California State Polytechnic University, Pomona

AUVSI Team

2018 Student UAS Competition

Technical Journal Paper

AUVSI Team Composition:

Team Lead: Alexander Rey
Department of Aerospace Engineering

Daniel Bannowsky, Michael Doan, Thomas Fergus, Christopher Hamilton, Andrew Moffatt, Mateus Pinheiro, Bryce Satterfield, Tristan Sherman
Department of Aerospace Engineering

Ronell Lim
Department of Mechanical Engineering

Dr. Subodh Bhandari
Faculty Adviser

Abstract

With this being the eighth year that Cal Poly Pomona has participated in AUVSI's SUAS competition, the successes and failures of previous architectures were leveraged to design this year's platform. Building off these, this year, the team will be attempting waypoint navigation, search area, air delivery, interoperability, and obstacle avoidance. In addition, a new airframe was chosen, a twin boom inverted V-tail aircraft. A new camera system and payload drop system were designed and built to improve the architecture's image recognition and air delivery tasks. Multiple flight tests were conducted on this platform to prove the performance of all the system elements, giving the team assurance that the system will successfully perform at this year's competition.

Table of Contents

Abstract 1
Table of Contents 2
1. System Engineering Approach 3
1.1 Mission Requirements Analysis 3
1.2 Design Rationale 4
2. System Design 4
2.1 Aircraft Design 4
2.1.1 Airframe 4
2.1.2 Power System 5
2.2 Autopilot 6
2.3 Obstacle Avoidance 7
2.3.1 Stationary Obstacle Avoidance 7
2.3.2 Moving Obstacle Avoidance 7
2.4 Imaging system 9
2.4.1 Camera 9
2.4.2 Camera Gimbal 9
2.5 Object Detection, Classification, Localization 9
2.5.1 Machine Learning Process 10
2.5.2 Identifying Images 10
2.5.3 GPS Localization 11
2.6 Communications 11
2.6.1 RF Transmitter Design 11
2.6.2 Radio Frequencies 11
2.6.3 Antenna Selection 11
2.6.4 Ground Control Station 12
2.6.5 Mission Planner computer 12
2.6.6 Telemetry Processing 13
2.6.7 Interoperability 13
2.7 Air Delivery 14
2.7.1 Air Delivery Mechanism 14
2.7.2 Air Delivery Software 14
2.8 Cyber Security 14
3. Safety 14
3.1 Developmental Risks and Mitigations 14
3.2 Mission Risks and Mitigations 15
4. Acknowledgements 15
References: 15

1. System Engineering Approach

1.1 Mission Requirements Analysis

The starting point for Cal Poly Pomona's UAS design was to first dissect the mission objectives and their corresponding requirements to determine the system level requirements. A detailed understanding of the system level requirements is critical as it allows for the determination of the tradeoffs and complexity of each task. These elements were looked at with respect to their corresponding point value to determine their priority in the overall design of the UAS. Table 1-1 shows the breakdown of a few of the higher point-value mission objectives.

Table 1-1: System level requirements of the AUVSI-SUAS Competition.

Mission Objective	Points	Objective Requirements (for full points)	System Requirements	Considerations and Trade Offs	Complexity
Autonomous Flight	12	Min. 3 minutes of automatic flight	Tune autopilot for platform	Maneuverability versus reliability	low
		No pilot takeovers			med
		Auto takeoff/landing	Reliable takeoff and landing sequence	Auto takeoff and landing is a potential failure point	high
Waypoint Capture	3	Fly within 100 ft of waypoints	GPS navigation on platform	Mass of onboard flight controller	low
Waypoint Accuracy	15	Acc. of waypoint nav. w/in 100 ft	Ability to handle wind	Turn radius vs lower air speed	high
			Vehicle turn radius		
		Valid telemetry at 1 Hz	Receive TLM at 1 Hz w/in 1 mi. radius	Power draw of TLM system	med
Stationary Obs. Avoidance	10	Avoid cylinder with 30-300ft radius and 30-750ft height	Customizable flight paths based on interoperability data	Non-linear flight paths vs efficient search pattern	med
Moving Obs. Avoidance	10	Avoid sphere with radius 30-300ft @ 0-40KIAS	Flight path able to be automatically be customized from live interoperability data	Sudden changes in flight pattern could result in autopilot failure	high
Target Characteristic	4	shape, shape color, alphanumeric, alphanumeric color, orientations	High res. camera/lens	Weight (flight time, agility)	low
			Attach cam to gimbal		low
			Onboard data mgmt.		high
Target Geolocation	6	Acc. of target w/in 150 ft	Integrate imaging & loc/TLM orientation	Transmission bandwidth usage	high
Target Actionable	6	Submit via USB	Save target on USB	Accuracy of target detection	low
		Submit via Interop.	Send target w/ Interop.		high
Target Autonomy	4	Submit targets w/o manual cmd	Program to autodetect targets	False negative occurrence	high
Air delivery	10	Acc. of target w/in 150 ft	Drop mechanism	Weight	low
			Drop prediction code	Precision vs acc.	med

1.2 Design Rationale

For a majority of the team, this will be their first year competing in the AUVSI-SUAS competition. Only two members have previous experience working with unmanned aerial vehicles. As such, the vehicle platform would have to be simple enough for new members to handle. Budget was not a major limiting issue with the project being funded by Cal Poly Pomona's aerospace engineering department.

Last year's team chose to go with a multirotor platform due to its predictability in waypoint navigation, its compatibility with our autopilot system, and the small amount of time the team would need to set it up. However, it was found that there were two main issues with using this type of aircraft. The first problem was related to endurance. Multirotors require more power than a fixed-wing aircraft, leading to a shorter flight time. This shorter flight time is not enough to finish all of the required tasks. The second problem was related to telemetry. Multirotor platforms typically do not have a lot of mounting space. This led to the telemetry modules for all of the systems being too close to each other. During the competition last year, the vehicle was unable to switch into autopilot due to telemetry issues. Because of these issues, multirotor platforms were taken out of consideration.

Going back to a fixed-wing aircraft, the team considered configurations that were used by the team in previous competitions. The Hangar 9 Valiant was assessed and taken out of consideration after consulting the previous team lead. Reasons for disregarding the Valiant platform included increased maintenance for the gasoline-powered configuration and inadequate flight time for the electric-powered configuration. Because of these reasons, the team had to find a new vehicle to fit the mission requirements. To save time, the team decided to buy a commercially available airframe. Configurations would be limited to what is currently on the market, but it would allow time for work to be done in other areas. The first major design decision was choosing between gas or electric propulsion. Electric propulsion was desirable for the lack of required maintenance and its reliance on chargeable energy instead of fuel. The second major design decision was choosing between a high-wing and low-wing design. The high-wing design was favorable for its natural stability. Lastly, the interior of the fuselage would need to be sufficiently large enough to accommodate the necessary hardware. It was found that the Ready-Made RC (RMRC) Anaconda met these characteristics and was the ideal platform for the competition.

2. System Design

2.1 Aircraft Design

2.1.1 Airframe

The airframe of the aircraft utilizes a modified RMRC Anaconda airframe, shown in Figure 2-1. The Anaconda is a twin boom, inverted V-tail aircraft that utilizes an electric pusher powerplant. This model was chosen over other fixed-wing vehicles as its smooth platform minimizes the vibrations that are experienced during flight. The airframe is primarily Styrofoam with the supporting components made from wood. The Styrofoam body allows for a light airframe, but it can be easily scratched or dented. Modification to the airframe was done by 3D-printing parts with PLA plastic, as opposed to using wood. This choice was made as 3D printed parts can be quickly prototyped, tested, and verified better than wooden parts. The faster manufacturing time lead us to use 3D printed parts over other alternatives. Examples of these 3D-printed parts include the Pixhawk mount, air-delivery mechanism support, and camera gimbal arms. The air-delivery system is mounted under the aircraft's center of gravity to minimize CG travel after releasing the water bottle. Similarly, the camera gimbal is mounted near the aircraft's CG to reduce the moment generated by the weight of the gimbal. The landing gear struts are made from aluminum and do not retract. The aircraft can also be easily disassembled and reassembled, making transportation easier. Modifications made to the fuselage include the air-deliver mechanism, camera gimbal, and longer landing struts. Important dimensions for the airframe are shown in Table 2-1 [1].

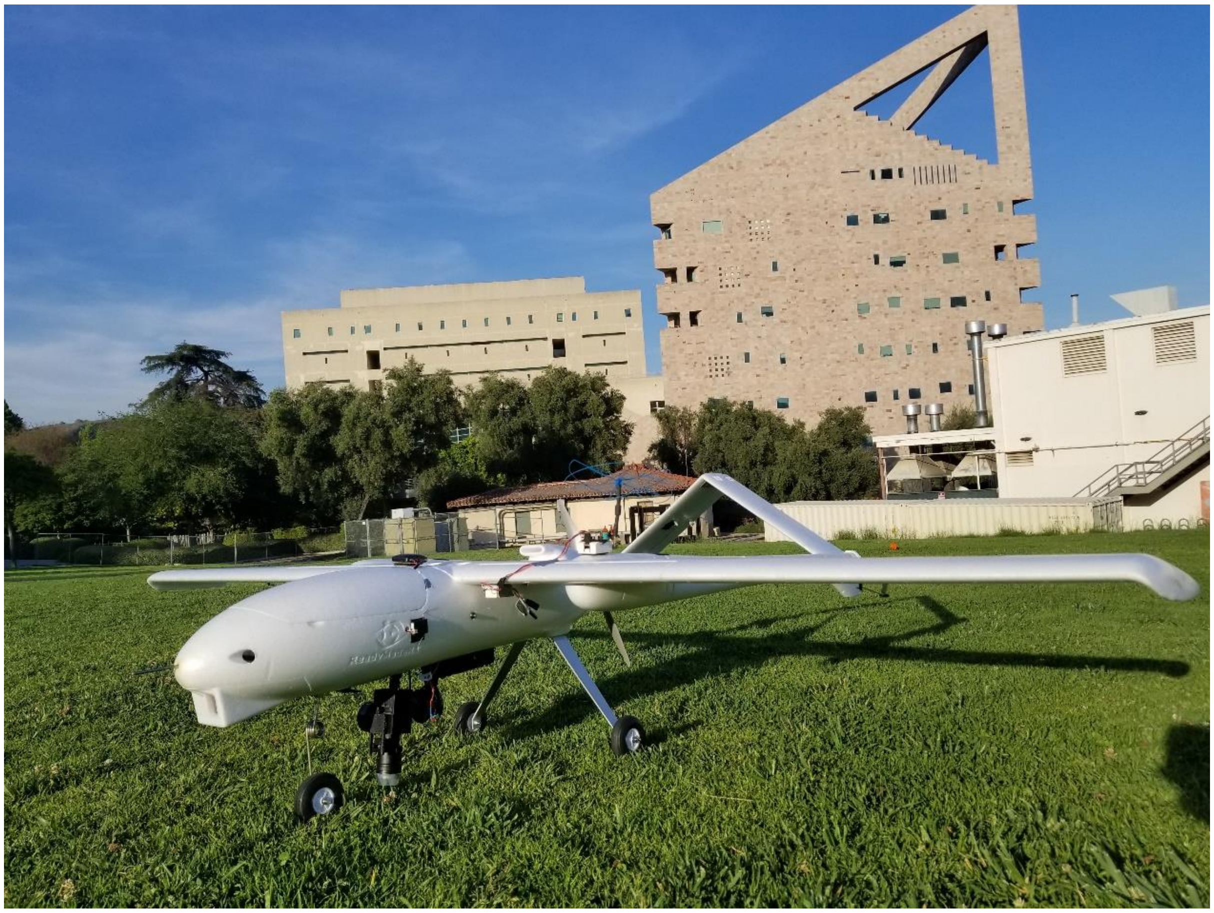


Figure 2-1: Picture of the RMRC Anaconda UAS that will be used in AUVSI-SUAS competition.

Table 2-1: Parameters of the RMRC Anaconda that will be used in the AUVSI-SUAS competition.

Parameter	Value
Wingspan	6.76 ft.
Wing Area	5.27 $ft.^2$
Maximum Take-Off Weight	12.0 lb.
Length	4.63 ft.
Propeller Size	14x7
Max. Interior Fuselage Length	26.5 in.
Max. Interior Fuselage Width	6 in.
Max. Interior Fuselage Height	3 in.
Max. Distance Between Fuselage and Ground	7.5 in.

2.1.2 Power System

The design of the power system for the vehicle is focused on accommodating the Cobra C-3520/12 brushless motor, Pixhawk autopilot system, camera gimbal, and imaging system. Three 2200 mAh 3S 25c lithium polymer batteries were used to power the Pixhawk, camera gimbal, and imaging system separately. These 3S lithium polymer batteries were selected as they were compact enough to fit within the limited space of the Anaconda’s fuselage, while still providing sufficient endurance to meet our flight time. For the main engine, two 5200 mAh 4S 35c lithium polymer batteries were used in parallel. Similar to the previous batteries, space is the limiting issue. These 4S lithium polymer batteries were selected over competitors for their higher power density while still fitting inside the aircraft. It was decided to power these subsystems individually, instead of together, to limit potential threats from battery failures.

2.2 Autopilot

The autopilot subsystem primarily utilizes a 3DR Pixhawk with Ardupilot software and is shown in Figure 2-2. Based on the team's experience with the Pixhawk in previous years, it has been shown to meet the system's needs. Despite other autopilots being available, such as the APM 2.6, the 3DR Pixhawk was chosen for its waypoint navigation capability, and the ease of modification with its open source code. It was also determined that the necessary time needed to learn another autopilot could be better allocated towards other aspects of the project, as time is a limiting variable for the development of the platform. In addition, the aircraft is equipped with a GPS receiver and a 915 MHz radio and antenna to transmit telemetry to the Ground Control Station (GCS). The GCS uses a dedicated laptop running the Mission Planner software for writing waypoints to the aircraft. For safety purposes, a 2.4 GHz radio is used to allow the safety pilot to take over the aircraft at any time.

With the attached GPS, compass, and magnetometer units on the airframe, the Pixhawk is capable of precise waypoint navigation through actuation of the aircraft's control surfaces. Due to this year's aircraft utilizing an inverted V-tail, the simultaneous actuation of both servos is needed to control yaw and pitch. This complication was handled through use of a Y-split cable attached to the Pixhawk. In addition to the control surfaces, the Pixhawk also has control over the bottle drop servo, the telemetry downlink to the ground station, and can function under various flight modes. The flight modes include manual, stabilize, and automatic. manual allows the pilot to use the Pixhawk as a receiver for flight commands and provides no automation. stabilize is an enhanced form of manual in which the Pixhawk attempts to correct to steady state flight in the absence of pilot input. automatic gives the Pixhawk gives complete authority over flight controls.

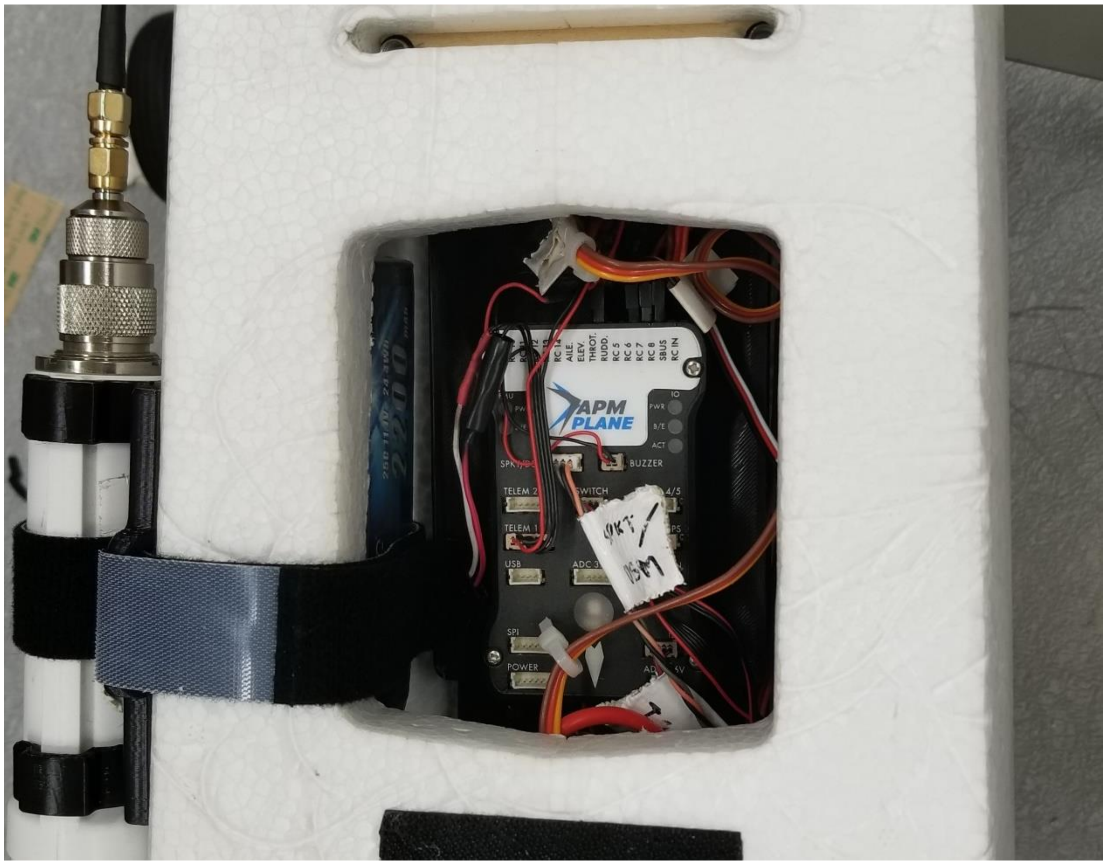


Figure 2-2: Picture of the 3DR Pixhawk Autopilot attached to the RMRC Anaconda.

2.3 Obstacle Avoidance

2.3.1 Stationary Obstacle Avoidance

To avoid the stationary objects, the obstacle avoidance algorithm takes the initial list of desired waypoints and the list of stationary objects and checks for certain conditions. First, the algorithm iterates through the list of waypoints and ensures that the path between any single waypoint and its following waypoint does not intersect with any of the stationary obstacles. In the event that the path puts the UAS on a collision trajectory, a set of waypoints to path around the obstacle is generated. This set of waypoints is then checked to make sure that the UAS does not run into another obstacle or goes outside the competition's boundary area. A visualization of this process is shown in Figure 2-3.

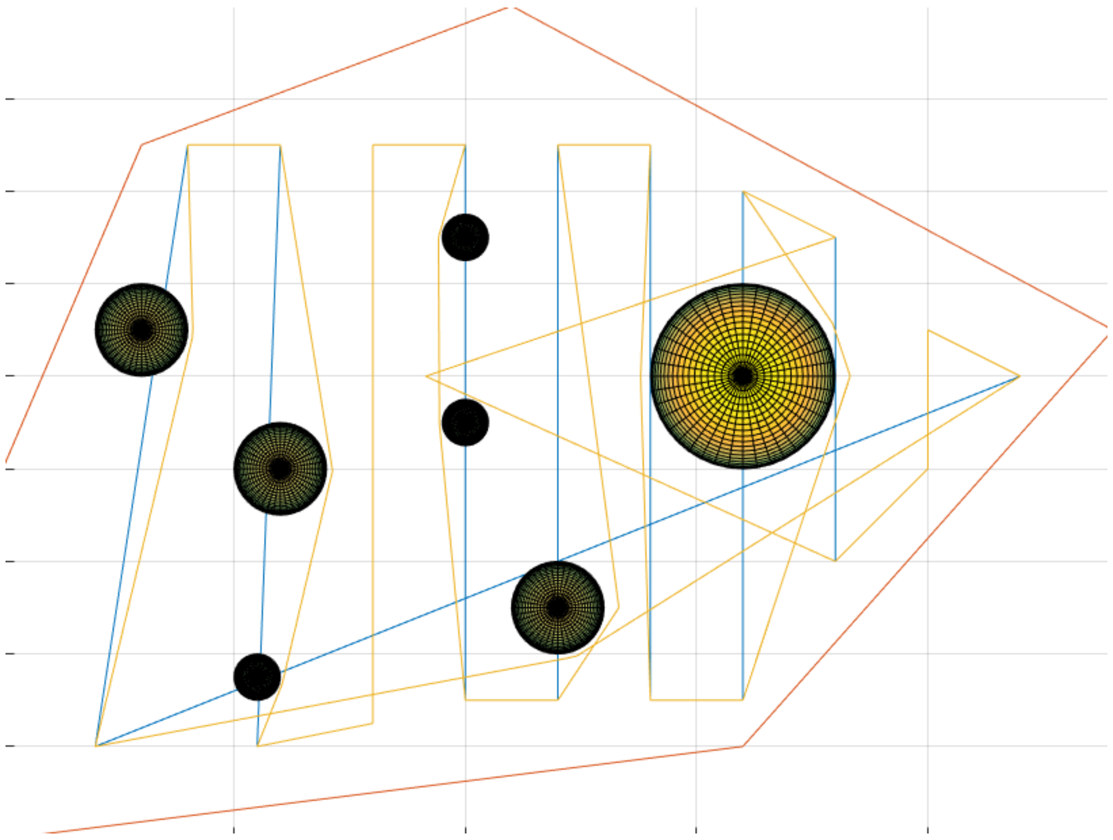


Figure 2-3: Visualization of Initial Waypoint Path (Blue) and Corrected Waypoint Path (Yellow).

2.3.2 Moving Obstacle Avoidance

To handle the dynamic flight path needed to avoid the moving obstacles, the Unity game engine was used. While unorthodox, the idea surfaced because AI in video games frequently need to determine the optimal route between two points while remaining within the bounds of the virtual map. This process is known as pathfinding. This same process of pathfinding can be used in this competition to accomplish the moving obstacle avoidance needs.

A trade study between using the Unity game engine and an independently developed code was done. From this, it was determined that the advantages of its ease of use, automatic visualization, and ability to be developed and tested in time for competition outweighed the overhead needed to run the engine.

The way the pathing works is at its core a single calculation. Below, in Figure 2-4, is a screen capture of what the virtual map looks like. White represents the drone, red spheres are the obstacles, green is the waypoint for the drone to follow, and gold is the end waypoint set by the team.

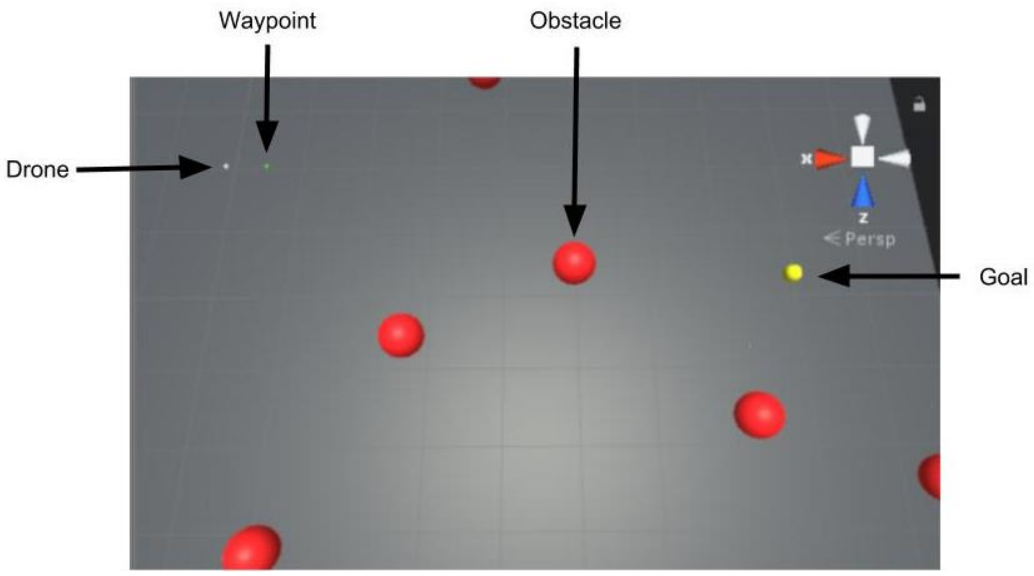


Figure 2-4: Screenshot of the unity engine virtual field with the UAS and obstacles.

To get to the goal, every frame the ground control station uses the built in Unity functions to orient the aircraft towards the gold ball. To avoid obstacles, an angular displacement is added to the orientation, as shown in Figure 2-5. This displacement is dependent on the distance every obstacle is away from the drone and the capabilities of the platform. As a red ball gets closer to the vehicle, the code increases how much the vehicles turns to avoid it. It then increments forward a set distance based on the platforms steady state speed. In actual flight, the movement is replaced by the current GPS location of the drone during that frame.

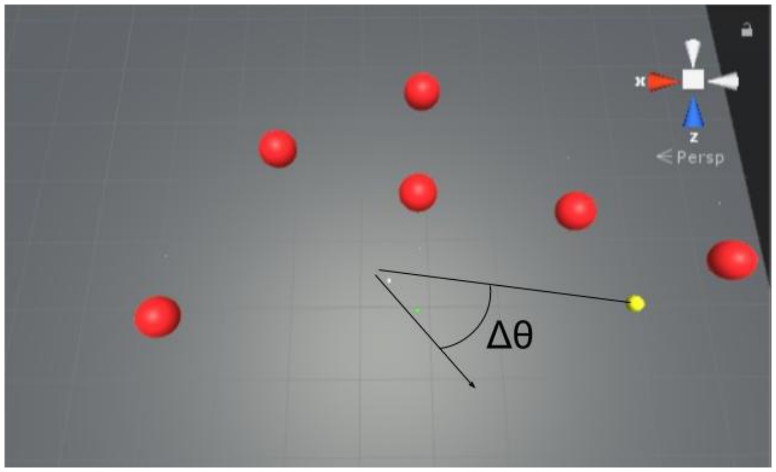


Figure 2-5: Unity waypoint navigation with maximum heading change angle.

The green waypoint is a child object of the aircraft that is dependent to its instantaneous position and orientation. This means that when the drone faces a direction, this green waypoint will always be in front of it and within the specified acceptable angular range. This green waypoint position is what is being passed to the real vehicle's flight path, whereas, the white sphere is what is being simulated in the program as the current vehicle's position.

To test the validity of this method, a simulation where the red obstacles were given random positions and headings was used. In this, it was gleaned that this method of pathfinding works for a vehicle, such as a plane, that needs to maintain a constant speed. For actual flight, each obstacle will be updated with a radius and position every frame by gathered from the interoperability server.

This method offers simplicity, by keeping the length of code to a minimum. However, drawbacks to using a reactionary based code is no predictive capability of the drone. In other words, an obstacle may be going in a simple circle pattern, but, the program would not recognize this and may take a slower route as a result. However, from the simulations, this situation is expected to be a nonissue, as the amount of lost time is generally minimal. For this reason, it was decided that a pure reaction, nonpredictive, based approach was taken.

Below is a list of special cases and necessary future developments that should be taken into consideration.

- When an obstacle's direction of motion is opposite to the side of the drone it is on, the drone would take the long path around the obstacle by attempting to outrun it. The solution was to make an invisible object that rested in front of each obstacle similar to the green waypoint on the drone. If the invisible object and the obstacle were on opposite sides of the drone, it would reverse the direction of angular displacement.
- Airfield boundaries need to be avoided so the edges are treated as obstacles too.
- Fine tuning of values such as sensitivity to obstacles.

2.4 Imaging system

2.4.1 Camera

A new camera had to be selected due to the switch in vehicle platforms. In previous years, a FLEA3 camera and a modified GoPro camera were used. The FLEA3 camera was not chosen due to its low resolution and its inability to produce large enough target images at the cruising altitude. The modified GoPro was a major success with last year's vehicle, but it was too large and heavy to integrate with the new vehicle. An increase in funding allowed the team to explore options that were previously too expensive to pursue. The chosen camera was the Grasshopper3 9.1 MP Color GigE Vision from Point Grey. When paired with the Fujinon YV3.3x15SA-2 CS mount lens, the camera provides high quality imaging with a small, lightweight body. A 15-50 mm lens is used to allow for adjustments to the field of view without having to change the flight altitude. The camera is capable of capturing video at a resolution of 1920x1080 with a frame rate of 16 frames per second. Power is provided to the camera through a GPIO cable connected to a 2200 mAh 3S 25c lithium polymer battery. Imaging data is sent through a CAT6 (1-Gbps) cable to an M5 Bullet (80-Mbps at 5-GHz), which connects to another M5 Bullet on the ground station. Lastly, the ground station's M5 Bullet connects to the image processing computer through a CAT6 cable. Live footage can be seen through Point Grey's FlyCap software.

2.4.2 Camera Gimbal

The two-axis gimbal from last year's platform was reused with slight modifications to accommodate the new vehicle. Since last year's system was proven to work, adapting it minimizes the development time needed to learn and integrate a new system. The previous configuration featured a microcontroller from BaseCam Electronics, a carbon fiber body, and brushless motors to stabilize the camera and lens. The carbon fiber parts were replaced with 3D-printed parts for easier integration with the vehicle and to decrease the overall size of the gimbal. The gimbal has the capability of rotating approximately ±30° for roll and ±30° for pitch, which is enough to fulfill its purpose of keeping the camera pointing normal to the ground when the vehicle is banking or changing altitude above a target.

2.5 Object Detection, Classification, Localization

This section discusses what was done to identify, classify, and store the target parameters and location. Each target has a shape and letter of varying colors and a GPS location that needs to be identified and stored after being detected from a video feed being streamed to the ground station from the airplane.

Two options were considered for this task: OpenCV2 color filtering with shape contour and neural network based identification. The OpenCV2 was the first option considered. It works by looking at each frame from the video stream and filters out the grass such that the only thing left on the picture is the target. Then a contour analysis is ran to identify the shape, letter, and colors. This process is quick and requires a small amount of code that can be run without using too much processing power, however, it comes with a number of disadvantages. First, the contours are mathematically difficult to interpret and sort into shapes. Secondly, letters and color filtering depend on multiple preset variables that need to be set on the day of, depending on the field, making it not very adaptable. The image filtering is also prone to errors, resulting in low confidence that images and shapes were correctly identified. This caused other options to be considered.

The method used for this neural network based identification works by sampling the video stream and passing it through a blob detection algorithm on the sampled frame that finds any potential anomalies. These anomalies are then fed to the neural network which has been trained to identify all the shapes, letters, and colors possible in the SUAS competition. This process has the high accuracy percentage that was desired and works in any field condition without the need to set variables or needing to adhere to a strict set of mathematical definitions for shapes and letters. This method has its disadvantages too. The process is slower and requires the video stream to be sampled rather than being able to use every frame directly. It also requires a hefty amount of processing power and training the machine learning software requires thousands of pictures to ensure high accuracy, which can be difficult to provide and requires more time to set up.

Despite the difficulty in setting up a neural network, it was chosen as the best method to identify the targets. Even with the limited time in developing this software, the accuracy it provides was required to meet the system's needs.

2.5.1 Machine Learning Process

In order to teach the machine learning software, it needed to be fed a high magnitude of images, which are organized in folders that categorize what the images are. there was no way to find or take enough images to properly train the machine, so the team decided to procedurally generate training images using python OpenCV2. A script was created that takes an input of a desired field image, a shape template image, a letter template image, a desired letter color, and a desired shape color. It then works by loading the desired shape and letter templates, changing the white color on the template to a desired color, and then "pasting" all the images together to form the final generated field, as shown in Figure 2-6. It is worth mentioning that colors are defined as ranges of RGB values in the code. When receiving a color input, it will output a random value that fits within the definition of the color, ensuring that the machine sees a vast array of colors. This function was then placed in a family of nested loops that generates every possible permutation of field, shape, letter, letter color, and shape color, and saves these images in folders based on the letter and shape.

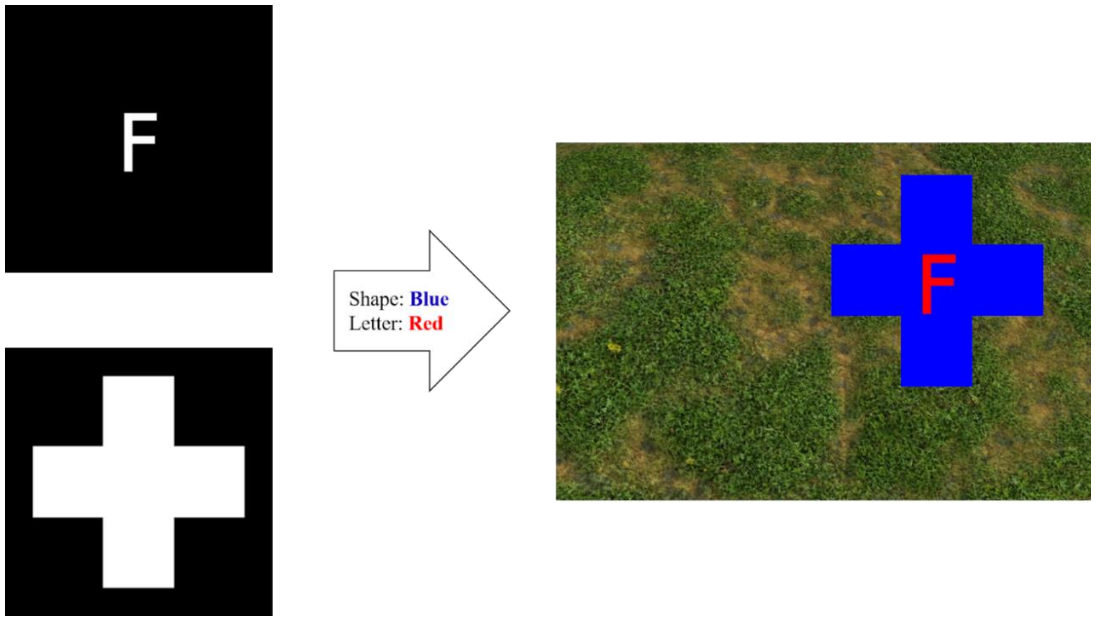


Figure 2-6: Example input and output of image generation script

In the example shown in Figure 2-6, the image generated would be saved in both the "cross" folder and the "F" folder. With a total of 5 fields, 14 shapes, 26 letters, and 13 colors, and with the added constraint of not allowing the shape and letter to be the same color, a total of 283,920 training images were generated with 20,280 images per shape and 10,920 images per letter

2.5.2 Identifying Images

The system utilizes a step by step process to attempt the detection of an object within the cameras view. As the vehicle surveys the field the camera feeds frames into a blob detection algorithm where areas of interest are identified. From there a retrained neural network is ran on the areas of interest to determine the characteristics of the object or if it is not an area of interest at all. Finally, the requested information is sent through our interoperability code to the judges.

Using OpenCV2's python wrapper's built-in blob detection functions the areas of interest on each image are identified. For our purposes the Maximally Stable Extremal Regions (MSER) blob detection functions were chosen, as they gave the best results for the images we were testing with. A python function then returns the pixel location of the frame that the MSER blob detection found.

For classifying the pixel locations returned by the blob detection function, the final layer of the inception model was retrained using the procedurally generated images mentioned in an earlier 2.5.1. From these images, 300 random images were selected for each shape and letter and used to train two separate neural networks; one for shapes, and one for letters. We feed the blob detection function straight into both neural networks and return the shape and letter for the given pixel location.

2.5.3 GPS Localization

If a target is identified, the pixel location of the target on the image is sent back to a code that determines the GPS location. This code trigonometrically uses the field of view of the camera, the height of the airplane, the heading of the plane, the GPS position of the airplane and the pixel location of the shape to determine how far the target is relative to the ground directly beneath the airplane. This relative distance is then converted into a change in latitude and longitude which are offset from the known position of the plane at the time of the photo. The GPS location of the aircraft is determined by correlating the time of the shot to a log of all time ordered telemetry saved on a shared folder that is discussed further in 2.6.6.

2.6 Communications

2.6.1 RF Transmitter Design

The 3DR telemetry module that was used for last year's competition was selected for its ease of use and its proven reliability to meet the system's needs. In addition, the Ubiquiti Bullet M5 wireless radio was selected to transmit the live video stream from the vehicle to the ground. The M5 radios were chosen because they can support a connection up to 50 km away with a rate up to 100 Mbps and remains compact enough to fit in the vehicle [2]. This performance allows us to meet the image processing needs while staying within the constraints of the selected platform.

2.6.2 Radio Frequencies

The UAS has three radio frequency (RF) sources for its data link. These three sources are for the manual control of the aircraft, telemetry, and video. The manual control for the aircraft is on a 2.4 GHz frequency to ensure no interference would occur for the safety pilot's control. The telemetry communication between the autopilot and the ground station is on a 915 MHz frequency. The video is streamed over Wi-Fi using a 5.8 GHz frequency. All of the radios use frequency hopping spread spectrum technology to mitigate risk of interference.

2.6.3 Antenna Selection

With this year's change in platform from a multirotor to a fixed-wing aircraft, it was necessary to reassess the previously selected ground station and vehicle antennas. Despite the large success with using the directional antennas in last year's design, the increase in airspeed from this platform change may alter the reliability of its directionality. The antenna selection for the UAS was narrowed down to what is shown in Table 2-2.

Table 2-2: Comparison of possible antenna selections for the ground station.

	Type of Antenna	Polarity	Gain (dB)	Beam Width (Degrees)	Weight (kg)	Price ($)
5.8 GHz Ground	parabolic	linear	24	12	1.4	64
	helical	circular	12.5	30	0.15	50
5.8 GHz Aircraft	whip	linear	5.5	180	0.05	11
	clover	circular	1.4	360	0.05	50
915 MHz Ground	parabolic	linear	15	18	2.29	94
	Patch	circular	8	65	0.45	52
915 MHz Aircraft	whip	linear	3	180	0.05	12
	clover	circular	1.4	360	0.05	34

The drawback to a linearly polarized antenna is that it does not maintain a strong data link if the two antennas are not properly aligned. When the antenna is rotated, a linearly polarized antenna undergoes changes in both amplitude and phase angle, whereas circularly polarized antennas only has changes in its phase [3]. Due to the aircraft constantly pitching and rolling, a linear antenna can potentially lose data. A linear antenna generally has higher gain and range capabilities compared to a circular antenna. This performance is due to the difficulties in manufacturing circularly polarized antennas. A major influence on antenna selection was based on the fact that both antennas have to have matching polarity to have the strongest connection. The trade study used to determine which antennas were selected is shown in Table 2-3.

Table 2-3: Trade study down-select for the ground station antenna.

	Type of Antenna	Polarity	Gain (dB)	Beam Width	Weight	Price	Overall
5.8 GHz Ground	parabolic	0	10	10	1	7	28
	helical	10	5	5	10	10	40
5.8 GHz Aircraft	whip	0	10	5	10	10	35
	clover	10	3	10	10	2	35
915 MHz Ground	parabolic	0	10	10	2	6	28
	Patch	10	5	3	10	10	38
915 MHz Aircraft	whip	0	10	5	10	10	35
	clover	10	5	10	10	5	40

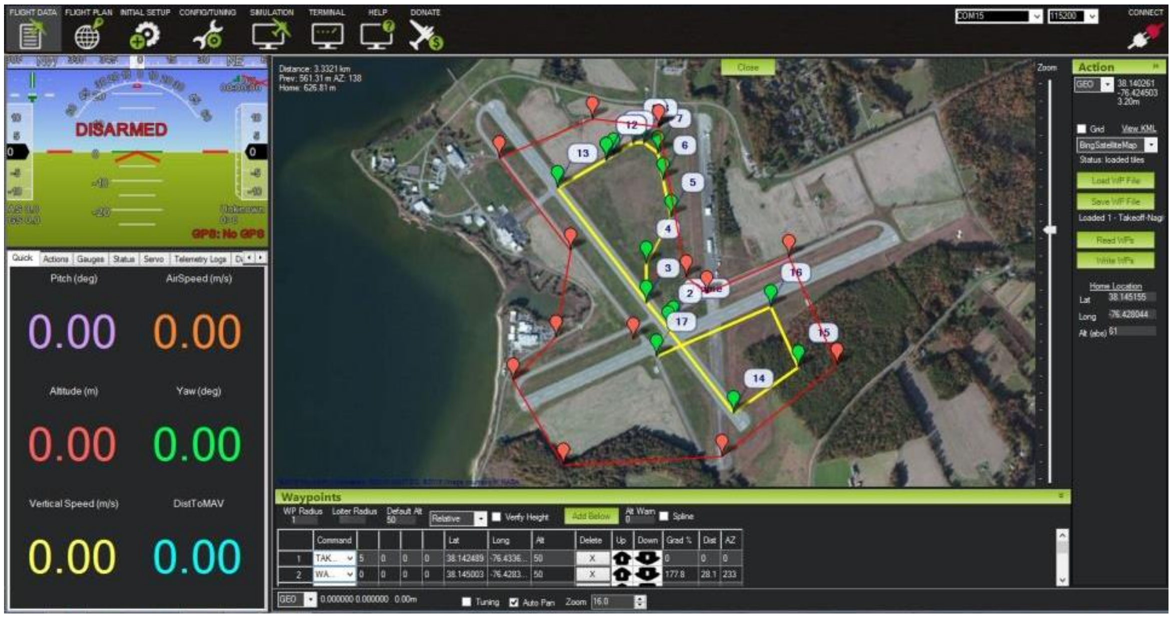
From this trade study, the circularly polarized antennas were chosen for both the 5.8 GHz frequency and 915 MHz frequency. With the selected directional antennas for the ground station, a tracking system was necessary to maintain a strong connection for the imagery and telemetry. This limitation was mitigated by having the ground station antenna attached to a mount that is able to be manually pointed towards the aircraft.

2.6.4 Ground Control Station

The Ground Control Station (GCS) consists of an antenna-tracking system and two computers. The first computer uses the Mission Planner software, and the other computer uses the image processing software which is further explained in 0. Handheld radios are used to ensure proper communication between the GCS crew and the safety pilot to minimize any potential errors.

2.6.5 Mission Planner computer

The objective of this computer is to use the modified Mission Planner software as the main interface between the aircraft and the GCS. Mission Planner will generate a flight path after retrieving the waypoint path and search area details from the interoperability server. This path along with the current aircraft altitude, speed, heading, no-fly zones and obstacles are then displayed in Mission Planner; all of which are constantly monitored by the GCS crew. The primary Mission Planner interface, with example flight path and ancillary vehicle information is shown in Figure 2-7.



	Command					Lat	Long	Alt	Delete	Up	Down	Grad %	Dist	AZ
1	TAK...	5	0	0	0	38.142489	-76.4336...	50	X			0	0	0
2	WA...	0	0	0	0	38.145003	-76.4283...	50	X			177.8	28.1	233

Figure 2-7: Image of primary Mission Planner Interface with example Flight Mission.

This information will be relayed to the image processing computer to provide the aircraft's telemetry, which is required for the target information. This computer will also connect to the interoperability server to collect the information provided by the server, which includes the mission details and obstacle locations. The team member at this station will be responsible for watching the aircraft's path for smooth flight as well as monitoring the interoperability program. The team member must press a button to activate each part of the mission for various tasks such as the bottle drop.

2.6.6 Telemetry Processing

Telemetry processing was added to the Mission Planner software. The software retrieves flight information from MAVLink, a communication link between Mission Planner and the Pixhawk. This flight information is transmitted at 10 Hz and includes the vehicle's latitude, longitude, altitude, airspeed, and heading. This data is used in five tasks: the primary objective; actionable intelligence; emergent target; interoperability; and obstacle avoidance. To accomplish these tasks, the retrieved data is first saved to a text file in a shared folder. The primary image processing computer then retrieves the information from the file so that the telemetry can be associated with an image. The data is also sent to the interoperability server as well as the collision avoidance software. Obstacle avoidance and interoperability are discussed further in 2.3 and 2.6.7, respectively.

2.6.7 Interoperability

The interoperability program works through Mission Planner and the Web Server that is provided during the competition to upload and download information to and from each. It was discovered that the code needed to be written in C# in order to properly communicate information to and from the Mission Planner software. In previous years, the interoperability program was directly added into a Mission Planner file. This year, a separate file with all of the interoperability functions was made and added into the Mission Planner Visual Studio project in order to make working with the code easy. Buttons and forms were added for configuring the interoperability program. The program is a modification of the Mission Planner code and is split into two parts. The first half of the program runs requests and functions that are not needed to be updated at 10 Hz. Those requests and functions include the login Post request to the Web Server and the Mission Details Get request. In order to ensure that all of the requests work properly, the login request is saved into a cookie. Each time a request is made, the cookie is called for the request to be made. The other half of the program includes the requests to receive obstacle information and to upload UAS telemetry data at 10 Hz. The program acquires the UAS telemetry information from the Mission Planner and then uploads that information to the server at 10 Hz to meet the objective requirement given in the competition rules. In previous years, all responses were saved as a string and then parsed by a function written to parse these specific responses. This year the Newtonsoft Json.Net is used to deserialize the Json response from the server in order to make working with the code easier for the future. The obstacle information is deserialized using Json.Net and then sent to the obstacle avoidance part of the code.

2.7 Air Delivery

2.7.1 Air Delivery Mechanism

The Anaconda's air delivery system utilizes E-flite's servoless payload release mechanism. This was chosen due to its size and practical functionality. Last year's air delivery mechanism was a 3D-printed casing that encloses the water bottle and opens when the servo is activated. Consideration was put into integrating the old mechanism with the vehicle, but it was found that it would require too much space inside the fuselage. The current mechanism minimizes the amount of space allocated to the air delivery system without interfering with functionality. The water bottle is attached to the mechanism through straps and a 3D-printed adapter. A 3D-printed mount secures the mechanism to the fuselage and handles loads in the longitudinal and lateral directions. The latch allows for effortless removal and attachment of the water bottle when testing, and the system can be easily integrated with the Pixhawk through the servo rail.

2.7.2 Air Delivery Software

To determine the optimal position to drop the water bottle, a combination of MATLAB and C# was used. To simplify the problem, the following assumptions were made: wind acted in a plane parallel to the ground with constant heading and velocity, and the aircraft will be in steady state flight parallel to the ground with the drop mechanism perpendicular to the ground. With these assumptions, the flight of the bottle would effectively be that of a simple projectile. Determining the optimal drop position is found by calculating the displacement of the bottle with an initial velocity equal to the aircraft's steady state flight speed, and an average wind speed and direction gathered before takeoff. To solve the nonlinear differential equation resulting from the aerodynamic forces, a 4th order Runge-Kutta Method was used in MATLAB. This process solves the bottle's positional equations of motion using an incremental numerical method accurate up to each equation's fourth derivative, with respect to time. Once this method determines the bottles displacement, the change in GPS position is then calculated in terms of the changes latitude and longitude. The optimal drop position is then determined by subtracting the change in position from the known coordinates of the intended target.

2.8 Cyber Security

The potential security threats are mostly between the ground station and the UAV. These potential threats are between the 3 radio connections from the UAV to the pilot or ground station. The connections are the Pixhawk telemetry connection, Grasshopper 3 video stream, and the radio control system connections.

The UAV is connected to the Mission Planner ground station using MAVLink which is well known for not being secure. MAVLink is designed to make sure the packets can be sent without loss but it is not designed to keep other people from connecting to the same device. To solve this issue, the Xbee radio system is used to connect the UAV to the Mission Planner ground station. The Xbee radios offer a 128 bit AES encrypted connection which helps prevent unwanted outside connections and unwanted data leaching. [4]

The Bullet M5 is used to transmit the Grasshopper3's video stream down to the ground station. The data is sent down using an HDMI extender which uses an TCP/IP to send the data to the receiving HDMI extender device. This connection is not secure, although a Bullet M5 is used to transmit the ethernet connection through a 5.8 Ghz radio. The Bullet M5 is setup to use the WPA2-AES to protect the video stream that is transmitted to the ground station.

The radio control system is a Spectrum DX18 transmitter and the Spectrum DSMX remote receiver uses DSM technology which features a Globally Unique Identification number which ensures the connection to the receiver remains between the two devices [5].

3. Safety

3.1 Developmental Risks and Mitigations

There were several developmental risks that were addressed and mitigated. The process of modifying the Anaconda's airframe presents concerns about structural strength. Punching and cutting out holes through the fuselage weakens the airframe's overall strength. The airframe must be able to sustain the loads imposed by flight and the weight of the camera gimbal and air-delivery systems. The risk of the airframe structurally failing was mitigated by 3D printing parts that spread out the stress throughout the fuselage. Another risk is presented with the location of the camera and the possibility of the lens breaking upon ground impact. This risk was mitigated by extending the landing gear struts and giving the camera lens a minimum ground clearance of one inch. For software, developmental risks are imposed by using a neural network for the image processing system. Concerns about finishing the neural network on time are raised due to the lack of prior experience and the complexity of the software. To mitigate this risk, the amount of images fed to the neural network were reduced from the desired 1,000,000 training images to 300,000.

3.2 Mission Risks and Mitigations

Most mission failures result from easily preventable errors during flight preparation. To negate this problem, a pre-flight checklist has been created in ensure that components such as the battery voltage, GPS, wiring connections, telemetry strength, and sensor outputs are checked before each mission. One of the risks that must be addressed prior to flight concerns hardware falling out of the camera gimbal hole in the fuselage. In order to mitigate this, the wiring and hardware components are neatly organized and secured to the fuselage.

The system's safety methodology is based on isolated subsystems to ensure that the aircraft never poses a threat to personnel or property. The electric motor, autopilot, camera and camera gimbal all have their own dedicated batteries. This ensures that the loss of one electrical subsystem does not cascade throughout the entire UAS. The autopilot telemetry frequency is separate from the safety pilot's radio control frequency. This prevents the failure of both autopilot and telemetry in the event of RF interference. In the event that both the autopilot, GCS and the safety pilot cannot communicate with the aircraft, the autopilot is programmed to loiter until connection is reestablished. If this does not occur in a predetermined time period, a failsafe is triggered where the aircraft will immediately land in order to prevent damage to personnel or property. There are two ways that the flight termination failsafe can be triggered. At any time, the ground control station operator can manually trigger an abort that will send a failsafe command to the aircraft. Alternatively, if the autopilot has lost its telemetry link with the ground for more than 20 seconds, it will automatically trigger the failsafe. This ensures that the flight can be terminated in a safe way in all possible scenarios. All failsafe procedures are tested at an airfield with only the flight test crew present to increase reliability and mitigate the risk of hurting a bystander.

4. Acknowledgements

The Cal Poly Pomona AUVSI team would like to thank Northrop Grumman for sponsoring the project. The team would also like to thank SolidWorks for providing access to the 3D solid modeling software used throughout the project. The team would finally like to thank our advisor Dr. Subodh Bhandari for his help and support in guiding the team in the correct direction.

References:

[1] ReadyMadeRC LLC (2018). RMRC Anaconda: *Data Sheet.* Retrieved from https://www.readymaderc.com/products/details/rmrc-anaconda-kit

[2] Ubiquiti Networks - Bullet™ M. (n.d.). Retrieved from https://www.ubnt.com/airmax/bulletm/

[3] Milligan, T. A. (1985). "Properties of Antennas." Modern Antenna Design. New York: McGraw-Hill.

[4] Digi International Inc. (2008). XBee-Pro 900: Data Sheet. Retrieved from https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-900-Manual.pdf

[5] Horizon Hobby, Inc. (2012). SPM9645 DSMX Remote Receiver User Guide. Retrieved from https://www.spektrumrc.com/ProdInfo/Files/SPM9645-Manual.pdf