Autonomous Target-Finding UAV

Final Project Document

Sponsor: Lockheed Martin

University of Central Florida
Department of Electrical and Computer Engineering
Dr. Lei Wei
Senior Design I
11/03/2017

Group 5
Dominic Williams (CpE)
Ley Nezifort (CpE)
Maverick Dusan (EE)
Julian D. Quitian (CpE)

Project Contacts: Dr. Kurt Stresau, Dr. Mark Steiner (UCF MAE), Lou Glaros (LM/UCF)
* Interdisciplinary project including additional MAE and CS groups & stakeholders

Table of Contents

List of Tables

Table of Figures

1. Executive Summary

This project aims to improve on current technologies for obstacle avoidance and target detection with symbology for tracking. There is a movement to improve the detection and avoidance capabilities on UAV aircraft. Common problems include blind spots due to inadequate numbers of object detection sensors as well as their strengths and response times. An emphasis by the customer was placed on synchronized target detection with symbology for tracking prey. This feature is important for future applications such as for defense application, for search and rescue missions, and optimization for general consumer uses. A functional and sharp targeting system eliminates mistakes in targeting objects and safety concerns in mission failure.

The primary user of this project was the customer company hosting the Robocopters Competition for UCF Senior Design; Lockheed Martin (LM) Missiles and Fire Control Research Sensor Systems and Technologies. Secondary users of the project are the University of Central Florida (UCF) students developing the project for credit in the UCF Senior Design course. Tertiary users include potential market users such as the US government.

Each team designed, built, and tested an autonomous unmanned aerial vehicle with commercial off-the-shelf (COTS) components that detected, tracked and collided with prey hobby drones. Student engineers working on the project were supposed to incorporate a targeting system that allowed for video streaming with symbology around prey or obstacles. The vehicle must also include certain flight modes with safety features. All features were examined in a course environment by the customer in the middle of April of 2018. Course objectives were to test capacity of the UAV by observing its capability of obstacle detection and avoidance as well as its tracking system ability to detect, target, and collide with prey UAV.

This project and incorporated research benefits senior design students in having an interesting real application of a relatively new technology of autonomous flight and benefits Lockheed Martin as well in connecting with future engineers and acquiring innovative solutions to the technology. The teams finished the copter’s concept design by the end of the 2017 fall semester and have presented the design in a Preliminary Design Review (PDR) to the sponsoring customer in January 2018. This report will cover Green Team’s design process and concept.

2. Project Overview

This project was meant to be a joint effort between subgroups of interdisciplinary students. The team first acquired a bare bones multirotor frame. Each subgroup was then in charge of the modules necessary to develop the RoboCopter's systems, including propulsion, avionics, and prey-recognition. The drone must be lightweight, agile enough to make any necessary movements, and completely autonomous once in flight. It was also expected to last at least the duration of the competition and be accurate enough to take down prey. Additionally, a protective cage was required in order to protect the RoboCopter from hard collisions. This section will focus on the design requirements and specifications pertaining to the electronics and software systems for the drone.

2.1 Problem Statement

The use of Unmanned Aerial Vehicles (UAVs) has been rapidly increasing for many different purposes including photography, agriculture, landscaping, and military surveillance. Most Commercial-Off-the-Shelf (COTS) UAVs, however, are remote controlled and do not possess the ability to autonomously modify their behavior based on changes in their environment. This project, sponsored by Lockheed Martin Missiles and Fire Control Applied Systems Sensor Research and Technologies, aims to serve as a challenging ground in which mechanical, aerospace, electrical, computer, and software engineers must work harmoniously to develop an autonomous UAV system encompassing propulsion, telemetry, aviation, and robot vision systems.

2.2 Goals & Objectives

The goal of this RoboCopter project was to develop an autonomous drone that could identify, track, and attack multiple prey UAVs in an obstacle course by causing a controlled-collision. Target prey, which can vary in type and size, are to be recognized through the RoboCopter's First-Person-Video (FPV) camera and then marked/highlighted; the recorded video stream was sent to a ground station outside of the stage/combat zone for further analysis. Multiple decoys and prey UAVs were expected, which means that the RoboCopter must be able to make quick decisions based on the input it receives from the environment.

RoboCopter acted as a combat UAV, where aggressive action came in the form of controlled collisions. Once completed, the design entered a competition which tested its features in a real-time environment. Our objective was for the RoboCopter to successfully fly, designate and attack prey, send the required data to the ground station, and land once the competition ends.

The following is a core list of minimal high-level objectives that our RoboCopter must be able to perform, as defined by our customer:

- Discriminate between decoys and real targets
- Navigate toward prey UAVs causing a collision while avoiding obstacles
- Transmit live First Person Video (FPV) to a ground station
- Combine two sensor modalities (optional)
    - Sensor modality examples: Mid-wave infrared imagery, Lidar point clouds, visible spectrum imagery, Radar returns, etc.
    - May aide in ability of RoboCopters to complete the project competition successfully

2.3 Customer Requirements

The requirements of our RoboCopter have been established by our sponsor and customer, Lockheed Martin. These requirements are set in order to standardize competition guidelines and simplify the design process. The list below notes the physical and functional requirements outlined by Lockheed Martin. These are the critical performance parameters as defined by our sponsor; they are further explained in their respective sections:

- Maximum UAV size of 4 ft. x 4 ft. x 4 ft. (L x W x H)
- Total price must be under $2000, with a maximum demonstrated cost of $1500.
- Minimally use 1 sensor modality
- Equip RoboCopter with a protective cage that still satisfies the physical platform requirement
- The drone must project FPV (first person view) from the drone to a data link on the ground station for judges to view.
- Competition Course Mission:
    - UAV must be able to autonomously operate and withstand at least two 10 minute rounds of competition.
    - UAV must stay within the boundaries of the arena.
    - UAV must be able to perform functions and avoid obstacles along the way. Failure to do so will result is one point taken away for each obstacle collision.
    - UAV must be able to track target prey UAV and instigate a controlled collision with prey detected.
- Symbology & Video Datalink
    - Provide video imagery overlays that highlight automatically detected vs. tracked targets.
        - Use blue symbology boxes for detections and red symbology boxes for tracks.
    - Utilize a wireless video data link to a ground station computer and display.
        - This portion of the project does not count against “as-demonstrated” cost.
    - Failure to broadcast video with appropriate symbology leads to 15 point penalty.

- Operational Modes (<u>all modes other than Repair require autonomous obstacle avoidance</u>)
  - Search: autonomously detect & track prey UAVs
  - Pursue: autonomously perform persistent tracking, navigation, and collision with prey UAVs
  - Reset: autonomously fly to another location within the arena and then reset search/pursue data buffers
  - Repair: manually fly to a teammate for repair

Evaluation Criteria	**Weight of Importance**
Autonomous Flight Capable UAV Copter	**30%**
1st Person POV Video Stream	**20%**
Prey Detection with Symbology	**20%**
Obstacle Avoidance	**15%**
Flight Modes	**10%**
Safety Cage	**5%**

Table 1: Component Priority Table

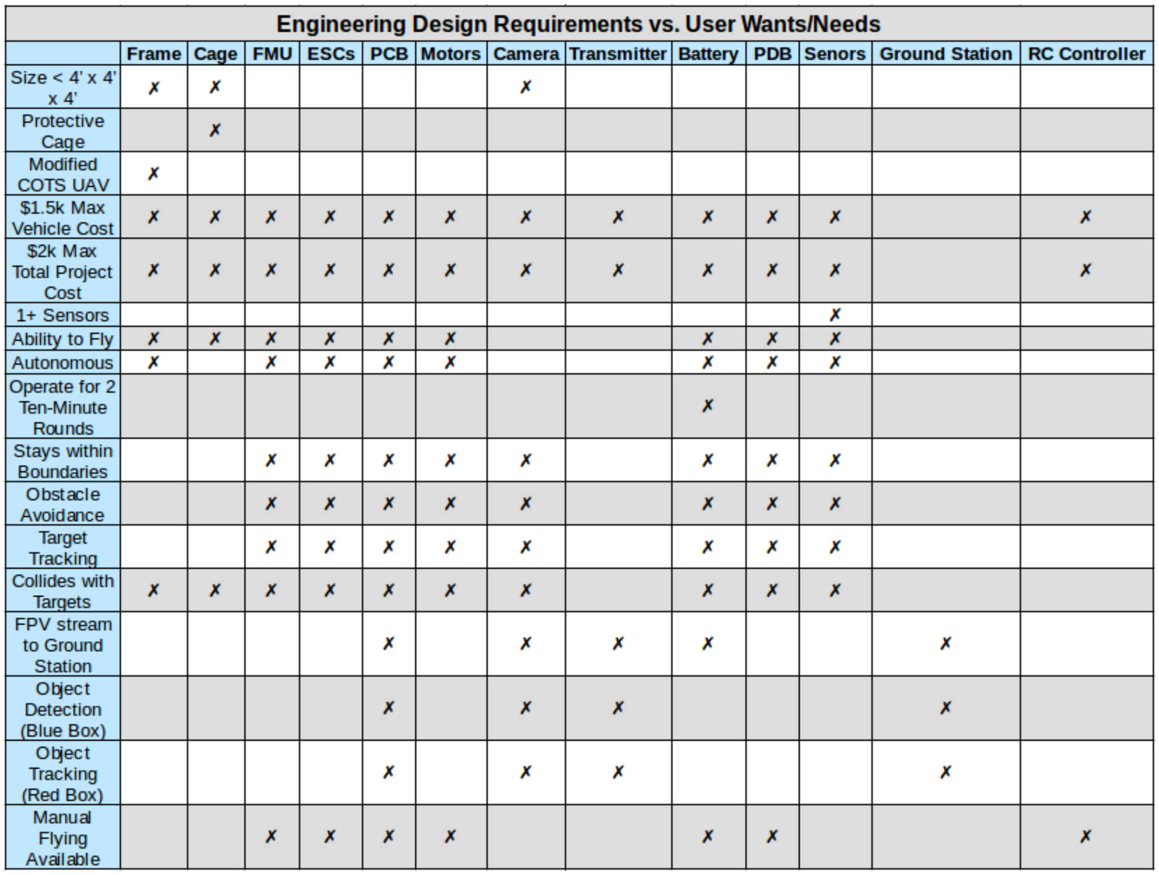
2.3.1 Design Requirements

Given the nature of UAVs and the customer requirements for this competition, there were a minimum number of components that was to be expected in our RoboCopter, regardless of whether said components are built from scratch or bought from a commercial provider:

- Flight Controller - May come in the form of a commercially available piece of hardware, or a combination of a microcontroller, a Real Time Operating System (RTOS), and the corresponding software to deal with sensor input.
- Electronic Speed Controller (ESC) - Help mediate motor rotation speed.
- Motor(s) - Used to rotate propellers; connected to ESC and Flight Controller.
- Printed Circuit Board (PCB) - Will integrate major UAV parts on a single board. It will also act as the main processor for flight planning. Must be designed as a diagram and built through a commercial PCB manufacturer.
- Protective Cage - Maximum size of drone plus cage must be 4 ft. x 4 ft. x 4 ft. (L x W x H).

In order to satisfy the requirements specified by our customer and maximize ease of development, our RoboCopter must meet the following requirements:

- Dynamically stable in flight
- Consistent and accurate autonomous flight
- UAV-to-ground wireless communication capacity
- Ability to receive fail-safe master landing/stabilization commands for testing and UAV protection
- On-board camera able to capture live video
- Image processing software which processes on-board camera stream for object recognition and autonomous flight
- Ability to recognize both prey and obstacles/decoys and modify behavior accordingly
- Inertial Measurement Units (IMU) for reading environment states, including a proper noise filtering algorithm for these sensors
- Ability of flight controller to send , Pulse-Position Modulation (PPM), Pulse-Width Modulation (PWM) or OneShot signal to ESC for motor rotation speed control



Engineering Design Requirements vs. User Wants/Needs

	Frame	Cage	FMU	ESCs	PCB	Motors	Camera	Transmitter	Battery	PDB	Senors	Ground Station	RC Controller
Size < 4' x 4' x 4'	✗	✗					✗						
Protective Cage		✗											
Modified COTS UAV	✗												
$1.5k Max Vehicle Cost	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗		✗
$2k Max Total Project Cost	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗		✗
1+ Sensors											✗		
Ability to Fly	✗	✗	✗	✗	✗	✗			✗	✗	✗		
Autonomous	✗		✗	✗	✗	✗			✗	✗	✗		
Operate for 2 Ten-Minute Rounds									✗				
Stays within Boundaries			✗	✗	✗	✗	✗		✗	✗	✗		
Obstacle Avoidance			✗	✗	✗	✗	✗		✗	✗	✗		
Target Tracking			✗	✗	✗	✗	✗		✗	✗	✗		
Collides with Targets	✗	✗	✗	✗	✗	✗	✗		✗	✗	✗		
FPV stream to Ground Station					✗		✗	✗	✗			✗	
Object Detection (Blue Box)					✗		✗	✗				✗	
Object Tracking (Red Box)					✗		✗	✗				✗	
Manual Flying Available			✗	✗	✗	✗			✗	✗			✗

Table 2: Requirements Relationship Matrix

2.3.2 Competition Guidelines

The Lockheed Martin RoboCopter competition consists of four teams and their autonomous predator drones competing for the most points by disabling enemy prey drones. Points was awarded and deducted based on the ability of the predator drone to quickly identify and track the prey drones while avoiding obstacles placed on the field.

Points are awarded as follows:

- +5 points for colliding with a prey drone without knocking it to the ground.
- +10 points for colliding with a prey drone and knocking it to the ground.
- -1 point for colliding with an obstacle.
- -15 for failing to provide real-time FPV video.

Points can also be deducted if the drone obviously leaves the field of play, the amount of points deducted is still to be determined. Drones will compete on a competition arena outlined in the figure below, with defined boundary lines, and may not fly higher than 20 to 30 feet.

The drone must be protected by a cage which can not exceed four cubic feet, including any appendages extending outside of the cage. The drone must make use of at least one type of sensor to detect and track the prey, but it is suggested to use at least two. The drone must also have four modes of operation for the competition.

Required Operational Modes:

- Search mode: Predator drone is autonomously searching for and identifying prey drones to pursue.
- Pursuit mode: Autonomously tracks the prey drone and attempts to collide while avoiding obstacles.
- Reset mode: Drone autonomously returns to original position to resume searching for prey drones.
- Repair mode: Flight controller switches to be controlled manually to the team’s “medic” to replace any damaged parts or a dead battery.

The drone must also transmit a real-time, first person view, stream from the camera onboard the drone. The video must contain symbology representing the actions of the searching and tracking algorithms. A blue box must be placed around identified drones and a red box must be placed around drones that are being pursued.

Each team was able to participate in two, ten minute rounds. The best score of the two rounds was the team’s final score. Scores was determined by the four judges at each corner of the arena. Any points awarded or deducted was left to discretion of these four judges, the average of their recorded scores was the score used for the round.

Prey drones was randomly swapped in and out by designated prey drone pilots because of their short flight time. They are about 4 x 4 x 4 inches with a flight time of about 5 minutes.To confuse the drone detection, pictures of the prey was placed as obstacles, as well as balloons, streamers, and office furniture. The complete list of obstacles are still to be determined.

2.3.3 Operating Environment

- Competition Course
  - The competition was held in a outdoor arena. The UAV must be able to operate autonomously and manually in outdoor conditions. In the case that there are different wind speeds throughout the rounds, the UAV will maintain stability during the search and pursue of a prey.
- Course Overview
  - A visual representation of the course provided by LM can be seen below. The outer dimensions of course are outlined in the figure. The drone will start in the color coded boxes labeled “Start Area A” and “Start Area B”, with prey starting in various locations. Around the robocopter arena was an area 5 feet thick that the pilots and medics was positioned in. A judge was positioned at each corner of the field with a total of four judges. The spectators was behind the judges and away from the arena.

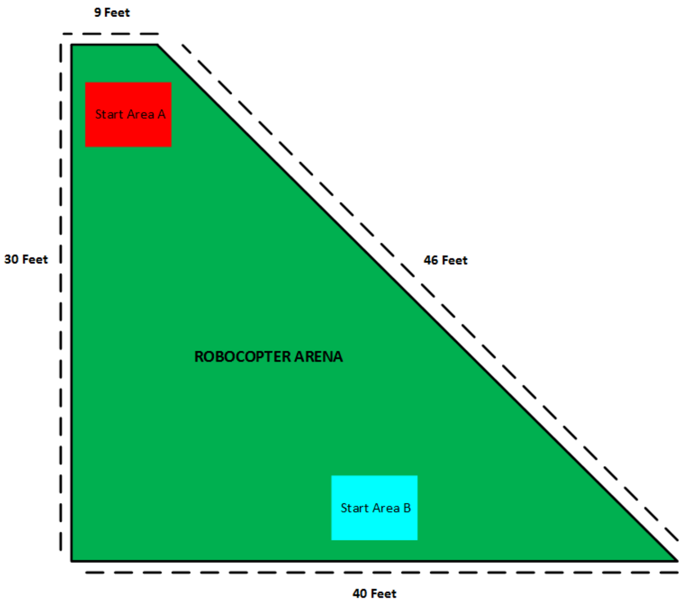


Figure 1: Competition Arena

- Obstacles
  - The course was changed from the original competition area design and removed obstacles that stood upright. Instead, decoy drones were 3-D printed and placed on the ground around the arena.

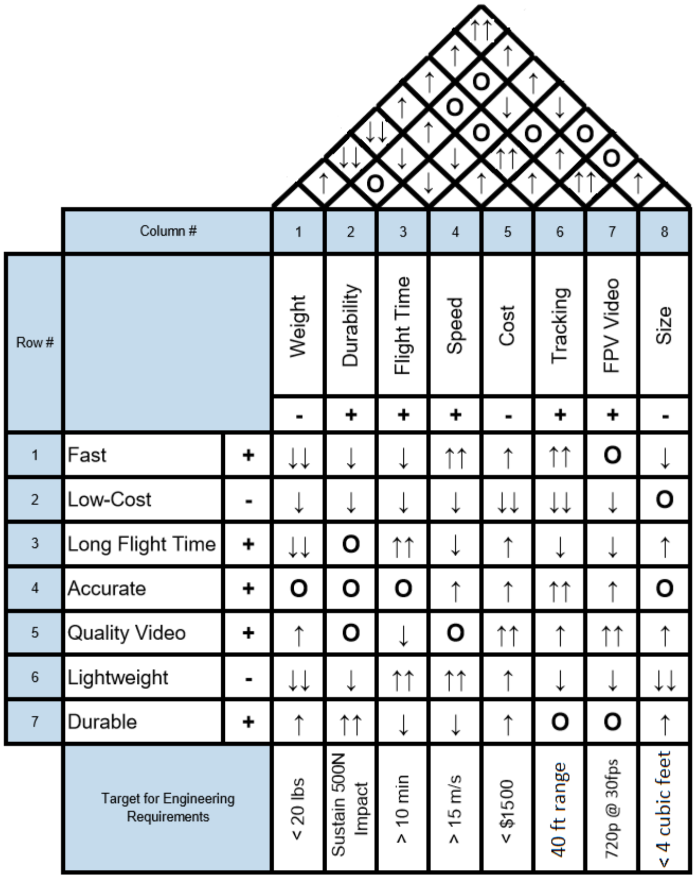
2.3.4 Financial Requirements

The maximum budget allowed was $2000, while the “as-demonstrated” maximum budget is $1500. The as-demonstrated cost is the cost of all the components on the drone itself. This means $500 dollars for things like the ground station for the FPV video, battery chargers, backup components, etc.

In order to request funds, the MAE team must contact Lockheed Martin and follow the purchasing procedure set up by the UCF Foundation. Components purchased out of pocket are still required to be recorded in the final drone budget. Those who purchase out of pocket components have the potential of being reimbursed by providing proof of purchase.

2.3.5 House of Quality

The House of Quality below is a collection of the customer's wants and desires, as well as some engineering characteristics that are deemed important to the group so that those desires can be met. The matrix helps to explain correlations between those desires and engineering characteristics, thus allowing the group to prioritize them throughout the duration of the project.



	Column #		1	2	3	4	5	6	7	8
Row #			Weight	Durability	Flight Time	Speed	Cost	Tracking	FPV Video	Size
			-	+	+	+	-	+	+	-
1	Fast	+	↓↓	↓	↓	↑↑	↑	↑↑	O	↓
2	Low-Cost	-	↓	↓	↓	↓	↓↓	↓↓	↓	O
3	Long Flight Time	+	↓↓	O	↑↑	↓	↑	↓	↓	↑
4	Accurate	+	O	O	O	↑	↑	↑↑	↑	O
5	Quality Video	+	↑	O	↓	O	↑↑	↑	↑↑	↑
6	Lightweight	-	↓↓	↓	↑↑	↑↑	↑	↓	↓	↓↓
7	Durable	+	↑	↑↑	↓	↓	↑	O	O	↑
	Target for Engineering Requirements		< 20 lbs	Sustain 500N Impact	> 10 min	> 15 m/s	< $1500	40 ft range	720p @ 30fps	< 4 cubic feet

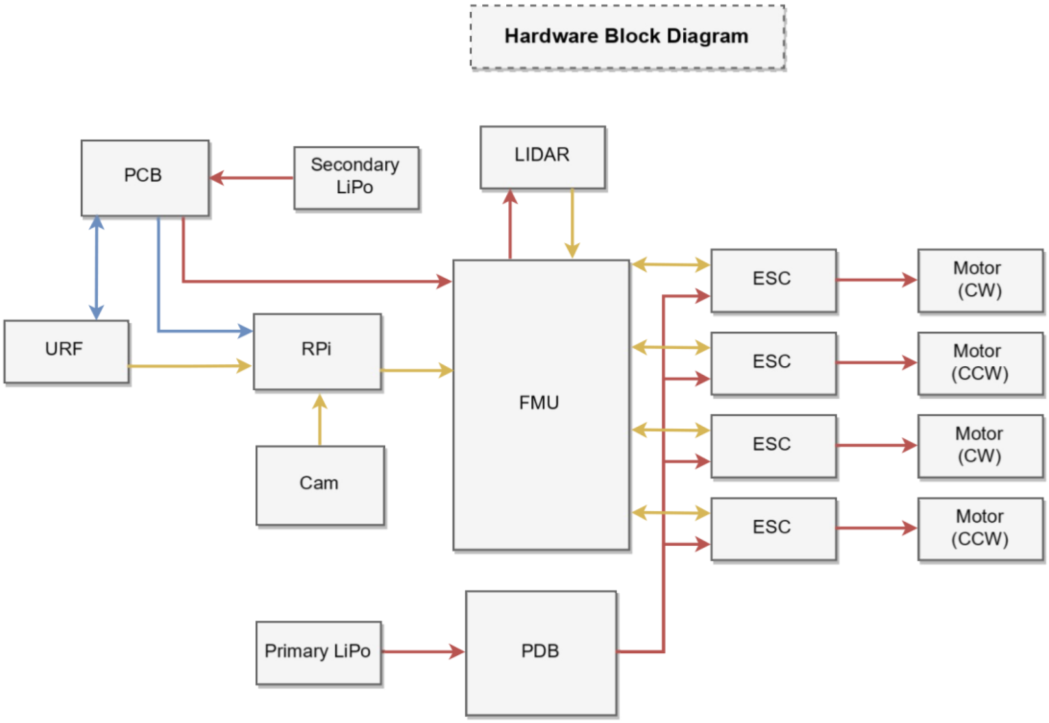
Legend

↑↑	Strong Positive Correlation
↑	Positive Correlation
O	No Correlation
↓	Negative Correlation
↓↓	Strong Negative Correlation
+	More Important
-	Less Important

Figure 2: House of Quality

2.3.6 Hardware Block Diagram

Below is the initial hardware block diagram for our project. It includes all of the electrical components that were incorporated into the project. No specific component decisions are included in this diagram. The legend explains the acronyms that are used in the diagram as well as the members who are responsible for doing majority of the research and design decisions for the respective section. The structure of this diagram is also subject to change.



Legend

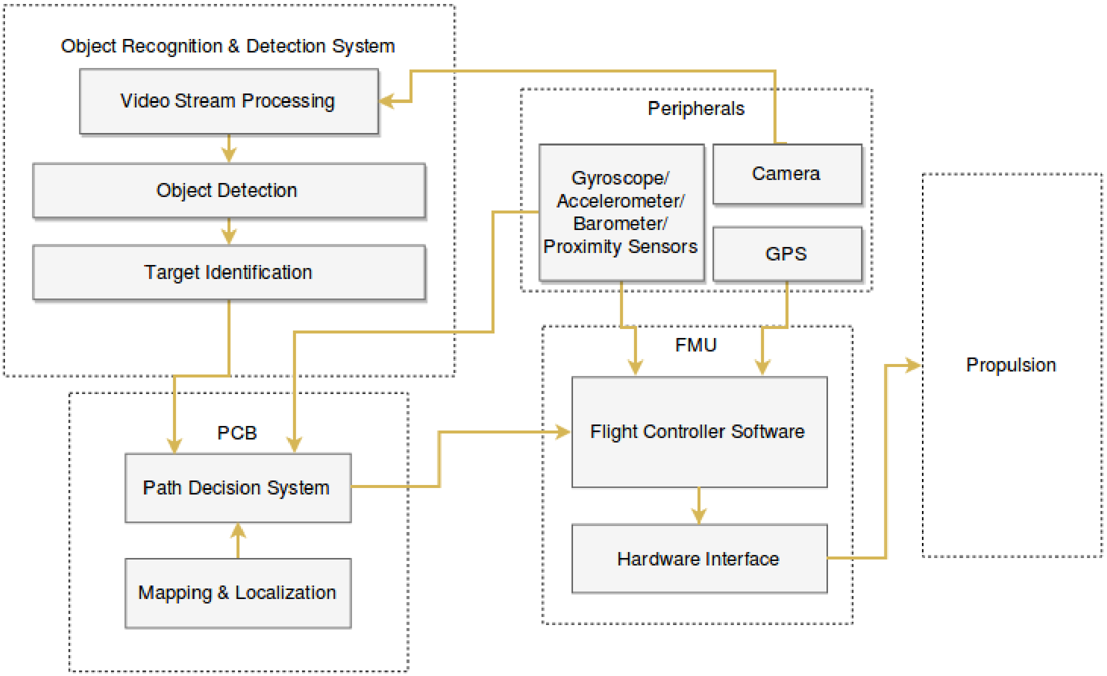
→ Data
→ Power
→ Both

Block Title	Description	Member(s) Responsible
FMU	Flight Management Unit	Julian
ESC	Electronic Speed Controller	Maverick
Motor (CW)	Motor (Clockwise)	All
Motor (CCW)	Motor (Counter-Clockwise)	All
Primary LiPo	5400 mAh Lithium Polymer Battery	Maverick
Secondary LiPo	1000 mAh Lithium Polymer Battery	Maverick
PDB	Battery Eliminator Circuit	Maverick
Cam	Mobius Action Cam	Ley
RPi	Raspberry Pi 3B	Julian/Ley
URF	Ultrasonic Range Finder (x5)	Dominic
PCB	Printed Circuit Board	Maverick
LIDAR	Light Detection Range Sensor	Ley/Dominic

Figure 3: Hardware Block Diagram

2.3.7 Software Block Diagram

The diagram below is the initial software block diagram for our project. The diagram is split up into five subsections that separate the software aspect of this project based on functionality. The sections include object recognition and detection, peripherals, the flight management unit, propulsion, and the software components to be controlled using the PCB. As with the hardware block diagram, this is subject to change as more information is gathered about the different software systems.



Legend

→ Data/Control

Block Title	Status	Member(s) Responsible
Flight Management Unit (FMU)	Testing	Julian
Flight Controller Software	Installed	Julian
Hardware Interface	Research	All
IMUs (Gyro, Accelerometer,etc...)	Acquired	Ley/Maverick
Camera	Research	Ley/Dominic
Vision System	Research	Ley/Dominic
Prey/Obstacle Identification System	Research	Ley/Dominic
Path Decision System	Research	Julian/Ley
Propulsion Systems	Research	Maverick

Figure 4: Software Block Diagram

3. Trade Studies and Downselection

Given the various topics that the design of an autonomous aircraft covers, most of the design process for this project consisted of preliminary research of the various components, concepts, and algorithms that must be considered. This section contains all the research that was done for this project, as well as design considerations, component options and alternatives, and decisions. Although most of this research consists of material directly related to the avionics and vision systems of the RoboCopter, it will also overlap with some of the work being performed by the MAE and CS student groups.

3.1 Flight Controller

The Flight Controller (FC)—also called Flight Controller Board(FCB) or Autopilot—is the term referring to system hardware for UAVs. In this project, the Flight Controller will come in the form of a commercial assembled product that can support the flight stack, providing the RoboCopter with the necessary features to reach controlled/pre-programmed autonomy [14]. This section will provide a detailed overview of UAV flight controllers, each of their components, and available options that are considered for this project.

3.1.1 Hardware

Flight controllers for multi-rotor UAVs typically, at minimum, contain a microprocessor, sensors, and input/output (I/O) pins. These allow the aircraft to obtain a percept which it uses in its decision making process, as well as communicate with any peripherals that may be desired. It is advised to note that, although specifications and statistics about each of the components that a flight controller contains are important, the decisions that are made focus on the product as a whole. The main considerations taken into account are supported software, optimization for real-time processing, and the presence of components necessary for the purposes of this project, as well as I/O pins for additional needed peripherals.

Main Processor

Similar to a Central Processing Unit (CPU) in computers, the Main Processor, also called Microcontroller Unit (MCU), acts as the brain of the Flight Controller, performing any calculation necessary for stable flight. The main MCU architectures used in Flight Controllers (and IOT devices overall) are the 8051, AVR, PIC, and ARM [17]. The performance of these, although clearly important, highly depends on the application and the software support they provide.

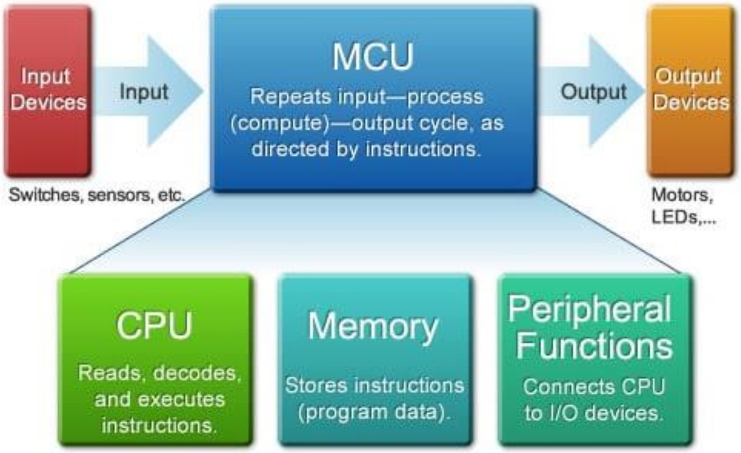


Figure 5: Microcontroller Unit Structure [20]

The 8051 is an 8-bit microcontroller with a Harvard architecture, meaning it has separate memory spaces for RAM and program memory. It has a limited stack space of 32 bytes of RAM, and can directly address all available RAM. Furthermore, the 8051 requires multiple clock cycles per instruction (either 6 or 12, depending on the instruction). It also has four 8-bit I/O pins, one UART port, two 16-bit timers/counters, and support for 6 internal and external interrupts [24]. The PIC, like the 8051, is historically a 8-bit MCU with a Harvard architecture. It has a small number of instructions (between 35 and 80 depending on the model quality), most of which take a single clock cycle to execute. It is inexpensive, simple, and has a wide range of interfaces including I2C, SPI, USB, UART, A/D digital converters, programmable comparators, PWM, LIN, CAN, PSP, and Ethernet [25]. Due to its simplistic design, however, it does have some computing limitations that it addresses with the cost of additional computation time. For example, because it can only directly address 256 bytes of RAM at a time, the PIC must use bank switching to extend it, which can lead to increased processing time.

The Atmel AVR MCU, like the 8051 and the PIC, has a modified Harvard architecture and is an 8-bit Reduced Instruction Set Computer (RISC) single-chip microcontroller. They are known for their presence in the Arduino microcontroller line of open source board designs [26]. It can directly address all available RAM, and can execute most instructions in a single clock-cycle. It has 32 single-byte registers, up to 8 GPIO ports, and internal EEPROM with which it stores relatively small amounts of data, while allowing individual bytes to be erased and reprogrammed. Furthermore, some AVR devices support clock speeds of up to 32 MHz, although 0-20 MHz is the norm. The AVR instruction set is concise, like its 8051 and PIC counterparts; however, it is much more orthogonal, meaning that most instruction types can use all addressing modes, as the instruction type and addressing mode vary independently. Besides a small number of exceptions with certain register and I/O port ranges, there is no requirement for an instruction to use a specific register [27]. Orthogonality is one of the major advantages that this microcontroller unit has over its other 8-bit competitors. Although work for this project does not directly include this concept, it is important to take it into account, as orthogonal MCUs are more likely to be more modern and have wide support, given the growth and relative inexpensiveness of high-performance MCUs. Lastly, the Atmel AVR MCU has a wide range of features including bidirectional GPIOs, I2C, SPI, USB, UART, A/D digital converters, analog comparators, SRAM, PWM, LIN, CAN, PSP, LCD controller support, Ethernet, and Fast AES and DES cryptographic support, among others [28] [29].

The ARM microcontroller RISC family is one of the most widely used on the IOT and 32-bit embedded system market. It has a Von Neumann architecture, much like the figure below, where program and RAM memory are shared. ARM MCUs have either a 16 or 32-bit architecture and can directly address all their available RAM. Like the AVR, most instructions in the ARM MCU are executed in one clock cycle. Due to its multiple applications, ARM microcontrollers, along with the AVR line, have the best compiler and application support [30]. Furthermore, the ARM is supported by a large number of embedded and real time operating systems. They feature up to 2MB of flash memory, operating frequencies of up to 120 MHz, low power consumption, and peripherals supporting I2C, UART, gigabit Ethernet, CAN, TFT LCD controller, camera interface, etc… Due to their versatility and efficiency, ARM microcontrollers and variations of them are used in various Flight Controllers across the hobby UAV market [31].

One major consideration for Microcontrollers is their CPU architecture, which usually represents the space, or width, in each of their primary registers. Registers are small spaces where both instructions and data can be quickly stored and retrieved, due to their close proximity to the CPU's processing components. This can define the CPU power and speed at which calculations and instructions are performed. CPUs tend to have 8-bit, 16-bit, 32-bit, or 64-bit architectures [18]. This architecture can define the type of operating system and other software that the MCU can support. The CPU also contains an Arithmetic Logic Unit (ALU), which performs pivotal arithmetic and logical operations that allow the CPU to receive the necessary operands and instructions and carry out any needed calculation. These instructions, which are defined by the CPU manufacturer, are usually performed automatically by the CPU, and are therefore not in the immediate context of this project. Hence, they will not be further discussed in this section [21].

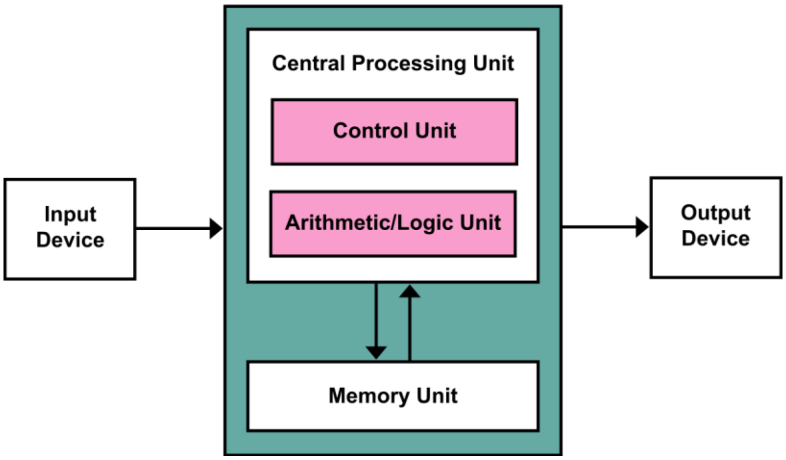


Figure 6: CPU in Von Neumann Architecture [23]

Another important characteristic to consider when selecting an MCU is its operating frequency, or clock rate. This refers to the “number of pulses generated by an oscillator, which is usually a quartz-crystal circuit, that sets the tempo for the processor” [22]. It is usually measured in megahertz (MHz), or millions of pulses per second. The performance of an MCU does not strictly positively correlate with its clock rate; instead, all its components must be able and optimized to work at the pace set by the clock rate. Therefore, when choosing an microprocessor, it is most important not to fixate on the clock rate, but rather consider the quality of the whole package, as that is what will decide the speed and efficiency at which instructions are performed, as well as the unit’s power consumption.

Memory Modules

Memory is a very important component of all microcontrollers that can decide whether a specific MCU is fit for a specific application. In this project’s case, the chosen MCU memory must be able to support real time computations in order for the aircraft to be aware of its current position and environment at all times. This not only includes supporting the chosen operating system, but also being able to store any sensory data and send commands to any output devices, which in this case could come in the form of Electronic Speed Controllers (ESC), video transmitter, sensors, or even another microcontroller. Flash memory, used to store the main code, and may also be “used to store in-flight data such as GPS coordinates, flight plans, automated camera movements, etc…”[17]. Static Random-Access Memory (SRAM) is power dependent memory with very high access times. It acts as the MCU’s cache and is mostly used to perform real-time calculations; it is also used to store sensor data which can continuously change. Electrically Erasable Programmable Read-Only Memory (EEPROM) is a type of non-volatile memory which allows bytes to be read, erased, and re-written individually, used to store information that does not change in-flight, such as data, waypoints, etc… Although the amount of flash, SRAM, and EEPROM memory available to are extremely important to the performance of our flight controller, these memory types and sizes are carefully added to integrated circuits and optimized for their corresponding commercially available flight controller. Furthermore, besides the memory load that all MCUs within flight controllers must support, we do not expect any need for additional computation, as most video and wayfinding processing was done on external microcontrollers which act as peripherals to the flight controller.

I/O Ports & A/D Converters

I/O ports are one of the most important features in the RoboCopter’s flight controller selection process, as a relatively big number of connections is expected. At the minimum, our flight controller must have enough ports to receive consistent power, send both data and power to four electronic speed controllers or motors, connect to a signal receiver, and hook up to the microcontroller unit in charge of the wayfinding and localization computation; it should also contain at least one port for GPS, one for compass if no internal one is present, and a beeper for emergency situations. Additional features that could be used are sensor ports, video transmitter or telemetry port, and USB port, among others. Furthermore, ports that can receive analog signal and turn in into digital signal using Analog-to-Digital Converters are important, as they widen the range of sensors and peripherals that can be connected to and from the flight controller. It should be implied that these ports must be supported by the software within the processor, and that the CPU, the clock cycle, the memory modules, and the power and communication ports must be optimized as a whole for maximum performance, which is the reason for acquiring a commercial flight controller rather than building one from scratch.

Sensors

Without sensors, a flight controller would not be able to obtain a precept of its environment, rendering it completely useless. Commercial flight controllers tend to include core sensors like the IMU, gyroscope, and barometer; GPS and compass, although usually not included, are necessary for flight. The flight controller should have space available for these and any other required sensors, be it internally or externally. For further explanation of the these and other sensors whose data the flight controller might receive, see the Navigation Sensors section.

After analyzing the various components that must work coherently in order to develop a flight controller, it is clear that, when choosing the appropriate flight controller, the package as a whole must be considered rather than each individual component included. Specifically, it is necessary to understand the inputs and outputs that are expected out of the controller, and choose an available product that meets those requirements. Because the RoboCopter was meant to have an additional component performing the wayfinding calculations, the flight controller needed to be capable of receiving and processing flight instructions that allow it to guide the copter in the desired/calculated direction. One of the major topics that come into play for this to work is supported software.

3.1.2 Power

Two voltage ranges are typically given for flight controllers: the voltage input range for the flight controller, and the input range for the main processor’s logic. For commercially obtained flight controllers, we only need to focus on the voltage input range for the flight controller, which is typically 5V. The flight controller, in general, should not be powered separately from the main battery. Each flight controller has its corresponding voltage input ranges and pinouts for them, so it is imperative to check the specs sheet of the corresponding controller. Power, however, should not be a limiting factor to the choice of a flight controller, as the flight controllers considered are optimized for performance and will work as intended as long as the proper voltage range is being passed.

3.1.3 Software

Given the popularity of commercially sold flight controllers among drone hobbyists, there is wide software support specifically geared towards both manual and autonomous flight. Given our RoboCopter's requirements, the chosen software must be able to support the core actions expected from an autonomous UAV, and must be malleable enough for modifications to be made. This leads to the conclusion that Open-Source software for flight controllers is the best approach, as it allows us to add or modify any necessary script supported by the hardware in order to achieve our goals. This section considers the various software stack options available for flight controllers; this software is meant to be flashed into the flight controllers, and can mostly be considered the firmware of the controller, running every time the controller boots up.

Cleanflight

Cleanflight is the most popular open-source flight controller software for multi-rotor and fixed-winged hobby UAVs in the World. It originated by forking off what is one of the oldest and most stables flight controller software, Baseflight, which had previously forked off a small yet impacting RC general purpose open-source project called MultiWii, which used the sensors in the Wii remotes and an Arduino board to create a flight controller. Cleanflight is a mature, yet simplified version of these various projects, and aims at providing quick setup times for as many types of aircraft as possible.

One of the major characteristics of Cleanflight is its support for both stable hardware and for bleeding-edge functionality (through its Betaflight version). It is geared towards 32-bit equipped flight controllers, runs on 8 different boards, and has a variety of features [32]. These include:

- Support for modern STM32 based processors F1/F3/F4/F7.
- Support for modern accelerometer/gyro/barometer/compass sensors.
- Support for modern ESC technologies DSHOT/ONESHOT and legacy PWM.
- Support for Multi-color RGB LED strip support.
- Advanced on-board telemetry logging (Blackbox).
- Wide support of receivers (SBus/iBus/SumD/SumH/PPM/PWM/CRSF/JetiExBus)
- Wide support of telemetry protocols (FrSky/SmartPort/S.Port/HoTT/iBus/LTM/MavLink/CRSF/SRXL).
- Built-in OSD support & configuration without needing third-party OSD software/firmware/comm devices.
- Support for external OSD slave systems.
- VTX support (RTC6705/Unify Pro(SmartAudio)/IRC Tramp/etc).

ArduCopter

ArduPilot is without question the most complete, documented, and reliable open-source autopilot software stack. While Cleanflight specializes in stable flight for manual mode, ArduPilot focuses on providing the versatility needed in order to support from anything between manual and fully autonomous flight for UAVs. It is used by both amateru hobbyists and professionals and has a very strong developer and open-source backbone, therefore making it very appealing for this project [33]. It provides a wide range of hardware support, as it was originally developed for 8-bit microcontrollers but has now evolved to become optimized for use with 32-bit microcontrollers. Furthermore, it supports Linux, giving it an even wider range of opportunity.

ArduCopter, the multi-rotor aircraft specific version of ArduPilot, is very appealing for this project as it is widely supported by the original developers, as well as by developers of other products who have made integration with components like the Raspberry PI much simpler. For example, DroneKit, which we will discuss later, allows for the communication between apps running on an onboard companion computer, like the Raspberry PI or a PCB's microcontroller. This is exactly the type of support that the RoboCopter might need. Below are some of the outstanding features of the core ArduCopter firmware [34]:

- High precision acrobatic mode.
- Auto-level and Altitude Hold modes: Fly level and straight with ease or add simple mode which removes the need for the pilot to keep track of the vehicle's heading. Just push the stick the way you want the vehicle to go, and the autopilot figures out what that means for whatever orientation the copter is in.
- Loiter and PosHold modes: the vehicle will hold its position using its GPS, accelerometers and barometer.
- Return to launch: Flip a switch to have Copter fly back to the launch location and land automatically.
- Ad-hoc commands in Flight : With a two-way telemetry radio installed, just click on the map and the vehicle will fly to the desired location.
- Autonomous missions: Use the ground station to define complex missions with up to hundreds of GPS waypoints. Then switch the vehicle to "AUTO" and watch it take-off, execute the mission, then return home, land and disarm all without any human intervention.
- Failsafes: The software monitors the state of the system and triggers an autonomous return-to-home in case of loss of contact with the pilot, low battery or the vehicle strays outside a defined geofence.
- Flexible and customizable: Copter can fly all shapes and sizes of vehicles because the user has access to hundreds of parameters that control its behaviour.
- No vendor lock-in: ArduPilot is fully open source with a diverse community of developers behind it.

	CleanFlight	ArduCopter
Open-Source	✓	✓
OneShot Support	✓	✓
Optimized for Autonomous Flight	✗	✓
Auto Takeoff & Landing	✗	✓
Wide Online Support	✓	✓

Table 3: Flight Software Decision Criteria

Although Clearflight and ArduPilot are both very complete flight software stacks, ArduPilot was chosen, as it was clearly better for our autonomous tracking objectives. However, just like with software, it is most important to select the flight controller that provides the most versatility, viewing the package as a whole rather than just specific hardware or software details. Below is a detailed review of the flight controllers that were considered, followed by the final design decision.

3.1.4 Flight Controller Selection

Although there is a very wide variety of flight controllers on the market, there is a limited number of them with enough developer, software, and hardware support to allow for autonomous flight. Although creating our own flight controller is an option, due to the time constraints of this project, time may be better spent on the PCB design and the flight optimization process. Furthermore, modern flight controllers reduce the amount of time that would previously have been spent implementing feedback systems like PID control loops and noise-reduction algorithms like a Kalman filter just to get stable flight; this allows the Electrical and Computer Engineering students to focus on working with the rest of the team in order to improve the tracking, obstacle detection, power system, pathfinding algorithms, and propulsion systems. This, in turn, allowed us to meet as many of our client’s requirements as possible, focus on working with groups of different disciplines for maximum integration, and learn to work on top of systems built by others in order to create the desired product. This section contains the flight controllers that were considered for this project.

APM 2.8

The APM began as a fixed-wing flight controller based on the Arduino, but grew quickly and incorporated different types of copters and rovers over time. The APM 2.8 is the last supported version of the APM line, produced through the widely known open-source flight controller project by 3D Robotics. It uses the ATMEGA2560 as its main processor, ATMEGA32U-2 to handle USB functionality, both of which are AVR microprocessors. It also includes the InvenSense MPU-6000, a 6-axis motion module which performs both gyroscope and accelerometer functions. The APM 2.8 has the appropriate interfaces for GPS and compass modules, which are required by our project. Overall, this flight controller is stable, well-known and well-documented. Some of its drawbacks for our specific project, however, included lack of support by the latest versions of ArduPilot, as well as the presence of only one telemetry port. The former drawback means that the latest optimizations for both autonomous flight and overall efficiency were not available. Furthermore, the lack of a second telemetry port may have posed problems for us, as we plan to have two telemetry sources – one coming from the transmitter for manual or emergency control, and another one from the on-board computing device that was to process the RoboCopter's percept, make flight course decisions, and send specific commands to the flight controller directly. Below is a compiled list of the APM 2.8's features:

- Features
    - Limited ArduPilot support (Support dropped after version 2.0)
    - 8-bit AVR
    - Bus Interface: UART, I2C, SPI, CAN
- Processor
    - 8 bit 265K flash memory ATMEGA2560
    - Main frequency: 16MHz
    - Throughput: 16 MIPS
    - Voltage: 4.5-5.5 V
- Sensors
    - MPU-6000 6 axis gyroscope / accelerometer
    - MS5611 high precision barometer
- Interface
    - 1 Telemetry Port
    - 2 UART Ports, 1 I2C
    - SPI
    - CAN
    - External USB MICRO interface
    - Optional Power Module
    - External magnetometer

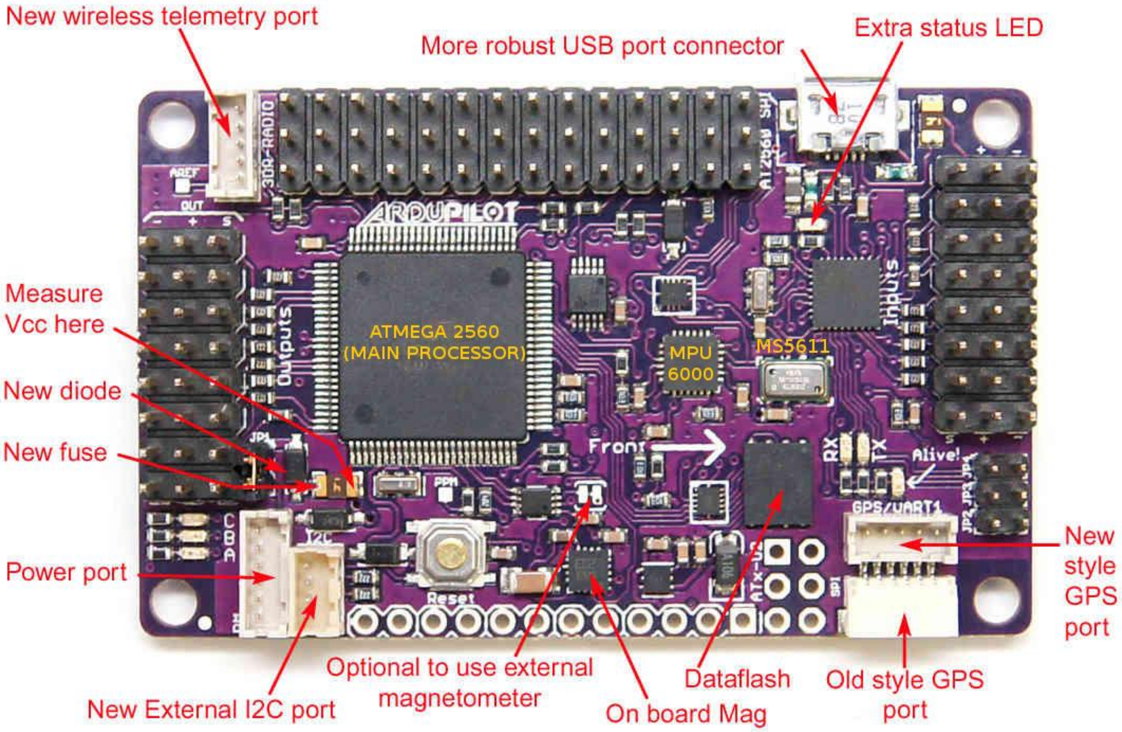


Figure 7: APM 2.8 Board Components [35]

Pixhawk PX4 2.4.8

Given the fast growth of the APM, 3D Robotics decided to develop a new platform that would support the necessary memory and computational processing, as the 8-bit ATMEGA2560 was struggling to keep up. 3D Robotics decided to build on the Swiss Federal Institute of Technology's PX4 hardware project, and hence released the Pixhawk. This flight controller uses the STM32F427 processor, which is a 32 bit ARM Cortex M4 core with onboard Floating-Point Unit (FPU) for improved accuracy and reduced computational time. The Pixhawk, by default, runs with "a very efficient real-time operating system (RTOS), which provides a POSIX-style environment" [36]. It includes multiple gyros, accelerometers, magnetometer, GPS, a MicroSD card for data logs, 5 UARTs, CAN, I2C, SPI, and ADC, among others. Below is a full compiled list of the Pixhawk's features:

- Features
    - Full ArduPilot support
    - 32-bit ARM
    - Bus Interface: UART, I2C, SPI, CAN
    - Integrated power supply & failure backup controller
    - SD slot to record flight data (8GB)
- Processor
    - 32 bit 2M flash memory STM32F427 Cortex M4, with hardware floating point processing unit (FPU)

  - Main frequency: 256K, 168MHZ RAM
  - 32 bit STM32F103 backup coprocessor
- On-board Sensors
  - L3GD20 3 axis digital 16 bit gyroscope
  - LSM303D 3 axis 14 bit accelerometer /magnetometer
  - MPU6000 6 axis accelerometer / magnetometer
  - MS5611 high precision barometer
- Interface
  - 2 Telemetry Ports
  - 5* UART, 1*compatible high voltage, 2*hardware flow control
  - 2*CAN
  - Spektrum DSM/DSM2/DSM-X satellite receiver compatible input
  - Futaba SBUS compatible input and output
  - PPM signal input
  - RSSI (PWM or voltage) input
  - I2C
  - SPI
  - 3.3 and 6.6V ADC input
  - External USB MICRO interface
- Dimensions
  - Weight 38g (1.3 oz)
  - Width 50mm (2.0”)
  - Height 15.5 mm (.6”)
  - Length 81.5 mm (3.2”)

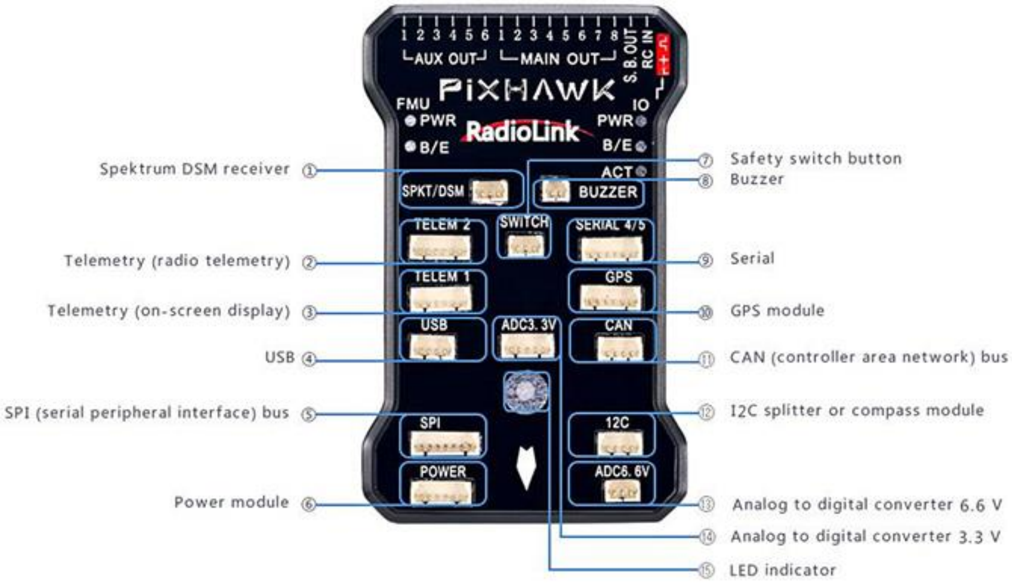


Figure 8: Pixhawk 2.4.8 Board Components [36]

After careful consideration, we chose to use the Pixhawk 2.4.8 flight controller for our project. This is due to the wide hardware and software support that is available, the abundant libraries, tutorials, and forums that can help during the testing and debugging process, the current support that it provides for the latest versions of ArduPilot, and the relative inexpensiveness in the current market. Further research, implementation, and testing is documented in the System Design section. The chart below compares the most important features considered for both the APM and the Pixhawk, for the purposes of this specific project.

	APM 2.8	**Pixhawk 2.4.8**
Open-Source Hardware	✓	✓
ArduCopter Compatible	✓	✓
Continuous Support	✗	✓
Official Versions Available	✗	✓
Wide Online Support	✓	✓
GPS + Compass	External	External
DroneKit Compatible	✓	✓
CPU Architecture	8-bit AVR	32-bit ARM
Processor	ATMEGA2560	STM32F427 Cortex M4
Telemetry Ports	1	2

Table 4: Flight Controller Decision Criteria

3.2 Proximity Sensors

Drones that come already equipped with obstacle avoidance technology are much more in demand by personal consumers and businesses. When flying drones indoors or outdoors obstacle avoidance technology is a must. The various obstacles that can be within the drones path can affect the drone’s flying capabilities and ease of use for the pilot. The use of obstacle avoidance sensors or cameras allow drones to be more stable in flight and allow the pilot controlling the drone to be able to safely fly the drone in conditions or areas that are out of direct sight. Essentially, the use of these sensors keep drones at a safe distance from obstacles that cause significant and costly impacts to the drone. Obstacle avoidance technologies have significantly reduced the fear of consumers crashing these expensive devices. Over the next few years we expect obstacle avoidance technologies to be improved and incorporated into many different engineering industries. With the continuous advancements within the automotive industry in regards to autonomous vehicles, obstacle avoidance features are at the forefront of their research.

The use of multiple sensors on a drone is very common in the industry. Sensor “fusion”, or the combination of multiple sensors, allow drones to have the ability to accurately detect and/or identify objects that are in the environment in which the drone is flying [1]. It is very common for some of the top selling drone manufacturers to combine sensors to increase the capabilities of their drones.

These sensors combine to provide important data to the flight controller so that the drone can make decisions for itself that ensure the safety of the drone itself, humans, and other objects within the surrounding area. Various combinations of sensors can be used to measure distances of obstacles to the drone. Information can then be sent to the flight controller to adjust the flying conditions of the drone. All of this computation can be done automatically, thus taking substantial load off the pilot.

Some of the most popular types of sensors used in the market for obstacle avoidance include: Infrared (IR), LIDAR, Stereo Vision, Time-of-Flight, (ToF) and Ultrasonic (US). Below is information about each of these sensor types and a comparison of these sensors based on the goals of this project.

Time-of-Flight

Time-of-Flight sensors in drones or ground vehicles have been used for many different applications. Some of the applications where ToF sensors have been used includes object scanning, indoor navigation, obstacle avoidance, gesture recognition, tracking objects, measuring volumes, reactive altimeters, 3D photography, augmented reality games and much more [4].

Time-of-Flight (ToF) distance measurement sensors emit infrared light pulses from the transmitter that light up the object, and a portion of that signal is then reflected back to the sensor’s receiver [4]. The phase shift between the illumination of the scene and the reflection is measured in seconds and translated to distance. The figure below is an example of the way a time-of-flight system works.

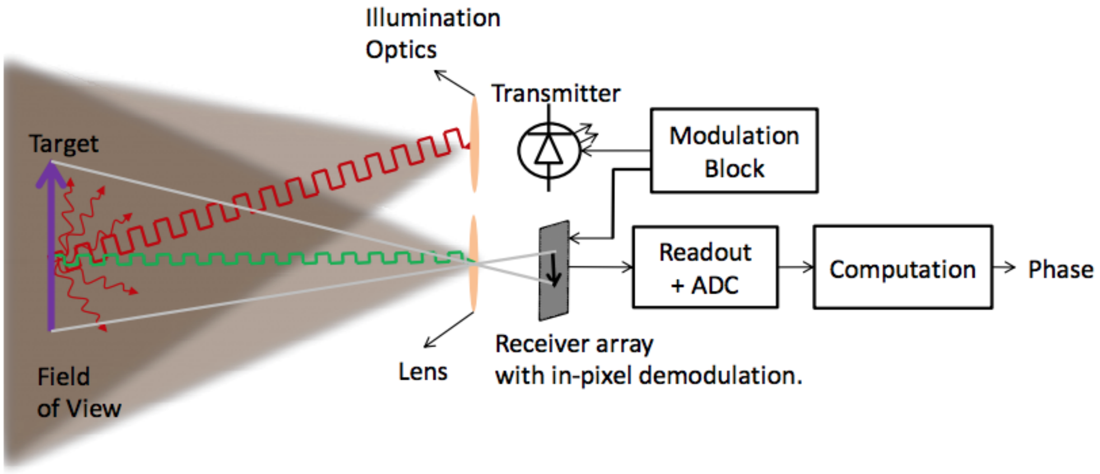


Figure 9: Time-of-Flight Sensor System

ToF sensors have many advantages and disadvantages. Some of the ones that apply for this project include:

Advantages:

- Measures distances within a scene in a single shot
- Cheap compared to other 3D range software
- Less processing power than stereo vision

Disadvantages:

- Light sensors are less accurate in poor lighting conditions such as fog
- Materials that are semitransparent may cause the transmitted rays to be reflected away from the sensor and results in inaccurate data

Infrared

Infrared radiation is the portion of the electromagnetic spectrum that has wavelengths of 0.75 µm to 1000 µm. The wavelengths are longer than the wavelengths of visible light which is the only portion of the spectrum that can be seen by the human eye.

Infrared sensors, as the name suggests, uses infrared rays to detect obstacles. An infrared obstacle detection sensor works in accordance with the infrared reflection principle to detect obstacles [1]. These types of sensors measure the distance between the object and the drone by calculating the time it takes for the infrared light to travel from the emitter to the object and from the object back to the transmitter. The figure below illustrates the concept of IR sensors. These are used quite frequently in applications where the object to be detected gives off a significant amount of heat that is different from the surrounding area.

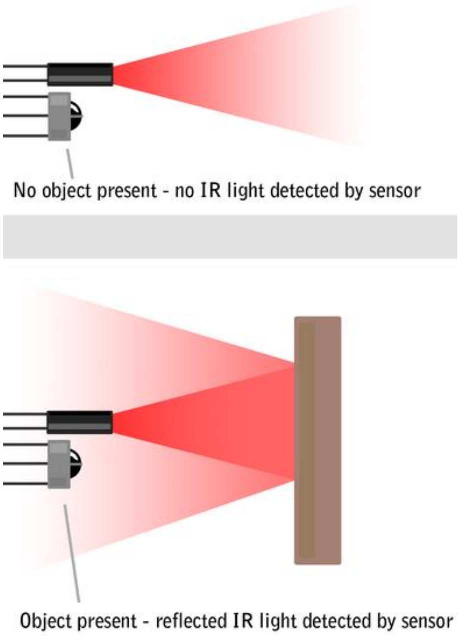


Figure 10: Infrared Sensor System

The use of infrared light to detect objects is very common. Infrared sensors have their advantages and disadvantages. Real time object detection is capable with the use of IR sensors and the speed of these sensors is well suited for the purposes of this project. With the sensor's method of determining distances, consistent readings which are less influenced by surface reflectivity, operating time, or environmental temperature can be taken [1]. On the other hand, IR sensors have disadvantages which may cause some issues in regards to the goals of this project. IR sensors become troublesome when it comes to poor lighting conditions and cannot detect diaphanous obstacles [2]. Infrared sensors also depend on weather conditions. The reliability of infrared sensors decreases with moisture and humidity. The range of some of the smaller IR sensors also can be a problem. To be able to detect obstacles and avoid them would require a sensor that gives the drone enough time to slow down and ultimately change its' flying direction. With the limited range of some of these lower powered IR sensors, the drone may end up crashing into obstacles even after detecting them.

LIDAR

LIDAR, or Light Detecting and Ranging, is a sensing technology which emits a pulsed laser light (usually ultraviolet, visible, or near infrared light) that is used to calculate the distance to and from an object. Lidar is one of the leading technologies for automobile collision avoidance and also in driverless cars [1]. The technology has been recently used on drones for obstacle avoidance.

The way that LIDAR sensors work is very similar to the Time-of-Flight sensors previously discussed. Sometimes ToF sensors are referred to as "Flash LIDAR" but the difference is ToF sensors are scannerless sensors and captures the entire scene in a single shot while LIDAR sensors scan the entire region point by point until the full image is captured [3]. LIDAR illuminates the object with a pulse of laser light. The distance is measured by tracking the time the pulse of light was transmitted and then received by the sensor. The system does an aggregate scan of multiple points which it then uses to compute a 3D view of its surroundings. As the sensor moves, the height, location and orientation of the instrument must be known to determine the position of the laser pulse at the time when the signal is sent and the time the signal is received [8]. The speed at which images can be developed is affected by the speed at which it can be scanned into the system [8].

Some of the advantages and disadvantages of LIDAR sensors include:

Advantages:

- Useful for making high resolution maps
- Lighting conditions do not affect the performance very much
    - Works well, even at night
- Higher range than most other sensor systems

Disadvantages:

- Accuracy depends on the condition of the object
    - Doesn’t work as well if the object is wet
- Cost of this sensor seems to be a lot higher than other object detection technologies for drones

Stereo Vision

Stereo vision works in a similar fashion to 3D sensing in human vision. Two cameras compile their respective 2D images to create one solid image that is then used to identify and avoid obstacles. One sensor identifies pixels in one image and the software tries to match the same location in the image taken by the other sensor [8]. This type of technology uses depth measurements to be able to understand its surroundings and determine how far the object is from the copter. The figure below explains the way stereo vision systems are set up.

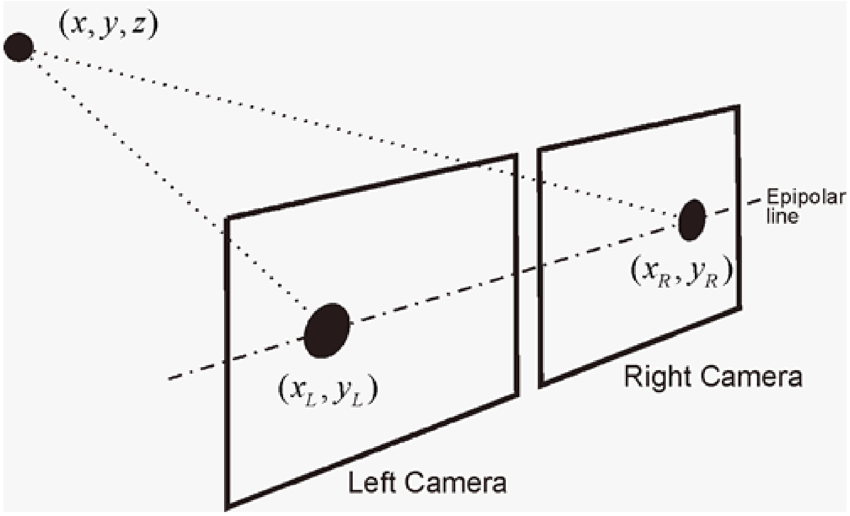


Figure 11: Stereo Vision System

Some of the advantages and disadvantages of stereo vision sensors include:

Advantages:

- The accuracy of the sensor system makes it useful for the task at hand
- Real time transformation of 3D data
- Easily integrates with other sensor techniques
- No need for “in-field” calibration because the sensors are usually factory calibrated to account for the lens distortion and camera misalignments

Disadvantages:

- Two sensors for each direction
- The total number of sensors would be doubled
- Needs to be combined with another sensor system for this project

Ultrasonic

Ultrasonic sensors have become very useful within the industry. They have been used in a wide number of applications and have proved very capable of meeting requirements for object detection. They are consistent and reliable sensors that get around the issues that optical camera sensors have come across and looks to be very capable of meeting our project goals.

Ultrasonic sensors send out a high-frequency sound pulse that gets reflected back off the object and returns to the sensor [1]. The sensor consists of two pieces, the transmitter and the receiver. The ultrasonic sensor uses the speed of sound (341 meters per second or 1100 feet per second) and the time that it takes to transmit the sound pulse to the object and then receive the sound pulse to determine how far the object is from the drone. The figure below shows an example of an ultrasonic sensor system detecting distance to an object.

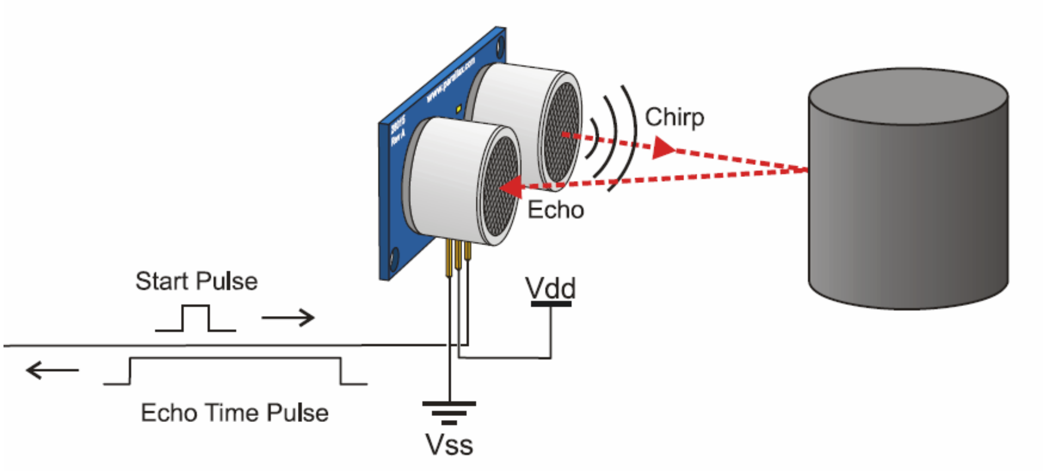


Figure 12: Ultrasonic Sensor system

$$D = (T * S)/2$$

D = Distance to the object
T = Time
S = Speed of sound

Some of the advantages and disadvantages of ultrasonic sensors include:

Advantages:

- Uses non-contact range detection to detect objects and measure distances
- Sensors on the bottom can be used to see how close the drone is to the ground
- Cost effective and reliable
- Accurate detection of even small objects
- More effective than other sensors on materials such as wood, plastic, and rubber

Disadvantages:

- Weather conditions can affect the performance
- Certain materials like paper and sponge can affect the performance
- Ultrasonic sensors used for these types of applications have a lower field of view than some other technologies

Based on the research that is outlined above, the group has decided that the best choice for a sensor that would meet the needs of this project is an ultrasonic sensor because of characteristics such as price, range, accuracy, and power requirements.

Each of the sensors chosen have high accuracy compared to other sensors of their type. The ToF sensors, as discussed previously, will work better in well-lit conditions because they are optical sensors. The ultrasonic sensors was less effective when the signal is reflected off materials that absorb sound waves and more effective than ToF sensors in poor lighting conditions. The following table will compare four sensors, two time-of-flight and two ultrasonic, that are frequently used in drone applications.

	Non-Optical		Optical	
	HC-SR04	Devantech SRF 10 Ultrasonic	TeraRanger One Type B	Adafruit VL53L0X
Min distance (mm)	20 mm	60 mm	200 mm	30 mm
Max distance (m)	5m	6m	14m	2 m
DC Voltage (V)	5V	5V	12 V	2.6 to 3.5 V
Weight (grams)	8.5g	3g	8g	1.3 g
Price per Sensor ($)	$1.96	$33.68	$133.00	$14.95

Table 5: Proximity Sensor Comparisons

Based on the comparisons outlined in the table above, the choice of sensor that we felt were best suited for the purposes of this project was the HC-SR04 ultrasonic sensor. The range, ease of use, and price of the HC-SR04 sensor was the biggest factors in selecting that device. Many applications that involve proximity sensors, whether for aerial vehicles or robots maneuvering on the ground, gravitate towards these two sensors.

The following picture displays the accuracy achieved by the HC-SR04 sensor depending on where the object is in the field of view. As the picture shows, the sensor works best when objects are within ±22.5°.

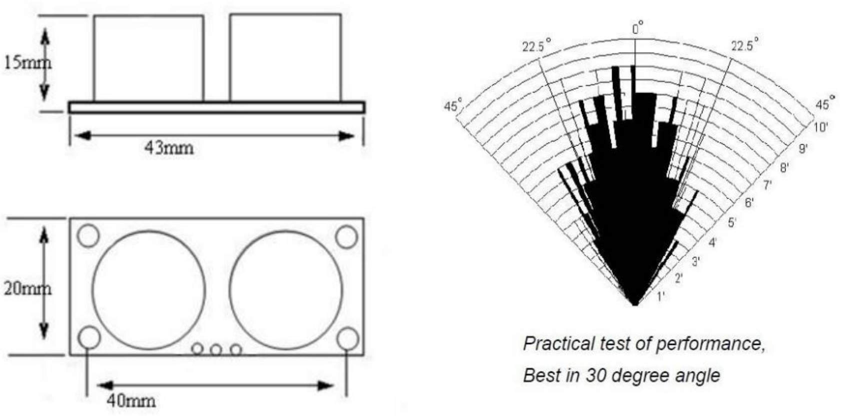


Figure 13: Performance of HC-SR04 Ultrasonic Sensor [14]

3.3 Navigation Sensors

Whether embedded within the Flight Controller or not, navigation sensors are necessary for a successful avionics system. Sensors read and provide the controller with the necessary information about its current environment state, which it can use to make critical flight decisions. This section will cover the various types of sensors that allow for efficient autonomous navigation. It will also cover different alternatives considered and provide an explanation for each design decision that was made regarding them. Note that, due to the common design conventions for small to mid-sized commercial and recreational UAVs, this section is strongly connected to the aircraft Flight Controller, which usually contains most of the sensors needed for basic stable flight.

Accelerometer

Accelerometers are sensors that measure proper acceleration. In other words, they measure the linear acceleration of a body in its own instantaneous rest frame in the x, y, and z axes. The output measurement units usually come in gravity, also known as 9.81 m/s/s or 32 ft/s/s; however, actual data output may also come in voltage units or, when sensed like weight, in G-forces. A major feature of accelerometers that allow them to greatly contribute to the stability of the vehicle they're on is their ability to detect gravity, which allows them to detect the downwards direction. Furthermore, they are able to detect coordinate acceleration, vibration, shock, falling in a reactive medium, orientation by measuring the direction of weight changes. Although position can be obtained from an accelerometer output, it is subject to drift.

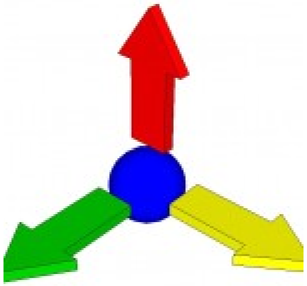


Figure 14: Accelerometer Axes [17]

An accelerometer uses the principle of Newton's second law of motion, force is equal to mass times acceleration. This equation is then adjusted to show that acceleration is directly proportional to force and inversely proportional to mass.

$$Force = Mass * Acceleration \rightarrow Acceleration = Force/Mass$$

The idea behind the way an accelerometer works is that it 'senses' a force from one of the axes with its electromechanical system and from that measurement true acceleration can be found. In some of the configurations the thin sensor has a wafer of movable mass is located above or in between fixed electrodes that detect the movement of the mass.

Accelerometers are economic and a good way to measure acceleration for flight stability. A drone needs to have sensors that detect its speeds and an accelerometer can satisfy that need. Some concerns that have arisen with this technology are that while it is accurate, use over time can wear the sensor down and cause less accurate readings. Another concern is that vibrations may also contribute to skewed instrument readings. These problems are important to be aware of in making informed decisions on selecting an accelerometer.

Gyroscope

Gyroscopes are used to measure or maintain orientation and angular velocity by monitoring the rate of angular change in up to three axes (α, β, and γ). The output of a gyroscope is usually analog or I2C, which is processed by the flight controller to obtain the actual angular change in degrees per second. Gyroscopes are critical for stable flight and movement, as they allow the flight controller to keep track of any change in angular velocity that the copter might experiencing, therefore potentially modifying how the copter's position and stability will change [17].

Figure 15: Gyroscope Axes [17]

The gyroscope is used to tell how fast the aircraft is rotating around its own axes which are yaw, pitch, and roll. Angular acceleration in gyroscopes are usually measured in degrees per second or revolutions per second for angular velocity. The PID controller uses this information directly for flight stability. Failure of the instrument typically results in a crash due to the lack of stability.

Inertial Measuring Unit

IMUs are boards that combine the functionality of accelerometers and gyroscopes; they are a common feature in off-the-shelf flight controllers. Normally, they contain a 3-axis accelerometer and a 3-axis gyrometer, but might contain a magnetometer as well. Overall output is the body's specific force, angular rate, and sometimes magnetic field.

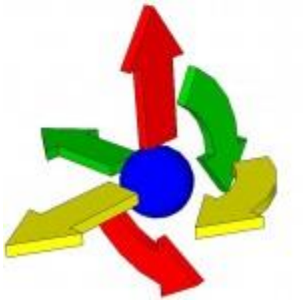


Figure 16: 6-Axis IMU [17]

Pressure/Barometer

Barometers are sensors used to measure air pressure to determine altitude. Pressure drops as altitude increases, and this can be used by the barometer as long as the absolute altitude local sea level pressure is known. A limitation with a barometer is that alterations may occur in readout when there is rapid air movement from wind; this effect, however, can be reduced by covering the sensor with foam. Barometers are needed for altitude hold and automatic modes such as return to home features. Most flight controllers take input from both the pressure sensor and GPS altitude to calculate a more accurate height above sea level.

GPS

A Global positioning system (GPS) is a common choice when it comes to navigation systems. However, it is important to note how it is used and what this information was used for and consider other potential systems. A GPS system is a sensor used to detect geographic positioning by means of satellites. It serves to find drone location so that it can be recovered and/or given instructions to go to specific coordinates. GPS systems need signal space and are truly beneficial when it is used in an outdoor space. In the use of autonomous flight, the flight controller can use the GPS to determine where to go by comparing its current location to the desired location. A component that is not always included in the GPS system, but is important to its function, is a magnetometer or a compass. For the GPS to fully function as it should, it needs to know where magnetic north is. The only issue with a magnetometer however, is that it cannot be placed near metal parts on the aircraft.

Although GPS chips can be found on some flight controllers, this is not optimal due to the common presence of a companion compass; the metals inside flight controllers tend to make the compass (and therefore GPS) readings slightly inaccurate. Therefore, most modern flight controllers include a port in which a cable connected to a GPS antenna, which holds the GPS chip, can be found. When a GPS detects incoming data from one or more satellites, it initiates a GPS lock sequence, which assures that there is continuous data being received from one or more satellites. As can be inferred, the more satellites that are sending continuous data to the GPS, the better [17]. For the purposes of this project, GPS data will only be used for the flight controller’s stabilization processes, but will not be used for wayfinding; therefore, the selection of a specific external GPS is not critical to the performance of our RoboCopter.

Compass/Magnetometer

Electronic magnetic compasses, also known as magnetometers, read the earth's magnetic field and use it to determine the direction that the UAV is facing, relative to magnetic north. Magnetometers are critical for the proper functionality of a GPS, and usually come embedded in external GPS antennas. As mentioned above, this project will not need the use of high-accuracy GPS modules, and therefore this section will not delve deeper towards the available options for GPS. Instead, a generic GPS module compatible with the selected flight controller was chosen and documented on the design section of this document.

Although navigation sensors are definitely a critical factor in the performance and stability of multirotor vehicles, we focus on selecting a flight controller that is complete and of high quality as a whole. Given flight controller should have most, if not all, of the sensors required for stable flight; these sensors, in turn, should be optimized for the specific design of the controller. Therefore, no selection process will take place; for details about the chosen flight controller and other design details, see the Flight Controller Selection and the System Design sections.

3.4 Object Detection & Tracking

In order for the drone to be able to take down prey drones, it must first have the ability to detect and track the other drones; which then introduces the topic of Object Detection & Tracking. Object Detection is a technology that is often associated with computer vision, more specifically image processing. It deals with detecting and finding different objects in a digital image or a video. Although the field itself is relatively new, there are quite a few interesting and exciting applications that are making use of it. For instances, face detection, self driving cars, people counting and aerial image analysis. To take it bit further, over recent years it's been commonly used in aerial photography and surveillance by using a drone. However, the use of the drone presents more challenges due to the top-down view angles, real-time, weight and area constraints [55]. Adding to that, the images are sometimes noisy and blurry because of the drone's motion. In this sections, basic aspects of object detection was discussed along with the architecture and the different algorithms that are commonly used for that.

3.4.1 Object Detection

Camera

Our drone needs to be able to avoid obstacles and take down enemy drones, and in order for it to do so, there are a couple hardware components that it's going to need; one of these components is a camera. There are a few factors that one must consider when considering cameras for a drone. For instance, the weight of the actual camera is often a big factor because a camera that's too heavy will cause the drone to be somewhat slower, and less agile. Another key aspect that one has to pay close attention when it comes to the camera is the Ground Sample Distance (GSD). The GSD is the real-world size of a pixel in images which generally sets a physical limit on the accuracy of the aerial survey; generally it affects how precise the models and map gets to be [52]. As an example, given the case where the GSD is 5 centimeters, then the model that's produced will not have an accuracy that's greater than 5 cm. In the camera itself, there are a few variables that can directly affect the GSD, and those are:

1. Camera resolution
2. The size of the camera sensor
3. The altitude of the flight

The camera resolution is essentially the amount of detail the camera can capture, and it is measured in picture element (pixel). For most drones, that value typically varies somewhere between 2 - 50 Megapixels. One will typically choose a camera with higher resolution, however, that doesn't often end up being an advantage especially if the sensor size is small. A camera sensor, also known as an image sensor is a solid-state device that happen to play a major part in the camera. It's main role is to capture light, and convert what the camera is pointing to into an image; it is often call the soul of the camera. They come in a variety of sizes such as: full frame, APS-C, Micro 4/3rds, ⅔", and 1". Since they control how much light comes in, larger sensors will have better light gathering ability at the same resolutions while smaller sensors will need greater exposure times to achieve the same effective outcome [52].

Besides the GSD, and the weight, the power consumption of the camera was another aspect to keep a close eye on as well. Since our drone was powered by a lithium polymer battery, it must be ensured that we choose a camera that uses it in the most efficient way possible.

In short, choosing the best camera for our drone was somewhat of a challenge, mainly because of the variety of options that are currently available on the market. Therefore, to make our decision easier, we choose a few criteria to focus on that will allow us to make the best decision. Those criterias are as followed:

1. Camera size & weight
2. The camera Ground Sample Distance
3. Power consumption
4. Cost

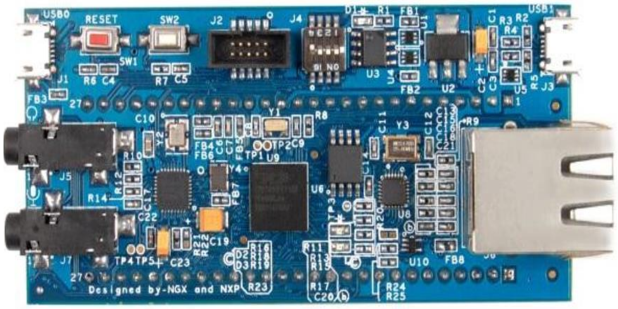
Pixy

The Pixy CMUcam5 is a small, low-cost, easy-to-use, embedded systems in the form of a intelligent camera. Pixy got its start as a Kickstarter campaign, and it's the result of a wonderful partnership between Charmed Labs and the Carnegie Mellon Robotics Institute. Charmed Labs was founded back in 2002, and their main goal is to make advanced technology affordable and easy to use which will potentially help bring these to new audiences. In the other hand, established way back in 1979, the Robotics

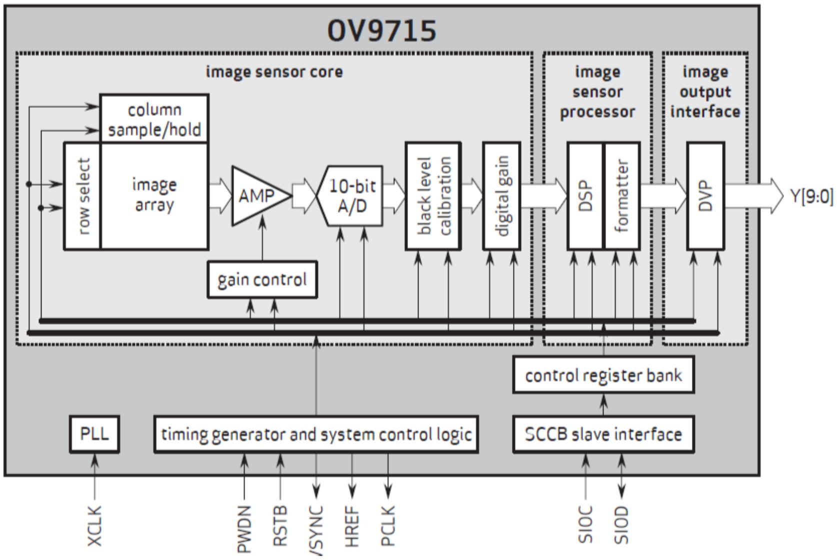
Institute at Carnegie Mellon University has been one of the world's leader in robotics research and education. Together, those two were able to build the Pixy camera which first started shipping in March of 2014. In short, The Pixy is a smart camera sensor that can detect many different objects based on training. Basically, one teaches Pixy one or more objects, then Pixy will start tracking that object based on it's color. Teaching is achieved by placing the desired object in front of the lens and holding down a button that's located on top of the board. While this is happening, there is a RDB LED under the lens that's providing feedback regarding what object the sensor is focused on. For instance, the LED will turn yellow when there's a yellow ball that is placed in front of the image sensor. Once the LED is the same color as the object, then Pixy is finished with the learning process; it can learn up to seven different colors. Pixy has the ability to report what it sees through many different interfaces such as: UART serial, I^2C, SPI, or it can provide it digital or analog output. Pixy has two main components a very powerful processor, and the actual image sensor. By pairing up those two, pixy filters all of the images that it sees through the image sensor and it only sends the useful information to the microcontroller that it's connected to. Most importantly, it is able to do at a frame rate of 50 cycles per second. As an illustration of how fast pixy is, it can process an entire 640x400 image in 20 milliseconds. Pixy uses a color-based filtering algorithm to detect objects which are filtering methods are popular because they are fast, efficient, and relatively robust [53]. Pixy is available for purchase online on websites such as amazon or ebay, and it typically cost about $70.

Technical Specifications

In its entirety, Pixy weighs about 27 grams with a dimension of 2.1" x 2.0" x 1.4". The microcontroller on it has about 264 Kilobytes of RAM, and 1 MegaBytes of Flash. As far as power consumption, it typically needs 140 mA with an input voltage that varies from 6 - 10 volts. The microcontroller that is in the pixy is called the NXP LPC4330. It is an ARM Cortex-M4 digital signal controller running at frequencies of up to 204 MHz and it is preferably designed for small embedded applications that requires signal processing. The ARM Cortex-M4 core offers single-cycle Multiply-Accumulate and SIMD instructions and a hardware floating-point unit to support signal processing while the M0 coprocessor handles I/O and digital control processing [54]. The MCU itself has 264 KB of data memory, advanced configurable peripherals, two High Speed USB 2.0, SPI Flash Interface (SPIFI), an external memory controller and multiple digital and analog peripherals. Some other features that the MCU has that are worth mentioning are as followed: Ultra-low power Real-Time Clock (RTC) crystal oscillator, Single 3.3 V power supply with on-chip DC-to-DC converter, Two 10-bit ADCs with DMA support and a data conversion rate of 400 k Samples per seconds, four general-purpose timer/counters with capture and match capabilities, last but not least, it has one fast mode plus 12C bus interface with rates up to 1Mbit per second.



The other main component of the pixy is the image sensor which happens to be an Omnivision OV9715. The 1/4-inch OV9715 sensor provides full-frame, sub-sampled or windowed 8-bit /10-bit images in RAW RGB format via the digital video port and with complete user control over image quality, formatting and output data transfer, and it offers a Chief Ray Angle (CRA) of zero degrees [114]. The sensor comes along quite a few advanced image processing functions such as white balance, gain control, lens correction and defective pixel correction and it is programmable through the serial camera control bus (SCCB) interface. In order for it to store images, the sensor has a one-time programmable memory. Some of the main features of the image sensor are: it support RAW Red Green Blue as output, it has phase lock loop (PLL) on the chip, it is ultra low power and really cheap, support for horizontal and vertical sub-sampling, and lastly it has a digital video port (DVP) parallel output interface. It has a resolution of 1 Megapixel and it’s frame rate is at 30 frames per second.



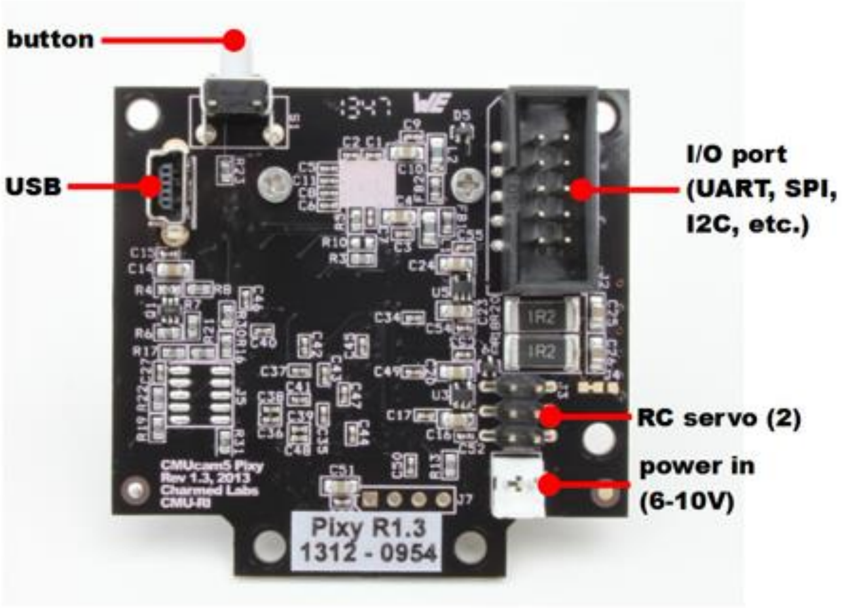


Figure 17: Pixy Circuit Board

The picture above is a layout of the back Pixy board. As illustrated, it has a button that is used for training purpose, a USB port, I/O port which allows the sensor to report what it sees through the many different interfaces, RC servo, and the power input port.

Lumenier CS-600 Super

The majority of the cameras that belong in the Lumenier family are specifically First Person View application. They are high resolution, high quality cameras, and they have a wide dynamic range for superior colors and clarity. The CS-600 Super started to become very popular within the FPV community because it is very affordable and it performs really well.As a few of the components, the camera comes with a dual 850 nm IR filter and that is pretty much in charge of blocking the IR light. The Sony Super HAD II CCD 600TVL FPV board camera with Nextchip 2040 DSP, Samsung and Panasonic resistance components and Tantalum capacitor by Lumenier [114]. The image sensor on the camera is the 1/3" SONY SUPER HAD II CCD which is a 1020 x 508 pixel sensor with a 5 µm x 7.40 µm unit cell size. The Super HAD CCD is a version of Sony's high performance CCD HAD (Hole-Accumulation Diode) sensor with sharply improved sensitivity by the incorporation of a new semiconductor technology developed by Sony Corporation [115]. In general, a CCD image sensor can have millions of pixels, and each pixel have a photo sensor that is used to convert the incoming into electrons. The camera board itself only weighs about 15g, but with an protective case, it weighs 44g and Its dimensions are 36mm x 36mm with the actual case, and 32mm x 32mm without. It is a very low power consumption system as it only needs about 9 to 12 volts to properly operate. To highlight the camera specifications, the CS-600 super has an electronic shutter speed that can vary anywhere from 1/60 - 1/100,000. It outputs its video using 1.0Vp-p 75 ohm BNC format. It has a horizontal resolution of 600TVL, and its display mode is through a LED monitor/CRT monitor. Additionally, it has other things such as white dot repair, edge light compensation, and Synchronize IR turn on and color turn to B&W. Its signaling system is the National Television System Committee (NTSC), which was the first widely adopted broadcast color system. Back in 1953, that system was responsible for developing protocols for Television broadcast transmission and reception in the US. Up until now, there haven’t been any significant changes in their approach except for the addition of a couple parameters for color signals. That’s the system that is used in most of the Americas and some of the Pacific island nations, except for Argentina, Paraguay, Brazil, and Uruguay. To briefly go over how it works, An NTSC TV image has 525 horizontal lines per frame (complete screen image), these lines are scanned from left to right, and from top to bottom while every other line is skipped;therefore, it takes two screen scans to complete a frame: one scan for the odd-numbered horizontal lines, and another scan for the even-numbered lines [116]. As a last couple of specs, all synchronization of the system happens internally, it has a signal to noise ratio of 60 decibels and its IR image optimizer provides an anti exposure of the object in the middle.

Overall, Compared to the Pixy, the Lumenier CS-600 Super is just an image sensor, it doesn’t have an actual microprocessor attached to. Therefore, it is unable to do task like object detection on site. It is really good when it comes to First Person View applications since that’s what it was designed for, however the other camera specifications aren’t all that convincing.

See3CAM_CU130

The See3CAM_CU130 is a UVC camera that has a dedicated, high performance Image Signal Processing (ISP) chip that has the ability to perform all the auto functions such as exposure control, white balance. It also comes provides best-in class images and videos due to the complete image signal processing pipeline. This camera part of the e-con systems family which itself comes with a lot of benefits.

Image stabilization is one of the biggest issues that one must face when working on a vision system for a drone. When a video is being recorded from a drone, the quality might not be as good as it could be due to mechanical vibrations. Therefore, e-con systems developed real-time Video stabilization algorithms that are implemented in their cameras in order to stabilize the camera motion. Video stabilization is the process of removing this kind of unwanted motion in the video feed and creating a smoothed video output this is performed by first detecting the induced motion on the camera and then compensating for it, so that the output video is free of unwanted shakes and jitters [117]. In the CU130, That stabilization is achieved based on Inertial Measurement Unit (IMU). Most video stabilization algorithms consists of three main steps. First is the motion estimation, then it go through some type of set filters, and lastly is the image compensation. The main point behind the first step was able to determine the camera is moving. Immediately after, a filter function is used to sort of smoother out the the motion information. Lastly, the image plane is then moved in the direction opposite of the unwanted motion.

Looking at the specifications of the camera, it has a 1/2.3" Optical form factor AR1820HS sensor with on-board high performance ISP, and it can shoot Ultra and Full HD at 30 frames per seconds, and HD at 60 frames per seconds. It is a 13 megapixels camera which has an electronic rolling shutter. It is a very low power consumption system, as it only requires about 5 volts for input voltage, and it is able in a wide variety of temperatures, from -30 degrees to 70 degrees. It weighs about 14 grams without its custom lens and approximately 19 with the lens, and its official dimensions are 30 x 30 x 32 for the length, base, and the height respectively. The camera supports multiple operating systems: Windows 7, 8, 10, a few linux distributions, and Android. Additionally, it has two different USB 3.0 connector types: the USB 3.0 Micro-B connector and USB 3.0 Type C connector, and it outputs its stream in a compressed MJPEG format which is a video compression format in which each frame is compressed individually as a JPEG image. Its output is enabled by the Aptina 8-lane HiSPI serial interface which is an open access scalable piece of technology that has been adopted by many companies to solve data transfer challenges. It is packaged with external hardware trigger input. Its signal to noise ratio is 36.3 decibels. As far as the image sensor itself, the AR1820HS is a 18 megapixels ½.3” Complementary Metal Oxide Semiconductor (CMOS) that comes with a variety of operation modes. Based on different environmental conditions, the sensor has the ability to optimize the resolution.

The AR1820HS utilizes Aptina™ A-PixHS^{TM} technology, which brings Aptina's BSI pixel technology together with advanced high-speed sensor architecture to enable a new class of high performance cameras, A-PixHS™ technology provides high quantum efficiency, low noise, and low power consumption, enabling uncompromised image quality for both still images and Full HD video [118]. With the introduction of this sensor, Original Equipment manufacturer now has the required resources to create a compelling new generation of cameras.

The See3CAM_CU130 certainly brings quite a few benefits, for instance, it provides a pretty good resolution, and it is implemented with a pretty good video stabilization algorithm. Therefore, one can be guaranteed that their video quality was really good if they choose to use that product.

Intel RealSense Depth D435

Before diving into the Intel RealSense, a brief explanation of a depth camera was given. A depth camera is pretty much a camera that captures depth images. In turn, a depth image is an image that indicates the distance from the camera to the captured image in each picture elements. For the most part, a depth sensor will have three parts: a Red Green Blue camera (optional), Infra-Red (IR) camera and an IR projector. In order for it to work properly as a unit, the IR projector will project an irregular pattern of dots on objects around it once it is powered up. Those patterns ends up being invisible to the human eyes because of their wavelength (ranges from 700 nm to 1mm). Secondly, the IR camera which has the capability to detect the infrared light that got bounced off the subjects is used to capture the intensities of the infrared light. Finally, and most critically, the video feed from the IR camera gets sent into the depth sensor processor. Using the displacement of the dots, the processor is able to calculate depth for every pixel in the scene. Even through the end, a small disparity can be observed between the recorded image and the projected pattern because of the distance of the objects and the separation in cameras. However, using the process above, results an image that indicates the depth.

The Intel® RealSense™ Camera D435 is part of the Intel® RealSense™ 400 Series of cameras, a lineup that takes Intel's latest depth-sensing hardware and software offerings and puts them into easy-to-integrate, packaged products [120]. Powerful, lightweight, and low-cost, this device gives other robotics systems the ability to really understand and interact with their surroundings. In order to calculate depth, the D435 uses stereo vision. Stereo vision in that case is mostly made of a right imager, left imager, and an optional infrared projector. The left and right imager are set with identical parts. In addition to the two imager, the depth module contains a color camera which provides texture information. Those information can include overlay on a depth image in order to create a color point cloud and overlay on a 3D model for reconstruction. It is powered through VBUS power of the USB connector and the D4 card itself power sources the Depth module; it normally needs about 5 volts to operate.As far as output, it does that through USB 3.0 for host system connection. The picture below contains all of the other components that makes up Depth module.

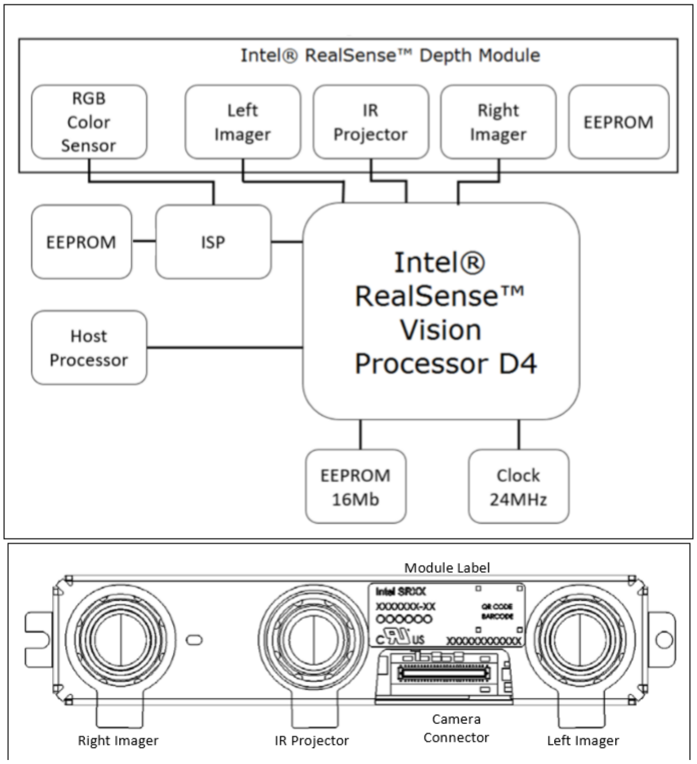


Figure 18: D435 System Block Diagram

The Intel RealSense D435 system is mainly composed of four major components:

1. RGB sensor to collect colored data
2. Infrared projectors to illuminate objects
3. Video Processor D4 to handle all the complex algorithms
4. Stereo image sensor so the disparity between images can be calculated

The Intel RealSense Vision Processor D4 is a purpose-built ASIC for computing real time depth and accelerating computer vision, at significantly faster speeds and fraction of the power compared to host based compute [Ley 2The D4 Vision processor included in the camera is an efficient and compact size card that provides advanced and complex algorithms to process raw depth camera image streams. This processor itself includes features such as active power gating, Time stamping and real-time synchronization across multiple sensors, Advanced stereo algorithms such as Semi Global matching, use of RGB channels, and Image rectification for camera optics and alignment compensation. This powerful vision processor that uses 28 nanometer (nm) process technology and supports up to 5 MIPI Camera Serial Interface 2 channels to compute real-time depth images and accelerate output, generating up to 90 frames per second (fps) in a depth video stream, that is 60 more fps compared to the 30 fps of the first-generation stereo depth camera [119].

Highlighting the specifications of the real sense camera, Its dimensions are 90 mm x 25 mm x 25 for the length, depth , and the height and it weighs 72G. Its depth stream output resolution can go up to 1280 x 720, and its frame rate can go up to 90 frame per seconds. The image sensor type is global which is kind of the same thing as a rotary shutters that have been adapted for digital use. With the global image shutter and wide field of view, the Intel® RealSense™ Depth Camera D435 offers the capability to capture and stream the depth data of moving objects effectively, providing high depth perception accuracy to your prototype in motion [119]. Depending on a few varying parameters, such as lighting conditions, calibration, scene the maximum range can vary anywhere from 10 meters and up. As far as compatibility, it supports different operating systems: Windows 10, Linux, and Ubuntu 16.04.

After conducting further research, the final camera selected was the mobius mini action camera. This camera provided a nice balance among the different criterias of interest in the sense of this project. As the name suggests, this camera is an action camera that has a 110 degree field of view, it has a dimension of 55 x 29 x 14, and it only weighs about 27 grams with the battery included.

Algorithms

Over the past decades, there have been a couple different methods/algorithms that were developed for Object detection. A lot of studies/research have been done in hope to find the best algorithm when it comes to object detection; however, most of them came to the conclusion there isn't necessarily a best one since that depends on the project's goals.

Viola Jones Framework

Proposed in 2001 by Michael Jones and Paul Viola, the Viola Jones framework was the first and very accurate approach to detect an object in real times [57]. Initially, it was focused to bring a solution to the problem of face detection, but later on as the use of it started to spread across different applications, they found that it can also be used to detect a variety of object classes given the proper training. Looking at it's approach, the Viola Jones Framework algorithm combines four key stages:

1. Simple rectangular/Haar-like features
2. Integral image for rapid features detection
3. AdaBoost machine-learning method
4. Cascade classifier

Haar Features

Haar features are typically used to identify different variations in the lighter and darker portion of an image. After identifying these differences, computations are made which results in a single rectangle around the face which was detected. Multiple research have shown that there are a few common properties in all human faces [122]. For instance, one's eye region is almost always darker than their upper-cheeks, and their nose region. Hence, Haar features can be used to match these regularities.

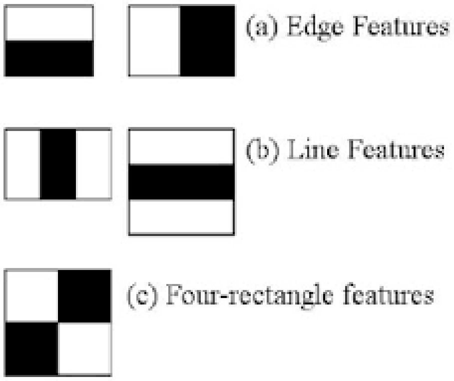


Figure 19: Different Examples of Haar Features

Integral Image

Sometimes referred to as summed area tables, Integral image is an algorithm for quickly and efficiently computing the sum of values in a rectangle subset of a pixel grid [57]. The algorithm was introduced by Frank Crow for its use with mipmaps in 1984; however, it wasn't until 2001 that people commonly started using it in computer vision. Historically, the principle is very well known in the study of multi-dimensional probability distribution functions, namely in computing 2D probabilities [124].

As it is suggested by the name (summed area tables), at any point, the value (x,y) cns the sum of the pixels above and to the left of x,y inclusive:

$$ii(x,y) = \sum_{x\prime \leq x, y\prime \leq y} i(x,y)$$

Where $i(x,y)$ is the pixel value of the original image and $ii(x',y')$ is the corresponding image integral value [57].

1	1	1
1	1	1
1	1	1

1	2	3
2	4	6
3	6	9

Figure 20: Example of Summed Area Table

Adaboost Machine Learning Method

Proposed by Yoav Freund and Robert Schapire in 1996, AdaBoost, which is short for Adaptive Boosting was actually the first boosting algorithm developed for binary classification which ended up being successful. It can be used to boost the performance of any machine learning algorithm [44]. The classification equation is:

$$F(x) = sign\,(\textstyle\sum_{m=1}^{M} \quad \theta m f m(x))$$

Where fm stands for the m-th weak classifier and θm is the corresponding weight [45].

The Viola Jones framework uses a modified version of the AdaBoost learning algorithm not only to select a small set of features, but to also train classifiers that use them. A single AdaBoost classifier consists of a weighted sum of many weak classifiers where each weak classifier is a threshold on a single Haar-like rectangular feature, and it’s defined as:

$$h(x, f, p, \theta = \{1\, pf(x) < p\theta, 0\ otherwise\}$$

Where f *denotes the feature value,* θ is the threshold and p is the polarity indicating the direction of the inequality [57].

Cascade Classifier

Given the fact that only .01% of all sub-windows are actually faces, the Cascade Classifier mainly act as a layer to filter out the negative windows. The cascade eliminates candidates by making stricter requirements in each stage with later stages being much more difficult for a candidate to pass [56]. The only two ways a candidate can exit the cascade is it fails at any stage, or if it passes all stages; if it does the latter, then a face is detected.

To sum it all up, as it was previously mentioned, though it can be trained, the Viola Jones Framework was initially focused on face detection. Therefore, it isn’t really the preferred algorithm when it comes to object detection on a drone. As of currently, it is primarily used in point and shoot cameras, security surveillance and a few other applications.

R-CNN

The introduction of deep structured learning, simply known as deep learning was a major game changer in the field of machine learning. Precisely,in computer vision, deep learning methods started to be used more often and they were crushing the methods that were being used previously. One of the methods that came around with the introduction of deep learning was Regions with Convolutional Neural Networks , mostly referred to as R-CNN. In order for one to fully understand how R-CNN, they must have some understanding of CNN to begin with. Convolutional Neural Networks (ConvNets or CNN’s) are a category of Neural Networks that have been proven very effective in areas such as image recognition and classification [46]. For instance, in recent years, other than powering self driving car and robot vision, CNN’s have had a lot of success when it comes to identifying objects like traffic signs and faces. A Convolutional Neural Network is comprised of one or more convolutional layers, and then followed by one or more fully connected layers [47]. Taking a look deeper at the approach, as a building block, every Convolutional Neural Network has four main operations, and those are : Convolution, Nonlinearity, Pooling, and Fully connected layer.

Convolution

Given an image, the primary purpose of the convolution operator is to extract the features from it. On most occasions, convolution was described

Mathematically, as it is an actual mathematical operation. Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data [46]. When applied to images, Convolution is usually applied in two dimensions; the height and the width of the image. Technically, two buckets are being mixed together, where the first bucket is image given as the input. That given image itself has three matrices of pixels, one for red, green, blue, respectively. The second bucket is the convolution kernel, a single matrix of floating point numbers where the pattern and the size of the numbers can be thought of as a recipe for how to intertwine the input image with the kernel in the convolution operation [47]. The convolution theorem can also be used to further develop the concept. On a higher level, the convolution theorem relates convolution in the time/space domain where convolution features an unwieldy integral or sum to a mere element wise multiplication in the frequency/Fourier domain [48].

$$h(x) = f \otimes g = \int_{-\infty}^{\infty} f(x-u)g(u)\,du = \mathcal{F}^{-1}\left(\sqrt{2\pi}\mathcal{F}[f]\mathcal{F}[g]\right)$$

$$\text{feature map} = \text{input}\otimes\text{kernel} = \sum_{y=0}^{\text{columns}}\left(\sum_{x=0}^{\text{rows}} \text{input}(x-a, y-b)\,\text{kernel}(x,y)\right) = \mathcal{F}^{-1}\left(\sqrt{2\pi}\mathcal{F}[\text{input}]\mathcal{F}[\text{kernel}]\right)$$

Figure 21: Convolution Theorem

In the picture above, the first equation is the one dimensional continuous convolution theorem of two general continuous functions; and the second equation is the 2D discrete convolution theorem for discrete image data. [48].

Non Linearity

After the convolution operation, the Non Linearity operation usually gets applied next. Shortly known as ReLU which stands for Rectified Linear Unit, is an operation that gets applied per pixel and it’s mainly in charge of replacing all the negative pixels with zeros. The purpose of ReLU is to introduce non-linearity in the ConvNet, since most of the real-world data would want the Convnet to learn would be non-linear [46].

Pooling

Convolutional networks sometimes include global or local pooling layers. Spatial pooling also known downsampling or subsampling pretty much keeps all the critical information after it reduce the dimensionality of each feature map; generally, there are three different types of which spatial pooling can be, and those are sum, average, max. When it comes to Max pooling, a spatial neighborhood gets defined at first, and from there the largest element from the rectified feature map gets taken. Otherwise, the average or the sum of all elements in that feature map window can be taken depending on whether one is considering average or sum pooling.

Fully Connected Layer

The fully connected layer basically implies that every neuron in a layer to every other neuron that’s in another layer. The output from the convolutional and pooling layers represent high-level features of the input image [46]. From there, these features are used to classify the image into different classes based on the trained dataset.

Tying it back to Regions with Convolutional Neural Networks, is an early application of CNNs to computer vision, specifically Object Detection. It’s inspiration came from the research of Hinton’s Lab at the University of Toronto. R-CNN’s main goal was able to take an image as input and correctly detect where the main objects are within the given image, and it will do that via a bounding box. Looking at it from a higher level, R-CNN is pretty much made up of three main steps which are as follows:

1. From the image given, come up with a set of proposals for bounding boxes.

2. Using the set of proposals from step 1, in order to see what object is actually in the box proposed, the images in the bounding boxes get ran through a pre-trained AlexNet and an Support Vector Machine (SVM).
3. Lastly, run the box through a linear regression model in a hope to be more precise, and to output tighter coordinates once the object has been identified.

In the first step, R-CNN is able to create these bounding boxes by using a process called selective search. Selective search combines the strength of segmentation and exhaustive search. As a quick overview, segmentation aims for a unique partitioning of the image through a generic algorithm, where there is one part for all object silhouettes in the image [58]. On the other hand, exhaustive search searches everywhere in an image because an object can be located at any given scale and position. From that combination, Selective search hopes to capture the location of all the possible objects, and the image structured is used to guide the sampling process. In most cases, any selective search algorithm is somewhat bounded by these three design principles: fast to compute, diversification, and capture all scales. The first design consideration, fast to compute pretty much highlights the fact that making the object locations set should never become a computational bottleneck. Diversification is because the algorithm should have a diverse set of to deal with all possible cases given the lack of an optimal strategy that allows the regions to be grouped together. For instance, different conditions can have somewhat of an impact how regions form an object; one of these conditions is the lighting condition. In selective search, all objects scales must be accounted for, hence the algorithm should capture all scales. Typically, that design consideration is done using hierarchical algorithm. Basically, to sum up selective search, it looks at the image through windows of different sizes, and for each size tries to group together adjacent pixels by texture, color, or intensity to identify objects [49]. To achieve that, selective search uses three main steps. The first step is to generate an initial sub-segmentation, the second is to combine the similar regions into larger ones recursively precisely using a greedy approach as far as the algorithm. The last one is to use the generated regions to produce candidate object locations.

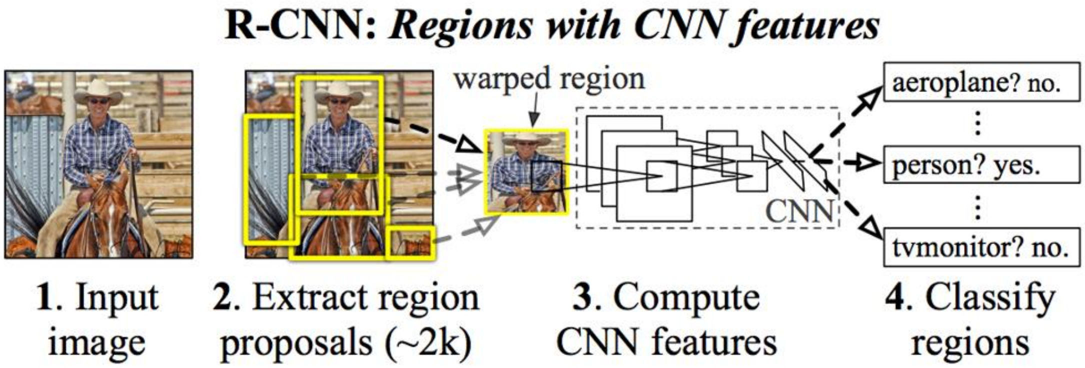


Figure 22: R-CNN passes image through AlexNet

Looking at the second step, once the set proposal is done, R-CNN passes it through a modified version of AlexNet as illustrated above. Briefly, AlexNet is the name of Convolutional Neural Network that was designed by SuperVision group whose members were Geoffrey Hinton,Ilya Sutskever, and Alex Krizhevsky. Originally written to run with Graphing Processing Unit (GPU) support , the network participated in the ImageNet Large Scale Visual Recognition Challenge back in 2012, and it did pretty well.

Lastly, focusing on the third step of the R-CNN adds a Support Vector Machine that specify what the object is, if it is indeed an object. After identifying the object, a simple linear regression gets ran on the region to generate closer bounding box coordinates for the object in the sub-region.

The Regions with Convolutional Neural Networks algorithm works pretty well when it comes object detection, however it gets really slow at times for a couple of reasons. For instance, every single region proposal has to go through a pass of AlexNet, and sometimes, that can take a lot of time. Additionally, it’s pipeline is extremely difficult to train given that it has to train three different models one by one. Those models are the Convolutional Neural Network model that generate image features, the classifier that predicts the class, and finally the regression model that gets used to output tighter coordinates in the bounding boxes. Knowing these drawbacks, people wanted to solve both of these issues, and with that, came the introduction of Fast Regions Convolutional Neural Networks

Fast R-CNN

One of the first author of Regions with Convolutional Neural Networks later noticed some flaws in the approach, so he took it upon himself to try to make it better. His main observation was in R-CNN, the same CNN computation was being ran again and again, nearly around 2000 times. Therefore, he thought it would be better to run the CNN computation just a single time, and then come up with an ultimate way to to share that computation across all the proposals. Ideally, this is what Fast R-CNN does using a technique called Region of Interest Pooling commonly referred to RoIPool. Region of interest pooling is another widely used operation in object detection. Its goal is to obtain fixed-size feature maps by performing max pooling on nonuniform sizes inputs. RoI pooling is a neural-net layer that takes two inputs. The first input is a fixed-size feature which will then contain several convolutions and max pooling layers. The second input is an N x 5 matrix that represents a list of the interested regions, where N would be the number of Region of Interests. Additionally the first column is the image index and the others are the coordinates of the top left and bottom right corners of the region.

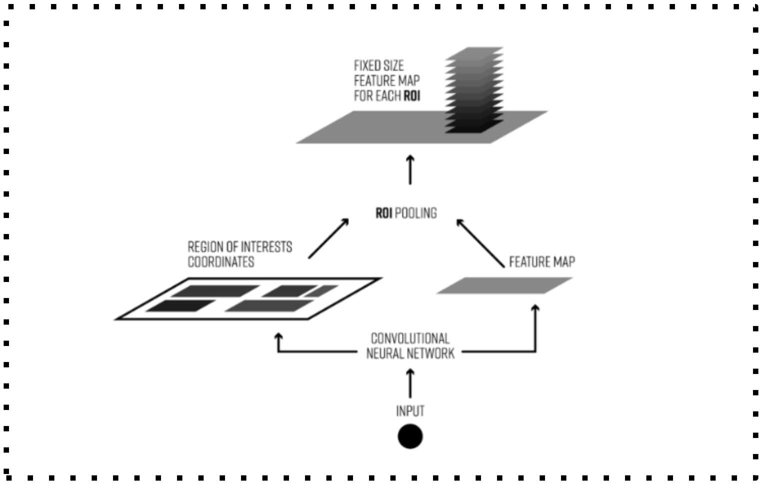


Figure 23: Object Detection pipeline with region of interest pooling

Taking a deeper look at what the RoI pooling actually does, one can say it takes a part of the feature map and scales it to some predefined size; the scaling itself is mostly done by:

1. Dividing the region proposal into equal-sized sections
2. Finding the largest value in each section
3. Copying these max values to the output buffer

The result is that from a list of rectangles with different sizes we can quickly get a list of corresponding feature maps with a fixed size [50]. The RoI pooling output's dimension has nothing to do with the size of the region proposals nor the size of the input feature map. It mostly depends on the number of sections that proposal gets divided into. The introduction of RoI pooling came along with a quite a couple benefits, and certainly its processing speed was one of the it's main one. The fact that convolutions computations especially at early stages of processing is a really expensive operation, this approach becomes a time safer.

The result is that from a list of rectangles with different sizes we can quickly get a list of corresponding feature maps with a fixed size [50]. The RoI pooling output's dimension has nothing to do with the size of the region proposals nor the size of the input feature map. It mostly depends on the number of sections that proposal gets divided into. The introduction of RoI pooling came along with a quite a couple benefits, and certainly its processing speed was one of the it's main one. The fact that convolutions computations especially at early stages of processing is a really expensive operation, this approach becomes a time safer.

After the region of interest pooling layer is done extracting feature vector from the feature map, it now becomes time for fine-tuning for detection. Just as the role of the RoI pooling layer, training all network is also an important capability of Fast R-CNN. In Fast R-CNN training, stochastic gradient descent (SDG) mini-batches are sampled hierarchically, first by sampling N images and then by sampling R/N RoIs from each image [124]. Through that process, one critical thing is that RoIs that are from the exact same image share memory and computation in both the backward and forward passes. Additionally, Fast R-CNN uses a streamlined training process with one fine-tuning stage that jointly optimizes a softmax classifier and bounding-box regressors [124].

As a last step, once a Fast R-CNN network is done with the fine-tuning phase, detection simply becomes a matter of running a forward pass; that is of course based on the assumption that all object proposals are pre-computed. While in that stage, the network takes a list of R objects proposals and an image or even a list of images as input. For each test RoI *r*, the forward pass outputs a class posterior probability distribution p and a set of predicted bounding-box offsets relative to r (each of the K classes gets its own refined bounding-box prediction) [124].

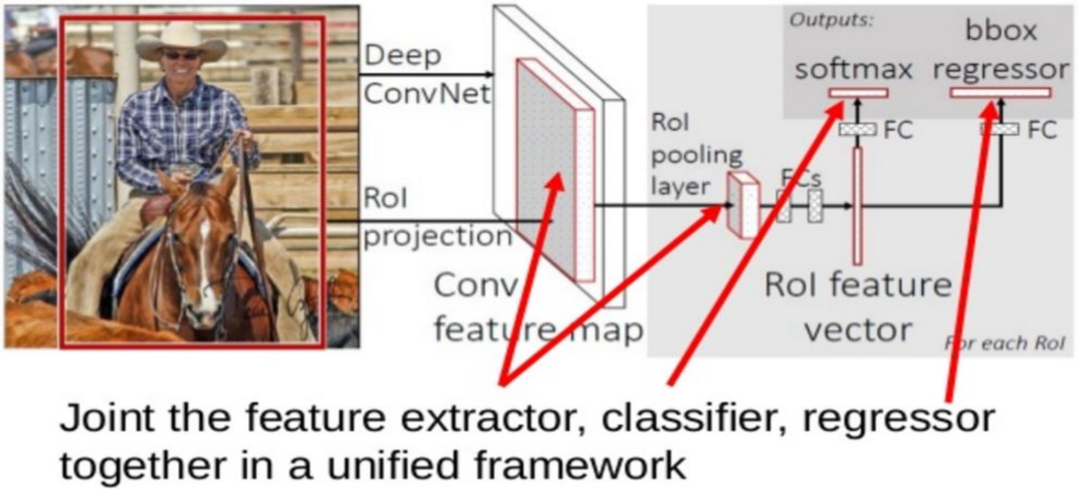


Figure 24: Fast R-CNN: Joint Training Framework

In summary, Fast R-CNN uses a single network to do three main operations: extract the image features, classify, and then finally tighten bounding boxes. In reference to the picture above, a softmax layer on top of the convolutional neural network replaced the SVM classifier in order to output a classification. Additionally, a linear regression layer that is parallel to the softmax layer get added to take care of the bounding box coordinates. As its main input, the model accepts an image or a list of images with region proposals, and classifications of each region gets outputted along with tighter bounding boxes.

You Only Look Once (YOLO)

The object detection algorithms described above (R-CNN, Fast R-CNN) belong in the two stage detection family. Unfortunately there are a couple drawbacks when it comes to most of the two steps detection algorithms. For instance, they cannot be ran in real time, because their speed is often slow. Each of their components are somewhat hard to optimize, and lastly, they have a very complex pipeline. Those drawbacks motivated computer scientist to come up with a better solution, and thus came the You Only Look Once algorithm. Commonly referred as YOLO, it is a new approach to object detection where it's entire pipeline is a single network. The goal behind the introduction to YOLO was to able to replicate an object detection algorithm that somewhat replicates the human visual system. The human visual system is very accurate and it is also very fast; at once glance, a person is usually able to look at an image or a set of images and instantly identify all of the objects that are in the image(s), and their location. Although it often gets overlooked, but it is because of such system, that humans are allowed to do very complex tasks such as reading, driving, shooting and a lot more, without putting much work into them. Therefore, having an object detection algorithm that behaves as close as possible to the human visual system (fast and accurate), computers would be allowed to drive cars without the need of specialized sensors, convey real-time scene information to humans using assistive devices, which will all sum up to us having better and more responsive robotics systems.

YOLO reframed object detection by considering the detection aspect as a regression problem; it goes directly from the pixels in an image to to bounding box coordinates and then to class probabilities. Using the system, an imaged only gets looked at once (hence, where the name came from), and it’s able to predict what objects are in the image, and where in the image they’re located. As a quick overview, YOLO is a three step system. The first step is to resize the given image, secondly it runs the convolutional network, and last but not least, the resulting detections gets thresholded by the model’s confidence. (See picture below).

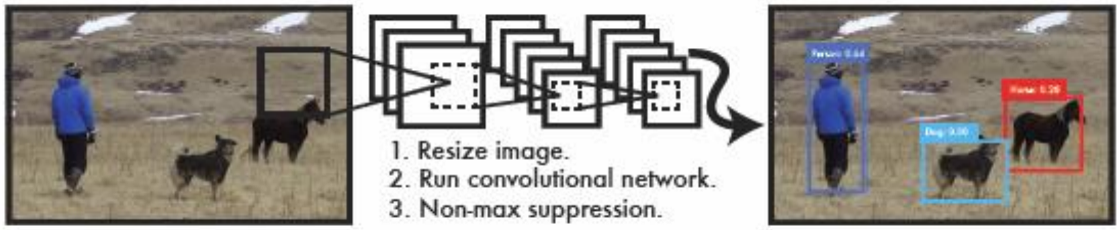


Figure 25: The YOLO Detection System

The introduction of YOLO also came along with the concept of an unified model; which is a lot more beneficial than the concepts used in the algorithms previously described. In short, the different components of object detection gets combined into a single network. The single convolutional network is able to predict multiple bounding boxes and class probabilities for these boxes at the same exact time. This technically means that the network thinks about the image, and all of the objects that are in it. Taking a deeper look at the first step of the system, it starts by resizing the input image; dividing it into an S x S grid. From there, each cell in the grid is responsible for predicting bounding boxes and confidences scores for those boxes. These confidence scores reflect how confident the model is that the box contains an objects and also how accurate it thinks the box is that it predicts [125]. Each bounding box that gets predicted contains 5 different predictions, and those are *x, y, w, h* and the confidence. The center of the box are represented by the (*x,y*) coordinates. The *w* represents the width, and the *h* represents the height; those are measured relative to the entire image. Lastly, the confidence prediction represents the intersection over union (IOU) between the predicted box and any ground truth box. When it comes to the confidence score in a bounding box, if there’s no object in that cell then the confidence score should be a zero. In any other case, the confidence score should equal the IOU between the predicted box and the ground truth. In addition to the 5 predictions, each cell also predicts a conditional class probabilities. It pretty much works like a classifier, and it's responsible for giving a probability distribution over a list of all possible classes. Some examples of the classes can be : an animal, a car, a person, etc. The confidence score for the bounding box and the class prediction are combined into one final score that tells us the probability that this bounding box contains a specific type of object [125]

$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}}$$

Figure 26: Confidence score equation

Considering the example where the network divides the image into a 10 by 10 grid, there was a total of 100 cells of which will individually predict 5 bounding boxes. Therefore, there was a total of 500 bounding boxes which are all generated simultaneously . Out of that 500, quite a few of them happen to have a very low confidence score, so they get discarded. Only the boxes whose final score is above a set threshold value are kept; that threshold value itself can be changed depending on accurate one want detector.

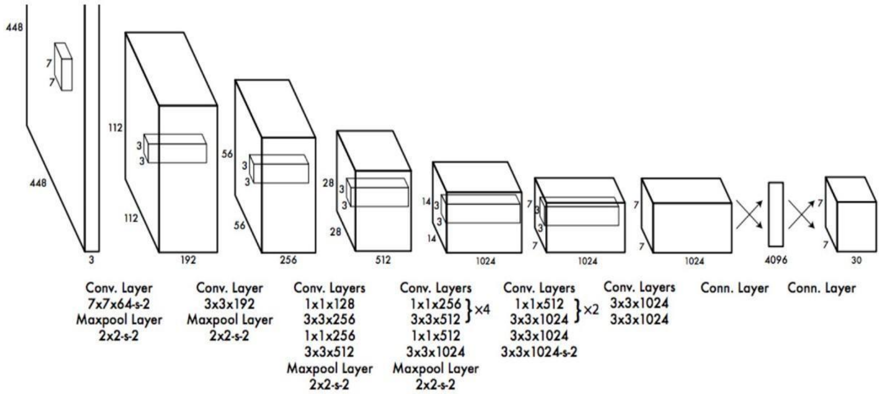


Figure 27: YOLO's architecture Con

Highlighting the architecture of the YOLO algorithm, the detection network contains 24 convolutional layers which are then followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. The convolutional layers gets pretrained on the ImageNet classification task at half the resolution and then double the resolution for detection [124].

Compared to the algorithms previously covered, YOLO is a really extremely fast. Given the fact that the approach reframed object detection as a regression problem, a complex pipeline is no longer needed. Secondly, YOLO has the ability to think globally about the image itself when making predictions.
Lastly, YOLO learns generalizable representations of objects. When trained on natural images and tested on artwork, YOLO outperforms top detection methods like DPM and R-CNN by a wide margin [124]. On the other hand, YOLO also have a few limitations.

For instance, it often struggles with small pictures and unusual aspect ratios, and it also struggles to precisely localize some objects.

3.4.2 Object Tracking

After the object(s) have been successfully detected, the drone now have to be able to track it in order to take it down. Through object tracking, a system gains the ability to predict future positions of one or multiple moving objects based on the history of these objects. Over the years, computer scientist and researchers have been able to develop some pretty good tracking algorithms. Below is a short list of the ones that was considered for the purpose of this project

Algorithms

1. Nearest Neighbor
2. Interactive Multiple Model
3. Probabilistic Data Association
4. Multiple Hypothesis Tracking

3.5 First Person View (FPV)

Prior to the invention of FPV technology, drones had to be flown in direct line of sight of the pilot to be safe. These days, after much more research and development, drones have much more advanced capabilities that allow the pilot to be stationed far away from the area the drone is being flown in. Drones with FPV technology have been used in a variety of areas, most notably aerial racing and cinematography. The popularity of drones has definitely skyrocketed because of FPV flying capabilities [10]. New manufacturing techniques and a reduction in component costs have allowed hobbyists to gravitate towards this new way of flying.

Throughout the RoboCopter’s competition the UAV drone must be able to transmit a live video stream down to a ground station. The video stream was a first person look at what the drone is able to see and will essentially allow viewers to see how the drone behaves during the competition. The on-board processor was responsible for ensuring a clear, continuous transmission during the flight.

To ensure a clear, consistent video feed during the competition, the transmission will have to overcome some common issues with FPV. Multipath fading and interference are two of the main issues that may be encountered, the latter being more of a concern for our group.

Multipath fading

Multipath fading is something to take into consideration during any form of radio communication. During multipath the signal is sent along two or more different paths to the receiver. Many factors such as atmospheric ducting, ionospheric reflection and refraction, and reflection from terrestrial objects, such as mountains and buildings can cause multipath. Fading refers to the distortion that a carrier-modulated telecommunication signal experiences over certain propagation media.

The location for the competition was in a crowded college campus, full of buildings, trees, students, etc. Even with a strong, noise free signal, a radio link can get sudden dropouts, especially in cluttered or urban environments [11]. Dropouts can be caused from the reflection of the signal on a number of different objects. The figure below shows an example of multipath propagation and some of the effects from surrounding objects such as buildings and trees.

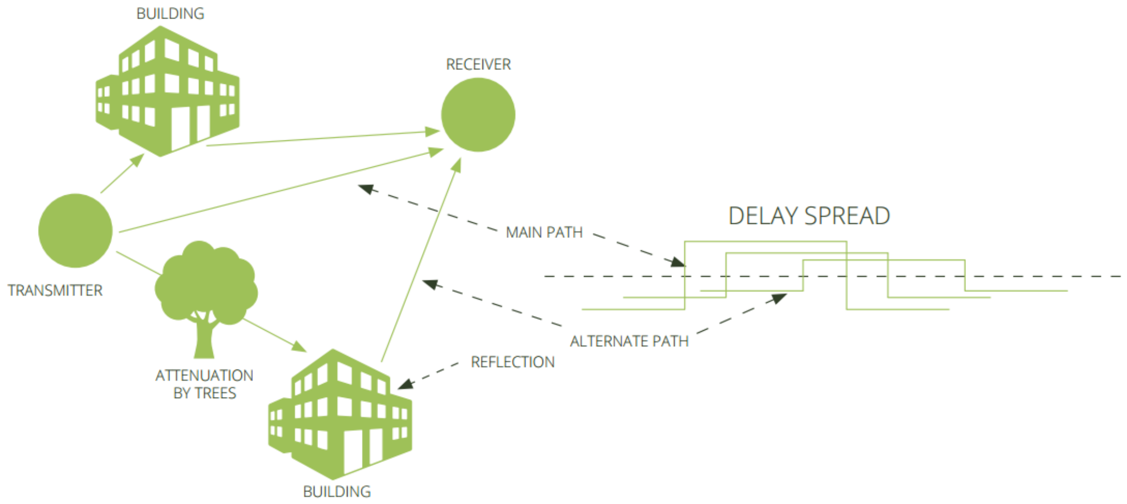


Figure 28: Multipath Propagation [12]

Interference

As with multipath fading, factors in the environment impact the amount of interference that can be encountered during transmission. If the signals causing the interference occur in the same frequency band as the wireless video link it will act as inband noise causing the signal to noise ratio to decrease [11]. When the signal to noise ratio decreases it results in a noisy video signal.

To overcome this common issue, the frequency used for transmission needs to be on one that is outside of the range of frequencies used by the common interfering objects. If the object’s interference is powerful then it is called a blocker. Blockers do not have to be on the same frequency range as the drone. The blocking signal can penetrate insufficient front-end channel filtering, and decrease the dynamics of the Low Noise Amplifier (LNA). The figure below shows a simple diagram of the receiver signal.

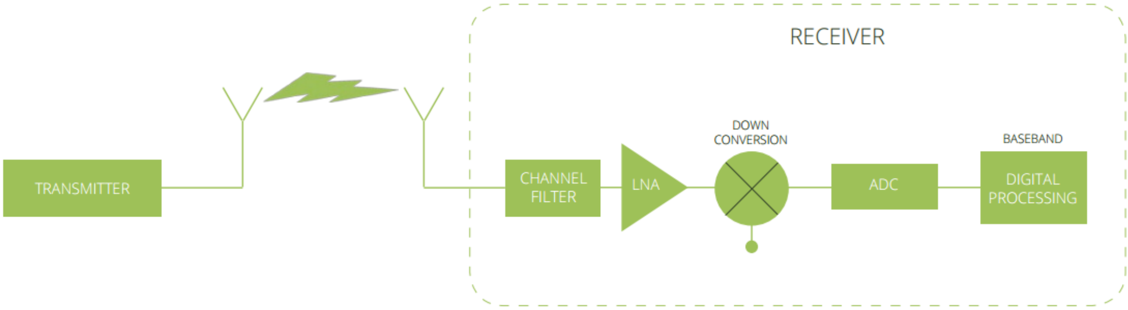


Figure 29: Signal Receiver Chain [12]

Frequency Selection

For the first person video to operate, a specific radio frequency will need to be decided upon. The two most common frequencies for FPV on drones include 2.4 GHz and 5.8 GHz. Other frequencies are available to use such as 900 MHz and 1.2 GHz but would need heavier, larger equipment and may also require specific licenses to operate at these frequencies so we will avoid comparisons with these frequencies for this project. Another way to transmit the video feed would be through WiFi which typically uses a digital signal at 2.4 GHz. The advantages and disadvantages of each option was explored to help determine the best route to take for the job based on the needs of this project.

2.4 GHz Analog

Analog transmission of video at 2.4 GHz has been outgrown over the years. At one point in time, during the beginning stages of FPV, it was the most widely used frequency. The technology operating at this frequency suffers interference from other devices in the area, especially in areas closer to homes and businesses. Therefore, the amount of interference and signal fading that could be encountered during the competition would not be suitable for the purposes of this project.

2.4 GHz Digital (WiFi)

Transmitting video over WiFi has many advantages. Most devices (smartphones, tablets, laptops) can communicate over a reliable WiFi network. Most people in the world own a device that is capable of using WiFi so to save from buying additional devices, such as a display to receive the video, the downlink can be directed to a device already owned by the customer. One significant drawback of using WiFi for video transmission is its range. The range of the signal can only go as far as allowed by the network. A solid, reliable connection such as WiFi would meet the needs of this project but the competition area may be conducted away from WiFi access points causing issues that may result in a failed product.

5.8 GHz Analog

Another form of transmitting the first person video is using the 5.8 GHz frequency range. Many of the top FPV drones out on the market use this frequency and it has proven to be very reliable. 5.8 GHz gives range and bandwidth that is more than capable of meeting the needs for this project. This frequency range also has less interference and attenuation issues like other frequencies that are used. Most hobbyists and professional grade drones use this frequency for first person video.

FPV Transmitter Power

The transmitter power is another aspect of the FPV transmission video that needs to be considered. The power was listed in mW and it is suggested for beginners to not exceed 500 to 600 mW. The transmitters that operate at these higher ranges are necessary in environments that have a lot of obstacles in between the drone and the receiver. The antenna is the key to success in terms of range with the FPV transmission. The antenna gives improved range but not more powerful transmission. Some countries will limit the power of transmitters for different reasons and will require licenses greater than certain power values. For the purposes of this project we will not need a lot of power for our transmitter. Our drone was operating within close proximity to our ground station and is suggested to work around the 25mW range.

3.6 Frame/Structure

In order to begin designing a PCB or calculating an accurate estimate of power consumption, a drone layout must be chosen. Having the basic design will make it possible to start making weight estimations and battery requirements and start looking at motors, propellers, batteries, etc.

Basic Quadcopter

Basic Quadcopters are generally lighter, cheaper and have less components than the other options which means less components to fail. They are usually smaller and more agile which is important for avoiding obstacles on the field. This layout has the most available resources in terms of components, code, and past projects to draw experience from. The downsides are few, but they typically have less potential for a heavy load than the other options [60].

Flat Hex and Octocopters

A flat or radial drone means it has an arm for each motor that are evenly spaced with a motor and propeller on each arm. These have more power than a basic quadcopter and can still fly if up to four of the motors go out, depending which ones [61]. This layout will have more arms which means it is heavier and larger. A larger drone will need a bigger cage and adding weight makes the drone exponentially more expensive with the need for bigger motors, battery, etc. Another big concern is writing the code for more motors. Complications need to be kept to a minimum in order to have software that runs smoothly and to avoid any failures during the competition. The setups have the potential for more power, but they are more suited for carrying professional, high-definition camera equipment because of their carrying capacity and stability [62]. For this competition, the price must be considered as well as the ability to avoid the obstacles.

Coaxial Hex and Octocopters

A coaxial multirotor drone could be a viable option however. A coaxial hexacopter (Y6) will have three arms with two motors on each, and a coaxial octocopter (X8) with four arms and two motors on each. The bottom propeller having a higher pitch than the top one to speed up the air coming from the top propeller. These setups have less arms per motor and would require a smaller cage than their radial counterparts giving it a higher power-to-weight ratio. There are many cons with this setup though.The two different pitch propellers would require different torque motors and testing to see which propeller combination works well [63].

Extra motors are louder and create excess noise at high frequencies that could interfere with the supersonic range finders and the extra vibration will affect the camera and other components [64]. The turning is also said to be not a responsive as a quadcopter because the heavier arms tend to not want to turn as quickly and accurately, which is big obstacle in being able to track down the prey and avoiding obstacles. A much larger battery with an high enough C rating and capacity would add much more weight. The bottom propeller pushing the “dirty” air from the top requires more energy and make this design less efficient [65].

A powerful quadcopter seems to be the best option because of their simplicity, agility, and the available resources in terms of components, software, and past projects to draw experience from. Certain restrictions like time constraints and moderate budget does not allow for a large powerful drone with more motors. Professional drones like this can easily cost thousands on their own.

3.7 Propulsion

Motor Requirements

To find a suitable motor for the drone, the MAE and ECE teams will collaborate to determine how much maximum thrust is needed and a propeller that works efficiently with the motor. As a general rule, the quadcopter should be able to hover at about 50% thrust to be able to maneuver well.

When choosing a motor, the KV rating will give insight on what type of applications the motor is suited for. When referring to brushless motors, the KV rating refers to the revolutions the motor will make when 1 volt is applied and no load. This is a theoretical value because a brushless motor should not be ran with no load, but it is a figure that is decided by the gauge and number of windings inside the motor [66]. A motor that has thinner wire with more winding gives a lower KV rating, but a higher torque. A motor with thicker wire and less windings gives a higher KV rating but less torque, in general. Larger propellers with a higher pitch require more torque, a lower KV rating [67].

This application required a moderately low KV rating because of the torque required to carry the load of the sensors, cameras, cage, etc, but has a decent amount of rpm to be quick and responsive to commands. High KV motors can spin the propeller at high RPMs and can move a smaller drone faster.

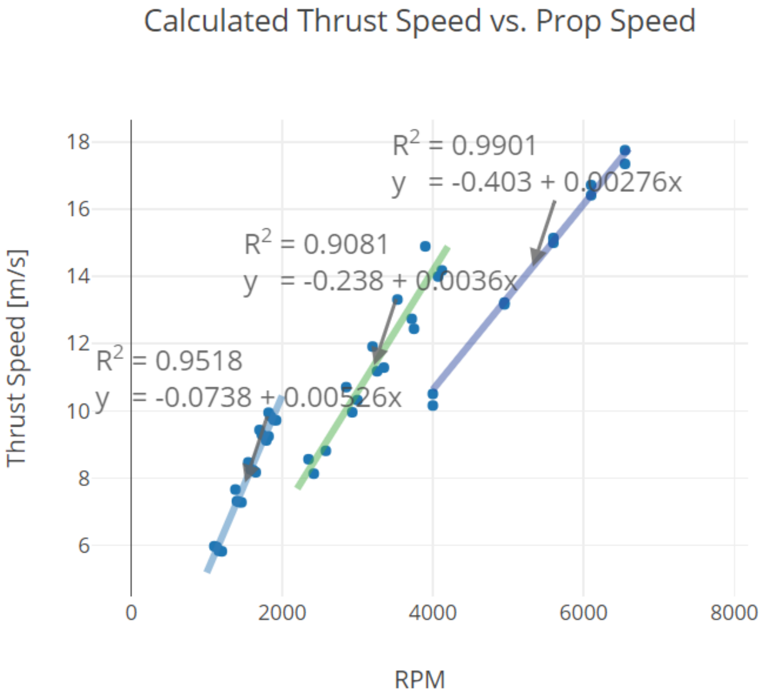


Figure 30: Propeller RPM compared to Thrust Speed [68]

The small arena did not require the drone to have a high top speed, so drag is not much of an issue, which is one reason why racing drones have relatively high KV ratings. Using the chart below, for a 450mm frame, a 1000KV or lower motor with 8" propellers or larger are recommended [69]. This will require a relatively large battery and speed controllers.

Frame Size	Prop Size	Motor Size	KV
150mm or smaller	3″ or smaller	1105 -1306 or smaller	3000KV or higher
180mm	4″	1806	2600KV – 3000KV
210mm	5″	2204-2208, 2306	2300KV-2600KV
250mm	6″	2204-2208, 2306	2000KV-2300KV
350mm	7″	2208	1600KV
450mm	8″, 9″, 10″ or larger	2212 or larger	1000KV or lower

Table 6: Propeller diameters and motor ratings for frame sizes

Taking these factor into account, it was decided to use Sunnysky 2216 800KV motors. They have a good balance between efficiency and thrust in a lightweight design. These motors are to be paired with 10 inch carbon fiber propellers with a pitch of 4.7. With this combination, each motor is able to provide a max of 960 grams of thrust.

3.8 Battery/Power Distribution

Multirotor drones almost exclusively use Lithium Polymer batteries because they have a higher energy density than the alternatives like Nickel Cadmium and are made in shapes to fit a compact quadcopter [70]. This makes Lipo's the obvious choice for the most efficient drone with the highest power to weight ratio.

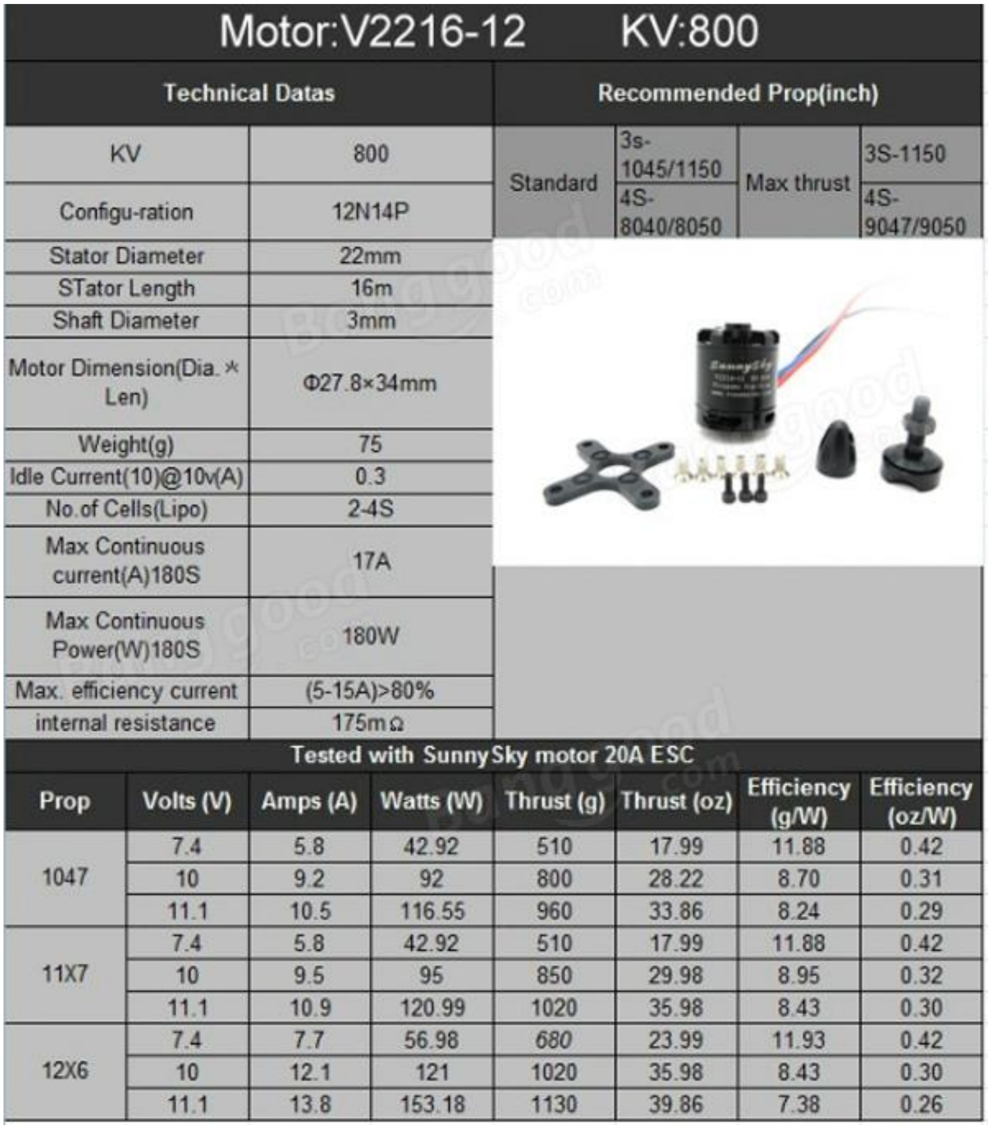
Battery Requirements

Choosing the right battery involves estimating the total weight and thrust requirements of the drone to determine the amp draw and energy consumption of the motors. Choosing the right battery is important to ensure the drone has enough battery capacity to complete the task, without overtaxing the battery which could result in overheating or a fire. Choosing a battery that exceeds the requirements will make the drone unnecessarily heavy and less efficient and maneuverable.

Evaluation Criteria	Weight of Importance
Max Discharge Rate	**35%**
Capacity	**30%**
Mass	**14%**
Cost	**7%**
Size	**5%**
Cycle Lifetime	**4%**
Noise	**3%**
Max Charge Rate	**2%**

Table 7: Battery Requirements Weight of Importance

The first step is to determine the components being used for the quadcopter. The motors used are the Sunnysky 2216 800KV motor, with 10 inch propellers that have a pitch of 4.7. The chart below shows the thrust and amp draw of this combination of motor and propeller for 11.1 volts, which is the more efficient voltage in this case [71]. This will give an idea of how many amps was drawn by the motors, which is the vast majority of power consumed.



Motor:V2216-12 KV:800

Technical Datas	
KV	800
Configu-ration	12N14P
Stator Diameter	22mm
STator Length	16m
Shaft Diameter	3mm
Motor Dimension(Dia.＊Len)	Φ27.8×34mm
Weight(g)	75
Idle Current(10)@10v(A)	0.3
No.of Cells(Lipo)	2-4S
Max Continuous current(A)180S	17A
Max Continuous Power(W)180S	180W
Max. efficiency current	(5-15A)>80%
internal resistance	175mΩ

Recommended Prop(inch)			
Standard	3s-1045/1150	Max thrust	3S-1150
	4S-8040/8050		4S-9047/9050

Tested with SunnySky motor 20A ESC

Prop	Volts (V)	Amps (A)	Watts (W)	Thrust (g)	Thrust (oz)	Efficiency (g/W)	Efficiency (oz/W)
1047	7.4	5.8	42.92	510	17.99	11.88	0.42
	10	9.2	92	800	28.22	8.70	0.31
	11.1	10.5	116.55	960	33.86	8.24	0.29
11X7	7.4	5.8	42.92	510	17.99	11.88	0.42
	10	9.5	95	850	29.98	8.95	0.32
	11.1	10.9	120.99	1020	35.98	8.43	0.30
12X6	7.4	7.7	56.98	680	23.99	11.93	0.42
	10	12.1	121	1020	35.98	8.43	0.30
	11.1	13.8	153.18	1130	39.86	7.38	0.26

Figure 31 - Thrust Chart of Sunnysky 2216 800KV motors with APC1047 Propellers [72]

Max draw is 10.5 amps for one motor, or 42 amps for four m

<u>Estimating Capacity</u>

- Frame = 240g
- 4 motors = 300g
- Camera = 70g
- Battery = 500g
- PCBs = 150g
- Flight Controller = 150g
- Cage = 300g
- 4 ESCs = 40g

Total = 1740g

As a rule of thumb, the drone should be able to hover at half thrust, having a thrust to weight ratio of 2 [73]. If the motors produce 960 grams of thrust each for a total of 3840 grams, the 1740 gram drone will have a thrust to weight ratio of 2.21. Assuming an average thrust of 2200 grams to calculate a capacity estimate requires 6 amps for each motor. Four motors will draw 24 amps and assuming another amp for other components gives a total of 25 amps. The drone will need to operate for ten minutes, or 0.167 hours. Supplying 25 amps for 0.167 hours requires 4175 mAh.

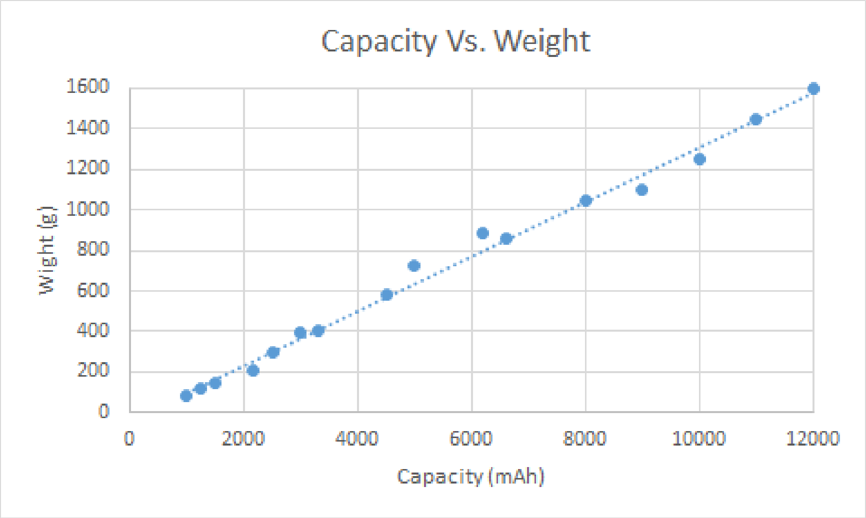


Figure 32 - Capacity vs. Average Battery Weights

It is common practice to add an extra 20% to the capacity to avoid draining the battery too low [74]. Draining the battery too low not only is risky for not being able to complete the task, it causes chemical reactions that are not intended and negatively affecting capacity and internal resistance, making the battery less efficient. It is also recommended to store LiPo batteries at around 50% charge for the same reason [75]. Taking this into account makes the capacity requirement closer to 5010 mAh.

The C rating of a battery is a theoretical value that gives an idea of how many amps can be drawn in proportion to the capacity of the battery [76]. To find the max amperage that the battery is rated for, the C rating is multiplied by the capacity of the battery. Since capacity and max amperage requirements are known, the amps needed are divided by the amp-hours. A safe estimate of 80 amps was made, meaning a minimum C rating of at least 16.

This Venom Lipo is an acceptable option that meets all the requirements. It has enough capacity and can supply enough amps with 108 continuous amp draw even for power spikes from the motors. It operates at 11.1 volts [77] which is the recommended voltage for the motor, and weighs less than the figure used when calculating weight estimates and has a higher capacity which will give more freedom when choosing motors and can deliver more current for longer to get more thrust.

Venom 3S LiPo

- 11.1v
- Capacity: 5400mAh
- Discharge Rate: 20C
- 1.75 x 8.88 x 3 in
- Weight: 390g

Figure 33: Venom LiPo

Power Distribution/Battery Eliminator

A power distribution board (PDB) or battery eliminator circuit (BEC) was needed to power the components of the drone. A battery eliminator circuit is a circuit that eliminates the need for a separate battery by regulating voltage to a component. Battery eliminator circuits are often included in the electronic speed controllers to power the flight controller [78]. This configuration has its pros and cons, but the ESCs with built-in BECs are prone to overheating or burning up, especially when dealing with an 11.1 volt, 3S or more battery. This is because an BEC is basically a voltage regulator that step-down the voltage by dissipating the excess energy as heat [79]. This should be considered when deciding how to power the flight controller.

The chosen frame, the DJI Flame Wheel, comes with top and bottom connector plates with drilled holes and pads to solder wires to and distribute power. With this particular frame the ESCs are often soldered to the plate and a male XT60 cable can be soldered so that the battery can be connected and disconnected for charging or replacement. The only concern is with relatively high amp loads, will the copper traces on this plate be enough to safely and efficiently transfer power. To make sure that this method was effective, the resistance between battery input pads and the pads for the ESCs were measured to see how much power would be dissipated.

Figure 34: DJI Flamewheel PDB

Figure 35: DJI Flamewheel PDB Resistance

The highest resistance recorded was less than 0.2Ω. Assuming a max total of 44 amps running through these copper traces, the maximum value that could be dissipated is shown below:

$$P = i^2R \ = \ 20^2 \times 0.2 \ = 387.2\, W$$

This is much more than expected and would be an unacceptable amount of lost power. However, resistor values below 1 Ohm are hard to measure accurately. Additional resistance could also be coming from the alligator clips used. Very low resistance should be done by setting up a circuit with the desired resistor to be measured and checking the voltage drop across the desired resistor and dividing by the amperage.

For this application, more voltages are required than the average quadcopter. There will need to be an output for the flight controller, the PCB, and the components used to output the autonomous controls. A Raspberry Pi or Arduino was used with the camera to output detection and tracking data. An Arduino can be powered with 7-12 volts using the V_{IN} pin and has a 3.3 and 5 volt output for the camera(s) [80]. A flight controller like the PixHawk Mini 3DR can take up to up to 45 volts for input and have 5 volts outputs that could be used to power the the PCB [81], meaning the speed controllers could be connected straight to the 11.1 volt battery.

High amp draw from the motors can cause a dip in the voltage from the primary battery and could cause a loss in functionality from the electronics [82]. Because of this, a secondary battery was used to power all electronics. For this, a small Turnigy 2S LiPo with 1000 mAh was used, meaning the PCB must convert a nominal 7.4 volts to 5 volts which is much easier and more efficient than trying to convert the 11.1 volts from the primary battery.

3.9 PCB Design

This drone will use a PCB that works alongside a flight controller to take that data from the camera and ultrasonic sensors and convert it into a signal that simulates a traditional RC transmitter. To design a PCB, a diagram should first be made to outline the components and give an idea of the requirements for the board [83]. The board was based around a microcontroller with pins for all the data collected by the sensors/cameras, and voltage out pins with the correct voltage for each component. The microcontroller will contain the algorithms for creating autonomous flight controls. From this, it can be determined if the board needs 3.3 volts or 5 volts, how many layers and how big the board was, and other important decisions. The basic components are shown in the hardware block diagram. Once a board design is finalized, the Gerber file was generated and sent to a manufacturer [84].

Design Software

To design the board, a PCB design software must be used. There are many different options in which design software to use that all have pros and cons. This section will cover some of the most popular design softwares and discuss these pros and cons to choose the one best suited for this project. To choose the most appropriate one, the requirements must be outline. In order to be viewed and edited by anyone in the group, on their own device, the software should be easily obtainable or downloaded and installed. The software should also be easy to use to save time in everyone working on the design becoming familiar with the software. The software should also have plenty of resources in terms of tutorials and previous designs.

Altium Designer

Altium is a very popular and powerful design package often used for professional applications. Altium has many features designed to increase design efficiency and ease. Altium is meant to be able to design a board for almost any application and has many features that may be unnecessary for a simple board. This can make the software more complicated and harder to learn. It is good for large projects with multi-layer boards [85]. Since Altium is for professional use, it requires a yearly subscription which is $7,245 for Altium Designer 2013 [86] and can only be ran on Windows computers. Being such a powerful application, the software uses a lot of system resources.

EAGLE

EAGLE is a design application used for both professional and educational uses. Students and professors use the full package for free. EAGLE stands for Easily

Applicable Graphical Layout Editor [86]. EAGLE can be ran on Windows, Mac, or Linux, which is a convenient feature when collaborating with members using their personal computers. EAGLE is also very lightweight and does not require a powerful computer [87]. This is good for working on the design and having it open while researching and recording information. There is also a large community of drone builders that use EAGLE which would be useful for researching and resolving issues and a huge library of components available to download.

KiCad

KiCad is a free and stable PCB design application. It is open source software, meaning anyone can alter the source code and there are versions that are tailored to the needs of the PCB for this project. It can be run on Windows, Mac, and Linux and is lightweight. There is plenty of tutorials and previous projects to look at and the software is easily learned with many useful keyboard shortcuts to speed up design [88].

For this project, Altium is too expensive and has more features than would be required. EAGLE and KiCad have similar pros and cons, they are both free, cross-platform, and have many resources to help in the design. KiCad, is said to be more intuitive for new users to learn with commands linked to keys [89]. The deciding factor is the amount of support for quadcopter projects. EAGLE has a larger library of components available for free and is the preferred program for multirotor hobbyists.

Design

Before starting the design, the components going on the board must be determined. The decided functionality is for the PCB to hold a microcontroller that receives the data from the sensors, and sends the data to a separate microcontroller with information about where the sensor is. The board must also be able to power these components with steady current and optimal voltages for the different components. The board should also have noise filtering circuitry for reference voltages of the microcontroller and efficient voltage regulation from the battery.

Microcontroller

The microcontroller being used in the PCB was the Atmega328p because of the amount of resources available in terms of past projects and troubleshooting. The Atmega328p is a 28-pin, 8-bit microcontroller with 23 general purpose I/O pins for receiving and sending data . The 328p is a low-power MCU operating at 1.8 - 5.5 volts with higher frequency clock rates achievable at higher voltages with the max 20MHz at voltages between 4.5 and 5.5 V. The clock rate achieved roughly translates to the amount of instructions per second [90] Having a Vcc of 5 volts was an ideal voltage for the chip and can also provide the 5 volts needed for the ultrasonic sensors, the flight controller, and the Raspberry Pi.

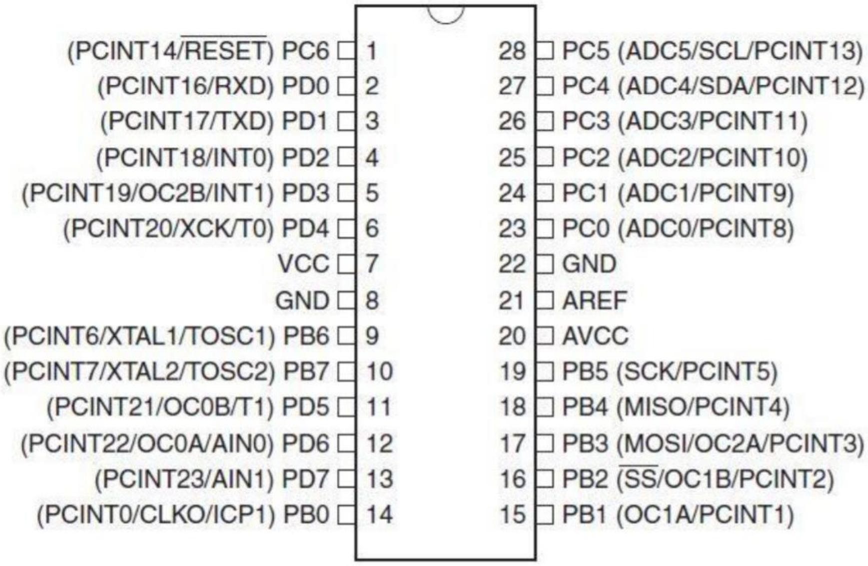


Figure 36: Atmega328 Pinout [91]

In the pinout diagram for the Atmega328, pin 1 is generally reserved at the reset pin and can be wired up to a pushbutton or signal that tells the chip to restart. Pins 2 through 6 are various digital pins with 2 and 3 being used for UART communication. Pin 7 is the voltage supply pin and should be connected to a 5 volt regulator that can be mounted on the board. Pins 8 and 22 are ground or common pins, which could be connected to a separate ground plate. Pins 9 and 10 are to be connected to a 16MHz crystal for the clock. Pins 10 through 19 are digital I/O pins that can be used for sensors with pins for programming. Pin 20 is the analog voltage supply, which should be externally connected to the Vcc even if the analog to digital converter is not being used.

If the analog power supply is not connected, it could result in unwanted power draw. Ideally, the analog voltage supply was connected to a passive low-pass filter to get rid of any unwanted noise, but might be unnecessary if the ADC is not in use. Pin 21 is the analog reference which is used to compare a voltage of a signal to and determines the values received in the analog pins. This pin is 5 volts by default and does not need to be connected to anything. Pins 23 through 28 are analog pins connected to an analog to digital converter. The ultrasonic sensors output digital data but can be connected to these pins with no difference in functionality from the digital I/O pins [92].

Voltage Regulators

Converting the 7.4 nominal voltage from the secondary battery to 5 volts for the board can be done using a number of different methods. An estimate of how much current was drawn should be made. The ultrasonic sensors draw about 15mA at 5 volts and the Atmega328p TQFP (surface mount) package will not draw more than 400mA at 5 volts. The Raspberry Pi has a max draw of 515 mA and the Pixhawk will draw 280mA with the telemetry module and other peripherals. This adds up to a total of current draw

1.27 amps. However, voltage spikes do occur, especially on startup, so this is an absolute minimum and there should be plenty of room for error.

There are two main types of voltage regulators, linear and switching. A linear regulator will drop to a lower voltage by dissipating the excess energy as heat. Switching regulators oscillate to achieve the desired voltage [93]. Switching regulators are more expensive but have better efficiency than linear regulators, this might not be a big problem because the the load is less than one watt. Heat is also not much of a concern for the same reason, so the main concern should be the quality of output power. The efficiency of suitable regulators should be analyzed and weighed against factors like size and price.

Linear voltage regulators can only converter a higher voltage to a lower voltage and are most efficient when the two voltages are closer and this regulator works better for applications with closer input and output voltages.To avoid the need for a heatsink a viable option would be a small Buck converter, which is a switching regulator that is able to take an input voltage and create a lower voltage with more current. These are more expensive but was more efficient and create less heat. At this point, the analog to digital converter is not being used so ripple voltage is not an issue. A switching regulator that meets all the requirements is the Recom R-78E-0.5. This converter is small, efficient, and relatively cheap. It supports input voltages from 7 to 28 volts and has an output voltage of 5 volts with a maximum current of 500mA, which is more than is necessary but not as much as many of the regulators [94].

In this low-power situation, a linear regulator might be a better option. The efficiency of a LDO regulator can be calculated using the following equation:

$$Efficiency = \frac{I_O V_O}{(I_O + I_Q)V_i} \times 100$$

Where I_O is the output current of 87mA, V_O is the output voltage which is 5 volts. I_q is the quiescent current, or the current draw when there is no load. A linear regulator like the LM1117 has an I_q of 5mA giving the efficiency calculated below.

$$Efficiency = \frac{1.27 \times 5}{(1.27+0.005)7.4} \times 100 = 67.3\%$$

Making the conversion from 7.4 volts to 5 volts requires that much of the energy is dissipated and the result for this particular regulator is 67.3% [97]. This means, 32.7% of the energy is being dissipated as heat. This is extremely inefficient. The battery chosen is however, able to provide the amount needed using this regulator for ten minutes This makes the final decision based on the amount of heat created by a linear regulator and the size and external components of a switching regulator.

Thermal resistance and maximum operating temperature is an issue when working with linear voltage regulators. If too much power is dissipated, the regulator may overheat and shut off. Most modern regulators have a temperature at which they shut off to avoid ruining the component. The LM1117 and LM7805 are similar regulators, but the LM7805 has a higher thermal resistance. The power dissipated, or P_D should be calculated to figure out how hot the regulator was getting. This calculation should be the worst case scenario, or the maximum amount of power being dissipated. This can be done using the formula below:

$$P_D = \ (V_{IN(MAX)} - V_{OUT(MIN)})I_{OUT} + (V_{IN(MAX)}I_{GND})$$

Where I_{GND} is the current going to ground or the quiescent current which is a maximum of 8mA for the lm7805 and the minimum output voltage for this converter is 4.9 volts. The maximum input voltage for a 2S battery is 8.4 volts when fully charged and the output current will still be 1.27A. Using the equation above, the maximum power dissipated is 4.36 watts [98]. Now the the maximum amount of power that was dissipated has been calculated, the value should be compared with the maximum amount of power that can be dissipated which can be done using the formula below:

$$P_{D,MAX} = \frac{T_{JMAX} - T_A}{\theta_{JA}}$$

The operating junction temperature of the converter is -40℃ to +125℃ meaning the converter should not reach more than 125℃, making T_{JMAX} = 125℃. T_A is the ambient temperature of the system, this could be different temperatures in different settings. The lower the ambient temperature, the bigger change in temperature can occur and therefore a higher amount of power can be dissipated. To be safe, an ambient temperature of 65°C can be used. This number is on the high side, but it is better to be safe than having a regulator that was shutting off and losing object avoidance capabilities.

For the lm7805 the thermal resistance from the junction to air is 65°C/W. Using these values gives a maximum possible amount of power. Using these values gives a maximum power dissipation of 0.923 watts. Using the same formula, the maximum ambient temperature the board should safely operate in is about 75°C which is 167°F so there could be a danger of overheating [99]. A general rule of thumb in the industry, however, is that for a TO-220 package three terminal linear regulator, which is the lm7805 package with the 65°C/W junction to air thermal resistance, less than 600mW dissipated does not need a heatsink.

To test the lm7805 regulator, a fuse was connected to the input of the regulator and a voltage of 12.6 volts was applied. A low Ohm resistor was connected from the output to ground to draw 400mA, less than the chosen components and any other 5 volt devices that will need to be powered such as the RaspberryPi. The circuit was left for a few minutes and the regulator got extremely hot and shut off.

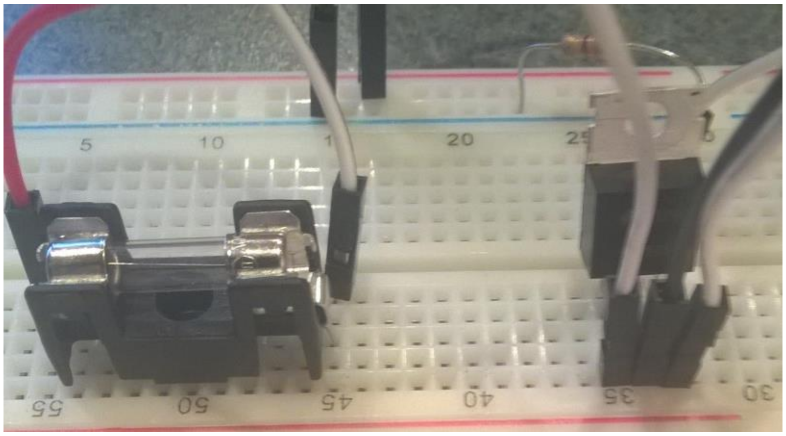


Figure 37: LM7805 Testing Circuit

Once the regulator proved to be working, the resistor was removed and the output of the LM7805 was connected to ground with a 1 Ohm resistor to simulate a short. It should be noted that the regulator did not shut off. The LM7805 got extremely hot, too hot to touch, and the resistor started smoking. This is an issue as the components connected to the LM7805 could be in danger in the occurrence of a fault.

It is recommended to add capacitors connecting the input to ground and the output to ground. These are called decoupling capacitors and have different functions for the input and output capacitors. For the input, the decoupling capacitor is to filter out any signals coming from the power supply or anything picked up along the way. A capacitor has practically infinite impedance for DC power, but an AC signal will see low impedance take the route to ground. The capacitor connected to the output to ground is there to keep the voltage at the desired level. If there are voltage spikes the capacitor will charge or discharge to reduce ripple [100]. The datasheet recommends a 0.33µF decoupling capacitor connected to the input and a 0.1µF decoupling capacitor connected to the output.

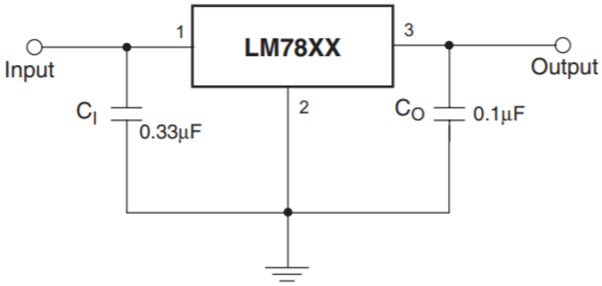


Figure 38: Application of LM7805 [101]

Because of the poor efficiency of the linear converter, the heat produced, and a need for an added heatsink, a simple switching regulator may be more suitable. The R-78E-0.5 is a good option, but requires external inductor to account for noise and also recommends using “soft-start” circuitry from the power supply to avoid the inrush of current to the capacitors damaging the converter. A similar package with simpler external circuitry is the OKI-78SR. This regulator already has decoupling capacitors and filters, it just requires an external fuse to avoid damaging the converter.

The recommended fuse in this circuit is a fast blow fuse with a rating of 2 amps. It is important not to choose a fuse too low or it will blow unnecessarily when the converter is not in danger of being damaged. This should not be an issue since the converter can handle much more than the 87mA load. The lm7805 and the OKI78SR have the same pinout diagrams, so both can be tested if noise from the OKI78SR is an issue. Capacitors can still be included so the lm7805 can still be used in place of the OKI-78SR. The OKI78SR datasheet recommends 10 to 22 μF for filtering the input and 10 to 47μF. This regulator is large and fairly expensive though. The best option may be to design a switching regulator than can be embedded into the PCB. This will save space and money but leaves room for errors and would be hard to replace.

A recommended tool for designing a regulator is TI’s WEBENCH. This is a simple program where the requirements are input and all the regulators that fit the requirements are shown. For this application, a fairly simple circuit is needed to create a regulator capable of 4 amps. The components required are five capacitors, two resistors, an inductor, and the TPS565201 regulator. This design seems to be the best option as it takes up very little space, can provide more than enough current, is cheap, and extremely efficient. The chart showing efficiency at different amperages is shown below.

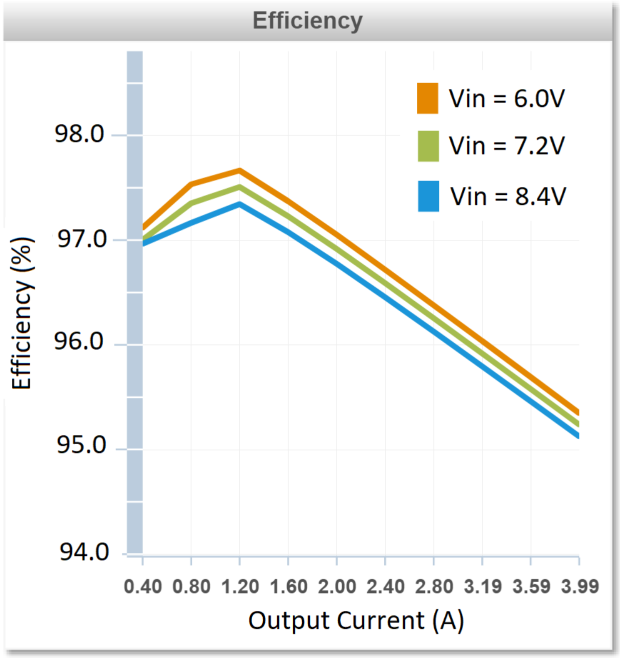


Figure 39: Efficiency of Designed Voltage Regulator [102]

Schematic and Board Layout

To include the switching regulator on the PCB, the schematic must first be made using all the desired parts. In this implementation, only surface mount components was used. This is where is was convenient to be using EAGLE as all parts were easily downloaded from TI or Sparkfun. The schematic can be seen below. This includes solder pads for all connection and and LED to indicate that the board is receiving power.

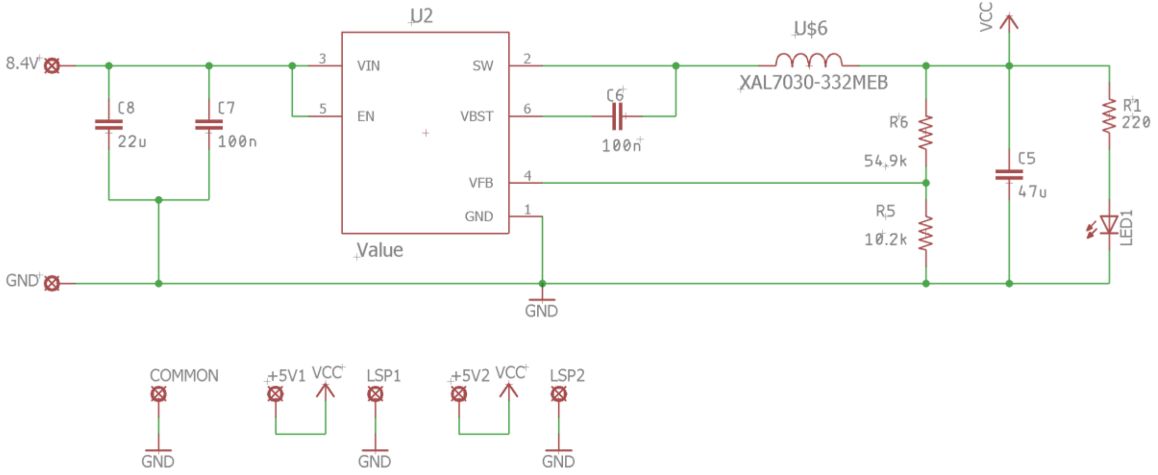


Figure 40: PCB Voltage Converter

The HCSR04 sensors have four pins each and five sensors was used. To manage the wires, a 20-pin male header can be soldered to the board and a four pin female-to-female wire connector with 0.1” spacing can be used to connect the sensors to the board. An in system programming header is included to program the Atmega and a header for sending the information to the Raspberry Pi. Two LEDs are connected to I/O pins to aid in troubleshooting. There are three decoupling capacitors going from Vcc to ground to filter the 5 volts and a 16MHz crystal to act as the clock for the MCU. A tactile switch is also connected to the reset pin.

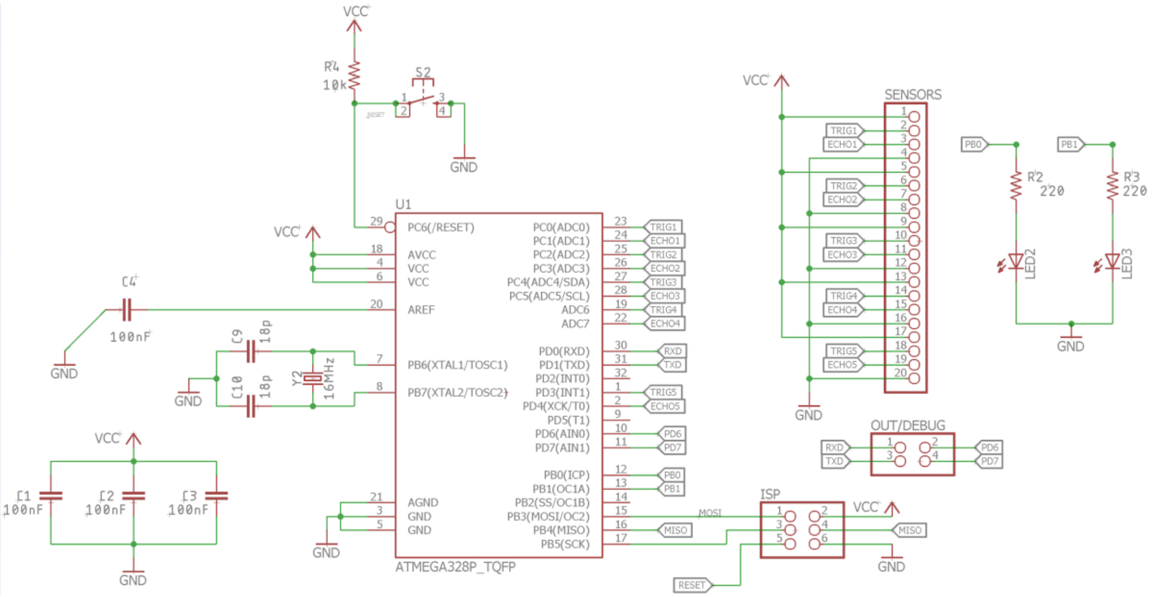


Figure 41: MCU Implementation Schematic

Once a satisfactory schematic has been made, a board view can be generated from the schematic. The board view starts out with all the components with the wires going directly to the next component. In EAGLE this is called the ratsnest because there are wires going everywhere and it is impossible to follow the diagram. The components should first be placed in their desired location, then the paths can be routed manually, or the software will autoroute. In this case, the traces were routed manually to get the shortest paths possible. Copper pours were also included for Vcc and ground to keep voltage drop to a minimum and to simplify routing.

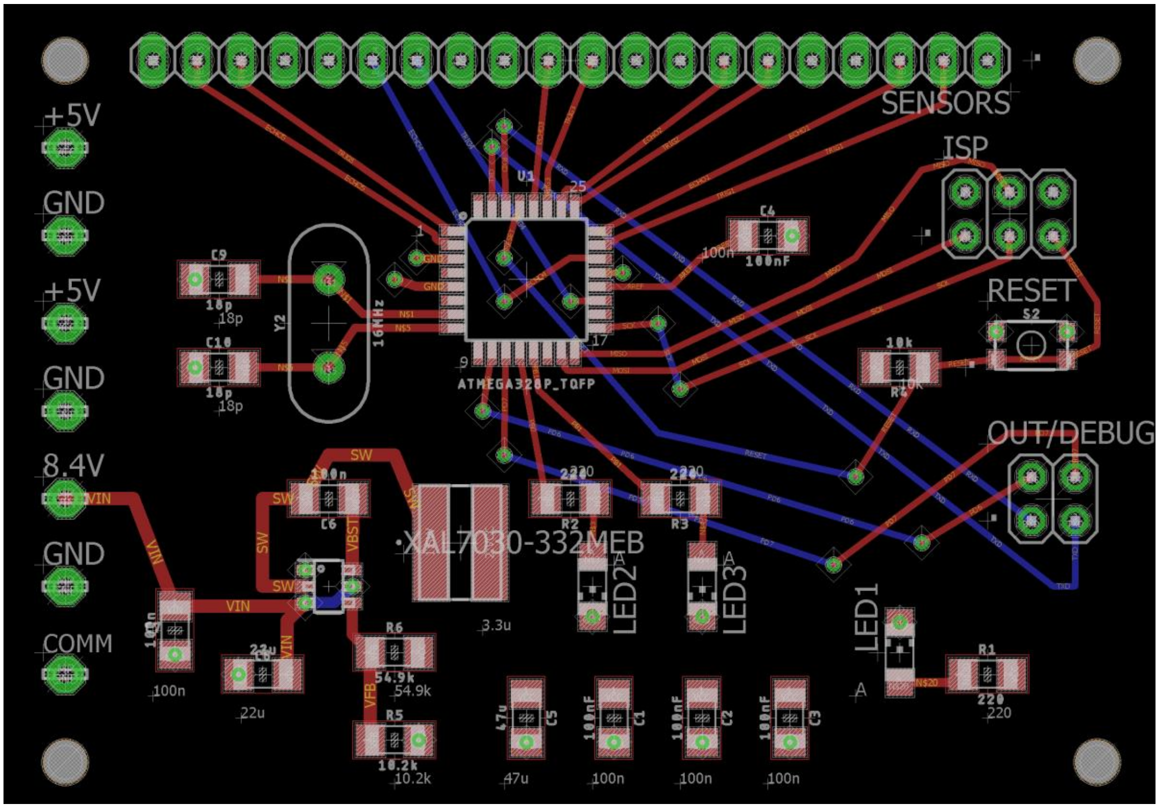


Figure 42: Final Board View

In most applications Gerber files must be generated and uploaded containing information about each layer. Since the chosen manufacturer was OSHPark, all that was required was the EAGLE board file. This made it much simpler to send to the sponsors to have them order the board. OSHPark was reasonably priced, quick to manufacture and ship, and came with three quality copies of the board. All the components were ordered from Mouser with most being surface mount packages. All capacitors, LEDs, and resistor were chosen to be 1206 because anything smaller might require a pick and place machine to solder to the board [103]. With the help of an electronics repair shop, all the surface mount components were soldered on.

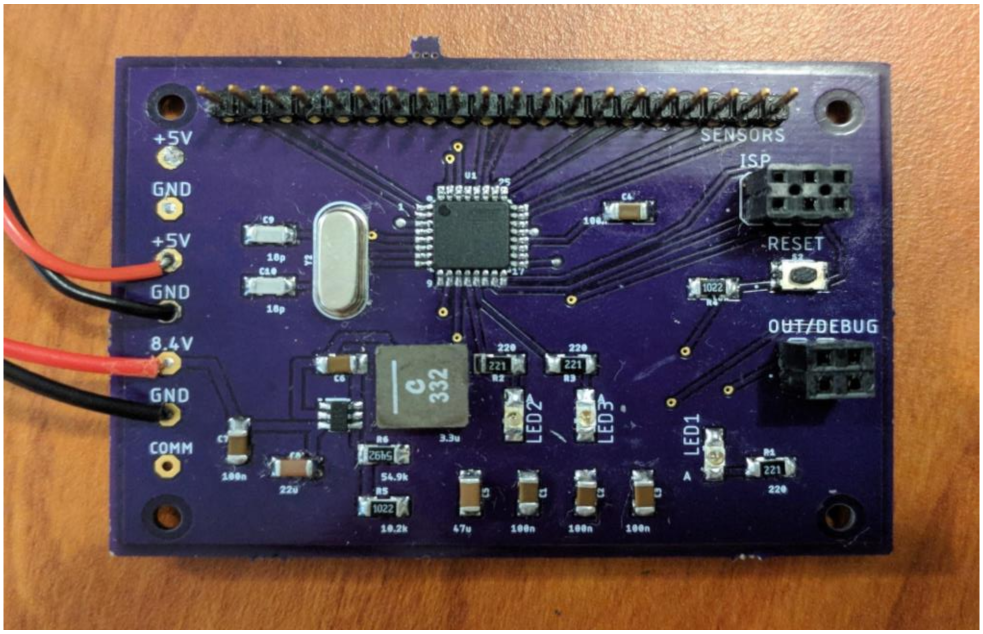


Figure 43: Manufactured PCB

3.10 Telemetry

As part of the competition guidelines, the drone may be guided to a designated area in order to make any necessary repairs or to replace a dying battery. This requires the drone to have a telemetry module to receive controls from a remote control transmitter rather than the controls generated by the autonomous system. This will not be a difficult task because the PixHawk flight controller supports most popular telemetry modules. The receiver modules generally do not weigh more than 60 grams and consume very little power. The required range for the module will not exceed 100 meters so that is also not a big concern. This leaves two main factors to look at when choosing a method of telemetry for this project: the cost of the transmitter/receiver combo and the reliability of the method. Telemetry can also include the transmission of data to the ground rather than just data to the flight controller. However, given the requirements to have FPV video with real-time tracking and identification information to a ground station, separating the two may be the best option.

RC transmitters generally have anywhere from two to 32 channels. The number of channels dictates the number of ways the transmitter can manipulate the aircraft's position by controlling motors speeds. A basic quadcopter requires at least four channels because the user needs control over throttle, rotation, forward and backwards translation, and side-to-side translation [104]. The transmitter will require an additional channel to switch from manual flight to autonomous flight. This means the controller will need an auxiliary switch to change channels. In addition the these four controls. The more channels that are included generally increases the price of of the transmitter with a 32-channel receiver being up to $500. In this application, manual controls was used very briefly and will not require advanced maneuvers. To keep the telemetry costs to a minimum, a five channel system was sufficient and eliminate unnecessary complications. It must be kept in mind that a 5 channel transmitter that has a switch to change flight modes only changes what the sticks on the RC transmitter do, not where the controls are being received. The transmitter can be modified to work with the autonomous code and switch modes using a 4 channel transmitter but the upgrade from four to five channels does not cost enough to jeopardize the reliability of switching modes.

Telemetry Protocols

There are several protocols that the receiver uses to send controls to the flight controller. The main methods supported by the PixHawk are receivers that use PPM, PCM, DSM, and a PPM-Sum receiver can be used to encode PWM controls to PPM controls because PWM is outdated and not supported by PixHawk [105]. PWM or pulse width modulation sends signal for each channel over its own separate wire. This allows fast transmission but limits the amount of channels supported. PPM or Pulse Position Modulation sends all the channels through the same wire allowing for more channels, this is similar to the updated version PCM or Pulse Code Modulation. DSM or Digital System Multiplexing used satellite to change transmission frequency in order to maintain the clearest signal [106]. More recently WiFi telemetry is being used, but there are no apparent advantages in this application.

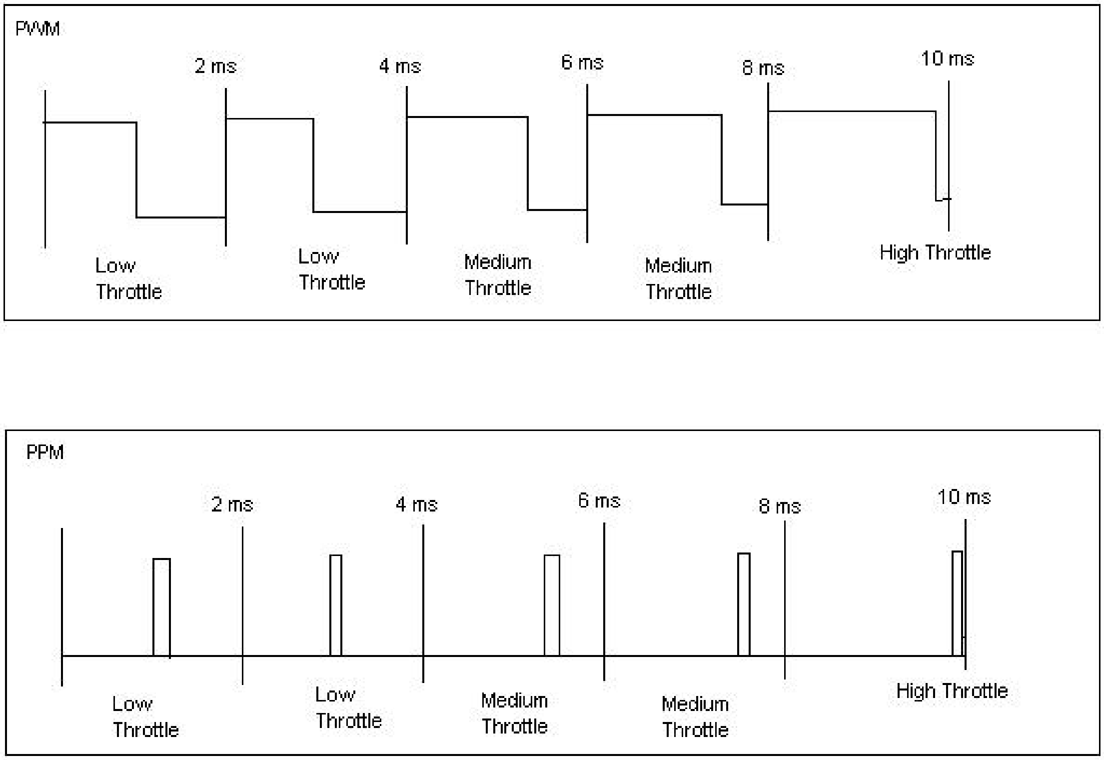


Figure 44: PWM versus PPM Protocols [107]

Receiver/Transmitter

The most widely protocol in receivers is PPM and PWM and was the most cost-effective and least complicated way to transmit manual controls. The Turnigy 5X Mini is a common, inexpensive option that comes with a receiver and suits the application well. However, this is a PWM receiver and will require an encoder to the PixHawk which is an additional $10 making this option $40. This is the case for many RC radio systems in this price range. For $50 a Turnigy 9X system can be purchased that includes an iA8 PPM receiver that can connect directly to the Pixhawk. This is a 2.4GHz option that comes with a higher quality, full-sized transmitter and is considered an industry standard with a large user base to help with troubleshooting and provide transmitter firmware and has great reviews. This transmitter has a GUI to select flight options and display information like battery level [108]. However, using such a basic transmitter could pose issues by not having the features required for this project. A popular transmitter that could have every feature needed is the Taranis Q X7. This transmitter can also record flight data and display it in real-time. A compatible receiver, the FrSky X8R, was used as well.

Figure 45: Taranis Q Transmitter [109]

3.11 PID Tuning

A multirotor drone, or any RC aircraft, is designed so that controls are entered to change the position of the vehicle, not maintain the position. This is achieved by using PID (Proportional Integral Derivative) controls in which the flight controller is being constantly updated to adjust its current state to match the desired state, or set point, which is the preset position to maintain stability. This is similar to how a car in cruise control applies more throttle when going uphill to counteract the extra force and match the desired speed. The system knows how much throttle to apply by calculating the amount of error in its current state, or the difference between current and desired state [110].

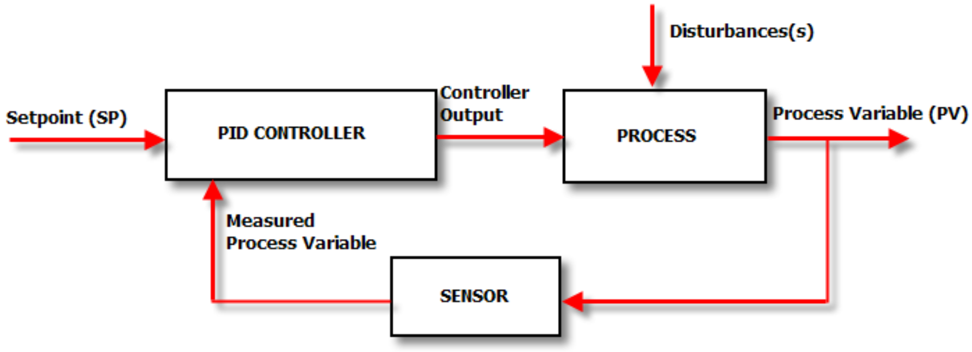


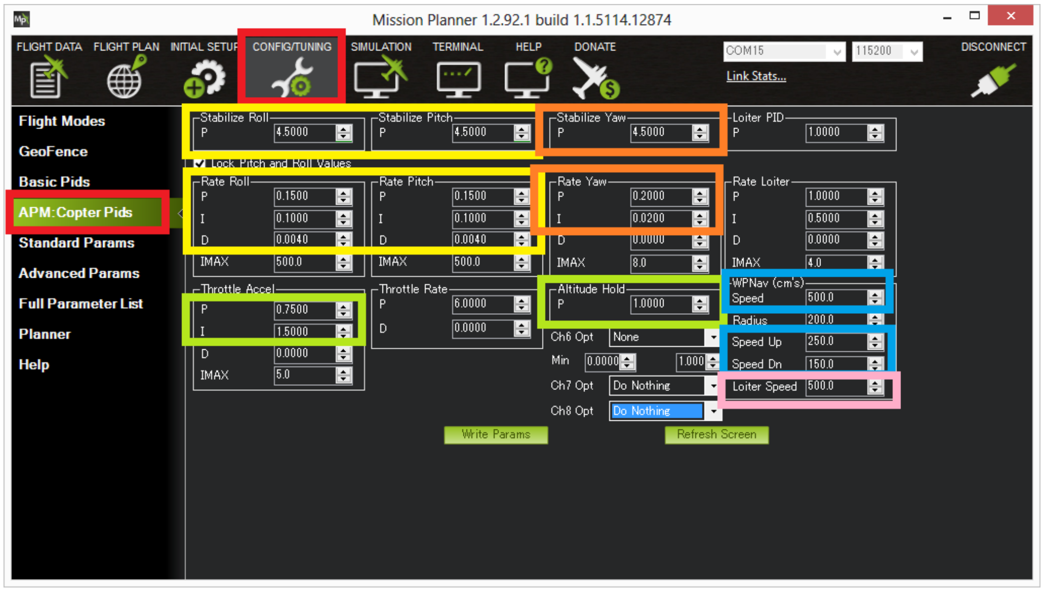
Figure 46: PID Control Loop [111]

PID tuning of a quadcopter is required for a heavily modified or newly created system and values containing information about size, weight, and thrust should be used for stability and accurate movement. These factors are very important in this application because in order for the drone to successfully identify and track prey, the camera must have a steady image and the quadcopter should be able to move quickly and accurately to strike the prey.

Obtaining PID Values

The proportional part of PID is how much the system reacts to error and is obtained by finding the ratio of error to the proportional band, or P-gain. Reacting too much to error will throw the drone off balance, create more error, and make stable flight impossible. The integral, or I-gain, determines how quickly the system resets itself to the set point. The amount of resets per minute is inversely proportional to the I-gain. For example, an I-gain of one will give 60 resets per minute. Derivative, or D-gain, determines how the system reacts to changes in movement or error. This helps the drone to anticipate disturbances and react accordingly to them depending on how fast the change in state is occurring.

Using Mission Planner, these values can be easily implemented to update the PixHawk and optimize flight by tuning control of throttle, roll, pitch, yaw, ect. Mission Planner has default values optimized for the 3DR Iris production drone and would work for a drone using the DJI Flame Wheel frame like this one, but tuning these values will increase performance. After the initial values have been fed into the program, the quadcopter should be fine-tuned through trial and error to account for unforeseen factors. This can be time-consuming but will greatly affect the how well the drone can hover and maneuver.



COM15 115200

Stabilize Roll: P 4.5000
Stabilize Pitch: P 4.5000
Stabilize Yaw: P 4.5000
Loiter PID: P 1.0000
☑ Lock Pitch and Roll Values
Rate Roll: P 0.1500, I 0.1000, D 0.0040, IMAX 500.0
Rate Pitch: P 0.1500, I 0.1000, D 0.0040, IMAX 500.0
Rate Yaw: P 0.2000, I 0.0200, D 0.0000, IMAX 8.0
Rate Loiter: P 1.0000, I 0.5000, D 0.0000, IMAX 4.0
Throttle Accel: P 0.7500, I 1.5000, D 0.0000, IMAX 5.0
Throttle Rate: P 6.0000, D 0.0000
Altitude Hold: P 1.0000
WPNav (cm's): Speed 500.0, Radius 200.0, Speed Up 250.0, Speed Dn 150.0, Loiter Speed 500.0
Ch6 Opt None
Min 0.0000 1.000
Ch7 Opt Do Nothing
Ch8 Opt Do Nothing

Figure 47: PID Tuning Page in Mission Planner [112]

To tune the PID values for the drone, it is suggested to start with a low value and slowly increase to avoid crashing. If the drone is symmetrical on the center axis, the values should remain the same for different controls. Setting P-gain involves analyzing how the drone rotates. A good P-gain will result in a responsive rotation with minimal prop wash. Prop wash is when thrust from one propeller interferes with the thrust of another and throws the drone off balance due to unexpected thrust. Prop wash comes from a P-gain that is too high because the drone is turning too fast. A low P-gain will result in a drone that responds to controls slowly and takes longer to rotate.

Turning the I-gain involves analyzing how the drone strafes left and right. The drone should hold a steady banking angle. Too high of an I-gain will cause the drone to flip over when strafing and too low of an I-gain will make it less maneuverable. The same approach should be taken, starting low and slowly increasing this value to avoid crashing the drone.

The D-gain is an important values that determines how much overshoot your drone has. This value is inversely proportional to the amount of overshoot which in counterintuitive. Too much D-gain and your drone is going to respond slowly to controls and too little will cause your drone to flip over. Most DIY drone builders want the most responsive drone possible without causing it to flip over. Tuning the drone for this value can be tricky because testing the limits of the D-gain will mean coming dangerously close to the threshold at which the drone flips or loses control [113].

Kalman Filters

Extended Kalman Filters (EKF) do not work directly with a PID control loop, but rather gives more accurate data from the drones sensors. Sensors often produce "noisy" data or readings with inaccurate outliers. A Kalman filter can be used to try and predict which readings are accurate and discard the ones that are not. This gives the flight controller a better idea of the drone’s velocity, acceleration, position, etc. An EKF for the object avoidance sensors would be necessary in order to not react to readings that might come from the cage surrounding the quadcopter.

An EKF can also provide more accurate readings by compensation for a constant error. For example, if altitude is constantly being overestimated by two feet, a Kalman filter can be used to implement the function of $actual\,altitude\ _{current} = altitude_{current} + noise$ to compensate for the inaccurate reading. Similarly, an EKF can implement a recursive function, or a function that uses a previous state value to calculate the current state. Once an equation has been defined, an EKF can be created using software like MATLAB.

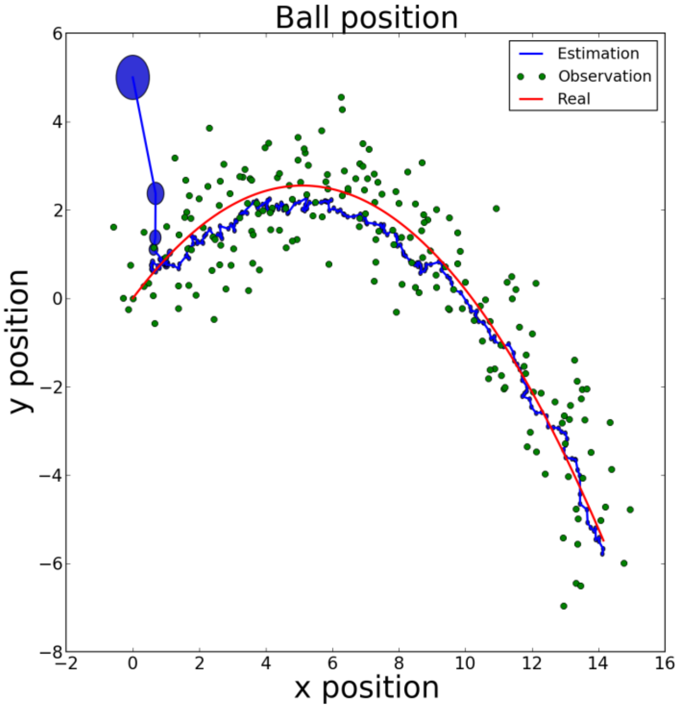


Figure 48: EKF Estimation from Observed Values [113]

3.12 Companion Computer

While our chosen flight controller, the Pixhawk, can easily handle both manual and planned autonomous missions given its memory and computing power, it is not made to handle any significant additional computations. This is especially true for wayfinding and computer vision algorithms. Therefore, an on-board companion computer is necessary, given that one of the project requirements is that all processing is done onboard of the vehicle. While it is possible to design one from scratch by buying the necessary processor, memory, etc.. and putting it all together in a PCB board, the time constraints for this project lead to the conclusion that a commercially available single-board computer was most appropriate. Vision algorithms, guidance, decision making, and transmission must take place in real-time. Computer vision tasks require significant processing power and must occur quickly. A high-powered processor allows the craft to perform with precision and respond to events quickly. This section considers various options that are available, and goes through the selection process for the chosen product.

Arduino Tian

The Arduino Tian is a popular Linux development board. “powered by Atmel’s SAMD21 MCU, featuring a 32-bit ARM Cortex M0+ and a Qualcomm Atheros AR9342, which is a highly integrated MIPS processor operating up to 533MHz and features an 802.11n 2.4 and 5GHz dual-band Wi-Fi module, IoT ready. The Tian also has the ability for users to switch off the Linux port from the MCU to reduce power and also use the board as a standalone device” [38].

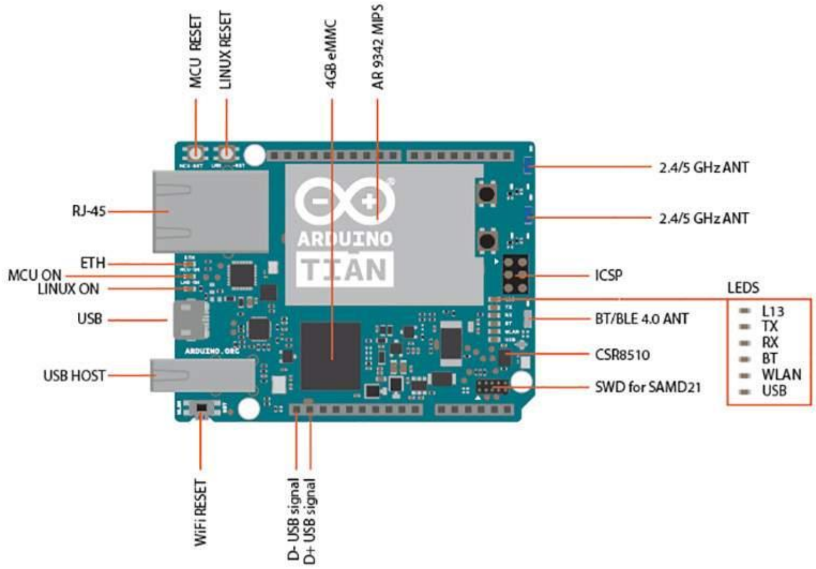


Figure 49: Arduino Tian Components and Ports [38]

The following is a list of the Arduino Tian’s technical specifications [39]:

- Microprocessor
    - Processor - Atheros AR9342
    - Architecture - MIPS
    - Operating Voltage - 3.3V
    - Flash Memory - 16MB + 4GB eMMC
    - RAM - 64MB DDR2
    - Clock Speed - 560 MHz
    - Wifi - 802.11 b/g/n 2.4 GHz dual-band
    - Ethernet - 802.3 10/100/1000 Mbit/s
    - USB - 2.0 Host
- Microcontroller
    - Microcontroller - SAMD21G18

  - Architecture - ARM Cortex-M0
  - Operating Voltage - 3.3V
  - Flash Memory - 256 KB
  - SRAM - 32 KB
  - Clock Speed - 48 MHz
  - Analog I/O Pins - 6
  - DC Current per I/O Pins - 7mA (I/O Pins)
- General
  - Input Voltage - 5V
  - PWM Output - 12
  - Power Consumption - 470mA
  - Size - 53 x 68.5 mm
  - Weight - 36g

While the Arduino Tian's ARM microcontroller supports Linux, it comes with its native operating system, ArduinoOS/LininoOS. Installing another operating system, based on the performed research, would require additional modification to both the Tian's hardware and software setup. Although the computer itself is very complete and definitely has enough power to withstand any amount of computational strain that this project might require, ease of first-time use is of great concern, as most of the project time was to be spent on the wayfinding and computer vision algorithms, as well as the overall component integration.

Raspberry Pi 3 (Model B)

The Raspberry Pi 3 is a small single-board computer. It features a QuadCore Broadcom BCM2837 SoC, which "includes four high-performance ARM Cortex-A53 processing cores running at 1.2GHz with 32kB Level 1 and 512kB Level 2 cache memory, a VideoCore IV graphics processor, and is linked to a 1GB LPDDR2 memory module on the rear of the board" [40]. Furthermore, it features a 40-pin GPIO header, 1 GB RAM, an HDMI port, as well as a USB chip to handle four USB 2 ports, among other things. The following is a list of features for this computer:

- Microprocessor
  - CPU - Quad-core 64-bit ARM Cortex A53 clocked at 1.2 GHz
  - GPU - 400MHz VideoCore IV multimedia
  - Memory - 1GB LPDDR2-900 SDRAM (i.e. 900MHz)
- General
  - USB ports - 4
  - Video outputs - HDMI, composite video (PAL and NTSC) via 3.5 mm jack
  - Network - 10/100Mbps Ethernet and 802.11n Wireless LAN
  - Peripherals - 17 GPIO plus specific functions, and HAT ID bus
  - Bluetooth - 4.1
  - Power source - 5 V via MicroUSB or GPIO header
  - Size - 85.60mm × 56.5mm
  - Weight - 31g

While the Raspberry Pi 3 might not be as complete or hardware-featured as some of its alternatives, it features a very interesting size vs weight vs power consumption ratio. Furthermore, the ARM Cortex A53 has well-documented support for various linux distributions including Debian-based linux distributions like Raspbian and Ubuntu MATE, as well as Arch Linux, Rise OS, OpenELEC and Pidora.

Figure 50: Raspberry Pi 3 Model B

NVIDIA Jetson TK1

The Jetson TK1 is a computation processor board. It features the Tegra K1 SOC which contains the NVIDIA Kepler GPU with 192 CUDA Cores and the NVIDIA 4-Plus-1 Quad-Core ARM Cortex-A15 CPU. It also provides support for GPIOs, UART, I2C, HSIC, SPI, and DP/LVDS. Below is a list of features available online for this computer [41].

- Microprocessor
    - NVIDIA 4-Plus-1™ Quad-Core ARM® Cortex™-A15 CPU
    - NVIDIA Kepler GPU with 192 CUDA Cores
- Input/Output
    - 2 GB x16 Memory with 64-bit Width
    - 16 GB 4.51 eMMC Memory
    - 1 Half Mini-PCIE Slot
    - 1 Full-Size SD/MMC Connector
    - 1 Full-Size HDMI Port
    - 1 USB 2.0 Port, Micro AB
    - 1 USB 3.0 Port, A
    - 1 RS232 Serial Port
    - 1 ALC5639 Realtek Audio Codec with Mic
    - In and Line Out
    - 1 RTL8111GS Realtek GigE LAN
    - 1 SATA Data Port
    - SPI 4 MByte Boot Flash
- Signals available through expansion port:

  - DP/LVDS
  - Touch SPI 1x4 + 1x1 CSI-2
  - GPIOs
  - UART
  - HSIC
  - I2C
- General
  - Weight - 907g
  - Size - 127 x 127 x 25.4 mm

The Jetson provides a fully functional NVIDIA CUDA platform for quickly developing and deploying compute-intensive systems for computer vision, robotics, medicine, and more. NVIDIA delivers the entire BSP and software stack, including CUDA, OpenGL 4.4, and Tegra-accelerated OpenCV. It comes with Ubuntu out of the box, which means software compatibility should be smooth. Overall, this is a very complete system, and it definitely provides the necessary hardware and software for this project’s computational needs. One drawback, however, is that it is more than four times heavier than the alternatives; on our copter design, this would have been a major drawback.

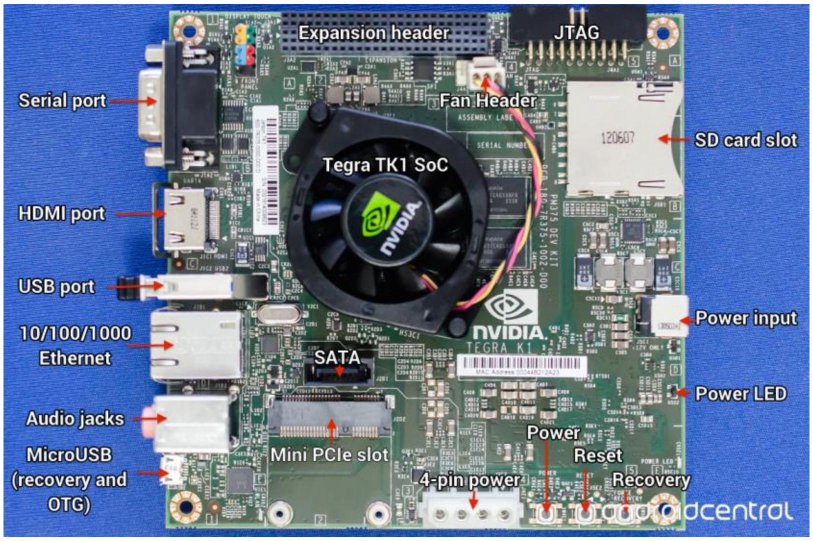


Figure 51: NVIDIA Jetson TK1 [42]

ODROID-XU4

The ODROID-XU4 is a 32-bit ARM single-board computer created by Hardkernel. It is one of the most powerful low-cost Single Board computers available, and an extremely versatile device. “Offering open source support, the board can run various flavors of Linux, including the latest Ubuntu 16.04 and Android 4.4 KitKat, 5.0 Lollipop and 7.1 Nougat. By implementing the eMMC 5.0, USB 3.0 and Gigabit Ethernet interfaces, the ODROID-XU4 boasts amazing data transfer speeds, a feature that is increasingly required to support advanced processing power on ARM devices” [59].

This computer features the Samsung Exynos5422 Cortex System-on-Chip (SoC), which includes both the ARM Cortex-A7 and the ARM Cortex-A15. This is the Octa Core SoC found on modern mobile phones such as the Samsung Galaxy S5. It also includes a 30-pin GPIO header supporting I2C, SPI, UART, and ADC, as well as an additional 12-pin GPIO header supporting I2C and I2S. It also includes two USB 3.0 ports, which might be very useful for compatibility purposes with other peripherals such as the Intel RealSense depth camera. Furthermore, it contains a serial console port, which could potentially be used as a telemetry port. Below is the list of theODROID-XU4's most important components.

- Microprocessor - Samsung Exynos5422 SoC
    - ARM Cortex-A7
    - ARM Cortex-A15
- Multimedia
    - GPU
        - ARM Mali – T628 MP6 (600 MHz)
        - OpenCL 1.1 Full profile
        - OpenGL ES 1.1, 2.0, and 3.0
    - JPEG
        - Enc/Dec
    - MFC
        - 1080p 60 Enc/Dec
- Ports:
    - Serial Console Port
        - I2C
        - SPI
        - I2S
        - UART
        - ADC
    - Ethernet RJ-45 Jack
    - MicroSD
    - HDMI Type A
    - USB 2.0 Host
    - 2 x USB 3.0
    - 30-pin GPIO header
        - I2C
        - SPI
        - UART
        - ADC
    - 12-pin GPIO header
        - I2C
        - I2S
- General
    - 2GB RAM
    - 5V/4A DC input

- ○ 3.2 x 2.3 x 0.9 inches
- ○ Weight - 198.446662 g

Overall, this was a very complete and appealing computer alternative. It is light, has more processing power than the RPi family, and has a price that is very accommodating to the available budget. Although integration and compatibility may be a concern before the testing phase, this is a computer that was definitely considered, and actually acquired, in the decision process.

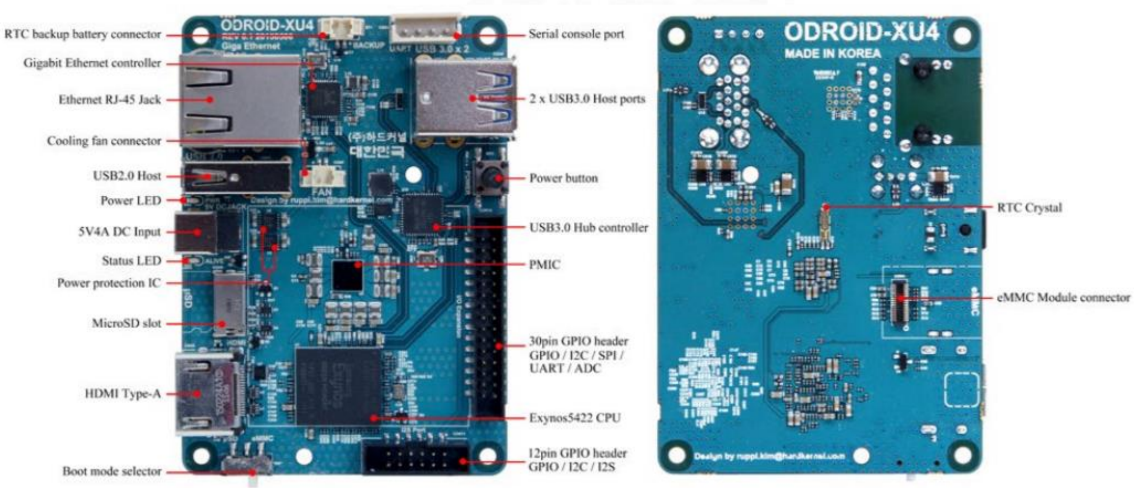


Figure 52: XU4 Block Diagram and Annotated Board Image [59]

The following table shows the major considerations for the selection of the RoboCopter’s onboard computer. Given that the expected frame size of our aircraft is approximately 450mm (in diameter), and that the motors will occupy a sizeable portion of our calculated maximum weight, size and weight are two major criteria for the selection decision. Given that the specific algorithms for prey detection have not been decided on, computational power is hard to calculate. However, a computer that supports a unix-based operating system was absolutely necessary, as this will significantly simplify the integration process with the other devices onboard. The ability to install ROS is also vital, as we will consider on the Flight Simulation section.

	Arduino Tian	**Raspberry Pi 3B**	**NVIDIA Jetson TK1**	**ODROID-XU4**
Well-Documented Pixhawk Support	✗	✓	✗	✓
Size (mm)	53 x 68.5	85.60 × 56.5	127 x 127 x 25.4	83 x 58 x 20
Weight (g)	36	42	907.19	38
Processor	Atheros AR9342, ARM Cortex-A0	Quad-Core ARM Cortex-A53	Quad-Core ARM Cortex-A15	ARM Cortex-A15 ARM Cortex-A7
Cost	$95.70	$34.50	$199.99	$67.45
Linux + ROS	✗	✓	✓	✓
DroneKit Documentation	✗	✓	✓	✓

Table 8: On-board Computer Selection Criteria

As can be seen on the table above, the selected computer at the moment was the Raspberry Pi 3 Model B due to its cost, weight, size, and performance. Furthermore, its compatibility with Unix-based OSes, the Pixhawk 2.8.1, and most common cameras ensured that component integration was as simple as possible. In case of the need for additional ports or interfaces, the Raspberry Pi Compute Module 3 was also an option, as it provided all the benefits of the Raspberry Pi 3 in a more flexible form factor, allowing for additional hardware configuration. These decisions were made on the integration and testing phase of this project, when all hardware was acquired. Either way, the computing characteristics of the Raspberry Pi 3 were deemed sufficient for the processing requirements of the RoboCopter. The pathfinding algorithms and flight decisions are primarily made on this computer, which in turn sends flight commands to the selected flight controller through the easy-to-use and supported MAVLINK communication protocol.

Although the Raspberry Pi 3 was the current selected on-board computer due in part to its availability, the complexity of the pathfinding and vision system algorithms made it necessary to select a computer with higher processing power. With the current research, the ODROID-XU4 appeared to be the most appropriate alternative. The specific computational requirements, however, were decided during the testing phase of this project, as they depend in part on the future work to be done during integration and software/algorithm analysis and implementation. Initially, testing was done with the Raspberry Pi 3; however, the ODROID XU4 was acquired for testing purposes as well. After testing, it was apparent that the Raspberry Pi's widely supported nature allowed for faster processing of the prey detection algorithms; hence, the Pi was ultimately selected in this project. In retrospect, the Jetson may have been a better alternative in order to fulfill all processing power requirements, while still providing the CUDA to allow for very fast image detection.

3.13 Flight Simulation

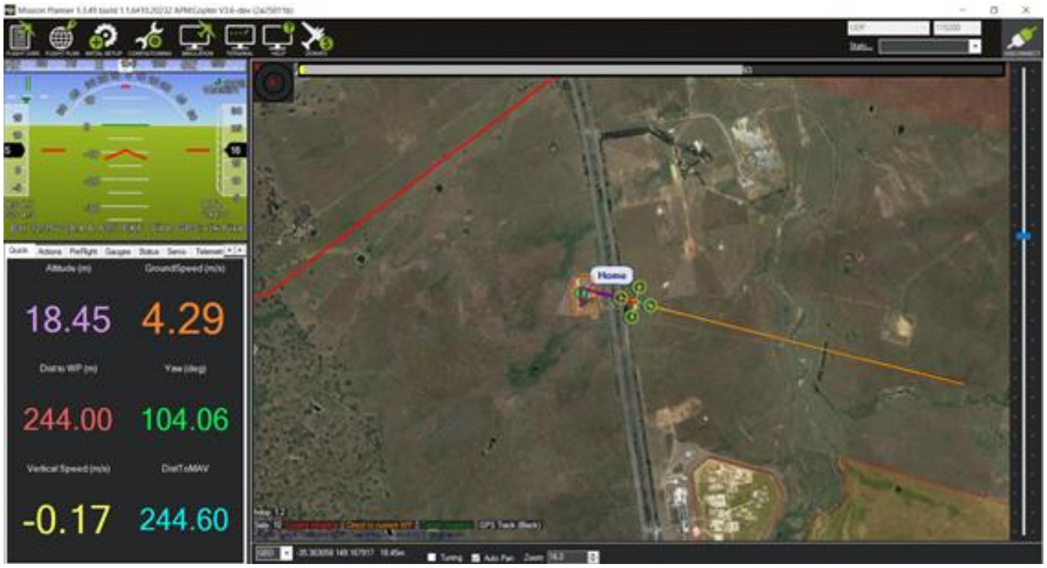
Given the wide number of available solutions for each component of the RoboCopter, it is understandable that physical component selection was one of the longest tasks in the design process. Because of this, flight simulators were extremely important to our cause, as they allowed for the testing of flight plans with varying components and environments. Best of all, they allowed the testing of flight algorithms without the need to have the vehicle physically completed. This section considers the different types of flight simulators that are available for our chosen flight controller, our on-board companion computer, and the different software components that our copter employed. It also covers the simulations that were run by the time that this report was written.

SITL

Software in the Loop (SITL) is a simulator supported by the ArduPilot project which allows the simulation of ArduPilot's Plane, Copter, or Rover instances without the need for any special hardware. It can be run directly on a PC, as it considers it simply another platform that ArduPilot can be built and run on. This takes advantage of the fact that ArduPilot is portable and can run on a wide variety of platforms. Simulations can take place on a Windows, Mac, or Linux OSes, and can even be run using a Virtual Machine (VM). SITL can simulate multiple types of aircraft, camera gimbals, antenna trackers, and even optional sensors such as LIDARs and optical flow sensors.

At the initial stage of this project, a basic takeoff and land script was simulated using SITL. The setup was the following:

- Host Computer: Windows 10.
  - Mission Planner connected to UDP port 14550.
  - Ubuntu 16.04 Guest VM running on VMWare Player.
    - SITL for copter script running
    - MavProxy running (can send commands to simulated copter)
      - Output to localhost port 14551 added
      - Output to host IP added on port 14550
  - Flight script running, connected to localhost UDP port 14551



Altitude (m): 18.45
GroundSpeed (m/s): 4.29
Dist to WP (m): 244.00
Yaw (deg): 104.06
Vertical Speed (m/s): -0.17
DistToMAV: 244.60

Figure 53: SITL Simulation

As can be seen in the picture above, the simulation with the script runs successfully. This allowed for the future structured testing of pathfinding algorithms that will need to be simulated during the testing and integration phase of the project. Furthermore, additional expected features such as a camera gimbal and LIDAR sensors can be implemented for more accurate results. This was the main simulation environment used for navigation.

Gazebo

Gazebo, like SITL, is a Robot Simulator provided by the OPen Source Robotics Foundation. It was designed to host an environment that developers can use to simulate life like conditions for the development of robot systems. Utilizing Gazebo will allow us to test guidance, navigation, control, vision algorithms, image processing, feature detection, and integration without any potential physical harm to our Robocopter UAV.

Gazebo runs on major Linux distributions as well as Mac. It boasts great integration with ROS using a set of ROS packages named gazebo_ros_pkgs. These packages provide necessary wrappers and interfaces to simulate a root in Gazebo using ROS message passing. Thanks to integration with ROS, Gazebo can run the same or similar code that we would be using on our physical Robocopter UAV, especially if compatible with the DroneKit framework.

Furthermore, Gazebo features dynamic simulations using up to four high performance physics engines while utilizing OGRE to provide realistic renderings. It can also generate sensor data and, with the use of gazebo_plugins, process the data using ROS. A graphical model editor can also be used to more accurately and efficiently simulate our operating environment.

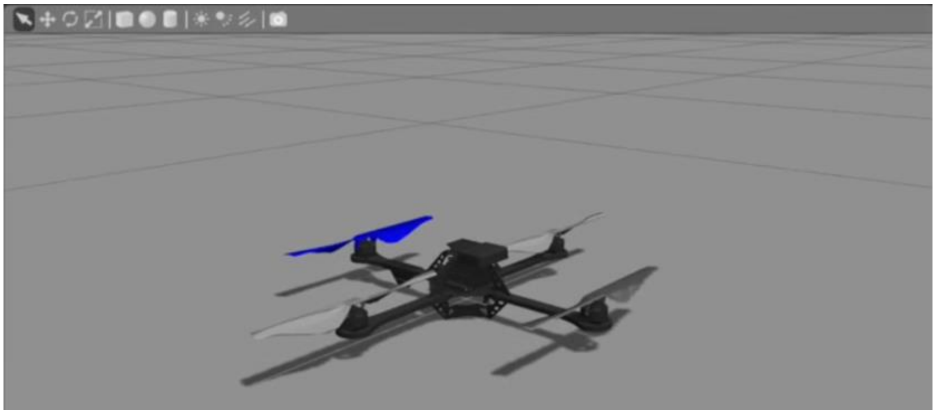


Figure 54: Gazebo Sample Quadcopter Model

DIYDrones developed ardupilot_sitl_gazebo_plugin, an ArduPilot plugin for Gazebo. Use of Ardupilot ROS and Gazebo SITL allows for improved simulation stability and allows for rapid testing of various combinations of sensor payloads and flight modes without hardware. This also allows proper function validation and debugging in real-time. Through the integration with ROS we are also able to use the package mavros to simulate sending and receiving mavlink packets. Mavlink packets can used as a communication backbone for the IMU and Microcontroller communication as well as for Linux interprocess and ground link communication. Gazebo was extremely important, although difficult to work with at times, in the simulation process. It provided very accurate and clear representations of the drone’s physical environment, as well as its missions.

4. Standards and Design Constraints

4.1 Related Standards

Standards are extremely important because they provide consistency among common products that ensure the safety of the device itself, other objects in the vicinity of the device during operation, and humans and/or animals that may be affected by the use of the device. Standards influence engineers when they are making decisions during the development phase of the product. They allow the engineer to take a directed approach instead of making educated guesses.

Standards are developed by groups of engineers who have the knowledge and the experience to make these types of decisions. Whether it’s the Institute of Electrical and Electronics Engineers (IEEE), the Federal Aviation Administration (FAA), or the International Organization for Standardization (ISO), credibility of the information is assured because the group is made of engineers who have years worth of experience in dealing with this information.

4.1.1 AC 91-57A - Model Aircraft Operating Standards

This advisory council is written to guide persons flying Unmanned Aerial Systems (UAS) for a hobby or recreation purposes. According to 14 CFR § 1.1 an aircraft is described as “a device that is used or intended to be used for flight in the air.” If the device meets the conditions for a “model aircraft” set in the Section 336 of Public Law 112-95, the *FAA Modernization and Reform Act* of 2012 then this standard applies. The following rules define a “model aircraft”:

1) The aircraft is flown strictly for hobby or recreational use;

2) The aircraft operates in accordance with a community-based set of safety guidelines and within the programming of a nationwide community-based organization (CBO)

3) The aircraft is limited to not more than 55 pounds, unless otherwise certified through a design, construction, inspection, flight test, and operational safety program administered by a CBO;

4) The aircraft operates in a manner that does not interfere with, and gives way to, any manned aircraft; and

5) When flown within 5 miles of an airport, the operator of the model aircraft provides the airport operator or the airport air traffic control tower (when an air traffic facility is located at the airport) with prior notice of the operation. Model aircraft operators flying from a permanent location within 5 miles of an airport should establish a mutually agreed upon operating

procedure with the airport operator and the airport air traffic control tower (when an air traffic facility is located at the airport).

The aircraft for this project satisfies rule 1, 3, and 4 so the operation of the drone needs to adhere to this standard.

This document is here to ensure the health and safety of others within the vicinity of the drone's operating environment. The document explains that the drone shall not be flown in restricted airspaces, areas with Temporary Flight Restrictions (TFR), and areas around aerial shows or sporting events. It also requests that operators of these vehicles avoid flying around the presence of a manned aircraft vehicle and that the aircraft should not be flown 400 ft above level ground. The operating requirements outlined in this document does not limit in anyway the requirements for this project. The drone that was built should easily adhere to all of these requirements.

Aircraft operations that endanger the National Airspace System (NAS) fail to comply with the rules set within this document may be subject to Federal Aviation Administration (FAA) enforcement actions.

4.1.2 IEEE Approved Draft Standard for Sensor Performance Parameter Definitions

This standard applies to this project's use of proximity sensors for object detection and object avoidance, and other sensors that help the drone stay stable during flight. IEEE describes this standard as follows:

A common framework for sensor performance specification terminology, units, conditions and limits is provided. Specifically, the accelerometer, magnetometer, gyrometer/gyroscope, accelerometer/magnetometer/gyroscope combination sensors, barometer/pressure sensors, hygrometer/humidity sensors, temperature sensors, light sensors (ambient and RGB), and proximity sensors are discussed.

4.1.4 IPC-221B Generic Standard on Printed Board Design

The IPC is an association who aims to standardize the assembly and production requirements of electronic equipment and assemblies. This standard refers to the design of our printed circuit board. The requirements contained in this document establishes principles and recommendations that must be followed for the design of the board. It explains the requirements for printed boards, whether they are single-sided, double-sided, or has multiple layers. The design manufacturer that we select should adhere to these standards. By following the requirements outlined by this standard, the printed boards that are manufactured should be able to last longer and have better overall performance.

4.2 Design Constraints

The standards described above affect the overall project's design in numerous ways. The standards also have affected the requirements of the project as well. While designing the aerial vehicle the group had to take into consideration factors that affect the design and implementation of this project. Some of those factors include economic, environmental, safety, manufacturing, etc. The following sections will highlight some of the design constraints that affected this project:

4.2.1 Economic Constraints

Before deciding on any hardware components and software to be used for this project we first had to the economic constraints. The budget for this project was given by our sponsor, Lockheed Martin. The total budget to be used for this project is $2,000. Out of that $2,000, only $1,500 can be used for the as-demonstrated cost for this project. The remaining $500 can be used for spare parts, manufacturing tools, testing stations, and other items that are deemed necessary for the success of this project.

To overcome the economic constraints, we have to consider the component's performance and importance versus the price and quality of the device. Whether we are considering the propellers, motors, flight controller, camera, proximity sensors or any other component we have to decide which component is more important to the goals of this project and thus which component we should decide to spend more money on. For example, one of the components we have considered is the TeraRanger One Type B time of flight sensor for object avoidance. One of these sensors costs $133 and can only handle one side of the drone at a time, whether it's the front, back, bottom, left, or right side. The manufacturer who designs this sensor sells another product that incorporates more than one of these sensors into one product so that it can see all the necessary sides of a drone but that product costs over $1,000 and is definitely too much to spend on proximity sensors for this project.

The total cost of our design continued to change as we make progress through this project. As we continue to research and learn more about the components and evaluate the importance of each component that we are considering, the money allotted for each will change. With the help of everyone involved in our Green Team group and the Lockheed Martin representatives, we will make design decisions that meet the goals of this project.

4.2.2 Time Constraints

The time to complete this project is consistent with the time allotted for the Fall and Spring semesters by the University of Central Florida. The Fall semester was dedicated to research and planning of the project while the Spring Semester was dedicated to implementation, integration, and testing of the device that was used for the competition.

The time frame for this project is short but at the end of this project we expect to have a fully functioning product that meets the goals set forth by our sponsor. Some of the design decisions may be affected by the lack of time. The time frame to build and test the device will also cause concerns throughout the project. Any additional features that do not positively affect the primary goals will have to be neglected.

Another time constraint involves the availability of the members of the team. Many of the members was taking full time credits, working or doing an internship or co-op, and being involved in other extracurricular activities. To make a lot of the design decisions, input from multiple members that have researched the respective portion of this project was necessary. With different schedules involved it was difficult to make decisions in a timely fashion.

4.2.3 Environmental, Social, and Political Constraints

For this project, there are not many environmental, social, or political constraints that was of concern. The project will result in a small scale product that doesn’t emit any hazardous wastes. The device will consist of electrical components, such as batteries, that need proper disposal. If improperly disposed, batteries can cause environmental issues but we do not believe that was a significant issue.

4.2.4 Ethical, Health, and Safety Constraints

In regards to ethical constraints, there are no ethical constraints that we feel will negatively impact this project. The research and physical work that needs to be done was well documented with proper citations and credit given to the creators of the information.

As with all engineering projects that involve devices being used on or around humans and/or animals, this project will take safety during operation very seriously. The drone was unmanned and autonomous so decisions that are made by the software will have to ensure the safety of the drone itself and potential bystanders. The weight of the drone should not mitigate some of the safety issues as well as the protective cage that was designed. Precautions was taken to protect the blades of the propellor, the onboard sensors, and any other components that have the capability of falling off the drone during flight. All components should be secured to mitigate that risk but because of the impacts that have to be made between our drone and the prey drones some objects may become loose.

In regards to health constraints, besides the safety constraints described above there should not be any issues with health. The drone will not be manned by a person so there is little risk of a person operating the device to be harmed. The device is also not to be operated on humans or animals so there are no biological factors to be concerned with. As described above, the software shall take care of the drone’s operations and ensure the drone operates in the way it is supposed to.

4.2.5 Manufacturability and Sustainability Constraints

This project was impacted significantly by manufacturability and sustainability constraints. The project consists of multiple rounds so the design should be able to be perform as expected during all rounds of the competition. Components such as the battery may die out after each round so having a backup battery is something that has been considered. The battery to be used was determined based on a variety of factors but will have to withstand the time allotted for each round in the competition.

Also, the wear and tear on the materials used to construct the body and caging was a concern. The materials will have to be able to withstand multiple impacts, possibly in the same location on the drone. The materials to be used was well researched and designed primarily by the Mechanical and Aerospace students. Matching parts will also need to be manufactured nearly identical to each other. They shall be tested to verify they perform within a certain tolerance of the spec that they are designed to be working at. Also, the other components, such as sensors, camera, propellers, motors, etc. may have to be replaced or reattached if they were to become loose during the round so spare parts should be considered during the construction of this drone. Components shall be compatible for ease of access upon installation as well as for maintenance. The system layout shall be done in a manner that makes design changes possible. Maintainability shall be heavily considered when making initial design decisions.

4.2.6 Course Constraints

A description of the course can be found in the Project Overview section of this document. As previously discussed, the outer dimensions of course are 40 feet by 40 feet and the drone will start in the corners of the course, with prey starting at the diagonal corner. The constraints set by the outer boundaries of the course and the obstacles that are positioned throughout the course will cause issues when designing and implementing the necessary software.

The setup of the course will impact the drone during operation. Possible moving targets can cause the drone to react in different ways. The drone needs to be able to distinguish between moving objects and recognize these objects as either a decoy that should be avoided or a prey drone that should be attacked. The design of the software will need to be carefully constructed and was an issue especially in addition to the timing constraints that we have previously described.

5. System Design

This RoboCopter design, as mentioned in the Executive Summary, is a joint effort between 13 engineering students pursuing different majors. Hence, one of the first major milestones for this project was to develop a set of high-level system, functional, software and hardware diagrams which allowed for further research, analysis, and cooperation. This section contains the various facets of the design process, and will delve through the many decisions that were made in order to create a working product. These decisions, as can be inferred, are heavily based on the information recorded in the Trade Studies and Research section.

5.1 Functional & Component Decomposition

In order to justify the necessity and identify the requirements of each component, the functionalities of each one should by analyzed. This will make determining the component best suited for the needs of the drone easier when finalizing decisions and purchasing the components. Having a functional decomposition of all the components will also ensure that all team members are informed on the desired functions of each component.

Camera

The camera or optical sensor served as the drones primary sensor. The desired function of the camera is to collect data about the position of prey drones in relation to the predator drone’s current location. The camera will also help with avoiding obstacles, making a map of the boundary lines, and returning to the search position while is reset mode.

Secondary Sensor

The secondary sensor provides supplemental information about the position of the drone. This component mainly gave information about the current altitude and any nearby obstacles. This can be accomplished by using a number of sensors. Ultrasonic range finders met these requirements because they are cheap and lightweight so multiple range finders can be placed on the drone to achieve better angles for more accurate feedback.

Flight Controller

The flight controller received manuals controls from the RC transmitter or automated controls generated from the autonomous flight algorithms on the microcontroller. The flight controller takes these controls and outputs data to the electronic speed controllers to maneuver the drone. The flight controller contains an accelerometer and gyroscope for data about the drones current orientation and movement.

Companion Computer

The purpose of the companion computer was to perform all computational activities that relate to pathfinding and, possibly, visual input processing. The companion computer, as covered in the Companion Computer part of the Trade Studies and Down Selection section, needed to have the computational capability to perform all necessary processing, and send low-level directional commands to the flight controller using the MavLink protocol. This allowed a modular approach that takes advantage of the function-specific hardware nature of both the flight controller and the on-board computer.

Printed Circuit Board

This was designed around a microcontroller that received data from the camera and sensors and contained the algorithm that turned this data into flight controls for the FCU. The board has a voltage input pin to power and voltage output pins to power the microcontroller, camera, and sensors.

Battery and Power Distribution Board

A single lithium polymer battery is used to power all components. A power distribution board is needed to step down the voltage from the battery to power the lower voltage components. The power distribution board may exist as a single voltage regulator or may not be necessary depending on the allowed input voltage of the flight controller and if the flight controller has the necessary output voltages for the PCB.

Electronic Speed Controller (ESC)

An electronic speed controller is generally required for each motor. The speed controller connects directly to the battery and regulates the amount of amps reaching the motors based on the controls given by the FCU. Speed controllers can also come with multiple outputs, which are generally suited for smaller drones because weight distribution is a big concern and a modular ESC can be placed in the center. A speed controller must be rated well within the max amount of amps drawn by the motors to avoid overheating and ensure maximum efficiency.

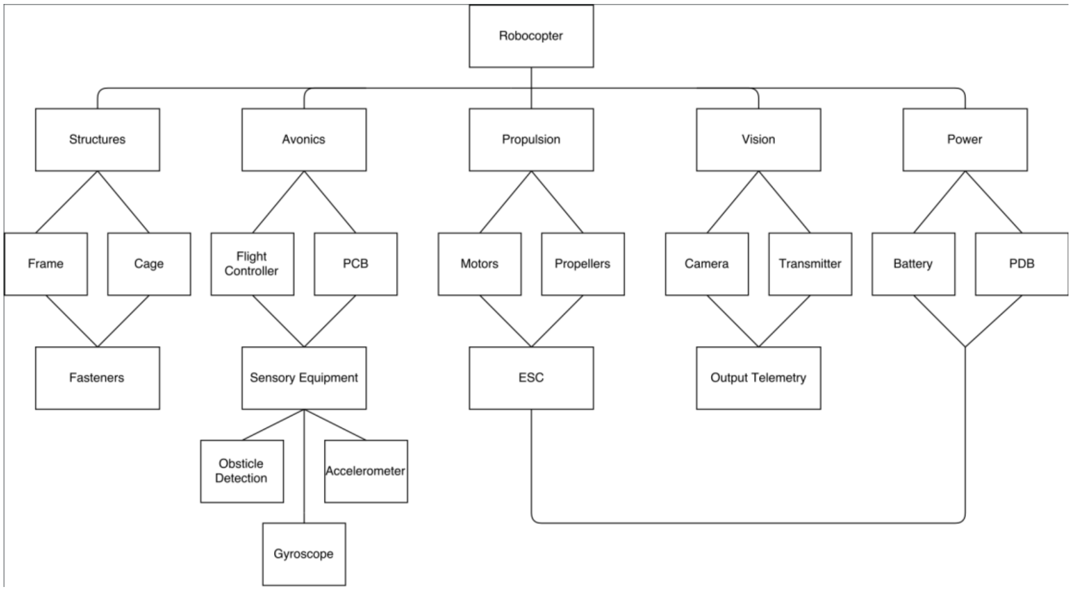


Figure 55: Component Decomposition

Our RoboCopter, as directed by the requirements, must have one or more sensors which update its percept at regular intervals, such that the companion computer can generate instructions for the flight controller to follow. The flight controller, in turn, controls the physical guidance of the copter. Below is a functional decomposition flowchart of the different logical and functional steps that our design follows:

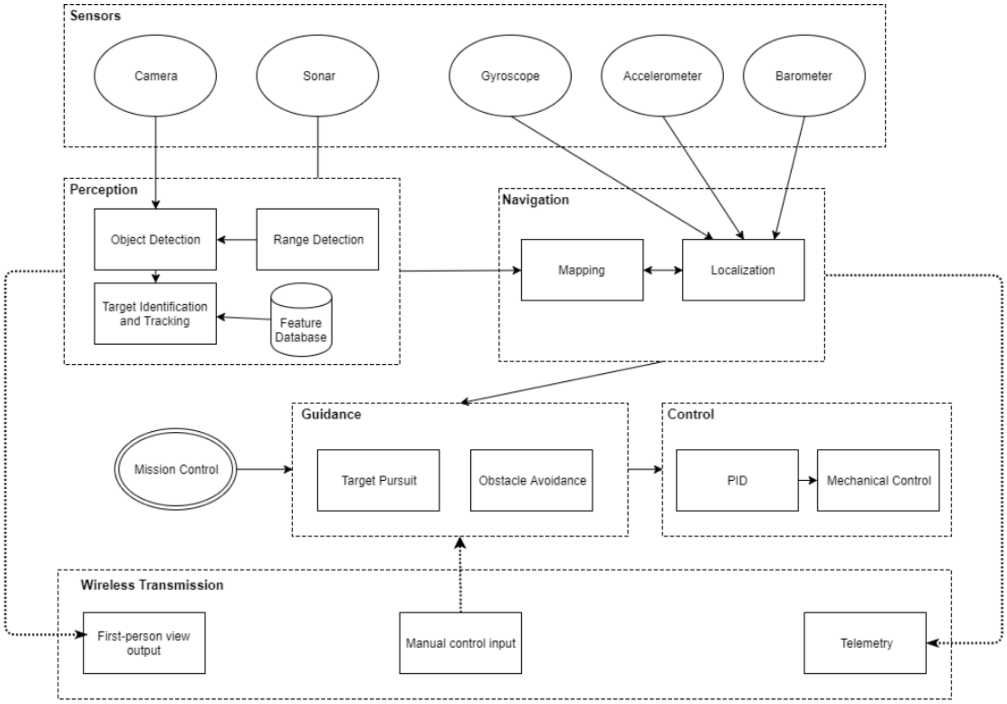
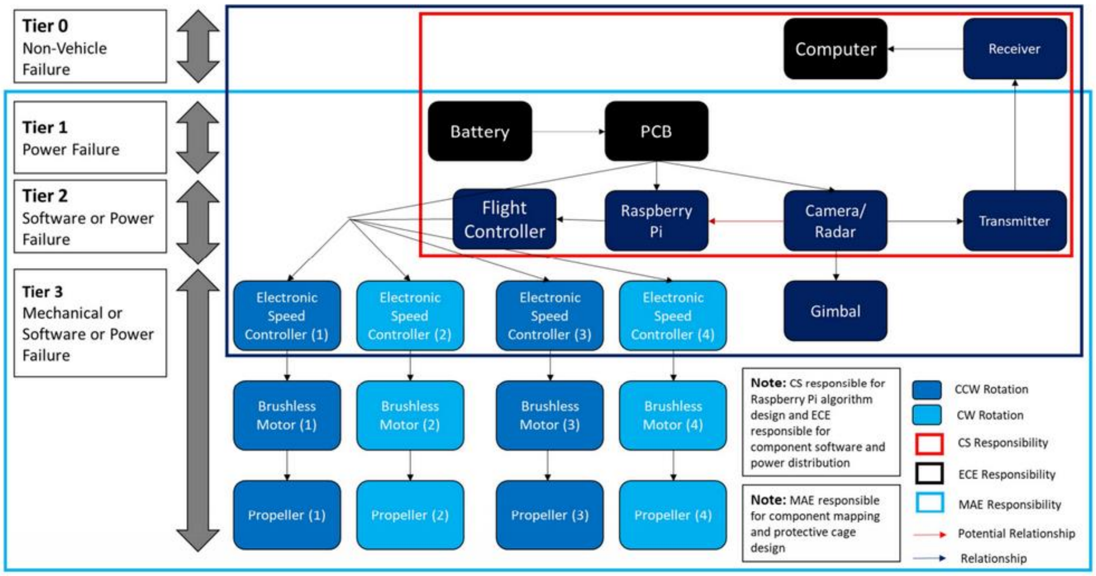


Figure 56: Functional Decomposition

5.2 Risk

This project involved mechanical, electrical, and software systems. Due to the number of subgroups available in our team, risk has been assigned to each respective subgroup. Each group was responsible for their components' risk avoidance and risk mitigation. The following figure is a preliminary chart of how risk responsibility was assigned for this project and its main components:



Note: CS responsible for Raspberry Pi algorithm design and ECE responsible for component software and power distribution

Note: MAE responsible for component mapping and protective cage design

Figure 57: Component Risk Assessment

5.2.1 Risk in Structures

The structural integrity of the RoboCopter greatly depended on the overall strength and reliability of some of its biggest components, including the frame, the cage, and the overall integration of the UAV's parts. A structural problem in these components could have lead to a wide range of consequences, from cosmetic damage, to inaccurate flight, to an unplanned collision. The immediate features of the copter that would be at risk given structural damage are weight, sensor input, and collision protection. a shift in the center of gravity in the quadcopter would lead to an inability to move to the desired locations; furthermore, damage to the frame could affect the stability of all onboard sensors, leading to inaccurate reading of the current percept. Given the intended use of the cage in this competition, where the drone was expected to continuously crash with other smaller drones, any damage to it could have lead one of the prey to go through the cage, or the collision to cause a fracture in any of the other components that are dependent on the cage. Lastly, a fault in the overall integration of the aircraft could lead to a loss of components, which would damage given components physically and affect the flight pattern of the drone.

Frame

The frame contained two major risk, the first being the available space for components and mounts. Most of the components was placed in the central part of the quadcopter. This posed many challenges in optimizing the placements of the sensors, flight controller, battery, PCB, camera mount and other potential components deemed necessary for the design. The second risk was the amount of weight placed in the center of the frame. The increase in mass in the center of the UAV will prevent it from turning efficiently and requiring more power to maneuver. To solve simultaneously solve both the problems, the team explored different mounting designs. Adding an extra platform above or below the top and bottom plate of the frame would have given more space for components. Designing two mounting sections that are away from the center and counterbalance each other for the camera and other components, enhancing the maneuverability of the frame. The team has developed solutions like these with different component configurations and tested which one work the best through trial and error.

Cage

The significant risk in the caging was in the deflection of the material and mounting methods. The material needed to not deflect too much, as it could possibly fracture on the actual cage. The deflection of the material could have also posed a risk to the propellers. If the cage deflected a significant amount under impact, the cage could have come in contact with a moving propellor and damage it. There was also a risk with the mounting of the cage to the frame, that under impact, the connections could possibly break and the cage could have collapsed onto the central housing or propellers. To overcome and account for these risks, two major checks were set in place. The first was to ensure that the cage material would not interfere with any moving components. This was accounted for by creating enough clearance for the cage to deflect without causing any interference. The second check was to ensure the mounting of the cage is tight and secure before any flight taking place. Both of these checks should be enough to ensure the caging around the quadcopter performs as intended and no unaccounted for accidents will occur during flight or landing.

Mounting

The main issue with the current mounting procedure was that if components need to be relocated it is likely that it was difficult to remove them from their initial position. Each component has a different risk associated with it becoming unfastened from the frame. For the motors it is nearly mission critical to secure them but for the raspberry pi, as long as it stayed connected to the other electrical equipment, should still operate as needed. While the goal was to have all components securely mounted by flight testing, if there was uncertainty in the performance of a certain component then a temporary mounting method like duct tape was used in order to reduce the likelihood of redoing the entire assembly.

5.2.2 Risk in Avionics

The next few subsections will cover the risks we have found with the avionics aspect of this project. We feel that we have encountered issues with the flight controller, the PCB, and the sensory equipment. These avionics risks almost turned into catastrophic issues during the day of the competition. The drone could potentially crash or veer of course if one of these components or any subset of these components were to fail during flight and could end up being a safety concern for the individuals within the surrounding area.

Flight Controller

The flight controller is the main component for all flight functions. The product chosen is a COTS Pixhawk that has plenty of online resources to work with and because it has been carefully chosen as a reliable product, there is little risk with the controller itself. However, the majority of the risk is evaluated when the programming is done. If the wrong commands are programmed into the controller, the copter could crash. To mitigate these risks, it is vital that the flight controller was tested for specific functions while being programmed in a simulated environment. This simulation environment comes from a software package called Ardupilot. ECE has more specific details with regard to how they plan to test the controller to ensure its success.

Printed Circuit Board

There is some risk in the PCB considering that the team was responsible for the design of the board. However, rather than all components being assembled on the board by the team, the components were built into the board by a vendor that is experienced with making such components. It is also helpful that this process of manufacturing is economical at around $25 per board.

Sensory Equipment

Sensors used are primarily the ultrasonic and lidar sensors and they do pose a level of high risk. The sensors are to be used for object detection and for altitude determination. Object detection was one of the fundamental requirements that must be met and a mistake in the competition can cost the team a loss of points. The ultrasonic sensors chosen are compatible with the system and have been tested by the ECE team to ensure performance. One or two lidar sensors were used to assist the ultrasonic sensors to have more accurate readings critical areas for obstacle detection, one the center front for forward motion and one underneath facing downward for descending motion. Sensors can be tested before being assembled with the flight controller to ensure that they will perform as expected in flight.

5.2.3 Risk in Propulsion

Propulsion is the ultimate step of the flight process. It is where all the computations that are performed by the on-board companion computer and the flight controller, based on sensor input, are turned into a specific combination of thrust for each motor, which was controlled by its corresponding ESC. The effect of a faulty motor or propeller would be critical in-flight, as it would certainly lead to the instability of the aircraft. Control is lost as soon as the flight controller loses the ability to manipulate the propellers as needed; the copter would immediately need to switch to repair or reset mode, and trigger its return-to-home function.

Motors

The main concern of the design in the propulsion area is the current draw of the motors. With the current draw of the chosen motors being so high, the flight time shrinks to below the 10 minute round time limit. This requires a battery switch halfway through each round, requiring the purchase of multiple batteries. This increases the cost of the overall project. It is possible to reduce the current draw by selecting a lower powered motor, sacrificing thrust and maneuverability. Ultimately, the drone’s time to target needs to be minimized to score the maximum amount of points, so a stronger motor is preferable in this regard. In this case, the risk is acceptable.

Another risk specific to the quadcopter design is the possibility of total flight failure if any of the motors break during flight. Maintenance and a protective cage will hopefully minimize the risk of this occurring.

Propellers

Mechanical safety is the primary concern for the propellers. Propellers are known to break during crashes, and the quadcopter design falls out of the sky with the loss of a single motor or propeller. It is paramount to protect the propellers during flight because of this flaw. The cage is designed to protect the propellers from collision. If a propeller breaks, it needs to be replaceable, so spares was purchased for emergency repairs.

5.2.4 Risk in Vision

The next two sections will cover risks that may affect the vision aspect of this project. The camera and the transmitter and receiver for the first person view video could have caused our project to fail. The camera was the component that makes or breaks this project. The camera needed to have good enough quality for the image recognition algorithms to be able to recognize the prey drones. The transmitter/receiver combination was necessary for the judges to see how our drone behaves during the competition. A failed transmission video would have caused our team to lose points on the day of competition.

Camera

The proposed solution is the Intel RealSense Robotic Development kit and includes a camera and processor selected to operate as the entire object detection system. The camera is 4 Megapixel, so low-resolution is not a risk to the detection software. The software does, however, rely on being introduced to the object of interest in order to “learn” it’s target, which introduces possible risk. The RealSense software can track a 3D object based on a two dimensional reference image and does not necessarily rely on color to confirm a target. However, there is risk associated with unknown prey drones that may have a different look or shape from the sample prey received by the team.

Something that could have presented itself as an issue was compatibility between the RealSense camera and the position kit used to mount the camera onto the frame. The mount, which clamps the camera onto positioning rails, was designed for a GoPro shaped camera, but is customizable. It is important for the camera to maintain its position and angle in reference to the frame to eliminate risk associated with object tracking and navigational/guidance accuracy. For this reason, a stable mount is vital.

Transmitter

The selected method of video transmission is a 5.8 GHz analog transmitter and receiver. While there possible issues with interference of the analog signal are always present, the performance of the long-range signal is much higher than what is needed by a factor of about 66 times conservatively. The transmitter also operates outside the line of site of the receiver, so maneuvering behind obstacles should not interrupt the feed whatsoever. The transmitter is lightweight and compact compared to alternatives, making it a low-risk selection. The largest possible risk in selecting this high strength analog method of FPV transmission would arise if the competition were located in an area where the signal is jammed, which is an incredibly unlikely scenario.

5.2.5 Risk in Power

This section will cover the risks associated with power for the drone. The battery and the power distribution board will need to power the entire system for as long as the round length, which is ten minutes. If power becomes an issue during the round, our quadcopter may end up crashing and could potentially become a safety concern.

Battery

The batteries were the main power supply and damage to its connection with the power distribution board will result in all systems failing to operate. This risk is mitigated by ensuring a secure connection of the XT60 discharge plug to the PDB and the battery has a discharge rating capable of delivering the required amount of amps to the motors. Another major risk with the battery is that it may not be have the capacity needed to complete the full ten minutes of flight. Should this issue be demonstrated during integration, we will either have to do a battery swap during mid-flight or add an additional battery. If an additional battery is added it will need to be added in parallel with the other battery in order to double the capacity and not the voltage.

Power Distribution Board

The power distribution board is connected to the battery and acts as a voltage regulator to supply power to all electrical components. Should the wrong voltage be supplied to certain components, they would likely fail to operate or even potentially overheat and cause a fire. This risk can be reduced by adding additional safety constraints into the PDB design.

5.3 Hardware

The hardware for an autonomous drone will differ slightly from a basic quadcopter layout. In this design, a customized circuit board is need to simulate the controls from a traditional RC transmitter. The battery is connected a power distribution board {PDB) or a battery eliminator circuit (BEC) to provide the appropriate voltage to each component. PDBs and BECs are usually small and lightweight, but can produce a lot of heat, so it is important to keep it away from any heat-sensitive components.

There are four electronic speed controllers (ESCs) that could be connected directly to the battery if a power distribution is not used. The four ESCs are directly connected to the motors and can be mounted on the arms of the frame to evenly distribute the weight. The ESCs was receiving information on how much power to supply from the flight controller unit (FCU). The PDB is also connected directly to the flight controller, but could also receive power from the ESCs if a PDB is not used. The PCB could get power directly from the flight controller depending on if the flight controller has the voltage out pin needed by the PCB (3.3v or 5v). The PCB will have the voltage out pins for the ultrasonic sensors, the camera, and the antenna and any other components required. The PCB will also have input pins for the data from all the components and output data to the flight controller.

The battery will need to be placed in between the top and bottom connector plates of the drone because the PCB needs to go on the bottom of the drone. The battery is also the heaviest component and the best place for it to go in terms of weight distribution is the very center. This will make the drone move accurately which was a challenge because the algorithms will work better if there are not certain motors that need more power because they are carrying a heavier load. If a PDB is used, it will go on top of the connector plate to keep it off the battery.

The PCB will need to go the the bottom because the ultrasonic sensors was attached directly to the board and needs a clear view of the ground to be able to sense how high the drone is. This is very important to be able to track drone prey drones autonomously because the drone needs to know what elevation it needs to be at to be able to make contact with the prey drone. The camera will also need to be on the bottom of the drone to have a good vantage point of the prey drones. If the drone has a high elevation with the camera pointed down looking for prey drones, it will have a clear view free of obstructions from obstacles. The antenna will also be connected to the PCB of possibly the flight controller if it supports the transmission of video in real time. The flight controller can also go on top of the top connector plate.

The picture below shows the setup of the PCB with the components that have already been purchased. The team has started testing these components to ensure they perform as expected. The picture includes the proposed setup with five HC-SR04 ultrasonic sensors which are the sensors that we plan on using for the project. The picture also includes the voltage regulator, crystal, a fuse, the ATMEGA328P processor. Other components, such as the resistors and capacitors can gathered and then tested in the engineering labs. New components was incorporated for the final printed board. The same testing was done when the printed board is purchased.

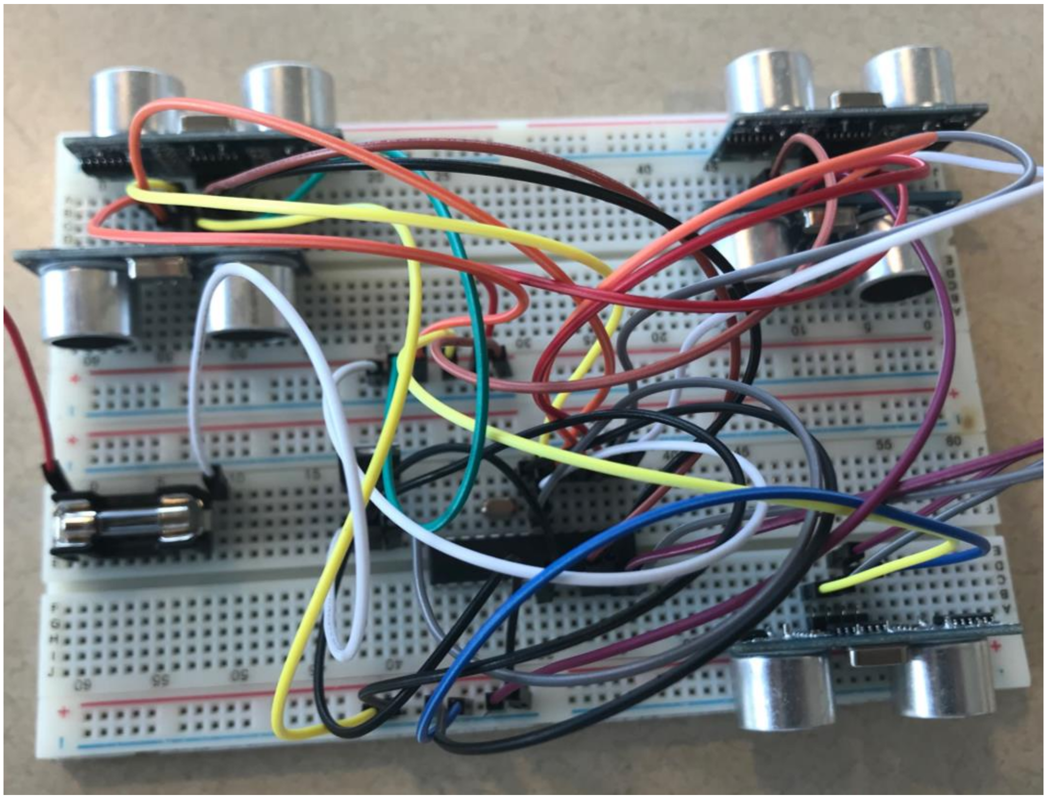


Figure 58: HC-SR04 Ultrasonic sensors and ATMega328P processor for PCB design

5.4 Software

Within this project code will have to be written in order to do a variety of functions, including but not limited to, general flying capabilities, avoiding objects by adjusting the flight path of the drone, detecting the other prey drones within the arena, safely landing the drone, and localization.

The software block diagram below gives a high level view of the system, illustrating the components that are grouped together to perform an overall function and how components will interact with other components during the flight. The legend at the bottom also explains the members of the groups who are overseeing the different functions of the software system.

Flight Mission

One of the most important steps in the software design process for the RoboCopter is the abstraction of the mission as a whole. This involves documenting the various steps that the UAV’s companion computer needs to go through in order to generate the action that best fits its current percept.

The figure below explains at a high level view the mission plan that the Green Team has constructed for this competition. The diagram explains the modes of operation (search, purse, reset and repair) from the start of the round to the end of the ten minute time limit. The black horizontal bars represent the beginning and end of our mission plan.

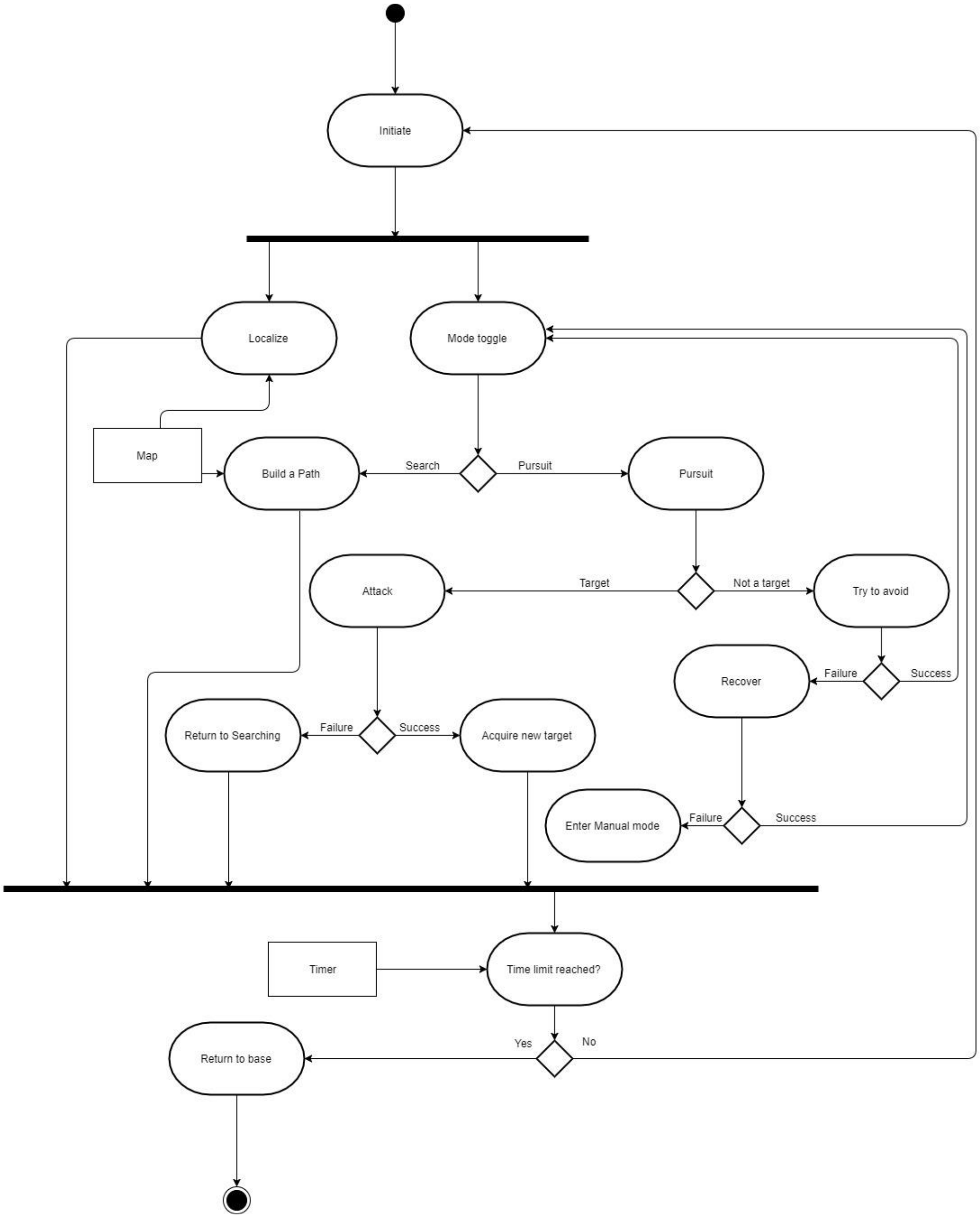


Figure 59: Mission Finite State Diagram

6. Overall Integration and Testing

This section will cover the steps that were taken during the project lifecycle that handled integrating hardware and software components into higher assemblies and the testing strategies to ensure the UAV drone was successful on the day of competition. Thorough testing done on the software and hardware components before, during, and after integration was crucial to the success of this project. One or more team members who have done the most research and/or has the most knowledge about specific sections were in charge of the testing for that section.

The testing that was highlighted in this section is mostly for the printed circuit board and other testing that were done by the electrical engineering and computer engineering students. The other tests that involves the camera, motors, propellers, overall prototype weight, materials, and strength of quadcopter were to be handled primarily by the mechanical engineering, aerospace engineering, and computer science students. The exact procedures and more in depth details of those tests will not be covered in this section but will follow the same general concept.

Multiple styles of testing, including unit, integration, and overall system testing, were conducted. A component that was initially included in the design may have fail due to compatibility oversight across systems, lack of research, or miscommunication. If a hardware component did not pass the tests, then troubleshooting steps was conducted, and if it still failed the test the component was replaced.

6.1 Hardware Testing

This section will cover all aspects of hardware testing. Each component that is used in our design was individually tested to ensure that it integrated well with the other components that it needed to interact with and is working correctly to give the desired performance results. The individual component testing was done prior to integration and also testing of the software.

6.1.1 Hardware Testing Environment

The competition is intended to be outdoors in a large open area but may have been relocated to an indoor environment if weather caused issues on the day of. Our team tested out the components in both indoor and outdoor environments to ensure we had the best chance to be successful no matter where the competition was ultimately held. Attempting to model the operating environment of the drone during the day of the competition was hard, especially for the case of outside environments. The environment cannot be totally controlled and is always changing. Testing was done multiple times, on different days, accounting for high or low wind, cloudy or sunny days, and cold and hot temperatures.

The components for the printed circuit board was tested using either personal equipment or equipment provided to us in one of the engineering labs. Breadboard testing before ordering a printed circuit board can verify that the circuit that we built and simulated on the computer met the requirements of the project. Input and output values was measured to ensure the components behave as expected.

6.1.2 Ultrasonic Sensor Testing

The picture below shows the testing setup done for the HC-SR04 ultrasonic sensors. The four leads of the sensor plug into the 5V, ground, and two I/O pins on the Arduino. Each sensor was individually tested to ensure distance measurements were accurate and that the sensors performed as expected. The testing was done indoors and outdoors on various types of materials. The testing of these sensors was conducted for measurements on different materials. Articles of clothing, couches, trees, people, and any other material that may be used as obstacles on the day of the competition were used during testing.

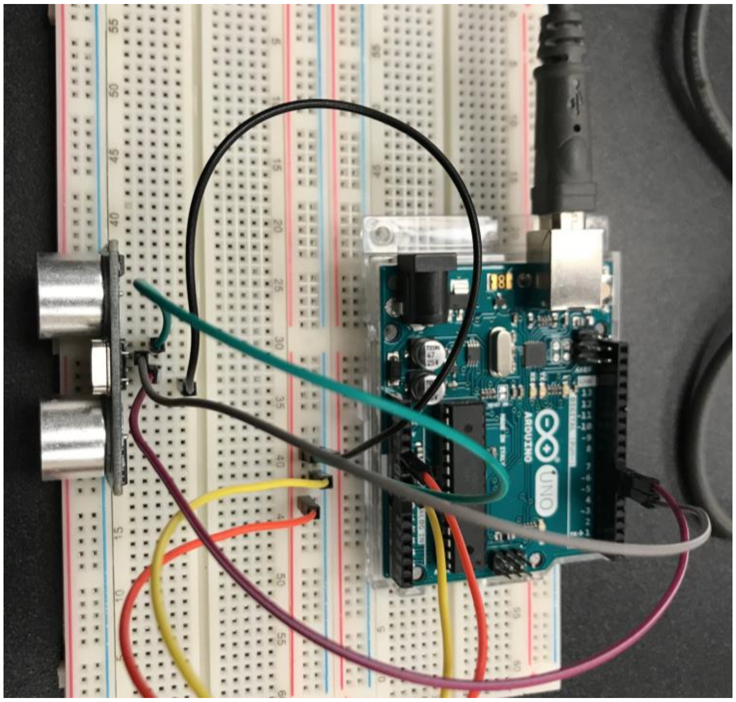


Figure 60: One HC-SR04 Ultrasonic Sensor setup

The next two pictures below shows the breadboard test setup for five ultrasonic sensors and the results of the five sensors printed on the serial monitor of the Arduino IDE. The testing was done using the components that was to be incorporated into the printed circuit board.

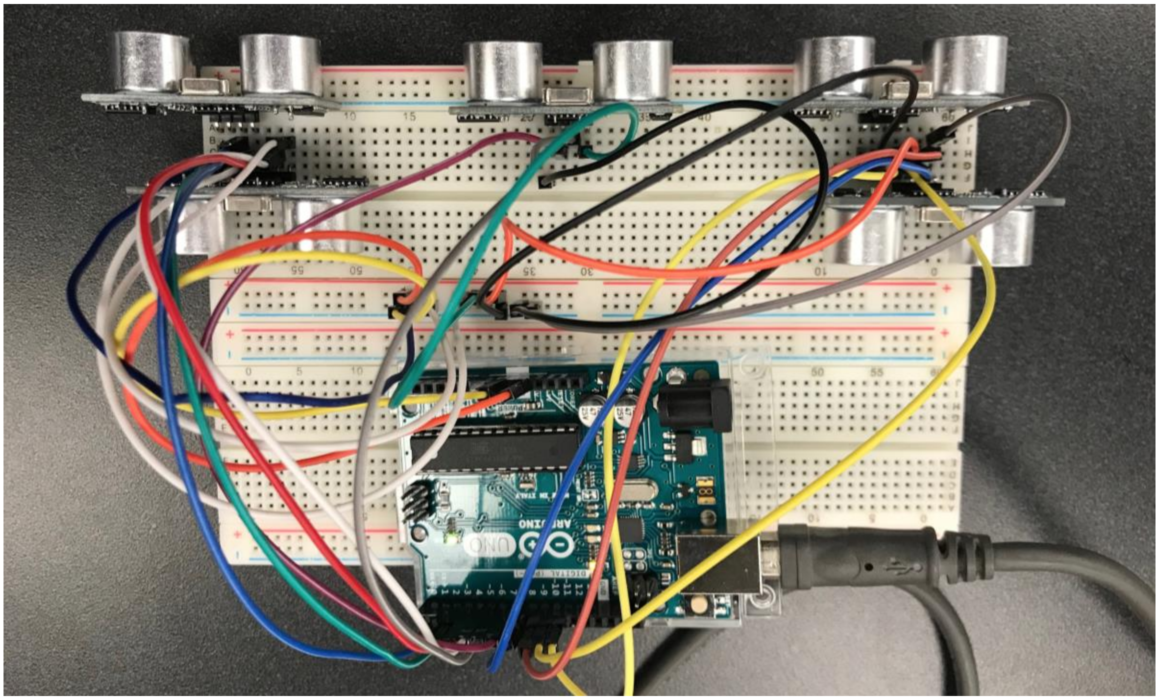
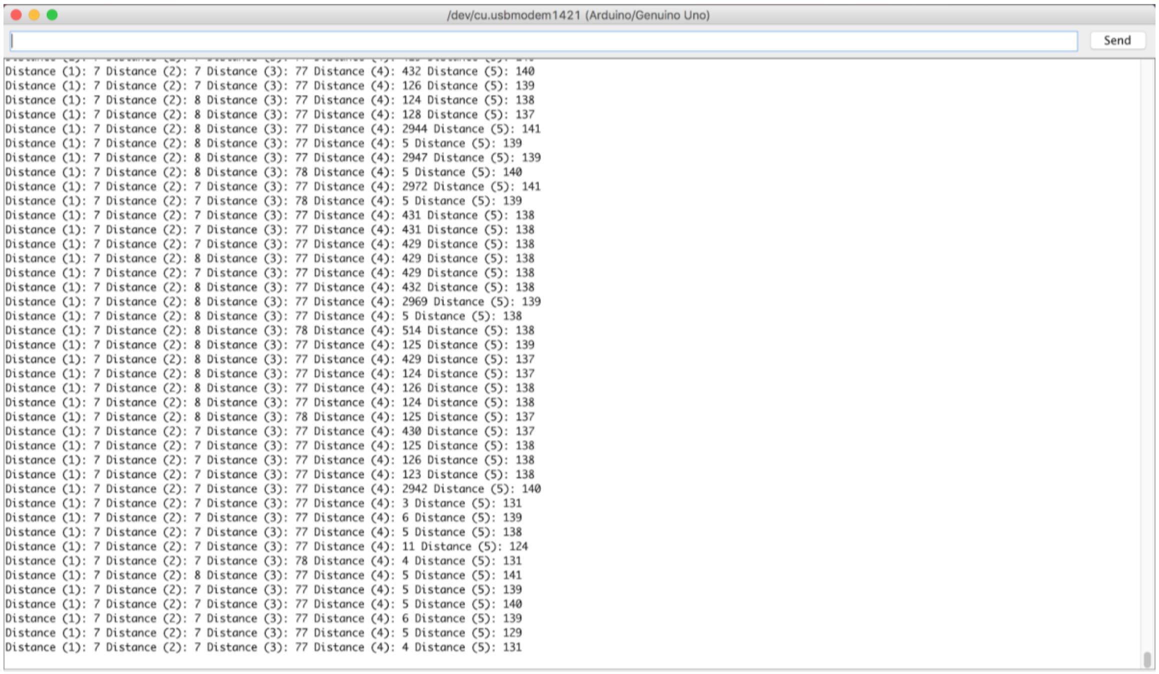


Figure 61: Five HC-SR04 Ultrasonic Sensors setup



```
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 432 Distance (5): 140
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 126 Distance (5): 139
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 124 Distance (5): 138
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 128 Distance (5): 137
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 2944 Distance (5): 141
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 5 Distance (5): 139
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 2947 Distance (5): 139
Distance (1): 7 Distance (2): 8 Distance (3): 78 Distance (4): 5 Distance (5): 140
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 2972 Distance (5): 141
Distance (1): 7 Distance (2): 7 Distance (3): 78 Distance (4): 5 Distance (5): 139
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 431 Distance (5): 138
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 431 Distance (5): 138
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 429 Distance (5): 138
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 429 Distance (5): 138
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 429 Distance (5): 138
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 432 Distance (5): 138
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 2969 Distance (5): 139
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 5 Distance (5): 138
Distance (1): 7 Distance (2): 8 Distance (3): 78 Distance (4): 514 Distance (5): 138
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 125 Distance (5): 139
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 429 Distance (5): 137
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 124 Distance (5): 137
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 126 Distance (5): 138
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 124 Distance (5): 138
Distance (1): 7 Distance (2): 8 Distance (3): 78 Distance (4): 125 Distance (5): 137
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 430 Distance (5): 137
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 125 Distance (5): 138
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 126 Distance (5): 138
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 123 Distance (5): 138
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 2942 Distance (5): 140
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 3 Distance (5): 131
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 6 Distance (5): 139
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 5 Distance (5): 138
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 11 Distance (5): 124
Distance (1): 7 Distance (2): 7 Distance (3): 78 Distance (4): 4 Distance (5): 131
Distance (1): 7 Distance (2): 8 Distance (3): 77 Distance (4): 5 Distance (5): 141
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 5 Distance (5): 139
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 5 Distance (5): 140
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 6 Distance (5): 139
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 5 Distance (5): 129
Distance (1): 7 Distance (2): 7 Distance (3): 77 Distance (4): 4 Distance (5): 131
```

Figure 62: HC-SR04 Ultrasonic Sensors Distance Measurements

The values that are shown above are verified for accuracy against hand calculations done using either a ruler or tape measurer. Multiple sensors can be tested at once facing in the same direction to verify that both sensors are reading and outputting the same data values.

The testing that is described above was done using the Arduino Uno and the ultrasonic sensors. The same setup and testing was conducted with the ATMEGA328P processor and the rest of the components in the printed circuit board when they were delivered to the team.

6.2 Software Testing

This section will cover the software testing that was done for this project. The overall software portion of this project was completed by a combination of computer science and computer engineering students. The computer science students was primarily focused on the computer vision aspect of the project that mostly dealt with object detection and recognition. The computer engineering students mostly worked on software that dealt with the avionics, object avoidance, and the first person view transmission with the ground station. Some topics was covered by multiple members to ensure the requirements are met and the workload isn't overbearing for one or two members of the team.

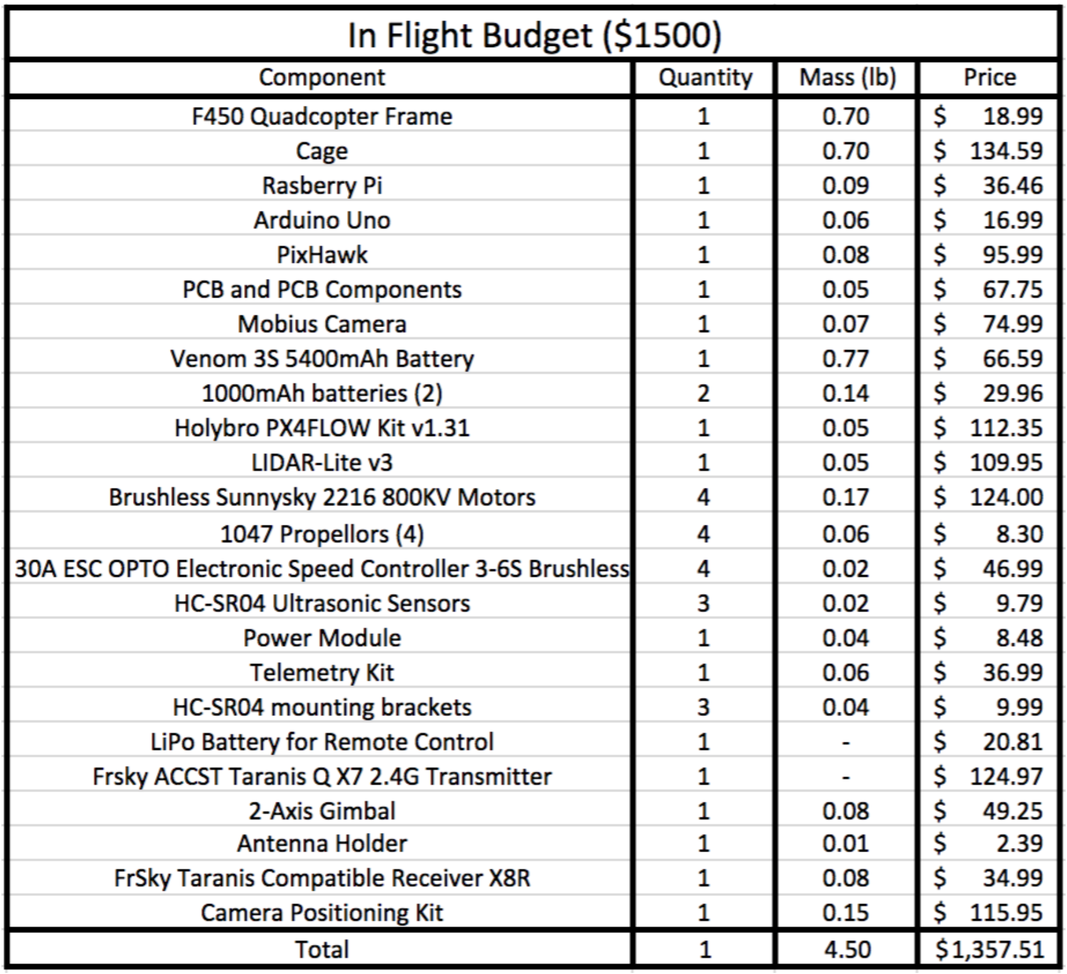
The testing phase for our software was mostly be a bottom-up approach. We tested each component of the system individually, sending inputs that model the operating environment and observing the expected outputs. Functions and variables were checked constantly while debugging the software. Our software development techniques were also continuously revised and altered as we saw fit. Once the lower components pass our test cases and we feel comfortable with integrating the software into a higher assembly, testing continued until all aspects of the system were tested and performed as best as possible. The details of each test case were constructed during the Spring semester as we gained more information.

7. Administration

This project is part of the University of Central Florida’s College of Electrical and Computer Engineering, and because it is a sponsored project, there are several budget and milestone requirements that must be met. Furthermore, given that this is an interdisciplinary project, working with groups of students from different majors is required. This section covers all administrative material that was considered for this project.

7.1 Budget

This project was funded and sponsored by Lockheed Martin. As mentioned on the Financial Requirements section, “The maximum budget allowed is $2000, the “as-demonstrated” maximum budget is $1500”. The table below briefly describes our financial budget for the project.



In Flight Budget ($1500)

Component	Quantity	Mass (lb)	Price
F450 Quadcopter Frame	1	0.70	$ 18.99
Cage	1	0.70	$ 134.59
Rasberry Pi	1	0.09	$ 36.46
Arduino Uno	1	0.06	$ 16.99
PixHawk	1	0.08	$ 95.99
PCB and PCB Components	1	0.05	$ 67.75
Mobius Camera	1	0.07	$ 74.99
Venom 3S 5400mAh Battery	1	0.77	$ 66.59
1000mAh batteries (2)	2	0.14	$ 29.96
Holybro PX4FLOW Kit v1.31	1	0.05	$ 112.35
LIDAR-Lite v3	1	0.05	$ 109.95
Brushless Sunnysky 2216 800KV Motors	4	0.17	$ 124.00
1047 Propellors (4)	4	0.06	$ 8.30
30A ESC OPTO Electronic Speed Controller 3-6S Brushless	4	0.02	$ 46.99
HC-SR04 Ultrasonic Sensors	3	0.02	$ 9.79
Power Module	1	0.04	$ 8.48
Telemetry Kit	1	0.06	$ 36.99
HC-SR04 mounting brackets	3	0.04	$ 9.99
LiPo Battery for Remote Control	1	-	$ 20.81
Frsky ACCST Taranis Q X7 2.4G Transmitter	1	-	$ 124.97
2-Axis Gimbal	1	0.08	$ 49.25
Antenna Holder	1	0.01	$ 2.39
FrSky Taranis Compatible Receiver X8R	1	0.08	$ 34.99
Camera Positioning Kit	1	0.15	$ 115.95
Total	1	4.50	$1,357.51

Table 9: Budget

7.2 Milestones

The following milestones were defined by the Green Team as a whole, based on the guidelines and requirements specified by the sponsor, the ECE Senior Design class, and the MAE and CS departments. Some of these milestones are approximated and are subject to change, as seen fit by the stakeholder requirements.

Key Milestones	Start Date	End Date
Senior Design 1		
Finalize Project Idea	08/21/17	08/28/17
Finalize Interdisciplinary Group Members	09/28/17	09/29/17
Research & Finalize Frame of Drone (quadcopter, octocopter, etc)	09/29/17	10/20/17
60 Page Submission	10/16/17	11/03/17
PCB Design	11/04/17	11/17/17
Order PCB	N/A	12/04/17
100 Page Submission	11/04/17	11/17/17
Order Parts	11/17/17	12/19/17
Final Paper Due	11/17/17	12/04/17
Senior Design 2		
Assemble Design	01/08/18	02/27/18
Software Development	01/08/18	02/27/18
Testing	03/1/18	04/13/18
Final Presentation	04/17/18	04/17/18
Final Product		04/20/18

Table 10: Project Milestones

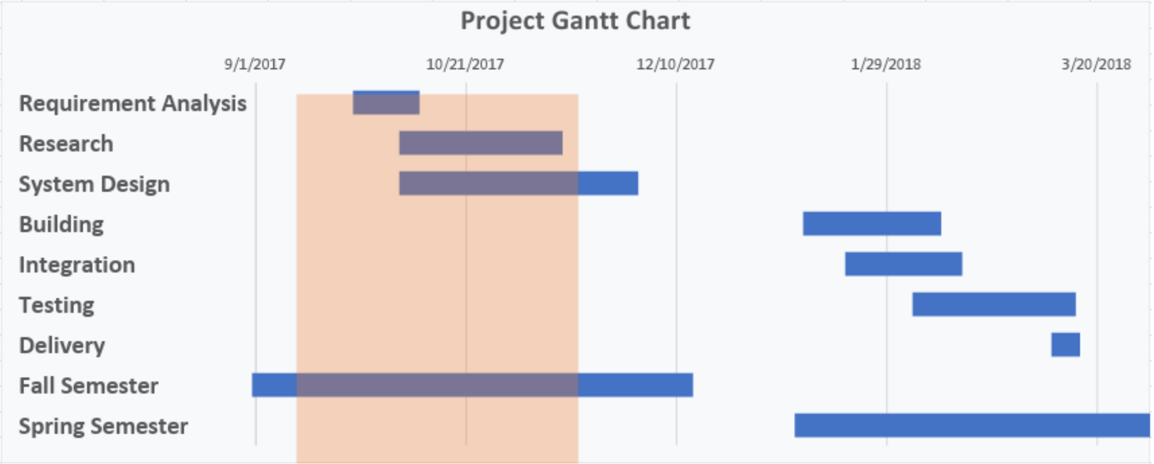


Figure 63: Project Gantt Chart

7.3 Team Roster

The Green Team for the Lockheed Martin RoboCopter's Senior Design Competition consisted of thirteen students: six students from the Department of Mechanical and Aerospace Engineering (MAE), three students from the Department of Computer Science (CS), and four students from the Department of Electrical and Computer Engineering (ECE). Every member of the Green Team contributed by bringing their prior knowledge and experiences, additional information gained through research, and overall energy to this project. Below is a list of the members from the Green Team as well as biographies from each of the students from the ECE Department who have contributed their research to this paper.

7.3.1 Electrical & Computer Engineering Students

Dominic Williams

Dominic is currently pursuing his Bachelor's degree in Computer Engineering. His interest in Engineering started around the age of twelve or thirteen years old. During high school is when he decided to direct his focus towards Computer Engineering specifically because of the courses he took and the projects he became involved with.

During his time at the University of Central Florida, Dominic became involved with multiple organizations such as the Caribbean Students' Association and the National Society of Black Engineers, where he held executive board positions within both organizations. He was also able to gain two co-op positions with Lockheed Martin during his time at UCF. He spent a little over a year working as a Quality Engineering student at the Missiles and Fire Control facility and is currently a Systems Engineering student working at the Rotary and Mission Systems facility. Through his leadership experiences on campus within student organizations and industry experience with Lockheed Martin, Dominic has been able to gain valuable experiences that will transfer into the industry.

Over the course of his undergraduate career Dominic’s interest in Software Engineering has only increased with each class he took. Through this project, he hopes to gain more experience with the software engineering lifecycle and more knowledge about different fields such as machine vision, sensing, and wireless communication.

Ley Nezifort

Ley is currently pursuing a Bachelor's degree in Computer Engineering along with a minor in Technological Entrepreneurship. He developed an interest in math and engineering at a very young age, as he was always trying to break into any toys, gadgets that uses electricity. Coming into college, he declared his major as a computer engineer; However, it wasn’t until the end of his sophomore year that he started developing a burning passion for the field of software engineering.

Ley has been able to use his passion for the field to better himself as a software engineer. He had the opportunity to intern with a couple industry leaders such as Intel, Dell, and Deloitte. From those experiences, he gained some relevant working experiences in most of the phases within the software development cycle, application development, process automation and software testing.

Throughout the project, Ley’s responsibilities will vary from time to time, but he was mostly focusing on topics such as Vision systems, more importantly Object Detection & Tracking, Pathing, and flight controller interfacing. He hopes to learn a lot about Computer Vision, which is something he’s been interested in for quite some time.

Maverick Dusan

Maverick is currently pursuing his Bachelor’s degree in Electrical Engineering at the University of Central Florida. An interest in physics and drew him to the Electrical Engineering field and the coursework proved to be the most rewarding. Unending curiosity about the world of power and electronics has made his academic career fulfilling and job prospects exciting.

Elective classes for Maverick have been geared towards signal processing and power systems with projects dealing with communication systems, signal encoding, microgrids, etc. Industry opportunities have given me experience in electrical equipment condition assessments, safety and hazard analysis for electrical systems, working on creating deliverables for private and government contracts, and more.

Maverick's major role in the group was dealing with the power requirements and power distribution to create a safe, efficient drone. Other major roles include collaborating with the Computer Science team and the Computer Engineers to bring the code, cameras, and sensors together in the PCB design. He expects to gain experience in PCB design and working with other disciplines to complete a project. Working with such a large group creates a large pool of knowledge and experience to draw from but also creates conflicting ideas. Maverick is excited to work through these issues and create the best final product as possible.

Julian Quitian

Julian is currently pursuing a BS Degree in Computer Engineering. His high school courses helped grow his interest in programming and software engineering. He enjoys working with people and developing efficient embedded systems.

Julian's previous experience comes mainly in the form of Virtual Reality, embedded systems, service management, and database development; this comes from research and extracurricular opportunities, as well as from internships with a consulting and a launch services company. He is comfortable performing a wide range of tasks including programming, scripting, IP networking, etc… as well as working with low level software. Furthermore, he enjoys using his engineering knowledge in various fields to smoothly integrate hardware with software for full transition and functionality.

Two of Julian's major roles in this project are working with different groups in order to maximize team dynamics, and take responsibility as the main flight controller engineer. He expects to gain real-world avionics, telemetry, and embedded engineering experience in a real-time processing environment. He also expects to contribute to the Vision Systems, Pathfinding, and electronics systems, learning how to leverage everyone's knowledge to develop a modular product with a simple, competitive, and reliable design.

7.3.2 Computer Science Students

Sean Reedy

Maxim Shelopugin

Michael Alexander

7.3.3 Mechanical & Aerospace Students

Alexander Dodson (AE)

Cameron Blastic (AE)

Josh Lajza (ME)

Karl Kage (ME)

William Yepes (ME)

Yasmin Coleman (AE)

8. Conclusion

The final design for this project demonstrated the technical skills and knowledge that was gained throughout the duration of this project. The design consisted of the software and hardware components that we as a team decided was best suitable to meet the needs of the project. Although we have many design constraints which include a limited amount of time to finish the product we believe that the decisions that have been made has showcased the diligence and cooperation of the members of the Green Team. The final components that were selected have been thoroughly researched and discussed with multiple members of the group before being implemented into the design.

This project has allowed members from multiple engineering disciplines and different cultures to come together and apply the knowledge they have learned from their respective majors. Aside from the individual research that was done by each member, we were also able to learn a lot from the other members of the group. As electrical and computer engineering students we were able to learn from the mechanical and aerospace students as well as the computer science students. The knowledge that was transferred between members of different majors did not only help us achieve the goals of this project but also allowed us to be more well rounded engineers. This project is representing real world experiences that happen every day within large engineering companies such as Lockheed Martin. Working alongside people who come from a different cultural and technical background is very common and it is something we will all have to do in our future careers.

Throughout the semester the team has made various changes to the project, including the projected cost of the budget, the components we planned to use, who would spearhead which sections of the project, etc. We continued to try and follow the project timeline that we have created so that we would stay on track to finish on time. We believe that the timeline we have created gave us sufficient time to build, integrate, and test the components of the project to ensure we met all the requirements and would be successful on the day of the competition.

Every member of our Green Team has personal goals for this project that they wished to achieve by the end of the Spring Semester. Some of the goals included just gaining new knowledge about drone technology and the components that are included in the production of UAV drones. Our members also wanted to learn how to integrate components to build up a larger system and work with simulations to help with testing of the design being proposed. Some members wanted to learn about machine vision, embedded systems, avionics, telemetry, how sensors can allow the drone to sense and then react to the conditions of its operating environment. In addition to the technical knowledge that can be gained from the project, members also just wanted the experience and to learn more about what the other engineering disciplines do in their field and the backgrounds of the members of our group. The Green Team consisted of a diverse group of students who come from many different backgrounds and have had different experiences that brings value to the project.

The Green Team also expected to accomplish various team goals. The team expected to have consistent effort and engagement from all the members of the group throughout the entire project. We as a team were able to discuss and express our opinions about design decisions in a friendly working environment. We expected to have built and integrated all the components in a timely fashion so we are not rushing to test the final design. Inadequate testing may have cause some issues on the day of the competition and we planned to mitigate those risks. At the end we intended to have a functioning drone and win the RoboCopter's Competition.

9. Appendices

This section will outline the different references that were used to gain information relative to the project. The section will also copyright permissions that we received from various sources.

Appendix A - References

[1] Corrigan, F. (2017, July 23). Top Collision Avoidance Drones And Obstacle Detection Explained. Retrieved September 7, 2017, from https://www.dronezon.com/learn-about-drones-quadcopters/top-drones-with-obstacle-detection-collision-avoidance-sensors-explained/

[2] Gageik, N., Benz, P., & Montenegro, S. (2015). Obstacle Detection and Collision Avoidance for a UAV With Complementary Low-Cost Sensors. IEEE Access, 3, 599-609. doi:10.1109/access.2015.2432455

[3] Corrigan, F. (2017, June 5). 10 Top Lidar Sensors For UAVs And So Many Great Uses. Retrieved September 16, 2017, from https://www.dronezon.com/learn-about-drones-quadcopters/best-lidar-sensors-for-drones-great-uses-for-lidar-sensors/

[4] Corrigan, F. (2013, June 3). Time of Flight Camera Sensors On Drones And 10 Terrific Uses. Retrieved September 16, 2017, from https://www.dronezon.com/learn-about-drones-quadcopters/best-uses-for-time-of-flight-tof-camera-depth-sensor-technology-in-drones-or-ground-based/

[5] Simplifying Time-of-Flight Distance Measurements. (2017, January 05). Retrieved October 1, 2017, from https://www.digikey.com/en/articles/techzone/2017/jan/simplifying-time-of-flight-distance-measurements

[6] Introduction to the Time-of-Flight (ToF) System Design. (2014, May). Retrieved October 2, 2017, from http://www.ti.com/lit/ug/sbau219d/sbau219d.pdf

[7] Time-of-Flight Camera – An Introduction. (2014, May). Retrieved from http://www.ti.com/lit/wp/sloa190b/sloa190b.pdf

[8] Corrigan, F. (2015, August 9). UAVs In The Lidar Applications Sector Increases Substantially. Retrieved October 6, 2017, from https://www.dronezon.com/learn-about-drones-quadcopters/uav-lidar-applications-services-technology-systems/

[9] Calin, G., & Roda, V. O. (2006, September 8). Real-time disparity map extraction in a dual head stereo vision system. Retrieved from http://www.scielo.org.ar/scielo.php?script=sci_arttext&pid=S0327-07932007000100005

[10] Corrigan, F. (2014, November 19). Understanding FPV Flying And FPV Equipment In Drones. Retrieved October 1, 2017, from https://www.dronezon.com/learn-about-drones-quadcopters/what-is-fpv-flying-drone-equipment/

[12] Bergersen, B. (n.d.). Drones and wireless video. Retrieved from https://www.datarespons.com/drones-wireless-video/

[13] What is an IR Sensor? (n.d.). Retrieved from http://education.rec.ri.cmu.edu/content/electronics/boe/ir_sensor/1.html

[14] Damien (n.d.). How to Build a Drone - A Definitive Guide For Newbies. Retrieved November 17, 2017, from http://beginnerflyer.com/build-a-drone/

[15] Dronecode. (n.d.). Retrieved November 17, 2017, from https://www.dronecode.org/

[16] Unmanned Aerial Vehicle. (2017, November 15). Retrieved November 17, 2017, from https://en.wikipedia.org/wiki/Unmanned_aerial_vehicle

[17] Benson, C. (2014, October 29). How to Make a Drone. Retrieved November 17, 2017, from http://www.robotshop.com/blog/en/robots/gorobotics/tutorials/how-to-make-a-drone-uav

[18] Beal, V. (n.d.). Register. Retrieved November 17, 2017, from https://www.webopedia.com/TERM/R/register.html

[19] Ltd., A. (n.d.). Cortex-M4 – Arm Developer. Retrieved November 17, 2017, from https://developer.arm.com/products/processors/cortex-m/cortex-m4

[20] Tutorial: Microcontrollers-MCU Basic Structure/Operation, Renesas Engineer School. (n.d.). Retrieved November 17, 2017, from https://www.renesas.com/en-us/support/technical-resources/engineer-school/mcu-01-basic-structure-operation.html

[21] What is arithmetic-logic unit (ALU)? - Definition from WhatIs.com. (n.d.). Retrieved November 17, 2017, from http://whatis.techtarget.com/definition/arithmetic-logic-unit-ALU

[22] What is clock speed? - Definition from WhatIs.com. (n.d.). Retrieved November 17, 2017, from http://whatis.techtarget.com/definition/clock-speed

[23] Von Neumann architecture. (2017, November 14). Retrieved November 17, 2017, from https://en.wikipedia.org/wiki/Von_Neumann_architecture

[24] 8051 microcontroller history|architecture |programming. (2015, June 08). Retrieved November 17, 2017, from http://microcontrollerslab.com/8051-microcontroller/

[25] PIC microcontroller. (2017, November 12). Retrieved November 17, 2017, from https://en.wikipedia.org/wiki/PIC_microcontroller

[26] Atmel AVR. (2017, November 14). Retrieved November 17, 2017, from https://en.wikipedia.org/wiki/Atmel_AVR

[27] Orthogonal instruction set. (2017, September 28). Retrieved November 17, 2017, from https://en.wikipedia.org/wiki/Orthogonal_instruction_set

[28] AVR042: Atmel AVR Design Considerations. (n.d.). Retrieved November 17, 2017, from http://www.atmel.com/images/atmel-2521-avr-hardware-design-considerations_applicationnote_avr042.pdf

[29] AVR Instruction Set Manual. (n.d.). Retrieved November 17, 2017, from http://www.atmel.com/images/Atmel-0856-AVR-Instruction-Set-Manual.pdf

[30] What is the difference between 8051, PIC, AVR and ARM? (2014, September 9). Retrieved November 17, 2017, from https://www.quora.com/What-is-the-difference-between-8051-PIC-AVR-and-ARM

[31] Atmel SMART ARM Processor Based MCUs . (n.d.). Retrieved November 17, 2017, from http://www.atmel.com/products/microcontrollers/arm/default.aspx

[32] Cleanflight. (2017, November 03). Retrieved November 17, 2017, from https://github.com/cleanflight/cleanflight

[33] Ardupilot About Page. (n.d.). Retrieved November 17, 2017, from http://ardupilot.org/about

[34] Copter Home. (n.d.). Retrieved November 17, 2017, from http://ardupilot.org/copter/index.html

[35] APM 2.5 and 2.6 Overview. (n.d.). Retrieved November 17, 2017, from http://ardupilot.org/copter/docs/common-apm25-and-26-overview.html

[36] Home - Pixhawk Hardware Project. (n.d.). Retrieved November 17, 2017, from https://pixhawk.org/

[37] Dronecode Connectors Standards. (n.d.). Retrieved November 17, 2017, from https://wiki.dronecode.org/workgroup/connectors/start

[38] Getting Started with the Arduino Tian. (n.d.). Retrieved November 17, 2017, from https://www.digikey.com/en/maker/blogs/getting-started-with-the-arduino-tian/41ce3252c11249ca94666ca5c3291c31

[39] Arduino Tian. (n.d.). Retrieved November 17, 2017, from https://store.arduino.cc/usa/arduino-tian

[40] Raspberry Pi 3 specs, benchmarks & more. (2017, June 02). Retrieved November 17, 2017, from https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks/

[41] Jetson TK1 Embedded Developer Kit from NVIDIA. (n.d.). Retrieved November 17, 2017, from http://www.nvidia.com/object/jetson-tk1-embedded-dev-kit.html

[42] Hildenbrand, J. (2014, August 03). A look at NVIDIA's Jetson TK1. Retrieved November 17, 2017, from https://www.androidcentral.com/look-nvidias-jetson-tk1

[43] Corrigan, F. (2017, October 26). Top Collision Avoidance Drones And Obstacle Detection Explained. Retrieved November 17, 2017, from https://www.dronezon.com/learn-about-drones-quadcopters/top-drones-with-obstacle-detection-collision-avoidance-sensors-explained/

[44] Boosting and AdaBoost for Machine Learning. (2016, September 21). Retrieved November 17, 2017, from https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/

[45] S. (2017, April 29). Boosting algorithm: AdaBoost – Towards Data Science. Retrieved November 17, 2017, from https://medium.com/towards-data-science/boosting-algorithm-adaboost-b6737a9ee60c

[46] U. (2017, May 28). An Intuitive Explanation of Convolutional Neural Networks. Retrieved November 17, 2017, from https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

[47] (n.d.). Retrieved November 17, 2017, from http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/

[48] Dettmers, T., Says, D. O., Says, T. D., Says, I., Says, A. A., Says, S. F., . . . Says, Y. T. (2015, October 10). Understanding Convolution in Deep Learning. Retrieved November 17, 2017, from http://timdettmers.com/2015/03/26/convolution-deep-learning/

[49] Parthasarathy, D. (2017, April 22). A Brief History of CNNs in Image Segmentation: From R-CNN to Mask R-CNN. Retrieved November 17, 2017, from https://blog.athelas.com/a-brief-history-of-cnns-in-image-segmentation-from-r-cnn-to-mask-r-cnn-34ea83205de4

[50] Region of interest pooling explained. (2017, July 07). Retrieved November 17, 2017, from https://blog.deepsense.ai/region-of-interest-pooling-explained/

[51] Hollemans, M. (n.d.). Retrieved November 17, 2017, from http://machinethink.net/blog/object-detection-with-yolo/

[52] Aero, P. (2017, January 19). What Makes a Good Camera for Drone Surveys and Inspections. Retrieved November 17, 2017, from https://blog.propelleraero.com/what-makes-a-good-camera-for-drone-surveys-and-inspections-460fb9fb7099

[53] LPC4330 Xplorer Board. (n.d.). Retrieved November 17, 2017, from https://www.element14.com/community/docs/DOC-48945/l/lpc4330-xplorer-board#documents

[54] OV9715. (n.d.). Retrieved November 17, 2017, from http://www.ovt.com/sensors/OV9715

[55] Hang, S., Shen, W., & Liu, Z. (n.d.). DeepDrone: ObjectDetectionandTrackingfor SmartDronesonEmbeddedSystem. Retrieved November 17, 2017.

[56] Chaudhari, M., Ondur, S., & Vanjare, G. (n.d.). A review on Face Detection and study of Viola Jones method .

[57] Cen , K. (n.d.). StudyofViola-JonesRealTimeFaceDetector. Retrieved November 17, 2017.

[58] Uijlings, J., Van de Sande, K., & Gevers, T. (n.d.). Selective Search for Object Recognition.

[59] ODROID-XU4. (n.d.). Retrieved December 02, 2017, from http://www.hardkernel.com/main/products/prdt_info.php?g_code=G143452239825

[60] *2014 Quadcopter vs Hexacopter vs Octocopter: Pros & Cons [EXPOSED]*. (2014). *Dronebly*. Retrieved 2 September 2017, from http://dronebly.com/quadcopter-vs-hexacopter-vs-octocopter-the-pros-and-cons

[61] Bridge, T. (2015). *Why fly an Octocopter? - The Broadcast Bridge - Connecting IT to Broadcast*. *Thebroadcastbridge.com*. Retrieved 2 September 2017, from https://www.thebroadcastbridge.com/content/entry/1211/why-fly-an-octocopter

[62] *Quadcopter vs Octocopter: Which One Is Better For Aerial Photographers and Videographers? - RC Crunch*. (2016). *RC Crunch*. Retrieved 2 September 2017, from https://www.rccrunch.com/quadcopter-vs-octocopter/

[63] Simoes, C. (2013). *Cite a Website - Cite This For Me*. *Fenix.tecnico.ulisboa.pt*. Retrieved 2 September 2017, from https://fenix.tecnico.ulisboa.pt/downloadFile/563345090412782/Resumo.pdf

[64] *Advanced MultiCopter Design — Copter documentation*. (2016). *Ardupilot.org*. Retrieved 5 September 2017, from http://ardupilot.org/copter/docs/advanced-multicopter-design.html

[65] Truong, A., & Papamoschou, D. (2013). *Cite a Website - Cite This For Me*. *Supersonic.eng.uci.edu*. Retrieved 5 September 2017, from http://supersonic.eng.uci.edu/download/AIAA-2013-0217.pdf

[66] *Brushless Motor Kv Constant Explained • LearningRC*. (2015). *LearningRC*. Retrieved 8 September 2017, from http://learningrc.com/motor-kv/

[67] *'Fly Electric' - Winding Density*. (2015). *Flyelectric.ukgateway.net*. Retrieved 9 September 2017, from http://www.flyelectric.ukgateway.net/winding.htm

[68] Allain, R. (2017). *Modeling the Thrust From a Quadcopter*. *WIRED*. Retrieved 9 September 2017, from https://www.wired.com/2014/05/modeling-the-thrust-from-a-quadcopter/

[69] Jacobs, A. (2015). *How to choose the right size motors & ESCs for your Drone, quadcopter, or Multirotor build - Quad Questions*. *Quad Questions*. Retrieved 9 September 2017, from https://quadquestions.com/blog/2017/02/22/choose-right-size-motors-drone/

[70] Dunn, T. (2015). *RC Battery Guide: The Basics of Lithium-Polymer Batteries - Tested.com*. *Tested*. Retrieved 15 September 2017, from http://www.tested.com/tech/502351-rc-battery-guide-basics-lithium-polymer-batteries/

[71] *Learning Brushless Motors*. (2013). *Rcboats.kiwi*. Retrieved 15 September 2017, from http://rcboats.kiwi/index.php/ct-menu-item-15/ct-menu-item-31/ct-menu-item-43

[72] *Sunnysky 2216 800KV*. *Banggood.com.com*. Retrieved 4 April 2018, from https://www.banggood.com/Sunnysky-V2216-800KV-Outrunner-Brushless-Motor-p-78175.html?cur_warehouse=CN

[73] *The guide to choose motors*. (2016). *Rcfpvplane.com*. Retrieved 15 September 2017, from http://rcfpvplane.com/the-guide-to-choose-motors-for-your-quadcopter-and-multirotor/

[74] *10 Ways to Extend Drone Battery Life*. (2015). *Beginner Flyer*. Retrieved 15 September 2017, from http://beginnerflyer.com/10-ways-to-extend-drone-battery-life/

[75] Wortel, M. *HOW TO CARE FOR YOUR LiPos*. *Site.petitrc.com*. Retrieved 23 September 2017, from http://site.petitrc.com/Tech/RCCA_How2TakeCareOfYourLiPo/

[76] *Lithium Polymer Battery - iRobotics Tech Wiki*. *Irobotics.illinois.edu*. Retrieved 23 September 2017, from http://irobotics.illinois.edu/wiki/Lithium_Polymer_Battery#C_Rating

[77] *Venom 5400mah Lithium Polymer Battery*. *Amazon.com*. Retrieved 27 September 2017, from https://www.amazon.com/Venom-5400mAh-Battery-Universal-Traxxas/dp/B000W7PZU6
[78] Vorkoetter, S. (2001). *To BEC or Not To BEC*. *http://www.stefanv.com*. Retrieved 27 September 2017, from http://www.stefanv.com/electronics/qf200105.html

[79] *Dimension Engineering - BECs for every R/C plane and helicopter - frequently asked questions*. *Dimensionengineering.com*. Retrieved 27 September 2017, from https://www.dimensionengineering.com/info/bec

[80] *Arduino Uno Rev3*. *Store.arduino.cc*. Retrieved 29 September 2017, from https://store.arduino.cc/usa/arduino-uno-rev3

[81] Autopilot, P. *Pixhawk Mini · PX4 User Guide*. *Docs.px4.io*. Retrieved 29 September 2017, from https://docs.px4.io/en/flight_controller/pixhawk_mini.html

[82] *Brownout due to voltage sag* (2013). *English*. Retrieved 30 September 2017, from https://www.rcgroups.com/forums/showthread.php?2034670-Brown-out-due-to-voltage-sag

[83] Dutta, P. (2008). *Cite a Website - Cite This For Me*. *Cs.berkeley.edu*. Retrieved 29 September 2017, from http://www.cs.berkeley.edu/~prabal/teaching/cs194-05-s08/cs194-designflow.ppt

[84] Tavernier, K. (2017). *Cite a Website - Cite This For Me*. *Ucamco.com*. Retrieved 4 October 2017, from https://www.ucamco.com/files/downloads/file/125/the_gerber_file_format_version_2_faq.pdf

[85] Feranec, R. (2017). *5x Why choose Altium Designer for Electronic Hardware Design - Welldone Blog - FEDEVEL*. *Welldone Blog*. Retrieved 4 October 2017, from https://www.fedevel.com/welldoneblog/2011/12/5x-why-to-choose-altium-designer-for-electronic-hardware-design/

[86] *Altium Designer 2013 Available | Altium.com*. *Altium.com*. Retrieved 4 October 2017, from http://www.altium.com/company/newsroom/press-releases/altium-designer-2013-available

[87] Scarpino, M. (2014). *Introducing EAGLE, the Easily Applicable Graphical Layout Editor for Circuit Board Design | 1.1 A Whirlwind Tour of EAGLE | InformIT*. *Informit.com*. Retrieved 4 October 2017, from http://www.informit.com/articles/article.aspx?p=2191413

[88] *Making PCBs in EAGLE*. *Princeton.edu*. Retrieved 4 October 2017, from http://www.princeton.edu/~mae412/HANDOUTS/PCBs_in_Eagle.html

[89] *Eagle vs. KiCad | Big Mess o' Wires*. (2010). *Bigmessowires.com*. Retrieved 4 October 2017, from https://www.bigmessowires.com/2010/05/03/eagle-vs-kicad/

[90] *AVR 8-bit MicrocontrollersAtmel.com*. Retrieved 4 October 2017, from http://www.atmel.com/Images/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf

[91] *AVR 16-bit MicrocontrollersWeb.csulb.edu*. Retrieved 10 October 2017, from http://web.csulb.edu/~hill/ee346/Lectures/08%20ATmega%20GPIO.pdf

[92] *Arduino - AnalogInputPins*. *Arduino.cc*. Retrieved 10 October 2017, from https://www.arduino.cc/en/Tutorial/AnalogInputPins

[93] *Switching Regulator Fundementals*. (2012). *Ti.com*. Retrieved 14 October 2017, from http://www.ti.com/lit/an/snva559a/snva559a.pdf

[94] *MAX16910 pdf, MAX16910 description, MAX16910 datasheets, MAX16910 view ::: ALLDATASHEET :::*. *Pdf1.alldatasheet.com*. Retrieved 15 October 2017, from http://pdf1.alldatasheet.com/datasheet-pdf/view/298714/MAXIM/MAX16910.html

[95] Power, R. (2017). *R-78E5.0-0.5 Recom Power | Power Supplies - Board Mount | DigiKey*. *Digikey.com*. Retrieved 15 October 2017, from https://www.digikey.com/product-detail/en/recom-power/R-78E5.0-0.5/945-1648-5-ND/2834904

[96] *RECOM DC/DC Converter*. (2017). *Recom-power.com*. Retrieved 20 October 2017, from https://www.recom-power.com/pdf/Innoline/R-78Exx-0.5.pdf

[97] Lee, B. (1999). *Understanding the Terms and Definitions of LDO Voltage Regulators*. *Ti.com*. Retrieved 25 October 2017, from http://www.ti.com/lit/an/slva079/slva079.pdf

[98] Wolbert, B. *Designing P.C. Board Heat Sinks*. *Ww1.microchip.com*. Retrieved 25 October 2017, from http://ww1.microchip.com/downloads/en/appnotes/ah-17.pdf

[99] Seshasayee, N. *Understanding Thermal Dissipation and Design of a Heatsink*. *Ti.com*. Retrieved 25 October 2017, from http://www.ti.com/lit/an/slva462/slva462.pdf

[100] *What is a decoupling capacitor and how do I know if I need one?*. *Electronics.stackexchange.com*. Retrieved 27 October 2017, from https://electronics.stackexchange.com/questions/2272/what-is-a-decoupling-capacitor-and-how-do-i-know-if-i-need-one

[101] *3-Terminal 1A Positive Voltage Regulator*. *Ee-classes.usc.edu*. Retrieved 3 November 2017, from http://ee-classes.usc.edu/ee459/library/datasheets/LM7805.pdf

[102] *TPS565201 Switching Regulator*. *webench.ti.com*. Retrieved 3 November 2017, from https://webench.ti.com/webench5/power/webench5.cgi?origin=ti_panel&lang_chosen=en_US&VinMin=7&VinMax=8.4&O1V=5&O1I=4&op_TA=30

[103] *Surface Mount Soldering*. (2017). *Ece.ucsb.edu*. Retrieved 3 November 2017, from http://www.ece.ucsb.edu/~long/ece145a/SMSoldering.pdf

[104] *Choosing the Right RC Transmitter Radio for Quadcopter | IndoorFPV.com*. *Indoorfpv.com*. Retrieved 8 November 2017, from http://www.indoorfpv.com/content/choosing-right-rc-transmitter-radio-quadcopter

[105] *Compatible RC Transmitter and Receiver Systems — Copter documentation*. *Ardupilot.org*. Retrieved 10 November 2017, from http://ardupilot.org/copter/docs/common-pixhawk-and-px4-compatible-rc-transmitter-and-receiver-systems.html

[106] *RC radio control protocols explained: PWM, PPM, PCM, SBUS, IBUS DSMX, DSM2*. *DroneTrest*. Retrieved 10 November 2017, from http://www.dronetrest.com/t/rc-radio-control-protocols-explained-pwm-ppm-pcm-sbus-ibus-dsmx-dsm2/1357

[107] *PWM vs. PPM*. Retrieved 10 November 2017, from https://www.rcgroups.com/forums/attachment.php?attachmentid=1173661

[108] *Turnigy 9X Review - 9 Channel RC Transmitter - Oscar Liang*. *Oscar Liang*. Retrieved 16 November 2017, from https://oscarliang.com/turnigy-9x-review-9-channel-rc-transmitter/

[109] *Turnigy 9X 9Ch Transmitter w/ Module & iA8 Receiver (Mode 2) (AFHDS 2A system)*. (2017). *Hobbyking*. Retrieved 16 November 2017, from https://hobbyking.com/en_us/turnigy-9x-9ch-transmitter-w-module-ia8-receiver-mode-2-afdhs-2a-system.html

[110] *PID Tuning Tutorial*. *Expertune.com*. Retrieved 20 November 2017, from http://www.expertune.com/tutor.aspx

[111] *PID for Dummies - Control Solutions*. *Csimn.com*. Retrieved 20 November 2017, from https://www.csimn.com/CSI_pages/PIDforDummies.html

[112] *Advanced Tuning — Copter documentation*. *Ardupilot.org*. Retrieved 28 November 2017, from http://ardupilot.org/copter/docs/tuning.html

[113] *Kalman filters*. *Jeremy.fix.free.fr*. Retrieved 1 December 2017, from http://jeremy.fix.free.fr/Softwares/kalman.html

[114] Flight Cameras. (n.d.). Retrieved December 03, 2017, from http://www.lumenier.com/products/flight-cameras

[115] What is Super HAD CCD camera? (n.d.). Retrieved December 03, 2017, from http://www.cctvco.com:9996/kb/a133/what-is-super-had-ccd-camera.aspx

[116] What is National Television Standards Committee (NTSC)? - Definition from WhatIs.com. (n.d.). Retrieved December 03, 2017, from http://whatis.techtarget.com/definition/National-Television-Standards-Committee-NTSC

[117] E-con Cameras for Drone. (n.d.). Retrieved December 03, 2017, from https://www.e-consystems.com/blog/camera/e-con-cameras-for-drone/

[118] Aptina Introduces 18MP BSI Sensor to Compact DSC Market. (2012, September 18). Retrieved December 03, 2017, from http://www.businesswire.com/news/home/20120918005455/en/Aptina-Introduces-18MP-BSI-Sensor-Compact-DSC

[119] A. (2017, October 04). Overview of the Intel® RealSense™ Depth Camera. Retrieved December 03, 2017, from https://software.intel.com/en-us/realsense/d400

[120] Intel RealSense Depth Camera D435. (n.d.). Retrieved December 3, 2017, from https://simplecore.intel.com/realsensehub/wp-content/uploads/sites/63/D435_ProductBrief.pdf

[121] Intel RealSense D400 Series . (n.d.). Retrieved December 3, 2017, from https://software.intel.com/sites/default/files/Intel_RealSense_Depth_Cam_D400_Series_Datasheet.pdf

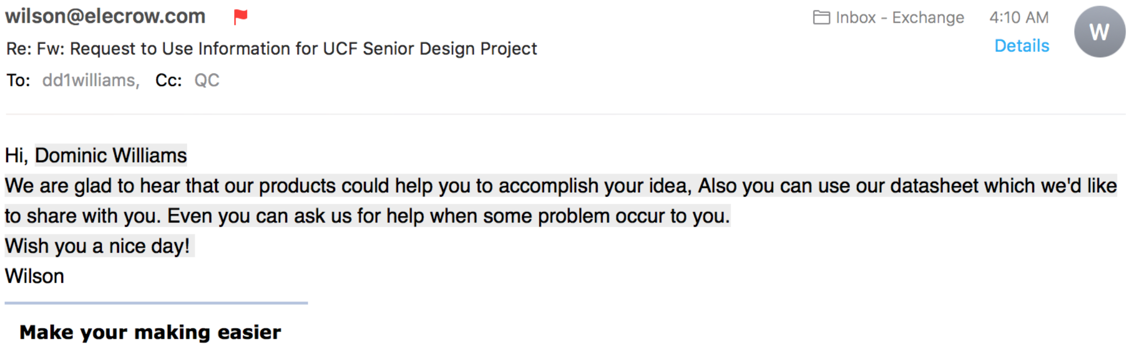
[122] Viola–Jones object detection framework. (2017, November 04). Retrieved December 03, 2017, from https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

[123] Summed-area table. (2017, November 25). Retrieved December 03, 2017, from https://en.wikipedia.org/wiki/Summed-area_table

[124] Girshick, R. (n.d.). Fast R-CNN. Retrieved December 3, 2017, from https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Girshick_Fast_R-CNN_ICCV_2015_paper.pdf

[125] Redmon, J., Divvala, S., & Girshick, R. (n.d.). YouOnlyLookOnce: Unified,Real-TimeObjectDetection. Retrieved December 3, 2017, from https://homes.cs.washington.edu/~ali/papers/YOLO.pdf

Appendix B - Copyright Permissions



wilson@elecrow.com

Re: Fw: Request to Use Information for UCF Senior Design Project

To: dd1williams, Cc: QC

Hi, Dominic Williams

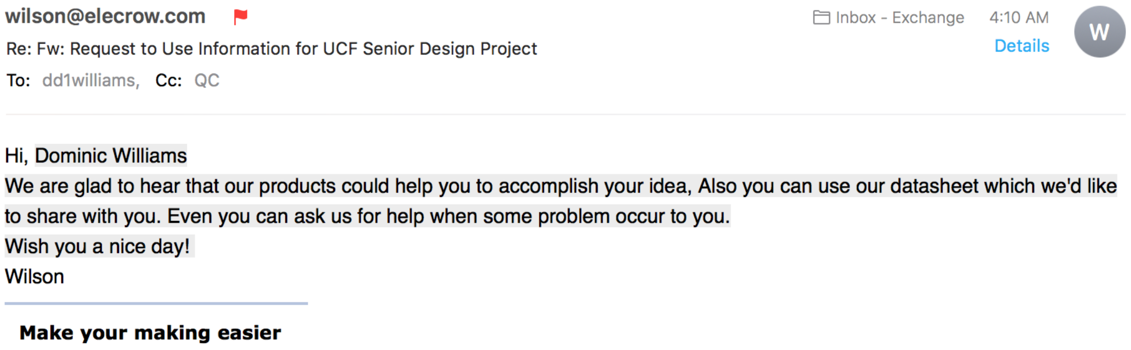
We are glad to hear that our products could help you to accomplish your idea, Also you can use our datasheet which we'd like to share with you. Even you can ask us for help when some problem occur to you.

Wish you a nice day!

Wilson

Make your making easier

Permission to Use: Figure (HC-SR04 Performance)



wilson@elecrow.com

Re: Fw: Request to Use Information for UCF Senior Design Project

To: dd1williams, Cc: QC

Hi, Dominic Williams

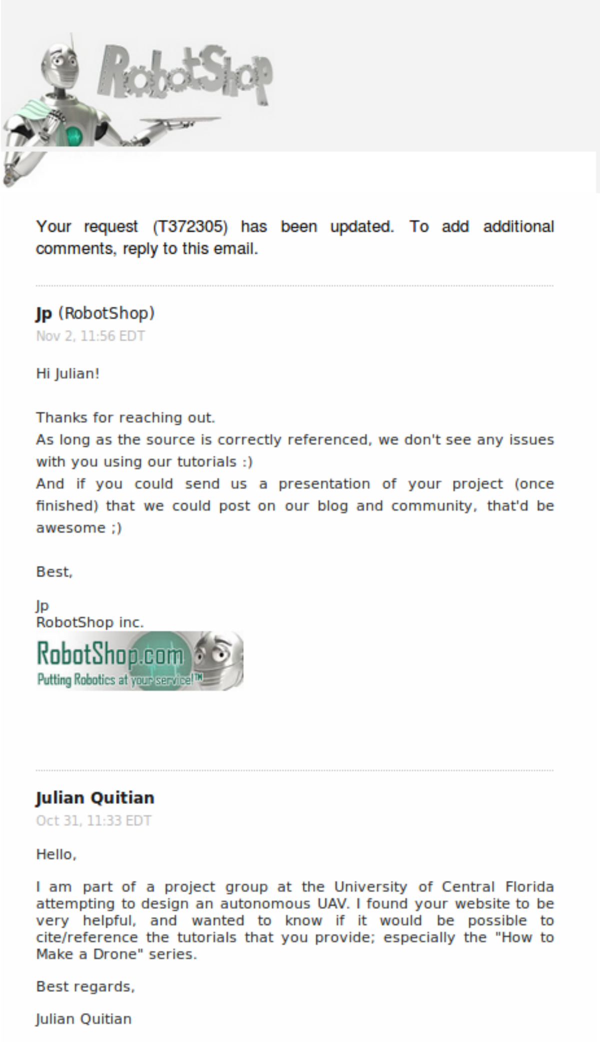
We are glad to hear that our products could help you to accomplish your idea, Also you can use our datasheet which we'd like to share with you. Even you can ask us for help when some problem occur to you.

Wish you a nice day!

Wilson

Make your making easier

[18]



Your request (T372305) has been updated. To add additional comments, reply to this email.

Jp (RobotShop)
Nov 2, 11:56 EDT

Hi Julian!

Thanks for reaching out.
As long as the source is correctly referenced, we don't see any issues with you using our tutorials :)
And if you could send us a presentation of your project (once finished) that we could post on our blog and community, that'd be awesome ;)

Best,

Jp
RobotShop inc.

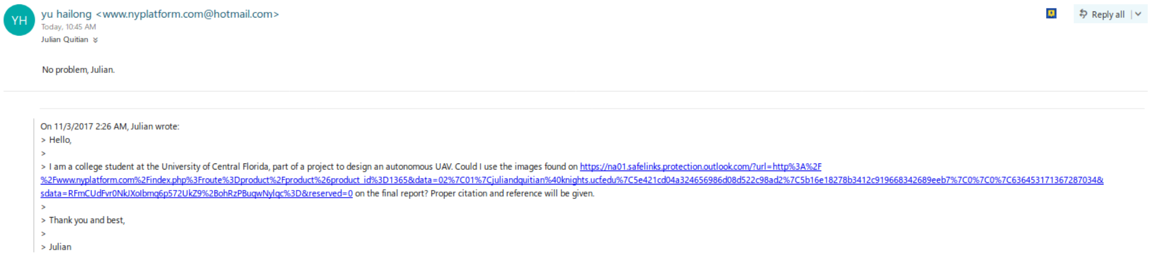
Julian Quitian
Oct 31, 11:33 EDT

Hello,

I am part of a project group at the University of Central Florida attempting to design an autonomous UAV. I found your website to be very helpful, and wanted to know if it would be possible to cite/reference the tutorials that you provide; especially the "How to Make a Drone" series.

Best regards,

Julian Quitian

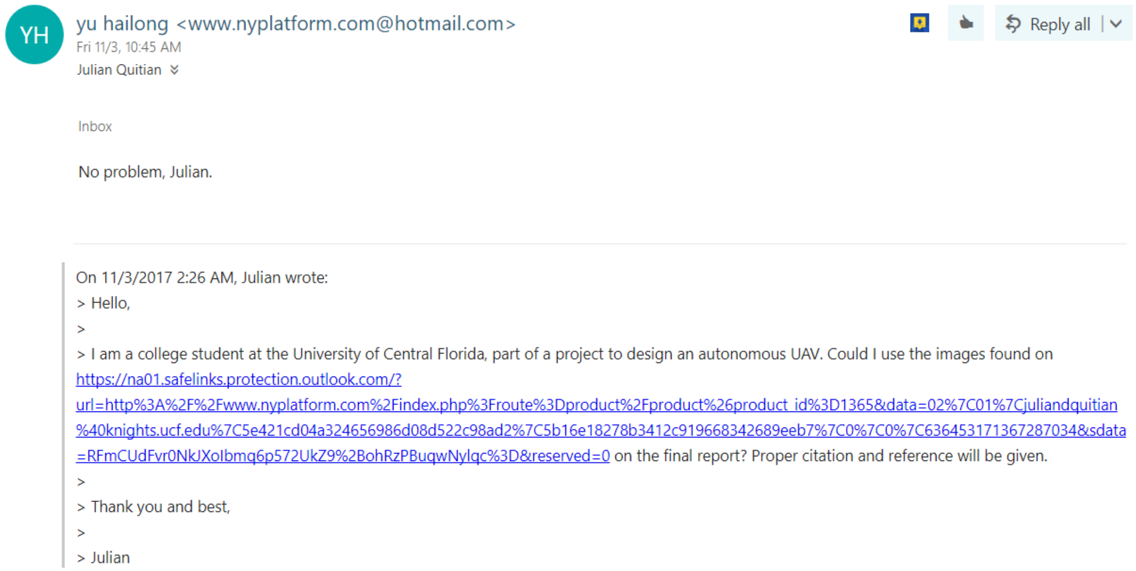


yu hailong <www.nyplatform.com@hotmail.com>
Today, 10:45 AM
Julian Quitian

No problem, Julian.

On 11/3/2017 2:26 AM, Julian wrote:
> Hello,
>
> I am a college student at the University of Central Florida, part of a project to design an autonomous UAV. Could I use the images found on https://na01.safelinks.protection.outlook.com/?url=http%3A%2F%2Fwww.nyplatform.com%2Findex.php%3Froute%3Dproduct%2Fproduct%26product_id%3D1365&data=02%7C01%7Cjuliandquitian%40knights.ucfedu%7C5e421cd04a324656986d08d522c98ad2%7C5b16e18278b3412c919668342689eeb7%7C0%7C0%7C636453171367287034&sdata=RFmCUdFvr0NkJXolbmq6p572UkZ9%2BohRzPBuqwNylqc%3D&reserved=0 on the final report? Proper citation and reference will be given.
>
> Thank you and best,
>
> Julian

[35]

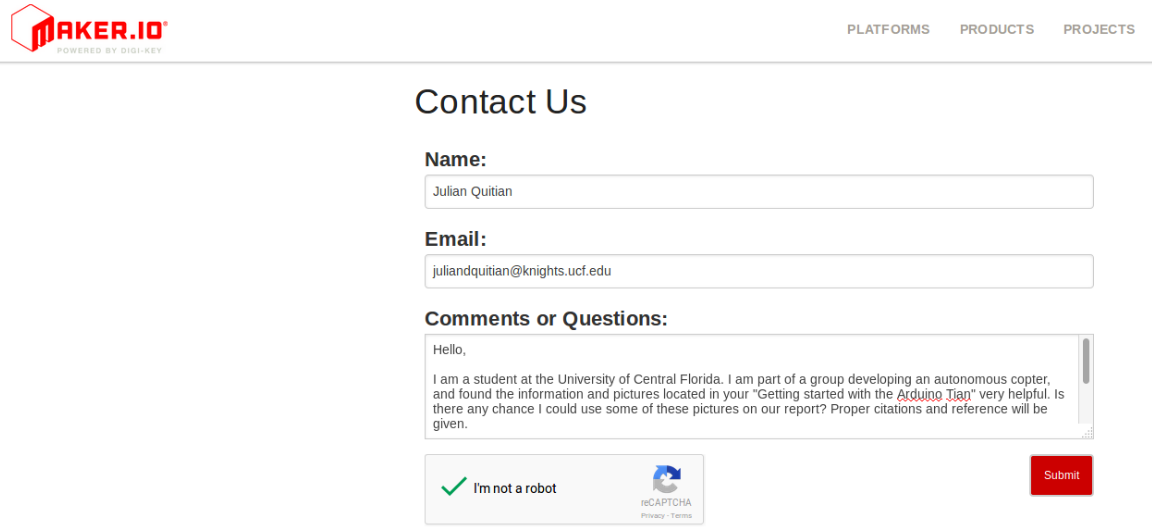


yu hailong <www.nyplatform.com@hotmail.com>
Fri 11/3, 10:45 AM
Julian Quitian

No problem, Julian.

On 11/3/2017 2:26 AM, Julian wrote:
> Hello,
>
> I am a college student at the University of Central Florida, part of a project to design an autonomous UAV. Could I use the images found on https://na01.safelinks.protection.outlook.com/?url=http%3A%2F%2Fwww.nyplatform.com%2Findex.php%3Froute%3Dproduct%2Fproduct%26product_id%3D1365&data=02%7C01%7Cjuliandquitian%40knights.ucf.edu%7C5e421cd04a324656986d08d522c98ad2%7C5b16e18278b3412c919668342689eeb7%7C0%7C0%7C636453171367287034&sdata=RFmCUdFvr0NkJXoIbmq6p572UkZ9%2BohRzPBuqwNylqc%3D&reserved=0 on the final report? Proper citation and reference will be given.
>
> Thank you and best,
>
> Julian

[38]



Contact Us

Name: Julian Quitian

Email: juliandquitian@knights.ucf.edu

Comments or Questions:

Hello,

I am a student at the University of Central Florida. I am part of a group developing an autonomous copter, and found the information and pictures located in your "Getting started with the Arduino Tian" very helpful. Is there any chance I could use some of these pictures on our report? Proper citations and reference will be given.

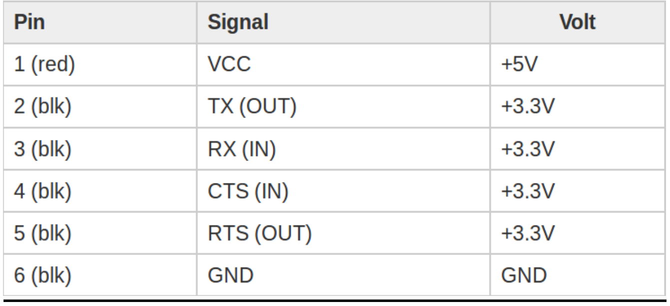
Appendix C - Dronecode Connector Standards

Dronecode is the open-source UAV platform from where most of our UAV's components obtained their design specifications. This section lists the connector standards which define the design of the RoboCopter's components including the ESCs, flight controller and motor [37].

Pinouts

<u>Telemetry Port</u>

______This pinout should be used for any serial port. Ports not supporting hardware flow control should leave the CTS and RTS lines floating. Cables should be 1:1 to the peripheral and TX / RX as well as CTS / RTS should be crossed on the peripheral side (e.g. a radio modem would have its RX port on pin 2, TX port on pin 3, RTS on port 4 and CTS on port 5).

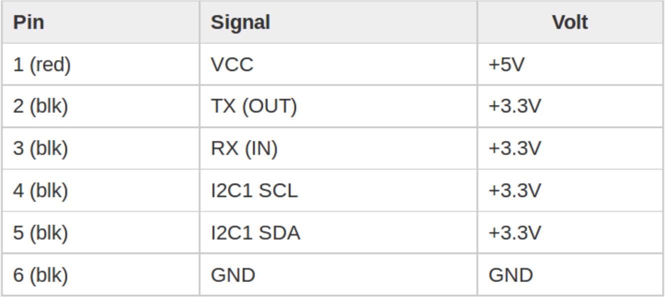


Pin	Signal	Volt
1 (red)	VCC	+5V
2 (blk)	TX (OUT)	+3.3V
3 (blk)	RX (IN)	+3.3V
4 (blk)	CTS (IN)	+3.3V
5 (blk)	RTS (OUT)	+3.3V
6 (blk)	GND	GND

Telemetry Pinout

GPS Port

This port is intended for combined serial plus I2C GPS units.

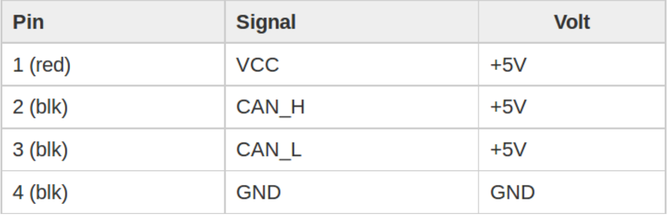


Pin	Signal	Volt
1 (red)	VCC	+5V
2 (blk)	TX (OUT)	+3.3V
3 (blk)	RX (IN)	+3.3V
4 (blk)	I2C1 SCL	+3.3V
5 (blk)	I2C1 SDA	+3.3V
6 (blk)	GND	GND

GPS Pinout

CAN

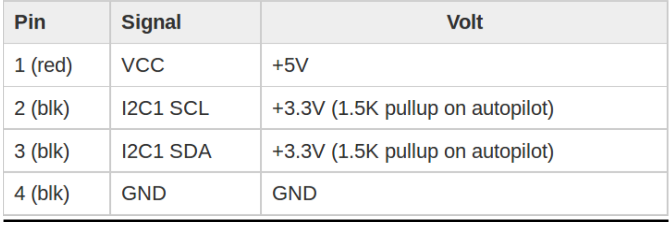
Recommended CAN transceivers: TJA1051TK3/118 or LTC2875



Pin	Signal	Volt
1 (red)	VCC	+5V
2 (blk)	CAN_H	+5V
3 (blk)	CAN_L	+5V
4 (blk)	GND	GND

CAN Pinout

I2C Port

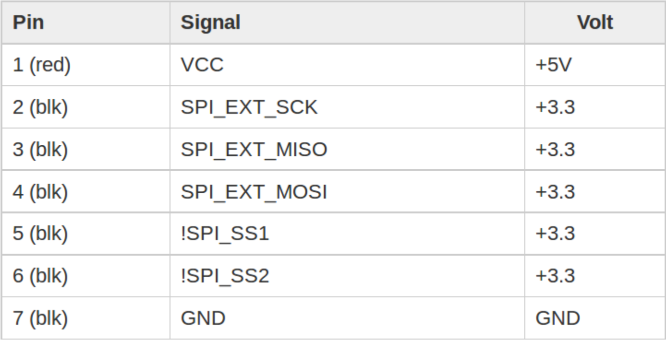


Pin	Signal	Volt
1 (red)	VCC	+5V
2 (blk)	I2C1 SCL	+3.3V (1.5K pullup on autopilot)
3 (blk)	I2C1 SDA	+3.3V (1.5K pullup on autopilot)
4 (blk)	GND	GND

I2C Pinout

SPI Port

This port is optional and for external SPI sensors.

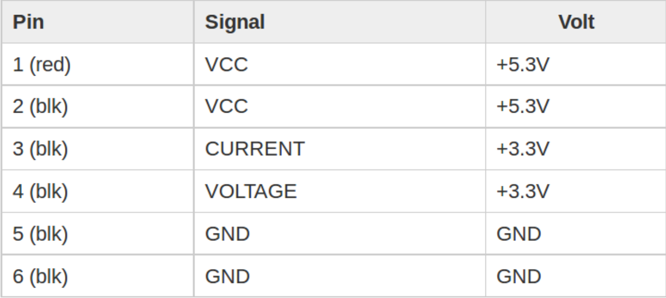


Pin	Signal	Volt
1 (red)	VCC	+5V
2 (blk)	SPI_EXT_SCK	+3.3
3 (blk)	SPI_EXT_MISO	+3.3
4 (blk)	SPI_EXT_MOSI	+3.3
5 (blk)	!SPI_SS1	+3.3
6 (blk)	!SPI_SS2	+3.3
7 (blk)	GND	GND

SPI Pinout

Analog Power

The CURRENT signal should carry an analog voltage from 0-3.3V for 0-60A as default. For high-power units the range should be 0-3.3V for 0-120A. The VOLTAGE signal should carry an analog voltage from 0-3.3V for 0-50A as default. The VCC lines have to offer at least 2.5A continuous and should default to 5.3V. A lower voltage of 5V is still acceptable, but discouraged.



Pin	Signal	Volt
1 (red)	VCC	+5.3V
2 (blk)	VCC	+5.3V
3 (blk)	CURRENT	+3.3V
4 (blk)	VOLTAGE	+3.3V
5 (blk)	GND	GND
6 (blk)	GND	GND

Analog Power Pinout

Appendix D - PCB Components