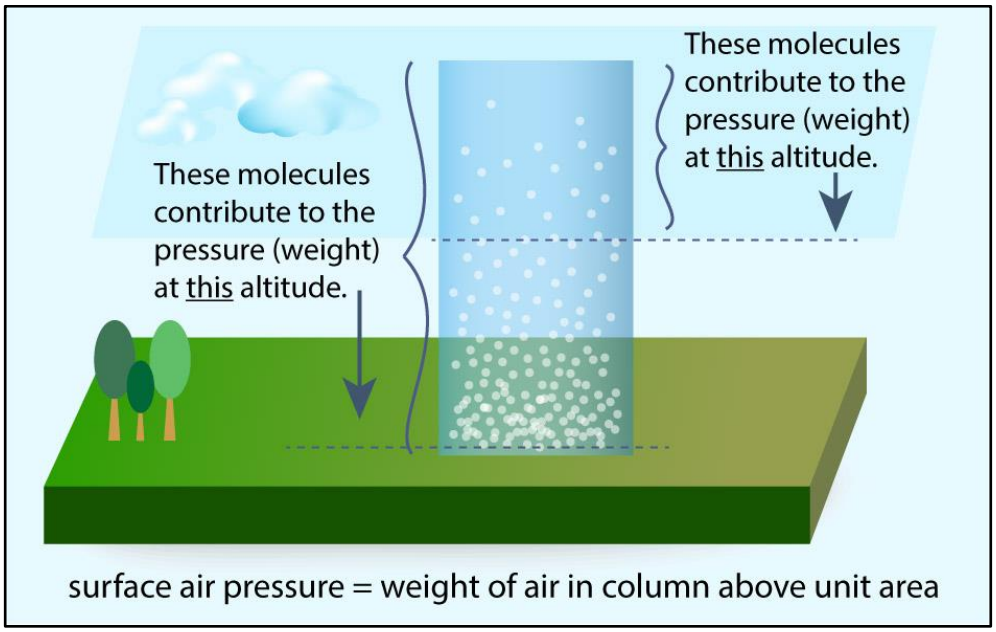


Figure 5.4.2.A: How the Density of Air Molecules Changes with Altitude

This makes a barometer very useful for the operation of the drone. By implementing a barometer, it can measure air pressure and it can be fed into the drone to determine if the correct height has been achieved for a mission. As mentioned earlier, determining height is very important for landing. As the drone approaches the ground it must slow down to avoid crashing and destroying itself. For height measurement the barometer considered for the project is the Bosch BMP280, as shown below in Fig. 5.4.2.B.

Figure 5.4.2.B: Bosch BMP280

Pros	Cons
Small size (Footprint: 2.0 × 2.5 mm², height: 0.95 mm)	Measurements can be affected by weather
Higher accuracy - measurement rate: 157 Hz	Needs to be calibrated
Can potentially complement GPS in positional accuracy	
Integrated temperature sensor	

Table 5.4.2.A: Pros and Cons of Bosch BMP280 as Barometer

Altimeter

Another option for measuring height is an altimeter. Altimeters function similarly to barometers in that they both measure atmospheric pressure. The difference between these two is that an altimeter is made to be used at different atmospheric pressures. Altimeters measure and compare

atmospheric pressure for determining altitude, while a barometer is supposed to be kept in a single location to measure small changes in pressure. Both equally work to determine height.

5.4.3. Measure Height: GPS Module

Alternatively, a GPS module could be added to the drone to measure height. Like a barometer, a GPS can measure height but with the benefit of not being affected by changes in air pressure. If a barometer is used without proper calibration in a zero-height reference point, it will not read the correct values. Adverse weather conditions can also affect it. A GPS module has no need for calibrations. It works by receiving a signal from satellites orbiting Earth which are used to determine position and height. The satellites send precise location and time data and the GPS receives and compares the time received with the time sent of the signals. This comparison, Time of Flight, is used to calculate the location on Earth [13].

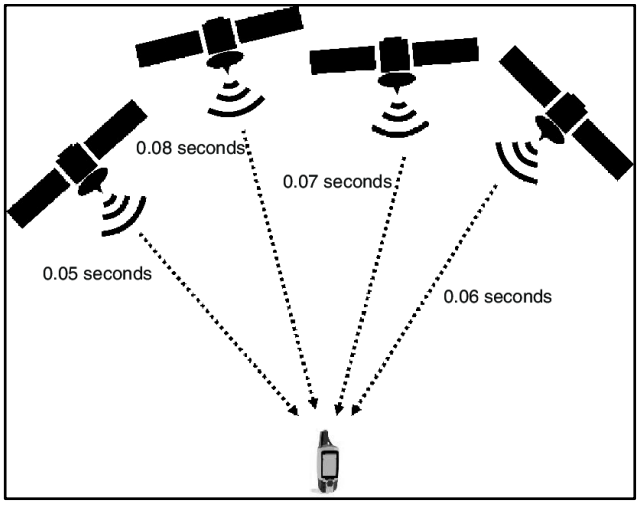


Figure 5.4.3.A: Functionality of a GPS Module

Since the project needs a GPS for autonomous flight A GPS module has been selected for the drone. The module used is FGPMMOPA6B. The Search and Rescue drone will receive a coordinate, travel to it, and then search the area. GPS technology will allow for the drone to transmit its location once it finds the target. It is also useful for determining the location of the drone so that it is not lost when out of sight. This is because the drone will fly without assistance and some missions require the drone to fly far from the GCS (Ground Control Station). In those situations, the GPS module will also be used to determine the location of the drone to see if it is on track to accomplish its task. Regardless of the inclusion of the GPS it is still worth considering the barometer for extra accuracy.

It is also important to identify the disadvantages of only using a GPS module for height detection. GPS modules can be affected by inclement weather. If the day is quite cloudy then the module will not be able to receive the satellite signals it needs to determine position and height.

To be able to obtain a good signal to determine height and location a GPS receiver needs four or more satellites.

Pros	Cons
Not affected by sudden wind gusts	Dependent on satellite signal
Already necessary for the drone	Buildings can block/affect signal
Can also provide drone location	Slower sample rate of 10 Hz
No need for calibration	
Tracks a maximum of 22 satellites for maintaining signal	

Table 5.4.3.A: Pros and Cons of Measure Height with GPS Module FGPMMOPA6B

5.4.4. Comparison & Selection of Height Detection Sensor

As mentioned above, the GPS model FGPMMOPA6B has been purchased as it is a requirement for the drone to function autonomously. It will permit the drone to compare its location to that of the target area and search the space within the perimeter. It will also allow for monitoring and detecting the location of the target if found. The GPS is able to measure height which is needed for setting a specific height for the search algorithm. But it is not as accurate since it has a 10 Hz sample rate. Its signal strength can also be affected by buildings that are near to the drone. This is why the barometer, BPM280, has been added to the drone.

While adding a barometer/altimeter to the drone would cost more and occupy space it is worth the investment. This is because it will complement the GPS by having a greater accuracy with a sample rate of 157 Hz. The space and weight added by the barometer BMP280 is miniscule enough that it will not affect the drone much. Another reason why the BMP280 will be added is

that it is integrated into the flight controller. This means there is no need to buy the module separately.

5.4.5. Inertial Measurement Unit

For navigation the drone will have an integrated Inertial Measurement Unit (IMU). The IMU will allow the drone's flight controller to determine orientation (rotation on an axis) and acceleration. Being able to measure these is essential for the drone to be able to fly by making corrections based on these measurements. This way the drone can stay stable while in flight. IMU is available in 6 DOF (Degrees of Freedom) or 9 DOF. Six DOF means that there are accelerometers and gyroscopes for each axis (x, y and z axis). A 9 DOF IMU adds a magnetometer to each axis. It helps understand each DOF that the IMU in the Search and Rescue drone would be measuring.

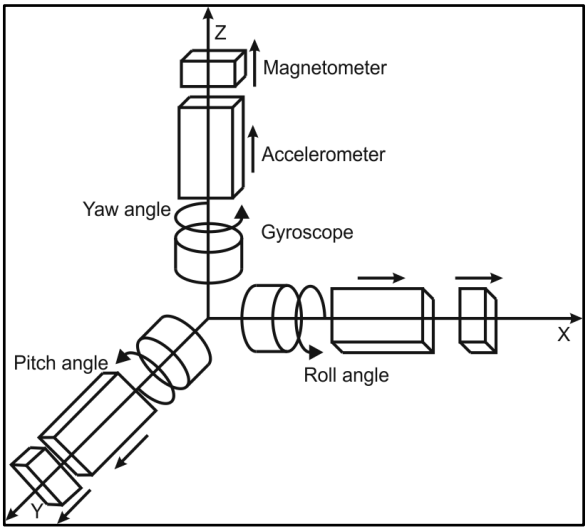


Figure 5.4.5.A: An Inertial Measurement Unit Body Coordinate System

For this project there is no need for a magnetometer. This is why a 6 DOF IMU has been selected. The IMU that will be used in the drone is the MPU6000 from InvenSense. The reason for this selection is mainly due to it being integrated into the flight controller selected for the drone. The flight controller meets all the project needs and has an IMU that satisfies the requirements.

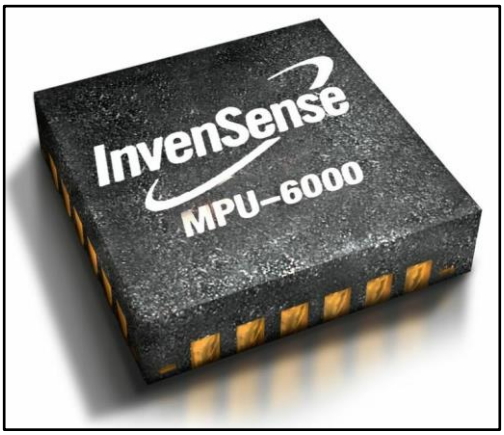


Figure 5.4.5.B: InvenSense MPU-6000 Sensor

MPU6000 Specifications

Gyroscope

- Triple-axis MEMS gyroscope
- Integrated 16-bit ADCs enable simultaneous sampling of gyros
- Digital-output X-, Y-, and Z-Axis angular rate sensors (gyroscopes) with a user-programmable full-scale range of ±250, ±500, ±1000, and ±2000°/sec
- Gyroscope operating current: 3.6mA
- Standby current: 5µA

Accelerometer

- Triple-axis MEMS accelerometer
- Integrated 16-bit ADCs enable simultaneous sampling of accelerometers
- Accelerometer normal operating current: 500µA
- Low power accelerometer mode current: 10µA at 1.25Hz, 20µA at 5Hz, 60µA at 20Hz, 110µA at 40Hz

Additional Specs

- 3.9mA operating current when all 6 motion sensing axes and the DMP are enabled
- QFN package dimensions: 4x4x0.9mm
- Digital-output temperature sensor
- 10,000 g shock tolerant

5.4.6. Range Finder

For object detections the drone will need a range finding sensor. This will allow the drone to avoid obstacles during flight. The range finding sensor can also aid in making precision landings. For the purposes of this project a sensor is needed that can work in close and medium range distances; that is approximately a range within 2.5 feet. The drone will be equipped with at least one sensor facing forward. This might change as the project goes into implementation. The types of sensors being considered are infrared, ultrasonic, and lidar sensors.

Infrared Sensor

The kind of infrared sensors that are considered for the project are active infrared sensors. These kinds of sensors function by emitting an infrared light through an LED and then a receiver measures the intensity [26]. The LED is constantly emitting this IR beam. When an object crosses the beam the reflected IR light's intensity (not the distance) is measured by the receiver. With the intensity the distance can be obtained.

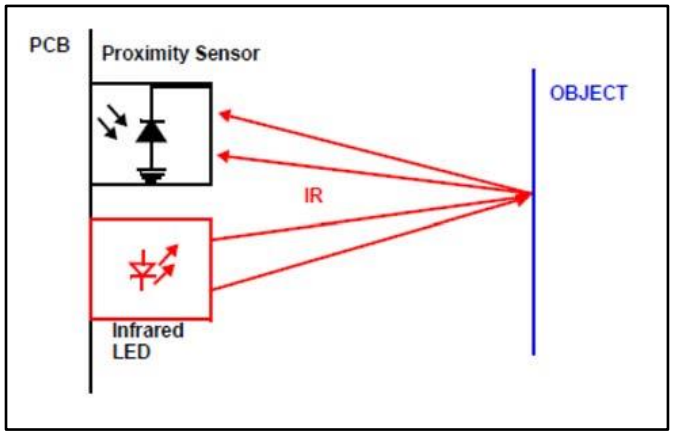


Figure 5.4.6.A: How an Infrared Sensor Works

Pros	Cons
Low price	Measurement can be affected by object color and/or the sun
Small dimensions	Bad short- and long-range reach
Acceptable resolution	

Table 5.4.6.A: Pros and Cons of Infrared Sensors with Drones

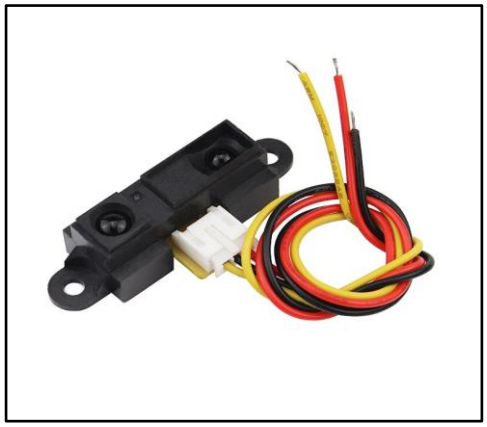


Figure 5.4.6.B: Sample Infrared Sensor

Ultrasonic Sensor

Ultrasonic sensors function by sending an ultrasonic wave through its transmitter. The wave is reflected back when it hits the object and then the receiver detects that reflected wave. The distance is calculated based on the time it takes for the receiver to detect the wave. These sensors are considered for the project since they are not affected by light nor heat. Also, the inexpensiveness of the sensor makes it a good candidate since one of the goals of the project is to spend below $500.

Figure 5.4.6.C: Ultrasonic Sensor Module HC-SR04

Pros	Cons

Inexpensive	Distance measurement can be affected by vibration from the drone
Have a decent range based on project requirements	
Not affected by heat, object color, or transparency	
Low current draw	

Table 5.4.6.B: Pros and Cons of Ultrasonic Sensor

Lidar Sensor

Lidar sensors function similar to ultrasonic sensors. These sensors function by emitting light pulses on a target and then the receiver detects the light's reflection. Through the time taken for the reflected light to be detected can the sensor measure light. These sensors excel in long range detection and can also be used to map an area due to their good accuracy. This makes this technology a good candidate for the drone. While the technology can be used for the drone the cost is a deterrent.

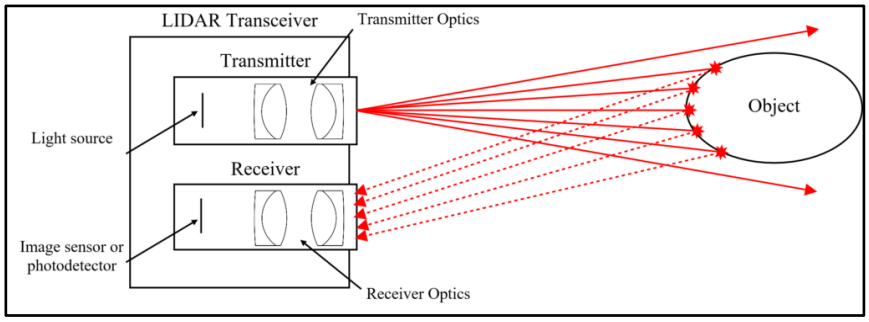


Figure 5.4.6.D: How LIDAR Sensor Technology Works

Figure 5.4.6.E: Sample LIDAR Sensor

5.4.7. Flight Controller

A drone is a complex system where many sensors and peripherals need to be connected. Every peripheral needs to be connected physically to the drone for these parts to be used in conjunction. For this a central piece of hardware is needed that will connect all devices, permit communication, and gather the information recorded by these peripherals. A sort of "brain" is in need that will manage the aircraft. This "brain" will send the appropriate voltage signals to the motors to maneuver the drone in whichever desired way. It will send signals to increase/decrease height to the motors; it will detect obstacles in its way with rangefinder sensors. This "brain" would be able to know where it is through a GPS module and it is able to send messages to its user through telemetry. This central piece of hardware is essential for the functioning of a drone because it connects every part and manages flight.

This part which can be considered the brain of the drone is called a flight controller. A flight controller has the necessary firmware to fly the drone. The firmware being the low-level software that interfaces with hardware to command it. A flight controller must be able to receive a user's flight commands, through a receiver module and a transmitter controller. It also needs to have physical connections to interface with all the peripherals connected to it. Therefore, a flight controller can be considered the brain of a drone; because it is what controls and stabilizes flight, and is able to interface with all the hardware in the drone.

Flight controllers are small computers that contain main memory (RAM), a CPU, and pins and for physical connections to interface with modules. These are essentially microcontrollers that are programmed to control and stabilize flight. For this project we are considering two types of computers to control and stabilize flight:

- Flight controllers
- Microcontrollers

If the flight controller does not have sensors built-in, then they can be connected physically. This communication can be carried out through various serial communication protocols such as UART, SPI and I2C. It is an essential part of the project to select the correct flight controller since there are different types of flight controllers made for various purposes. Some flight controllers are made for racing, these are made small with many built in sensors and with VTX communication ports. VTX is used for FPV (First Person View) flying which gives a user a different viewing experience. In contrast, other flight controllers are not meant for racing, these do not need higher speed processors or FPV.

5.4.8. Microcontroller

This project requires careful consideration of the controller to be used. This is important because there are two main kinds of flight controllers: flight controllers, and microcontrollers. Each kind of controller has its advantages and disadvantages that will need to be considered in a drone build. The main difference among these kinds of computers is the kind of firmware used. Flight controllers are easily programmed with many kinds of software available. This is not the case for microcontrollers; these need to be programmed manually. If a drone is built with a microcontroller, then the person building it must create the firmware code that enables flight and communication with the peripherals through the serial communication ports available in the microcontroller. The main difference is the difficulty that comes from programming the drone. At the same time using a microcontroller provides flexibility with the hardware in that the user can decide how to use each part of the microcontroller.

Originally there were no commercial flight controllers. To create a drone a developer would have to use a microcontroller and create a firmware that would be able to control the drone for flight. To the benefit of microcontrollers, they can be used for different purposes; they are small computers that are embedded into many devices to accomplish a specific objective. Microcontrollers are in cars, clocks, microwaves, simple cellphones (“dumb phones”), etc. The flexibility that microcontrollers provide has been used to create the firmware for a drone.

For this project we could use a microcontroller such as an msp430 from Texas Instruments. Using a microcontroller allows for the development of a firmware that is tailored to the project’s needs. This is because the developer has full control of the hardware present in the device. The GPIO pins in microcontrollers can be used for whatever purpose is needed.

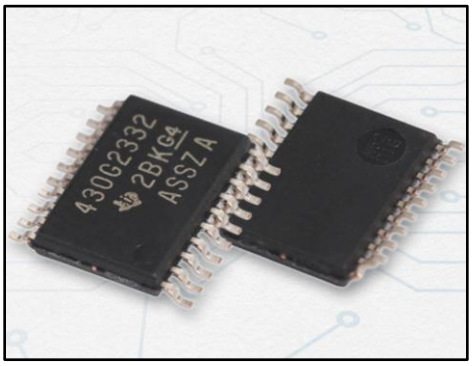


Figure 5.4.8.A: msp430G2332 Microcontroller

Pros	Cons
Inexpensive	Difficult to program
More flexibility in use of hardware	Time-consuming to program

Table 5.4.8.A: Pros and Cons of a Microcontroller

A microcontroller programmer needs to have knowledge of how the hardware works. This means that the developer would have to know how to control the pins and ports in the device and set them to the correct functionality. This means that pins used to connect sensors must be set as input and to change it requires knowledge of bit masking; this is the case with msp430FR6989 and msp430G2. The microcontroller is programmed with the C programming language and bit masking is needed to determine which pin of a port is to be set or reset. To do this, hexadecimal values can be used. The developer also needs to verify the schematics for the microcontroller to determine which port's pin must be changed to input for a sensor. Bit-masking is needed to change that pin solely since changing other pins could disrupt the functionality of other pins, which might have some other peripheral connected to it and cause it to not work. When the pin is set, there are more details to be taken care of. The correct communication protocol must be used and initialized in code. This is because peripherals do not share the same serial communication protocol. Some use UART, others use I2C or SPI; CAN is also another protocol. It is possible that some peripherals support more than one communication protocol, but this demonstrates that the programmer needs to have knowledge of the underlying functionality of a communication protocol to be able to use it in code.

This contrasts with a commercial flight controller where the firmware is provided, and the connections are plug-and-play. There is no need for the builder to know how the protocols work. When creating firmware for the microcontroller the developer also must verify that he/she implemented the communication protocol correctly and troubleshoot and test if the peripheral is functioning as intended. This sometimes requires the need of an oscilloscope to determine what voltage signals are being sent. The firmware developer also needs to understand how a drone flies to be able to implement this in code.

This means the firmware developer must understand how to use PWM (Pulse Width Modulation) to send the correct signal to the motors. If the developer wants the drone to increase its height, then he/she would need to send the same PWM signal to all motors so that they spin equally fast and raise the drone. Also, if the developer desires to make the drone move forward then the drone must be made to spin the back motors faster and the front motors slower. This allows the drone to pitch downwards and move forward. This means the firmware must implement all the drone's movement in code and be able to detect the different instructions sent through telemetry to it. All of this is complex, someone without knowledge of these concepts will have a difficult time making firmware for a microcontroller. As an added layer of difficulty, the drone must be stable in flight. Therefore, it must be tested until the firmware works correctly in controlling the drone. This is something that is of no concern if the drone uses a flight controller with downloaded firmware.

Another benefit of a microcontroller is that it can be set to different modes that reduce energy consumption. This is important for extending the battery charge in the device. To accomplish this, certain clocks in the microcontroller are turned off so that power is not unnecessarily wasted. Interrupts are used to wake the hibernating parts and regain the functionality of the peripherals connected to the clock. An interrupt allows a sensor to turn on when a certain trigger event occurs, then the device turns off again. This can be used on the search and rescue drone for turning off the camera and sensors when not in use.

5.4.9. Commercial Flight Controller

For this project, a microcontroller would be excellent for a search and rescue drone. It would give the flexibility of using the hardware as desired and to connect many different peripherals that might not be able to interface with a regular flight controller. But when compared to a commercial flight controller it is much more complex, needing to code and optimize the firmware for the drone to fly. For these reasons, this project will use a commercial flight controller.

The ports in the selected commercial flight controller need to be considered for this project. The flight controller requires a VTX communication port for the FPV video feed. The FPV video

feed is fed into the computer algorithm which will detect humans and send a signal to the flight controller that it has found someone. Fortunately, many racing flight controllers have a VTX port available. Additionally, most flight controllers have the necessary serial communication ports to connect with other peripherals. Most have UART ports for these modules to connect to.

This simplifies the project since these kinds of flight controllers are more-or-less plug-and-play. The autopilot software is flashed on it and then the peripherals are connected to each port and the software takes care of the rest. Many autopilot software provides a section for calibrating sensors for flight. There is no need for understanding how UART or any other serial communication protocol functions because the open-source software takes care of this. This is beneficial for the project because it allows the embedded systems programmer to concentrate on programming the drone to receive GPS coordinates and travel to the location and search if there is a person in need of rescue. The flight controller software is already tested and is sure to keep the drone flying. Besides reducing the complexity of the project and not having to be concerned with programming the hardware correctly. It gives the team members more time to finish the project since the group is time constraint and has other classes that take up time. It also has some sensors built in.

Commercial flight controllers are built specifically for running autopilot software and controlling drones. Since these flight controllers are made for this sole purpose, they have some peripherals useful for flight integrated into them. For instance, many flight controllers have an IMU (Inertial Measurement Unit) integrated into the controller. These modules report body forces, angular rates, and orientation of the body. These measurements are needed to be able to determine the speed of the controller, the height, role (x-axis orientation), pitch (y-axis orientation), and yaw (z-axis orientation). These IMUs can have accelerometers, gyroscopes, magnetometers, barometers, GPS receivers, etc. Flight controllers feature several degrees of freedom (DOF); 6 DOF means a 3-axis gyroscope and accelerometer (a typical IMU). They also come in 9, 10, and 11 DOF. Nine DOF adds a compass to 6 DOF; 10 DOF adds a barometer, and 11 DOF adds a GPS receiver. For this project, a 6 DOF IMU will suffice, the GPS module used will be external. Not all flight controllers bring an IMU, but many do. and this is another benefit of picking this kind of flight controller over a microcontroller.

Moreover, flight controllers can have Electronic Speed Sensors (ESC) built in. Some flight controllers do have this size and cost reducing feature. The alternative would be to buy 4 separate ESCs, one for each motor, but this saves up additional ports for more mission specific peripherals. Microcontrollers do not have built in ESCs since this is a module mostly used in flight controllers, external ones would need to be purchased if an MCU were used in our project.

Pros	Cons

Firmware is provided (no need to program)	Less flexibility with hardware use
Simple to connect sensors and other peripherals (no interfacing protocol knowledge needed)	
Specifically built for flight with integrated sensors such as IMU	
Less time spent programming	

Table 5.4.9.A: Pros and Cons of a Flight Controller

5.4.10. Selected Fight Controller Type

For this project we have decided to use a flight controller over a microcontroller because of the time constraints. This project needs to be finished in December and programming the firmware in a microcontroller would take much time. It would also require much testing to ensure flight stability; time that is better used in programming the autonomous functionality of the drone to search for a target. For this reason, we have selected flight controllers as the best controller for the drone.

There have been two main flight controllers that we have considered for this project:

- Holybro Kakute F7 AIO V1.5 Flight Controller
- Matek F405-STD Flight Controller

These two flight controllers would serve well for the project since they are both inexpensive and – most importantly – support Ardupilot. Ardupilot is the firmware that will be flashed in the flight controller to control flight.

Pros	Cons
Faster integrated F7 MCU (STM32F745: CPU 216 MHz; Flash Mem. 1MB)	Expensive

More UART ports (total: 7)	Weighs 11.2 grams (4.2 grams more than Matek)
Integrated 6 DOF IMU	
Supports Ardupilot	
OSD and VTX support	
Voltage filter for better video image	

Table 5.4.10.A: Pros and Cons of Holybro Kakute F7 AIO V1.5 Flight Controller

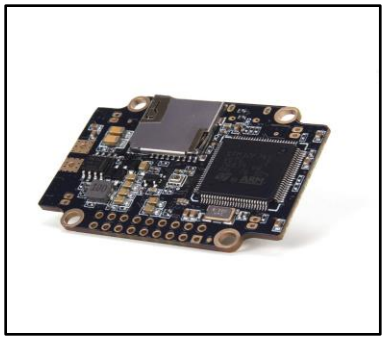


Figure 5.4.10.A: Holybro Kakute F7 AIO V1.5 Flight Controller

Pros	Cons
Less expensive than Holybro Kakute	Has less UART ports (total: 5)
Weighs less than Holybro Kakute at 7 grams	Slower F4 MCU (STM32F405RGT6: CPU 165 MHz; Flash Mem 1 MB)
6 DOF IMU	

Supports Ardupilot	
OSD and VTX support	

Table 5.4.10.B: Pros and Cons of Matek F405-STD Flight Controller

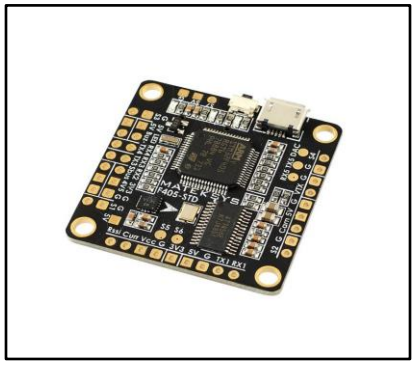


Figure 5.4.10.B: Matek F405-STD Flight Controller

5.4.11. Final Flight Controller Decision

For this project we have selected the Holybro Kakute F7 AIO V1.5 Flight Controller – mainly because it has a faster F7 processor. This will avoid any problems running the firmware during flight. The second most important reason is that it has more ports. As mentioned above, it has 7 UART ports. This is very useful because we can connect all our sensor modules and still have some extra ports available for additions or ultrasonic sensors for object detection.

Companion Computer

The companion computer or trip computer is an integral part of the drone because it has the decision making power. It can run the autonomous algorithm and make decisions based on inputs and what it encounters. The flight controller cannot run this algorithm since it would be overwhelmed. Another reason is that the library to be used for this algorithm, Dronekit (discussed later), cannot be run on the FC since it already is running the firmware. For this project we considered using the Raspberry Pi 4 Model B (2018). Some of its technical specifications are:

- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz

- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- 2 USB 3.0 ports; 2 USB 2.0 ports.
- 2 × micro-HDMI ports (up to 4kp60 supported)
- 4-pole stereo audio and composite video port
- 40 pin GPIO header (For more module connections
- Input Voltage 5V DC (Min. 3A - Can use 2.5A with good supply and USB peripherals consume less than 500 mA)

This Raspberry Pi would allow us to run the search and rescue algorithm adequately. Below is an image of the Raspberry Pi 4 Model B (2018).

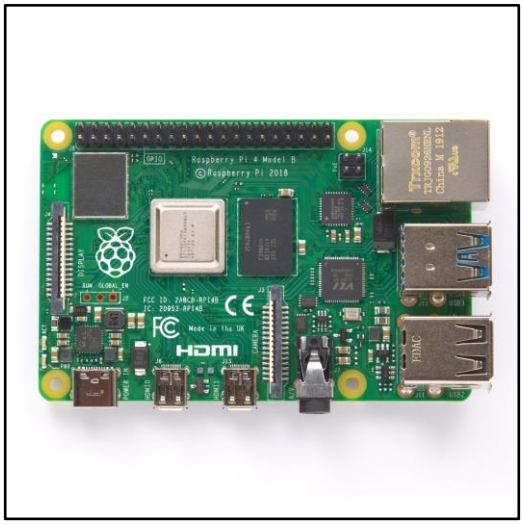


Figure 5.4.11.A: Raspberry Pi 4 Model B (2018)

There are other options that would work fine as a companion computer such as the Arduino family of products, NVidia TX2, and ODroid. The main reason the Raspberry Pi 4 has been selected is that it does not need to be purchased; one of the team members has it available. One of the goals of this project is to maintain cost below $500 and this will help reduce the project expenses. Besides this, the Raspberry Pi satisfies the project's needs.

The connection with the flight controller can be done through UART. The Raspberry Pi can be connected to the 5v DC output provided by the flight controller and then one of its pins will be connected to one of the UART ports. The connection goes like this: TX to TX; RX to RX; and GND to GND. Below is an image of a Raspberry Pi Zero connecting to a flight controller. The connection is more or less the same for the Raspberry Pi 4.

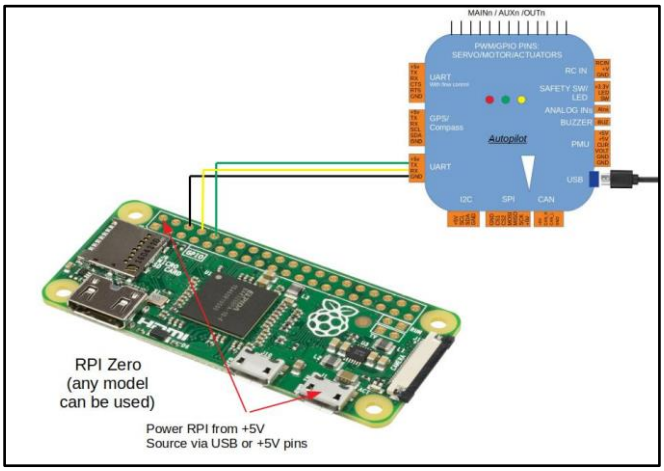


Figure 5.4.11.B: Raspberry Pi Zero Model

Though the Raspberry Pi 4 is adequate for the project we determined that it is not needed. By using RF communication (RF USB dongle) we were able to run the automation script on the GCS and control the flight controller (see section 6.4). This allowed us to reduce weight; cut development time spent on the RPI 4; and run a single FLASK script that contains the CV algorithm, automation (dronekit), and user interface code. By eliminating the RPI we also facilitated communication, saving in development time on a communication means between the UI code and the automation script.

5.4.12. Software Stack Comparison

Previously it was determined that a flight controller is better suited to control the drone versus a microcontroller. This decision is based on the project's time constraint. Flight controllers have software (firmware) that is available to use; whereas a microcontroller would require the team to program the firmware. Downloading firmware allows more time to be allocated, to the more important, higher-level aspects of programming the drone. But this is not the only piece of software needed to achieve this project's goal. A software stack is needed with three basic layers of abstraction:

- Fight Controller Software (firmware)
- Middleware
- Software layer

The lowest level of abstraction is the flight controller software (firmware), sometimes referred to as autopilot. The next layer is the middleware, which enables communication with the Ground

Control Station (GCS) and the drone; and between the flight controller and the different peripherals connected to it. Lastly, there is the software layer that uses APIs to write the autonomous search algorithm for the drone. There is also software for the Ground Control Station which helps monitor the drone.

There are different options for the layers of the software stack, but it is important to determine which tools are to be used to start programming the drone. Not all available software is the same. Furthermore, flight controllers are not compatible with all flight controller software. In other words, the flight controller you select will determine what software you can use. This is because some flight controllers have different purposes. For instance, racing flight controllers are meant to be used with racing-oriented firmware such as Betaflight. While other autopilots are better suited for fixed wing UAVs or autonomous drones that need GPS functionality.

For the firmware there were many options available to use in this project such as: Betaflight, iNavflight, Ardupilot, PX4 Autopilot, and Naze32. The two main options that have been identified for the project are Ardupilot and PX4 Autopilot. This is because they allow for GPS tracking; have a good software stack; good documentation; and have many flight controllers that are compatible with the software.

The project will be programmed in the python programming language using the dronekit api. The protocol MavLink will be used to communicate between the drone and the modules. Lastly, Ardupilot will be used as the flight controller software. Below is a table of the different software used for each layer of the stack. In the next section Ardupilot and PX4 will be compared; the following sections will discuss the other software layers.

Software Layers	Software Used
Firmware	Ardupilot
Middleware	MavLink
Software	Python with Dronekit API

Table 5.4.12.A: Software Stack

Firmware: Flight Controller Software

This section compares flight controller firmware and states reasons for the selection of one over the other. The firmwares to be compared are Ardupilot and PX4 Autopilot. Something to note is that the firmware selected dictated which flight controllers could be used in the project. This is because flight controllers are not compatible with all firmware.

Ardupilot

Ardupilot firmware was released in 2009. It has many features that are beneficial for the drones' autonomy. Due to the years it has been available there are many resources that can be consulted for programming the drone. This section will discuss the features of Ardupilot that are beneficial for the Search and Rescue drone.

Ardupilot Features

- Released earlier; more resources to consult
- Fly modes conducive to autonomy
- Supports GPS
- Failsafes for monitoring state of drone

Ardupilot Benefits Elaboration

Ardupilot has many key features that are useful. One feature is the ability to change flight modes for the drone. These flight modes determine how the drone is controlled in flight. Two necessary modes for the project are Loiter and PosHold. These modes will hold the drones position using the GPS module in it, accelerometer, and barometer. More specifically, Loiter Mode makes the drone maintain its current location, heading and altitude. This mode allows the operator to control the drone with a degree of assistance from the flight controller. When the sticks are released in the transmitter the drone will attempt to stop and hold its position. PosHold Mode is very similar to Loiter; it attempts to stop the drone when the transmitter's stick is released but it is preferred by some for its control. It allows the operator to control the lean angle with the sticks on the transmitter. These two modes are useful for the search and rescue drone since the vehicle will have a manual mode where the operator can manually control the drone. With these modes Ardupilot will facilitate flight for those with little experience and will ultimately help avoid crashes or accidents. To enable this mode the drone must have a GPS and in the case of PosHold a compass. Perhaps Loiter will be a better option since a compass may not be necessary for the drones mission. The GPS and sensors must be calibrated in Ardupilot before using these modes.

Ardupilot has a Return To Launch(RTL) feature which makes the drone return to its starting position. When this mode is initiated the drone will automatically leave its current position to travel to its home position, in this case the Ground Control Station, and hover above the location. RTL mode can also allow the drone to land. It has some adjustable parameters like, LAND_SPEED which sets the descent speed for the final landing stage in units of centimeter per second. This speed can be adjusted from 20 to 200 cm/sec. The time the drone hovers over home position before landing can also be changed with RTL_LOIT_TIME. Besides these parameters, RTL mode permits the user to adjust the height the copter will be in while in RTL mode; the speed can also be changed; and allows for controlling yaw in this mode. This mode will be useful

since the ability to cancel a mission at any time is crucial and then the drone returns on its own. This allows the operators to focus on more important tasks for a search and rescue mission. To utilize this mode the drone must have a barometer installed to measure height through changes in air pressure. Ardupilot warns that the drone could deviate from the actual height set for RTL mode if in an area where air pressure is changing. Now, if the drone is 20 feet from the ground and has sonar installed then this can be solved. The search and rescue drone should not have this problem since it will have ultrasonic sensors. Automation is necessary for the project.

Auto mode makes the drone follow a pre-programmed mission script which is stored in the flight controller. This script has different navigation commands that tell the drone where to go and what to do. The script is the mission that is required of the drone. This mode is a central part for the search and rescue drone since it allows the drone to be autonomous and execute a mission without human intervention. With this the drone can be made to receive a GPS waypoint and then made to go there. Once at the location the drone can be commanded through the script to search the area for people in need of rescue. While flying within the perimeter the drone will be commanded to send its GPS location if the computer vision algorithm detects a person. To enable Auto mode AltHold and Loiter mode must be incorporated. Loiter mode was previously mentioned but not AltHold. AltHold maintains the altitude of the drone while permitting the operator to control roll, pitch, and yaw. The drone should be able to maintain hover at 50% throttle since it could lose control.

Auto mode will ignore any command input with the transmitter so that the drone executes the script in memory. This mode can be changed with a switch on transmitters so that the operator can regain control at any time. This will be useful in the case of the drone being near some hazard and not avoiding it. This mode can be started while the drone is on the ground or flying. If the mission commences with the vehicle already in flight, then the drone will execute the same commands. If the first command is to reach a certain height, then the drone will ignore that command if it is already at that height or higher. This can be useful if the drone will execute another mission immediately before landing. Since new missions can be given to the drone through the Ground Control Station. Also, when a mission script ends the drone should be landed to conserve energy. The drone can land at its starting location or it can be made to land somewhere else. If a land command is not provided, then the drone will stop at its last GPS waypoint and the pilot will have to take control. This mode will also arm the drone at the beginning of the mission and disarm it at the end.

The script programmer can also select the speed at which the drone will travel autonomously. The maximum speed is 5m/s. In Auto mode vertical speed can also be adjusted with WPNAV_SPEED_UP and WPNAV_SPEED_DN parameters. Lastly, this mode also permits for the control of the radius of a waypoint. This is the acceptable radius in which the drone can be considered in the waypoint location. With the previously mentioned functionality of Auto mode, it is evident that it will be needed for the success of the project.

For testing the drone at a more basic level Ardupilot's Guided mode can be used. This mode is like Auto mode in that it can be made to fly to a location autonomously. But this mode is only enabled through the Ground Control Station. The pilot can select different waypoints in the Ground Control Station and the drone will go there and hover until a new waypoint is entered. The drone will not do anything else other than go to the waypoint. As mentioned earlier this mode is needed for testing basic flight in the drone while the project progresses. It will allow the team to test if the drone can fly and follow some simple commands. This mode also has Follow Me mode which makes the drone follow the pilot.

Besides the mentioned modes in which the drone can operate, Ardupilot provides Failsafes. Failsafes are software monitors that check the state of the drone. If a trigger occurs, then the drone will abandon the mission and autonomously return to home to avoid any dangers. This is needed for the search and rescue drone to avoid as many accidents as possible. The kinds of Failsafe available are: Radio, Battery, GCS, EKF, Vibration, Terrain Data Loss, Crash, Parachute, and Independent Watchdog. The most important Failsafes for the project are Radio, Battery, GCS, and EKF.

The Radio Failsafe is triggered when the transmitter is turned off for more than 0.5 seconds; the drone is outside the range of the transmitter; and if the receiver loses power. If the Failsafe is triggered, then the drone can take a few different actions. It can land where it is immediately; it can do nothing; and it can be set to RTL mode so that it lands home. With this feature the drone can return safely if it loses connection with the transmitter.

The battery Failsafe is triggered if the voltage is at or below 10.5 volts. This voltage parameter can be changed to any desired voltage. When it triggers a message will be sent to the Ground Control Station and the vehicle will take whatever action is programmed. The vehicle can be made to land where it is or land home. If it is landing home the drone is switched to RTL mode and an estimate is made to see if the drone will make it with the remaining charge. If it cannot make it home, it lands where it is. This will benefit the project by protecting the drone from crashing if its battery is about to die.

GCS Failsafe will interrupt the drones flight if it loses connection with the Ground Control Station. This is important since the drone needs to send it its status information through telemetry constantly so that any errors can be checked. This event is triggered if the Ground Control Station is disconnected; the drone is out of range for the Ground Control Station; or the drone telemetry module of Ground Control Station loses power. Similar to the Battery Failsafe the drone can be configured through parameters. The actions are: it can do nothing, land immediately, go into RTL mode, or smart RTL.

Lastly, the last important Failsafe for the drone is the EKF Failsafe. This checks for the health of EKF (the position and altitude estimation system) to find problems with the drones position estimation. This problem occurs when the GPS has a glitch (or the compass), and it helps avoid letting the drone fly away. This triggers when the position or velocity are higher than the

FS_EKF_THRESH parameter for 1 second. If in Auto mode when this event triggers, then the FS_EKF_ACTION parameter determines what the drone will do.

Besides the Failsafes, Ardupilot is completely open source. This gives the team the flexibility of using the code free of cost and modifying it if necessary. This also allows the team to share and run the software freely. Since Ardupilot is open source there is a community of developers that can assist with using Ardupilot. The documentation also has useful information and teaches how to interact with the GitHub repository to make changes to the code. It also teaches what kind of flight controllers work; how multicopters work; the different kinds of peripherals that can be connected; etc.

Ardupilot is also compatible with many Ground Control Software. The Ground Control Station is a software application that runs on the base computer. This software is used to communicate with the drone and display real-time data of the drone. The kind of data it can provide is position and performance. This software provides a virtual cockpit and lets you send GPS waypoints. Ardupilot supports the following Ground Control Software: Mission Planner (Mac OS X), APM Planner 2.0 (Windows, Mac OS X, and Linux), MAVProxy (Linux), QGroundControl (Windows, Mac OS X, Linux, Android and IOS), and UgCS (Universal Ground Control Station; Windows, Mac OS X, Linux). Besides the computer applications there are mobile app Ground Control Stations such as: Tower (Android, IOS), MAV Pilot 1.4 (IOS); Side Pilot (IOS); and AndroPilot (IOS). Ardupilot also supports use of a companion computer. A companion computer is a computer that communicates through MAVLink with the flight controller and can be used to do calculations and make decisions during flight. This computer can be a Raspberry Pie that runs a computer vision algorithm or a Nvidia Jetson Nano. These calculations would be too intense for the flight controller to run by itself. What occurs is that the Flight Controller controls flight, and the companion computer runs a side algorithm that determines decisions for the mission. This was an option for this project since the drone could have run the computer vision algorithm on the drone, but this would make the drone weigh more. The project expenses would increase if a companion computer is purchased. By using a FPV, the video feed can be provided to the computer vision algorithm in the Ground Control Station and then the algorithm would notify the drone that the target has been found through the MAVLink protocol.

PX4 Autopilot

PX4 Autopilot is a firmware that would function effectively for the Search and Rescue drone. It has the GPS support needed for autonomous flight as well as flight modes that are helpful for the project. Although newer, the developers provide a good documentation source that explains many things such as setting up a companion computer and explaining flight modes. It also supports many different software and APIs to program the flight controller. Many of the features available in Ardupilot are present in PX4 Autopilot.

PX4 Autopilot Features

- Good flight modes for autonomy
- Good documentation
- GPS support

Features Elaboration

A feature that is useful for the search and rescue drone is offboard control. Offboard Control allows a developer to control the PX4 flight by running software remotely. This means that missions could be executed in another computer and the commands sent through MAVLink to the flight controller. To do this the SET_POSITION_TARGET_LOCAL_NED and SET_ATTITUDE_TARGET MAVLink messages must be used. The former sets the desired vehicle position in a north-east-down coordinate frame and is used by an external controller. The latter is similar; it is used to set the position in a local north-east-down coordinate frame with an external controller. This feature is useful because the search and rescue drone could receive commands from Ground Control Station and so this way the computer vision algorithm could tell the drone when a person is found. The negative aspect of this feature is that more testing must be done to ensure the drone can be controlled correctly. This feature also requires a companion computer which will not be used in this project.

PX4 Autopilot also supports ROS (Robot Operating system). ROS is a framework that is used for writing robot software. With it you can make all kinds of robots, this includes drones. Creating extra software complicates a project since more time must be ascribed to it and debugging and checking functionality. Writing software for robots is even more difficult. ROS is something that is very useful and not present in Ardupilot. It could allow for the addition of extra functionality to the drone to execute more complex tasks. This is something that is alluring but currently does not benefit the project due to time constraints. Anything that would be added with ROS would be a stretch goal for our drone. Using ROS requires a companion computer which is not going to be purchased for the drone.

Like Ardupilot, PX4 Autopilot supports Failsafes. Previously it was discussed how these are extremely useful for protecting the drone in undesirable situations. In PX4 Autopilot most Failsafes are configured in QGroundControl (a Ground Control Station software), others are configured through parameters. Some of the Failsafe actions are:

- None/Disable
- Warning
- Holdmode
- Return mode
- Land mode
- Flight termination

- Lockdown

In None/Disabled, any Failsafe will be ignored. The Warning action sends a warning message to QGroundControl. These two are not too relevant for the search and rescue drone.

Hold mode action stops the drone and it hovers in place. The drone will keep its GPS location and altitude. This mode is intended as a pause during missions. If there is some interference in the mission, such as an object or an event that forces the mission to stop then this is used. This mode can also be used to regain control of the drone in an emergency. This mode is activated by a pre-programmed switch in the transmitter. It is a useful safety feature that is not available in Ardupilot. All though this same effect can be achieved by using the transmitter and changing modes. This mode requires the drone to have a GPS module. The parameter MIS_LTRMIN_ALT is used to set the minimum height of Hold mode. There is another parameter that allows the pilot to regain control.

Another mode present in PX4 is Return mode. Just like Ardupilot's RTL mode, it allows the drone to go somewhere safe when the mode is triggered. Unlike Hold mode this mode is triggered automatically. With Return mode the action to be taken by the drone can be selected. With the return type Home/rally point return the drone reaches a safe altitude and travels to home or a specified point. There is also Mission landing/rally point return; this functions similarly to the previous one but differs in that the drone will go to the closest mission destination point. If no such point exists, then the drone goes home or to a rally point. If Mission path return type is used, then the drone will try to land at the last GPS waypoint for the mission. Lastly, closest safe destination return makes the drone return to whatever is closest, a rally point, home, or mission point. This mode is like the Land mode which lands the drone at the place where the mode was engaged. In this mode the rate of descent during landing can be set. Interestingly, this mode shares similarities with Ardupilot's RTL mode. Both these modes are needed for the project during testing and missions.

PX4 has a more general failsafe action called Flight Termination. This covers different cases where safety is a concern and so the mission/flight is terminated. In other words, Failsafes permit the pilot to specify conditions for which it is safe to fly. For instance, in PX4 a battery failsafe action can be set through the COM_LOW_BAT_ACT parameter. With this parameter the drone can be made to return or land somewhere safe. There exists a RC Loss failsafe as well. The action for it is set with the parameter NAV_RCL_ACT and it causes the drone to take one of the following actions: be disabled, Hold mode, Return mode, Land mode, Terminate, Lockdown. Also, these features further confirm the similarities among Ardupilot and PX4 Autopilot.

An API that is not available in Ardupilot is MAVSDK. This is a C++ MAVLink library for managing communication between a flight controller and the Ground Control Station; and communication between the flight controller and the different peripherals connected to it. It also permits the communication between different vehicles. If a drone has a companion computer then this is the protocol needed for the two computers to communicate. In the case of this project

one way this protocol can be used is by sending from the Ground Control Station a MAVLink message to the drone that the computer vision algorithm found the target. This library supports blocking synchronous and asynchronous calls. These calls would be made in the script the drone will be running on and will wait for a message from the Ground Control Station. This API allows for fast communication among onboard devices. And allows up to 255 vehicles connected via TCP or UDP. Another important feature is that vehicle information and state can be extracted through telemetry. If used in the project it would allow the team to have information on GPS coordinates, the RC connection, flight mode, battery level, etc. This is something crucial for the project, with this, the drone can be monitored, and errors detected.

The MAVSDK library would also allow for the Ground Control Station to disarm/arm the drone. It would also permit external commands such as return to launch. Another benefit is that the drone can be controlled directly without a transmitter. Missions can be created and managed with this API. This library is implemented in other programming languages. It is available in Java as MAVSDK-Java and in python with MAVSDK-Python. Although this library does not exist in Ardupilot, it has a similar API. The MAVLink message protocol can be used in Ardupilot through the pymav library. With either firmware the functionality can be accessed.

Software Selection: Ardupilot vs PX4 Autopilot

Both Ardupilot and PX4 Autopilot are very similar. Both firmwares provide extensive features for piloting a drone. They both have failsafe features; they share the MAVLink protocol for communication; both have extensive documentation; and they support GPS navigation for autonomy of the drone. The similarities are great enough that either software would work for this project. When everything is considered, the firmware selected for the project is Ardupilot. The main reason is that Ardupilot has a few more resources that will be beneficial when programming the drone. Besides this either firmware would function for the project.

Middleware: MavLink

Mavlink is a communication protocol that will be used for the drone. This protocol permits the communication between the drone (Flight Controller) and the components connected to the flight controller. MavLink structures messages in frames of 8 - 263 Bytes. To differentiate messages a messageID is used. Message size varies according to the message ID and each messageID is a different command. MavLink will be used to enable communication between the Ardupilot and Dronekit to program the drone.

Software: Dronekit

DroneKit is an API written in python that will be used to program the drone. This library is used to control the drone through a python script. It gives the user the ability to command the drone by sending MavLink messages to Ardupilot. This is very important because it is what will be used to program the drone for autonomous search and rescue. It also provides functions for accessing MavLink messages to accomplish actions. This is useful for the project because the programmers do not need to know how to structure MavLink messages. This API will be run on the Raspberry Pi 4 Model B since as the companion computer it will be commanding the drone.

5.4.13 RF Interference

During field testing, we experienced significant disruptions in communication between the ground station and the drone. Upon further investigation, we discovered that the field we used for flight tests was subject to a significant amount of radio signals, which was causing interference with our communication and resulting in the failure of some commands being transmitted.

The ground control station relies on radio waves to communicate with the drone. “Radio waves” refer to waves that are transmitted in the radio band of the electromagnetic spectrum, and there are many applications for this type of communication. Because some of those applications are military or otherwise sensitive, there are limited frequency ranges within the radio band that are legal for general purpose use, reserving other frequencies for high-priority applications. The most commonly used frequencies for drones and other wireless communications are 2.4GHz or 5.8GHz.

Whenever multiple devices are attempting to transmit messages through the same space at the same frequency, the possibility for interference exists. Interference in this case is caused when a device is listening for a message from its transmitter and expects it to be on a specific frequency, but mistakenly intercepts a signal from a different transmitter instead. When this happens, the intended message can get dropped because the receiver was already receiving the wrong message when the right message was transmitted. This type of interference is most common when there are many wireless devices in use in the same area. Devices that can cause this type of interference include cellular phones and internet hardware for wifi networks.

Signal scrambling is another type of interference that can affect communication between the drone and the ground control station. In this type of interference, the message that is being transmitted is overpowered by a much stronger electromagnetic wave, even if the stronger wave is of a dramatically different frequency. This type of interference is most commonly experienced around radio antennae, and the area of effect varies with the type of antenna. Tower antennas emit electromagnetic waves in a donut shape radiating outward from the tower parallel to the ground, and the strength of the signal is spread more or less evenly across that entire area. The

configuration means that approaching the tower from the top or flying over the tower is generally safe and relatively free from interference. Dish or parabolic antenna concentrate the strength of the generated electromagnetic waves in one direction, and flying through the path of one while it is transmitted can have catastrophic consequences due to the amount of interference that will be experienced by the drone.

A less likely but still possible source of electromagnetic waves strong enough to scramble an RF signal is power lines. Power lines are not an electromagnetic field in and of themselves, but when there is current moving through the lines, the movement of the electrons generates an electromagnetic field that can be strong enough to affect the transmission of RF messages between the ground station and the drone. The more current flowing through the line, the stronger the EMF and the more likely it is that there will be signal disruption in the vicinity of the lines. Lines that distribute power to individual buildings are not likely to have a strong enough EMF to affect the flight of a drone, but high voltage (high current) lines between power stations can easily cause interference.

Interference can also be caused by physical objects around the drone, or between the drone and the ground control station. Because the messages are being transmitted as waves, they are subject to reflection and refraction as they encounter different surfaces enroute to the recipient. Hard surfaces such as buildings or canyon walls can drastically reduce the distance an RF wave can be transmitted successfully, as the reflection of the waves off of the surface can interfere with the original wave. RF signals generally cannot be transmitted through hard obstructions, so a lack of clear path between the drone and its associated ground control station can also make reliable communication impossible.

5.5. Thrust/Power System

This section will be explaining how we reached our conclusions on picking the following components and their specifications:

- Battery
- Propeller (Length, blade count, pitch)
- Motor Size
- ESC

My main goal when selecting parts in this section is to find what combination of the components will lead to the best flight time while also being both available and within our budget. When testing for each component I used eCalc for multirotor. This is a tool that allows me to input the drones weight, battery capacity, motor, and propeller specifications amongst other things. In

return the tool will calculate a large amount of data. Below is an image of what the tool looks like, highlighted are the main specifications I changed in my testing.

The most important output we get from the tool is hover flight time since our drone will mostly be hovering slowly around a field in its search. We want a flight time of around 15-20 minutes so we can accomplish our goal. We do not have to worry about maximum flight speed, since even the slowest of the setups tested travel at maximum speeds of at least 100mph, making all of them capable of the slower travel speeds we require for the search.

Going into testing I needed to get a baseline of what the rest of the components would weigh, as the base weight of the drone would greatly affect our final target and the strength of the power system we would need. We took the weights of the components we know we are using, and took an average of the possible candidates for the other components, and placed the results in the table below:

Part	**Weight / Unit (g)**	**# of units**	**Total weight (g)**
Motors	?	4	?
ESC	15	1	15
GPS	9	1	9
VTX	7	1	7
Camera	7	1	7
FC	6	1	6
Propellers	?	4	?

Receiver	4	1	4
Frame	128	1	128
Battery	220	?	?
Hardware	84	1	84
Total			360

Table 5.5.A: Table of Components – Their Count and Weight

In the following tests, the weight of the motors, propellers, and batteries will change, and the total weight will be reflected in that specific test.

5.5.1. Battery Type

One of the biggest influences on flight time of course is the battery. There are many types of flight batteries used out there but for our project we will be using LiPo (Lithium Polymer) batteries. These batteries have a decent energy density, more than capable discharge rates, and stable voltage over the batteries discharge. The biggest reason for our choice however is price. Since team member Joseph Manalo has an abundance of 1300mah 6 cell LiPo packs, we will free up possibly over a fifth of our budget that can be used elsewhere. However, if we find that we have extra budget, we may consider buying or building Lithium-Ion battery packs. Lithium-Ion batteries have almost double the energy density of a LiPo battery, but they have a very limited discharge rate. It is quite possible with careful planning and minor alterations to our initial setup, that we can achieve a low enough discharge rate to accommodate the Lithium-Ions shortcomings. But for our initial setup, we plan on running multiple 1300mah 6s LiPo batteries in parallel.

5.5.2. eCalc Testing

Most of my conclusions and graphs will stem from a series of computations I ran through the eCalc tool. The test consisted of first gathering a range of test components.

For motors I examined the range of motors popular manufacturer T-Motor offers for a medium sized quadcopter. The size of the motor is a compound number formatted as XXYY. The first half is the stator width in millimeters, and the second half is the stator height in millimeters as well. Included in this test are the following motors:

Motor Name	Motor Size	Weight (g)
F40	2205	123.2
F60	2207	134
F80	2407	158.8
F90	2806.5	186.4

Table 5.5.2.A: Table of Motors – Their Size and Weight

For the propellers I changed both the propeller length and blade count, as these parameters have a large effect on both the weight and efficiency of the drone. My range for propeller size goes from 5in to 8in, since these are the most used, realistic, and available propellers for mid-sized long-range quadcopters. In these sizes they have both two blade and three blade variants. For all the tests in this section I used a constant pitch to keep things organized. Most propellers range from 3in to 5in so for these tests I used a constant 4in pitch. In another test I use a variable pitch alongside the more efficient setups I gathered from this section of tests. I gathered average weights of a wide selection of commercial propellers with the desired length and blade count and added all the propeller data to the table below:

Propeller Size (in)	Two-Blade Weight (g)	Three-Blade Weight (g)
5	12.4	15.2

6	20.4	25.6
7	26.4	36.4
8	30.8	44.2

Table 5.5.2.B: Table of Propellers – Their Size, Two-Blade Weight, and Three-Blade Weight

The last variable in my testing was the number of batteries used. As explained in section X.X.2 we are using 1300mah 6s LiPos. These batteries weigh 220g a piece and I used a range from one to four batteries. I stopped at four batteries as the weight of the drone would start becoming much too heavy for a mid-sized drone, and the frame would need to be quite large to accommodate the space for all four batteries.

With all the calculated weights I was able to input the weight of the drone more accurately into the eCalc tool. In this test I would input the correct specifications and note down the calculated flight time at a hover. Most of my conclusions in the following section are derived from these calculated flight times. A full list of the calculations can be found at the end of this section.

5.5.3. Battery Count

The biggest factors to keep in mind as we look at the data for the batteries is the fact that the batteries weigh 220g a piece and are 3x1.5x1.5 inches in size. This means for each battery added on, we will need to accommodate for the space it will take up, and the sheer amount of added thrust needed for a hover. The data for this graph was derived from the average flight times of all the combinations of motor and propeller for each number of batteries.

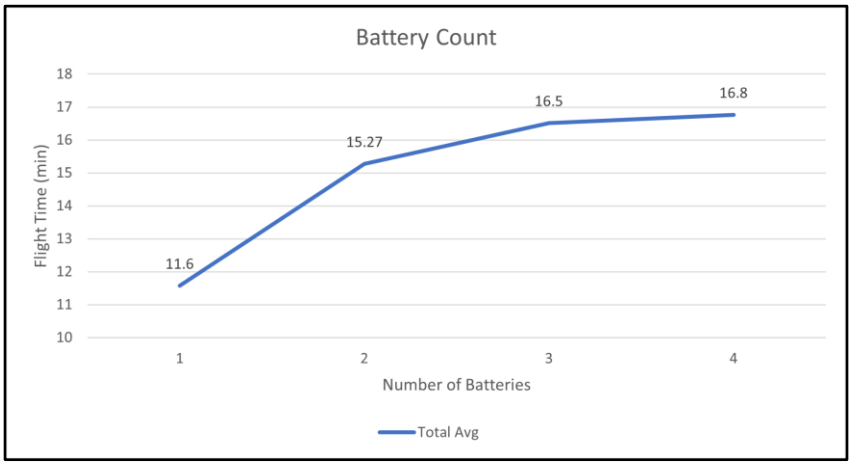


Figure 5.5.3.A: Graph Depicting Relationship between Flight Time and Battery Count

As we already knew that adding additional batteries will increase the flight time. However, there is a clear trend of diminishing returns, as the difference between one and two batteries is an additional 3.67 minutes on average, and the difference between three and four batteries is only an additional 0.4 minutes. As we make our conclusion on how many batteries we want to have onboard at once, we must determine if the overall weight and cargo space is worth the additional flight time.

Conclusion

Currently it looks like either two or three batteries will be optimal for our build, as one battery may not provide us with enough flight time, and four batteries may not be worth the extra space needed for such a small return.

5.5.4. Motor Size and Propeller Length

In this section, we will learn about the relationship between the size of the motor and the length of the propeller. From scanning the data, it seemed like these two had the largest affectual range when paired together properly and were heavily reliant on each other. Below are multiple flight-time tables consisting of the different pairings between motor and propeller length. Each table shows the flight time in minutes for a different battery count, and a heat map has been overlaid on top of all of them to show their relative flight time to the rest of the data.

Motor/Propeller Length Flight Time (min) Heat Map				
1 Battery	5in	6in	7in	8in
2205	9.8	11.5	12.95	14.15
2207	9.65	11.4	13	14.35
2407	9.15	10.9	12.45	13.95
2806.5	8	9.8	11.4	12.7
2 Batteries	5in	6in	7in	8in
2205	12.7	14.9	16.75	18.25
2207	12.65	15	16.95	18.7
2407	12.15	14.55	16.6	18.5
2806.5	10.85	13.3	15.35	17.15
3 Batteries	5in	6in	7in	8in
2205	13.6	15.95	17.85	19.4
2207	13.6	16.1	18.25	20.05
2407	13.3	15.85	18.1	20.05
2806.5	11.95	14.6	16.85	18.8
4 Batteries	5in	6in	7in	8in
2205	13.7	16.05	17.95	19.4
2207	13.85	16.3	18.45	20.25
2407	13.55	16.15	18.4	20.4
2806.5	12.25	15	17.3	19.25

Figure 5.5.4.A: Heat Map Depicting Relationship Between Motors and Propeller Length

A few conclusions can be made from this data. As we learned before, more batteries equate to more flight time with diminishing returns. We can also conclude that a larger size propeller will increase efficiency as well. In every table the better flight time comes from both the 7in and 8in columns. We can also see that the larger motors are less efficient with the smaller propellers when compared to the smaller motors. However, as we move to the larger propellers, the larger motors begin to catch up in efficiency. This becomes clearer with the heavier setups that have more batteries, although the largest motor 2806.5 never catches up to the others. This means it is too large of a motor to be efficient with our build.

Since we concluded before that two or three batteries will be best for our build, we should focus on those tables to conclude the best motor size and propeller length. It is clear that 2207 and 2407 are the most efficient motors in our lineup. As the build becomes heavier, and we swing a larger prop, 2407 slowly becomes more efficient than 2207.

Conclusion

In the end, the frame's design and component weight will determine what combination we will use. If we must limit ourselves to a 7in prop, the 2207 may be the better option if we can stay on the lighter side. If the frame design can account for the larger 8in prop, having the 2407 will be our best choice. Having the 2407 will also be better if we see an increase in the total drone's weight than what we originally predicted.

5.5.5. Propeller Blade Count

The propeller blade count is the number of blades on each propeller. In my research I found that there are mostly only two and three blade propellers in the sizes we are looking at. As a baseline we know that more blades will increase thrust at the same RPM as a propeller with less blades. However, due to the added surface area you will also see an increase in resistance at the same RPM, and therefore more power use on the propeller with more blades. There is also the additional weight added from the extra blade. This does not matter as much with the smaller 5in propellers but as you get to the larger propellers it can add over 10 grams. The following graph shows the average times for blade count across the different number of battery setups.

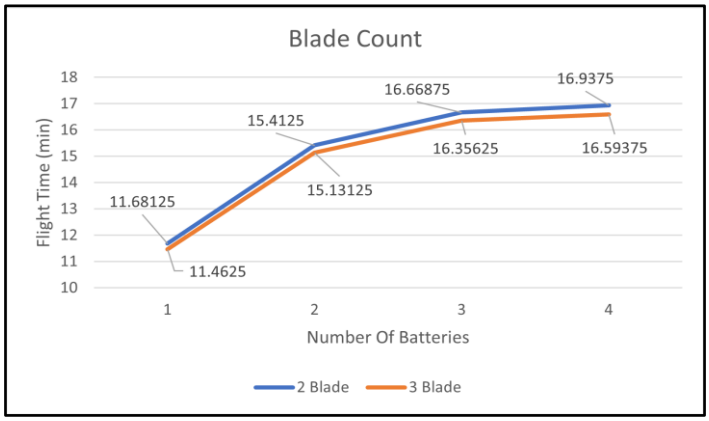


Figure 5.5.5.A: Graph Depicting Relationship Between Flight Time and Battery Count with Two- and Three-Blades

Conclusion

It's pretty easy to see that along the entire graph the two blade propellers have better efficiency than the three blade propellers. So, for the purpose of maximizing the efficiency of our build the two blade propellers will be our choice. If for some reason we need more thrust per RPM we can switch to the three blades without too much of an impact on our flight time. But since our goal is slow stable flight, we will most likely stay with the two blades.

5.5.6. Propeller Pitch

Since propeller pitch is being tested after I have conducted the other tests, I will be specifically testing the different propeller pitches on the setups we have settled on. There is also a very

limited number of pitches available for each size prop that we have, but we will be testing over the entire range of propeller pitches out there. Then we can create a graph that will be referenced when purchasing the propellers.

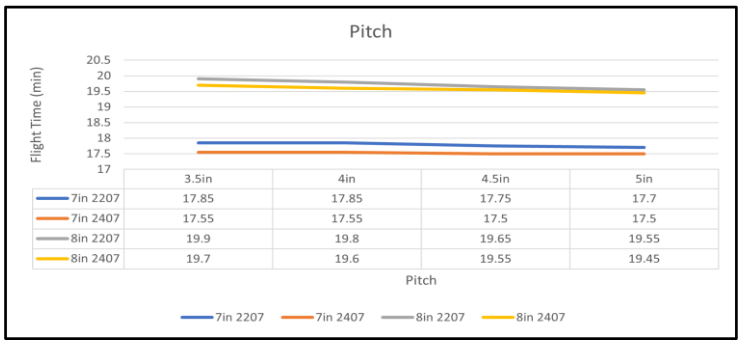


Figure 5.5.6.A: Graph Depicting Relationship Between Flight Time and Propeller Pitch

This graph is composed of the flight times of the 7in and 8in propellers between the two motors and averaged across the two battery counts we came up with earlier. The propeller pitch ranges from 3.5in to 5in.

Conclusion

It seems like there is a very shallow downward slope. This slope however does not even change the flight time by a half minute over the course of the graph for any setup. Because of this, we will try to stay away from propellers with steeper pitches, but ultimately price and availability will be the bigger deciding factor on which propeller pitch we go with.

5.5.7. Electronic Speed Controller

The ESC (Electronic Speed Controller) is responsible for interpreting the flight controllers output signals and diverting power accordingly to the motors. For us to choose the ESC we first need to choose whether we want a 4-in-1 ESC or four individual ESCs. 4-in-1 ESCs incorporate all four of the ESCs into one board that will be mounted in the main stack of the quadcopter. The 4-in-1 also acts as the power control board in that it is the main battery connection to the system. Having a 4-in-1 also simplifies the build process and is also usually cheaper than the alternative. For these reasons we will be using a 4-in-1 ESC, as the only real downside is if you burn an ESC in a crash or flight related issue, you must replace the entire board.

The only other parameters that come into play when selecting the ESC is the voltage and current rating, and of course price. Since we plan on using 6 cell LiPo batteries, the ESC must be rated to handle the 25.2 volts the batteries put out when fully charged. The ESC must also be rated for the number of amps we plan on pulling through it. Our most amp hungry setup will pull somewhere between 10 and 60 amps across it's powerband. So, our board should sit around a rating of 40A continuous and 60A burst. Ultimately the ESC will be one of the more important parts when it comes down to quality. If we run a cheap ESC, it is very possible that it will burn itself out during our testing. Because of this we want to get a mid to high priced ESC.

5.5.8. Full eCalc Data List

1300 Mah

3 Blade Weight (g)					2 Blade Weight (g)				
	5	6	7	8		5	6	7	8
2205	618.4	628.8	639.6	647.4	2205	615.6	623.6	629.6	634
2207	629.2	639.6	650.4	658.2	2207	626.4	634.4	640.4	644.8
2407	654	664.4	675.2	683	2407	651.2	659.2	665.2	669.6
2806.5	681.6	692	702.8	710.6	2806.5	678.8	686.8	692.8	697.2

3 Blade Flight Time (min)					2 Blade Flight Time (min)				
	5	6	7	8		5	6	7	8
2205	9.8	11.4	12.7	13.8	2205	9.8	11.6	13.2	14.5
2207	9.7	11.4	12.8	14	2207	9.6	11.4	13.2	14.7
2407	9.2	10.9	12.3	13.7	2407	9.1	10.9	12.6	14.2
2806.5	8.1	9.8	11.3	12.5	2806.5	7.9	9.8	11.5	12.9

2600 Mah

3 Blade Weight (g)					2 Blade Weight (g)				
	5	6	7	8		5	6	7	8
2205	838.4	848.8	859.6	867.4	2205	835.6	843.6	849.6	854
2207	849.2	859.6	870.4	878.2	2207	846.4	854.4	860.4	864.8
2407	874	884.4	895.2	903	2407	871.2	879.2	885.2	889.6
2806.5	901.6	912	922.8	930.6	2806.5	898.8	906.8	912.8	917.2

3 Blade Flight Time (min)					2 Blade Flight Time (min)				
	5	6	7	8		5	6	7	8
2205	12.7	14.8	16.5	17.8	2205	12.7	15	17	18.7
2207	12.7	14.9	16.7	18.3	2207	12.6	15.1	17.2	19.1
2407	12.2	14.5	16.4	18.2	2407	12.1	14.6	16.8	18.8
2806.5	11	13.3	15.2	16.9	2806.5	10.7	13.3	15.5	17.4

3900 Mah

3 Blade Weight (g)					2 Blade Weight (g)				
	5	6	7	8		5	6	7	8
2205	1058.4	1068.8	1079.6	1087.4	2205	1055.6	1063.6	1069.6	1074
2207	1069.2	1079.6	1090.4	1098.2	2207	1066.4	1074.4	1080.4	1084.8
2407	1094	1104.4	1115.2	1123	2407	1091.2	1099.2	1105.2	1109.6
2806.5	1121.6	1132	1142.8	1150.6	2806.5	1118.8	1126.8	1132.8	1137.2

3 Blade Flight Time (min)					2 Blade Flight Time (min)				
	5	6	7	8		5	6	7	8
2205	13.6	15.8	17.5	18.9	2205	13.6	16.1	18.2	19.9
2207	13.6	16	18	19.6	2207	13.6	16.2	18.5	20.5
2407	13.4	15.8	17.9	19.7	2407	13.2	15.9	18.3	20.4
2806.5	12.1	14.6	16.7	18.5	2806.5	11.8	14.6	17	19.1

5200 Mah

3 Blade Weight (g)					2 Blade Weight (g)				
	5	6	7	8		5	6	7	8
2205	1278.4	1288.8	1299.6	1307.4	2205	1275.6	1283.6	1289.6	1294
2207	1289.2	1299.6	1310.4	1318.2	2207	1286.4	1294.4	1300.4	1304.8
2407	1314	1324.4	1335.2	1343	2407	1311.2	1319.2	1325.2	1329.6
2806.5	1341.6	1352	1362.8	1370.6	2806.5	1338.8	1346.8	1352.8	1357.2

3 Blade Flight Time (min)					2 Blade Flight Time (min)				
	5	6	7	8		5	6	7	8
2205	13.7	15.9	17.6	18.9	2205	13.7	16.2	18.3	19.9
2207	13.9	16.2	18.2	19.8	2207	13.8	16.4	18.7	20.7
2407	13.6	16.1	18.2	20	2407	13.5	16.2	18.6	20.8
2806.5	12.4	15	17.1	18.9	2806.5	12.1	15	17.5	19.6

Figure 5.5.8.A: Main Test

Pitch Test is below:

2600 mah										
Weight (g)										
	7	8								
2207	860.4	864.8								
2407	885.2	889.6								
7in						8in				
	3.5	4	4.5	5			3.5	4	4.5	5
2207	17.2	17.2	17.1	17.1		2207	19.2	19.1	19	18.9
2407	16.8	16.8	16.8	16.8		2407	18.9	18.8	18.8	18.7
3900 mah										
Weight (g)										
	7	8								
2207	1080.4	1084.8								
2407	1105.2	1109.6								
7in						8in				
	3.5	4	4.5	5			3.5	4	4.5	5
2207	18.5	18.5	18.4	18.3		2207	20.6	20.5	20.3	20.2
2407	18.3	18.3	18.2	18.2		2407	20.5	20.4	20.3	20.2

Figure 5.5.8.B: Pitch Test

5.5.9 Final Thoughts on Propeller Length vs. Motor Size

In our eCalc testing we were able to determine the best combination of motor size and propeller length. We are going to take a quick look at why we reached the conclusion that we did.

As a basis we understand that a propeller is spun to create lift. The larger the propeller the harder it is to spin, but the more lift it is able to create. Through our testing we found that the larger the propeller we had the more efficient the system became. This is because the larger propeller creates more lift at a lower RPM than the smaller propeller. However, the motor has to have more torque to be able to spin the propeller with its increased resistance. This is why the 2407 motor became more efficient with the larger propellers than the 2207, but we must look at the definition of torque to understand this relationship better.

$$\tau = rF\sin\theta$$

τ = Torque
r = Radius Of The Moment Arm
F = Force Being Applied
θ = Angle Between the Force vector and Moment Arm

The main variable we are changing when we are looking at motor size is the radius of the moment arm. Since we know that the larger propeller will need more torque to spin effectively, we must have a larger radius so we can create more torque. This is why the 2407 motor is better than the 2207. The first two digits in the motor size is the diameter of the stator in millimeters,

while the second two digits are the stator height. The 2407 motor is 2mm wider than the 2207, which means it has the longer radius we are looking for.

In our project bigger doesn't always mean better though. Since the weight of our components is directly proportional to the thrust our system needs to create, there becomes a point where we start to lose efficiency. We knew that to stay a relatively small sized drone that we shouldn't go larger than an 8in propeller. So with that size propeller we tested the motor sizes. We can see that once you go past the 2407 motor size we have a decent drop off in efficiency. This is because the 2407 is fully capable of spinning the 8in propeller already, going to a larger motor will indeed add more torque, but that torque will not be put to use unless we have a bigger propeller. However, the larger 2806.5 motor weighs almost 30 grams more than the 2407, which increases the amount of thrust needed to lift the drone. So with the larger motor, we have more thrust needed to fly but we don't produce any more thrust than we did with the smaller motor.

5.6. Video System

Since our project involves a computer vision algorithm, we need to choose a video system for our drone. Our video system needs to provide adequate video for the computer algorithm and needs to be quickly sent to the video processor so we can operate in real time. The biggest question we must answer in determining a video system is if we want to process the video onboard the drone or send the video data to an offboard workstation.

5.6.1. Onboard Video Processing

For onboard processing we will be filming data using an onboard camera and feeding that data directly into an onboard computer loaded with the computer vision software. We have found an excellent microcomputer, the Nvidia Jetson Nano that will be able to run our software in real time with the live video coming into the computer. However, with this additional hardware we would have to make extra room for the processor and account for the extra weight and power usage. There is also the additional price of the computer on the budget and the extra time it will take to learn the device.

With onboard computing we can use much better-quality cameras since we do not have to transfer that data wirelessly. Even though we are transferring less data in terms of video, we will have to expand the interface of the host computer to be able to interact with the computer on the drone.

Pros	Cons
Not limited by video range or data size	Need to account for the added price, weight, power usage, and space of the processor and added connections
Expandable and allows for better options to include other parts of the system	Need to expand UI interaction between the host computer and the drone
	CV may need to be adapted to work on the Nano

Table 5.6.1.A: Table Depicts Pros and Cons of Onboard Video Processing

5.6.2. Offboard Video Processing

For offboard processing we can use a 5.8Ghz video system to transfer the data in real time to an off-board workstation. The system will consist of a camera, the VTX (video transmitter), and a video receiver that will feed the video into the offboard computer. This system, if set up correctly - and with a powerful enough VTX - can transmit the video in less than 30 milliseconds and a distance of over a mile with no interference. With this system the same computer processing the video can also be controlling the drone. So, there will be no need to add the extra interface between two separate computers.

The biggest issue with this system is the quality of the video. Since we are transmitting the data wirelessly, we must use rather low-quality cameras. Our cheapest option comes in the form of analog video which will hit a price point of under $100. We can also use more expensive digital systems as well for improved image quality, but this upgrade may cost us over $100 in addition.

Pros	Cons

Has very cheap options	Questionable video quality
Low latency	Eventually limited by video range
Simple to set up, low power usage (<0.5A) and lightweight (< 10g)	

Table 5.6.2.A: Table Depicts Pros and Cons of Off-Board Video Processing

5.6.3. Possible Video Processing Devices

For any human detection software that we have, we will need a type of device to run it. We came to the conclusion that we only have two options to process the drone vision to detect people, but of course both of these come with their own pros and cons. One option is to transmit the analog video to a computer on the ground and then we run the person detection program in real time to process the input from the computer. The second option is to use a small device called the Jetson Nano and have it located on the drone to scan the video footage while it's still on the drone level.

Figure 5.6.3.A: Jetson Nano - Processing Device Design to Run AI and CV Models

Figures 5.6.3.B & 5.6.3.C: Computer Components that allow for us to run heavier software compared to a Jetson Nano

5.6.4. Conclusion

Since our project is a flying drone, the cost of having an onboard computer will have some serious impacts on our design. The added weight, space, and power usage will limit our drones' capabilities and multiple design aspects of the drone. Even though with the onboard solution we can have much higher quality video, if our offboard solution can provide the computer vision algorithm with adequate video, there is no reason we should go with an onboard computer. The offboard system is cheaper, easier to set up, and has less mechanical and electrical strain on the drone. Furthermore, even though off-board CV model processing comes with issues such as latency and potential packet loss, it allows for easier drone construction and less battery drainage.

In light of this, we chose to process our video using the offboard solution with an onboard camera. Our camera will feed the footage via the video transmitter on the drone to the video receiver plugged into the computer. We can then open our CV model, and using this camera, begin detecting any humans within the frame.

5.7. Computer Vision

This section explains how we reached our conclusions regarding the selection of a suitable Computer Vision (CV) approach and model.

Computer Vision is the field of study that focuses on "helping computers to see" [17]. To put it in more technical terms, at "an abstract level, the goal of computer vision problems is to use the observed image data to infer something about the world" [17].

CV helps computers - whether the processing be done onboard another platform or off-board - to identify objects or events, such as the presence of a dog in a yard, a car on the highway, or - for this project - a human being in a field.

As our drone moves throughout the field, we will have an onboard camera that takes real-time video data of the ground beneath the drone as it flies. This real-time video data must be processed effectively to ensure that a human (and the accompanying location) can be classified as such by the drone, and return the location.

We intend to utilize CV to process this real-time video data and use it to identify any humans the drone may encounter.

5.7.1. Computer Vision Approaches

CV is a subset of both Artificial Intelligence (AI) and Machine Learning (ML) - terms that are heard all too commonly today. What this means is that CV borrows from strategies used in both AI and ML to accomplish its goals - Fig. 5.7.1.A below illustrates the relationship.

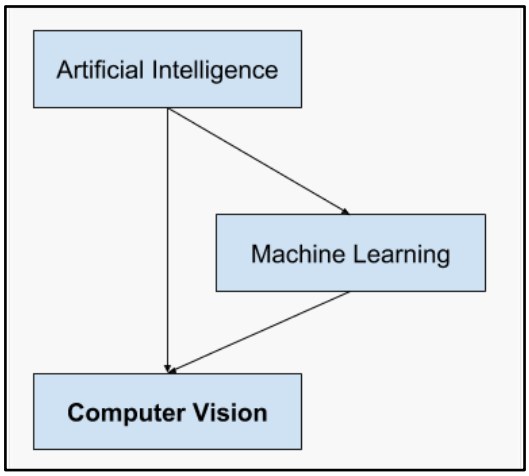


Figure 5.7.1.A: Visual Relationship between Artificial Intelligence, Machine Learning, and Computer Vision

As said above, CV utilizes strategies from general AI and ML, in that it uses four main modes of training a model:

- Supervised Learning
- Unsupervised Learning
- Transfer Learning

Supervised Learning

Supervised learning is the process of training a ML model by mapping input variables (called "features") to a specified output(s) called a "label(s)" or "ground truth" [18], so that the model can use an algorithm to learn the necessary mapping functions from the inputs to the output.

An everyday example can be seen in showing a child several images of an elephant. If a child is shown 200 images of an elephant, that child is able to make the necessary mapping connections between what features an elephant has and what makes it an elephant and not a giraffe (i.e. long nose, grey-brown skin, big ears, four legs, etc.). If this child, then, is taken to the zoo, and shown a certain animal, that child will - by virtue of learning the features of an elephant - readily be able to identify that animal as being an elephant or not.

In the same way, CV models can be trained using supervised learning. If specific objects are highlighted to the model - and the model is training to understand what pattern makes a dog a dog, for instance, and not a cat - then the model is being trained using a supervised learning approach.

Fig. 5.7.1.B illustrates the supervised learning process as it relates to ML.

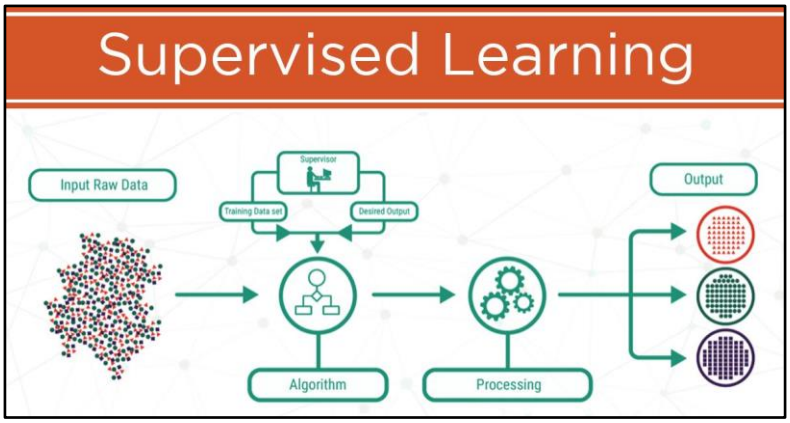


Figure 5.7.1.B: Supervised Learning Process in Machine Learning

Unsupervised Learning

Unsupervised learning is the process wherein the ML or CV model is not given ground truth, and must work on its own to discover patterns existent within the input dataset [18].

Whereas supervised learning deals with more structured data such as labelled images, text in spreadsheets, etc., unsupervised learning is used for unlabelled data, such as grouping customers via purchase history, or providing video recommendations based on recent Internet searches, etc.

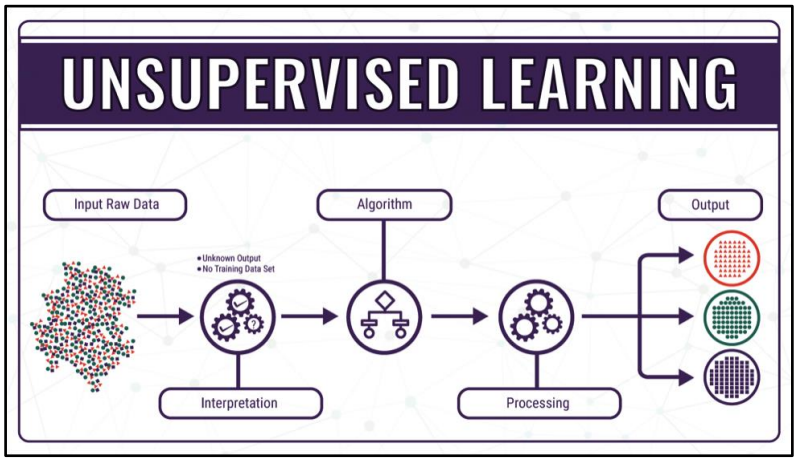


Figure 5.7.1.C: Unsupervised Learning Process in Machine Learning

Transfer Learning

Transfer learning is the process whereby a ML or CV model that was designed and developed for a task is utilized for another task of similar nature. For instance, if a CV model existed that was designed to detect dogs playing in a park, that same model could then be reused to detect dogs playing on the beach, since these two tasks are of similar yet inexact nature (dogs are playing within the video frame, but in different locations).

Fig. 5.7.1.D illustrates the difference between traditional ML approaches (such as supervised and unsupervised models), and transfer learning.

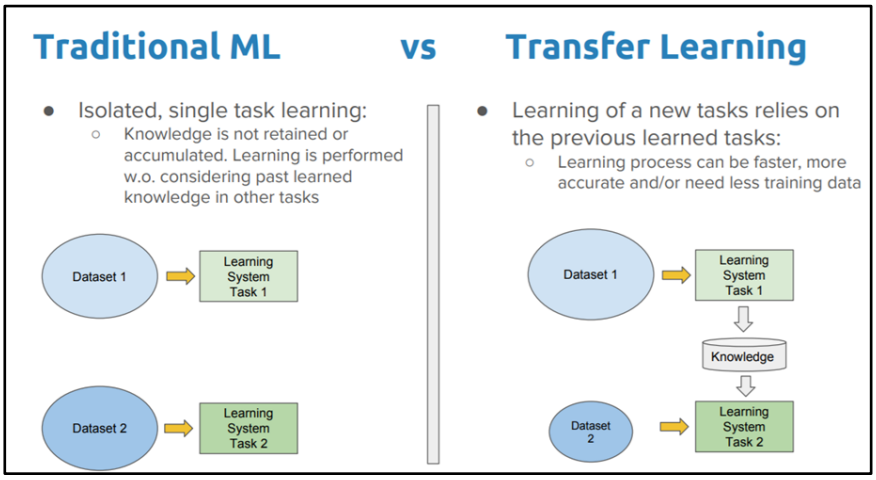


Figure 5.7.1.D: Transfer Learning vs. Traditional Machine Learning Approaches

Transfer learning approaches and techniques tend to be less expensive both computationally and monetarily because, since the model being used was not trained on any specific dataset - but was, in fact, reused - the developer does not have to spend computational power or extra man-hours to develop the model for the task at hand. The model can simply be reused from one of the many myriads of resources on such tasks, and, if necessary, adjusted to meet the requirements and demands of the project.

5.7.2. CV Approach for this Project

For this project, we have adopted an approach solely based on transfer learning. As was said before, this approach reuses existing models for similar yet inexact purposes, and, when compared to supervised or unsupervised learning, is less computationally and monetarily expensive because a dataset is not necessary to train on.

For our project, we will be detecting humans within a field. To create a dataset that was large enough to both train and test on if we adopted a supervised learning approach (which would have been more accurate than unsupervised) would have taken weeks if not months, because we would first have to acquire sufficient video footage, preprocess that footage to highlight features, remove excess noise to prevent overfitting, select the correct algorithm to train the model, and - having completed all this - waited for the model to finish training, a process which could take several hours.

Since the workload was more than could be conceivably completed during the semester, we investigated transfer learning, and found that since the nature of our problem of human detection is so common, there are a number of existing models to choose from. These models were not specifically trained to detect humans within a field, but were used to detect humans on a beach, in a park, in the city, etc., and could be utilized in our necessary setting.

5.7.3. Possible Pre-Trained Models

Upon further investigation, we found that some pre-trained models suited our purpose while others did not, as is elaborated below.

Support Vector Machines (SVMs) Classifier with Histogram of Oriented Gradients (HOG) Features

The HOG Descriptor is a feature descriptor that extracts features from image data by focusing on the structure or shape of an object. It provides information as to whether a pixel is an edge or not,

as well as the direction of the edge by extracting the magnitude (gradient) and direction (orientation) of the edges.

These orientations are calculated in localized portions – the complete image is broken into smaller regions, and the gradient and orientation are calculated for each region. Then, a histogram for each region is created, using the gradients and orientations of the pixel values [21].

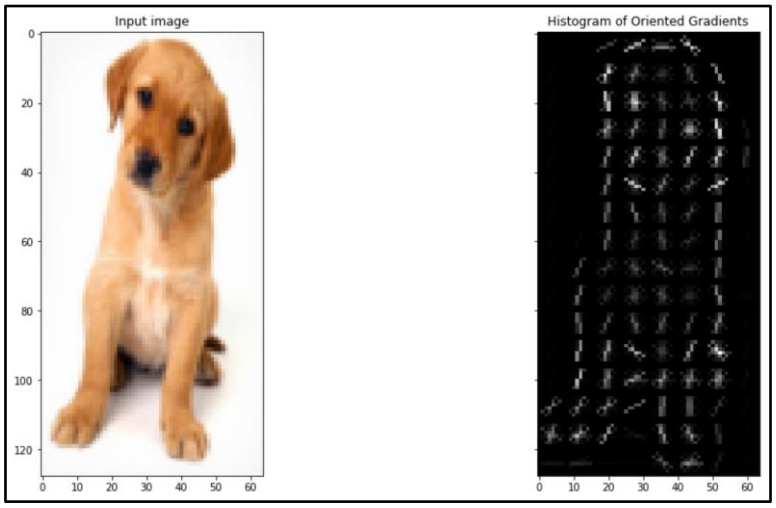


Figure 5.7.3.A: Left – Input Image of a Dog;
Right – HOG Feature Extraction of Input Image

Upon the HOG Descriptor completing the extraction of the image's features, SVMs then classify the image frame as containing a human or not. The model can be written in Python – with OpenCV's function *HOGDescriptor()* for HOG feature extraction – and then uses the *setSVMDetector()* method to run the SVM classifier on the features, as shown below in Fig. 5.7.3.B.

```
124 if __name__ == "__main__":
125     HOGCV = cv2.HOGDescriptor()
126     HOGCV.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
127
128     args = argsParser()
129     humanDetector(args)
130
```

Figure 5.7.3.B: Python Code Implementation of HOG Descriptor with OpenCV

EfficientDet

The EfficientDet model is an object detection model that uses various backbone and optimization techniques. Essentially, it extracts features from the image, classifies them, and fuses them together to predict the object. Some of the techniques that it uses is a weighted Bi-directional Feature Pyramid Network (BiFPN) - which is a feature pyramid network for multi-scale feature fusion - as well as a compound scaling method that uniformly scales the resolution, depth and width for all backbones [22].

Figure 5.7.3.C below showcases the model architecture.

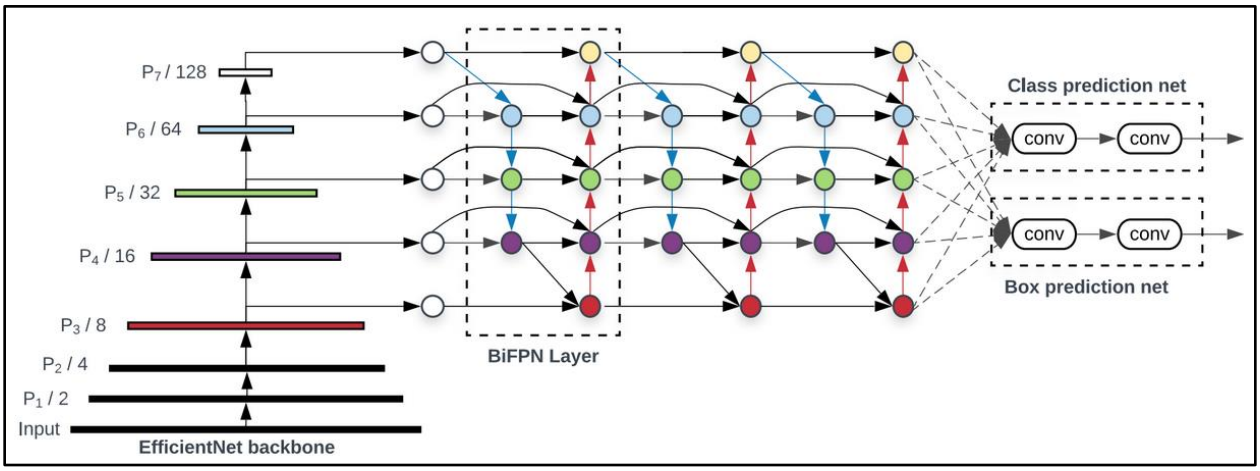


Figure 5.7.3.C: EfficientDet Model Architecture

You Only Look Once (YOLO)

YOLO models were developed by Joseph Redmon, et al [19] and were first described in his 2015 paper You Only Look Once: Unified Real-Time Object Detection. This paper disregarded prior work done on object detection in which “classifiers were repurposed to perform detection”. Rather, they “framed object detection as a regression problem to spatially separated bounding boxes and associated class probabilities[, wherein a] single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation” [19].

Fig. 5.7.3.D below depicts the result of a YOLO model [20].

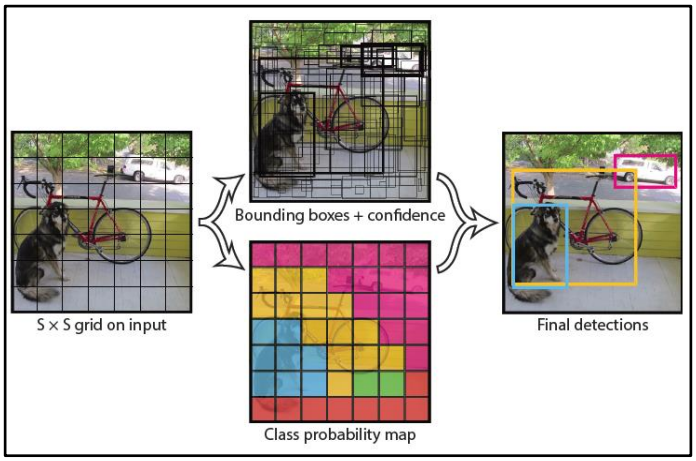


Figure 5.7.3.D: Visual Depiction of YOLO Model in Detecting a Dog, Bicycle, and Car

5.7.4. Our Model

Originally, we had selected HOG Descriptor as our model of choice. It was fast, detected tolerably well, and did not take a lot of effort to set up, despite the hyperparameters needing to be adjusted to account for processing time. Figure 5.7.4.A shows the correct detection of a human using the HOG Descriptor.

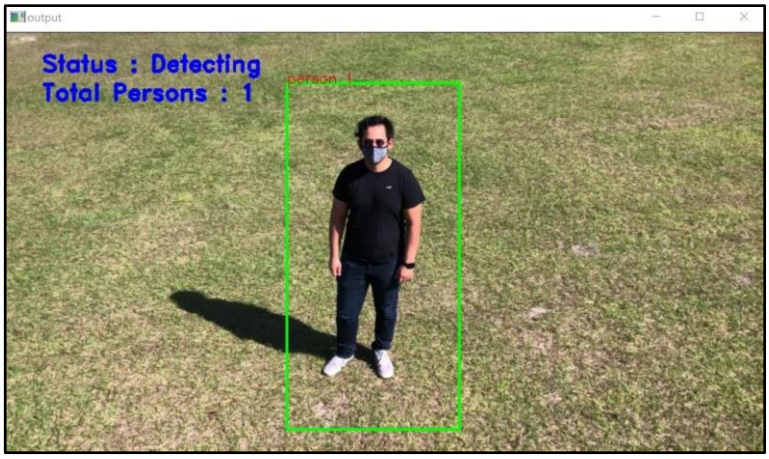


Figure 5.7.4.A: HOG Descriptor Correctly Detects Human within Drone Camera Frame with Digital Footage

However, the HOG Descriptor classified too many objects incorrectly as humans, as shown below in Figure 5.7.4.B. So, we had to look for other options.

Figure 5.7.4.B: HOG Descriptor Incorrectly Identifies a Tree as a Person

This led us to YOLO, specifically YOLOv4. Currently, there are five versions of the YOLO model (with Version 5 being released in 2020). However, YOLOv5 ran slowly on our computer, and unfortunately, we did not have access to a GPU to speed the process up. So, we fell back to YOLOv4, which ran just as accurately and somewhat faster on our CPU, but was still not as fast as the live footage we ran from the drone - operating at about 4-5 frames per second (FPS). Fortunately, after doing extensive research, we found that there exists a compressed version of YOLOv4 called YOLOv4-Tiny. After running it, we found that it ran more than twice as fast as YOLOv4 (at 10-12 FPS), with a slight drop in accuracy.

Unfortunately, YOLOv5 did not have a compressed version ("YOLOv5-Tiny") that we could find, so we settled with YOLOv4-Tiny. After testing it with live drone footage, we found that it detected a human standing ~9 feet away with 80% confidence (shown below in Figure 5.7.4.C - the other information in the picture frame is recorded from the camera, and could not be removed).

Figure 5.7.4.C: YOLOv4-Tiny Detecting Human Standing 9 Feet Away with 80% Confidence

After these results, we concluded that YOLOv4-Tiny suited our purposes for the extent of this project.

Conclusion

Currently, other models such as RetinaNet are being evaluated to determine their validity and performance, but because of the makeup and effectiveness of YOLOv4-Tiny, our design choice is unlikely to change.

Our drone will be flying with a camera angle of at least 20 degrees at a height of 5 meters (~16 feet) above the ground over an open field. At this height and angle, no clouds, trees, or other objects are likely to be visible, and as such, the drone will move forward, only being able to pick up grass and humans (if any are present within the field).

The CV model is only required to detect humans with at least 75-80% confidence, a feat it can easily do, hence our decision to stick with YOLOv4-Tiny.

5.7.5. How to Run the Model

Our model runs as a function within the backend Python script (shown below in Figure 5.7.5.A).

```
# Code for CV Model (YOLOv4) below
# YOLOv4 Tiny Weights and CFG Files must be within folder of app.py
def CV_Model(cap, multi_rescue):
        starting_time = time.time()
        frame_id = 0

        _,frame= cap.read() # Read the frame from the 'cap' tuple
        frame_id+=1

        height,width,channels = frame.shape

        # Detecting objects within the frame
        blob = cv2.dnn.blobFromImage(frame,0.00392,(320,320),(0,0,0),True,crop=False) #reduce 416 to 320

        net.setInput(blob)
        outs = net.forward(outputlayers)

        stop = False

        #Showing info on screen/ get confidence score of algorithm in detecting an object in blob
        class_ids = []
        confidences = []
        boxes = []
        for out in outs:
            for detection in out:
                scores = detection[5:]
                class_id = np.argmax(scores)
                confidence = scores[class_id]
```

Figure 5.7.5.A: Portion of the CV Model Within Backend Python Script

The fact that it is a function allowed us to test other aspects of the script without involving the CV model, something that helped us immensely in the beginning stages of development. We were also able to pass in a boolean that will most likely be used if we manage to locate multiple people in a single run.

6. Project Hardware & Software System Architecture

6.1. General Block Diagram

Below is a block diagram image of the project – with each member's responsibilities and areas of work color-coded accordingly.

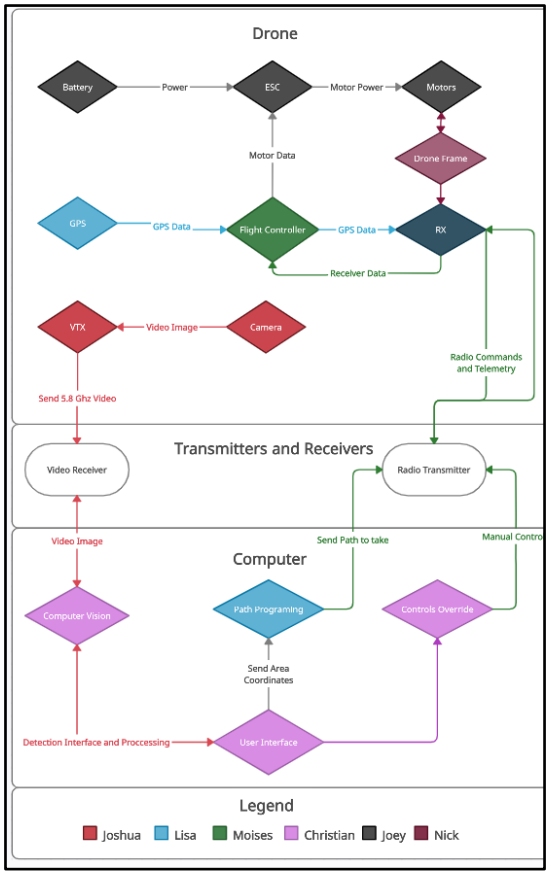


Figure 6.1.A: Team Block Diagram

6.2. Ground Computer User Interface Diagram

Below is a diagram of the User Interface (UI) as it relates to the drone's movements and components.

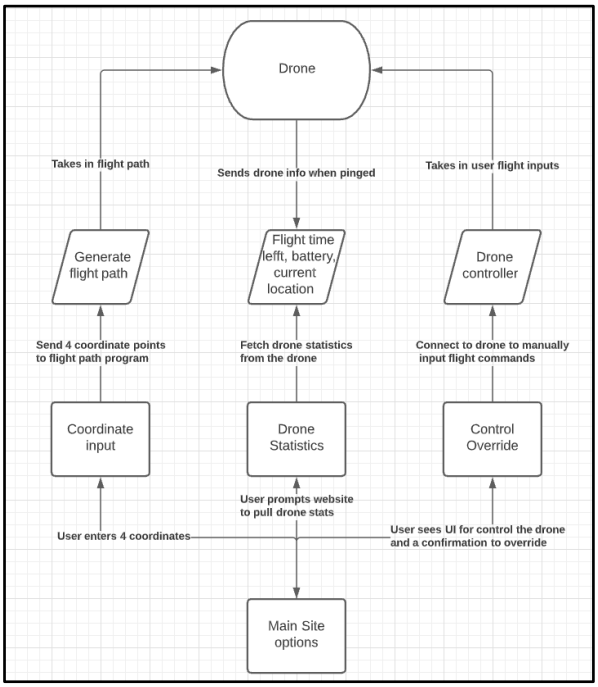


Figure 6.2.A: UI – Drone Diagram

6.3. Drone to Ground Control Station Communication & Diagram

Communication between the drone and the GCS(Ground Control Station) is achieved through RF. On the GCS side an USB RF transmitter is used to send commands and the drone has an RF receiver attached to the flight controller. Below is an image of the RF dongle used in the drone.

Figure 6.3.A: Holybro Transceiver Telemetry Radio V3 915MHz

Through RF waves MavLink messages are sent to the drone to command it; as well as received telemetry data. To initiate communication MavProxy must be started. Mavproxy will connect to COM5, which is the serial connection port to the RF transmitter (This will vary for each

computer). The user must open UDP port 14550. This port is used to enable communication between the flight controller, and the automation script and user interface.

When the UI (app.py) is started the user is first prompted for the comm port. The user must enter 14550. This will allow the automation script to connect to the drone through the UDP port and send/receive MavLink messages. Then the dronekit(automation) code can begin to control the drone and provide telemetry data to the user interface. Since the dronekit code is part of the user interface script the UI will receive telemetry data and can send land and retreat (RTL) commands to the drone.

All the commands are received by the flight controller and through the GPS it can determine its position and where it has to go. Below is a diagram that shows how the drone communicates with the GCS.

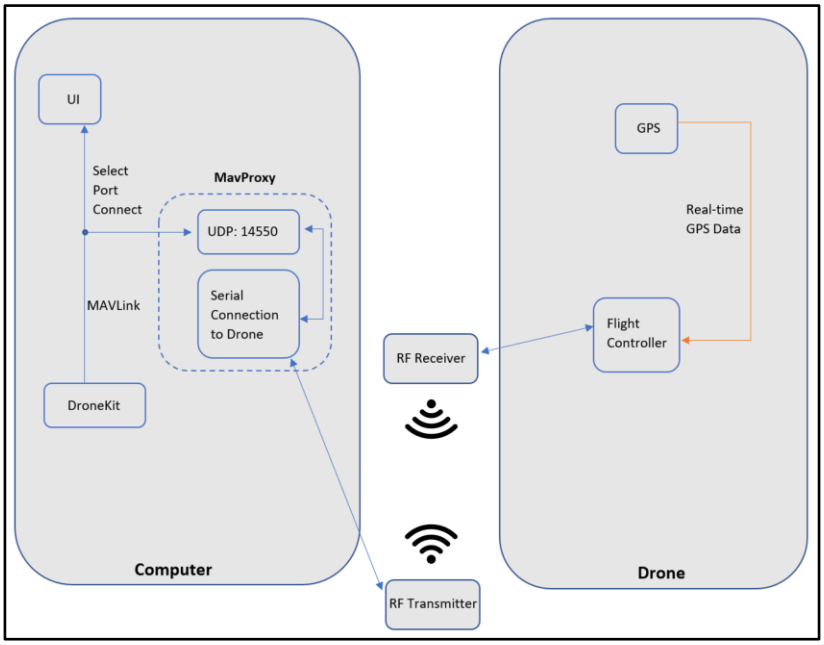


Figure 6.3.B: Drone to GCS Communication Diagram

6.4. Component List

Onboard Drone Components	
Flight Controller: *Holybro Kakute AIO V1.5* - STM32F745 Processor - Barometer	**Video Transmitter:** *RDQ Mach 3* - MMCX - 5.8Ghz

- 6 UARTS - Supports SBUS and Ardupilot - 2s-10s Voltage - 30x30 mounting - 11.2g	- 1000mw max output - 2s-6s Voltage - Built in Microphone - 6.5g
Propeller: *APC B8x4.5MR-B4 Bi-Blade* - 8in length - 4.5in pitch - Bi-Blade - 9g	**Camera:** *Foxeer T-Rex Micro* - 1500TVL - 1.7 Lens - 4.5-16V Input - 8.6g
Motor: *T-Motor F80 Pro* - 1900 KV - 2407 - 3s-6s - 49.16A Max Draw - 39.7g	**Battery:** *Tattu R-Line V3* - LiPo - 6s - 1300mah - 120C - 208g
Receiver: *FrSky R-XSR* - SBUS - Telemetry - 4-10V Input - 1.5g	**Vtx Antenna:** *Foxeer Lollipop w/ pigtail* - RHCP - MMCX To SMA Pigtail - 9.5g
Onboard Computer: *Raspberry Pi 4 Model B (2018)* - 2GB Ram -CPU:Broadcom BCM2711, 4 core, 64-bit @ 1.5GHz - 5V Input - 3A Minimum(2.5A if USB peripherals consume less than 500mA) - 50g	**Electronic Speed Controller:** *Hobbywing XRotor Micro* - 4s-6s Voltage - 4in1 - 60A - BLHeli32 - 30x30 mounting - 15g

Offboard Components

Video Receiver: *R600* - 5.8Ghz - AV output	**Capture Card:** *UCEC Capture Card* - AV input - DIGITAL output
Transmitter: *TX16S Max* - MultiProtocol (Including FrSky Protocols)	

7. Project Tests and Logs

Below are the outlines for the tests for this project.

7.1. Analog Video Test

7.1.1. Goal:

The goal of this test is to help understand what level of video quality is clear enough for the computer vision algorithm. Specifically, we seek to find if the video achievable with 5.8Ghz analog video technology is usable for our project.

7.1.2. Requirements:

- A 5.8Ghz analog video system capable of recording and transferring footage
- Test Field (at least 10,000 square feet)
- Human test object
- Way to simulate a drone's flight with the video system
- Computer with the computer vision algorithm

7.1.3. Procedure:

1. Find a large mostly empty field and have the human test subject stand out in the field.
2. Using the video system at a height of at least 15ft, record the subject standing in the field. If possible, move the video system over the subject, making a few passes in both directions.
3. Have the subject lay down in the field and repeat the recording procedure.
4. Transfer the footage to the computer and run it through the computer vision algorithm, logging the algorithm's ability to locate the subject in the field.

7.1.4. Timeline:

This test should be executed early on in development to help determine what video system should be used for the project. Should be done before tuning the algorithm and determining on board components.

7.2. First Prototype Flight Test

7.2.1. Goal:

This test is to be done once the manually controlled prototype is built. This test to ensure that the drone is operating as it should and is ready for tuning and then automated implementation.

7.2.2. Requirements:

- Prototype drone
- Test Field (Safe of any people and at least 10,000 square feet)
- Basic set of drone tools (Screwdrivers, prop tools, soldering iron, etc.)
- Computer with firmware configurator loader

7.2.3. Procedure:

1. With the propellers off, test the motors, spooling them up one at a time. Ensure that the motor order is correct and that the motor direction is correct. Also ensure that the flight controller board orientation is correct
2. Check the radio inputs in the firmware and do a failsafe test, arming the drone (still with the props off) and powering off the radio.
3. Apply a light and safe tune to make sure the drone is stable. A lighter tune will have reduced PIDS and more filtering, as well as an amp limit and reduced spool up power on the ESC.
4. Load charged batteries on the drone and put the propellers on, checking the propeller direction while doing so and making sure to tighten the nuts down all the way.
5. Plug in the battery, then check the radio and video connection.
6. After checking your surroundings to make sure it is safe, do a quick test arm, arming the drone and after a second disarming just in case the drone tries to run away.
7. Once you feel confident that the drone will fly, arm the drone again, and slowly bring it to a hover a few feet off the ground, being constantly on the disarm switch in case anything goes wrong.
8. After hovering for 10-20 seconds bring the drone down, disarming on touch down. Check the motor temperature to make sure they did not rapidly get hot during the hover.
9. Perform another test hover, this time performing some basic maneuvers. Roll the drone side to side, pitch the drone forward and back, returning the drone to a hover after performing each maneuver. Do a full rotation around the yaw axis in each direction. Then perform some small throttle changes. Rapidly move the stick, increasing throttle by 10%,

if the drone seems stable do it again this time by about 25%. This process should not take longer than one minute, in which you should then bring it down, disarming, and performing another motor temperature check.

10. At this point you now have a functional drone, and the drone should be tuned in, following a drone specific tuning guide.

7.2.4. Timeline:

This test should be done as soon as the prototype is finished. The prototype should be assembled as soon as all the parts for it have been selected and approved by the team. This should be done early enough in the project so that there is time for tuning and implementing the automation system.

7.3. Flight Time Test

7.3. Goal:

The goal of this test is to understand the range of flight times we can get out of the drone. Doing this can help us spot issues that we may have further in development once we begin implementing the search algorithm and computer vision sides of the project.

7.3.2. Requirements:

- Prototype drone
- Test Field (at least 10,000 square feet)

7.3.3. Procedure:

1. First run a hover flight time test. The batteries should be charged to 4.2 volts per cell. Hover the drone for 3 minutes at a time, recording the voltage of the batteries after each section. If the voltage hits a resting voltage of 3.7 volts end the test.
2. Now do the same while keeping a speed of about at least 10mph, keeping a separate record of the voltages.

7.3.4. Timeline:

This test can be executed once the drone is in a state where it can be comfortably flown manually. Does not have to be completely tuned in at this time.

7.4 UI Comms Test

7.4.1 Goal:

The goal of this test is to see if we are ready to perform the field test. All parts of the test will be performed on the bench and make sure all the functions work as intended.

7.4.2 Requirements:

- Basic UI Running on PC
- Command Ready Drone

7.4.3 Procedure:

1. With the props off, plug in the drone and open the UI on the PC.
2. Select the appropriate connection settings and open the connection to the drone from the UI.
3. When the connection is opened, check if the telemetry data is correctly displayed.
4. Enter a dummy set of GPS coordinates to create a waypoint path.
5. While having someone ready to pull the plug on the drone, hit the start mission button. Check if the motors do indeed start up. If they do, immediately hit the stop mission button. If the drone does not disarm, pull the plug.
6. Check any flight logs to confirm that everything is working as intended.

7.4.4. Timeline:

This test should be completed once the UI is created and able to be connected with the drone and before you run the automated test.

7.5. Basic Automated Navigation Test

7.5.1. Goal:

This test will help us understand the drones flight system capabilities when it comes down to automated navigation. This test should provide us with enough information to go into implementing the search algorithms.

7.5.2. Requirements:

- Prototype drone (Tuned and basic automation set up)
- Test Field (At least 10,000 square feet)

7.5.3. Procedure:

1. Start the automated test by having the drone fly into the middle of the field, staying about 15 feet above the ground and not exceeding speeds of 10mph.
2. From there, have the drone fly to one side of the field and stop. Then from there fly back into the center and stop.
3. To test continuous flying with direction changes, have the drone fly back to the side of the field, and without stopping fly to the opposite side of the field. Without stopping, have the drone return to the center of the field.
4. To test height changes, have the drone increase its height to 20ft. Then have the drone fly to the side of the field, while also returning its height to 15ft. Then have it fly back to the center of the field, increasing its height back to 20ft. Then while the drone is in the center of the field, have it return to a height of 15ft.
5. To test speed changes, have the drone begin circling the test field continuously. While the drone circles, have it increase speed by 5 mph, then reduce speed by 10mph, and lastly return speed back to its original speed while returning to the center of the field.
6. To test dynamic changes in flight path, have the drone fly to one side of the field, and while it is in flight, interrupt it to return to the center of the field.
7. Lastly, have the drone perform a failsafe, which should have the drone return to its takeoff position and land.

7.5.4. Timeline:

This test should be executed after the prototype has been manually tested and tuned. The flight controller will have to be properly set up to allow for the automation. As soon as the flight

controller is set up, this test should be performed. Once this test is performed, development on incorporating the search UI can begin.

Note: This test was replaced with 7.6 since it's easier to run the UI script than to send individual commands.

7.6. Automation Test

7.6.1 Goal:

This test is to see if the basic autonomous functions of the drone are finished and ready for refining.

7.6.2 Requirements:

- UI Comms Test Passed
- Basic UI Running on PC
- Command Ready Drone
- Test Field

7.6.3 Procedure:

1. Plug the drone in and connect to the UI.
2. Select the search area for the drone.
3. Check the telemetry data for GPS accuracy.
4. With the drone in the field and the field is clear, hit the start mission button.
5. While the drone is flying along its path, be sure to take note of the telemetry data and how it changes over the course of the flight.
6. You should also have someone watching the video as it flies to see if any adjustments need to be made.
7. Once the drone has completed its mission, unplug it and record the outcome of the test.

7.6.4. Timeline:

This test should be executed once the UI comms test is passed and the UI automation is ready to be tested.

7.7. Full End-To-End Test

7.7.1 Goal:

This is the final passing test to see if our goal has been achieved.

7.7.2 Requirements:

- Automation Test Passed
- Final UI Running on PC w/ CV Algorithm
- Final Drone
- Test Field

7.7.3 Procedure:

1. Plug the drone in and connect to the UI.
2. Select the search area for the drone.
3. Check the telemetry data for GPS accuracy.
4. With the drone in the field and the one or two targets set out in the field, hit the start mission button.
5. Let the drone continue and take note of what happens as the drone passes over the targets.
6. The drone should stop at some point and RTL when it finds its target.
7. Any adjustments should be made to increase search accuracy and this test should be rerun until optimal

7.7.4. Timeline:

This is the last test to be executed. It requires all parts to be finished and working together.

7.8. Build & Testing Log

5/14/2021 - Prototype Manual Build

- Primed esc pads
- Soldered XT60 onto ESC
    - Motors screwed onto stand-in frame
- ESC mounted
    - Motor 1 soldered and smoke stopper tested successfully
    - Motor 2 soldered and smoke stopper tested successfully
    - Motor 3 soldered and smoke stopper tested successfully
    - Motor 4 soldered and smoke stopper tested successfully
- Motors all soldered and good to go
    - FC would not connect to computer due to driver issues
    - We could not update the FC to ardupilot
- Build halted - Still need all FC and peripheral work

6/11/2021 - Prototype Manual Build cont.

- Prior to build meet FC was updated to Ardupilot and the configurator has been installed on laptop
- Connected +/- from ESC connector to +/- battery leads on flight controller
    - No sufficient power
- Connected positive and negative on flight controller directly to battery leads on ESC
    - Power is now sufficient
- Connected motor signal wires from ESC connector to FC
    - No ESC connect tones
- Receiver wired to +/- and R6 and bound to radio
- Ardupilot configured for receiver
    - Protocol set to FRSKY D8
    - Successfully reading radio input
- Attempted to reconfigure ardupilot to connect with ESC
    - ESC protocol set to Dshot 600
    - Ardupilot set to communicate to BLHeli
    - Attempted ESC configurator in BLHeli
    - Still no ESC tones and motor test does not work
- Video Transmitter wired to +/- and VO pin
    - Successfully powering on and transmitting OSD
- Camera wired to +/- and VI pin

    - Successfully powering on and VTX is now transmitting camera feed
- Build Halted - Still need to connect FC and ESC, configure ESC

6/30/2021 - Prototype Manual Build cont.

- Restarted FC configuration process
    - Reset FC parameters to default
    - Selected X Frame configuration
- RC Transmitter setup
    - FRSKY D8 protocol
    - Radio configured
- Accelerometer configured
- ESC calibration
    - ESC Type set DShot600
    - Output PWM Min 1000 - Max 2000
    - ESC Connection Successful
- Motor Spin test
    - Successfully spins all motors
- Motor Map test
    - Mapped: (ArdupilotMap, ESC#) | (A,2) (B,1) (C,3) (D,4)
- Motor Direction test
    - (ESC#,Direction[CW,CCW]) | (1,CW) (2,CW) (3,CW) (4,CCW)
- Flight Mode - Aux 5 - pos0 Stabilize - pos1 Stabilize
    - Flight Modes Configured
    - Flight Mode - Aux 6 - pos0 Stabilize - pos1 Stabilize - pos2 Acro
- Throttle Cap in Radio setup - pos0 85% - pos2 100% - pos3 50%
- Connect to BLHeli to change motor direction
    - Connection failed
- Parameters changed for BLHeli passthorugh - SERVO_BLH-AUTO 1
- Connect to BLHeli to change motor direction
    - Connection successful
- Motors Configured in BLHeli for PROPSIN
    - (ESC#,Direction[CW,CCW]) | (1,CW) (2,CCW) (3,CCW) (4,CW)

7/20/2021 - Prototype Manual Build cont.

- Not Arming Debugging
    - Plugging in battery

- OSD shows bootup sequence
- Barometer calibration successful
- EKF2 IMUO tilt alignment successful
- Prearm: compass not healthy
- Prearm: Logging failed
- Prearm: EKF2 still initialising
    - Stays there
- Added SDCard to fix Logging Failed
- Removing irrelevant prearm checks
- Removing all compasses from compass tab since FC has no compasses
- Initialisation now completes
- Arms successfully

7/21/2021 - Prototype Manual Build cont.

- Finishing Prototype Build
    - Organized and secured all parts to temporary frame
    - Top plate secured on, added battery strap and 3in props.
    - Started basic hover test, testing for runaways
- Test 1: 3in props, 3s battery, smoke stopper on
    - Drone armed successfully, smoke stopper trips when throttle is added
- Test 2: 3in props, 3s battery, no smoke stopper
    - Drone will throttle out to 100%
    - Not enough lift to hover
    - Drone will roll and yaw correctly, pitch control is opposite than normal
- Test 3: 5in props, 3s battery, no smoke stopper
    - Drone now oscillates when throttle is added
    - Drone can hover, but the oscillations make it unsafe/hard to fly
- Conclusion: The drone is configured correctly and capable of flight once tuned.
    - Oscillation issue could be improperly secured wires/parts moving in flight, PIDs, or bad props

9/14/2021 - V1 Automated Drone Build

- Connecting RF module
    - RC reciever removed
    - Module wired power to 5v, ground to gnd, rx to r6 - No connection
    - Switched protocol to mavlink - No connection

    - Connected tx to t6 - No connection
    - Switched protocol to mavlink 2 - No connection
- Moved rx and tx to UART 1 - No connection
- Switched protocol to mavlink 1 - no connection
- Switched rx and tx wires - Connected and heartbeat recieved
- Migrating Frames
    - All electronics removed from old frame
    - Motors removed from esc
    - Motor Wires extended
    - Motors reconnected to esc
    - Parts temporarily mounted
- Everything is now migrated and mounted

9/20/2021 - V1 Automated Drone Build cont.

- Flight controller flipped to correct orientation
- RC receiver reconnected for manual control
- GPS soldered to UART3 5v ground and extra GPS specific ports
- Motor data wires moved around to compensate for ESC orientation change
- Everything connected and mounted in temporary positions for testing

9/21/2021 - V1 Automated Drone Build cont.

- FC connected to mission planner
- RC receiver reconfigured
- All motors connect and spin
- Motor shafts contact carbon and need additional clearance
- Motor direction fixed but motors aren't mapped correctly
- Fixed motor order
- Attempted to configure GPS but got errors

9/25/2021 - V1 Automated Drone Build cont.

- GPS rewired correctly, switching data wires
    - GPS reconfigured in Mission Planner
- GPS connects and accurately tracks
    - Battery temporarily strapped to arm

9/25/2021 - Mission Planner Drone Test

- Build brought out to field and tethered to the ground for testing
- Sent simple commands using Mission Planner
- Was able to hover, fly to waypoints and land in a shaky manner
- Postponing any additional testing till mounts are finished and camera is mounted

10/2/2021 - Mission Planner Drone Test cont.

- Previous to testing, new brackets added and camera has been mounted
- Camera mount rotates around rail and needs to be fixed
- Drone tethered and sent commands from Mission Planner
- Drone took a hard landing from running out of battery
- ESC fire on while hovering on next flight
- Fire most likely due to ESC wires touching after hard landing
- Drone taken apart for cleaning

10/6/2021 - Post Fire Rebuild

- New ESC connected to FC
- All peripherals except GPS and motors reconnected
- Everything seems to be in working order

10/10/2021 - Post Fire Rebuild cont.

- Motors and GPS reconnected
    - Everything has been remounted and reconfigured
    - Hover tested and flies successfully
- Camera has OSD menu appearing due to melted wires
- Camera wires cut and rewired, other melted parts removed/fixed
- Stack is now crooked by about 10 degrees but doesn't affect flight
- Still needs GPS/battery and new camera mount and motor shaft clearance

10/11/2021 - Mission Planner Drone Test

- Drone flown without tether
- Sent commands from Mission Planner
- Was able to hover, fly to waypoints and land
- This time was much more accurate and consistent than last test
- Tested with basic script and was able to take off and land

10/15/2021 - UI Drone Testing

- Drone connected and ran with UI
- Drone goes through most of the waypoints but sometimes will randomly stop
- Does not RTL at the end and force landing commands don't work

10/26/2021 - UI Drone Testing cont.

- Battery mounts added
- Updated Camera mount added
    - Tuned drone for 6s battery
- Drone crashes when switched to automatic modes
    - Switching back to 3s fixes it
- Drone ran with UI, still inconsistently goes through waypoints
- Drone still does not RTL at the end and force landing commands still don't work
- After some configuration, computer picks up drone video and runs CV algorithm separate to the UI scripts

11/8/2021 - UI Drone Testing cont.

- After reviewing flight logs, we realized the flight field has radio interference which stops the connection between the drone and UI
- Moved away from radio interference
- Drone ran with UI goes through all waypoints but doesn't RTL
- Bad GPS coverage ends flight testing early

11/10/2021 - UI Drone Testing cont.

- UI Code updated
- Drone ran with UI goes through all waypoints and RTL's
- Forced landing commands work

11/12/2021 - Computer Vision Drone Testing

- Connected Computer Vision side of UI
- Ran drone with UI without any target in the field
- Got through all waypoints and returned
- Height seems somewhat inconsistent
- Tried testing with CV, wasn't working due to compatibility issues

11/16/2021 - Computer Vision Drone Testing cont.

- Fixed compatibility issues with CV
- Drone ran with UI and targets in the field
- CV picks up targets and returns drone
- Adjusted flight height to help CV find targets
- Height is still inconsistent but drone fully completes entire end-to-end mission

8. Administrative Content

8.1. Budget Discussion

The budget for this project is $500, but we expect the cost to be lower. Our aim is to make the project affordable. This means we will not pick the most expensive parts to build the drone; but it does not mean that we will select the most inexpensive parts due to reliability concerns. To pay for all the expenses we plan on evenly dividing costs among the six members of the team. If a team member purchases a part that he/she needs, then the rest of the teammates will pay that person their fair share. To keep track of this we have a document to record all expenses; in it we record who bought a part so that we can keep track of the money and be fair to everyone.

Another reason we expect to spend less than the budgeted amount is that one of our teammates, Joseph Manalo, enjoys working with drones and is part of the FPV drone club at UCF. Joseph has some spare parts that we expect to use in the project. One of the parts is the flight controller, which is necessary for controlling the drone when it flies. At this early stage we do not know what other parts we will repurpose that Joseph may have since we are still investigating parts to use in this project.

To further reduce cost, we plan on using any free resources and services available for our team. It has not been determined but possibly could 3D print the body of the drone in UCF, which will reduce build costs. There might be other materials we source from the MAE department for free. As mentioned before, we are in the early phases of the project and all our plans are not completely certain. Below you will find a table with all the expenses and the total expected cost of the project. The table below assumes we did not use one of Joseph's flight controllers.

- More expensive parts

Part	Build Area	Cost
7 Inch Folding Props Long Range	Propellers	$3.99
HD1-XR 7" Deadcat Frame (if frame not built)	Frame	$59.99
Lumenier LUX F7 (may change)	Fight Controller	$44.99

SP Racing H7 EXTREME PX4	Fight Controller	$75.99
T-Motor Velox V2 V45A 3-6S BLHeli_32 4-in-1 ESC	ESC	$39.99
HobbyWing XRotor	ESC	$49.99
Lumenier 2407 Motors (**x4**)	Motors	$25.99 (x4) = $103.96
FrSky R-XSR Receiver	RX	$19.99
Xilo Stax	Video Transmitter	$21.99
RushTank Ultimate II	Video Transmitter	$35.99
Adafruit Ultimate GPS Breakout – Version 2	GPS Module	$39.95
RunCam Phoenix 2	Camera	$29.99
Foxeer Trex Micro	Camera	$44.99
Foxeer Lollipop V3	Antenna	$9.99
Total Base Cost		**$374.83**
Total Maximum Cost		**$444.83**

Table 8.1.A: Table of Necessary Components to Purchase & Their Anticipated Cost

Our final budget is in the following table. We were able to get most of the project paid for by the ME Department and by using parts that we already had. It should also be noted that the ESC listed is the one we intended to use on the final project, but we used a spare one after the first ESC caught on fire.

Part	Cost ($)
Flight Controller	48.99
Electronic Speed Controller	49.99
Video Transmitter	24.99
RF Telemetry Module	29.00
GPS	39.90
Camera	44.90
Motors	107.96
Propellers	7.55
Batteries (One Set)	73.98
Frame	66.23
Total Cost	493.49

ME Department Paid (Including price of items already in stock)	$345.73
What We Already Had	$73.98
Teams Total Expenses	$66.23

Table 8.1.B: Table of Final Components and Cost

8.2. Milestone Discussion

8.2.1. Individual Gantt Charts for Senior Design 2

Individual Gantt charts have been created for planning out progress for Senior Design 2, and are presented below. These Gantt charts serve as a visual guide to plot out future steps for moving forward with the project as the semester approaches.

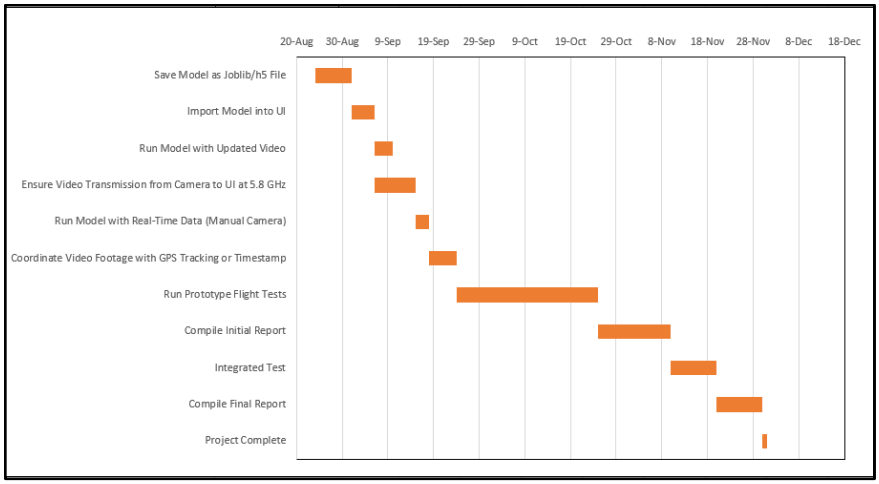


Figure 8.2.1.A: Computer Vision Gantt Chart for Senior Design 2

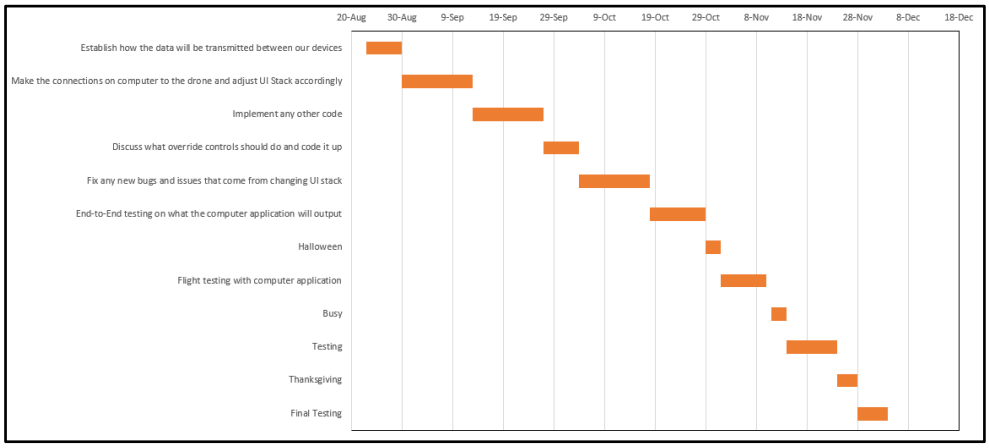


Figure 8.2.1.B: UI/UX Gantt Chart for Senior Design 2

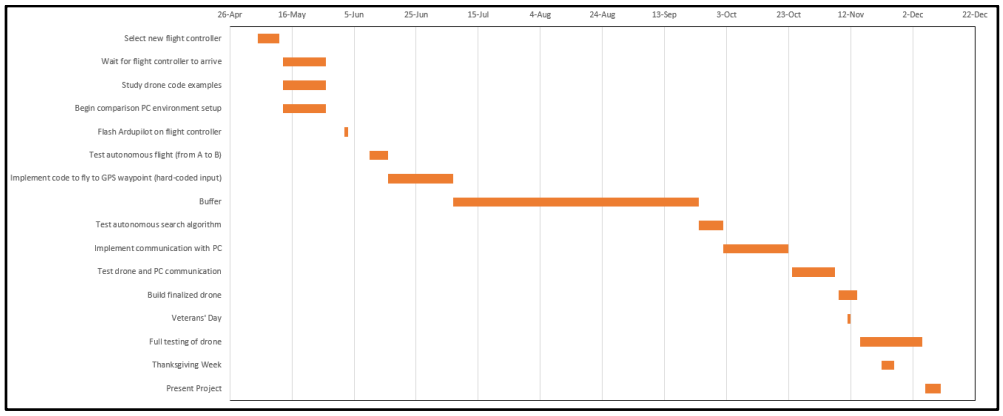


Figure 8.2.1.C: Flight Controller & Integration Gantt Chart for Senior Design 2

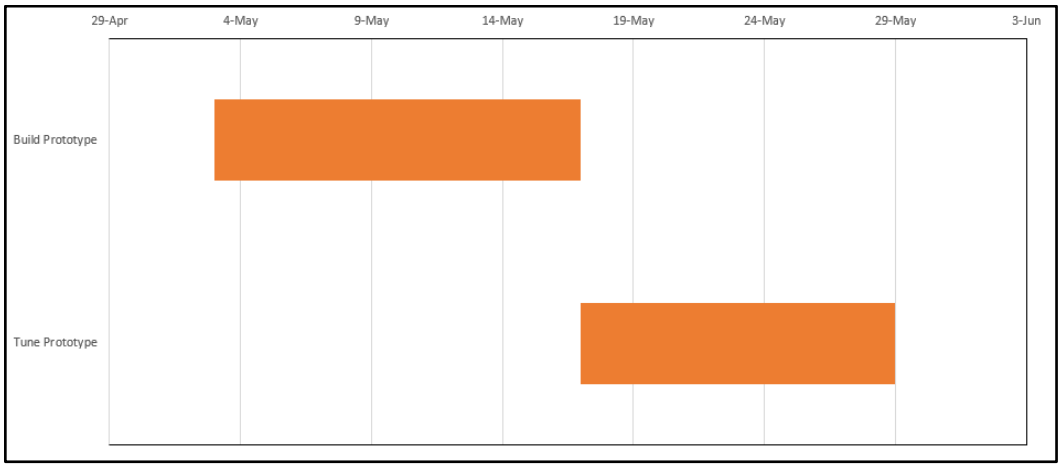


Figure 8.2.1.D: Drone Build Gantt Chart for Senior Design 2

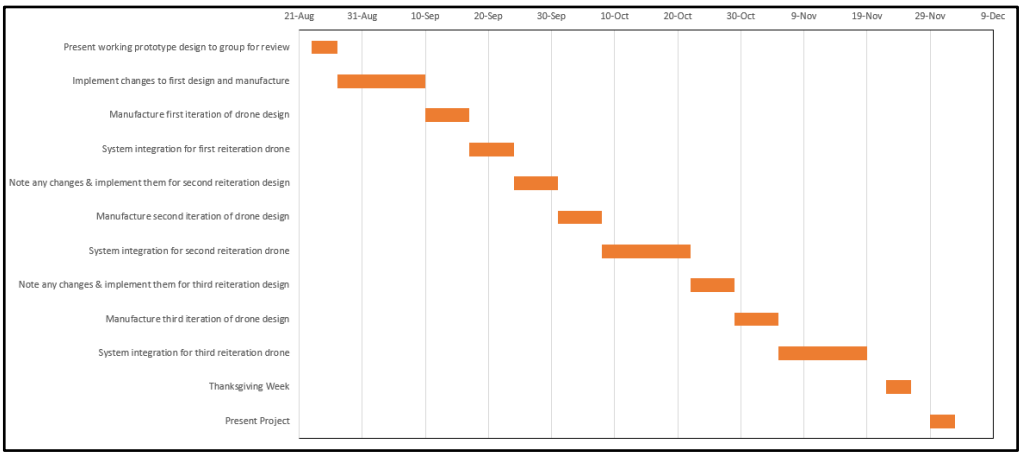


Figure 8.2.1.E: Drone Frame Design Gantt Chart for Senior Design 2

8.2.2. Overall, Administrative Gantt Chart for Senior Design 2

Below is an overall Gantt chart, factoring in administrative items such as university breaks and closures, hardware/software integration and testing, final report and testing completion, etc. The individual Gantt charts supplement the below Gantt chart in that they offer in-depth analysis of the general project components.

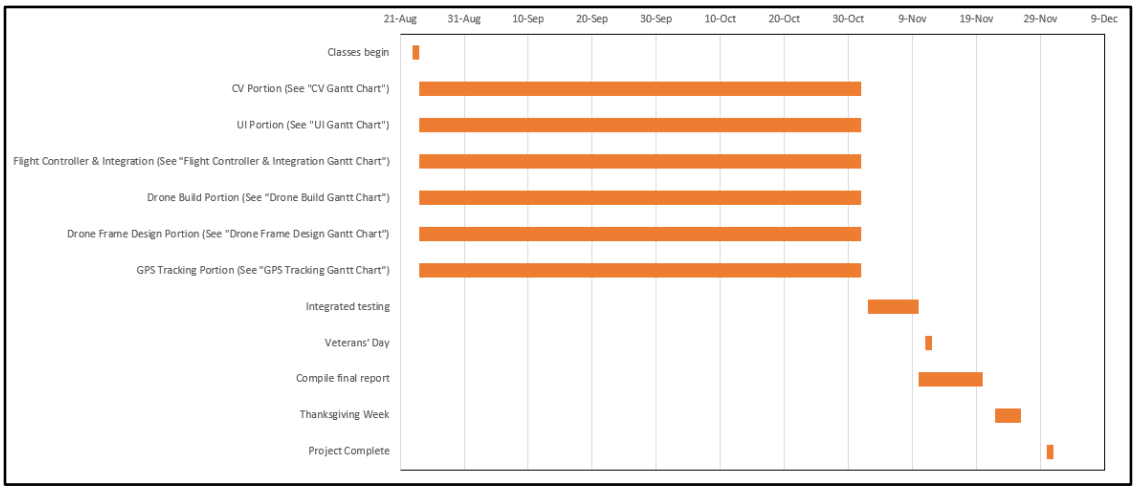


Figure 8.2.2.A: Overall, Administrative Gantt Chart for Senior Design 2

8.2.3. Project Timeline

Below is a list of the milestones that we, as a team, planned to meet to deliver our product throughout our first and second semesters:

May 3 - Initial report on CV model performance

May 7 - Save all work pertaining to project to be resumed in Fall 2021

Aug. 23 - Resume work from Spring 2021

Sept. 6 - Have first successful flights

Sept. 20 - Have five consecutive successful flight tests

Sept. 27 - Identify all feasible stretch goals to add to the project (e.g., swarm drones, etc.)

Oct. 4 - Begin work on stretch goals

Oct. 25 - Complete work on stretch goals

Nov. 8 - Have first successful flight test with added stretch goals

Nov. 11 - Veterans' Day

Nov. 19 - Have three consecutive successful flight tests with added stretch goals

Nov. 22 - 26 - Thanksgiving Week

Nov. 28 - Complete Final Design document

Nov. 29 - Complete Final Presentation

Nov. 30 - Deliver Final Presentation

As the Agile Project Management methodology was implemented in our project, these milestones turned into 2-week sprints, where major deliverables were chunked into pieces called stories and assigned to teammates. Each of the 2-week sprints focused on the current most

important feature or goal of a certain aspect of the project. These are the following goals of each sprint for the 2021 Fall semester:

Sprint 1 (8/30 - 9/13)	**Develop MVP** (minimum viable product)	Manually flyable drone as well as incorporating past work completed in SDI. This includes basic MAV-proxy design, first iteration drone frame, deciding Ground Station Computer communication method.
Sprint 2 (9/13-9/27)	**Develop a Command-ready drone**.	This includes completing a mission planner capable drone, physically tuning the actual drone and migrating electronics to MVP frame, as well as converting CV script to process live footage.
Sprint 3 (9/27-10/11)	**Basic Integrated Automated Drone and UI.**	The goal was to further establish RF communications with ground station, basic UI, and embed CV algorithm into UI.
Sprint 4 (10/12-10/25)	**Add CV Functions and Refine Drone & UI.**	Finishing any drone frame tasks for before further testing and weight reduction is in mind, as well as refining basic UI with CV functionality and integrating further communication components into MVP frame.
Sprint 5 (10/26-11/8)	**Validate possibility for Backburner Functionality and prepare for CDR Presentation.**	The primary focus was to gauge any potential add-ons that were feasible as well as prepare to present in the upcoming CDR design review presentation**.**
Sprint 6 (11/9-11/22)	**Final Report and Presentation Documentation.**	This time would be used to prepare for end-of-semester design documentation and final presentation of the project.
Sprint 7 (11/29-12/5)	**Close out SAR project and finish Senior Design II**	Complete any remaining administrative tasks.

Table 8.2.3.A: Sprint Table for Senior Design 2

While the overall sprint goals were mostly achieved, many tasks were pushed back as their realistic window of time was not realized. The UI and CV algorithms, for example, were not marked until two sprints after their assigned due dates. This was due to several complications, as

well as miscommunication of the work progress between team members. Other complications included delays in drone frame components due to manufacturing issues with the 3D printer and delays in testing of newly designed frame components.

Overall, while not a perfect solution for our current collaborative capabilities as a team, the 2-week sprints greatly helped our team to focus on the most important issues at hand, and finish out on them before moving to less important tasks.

One example of this was the computer-drone communication dilemma at the beginning of the semester. Instead of explicitly focusing on a physical, mission capable manual drone as Sprint 1 required, we were encouraged to tackle the issue of communication, which surely would have caused delays if procrastinated on. Instead, this major issue was resolved in Sprint 1, and allowed for the rest of the project to continue on without any critical delays.

Our progress has been steady, and each team member is able to track current progress and workload easily with the Jira Scrum Board. Team communication can always be improved upon, but the ease of project planning for this semester has proven itself to be evident in our finishing a successful mission a month before delivering the final presentation.

In the future, clearer long-term goals will need to be planned, and further broken down into respective stories and tasks. This will bridge the apparent gap between Waterfall and Scrum project management, which is a valuable skill to have when managing complex efforts.

Appendix I: References

Drone Frame Design References

[1] Rodriguez, Emily and Jain, Parul. *Airframe*. Britannica.com, 2019, Feb. 21. https://www.britannica.com/technology/airframe.

[2] Nodes, D. (Ed.). (n.d.). *How to choose drone frame for racing or freestyle?* Drone Nodes. https://dronenodes.com/drone-frame-racing-freestyle/.

[3] Kuantama, Endrowednes and Tarca, Radu. *Quadcopter thrust optimization with ducted-propeller*. Annual Session of Scientific Papers IMT ORADEA 2017, 2017, https://www.matec-conferences.org/articles/matecconf/pdf/2017/40/matecconf_imtoradea2017_01002.pdf.

[4] RCModelReviews. *How ducting a propeller increases efficiency and thrust*. YouTube.com, 2015, July 22, https://www.youtube.com/watch?v=Cew5JF8q6eY.

[5] *PC-1 Multipurpose Quadcopter*. Airforce-Technology.com. https://www.airforce-technology.com/projects/pc-1-multipurpose-quadcopter/.

[6] Lockheed Martin Indago 3 - UAV. LockheedMartin.com. https://www.lockheedmartin.com/en-us/products/indago-vtol-uav.html.

[7] Boon, M.A. et al. *Comparison of a Fixed-Wing and Multi-Rotor Uav for Environmental Mapping Applications: a Case Study*. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2017, Aug. 23, https://ui.adsabs.harvard.edu/link_gateway/2017ISPAr42W6...47B/doi:10.5194/isprs-archives-XLII-2-W6-47-2017.

[8] *Armoured Fighting Vehicles: Tracks vs. Wheels*. GlobalDefenceTechnology.com. https://defence.nridigital.com/global_defence_technology_oct19/armoured_fighting_vehicles_tracks_vs_wheels#.

[9] GEN 3.6 Search and Rescue. Aeronautical Information Publication - AIP. https://www.faa.gov/air_traffic/publications/atpubs/aip_html/part1_gen_section_3.6.html
.

[10] *Airborne Drones Discusses Drones for Commercial Use - Fixed Wing or Long Range Multi-Rotor?* Airborne Drones, 2017, Aug. 1, https://www.airbornedrones.co/fixed-wing-long-range-multi-rotor/.

[11] *UAV Transforms between Fixed-wind and Quad-rotor Mid-flight*. Available Technologies.com, https://license.umn.edu/product/uav-transforms-between-fixed-wing-and-quad-rotor-mid-flight.

[12] Damlamian, Herve. *Use of Multirotor and Fixed Wing UAV to Assess Impacts of TC Pam (Cat. 5) in Vanuatu*. Pacific Community, https://www.preventionweb.net/files/45270_216.pdf.

[13] LAND SEARCH AND RESCUE ADDENDUM. National Search and Rescue Committee, 2011, Nov., https://www.dco.uscg.mil/Portals/9/CG-5R/nsarc/Land_SAR_Addendum/Published_Land%20SAR%20Addendum%20(1118111)%20-%20Bookmark.pdf.

[14] *G1000 Integrated Flight Deck: Search and Rescue Pilot's Guide*. Garmin. 2007, https://www.gocivilairpatrol.com/media/cms/Garmin_SAR_Software_Pilots_Guide_AA_____C776EBBA500.pdf.

Flight Controller References

[15] Shaffer, Christopher. *GPS/GIS*. The International Encyclopedia of Primatology, pp. 1 - 2, https://www.researchgate.net/publication/316658355_GPSGIS.

[16] Jost, Danny. *What is an IR Sensor?* FierceElectronics.com, 2019, June, https://www.fierceelectronics.com/sensors/what-ir-sensor.

Computer Vision References

[17] Brownlee, Jason. *A Gentle Introduction to Computer Vision*. MachineLearningMastery.com, 2019, March 19, https://machinelearningmastery.com/what-is-computer-vision/.

[18] Brownlee, Jason. *Supervised and Unsupervised Machine Learning Algorithms*. MachineLearningMastery.com, 2016, March 16, https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/.

[19] Redmon, Joseph et al. *You Only Look Once: Unified, Real-Time Object Detection*. Cornell University, 2015, Jun. 8, https://arxiv.org/abs/1506.02640.

[20] Tsang, Sik-Ho. *Review: YOLOv1 - You Only Look Once (Object Detection)*. TowardsDataScience.com, 2018, Oct. 17, https://towardsdatascience.com/yolov1-you-only-look-once-object-detection-e1f3ffec8a89.

[21] Singh, Aishwarya. *Feature Engineering for Images: A Valuable Introduction to the HOG Feature Descriptor*. AnalyticsVidhya.com, 2019, Sept. 4, https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/.

[22] Tan, Mingxing. *EfficientDet: Scalable and Efficient Object Detection*. Google Research, Brain Team, 2020, Jul. 27, https://arxiv.org/pdf/1911.09070v7.pdf.

UI/UX References

[23] Detimo. *Python Flask: pros and cons*. Dev.to, 2019, Nov. 18, https://dev.to/detimo/python-flask-pros-and-cons-1mlo.

[24] Google Developers. *Google Maps Platform Documentation*. Google Maps Platform. https://developers.google.com/maps/documentation.

[25] Convert NMEA sentence Lat and Lon to Decimal Degrees. RaspberryPi.org, 2017, Feb. 21, https://www.raspberrypi.org/forums/viewtopic.php?t=175163.

GPS References

[26] Jost, D. (2019, July 30). *What is an IR sensor?* FierceElectronics. https://www.fierceelectronics.com/sensors/what-ir-sensor.